SELEÇÃO DE ABORDAGENS DE TESTE PARA APLICAÇÕES WEB

Silvia Lopes Santa Isabel

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Engenharia de Sistemas e Computação, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Mestre em Engenharia de Sistemas e Computação.

Orientador: Guilherme Horta Travassos

Rio de Janeiro
Julho de 2011

SELEÇÃO DE ABORDAGENS DE TESTE PARA APLICAÇÕES WEB

Silvia Lopes Santa Isabel

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA (COPPE) DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Examinada por:

_____

Prof. Guilherme Horta Travassos, D.Sc.

_____

Profa. Cláudia Maria Lima Werner, D.Sc.

_____

Profa. Silvia Regina Vergilio, D.Sc.

RIO DE JANEIRO, RJ – BRASIL
JULHO DE 2011

Santa Isabel, Silvia Lopes

Seleção de Abordagens de Teste para Aplicações Web/ Silvia Lopes Santa Isabel. – Rio de Janeiro: UFRJ/COPPE, 2011.

XIII, 147 p.: il.; 29,7 cm.

Orientador: Guilherme Horta Travassos.

Dissertação (Mestrado) – UFRJ/COPPE/Programa de Engenharia de Sistemas e Computação, 2011.

Referências Bibliográficas: p. 108-114.

1. Teste de Software. Abordagens de Teste para Aplicações Web. 3. Seleção de Tecnologias de Software. 4. Engenharia de Software Experimental. I. Travassos, Guilherme Horta II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia de Sistemas e Computação. III. Título.

Aos meus pais, à minha irmã e ao meu amor.

Agradecimentos

Agradecer é bom e necessário. Dedico o meu reconhecimento inicial ao apoio e solidariedade das pessoas que sempre estiveram ao meu lado. A todos aqueles que me inspiraram e estimularam para que eu pudesse alcançar esse objetivo.

Em especial, agradeço a aqueles que mais amo e que mais me fazem sentir amada: minha Mãe, Rosimar, meu Pai, Marinêu, minha Irmã, Simone e meu namorado Marcos. Eu simplesmente não teria conseguido sem a paciência, compreensão e incentivo de vocês. Agradeço também a toda a minha família e aos meus amigos queridos, dos quais subtraí parte do nosso tempo de convívio, mas em favor de uma boa causa. Além deles, somo meu reconhecimento:

Ao meu gerente e amigo Marcus Estrella pela liberação, apoio e incentivo.

Ao meu Orientador, Guilherme Travassos, pela oportunidade e pela orientação durante todo este período.

Aos Companheiros da COPPE, especialmente aqueles que ingressaram nessa jornada junto comigo, Rafael, Wallace e aos mineirinhos Marcelo e Natália.

A todo o grupo de Engenharia de Software Experimental, especialmente Arilo, Jobson e Rodrigo, a ajuda de vocês foi fundamental para a realização deste trabalho.

Ao aluno Thiago pelo apoio na publicação do survey.

À Taisa, sempre atenciosa e prestativa.

Às professoras Claudia Werner e Silvia Vergílio por participarem de minha banca de defesa de mestrado.

Resumo da Dissertação apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

SELEÇÃO DE ABORDAGENS DE TESTE PARA APLICAÇÕES WEB

Silvia Lopes Santa Isabel

Julho/2011

Orientador: Guilherme Horta Travassos

Programa: Engenharia de Sistemas e Computação

Este trabalho apresenta uma instância especializada do procedimento Porantim, denominada Porantim-WAT, para apoiar a seleção de técnicas de teste para projetos de software Web, organizada com apoio de uma metodologia baseada em evidência, que utiliza um repositório de conhecimento com as informações que caracterizam as abordagens de teste para aplicações Web (WAT) e um mecanismo que avalia o grau de adequação entre as abordagens e as características de um projeto de software Web. Neste sentido, além de uma revisão informal da literatura que permitiu identificar os principais conceitos associados às aplicações Web, é apresentada a proposta e avaliação de uma estrutura de caracterização de abordagens de teste para aplicações Web organizada a partir dos resultados obtidos através de uma quasi-revisão sistemática e avaliada por um survey realizado com pesquisadores da área. Ao final, é apresentada a adaptação de uma infra-estrutura computacional para apoiar o procedimento de seleção proposto.

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

SELECTION OF TESTING APPROACHES FOR WEB APPLICATION

Silvia Lopes Santa Isabel

July/2011

Advisor: Guilherme Horta Travassos

Department: Computer Science and Systems Engineering

This work presents a proposal extending an approach developed to support the selection of web application testing (WAT) approaches, named Porantim-WAT. It has been developed by following an evidence-based scientific methodology, which uses a body of knowledge providing a set of information concerned with Web application testing approaches and a mechanism supporting the evaluation of the adequacy level between the WAT approaches and Web software project characteristics. Besides an informal review of the literature which identified the WAT approaches main concepts, it was also presented the proposal and evaluation of a testing characterization structure for WAT approaches organized based on the results of a quasi-systematic review. Such characterization strucuture was assessed through a survey performed with researchers. Using this knowledge set, a computational infrastructure was tailored to support the proposed selection approach. A proof of concept into the context of an industrial web application is then presented.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABELAS

CAPÍTULO 1 - INTRODUÇÃO

Neste capítulo é apresentada a motivação para desenvolvimento deste trabalho (seção 1.1), os objetivos da pesquisa (seção 1.2), a metodologia utilizada (seção 1.4) e a organização do texto (seção 1.5).

1.1 Motivação

Inicialmente concebida como uma forma de publicação de hipertextos estáticos, a Web está se tornando cada vez mais complexa. As aplicações construídas sobre a Internet, utilizando tecnologias de padrão aberto trazem novos desafios aos pesquisadores, como o comportamento dinâmico, representações heterogêneas e novo controle de fluxo de dados (QI, 2005).

Este ambiente, com novas funcionalidades e limitações fazem da atividade de teste de software baseado na Web uma tarefa desafiadora (MUSTAFA, 2007). Em (MANSOUR, 2007), é destacado que as técnicas tradicionais de testes não são adequadas para aplicações baseadas na Web, uma vez que elas não consideram todas as características destas aplicações, tais como a sua natureza multi-camada, estrutura baseada em *hiperlink*, e dirigida a evento.

A rápida evolução tecnológica proporciona novos desafios e faz aumentar a demanda por técnicas e ferramentas que tratem a questão da construção e garantia da qualidade das aplicações Web. Por exemplo, em um estudo secundário realizado inicialmente por CONTE et. al. (2005) e atualizado por MASSOLAR (2008), foi identificado um conjunto de metodologias para apoiar a construção das aplicações Web. Entretanto, a simples utilização de um destes métodos não garante a qualidade da aplicação.

Dentre as atividades de garantia da qualidade, teste de software aparecem como uma importante e motivante área de investigação. Um levantamento inicial sobre possíveis trabalhos nesta área revela que várias publicações focam na definição de novas estratégias de teste.

Desta forma, visando melhor compreender as características das aplicações Web e quais abordagens de teste de software estariam disponíveis na literatura técnica para serem selecionadas para os projetos de software, acreditamos que uma importante questão de pesquisa a ser respondida é:

Q: "Que abordagens de testes para aplicações Web ou para aplicações que utilizam a Web têm sido propostas e quais suas principais características?"

A revisão realizada revelou uma quantidade expressiva de publicações com propostas de novas abordagens de teste para aplicações Web. Portanto, como se depreende, existe disponibilidade de tecnologias de software, especificamente testes de software, que podem ser selecionadas para estas aplicações.

A existência de diferentes métodos e categorias de aplicações envolvendo a Web implicam em características diferenciadas, que aliadas à diversidade das tecnologias de software disponíveis, aumentam a dificuldade de escolha de técnicas de garantia da qualidade adequadas a um projeto de software Web em particular.

Neste contexto, VEGAS e BASILI (2005) descreveram um mecanismo para apoiar a seleção de técnicas de teste em geral, baseado em um esquema de caracterização, onde um catálogo de técnicas de teste de software é instanciado para um projeto em particular.

No entanto, o uso do referido esquema de caracterização em projetos de software Web pode levar a identificação de técnicas de teste não necessariamente adequadas, tendo em vista a generalidade dos atributos utilizados no esquema de caracterização frente às características específicas dos projetos de aplicações Web.

Situação semelhante foi observada por DIAS NETO (2009) em relação ao uso do esquema de caracterização para apoiar a seleção de técnicas de teste baseado em modelos para projetos de software.

Desta forma, considerando a diversidade de técnicas de teste disponível na literatura técnica e as características dos projetos de aplicação Web, a seleção de técnicas de teste para aplicações Web necessita de investigação adicional e a organização de um corpo de conhecimento inicial sobre estas técnicas de teste em particular torna-se importante para aprimorar o procedimento de caracterização e seleção destas técnicas para projetos de software Web.

1.2 Objetivo da Dissertação

O objetivo desta dissertação é apresentar um procedimento para apoiar a seleção de técnicas de teste para projetos de software Web, denominado Porantim-WAT, organizada com apoio de uma metodologia baseada em evidência, que utiliza um repositório de conhecimento com as informações que caracterizam as abordagens de teste para aplicações Web (WAT) e um mecanismo que avalia o grau de adequação entre abordagens e as características de um projeto de software Web.

Para atingir este objetivo, foi necessário identificar e caracterizar as abordagens de testes para aplicações Web publicadas na literatura técnica, e partir desses resultados, definir a estrutura de um corpo de conhecimento para tais abordagens, desenvolvida a partir da adaptação da estrutura definida por VEGAS e BASILI (2005) e DIAS NETO (2009). Adicionalmente, esta pesquisa apresenta a adaptação de uma infra-estrutura computacional para apoiar o procedimento de seleção proposto.

1.3 Metodologia de Pesquisa

A metodologia de pesquisa utilizada para a realização deste trabalho foi orientada pela abordagem baseada em evidências proposta por SPÍNOLA et al. (2008), que evoluiu a metodologia originalmente proposta por SHULL et al. (2001) e estendida por MAFRA et al.(2006). A metodologia é composta por duas etapas: concepção e avaliação da abordagem proposta, conforme descrito na Figura 1.1. e possui grande cunho experimental, provendo diversos estudos para verificar a efetividade de uma tecnologia de software antes que ela seja introduzida na indústria.

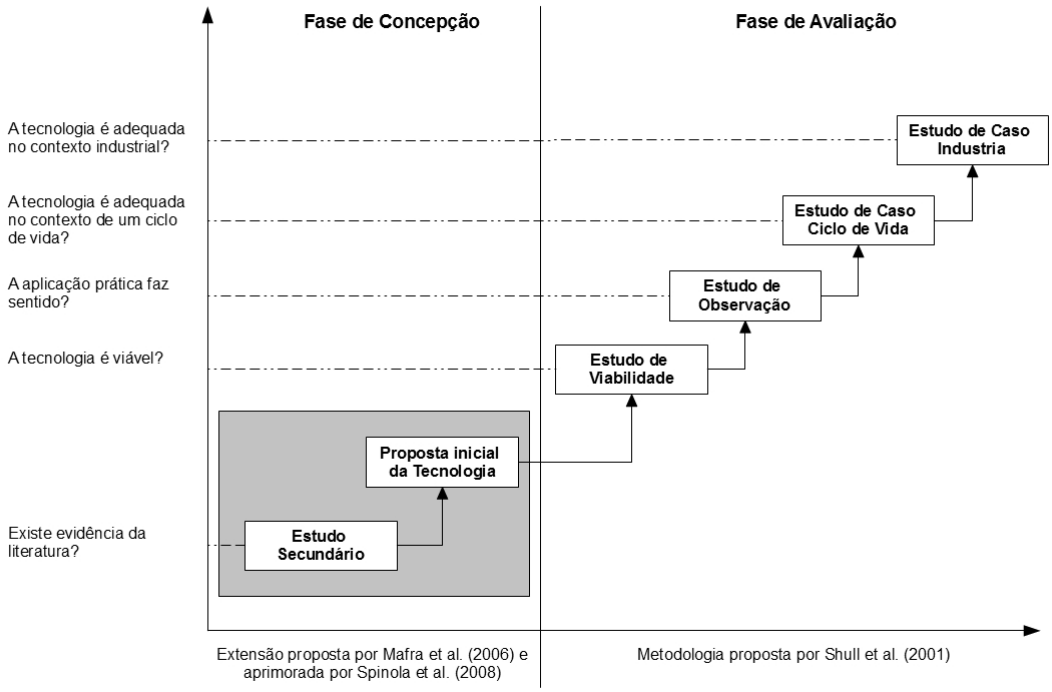


Figura 1.1. Metodologia de Pesquisa adotada (SPÍNOLA et al., 2008).

Para a realização deste trabalho foram executadas as atividades da etapa de concepção, na qual SPÍNOLA et al. (2008) descreve a forma como os estudos secundários devem ser executados, e inclui a realização de estudos primários para

avaliar o conhecimento obtido através dos estudos secundários. As atividades estão descritas na Figura 1.2 e detalhadas abaixo:

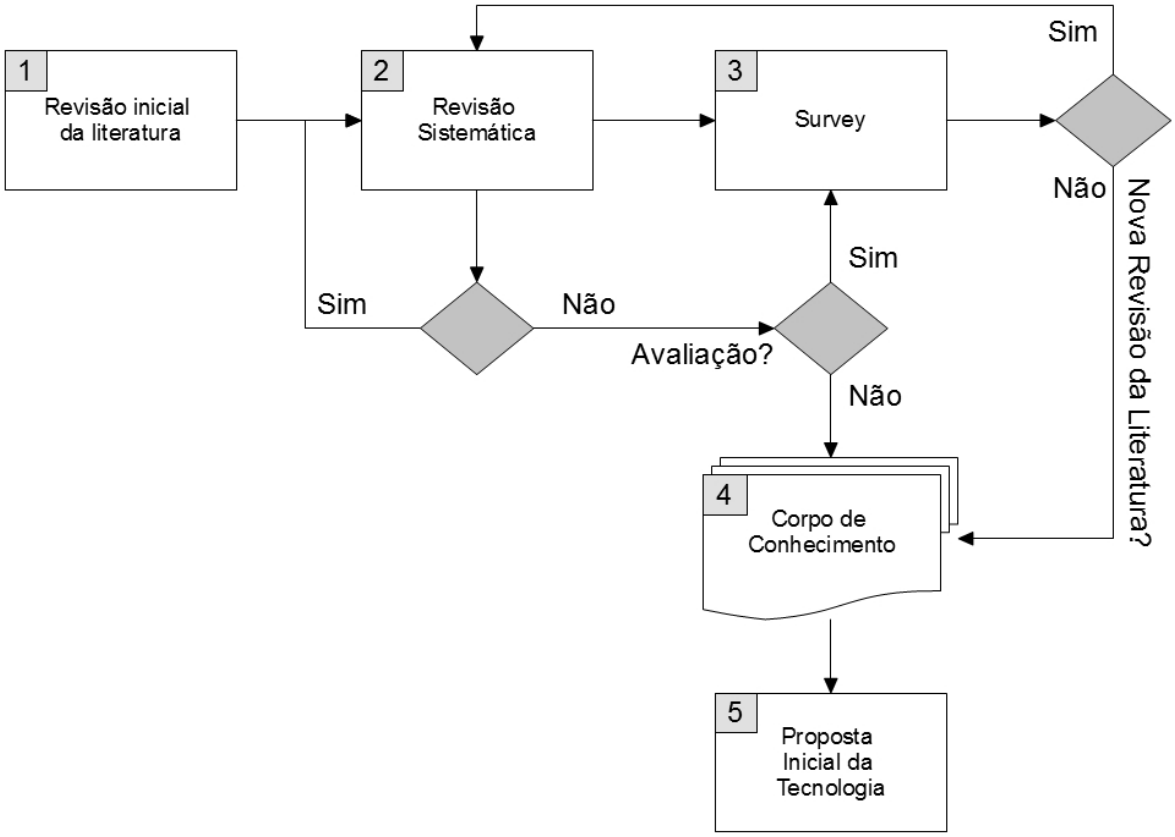


Figura 1.2. Metodologia de Pesquisa – Fase de Concepção (SPÍNOLA et al., 2008).

Atividade 1) Revisão inicial da literatura: o objetivo é identificar os conceitos básicos para apoiar a definição de um protocolo de revisão sistemática da literatura mais preciso e abrangente. Neste trabalho foi realizada uma revisão da literatura sobre aplicações Web, identificando seus conceitos e definições, evolução histórica, categorias de aplicações e as principais diferenças apresentadas em relação às aplicações convencionais. Esta pesquisa incluiu a busca de atributos relevantes para a caracterização de abordagens de teste para aplicações Web.

Atividade 2) Revisão sistemática: Nesta atividade, o protocolo da revisão sistemática é elaborado e executado. Baseado nos resultados obtidos a partir da análise dos artigos identificados, os pesquisadores envolvidos definem se é necessário refinar o estudo executado. Caso positivo, este passo é repetido. Caso contrário, decide-se se o conjunto de conhecimento obtido deve ser avaliado através de um survey (executar atividade 3) ou não (ir para a atividade 4);

Neste trabalho foi realizada uma quasi-revisão sistemática em Agosto de 2009, cujos resultados foram refinados através de uma re-execução realizada em

Outubro de 2010. O objetivo foi identificar e caracterizar as abordagens publicadas na literatura técnica que apóiam a atividade de teste em aplicações Web. Os resultados destes estudos são apresentados no Capítulo.

Atividade 3) Survey: Um survey é planejado e executado para avaliar o corpo de conhecimento organizado através da revisão sistemática executada na atividade 2. Ao final desta etapa tem-se um corpo de conhecimento a ser estruturado.
Neste trabalho foram realizadas duas execuções do survey, com o objetivo de avaliar os atributos das abordagens WAT(obtidos a partir da execução da atividade 2 desta metodologia) com o propósito de caracterizá-los quanto a sua importância e relevância no procedimento de seleção de abordagens WAT para projetos de software Web.
A primeira execução foi realizada em Setembro de 2010 e a segunda em Janeiro de 2011. Os resultados destes estudos são apresentados no Capítulo 3.

Atividade 4) Corpo de Conhecimento: Após as atividades anteriores, o corpo de conhecimento é consolidado e organizado e pode ser utilizado no apoio à concepção de uma nova tecnologia (atividade 5).
Nesta etapa foi consolidada a definição da estrutura e conteúdo do corpo de conhecimento de abordagens de teste para aplicações Web, desenvolvido a partir dos resultados das revisões executadas (atividade 2) e da avaliação de especialistas (atividade 3). O corpo de conhecimento das abordagens WAT é apresentado no Capítulo 3 deste trabalho.

Atividade 5) Proposta inicial da tecnologia: A última atividade é a concepção de uma nova tecnologia apoiada pelo corpo de conhecimento desenvolvido.
Neste trabalho foi definido um procedimento de seleção de técnicas de teste para aplicações Web; bem como a identificação dos requisitos visando à adaptação da infra-estrutura computacional Maraká, proposta em (DIAS NETO 2009), para apoiar a seleção de técnicas de teste baseadas em modelos.
Os resultados desta etapa são apresentados no Capítulo 4.

1.4 Organização da Dissertação

Esta dissertação está organizada em cinco capítulos. O presente capítulo apresentou a motivação para desenvolvimento deste trabalho, os objetivos da pesquisa, a metodologia utilizada e a organização do texto.

Capítulo 2 – Abordagens de Teste para aplicações Web – apresenta uma breve descrição sobre a origem e evolução histórica das aplicações Web, bem como algumas definições e categorias encontradas na literatura. Serão apontadas as diferenças entre os sistemas convencionais e as aplicações Web e processo de teste na Engenharia Web. Além disso, serão apresentados o planejamento, execução e análise de resultados de uma quasi-revisão sistemática realizada com o objetivo de caracterizar as abordagens de teste para aplicações Web disponíveis da literatura técnica.

Capítulo 3 – Estrutura para a caracterização de abordagens de teste para aplicações Web. Este capítulo apresenta a proposta e avaliação de uma estrutura de caracterização de abordagens de teste para projetos de software Web organizada a partir dos resultados obtidos através de uma quasi-revisão sistemática da literatura e avaliada através de um survey realizado com pesquisadores da área. Serão apresentados detalhes do planejamento, execução e análise de resultados do survey que avaliou a estrutura proposta.

Capítulo 4 – Porantim-WAT – Procedimento para apoiar a seleção de abordagens de teste para projetos de software Web – apresenta o procedimento que provê apoio à seleção de abordagens de teste para projetos de software Web, denominado Porantim-WAT. Trata-se de uma instância especializada de Porantim, e da mesma forma, será fundamentada nos mesmos elementos: um corpo de conhecimento sobre abordagens WAT, desenvolvido nesta pesquisa e descrito no Capítulo 3, e um mecanismo que avalia o grau de adequação entre as abordagens de teste e as características de um projeto de software Web.

O procedimento consiste em capturar as informações sobre o projeto de software Web no qual as abordagens WAT devem ser aplicadas. A partir das informações do projeto, o procedimento consulta o corpo de conhecimento das abordagens WAT indica aquelas mais adequadas ao projeto.

Adicionalmente, é apresentada a adaptação de uma infra-estrutura computacional para apoiar a execução das atividades que compõem o mecanismo de seleção de abordagens de teste para projetos Web. A utilização dessa infra-estrutura foi demonstrada a partir da sua aplicação em um projeto de software Web denominado SIGIC, a fim de observar a viabilidade de seu uso no apoio à seleção de técnicas de teste.

Por fim, o quinto capítulo, ***Conclusão***, apresenta as considerações finais deste trabalho, bem como as contribuições da dissertação, suas limitações e perspectivas futuras. Adicionalmente, este trabalho apresenta três apêndices:

- ***Apêndice A – Lista de Artigos Identificados na Quasi Revisão Sistemática.***

Este apêndice apresenta a lista completa dos artigos identificados ao longo das execuções do protocolo da revisão realizadas neste trabalho, identificando os artigos selecionados para escopo desta pesquisa e aqueles que foram descartados, sendo possível verificar o critério que motivou a exclusão de cada publicação e a etapa da revisão na qual a publicação foi excluída.

Os trabalhos são identificados pelo o ano da publicação, os autores e o título. Nos casos das publicações excluídas é apresentado também o motivo da exclusão.

- ***Apêndice B – Evolução da Infra-Estrutura Computacional Maraká***

Este apêndice descreve os requisitos funcionais e não funcionais de adaptação da infra-estrutura Maraká para suportar o procedimento de seleção de abordagens de teste para projetos Web. Serão descritos os ajustes realizados no banco de dados, necessários para armazenar o corpo de conhecimento das abordagens de tese para aplicações Web, bem como os ajustes realizados no código fonte com a criação de um novo componente que automatiza o procedimento Porantim-WAT.

- ***Apêndice C – Glossário***

Ao longo da realização da revisão da literatura e da quasi revisão sistemática para identificar e caracterizar as abordagens de testes para aplicações Web, foi identificada uma grande variedade de tecnologias empregadas nesta área de pesquisa. Este apêndice apresenta um glossário bilíngüe com o objetivo de reunir, de forma breve e objetiva, os significados dos variados termos, expressões e palavras utilizadas para referenciar as tecnologias empregadas nas aplicações Web. Trata-se de uma coletânea de expressões com a respectiva explicação de conceitos dos termos encontrados descritos em inglês e em português.

CAPÍTULO 2 - ABORDAGENS DE TESTE PARA APLICAÇÕES WEB

Neste capítulo é apresentada uma breve descrição sobre a origem e evolução histórica das aplicações Web (seção 2.1), bem como algumas definições (seção 2.2) e categorias encontradas na literatura (seção 2.3). São apontadas as diferenças entre os sistemas convencionais e as aplicações Web (seção 2.4) e processo de teste na Engenharia Web (seções 2.5 e 2.6). Além disso, são apresentados o planejamento, execução e análise de resultados de uma quasi-revisão sistemática realizada com o objetivo de caracterizar as abordagens de teste para aplicações Web disponíveis da literatura técnica (seção 2.7).

2.1 Evolução Histórica das Aplicações Web

A Web foi originalmente concebida como um ambiente capaz de compartilhar informações no formato de hipertexto entre indivíduos geograficamente dispersos apoiados pela utilização de um *browser*. A arquitetura utilizada era cliente-servidor duas camadas, com restrições quanto à flexibilidade e escalabilidade e normalmente apoiava aplicações simples e com limitações de funcionalidade (OFFUT, 2002).

NIELSEN (apud ANDREWS et. al., 2005) afirma que em 1995, 100% das aplicações Web eram compostas por interfaces estáticas, em 1998 esse número reduziu para quase 90%, e que no ano 2000 representavam apenas cerca de 50% das aplicações Web.

Ao longo dos anos a Web sofreu sucessivas evoluções e modificações, apoiando aplicações de pequena e larga escala, desenvolvidas por equipes multidisciplinares com habilidades diversas e com o emprego de novas e variadas tecnologias (MENDES et al., 2006). A configuração foi estendida do modelo cliente-servidor duas camadas para multi-camadas (DELAMARO et al., 2007), que aliadas à rápida evolução tecnológica proporciona novos desafios para as técnicas utilizadas no desenvolvimento de software.

2.2 Definição

Na literatura técnica encontramos diversos termos empregados ao se tentar denominar aplicações Web, dentre eles: Web site, sistemas Web, aplicações de Internet, aplicações baseadas na Web (MENDES et al., 2006). O objetivo desta seção

é discorrer sobre algumas definições encontradas na literatura técnica e apresentar o conceito a ser utilizado ao longo deste trabalho.

Em (KAPPEL, 2004), encontramos a seguinte definição para aplicações Web: “Uma aplicação Web é um sistema de software baseado em tecnologias e padrões do *World Wide Web Consortium* (W3C) que provê recursos específicos de Web, como conteúdo e serviços através de uma interface de usuário, o Web *browser*”. Para XU et al. (2004) as aplicações Web são aplicações hipermídia, distribuídas, multi-plataforma, interativa, dinâmicas, heterogêneas e autônomas.

Embora muitos autores considerem a utilização do *browser* na definição de aplicações Web, esta definição não será completamente adotada no contexto deste trabalho visto que na medida em que ela inclui a utilização de um *browser* como instrumento de interface com usuário ela exclui dessa definição as aplicações que utilizam a infra-estrutura Web com emprego de outros componentes para apresentação e exploração da informação. Dentre outros exemplos, podemos citar as aplicações baseadas em *workflow*, serviços Web, sistemas colaborativos e aplicações envolvendo características de ubiqüidade.

Segundo CHRISTODOULOU et al. (apud MENDES et. al., 2006), a Web tem sido utilizada para prover três tipos de aplicações: Aplicações Hipermídia Web, Aplicações de Software Web e Aplicações Web. Aplicações Hipermídia Web são aplicações não convencionais caracterizadas pela publicação de informações através da Web utilizando nós, *links*, âncoras e estruturas de acesso. Aplicações de Software Web são aplicações de software convencionais que dependem da Web ou do uso da sua infra-estrutura. As aplicações Web são definidas como aplicações distribuídas na Web combinando características das Aplicações Hipermídia e Aplicações de Software Web.

Em (DI LUCCA e FASOLINO, 2006), é mencionado que as aplicações Web podem ser consideradas como um sistema distribuído, com arquitetura cliente-servidor multicamadas, incluindo as seguintes características: grande número de usuários com acesso concorrente; execução em ambientes heterogêneos compostos por diferentes hardwares, conexões de rede, sistemas operacionais, servidores Web e navegadores diferentes; natureza heterogênea devido à variedade de componentes e tecnologias empregadas no seu desenvolvimento (diferentes linguagens, modelos, etc.); e habilidade de geração de componentes dinâmicos, em tempo de execução, de acordo com a entrada dos usuários ou status do servidor. TARHINI et al. (2008) definem um aplicativo da Web como aquele que pode ser acessado através de um navegador da Web e que contém código do lado do cliente e código do lado do servidor.

MASSOLAR (2011) apresenta a definição adotada nesse trabalho, elaborada a partir de uma adaptação de (KAPPEL et al. 2004): “Uma aplicação Web é um sistema de software baseado em tecnologias e padrões do *World Wide Web Consortium* (W3C) que provê recursos específicos de Web, como conteúdo e serviços, através de um cliente Web”. Complementa ainda que as aplicações Web representam sistemas de software complexos, orientados a processos ou a dados, e que integram a exploração não-linear da informação com a capacidade de execução de processos de forma distribuída.

2.3 Categorias de aplicações Web

Frente às várias definições de aplicações Web encontradas, é importante categorizar tais aplicações para melhor compreender suas características, peculiaridades e similaridades. Na literatura encontramos duas categorizações de aplicações Web apresentadas a seguir.

GINIGE e MURUGESAN (2001) apresentam sete categorias de aplicações baseadas na Web: (1) Aplicações Informacionais, tais como jornais on-line, catálogos de produtos, *newsletters*, classificados *on-line*, livros eletrônicos; (2) Aplicações Interativas, tais como formulários de registro, jogos *on-line*; (3) Transacionais: shopping eletrônico, internet *banking*; (4) *Workflow*; (5) Ambientes de trabalho colaborativo; (6) Comunidades e mercados *on-line*, como, por exemplo, leilões *on-line*; e (7) Portais Web.

Em 2004, KAPPEL apresenta uma nova categorização, acompanhando a evolução histórica das aplicações Web (Figura 2.1). As categorias são apresentadas a partir de um referencial histórico e por níveis de complexidade, onde as categorias mais novas apresentam um maior nível de complexidade. A primeira categoria apresentada é a centrada em documentos. Ela remete às origens das aplicações Web, visto que representam aplicações hipermídia estáticas. A segunda categoria é a Interativa representada por páginas Web que possibilitam uma pequena interação com o usuário.

Seguindo a linha de evolução temporal, a terceira categoria é a transacional. Mais complexas que as anteriores, elas são apoiadas por sistemas de banco de dados e apresentam maior interatividade com o usuário. A próxima categoria é a baseada em *workflow*, seguidas pelas categorias de aplicações colaborativas (*groupwares*) e orientadas a portal. As categorias mais recentes e conseqüentemente consideradas mais complexas são as aplicações ubíquas e as aplicações baseadas no conhecimento.

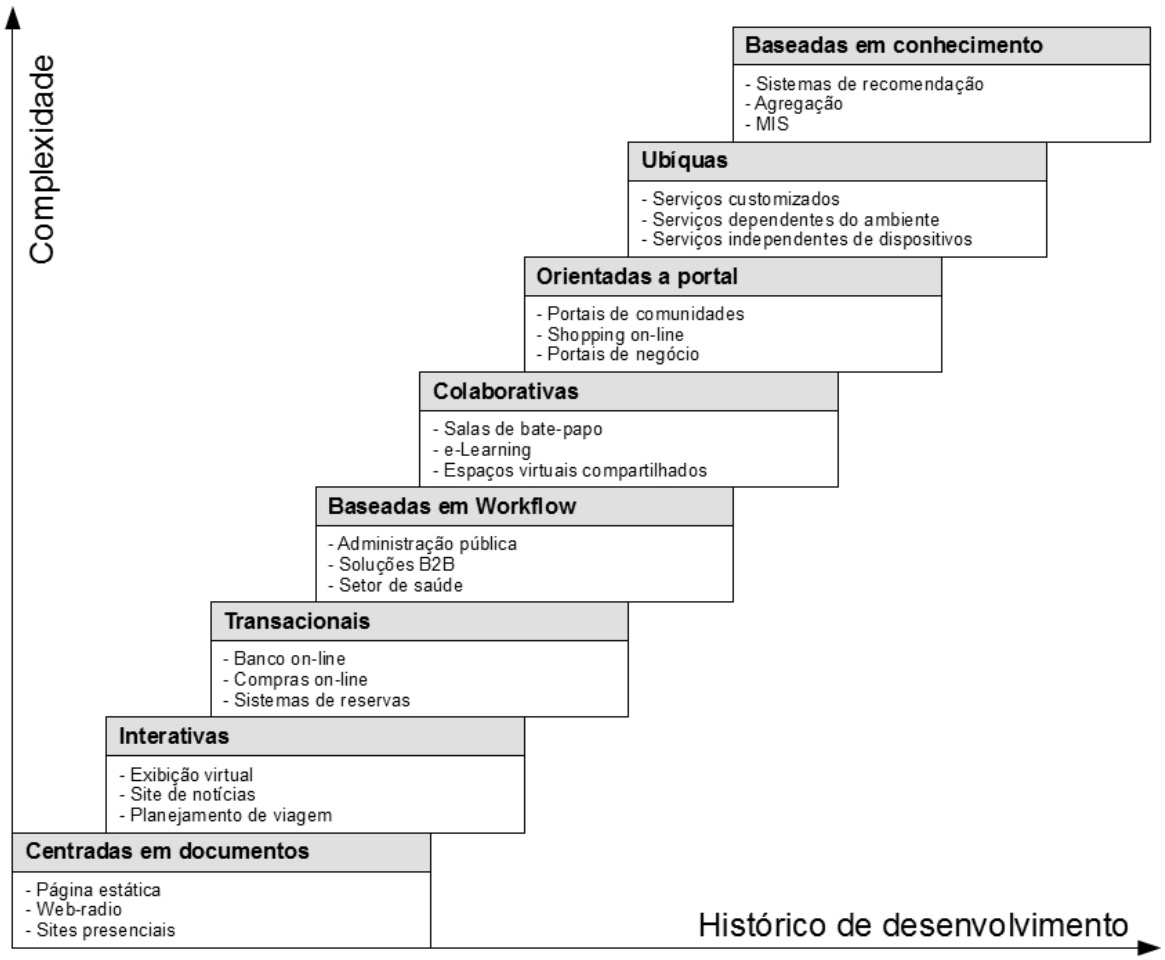


Figura 2.1. Categorias de Aplicações Web (Adaptado de KAPPEL, 2004)

2.4 Diferenças entre aplicações convencionais e aplicações Web

As aplicações Web e aplicações convencionais diferem em vários aspectos, dentre eles, GINIGE et al. (2001) citam que, comparados com a engenharia de sistemas tradicional, o desenvolvimento de aplicações Web destaca-se pelo rápido crescimento de requisitos, conteúdo e funcionalidade, e pelas constantes alterações sofridas durante o ciclo de vida. O desenvolvimento de sistemas baseado em Web é uma atividade contínua e sem *releases* específicas como ocorre no desenvolvimento de software convencional.

MENDES *et al.* (2006) discutem as principais diferenças entre desenvolvimento de software convencional e Web, e discorrem sobre as diversas áreas que apresentam divergências no desenvolvimento de tais aplicações. Destacam que as principais diferenças envolvem as pessoas envolvidas no desenvolvimento (equipe multidisciplinar), as características intrínsecas das aplicações Web (tecnologias

empregadas, disciplinas envolvidas, arquitetura diferenciada) e o público para o qual as aplicações foram desenvolvidas (geralmente as aplicações convencionais são desenvolvidas para um grupo bem definido e conhecido enquanto que as aplicações Web para um grupo de usuários extenso e desconhecido).

XU et al. (2006) descrevem que originalmente as aplicações Web consistiam principalmente da apresentação de textos estáticos, mas atualmente a maioria delas incorpora elementos interativos e dinâmicos. Estes elementos aumentaram as funcionalidades das Aplicações Web, mas também elevou o seu nível de complexidade. Com o mesmo entendimento, DI LUCCA e FASOLINO (2006) destacam que as aplicações Web são dinâmicas e acrescenta que as análises estáticas não são precisas para este tipo de aplicação. Ele ressalta que habilidade de geração de componentes dinâmicos, em tempo de execução, de acordo com a entrada dos usuários é uma das características das aplicações Web que podem inviabilizar o emprego de métodos tradicionais de teste neste tipo de aplicação.

Apesar das diversas percepções e caracterizações das aplicações Web existe o entendimento comum de que tais aplicações apresentam um alto grau de complexidade e o questionamento sobre a aplicabilidade e adequação dos métodos convencionais de engenharia de software no desenvolvimento de Aplicações Web. Nesse contexto, sugiram novas preocupações à cerca do processo de desenvolvimento e garantia de qualidade.

2.5 Engenharia das Aplicações Web

Inicialmente, a maior parte das aplicações Web foi desenvolvida sem um modelo de processo formal. Os requisitos não eram capturados e a arquitetura e do projeto detalhado do sistema não eram considerados (PRESSMAN, 2000). Os sistemas rapidamente passavam da fase de implementação para entrega, sem serem avaliados através da atividade de teste. Usualmente, nenhuma documentação era produzida para descrever a organização interna da aplicação. Este tipo de prática foi motivado pelas características da primeira geração dos sites Web.

A entrega dos sistemas baseados na Web desenvolvidos com métodos *ad hoc* ou não considerando os princípios de engenharia de software era justificada pelo tamanho dessas aplicações, tipicamente pequenas, com pequena expectativa de vida e pelas dificuldades de capturar as necessidades do usuário. Além disso, a primeira geração de sites Web era um pouco mais do que simples materiais de propaganda à disposição do público, sem muita relevância para os negócios.

Rapidamente as empresas perceberam que a Web não era apenas um canal para promover a sua imagem, mas poderia ser explorada como um meio de prestação de serviço. Ao mesmo tempo, diversas tecnologias foram desenvolvidas para suportar a produção da crescente complexidade das aplicações Web, as quais poderiam ser efetivamente exploradas para apoiar os negócios principais das empresas. Conseqüentemente, estas aplicações começaram a incorporar funcionalidades avançadas, tornando-se críticas para as empresas (RICCA e TONELLA, 2002).

O próximo passo nesta evolução foi absorver algumas das lições aprendidas durante a história da engenharia de software, que começou a ser considerada quando a prática de desenvolvimento de aplicações tradicionais sofreu problemas semelhantes aos sistemas baseados em Web.

A Engenharia de Aplicações Web (Web Engineering) surgiu por volta do ano 2000, devido à necessidade de melhorar o desenvolvimento de aplicações Web. A situação do desenvolvimento de aplicações Web era similar às práticas de software dos anos 60, usualmente feito de forma espontânea, não repetível; baseado apenas no conhecimento, experiências e práticas de desenvolvedores individuais; e com ausência de documentação apropriada sobre as decisões de projeto (KAPPEL, 2004).

O principal foco da Engenharia de Aplicações Web é o estudo de abordagens sistemáticas e quantificáveis (conceitos, métodos, técnicas e ferramentas) para a análise de requisitos, projeto, implementação, teste, operação e manutenção de aplicações Web de alta qualidade. KAPPEL (2004) destaca ainda os seguintes princípios da Engenharia de Aplicações Web:

- Objetivos e requisitos claramente definidos;
- Desenvolvimento sistemático de aplicações Web em fases;
- Planejamento cuidadoso de cada fase do desenvolvimento, e;
- Avaliação contínua de todo o processo de desenvolvimento.

Neste contexto, emergiu uma crescente demanda por melhores técnicas, metodologias e processos, e a partir de então um esforço substancial foi dedicado para investigar modelos e formalismos destinados a apoiar o projeto de aplicações Web. Um conjunto de metodologias de desenvolvimento de aplicações Web foi identificado a partir de estudo secundário realizado inicialmente por CONTE et. al.

(2005) e atualizado por MASSOLAR (2011)[1], onde a principal questão de pesquisa era:

"Quais processos de desenvolvimento têm sido utilizados para desenvolver aplicações Web?"

Estão detalhadas a seguir as metodologias identificadas na revisão para apoiar a construção das aplicações Web:

- HDM (GARZOTTO et. al., 1993). Permite a criação de um modelo para aplicações hipermídia. Baseia-se em métodos de projeto de banco de dados (entidade-relacionamento) e influenciou vários métodos subseqüentes.
- RMM (ISAKOWITZ et. al., 1995). Estende a HDM propondo estruturas de acesso de roteiros guiados e índices mais ricos, além de um processo de desenvolvimento detalhado.
- OOHDM (SCHWABE et. al., 1996). Método baseado em modelos orientados a objetos concentrando-se em dois aspectos: navegacional e estrutural.
- WebML (CERI et al., 2000). Linguagem conceitual que permite a descrição em alto nível de uma aplicação Web, independente da plataforma tecnológica de implementação. Propõe a especificação das características dos Web Sites *Data-Intensive* em diferentes modelos (Dados, Hipertexto e Apresentação).
- OOWS (PASTOR et. al., 2001). Método de desenvolvimento de aplicações Web que introduz novos conceitos OO (primitivas de modelagem) para a modelagem da semântica navegacional e de apresentação.
- OO-H (CACHERO et. al., 2001). Abordagem de desenvolvimento de aplicações Web que adota a combinação de três perspectivas: estrutural, comportamental e de apresentação.
- WAE (CONALLEN, 2002). Extensão da UML para aplicações Web.
- UWE (KOCH & KRAUS, 2002). Processo de desenvolvimento para aplicações Web que utiliza como base apenas a notação e os mecanismos de extensão previstos na UML. Propõe a modelagem de

[1] O protocolo completo utilizado na condução da referida revisão, detalhando o procedimento adotado, as questões de pesquisa, as strings de busca e as limitações da revisão podem ser encontrados em (CONTE et. al., 2005) e (MASSOLAR, 2008)

uma aplicação Web através de três perspectivas de projeto: conceitual, navegacional e apresentação.

- ADM (DIAZ et. al., 2004). Processo flexível e detalhado que considera diferentes níveis de abstração: estrutural, navegacional, apresentação, interação entre usuário e componentes do sistema, requisitos funcionais e regras de acesso para prover um ambiente seguro.
- W2000 (BARESI et. al., 2006). Notação para a modelagem de aplicações Web originada da HDM. Integra conceitos de diferentes domínios e incorpora conceitos da UML. Adota uma abordagem dirigida a modelos.

Entretanto, a simples utilização de um destes métodos não garante a qualidade da aplicação. A existência de diferentes métodos e categorias de aplicações envolvendo a Web implica em características diferenciadas, que aliada à diversidade das tecnologias de software disponíveis, aumentam a dificuldade de escolha de técnicas de garantia da qualidade adequadas a um projeto de software Web em particular.

2.6 Processo de Teste das Aplicações Web

GINIGE e MURUGESAN (2001) discutem os resultados de uma pesquisa sobre projetos de aplicações Web de larga escala e destacam que 84% dos sistemas produzidos não atenderam às necessidades de negócio; 53% dos sistemas não contemplavam todas as funcionalidades requisitadas; e 52% dos sistemas entregues eram de baixa qualidade.

Dentre as atividades de garantia da qualidade, teste de software aparecem como uma importante e motivante área de investigação. A atividade de teste pode ser definida como uma atividade de verificação e validação que analisa um produto ao longo de um processo de desenvolvimento de um software com o objetivo de aprimorar a sua qualidade, expondo possíveis defeitos e os desvios do comportamento desejado da aplicação (adaptado de RICCA e TONELLA, 2005).

2.6.1 Níveis de Teste

A atividade de teste é uma tarefa complexa e por isso é normalmente dividida em fases com objetivos distintos, também conhecidas como níveis ou etapas de teste. Os principais níveis de teste encontrados na literatura são:

- Teste de Unidade: Este teste tem como foco as menores unidades de uma aplicação que podem ser funções, procedimentos, métodos ou classe. No

contexto das aplicações Web, a unidade pode ser considerada como uma página Web e seus elementos (campos, texto, formulários, botões, *links*, scripts) ou um serviço Web. RICCA e TONELLA (2002) descrevem que nesta etapa, cada unidade é testada separadamente e a atividade de teste pode ser aplicada à medida que ocorre a implementação sem a necessidade de dispor-se do sistema totalmente finalizado, sendo necessário em alguns casos a utilização de *"drivers"* ou *"stubs"* em substituição às funcionalidades ausentes. Por exemplo, um script de servidor é chamado por um *"driver"* que simula o navegador ("*browser*") e recebe a página HTML como a saída resultante. Outro exemplo é um programa Java script que valida os campos em uma página do lado do cliente e envia uma solicitação para um servidor fictício, simulados por um "*stub"*.

- Teste de Integração: Normalmente realizado após as unidades terem sido testadas individualmente. À medida que as diversas partes são colocadas para trabalhar juntas, é preciso verificar se a interação entre elas funciona adequadamente. No contexto das aplicações Web, pode ser considerada a integração de páginas Web ou a composição de serviços Web.
- Teste de Sistema: Em geral este teste é realizado quando o sistema está completo, com todas as suas partes integradas. O objetivo é verificar se os requisitos funcionais e não funcionais especificados foram corretamente implementados. O sistema é validado como um todo em um ambiente o mais semelhante possível do ambiente alvo real.
- Teste de Aceitação: Este teste é geralmente realizado pelos usuários finais do sistema para avaliar se o produto entregue está de acordo com o solicitado. Em geral, o cliente instala e executa o aplicativo em seu próprio ambiente.

2.6.2 Técnicas de Teste e Critério de Cobertura

Além das fases de execução, outra característica da atividade de teste está relacionada à técnica empregada na sua execução. As principais técnicas de teste são: Funcional e Estrutural. O que distingue essencialmente as técnicas de teste é a fonte utilizada para definir os requisitos de teste e os critérios de cobertura utilizados para explorar, exercitar e avaliar as aplicações sob análise.

Teste Funcional é uma técnica utilizada considerando o programa ou aplicação a ser avaliada como uma caixa preta. Para testá-lo, são fornecidas entradas e

avaliadas as saídas geradas para verificar se estão em conformidade com os objetivos especificados. Nessa técnica, os detalhes de implementação não são considerados.

Os critérios de cobertura mais conhecidos da técnica de teste funcional são o Particionamento por Equivalência, Análise de Valor Limite, e Grafo Causa-Efeito. Uma das vantagens dos critérios da técnica funcional é que eles requerem somente a especificação do produto para derivar os requisitos de teste, não sendo necessário o acesso ao código-fonte. Por outro lado, não é possível assegurar, por exemplo, que trechos críticos do código tenham sido percorridos e validados.

Em geral, a técnica de teste caixa preta para aplicações Web não é diferente da técnica utilizada em aplicações de software convencionais. Em ambos os casos o uso de um critério de cobertura e um adequado conjunto de requisitos de teste são definidos com base na especificação da funcionalidade dos itens que serão testados. Entretanto, algumas características específicas das aplicações Web podem afetar o projeto e a execução dos testes. Por exemplo, o teste de componentes gerados dinamicamente durante a execução pode ser uma tarefa mais árdua, devido à dificuldade de identificar e reproduzir as mesmas condições que cada componente produziu. Nesse caso, as especificações utilizadas para representar o comportamento de aplicações tradicionais deverão ser adaptadas para as aplicações Web (DI LUCCA e FASOLINO, 2006).

A técnica de teste estrutural, ou caixa branca, avalia o comportamento interno do componente de software. Essa técnica trabalha diretamente sobre o código fonte para avaliar aspectos tais como caminhos lógicos e conjuntos específicos de condições. Os critérios de cobertura associados à técnica estrutural são classificados com base na complexidade, no fluxo de controle e no fluxo de dados.

Dentre as limitações desta técnica, destaca-se o fato de que os requisitos de teste são baseados na estrutura interna da aplicação e desta forma eventuais problemas no código implicam em testes incompletos (DELAMARO et al., 2007).

No contexto das aplicações Web, RICCA e TONELLA (2002) propõem alguns critérios de teste caixa branca:

- Teste de todas as páginas (“*all-pages”*): onde cada página do aplicativo Web deve ser visitada nas atividades de teste pelo menos uma vez;
- Teste de todos os *Hyperlinks* (*“all-links”*): onde cada link de cada página deve ser percorrido nas atividades de teste pelo menos uma vez;
- Teste de todos os caminhos (*“all-pathes”*): onde todos os caminhos da aplicação Web deverão ser percorridos em algum teste pelo menos uma vez.

• Teste de todos os usos (“*all-use*”): onde todos os caminhos de navegação serão exercidos considerando todas possibilidade de uso para todas definições de variáveis, formando uma dependência de dados.

Além das técnicas de caixa branca e caixa preta apresentadas, DI LUCCA e FASOLINO (2006) destacam que a técnica de teste caixa cinza também pode ser utilizada nos testes das aplicações Web. Trata-se de uma combinação entre as técnicas funcionais e estruturais, que considera tanto o comportamento da aplicação quando a sua estrutura interna. É esperado que a utilização desta técnica ajude a revelar problemas que não são facilmente capturados com uso individual das técnicas funcionais e estruturais, especialmente problemas associados ao fluxo de informações e compatibilidades quanto à distribuição de hardware e configurações de software dos sistemas.

É possível ainda distinguir o tipo de análise de teste que a abordagem é capaz de suportar. RICCA e TONELLA (2001) definem que as páginas da Web podem ser estáticas ou dinâmicas. Embora o conteúdo de uma página Web estática seja fixo, o conteúdo de uma página dinâmica é computado em tempo de execução pelo servidor e pode depender das informações fornecidas pelos usuários através de campos de entrada. Uma distinção semelhante é proposta em ANDREWS et al., (2005).

2.6.3 Características de Qualidade

As aplicações Web apresentam algumas características distintas quando comparadas às aplicações convencionais, tais como junção de várias tecnologias em uma mesma aplicação, rápida evolução tecnológica, evolução constante de seus requisitos, acesso via *browser* (ainda em grande maioria) e arquitetura diferenciada (possibilitando a execução de funções através de servidores ou do próprio cliente). Essas particularidades fazem com que as características de qualidade de uma aplicação Web precisem ter uma avaliação diferenciada.

A norma ISO/IEC 9126 (2002) apresenta um conjunto genérico de características de qualidade para o software, que são confiabilidade, eficiência, funcionalidade, manutenibilidade, usabilidade e portabilidade. Cada característica de qualidade é composta por sub-características que detalham um pouco mais cada requisito de qualidade a ser atingido em uma aplicação:

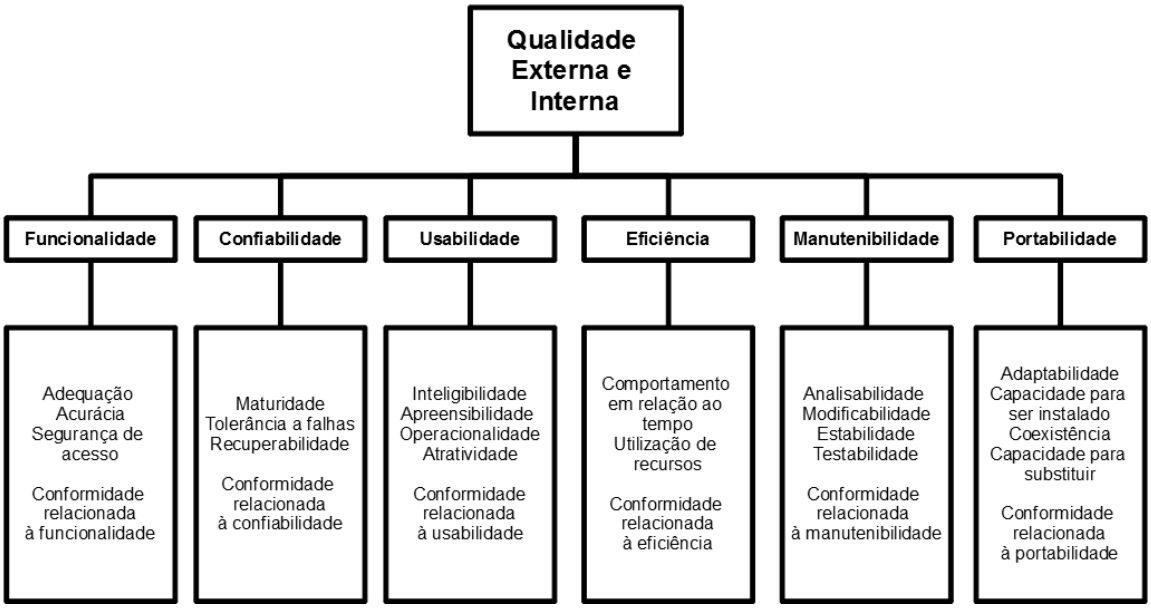


Figura 2.2. Características e Subcaracterísticas de Qualidade (NBR ISO/IEC 9126-1, 2003).

- **Funcionalidade**: capacidade do software em prover funções que atendam às necessidades quando é usado sobre determinadas condições.

Adequação: prover um conjunto apropriado de funções para tarefas e objetivos específicos do usuário.

Acurácia: prover resultados ou comportamentos corretos e de acordo com o nível de precisão necessário.

Interoperabilidade: interagir com um ou mais sistemas.

Segurança: proteger informações e dados que pessoas ou sistemas não autorizados possam ler ou modificá-los, e pessoas ou sistemas não autorizados possuem acesso negados a eles.

Conformidade à funcionalidade: estar de acordo com normas ou convenções relacionadas à funcionalidade.

Muitos métodos utilizados para testar os requisitos funcionais de uma aplicação “tradicional” também podem ser utilizados no teste de aplicações Web. Em relação a subcaracterística de segurança, a natureza das aplicações Web, no que diz respeito à junção de várias tecnologias em um ambiente heterogêneo associado ao grande número de usuários e possibilidades de acesso, podem tornar o teste de segurança uma tarefa mais árdua (DI LUCCA e FASOLINO, 2006).

