O uso de arquitetura orientada a componentes no desenvolvimento de uma plataforma de análise sintática

Guilherme W. S. Lopes[1], Andreia G. Bonfante[1], Cláudia A. Martins[1], Josiel M. Figueiredo[1]

[1]Instituto de Computação – Universidade Federal do Mato Grosso (UFMT)
Cuiabá – MT - Brasil

gwslopes@gmail.com, {andreiab,claudia,josiel}@ufmt.br

Abstract. *This article describes a syntactic analysis environment based on SCA (service-component architecture). The prototype intends to be a testing platform of sentence processing, grammar learning and parsing algorithms.*

Resumo. *Este artigo descreve um protótipo de uma plataforma de análise sintática automática cujo desenvolvimento segue os padrões de arquitetura orientada a componentes e a serviços. A proposta é que a plataforma seja usada como um ambiente de testes de algoritmos envolvidos no processo, seja de preparação de sentenças, de aquisição de gramáticas, de análise sintática e de testes e avaliação, e que seja flexível o suficiente para aceitar algoritmos implementados em diversas linguagens.*

1. Introdução

As arquiteturas orientadas a componentes (SCA) estão sendo cada vez mais utilizadas com o advento dos sistemas distribuídos [Braga et al 2009]. Essas arquiteturas possibilitam o desenvolvimento de aplicações dos mais diversos tipos. A arquitetura também permite que as aplicações possam conversar entre elas e com outras arquiteturas, utilizando algum protocolo de comunicação, como por exemplo, SOAP (*Simple Object Access Protocol*) utilizado para comunicação com serviços web e Java RMI (*Remote Method Invocation*) [Senturier et al 2009].

Neste trabalho, o uso de tal tecnologia está sendo explorado na construção de um ambiente de preparação de sentenças, aprendizado de gramáticas e testes para análise sintática automática. Os algoritmos podem ser combinados das mais diversas formas, sendo que o encadeamento necessário fica implícito. Assim, por exemplo, qualquer usuário que queira testar um algoritmo específico, como por exemplo, de aquisição de gramática, poderá incluí-lo na plataforma, mantendo os outros algoritmos de análise sintática e de testes e avaliação de resultados os mesmos já implementados. Essa flexibilidade permite que o algoritmo seja implementado em diferentes linguagens de programação, bastando apenas que sejam obedecidos os padrões de comunicação.

Nas seções seguintes são apresentados o referencial teórico utilizado para a elaboração deste trabalho (seção 2) e a descrição do protótipo já implementado (seção 3). Na seção 4 são apresentadas a conclusão e os trabalhos futuros.

2. Referencial Teórico

A arquitetura orientada a componentes (*Service Component Architecture* - SCA) é uma alternativa para a arquitetura orientada a serviços. Ela é uma padronização para composição e implantação de aplicações, permitindo que sejam empregadas tecnologias distintas em sua implementação [Chapell 2007] [Senturier 2009]. Curbera (2007) define

uma arquitetura orientada a componentes como uma representação de um componente de serviço reutilizável que encapsula a lógica de negócio que suporta um ou mais serviços.

Como é mostrado na Figura 1, uma aplicação SCA pode ser acessada por outro software que não seja SCA, como um serviço Web (na figura, descrito como "Cliente"), ou qualquer outro modelo de software. As aplicações também podem acessar e gravar dados em bancos de dados (na figura, descrito como "DBMS", *Database management system* ou Sistema gerenciador de banco de dados). As composições (*composites)* são descritas através de uma linguagem de descrição (*Service Configuration Description Language*-SCDL) em um arquivo (baseado em XML) que especifica como os componentes irão se relacionar (na figura, descrito como "Configuração SCDL").

