# 6
# Discussão

Além das técnicas de teste usando modelos gramaticais, existem outras abordagens de teste funcional de sistemas que estão sendo estudadas pela comunidade científica. Algumas delas se dedicam a validações de modelos, testes em ambientes distribuídos e verificação funcional de componentes de software. O confronto com outras técnicas favorece o entendimento do teste baseado em gramática ao passo que identifica vantagens e desvantagens na sua utilização. Somado a isso, essas técnicas podem tratar de situações não previstas, funcionando como fonte de novas idéias para o seu aprimoramento.

A comparação com outras técnicas promovem uma discussão em prol de possíveis contribuições desta pesquisa para o estudo e a prática de testes funcionais. Este capítulo faz um contraponto com outras abordagens de teste (seção 6.1) e, ao final, identifica as contribuições (seção 6.2) com a disciplina de teste funcional.



## 6.1.
## Teste Baseado em Gramática e Outras Abordagens de Teste

Uma abordagem foi proposta por Atkinson & Groß (2002) para reduzir o esforço de verificação manual de sistemas. A idéia é equipar componentes com testes incluídos na sua codificação. Estes testes tornam os componentes hábeis a verificar sua integridade e seu ambiente em tempo de execução. Quando instalados em novos sistemas, estes componentes verificam a conformidade de seus contratos (assertivas) e verificam automaticamente se suas obrigações podem ser atendidas. Tal técnica é denominada *built-in contract testing*.

A técnica consiste em adicionar uma interface de teste à interface funcional de um componente. Esta interface de teste expõe os estados lógicos de um componente e os tornam acessíveis tanto para sua configuração quanto para sua verificação. Os estados lógicos de um componente são os estados visíveis externamente que o usuário do componente precisa conhecer para utilizá-lo

adequadamente. Os estados são parte da especificação do componente e não podem ser confundidos com o modelo de estado de sua programação. Estes estados estão associados à abstração que o componente representa.

Essa técnica provê uma arquitetura e uma metodologia que são particularmente apropriadas a sistemas distribuídos e sistemas dinâmicos, como aplicações de Internet e sistemas com reconfiguração dinâmica. Uma das dificuldades observadas no teste deste tipo de sistema usando teste baseado em modelos gramaticais, a exemplo do estudo de caso (capítulo 5), foi iniciar os testes a partir de um estado conhecido de um componente. Se este não prover métodos de acesso a sua estrutura, é praticamente "impossível" observar os efeitos produzidos pela execução de suas operações.

Uma arquitetura para verificação e teste baseado em modelo usando uma especificação de perfil de UML foi sugerida por Cavarra et al. (2002). Nesta abordagem, diagramas de classes, objetos e estados são usados para definir modelos essenciais e diretivas de teste.

Os modelos caracterizam possíveis comportamentos de entidades de um sistema em termos de ações e eventos. As diretivas de teste são introduzidas por meio de diagramas de objetos e de estados que definem restrições de teste, critérios de cobertura e propósito de teste (padrão de comportamento).

Os modelos descritos segundo este perfil são convertidos numa ferramenta de linguagem IF – *Intermediate Format* – com descrições que podem ser verificadas e usadas para gerar testes. Diferente do teste baseado em modelos gramaticais, esta abordagem utiliza uma linguagem baseada em máquinas de estados finitos. As duas abordagens tratam de critérios de cobertura funcional, propósitos de teste e restrições aos testes, mas as duas não foram avaliadas quanto ao grau de escala da solução de software proposta.

Um estudo de Javed (2005) indica uma abordagem baseada em gramática usada para validação de diagramas de classes. A técnica envolve a conversão de representações UML numa forma gramatical equivalente. Essa gramática define uma linguagem específica de domínio (DSL – *domain specific language*) que será usada na descrição de casos de uso do sistema.

A conversão do diagrama é feita com o auxílio de ferramentas de transformação de linguagens (ex. diagramas representados em XML – *Extensible Markup Language* – convertidos por meio de XSLT – *Extensible Stylesheet*

*Language Transformations* – para uma DSL). O processo de validação de casos de uso é assistido por ferramentas de desenvolvimento como um tradutor (*parser*), que testa as cadeias de caracteres que o descrevem e verifica se a mesma é valida para a DSL em questão. Uma métrica de comparação de cadeias de caracteres é fornecida pela solução para permitir a identificação de cadeias de caracteres similares válidas. Com base nestas métricas, o usuário modifica o diagrama de classes original de acordo com a funcionalidade desejada. Essa técnica ainda lida com questões como relacionamentos cíclicos e herança múltipla em diagrama de classes.

A técnica citada trata apenas da parte estrutural de um sistema, representada pelos seus diagramas de classes. Enquanto isso, o teste baseado em modelos gramaticais proposto nesta dissertação avalia um escopo mais amplo que abrange o comportamento do sistema e a configuração do ambiente.