- **Confiabilidade**: Capacidade do software em manter um nível especificado de desempenho quando usado sobre condições específicas.

Maturidade: evitar falhas como um resultado de um erro presente no software.

Tolerância à falhas: manter um nível especificado de desempenho no caso de erros ou por ter infringido a interface definida.

Recuperabilidade: re-estabelecer um nível especificado de desempenho e recuperar os dados diretamente afetados no caso de uma falha.

Conformidade à confiabilidade: estar de acordo com normas ou convenções relacionadas à confiabilidade.

No contexto Web, o teste de confiabilidade está muitas vezes associado ao processamento correto do fluxo de navegação sem erros (REZA et al., 2008), à integridade dos *links* para evitar situações de páginas inacessíveis ou sem ponteramento adequado; à apresentação de mensagens de erros permitindo o retorno às funcionalidades, e ao processamento cancelado em caso de expiração da sessão.

- **Usabilidade**: capacidade do software de ser entendido, aprendido, utilizado, e atrativo para seus usuários, quando usado sobre condições específicas.

Facilidade de entendimento: possibilitar ao usuário o entendimento da aplicação, e como ela pode ser usada para uma tarefa ou condição de uso particular.

Facilidade de aprendizado: possibilitar ao usuário aprender como usar a aplicação.

Operabilidade: possibilitar ao usuário operar e controlar a aplicação.

Atratividade: ser atrativo ao usuário.

Conformidade à usabilidade: estar de acordo com normas ou convenções relacionadas à usabilidade.

O teste de usabilidade é uma questão crítica para uma aplicação Web. De fato, essa característica de qualidade pode determinar o sucesso da aplicação. Como conseqüência, o *"front-end"* da aplicação e como os usuários interagem com ele, muitas vezes é foco de maior cuidado e atenção neste tipo de teste (DI LUCCA e FASOLINO, 2006).

- **Eficiência**: capacidade do software em prover desempenho apropriado, relativo à quantidade de recursos usados sob certas condições.

Comportamento de tempo: prover tempo apropriado de resposta e processamento.

Utilização de recursos: usar quantidade apropriada e tipos de recursos quando o software realiza suas funções sobre determinada condição.

Conformidade à eficiência: estar de acordo com normas ou convenções relacionadas à eficiência.

No contexto Web, o desempenho da aplicação é uma questão crítica, pois os usuários não querem esperar muito tempo por uma resposta às suas requisições. É esperado ainda que os serviços estejam sempre disponíveis. Esta é uma tarefa difícil, porque não é possível saber de antemão quantos usuários vão realmente estar conectados a um aplicativo em execução.

As falhas que podem ser descobertas através de testes de desempenho estão associadas às falhas de execução do ambiente (por exemplo, recursos escassos ou recursos mal utilizados).

- **Manutenibilidade**: capacidade do software de ser modificado. Modificações podem incluir correções, melhorias ou adaptações do software a mudanças no ambiente e nos requisitos e especificações funcionais.

Analisabilidade: permitir diagnóstico de deficiências ou causas de falhas, ou identificação das partes que precisam ser modificadas.

Modificabilidade: possibilitar que uma modificação específica seja implementada.

Estabilidade: evitar efeitos não-esperados após modificações.

Testabilidade: possibilitar que modificações sejam validadas.

Conformidade à manutenibilidade: estar de acordo com normas ou convenções relacionadas à manutenibilidade.

Não foi possível identificar questões específicas do teste de manutenibilidade associada ao teste de aplicações Web. Desta forma, os métodos utilizados para testar uma aplicação “tradicional”, a princípio, também podem ser utilizados no teste de aplicações Web.

- **Portabilidade**: capacidade do software de ser transferido de um ambiente para outro.

Adaptabilidade: poder ser adaptado para diferentes ambientes sem aplicação de ações ou qualquer outro ajuste além daqueles já fornecidos pela aplicação.

Facilidades de instalação: poder ser instalado em um ambiente específico.

Co-existência: capacidade de co-existir com outras aplicações em um ambiente comum com compartilhamento de recursos.

Facilidade de substituição: capacidade de ser usado no lugar de outro software com o mesmo propósito existente no mesmo ambiente.

Conformidade à portabilidade: estar de acordo com normas ou convenções relacionadas à portabilidade.

No caso das aplicações Web, o teste da portabilidade irá descobrir falhas associadas ao uso de diferentes plataformas de servidores web ou diferentes navegadores (*"browsers"*) com diferentes versões ou configurações (EATON e MEMON , 2007).

A grande variedade de combinações possíveis de todos os componentes envolvidos na execução de uma aplicação web torna inviável o teste de todos os cenários possíveis de utilização. O que normalmente ocorre é que apenas as combinações mais comuns são consideradas. Como conseqüência, apenas um subconjunto de defeitos de compatibilidade pode ser descoberto (DI LUCCA e FASOLINO, 2006).

Cada característica da norma foi associada a uma característica específica, necessária ou presente em aplicações Web. A Tabela 2.1 apresenta o mapeamento das características de qualidade de acordo com a norma ISO/IEC 9126 (2002), destacando apenas segurança que representa uma sub-característica de funcionalidade, devido a sua importância para as aplicações Web.

Tabela 2.1. Mapeamento das Características de Qualidade.

Característica ISO 9126	Característica necessária ou presente em aplicações Web
Funcionalidade	Atendimento aos requisitos funcionais solicitados pelos usuários. Precisão dos resultados e do comportamento da aplicação de acordo com os requisitos funcionais e não funcionais. **** Sub-característica SEGURANÇA**: Acesso às funcionalidades somente por pessoas autorizadas. Evitar "ataques" diretos e indiretos ao servidor de aplicação e de banco de dados. Evitar acessos diretos às páginas dinâmicas (informando valores através de *links*)
Confiabilidade	Processamento correto de *links* (fluxo de navegação).

	Integridade dos *links* (evitar páginas inacessíveis, *links* quebrados ou sem ponteramento adequado). Apresentação de mensagens de erros, permitindo o retorno às funcionalidades. Processamento cancelado em caso de expiração da sessão.
Usabilidade	Preocupações com o nível de apresentação, tais como: Facilidade no uso da aplicação Aprendizagem das terminologias, menus e navegação através de diferentes páginas da Web
Eficiência	Tempo de resposta a uma solicitação. Tempo para carregar uma página. Carga de servidor de aplicação (quantidade de solicitações). Carga de servidor de banco de dados (quantidade de acesso).
Manutenibilidade	Não foi possível mapear atributos específicos para aplicação Web.
Portabilidade	Verificação de recursos específicos de *browser* e restrições para carregar uma determinada página (*scripts*, *plugin*, etc.). Adaptação a diferentes servidores, ou sistemas operacionais.

Um levantamento inicial sobre possíveis trabalhos na área teste de aplicações *Web* revela que várias publicações focam na definição de novas estratégias de teste. Visando melhor compreender as características das abordagens de teste para aplicações Web disponíveis, de forma a identificar, por exemplo, as características de qualidade cobertas, técnica de teste empregada e nível de teste no qual a abordagem está apta a avaliar, optou-se por realizar um estudo secundário na literatura técnica, detalhado a seguir.

2.7 *Quasi* Revisão Sistemática

2.7.1 Introdução

Segundo KITCHENHAM (2004), uma revisão sistemática da literatura é um meio de identificar, avaliar e interpretar toda pesquisa relevante e disponível sobre uma questão de pesquisa particular, área ou fenômeno de interesse. Os estudos individuais que contribuem para uma revisão sistemática são chamados de estudos primários. A revisão sistemática é denominada como estudo secundário.

Revisões sistemáticas podem ser divididas em três fases principais: (1) Planejamento da revisão, que consiste na identificação da necessidade de uma revisão e no desenvolvimento do respectivo protocolo de revisão; (2) Condução da Revisão, que consiste na identificação da pesquisa, seleção dos estudos primários, avaliação da qualidade dos estudos, extração, monitoramento e síntese dos dados; e (3) Análise de Resultados.

A questão de pesquisa deve ser elaborada utilizando um formato claro e explícito. Para isso, é importante que ela possa ser, após definida, estruturada de acordo com a estratégia PICO que explora as seguintes dimensões: População, Intervenção, Comparação e Resultado ("*Outcome*"). A população descreve o grupo de

pessoas, tipos de projetos ou aplicações onde será observada a intervenção, que por sua vez, define o que será observado no contexto da revisão. Na comparação devem ser descritas quaisquer restrições ou resultados anteriores que possam ser usados para comparar com a intervenção. Os resultados devem relatar os fatores importantes para os profissionais da prática, tais como aperfeiçoamento na confiabilidade, redução de custos. Em alguns casos, são requeridas intervenções que reportam apenas o aperfeiçoamento de algum aspecto na produção de software (KITCHENHAM, 2004).

O estudo proposto nesse trabalho pode ser caracterizado como uma quasi-revisão sistemática, pois a revisão não apresenta todas as dimensões da estratégia PICO. Em geral, quasi-revisões sistemáticas representam revisões de caracterização (revisões iniciais) que não realizam qualquer tipo de comparação. Entretanto, por tal revisão explorar o mesmo rigor e formalismo nas fases metodológicas de preparação e execução do protocolo, sem fazer uso de meta-análise, ela pode ser referenciada como uma quasi-revisão sistemática. Estas revisões são importantes para estabelecer uma base inicial de conhecimento para a comparação de futuros estudos primários e pode ajudar a evidenciar conhecimento de grande valor para os engenheiros de software (TRAVASSOS et al., 2008).

As próximas seções descrevem as etapas de planejamento, que consiste na apresentação do protocolo da revisão, questão de pesquisa, execução e análise dos resultados.

2.7.2 Planejamento da Revisão

A etapa de planejamento consiste na elaboração de um protocolo que especifica os métodos que serão aplicados na condução da revisão e a sua importância está associada à redução da possibilidade do viés do pesquisador e na possibilidade de sua avaliação e repetição por outros pesquisadores.

2.7.2.1 Questão de Pesquisa

O objetivo dessa pesquisa é realizar uma quasi-revisão sistemática a fim de caracterizar as abordagens de teste de software para aplicações Web descritas na literatura técnica, identificando os elementos que compõem as diferentes abordagens, suas características e instrumentos de apoio empregados nas atividades do processo de teste.

Desta forma, visando melhor compreender as características das aplicações Web e quais abordagens de teste de software estariam disponíveis na literatura técnica para serem selecionadas para os projetos de software Web, acreditamos que uma importante questão de pesquisa a ser respondida é:

Q: "Que abordagens de testes para aplicações Web ou para aplicações que utilizam a Web têm sido propostas e quais suas principais características?"

2.7.2.2 PICO

O objetivo da revisão é caracterizar as abordagens de teste de software para aplicações Web descritas na literatura técnica, identificando os elementos que compõem as diferentes abordagens. Para isso, a questão de pesquisa Q deve ser estruturada de acordo com a estratégia PICO:

(P) População: Aplicações Web, aplicações que dependem da infra-estrutura Web para sua execução e as categorias de aplicações Web identificadas por GINIGE e MURUGESAN (2001) e por KAPPEL (2004) ;

(I) Intervenção: Abordagens de teste aplicadas em projetos de software Web.

(C) Comparação: Não há.

(O)Resultados ("*Outcomes*"): Elementos que compõem as diferentes abordagens de teste para aplicações Web.

A expressão de busca é formada pelas palavras chave definidas a partir da combinação lógica das dimensões PICO (População, Intervenção, Comparação e Resultado):

S: (Termos da População) AND (Termos da Intervenção) AND (Termos da Comparação) AND (Termos do Resultado - "Outcome")

Entretanto, para não restringir demais os elementos que compõem as diferentes abordagens de teste de software, o elemento Resultado não foi incluído na definição dos termos chave, já que estes termos representam as definições que deverão ser extraídas dos artigos. Desta forma, a expressão de busca é estruturada apenas a partir das dimensões População e Intervenção, visto que não foram identificados elementos de Comparação. Convertendo as dimensões pelas palavras chave definidas, teremos:

S: (Termos da População) AND (Termos da Intervenção)

Onde:

Termos da População = conjunto de termos que identificam aplicações Web.

Termos da Intervenção = conjunto de termos sobre Abordagem e Teste.

2.7.2.3 Palavras Chave

As palavras chave foram definidas a partir dos elementos PICO (População, Intervenção, Comparação e Resultado) (PAI et a. 2004). A população será representada pelo conjunto de termos que identificam as aplicações Web e aplicações que dependem da infra-estrutura Web para sua execução. A intervenção será representada pelos termos que descrevem abordagens de teste de software. Não foram identificados para esse estudo elementos de comparação e optamos não incluir o elemento resultado na definição dos termos chave, a fim de caracterizar as diferentes abordagens de teste de software para aplicações Web a partir das questões secundárias. Segue abaixo a lista dos termos identificados:

Aplicações: Web Application, Web system, Web-based system, Web-based application, Web software, Web Services, Internet applications, agent-based system, agent-based software, agent-based application, Knowledge-based system, Knowledge-based software, Knowledge-software, Workflow, Ubiquitous system, Ubiquitous software, Ubiquitous application, e-commerce, e-government, Collaborative System, Network sensor, client-server, multi-layer, grid computing.

Abordagem: process, approach, method, technique, methodology, strategy.

Teste: Software Test, Software Testing, System Test, System Testing, Test Web, Testing Web, Web test, Web testing, Web application testing.

2.7.2.4 Definição das máquinas de busca

A máquina de busca escolhida para a execução da revisão foi a Scopus (http://www.scopus.com/) visto que a sua base de dados cataloga a maior parte dos artigos existentes em diferentes bibliotecas digitais ou outras máquinas de busca (Compendex, IEEE , ACM Digital Library, Springer, Web of Science). Desta forma, foi considerado que a cobertura oferecida por esta máquina de busca seria suficiente para o escopo desta pesquisa.

2.7.2.5 Expressão de Busca

A expressão de busca submetida à máquina Scopus foi:

TITLE-ABS-KEY(({Web Application} OR {Web Applications} OR {Web system} OR {Web systems} OR "Web-based system" OR "Web-based application" OR {Web software} OR {Web Service} OR {Web Services} OR {Internet application} OR {Internet applications} OR "agent-based system" OR "agent-based software" OR "agent-based application" OR "Knowledge-based system" OR "Knowledge-based software" OR "Knowledge-based application" OR "Workflow-based system" OR "Workflow-based software" OR "Workflow-based application" OR {Ubiquitous System} OR {Ubiquitous Systems} OR {Ubiquitous Software} OR {Ubiquitous application} OR {Ubiquitous applications} OR {e-commerce} OR {e-government} OR {Collaborative System} OR {Collaborative Systems} OR {Collaborative Software} OR {Collaborative application} OR {Collaborative applications} OR {Network sensor} OR "client-server" OR "multi-layer" OR {grid computing}) AND (({Software Test} OR {Software Testing} OR {System Test} OR {System Testing} OR (Web W/2 test)) AND (process OR approach OR method OR technique OR methodology OR strategy)))*

É possível observar que alguns termos aparecem entre aspas e outros entre chaves. De acordo com as instruções descritas no *help* da máquina de busca Scopus, o uso de chave retorna exatamente o termo nele descrito, sem variações, enquanto o uso de aspas retorna algumas variações do termo descrito, como por exemplo, o uso de hífen entre os termos, visto que a pontuação é ignorada. Para alguns casos é interessante que essas variações sejam retornadas. Foi utilizado também o operador de proximidade W/n (“Within”) que retorna trabalhos onde os termos descritos são encontrados a uma distância de no máximo “n” termos.

A expressão de busca foi aplicada aos títulos, resumos (abstracts) e palavras-chave dos artigos.

2.7.2.6 Critérios de Inclusão e Exclusão de Artigos

Os critérios de inclusão adotados foram:

- Os artigos devem estar disponíveis na Web;
- Os artigos devem estar escritos em inglês;
- Os artigos devem descrever uma abordagem de testes para aplicações Web;
- Os artigos devem apresentar um exemplo, aplicação, prova de conceito ou estudo sobre a abordagem proposta.

Os critérios de exclusão adotados foram:

- artigos que não apresentam abordagem de teste para aplicações Web;
- artigos que apresentam apenas o apoio automatizado sem se preocupar em descrever esta abordagem;
- artigos que apresentam apenas uma estratégia de geração automática de casos de teste;
- artigos que apresentam apenas uma estratégia de geração automática de dados de teste;
- abordagens que dependam de alguma forma de transformação dos modelos em outras representações, como redes de Petri, algoritmos, representação matemática ou cadeia de Markov;
- abordagens baseadas em código de teste ou baseadas em agentes;
- artigo duplicado;
- artigos que não apresentam nenhum tipo de avaliação;
- artigos não disponíveis na Web ou não disponibilizados pelos autores.
- Artigos que descrevem chamadas de eventos (*"proceedings"*)

2.7.2.7 Procedimentos de seleção

O procedimento de seleção será dividido em duas etapas. A primeira etapa consiste na análise dos títulos e *abstracts* de todos os artigos retornados na pesquisa quanto aos critérios de inclusão e exclusão. Entretanto, nesse momento não foi avaliado se o trabalho apresentava algum estudo envolvendo a abordagem de teste proposta, pois muitas vezes essa informação não está descrita nos *abstracts*. Após a avaliação, os artigos foram classificados como Candidato à Inclusão, Excluído ou Indefinido. Foram excluídos os trabalhos que apresentavam qualquer característica descrita nos critérios de exclusão. Um pesquisador irá executar este procedimento que deverá ser revisado por um segundo pesquisador por amostragem. Não havendo consenso sobre determinado estudo, o mesmo será classificado como indefinido.

Na segunda etapa do procedimento de seleção, todos os artigos classificados como Candidato à Inclusão ou Indefinido foram analisados qualitativamente e novamente avaliados sob a perspectiva da presença dos critérios de inclusão ou ausência dos critérios de exclusão. Apenas nesse momento foi avaliado se o trabalho apresentava algum estudo envolvendo a abordagem de teste proposta. Após esta análise eles foram classificados como Incluído, Excluído ou Indefinido. Foram

excluídos os trabalhos que apresentavam qualquer característica descrita nos critérios de exclusão. Esta etapa foi executada por um pesquisador e revisada por outro envolvido no estudo, por amostragem. Não havendo consenso sobre algum trabalho, o mesmo foi classificado como indefinido.

A estratégia de extração dos dados foi aplicada aos artigos classificados como Incluído ou Indefinido que foram caracterizados sob as dimensões definidas pelas questões secundárias apresentadas a seguir.

2.7.2.8 Questões Secundárias

A partir do universo de artigos selecionados na condução da revisão, foram analisadas diferentes características relacionadas às abordagens de teste para aplicações Web. Essa análise foi direcionada pelas questões secundárias da pesquisa, descritas a seguir:

QS1) A abordagem utiliza alguma ferramenta de apoio?

As abordagens de teste normalmente requerem passos complexos e um dos fatores que facilitam a sua utilização é a existência de apoio ferramental. Nesse contexto, as abordagens de teste serão categorizadas quanto à existência de algum apoio ferramental para sua utilização.

É importante destacar que as abordagens selecionadas apresentam algum método, estratégia ou processo para testar aplicações Web e que as ferramentas utilizadas devem ser instrumentos de apoio que facilitam sua execução. A ausência de ferramentas não deve inviabilizar a execução da abordagem.

As ferramentas de teste normalmente automatizam algumas tarefas requeridas pelo processo proposto, auxiliando, por exemplo, na geração dos casos de teste, na execução dos testes e na avaliação dos resultados. Além disso, as ferramentas de teste podem apoiar na produção de documentação de teste na sua gestão de configuração (DI LUCCA e FASOLINO, 2006).

QS2) A abordagem é aplicável para quais categorias de aplicações que utilizam a Web?

Os trabalhos selecionados foram classificados quanto à categoria da aplicação na qual a abordagem pode ser aplicada. Na revisão realizada foi considerado como categorias de aplicações Web aquelas definidas por GINIGE E MURUGESAN (2001) e KAPPEL (2004): Sistemas baseados no conhecimento, *workflow*, sistemas ubíquos, comércio eletrônico, sistemas colaborativos, dentre outras detalhadas no item 2.3 deste trabalho.

QS3) Quais características de qualidade são avaliadas pela abordagem?

Foi indicado o conjunto de características de qualidade descritas na norma ISO/IEC 9126 (2002) (funcionalidade, eficiência, usabilidade, confiabilidade, portabilidade e a subcaraterísitca segurança), que a abordagem de teste para aplicações Web é capaz de avaliar, considerando o mapeamento descrito na Tabela 2.1.

QS4) Qual nível de abstração de teste é usado pela abordagem?

As abordagens de teste para aplicações Web também foram classificadas de acordo com o nível de teste no qual elas podem ser utilizadas. Foram considerados testes de unidade, teste de integração, teste de sistema e teste de aceitação.

Apesar de não ser considerado um nível de teste, o Teste de Regressão também foi tratado como tal para fins dessa classificação. Esse tipo de teste é realizado em cada manutenção do sistema com o objetivo de garantir que os novos requisitos implementados não afetaram funcionalidades já existentes.

QS5) Qual técnica de teste é utilizada pela abordagem?

Essa classificação considerou o tipo de técnica de teste que é utilizada pela abordagem de teste para aplicações Web. Será Funcional quando a técnica utilizada considerar a aplicação a ser avaliada como uma caixa preta; e Estrutural (ou caixa branca) quando a abordagem considerar o código-fonte da aplicação ou modelos de cobertura que especificam a representação de elementos que deverão ser exercitados na atividade de teste.

A observação dos critérios de cobertura para explorar, exercitar e avaliar as aplicações sob análise será importante para ajudar a distinguir a técnica de teste utilizada pela abordagem.

QS6) Qual tipo de análise de teste é realizada pela abordagem?

O tipo de análise de teste que a abordagem realiza será classificado como Estático ou Dinâmico. Será Estático quando o comportamento da aplicação que se deseja avaliar não é influenciado pela interação do usuário. Caso contrário, será classificado como Dinâmico (ex.: avaliação de páginas que podem ser construídas dinamicamente em resposta a entradas de usuários).

QS7) A abordagem está inserida em alguma metodologia de desenvolvimento de aplicações Web (OOHDM, WebML, W2000, OOWS, OO-H, WAE, UWE, ADM, HDM, RMM)?

As abordagens de teste para aplicações Web podem ser associadas às metodologias de desenvolvimento como um mecanismo de integração entre as atividades e redução de esforço e custo para construção dos testes. Para classificar as abordagens retornadas no estudo, serão consideradas as metodologias de desenvolvimento Web identificadas a partir de estudo secundário realizado inicialmente por CONTE et. al. (2005) e atualizado por MASSOLAR (2008), destacadas no item 2.5.

QS8) Qual estudo envolvendo a abordagem de teste proposta foi descrito?

O foco da revisão está na identificação de estudos que descrevam abordagens de teste para aplicações web que apresentem algum relato de uso ou estudo experimental que retrate a aplicabilidade das abordagens identificadas. Nesse sentido, os trabalhos selecionados serão classificados de acordo com o estudo apresentado, conforme detalhado a seguir:

- Estudo de caso: São estudos que investigam um fenômeno contemporâneo dentro de um contexto real, especialmente quando os limites do fenômeno e do contexto não são claros (YIN, 2003). São conduzidos com o propósito de se investigar uma entidade ou um fenômeno dentro de um espaço de tempo específico (MAFRA et al., 2006).
- Estudo Experimental: É uma forma de estudo onde o pesquisador tem controle sobre algumas das condições na qual o estudo está sendo executado e também sobre as variáveis que estão sendo estudadas. Representam a idéia geral de estudos em Engenharia de Software. Entretanto, para serem considerados como um experimento no sentido geral do termo, é preciso considerar a aleatoriedade na escolha da população, o que é muito difícil em Engenharia de Software. (WIKI – ESE, 2011).
- Relato de Uso (Exemplo ou Prova de Conceito): Qualquer relato de uso demonstrando a aplicação da abordagem através de um exemplo ou prova de conceito.

2.7.2.9 Extração de Informações

Para analisar os estudos retornados na pesquisa é necessário que as questões secundárias sejam respondidas. Para isso, será extraído do conteúdo de cada artigo selecionado as informações abaixo listadas. Quando cabível, foi identificada a questão secundária que orientou a extração da respectiva informação.

- Título do Artigo
- Autores
- Fonte
- Ano de Publicação
- Descrição da abordagem de teste para aplicações Web
- Ferramenta utilizada para automação da abordagem (QS1)
- Categoria aplicável (QS2)
- Características de Qualidade avaliadas pela abordagem de teste proposta (QS3)
- Nível de abstração de teste (sistema, integração, unidade, aceitação, regressão) usado pela abordagem (QS4)
- Técnica de Teste (Funcional, Estrutural) utilizada pela abordagem (QS5)
- Tipo de análise de teste (Estática, Dinâmica) realizada pela abordagem (QS6)
- Metodologia de desenvolvimento de aplicações Web (QS7)
- Tipo do Estudo utilizado para avaliar a abordagem (QS8)

2.7.3 Execução da Revisão

Ao longo da pesquisa, o protocolo foi executado duas vezes:

- **Agosto de 2009**. Na primeira execução foram retornados 564 trabalhos. O procedimento de seleção foi dividido em duas etapas. A primeira etapa consistiu na análise de todos os "*abstracts"* dos artigos retornados na pesquisa quanto aos critérios de exclusão e inclusão, exceto aquele que determina a apresentação de um estudo sobre a abordagem proposta. Dessa forma, 60 trabalhos puderam ser classificados como Indefinidos, 122 como Candidatos a Inclusão e 382 foram excluídos.

  No entanto, dos 182 trabalhos selecionados (Candidatos a Inclusão + Indefinidos), 62 não permitiam acesso através do Portal de Periódicos da CAPES. Dentro desse conjunto, foi possível identificar o e-mail dos autores de 47 artigos. No período de 27 a 28 de setembro de 2009, foram enviados e-mails para os

autores solicitando o envio dos trabalhos não disponíveis. Destes, 13 autores enviaram cópias de seus trabalhos.

Na segunda etapa do procedimento de seleção, todos os artigos classificados como Candidato à Inclusão ou Indefinido foram analisados qualitativamente e avaliados sob a perspectiva dos critérios de inclusão e exclusão. Nesse momento, 12 trabalhos foram classificados como Indefinidos, 59 foram Incluídos e 111 foram excluídos. Desta forma, 71 artigos foram selecionados (Incluídos + Indefinidos).

Na pesquisa realizada, foram consideradas como categorias de aplicações Web aquelas definidas por GINIGE e MURUGESAN (2001) e por KAPPEL (2004): Sistemas baseados no conhecimento, *workflow*, sistemas ubíquos, comércio eletrônico, sistemas colaborativos. Em complemento, buscou-se ainda por sistemas baseados em agentes, *e-government*, sensores de rede, aplicações cliente-servidor, multicamadas, grids computacionais e aplicações *Network sensor* devido ao entendimento de que tais aplicações estão inseridas na definição de aplicações Web adotada nesse trabalho. Entretanto, após a análise do resultado da busca, foi possível observar que alguns trabalhos retornados em razão da utilização desses termos, na maioria das vezes, não estavam relacionados às aplicações Web. Nesse sentido, o protocolo foi evoluído quantos aos termos utilizados e foi elaborada uma nova string de busca. Foram removidas as palavras-chaves "*Network sensor", "client-server", "multi-layer"* e *"grid computing"*. Além disso, os termos referentes às categorias de aplicações identificadas por GINIGE e MURUGESAN (2001) e por KAPPEL (2004) foram associados à palavra Web para que os trabalhos retornados em razão da utilização desses termos estejam relacionados às aplicações Web. Segue abaixo a nova string de busca com as alterações mencionadas:

TITLE-ABS-KEY((({Web Application} OR {Web Applications} OR {Web system} OR {Web systems} OR "Web-based system" OR "Web-based application" OR {Web software} OR {Web Service} OR {Web Services} OR {Internet application} OR {Internet Applications} OR {e-commerce} OR {e-government} OR (("agent-based system" OR "agent-based software" OR "agent-based application" OR "Knowledge-based system" OR "Knowledge-based software" OR "Knowledge-based application" OR "Workflow-based system" OR "Workflow-based software" OR "Workflow-based application" OR {Ubiquitous System} OR {Ubiquitous Systems} OR {Ubiquitous Software} OR {Ubiquitous application} OR {Ubiquitous applications} OR {Collaborative System} OR {Collaborative Systems} OR

{Collaborative Software} OR {Collaborative application} OR {Collaborative applications})**AND Web**))AND (({Software Test} OR {Software Testing} OR {System Test} OR {System Testing} OR (web W/2 test*)) AND (process OR approach OR method OR technique OR methodology OR strategy))))

Verificou-se que todos os trabalhos selecionados na primeira execução retornaram após a modificação da string de busca.

- **Setembro de 2010:** Na re-execução da revisão, utilizando o protocolo evoluído, foram retornados mais 237 novos trabalhos, dentre os quais 43 classificados como Indefinidos, 48 como Candidatos a Inclusão e 146 foram excluídos após a análise do título e "*abstract"*. Dos 91 trabalhos selecionados (Candidatos a Inclusão + Indefinidos), 24 não permitiam acesso através do Portal de Periódicos da CAPES. Entretanto, para todos eles foi possível identificar o e-mail dos autores. No período de 26 a 28 de outubro de 2010, foram enviados e-mails para os autores solicitando o envio dos trabalhos não disponíveis. Destes, 8 autores enviaram cópias de seus trabalhos.

  Na segunda etapa do procedimento de seleção, todos os artigos classificados como Candidato à Inclusão ou Indefinido foram analisados qualitativamente e avaliados sob a perspectiva dos critérios de inclusão e exclusão. Nesse momento, 5 trabalhos foram classificados como Indefinidos, 25 foram Incluídos e 61 foram excluídos. Desta forma, 30 artigos foram selecionados (Incluídos + Indefinidos).

A lista completa dos artigos identificados ao longo das duas execuções do protocolo da revisão está disponível no Apêndice A deste trabalho, onde é possível verificar o critério que motivou a exclusão de cada publicação. A Tabela 2.2 informa os motivos de exclusão e quantidade de artigos associados.

Tabela 2.2. Classificação dos Artigos Excluídos.

Motivo da Exclusão	Quantidade de Artigos
Não apresenta abordagem de teste para aplicações Web	359
Apresenta apenas o apoio automatizado sem se preocupar em descrever esta abordagem	52
Apresenta estratégia de geração automática de casos de teste.	26
Apresenta estratégia de geração automática de dados de teste.	07
Abordagem depende de alguma forma de transformação dos modelos em outras representações, como redes de Petri, algoritmos, representação matemática ou cadeia de Markov	40
Abordagens baseadas em código de teste ou baseadas em agentes;	14
Artigo duplicado	38

Não apresenta nenhum tipo de avaliação	41
"*Proceedings*"	58
Artigo não disponibilizado	65
Total:	**700**

A partir do universo dos 101 artigos selecionados nas execuções da revisão, as informações e características relacionadas a cada abordagem de teste para aplicações Web foram extraídas e analisadas. A lista das abordagens identificadas é apresenta na Tabela 2.3, que apresenta uma categorização inicial das abordagens de teste selecionadas. Na seção 2.7.4, será apresentado um detalhamento das análises realizadas. Esses dados foram utilizados na construção da estrutura de caracterização (corpo de conhecimento) de tais abordagens cuja proposta e avaliação serão apresentados no Capítulo 3.

Tabela 2.3. Caracterização das Abordagens WAT.

Autores	Categoria	Característica de Qualidade	Ferramenta	Nível de Teste	Técnica de Teste	Tipo de Análise	Metodolog. de Desenv.	Tipo de Estudo
Amyot et al. 2005	Browser-Based	Funcionalidade Confiabilidade	Sim	Aceitação	Funcional	Estático	-	Estudo de Caso
Anandhan et al. 2006	Browser-Based	Usabilidade	Não	Aceitação	Funcional	Dinâmico	-	Estudo de Caso
Andreolini et al. 2005	Browser-Based	Eficiência	Não	Sistema	Funcional Estrutural	Dinâmico	-	Estudo de Caso
Andrews et al. 2005	Browser-Based	Funcionalidade Confiabilidade	Sim	Sistema	Funcional	Dinâmico	-	Exemplo
Antunes e Vieira 2009	Serviços Web	Segurança	Sim	Sistema	Estrutural	Dinâmico	-	Estudo Experimental
Avritzer e Weyuker 2004	e-commerce	Eficiência	Sim	Sistema	Funcional	Dinâmico	-	Estudo de Caso
Barber 2004	Browser-Based	Eficiência	Não	Sistema	Funcional	Dinâmico	-	Aplicação
Baresi et al. 2005	Browser-Based	-	Sim	Sistema	Funcional	Dinâmico	WebML	Estudo de Caso
Bartolini et al. 2009	Serviços Web	-	Não	Unidade Integração	Funcional	Dinâmico	-	Estudo de Caso
Bellettini et al. 2005	Browser-Based	Funcionalidade	Sim	Sistema	Estrutural	Estático Dinâmico	-	Estudo de Caso
Belli and Linschulte 2008	Serviços Web	Eficiência Segurança	Não	Sistema	Funcional	Dinâmico	-	Estudo de Caso
Belli and Linschulte 2010	Serviços Web	Confiabilidade	Sim	Sistema	Estrutural	Estático	-	Estudo de Caso
Bertolino et al. 2006	Serviços Web	Confiabilidade	Não	Sistema	Funcional	Dinâmico	-	Exemplo
Bezemer et al 2009	AJAX	Segurança	Sim	Sistema	Funcional	Dinâmico	-	Estudo de Caso
Cavalli et al. 2007	e-learning	Eficiência Funcionalidade	Sim	Sistema Regressão	Funcional	Dinâmico	-	Aplicação
Chang 2006	Serviços Web e-Learning	Usabilidade	Sim	Sistema	Funcional	Dinâmico	-	Exemplo
Chengying 2008	Serviços Web	Funcionalidade	Sim	Sistema	Funcional	Dinâmico	-	Exemplo
Cho et al. 2005	Browser-Based	Funcionalidade Confiabilidade	Sim	Unidade Integração	Estrutural	Dinâmico	WebML	Exemplo
Ciampa et al 2010	Browser-Based	Segurança	Sim	Sistema	Funcional	Dinâmico	-	Estudo Experimental
Conroy et al. 2007	Serviços Web	Funcionalidade	Sim	Unidade Regressão	Funcional	Estático	-	Estudo Experimental
Draheim et al. 2006	Browser-Based	Eficiência	Sim	Sistema	Funcional	Dinâmico	-	Exemplo
Eaton and Memon 2007	Browser-Based	Portabilidade	Sim	Sistema	Estrutural	Estático	-	Estudo Experimental
Elbaum et al 2003	Browser-Based	Funcionalidade	Sim	Sistema	Funcional	Dinâmico	-	Estudo Experimental
Elbaum et al 2005	Browser-Based	Funcionalidade	Sim	Sistema	Funcional	Dinâmico	-	Estudo Experimental
Endo et al. 2008	Serviços Web	-	Sim	Integração	Estrutural	Estático	-	Estudo de Caso
Feng et al. 2004	Browser-Based	Confiabilidade Funcionalidade Segurança	Não	Unidade Integração Regressão	Funcional Estrutural	Dinâmico	-	Exemplo
Fu et al. 2004	Serviços Web	Confiabilidade	Sim	Unidade	Estrutural	Estático	-	Prova de Conceito
Fugini et al. 2008	Serviços Web	Confiabilidade Eficiência Funcionalidade	Sim	Integração	Funcional	Dinâmico	-	Estudo de Caso
Gutiarrez et al 2006	Browser-Based	Confiabilidade	Sim	Unidade	Funcional	Dinâmico	-	Estudo de Caso
Halfond 2009	Browser-Based	Segurança	Sim	Sistema	Funcional Estrutural	Estático	-	Estudo Experimental

Hanna and Munro 2008	Serviços Web	Confiabilidade	Sim	Sistema	Funcional	Dinâmico	-	Prova de Conceito
Haydar et al. 2008	Browser-Based	Funcionalidade Confiabilidade Segurança Usabilidade	Sim	Sistema	Funcional	Dinâmico	-	Estudo Experimental
Huang et al. 2004	Browser-Based	Segurança	Sim	Unidade	Funcional	Dinâmico	-	Estudo Experimental
Jiang e Jiang 2009	Browser-Based	Eficiência	Sim	Sistema	Funcional	Dinâmico	-	Estudo Experimental
Jokhio et al. 2009	Serviços Web	Funcionalidade	Sim	Sistema	Funcional	Estático	-	Exemplo
Almeida Júrior and Vergilio 2006	Serviços Web	Confiabilidade	Sim	Integração Regressão	Funcional	Dinâmico	-	Estudo de Caso
Karam et al. 2006	Workflow	Confiabilidade	Não	Unidade Integração	Estrutural	Estático	-	Estudo de Caso
Karam et al. 2007	Serviços Web	-	Não	Integração	Estrutural	Dinâmico	-	Exemplo
Keum et al. 2006	Serviços Web	-	Não	Sistema	Estrutural	Dinâmico	-	Exemplo
Koopman et al. 2007	Browser-Based	Confiabilidade	Sim	Sistema	Estrutural	Estático	-	Exemplo
Kuk e Kim 2009	Serviços Web	Confiabilidade Funcionalidade	Não	Integração	Funcional	Dinâmico	-	Estudo de Caso
Lertphumpanya and Senivongse 2008	Serviços Web Workflow	Confiabilidade	Sim	Integração	Estrutural	Estático	-	Estudo de Caso
Li et al. 2008 (a)	Serviços Web	Confiabilidade	Não	Sistema	Funcional	Dinâmico	-	Estudo Experimental
Li e Miao 2008	Browser-Based	Funcionalidade Confiabilidade Segurança	Não	Sistema	Funcional	Estático	-	Exemplo
Li et al. 2008 (b)	Browser-Based	Funcionalidade	Não	Sistema	Funcional	Dinâmico	-	Exemplo
Li et al. 2008	Browser-Based	Funcionalidade Confiabilidade	Sim	Sistema	Funcional	Dinâmico	-	Exemplo
Li et al. 2007	Browser-Based	Confiabilidade	Sim	Sistema	Funcional	Dinâmico	-	Estudo Experimental
Li et al. 2005	Serviços Web	Confiabilidade	Não	Unidade	Estrutural	Estático	-	Exemplo
Li e Sun 2006	Serviços Web	Funcionalidade	Sim	Unidade Regressão	Funcional	Dinâmico	-	Exemplo
Li et al. 2009	Serviços Web	Funcionalidade Confiabilidade	Sim	Sistema Regressão	Funcional	Dinâmico	-	Exemplo
Lin et al. 2006	Serviços Web	Funcionalidade	Sim	Regressão	Funcional	Dinâmico	-	Exemplo
Lin et al. 2009	Browser-Based	Segurança	Sim	Unidade	Funcional	Dinâmico	-	Estudo Experimental
Liu et al. 2008	Serviços Web	Confiabilidade	Não	Integração	Estrutural	Estático	-	Exemplo
Liu et al. 2001	Browser-Based	Funcionalidade Confiabilidade	Não	Unidade Integração	Estrutural	Estático Dinâmico	-	Exemplo
Liu et al. 2000	Browser-Based	Funcionalidade Confiabilidade	Não	Unidade Integração Sistema	Estrutural	Estático	-	Exemplo
Louw e Venkatakrishnan 2009	Browser-Based	Segurança	Sim	Sistema	Funcional	Dinâmico	-	Estudo Experimental
Lucca et al. 2002	Browser-Based	Funcionalidade Segurança	Sim	Unidade Integração	Funcional	Estático Dinâmico	-	Estudo de Caso
Lucca et al. 2006	Browser-Based	Funcionalidade	Sim	Sistema	Funcional	Dinâmico	-	Estudo de Caso
Luo et al. 2009	Browser-Based	Confiabilidade	Sim	Sistema	Estrutural	Dinâmico	-	Estudo Experimental
Mao 2009	Serviços Web	Confiabilidade	Não	Unidade Integração	Funcional	Dinâmico	-	Estudo de Caso
Marchetto 2008	Browser-Based	Funcionalidade Confiabilidade	Sim	Sistema	Funcional	Dinâmico	-	Estudo Experimental
Marchetto et al. 2008 (a)	AJAX	Confiabilidade Funcionalidade Segurança Portabilidade	Não	Sistema	Funcional	Dinâmico	-	Estudo de Caso
Marchetto et al. 2008 (b)	Browser-Based	Confiabilidade Usabilidade Funcionalidade	Não	Aceitação	Funcional	Dinâmico	-	Estudo Experimental
Mayer e Labke 2006	Serviços Web	Funcionalidade	Não	Unidade	Estrutural	Estático	-	Exemplo
Mesbah e Deursen 2009	AJAX	Confiabilidade	Sim	Unidade Regressão	Estrutural	Dinâmico	-	Estudo de Caso
Mostefaoui e Simpson 2008	Serviços Web	Eficiência Funcionalidade Segurança	Sim	Unidade Regressão	Funcional Estrutural	Estático Dinâmico	-	Exemplo
Noikajana e Suwannasart 2008	Serviços Web	Confiabilidade Funcionalidade	Sim	Sistema	Funcional	Dinâmico	-	Exemplo
Offutt et al. 2008	Browser-Based	Confiabilidade Segurança	Sim	Sistema	Funcional	Dinâmico	-	Estudo de Caso
Offutt et al. 2004	Browser-Based	Confiabilidade Segurança	Não	Sistema	Funcional	Dinâmico	WebML	Estudo Experimental
Pentafronimos et al. 2008	Serviços Web e-government	-	Não	Sistema	Funcional	Dinâmico	-	Exemplo
Perumal e Dhavachelvan 2008	Browser-Based	Eficiência	Sim	Sistema	Funcional	Dinâmico	-	Estudo Experimental
Probert et al. 2003	e-commerce	Funcionalidade Segurança	Não	Unidade	Funcional	Dinâmico	-	Estudo de Caso

		Eficiência Confiabildiade						
Raffelt et al. 2008	Browser-Based	-	Sim	Regressão	Funcional Estrutural	Estático Dinâmico	-	Exemplo
Rajan et al. 2009	Browser-Based	Funcionalidade Segurança	Sim	Unidade	Estrutural	Estático	-	Estudo de Caso
Reza et al. 2008	Browser-Based	Funcionalidade Confiabildiade	Não	Unidade Integração	Estrutural	Dinâmico	-	Estudo de Caso
Ricca e Tonella 2002	Browser-Based	-	Sim	Integração Regressão	Estrutural	Estático	-	Estudo de Caso
Ricca e Tonella 2001	Browser-Based	-	Sim	Regressão	Estrutural	Estático	RMM	Estudo de Caso
Ruth et al. 2007	Serviços Web	-	Não	Regressão	Estrutural	Estático	-	Estudo de Caso
Salva e Rabhi 2010	Serviços Web	Confiabilidade	Não	Sistema	Funcional	Dinâmico	-	Exemplo
Sampath et al. 2008	Serviços Web	-	Não	Regressão	Funcional	Dinâmico	-	Estudo Experimental
Shahriar e Zulkernine 2008	Browser-Based	Segurança Confiabilidade	Sim	Unidade	Estrutural	Dinâmico	-	Estudo Experimental
Shahriar e Zulkernine 2010	Browser-Based	Segurança	Sim	Sistema	Funcional	Dinâmico	-	Estudo Experimental
Shams et al. 2006	Browser-Based	Eficiência	Sim	Sistema	Funcional	Dinâmico	-	Estudo Experimental
Shimomura e Ikeda 2010	Browser-Based	Funcionalidade	Sim	Regressão	Estrutural	Dinâmico	-	Exemplo
Siblini e Mansour 2005	Serviços Web	Confiabilidade Funcionalidade	Não	Sistema	Funcional	Dinâmico	-	Exemplo
Solino e Vergilio 2009	Serviços Web	Confiabilidade	Sim	Sistema	Funcional	Dinâmico	-	Estudo de Caso
Sprenkle et al. 2005	Browser-Based	Confiabilidade	Sim	Sistema	Funcional	Dinâmico	-	Estudo de Caso
Sun et al. 2009	AJAX	Segurança	Sim	Sistema	Funcional	Dinâmico	-	Estudo Experimental
Tappenden e Miller 2008	Browser-Based	Confiabildiade Segurança	Não	Sistema	Estrutural	Estático	-	Exemplo
Tarhini et al. 2008	Browser-Based	-	Não	Regressão	Fucnional	Dinâmico	-	Estudo de Caso
Tarpe et al. 2008	Browser-Based	Segurança	Sim	Sistema	Funcional Estrutural	Dinâmico	-	Exemplo
Tian e Ma 2006	Browser-Based	Confiabilidade Funcionalidade Eficiência Usabilidade	Não	Unidade Integração	Funcional	Dinâmico	-	Estudo de Caso
Tonella e Ricca 2002	Browser-Based	Confiabilidade	Não	Sistema	Estrutural	Estático Dinâmico	-	Estudo de Caso
Tsai et al. 2005	Serviços Web	Funcionalidade	Sim	Unidade Integração	Funcional	Dinâmico	-	Estudo de Caso
Wang e Huang 2008	Serviços Web	Confiabilidade	Não	Integração	Funcional	Dinâmico	-	Estudo de Caso
Wu et al. 2010	Browser-Based	Segurança Confiabilidade	Sim	Unidade	Estrutural	Estático	-	Exemplo
Xie et al. 2007	Serviços Web	-	Sim	Unidade	Funcional	Dinâmico	-	Exemplo
Yang et al. 2010	Serviços Web	-	Sim	Integração Regressão	Funcional	Dinâmico	-	Estudo de Caso
Yu et al. 2009	Browser-Based	Funcionalidade Confiabilidade	Não	Aceitação	Funcional	Dinâmico	-	Estudo Experimental
Zhang et al. 2007	Serviços Web	-	Não	Sistema	Estrutural	Dinâmico	-	Estudo de Caso
Zheng e Chen 2007	Browser-Based	-	Sim	Regressão	Estrutural	Dinâmico	-	Estudo Experimental

A Figura 2.3 apresenta a divisão dos artigos selecionados por ano de publicação, na qual é possível observar o aumento da quantidade de artigos descrevendo abordagens WAT até 2008. O declínio observado nos anos seguintes pode ser atribuído à demora na publicação dos trabalhos nas bibliotecas digitais. Isso foi observado nas duas execuções da revisão: A primeira execução, realizada em agosto de 2009, não retornou trabalhos publicados no referido ano, que só foram capturados na execução realizada em setembro de 2010.

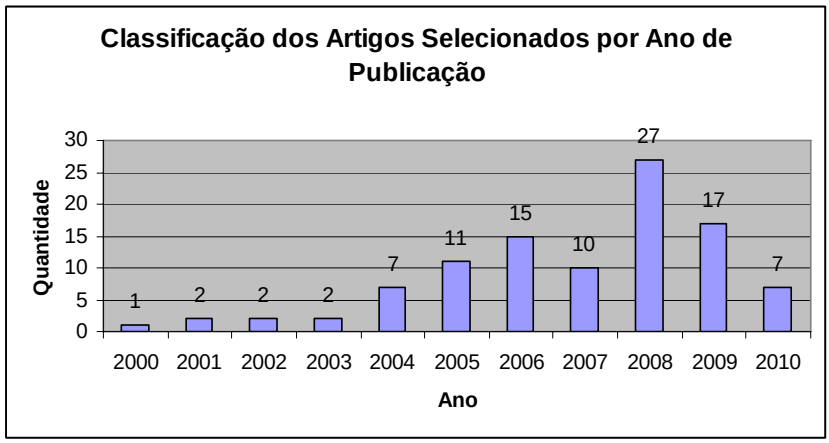


Figura 2.3. Classificação dos Artigos Selecionados por Ano de Publicação.

2.7.3.1 Ameaças à Validade

Como ameaças à validade do estudo, ressaltamos que não foi possível recuperar 49 dos 182 artigos selecionados na primeira etapa da primeira execução da revisão. Isto porque, não foi possível identificar os e-mails dos autores de 15 trabalhos e 34 não foram disponibilizados pelos autores identificados. Pela mesma razão, não foram recuperados 16 trabalhos selecionados na reexecução da revisão. Entretanto, é possível considerar que os artigos incluídos apresentam representatividade e trazem informação sobre um conjunto abrangente de abordagens de teste de software aplicáveis a projeto de software Web.

Outra ameaça identificada está relacionada à execução da string de pesquisa apenas na máquina Scopus. Entretanto, destaca-se que trabalhos executados ao longo dos anos pelo grupo de Engenharia de Software Experimental da COPPE/UFRJ identificaram que a cobertura oferecida pela Scopus é considerada suficiente, na medida que ela retorna maior parte dos artigos existentes em diferentes bibliotecas digitais ou outras máquinas de busca (Compendex, IEEE , ACM Digital Library, Springer, Web of Science)

2.7.4 Análises dos Resultados

A partir do universo de artigos selecionados na execução da revisão, foram analisadas diferentes características relacionadas às abordagens de teste para aplicações Web. Essa análise foi direcionada pelas questões secundárias da pesquisa, descritas a seguir:

QS1: A abordagem utiliza alguma ferramenta de apoio?

QS2: A abordagem é aplicável para quais categorias de aplicações que utilizam a Web?

QS3: Quais características de qualidade são avaliadas pela abordagem?

QS4: Qual nível de abstração de teste é usado pela abordagem?

QS5: Qual técnica de teste é utilizada pela abordagem?

QS6: Qual tipo de análise de teste é realizada pela abordagem?

QS7: A abordagem está inserida em alguma das metodologias conhecidas para o desenvolvimento de aplicações Web (OOHDM, WebML, W2000, OOWS, OO-H, WAE, UWE, ADM, HDM, RMM)?

QS8: Foi descrito algum estudo envolvendo a abordagem de teste proposta?

Esquema de classificação por ferramenta de apoio (QS1)

A Tabela 2.4 apresenta o resultado da análise quanto à existência de ferramenta de apoio nas abordagens WAT selecionadas. Dentre os 101 artigos selecionados, 64 apresentam algum tipo de apoio ferramental. As ferramentas utilizadas foram classificadas em 2 tipos: protótipos e ferramenta comercial. Verificamos que a maior parte das abordagens de teste selecionada (aproximadamente 75%) faz uso de protótipos.

Uma análise complementar pode ser realizada no sentido de verificar quais protótipos foram avaliados através de algum estudo, ou ainda, verificar se alguma ferramenta foi utilizada ou referenciada em outros trabalhos, reforçando a idéia da viabilidade de seu aproveitamento em algum projeto.

Tabela 2.4. Análise das Abordagens pelo uso de Ferramentas.

Faz uso de ferramentas	Não há indicação de apoio ferramental
64	37

A utilização de ferramentas indica que tais abordagens automatizam algumas tarefas requeridas pelo processo proposto. Nesse sentido, é interessante realizar uma análise complementar para verificar o nível de complexidade dos passos não automatizados para estimar o esforço, custo e tempo de execução da abordagem WAT.

Esquema de classificação por categoria da aplicação (QS2)

Após a análise preliminar do resultado da busca, foi possível observar que os trabalhos retornados em razão da utilização dos termos associados às categorias, na maioria das vezes, não estavam relacionados às aplicações Web. A Tabela 2.5

apresenta o resultado da análise quanto ao tipo da aplicação que a abordagem pode ser aplicada.

Tabela 2.5. Análise das Abordagens por Categoria da Aplicação.

Categoria	Quantidade de Artigos
Browser-Based	55
Serviços Web	38
Comércio Eletrônico	2
Workflow	1
Aplicações Web Colaborativas	2
Aplicações Web Ubíquas	0
Aplicações Web Baseadas no Conhecimento	0
Aplicações Web Baseadas em Agentes	0
e-gorvernment	1
RIAS (*Rich Internet Applications*)	4

Basicamente as categorias de aplicações identificadas puderam ser divididas em dois grandes grupos: abordagens de teste para aplicações web baseadas na utilização de *"browsers"* ou navegadores que não se enquadram em nenhuma classificação específica (55 artigos) e abordagens para o teste de Serviços Web (38 artigos).

Além dessas, foram identificadas abordagens de teste para aplicações de comércio eletrônico (2 artigos), *e-learning* (tipo colaborativo) (2 artigos), *workflow* (2 artigo) e *e-government* (1 artigo). Identificamos apenas 3 artigos relacionados a abordagens que podem ser aplicáveis a mais de uma categoria de aplicação web.

Foram encontradas ainda abordagens de teste para aplicações denominadas AJAX - *Asynchronous Javascript And XML*. Trata-se de aplicações que utilizam o recurso de solicitações assíncronas de informações, para tornar páginas Web mais interativas com o usuário.

Por essa razão, são classificadas como RIAS (*Rich Internet Applications*), ou seja, os aplicativos da Web que incluem interações avançadas e sofisticadas com o usuário, que não seriam possíveis de serem oferecidas no paradigma multipágina das aplicações Web tradicionais (MARCHETTO et al., 2008).

