1

Introdução

Atualmente, grande parte das atividades do cotidiano estão relacionadas direta ou indiretamente ao uso de software. Seja numa conversa ao telefone celular, no uso de um cartão de transporte público, ou no pagamento eletrônico de uma compra, o software é fundamental para que estas atividades ocorram rapidamente e com sucesso.

Por outro lado, é notório o fato de que o desenvolvimento de software muitas vezes não consegue entregar um produto final que atenda aos seus requisitos de forma satisfatória, devido à complexidade crescente dos requisitos dos sistemas atuais. É grande a incidência de defeitos na criação de software, que vão desde a análise e especificação até a implementação do código.

Em virtude desta realidade, existe uma busca contínua pelo aperfeiçoamento da maneira de se produzir software, para que seja possível lidar com a complexidade dos sistemas e com as restrições de custo e prazo dos projetos. A crescente demanda por software de qualidade, nem sempre acompanhada pela oferta de profissionais bem qualificados, exige que se aumente a produtividade das equipes através da adoção de técnicas e ferramentas que auxiliem o processo de desenvolvimento e, conseqüentemente, promovam uma redução de defeitos no produto final.

Entre as técnicas que vêm sendo adotadas na busca pelo aumento da qualidade e produtividade das equipes de desenvolvimento está o uso de componentes de software. Gradativamente, novos padrões e ferramentas de desenvolvimento com suporte ao uso de componentes têm surgido, com o objetivo de facilitar a modelagem e a implementação de sistemas complexos [1].

Além do uso de componentes, uma outra técnica que tem sido objeto de pesquisa neste mesmo contexto é o uso de contratos de software. Contratos são, em essência, mecanismos para expressar formalmente as interfaces de

objetos ou componentes de software, favorecendo a verificação continuada de sua correção e aumentando o rigor de sua especificação.

O uso de contratos de software baseia-se na metáfora de contratos legais, em que as partes envolvidas possuem direitos e deveres a serem observados. No contexto da interface de um objeto, um contrato especifica quais as condições que devem ser previamente satisfeitas para que uma operação possa ser invocada (pré-condições), e quais devem ser garantidas pelo objeto ao final da execução da operação (pós-condições). Ainda na analogia com contratos legais, as pré-condições representam as obrigações do cliente e os direitos do objeto sendo chamado. Por outro lado, as pós-condições significam as obrigações que um objeto deve cumprir de sua parte, garantindo ao cliente o direito de contar com as premissas por elas definidas.

Outro aspecto importante no uso de contratos são as invariantes, cláusulas que devem ser satisfeitas durante o tempo de vida do objeto. As invariantes devem ser verificadas na chamada de uma operação do objeto e no seu retorno. É possível que esta seja violada de forma transiente durante o processamento da operação invocada, porém ela deve ser novamente garantida ao retornar o controle para o cliente da chamada.

O desenvolvimento de software baseado em componentes (ou CBSD - *Component Based Software Development*) é uma disciplina que pode muito se beneficiar do uso de contratos, uma vez que a divisão de papéis entre provedor e cliente de componentes aumenta a necessidade de se garantir uma especificação mais formal para o seu uso correto. O uso de contratos também auxilia na documentação correta do componente, pois a geração da documentação a partir da especificação dos contratos diminui a chance de esta se tornar ambígua ou desatualizada.

Clemens Szyperski, um importante pesquisador na área de componentes, define que um componente de software é “uma unidade de composição que possui interfaces especificadas contratualmente e dependências de contexto explícitas” [1], o que suporta o argumento a favor do uso de contratos com componentes.

Apesar de representar uma técnica importante no auxílio à qualidade do produto final em projetos de software, o uso de contratos não é uma prática muito comum, principalmente no âmbito corporativo. No contexto acadêmico, existem diversos trabalhos sobre o tema, porém são poucas as linguagens, *frameworks* e *middlewares* que oferecem ao desenvolvedor recursos para o uso de contratos. Entre os ambientes e linguagens de sucesso comercial, é

praticamente inexistente o suporte nativo ao uso de contratos.

O uso de componentes de software está intimamente ligado ao conceito de reúso, que é uma das suas principais motivações. O uso de contratos contribui para garantir que um componente funcione corretamente com diversos clientes, o que é uma premissa para o reúso ser bem sucedido. Quando um componente é desenvolvido e testado sem que seus contratos sejam devidamente verificados, maiores são as chances de que alguns defeitos possam passar despercebidos, sendo descobertos mais tarde no ambiente de produção, quando este é utilizado em condições distintas daquelas em que foi testado.

Ao definir contratos para as operações do componente, é necessário formalizar seu comportamento, o que exige um esforço maior por parte dos desenvolvedores. Porém, em geral esse esforço é compensado pela prevenção e correção de defeitos, além de promover uma elaboração mais cuidadosa das interfaces implementadas pelos componentes e uma análise mais profunda dos seus requisitos, relacionamentos e comportamentos.

O uso de contratos possui diversos aspectos que podem ser objeto de pesquisa, tais como impactos de desempenho, dificuldades associadas à manutenção, questões relacionadas ao salvamento do estado do objeto para verificações de pós-condições, descrição dos estados do objeto e suas transições, entre outros. Para cada um destes aspectos, surgem novos desafios quando os contratos se aplicam ao contexto de desenvolvimento em ambientes distribuídos, em que a concorrência quase sempre está presente e impõe uma série de dificuldades à verificação de contratos.