O teste de sistemas complexos representa não só uma demanda crescente quanto uma tarefa crítica. Neste contexto, o trabalho de Riebisch & Hübner (2005) considera o uso de modelos para a geração de dados de teste e propõe o seu refinamento para a geração de casos de teste. Este refinamento quando automatizado pode reduzir consideravelmente o esforço e o risco de erros na geração dos testes.

Os requisitos descritos na maioria das vezes por textos em linguagem natural representam a fonte utilizada pelo método desenvolvido por Riebisch & Hübner para a geração de casos de teste. O processo inicia com a definição de casos de uso. As descrições dos casos de uso são transformadas em expressões formais com sintaxe e semântica definidas. Então, sofre um processo de refinamento onde termos genéricos são substituídos por termos mais concretos ou informação detalhada é adicionada à descrição. Sugestões para termos ausentes ou ambíguos são derivadas de um glossário e da análise de sua integração às descrições de casos de uso por intermédio de elos de rastreabilidade e modelos de características (*feature models*). O resultado da formalização é um diagrama de atividades que posteriormente é transformado em diagramas de estado e seqüência. Estas representações descrevem os casos de teste e servem como base para a sua execução.

O mais interessante desta abordagem é a questão do tratamento de texto em linguagem natural que poderia ser adicionada à estrutura de teste baseado em

modelos gramaticais para facilitar a construção dos modelos a partir de descrições de casos de uso.

O estudo de Zheng & Bundell (2007) introduz uma nova metodologia de teste de componente de software baseado em modelos UML. A aplicação da modelagem UML ao teste e ao desenvolvimento de componentes de software permite que as fases de teste e projeto utilizem uma abordagem consistente de especificação. Segundo o trabalho, isto torna mais eficiente e eficaz a produção de componentes de software funcionais e confiáveis.

Seguindo o framework proposto para o desenvolvimento de casos de teste de componentes, são construídos modelos de teste suportados por técnicas de teste de integração de componentes baseado em cenários (*scenario-based component integration testing*) e de teste por contrato (*test by contract*). A seguir, casos de teste de componentes são projetados e gerados a partir dos modelos com apoio de uma técnica de mapeamento de componentes de teste. Esta técnica transforma artefatos UML e contratos de teste em dados de teste para construção de elementos e seqüências de teste dos quais casos de teste e suas especificações são derivados e gerados.

A infra-estrutura do teste baseado em modelos gramaticais gera casos de teste executáveis, não se restringindo somente a suas especificações, como é a situação atual desse trabalho. Embora não use uma estrutura dedicada ao teste por contrato, o mesmo efeito pode ser obtido pelo uso de atributos nos modelos gramaticais. Eles estabelecem condições e restrições à geração e à execução dos casos de teste. Somado a isto, é preciso fazer a extensão de uma nova estratégia de teste compatível com estes atributos.

## 6.2. Contribuições

As representações da estrutura e da funcionalidade do sistema somada à modelagem de seu ambiente, com descrições de fácil entendimento, tão formais quanto práticas, permitem compreender suas características e analisar o seu comportamento. Influenciam ainda na produtividade e na geração e execução de casos de teste.

Por isso, essa dissertação propôs o uso de modelos gramaticais como ponto de referência para realização de testes funcionais de sistemas. As gramáticas que originaram estes modelos puderam ser minimizadas a ponto de não sacrificarem sua semântica. Os modelos são capazes de omitir informação não relacionada aos objetivos de teste e excluir informação inútil ou redundante. Sua legibilidade facilita a compreensão por parte de usuários, desenvolvedores e testadores. Sua estrutura permite mudanças rápidas.

A construção destes modelos é flexível, podendo ser feita de várias maneiras. *Top-down* – construção de gramáticas a partir de especificações de requisitos de sistema. *Bottom-up* – construção de gramáticas de teste a partir de outras gramáticas de teste. *Middle-out* – um misto das maneiras anteriores. Essa flexibilidade resulta em facilidade de adoção e operação.

Os modelos gramaticais ainda estabelecem uma forma estruturada de comunicação entre os atores envolvidos com o teste de sistema, pois são descritos numa linguagem com níveis de concretude variável segundo a necessidade de avaliação. Os modelos gramaticais definem uma visão consistente do critério de teste e com isto permitem a sistematização da geração de casos de teste. Estratégias e heurísticas de teste podem ser desenvolvidas com base nelas a fim de especificar o critério de parada e o critério de adequação dos testes.

O uso do paradigma orientado a objetos na representação computacional dos modelos permitiu simplificar a notação utilizada na sua forma escrita e, com isto, propiciou um novo estágio de pesquisa em testes funcionais baseados em modelos.