Por apresentar características específicas, foram identificadas na literatura técnica abordagens de teste igualmente específicas para essa classe de aplicações Web. Na revisão realizada foram identificados 4 artigos com tal direcionamento.

Esquema de classificação por característica de qualidade coberta (QS3)

A Tabela 2.6 apresenta o mapeamento entre características e número de trabalhos relacionados, no qual é possível observar que as abordagens WAT em sua maioria avaliam a confiabilidade e a funcionalidade das aplicações.

Tabela 2.6. Análise das Abordagens por Característica de Qualidade Coberta.

Característica de Qualidade	Quantidade de Artigos
Confiabilidade	47
Funcionalidade	38
Segurança	24
Eficiência	13
Usabilidade	5
Portabilidade	2
Não Identificado	16

Na avaliação da confiabilidade, grande parte das abordagens verifica o fluxo de navegação das aplicações quanto ao processamento correto e a integridade dos *links* (para evitar páginas inacessíveis, *links* quebrados ou sem ponteramento adequado), enquanto que, na avaliação da funcionalidade, é verificado o comportamento da aplicação de acordo com os requisitos funcionais e não funcionais.

Também merece destaque a quantidade de abordagens preocupadas com a segurança das aplicações. Neste caso, normalmente é avaliado o controle de acesso das aplicações para garantir o acesso às funcionalidades somente por pessoas autorizadas, as vulnerabilidades da aplicação e a sua capacidade de evitar “ataques” através da injeção de códigos maliciosos.

Existe ainda um grupo expressivo de abordagens que avalia a eficiência, preocupando-se com o desempenho das aplicações normalmente avaliado a partir de análises do tempo de resposta da aplicação em situações extremas (alto número de acesso concorrente). Tais abordagens são normalmente associadas à utilização de ferramentas para simular grande quantidade de acesso às aplicações.

Foram identificadas apenas 5 abordagens de teste para aplicações Web que avaliam a usabilidade, verificando a facilidade no uso da aplicação e aprendizagem das terminologias, menus e navegação através de diferentes páginas da Web; e 2 abordagens que avaliam a portabilidade das aplicações, buscando identificar erros associados à utilização de diferentes navegadores e sistemas operacionais.

Esquema de classificação por nível de teste (QS4)

As abordagens de teste para aplicações Web foram classificadas de acordo com o nível de teste no qual elas podem ser utilizadas. Esse mapeamento foi realizado considerando as definições dos níveis de teste encontrados na literatura ou a perspectiva do autor, quando explicitada na abordagem proposta. Foi observado que algumas abordagens cobrem mais de um nível de teste (exemplo: teste de unidade e integração). A Tabela 2.7 descreve os resultados da análise das abordagens WAT por nível de teste:

Tabela 2.7. Análise das Abordagens por Nível de Teste.

Nível de Teste	Quantidade de Artigos
Unidade	26
Integração	21
Sistema	54
Aceitação	4
Regressão	18

É possível observar uma maior incidência de abordagens projetadas para realizar Teste de Sistema, que em geral é realizado quando o sistema está completo, com todas as suas partes integradas. Existe um equilíbrio entre a quantidade de abordagens projetadas para realizar Teste de Unidade e Teste de Integração, não sendo rara a existência de abordagens que podem ser aplicadas nos dois níveis de teste.

Teste de Aceitação não é ainda um nível de abstração tão usual nas abordagens de teste para aplicações Web, apresentando a menor quantidade de abordagens proposta para avaliar tal nível de teste. Por outro lado, foi identificado um número expressivo de abordagens WAT que avaliam o Teste de Regressão, que em sua maioria, buscam identificar o conjunto de testes que devem ser executados para garantir a confiança da aplicação após a execução de manutenções.

Destaca-se, porém, que não há um senso comum sobre a definição de unidade nas aplicações Web, que por sua vez tem conseqüência nas definições de teste de unidade, integração e sistema. Isto porque o conceito da unidade da aplicação varia de acordo com as características da abordagem, como por exemplo, a categoria da aplicação Web.

Nas aplicações baseadas na utilização de *browsers*, boa parte das abordagens apresenta o entendimento semelhante a aquele apresentado em (DI LUCCA et al., 2006), onde é definido que as aplicações Web são compostas por páginas que podem ser de dois tipos: páginas clientes e páginas do servidor. As páginas Web podem ser

divididas por componentes, tais como formulários, texto, imagens, *scritps*, *cookies*, e integradas por *links*. Na abordagem proposta, o teste de unidade é realizado para validar os componentes de uma página Web, enquanto que os testes de integração são feitos considerando o relacionamento entre páginas.

No caso das aplicações categorizadas como serviços Web, em geral, são analisados o comportamento e operações dos serviços Web. Nas aplicações colaborativas, as unidades consideradas são: fórum, *faqs*, armazenamento de arquivos, enquanto que nas abordagens de teste para aplicações *e-commerce*, as unidades consideradas são transações de comércio eletrônico realizadas. Nesse sentido, foi realizada uma análise qualitativa para identificar as unidades de teste consideradas nas abordagens.

Esquema de classificação por técnica de teste (QS5)

As abordagens de teste para aplicações Web foram classificadas de acordo com o tipo de técnica que é utilizada pela abordagem. Esse mapeamento foi realizado considerando as definições das técnica de teste encontradas na literatura (apresentadas na Seção 2.6.2) e o critério de cobertura utilizado. A Tabela 2.8 descreve os resultados da análise das abordagens WAT por tipo de técnica de teste:

Tabela 2.8. Análise das Abordagens por Técnica de Teste.

Técnica de Teste	Quantidade de Artigos
Funcional (Caixa-Preta)	64
Estrutural (Caixa-Branca)	31
Ambas (Caixa-Cinza)	6

A técnica de teste funcional é utilizada pela maioria das abordagens WAT selecionadas na revisão. As abordagens de teste para serviços web, por exemplo, refletem a predominância da utilização da técnica de teste funcional, pois muitas vezes a estrutura do serviço não está disponível. Foram identificadas em menor número, abordagens que utilizam técnica de teste caixa cinza, ou seja, uma combinação entre as técnicas funcionais e estruturais, que considera tanto o comportamento da aplicação quando a sua estrutura interna.

Adicionalmente, foi realizada uma análise qualitativa para identificar os critérios de coberturas considerados pelas abordagens WAT, visto que tal informação também foi considerada no mapeamento apresentado.

Esquema de classificação por análise de teste (QS6)

As abordagens de teste para aplicações Web foram classificadas de acordo com o tipo de análise realizada. Esse mapeamento foi realizado considerando se o comportamento da aplicação que se deseja avaliar é ou não influenciado pela interação do usuário. A Tabela 2.9 descreve os resultados da análise das abordagens WAT por análise de teste:

Tabela 2.9. Análise das Abordagens por Tipo de Análise de Teste.

Tipo de Análise	Quantidade de Artigos
Dinâmica	72
Estática	23
Ambos	6

A análise dinâmica é utilizada pela maioria das abordagens WAT selecionadas na revisão, o que se justifica devido ao aspecto dinâmico que predomina neste tipo de aplicação.

Esquema de classificação por metodologia de desenvolvimento de aplicações Web (QS7)

Nesta pesquisa, foram encontradas apenas 4 abordagens de testes associadas a 2 metodologias de desenvolvimento Web identificadas a partir de estudo secundário realizado inicialmente por CONTE et. al. (2005) e atualizado por MASSOLAR (2008).

A WebML foi referenciada por 3 artigos, enquanto que a RMM foi citada por apenas um trabalho. As demais metodologias não foram referenciadas nos trabalhos selecionadas na revisão.

BARESI et al. (2005) utilizam a metodologia WebML, porém ressalta que os resultados obtidos independem da linguagem de modelagem utilizada. O modelo apresentado em OFFUT et al. (2004) é complementar ao WebML. CHO et al. (2005) propõem uma abordagem e a compara com a WebML. A metodologia proposta por RICCA e TONELLA (2001) é baseada na RMM.

Esquema de classificação de acordo com o estudo apresentado (QS8)

Esse mapeamento considerou a perspectiva do autor da abordagem sobre a avaliação realizada na proposta. Nesse contexto, em 38 artigos foi informado que a abordagem foi avaliada através de um estudo de caso, 26 foram avaliadas por estudos experimentais e 37 apresentaram algum relato de uso, como a apresentação de exemplos, aplicação ou prova de conceito, conforme descrito na Tabela 2.10.

Tabela 2.10. Análise das abordagens por Tipo de Estudo.

Tipo de Estudo	Quantidade de Artigos
Estudo de Caso	38
Estudo Experimental	26
Relato de Uso	37

2.8 Considerações Finais do Capítulo

Este capítulo apresentou a caracterização de abordagens de teste para aplicações Web, a partir dos resultados obtidos em uma quasi-revisão sistemática da literatura técnica que classificou as abordagens quanto à:

- Utilização de apoio ferramental,
- Categorias de aplicações cuja abordagem é aplicável,
- Característica de qualidade avaliada,
- Inserção da abordagem de teste em alguma metodologia de desenvolvimento de aplicações Web conhecidas e identificadas por trabalhos anteriores na área,
- Nível de abstração de teste utilizado,
- Técnica de Teste utilizada,
- Análise de teste realizada, e
- Tipo de estudo utilizado para avaliar a abordagem.

Existe, portanto, uma disponibilidade de diferentes estratégias de teste para aplicações Web e a caracterização de tais abordagens representa um importante passo para a organização de um corpo de conhecimento que possa ser utilizado para apoiar a seleção das técnicas de teste mais adequadas para um determinado projeto de software web.

No entanto, é importante destacar que as abordagens de teste para aplicações Web, selecionadas na quasi-Revisão Sistemática, foram classificadas de acordo com os elementos apresentados nos aritgos, que eventualmente podem não explicitar todas as características da abordagem proposta.

O próximo capítulo apresenta a construção e avaliação de um corpo de conhecimento de abordagens de teste para aplicações Web, seguindo a metodologia baseada em evidencia proposta em (DIAS NETO et al., 2010).

CAPÍTULO 3 - ESTRUTURA PARA A CARACTERIZAÇÃO DE ABORDAGENS DE TESTE DE APLICAÇÕES WEB

Este capítulo apresenta a proposta e avaliação de uma estrutura de caracterização de abordagens de teste para projetos de software Web organizada a partir dos resultados obtidos através de uma quasi-revisão sistemática da literatura e avaliada através de um survey realizado com pesquisadores da área. São apresentados trabalhos relacionados (seção 3.2), detalhes da estrutura de caracterização (seção 3.3), bem como o planejamento, execução e análise de resultados do survey que avaliou a estrutura proposta (seção 3.4).

3.1 Introdução

A revisão para identificar possíveis trabalhos sobre abordagens de teste para aplicações Web revelou a existência de várias publicações que focam na definição de novas estratégias para o teste deste tipo de aplicação. Portanto, como se depreende, existe disponibilidade de tecnologias de software, especificamente testes de software, que podem ser selecionadas para estas aplicações.

Neste contexto, VEGAS e BASILI (2005) descreveram um mecanismo para apoiar a seleção de técnicas de teste em geral, baseado em um esquema de caracterização, onde um catálogo de técnicas de teste de software é instanciado para um projeto em particular.

No entanto, o uso do referido esquema de caracterização em projetos de software Web pode levar a identificação de técnicas de teste não necessariamente adequadas, tendo em vista a generalidade dos atributos utilizados no esquema de caracterização frente às características específicas dos projetos de aplicações Web.

Situação semelhante foi observada por DIAS NETO (2009) em relação ao uso do esquema de caracterização para apoiar a seleção de técnicas de teste baseado em modelos para projetos de software.

Desta forma, considerando a diversidade de técnicas de teste disponível na literatura técnica e as características dos projetos de aplicação Web, a seleção de abordagens de teste para projetos Web necessita de investigação adicional e a organização de um corpo de conhecimento inicial sobre estas abordagens em

particular torna-se importante para aprimorar o procedimento de caracterização e seleção destas técnicas para projetos de software Web.

A partir dos resultados obtidos na quasi-revisão sistemática, é apresentada a estrutura de caracterização de abordagens de teste para aplicações Web. Antes, porém, são apresentados os trabalhos associados à caracterização de abordagens de teste de software identificados na literatura técnica, com o objetivo de auxiliar a escolha da abordagem a ser empregada na validação de uma aplicação.

3.2 Trabalhos Relacionados

3.2.1 Corpo de Conhecimento de técnicas de teste de software (V&B2005)

VEGAS e BASILI (2005) abordaram o problema relacionado à identificação de informações relevantes para a seleção de abordagens de teste, levando em consideração que as abordagens de teste são usadas para encontrar um conjunto adequado de casos de teste e que existem várias técnicas disponíveis na literatura técnica, algumas cujo uso nunca foi considerado e outras que são repetidamente utilizadas em diferentes projetos, sem que a adequação do seu uso seja avaliada.

Nesse contexto, foi apresentada uma proposta denominada *Esquema de Caracterização* que provê uma infra-estrutura para armazenar informações sobre técnicas de teste com o objetivo de auxiliar os testadores na escolha da técnica mais adequada para um determinado projeto (Tabela 3.1).

Tabela 3.1. Atributos utilizados no Esquema de Caracterização de Técnicas de Teste (VEGAS e BASILI, 2005).

Nível	Elemento	Atributo	Descrição
Operacional	Agentes	Conhecimento	Habilidade requerida para aplicação da técnica
		Experiência	Experiência requerida para aplicação da técnica
	Ferramentas	Identificador	Nome da ferramenta
		Automação	Parte da técnica automatizada pela ferramenta
		Custo	Custo de compra ou manutenção da ferramenta
		Ambiente	Plataforma (Software e Hardware)
		Suporte	Suporte provido pelo fabricante
	Técnicas	Comprensibilidade	O quanto a técnica é de fácil entendimento
		Custo da aplicação	Esforço para aplicação da técnica
		Entradas	Entradas requeridas para aplicação da técnica
		Critérios de Adequação	Critério de parada para geração de casos de teste
		Custo do dado do teste	Custo na identificação dos dados para aplicação da técnica
		Dependências	Relacionamentos com outras técnicas
		Repetibilidade	Se 2 ou mais pessoas geram os mesmos casos de teste

		Fontes de Informação	Onde encontrar mais informações sobre a técnica
	Casos de Teste	Completude	Cobertura provida pelos casos de teste
		Precisão	Quantidade de casos de teste repetidos gerados pela Técnica
		Efetividade	Capacidade de encontrar defeitos
		Tipo de defeito	Tipos de defeitos encontrados
		Nº de casos de testes gerados	Número de casos de testes gerados
	Objeto	Elemento	Elemento considerado em cada teste
		Aspecto	Funcionalidade a ser testada
		Tipo de Software	Tipo de software que pode ser testado pela técnica
		Linguagem de Programação	Linguagem de programação que pode ser utilizada
		Método de desenvolvimento	Método de desenvolvimento
		Tamanho	Tamanho do software
Histórico	Projeto	Projetos referenciados	Projetos anteriores que utilizaram a técnica
		Ferramentas utilizadas	Ferramentas utilizadas em projetos anteriores
		Equipe	Equipe que trabalhou nos projetos anteriores
	Satisfação	Opinião	Opinião geral sobre a aplicação da técnica
		Benefícios	Benefícios no uso da técnica
		Problemas	Problemas no uso da técnica

O modelo proposto foi muito útil como ponto de partida para começar a identificar os tipos de atributos gerais que são importantes para caracterizar as técnicas de teste. Entretanto, se consideramos algumas abordagens específicas de teste, alguns atributos específicos precisam ser adicionados ao esquema original para melhor caracterizá-las.

3.2.2 Corpo de Conhecimento de abordagens para MBT (DIAS NETO 2009)

Em DIAS NETO (2009), é apresentada uma solução para o problema da seleção de técnicas de teste baseado em modelos (TTBM) com base no esquema de caracterização de VEGAS e BASILI (2005), a partir do desenvolvimento do corpo de conhecimento sobre TTBMs e de um procedimento de seleção responsável por prover informações para a equipe de teste avaliar a adequabilidade e o impacto de TTBM a partir das características de um projeto de software onde elas seriam aplicadas.

A construção do corpo de conhecimento sobre TTBMs foi realizada a partir dos resultados de dois estudos, um estudo secundário com o propósito de definir o conteúdo do corpo de conhecimento e um estudo primário com propósito de definir a sua estrutura. A Tabela 3.2 apresenta a estrutura proposta para caracterizar técnicas de teste baseadas em modelos.

Tabela 3.2. Atributos utilizados para caracterizar uma TTBM (DIAS NETO, 2009).

Categoria	Atributo	Descrição
Ferramenta	Ferramenta de Apoio	Define a existência ou não de uma ferramenta de apoio. Caso exista, é importante observar quais passos são apoiados pela ferramenta e quais são suas limitações.
	Necessidade de Ferramenta Externa	Define a necessidade do uso de alguma ferramenta externa à TTBM para viabilizar sua aplicação (ex: ferramenta para geração do modelo usado pela TTBM para geração dos testes).
	Plataforma	Plataforma em que a ferramenta de apoio opera
	Custo	Custo associado à ferramenta de apoio.
Histórico	Tipos de Falhas	Tipos de Falhas que podem ser reveladas pela TTBM.
Objeto	Característica de Qualidade que a técnica está apta a avaliar	Normalmente, TTBMs estão aptas a avaliar apenas um conjunto restrito de características de qualidade do software (ex.: usabilidade, desempenho, segurança, funcionalidade, eficiência, confiabilidade, controle de acesso, etc.).
	Plataforma de Execução	Define a plataforma de execução no qual uma TTBM pode ser usada.
	Paradigma de Desenvolvimento	Define o paradigma de desenvolvimento no qual uma TTBM pode ser usada.
	Linguagem de Programação	Define a linguagem de programação na qual uma TTBM pode ser usada.
	Limitações/Restrições para utilizar a técnica	Essas informações dizem respeito ao que uma TTBM não está apta a fazer ou algum cenário onde ela não pode ser aplicada (ex: a técnica só pode ser aplicada em projetos com características específicas, tais como tamanho, habilidades, ciclo de vida). Isso pode restringir seu uso em projetos de software.
	Habilidades Requeridas	Habilidades requeridas para o uso da TTBM.
	Nível de Teste	Define qual nível de abstração de teste é usado pela TTBM para avaliar o software sob teste. Os níveis usados são: teste de sistema, integração e unidade.
	Tipo de Técnica de Teste	Algumas TTBMs são aplicadas para teste funcional e outras para teste estrutural. É importante observar esta característica para apoiar a seleção de uma técnica para um projeto.
Técnica	Critério de Cobertura de Teste	Define as regras usadas para gerar casos de teste a partir do modelo do software. Existem diversos tipos de critérios, como: fluxo de dados, fluxo de controle e análise de mutantes. Eles definem o esforço e qualidade dos resultados gerados automaticamente pela TTBM.
	Critério de Geração de Caso de Teste	Descreve os passos a serem seguidos pela TTBM desde a construção do modelo até a geração ou execução dos testes. Eles definem o que pode ser automatizado e o que não pode, e como integrar esses passos.
	Entradas requeridas	TTBMs comumente requerem diferentes entradas (artefatos) para iniciar o processo de construção do modelo para geração dos casos de teste. Normalmente, essas entradas possuem diferentes complexidades para serem interpretadas. É importante observar se o processo de desenvolvimento de software está apto a produzir as entradas necessárias para usar a TTBM.
	Modelo Comportamental/Estrutural	Representa as características do software que podem ser testadas por uma TTBM. O modelo pode ser a maior limitação de uma técnica, pois indica que informações do software podem ser representadas ou não. Algumas vezes o modelo é usado para um domínio de aplicação específico e não pode ser usado em outro contexto. Três aspectos são importantes a respeito dos modelos: (1) quão fácil e automatizado é seu desenvolvimento, (2) se ele contém as informações necessárias para a geração dos testes, e (3) quão fácil é para extrair casos de teste a partir dele.
	Nível de Complexidade dos passos não automatizados	Indicação a respeito do esforço, custo e tempo necessário para realizar os passos requeridos em uma TTBM. O grau de complexidade dos passos automatizados é normalmente baixo. No entanto, os passos não automatizados podem requerer diferente grau de esforço, custo e habilidade para ser realizado. Assim, torna-se necessário analisar seus graus de complexidade indicando mais precisamente o nível de automação de uma TTBM.
	Proporção de passos automatizados	Relação entre a quantidade de passos automatizados que compõem a TTBM e a quantidade de passos total. A principal característica de TBM é a possibilidade de geração e execução automática de casos de teste, pois isso pode resultar em menos tempo e esforço para o processo de testes.
	Resultados gerados pelas técnicas	TTBMs podem gerar resultados em diferentes formatos (ex: roteiros de teste a serem executados em plataformas específicas ou resultados dos testes sumarizados após a execução dos casos de teste). É importante observar se os resultados gerados estão de acordo com os artefatos planejados para o processo de desenvolvimento de software.
	Tecnologia de Geração dos Testes	Diferentes TTBMs usam diferentes tecnologias para geração dos casos de testes (ex: UML, Linguagem B, Especificação Z, TSL e outros). É

		importante observar se o processo de desenvolvimento usa a mesma tecnologia para modelagem do software que a adotada pela TTBM visando reduzir o esforço e risco associado ao uso de diferentes tecnologias em um mesmo projeto.
	Uso de Modelos Intermediários	Define se a TTBM adota ou não modelos intermediários entre o modelo comportamental/estrutural e os casos de teste para apoiar a geração automática de casos de teste. A existência de modelos intermediários pode introduzir um esforço adicional para a técnica, pois requer uma transformação do modelo original provido pelo processo de desenvolvimento para um novo modelo a ser construído pela equipe de teste.
	Avaliação Experimental	Indicador do tipo de avaliação experimental que a TTBM foi submetida.

3.3 Definição de um corpo de conhecimento para abordagens WAT

Nesta seção é definida uma estrutura adaptada às características das aplicações Web, a partir dos resultados obtidos na quasi-revisão sistemática. Foram também adaptados para este contexto, com base nos resultados da revisão, alguns atributos definidos no esquema de caracterização proposto por VEGAS e BASILI (2005) e na estrutura definida por DIAS NETO (2009).

Os níveis da estrutura de caracterização a serem utilizados são os mesmos propostos em (DIAS NETO, 2009), compostos por Atributo e Categoria do Atributo. Os atributos são agrupados em 4 categorias: “Ferramenta” que possui atributos para caracterizar as ferramentas utilizadas para apoiar as abordagens de teste, “Histórico” que possui atributos para caracterizar as abordagens de teste de acordo com os resultados obtidos com a sua aplicação, “Objeto” que possui diversos atributos para caracterizar o objeto de teste das abordagens (esses atributos são influenciados pelas características do projeto ou da aplicação); e a “Técnica” cujos atributos descrevem algumas características da abordagem que não são influenciadas pelas características do projeto ou da aplicação que será avaliada.

Tabela 3.3. Estrutura inicial para caracterizar as abordagens de teste para aplicações Web.

Categoria	Atributo	Descrição	Origem
Ferramenta	Disponibilidade de ferramentas de apoio	Informações sobre as ferramentas que apóiam a abordagem WAT, incluindo o custo da plataforma, tipo (protótipo / comercial / shareware / freeware).	V&B (2005) DN (2009)
Histórico	Citação em artigos científicos anteriores	Trabalhos anteriores em que a abordagem WAT foi referenciada.	NOVO
	Utilização em projetos de software anteriores	Informações históricas sobre o uso da abordagem WAT em projetos de software *Web* anteriores para avaliar a sua viabilidade.	V&B (2005) DN (2009)
	Benefícios	Indicação dos benefícios obtidos com o uso da abordagem WAT em projetos de software anteriores.	V&B (2005)
	Problemas	Indicação dos problemas encontrados com o uso da abordagem WAT em projetos de software anteriores.	V&B (2005)
	Avaliação Experimental	Indicação sobre a estratégia experimental (ex.: estudo de caso, estudo experimental, relato de uso) e as evidências adquiridas através da experimentação da abordagem WAT.	DN (2009)
Objeto	Categoria da	Informações sobre que tipo de aplicação (tais como *Web*	NOVO

	aplicação *Web*	Services, *Workflow*, colaboração, *e-commerce*) a abordagem pode ser usada.	
	Característica de Qualidade	Indicação sobre o conjunto de características de qualidade (por exemplo: usabilidade, desempenho, funcionalidade, segurança, confiabilidade, eficiência) a abordagem WAT é capaz de avaliar.	DN (2009)
	Unidade	Descreve a unidade da aplicação na qual a abordagem de teste atua, como páginas *Web*, componentes de páginas da *Web*, *links* e outros.	NOVO
	Perspectiva	Indicação de qual perspectiva de Projeto do Software *Web* a abordagem WAT pode ser aplicada: Conceituação (elementos conceituais que fazem parte do domínio da aplicação), Apresentação (aspecto visual e características de interface de usuário relacionados ao projeto), Navegação (estrutura da perspectiva do usuário mostrando como as informações estão disponíveis e são acessadas) e Estrutura (como a aplicação está estruturada em termos de classes, componentes).	NOVO
	Camada da aplicação	Informações sobre qual camada (cliente, servidor ou ambas) a abordagem WAT pode ser aplicada.	NOVO
	Nível de Teste	Os níveis de teste (unidade, integração, sistema, aceitação dos usuários, teste de regressão), exploradas pela abordagem WAT para testar o comportamento/estrutura do software *Web*.	DN (2009)
Técnica	Limitações	Informações sobre as restrições para utilizar a abordagem WAT (por exemplo: a abordagem é aplicada apenas para um projeto de software com características específicas, tais como o tamanho, a habilidade da equipe, ciclo de vida, escalabilidade).	DN (2009)
	Tipos de Falhas	Tipo de falhas que a abordagem WAT pode ajudar a revelar, tais como falha de dados, atrasos, erros de navegação, entre outros	V&B (2005) DN (2009)
	Métodos de desenvolvimento *Web*	Indicação sobre quais os métodos de desenvolvimento de software *Web* usam a mesma tecnologia adotada pela Abordagem WAT.	V&B (2005)
	Modelo utilizado para a geração de teste	Informações sobre o modelo (por exemplo, diagrama de casos de uso, diagrama de estados) utilizado pela abordagem WAT.	DN (2009)
	Tecnologia de Modelagem	Indicação sobre a tecnologia usada para a geração dos modelos utilizados para geração dos testes (ex.: UML, FSM, WSDL e outros).	DN (2009)
	Tipo de Técnica	Indicação sobre a técnica de testes (funcional, estrutural) que a abordagem WAT pode suportar.	DN (2009)
	Critérios de Cobertura	Informação sobre as regras utilizadas pela abordagem WAT para gerar casos de teste, tais como o fluxo de dados e de controle de fluxo.	V&B (2005) DN (2009)
	Tipo de Análise	Indicação sobre o tipo de análise de teste a abordagem WAT é capaz de suportar: comportamento estático (não é influenciado pela interação do usuário) ou dinâmica (ex.: páginas que podem ser construídas dinamicamente em resposta a entradas de usuários).	NOVO
	Descrição	Uma breve descrição sobre a abordagem WAT.	NOVO
	Entradas	Informações sobre os artefatos necessários / entrada de dados para iniciar o processo de geração de casos de teste com a abordagem WAT	V&B (2005) DN (2009)
	Saídas	Informações sobre as saídas (artefatos) produzidas pela abordagem WAT.	DN (2009)
	Tarefas executadas	Descrição sobre as tarefas demandadas pela abordagem WAT para gerar e / ou executar casos de teste.	DN (2009)
	Tarefas automatizadas	Descreve como a abordagem WAT é automatizada.	V&B (2005) DN (2009)
	Nível de Complexidade dos Passos Não-automatizados	Indicação sobre o esforço, tempo, custo e para executar as tarefas não-automatizadas na utilização da abordagem WAT.	DN (2009)

A Tabela 3.3 lista os atributos identificados como necessários para caracterização de abordagens de teste para aplicações Web, combinando os resultados da quasi-revisão sistemática e os trabalhos relacionados descritos anteriormente, indicados na coluna “Origem” se VEGAS e BASILI (2005), DIAS NETO (2009) ou Novo.

Os novos atributos visam à adaptação da estrutura para capturar as características específicas das abordagens de teste para aplicações Web. Tal proposta foi orientada por diferentes fontes de pesquisa:

- Revisão inicial da literatura sobre aplicações Web descrita no Capitulo 2;
- Diferentes características relacionadas às abordagens de teste para aplicações Web analisadas na quasi-revisão sistemática, descrita na Seção 2.7.

Serão detalhadas a seguir as razões que motivaram a inclusão dos novos atributos na estrutura proposta:

Categorias de aplicações Web: Frente às várias definições de aplicações Web encontradas, foi considerado importante categorizar tais aplicações para melhor compreender suas características, peculiaridades e similaridades. Além disso, devido ao fato de que algumas categorias de aplicação apresentam características específicas, foram identificadas na literatura técnica abordagens de teste igualmente específicas para as diferentes categorias deste tipo de aplicação.

Unidade: Foi observado ao longo da revisão da literatura e da quasi-revisão sistemática que não há um senso comum sobre a definição de unidade nas aplicações Web, que por sua vez tem conseqüência nas definições de teste de unidade, integração e sistema.

Perspectiva: Indicação da perspectiva de projeto de software Web na qual a abordagem de teste pode ser aplicada. Em (CONTE, 2009), é apontado que as perspectivas de projeto comumente utilizadas no desenvolvimento de aplicações Web são: Conceituação (elementos conceituais que fazem parte do domínio da aplicação), Apresentação (aspecto visual e projeto de interface com o usuário características relacionadas), Navegação (perspectiva do usuário, mostrando como a informação está disponível e acessada) e Estrutura (como a aplicação é estruturada em termos de classes, componentes).

Tipo de Análise: Boa parte das abordagens de teste analisadas distingue o tipo de análise realizada pela abordagem em Dinâmica a Estática. Um exemplo de aplicações Web dinâmicas são as aplicações AJAX. Trata-se de um conjunto de tecnologias utilizadas para desenvolver aplicativos ricos e interativos da Web. Um cliente típico AJAX é executado localmente no navegador Web do usuário e atualiza sua interface em tempo real, em resposta a entrada do usuário.

Além desses, foram incluídos os atributos “Descrição Breve” capaz de prover uma ideia geral da abordagem e “Citação em Artigos Anteriores” que informa a quantidade de vezes que a abordagem foi referenciada.

3.4 Survey de avaliação da estrutura

3.4.1 Objetivo

Foi executado um survey com pesquisadores da área com objetivo de identificar quais das informações presentes no conjunto de atributos podem ser pertinentes para caracterizar uma abordagem de teste para aplicações Web e sua respectiva relevância no contexto da seleção de abordagem WAT para um projeto de software Web. O estudo foi planejado e executado utilizando o padrão e a infra-estrutura adotada pelo grupo de Engenharia de Software Experimental da COPPE-UFRJ em trabalhos anteriores (SPÍNOLA et al., 2010) (DIAS NETO, 2009).

3.4.2 Questões de Pesquisa

As questões de pesquisa associadas ao survey são:

Q1: Os atributos extraídos da literatura técnica, apresentados na Tabela 3.3, são importantes para caracterizar uma abordagem WAT?

Q2: Existe algum atributo adicional considerado importante para caracterizar uma abordagem WAT que não está presente no conjunto inicial?

Q3: Qual a ordem de relevância, dos atributos considerados importantes, para apoiar a seleção de uma abordagem WAT a ser aplicada em um projeto de software Web?

A população do estudo foi representada pelo conjunto de autores dos artigos científicos descrevendo abordagens WAT, publicados na literatura técnica e identificadas através da quase revisão sistemática descrita no Capítulo 2.

Como instrumentação do estudo, foi adaptado um questionário on-line, anteriormente desenvolvido pelo grupo de Engenharia de Software Experimental da COPPE-UFRJ, que foi ajustado para atender as necessidades desse estudo. O preenchimento do questionário é realizado em três passos. Os participantes do estudo foram contatados por email, no qual receberam um login e senha para acessar o questionário, evitando que participantes não convidados participassem do estudo. A Figura 3.1 apresenta a tela inicial do survey.

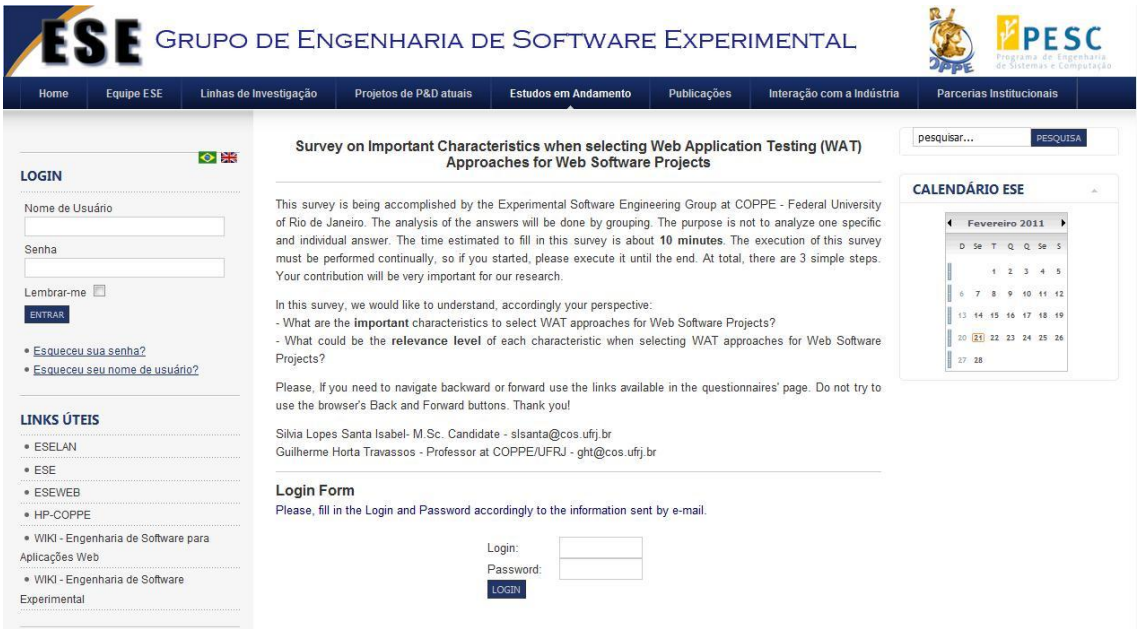


Figura 3.1. Tela inicial do Survey.

<u>1º Passo</u>) Caracterização do Participante.

Nesse passo os participantes são convidados a informar seus dados pessoais (nome, e-mail, afiliação e país), nível de formação acadêmica, número de artigos publicados sobre abordagens WAT e nível de experiência em teste de aplicações Web na indústria (Figura 3.2). Os dados pessoais do participante, exceto o email, são opcionais. Para todas as demais, o preenchimento é obrigatório para que o participante seja considerado na análise de resultados desse survey.

Figura 3.2. Tela de caracterização do participante.

Cada variável pode assumir os seguintes valores:

Nível de Formação:

= 0, se o participante possui apenas nível superior;

= 1, se o participante possui especialização em engenharia de software;

= 2, se o participante possui mestrado;

= 3, se o participante possui doutorado;

Número de Artigos: valor numérico de preenchimento livre.

Nível de Experiência:

= 0, se o nível de experiência é baixo;

= 1, se o nível de experiência é médio;

= 2, se o nível de experiência é alto;

= 3, se o nível de experiência é excelente;

Número de Projetos: valor numérico de preenchimento livre.

2º Passo) Identificação dos atributos importantes ou não importantes para caracterizar uma abordagem WAT.

Nesse momento são listados todos os 26 atributos que compõem a estrutura inicial para caracterizar as abordagens de teste para aplicações Web (Figura 3.3), apresentada na Tabela 3.3 deste trabalho.

Para cada atributo, o participante deve indicar se ele é ou não importante para caracterizar uma abordagem WAT, ou seja, se ele deve compor a estrutura de caracterização. Além disso, o participante pode inserir até 5 atributos adicionais que considere importantes e que não estão incluídos no conjunto inicial;

3º Passo) Definição do nível de relevância dos atributos para seleção de abordagens de teste para projetos de software Web.

Neste passo, os atributos indicados pelo participante como importantes para a seleção de uma abordagem WAT para um projeto de software Web (Passo 2) são avaliados quanto ao seu nível de relevância no procedimento de seleção. Seis níveis de relevância foram definidos:

(1) Muito Baixa Relevância: Indica que a característica não afeta significativamente a seleção de abordagens WAT para projetos de software Web. Essa característica deve ser considerada apenas em projetos de software Web muito específicos.

(2) Baixa Relevância: Indica que a seleção de uma abordagem WAT para um projeto de software Web será mais precisa caso considere esta característica. Em

alguns cenários específicos, poderá ser mais relevante, mas, em geral, a seleção não é afetada na ausência dessa característica.

(3) Média Relevância: Indica que a característica tem um efeito considerável na escolha de uma abordagem WAT para um projeto de software Web. Em geral, a seleção é afetada na ausência desta característica, mas ainda será possível usar a abordagem WAT selecionada.

(4) Alta Relevância: Indica que a característica deve ser considerada ao selecionar abordagens WAT para projetos de software Web. Apenas para um número restrito de cenários específicos, esta característica não deve ser considerada na seleção de abordagens WAT para projetos de software Web.

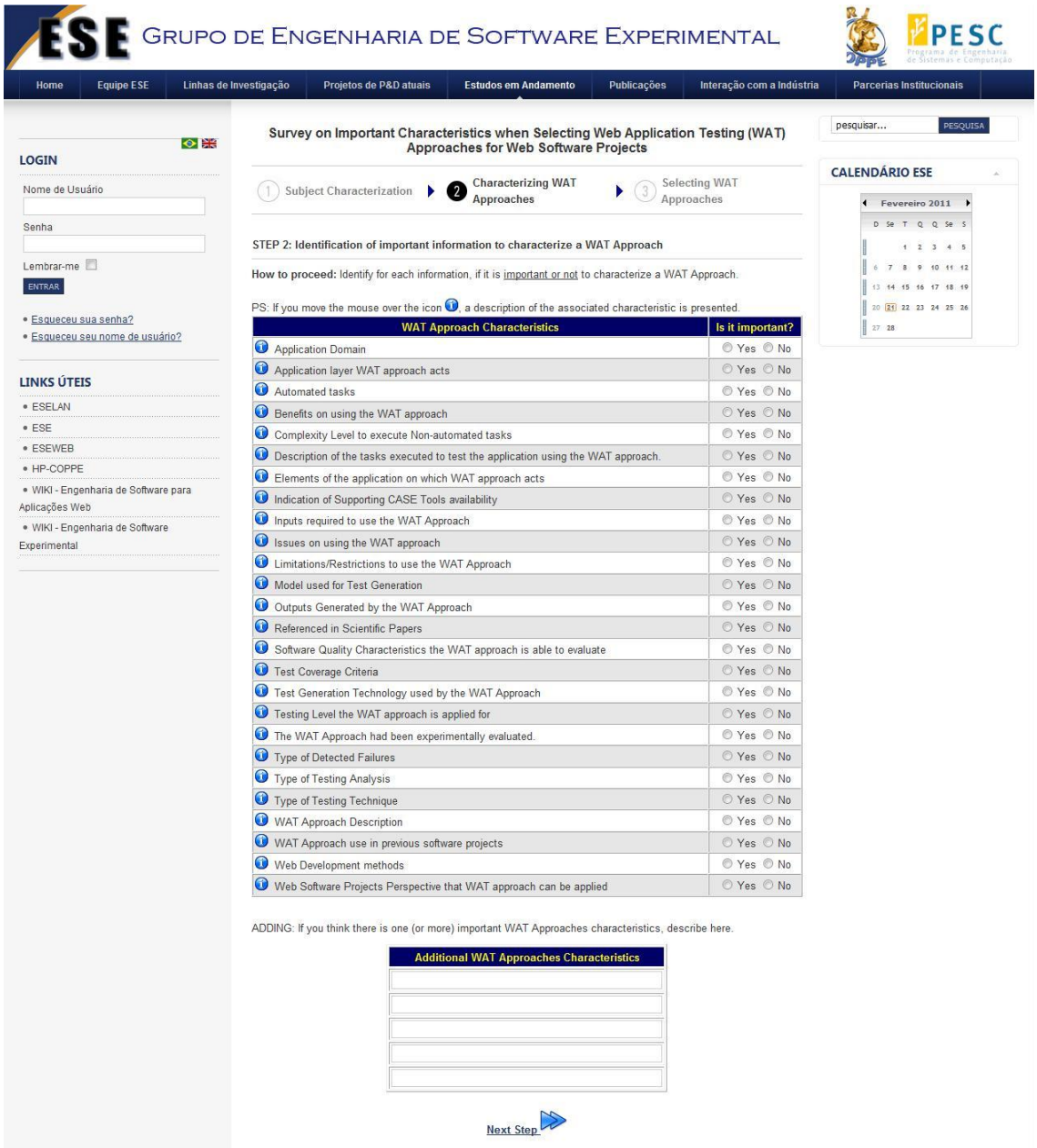


Figura 3.3. Tela de identificação de pertinência.

(5) Muita Alta Relevância: indica que a característica é absolutamente necessária para selecionar abordagens WAT para projetos de software Web. Sua ausência (não usá-la) indica que a abordagem escolhida WAT não deve ser capaz de ser utilizada.

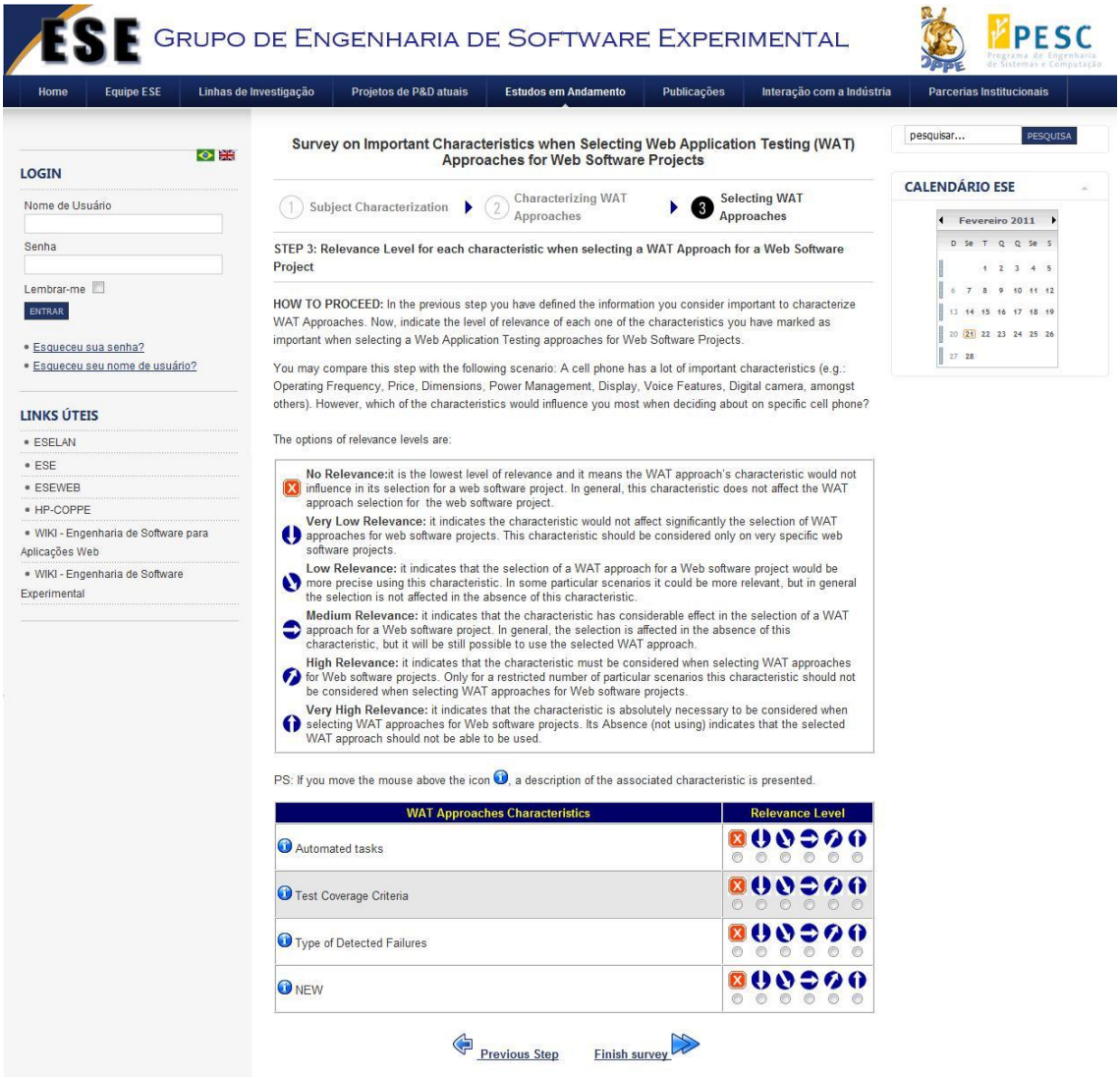


Figura 3.4. Tela de identificação de relevância.

Na Figura 3.4 é apresentado um exemplo da tela de identificação de relevância dos atributos para seleção de abordagens WAT para um determinado projeto de software web. Neste passo são listados apenas os atributos indicados pelo participante como importantes no passo anterior, que no exemplo foram: Passos Automatizados, Critérios de Cobertura, Tipos de Defeitos, e um novo atributo inserido pelo participante que não estava incluído no conjunto inicial. A Figura 3.5 mostra a tela de agradecimento apresentada ao participante ao término no survey.

Figura 3.5. Tela de agradecimento.

3.4.3 Execução

Ao longo da pesquisa o survey foi executado duas vezes:

Setembro de 2010. Na primeira execução, o questionário do estudo ficou ativo no período de 21/08/2010 a 17/09/2010 no endereço http://ese.cos.ufrj.br. Foram enviados convites de participação do survey por email para 169 pesquisadores, autores dos artigos científicos descrevendo abordagens WAT identificados na primeira execução da quasi-revisão sistemática descrita no Capítulo 2. Desse conjunto, 42 emails retornaram com mensagem de erro indicando que o endereço de e-mail estaria errado ou não existia. No total, o envio de e-mails para 127 pesquisadores não resultou em falha, o que nos leva a crer que essa seria a população disponível para o estudo.

Foram obtidas apenas 9 respostas, apesar dos lembretes enviados ao longo do período da pesquisa, o que resulta em um nível de confiança em torno de 68% (Figura 3.7) para esta população, considerando os dados coletados de acordo com a fórmula de Cálculo do Nível de Confiança de uma Amostra (HAMBURG,1980), apresentada na figura 3.6.

- N = Tamanho da População
- $1 - E_o$ = Nível de Confiança
- N = Tamanho da Amostra

$$n = \frac{N \cdot \frac{1}{E_o^2}}{N + \frac{1}{E_o^2}} \rightarrow E_o = \sqrt{\frac{N - n}{N \cdot n}}$$

Figura 3.6. Fórmula para cálculo do nível de confiança (Adaptado de HAMBURG, 1980).

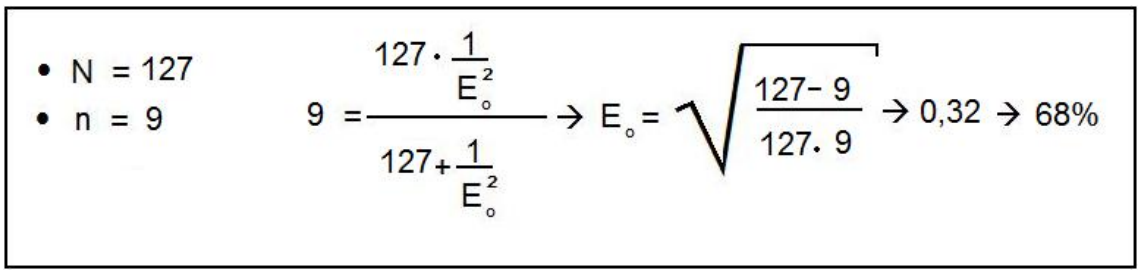


Figura 3.7. Nível de confiança da primeira execução do survey.

Devido ao um baixo número de respostas obtidas na primeira execução do survey, optou-se por reexecutá-lo com o objetivo melhorar o nível de confiança dos resultados obtidos.

Fevereiro de 2011. Nesta execução, o questionário do estudo ficou ativo no período de 05/01/2011 a 11/02/2011 no endereço http://ese.cos.ufrj.br. Foram enviados convites de participação do survey para 73 pesquisadores, autores dos novos artigos científicos descrevendo abordagens WAT identificados na segunda execução da quasi-revisão sistemática descrita no Capítulo 2. Desse conjunto, 10 emails retornaram com mensagem de erro indicando que o endereço de e-mail estaria errado ou não existia.

No total, o envio de e-mails para 63 pesquisadores não resultou em falha, sendo que 47 representavam novos autores e 16 autores já contatados na primeira execução do survey. Desta forma, a população disponível para o estudo passou a ser 174 (127 + 47).

Foram obtidas 11 novas respostas nesta execução, totalizando uma amostra de 20 respostas obtidas. Com esses novos dados, foi possível melhorar o intervalo de confiança de 68% para 79% (Figura 3.8).

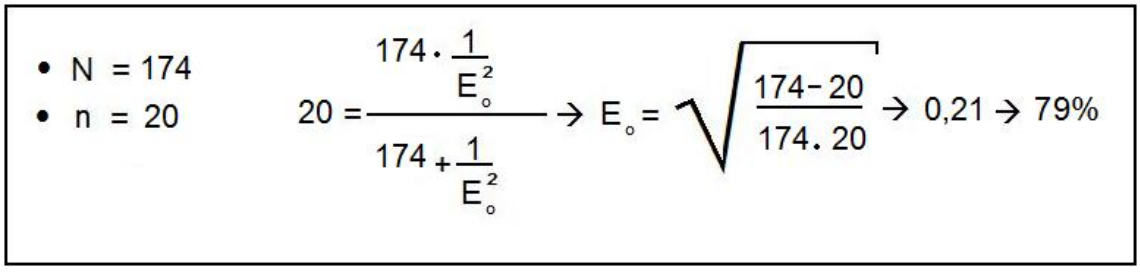


Figura 3.8. Nível de confiança após segunda execução do survey.

Os resultados das duas execuções do survey foram agregados para a realização do cálculo do nível de confiança. Isto foi possível porque as amostras das duas execuções foram selecionadas o obedecendo mesmo critério (autores dos

artigos selecionados nas duas execuções da quasi-revisão sistemática). Além disso, foram mantidas, nas duas execuções, a estratégia, a instrumentação utilizada e o objeto da análise. É importante destacar também o curto espaço de tempo entre as duas execuções, o que minimiza possíveis interferências nas opiniões capturadas pelo survey.

3.4.4 Análise dos Resultados

Para a análise dos dados, foi atribuído um peso para cada participante de acordo com a fórmula apresentada na Figura 3.9, que considera o nível de formação acadêmica, número de artigos publicados sobre abordagens WAT, nível de experiência em teste de aplicações Web, e o número de projetos de teste de aplicações Web que cada um participou na indústria. A fórmula usada para definir os pesos dos participantes foi baseada na proposta de DIAS NETO (2009). A Tabela 3.4 apresenta a caracterização dos participantes.

$$\text{Peso} = \text{Nível de Formação} + \frac{\text{Nº de Artigos Publicados}}{\text{Mediana do Nº Artigos Publicados por todos os Participantes}} + \text{Nível de Experiência} + \frac{\text{Nº de Projetos WAT}}{\text{Mediana do Nº de Projetos de todos os Participantes}}$$

Figura 3.9. Fórmula para caracterizar os participantes do survey.

Tabela 3.4. Caracterização dos participantes do survey.

País	Afiliação	Formação	Artigos Publicados	Nível de Experiência	Nº Projetos usando WAT	Peso
Brasil	Universidade de São Paulo	Mestrado	4	Alto	3	5,80
Itália	Politecnico di Milano	Doutorado	2	Alto	5	7,06
EUA	University of Nebraska–Lincoln	Doutorado	3	Alto	4	6,93
EUA	University of New Orleans	Doutorado	7	Alto	4	7,73
Grécia	University of Pireaus	Doutorado	6	Alto	4	7,53
Alemanha	University of Paderborn	Mestrado	10	Alto	2	6,66
Canadá	Universite de Montreal	Doutorado	12	Excelente	3	9,40
EUA	University of Maryland	Doutorado	15	Excelente	3	10,0
Itália	Politecnico di Torino	Doutorado	4	Alto	3	6,80
Canadá	Queen's university	Doutorado	4	Alto	2	6,46
EUA	University of Delaware	Doutorado	10	Alto	5	8,66
EUA	Fujitsu Labs of América	Doutorado	5	Excelente	7	9,33
Taiwan	Tatung University	Doutorado	3	Alto	1	5,93
Portugal	University of Coimbra	Mestrado	5	Alto	1	5,33
EUA	Georgia Tech	Mestrado	5	Excelente	7	8,33

Itália	University of Sannio	Doutorado	10	Alto	2	7,66
Brasil	Universidade Federal do Paraná	Doutorado	8	Alto	2	7,26
Itália	University of Naples	Especialização	4	Alto	1	6,13
EUA	Fujitsu Labs of América	Doutorado	1	Alto	2	3,86
Itália	Istituto di Scienza e Tecnologie dell'Informazione	Doutorado	6	Alto	3	7,20

Após o cálculo dos pesos dos participantes, passou-se à análise do **Índice de Pertinência** que indica a importância dos atributos avaliados na caracterização das abordagens WAT. Se o atributo foi considerando importante pelo participante do survey, ele tem valor 1, caso contrário ele tem valor 0 (zero).