Embora haja diversos trabalhos sobre o uso de contratos e sistemas de componentes de software, poucos abordam o tema sob o ponto de vista do desenvolvedor, ou seja, em geral são apresentadas as características gerais do sistema ou *framework* criado sem entrar nos detalhes do seu uso e sem mostrar como os contratos devem ser criados, em que situações eles são mais ou menos úteis, o que deve ser evitado, quais são as limitações existentes, além das particularidades da criação de contratos em ambiente distribuído.

Portanto, o uso de contratos em sistemas baseados em componentes representa um tema de pesquisa relevante dentro do contexto atual da disciplina de Engenharia de Software, especialmente para Sistemas Distribuídos.

1.1

Objetivos

O tema desta dissertação consiste na avaliação do uso de contratos em sistemas de componentes de software distribuídos, baseado na argumentação descrita na seção anterior. Para viabilizá-lo, construímos um subsistema de suporte a contratos baseado na plataforma do OiL [2], que é uma implementação do padrão CORBA [3] na linguagem Lua [4, 5]. Esta ferramenta serve de apoio à dissertação, fornecendo insumos para a exposição das idéias que são tema deste trabalho e sua argumentação.

O sistema de componentes SCS [6] também foi utilizado no desenvolvimento da pesquisa. Ele implementa um modelo simplificado de componentes, baseado no CORBA CCM [7] e no COM [8], e oferece uma arquitetura de desenvolvimento de aplicações bem definida, que inclui mecanismos de introspecção, conexão entre componentes e um ambiente de execução baseado em contêineres, que por sua vez promovem a padronização dos serviços de carga, execução e informação dos componentes. O SCS possui implementações em Lua/OiL e Java.

Durante o trabalho realizado, procuramos investigar os principais aspectos relacionados ao uso de contratos em sistemas distribuídos baseados em componentes, tais como: benefícios, dificuldades, limitações e desafios. São abordados ao longo do texto tópicos como a avaliação distribuída de contratos, concorrência na avaliação, elaboração de contratos de componentes, sincronização de chamadas, herança de contratos, entre outros. Realizamos também alguns experimentos para avaliar os aspectos citados, e que serão descritos mais adiante no texto.

O estudo proposto neste trabalho está voltado para os aspectos dinâmicos do uso de contratos e sua avaliação em tempo de execução. Portanto, a análise estática de contratos não foi incluída no escopo do trabalho.

1.2

Contribuições

A principal contribuição do trabalho realizado é a avaliação do uso de contratos no desenvolvimento de sistemas baseados em componentes, através da discussão sobre as principais características, requisitos, dificuldades e li-

mitações existentes. Os experimentos realizados ajudam a ilustrar os pontos abordados durante o texto, dando um caráter prático ao estudo realizado.

Adicionalmente, os aspectos do uso de contratos tratados nesta dissertação servem de insumos para avaliar as vantagens e desvantagens da adoção de técnicas mais formais na especificação e implementação de componentes de software. Como a adoção de contratos ainda não é uma prática tão comum, as idéias aqui apresentadas contribuem para promover a discussão em torno do tema e melhorar sua compreensão, além de inspirar trabalhos futuros.

Um dos produtos do trabalho desta dissertação é a criação de um subsistema de suporte ao uso de contratos em Lua, seja para objetos Lua locais, objetos remotos criados na plataforma OiL ou componentes SCS. Devido à sua simplicidade, dinamismo e eficiência, a linguagem Lua tem sido amplamente utilizada nos ambientes acadêmico e corporativo para o desenvolvimento de protótipos e sistemas de pequeno porte, além do seu uso mais tradicional como uma linguagem de extensão. Com isso, o subsistema de contratos criado aumenta a oferta de ferramentas para auxiliar o desenvolvimento em Lua, além de servir de base para experimentos e trabalhos acadêmicos em disciplinas relacionadas ao tema. Alguns detalhes da implementação da ferramenta foram incluídos nesta dissertação e contribuem para enriquecer o seu conteúdo. Por outro lado, as limitações identificadas na ferramenta também serão tema de discussão ao longo do texto.

1.3

Estrutura do Documento

O capítulo 2 apresenta os principais conceitos associados a contratos de software, sua classificação de tipos, além de aspectos do seu uso com objetos e componentes.

O capítulo 3 descreve alguns trabalhos relacionados ao tema e suas principais características, para contextualizar o estudo realizado em relação à pesquisa recente na área.

O capítulo 4 descreve o sistema de suporte a contratos desenvolvido, suas principais características, detalhes de implementação, além das restrições, dificuldades e decisões do projeto.

O capítulo 5 contém alguns experimentos com aplicações baseadas em

objetos e componentes distribuídos utilizando contratos, servindo de insumos para o tema proposto neste trabalho. São avaliados alguns aspectos práticos do desenvolvimento neste ambiente, além de questões sobre desempenho, dificuldades de implementação, resultados obtidos, limitações, prós e contras.

O capítulo 6 apresenta as conclusões do trabalho realizado, suas contribuições, principais desafios, resultados atingidos, além de possíveis extensões e trabalhos futuros.