O índice de pertinência de cada atributo foi obtido pela soma das respostas de cada um dos participantes (1 ou 0), multiplicadas pelos seus respectivos pesos. Se todos os participantes considerarem um atributo importante (ou seja, atribuírem valor =1), o índice de importância alcançará o valor máximo, que no caso, será igual a soma dos pesos dos participantes (144.13).

A definição de que um atributo é importante ou não para caracterizar uma abordagem WAT deve ser limitada por um ponto de corte. Nesse sentido, foram analisadas 3 medidas estatísticas: Média, Mediana e Quartil. A média é uma medida de tendência central a partir da qual é possível verificar uma tendência dos dados em torno dos valores centrais. A média dos índices de pertinência é 118,56.

A mediana de um conjunto de valores, dispostos segundo uma ordem (crescente ou decrescente), é o valor situado de tal forma no conjunto que o separa em dois subconjuntos de mesmo número de elementos. A mediana depende da posição e não dos valores dos elementos na série ordenada. Essa é uma das diferenças marcantes entre mediana e média (que se deixa influenciar, e muito, pelos valores extremos). A mediana dos índices de pertinência é 124,30.

Além das medidas de posição (média e mediana), há outras que, se consideradas individualmente, não são medidas de tendência central, mas estão ligadas à mediana relativa à sua característica de separar a série em duas partes que apresentam o mesmo número de valores.

Essas medidas são, juntamente com a mediana, conhecidas pelo nome genérico de separatrizes. Nesse conjunto se incluem os quartis. Denomina-se quartis os valores de uma série que a dividem em quatro partes iguais. É necessário, portanto, 3 quartis (Q1 , Q2 e Q3) para dividir uma série em quatro partes iguais. O quartil 2 (Q2) sempre será igual à mediana da série. O primeiro quartil dos índices de pertinência é 109,41.

Os índices de pertinência de cada atributo estão descritos na Tabela 3.5. Para facilitar a visualização das medidas estatísticas analisadas, foram hachurados em vermelho os atributos cujos índices ficaram localizados no primeiro quartil; em amarelo, aqueles cujo valor da pertinência é menor do que a média dos índices; e em verde os atributos com índices localizados abaixo da mediana.

Tabela 3.5. Índice de Pertinência dos atributos.

Atributo	Índice de Pertinência	% de Pertinência	
Unidade	138,33	95,98%	
Tipos de Defeito	138,20	95,88%	
Modelo utilizado para a geração de teste	137,33	95,28%	
Tarefas automatizadas	137,06	95,10%	
Entradas	136,40	94,63%	
Critérios de Cobertura	136,40	94,63%	
Limitações	135,80	94,22%	
Tipo de Técnica	132,40	91,86%	
Nível de Complexidade dos Passos Não-automatizados	131,06	90,93%	
Tecnologia de Geração dos Testes	129,60	89,92%	
Benefícios	129,33	89,73%	
Característica de Qualidade	128,06	88,85%	
Nível de Teste	126,13	87,51%	
Categoria da aplicação Web	122,46	84,97%	Mediana
Saídas	121,26	84,14%	
Problemas	120,00	83,26%	
Avaliação Experimental	119,00	82,56%	
Camada da aplicação	117,53	81,54%	Média
Tipo de Análise	116,26	80,67%	
Descrição	107,13	74,33%	1º Quartil
Tarefas executadas	99,86	69,29%	
Perspectiva	90,26	62,63%	
Métodos de desenvolvimento Web	87,8	60,92%	
Utilização em projetos de software anteriores	84,93	58,93%	
Disponibilidade de ferramentas de apoio	80,86	56,11%	
Citação em artigos científicos anteriores	79,13	54,90%	

O patamar escolhido para indicar se o atributo deve ser excluído ou não da estrutura de caracterização foi o primeiro quartil dos índices de pertinência. Na Figura 3.10, é possível observar que nesse intervalo há uma queda mais acentuada dos índices de pertinência dos atributos. Desta forma, os atributos cujos índices ficaram localizados no primeiro quartil, abaixo de 109,41, foram descartados.

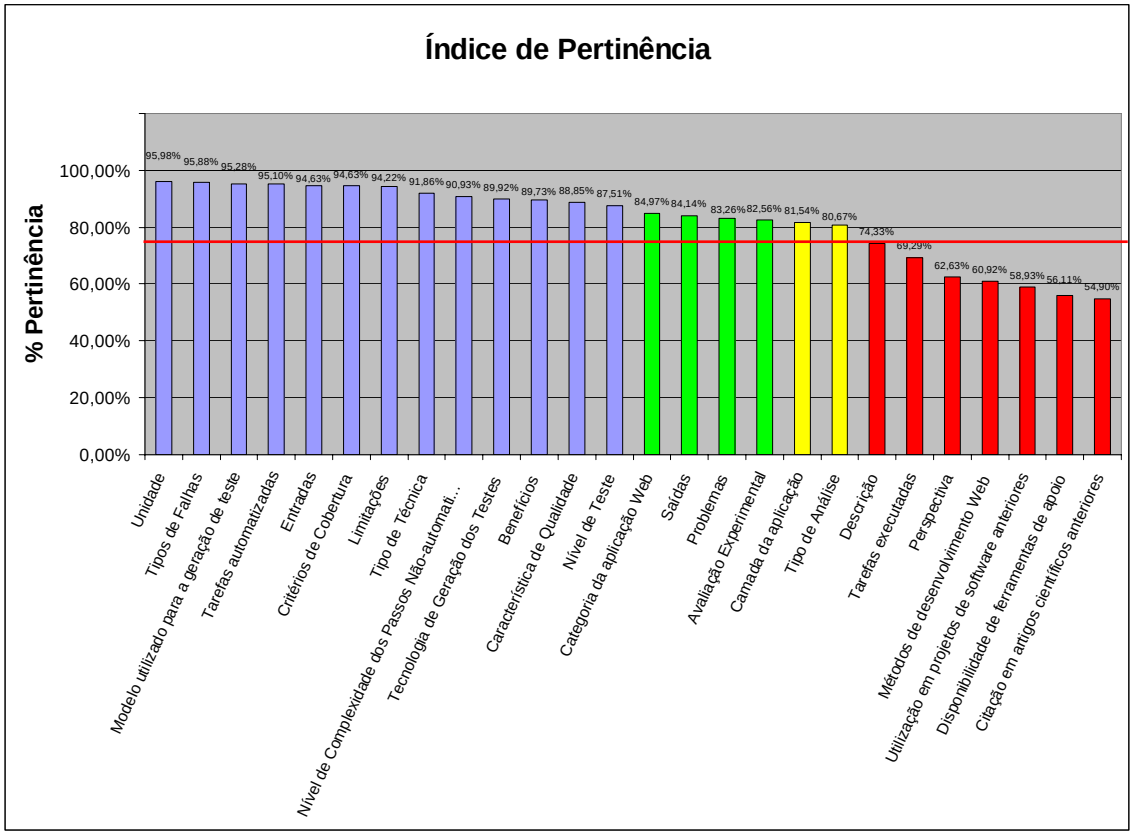


Figura 3.10. Representação Gráfica do Índice de Pertinência dos Atributos.

A escolha foi influenciada também pelo nível de confiança do survey (79%) que, caso fosse escolhido como patamar de corte frente aos percentuais de pertinência apresentados pelos atributos, iria coincidir com o conjunto selecionado para compor a estrutura de caracterização a partir critério do primeiro quartil, cujo último atributo apresenta 80,67% (percentual do índice de pertinência do atributo 'Tipo de Análise').

Após a identificação dos índices de pertinência, foi calculado o **Nível de Relevância** dos atributos para caracterizar as abordagens WAT, que indica o quanto cada atributo é importante para apoiar o processo de seleção de abordagens de teste para projetos de software Web. Cada participante indicou um valor para cada atributo definido anteriormente como importante. Os valores podiam variar de 0 a 5, conforme detalhado no item 3.2. Aos atributos que não foram indicados como importantes, foi atribuído o valor -5 como nível de relevância.

Nesse sentido, o nível de relevância de cada atributo foi obtido pela soma das respostas de cada um dos participantes (0 a 5, ou -5 para os atributos indicados como não importantes), multiplicadas pelos seus respectivos pesos. Se todos os participantes atribuírem o valor máximo de relevância para um atributo (ou seja, atribuírem valor =5), o seu nível alcançará o valor máximo que, no caso, será igual a soma dos pesos dos participantes (144.13) multiplicado por 5 (=720,65).

A informação do nível de relevância irá auxiliar na identificação dos atributos que deverão apresentar um peso diferenciado no procedimento de seleção das abordagens de teste para projetos de software Web. Por esta razão, os níveis de relevância de cada atributo também foram analisados utilizando medidas estatísticas: Média, Mediana e Quartil.

Os níveis de relevância de cada atributo estão descritos na Tabela 3.6. Para facilitar a visualização das medidas estatísticas analisadas, foram hachurados em vermelho os atributos cujos índices ficaram localizados no primeiro quartil; em amarelo, aqueles cujo valor da pertinência é menor do que a média dos índices; e em verde os atributos com índices localizados abaixo da mediana.

Tabela 3.6. Nível de relevância dos atributos.

Atributo	Nível de Relevância	% de Relevância	Ordem para Seleção
Tipos de Falhas	528,40	73,32%	1º Grupo
Tarefas automatizadas	502,66	69,75%	1º Grupo
Critérios de Cobertura	498,33	69,15%	1º Grupo
Entradas	471,53	65,43%	1º Grupo
Modelo utilizado para a geração de teste	470,20	65,25%	1º Grupo
Limitações	463,46	64,31%	1º Grupo
Tipo de Técnica	444,33	61,66%	1º Grupo
Unidade	443,20	61,50%	1º Grupo
Nível de Complexidade dos Passos Não-automatizados	415,33	57,63%	1º Grupo
Tecnologia de Geração dos Testes	413,86	57,43%	1º Grupo
Característica de Qualidade	405,93	56,33%	1º Grupo
Benefícios	403,00	55,92%	1º Grupo
Saídas	396,73	55,05%	1º Grupo
Nível de Teste	363,33	50,42%	2º Grupo
Avaliação Experimental	336,40	46,68%	2º Grupo
Categoria da aplicação Web	325,60	45,18%	2º Grupo
Tipo de Análise	322,93	44,81%	2º Grupo
Problemas	318,06	44,14%	2º Grupo
Camada da aplicação	283,80	39,38%	2º Grupo
Descrição	164,53	22,83%	-
Tarefas executadas	99,86	13,86%	-
Perspectiva	3,40	0,47%	-
Métodos de desenvolvimento Web	-18,13	-2,52%	-
Disponibilidade de ferramentas de apoio	-29,60	-4,11%	-
Citação em artigos científicos anteriores	-45,53	-6,32%	-
Utilização em projetos de software anteriores	-104,26	-14,47%	-

Mediana

Média

1º Quartil

Apesar do conjunto de atributos indicados ser o mesmo, considerando o quartil como patamar de corte, é possível observar que a ordem de pertinência dos atributos não necessariamente coincide com a ordem de relevância para seleção de uma

abordagem WAT para projetos de Software Web. A análise das medidas estatísticas sugere que apesar de importantes, alguns atributos podem ser mais influentes que outros no procedimento de seleção.

Nesse sentido, o conjunto de atributos pode ser dividido em dois grupos para verificar a adequação das características das abordagens WAT frente a um projeto de software Web. O primeiro grupo, composto pelos atributos não hachurados na Tabela 3.6 (atributos acima da mediana), pode ser utilizado na verificação de adequação, e o segundo grupo (atributos entre a mediana e o primeiro quartil) pode ser utilizado como critério de desempate. O Capítulo 4 irá apresentar, mais detalhadamente, como a análise das medidas estatísticas irá influenciar o procedimento de seleção proposto.

Adicionalmente, cada participante foi questionado sobre a existência de algum atributo considerado importante para caracterizar uma abordagem WAT que não estava presente no conjunto apresentado. Apenas dois participantes sugeriram novos atributos. Um deles indicou o atributo “Disponibilidade de Ferramentas WAT de código aberto” e associou a tal atributo o nível de relevância 4.

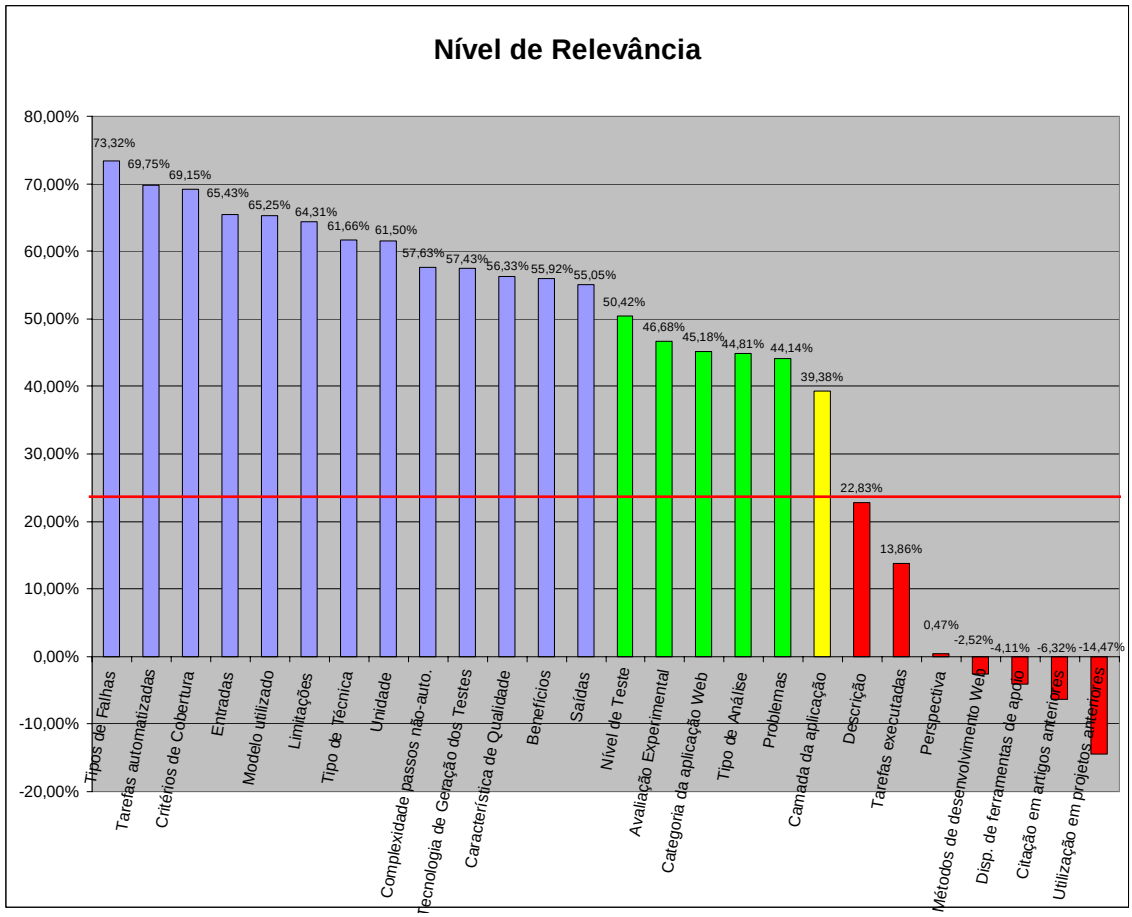


Figura 3.11. Representação Gráfica do Nível de Relevância dos Atributos.

Entretanto, ao analisar as respostas de todos os participantes do survey verifica-se que o atributo “Disponibilidade de ferramentas de apoio” alcançou os menores índices de pertinência e nível de relevância, abaixo inclusive do patamar de corte definido para compor a estrutura de caracterização de abordagens WAT, razão pela qual não será incorporado na estrutura proposta. Da mesma forma, o novo atributo sugerido “Disponibilidade de Ferramentas WAT de código aberto”, também foi descartado.

Outro participante indicou o atributo “Comportamento Não Determinístico” para o qual ele atribuiu o nível máximo de relevância (=5). Comportamento não determinístico pode ser definido como um comportamento regido por fenômenos aleatórios. Esse atributo apresenta certa familiaridade com o atributo Tipo de Análise, no sentido em que, conhecendo-se o estado do sistema, não é possível prever o seu comportamento final, devido à interferência do usuário no fluxo de execução e as características dinâmicas da aplicação. O atributo Tipo de Análise, incluído no corpo de conhecimento, também foi identificado como pertinente pelo mesmo participante. Por esta razão, o novo atributo sugerido foi descartado.

A estrutura final para caracterizar abordagens WAT está descrita na Tabela 3.7. Em complemento, foi associado um conjunto de valores pré-definidos a alguns atributos com o objetivo de classificar os dados, que poderá auxiliar na identificação das abordagens adequadas a determinados projetos de software Web.

A obtenção desses valores foi orientada por diferentes fontes de pesquisa,tais como, revisão inicial da literatura sobre aplicações Web descrita no Capítulo 2; diferentes características relacionadas às abordagens de teste para aplicações Web analisadas na quasi-revisão sistemática (descrita na Seção 2.7), e o conjunto de valores associados aos atributos originalmente proposto em DIAS NETO (2009) e VEGAS e BASILI (2005).

Para o atributo “Avaliação Experimental” deverá ser considerada a perspectiva do autor sobre a avaliação realizada na abordagem proposta, e poderá assumir os seguintes valores: prova de conceito, aplicação na indústria, estudo de caso, estudo experimental. No caso das “Categorias da Aplicação” foram utilizados os valores definidos por Ginige e Murugesan [8] e Kappel [10], além dos termos *Rich Internet Application* (RIAS), Serviços Web, *e-commerce*, *e-government*, *e-learning*, utilizados na revisão realizada.

As “características de qualidade” serão classificadas de acordo com as características descritas na norma ISO/IEC 9126 (2002), destacando apenas a segurança, que é uma sub-característica de funcionalidade, devido a sua importância para aplicações Web. O atributo “Nível de Teste” deverá considerar as seguintes

etapas de teste: Unidade, Integração, Sistema e Aceitação. Apesar de não ser um nível de teste, o Teste de Regressão também foi considerado para fins dessa classificação. O atributo "Camada de Aplicação" poderá assumir os valores "*Front-End*" ou "*Back-End*".

O atributo "Tipo de Técnica" poderá assumir os valores Funcional ou Estrutural, enquanto o "Tipo de Análise" poderá ser Dinâmica ou Estática, caso a análise seja influenciada ou não pela interação do usuário. O atributo "Nível de Complexidade dos Passos Não-automatizados" poderá ser classificado como Baixo, Médio ou Alto de acordo com esforço, custo, tempo e habilidades requeridas para realizar os passos da abordagem. Os demais atributos são descritos a partir de um texto livre.

Tabela 3.7. Estrutura final para caracterizar abordagens WAT.

Categoria	Atributos	Valores
Histórico	Avaliação Experimental	[prova de conceito, aplicação na indústria, estudo de caso, estudo experimental]
	Benefícios	[Texto livre]
	Problemas	[Texto livre]
Objeto	Categoria da aplicação Web	[Aplicações *Browser-Based*, *Rich Internet Application* (RIAS), Serviços Web, *Workflow*, Colaboração, baseada no Conhecimento, Ubíquas, *e-commerce, e-government, e-learning*]
	Característica de Qualidade	[Eficiência, Funcionalidade, Confiabilidade, Segurança, Usabilidade, Portabilidade]
	Unidade	[Texto livre]
	Nível de Teste	[Unidade, Integração, Sistema, Aceitação, Regressão]
	Camada da aplicação	["Front-End", "Back-End"]
Técnica	Tipo de Técnica	[Funcional, Estrutural]
	Tipo de Análise	[Estático, Dinâmico]
	Tipo de Falhas	[Texto livre]
	Critérios de Cobertura	[Texto livre]
	Entradas	[Texto livre]
	Saídas	[Texto livre]
	Limitações	[Texto livre]
	Modelo utilizado para a geração de teste	[Texto livre]
	Tecnologia de Geração dos Testes	[Texto livre]
	Tarefas automatizadas	[Sim, Não]
	Nível de Complexidade dos Passos Não-automatizados	[Baixo, Médio, Alto]

A Tabela 3.8 mostra um exemplo de caracterização utilizando a estrutura proposta. Foi utilizada a abordagem de teste para aplicações Web definida por REZA et al.(2008).

Tabela 3.8. Exemplo: "A model based testing technique to test Web applications using StateCharts" (REZA et al 2008).

Categoria	Atributos	Valores
Histórico	Avaliação Experimental	Estudo de Caso
	Benefícios	Não relatado.
	Problemas	Não relatado.
Objeto	Categoria da aplicação Web	Aplicação *Browser-Based*
	Característica de Qualidade	Funcionalidade, Confiabilidade
	Unidade	*Links*, Formulários e Imagens
	Nível de Teste	Teste de Unidade, Teste de Integração
	Camada da aplicação	"*Front-End*"
Técnica	Tipo de Técnica	Estrutural
	Tipo de Análise	Dinâmico
	Tipo de Falhas	Lógica Incorreta no Código, Falhas nos *links* (Páginas Inacessíveis, *links* quebrados), Incorreta transições de estado.
	Critérios de Cobertura	Todas as imagens, todas as transações, todos os pares, todas as condições e todos os caminhos.
	Entradas	Diagrama de Estado
	Saídas	Casos de Teste
	Limitações	Não é considerado problema de compatibilidade dos navegadores Web. Não encontraram solução para problemas relacionados à modelagem de concorrência e "*back-ends*" das aplicações Web.
	Modelo utilizado para a geração de teste	Diagrama de Estado
	Tecnologia de Geração dos Testes	UML
	Tarefas automatizadas	Sim
	Nível de Complexidade dos Passos Não-automatizados	Baixa, apenas a modelagem inicial é requerida

3.5 Considerações Finais do Capítulo

Este capítulo apresentou uma estrutura de caracterização de abordagens de teste de aplicações Web, a partir dos resultados obtidos na quasi-revisão sistemática descritos anteriormente e de trabalhos relacionados identificados na literatura. Além disso, foi apresenta a avaliação da estrutura proposta através de um survey realizado com especialistas.

A motivação para organizar esta estrutura de caracterização se apóia na premissa de que a existência de tal estrutura pode prover informações para apoiar a tomada de decisão a respeito da seleção de abordagens de teste para determinados projetos de software.

Nesse sentido, é importante manter a atualizar o corpo de conhecimento e os índices de relevância dos seus atributos, a partir da realização de novas revisões sistemáticas. Além disso, uma importante contribuição a ser realizada em trabalhos futuros seria submeter à avaliação da estrutura do corpo de conhecimento aos profissionais da indústria também, visto que o survey foi submetido a membros da comunidade acadêmica.

O próximo capítulo apresenta o procedimento Porantim-WAT, uma instância especializada de Porantim (DIAS NETO, 2009) cujo objetivo é indicar a abordagem de teste mais apropriada para um projeto, a partir do cálculo do grau de adequação entre os atributos das abordagens existentes no corpo de conhecimento e as características do projeto onde elas seriam aplicadas.

CAPÍTULO 4 - PORANTIM-WAT - PROCEDIMENTO PARA APOIAR A SELEÇÃO DE ABORDAGENS DE TESTE PARA PROJETOS DE SOFTWARE WEB

Neste capítulo é apresentado o procedimento Porantim-WAT, que provê apoio à seleção de abordagens de teste para projetos de software Web (seção 4.3). Trata-se de uma instância especializada de Porantim (seção 4.2) e, da mesma forma, é fundamentada nos mesmos elementos: um corpo de conhecimento sobre abordagens WAT e um mecanismo que avalia o grau de adequação entre as abordagens de teste e as características de um projeto de software Web. Ao final, é apresentada a adaptação de uma infra-estrutura computacional para apoiar o procedimento de seleção proposto (seção 4.4).

4.1 Introdução

A partir dos resultados obtidos na quasi-revisão sistemática, foi apresentada a estrutura de caracterização de abordagens de teste para aplicações Web. A motivação para organizar esta estrutura de caracterização se apóia na premissa de que a existência de tal estrutura pode auxiliar a seleção da abordagem de teste a ser aplicada em um projeto de software Web.

Em (DIAS NETO, 2009), foi apresentada uma solução para o problema da seleção de técnicas de teste baseado em modelos (TTBM) com base no esquema de caracterização de VEGAS e BASILI (2005), a partir do desenvolvimento do corpo de conhecimento sobre TTBMs e de um procedimento de seleção automatizado denominado Porantim, que calcula o grau de adequação entre atributos das TTBMs e as características do projeto de software onde elas seriam aplicadas.

Neste capítulo, será apresentada a criação de uma instância especializada do procedimento Porantim para apoiar a seleção de abordagens de teste para projetos de software Web.

4.2 Porantim: Procedimento de apoio para seleção de tecnologias

4.2.1 Visão Geral

O procedimento Porantim, proposto por DIAS NETO (2009), objetiva prover informações para apoiar a tomada de decisão a respeito da seleção de técnicas de teste baseado em modelos para determinados projetos de software. Trata-se de uma evolução do mecanismo de apoio à seleção de técnicas de teste chamada de Esquema de Caracterização, proposta por VEGAS e BASILI (2005), cuja caracterização dos atributos para a técnica de teste foram atualizados com as características específicas do MBT para elaboração de um corpo de conhecimento específico.

O procedimento é baseado em dois elementos: a) Corpo de Conhecimento sobre técnicas TBM, que funciona como um repositório de técnicas que podem ser utilizados em um projeto de software; e b) Mecanismo de Seleção que indica quais as técnicas teste mais adequadas, além de analisar o impacto do seu uso sobre um projeto de software.

O Mecanismo de Seleção é composto por 5 atividades:

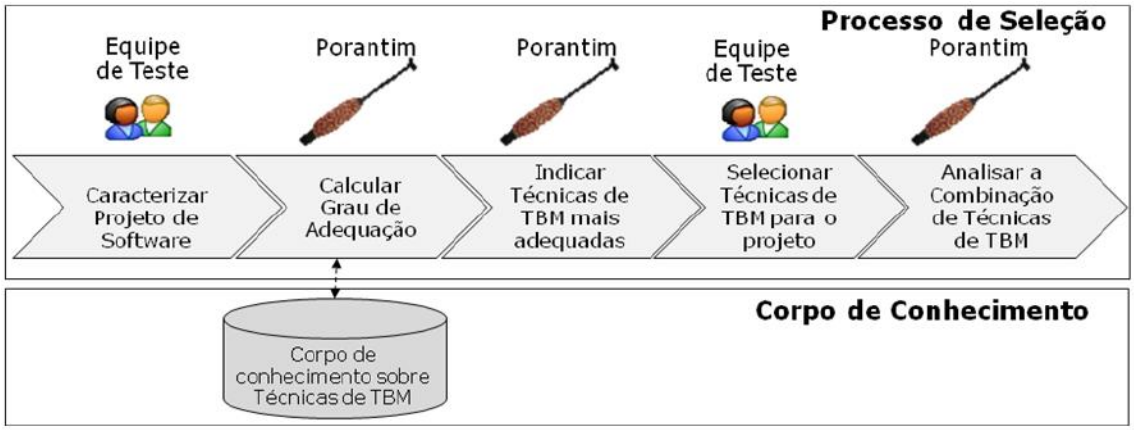


Figura 4.1. Visão Geral do Procedimento Porantim (DIAS NETO 2009).

- **Caracterizar Projeto de Software**: a equipe de teste precisa preencher um questionário sobre as características de projetos de software onde as técnicas de teste serão aplicadas. No total, 16 atributos em relação ao projeto de software devem ser preenchidos, tais como: plataforma de execução, a linguagem de programação, modelos fornecidos pelo processo de desenvolvimento, nível de teste exigido, as características de qualidade do software a ser avaliado, entre outros.

- **<u>Calcular o Grau de Adequação</u>**: internamente, Porantim calcula o grau de adequação entre o projeto de software caracterizado na etapa anterior para cada técnica TBM incluída no repositório, utilizando o conceito matemático de distância euclidiana. A noção de distância conceitual é realizada, matematicamente, pela norma da diferença entre dois vetores v1 e v2, conforme apresentado na Figura 4.2.

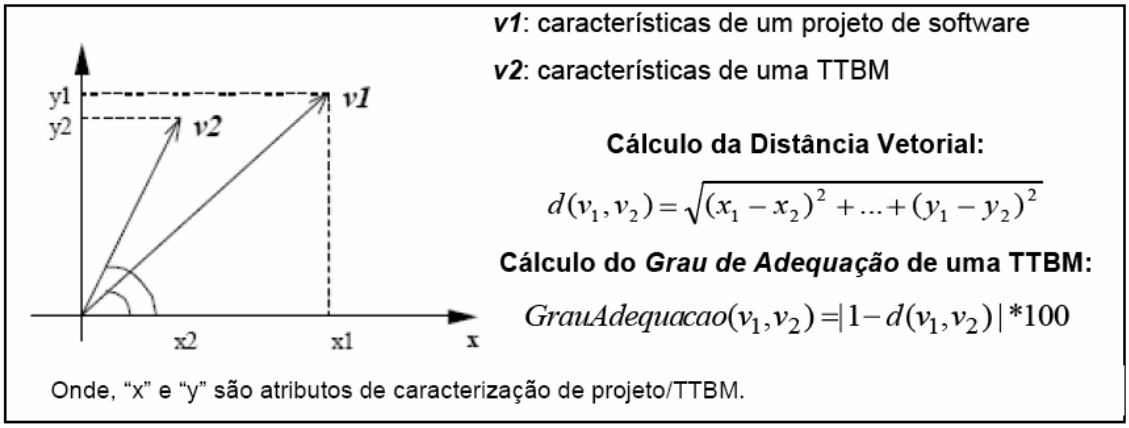


Figura 4.2. Fórmula para cálculo de distância entre vetores (DIAS NETO 2009).

Para o cálculo do Grau de Adequação, as características do projeto e de cada TTBM são transformadas em valores que posteriormente são representados através de vetores (v1 – vetor das características do projeto de software) e (v2 – vetor das características de cada TTBM). A distância entre eles indica o grau de adequação de uma TTBM para o projeto de software. Quanto mais próximos eles estiverem, mais adequada ao projeto é a TTBM.

Cada atributo do projeto de software que pode ser confrontado com os atributos de caracterização de TTBMs possui um peso[2] específico. A lista dos pesos de cada atributo pode ser encontrada em DIAS NETO (2009). A regra de transformação utilizada para representar o vetor v1 (vetor das características do projeto de software) foi utilizar o valor do peso de cada atributo. A regra de transformação utilizada para representar o vetor v2 (vetor das características de cada TTBM) está descrita a seguir:

SE [atributo do projeto de software = atributo da TTBM]
ENTÃO atributo da TTBM é transformado em 1 x influência do atributo;
CASO CONTRÁRIO recebe o valor 0.

[2] Para cada atributo de caracterização das técnicas de teste baseadas em modelos foi atribuído um peso obtido através dos resultados de um survey. Esses pesos simbolizam o nível de relevância de cada atributo na seleção das técnicas para um determinado projeto.

Uma vez definidas as regras de transformação, o passo seguinte consiste na formalização da representação vetorial para projeto de software e TTBM e cálculo da distância entre os vetores.

- **Indicar Técnicas mais adequadas**: depois de calcular o grau de adequação, Porantim lista, em ordem decrescente, o subconjunto de técnicas de teste que mais se adequam ao projeto, indicando o grau de adequação de cada uma delas. É possível consultar os detalhes das características das técnicas listadas.

- **Selecionar Técnicas**: a equipe de teste precisa selecionar o subconjunto das técnicas, sugeridas na etapa anterior, que deseja utilizar no projeto de software.

- **Analisar combinação das Técnicas**: após a equipe selecionar o subconjunto de técnicas que deseja utilizar no projeto de software, Porantim analisa o impacto da combinação das técnicas selecionadas sobre algumas variáveis do processo de teste, tais como a cobertura de projetos de software, modelagem, esforço e recursos humanos necessários para utilizar as técnicas.

Apresentada a visão geral do procedimento Porantim, serão descritas a seguir as adaptações necessárias que deverão ser realizadas no mecanismo de seleção das abordagens de teste voltado para aplicações Web. A construção de um repositório com atributos de caracterização específicos para o domínio de teste de aplicações Web, descrito no capítulo anterior, representa o primeiro passo na direção da criação de uma instância especializada de Porantim para apoiar a seleção de técnicas de teste para projetos de software Web.

4.3 Porantim-WAT: Adaptação de Procedimento para apoiar a seleção de abordagens de teste para projetos Web

Porantim-WAT representa uma instância especializada do procedimento de apoio à seleção de técnicas de teste baseado em modelos, proposto por DIAS NETO (2009). Sendo uma extensão de Porantim, este procedimento é fundamentado nos mesmos elementos principais: Corpo de Conhecimento e Mecanismo de Seleção de Técnicas de Teste, composto por 5 atividades.

Entretanto, para a criação de uma instância especializada do procedimento Porantim para apoiar a seleção de abordagens WAT para projetos de software Web,

além da criação de um repositório com atributos de caracterização específicos para o domínio de teste de aplicações Web, será necessário ajustar todas as atividades do mecanismo de seleção das abordagens.

Porantim-WAT também irá apresentar 5 atividades no mecanismo de seleção. Entretanto apresentação um caminho alternativo adicional que possibilitará o refinamento da seleção de abordagens de teste para aplicações Web. Serão apresentados a seguir os requisitos de adaptação requeridos em cada atividade do mecanismo de seleção, nas quais serão necessárias implementações de ajustes.

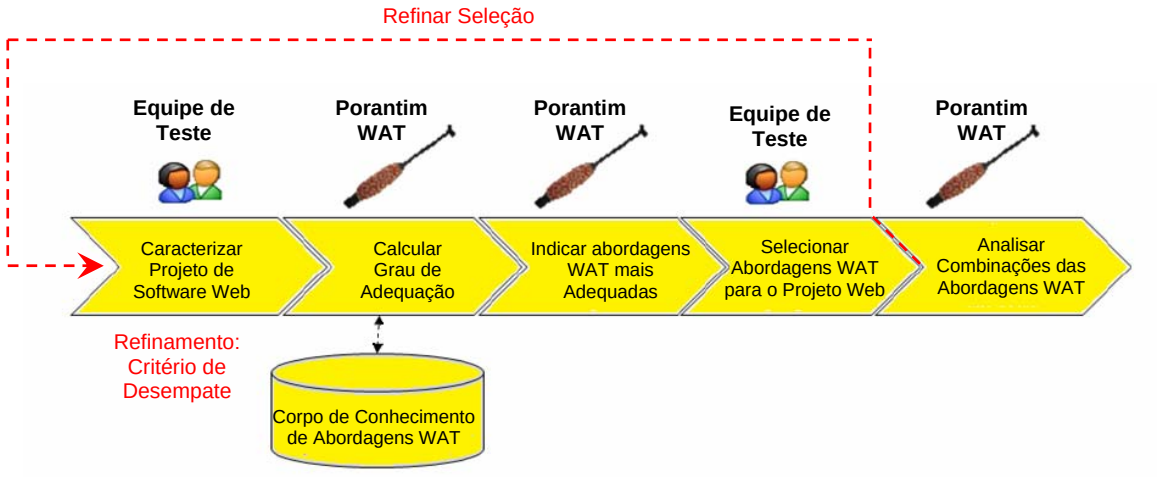


Figura 4.3. Visão Geral Procedimento Porantim-WAT.

- **Caracterizar Projeto de Software Web**: a equipe de teste irá preencher um questionário sobre as características de projetos de software Web onde as abordagens de teste serão aplicadas. Inicialmente, 9 atributos em relação ao projeto de software Web devem ser preenchidos, tais como: Modelos fornecidos pelo processo de desenvolvimento, Tipo de falhas, as características de qualidade de software a ser avaliado, entre outros.

  Ao final do procedimento de seleção, caso haja empate no grau de adequação entre as abordagens selecionadas por Porantim-WAT, ou ainda, caso seja desejável refinar a seleção das abordagens de teste por qualquer motivo, a etapa “Caracterizar Projeto de Software Web” poderá ser novamente submetida à equipe de teste. Nesse momento, um segundo grupo de informações sobre a caracterização do projeto, composto por 5 novos atributos, será informado pela equipe de teste e será utilizado como critério de desempate. Desta forma, espera-se obter um resultado mais refinado da busca, visto que 14 atributos serão considerados pelo mecanismo de seleção.

- **Calcular o Grau de Adequação**: analogamente a Porantim, o procedimento Porantim-WAT calcula o grau de adequação entre o projeto de software Web, caracterizado na etapa anterior, para cada abordagem WAT incluída no corpo de conhecimento, utilizando o conceito matemático de distância euclidiana.

  Para o cálculo do Grau de Adequação, as características do projeto de software Web e de cada abordagem WAT são transformadas em valores que posteriormente são representados através de vetores. A distância entre eles indica o grau de adequação de uma abordagem WAT para o projeto de software Web. Na Seção 4.3.6 será apresentado um exemplo da transformação em vetores e do cálculo do grau de adequação.

- **Indicar Abordagens WAT mais Adequadas**: depois de calcular o grau de adequação, Porantim-WAT lista, em ordem decrescente, o subconjunto das abordagens WAT que mais se adequaram ao projeto, indicando o grau de adequação de cada uma delas. É possível consultar os detalhes das características das abordagens listadas.

- **Selecionar Abordagens WAT para um Projeto Web**: nesse momento serão apresentadas duas opções à equipe de teste. A primeira consiste na seleção de um subconjunto das abordagens WAT, sugeridas na etapa anterior, que serão utilizadas no projeto de software Web caracterizado. Na segunda, a equipe de teste poderá optar pelo refinamento da busca. A equipe então deverá escolher quais abordagens serão submetidas ao refinamento do procedimento de seleção. Nesse caso, Porantim-WAT irá retornar à etapa “Caracterizar Projeto de Software Web” onde novos atributos do projeto deverão ser informados e serão utilizados como critério de desempate.

- **Analisar Combinação das Abordagens WAT**: após a equipe selecionar o subconjunto de abordagens WAT que deseja utilizar no projeto de software Web, Porantim-WAT analisa o impacto da combinação das abordagens selecionadas sobre algumas variáveis do processo de teste, tais como a cobertura de projetos de software, modelagem, esforço e recursos humanos necessários para utilizar as técnicas.

4.3.1 Adaptação da etapa “Caracterizar Projeto”

Os atributos de caracterização de projeto de software utilizados em Porantim estão apresentados na Tabela 4.1, apresentada a seguir, junto com um conjunto de exemplos de possíveis valores que podem ser associados a eles. Tratam-se de 16 atributos divididos em 2 categorias:

- Características do projeto de software a ser desenvolvido, e;
- Requisitos de teste para o projeto de software.

Tabela 4.1. Atributos de Caracterização de Projeto de Software (DIAS NETO, 2009).

Categoria	Atributos de Caracterização	Exemplo de Valores
Características do Projeto	1. Plataforma de execução do software	Ex: web, desktop, sw embarcado
	2. Paradigma de desenvolvimento adotado no projeto	Ex: Estruturado, Orientado a Objetos, Orientado a Aspectos
	3. Linguagem de programação adotada para construir o software	Ex: Java, C++, PHP, Phyton, Ruby
	4. Modelo comportamental/estrutural de software provido pelo projeto	Ex: Diagramas UML, Máquina de Estado Finito, Grafos
	5. Tecnologia usada para modelagem do software	Ex: UML, TSL, OCL
	6. Duração estimada para o projeto	*Em meses*
	7. Indicador de complexidade do problema	*Alta, Média, Baixa*
	8. Indicador de volatilidade dos requisitos	*Alta, Média, Baixa*
	9. Indicador de tamanho estimado da aplicação	*Grande, Média, Pequena*
	10. Habilidade provida pela equipe de teste alocada para o projeto	*Ex: conhecimento em uma linguagem de programação ou de modelagem, conhecimento sobre um tipo de técnica de teste aplicável a uma plataforma de execução*
Requisitos de Teste	11. Nível(is) de Teste desejado(s) para o projeto	*Ex: unidade, integração, Sistema, aceitação*
	12. Tipo de Técnica de Teste a ser aplicada	Ex: Funcional ou Estrutural
	13. Características de qualidade definidas nos requisitos e que devem ser avaliadas ao longo do projeto	Características de qualidade da norma ISO/IEC 9126 (ex: eficiência, funcionalidade, usabilidade)
	14. Tipos de Defeitos que desejam ser reveladas	*Ex: Tipo de Dados Incorreto, Falha de Banco de Dados, Falha de Navegação*
	15. Apoio Ferramental requerido para testes	*Não requer o uso de ferramenta,* *Apenas ferramentas gratuitas,* *Possibilidade de adquirir Ferramenta*
	16. Custo esperado para os testes	*Alto, Médio, Baixo*

O requisito de adaptação para caracterização de um projeto de software Web consiste na adequação do conjunto de atributos para este tipo de aplicação, orientado por diferentes fontes de pesquisa:

- O conjunto de atributos originalmente proposto em DIAS NETO (2009) para caracterizar projetos de software;
- *quasi*-revisão sistemática para identificar as características das abordagens de teste para aplicações Web disponíveis na literatura técnica, e ;
- Conjunto de atributos de caracterização das abordagens WAT confirmados a partir do survey que apoiou a definição da estrutura do corpo de conhecimento de abordagens de teste para aplicações Web, descrito no Capítulo 3.

Além disso, a partir da análise de resultados do survey no capítulo anterior, foi possível observar que alguns atributos seriam mais influentes no procedimento de seleção. Nesse sentido, o conjunto de atributos foi dividido em dois grupos para verificar a adequação das características das abordagens WAT frente a um projeto de software Web. O primeiro grupo, será utilizado na verificação de adequação propriamente dita. O segundo será utilizado como critério de desempate, caso seja necessário realizar um refinamento das abordagens selecionadas.

Desta forma, os atributos de caracterização de um projeto de software Web, utilizado em Porantim-WAT, estão apresentados na Tabela 4.2.

Tabela 4.2. Atributos de Caracterização de Projeto de Software Web.

Categoria	Atributos de Caracterização	Exemplo de Valores
Características do Projeto	1. Modelo comportamental/estrutural de software provido pelo projeto	Ex: Diagramas UML, Máquina de Estado Finito, Grafos
	2. Tecnologia usada para modelagem	Ex: UML, WSDL
	3. Habilidade provida pela equipe de teste alocada para o projeto	*Ex: conhecimento em um tipo de modelagem, conhecimento sobre um tipo de técnica de teste aplicável a uma plataforma de execução*
Requisitos de Teste	4. Tipos de Defeitos que desejam ser reveladas	*Ex: Tipo de Dados Incorreto, Falha de Banco de Dados, Falha de Navegação*
	5. Tipo de Técnica de Teste a ser aplicada	Ex: Funcional ou Estrutural
	6. Critério de Cobertura	*Ex: Classe de Equivalência, Todos os caminhos, etc.*
	7. Características de qualidade definidas nos requisitos e que devem ser avaliadas ao	Características de qualidade da norma ISO/IEC 9126 (ex:

	longo do projeto	eficiência, funcionalidade, usabilidade)
	8. Tarefas Automatizadas	*Ex: Sim ou Não*
	9. Nível de Complexidade dos passos não automatizados	*Ex: Baixo, Médio, Alto*

A adequação do conjunto de atributos de caracterização de projeto de software Web resultou na necessidade de evoluir a estrutura do corpo de conhecimento sobre abordagens WAT definido no Capítulo 3. Foi necessária a inclusão do atributo "Habilidades Requeridas pela Abordagem" para que ficassem compatíveis aos atributos usados para caracterização do projeto. Este atributo herdará o peso obtido no atributo que indica as limitações /restrições para utilização da abordagem WAT.

Em caso de empate, ou seja, quando duas ou mais abordagens apresentam o mesmo nível de adequação para um determinado projeto de software Web, ou ainda, quando for necessário refinar a seleção das abordagens de teste, um segundo grupo de informações sobre a caracterização do projeto, será utilizado como critério de desempate. Desta forma, outros atributos de caracterização de um projeto de software Web deverão ser informados. Os atributos utilizados em Porantim-WAT como critério de desempate, estão apresentados na tabela 4.3.

Tabela 4.3. Atributos utilizados como critério desempate na Caracterização de Projeto.

Categoria	Atributos de Caracterização	Exemplo de Valores
Características do Projeto	1. Categoria da Aplicação Web	Ex: *Browser-Based*, Serviços Web, *e-commerce, workflow*, colaborativas
Requisitos de Teste	2. Nível(is) de Teste desejado(s) para o projeto	*Ex: unidade, integração, Sistema, aceitação*
	3. Análise de Teste	Ex: Dinâmica ou Estática
	4. Tipo de Avaliação da Abordagem	*Ex: Estudo de Caso, Estudo Experimental*
	5. Camada da aplicação a ser testada	Ex: Front-End ou Back-End

4.3.2 Adaptação da etapa "Calcular o Grau de Adequação"

Para o cálculo do Grau de Adequação, os atributos do projeto de software deverão ser confrontados com os atributos de caracterização das abordagens WAT, seguindo o algoritmo proposto em DIAS NETO (2009) que utiliza o conceito matemático de distância euclidiana (BOLDRINI et al., 1980).

Este mecanismo transforma as características do projeto e de cada abordagem de teste em valores, que posteriormente são representados através de vetores (v1 –

vetor das características do projeto de software; e v2 – vetor das características de cada abordagem de teste). A distância entre os vetores indica o grau de adequação de uma abordagem de teste para o projeto de software. Quanto mais próximos, maior o grau de adequação.

Nesse contexto, o cálculo de adequação deverá ser adaptado apenas em relação aos atributos específicos das abordagens WAT (descrito no Capítulo 3) e aos atributos de caracterização do projeto de software Web (descrito na seção anterior).

Tabela 4.4. Relacionamento entre Atributos de Caracterização de Projeto Web e abordagens WAT.

Atributos de Caracterização	Atributos WAT	Observação
1. Categoria da Aplicação Web	Categoria da aplicação que a abordagem pode ser utilizada	Utilizado apenas como critério de desempate
2. Modelo comportamental/estrutural de software provido pelo projeto	Modelos utilizados pela abordagem	
3. Tecnologia usada para modelagem do software	Tecnologia utilizada para geração dos testes	
4. Habilidade provida pela equipe de teste alocada para o projeto	*Habilidade requerida para ser operada*	
5. Nível(is) de Teste desejado(s) para o projeto	*Níveis de Teste explorados ela abordagem*	*Utilizado apenas como critério de desempate*
6. Tipo de Técnica de Teste a ser aplicada	Indicação da técnica que a abordagem utiliza	
7. Análise de Teste a ser realizada	Tipo de análise realizada pela abordagem	Utilizado apenas como critério de desempate
8. Características de qualidade definidas nos requisitos e que devem ser avaliadas ao longo do projeto	Características de qualidade que a abordagem WAT é capaz de avaliar	
9. Tipos de Defeitos que desejam ser reveladas	*Tipos de defeitos que a abordagem é capaz de revelar.*	
10. Tipo de Avaliação da Abordagem	*Indicação sobre a estratégia experimental e as evidências adquiridas através da experimentação da abordagem WAT.*	*Utilizado apenas como critério de desempate*
11. Camada da aplicação a ser testada	*Informações sobre qual camada a abordagem WAT pode ser aplicada.*	*Utilizado apenas como critério de desempate*
12. Tarefas Automatizadas	*Descreve se a abordagem WAT é automatizada*	
13. Nível de Complexidade dos	*Indicação sobre o*	

passos não automatizados	*esforço, tempo, custo e para executar as tarefas não-automatizadas na utilização da abordagem WAT.*	
14. Critério de Cobertura	*Informação sobre as regras utilizadas pela abordagem WAT para gerar casos de teste*	

Além disso, cada atributo possui um peso específico definido pelo Índice de Relevância apresentado no Capítulo 3. Desta forma, os atributos exercem influências diferenciadas no procedimento de seleção. A influência de cada atributo é calculada pela divisão do peso do atributo pela soma dos pesos de todos os atributos. Os atributos das abordagens WAT que não podem ser confrontados com os atributos do projeto não são usados para calcular o Grau de Adequação. No entanto, eles compõem a abordagem para prover mais informações que podem ser consultadas para apoiar a tomada de decisão no momento da seleção das técnicas. A lista de atributos, seus pesos e influência estão apresentados na Tabela 4.5

Tabela 4.5. Atributos de Caracterização de Projeto de Software e seus pesos.

Atributos de Caracterização	Pesos	Influências
1. Modelos	0,6525	8,14%
2. Tecnologia de modelagem	0,5743	7,17%
3. Habilidades	0,6431	8,03%
4. Tipos de Falhas	0,7332	9,15%
5. Técnica de Teste	0,6166	7,69%
6. Critério de Cobertura	0,6915	8,63%
7. Características de qualidade	0,5633	7,03%
8.Tarefas Automatizadas	0,6975	8,70%
9.Nível de Complexidade dos passos não automatizados	0,5763	7,19%
10. Categoria da Aplicação Web	0,4518	5,64%
11. Nível de Teste	0,5042	6,29%
12. Análise de Teste	0,4481	5,59%
12.Tipo de Avaliação da Abordagem	0,4668	5,83%
14.Camada da aplicação a ser testada	0,3938	4,91%

De acordo com o algoritmo proposto em DIAS NETO (2009), para a representação do vetor de um determinado projeto de software, cada atributo de caracterização do projeto será transformado nos valores definidos pela sua influência, conforme detalhado no quadro abaixo:

$V_1 = (X_1 \mid X_2 \mid X_3 \mid X_4 \mid X_5 \mid X_6 \mid X_7 \mid X_8 \mid X_9 \mid X_{10} \mid X_{11} \mid X_{12} \mid X_{13} \mid X_{14})$

Onde:

V_1 é o vetor representa um projeto de software Web.

X_i é a valor da influência do atributo i que caracteriza o projeto de software Web (identificados na tabela 4.5).

Desta forma, o vetor será composto pela influência dos atributos abaixo listados:

X_1 – Valor da influência do atributo "Modelos".

X_2 – Valor da influência do atributo "Tecnologia de modelagem".

X_3 – Valor da influência do atributo "Habilidades".

X_4 – Valor da influência do atributo "Tipos de Falhas".

X_5 – Valor da influência do atributo "Técnica de Teste".

X_6 – Valor da influência do atributo "Critério de Cobertura".

X_7 – Valor da influência do atributo "Características de qualidade".

X_8 – Valor da influência do atributo "Tarefas Automatizadas".

X_9 – Valor da influência do atributo "Nível de Complexidade dos passos não automatizados".

X_{10} – Valor da influência do atributo "Categoria da Aplicação Web".

X_{11} – Valor da influência do atributo "Nível de Teste".

X_{12} – Valor da influência do atributo "Análise de Teste".

X_{13} – Valor da influência do atributo "Tipo de Avaliação da Abordagem".

X_{14} – Valor da influência do atributo "Camada da aplicação a ser testada".

Para a representação do vetor que irá representar uma determinada abordagem WAT, as transformações de cada atributo em valores serão realizadas obedecendo à seguinte regra:

SE *[atributo do projeto de software = atributo da abordagem de teste]*
ENTÃO *atributo da abordagem de teste é transformado em 1 x influência do atributo*
CASO CONTRÁRIO *recebe valor 0*

4.3.3 Adaptação da etapa “Indicar Abordagens WAT mais Adequadas”

Nesta etapa do procedimento de seleção lista, em ordem decrescente, o subconjunto das abordagens WAT que mais se adequam ao projeto. Além disso, é possível consultar os detalhes das características das abordagens listadas.

Neste contexto a apresentação das abordagens deverá ser adaptada para apresentar informações associadas às abordagens WAT.

4.3.4 Adaptação da etapa “Selecionar Abordagens WAT para Projeto de Software Web”

O requisito de adaptação desta etapa consiste principalmente em oferecer um fluxo alternativo para o usuário. A motivação para este novo fluxo de execução baseia-se no fato de que cada atributo possui um peso específico definido pelo Índice de Relevância (apresentado no Capítulo 3) e que, desta forma, exercem influências diferenciadas no procedimento de seleção.

Nesse sentido, o conjunto de atributos que caracterizam as abordagens WAT foi dividido em dois grupos para verificar a sua adequação frente a um projeto de software Web. O primeiro grupo é composto pelos 9 atributos apresentados na Tabela 4.2, e será utilizado na primeira iteração do cálculo do grau de adequação.

Ao final da primeira iteração do procedimento de seleção, caso haja empate no grau de adequação entre as abordagens selecionadas por Porantim-WAT, ou ainda, caso seja desejável refinar a seleção das abordagens de teste por qualquer motivo, será disponibilizada ao usuário a opção “Refinar Seleção”.

Ao escolher essa opção, o procedimento será retornado à etapa “Caracterizar Projeto de Software Web”. Nesse momento a equipe de teste irá complementar a caracterização do projeto Web, informando 5 novos atributos, descritos na Tabela 4.3. Desta forma, espera-se obter um resultado mais refinado da busca, visto que, no total, 14 atributos serão considerados pelo mecanismo de seleção.

4.3.5 Adaptação da etapa “Analisar a Combinação das Abordagens WAT”

A análise do uso combinado de técnicas de teste baseado em modelos utilizado por Porantim não será alterada no procedimento Porantim-WAT proposto. No entanto, é necessário adaptar o mecanismo de análise em relação aos atributos específicos das abordagens WAT (descrito no Capítulo 3) e aos atributos de caracterização do projeto de software Web (descrito na seção anterior).

4.3.6 Exemplo

As Tabelas 4.6 e 4.7 apresentam exemplos fictícios de instanciação da caracterização de um projeto de software Web e a caracterização referente a duas abordagens WAT, identificadas como WAT#1 e WAT#2. Essas abordagens serão utilizadas para exemplificar o cálculo da distância entre projetos e abordagens WAT, de acordo com o procedimento Porantim-WAT proposto. São apresentados dois cenários:

Cenário 1) Não haverá empate entre as abordagens quanto à adequação ao projeto de software web caracterizado.

Cenário 2) Haverá empate entre as abordagens quanto à adequação ao projeto de software web caracterizado. Nesse caso, será necessário utilizar os atributos de caracterização adicionais, como critério de desempate.

Tabela 4.6. Exemplo de caracterização de projeto e abordagens WAT (Cenário 1).

Atributos de Caracterização	**Projeto**	**WAT #1**	**WAT #2**
1. Modelos	Diagrama de Classe	Diagrama de Classe	Diagrama de Casos de Uso
2. Tecnologia de modelagem	UML	UML	UML
3. Habilidades	*Conhecimento em UML*	*Conhecimento em UML*	*Conhecimento em UML*
4. Tipos de Defeitos	*Tipo Incorreto de Dados*	*Tipo Incorreto de Dados*	*Tipo Incorreto de Dados*
5. Técnica de Teste	Funcional	Funcional	Funcional
6. Critério de Cobertura	Valores Limítrofes	Valores Limítrofes	Classe de Equivalência
7. Características de qualidade	Confiabilidade	Confiabilidade	Confiabilidade
8. Tarefas automatizadas	NÃO	NÃO	NÃO
9. Complexidades Passo não automatizados	*Baixo*	*Baixo*	*Médio*
Grau de Adequação		*100%*	*86,12%*

<u>Representação vetorial do Projeto e das abordagens WAT#1 e WAT#2</u>

- Transformação das características do projeto no vetor $V_{Projeto}$

$V_{Projeto}$= *(0,0814 | 0,0717 | 0,0803 | 0,0915 | 0,0769 | 0,0863 | 0,0703 | 0,0870 | 0,0719)*

- Transformação das características da abordagem WAT#1 no vetor $V_{Wat\#1}$

$V_{Wat\#1}$= *(0,0814 * 1 | 0,0717 * 1 | 0,0803 * 1 | 0,091 5* 1 | 0,0769 * 1 | 0,0863 * 1 | 0,0703 * 1 | 0,0870 * 1 | 0,0719 * 1)*

$V_{Wat\#1}$= *(0,0814 | 0,0717 | 0,0803 | 0,091 5 | 0,0769 | 0,0863 | 0,0703 | 0,0870 | 0,0719)*

- Transformação das características da abordagem WAT#2 no vetor $V_{Wat\#2}$

$V_{Wat\#2}$= *(0,0814 * 0 | 0,0717 * 1 | 0,0803 * 1 | 0,0915 * 1 | 0,0769 * 1 | 0,0863 * 0 | 0,0703 * 1 | 0,0870 * 1 | 0,0719 * 0)*

$V_{Wat\#2}$= *(0 | 0,0717 | 0,0803 | 0,0915 | 0,0769 | 0 | 0,0703 | 0,0870 | 0)*

Cálculo da distância entre a representação vetorial do Projeto e a representação vetorial de cada abordagem:

- Cálculo do nível de adequação entre $V_{Projeto}$ e $V_{Wat\#1}$

Nível de Adequação ($V_{Projeto}$, $V_{Wat\#1}$) = $|1 - distancia(V_{Projeto}, V_{Wat\#1})| * 100$

distância ($V_{Projeto}$, $V_{Wat\#1}$) = $\sqrt{(V_{Projeto} - V_{Wat\#1})^2}$

distância ($V_{Projeto}$, $V_{Wat\#1}$) = $\sqrt{0^2} = 0$

Nível de Adequação ($V_{Projeto}$, $V_{Wat\#1}$) = $|1 - 0| * 100 - 100\%$

- Cálculo do nível de adequação entre $V_{Projeto}$ e $V_{Wat\#2}$

Nível de Adequação ($V_{Projeto}$, $V_{Wat\#2}$) = $|1 - distancia(V_{Projeto}, V_{Wat\#2})| * 100$

distância ($V_{Projeto}$, $V_{Wat\#2}$) = $\sqrt{(V_{Projeto} - V_{Wat\#2})^2}$

distância ($V_{Projeto}$, $V_{Wat\#2}$) = $\sqrt{(0,0814 - 0)^2 + 0^2 + 0^2 + 0^2 + (0,0863 - 0)^2 + 0^2 + 0^2 + (0,0719 - 0)^2}$

distância ($V_{Projeto}$, $V_{Wat\#1}$) = $\sqrt{0,00662596 + 0,00744769 + 0,00516961}$

distância ($V_{Projeto}$, $V_{Wat\#2}$) = $\sqrt{0,01924326}$

distância ($V_{Projeto}$, $V_{Wat\#2}$) = $0,13872007$

Nível de Adequação ($V_{Projeto}$, $V_{Wat\#2}$) = $|1 - 0,13872007| * 100$

Nível de Adequação ($V_{Projeto}$, $V_{Wat\#2}$) = $0,86127992 * 100 = 86,12\ \%$

Tabela 4.7. Exemplo de caracterização de projeto e abordagens WAT (Cenário 2).

Atributos de Caracterização	Projeto	WAT #1	WAT #2
1. Modelos	Diagrama de Classe	Diagrama de Classe	Diagrama de Classe
2. Tecnologia de modelagem	UML	UML	UML
3. Habilidades	*Conhecimento em UML*	*Conhecimento em UML*	*Conhecimento em UML*
4. Tipos de Falhas	*Tipo Incorreto de Dados*	*Erro de Interface*	*Erros de Formulário*
5. Técnica de Teste	Funcional	Funcional	Funcional
6. Critério de Cobertura	Valores Limítrofes	Classe de Equivalência	Classe de Equivalência
7. Características de qualidade	Confiabilidade	Confiabilidade	Confiabilidade
8. Tarefas automatizadas	Não	Não	Não
9. Complexidades Passo não automatizados	*Baixo*	*Médio*	*Alto*

Grau de Adequação		*85,34%*	*85,34%*
10. Categoria da Aplicação	*Browser-Based*	*Serviços Web*	*Browser-Based*
11. Nível de Teste	*Unidade*	*Integração*	*Unidade*
12. Análise de Teste	*Estático*	*Dinâmico*	*Estático*
13.Tipo de Avaliação	*Estudo de Caso*	*Estudo de Caso*	*Estudo de Caso*
14. Camada da Aplicação	*Cliente*	*Cliente*	*Cliente*
Grau de Adequação – Critério Desempate		*82,18%*	*85,34%*

Representação vetorial do Projeto e das abordagens WAT#1 e WAT#2

- Transformação das características do projeto no vetor $V_{Projeto}$

$V_{Projeto}$= *(0,0814 | 0,0717 | 0,0803 | 0,0915 | 0,0769 | 0,0863 | 0,0703 | 0,0870 | 0,0719)*

- Transformação das características da abordagem WAT#1 no vetor $V_{Wat\#1}$

$V_{Wat\#1}$= *(0,0814 * 1 | 0,0717 * 1 | 0,0803 * 1 | 0,091 5* 0 | 0,0769 * 1 | 0,0863 * 0 | 0,0703 * 1 | 0,0870 * 1 | 0,0719 * 0)*

$V_{Wat\#1}$= *(0,0814 | 0,0717 | 0,0803 | 0 | 0,0769 | 0 | 0,0703 | 0,0870 | 0)*

- Transformação das características da abordagem WAT#2 no vetor $V_{Wat\#2}$

$V_{Wat\#2}$= *(0,0814 * 1 | 0,0717 * 1 | 0,0803 * 1 | 0,091 5* 0 | 0,0769 * 1 | 0,0863 * 0 | 0,0703 * 1 | 0,0870 * 1 | 0,0719 * 0)*

$V_{Wat\#2}$= *(0,0814 | 0,0717 | 0,0803 | 0 | 0,0769 | 0 | 0,0703 | 0,0870 | 0)*

Cálculo da distância entre a representação vetorial do Projeto e a representação vetorial de cada abordagem:

- Cálculo do nível de adequação entre $V_{Projeto}$ *e* $V_{Wat\#1}$ *ou* $(V_{Wat\#2})$

Nível de Adequação $(V_{Projeto}, V_{Wat\#1}) = |1 - distancia(V_{Projeto}, V_{Wat\#1})| * 100$

distância($V_{Projeto}, V_{Wat\#1}) = \sqrt{(V_{Projeto} - V_{Wat\#1})^2}$

distância($V_{Projeto}, V_{Wat\#1}) = \sqrt{0^2 + 0^2 + 0^2 + (0{,}0915-0)^2 + 0^2 + (0{,}0863 - 0)^2 + 0^2 + 0^2 + (0{,}0719 - 0)^2}$

distância($V_{Projeto}, V_{Wat\#1}) = \sqrt{0{,}00837225 + 0{,}00744769 + 0{,}00516961}$

distância($V_{Projeto}, V_{Wat\#1}) = \sqrt{0{,}02148955}$

distância($V_{Projeto}, V_{Wat\#1}) = 0{,}14659314$

Nível de Adequação $(V_{Projeto}, V_{Wat\#1}) = |1 - 0{,}14659314| * 100$

Nível de Adequação $(V_{Projeto}, V_{Wat\#1}) = 0{,}85340685 * 100 = 85{,}34\ \%$

Podemos observar que $V_{Wat\#1}$ é igual a $V_{Wat\#2}$. Desta forma, a distância ($V_{Projeto}$, $V_{Wat\#1}$) também será igual à distância ($V_{Projeto,}$, $V_{Wat\#2}$). Nesse cenário houve empate entre as abordagens quanto à adequação ao projeto de software web caracterizado. Nesse caso, será necessário utilizar os atributos de caracterização adicionais, como critério de desempate.

Representação vetorial do Projeto e das abordagens WAT#1 e WAT#2, observando os atributos do critério de desempate

- Transformação das características do projeto no vetor $V_{Projeto}$

$V_{Projeto}$= (0,0814 | 0,0717 | 0,0803 | 0,0915 | 0,0769 | 0,0863 | 0,0703 | 0,0870 | 0,0719 | 0,0564 | 0,0629 | 0,0559 | 0,0583 | 0,0491)

- Transformação das características da abordagem WAT#1 no vetor $V_{Wat\#1}$

$V_{Wat\#1}$= (0,0814 * 1 | 0,0717 * 1 | 0,0803 * 1 | 0,091 5* 0 | 0,0769 * 1 | 0,0863 * 0 | 0,0703 * 1 | 0,0870 * 1 | 0,0719 * 0 | 0,0564 * 0 | 0,0629 * 0 | 0,0559 * 0 | 0,0583 * 1 | 0,0491 * 1)

$V_{Wat\#1}$= (0,0814 | 0,0717 | 0,0803 | 0 | 0,0769 | 0 | 0,0703 | 0,0870 | 0 | 0 | 0 | 0 | 0,0583 | 0,0491)

- Transformação das características da abordagem WAT#2 no vetor $V_{Wat\#2}$

$V_{Wat\#2}$= (0,0814 * 1 | 0,0717 * 1 | 0,0803 * 1 | 0,091 5* 0 | 0,0769 * 1 | 0,0863 * 0 | 0,0703 * 1 | 0,0870 * 1 | 0,0719 * 0 | 0,0564 * 1 | 0,0629 * 1 | 0,0559 * 1 | 0,0583 * 1 | 0,0491 * 1)

$V_{Wat\#2}$= (0,0814 | 0,0717 | 0,0803 | 0 | 0,0769 | 0 | 0,0703 | 0,0870 | 0 | 0,0564 | 0,0629 | 0,0559 | 0,0583 | 0,0491)

Cálculo da distância entre a representação vetorial do Projeto e a representação vetorial de cada abordagem:

- Cálculo do nível de adequação entre $V_{Projeto}$ e $V_{Wat\#1}$

$Nível\ de\ Adequação\ (V_{Projeto},\ V_{Wat\#1}) = |1 - distancia(V_{Projeto},\ V_{Wat\#1})| * 100$

$distância(V_{Projeto},\ V_{Wat\#1}) = \sqrt{(V_{Projeto} - V_{Wat\#1})^2}$

$distância(V_{Projeto},\ V_{Wat\#1}) = \sqrt{0^2 + 0^2 + 0^2 + (0,0915-0)^2 + 0^2 + (0,0863 - 0)^2 + 0^2 + 0^2 + (0,0719 - 0)^2 + (0,0564 - 0)^2 + (0,0629 - 0)^2 + (0,0559 - 0)^2 + 0^2 + 0^2}$

$distância(V_{Projeto},\ V_{Wat\#1}) = \sqrt{0,00837225 + 0,00744769 + 0,00516961 + 0,00318096 + 0,00395641 + 0,00312481}$

$distância(V_{Projeto},\ V_{Wat\#1}) = \sqrt{0,03175173}$

$distância(V_{Projeto},\ V_{Wat\#1}) = 0,17819015$

$Nível\ de\ Adequação\ (V_{Projeto},\ V_{Wat\#1}) = |1 - 0,17819015| * 100$

$Nível\ de\ Adequação\ (V_{Projeto},\ V_{Wat\#1}) = 0,82180984 * 100 = 82,18\ \%$

- Cálculo do nível de adequação entre $V_{Projeto}$ e $V_{Wat\#2}$

$Nível\ de\ Adequação\ (V_{Projeto},\ V_{Wat\#2}) = |1 - distancia(V_{Projeto},\ V_{Wat\#2})| * 100$

$distância(V_{Projeto},\ V_{Wat\#2}) = \sqrt{(V_{Projeto} - V_{Wat\#2})^2}$

$distância(V_{Projeto},\ V_{Wat\#2}) = \sqrt{0^2 + 0^2 + 0^2 + (0,0915-0)^2 + 0^2 + (0,0863-0)^2 + 0^2 + 0^2 + (0,0719-0)^2 + 0^2 + 0^2 + 0^2 + 0^2 + 0^2}$

$distância(V_{Projeto},\ V_{Wat\#2}) = \sqrt{0,00837225 + 0,00744769 + 0,00516961}$

$distância(V_{Projeto},\ V_{Wat\#2}) = \sqrt{0,02148955}$

$distância(V_{Projeto},\ V_{Wat\#2}) = 0,14659314$

Nível de Adequação ($V_{Projeto,\ V_{Wat\#2}}$) =|1− 0,14659314 | *100

Nível de Adequação ($V_{Projeto,\ V_{Wat\#2}}$) = 0,85340685 *100 = 85,34 %

Resultado: WAT#2 é mais adequada ao Projeto caracterizado.

4.4 Maraká: Infraestrutura computacional para apoiar o procedimento Porantim-WAT

4.4.1 Visão Geral e Evolução

Maraká é uma infra-estrutura computacional que apóia o planejamento e controle do processo de teste, proposta por DIAS NETO e TRAVASSOS (2006), desenvolvida sobre o *framework* de uso livre Joomla! (www.joomla.org) e utiliza as tecnologias PHP+MySQL.

A infra-estrutura Maraká foi originalmente projetada para apoiar o planejamento e controle de testes de software através do acompanhamento do processo de testes e a documentação das atividades realizadas ao longo deste processo. As funcionalidades inicialmente fornecidas envolvem: (a) o gerenciamento da equipe de teste; (b) o gerenciamento das atividades de teste, a partir da criação de novas atividades de teste a serem acompanhadas através da infra-estrutura e da consulta aos artefatos produzidos; e (c) o gerenciamento do processo de testes, através do acompanhamento do processo (planejamento, projeto e execução de casos/procedimentos de teste), para cada atividade de teste criada.

Posteriormente, a infra-estrutura foi evoluída para prover apoio ao procedimento de seleção de TTBMs, a partir da inclusão do repositório de Técnicas TBM e do componente Gerenciador de Porantim, que está diretamente associado ao componente de Maraká responsável pelo gerenciamento do processo de teste para um projeto de software.

Este trabalho propõe uma nova evolução de Maraká para automatizar o procedimento Porantim-WAT, cujo objetivo é apoiar a seleção de abordagens WAT para projetos de software Web. Para isso, foram incluídos o repositório de abordagens de teste para aplicações Web e o componente Gerenciador Porantim-WAT, ilustrado na Figura 4.2.

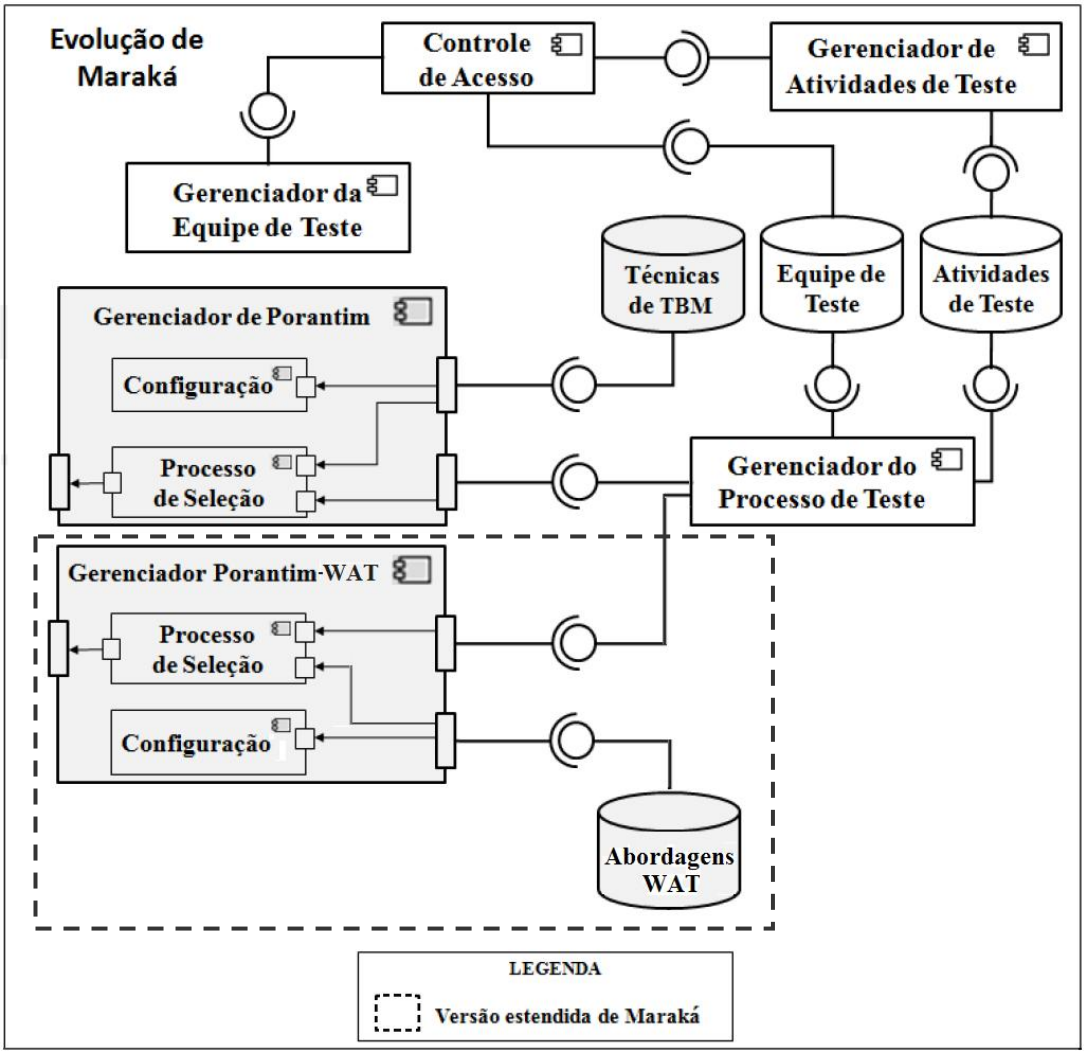


Figura 4.4. Evolução da Arquitetura de Maraká.

O repositório das abordagens WAT implementa o corpo de conhecimento de abordagens de teste para aplicações Web, apresentado no Capítulo 3 deste trabalho, enquanto que o componente Gerenciador Porantim-WAT implementa o procedimento de seleção das abordagens WAT para um projeto de software Web (apresentado na Seção 4.3). O componente Gerenciador Porantim-WAT é responsável pelas seguintes funcionalidades:

- **Configuração do corpo de conhecimento**: trata-se do gerenciamento das abordagens WAT através de mecanismos de cadastro, edição, exclusão, visualização e importação.
- **Configuração dos parâmetros adotados por Porantim-WAT**: o procedimento de seleção das abordagens de teste para projetos de software Web inclui a definição dos pesos de cada atributo de

caracterização das abordagens WAT e definição da quantidade máxima de abordagens a serem exibidas no procedimento de seleção.

- **Execução do procedimento de seleção de abordagens WAT**: realizado para cada projeto de teste através do cálculo de adequação entre as características do projeto de software Web e as características das abordagens WAT.

A seguir serão descritas as funcionalidades implementadas no componente Gerenciador de Porantim-WAT para apoiar o uso do procedimento através da infra-estrutura Maraká.

4.4.2 Funcionalidades Adaptadas

4.4.2.1 Configuração do Corpo de Conhecimento

A configuração do corpo de conhecimento de *Porantim-WAT*, inclui ações de cadastro, edição, exclusão, visualização e importação das abordagens de teste para aplicações Web. Para acessar esta funcionalidade, a partir da tela inicial de Maraká, escolha a opção **Configuração → Abordagens WAT** (Figura 4.5).

Desta forma, será apresentada uma tela que irá listar as abordagens WAT existentes no repositório e as funcionalidades de cadastro, edição, exclusão e visualização (Figura 4.6).

Figura 4.5. Tela de Configuração da Infra-Estrutura Maraká.

Figura 4.6. Tela de Gerenciamento do Corpo de Conhecimento.

Além do cadastro manual das abordagens de teste, Maraká apresenta a funcionalidade de importação, que também pode ser acessada a partir da página apresentada na Figura 4.6. Tal funcionalidade permite que as abordagens WAT listadas na ferramenta JabRef[3] tenham seus dados importados em lotes para o repositório mantido por Maraká.

A carga é realizada utilizando arquivos BibTeX (extensão .bib) que deverá apresentar um formato pré-definido, contendo um conjunto de campos que descrevem os atributos de caracterização de uma abordagem WAT. Para definir os campos que irão compor o formulário, acessar o menu **Options → Set up General Fields** no JabRef, e inserir as linhas destacadas na Figura 4.7.

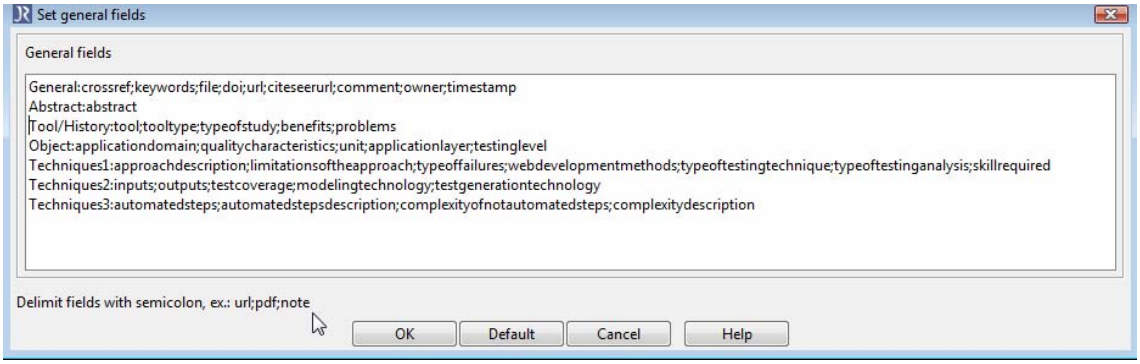


Figura 4.7. Configuração dos campos de caracterização das abordagens WAT no JabRef.

Além disso, algumas regras para preenchimento dos campos de caracterização das abordagens WAT na ferramenta JabRef devem ser respeitadas. Essas regras estão descritas na Tabela 4.8.

[3] JabRef - Ferramenta de gerenciamento de referências bibliográficas (http://jabref.sourceforge.net)

Tabela 4.8. Regras de Mapeamento e Preenchimento do Jabref.

Atributo da Abordagem WAT	Tag JabRef	Regra de Preenchimento
Benefícios	Benefits	Texto Livre
Camada da Aplicação	Applicationlayer	Utilizar "," como separador
Característica de Qualidade	qualitycharacteristics	Utilizar "," como separador
Categoria da Aplicação	applicationdomain	Utilizar "," como separador
Complexidade dos Passos não-automatizados	complexityofnotautomatedsteps	1- Baixo, 2- Médio, 3- Alto
Descrição da Complexidade	complexitydescription	Texto Livre
Critérios de Cobertura	Testcoverage	Utilizar "," como separador
Descrição	approachdescription	Texto Livre
Entradas	Inputs	Texto Livre
Habilidades	Skillrequired	Utilizar "," como separador
Limitações	limitationsoftheapproach	Texto Livre
Metodologia de Desenvolvimento	webdevelopmentmethods	Texto Livre
Nível de Teste	Testinglevel	Utilizar "," como separador
Modelo	modelingtechnology	Utilizar "," como separador
Automação	Automatedsteps	0 - Não 1 - Sim
Descrição das tarefas automatizadas	Automatedstepsdescription	Texto Livre
Ferramenta	Tool	Texto Livre
Tipo de Ferramenta	Tooltype	[Protótipo, Comercial]
Problemas	Problems	Texto Livre
Saídas	Outputs	Texto Livre
Tecnologia de Modelagem	Testgenerationtechnology	Utilizar "," como separador
Tipo de Estudo	Typeofstudy	[Estudo de Caso, Estudo Experimental, Prova de Conceito, Exemplo]
Tipo de Técnica	Typeoftestingtechnique	Utilizar "," como separador
Tipo de Análise	Typeoftestinganalisys	Utilizar "," como separador
Unidade	Unit	Texto Livre

Para executar a importação basta clicar no botão **Importar Abordagem WAT** disponível na tela mostrada na Figura 4.8, que irá solicitar a indicação de um arquivo. O próximo passo consiste na escolha das abordagens WAT identificadas no arquivo que deverão ser importadas para o repositório em Maraká.

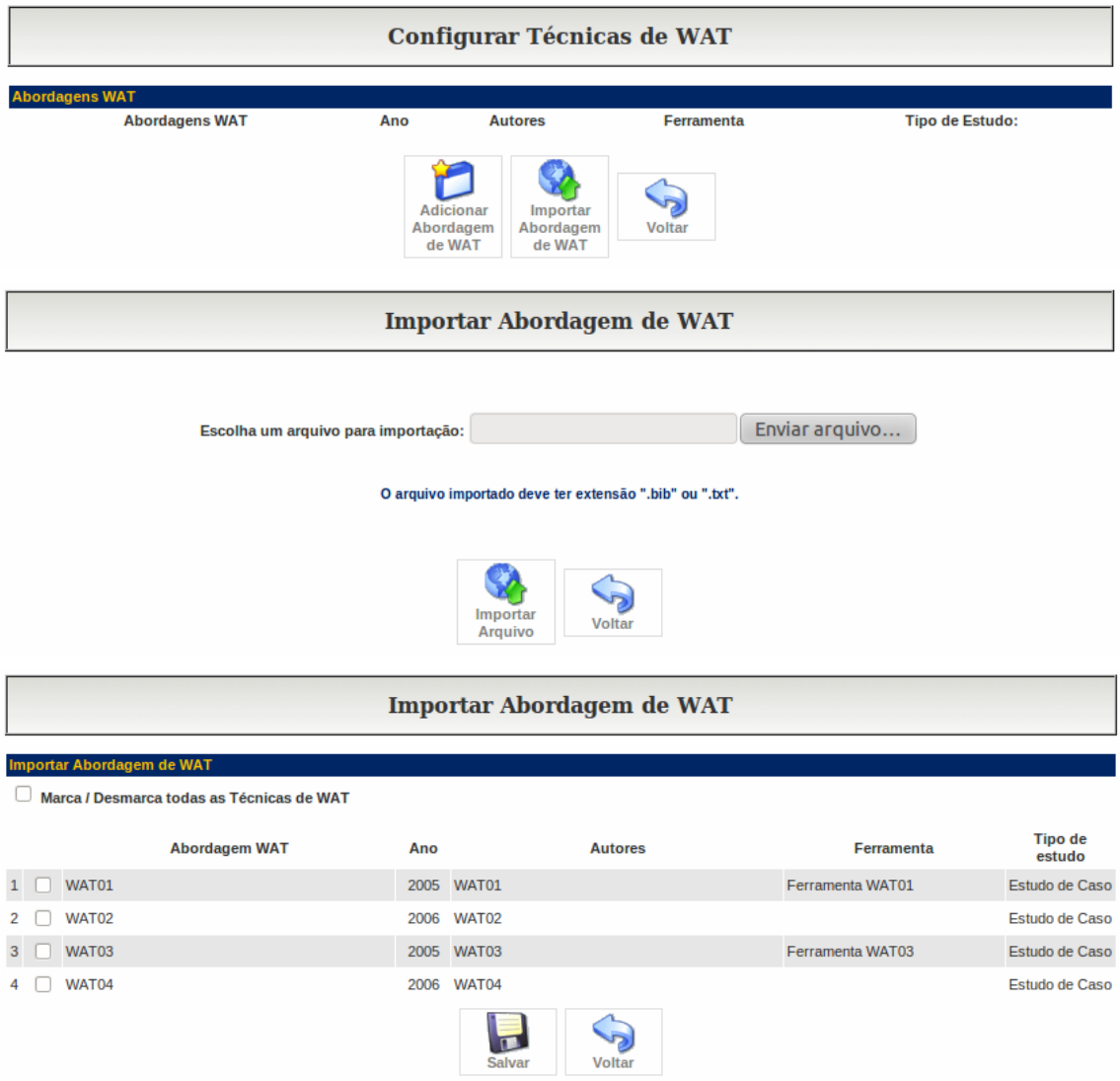


Figura 4.8. Passos para importação de abordagens utilizando arquivo bibtex.

4.4.2.2 Configuração dos Parâmetros adotados por Porantim-WAT

É possível ajustar o peso dos atributos de caracterização das abordagens WAT para o procedimento de seleção. Os valores *default* dos pesos foram atribuídos a partir do Índice de Relevância obtido através do survey publicado em (SANTA ISABEL e TRAVASSOS, 2011) e descrito no Capítulo 3.

É importante destacar que o valor dos pesos pode ser alterado a qualquer tempo, de acordo com as características e necessidades do projeto de software que esteja utilizando o procedimento Porantim-WAT. Entretanto, a soma dos pesos de todos os atributos de caracterização sempre deverá ser igual a 1, representando 100%.

Além disso, é possível configurar se o atributo será obrigatoriamente considerado no procedimento de seleção através da marcação de opcionalidade, localizado ao lado do campo de peso de cada atributo de caracterização (Figura 4.9).

Outra configuração possível é a identificação da quantidade máxima de abordagens WAT a serem sugeridas pela infra-estrutura. Todas as funcionalidades que acabam de ser descritas podem ser acessadas na opção **Configuração → Configurar Porantim WAT**, a partir da tela inicial de Maraká.

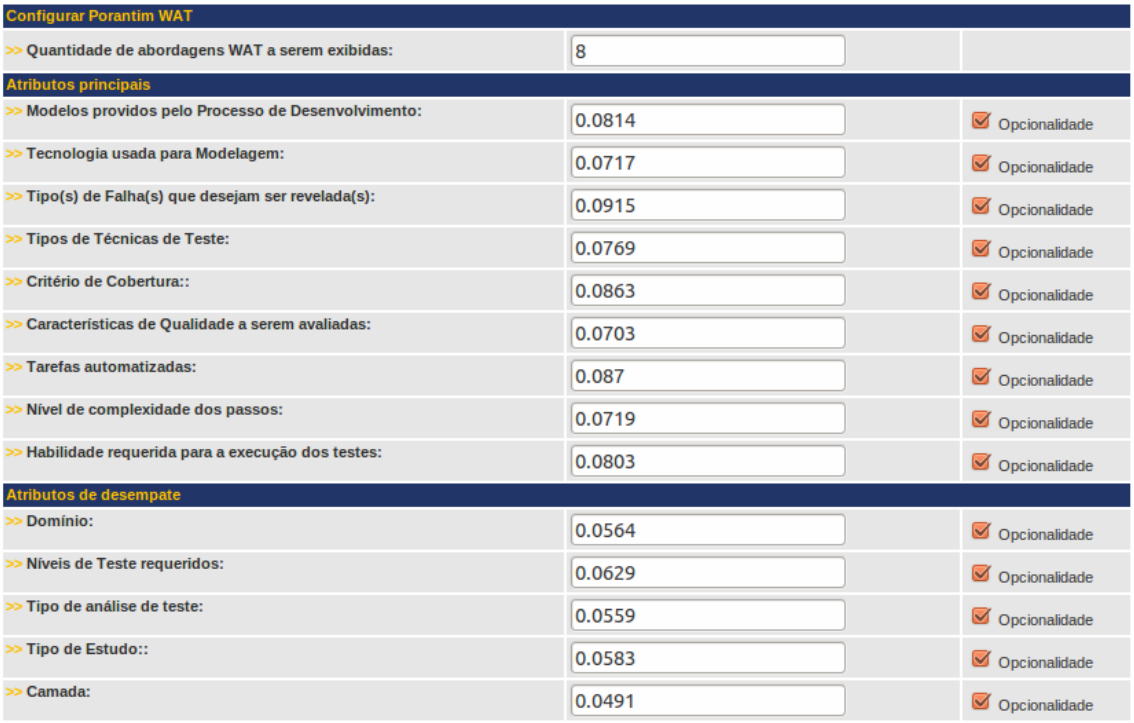


Figura 4.9. Tela de Configuração dos pesos dos Atributos de Caracterização de abordagens WAT e seus pesos.

4.4.2.3 Execução do Procedimento de Seleção das Abordagens WAT

A execução do procedimento de seleção das abordagens WAT para um projeto de software Web ocorre na sub-atividade chamada **Definir Técnicas de Teste** durante a atividade **Planejar Testes**. Na versão anterior de Maraká, esta atividade podia feita através da descrição manual das técnicas de teste ou com o auxílio de Porantim, lembrando que Porantim só se aplica ao contexto de TTBMs.

Com o advento de Porantim-WAT surge uma nova funcionalidade em Maraká e uma nova opção é apresentada durante a execução da sub-atividade chamada **Definir Técnicas de Teste**: Porantim-WAT – Procedimento de apoio na seleção de abordagens de teste para aplicações Web, conforme apresentado na figura 4.10.

Figura 4.10. Tela de Escolha do Procedimento para Seleção de Técnicas de Teste em Maraká.

O botão “Iniciar o uso de Porantim-WAT” implementa o mecanismo de seleção, apresentado na Seção 4.3. Na etapa **Caracterizar Projeto de Software,** é apresentado um formulário (Figura 4.11) que possibilita a caracterização de um projeto de software Web.

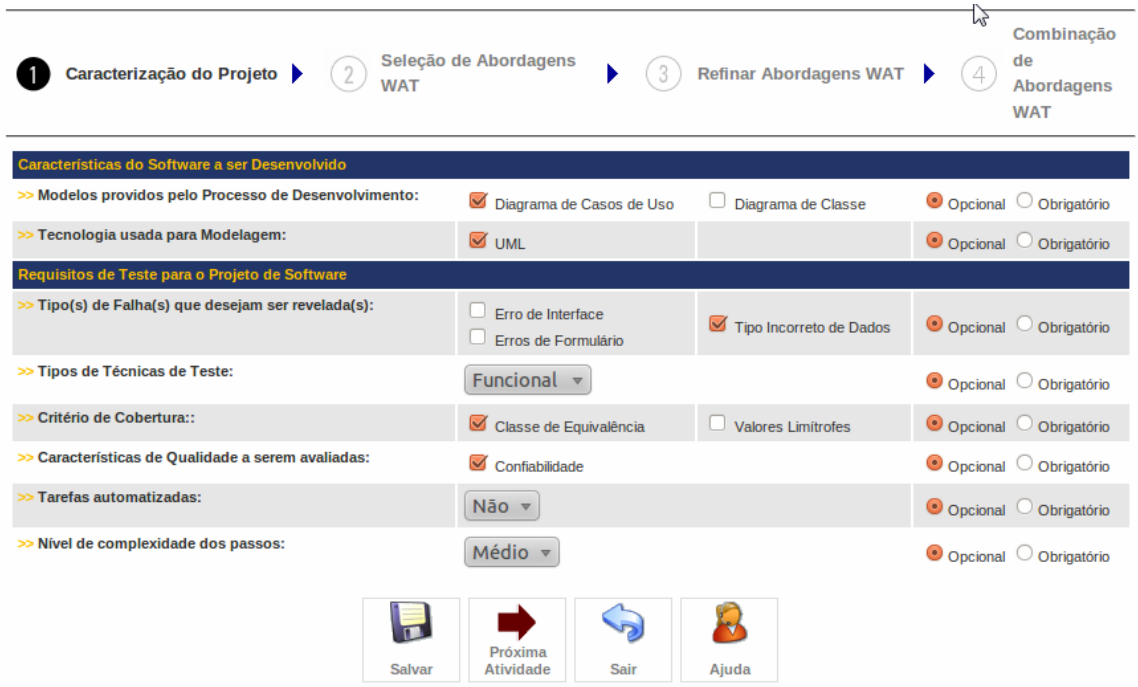


Figura 4.11. Formulário de caracterização de projeto software Web.

Após o cálculo, a ferramenta apresenta uma lista que informa as abordagens WAT selecionadas para o projeto informado, exibidas em ordem decrescente por grau de adequação.

Nesse momento, é possível selecionar as abordagens que serão utilizadas no projeto, acionando o botão “Próxima atividade”, ou optar por um refinamento na seleção, clicando no botão “Refinar” (Figura 4.12).

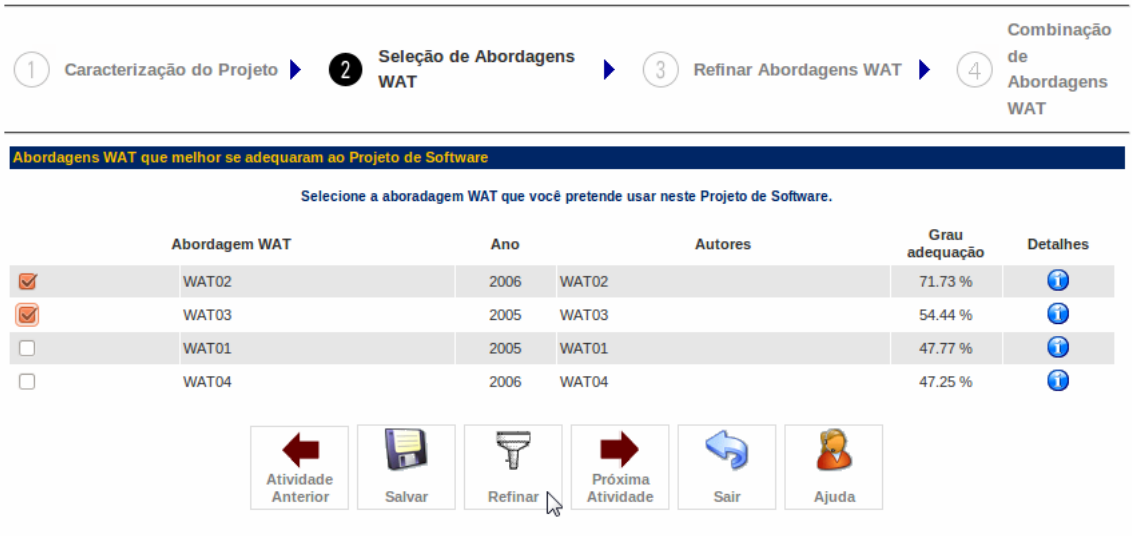


Figura 4.12. Abordagens WAT selecionadas para refinamento da seleção.

Ao optar pelo refinamento, são requeridas informações adicionais para a caracterização do projeto, conforme ilustrado na Figura 4.13. Desta forma, o cálculo do grau de adequação é realizado novamente considerando as novas características informadas (Figura 4.14).

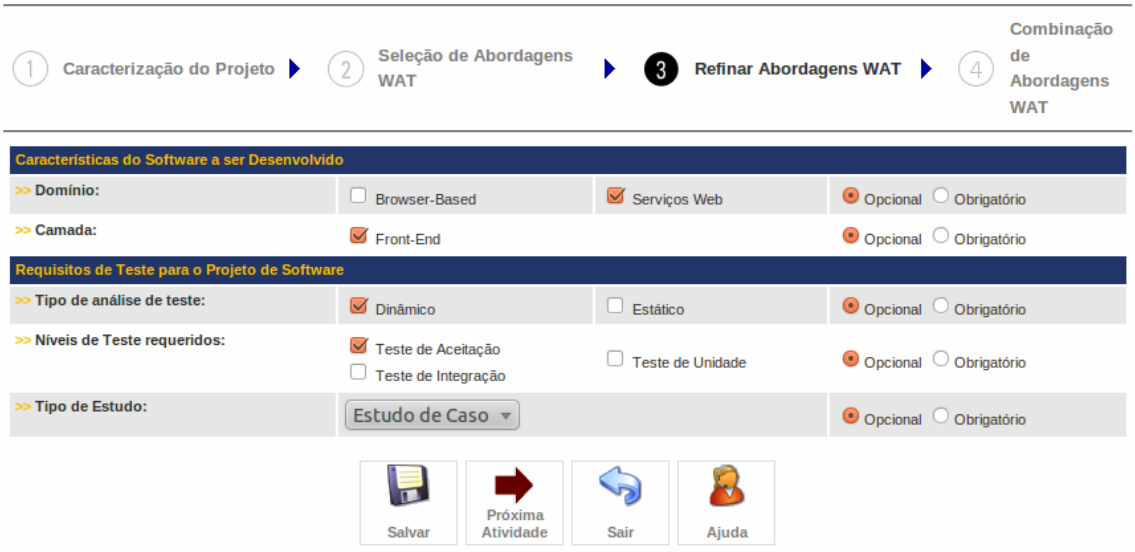


Figura 4.13. Refinamento da seleção – Caracterização adicional do projeto de teste.

Figura 4.14. Novo grau de adequação entre as abordagens.

Ao final é apresentada a lista de abordagens WAT selecionadas para o projeto (Figura 4.15), indicando o grau de adequação, cujo cálculo considera as novas características informadas para o projeto. Além disso, também é apresentada a análise do uso combinado dessas abordagens, onde os requisitos de teste para o projeto de software são confrontados com as características das abordagens WAT selecionadas.

Figura 4.15. Resultado do Refinamento e Análise da combinação das abordagens WAT.

4.4.3 Exemplo

Nesta seção, o procedimento Porantim-WAT será aplicado em um projeto de software Web real, apoiado pela utilização da ferramenta Maraká. O objetivo deste exemplo é avaliar as funcionalidades providas pela infra-estrutura computacional de apoio ao procedimento Porantim-WAT, analisando o comportamento e viabilidade da infra-estrutura adaptada.

Para execução deste exemplo foi selecionado o projeto SIGIC - Sistema de Gestão de Informações (SIGIC) da Fundação COPPETEC. O conjunto de abordagens de teste para aplicações Web utilizado será composto pelas 101 abordagens WAT selecionadas na quasi-revisão sistemática, descrita na Seção 2.7

4.4.3.1 Caracterização do Projeto

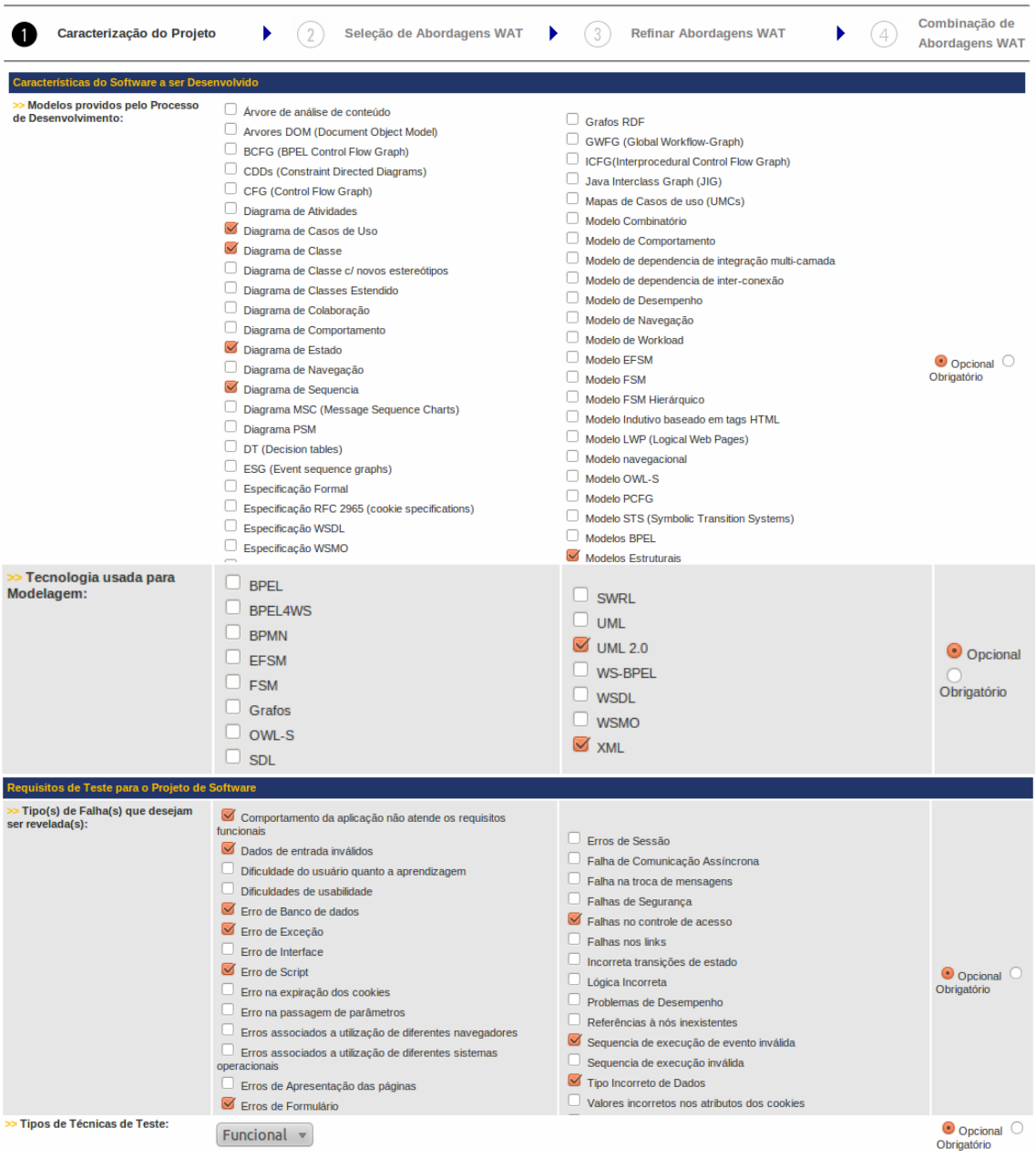
O projeto SIGIC consiste no desenvolvimento e manutenção do Sistema de Gestão de Informações (SIGIC) da Fundação COPPETEC. Trata-se de uma aplicação Web que faz a gestão das solicitações de desembolsos e relatórios de acompanhamento financeiro nos âmbitos dos projetos da fundação. A Tabela 4.9 apresenta a caracterização do projeto, de acordo com os atributos propostos neste trabalho (detalhados na seção 4.3.1), realizada por um engenheiro de software que integra a equipe responsável pela manutenção do sistema.

Tabela 4.9. Caracterização do Projeto SIGIC.

Atributos de Caracterização	**Projeto**
1. Modelos	Diagrama de Casos de Uso Diagrama de Classe Diagrama de Estado Diagrama de Seqüência Modelos Estruturais
2. Tecnologia de modelagem	UML2.0 XML
3. Habilidades	*Conhecimento em UML*
4. Tipos de Falhas	*Comportamento da aplicação não atende os requisitos funcionais* *Dados de entrada inválidos* *Erro de Banco de dados* *Erro de Exceção* *Erro de execução de Script* *Erros de Formulário* *Erros de Interface* *Erros de Navegação* *Falhas no controle de acesso* *Seqüência de execução inválida*
5. Técnica de Teste	*Funcional*
6. Critério de Cobertura	*Classes de equivalência* *Combinação de diferentes sessões de usuário* *Fluxo de Controle* *Fluxo de Dados* *Fluxos de Exceção*

7. Características de qualidade	*Usabilidade* *Funcionalidade*
8. Tarefas automatizadas	*Sim*
9. Complexidades Passo não automatizados	*Médio*
Atributos de Caracterização – Critério Desempate	
10. Categoria da Aplicação	*Browser-Based* *Workflow*
11. Nível de Teste	*Sistema*
12. Análise de Teste	*Dinâmico, Estático*
13.Tipo de Avaliação	*Prova de Conceito*
14. Camada da Aplicação	*Front-End*

As informações que caracterizam o projeto SIGIC foram submetidas na ferramenta Maraká, conforme pode ser observado na Figura 4.16.



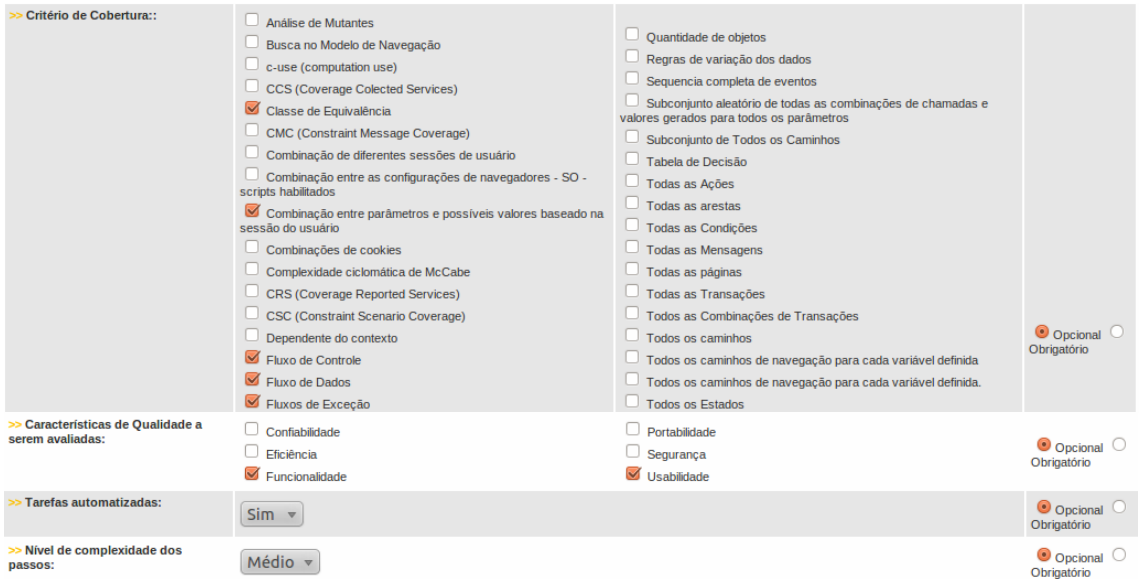


Figura 4.16. Caracterização inicial do projeto SIGIC

4.4.3.2 Aplicação Porantim-WAT

Após a caracterização do projeto SIGIC, Maraká irá calcular automaticamente o grau de adequação entre o projeto de software Web, caracterizado na etapa anterior, para cada uma das 101 abordagens WAT descritas no corpo de conhecimento, utilizando as fórmulas providas por Porantim-WAT.

As abordagens mais adequadas ao projeto SIGIC e o respectivo grau de adequação são listados pela ferramenta, conforme mostrado na Figura 4.17.

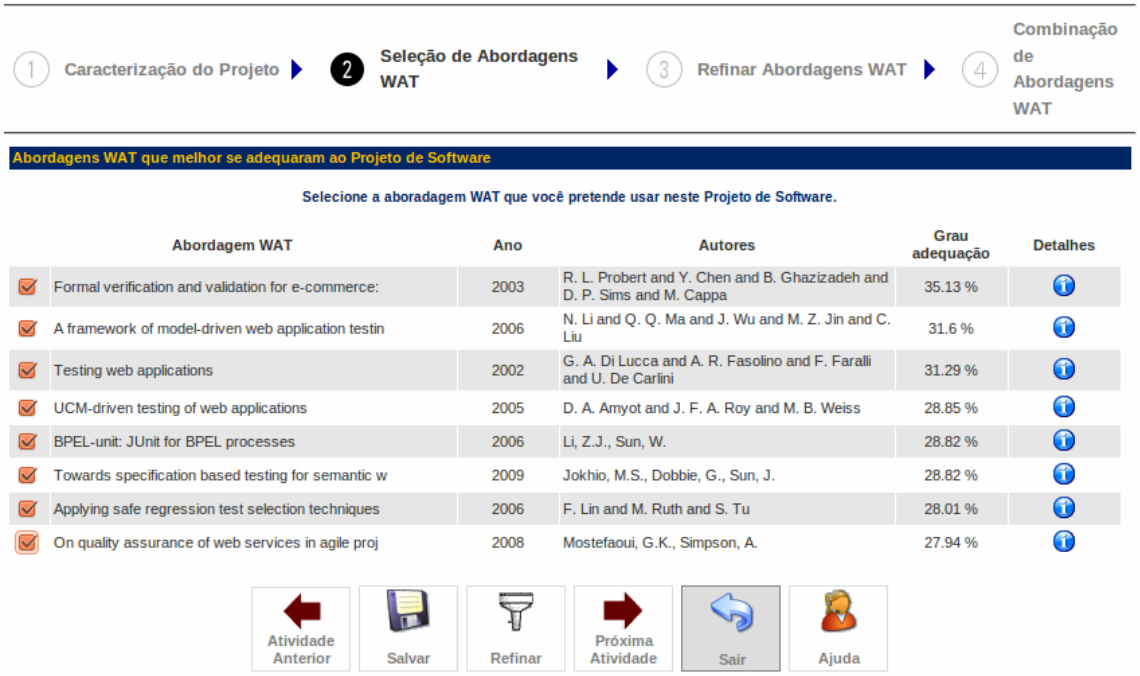


Figura 4.17. Abordagens WAT mais adequadas ao projeto SIGIC.

Devido ao grau de adequação das diferentes abordagens retornadas apresentarem valores muito próximos, optou-se pelo refinamento do procedimento de seleção. Desta forma, informações adicionais para a caracterização do projeto foram requeridas e preenchidas, conforme ilustrado na Figura 4.18. Após novo cálculo do grau de adequação (Figura 4.19), foram selecionadas duas abordagens que juntas apresentam maior grau de cobertura ao projeto SIGIC (Figura 4.20.).

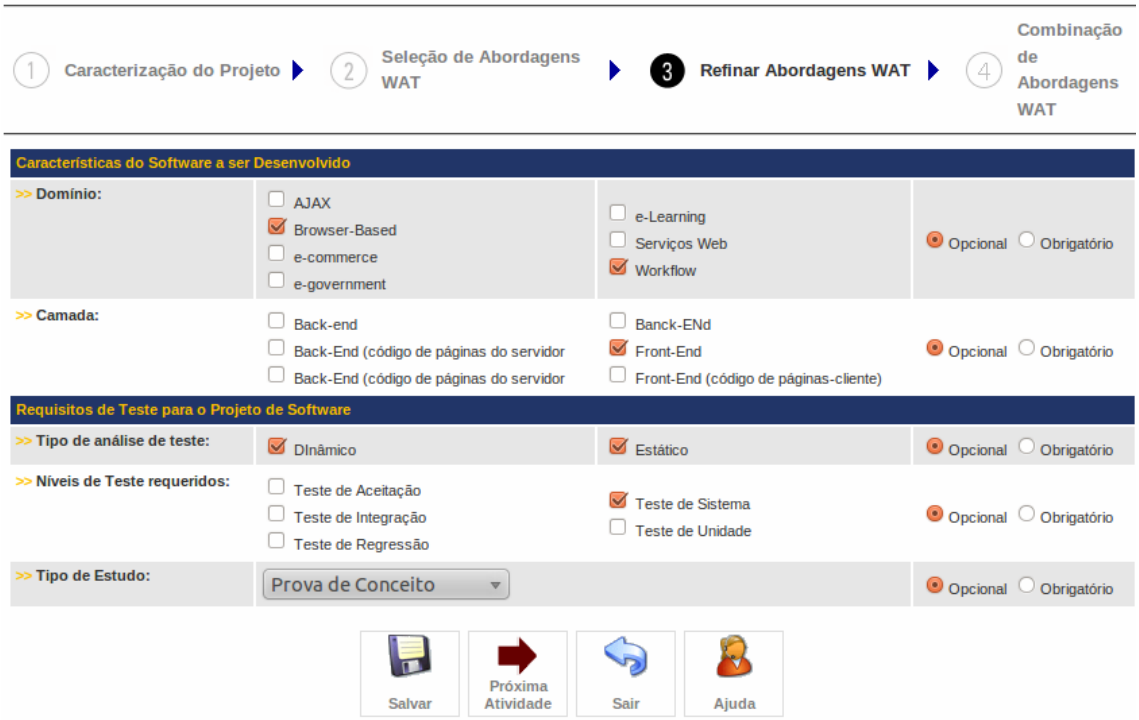


Figura 4.18. Caracterização adicional do projeto SIGIC.

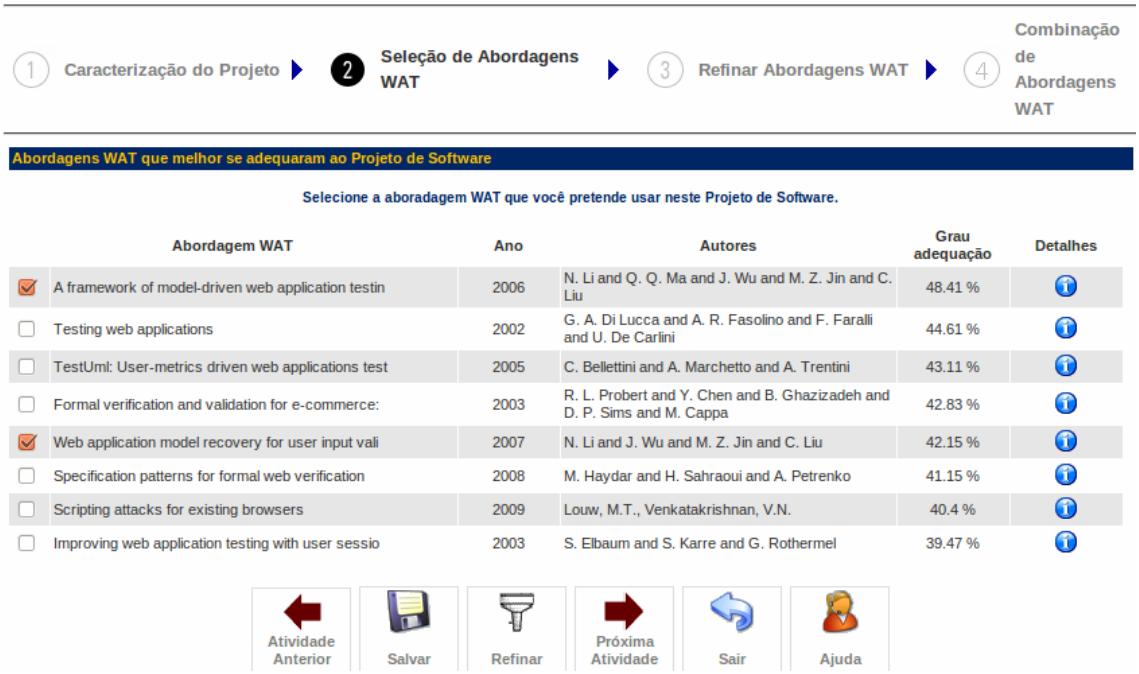


Figura 4.19. Novo grau de adequação entre as abordagens WAT e o projeto SIGIC.

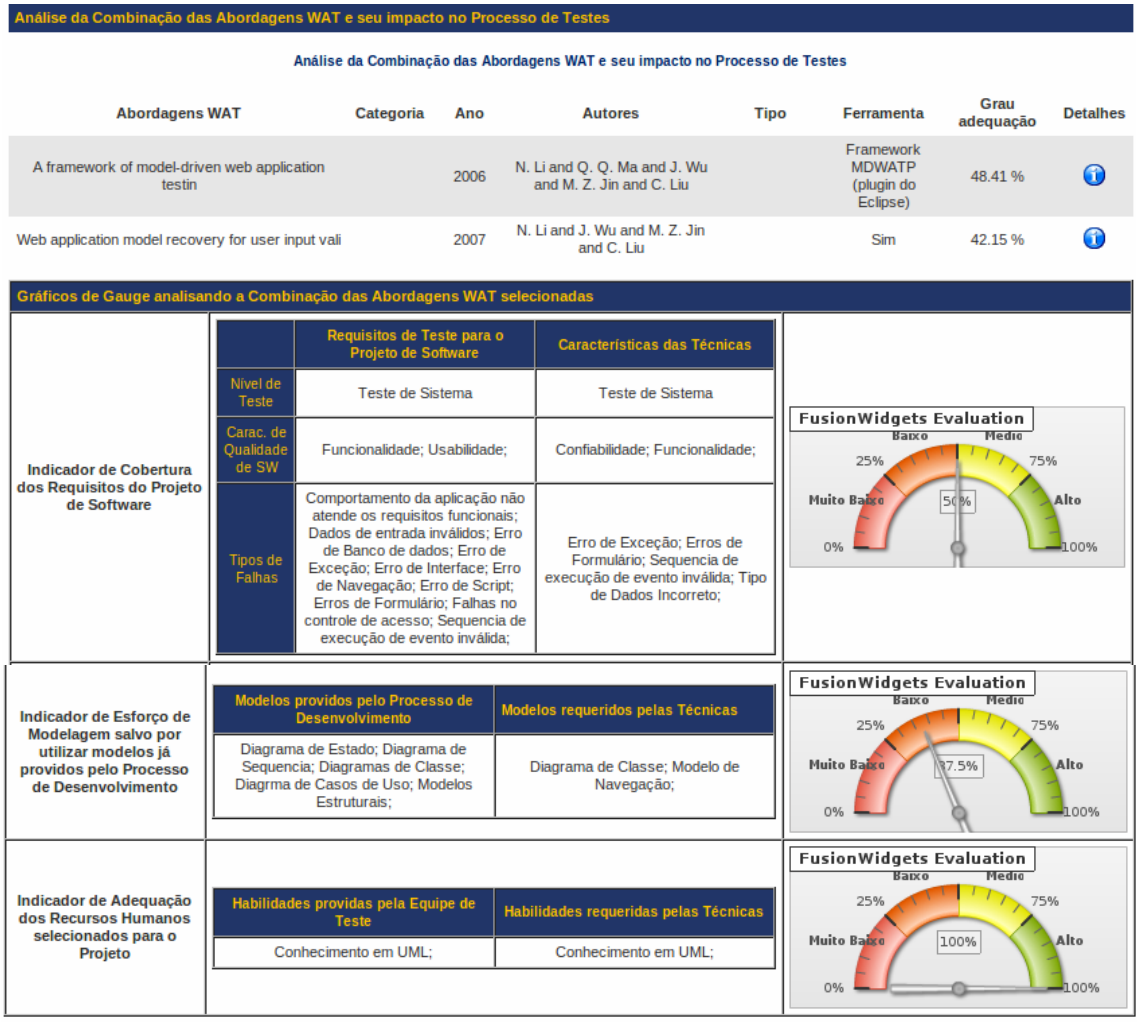


Figura 4.20. Resultado do procedimento de seleção para o projeto SIGIC.

4.5 Considerações Finais do Capítulo

Neste capítulo, foi definido o procedimento Porantim-WAT para apoiar à seleção de abordagens WAT para projetos de software Web. Este procedimento representa uma evolução de Porantim, proposto por DIAS NETO (2009).

Além disso, foi apresentada também a extensão da infra-estrutura computacional Maraká através da criação de um repositório de abordagens WAT e do componente Porantim-WAT, para aplicação do procedimento de apoio à seleção de abordagens de teste para projetos de software Web, proposto neste trabalho.

A utilização dessa infra-estrutura foi demonstrada a partir da sua aplicação em um projeto de software Web denominado SIGIC, a fim de observar a viabilidade de seu uso no apoio à seleção de técnicas de teste.

O próximo capítulo apresenta as considerações finais deste trabalho, descrevendo suas conclusões, resultados obtidos, limitações e futuras linhas de pesquisas a serem seguidas para continuidade desta pesquisa.

CAPÍTULO 5 - CONCLUSÃO

Neste capítulo são apresentadas as considerações finais deste trabalho (seção 5.1), bem como as contribuições da dissertação (seção 5.2), suas limitações (seção 5.3) e perspectivas futuras (seção 5.4).

5.1 Considerações Finais

A evolução tecnológica das aplicações Web aumentou a demanda por técnicas e ferramentas que abordam o problema da garantia de qualidade dessas aplicações. A partir da execução de revisões na literatura técnica, foi identificada a disponibilidade de diferentes estratégias de teste para estas aplicações. A existência de diferentes métodos e tipos de aplicações que envolvem a Web aumenta a dificuldade da seleção de técnicas de teste para um projeto de software Web.

Este trabalho apresentou os resultados de uma pesquisa sobre aplicações Web e definiu um procedimento de seleção de abordagens de teste voltadas para este tipo de aplicação, além da construção de uma infra-estrutura computacional elaborada para apoiar à execução do procedimento proposto em organizações de desenvolvimento de software Web.

Para isso, foi organizado um corpo de conhecimento inicial que contempla a caracterização de 101 abordagens de teste para aplicações Web, a partir dos resultados obtidos em uma quasi-revisão sistemática da literatura técnica que classificou as abordagens quanto à utilização de apoio ferramental, categorias de aplicações cuja abordagem é aplicável, característica de qualidade avaliada, nível de abstração de teste utilizado, tipo de análise realizada, tipo de estudo utilizado para avaliar a abordagem e quanto à inserção em alguma metodologia de desenvolvimento de aplicações Web conhecidas e identificadas por trabalhos anteriores na área.

A motivação para organizar o corpo de conhecimento se apoiou na premissa de que a existência de tal estrutura pode prover informações para apoiar a tomada de decisão a respeito da seleção de abordagens de teste para projetos de software Web.
A estrutura de caracterização proposta foi avaliada através de um survey realizado com pesquisadores da área e autores das abordagens selecionadas na quasi-revisão sistemática realizada.

Após a definição da estrutura e da construção do corpo de conhecimento, foi apresentado o procedimento Porantim-WAT. Trata-se de um procedimento para apoiar a seleção de técnicas de teste para projetos de software Web, que consulta o repositório de conhecimento com as informações que caracterizam as abordagens de teste para aplicações Web (WAT) e utiliza um mecanismo que avalia o grau de adequação entre as abordagens e as características de um projeto de software Web. Ao final, foi apresentada a adaptação de uma infra-estrutura computacional para apoiar o procedimento de seleção proposto, cuja utilização foi ilustrada por um exemplo.

5.2 Contribuições

As principais contribuições desta dissertação são:

- **Definição de um protocolo de pesquisa para identificar e caracterizar abordagens de teste para aplicações Web;**

  Foi realizado um estudo na área de teste de aplicações Web visando identificar as abordagens propostas e suas principais características. Isto implicou na definição de um protocolo para realização de uma quasi-revisão sistemática. Este protocolo pode ser utilizado/estendido em novas pesquisas sobre este tema.

- **Corpo de conhecimento de abordagens WAT;**

  Foi construído um corpo de conhecimento contendo a caracterização de 101 abordagens WAT identificadas na literatura técnica a partir de uma quasi-revisão sistemática. A estrutura do corpo de conhecimento foi avaliada por pesquisadores da área através de um survey (pesquisa de opinião), cujo objetivo era identificar o conjunto de atributos que caracterizam as abordagens WAT e qual a relevância de cada atributo para a seleção de uma abordagem em um projeto de software Web.

  Foi criado um banco de dados na infra-estrutura Maraká para armazenar o corpo de conhecimento de abordagens de teste para aplicações Web. Além dessa estrutura, o corpo de conhecimento também está armazenado em um banco de dados da ferramenta JabRef. Ambas estruturas foram disponibilizadas como resultado desta pesquisa.

- **Definição do procedimento Porantim-WAT.**

Foi proposto um procedimento para apoiar a seleção de abordagens de teste para projetos de software Web, denominado Porantim-WAT. Trata-se de uma instância especializada do procedimento Porantim proposto por DIAS NETO (2009) que apóia a seleção de técnicas de teste baseado em modelos.

O procedimento consiste em capturar as informações sobre o projeto de software Web no qual as abordagens WAT devem ser aplicadas. A partir das informações do projeto, o procedimento consulta o corpo de conhecimento das abordagens WAT e indica aquelas mais adequadas ao projeto.

- **Adaptação da Infra-estrutura computacional Maraká para apoiar o uso de Porantim-WAT;**

Foi apresentada a extensão da infra-estrutura computacional Maraká para apoiar nas atividades que compõem o mecanismo de seleção de abordagens teste para projetos Web. Foi construído um banco de dados contendo as abordagens de teste para aplicações Web e um componente que automatiza o procedimento Porantim-WAT.

A utilização dessa infra-estrutura foi demonstrada a partir da sua aplicação em um projeto de software Web denominado SIGIC, a fim de observar a viabilidade de seu uso no apoio à seleção de técnicas de teste.

- **Glossário bilíngüe de termos referentes às tecnologias empregadas nas aplicações Web.**

Ao longo da realização da revisão da literatura e da quasi-revisão sistemática para identificar e caracterizar as abordagens de testes para aplicações Web, foi identificada uma grande variedade de tecnologias empregadas nesta área de pesquisa. Foi criado então um glossário bilíngüe com o objetivo de reunir, de forma breve e objetiva, os significados dos variados termos, expressões e palavras utilizadas para referenciar as tecnologias empregadas nas aplicações Web. Trata-se de uma coletânea de expressões com a respectiva explicação de conceitos dos termos encontrados descritos em inglês e em português.

5.3 Limitações

Diversos aspectos podem influenciar na seleção de uma tecnologia para um projeto de software, tais como cronograma do projeto, esforço, custo e qualificação da

equipe. No entanto, a proposta desta dissertação foca apenas em aspectos técnicos para caracterizar abordagens WAT e avaliar a sua adequação a um projeto de software Web, sem considerar os outros aspectos, que também possuem relevância para apoiar a tomada de decisão sobre a seleção da abordagem mais adequada para determinado projeto.

Além disso, as abordagens de teste para aplicações Web, selecionadas na quasi-Revisão Sistemática, foram classificadas de acordo com os elementos apresentados nos artigos, que eventualmente podem não explicitar todas as características da abordagem proposta.

Outra limitação a ser destacada nesse trabalho está associada ao fato de que o survey que avaliou a estrutura do corpo de conhecimento para caracterizar as abordagens WAT foi submetido a membros da comunidade acadêmica, não tendo sido observado a opinião de profissionais da indústria sobre o tema.

Também não foi possível executar estudos experimentais para avaliar a viabilidade do procedimento proposto bem como da ferramenta de apoio, muito embora o procedimento e a ferramenta tenham sido avaliados em sua origem. Entretanto, foi apresentado um exemplo (prova de conceito) da utilização do procedimento proposto, a fim de observar a viabilidade de seu uso no apoio à seleção de técnicas de teste.

5.4 Perspectivas Futuras

A realização desse trabalho de pesquisa levou ao desenvolvimento de um procedimento de seleção de abordagens de teste voltada para aplicações Web, além da construção de uma infra-estrutura computacional elaborada para apoiar à execução do procedimento proposto em organizações de desenvolvimento de software Web.

O corpo de conhecimento, construído a partir abordagens de teste identificadas por uma quasi-Revisão Sistemática e avaliado por pesquisadores através de um survey, é o elemento que apóia o processo de seleção proposto. Nesse sentido, é importante evoluir o protocolo da revisão realizada, considerando a utilização de outras fontes de pesquisa, com o objetivo de manter a atualizar o corpo de conhecimento e os índices de relevância dos seus atributos. Além disso, uma importante contribuição a ser realizada em trabalhos futuros é realizar a avaliação da estrutura do corpo de conhecimento pelos profissionais da indústria.

Para a elaboração da proposta, esta pesquisa se baseou em uma metodologia baseada em evidências (SHULL et al. 2001; MAFRA et al. 2006; SPÍNOLA 2008). A metodologia é composta por duas etapas: concepção e avaliação da abordagem proposta. Para a realização deste trabalho foram executadas as atividades da etapa

de concepção, na qual SPÍNOLA et al. (2008) descreve a forma como os estudos secundários devem ser executados, e inclui a realização de estudos primários para avaliar o conhecimento obtido através dos estudos secundários.

Apesar da utilização da infra-estrutura ter sido demonstrada a partir da sua aplicação em um projeto de software Web denominado SIGIC, a fim de observar a viabilidade de seu uso no apoio à seleção de técnicas de teste, não foi possível executar estudos experimentais para avaliar a viabilidade do procedimento proposto bem como da ferramenta de apoio.

Nesse sentido, é interessante a realização de trabalhos futuros para conduzir estudos experimentais em diferentes contextos para avaliar o procedimento proposto até que seja considerado maduro o suficiente para ser utilizado na indústria.

REFERÊNCIAS BIBLIOGRÁFICAS

ANDREWS, A. A., OFFUTT, J., ALEXANDER, R.T. (2005) Testing Web applications by modeling with FSMs, Software and Systems Modeling Journal, Vol. 4, p. 326-345.

ARAÚJO, M. A. P., TRAVASSOS, G. H. (2008) A System Dynamics Model based on Cause and Effect Diagram to Observe Object-Oriented Software Decay. Relatório Técnico, COPPE/UFRJ.

ARTZ, S. , KIEZUN, A. , DOLBY, J. , TIP, F. , DIG, D. , PARADKAR, A. , ERNSF, M.D. (2008) “Finding bugs in dynamic web applications”, Conference of 2008 International Symposium on Software Testing and Analysis, ISSTA.

BEIZER, B. (1995) “Black-Box Testing: Techniques for Functional Testing of Software and Systems”, John Wiley & Sons.

BARESI, L., FRATERNALI,P., TISI, M., MORASCA, S. (2005)“Towards model-driven testing of a Web application generator”, International Conference on Web Engineering, ICWE, vol 3579, p. 75-86.

BARESI, L., COLAZZO, S. , MAINETTI, L., MORASCA, S., (2006) “W2000: A Modeling Notation for Complex Web Applications”. In E. Mendes and N. Mosley (eds.) Web Engineering: Theory and Practice of Metrics and Measurement for Web Development.

CACHERO, C., GÓMEZ, J., PASTOR, O., (2001) “Conceptual Modeling of Device-Independent WebApplications”, IEEE Multimedia, (Abr-Jun), p. 26 – 39.

CERI, S., FRATERNALI, P., BONGIO, A. (2000) “Web Modeling Language (WebML): a Modeling Language for Designing Web Sites”. WWW9 Conference, Amsterdam.

CHO, Y., LEE, W., CHONG, K. (2005) “The technique of test case design based on the UML sequence diagram for the development of Web applications”, International Conference on Computational Science and Applications, ICCSA, LNCS 3480, p. 1–10.

CONALLEN., J. (2002) “Building Web Applications with UML”, Addison-Wesley Publishing Company.

CONTE, T. U., MENDES, E., TRAVASSOS, G.H. (2005) “Revisão Sistemática sobre Processos de Desenvolvimento para Aplicações Web”, Relatório Técnico, PESC/COPPE/UFRJ.

CONTE, T. U. (2009) “Técnica de Inspeção de Usabilidade Baseada em Perspectivas de Projeto Web”, Tese de Doutorado, COPPE/UFRJ.

DELAMARO, M. E., MALDONADO, J. C., JINO, M. (2007) “Introdução ao Teste de Software”, Elsevier, Rio de Janeiro.

DESHPANDE, Y.; HANSEN, S. (2001) “Web engineering: creating a discipline among disciplines”. Multimedia, IEEE. Volume 8, Issue 2, p. 82 – 87.

DI LUCCA, G. A., FASOLINO, A., FARALLI, F., DE CARLINI, U. (2002) “Testing Web applications”, International Conference on Software Maintenance (ICSM’02), Montreal, Canada.

DI LUCCA, G.A., FASOLINO, A.R. (2006) “Testing Web Applications: The state of the art and future trends”, J. Inf. Softw. Technol. 48(2), p. 1172–1186.

DIAS NETO, A. C., NATALI, A. C. C., ROCHA, A. R. C., TRAVASSOS, G. H. (2006) “Caracterização do Estado da Prática das Atividades de Teste em um Cenário de Desenvolvimento de Software Brasileiro” Simpósio Brasileiro de Qualidade de Software, 2006, Vila Velha. Simpósio Brasileiro de Qualidade de Software.

DIAS NETO, A. C. (2006) “Uma Infra-estrutura Computacional para Apoiar o Planejamento e Controle de Testes de Software”. Dissertação de M.Sc., COPPE/UFRJ.

DIAS NETO, A.C., TRAVASSOS, G.H. (2006) “Maraká: Apoio ao Planejamento e Controle de Testes de Software” XIII Sessão de Ferramentas (20º Simpósio Brasileiro de Engenharia de Software), Florianópolis, SC.

DIAS NETO, A. C. (2009) “Porantim: Uma Abordagem para Apoiar a Seleção Combinada de Técnicas de Teste Baseado em Modelos em Projetos de Software”, Tese de Doutorado, COPPE/UFRJ, 2009.

DIAS NETO, A. C., SPINOLA, R. O., TRAVASSOS, G. H. (2010) “Developing Software Technologies through Experimentation: Experiences from the Battlefield”. CIBSE – Conferencia Ibero Americana de Software Engineering.

DIAZ, P., MONTERO, S., AEDO, I. (2004) “Modelling hypermedia and Web applications: the Ariadne Development Method”, Information Systems.

EATON, C., MEMON, A.M. (2007) “An empirical approach to evaluating web application compliance across diverse client platform configurations”, International Journal of Web Engineering and Technology, Vol. 3, p. 227-253.

EBSE (2007) “Guidelines for performing Systematic Literature Reviews in Software Engineering”, EBSE Technical Report EBSE-2007-001, Available at http://www.dur.ac.uk/ebse/resources/guidelines/Systematic-reviews-5-8.pdf. Last access in 12/08/2009.

GARZOTTO, F., PAOLINI, P., SCHWABE, D. (1993) “HDM – A Model-based Approach to Hypertext Application Design”, ACM Transactions on Information Systems, vol.11, nº. 1, p.1-26.

GINIGE, A., MURUGESAN, S. (2001) “Web Engineering: an Introduction, IEEE Multimedia”, Vol. 8, Issue: 1, p. 14 -18.

HAMBURG, M. (1980) “Basic Statistics: A Modern Approach, Journal of the Royal Statistical Society”, Series A (General), Vol. 143, No. 1, 2a edição.

ISAKOWITZ, T., STOHR, E., BALASUBRAMANIAN, P. (1995) “RMM: a methodology for structured hypermedia design”, Communications ACM, vol. 38, no. 8, p. 34 – 44.

ISO/IEC (2002) “ISO-9126-1: Software engineering – Product Quality – Part 1 Quality model”, The International Organization for Standardization and the International Electrotechnical Commission.

MAFRA, S.N., TRAVASSOS, G.H. (2006) “Estudos Primários e Secundários apoiando a busca por evidências em Engenharia de Software”, Relatório Técnico PESC 687/06 - COPPE/UFRJ.

KAPPEL, G. (2004) “Web Engineering - Old Wine in New Bottles?”. Invited Talk, Fourth International Conference on Web Engineering (ICWE’04), disponível emhttp://www.icwe2004.org/download/ICWE_Kappel.pdf , Munique.

KITCHENHAM, B. (2004) “Procedures for Performing Systematic Reviews”, Keele technical report SE0401 and NICTA technical report 0400011T.1.

KITCHENHAM, B. , PEARL BRERETON, O. , BUDGEN, D. , TURNER, M. , BAILEY, J. , LINKMAN, S. (2009) "Systematic literature reviews in software engineering - A systematic literature review", Information and Software Technology, Volume 51, Issue 1, Pages 7-15.

KOCH, N., KRAUS, A. (2001) “The expressive Power of UML-based Web Engineering”, Second Int. Workshop on Web-oriented Software Technology (IWWOST´02).

MAFRA, S.N., BARCELOS, R.F., TRAVASSOS, G.H. (2006) “Aplicando uma Metodologia Baseada em Evidência na Definição de Novas Tecnologias de Software”, XX SBES, pp. 239-254.

MANSOUR, N., HOURI, M. (2007) “Testing Web applications”, Information and Software Technology 48 (1), pp. 31-42.

MASSOLLAR, J. (2008) “Uma abordagem baseada na engenharia para desenvolvimento e garantia a qualidade de Aplicações Web”, Exame de Qualificação, COPPE/UFRJ.

MASSOLLAR, J. (2011) “Uma abordagem para especificação de requisitos dirigida por modelos integrada ao controle de qualidade de Aplicações Web”, Tese de Doutorado, COPPE/UFRJ.

MENDES, E., MOSLEY, N., COUNSELL, S. (2006) “The Need for Web Engineering: An Introduction”. In: Mendes, E., Mosley, N. (eds), Web Engineering, Chapter 1, New York, Spinger Verlag.

MUSTAFA, G., SHAH, A. A., ASIF, K. H., ALI, A. (2007) “A strategy for testing of Web based software”, Information Technology Journal 6 (1), p. 74-81, 2007.

NBR ISO/IEC 9126-1 (2003) “Engenharia de software - Qualidade de produto - Parte 1: Modelo de qualidade”, ABNT - Associação Brasileira de Normas Técnicas.

OFFUT, A. J. (2002) “Quality atributes of Web software applications”, IEEE Software: Special Issue on Software Engineering of Internet Software, 19(2):25-32.

OFFUTT, J., WU, Y., DU, X., HUANG, H. (2004) “Bypass testing of Web applications” ISSRE 2004, p. 187-197.

PAI, M. (2004) “Systematic reviews and meta-analyses: an illustrated, step-by-step guide” The National Medical Journal of India, vol. 17, no. 2, 2004.

PASTOR, O., GÓMEZ, J., INSFRÁN, E., PELECHANO, V. (2001) “The OO-Method Approach for Information Systems Modeling: From Object-Oriented Conceptual Modeling to Automated Programming”, Information Systems, Elsevier Science, vol. 26, p. 507-534, no. 7.

PRESSMAN, R.S. (2000), “What a Tangled Web We Weave,” IEEE Software 17, 1, 18–21.

QI, Y., KUNG, D., WONG, E. (2005) “An agent-based testing approach for Web applications”, Proceedings - International Computer Software and Applications Conference 2, art. no. 1508082, p. 45-50.

REZA, H., OGAARD, K., MALGE, A. (2008) A model based testing technique to test Web applications using StateCharts. In: Conference of International Conference on Information Technology: New Generations, ITNG 2008, p. 183-188.

RICCA, F., TONELLA, P. (2001) “Analysis and testing of Web applications”, Proceedings of the 23rd International Conference on Software Engineering.

RICCA, F., TONELLA, P. (2002), “Testing processes of web applications”, Annals of Software Engineering (14:1-4),pp. 93-114.

RICCA, F., TONELLA, P. (2005) “Anomaly detection in Web applications: a review ofalready conducted case studies”, Ninth European Conference on Software Maintanence and Reegineering.

SANTA ISABEL, S. L.; TRAVASSOS, G. H. (2010) “Abordagens de Teste para Projetos de Software Web”. Relatório Técnico, COPPE/UFRJ (2010). Disponível em http://lens-ese.cos.ufrj.br.

SANTA ISABEL, S. L., TRAVASSOS, G. H. (2011) “Características de Técnicas de Teste de Software para Uso em Projetos WEB”. CIBSE – Conferencia Ibero Americana de Software Engineering.

SCHWABE, D., ROSSI, G., BARBOSA, S. (1996) “Systematic Hypermedia Design with OOHDM”, ACM Conference on Hypertext.

SPÍNOLA, R.O., PINTO, F.C.D.R., TRAVASSOS, G.H. (2010) “UbiCheck: An approach to support requirements definition in the ubicomp domain”. SAC '10 Proceedings of the 2010 ACM Symposium on Applied Computing, pp. 306-310.

SPINOLA, R. O. ; DIAS-NETO, A. C.; TRAVASSOS, G. H. (2008) Abordagem para Desenvolver Tecnologia de Software com Apoio de Estudos Secundários e Primário. In: Experimental Software Engineering Latin American Workshop (ESELAW), Salvador, Novembro.

SHULL, F., CARVER, J., TRAVASSOS, G. (2001) “An Empirical Methodology for Introducing Software Processes”, No: Joint 8th ESEC and 9th ACM SIGSOFT FSE-9, pp. 288-296.

TARHINI, A., ISMAIL, Z., MANSOUR, N. (2008) “Regression testing Web applications”, International Conference on Advanced Computer Theory and Engineering, ICACTE, art. no. 4737088, p. 902-906.

TRAVASSOS, G. H., SANTOS, P. S. M., MIAN, P., DIAS NETO, A. C., BIOLCHINI, J. (2008) “An Environment to Support Large Scale Experimentation in Software Engineering”, In: IEEE International Conference on Engineering of Complex Computer Systems, Belfast. Proceedings of ICECCS p. 193-202.

VEGAS, S., BASILI, V. (2005) “A Characterization Schema for Software Testing Techniques”, Empirical Software Engineering, vol.10 no.4, p.437-466.

XU, L., XU, B. (2006) “Testing forms in Web applications automatically”. Wuhan University Journal of Natural Sciences, vol 11, p. 561-566.

YIN, R. K. (2003) “Case Study Research: Design and Methods (3rd edition)”, London: Sage.

WU, Y., OFFUTT, J. (2002) “Modeling and Testing Web-based Applications”, Technical Report ISE-TR-02-08, George Mason University, Fairfax, VA, USA. Disponível em http://cs.gmu.edu/~tr-admin/papers/ISE-TR-02-08.pdf.

WIKI – ESE (2011), “Glossário de Termos ESE”, Grupo de Engenharia de Software Experimental - COPPE/UFRJ. Disponível em http://lens-ese.cos.ufrj.br/wikiese/index.php/Experimental_Software_Engineering_-_Glossary_of_Terms

APÊNDICES

APÊNDICE A – LISTA DE ARTIGOS IDENTIFICADOS NA REVISÃO SISTEMÁTICA

Este apêndice apresenta a lista completa dos artigos identificados ao longo das duas execuções do protocolo da revisão descrito no item 2.7 deste trabalho.

Este apêndice apresenta a lista completa dos artigos identificados ao longo das execuções do protocolo da revisão realizadas neste trabalho. Foram realizadas duas execuções do protocolo: A primeira em Agosto/2009 e a segunda em Setembro/2011.

O procedimento de seleção foi dividido em duas etapas. A primeira etapa consiste na análise dos títulos e abstracts de todos os artigos retornados na pesquisa quanto aos critérios de inclusão e exclusão descritos no item 2.7.2.6. Após a avaliação, os artigos foram classificados como Candidato à Inclusão, Excluído ou Indefinido, sendo excluídos os trabalhos que apresentam qualquer característica descrita nos critérios de exclusão.

Na segunda etapa do procedimento de seleção, todos os artigos classificados como Candidato à Inclusão ou Indefinido foram analisados qualitativamente e novamente avaliados sob a perspectiva da presença dos critérios de inclusão ou ausência dos critérios de exclusão. Após esta análise eles foram classificados como Incluído, Excluído ou Indefinido.

O objetivo deste apêndice é identificar não apenas os artigos selecionados para escopo desta pesquisa (detalhados no item 2.7.3), mas também identificar aqueles que foram descartados, sendo possível verificar o critério que motivou a exclusão de cada publicação e a etapa da revisão na qual a publicação foi excluída. Os critérios de exclusão adotados foram:

[01] O artigo não apresenta abordagem de teste para aplicações Web

[02] O artigo apresenta apenas o apoio automatizado, sem se preocupar em descrever esta abordagem;

[03] O artigo apresenta apenas uma estratégia de geração automática de casos de teste, sem se preocupar em descrever esta abordagem;

[04] O artigo apresenta apenas uma estratégia de geração automática de dados de teste, sem se preocupar em descrever esta abordagem;

[05] O artigo apresenta uma abordagem depende de alguma forma de transformação dos modelos em outras representações, como redes de Petri, algoritmos, grafos, representação matemática ou cadeia de Markov.

[06] O artigo apresenta uma abordagem baseada em código de teste ou baseadas em agentes;

[07] Artigo duplicado;

[08] O artigo apresenta uma abordagem de teste para aplicações Web, entretanto não apresenta nenhum tipo de avaliação da abordagem proposta;

[09] Não é um artigo, e sim um "Proceedings".

[10] Artigo não está disponível pelo acesso à CAPES e não foi disponibilizado pelo autor.

Serão apresentados a seguir todos os artigos retornados nas duas execuções do protocolo da revisão (Agosto/2009 e Setembro/2011), identificando o ano da publicação (coluna “Year”), os autores (Coluna “Author”), e o título (Coluna “Title”). Além disso, é possível verificar o critério que motivou a exclusão de cada publicação e a etapa na qual a publicação foi excluída. A coluna “Decisão” descreve a classificação do trabalho após a conclusão da primeira etapa, que pode ser EXCLUÍDO, INDEFINIDO ou CANDIDATO A INCLUSÃO. O campo “Visto1” indica o critério de exclusão adotado na primeira etapa da revisão.

Apenas os artigos classificados como Candidato à Inclusão ou Indefinido foram submetidos à análise da segunda etapa da revisão. A coluna “Decisão2” descreve a classificação do trabalho após a conclusão da segunda etapa, que pode ser EXCLUÍDO, INDEFINIDO ou INCLUÍDO. O campo “Visto2” indica o critério de exclusão adotado na segunda etapa da revisão. Os campos “Decisão2” e “Visto2” não possuem valores para os trabalhos excluídos na primeira etapa da revisão.

1ª Execução (Agosto de 2009): Foram retornados 564 trabalhos

Year	Author	Title	Decisao	Visto1	Decisao2	Visto2
2008	Abu Talib	Assessment of real-time software specifications quality using COSMIC-FFP	EXCLUÍDO	[01]		
2006	Alava	Automatic validation of java page flows using model-based coverage criteria	INDEFINIDO		EXCLUÍDO	[05]
2008	Albunni	Using UML for modelling cross-cutting concerns in aspect oriented software engineering	EXCLUÍDO	[01]		
2008	Alshahwan	Automated session data repair for web application regression testing	CANDIDATO A INCL...		EXCLUÍDO	[05]
2008	Alvarez	Evalua-Test: A random test generator	EXCLUÍDO	[01]		
2007	Alwardt	Utilizing a Service-Oriented Architecture to perform closed-loop diagnostics in network centric support environ...	EXCLUÍDO	[01]		
2005	Amyot et al.	UCM-driven testing of web applications	INDEFINIDO.		INCLUÍDO	
2003	Anagnostopoulos	An artificial neural network approach for classifying e-Commerce Web pages	EXCLUÍDO	[01]		
2006	Anandhan et al.	Web usability testing - CARE methodology	INDEFINIDO		INCLUÍDO	
2005	Andersson	Multithreading and web applications: Further adventures in acceptance testing	EXCLUÍDO	[02]		
2005	Andreolini et al.	Design and testing of scalable Web-based systems with performance constraints	CANDIDATO A INCL...		INDEFINIDO	
2006	Andreou	Intelligent classification and retrieval of software components	EXCLUÍDO	[01]		
2005	Andrews et al.	Testing Web applications by modeling with FSMs	CANDIDATO A INCL...		INCLUÍDO	
1985	Anon	PROCEEDINGS - COMPSAC 85: THE IEEE COMPUTER SOCIETY'S NINTH INTERNATIONAL COMPUTER S...	EXCLUÍDO	[09]		
2004	Antoniol	Understanding web applications through dynamic analysis	EXCLUÍDO	[01]		
1995	Apfelbaum	Automated functional test generation	EXCLUÍDO	[01]		
2008	Arantes	Automatic test case generation through a collaborative web application	EXCLUÍDO	[03]		
2007	Araujo	A new method for measuring the activity of ABC transporters proteins in microgravity	EXCLUÍDO	[01]		
2007	AraÃ°jo	Web-based tool for automatic acceptance test execution and scripting for programmers and customers	EXCLUÍDO	[02]		
2005	Arendall III	A test of enhancing model accuracy in high-throughput crystallography	EXCLUÍDO	[01]		
2008	Artz	Finding bugs in dynamic web applications	CANDIDATO A INCL...		EXCLUÍDO	[05]
2004	Astrachan	Non-competitive programming contest problems as the basis for just-in-time teaching	EXCLUÍDO	[01]		
2004	Avritzer	Estimating the CPU utilization of a rule-based system	EXCLUÍDO.	[01]		
2004	Avritzer	The role of modeling in the performance testing of e-commerce applications	CANDIDATO A INCL...		EXCLUÍDO	[02]
2008	Ayhan	Implementing geospatially enabled aviation web services	EXCLUÍDO.	[01]		
2008	Babik	A testing framework for OWL-DL reasoning	EXCLUÍDO.	[01]		
2002	Bagnasco	XML data representation for testing automation	EXCLUÍDO.	[01]		
2007	Bai	Adaptive web services testing	INDEFINIDO		EXCLUÍDO	[08]
2007	Bai	Design of a trustworthy service broker and dependence-based progressive group testing	CANDIDATO A INCL...		EXCLUÍDO	[06]
2008	Bai	Ontology-based test modeling and partition testing of web services	EXCLUÍDO.	[01]		
2005	Bajwa	Evaluating current testing processes of Web-portal applications	CANDIDATO A INCL...		EXCLUÍDO	[10]
2008	Balasubramanian	Enhancing interactive web applications in hybrid networks	EXCLUÍDO.	[01]		
1990	Banisoleiman	Diesel engine fuel oil support system test programme	EXCLUÍDO.	[01]		
2003	Baolu	A New Testing Approach for the Whiteboard System	CANDIDATO A INCL...		EXCLUÍDO	[10]
2004	Barber	Creating effective load models for performance testing with incomplete empirical data	INDEFINIDO		INDEFINIDO	
2005	Baresi et al.	Towards model-driven testing of a Web application generator	CANDIDATO A INCL...		INDEFINIDO	
2008	Baresi	Towards a unified framework for the monitoring and recovery of BPEL processes	EXCLUÍDO.	[01]		
2008	Bartolini	Towards automated WSDL-based testing of web services	EXCLUÍDO.	[02]		
2007	Batra	Web usability and evaluation: Issues and concerns	INDEFINIDO		EXCLUÍDO	[01]
2001	Beckert	Trade exchanges set up shop in cyberspace	EXCLUÍDO.	[01]		
2005	Bellettini et al.	TestUml: User-metrics driven web applications testing	INDEFINIDO.		INCLUÍDO	
2008	Belli and Linschul...	Event-driven modeling and testing of web services	CANDIDATO A INCL...		INCLUÍDO	
2006	Benharref	Towards the testing of composed web services in 3rd generation networks	EXCLUÍDO.	[01]		
2007	Benharref	New approach for EFSM-based passive testing of web services	INDEFINIDO		EXCLUÍDO	[05]
2009	Benharref	Efficient traces' collection mechanisms for passive testing of Web Services	EXCLUÍDO.	[01]		
2006	Bernstein	Imprecise RDQL: Towards generic retrieval in ontologies using similarity joins	EXCLUÍDO.	[01]		
2006	BernÃjbe Loranca	Study for classification of quality attributes in argentinean E-commerce sites	EXCLUÍDO.	[01]		
2006	Bertolino et al.	Audition of web services for testing conformance to open specified protocols	CANDIDATO A INCL...		INCLUÍDO	
2007	Bertolino	Automatic generation of test-beds for pre-deployment QoS evaluation of Web services	EXCLUÍDO	[07]		
2007	Bertolino	Automatic test data generation for XML schema-based partition testing	EXCLUÍDO.	[04]		
2005	Bertolino	The audition framework for testing Web services interoperability	INDEFINIDO		EXCLUÍDO	[08]
2004	Bhasin	Analysis and prediction of affinity of TAP binding peptides using cascade SVM	EXCLUÍDO.	[01]		
2009	Bhattacharjee	Property evaluation in The Expert System for Thermodynamics ("TEST") web application	EXCLUÍDO.	[01]		
2009	Bitters	Spatial relationship networks: Network theory applied to GIS data	EXCLUÍDO.	[01]		
2006	BogÃjrdi-MÃ©szoly	Performance factors in ASP.NET web applications with limited queue models	EXCLUÍDO.	[07]		
2005	BogÃjrdi-MÃ©szÃ...	Investigating factors influencing the response time in ASP.NET web applications	EXCLUÍDO.	[07]		
2006	BogÃjrdi-MÃ©szÃ...	Using queueing model in predicting the response time of ASP.NET web applications	EXCLUÍDO.	[01]		
2004	Boldyreff	Web Site Evolution	EXCLUÍDO.	[01]		
2001	Bratt	The International Data Centre of the Comprehensive Nuclear-Test-Ban Treaty: Vision and progress	EXCLUÍDO.	[01]		
2007	Brenner	Strategies for the run-time testing of third party Web	INDEFINIDO		EXCLUÍDO	[08]
2003	Brown	End-User Testing for the Lyee Methodology using the Screen Transition Paradigm and WYSIWYT	EXCLUÍDO.	[01]		
1999	Brown	No free lunch: institutional preparations for computer-based patient records.	EXCLUÍDO.	[01]		
2007	Bryce	Biased covering arrays for progressive ranking and composition of Web Services	EXCLUÍDO.	[01]		
2003	Cai	Testable model specification via statechart schema	INDEFINIDO		EXCLUÍDO	[08]
2008	Cai	Extended hierarchical color petri net-based test case generation for composite services	EXCLUÍDO	[05]		
2008	Cai	The web application test based on page coverage criteria	CANDIDATO A INCL...		EXCLUÍDO	[10]
2005	Campbell	Testing concurrent object-oriented systems with spec explorer extended abstract	EXCLUÍDO	[02]		
2004	Campos	Towards a methodology for developing agent-based simulations: The MASim methodology	EXCLUÍDO.	[01]		
2004	Cane	Measuring performance of web applications: Empirical techniques and results	INDEFINIDO.		EXCLUÍDO	[10]
2005	Cao	Design of Web test system on MVC design pattern	INDEFINIDO		EXCLUÍDO	[10]
2008	Cao	Method for test case selection and execution of web application regression testing	INDEFINIDO		EXCLUÍDO	[10]
1995	Carlson	Monolithic glass block lasercom terminal: hardware proof of concept and test results	EXCLUÍDO.	[01]		
2004	Carr	Giving Credit to Secure Applications	EXCLUÍDO	[02]		
2007	Cavalli et al.	Regression and performance testing of an e-learning web application: DotIrn	INDEFINIDO		INCLUÍDO	

1ª Execução (Agosto de 2009): Foram retornados 564 trabalhos

Year	Author	Title	Decisao	Visto1	Decisao2	Visto2
2005	Cavalli	From UML models to automatic generated tests for the dotLRN e-learning platform	INDEFINIDO		EXCLUÍDO	[08]
2006	Chan	Restricted random testing: Adaptive random testing by exclusion	EXCLUÍDO.	[01]		
2007	Chan	A metamorphic testing approach for online testing of service-oriented software applications	CANDIDATO A INCL...		EXCLUÍDO	[10]
2007	Chandel	Sensei: Spoken language assessment for call center agents	EXCLUÍDO.	[01]		
1989	Chang	Testing integrated knowledge-based systems	EXCLUÍDO.	[01]		
1996	Chang	A performance evaluation of heuristics-based test case generation methods for software branch coverage	EXCLUÍDO.	[01]		
2006	Chang	Web service testing and usability for mobile learning	INDEFINIDO		INDEFINIDO	
2008	Chen	Nuclear threat detection via the nuclear web and dark web: Framework and preliminary study	EXCLUÍDO.	[01]		
2007	Chen	Automatic generating test cases for testing Web applications	EXCLUÍDO	[03]		
2006	Chen	BEAT: A web-based Boolean expression fault-based test case generation tool	EXCLUÍDO	[07]		
2007	Chen	Initial trust and online buyer behaviour	EXCLUÍDO.	[01]		
2009	Chen	Model-based method for web service performance testing	INDEFINIDO		EXCLUÍDO	[10]
2006	Chen	E-commerce system simulation for construction and demolition waste exchange	EXCLUÍDO.	[01]		
2004	Cheng	On the development of software tools for testing Web service	INDEFINIDO		EXCLUÍDO	[10]
2008	Chengying	Performing combinatorial testing on web service-based software	CANDIDATO A INCL...		INCLUÍDO	
2008	Chester	A system for dynamic server allocation in application server clusters	EXCLUÍDO.	[01]		
2003	Chi	Improving the evaluation of programming courses	EXCLUÍDO.	[01]		
2005	Cho et al.	The technique of test case design based on the UML sequence diagram for the development of web applicati...	INDEFINIDO		INCLUÍDO	
2005	Cho	The technique of business model driven analysis and test design for development of web applications	CANDIDATO A INCL...		EXCLUÍDO	[07]
2007	Chohan	LHC magnet tests: Operational techniques and empowerment for successful completion	EXCLUÍDO.	[01]		
2006	Choi	Automatic test approach of web application for security (AutoInspect)	EXCLUÍDO	[02]		
2008	Choo	ILC-CMAS model, summary of research findings and implication for Content Creation and Management Auto...	EXCLUÍDO.	[01]		
2001	Choukair	Auto-adaptive preliminary approach to compose multimedia telecom services	EXCLUÍDO.	[01]		
2008	Chukmol	A framework for web service discovery: Service's reuse, quality, evolution and user's data handling	EXCLUÍDO	[01]		
2008	Chunyan	WSDL-based automated test data generation for web service	EXCLUÍDO.	[04]		
2007	Conroy et al.	Automatic test generation from GUI applications for testing web services	INDEFINIDO		INCLUÍDO	
2006	Cooper	Data mining for correlations between diet and Crohn's disease activity.	EXCLUÍDO.	[01]		
2004	Costagliola	A Web based tool for assessment and self-assessment	EXCLUÍDO.	[01]		
2008	Costagliola	Logging and visualization of learner behaviour in Web-based E-testing	EXCLUÍDO.	[01]		
1991	Cross II	Expert system assisted test data generation for software branch coverage	EXCLUÍDO.	[01]		
1997	Cusick	Guiding reengineering with the operational profile	EXCLUÍDO.	[01]		
2007	Dai	Contract-based testing for web services	EXCLUÍDO	[03]		
2003	Dascalu	Specification of the Verity Learning Companion and Self-Assessment Tool	EXCLUÍDO.	[01]		
2006	Davis	The use of partnered usability testing to help to identify GAPS in online work flow	EXCLUÍDO.	[01]		
2008	Davis	Generating personalised cardiovascular risk management educational interventions linking SCORE and beh...	EXCLUÍDO.	[01]		
2005	Dawson	Testing Web applications	EXCLUÍDO	[02]		
2007	De Barros	Web services wind tunnel: On performance testing large-scale stateful web services	EXCLUÍDO.	[01]		
2006	De La Riva	A partition-based approach for XPath testing	EXCLUÍDO.	[01]		
2008	Della Penna	An XML based methodology to model and use scenarios in the software development process	EXCLUÍDO.	[01]		
2006	Deng	Research on model driven test modeling of EJB component	INDEFINIDO		EXCLUÍDO	[10]
2007	Deng	Progress in testing for web applications	INDEFINIDO		EXCLUÍDO	[10]
1997	Devanbu	The use of description logics in KBSE systems	INDEFINIDO		EXCLUÍDO	[01]
2004	Dhavachelvan	Reliability enhancement in software testing an agent-based approach for complex systems	INDEFINIDO.		EXCLUÍDO	[10]
2005	Dhavachelvan	Complexity measures for software systems: Towards multi-agent based software testing	EXCLUÍDO	[07]		
2006	Dhavachelvan	A new approach in development of distributed framework for automated software testing using agents	excluído	[06]		
2008	Din	A workload model for benchmarking BPEL engines	EXCLUÍDO.	[01]		
2007	Domenech	A tool for web usage mining	EXCLUÍDO.	[01]		
2008	Dong	Multi-agent test environment for BPEL-based web service composition	EXCLUÍDO	[06]		
2008	Dong	Test case reduction technique for BPEL-based testing	INDEFINIDO		EXCLUÍDO	[05]
2006	Dong	Testing BPEL-based web service composition using high-level petri nets	EXCLUÍDO	[05]		
2006	Dorsz	Automatic software validation process	EXCLUÍDO	[01]		
2008	Downey	Using SPI to achieve delivery objectives in e-commerce software development	INDEFINIDO		EXCLUÍDO	[10]
2006	Draheim et al.	Realistic load testing of web applications	INDEFINIDO		INCLUÍDO	
2006	Du	Automatically building service evaluation metadata in a grid environment	EXCLUÍDO.	[01]		
2007	Duan	Enhanced traverse of web pages	CANDIDATO A INCL...		EXCLUÍDO	[10]
2006	Duarte	GridUnit: Software testing on the grid	EXCLUÍDO	[01]		
2007	DÃ¤stner	An object oriented development suite for data fusion: Design, generation, simulation and testing	EXCLUÍDO.	[01]		
2007	Eaton and Memon	An empirical approach to evaluating web application compliance across diverse client platform configurations	CANDIDATO A INCL...		INCLUÍDO	
2009	Eaton	Chapter 5 Advances in Web Testing	EXCLUÍDO	[01]		
2003	Edmonds	Uzilla: A new tool for Web usability testing	EXCLUÍDO.	[02]		
2006	El Saddik	Performance measurements of web services-based applications	INDEFINIDO	[01]		
2006	El Yamany	A multi-agent framework for testing distributed systems	EXCLUÍDO	[06]		
2003	Elbaum et al.	Improving web application testing with user session data	CANDIDATO A INCL...		INCLUÍDO	
2005	Elbaum et al.	Leveraging user-session data to support web application testing	CANDIDATO A INCL...		INCLUÍDO	
2005	Emer	A testing approach for XML schemas	EXCLUÍDO.	[01]		
2008	Endo et al.	Web services composition testing: A strategy based on structural testing of parallel programs	CANDIDATO A INCL...		INCLUÍDO	
1996	Everett	DQS's experience with SRE	EXCLUÍDO.	[01]		
2007	Falcone	A compositional testing framework driven by partial specifications	EXCLUÍDO.	[01]		
2008	Farooq	Challenges in evaluating SOA test processes	INDEFINIDO		EXCLUÍDO	[10]
2006	Feng	Addressing the testing challenge with a web-based e-assessment system that tutors as it assesses	EXCLUÍDO.	[01]		
2004	Feng et al.	A practical approach to test case design for module based Web applications and reusability	CANDIDATO A INCL...		INCLUÍDO	
2008	Florea	A web services approach for the design of a multi-sensor data fusion system	EXCLUÍDO.	[01]		
2008	Fong	Building a test suite for web application scanners	EXCLUÍDO	[02]		
2007	Fonseca	Testing and comparing web vulnerability scanning tools for SQL injection and XSS attacks	EXCLUÍDO.	[02]		

1ª Execução (Agosto de 2009): Foram retornados 564 trabalhos

Year	Author	Title	Decisao	Visto1	Decisao2	Visto2
2006	Foster	LTSA-WS: A tool for model-based verification of Web service compositions and choreography	EXCLUÍDO	[02]		
2004	Fu et al.	Testing of java web services for robustness	CANDIDATO A INCL...		INDEFINIDO	
2008	Fu	APOGEE-automated project grading and instant feedback system for web based computing	EXCLUÍDO.	[01]		
2008	Fu	SAFELI - SQL injection scanner using symbolic execution	EXCLUÍDO	[02]		
2008	Fugini et al.	Quality analysis of composed services through fault injection	CANDIDATO A INCL...		INCLUÍDO	
2005	Gamage	Where the speed matters...Zero-response-time search engine for small collections	EXCLUÍDO.	[01]		
1995	Gedike	Intelligent ecology-medical system	EXCLUÍDO.	[01]		
2007	George	Specification for testing	EXCLUÍDO.	[01]		
2006	Giordano	Experimenting ontology web services	EXCLUÍDO.	[01]		
1996	Gonzalez	Performance evaluation of a large diagnostic expert system using a heuristic test case generator	EXCLUÍDO	[01]		
2008	Gotel	Teaching software quality assurance by encouraging student contributions to an open source web-based sys...	EXCLUÍDO.	[01]		
2008	Goth	"Googling" test practices? Web giant's culture encourages process improvement	INDEFINIDO		EXCLUÍDO	[01]
2008	Gracia	Web-based measure of semantic relatedness	EXCLUÍDO.	[01]		
2006	GradiÃ¡ar	Factors affecting the readability of colored text in computer displays	EXCLUÍDO.	[01]		
2008	Grieskamp	Model-based quality assurance of windows protocol documentation	EXCLUÍDO.	[01]		
2008	Gu	An extended MM-path approach to component-based web application testing	INDEFINIDO		EXCLUÍDO	[05]
2009	Guo	Semantic information integration and question answering based on pervasive agent ontology	EXCLUÍDO.	[01]		
2008	Guo	Web application fault classification-an exploratory study	EXCLUÍDO	[01]		
2006	Guo	Service oriented grid software testing environment	INDEFINIDO		EXCLUÍDO	[10]
2006	Gutiarrez et al.	An approach to generate test cases from use cases	INDEFINIDO		INCLUÍDO	
2002	GÃ³mez	A web-based self-monitoring system for people living with HIV/AIDS	EXCLUÍDO.	[01]		
2007	Halfond	Improving test case generation for web applications using automated interface discovery	EXCLUÍDO	[05]		
1994	Hall	Systematic incremental validation of rule-based reactive systems	EXCLUÍDO.	[01]		
2008	Hamill	Unit testing web services	INDEFINIDO		EXCLUIDO	[10]
2009	Han	Proposing-testing generic reasoning and its application in railway location	EXCLUÍDO.	[01]		
2008	Hanna and Munro	Fault-based web services testing	INDEFINIDO		INCLUÍDO	
2007	Hanna	An approach for specification-based test case generation for Web services	EXCLUÍDO	[03]		
2006	Hao	Usage-based statistical testing of web applications	EXCLUÍDO	[05]		
2005	Haque	Post-deployment specification, analysis and testing of enterprise web applications	INDEFINIDO		EXCLUÍDO	[10]
2007	Hartman	Domain specific approaches to software test automation	EXCLUÍDO.	[07]		
2007	Hartman	Domain specific approaches to software test automation	EXCLUÍDO.	[09]		
2008	Haydar et al.	Specification patterns for formal web verification	INDEFINIDO		INDEFINIDO	
2003	Heckel	Towards model-driven testing	INDEFINIDO		EXCLUÍDO	[08]
2008	Hector	A software engineering process for BDI agents	EXCLUÍDO.	[01]		
2006	Ho	Using a Petri net model approach to web applications testing	EXCLUÍDO	[05]		
2002	Holzmann	An automated verification method for distributed systems software based on model extraction	EXCLUÍDO.	[01]		
2008	Hoshino	A cloze test authoring system and its automation	EXCLUÍDO	[01]		
2008	Hou	A testing method for Web services composition based on data-flow	INDEFINIDO		EXCLUÍDO	[05]
2008	Hou	Quota-constrained test-case prioritization for regression testing of service-centric systems	CANDIDATO A INCL...		EXCLUÍDO	[05]
2008	Huachuan	The implementation of a distributed system based on a parallel algorithm for selfsimilar network traffic simul...	EXCLUÍDO	[01]		
2007	Huaikou	A formal open framework based on agent for testing web applications	EXCLUÍDO	[06]		
2005	Huang	Automated model checking and testing for composite Web services	EXCLUÍDO	[02]		
2004	Huang	Mining of web-page visiting patterns with continuous-time Markov models	EXCLUÍDO.	[01]		
2004	Huang et al.	Non-detrimental Web application security scanning	CANDIDATO A INCL...		INCLUÍDO	
2005	Huang	A testing framework for Web application security assessment	EXCLUÍDO	[02]		
2008	Hughes	Client and server verification for web services using interface grammars	EXCLUÍDO	[02]		
2007	Hwang	Development of a web-based system for diagnosing student learning problems on english tenses	EXCLUÍDO.	[01]		
2008	Insfran	A systematic review of usability evaluation in Web development	EXCLUÍDO.	[01]		
2008	Ivask	Web-based framework for distributed remote laboratory in the field of digital system test	EXCLUÍDO.	[01]		
2005	Jenkins	A web-based system to gather and distribute specifications for wind tunnel tests	EXCLUÍDO	[01]		
2008	Jia	An automatic execution system for web functional test base on modelling user's behaviour	EXCLUÍDO.	[02]		
2005	Jia	Techniques for testing software based on architectural description	EXCLUÍDO.	[01]		
2004	Jiang	Research on a testing technology based on design-by-contract	INDEFINIDO		EXCLUÍDO	[10]
2005	Jiang	Method of automated test data generation for web service	EXCLUÍDO.	[04]		
2008	Jin-Mei	A cryptosystem based on multiple chaotic maps	EXCLUÍDO.	[01]		
2008	Jingmin	A configurable web service performance testing framework	EXCLUÍDO	[02]		
1992	Jones	Probabilistic parser applied to software testing documents	EXCLUÍDO.	[01]		
2006	Jr and Vergilio	Exploring perturbation based testing for web services	CANDIDATO A INCL...		INCLUÍDO	
2009	Juristo	A look at 25 years of data	EXCLUÍDO	[01]		
2008	Juszczyk	GENESIS - A framework for automatic generation and steering of testbeds of complex Web services	EXCLUÍDO.	[01]		
2001	Kallepalli and Tian	Measuring and modeling usage and reliability for statistical web testing	INDEFINIDO		EXCLUIDO	[05]
2001	Kallepalli	Usage measurement for statistical web testing and reliability analysis	EXCLUÍDO.	[01]		
2004	Kaner	Panel: Research challenges in testing Web applications	EXCLUÍDO.	[09]		
2006	Karam et al.	An abstract model for testing MVC and workflow based Web applications	INDEFINIDO		INCLUÍDO	
2007	Karam et al.	An abstract workflow-based framework for testing composed web services	INDEFINIDO		INDEFINIDO	
2008	Karusseit	Policy expression and checking in XACML, WS-Policies, and the jABC	EXCLUÍDO.	[01]		
1996	Katragadda	Real-time ultrasonic weld evaluation system	EXCLUÍDO.	[01]		
2006	Keum et al.	Generating test cases for web services using extended finite state machine	CANDIDATO A INCL...		INDEFINIDO	
2008	Kim	Usability evaluation framework for ubiquitous computing device	INDEFINIDO		EXCLUÍDO	[08]
2006	Kim	An agent-oriented approach to semantic web services	EXCLUÍDO.	[01]		
2006	Kim	An agent system for automated Web service composition and invocation	EXCLUÍDO.	[01]		
2005	Kinugasa	Operation-style answering in multimedia testing system DrilLs-M for Kanji letter shape learning	EXCLUÍDO.	[01]		
2007	Kissoum	A formal approach for functional and structural test case generation in multi-agent systems	EXCLUÍDO	[01]		
2007	Kohavi	Practical guide to controlled experiments on the web: Listen to your customers not to the hippo	EXCLUÍDO.	[07]		

1ª Execução (Agosto de 2009): Foram retornados 564 trabalhos

Year	Author	Title	Decisao	Visto1	Decisao2	Visto2
2009	Kohavi	Controlled experiments on the web: Survey and practical guide	EXCLUÍDO.	[01]		
2008	Kolupaev	CAPTCHAs: Humans vs. bots	EXCLUÍDO.	[01]		
2007	Kona	Automatic composition of semantic web services	EXCLUÍDO.	[01]		
2007	Kongsli	Security testing with selenium	EXCLUÍDO.	[02]		
2006	Kongsli	Towards agile security in web applications	EXCLUÍDO.	[07]		
2008	Kooper	3D medical volume reconstruction using web services	EXCLUÍDO	[01]		
2007	Koopman et al.	Model-based testing of thin-client web applications and navigation input	CANDIDATO A INCL...		INCLUÍDO	
2007	Kosuga	Sania: Syntactic and semantic analysis for automated testing against SQL injection	EXCLUÍDO	[02]		
2008	Kou	A standard process for data mining based software debugging	EXCLUÍDO.	[01]		
2006	Krishnamurthy	A synthetic workload generation technique for stress testing session-based systems	EXCLUÍDO.	[01]		
2000	Kung	Object-oriented Web test model for testing Web applications	INDEFINIDO		EXCLUÍDO	[10]
2006	Kuo	Web-based adaptive testing system	EXCLUÍDO	[01]		
2007	Kurniawan	An integrated grid development environment in eclipse	EXCLUÍDO	[01]		
2008	KÃ¼ster	On the empirical evaluation of semantic web service approaches: Towards common SWS test collections	EXCLUÍDO.	[01]		
2008	KÃ¼ster	OPOSSum - An online portal to collect and share SWS descriptions	EXCLUÍDO.	[01]		
2006	Labiche	Planning and scheduling from a class test order	EXCLUÍDO.	[01]		
2006	Lai	A model and implementation of function test automation for web applications	INDEFINIDO		EXCLUÍDO	[10]
2007	Lallali	Timed modeling of Web services composition for automatic testing	EXCLUÍDO	[01]		
2008	Lallali	Automatic timed test case generation for web services composition	CANDIDATO A INCL...		EXCLUÍDO	[10]
2000	Laney	Creating interactive Web pages	INDEFINIDO.		EXCLUÍDO	[10]
2002	Lau	Asynchronous web-based patient-centered home telemedicine system	EXCLUÍDO.	[01]		
2008	Lee	Automatic mutation testing and simulation on OWL-S specified Web services	EXCLUÍDO	[02]		
2008	Lertphumpanya a...	Basis path test suite and testing process for WS-BPEL	CANDIDATO A INCL...		INDEFINIDO	
2008	Li	Development and key techniques of Petri net-based discrete event simulation and control software	EXCLUÍDO.	[01]		
2004	Li	Code-coverage guided prioritized test generation	CANDIDATO A INCL...		EXCLUÍDO	[06]
2008	Li et al.	Control flow analysis and coverage driven testing for web services	INDEFINIDO.		INDEFINIDO	
2008	Li and Miao	An approach to modeling and testing Web applications based on use cases	INDEFINIDO		INCLUÍDO	
2008	Li et al.	A UML-based approach to testing Web applications	INDEFINIDO		INCLUÍDO	
2006	Li et al.	A framework of model-driven web application testing	INDEFINIDO		INCLUÍDO	
2007	Li et al.	Web application model recovery for user input validation testing	INDEFINIDO		INCLUÍDO	
2008	Li	Multiple-implementation testing for XACML implementations	EXCLUÍDO.	[01]		
2008	Li	Framework for interoperability of BPEL-based workflows	INDEFINIDO		EXCLUÍDO	[05]
2004	Li	Web services testing, the methodology, and the implementation of the automation-testing tool	INDEFINIDO		EXCLUÍDO	[10]
2008	Li	An adaptive and trustworthy software testing framework on the grid	EXCLUÍDO	[01]		
2006	Li	Large-scale software unit testing on the grid	EXCLUÍDO	[07]		
2005	Li et al.	BPEL4WS unit testing: Framework and implementation	CADIDATO A INCLU...		INCLUÍDO	
2003	Liang	Simulation-driven performance testing method for Web applications	EXCLUÍDO.	[10]		
2008	Lim	KnowledgeSeeker - An ontological agent-based system for retrieving and ANalyzing Chinese web articles	EXCLUÍDO.	[01]		
2006	Lin et al.	Applying safe regression test selection techniques to java web services	INDEFINIDO		INCLUÍDO	
2008	Lin	Personalized E-eommerce recommendation based on ontology	EXCLUÍDO.	[01]		
2007	Lin	Comparing cancer and normal gene regulatory networks based on microarray data and transcription factor a...	EXCLUÍDO.	[01]		
2008	Liu et al.	A WS-BPEL based structural testing approach for Web service compositions	INDEFINIDO		INCLUÍDO	
2001	Liu et al.	An object-based data flow testing approach for web applications	INDEFINIDO		INCLUÍDO	
2000	Liu et al.	Structural testing of Web applications	INDEFINIDO		INCLUÍDO	
2006	Liu	Data flow analysis and testing of JSP-based Web applications	INDEFINIDO		EXCLUÍDO	[08]
2007	Liu	Remote measurement framework - Multi-agent and web service solutions	EXCLUÍDO.	[01]		
2003	Liu	Agent-based automated compatibility software test for NLSF	EXCLUÍDO.	[01]		
2004	Liu	Software architecture of a network fault management system in real time	EXCLUÍDO.	[01]		
2007	Liu	A multi-agent-based architecture for enterprise customer and supplier cooperation context-aware information ...	EXCLUÍDO	[01]		
2008	Liu	Queueing-model-based adaptive control of multi-tiered web applications	EXCLUÍDO.	[01]		
2006	Liu	Adaptive control of multi-tiered Web applications using queueing predictor	EXCLUÍDO.	[07]		
2008	Liu	The architecture of an event correlation service for adaptive middleware-based applications	EXCLUÍDO.	[01]		
2008	Lokasyuk	Neural nets method for estimation of the software retesting necessity	EXCLUÍDO.	[01]		
2007	Looker	Determining the dependability of Service-Oriented Architectures	INDEFINIDO		EXCLUÍDO	[10]
2007	Lopez-Ramos	Designing a novel SOA architecture for security and surveillance WSNs with COTS	EXCLUÍDO.	[01]		
2006	Lucca and Fasoli...	Testing Web-based applications: The state of the art and future trends	INDEFINIDO		EXCLUIDO	[01]
2002	Lucca et al.	Testing web applications	CANDIDATO A INCL...		INCLUÍDO	
2006	Lucca et al.	A technique for reducing user session data sets in web application testing	CANDIDATO A INCL...		INCLUÍDO	
2005	Luo	Web services-based test report generation	EXCLUÍDO.	[01]		
2008	Lutteroth	Modeling a realistic workload for performance testing	INDEFINIDO		EXCLUÍDO	[08]
2008	Ma	A method of end-user testing for Web services	CANDIDATO A INCL...		EXCLUÍDO	[10]
2008	Ma	Checking active XML validation	EXCLUÍDO.	[01]		
2005	Ma	Web based system performance testing and optimization	CANDIDATO A INCL...		EXCLUÍDO	[10]
2009	MaclÃ¡-PÃ©rez	Energy management system as an embedded service: Saving energy consumption of ICT	EXCLUÍDO.	[01]		
2008	Maidantchik	Web system to support analysis of the Tile Calorimeter commissioning	EXCLUÍDO.	[01]		
2004	Majumdar	Optimization of B2C E-services solutions with the systems of library	EXCLUÍDO.	[01]		
2006	Mansour	Testing web applications	INDEFINIDO		EXCLUÍDO	[08]
2004	Mao	Testing and evaluation for Web usability based on extended Markov chain model	EXCLUÍDO	[05]		
2008	Marchetto	Talking about a mutation-based reverse engineering for Web testing: A preliminary experiment	CANDIDATO A INCL...		INCLUÍDO	
2008	Marchetto	Special section on testing and security of Web systems	EXCLUÍDO.	[09]		
2008	Marchetto et al.	A case study-based comparison of web testing techniques applied to AJAX web applications	CANDIDATO A INCL...		INCLUÍDO	
2008	Marchetto et al.	Comparing "traditional" and web specific fit tables in maintenance tasks: A preliminary empirical study	INDEFINIDO.		INCLUÍDO	
2005	Marchetto	Evaluating web applications testability by combining metrics and analogies	excluído	[01]		
2008	Marchetto	State-based testing of Ajax Web applications	CANDIDATO A INCL...		EXCLUÍDO	[07]

1ª Execução (Agosto de 2009): Foram retornados 564 trabalhos

Year	Author	Title	Decisao	Visto1	Decisao2	Visto2
2004	Martins Jr.	Using support vector machines to recognize products in E-commerce pages	EXCLUÍDO.	[01]		
2007	Mashhadi	Investigating customers' decision to accept e-banking services	EXCLUÍDO.	[01]		
2002	Maurer	Extreme programming: Rapid development for web-based applications	EXCLUÍDO.	[01]		
2006	Mayer and Labke	Towards a BPEL unit testing framework	INDEFINIDO		INCLUÍDO	
2008	McAllister	Leveraging user interactions for in-depth testing of web applications	EXCLUÍDO	[02]		
2001	McGough	A web-based testing system with dynamic question generation	EXCLUÍDO.	[01]		
2005	Mei	A framework for testing Web services and its supporting tool	CANDIDATO A INCL...		EXCLUÍDO	[04]
2008	Mei	Data flow testing of service-oriented workflow applications	INDEFINIDO.		EXCLUÍDO	[05]
2002	MenascÃ©	Load testing of Web sites	INDEFINIDO		EXCLUÍDO	[08]
2001	Mendes	Using an engineering approach to understanding and predicting web authoring and design	EXCLUÍDO.	[01]		
2006	Merdes	Ubiquitous RATs: How resource-aware run-time tests can improve ubiquitous software systems	INDEFINIDO		EXCLUÍDO	[01]
2007	Miao	An approach to generating test cases for testing component-based web applications	INDEFINIDO		EXCLUÍDO	[10]
2008	Miao	Towards automatically generating test paths for Web application testing	CANDIDATO A INCL...		EXCLUÍDO	[05]
1998	Mihajlovi?	A knowledge-based test plan generator for incremental unit and integration software testing	EXCLUÍDO	[01]		
2003	Min	A replicated server architecture supporting survivable services	EXCLUÍDO.	[01]		
2008	Minghui	A static analysis approach for automatic generating test cases for web applications	INDEFINIDO		EXCLUÍDO	[03]
2002	Moe	Using execution trace data to improve distributed systems	EXCLUÍDO.	[01]		
2004	Mosorov	Internet application for distant testing	EXCLUÍDO.	[01]		
2008	Moss	Constraint patterns and search procedures for CP-based random test generation	EXCLUÍDO.	[01]		
2003	Mugridge	A customer test generator for web-based systems	INDEFINIDO		EXCLUÍDO	[02]
2007	Murlewski	Comprehensive approach to web applications testing and performance analysis	INDEFINIDO.		EXCLUÍDO	[08]
2007	Murphy	Work in progress - Testing right from the start	EXCLUÍDO.	[01]		
1997	Murrell	A survey of tools for the validation and verification of knowledge-based systems: 1985-1995	EXCLUÍDO.	[01]		
2007	Mustafa et al.	A strategy for testing of web based software	INDEFINIDO		EXCLUÍDO	[08]
2008	Nanda	Dynamic multi-process information flow tracking for web application security	EXCLUÍDO.	[07]		
2008	Nanda	Web application attack prevention for tiered internet services	EXCLUÍDO	[02]		
2004	Nguyen	Web application testing beyond tactics	INDEFINIDO		EXCLUÍDO	[10]
2008	Noikajana and Su...	Web service test case generation based on decision table	INDEFINDO		INCLUÍDO	
2008	Noikajana	An approach for web service test case generation based on web service semantics	INDEFINIDO		EXCLUÍDO	[10]
2002	O'Connor	Beta testing a web-based interactive coaching system for team skill development	EXCLUÍDO	[01]		
2008	Offutt et al.	An industrial case study of bypass testing on Web applications	CANDIDATO A INCL...		INCLUÍDO	
2004	Offutt et al.	Bypass testing of web applications	CANDIDATO A INCL...		INCLUÍDO	
2004	Offutt	Web application bypass testing	CANDIDATO A INCL...		EXCLUÍDO	[10]
2006	Oh	WSBen: A Web services discovery and composition benchmark	EXCLUÍDO.	[01]		
2000	Ohara	Database approach to testing and evaluating of object-oriented programs	EXCLUÍDO.	[01]		
2007	Okubo	Secure software development through coding conventions and frameworks	INDEFINIDO		EXCLUÍDO	[01]
2008	Othman	Trust mechanisms: An integrated approach for e-commerce website development process	INDEFINIDO		EXCLUÍDO	[08]
2008	Palomo-Duarte	Improving Takuan to analyze a meta-search engine WS-BPEL composition	EXCLUÍDO	[01]		
2008	Palonio-Duarte	Takuan: A dynamic invariant generation system for WS-BPEL compositions	EXCLUÍDO.	[07]		
2001	Pan	Robustness testing and hardening of CORBA ORB implementations	EXCLUÍDO.	[01]		
2007	Paradkar	Automated functional conformance test generation for semantic web services	EXCLUÍDO	[02]		
2008	Parveen and Tilley	A research agenda for testing SOA-based systems	INDEFINIDO			[01]
2005	Pasala	An approach for test suite selection to validate applications on deployment of COTS upgrades	EXCLUÍDO	[01]		
2006	Pasala	An approach based on modeling dynamic behavior of the system to assess the impact of COTS upgrades	EXCLUÍDO	[07]		
1999	Paul	Software metrics knowledge and databases for project management	EXCLUÍDO.	[01]		
2008	Perumal and Dha...	Performance analysis of distributed web application: A key to high perform computing perspective	INDEFINIDO		INCLUÍDO	
2008	Peyton	Integration testing of composite applications	EXCLUÍDO	[06]		
2003	Pipka	Test-driven web application development in Java	EXCLUÍDO	[10]		
1991	Plant	Factors in software quality for knowledge-based systems	EXCLUÍDO.	[01]		
2005	Polini	Interoperability testing of Web Services for e-learning	EXCLUÍDO.	[10]		
2005	Ponnurangam	Fuzzy complexity assessment model for resource negotiation and allocation in agent-based software testing f...	EXCLUÍDO.	[01]		
2003	Probert et al.	Formal verification and validation for e-commerce: Theory and best practices	INDEFINIDO		INDEFINIDO	
2008	Psarris	Symbolic analysis for increased program execution performance	EXCLUÍDO.	[01]		
2002	Pugh	Automated testing \& windows CE	EXCLUÍDO.	[02]		
2006	Qi	An agent-based data-flow testing approach for Web applications	EXCLUÍDO	[06]		
2005	Qi	An agent-based testing approach for web applications	EXCLUÍDO	[07]		
2007	Qian	Recursion detection testing on software components	INDEFINIDO		EXCLUÍDO	[01]
2006	Qian	Decoupling metrics for services composition	EXCLUÍDO.	[01]		
2008	Raffelt et al.	Hybrid test of web applications with webtest	INDEFINIDO		INCLUÍDO	
2008	Raffelt	Dynamic testing via automata learning	EXCLUÍDO.	[01]		
2004	Ran	AutoDBT: A framework for Automatic Testing of Web Database applications	EXCLUÍDO	[02]		
2009	Ran	Building test cases and oracles to automate the testing of web database applications	EXCLUÍDO	[02]		
2002	Rankin	The software testing automation framework	EXCLUÍDO.	[01]		
2006	Reich	Continuous software test distributed execution and integrated into the globus toolkit	INDEFINIDO		EXCLUÍDO	[08]
2008	Reza et al.	A model based testing technique to test web applications using StateCharts	CANDIDATO A INCL...		INCLUÍDO	
2006	Rezgui	Ontology-centered knowledge management using information retrieval techniques	EXCLUÍDO.	[01]		
2005	Ricca and Tonella	Anomaly detection in Web applications: A review of already conducted case studies	CANDIDATO A INCL...		EXCLUÍDO	[01]
2005	Ricca and Tonella	Web testing: A roadmap for the empirical research	CANDIDATO A INCL...		EXCLUÍDO	[01]
2002	Ricca and Tonella	Testing processes of web applications	CANDIDATO A INCL...		INCLUÍDO	
2001	Ricca and Tonella	Analysis and testing of web applications	INDEFINIDO		INCLUÍDO	
2001	Ruegsegger	A process control strategy based upon device performance metrics	EXCLUÍDO.	[01]		
2004	Ruffo	WALTy: A user behavior tailored tool for evaluating web application performance	EXCLUÍDO.	[02]		
2007	Ruth et al.	Towards automatic regression test selection for web services	INDEFINIDO		INCLUÍDO	
2007	Ruth	Towards automating regression test selection for web services	INDEFINIDO		EXCLUÍDO	[07]

1ª Execução (Agosto de 2009): Foram retornados 564 trabalhos

Year	Author	Title	Decisao	Visto1	Decisao2	Visto2
2008	Ruth	Empirical studies of a decentralized regression test selection framework for web services	CANDIDATO A INCL...		EXCLUÍDO	[05]
2008	Sagarna	Dynamic search space transformations for software test data generation	EXCLUÍDO.	[01]		
2006	Sakkopoulos	Adaptive mobile web services facilitate communication and learning internet technologies	EXCLUÍDO.	[01]		
2001	Saleh	Anomaly detection in concurrent Java programs using dynamic data flow analysis	EXCLUÍDO.	[01]		
2008	Sama	Multi-layer faults in the architectures of mobile, context-aware adaptive applications: A position paper	INDEFINIDO		EXCLUÍDO	[01]
2008	Sampath et al.	Prioritizing user-session-based test cases for web applications testing	CANDIDATO A INCL...		INCLUÍDO	
2007	Sampath et al.	Applying concept analysis to user-session-based testing of web applications	EXCLUÍDO	[03]		
2006	Sampath	Integrating customized test requirements with traditional requirements in Web application testing	EXCLUÍDO	[05]		
2004	Sampath	Composing a framework to automate testing of operational web-based software	EXCLUÍDO	[02]		
2004	Sampath	A scalable approach to user-session based testing of web applications through concept analysis	EXCLUÍDO	[03]		
2006	Sanchez	Automatic support for testing web-based enterprise applications	EXCLUÍDO.	[02]		
2004	Sanghi	A testbed for performance evaluation of load-balancing strategies for Web server systems	EXCLUÍDO.	[01]		
2007	Sari	Croquet based virtual museum implementation with grid computing connection	EXCLUÍDO.	[01]		
2004	Saward	Assessing usability through perceptions of information scent	EXCLUÍDO.	[01]		
2004	Schell	Web-based minimally invasive surgery training: Competency assessment in PGY 1-2 surgical residents	EXCLUÍDO.	[01]		
2001	Scholtz	Adaptation of traditional usability testing methods for remote testing	EXCLUÍDO	[02]		
1998	Scholtz	WebMetrics: A methodology for producing usable Web sites	INDEFINIDO		EXCLUÍDO	[10]
2004	Sen	Efficient decentralized monitoring of safety in distributed systems	EXCLUÍDO	[01]		
2006	Sengupta	A research agenda for distributed software development	EXCLUÍDO.	[01]		
2004	Seo	Web server attack categorization based on root causes and their locations	EXCLUÍDO.	[01]		
2008	Seung	Automatic generation of testing environments for web applications	EXCLUÍDO	[02]		
2008	Shahriar and Zulk...	MUSIC: Mutation-based SQL injection vulnerability checking	INDEFINIDO		INCLUÍDO	
2006	Shams et al.	A model-based approach for testing the performance of Web applications	INDEFINIDO		INCLUÍDO	
2005	Shan	Research progress in software testing	INDEFINIDO		EXCLUÍDO	[10]
2007	Shanmugam	Risk mitigation for cross site scripting attacks using signature based model on the server side	EXCLUÍDO.	[01]		
2005	Sharifi	A new method on automated web application testing	INDEFINIDO		EXCLUÍDO	[10]
1985	Shaughnessy	IMPLEMENTATION OF A MULTILAYER IMAGING CHEMISTRY INTO A VLSI CMOS PROCESS.	EXCLUÍDO.	[01]		
1998	Shinohara	An optimal software release problem under cost rate criterion: Artificial neural network approach	EXCLUÍDO.	[01]		
2005	Siblini and Manso...	Testing web services	CANDIDATO A INCL...		INCLUÍDO	
2003	Siegel	The National Library of Medicine's strategy for assessing the impacts of health information web sites	EXCLUÍDO.	[01]		
2007	Simion	A hybrid algorithm for scheduling workflow applications in grid environments (ICPDP)	EXCLUÍDO.	[01]		
2003	Simmonds	Towards scalable network emulation	EXCLUÍDO.	[01]		
1993	Simpson	Impact of Commercial Off-the-Shelf (COTS) equipment on system test and diagnosis	EXCLUÍDO.	[01]		
2005	Singh	Adaptation of cultural content: Evidence from B2C e-commerce firms	EXCLUÍDO	[01]		
2006	Sinha	Model-based functional conformance testing of Web services operating on persistent data	EXCLUÍDO	[05]		
2008	Smith	On guiding the augmentation of an automated test suite via mutation analysis	EXCLUÍDO.	[01]		
2007	Sneed	The design and use of WSDL-Test: A tool for testing Web services	EXCLUÍDO.	[02]		
2008	Song	Modeling Web browser interactions and generating tests	INDEFINIDO		EXCLUÍDO	[08]
2009	Song	Extracting database interactions and generating test for web applications	INDEFINIDO		EXCLUÍDO	[10]
2005	Sprenkle et al.	An empirical comparison of test suite reduction techniques for user-session-based testing of web applications	INDEFINIDO		EXCLUIDO	[08]
2006	Sprenkle	A case study of automatically creating test suites from Web application field data	EXCLUÍDO	[03]		
2007	Sprenkle	Learning effective oracle comparator combinations for web applications	EXCLUÍDO.	[07]		
2007	Sprenkle	Automated oracle comparators for testing web applications	EXCLUÍDO.	[02]		
2008	Srivastava	Extension of object-oriented software testing techniques to agent oriented software testing	EXCLUÍDO.	[01]		
2008	Srivastava	Regression testing techniques for agent oriented software	EXCLUÍDO.	[01]		
2009	Stepien	Integration testing of web applications and databases using TTCN-3	EXCLUÍDO	[06]		
2008	Stepien	Framework testing of web applications using TTCN-3	INDEFINIDO		EXCLUÍDO	[07]
2008	Stoehel	Testing optimization for mission-critical, complex, distributed systems	EXCLUÍDO.	[01]		
2008	Swan	Web services test bed	EXCLUÍDO.	[01]		
2007	Swierenga	Implementing a corporate web accessibility compliance program	EXCLUÍDO.	[09]		
2006	Taipale	Improving software testing by observing practice	EXCLUÍDO.	[01]		
2007	Takagi	Analysis of navigability of Web applications for improving blind usability	EXCLUÍDO.	[01]		
2007	Takagi	An overview and case study of a statistical regression testing method for software maintenance	EXCLUÍDO.	[01]		
2006	Tamura	Software reliability modeling in distributed development environment	EXCLUÍDO.	[01]		
2001	Tang	On improving model and strategy of testing C/S application software	EXCLUÍDO.	[01]		
2008	Tappenden and ...	A three-tiered testing strategy for cookies	INDEFINIDO		INCLUÍDO	
2005	Tappenden	Agile security testing of web-based systems via HTTPUnit	INDEFINIDO		EXCLUÍDO	[08]
2008	Tarhini et al.	Regression testing web applications	INDEFINIDO		INCLUÍDO	
2006	Tarhini	A simple approach for testing Web service based applications	EXCLUÍDO.	[01]		
2006	Tarhini	Regression testing web services-based applications	EXCLUÍDO	[05]		
2008	Tarpe et al.	Supporting security testers in discovering injection flaws	INDEFINIDO		INCLUÍDO	
2006	Tian and Ma	Web Testing for Reliability Improvement	CANDIDATO A INCL...		INCLUÍDO	
2003	Tian	A Hierarchical Strategy for Testing Web-Based Applications and Ensuring Their Reliability	INDEFINIDO		EXCLUÍDO	[10]
2004	Tonella and Ricca	A 2-layer model for the white-box testing of web applications	INDEFINIDO		INCLUÍDO	
2004	Tonella	Statistical testing of Web applications	INDEFINIDO		EXCLUÍDO	[05]
2008	Tonella	Dynamic model extraction and statistical analysis of web applications: Follow-up after 6 years	EXCLUÍDO.	[01]		
2007	Torkey	A new methodology for Web testing	EXCLUÍDO.	[01]		
2007	Tran	Composing OWL-S web services	EXCLUÍDO.	[01]		
2005	Tsai et al.	Developing and assuring trustworthy web services	CANDIDATO A INCL...		INCLUÍDO	
2003	Tsai	Scenario-Based Web Services Testing with Distributed Agents	INDEFINIDO		EXCLUÍDO	[10]
2005	Tsai	Swiss cheese test case generation for web services testing	CANDIDATO A INCL...		EXCLUÍDO	[10]
2005	Tsai	Web service group testing with windowing mechanisms	INDEFINIDO		EXCLUÍDO	[05]
2004	Tsai	Testing Web Services Using Progressive Group Testing	CANDIDATO A INCL...		EXCLUÍDO	[10]
2005	Tsai	Adaptive testing, oracle generation, and test case ranking for web services	CANDIDATO A INCL...		EXCLUÍDO	[01]

1ª Execução (Agosto de 2009): Foram retornados 564 trabalhos

Year	Author	Title	Decisao	Visto1	Decisao2	Visto2
2002	Tsai	Adaptive scenario-based object-oriented test frameworks for testing embedded systems	EXCLUÍDO.	[01]		
2005	Tsai	Stochastic voting algorithms for web services group testing	EXCLUÍDO	[05]		
2004	Tsai	Cooperative and group testing in verification of dynamic composite web services	INDEFINIDO		EXCLUÍDO	[10]
2005	Tsai	Web services-based collaborative and cooperative computing	INDEFINIDO		EXCLUÍDO	[08]
2005	Tsai	A robust testing framework for verifying Web Services by completeness and consistency analysis	EXCLUÍDO	[05]		
2007	Tsai	A coverage relationship model for test case selection and ranking for multi-version software	EXCLUÍDO.	[01]		
2008	Turner	An automated test code generation method for Web applications using activity oriented approach	INDEFINIDO		EXCLUÍDO	[08]
2008	Ummu Salima	Enhancing the efficiency of regression testing through intelligent agents	EXCLUÍDO.	[01]		
2009	Vedrines	Moving web-tension determination by out-of-plane vibration measurements using a laser	EXCLUÍDO.	[01]		
2007	Vedrines	Moving web tension determination by out of plane vibrations measurements using a laser	EXCLUÍDO.	[07]		
2007	Vieira	Benchmarking the robustness of web services	INDEFINIDO		EXCLUÍDO	[01]
2005	Vituzzi	Remote usability testing for paediatric web sites	EXCLUÍDO.	[01]		
2007	Voegele	A Laboratory Information Management System (LIMS) for a high throughput genetic platform aimed at candida...	EXCLUÍDO.	[01]		
2008	Walden	Integrating web application security into the IT curriculum	EXCLUÍDO.	[01]		
2008	Wang	An interaction-based test sequence generation approach for testing webapplications	INDEFINIDO		EXCLUÍDO	[05]
2007	Wang	Ontology-based test case generation for testing web services	EXCLUÍDO	[05]		
2008	Weichselbraun	Strategies for optimizing querying third party resources in Semantic Web applications	EXCLUÍDO.	[01]		
2007	Weidong	An embedded web services framework and data representation for distributed testing environment	EXCLUÍDO	[02]		
1999	Williams	Efficient regression testing of multi-panel systems	EXCLUÍDO	[01]		
2004	Wilson	The feasibility of cryogenic storage in space	EXCLUÍDO.	[01]		
2004	Wilson	The feasibility of cryogenic storage in space	EXCLUÍDO.	[07]		
2006	Winbladh	An automated approach for goal-driven, specification-based testing	EXCLUÍDO	[01]		
2001	Wroblewski	Design considerations for Web-based applications	EXCLUÍDO.	[01]		
2003	Wuthrich	Instructional testing through wireless handheld devices	EXCLUÍDO.	[01]		
2009	Xiong	Model-based penetration test framework for web applications using TTCN-3	EXCLUÍDO	[05]		
2007	Xu	Semantic web services discovery in P2P Environment	EXCLUÍDO.	[01]		
2004	Xu	Applying agent into web testing and evolution	EXCLUÍDO	[02]		
2006	Xu	A tuple-space-based coordination architecture for test agents in the MAST framework	EXCLUÍDO	[02]		
2006	Xu	Automatic GUI testing for an OCR system	EXCLUÍDO.	[01]		
2004	Xu and Xu	A framework for web applications testing	EXCLUÍDO	[07]		
2006	Xu	Testing forms in web applications automatically	INDEFINIDO		EXCLUÍDO	[02]
2007	Xu	Applying Agent into intelligent Web application testing	EXCLUÍDO.	[02]		
2004	Xu	Research on testing framework for Web applications	INDEFINIDO		EXCLUÍDO	[10]
2003	Xu	Regression Testing for Web Applications Based on Slicing	INDEFINIDO		EXCLUÍDO	[08]
2005	Xu	Research on the analysis and measurement for testing results of web applications	EXCLUÍDO.	[01]		
2005	Xu	Testing and fault diagnosis for Web application compatibility based on combinatorial method	INDEFINIDO.		EXCLUÍDO	[10]
2003	Xu	Applying users' actions obtaining methods into Web performance testing	INDEFINIDO		EXCLUÍDO	[10]
2003	Xu	Test Web applications based on Agent	EXCLUÍDO	[06]		
2005	Xu et al.	Testing web services by XML perturbation	EXCLUÍDO.	[03]		
2005	Xu	Research on security solution for software update	EXCLUÍDO.	[01]		
2000	Yamada	Software reliability growth model for a distributed development environment	EXCLUÍDO.	[01]		
2006	Yang	Software configuration management based on logical element relationships	EXCLUÍDO.	[01]		
2007	Yang	Using AJAX to build an online assessment management system based on QTI and Web 2.0	EXCLUÍDO.	[01]		
2008	Yang	Study on architecture and key technologies of shanghai ocean disaster grid system (SODGS)	EXCLUÍDO	[01]		
2003	Yang	Research on personalized Web service	EXCLUÍDO.	[01]		
2006	Yanyan	Web application performance analysis based on comprehensive load testing	CANDIDATO A INCL...		EXCLUÍDO	[10]
	Yeganeh	Semantic web service composition testbed	EXCLUÍDO.	[02]		
2008	Yoon	Generating test requirements for the service connections based on the layers of SOA	EXCLUÍDO	[01]		
2008	Younis	A strategy for grid based T-Way test data generation	EXCLUÍDO.	[01]		
2008	Younis	IRPS - An efficient test data generation strategy for pairwise testing	EXCLUÍDO.	[01]		
2007	Yu	Modeling the measurements of QoS requirements in Web service systems	EXCLUÍDO.	[01]		
2006	Yu	Multi-agent testing architecture for monitoring and analyzing the performance of Web applications	EXCLUÍDO.	[06]		
2007	Yu	A workflow-based test automation framework for Web based systems	EXCLUÍDO	[02]		
2007	Yu	OWL-S based interaction testing of Web Service-based System	INDEFINIDO		EXCLUÍDO	[05]
2005	Yu	Protocol-based interoperability test of Web application	INDEFINIDO		EXCLUÍDO	[10]
2005	Yu	Web services interoperability testing based on ontology	EXCLUÍDO.	[07]		
2003	Yu	On the use of the classification-tree method by beginning software testers	EXCLUÍDO.	[01]		
1999	Yu	Strategy based on a knowledge base for black box software testing	EXCLUÍDO.	[01]		
2004	Yuen	A testing framework for web applications based on the MVC model with behavioral descriptions	CANDIDATO A INCL...		EXCLUÍDO	[10]
2005	Zeinert	Database application streamlines VW's durability testing operations	EXCLUÍDO.	[01]		
2007	Zeng	Research of electronic document certification testing system base on characteristics getting	EXCLUÍDO.	[01]		
2007	Zeng	Auto-generating test sequences for web applications	INDEFINIDO		EXCLUÍDO	[08]
2007	Zeng	Model checking-based testing of Web applications	INDEFINIDO		EXCLUÍDO	[08]
2009	Zhai	A concept planning algorithm for pragmatics web service discovery	EXCLUÍDO.	[01]		
2007	Zhang	A business process of web services testing method based on UML2.0 activity diagram	INDEFINIDO		INCLUIDO	
2007	Zhang	Research on application of modeling and simulation in launch vehicle virtual test	EXCLUÍDO.	[01]		
2004	Zhang	An approach to facilitate reliability testing of Web services components	EXCLUÍDO.	[01]		
2006	Zhang	Fault injection-based test case generation for SOA-oriented software	EXCLUÍDO	[03]		
2005	Zhang	Web services quality testing	INDEFINIDO		EXCLUÍDO	[10]
2005	Zhang	Criteria analysis and validation of the reliability of web services-oriented systems	CANDIDATO A INCL...		EXCLUÍDO	[06]
2008	Zhang	OWL-S based test case generation	EXCLUÍDO	[01]		
2008	Zhao	Using web services for distributed medical visualisation	EXCLUÍDO.	[01]		
2007	Zheng and Chen	Maintaining multi-tier web applications	CANDIDATO A INCL...		INDEFINIDO	
2007	Zheng	Analysis of BPEL data dependencies	EXCLUÍDO	[03]		

1ª Execução (Agosto de 2009): Foram retornados 564 trabalhos

Year	Author	Title	Decisao	Visto1	Decisao2	Visto2
2007	Zheng	A model checking based test case generation framework for web services	INDEFINIDO		EXCLUÍDO	[08]
2008	Zheng	A distributed replication strategy evaluation and selection framework for fault tolerant Web services	EXCLUÍDO.	[01]		
2008	Zhong	BPEL4WS unit testing: Framework and implementation	CANDIDATO A INCL...		EXCLUÍDO	[07]
2007	Zhongsheng	A practical web testing model for web application testing	EXCLUÍDO	[05]		
1996	Zhou	System testing strategy for new generation network applications. Phase-based state control method in BECO...	EXCLUÍDO.	[01]		
2006	Zhu	A framework for service-oriented testing of web services	INDEFINIDO		EXCLUÍDO	[08]
2004	Zhu	Cooperative agent approach to quality assurance and testing web software	EXCLUÍDO	[06]		
2008	Zhu	Research on a method of Ajax-based web user behavior collection	EXCLUÍDO.	[01]		
2005	Zunliang	Actionable knowledge model for GUI regression testing	EXCLUÍDO.	[01]		
2000			EXCLUÍDO	[09]		
2009		Lecture Notes in Business Information Processing E-Technologies: Innovation in an Open World: 4th Intenati...	EXCLUÍDO	[09]		
2005		Web Engineering: 5th International Conference, ICWE 2005. Proceedings	EXCLUÍDO	[09]		
2005		Fundamental Approaches to Software Engineering - 8th International Conference, FASE 2005, held as part of ...	EXCLUÍDO	[09]		
2008		Proceeding of the 2008 International Conference on Semantic Web and Web Services, SWWS 2008	EXCLUÍDO:	[09]		
2008		TAV-WEB 2008 - Proceedings of the Workshop on Testing, Analysis and Verification of Web Software	EXCLUÍDO	[09]		
2008		Proceedings - testing: Academic and industrial conference practice and research techniques, TAIC part 2008	EXCLUÍDO	[09]		
2008		Proceedings - Thirteenth IEEE International Conference on the Engineering of Complex Computer Systems, I...	EXCLUÍDO	[09]		
2008		Proceedings of the 1st International Conference on Software Testing, Verification and Validation, ICST 2008	EXCLUÍDO	[09]		
2008		2008 IEEE International Conference on Software Testing Verification and Validation Workshop, ICSTW'08	EXCLUÍDO	[09]		
2008		Recent Advances in Intrusion Detection - 11th International Symposium, RAID 2008, Proceedings	EXCLUÍDO	[09]		
2007		2007 IEEE/ACS International Conference on Computer Systems and Applications, AICCSA 2007	EXCLUÍDO	[09]		
2007		Proceedings - 18th IEEE International Symposium on Software Reliability Engineering, ISSRE 2007	EXCLUÍDO	[09]		
2007		Agent-Oriented Software Engineering VII: 7th International Workshop, AOSE 2006 Revised and Invited Papers	EXCLUÍDO	[09]		
2007		Proceedings - 7th International Conference on Quality Software, QSIC 2007	EXCLUÍDO	[09]		
2007		DoSTA 2007: Workshop on Domain-Specific Approaches to Software Test Automation - In conjunction with th...	EXCLUÍDO	[09]		
2007		Proceedings - NWeSP 2007 Third International Conference on Next Generation Web Services Practices	EXCLUÍDO	[09]		
2007		Proceedings - Eighth International Symposium on Autonomous Decentralized Systems ISADS 2007	EXCLUÍDO	[09]		
2007		Testing of Software and Communicating Systems: 19th IFIP TC6/WG6.1 International Conference, TestCom 2...	EXCLUÍDO	[09]		
2007		FOSE 2007: Future of Software Engineering	EXCLUÍDO	[09]		
2006		Innovative Internet Community Systems - 5th International Workshop, IICS 2005, Revised Papers	EXCLUÍDO	[09]		
2006		Testing of Communicating Systems - 18th IFIP TC 6/WG 6.1 International Conference, TestCom 2006, Proce...	EXCLUÍDO	[09]		
2006		Architecting Systems with Trustworthy Components -International Seminar, Revised Selected Papers	EXCLUÍDO	[09]		
2006		Formal Approaches to Software Testing and Runtime Verification First Combined International Workshops FA...	EXCLUÍDO	[09]		
2006		Proceedings of the Eighth IEEE International Symposium on Web Site Evolution, WSE 2006	EXCLUÍDO	[09]		
2006		ICWE'06: The Sixth International Conference on Web Engineering	EXCLUÍDO	[09]		
2005		Service Availability - Second International Service Availability Symposium, ISAS 2005, Revised Selected Papers	EXCLUÍDO	[09]		
2005		Proceedings: Seventh IEEE International Symposium on Web Site Evolution (WSE 2005)	EXCLUÍDO	[09]		
2005		Proceedings - 31st EUROMICRO Conference on Software Engineering and Advanced Applications - EUROMI...	EXCLUÍDO	[09]		
2005		Proceedings 2005 Workshop on Techniques, Methodologies and Tools for Performance Evaluation of Compl...	EXCLUÍDO	[09]		
2005		Proceedings: Fifth International Conference on Quality Software (QSIC 2005)	EXCLUÍDO	[09]		
2005		Proceedings SOSE 2005 IEEE International Workshop on Service-Oriented System Engineering	EXCLUÍDO	[09]		
2004		Proceedings: 15th International Symposium on Software Reliability Engineering, ISSRE 2004	EXCLUÍDO	[09]		
1996		Proceedings of the 1996 IEEE 20th Annual International Computer Software \& Applications Conference, COM...	EXCLUÍDO	[09]		
1992		Symposium on Assessment of Quality Software Development Tools	EXCLUÍDO	[09]		
1990		COMPEURO'90: Proceedings of the 1990 IEEE International Conference on Computer Systems and Software...	EXCLUÍDO	[09]		

2ª Execução (Setembro de 2010): Foram retornados mais 237 novos trabalhos

Year	Author	Title	Decisao	Visto1	Decisao2	Visto2
2009	?ezankovÃ¡	Cluster, SOM and NMF analyses of web patterns	EXCLUÍDO	[01]		
2009	Alalfi	Modelling methods for web application verification and testing: State of the art	EXCLUÍDO	[01]		
2009	Alalfi	Automated reverse engineering of UML sequence diagrams for dynamic web applications	EXCLUÍDO.	[01]		
2009	Alrouh	A performance evaluation of security mechanisms for web services	EXCLUÍDO	[01]		
2009	Alwan	Ranking and selecting integrity tests in a distributed database	EXCLUÍDO	[01]		
2010	Amalfitano	Rich internet application testing using execution trace data	EXCLUIDO	[03]		
2010	Anchlia	A novel injection aware approach for the testing of database applications	EXCLUÍDO	[01]		
2010	Andrews	Scalability issues with using FSMWeb to test web applications	CANDIDATO A INCL...		INCLUIDO	[07]
2009	Antunes	Effective detection of SQL/XPath Injection vulnerabilities in web services	CANDIDATO A INCL...		EXCLUIDO	[07]
2009	Antunes	Detecting SQL injection vulnerabilities in web services	CANDIDATO A INCL...		INCLUIDO	
2009	Apilli	Fault-based combinatorial testing of web services	CANDIDATO A INCL...		EXCLUIDO	[08]
2008	Arantes	WEB-PerformCharts: A collaborative web-based tool for test case generation from Statecharts	EXCLUIDO	[01]		
2010	Armando	Model-checking driven security testing of web-based applications	INDEFINIDO		EXCLUIDO	[03]
2010	Avancini	Towards security testing with taint analysis and genetic algorithms	EXCLUIDO	[05]		
2010	Bagnato	Practical experience gained from passive testing of Web based systems	INDEFINIDO		EXCLUIDO	[05]
2010	Bai	Message broker using asynchronous method invocation in web service and its evaluation	EXCLUÍDO	[01]		
2009	Bai	Risk-based adaptive group testing of semantic web services	EXCLUÍDO	[01]		
2009	Bangalore	Securing web servers using self cleansing intrusion tolerance (SCIT)	CANDIDATO A INCL...		EXCLUIDO	[02]
2009	Bartolini	Whitening SOA testing	CANDIDATO A INCL...		INCLUIDO	
2009	Bartolini	WS-TAXI: A WSDL-based testing tool for web services	CANDIDATO A INCL...		EXCLUIDO	[02]
2010	Bartsch	Supporting authorization policy modification in agile development of web applications	EXCLUÍDO	[01]		
2010	Belli	Event-driven modeling and testing of real-time web services	CANDIDATO A INCL...		INCLUIDO	
2009	Belli	Testing compositeweb services - An event-based approach	EXCLUÍDO	[07]		
2010	Benjamin	Some modeling challenges when testing rich internet applications for security	EXCLUÍDO	[01]		
2010	Bertolini	A framework for GUI testing based on use case design	EXCLUIDO	[01]		
2009	Bertolino	Approaches to testing service-oriented software systems	EXCLUIDO	[01]		
2009	Bertolino	Automatic synthesis of behavior protocols for composable web-services	EXCLUIDO	[01]		
2009	Bessayah	A formal approach for specification and verification of fault injection process	EXCLUIDO	[01]		
2009	Bezemer	Automated security testing of web widget interactions	INDEFINIDO		INCLUÍDO	
2009	Bhattacharjee	The chemical thermodynamic module of the expert system for thermodynamics ("Test") web application	EXCLUIDO	[01]		
2010	Bjerkan	Patients and professionals in collaborative testing of a web-based tool for integrated care: An evaluation study	EXCLUIDO	[01]		
2009	Blanco	A first approach to test case generation for BPEL compositions of web services using Scatter Search	EXCLUÍDO	[03]		
2009	Bo	Modeling web applications and generating tests: A combination and interactions-guided approach	EXCLUÍDO	[01]		
2007	Cai	Synthesizing client load models for performance engineering via web crawling	EXCLUÍDO	[03]		
2009	Cao	Testing Web services composition using the TGSE tool	EXCLUÍDO	[03]		
2010	Carbo	An extension of a fuzzy reputation agent trust model (AFRAS) in the ART testbed	EXCLUIDO	[01]		
2009	Chen	Turning access into a web-enabled secure information system for clinical trials	EXCLUIDO	[01]		
2009	Chen	UML activity diagram-based automatic test case generation for java programs	EXCLUIDO	[03]		
2009	Chen	Server-side regression test for Web application developments	INDEFINIDO		EXCLUIDO	[02]
2009	Chen	Prescription map generation intelligent system of precision agriculture based on web services and WebGIS	EXCLUIDO	[01]		
2008	Choi	A collaborative face recognition framework on a social network platform	EXCLUIDO	[01]		
2009	Choudhary	Automated client-side monitoring for web applications	EXCLUIDO	[01]		
2010	Ciampa	A heuristic-based approach for detecting SQL-injection vulnerabilities in web applications	CANDIDATO A INCL...		INCLUIDO	
2009	Costa	Dependability benchmarking using software faults: How to create practical and representative faultloads	EXCLUIDO	[01]		
2009	Dao	Idea: Automatic security testing for web applications	CANDIDATO A INCL...		EXCLUIDO	[10]
2009	De la Calle	BIRI: A new approach for automatically discovering and indexing available public bioinformatics resources fro...	EXCLUIDO	[01]		
2009	Deng	A QoS-oriented approach for web service group testing	EXCLUIDO	[01]		
2010	Ding	Splitter: A proxy-based approach for post-migration testing of web applications	EXCLUÍDO	[01]		
2010	Dobolyi	Addressing high severity faults in web application testing	EXCLUIDO	[01]		
2009	Dobolyi	Harnessing web-based application similarities to aid in regression testing	EXCLUIDO	[02]		
2010	DomÃ-nguez-Jim...	GAmera: A tool for WS-BPEL composition testing using mutation analysis	EXCLUIDO	[04]		
2009	DomÃ-nguez-Jim...	GAmera: An automatic mutant generation system for WS-BPEL compositions	EXCLUIDO	[07]		
2009	Dong	Construction and test of web service solution for E-government	EXCLUIDO	[01]		
2009	Dong	Test case generation method for BPEL-based testing	EXCLUIDO	[03]		
2009	Dong	Testing WSDL-based Web Service automatically	INDEFINIDO		EXCLUIDO	[03]
2006	Dong	Web service testing method based on fault-coverage	EXCLUIDO	[05]		
2009	Dong	Test method for BPEL-based Web Service composition based on data flow analysis	INDEFINIDO		EXCLUIDO	[10]
2010	Du	An effect evaluation model for vulnerability testing of web application	INDEFINIDO		EXCLUIDO	[01]
2010	Eisenberg	Apatite: A new interface for exploring APIs	EXCLUIDO	[01]		
2010	El-Adaway	Multiagent system for construction dispute resolution (MAS-COR)	EXCLUIDO	[01]		
2009	Escobedo	Observability and controllability issues in conformance testing of web service compositions	CANDIDATO A INCL...		EXCLUIDO	[10]
2010	Estero-Botaro	Quantitative evaluation of mutation operators for WS-BPEL compositions	EXCLUIDO	[01]		
2009	Frantzen	On-the-fly model-based testing of web services with jambition	CANDIDATO A INCL...		EXCLUIDO	[10]
2009	Fraternali	A higher order generative framework for weaving traceability links into a code generator for web application te...	EXCLUIDO	[05]		
2010	GonzÃ¡lez	A collaborative benchmarking framework for multibody system dynamics	EXCLUIDO	[01]		
2010	Gu	Coverage criteria and test requirement reduction for component-based web application	EXCLUIDO	[05]		
2010	Guo	A novel approach for question answering and automatic diagnosis based on pervasive agent ontology in med...	EXCLUIDO	[01]		
2010	Guo	Design and implementation of performance testing model for web services	CANDIDATO A INCL...		EXCLUIDO	[08]
2006	GutiÃ©rrez	Generating test cases from sequences of use cases	EXCLUIDO	[03]		
2009	Halfond	Penetration testing with improved input vector identification	CANDIDATO A INCL...		INCLUIDO	
2008	Halfond	Web application modeling for testing and analysis	EXCLUIDO	[01]		
2010	HallÃ¯	Runtime verification of Web service interface contracts	INDEFINIDO		EXCLUIDO	[02]
2009	Hamed	Performance testing for web based application architectures (.NET vs. Java EE)	EXCLUIDO	[01]		
2009	Hamill	Web service validation enabling test-driven development of service-oriented applications	CANDIDATO A INCL...		EXCLUIDO	[01]

2ª Execução (Setembro de 2010): Foram retornados mais 237 novos trabalhos

Year	Author	Title	Decisao	Visto1	Decisao2	Visto2
2009	Han	Designing a virtualized testbed for dynamic multimedia service composition	EXCLUIDO	[01]		
2005	Haydar	Properties and scopes in web model checking	EXCLUIDO	[01]		
2009	He	Design and implementation of student attendance management system based on MVC	EXCLUIDO	[01]		
2009	Herbold	Automated refactoring suggestions using the results of code analysis tools	EXCLUIDO	[01]		
2009	Huachuan	Research on the parallel algorithm for self-similar network traffic simulation	EXCLUIDO	[01]		
2009	Huang	BestRec: A behavior similarity based approach to services recommendation	EXCLUIDO	[01]		
2010	Idrus	Using three layer model (TLM) in web form design: WeFDeC checklist development	EXCLUIDO	[01]		
2009	Jati	Quality evaluation of e-government website using web diagnostic tools: Asian case	INDEFINIDO		EXCLUIDO	[02]
2009	Jiang	A quick testing model of web performance based on testing flow and its application	CANDIDATO A INCL...		INCLUIDO	
2010	Jiang	Exploration of real-time middleware for Web database	EXCLUIDO	[01]		
2009	Jiang	Test-data generation for web services based on contract mutation	EXCLUIDO	[04]		
2010	Jin	Automated behavioral regression testing	EXCLUIDO	[01]		
2009	Jokhio	Goal-based testing of semantic web services	EXCLUIDO	[07]		
2009	Jokhio	A framework for testing semantic web services using model checking	CANDIDATO A INCL...		EXCLUIDO	[07]
2009	Jokhio	Towards specification based testing for semantic web services	CANDIDATO A INCL...		INDEFINIDO	
2009	Juan Jo?e	A framework for mutant genetic generation for WS-BPEL	EXCLUIDO	[05]		
2009	Kam	Lessons learned from a survey of web applications testing	EXCLUIDO	[01]		
2009	Karimpour	Conceptual discovery of web services using WordNet	EXCLUIDO	[01]		
2009	Khan	A methodology for model-based regression testing of web services	INDEFINIDO		EXCLUIDO	[08]
2010	Kourtesis	Increased reliability in SOA environments through registry-based conformance testing of Web services	INDEFINIDO		EXCLUIDO	[10]
2010	Kremenova	Proposal of key factors for user-oriented web applications	INDEFINIDO		EXCLUIDO	[10]
2010	Krishnamurthy	A model-based performance testing toolset for web applications	CANDIDATO A INCL...		EXCLUIDO	[10]
2009	Kuk	Robustness testing framework for web services composition	INDEFINIDO		INCLUIDO	
2009	Kumar Ashok	Automated regression suite for testing web services	INDEFINIDO		EXCLUIDO	[08]
2009	Kustanto	Automatic source code plagiarism detection	EXCLUIDO	[01]		
2010	Kvadsheim	Web-Based automatic feedback on assignments in statistics: How can it help students learn statistics and u...	EXCLUIDO	[01]		
2008	Lallali	Transforming BPEL into intermediate format language forweb services composition testing	INDEFINIDO		EXCLUIDO	[08]
2009	Laranjeiro	Extending test-driven development for robust web services	CANDIDATO A INCL...		EXCLUIDO	[08]
2009	Laranjeiro	Improving web services robustness	EXCLUIDO	[07]		
2009	Li	What are the problem makers: Ranking activities according to their relevance for process changes	EXCLUIDO	[01]		
2010	Li	A test platform for evaluation of web service composition algorithm	INDEFINIDO		EXLCUIDO	[02]
2009	Li	A combinatorial approach to multi-session testing of stateful web services	EXCLUIDO	[07]		
2009	Li	An Abstract GFSM Model for optimal and incremental conformance testing of web services	INDEFINIDO		EXCLUIDO	[05]
2008	Li	An approach to modeling and testing Web applications based on use cases	INDEFINIDO		EXCLUIDO	[08]
2009	Li	An approach to testing Web applications on-the-fly	CANDIDATO A INCL...		EXCLUIDO	[03]
2006	Li	BPEL-unit: JUnit for BPEL processes	INDEFINIDO		INCLUIDO	
2009	Li	Towards a practical and effective method for web services test case generation	CADIDATO A INCLU...		INDEFINIDO	
2009	Lin	A multi-level sanitizing strategy based on injection point	CANDIDATO A INCL...		INDEFINIDO	
2009	Lin	Incorporating domain knowledge and information retrieval techniques to develop an architectural/engineering...	EXCLUIDO	[01]		
2009	Liping	Test purpose-based test generation for web applications	EXCLUIDO	[01]		
2009	Liu	The applications of pressure test in the B/S system	INDEFINIDO		EXCLUIDO	[01]
2009	Liu	A Unified test framework for continuous integration testing of SOA solutions	EXCLUIDO	[01]		
2009	Liu	Semantic web service modeling and composition based on description logic rule	EXCLUIDO	[01]		
2009	Liu	Targeted advertising based on intelligent agents in e-commerce	EXCLUIDO	[01]		
2009	Louw	Scripting attacks for existing browsers	CANDIDATO A INCL...		INCLUÍDO	
2009	Luo	Clustering and tailoring user session data for testing web applications	CANDIDATO A INCL...		INCLUIDO	
2010	Ma	Research on spacecraft test language and executing platform	EXCLUIDO	[01]		
2009	Magedanz	Control framework design for future internet testbeds	EXCLUÍDO	[01]		
2010	Mahmood	Replicating web contents using a hybrid particle swarm optimization	EXCLUIDO	[01]		
2009	Mani	Efficient testing of service-oriented applications using semantic service stubs	EXCLUIDO	[01]		
2009	Mao	Towards a hierarchical testing and evaluation strategy for web services system	CANDIDATO A INCL...		INCLUIDO	
2009	Mao	A specification-based testing framework for web service-based software	EXCLUIDO	[07]		
2009	Marchetto	OQMw: An OO quality model for web applications	EXCLUIDO	[01]		
2010	Mateo	Mutation at system and functional levels	EXCLUIDO	[01]		
2010	Matias Jr.	Accelerated degradation tests applied to software aging experiments	EXCLUIDO	[01]		
2010	de Matos	From formal requirements to automated web testing and prototyping	INDEFINIDO		EXCLUIDO	[02]
2009	Medina	Proposal of a methodology for implementing a service-oriented architecture in distributed manufacturing syst...	EXCLUIDO	[01]		
2009	Mei	A context-aware orchestrating and choreographic test framework for service-oriented applications	EXCLUIDO	[01]		
2009	Mei	Data flow testing of service choreography	EXCLUIDO	[01]		
2009	Mei	Tag-based techniques for black-box test case prioritization for service testing	CANDIDATO A INCL...		EXCLUIDO	[01]
2009	MeiJunjin	An approach for SQL injection vulnerability detection	CANDIDATO A INCL...		EXCLUIDO	[01]
2009	Merchant	A browser agnostic web application UI test framework: Motivation, architecture, and design	INIDEFINIDO		EXCLUIDO	[08]
2009	Mesbah	Invariant-based automatic testing of Ajax user interfaces	CANDIDATO A INCL...		INCLUIDO	
2000	Meyerhoff	Quality goals and testing approaches on the Internet [QualitÃ¤tsziele und testaufgaben im internet]	INDEFINIDO		EXCLUIDO	[10]
2009	Ming	Integrated development environment based on resource library for intelligent opto-electric instruments	EXCLUIDO	[01]		
2009	Montoto	Automating navigation sequences in AJAX websites	EXCLUIDO	[01]		
2009	Montoto	Web navigation sequences automation in modern websites	EXLCUIDO	[01]		
2008	Mostefaoui	On quality assurance of web services in agile projects: An experience report	INDEFINIDO		INDEFINIDO	
2009	Mustafa	Classification of software testing tools based on the software testing methods	EXCLUIDO	[01]		
2009	NerbrÃ¥ten	HACME game: A tool for teaching software security	EXCLUIDO	[01]		
2009	Noikajana	An improved test case generation method for web service testing from WSDL-S and OCL with pair-wise testin...	EXCLUIDO	[03]		
2009	Offutt	Modeling presentation layers of web applications for testing	EXCLUÍDO.	[01]		
2010	Ogata	A method of automatic integration test case generation from UML-based scenario	INDEFINIDO		EXCLUIDO	[08]
	Padovani	Contract-based discovery of Web services modulo simple orchestrators	EXCLUIDO	[01]		

2ª Execução (Setembro de 2010): Foram retornados mais 237 novos trabalhos

Year	Author	Title	Decisao	Visto1	Decisao2	Visto2
2009	Palomo-Duarte	Enhancing WS-BPEL dynamic invariant generation using xml schema and xpath information	INIDEFINIDO		EXCLUIDO	[10]
2009	Park	Automatic discovery of web services based on dynamic black-box testing	EXCLUIDO.	[01]		
2009	Paskalev	Runtime generation of a user interface, described in a database	EXCLUIDO	[01]		
2009	Pava	A self-configuring test harness for web applications	CANDIDATO A INCL...		*EXCLUÍDO	[02]
2008	Pentafronimos	Interoperability testing for e-Government Web services	INDEFINIDO		INCLUIDO	
2009	Pina	E-government evolution in EU local governments: A comparative perspective	EXCLUIDO	[01]		
2009	Praba	Fuzzy semi-Markov model for web testing	EXCLUIDO	[05]		
2009	Pretre	Using common criteria to assess quality of web services	EXCLUIDO	[01]		
2008	Pretre	Automating UML models merge for web services testing	EXCLUIDO	[07]		
2009	Pretre	Automated UML models merging for web services testing	INDEFINIDO		EXCLUIDO	[10]
2009	Pun	Audit trail analysis for traffic intensive web application	INDEFINIDO		EXCLUÍDO	[08]
2009	Qian	Testing component-based web applications using component automata	CANDIDATO A INCL...		EXCLUIDO	[08]
2009	Rajan	WEAVE: WEb applications validation environment	INDEFINIDO		INCLUIDO	
2009	Ramollari	Leveraging semantic web service descriptions for validation by automated functional testing	EXCLUIDO	[03]		
2008	Rivolli	An approach for tracking user interaction with interactive applications [Uma abordagem para o rastreamento ...	EXCLUIDO	[01]		
2009	Robles Luna	Bridging test and model-driven approaches in web engineering	CANDIDATO A INCL...		EXCLUIDO	[01]
2010	Roest	Regression testing Ajax applications: Coping with dynamism	CANDIDATO A INCL...		EXCLUIDO	[03]
2009	Rollet	Testing robustness of communicating systems using ioco-based approach	EXCLUIDO	[01]		
2009	Rupprecht	Evaluation of web user interfaces for the online retail of apparel	EXCLUIDO	[01]		
2008	Ruth	Concurrency in a decentralized automatic regression test selection framework for web services	EXCLUIDO	[02]		
2010	Salva	Stateful web service robustness	CANDIDATO A INCL...		INCLUIDO	
2010	Salva	A preliminary study on BPEL process testability	EXCLUIDO	[01]		
2010	Shahriar	PhishTester: Automatic testing of phishing attacks	CANDIDATO A INCL...		INCLUIDO	
2009	Shahzad	Automated optimum test case generation using web navigation graphs	EXCLUIDO	[03]		
2010	Shelly	Application of traditional software testing methodologies to web accessibility	INDEFINIDO		EXCLUIDO	[01]
2010	Shimomura	Server-driven regression test for Web applications and its performance	INDEFINIDO		INCLUIDO	
2009	Shimomura	Synchronization of multi-window requests for server-side regression test of web applications	EXCLUIDO	[07]		
2008	Siddavatam	Comprehensive test mechanism to detect attack on web services	INDEFINIDO		EXCLUIDO	[08]
2009	Silva Solino	Mutation based testing of web services	CANDIDATO A INCL...		INCLUIDO	
2009	Silveira	Exploring XML perturbation techniques for web services testing	CANDIDATO A INCL...		EXCLUIDO	[10]
2009	Simanta	Information assurance challenges and strategies for securing SOA environments and web services	EXCLUIDO	[01]		
2010	Simon	Evaluation of WS-* standards based interoperability of SOA products for the hungarian e-government infrastru...	INDEFINIDO		EXCLUIDO	[01]
2009	Sitar-Taut	A case study on usability metrics applied in Romanian E-commerce environment	INDEFINIDO		EXCLUIDO	[10]
2010	Smith	Idea: Using system level testing for revealing SQL injection-related error message information leaks	CANDIDATO A INCL...		EXCLUIDO	[10]
2009	Smith	Should software testers use mutation analysis to augment a test set?	EXCLUIDO	[01]		
2009	Song	Modeling database interactions in Web applications and generating test cases	EXCLUIDO	[07]		
2009	Song	Considering web frameset and browser interactions in modeling and testing of web applications	INDEFINIDO		EXCLUIDO	[07]
2009	Song	A method of the web service composition based on semantics	EXCLUIDO	[01]		
2005	Sprenkle	Automated replay and failure detection for web applications	CANDIDATO A INCL...		INCLUIDO	
2009	Sun	Client-side detection of XSS worms by monitoring payload propagation	CANDIDATO A INCL...		INCLUIDO	
2010	Takamatsu	Automated detection of session fixation vulnerabilities	CANDIDATO A INCL...		EXCLUIDO	[02]
2009	Tappenden	Cookies: A deployment study and the testing implications	INDEFINIDO		EXCLUIDO	[01]
2009	Tewari	An improved discovery engine for efficient and intelligent discovery of web service with publication facility	EXCLUIDO	[01]		
2009	Tsai	Optimum tuning of defense settings for common attacks on the web applications	INDEFINIDO		EXCLUÍDO	[08]
2009	Tuglular	Protocol-based testing of firewalls	EXCLUIDO	[01]		
2009	Ueno	Early capacity testing of an enterprise service bus	EXCLUIDO	[01]		
2009	Vahid Dastjerdi	SCAIMO - A case for enabling security in semantic web service composition	EXCLUIDO	[01]		
2010	Veanes	Rex: Symbolic regular expression explorer	EXCLUIDO	[01]		
2008	Wang	Requirement model-based mutation testing for web service	CANDIDATO A INCL...		INCLUIDO	
2009	Wang	Mutation test based on OWL-S requirement model	EXCLUIDO	[07]		
2010	Watkins	Introducing fault-based combinatorial testing to web services	INDEFINIDO		EXCLUIDO	[08]
2010	Witherell	Improved knowledge management through first-order logic in engineering design ontologies	EXCLUIDO	[01]		
2010	Wu	Unit testing and action-level security solution of struts web applications based on MVC	CANDIDATO A INCL...		INCLUIDO	
2009	Wusteman	OJAX: A case study in agile Web 2.0 open source development	EXCLUIDO	[01]		
2007	Xie	Smart: A tool for application reference testing	CANDIDATO A INCL...		INDEFINIDO	
2009	Xing	Semantic Web service discovery based on WordNet ontology and PLSA	EXCLUIDO	[01]		
2010	Yang	A regression testing method for Composite Web Service	CANDIDATO A INCL...		INCLUIDO	
2010	Yang	Analysis and test design for development of web applications	CANDIDATO A INCL...		EXCLUIDO	[10]
2010	Yin	A compensation-supported substitution QoS model based on detailed transaction	EXCLUIDO	[01]		
2009	Yu	Tabu search and genetic algorithm to generate test data for BPEL program	EXCLUIDO	[04]		
2009	Yu	Towards call for testing: An application to user acceptance testing of web applications	CANDIDATO A INCL...		INCLUIDO	
2009	Zakaria	Unit testing approaches for BPEL: A systematic review	EXCLUIDO	[01]		
2008	Zhai	WSDL-based ICS proforma and generation method	EXCLUIDO	[01]		
2009	Zhang	Detecting fence-sittings among categories of web services	EXCLUIDO	[01]		
2010	Zhang	A Mobile Agent-Based Tool Supporting Web Services Testing	EXCLUIDO	[06]		
2009	Zhang	Themes4BPEL: An efficient aspect-oriented web service composition design approach	EXLCUIDO	[01]		
2009	Zhongsheng	An approach to testing web applications based on probable FSM	INDEFINIDO		EXCLUIDO	[08]
2009	Zhou	Research on some issues of software testing technology	EXCLUIDO	[01]		
2009	Zhou	Improved fuzzy set information retrieval approach on duplicate webpage detection	INDEFINIDO		EXCLUIDO	[01]
2009	Zhu	Testing a web application involving web browser interaction	CANDIDATO A INCL...		EXCLUIDO	[08]
2009	Zhu	Generating test case from functional requirement of Web applications	EXCLUIDO	[03]		
2010	Zhu	Study on beta testing of web application	INDEFINIDO		EXCLUIDO	[05]
2010		Proceedings - 21st Australian Software Engineering Conference, ASWEC 2010	EXCLUÍDO	[09]		
2010		W4A 2010 - International Cross Disciplinary Conference on Web Accessibility Raleigh 2010	EXCLUÍDO	[09]		

2ª Execução (Setembro de 2010): Foram retornados mais 237 novos trabalhos

Year	Author	Title	Decisao	Visto1	Decisao2	Visto2
2010		EuroSys'10 - Proceedings of the EuroSys 2010 Conference	EXCLUíDO	[09]		
2009		Proceedings - 2009 33rd Annual IEEE International Computer Software and Applications Conference, COMP...	EXCLUíDO	[09]		
2009		Proceedings - 2009 33rd Annual IEEE International Computer Software and Applications Conference, COMP...	EXCLUíDO	[09]		
2009		2009 4th South-East European Workshop on Formal Methods: Formal Methods for Web Services; Formal Met...	EXCLUíDO	[09]		
2009		Proceedings of the IASTED International Conference on Software Engineering, SE 2009	EXCLUíDO	[09]		
2009		Proceedings - 16th Asia-Pacific Software Engineering Conference, APSEC 2009	EXCLUíDO	[09]		
2009		QSIC 2009 - Proceedings of the 9th International Conference on Quality Software	EXCLUíDO	[09]		
2009		Testing of Software and Communication Systems - 21st IFIP WG 6.1 International Conference, TESTCOM 20...	EXCLUíDO	[09]		
2009		1st International Conference on Advances in System Testing and Validation Lifecycle, VALID 2009	EXCLUíDO	[09]		
2009		Web Services and Formal Methods - 5th International Workshop, WS-FM 2008 Revised Selected Papers	EXCLUíDO	[09]		
2009		Proceedings - 1st International Symposium on Search Based Software Engineering, SSBSE 2009	EXCLUíDO	[09]		
2009		Proceedings - 2009 Australian Software Engineering Conference, ASWEC 2009	EXCLUíDO	[09]		
2009		Proceedings of the 2009 ICSE Workshop on Automation of Software Test, AST 2009	EXCLUíDO	[09]		
2009		2009 31st International Conference on Software Engineering - Companion Volume, ICSE 2009	EXCLUíDO	[09]		
2009		Autonomic and Trusted Computing - 6th International Conference, ATC 2009, Proceedings	EXCLUíDO	[09]		

APÊNDICE B – EVOLUÇÃO DA INFRA-ESTRUTURA COMPUTACIONAL MARAKÁ

Este apêndice descreve os requisitos de adaptação da infra-estrutura Maraká para suportar o procedimento Porantim-WAT. Serão descritos os ajustes realizados no banco de dados e no código fonte.

REQUISITO 1) Criação de repositório das abordagens WAT

Deverá ser criada uma tabela que armazene os atributos propostos na estrutura de caracterização de abordagens WAT. Essa tabela será denominada **mos_tecnicasWAT**. Alguns atributos serão associados a tabelas de domínio, que também deverão ser criadas.

Deverão ser criadas ainda as tabelas de relacionamento entre **mos_tecnicasWAT** e as tabelas de domínio de valores. Segue abaixo o detalhamento das tabelas a serem criadas.

Nome da tabela: mos_tecnicasWAT			
Campo	Tipo	Tamanho	Observação
Id	bigint	12	Chave auto-incremental
Ano	varchar	4	
Artigo	varchar	100	
Autores	varchar	100	
Descrição	text	-	
Entradas	text	-	
Limitações	text	-	
metodoDesenv	varchar	20	
Ferramenta	int	11	0-Não, 1-Sim
nomeFerramenta	varchar	100	
tipoferramenta	varchar	50	
refereciaCompleta	varchar	100	
Saídas	Text	-	
Unidade	Varchar	100	
complexidadePassos	Int	11	1-Baixo, 2-Médio, 3-Alto
descricaoPassos	longtext	-	
tarefasAutomaticas	Int	11	0-Não, 1-Sim

tarefasAutomaticasDescricao	varchar	1000	
tipoEstudo	varchar	50	
Benefícios	varchar	100	
Problemas	varchar	100	

Tabelas de domínio de valores. As tabelas de domínio de valor serão compostas por 2 colunas: id da tabela, que será uma chave auto-incremental, e a coluna que irá descrever o domínio de valores de determinados atributos da técnica WAT. Seguindo essa formação, serão criadas as seguintes tabelas:

- mos_tecnicaTeste
- mos_analiseTeste
- mos_dominio
- mos_camada
- mos_criterio

Exemplo:

Tabela: mos_ tecnicaTeste			
Campo	Tipo	Tamanho	Observação
Id	Bigint	12	Chave auto-incremental
Nome	Texto	50	Ex: Funcional, Estrutural

Tabelas de relacionamento. As tabelas de relacionamento serão compostas por 3 colunas: id da tabela, que será uma chave auto-incremental, id da técnica WAT e o id da tabelas de domínio de valor que se relaciona com a técnica. Seguindo essa formação, serão criadas as seguintes tabelas de relacionamento:

- mos_tecnicasWAT_ tecnicaTeste
- mos_tecnicasWAT_ analiseTeste
- mos_tecnicasWAT_dominio
- mos_tecnicasWAT_camada
- mos_tecnicasWAT_criterio

Exemplo:

Tabela: mos_tecnicasWAT_tecnicaTeste			
Campo	Tipo	Tamanho	Observação
Id	Bigint	12	Chave auto-incremental
idTecnicaWAT	Bigint	12	Chave estrangeira mos_tecnicasWAT
idTopoTecnica	Bigint	12	Chave estrangeira mos_ tecnicaTeste

REQUISITO 2) Caracterização de Projetos de software Web

A tabela que armazena os atributos de caracterização de projeto de software, denominada **mos_caracterizacaoprojeto**, deverá ser adaptada para armazenar os atributos prospostos para a caracterização de projetos de software Web. Alguns atributos serão associados a tabelas de domínio, descritas anteriormente.

Deverão ser criadas ainda as tabelas de relacionamento entre **mos_caracterizacaoprojeto** e as tabelas de domínio de valores. Segue abaixo o detalhamento das tabelas a serem adaptadas/criadas.

Nome da tabela: mos_caracterizacaoprojeto				
Campo	Tipo	Tamanho	Exibir	Observação
id	Bigint	12	Não	Chave auto-incremental
idPlano	Bigint	12	Não	
idplataforma	Bigint	12	Não	
idparadigma	Bigint	12	Não	
idlinguagem	Bigint	12	Não	
idtecnologia	Bigint	12	Sim	
tipoteste	Smallint	6	Sim	Combo Box.
duracao	Varchar	40	Não	
complexidade	Smallint	6	Não	
frequencia	Smallint	6	Não	
tamanho	Smallint	6	Não	
custoteste	Smallint	6	Não	
ferramenta	Smallint	6	Não	
iddominioAplicacao	Smallint	6	Cond	
idcamadaAplicacao	Smallint	6	Cond	
idcomplexidadePassos	Smallint	6	Sim	Novo. Combo Box.
idnivelautomação	Smallint	6	Sim	Novo. Combo Box.
idtipoestudo	Smallint	6	Cond	Novo. Combo Box.
obrigatoriedadedominioAplicacao	Smallint	6	Cond	Novo.

obrigatoriedadecamadaAplicacao	Smallint	6	Cond	Novo.
obrigatoriedadecomplexidadePassos	Smallint	6	Sim	Novo.
obrigatoriedadenivelautomação	Smallint	6	Sim	Novo.
obrigatoriedadetipoestudo	Smallint	6	Cond	Novo.
obrigatoriedadePlataforma	Smallint	6	Não	
obrigatoriedadeParadigma	Smallint	6	Não	
obrigatoriedadeLinguagem	Smallint	6	Não	
obrigatoriedadeModelo	Smallint	6	Sim	
obrigatoriedadeTecnologia	Smallint	6	Sim	
obrigatoriedadeNivelTeste	Smallint	6	Sim	
obrigatoriedadeTipoTeste	Smallint	6	Sim	
obrigatoriedadeCaracteristica	Smallint	6	Sim	
obrigatoriedadeTipoFalha	Smallint	6	Sim	
obrigatoriedadeFerramenta	Smallint	6	Não	
indicadorCobertura	Float	Default	Não	Utilizado na combinação (Passo 4)
indicadorModelagem	Float	Default	Não	Utilizado na combinação (Passo 4)
indicadorRH	Float	Default	Não	Utilizado na combinação (Passo 4)

Tabelas de relacionamento. As tabelas de relacionamento serão compostas por 3 colunas: id da tabela, que será uma chave auto-incremental, id da técnica WAT e o id da tabelas de domínio de valor que se relaciona com a técnica. Seguindo essa formação, será criada a seguinte tabela de relacionamento:

- mos_ caracterizacaoprojeto_criterio

Exemplo:

Tabela: mos_ caracterizacaoprojeto_criterio			
Campo	Tipo	Tamanho	Observação
Id	Numérico	12	Chave auto-incremental
idTecnicaWAT	Numérico	12	Chave estrangeira mos_tecnicasWAT
idCriterio	Numérico	12	Chave estrangeira mos_ criterio

Observação: Tabelas de relacionamento que já existem e que serão utilizadas:

- mos_caracterizacaoprojeto_caracteristicas (A ser exibido)
- mos_caracterizacaoprojeto_modelos (A ser exibido)
- mos_caracterizacaoprojeto_niveis (Exibição Condicional)

- mos_caracterizacaoprojeto_tipofalha (A ser exibido)

REQUISITO 3) Inclusão do componente Gerenciador de Porantim-WAT

Criação de um componente para gerenciar as funcionalidades que serão providas pelo Porantim WAT.

Requisito 3.1) Maraká deverá prover funcionalidades para a manutenção do corpo de conhecimento Porantim-WAT, tais como cadastro, edição, exclusão e visualização das abordagens de teste para aplicações Web.

Na visualização das abordagens, deverão ser exibidos os atributos abaixo listados:

- Ano
- Autores
- Título
- Ferramenta
- Tipo de Estudo

Requisito 3.2) Maraká deverá prover funcionalidade para realizar a carga em lote de abordagens WAT para o corpo de conhecimento, a partir de um arquivo no formato bibtex.

Atributo Abordagem WAT	Tabela.Campo	Tag JabRef	Regra de Preenchimento
Ano	mos_tecnicasWAT.ano	Year	-
Artigo	mos_tecnicasWAT.artigo	Title	-
Autores	mos_tecnicasWAT.autores	Autor	-
refereciaCompleta	Mos_tecnicasWAT.ReferênciaCompleta	Concatenação de várias tags	-
Ferramenta	Mos_tecnicasWAT.nomeFerramenta	Tool	Texto Livre
Tipo de Estudo	mos_tecnicasWAT.tipoEstudo	typeofstudy	Texto Livre
Benefícios	mos_tecnicasWAT.beneficios	Benefits	Texto Livre
Problemas	mos_tecnicasWAT.problemas	Problems	Texto Livre
Domínio da Aplicação	mos_tecnicaswat_dominios	applicationdomain	Texto Livre
Característica de Qualidade	mos_tecnicasWAT_caracteristicas	qualitycharacteristics	Separador “,”
Unidade	mos_tecnicasWAT.unidade	Unit	Texto Livre
Camada da	mos_tecnicasWat_camada	Applicationlayer	Separador “,”

Aplicação			
Nível de Teste	mos_ tecnicasWAT_niveis	Testinglevel	Separador “,”
Descrição da Abordagem	mos_tecnicasWAT.descricao	approachdescription	Texto Livre
Limitações	mos_tecnicasWAT.limitacoes	limitationsoftheapproach	Texto Livre
Tipos de Falhas	mos_tecnicasWAT_tiposFalhas	typeoffailures	Separador “,”
Tipo da Técnica	mos_ tecnicasWAT_tecnicaTeste	typeoftestingtechnique	[Funcional, Estrutural]
Tipo de Análise	mos tecnicasWAT_analiseTeste	typeoftestinganalisys	[Estático, Dinâmico]
Habilidade Requeridas	mos_ tecnicasWAT_habilidades	skillrequired	Separador “,”
Entradas	mos_tecnicasWAT.entradas	Inputs	Texto Livre
Saídas	Mos_tecnicasWAT.saidas	Outputs	Texto Livre
Criterios de Cobertura	mos_ tecnicasWAT_criterioCobertura	testcoverage	Separador “,”
Modelo	mos_ tecnicasWAT_modelos	modelingtechnology	Separador “,”
Tecnologia de Modelagem	mos_ tecnicasWAT_tecnologias	testgenerationtechnology	Separador “,”
Nível de Automação	Mos_tecnicasWAT.tarefasAutomaticas	Automatedsteps	0 – Não 1 – Sim
Descrição Nível de Automação	mos_tecnicasWAT. tarefasAutomaticasDescricao	Automatedstepsdescription	Texto Livre
Complexidade dos Passos não automatizados	Mos_tecnicasWAT.complexidadePassos	complexityofnotautomatedsteps	1- Baixo 2- Médio 3- Alto
Descrição da Complexidade	Mos_tecnicasWAT.descricaoPassos	complexitydescription	Texto Livre

Requisito 3.3) Maraká deverá prover funcionalidade que permita a definição dos pesos de cada atributo de caracterização das abordagens WAT. Esses pesos serão considerados no cálculo de adequação entre as características do projeto de software Web e as abordagens WAT. A definição dos pesos poderá ser realizada também para os atributos que poderão ser utilizados como critério de desempate.

A tabela que armazena o peso dos atributos no procedimento de seleção, denominada **mos_configporantim**, deverá ser adaptada para armazenar os atributos propostos para as abordagens WAT. Nesse sentido, deverão ser incluídos os campos abaixo listados:

Tabela: **mos_configporantim**		
Campo	Tipo	Tamanho
dominioAplicacao	Float	
camadaAplicacao	Float	
complexidadePassos	Float	

Nivelautomação	Float	
tipoestudo	Float	
Critério	Float	

Requisito 3.4) Maraká deverá prover funcionalidade que permita a e definição da quantidade máxima de abordagens a serem exibidas no procedimento de seleção;

Requisito 3.5) Maraká deverá ser adaptado para executar o procedimento de seleção das abordagens de teste para aplicações Web provido por *Porantim-WAT*, realizado para cada projeto de teste de software Web.

Para implementar os requisitos listados, foram criados os arquivo abaixo:

- Diretório: /Componentes/Com_configferramenta
- Arquivo: tecnicaWAT.php e tecnicaWAT.html.php

- Diretório: /Componentes/com_processmanager
- Arquivo: porantim-wat.html.php e porantim-wat.php

Adicionalmente, foram incluídas algumas funções nos arquivos abaixo:

- Diretório: /Componentes/Com_configferramenta
- Arquivo: configferramenta.php e configferramenta.html.php

- Diretório: /Componentes/com_processmanager
- Arquivo: processmanager.php

- Diretório: /Componentes/com_processmanager/languages
- Arquivo: brazilian_portuguese.php e english.php

APÊNDICE C – GLOSSÁRIO

Este apêndice descreve o glossário de termos referentes às tecnologias empregadas nas aplicações Web, identificadas nas abordagens WAT publicadas na literatura técnica e selecionadas na quasi-revisão sistemática executada neste trabalho.

Ao longo da realização da revisão da literatura e da revisão quasi sistemática para identificar e caracterizar as abordagens de testes para aplicações Web, foi identificada uma grande variedade de tecnologias empregadas nesta área de pesquisa. Este apêndice apresenta um glossário bilíngüe com o objetivo de reunir, de forma breve e objetiva, os significados dos variados termos, expressões e palavras utilizadas para Referênciar as tecnologias empregadas nas aplicações Web. Trata-se de uma coletânea de expressões com a respectiva explicação de conceitos dos termos encontrados descritos em inglês e em português.

- **AJAX**

Português:

Acrônimo para "Assíncrono de javascript e XML". Aplicações AJAX se baseiam na comunicação cliente / servidor assíncrono, na manipulação da árvore DOM em tempo de execução do lado do cliente. Isto não só torna este tipo de aplicação fundamentalmente diferente das aplicações web tradicionais, mas também a torna mais propensa a erros e mais difícil de testar.

Ajax é um conjunto de tecnologias utilizadas para desenvolver aplicativos ricos e interativos da Web. Um cliente típico Ajax é executado localmente no navegador Web do usuário e atualiza sua interface em tempo real em resposta a entrada do usuário.

Inglês:

Asynchronous Javascript and XML acronym. AJAX-based Web applications rely on stateful asynchronous client/server communication, and client-side runtime manipulation of the DOM tree. This not only makes them fundamentally different from traditional web applications, but also more error-prone and harder to test.

Ajax is a collection of technologies used to develop rich and interactive Web applications. A typical Ajax client runs locally in the user's Web browser and refreshes its interface on the fly in response to user input.

Referência: (Mesbah, 2009) (HallÃ¨, 2010)

- **BPEL**

Português:

Linguagem de Execução de Processo de Negócios - BPEL - é uma linguagem padrão para a composição de Serviços Web.

Inglês:

Business Process Execution Language – BPEL – is a standard language for Web services composition.

Referência: (Lallali, 2008)

- **BPEL4WS**

Português:

BPEL4WS ou WS-BPEL ou ainda apenas BPEL é um exemplo de Linguagem de Execução de Processo de Negócios de composição de Serviços Web. Existem outras linguagens: BPMN, WfXML, XPDL, XLANG, WSFL, etc. Todas elas descrevem um processo de negócio através da composição de serviços web.

Inglês:

The Business Process Execution Language for Web Services (BPEL4WS, or WS-BPEL as the new name, abbr. BPEL) is an example of business process programming language for web service composition. Other languages include: BPMN, WfXML, XPDL, XLANG, WSFL, etc. They all describe a business process by composing web services.

Referência: (Li, 2006)

- BPMN

Português:

Sigla para Notação para Modelagem de Processos de Negócio. A BPMN é uma notação gráfica padrão proposta pelo Business Process Management Initiative

(BPMI) para representar processos de negócio. Ao adaptar a notação BPMN, o modelo analítico proposto pode servir como um mecanismo de comunicação entre os desenvolvedores e os testadores do processo WS-BPEL de modo a facilitar as atividades de análise e teste das composições de serviços.

Inglês:
Abbrevitation for Business Process Modeling Notation. The BPMN is a standard graphical notation proposed by Business Process Management Initiative (BPMI) to represent business processes. By adapting the BPMN notation, the proposed test model can serve as a communication mechanism between the WS-BPEL process developers and testers so as to facilitate the service compositions analyzing and testing.

Referência: (Liu et al, 2008)

- **ECS**

Português:
Sigla para serviço de comércio eletrônico.

Inglês:
Abbrevitation for e-Commerce Service.

Referência: (HallÃ¨, 2010)

- LWP

Português:
LWP é uma página da Web inteira ou parte de uma página da Web que aceita dados do usuário através de um formulário HTML e, em seguida, envia os dados para um módulo de software específico.

Inglês:
An LWP is either an entire physical Web page or the portion of a Web page that accepts data from the user through an HTML form and then sends the data to a specic software module.

Referência: (Andrews et al, 2005)

- OWL-S

Português:

OWL-S facilita a especificação de Serviços Web com um conjunto de construções de linguagem de marcação para descrever as propriedades e capacidades de WS sem ambigüidades. OWL-S suporta a automatização de tarefas, incluindo automação na identificação, invocação, interoperabilidade, composição e monitoramento da execução de serviços Web.

Inglês:

OWL-S facilitates WS specification with a core set of markup language constructs for describing the properties and capabilities of WS in unambiguous and computerinterpretable form. OWL-S supports the automation of WS tasks including automated WS discovery, invocation, interoperation, composition, and execution monitoring.

Referência: (Tsai et al, 2005)

- **Phishing**

Português:

O phishing é um ataque baseado na web que atrai os usuários finais a visitar sites fraudulentos e dar informações pessoais (por exemplo, userid, senha). A informação é utilizada para realizar atividades ilegítimas, como serviços bancários online.

Inglês:

Phishing is a web-based attack which allures end users to visit fraudulent websites and give away personal information (e.g., userid, password). The information is used to perform illegitimate activities such as online banking.

Referência: (Shahriar, 2010)

- **RIAS**

Português:

Aplicação de Internet Rica utiliza tecnologias como Ajax, Flex ou Silverlight, e rompe com a abordagem tradicional de aplicações Web devido à camada servidora de computação e comunicação síncrona entre o cliente e servidores web. Este tipo de aplicação introduz novos desafios, novas vulnerabilidades de segurança, e seu comportamento torna a atividade de teste difícil.

Inglês:

Rich Internet applications (RIAs), using technologies such as Ajax, Flex, or Silverlight, break away from the traditional approach of Web applications having server-side computation and synchronous communications between the web client and servers. RIAs introduce new challenges, new security vulnerabilities, and their behavior makes it difficult or impossible to test with current web application security scanners

Referência: (Benjamin, 2010)

- **SDL**

Português:

Sigla para Linguagem de Especificação e Descrição. É uma poderosa linguagem para projetar, desenvolver e testar sistemas e-commerce baseados em CORBA.

Inglês:

Specification and Description Language (SDL) is a powerful design language, to develop and test CORBA-based e-commerce systems

Referência: (Probert et al, 2003)

- **SESSION**

Português:

Uma sessão pode ser considerada como uma seqüência de solicitações interdependentes submetidas por um único usuário

Inglês:

A session being a sequence of inter-dependent requests submitted by a single user

Referência: (Krishnamurthy, 2010)

- **SOAP**

Português:

SOAP, um protocolo de comunicação padrão para a transmissão de mensagens XML. O Protocolo Simples de Acesso a Objetos (SOAP) é um protocolo leve baseado em XML para troca de informações em um ambiente descentralizado e distribuído.

Inglês:

SOAP, a standard communication protocol for transmitting XML messages.
The Simple Object Access Protocol (SOAP) is a lightweight, XML-based protocol for exchanging information in a decentralized distributed environment.

Referência: (HallÃ¨, 2010)

- **SQL injection**

Português:

Consiste na possibilidade em que o usuário tem de injetar fragmentos de consultas SQL em campos de entrada do aplicativo da Web. Se esses campos ou os resultados da consulta SQL a serem enviados para o banco de dados não são devidamente validados, então poderá ser possível para o invasor acessar dados não autorizados, fazer engenharia reversa da estrutura do banco, ou mesmo inserir / excluir dados.

Inglês:

Consists in the possibility the user has to inject fragments of SQL queries in Web application input fields. If these fields or the resulting SQL query to be sent to the database are not properly validated, then it might be possible for the

attacker to access unauthorized data, reverse engineer the database structure, or even to insert/delete data.

Referência: (Ciampa, 2010)

- **SWRL**

Português:

Sigla para Regras semânticas de Linguagem Web – representa linguagem de regra para a web semântica que consiste em antecedentes (corpo) e subseqüentes (cabeça), como uma forma de equação antecedentes→subseqüentes. Pode ser escrita como uma expressão lógica com conjunções positivas.

Inglês:

The semantic web rule language (SWRL) represents rule language for semantic web which consists antecedent (body) and consequent (head), such as a form antecedent -> consequent. It can be written as an logic expression with positive conjunctions only.

Referência: (Noikajana and Suwannasart, 2008)

- **TTCN-3**

Português:

TTCN-3 significa Notação de Teste e Controle de Teste Versão 3.

É uma linguagem de testes padronizada internacionalmente e desenvolvida pelo Comitê Técnico ETSI MTS (Métodos de Teste e Especificação), especificamente concebida para teste e certificação. A TTCN pode ser interpretada apenas por um sistema de teste equipado adequadamente, visto que não é tão fácil para um ser humano para ler e entender. A principal desvantagem é o fato de que, a fim de executar casos de teste TTCN-3, é necessário compilador adequado. Entretanto, até o momento não existem ferramentas open source para compilar e interpretar TTCN-3.

Inglês:

TTCN-3 means Testing and Test Control Notation Version 3.

It is a core language, an internationally standardizd testing language developed by the ETSI Technical Committee MTS (Methods for Testing and Specification), specifically designed for testing and certification. The TTCN can be exactly and repeatedly interpreted only by a suitably equipped test system, while it is not so easy for a human to read and understand it. A major drawback is the fact that in order to run and execute the TTCN-3 test cases, an appropriate TTCN-3 compiler is needed.It should be noted, that until now there are no open source tools for compiling and interpreting TTCN-3.

Referência: (Dong 2009) (Pentafronimos, 2008)

- **UCTMs**

Português:

UCTMs significa Modelos de Transição de Casos de Uso. Trata-se de um perfil hierárquico de diagramas de caso de uso.

Inglês:

UCTMs means Use Case Transition Models. It is an hierarchical profile use-case diagrams.

Referência: (Li, L.a b , Miao, H.a , 2008)

- **Widget**

Português:

Páginas Ajax podem ser compostas de componentes de interface independente do usuário, muitas vezes chamados widgets. Esses widgets são mini-aplicativos compostos por um pedaço de código que pode ser incorporado em uma página HTML e executar de forma independente e próximas umas das outras, oferecendo conteúdo dinâmico e funcionalidades para uma variedade de tarefas como mostrar as últimas notícias, previsão do tempo, ou uma lista de novas mensagens de e-mail. Exemplos altamente visíveis incluem sites como o iGoogle e Netvibes que permitem aos usuários selecionar widgets a partir de um catálogo, personalizar e hospedá-los em sua própria página inicial. Eles também fornecem APIs para desenvolvedores externos criarem e incluírem novos widgets nos catálogos do widget.

Inglês:

Ajax pages can be composed from independent user interface components, often called web widgets. These widgets are mini-applications composed of a chunk of code that can be embedded in an HTML page, and run independently and next to each other, providing dynamic content and functionality for a wide variety of tasks such as showing the latest news headlines, weather predictions, or a list of new email messages. Highly visible examples include Mashup sites such as iGoogle and Netvibes, which allow users to select widgets from a catalog, and customize and host them on their own start page. They also provide APIs for external developers to build and include new widgets in the widget catalogs.

Referência: (Bezemer, 2009)

- **WS - Web Services**

Português:

Um serviço Web é definido pelo W3C como "um sistema de software projetado para suportar interação máquina-máquina interoperáveis sobre uma rede".

Serviços Web interagem entre si trocando mensagens codificadas usando XML. Em sua forma básica, a arquitetura de serviços Web consiste em um modelo RPC simples em que um cliente invoca operações exportadas por um provedor de serviços usando SOAP. Cada serviço Web publica a sua interface de chamada (por exemplo, endereço de rede, portas operações previstas, e com os formatos esperados de mensagens XML para invocar o serviço), utilizando o WSDL - Web Service Description Language.

Serviços da Web podem ser combinadas de forma colaborativa para criar um novo sistema de serviços Web (WSS), para resolver problemas mais complexos.

Inglês:

A Web service is defined by the W3C as “a software system designed to support interoperable machine-to-machine interaction over a network”.

Web services interact with each other by exchanging messages encoded using XML. In its basic form, the Web service architecture consists of a simple RPC model in which a client invokes operations exported by a service provider using

SOAP, a standard communication protocol for transmitting XML messages. Each Web service publishes its invocation interface (for example, network address, ports, operations provided, and the expected XML message formats to invoke the service) using the Web Service Description Language. The WSDL specification serves as a contract between the client and the server that (in part) defines the valid interactions.
Web services can be combined in a collaborative way to create a new Web services system (WSS) to solve more complex problems.

Referência: (HallÃ¨, 2010) (Mao, 2009)

- **WS-BPEL**

Português:

Mesmo que BPEL4WS ou apenas BPEL. É um exemplo de Linguagem de Execução de Processo de Negócios de composição de Serviços Web.

Inglês:

The Business Process Execution Language for Web Services (BPEL4WS, or WS-BPEL as the new name, abbr. BPEL) is an example of business process programming language for web service composition.

Referência: (Li, 2006)

- **WSMO**

Português:

Sigla para Ontologia para Modelagem de Serviços Web. É um framework que fornece uma plataforma semanticamente enriquecida para documentar, identificar e executar serviços web.

Inglês:

Web Service Modelling Ontology (WSMO) framework. It provides a semantically enriched platform for documenting, discovering and executing SWS

Referência: (Jokhio, 2009)

- **WSC**

Português:

Sigla para Composição de Serviços Web. WSC é uma das idéias mais promissoras relacionadas com os Serviços Web (WSs), onde os desenvolvedores e os usuários podem resolver problemas complexos através da combinação do WSS disponíveis. WSC reduz o tempo de desenvolvimento e provê melhores possibilidades de reutilização. WSC pode ser desenvolvido em dois paradigmas: coreografia e orquestração. Na coreografia, cada serviço envolvido sabe exatamente quando executar suas operações e com quem interagir. A coreografia é mais colaborativa e deixa que cada parte envolvida descreva a sua participação. Na orquestra, um coordenador central possui o controle do WSs envolvidos e coordena a execução de diferentes operações de WS, de acordo com requisitos de orquestração pré-estabelecidos.

Inglês:

Web Service Compositions abreviation. WSC is one of the most promising ideas related to WSs, where developers and users can solve complex problems by combining available WSs. WSC reduces the development time, improves the reuse of WSs and the consummation of complex WSs. WSC can be developed in two paradigms: choreography and orchestration. In choreography, each involved WS knows exactly when it should execute its operations and with whom it should interact. Choreography is more colaborative and let each involved part describes its participation. In orchestration, a central coordinator possesses the control of involved WSs and coordinates the execution of different WS operations, according to pre-established orchestration requirements.

Referência: (Endo et al, 2008)

- **WSDL**

Português:

Descrição de Linguagem de Serviços Web (WSDL) é um formato XML para descrever serviços de rede com informações do orientado a RPC e orientado a mensagem. Programadores ou ferramentas de desenvolvimento automatizado

podem criar arquivos WSDL para descrever um serviço e pode disponibilizar essa descrição através da Internet.

Inglês:
Web Services Description Language (WSDL) is an XML format for describing network services containing RPC-oriented and message-oriented information. Programmers or automated development tools can create WSDL files to describe a service and can make this description available over the Internet.

Referência: (Kumar Ashok, 2009)

- **WSDL-S**

Português:
A especificação WSDL serve como um contrato entre o cliente e o servidor que (em parte) define as interações válidas e provê uma descrição sintática de um serviço. WSDL opera no nível sintático e carece de descrição semântica e contexto do comportamento.

Inglês:
The WSDL specification serves as a contract between the client and the server that (in part) defines the valid interactions and provides the syntactic description of a single service. WSDL operates at the syntactic level and lacks semantic description and behavior context.

Referência: (HallÃ¨, 2010) (Dong, 2009)