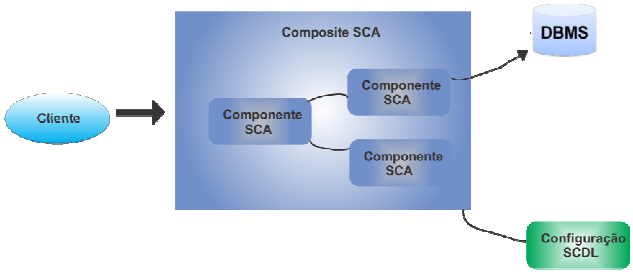

Figura 1. Exemplo de uma composição (*composite)* formado por três componentes conectados a um banco de dados [adaptado de Chappel 2007]

O protótipo que está sendo desenvolvido utiliza a implementação Apache Tuscany[1], um servidor que implementa SCA. A escolha deve-se ao fato de que ele fornece uma infraestrutura de código aberto que visa facilitar o desenvolvimento, montagem, edição e gerenciamento de aplicações compostas e de processamento de dados [Chu et al. 2009].

3. O protótipo

Os componentes estão sendo implementados na linguagem de programação Java e para a programação, o ambiente Eclipse está sendo usado. Foram criados 3 componentes, sendo eles: componente de processamento do texto; componente de conexão com o banco de dados e componente para gerenciar os arquivos intermediários criados durante a execução dos processos.

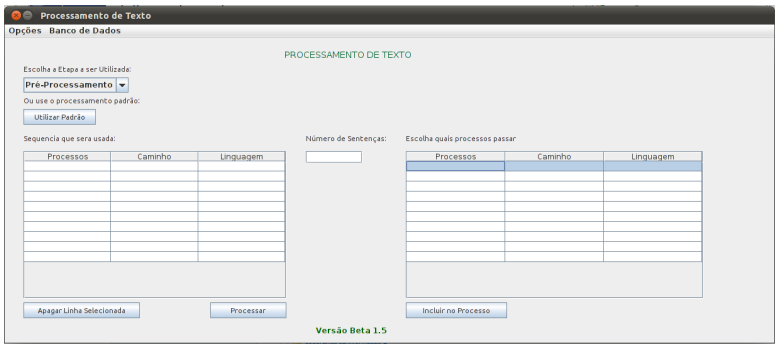

Figura 2. Interface do protótipo

Na Figura 2 é mostrada a interface desenvolvida até a elaboração deste artigo. Como é um ambiente de testes, a plataforma permite que sejam encadeados/combinados

1 http://tuscany.apache.org/

todos os algoritmos de um processo de análise sintática, desde os envolvidos na preparação de sentenças, até os de aquisição automática de gramática, análise sintática e avaliação. Isto permite que sejam usados diversos algoritmos em diferentes momentos.

Os processos (algoritmos) utilizados no processamento são armazenados em um banco de dados PostgreSQL e podem ser cadastrados através do menu “Banco de Dados”. Estes processos podem ser cadastrados em linguagens como Perl e Python, sendo que outras linguagens podem ser adicionadas em versões futuras.

O usuário também pode salvar a sua sequência de processamento utilizando o menu de “Opções” caso este queira utilizá-la de novo em outro texto.

4. Conclusão e Trabalhos Futuros

Este artigo apresentou a proposta de elaboração de uma plataforma de testes para algoritmos relacionados à análise sintática automática de sentenças. Tal plataforma trará maior versatilidade aos usuários que queiram testar algoritmos específicos, sem a necessidade de implementar todo o processo de análise, uma vez que a plataforma disponibilizará a integração de diversos algoritmos, possibilitando reutilizar outros códigos. A flexibilidade também ocorrerá em termos de linguagens de programação, uma vez que se pretende definir protocolos de comunicação entre diversas linguagens dentro da plataforma. Como trabalho futuro pretende-se finalizar a plataforma e adicionar um banco de dados contendo uma *treebank* que servirá de base para os testes, e ainda alguns algoritmos de preparação de sentenças, aquisição de gramáticas, análise sintática e testes.

Referências

Braga, E. et al. (2009) “Web Services: Conceitos e Práticas de Desenvolvimento de Aplicações Distribuída”. Luziania: Centro Universitário de Desenvolvimento do Centro-Oeste.

Chappell, D. (2007) “Introducing SCA”, <http://www.davidchappell.com/ articles/Introducing _SCA.pdf/>, julho.

Curbera, F. (2007) Component Contracts in Service-Oriented Architectures. In: IEEE Computer Society, vol 40, Nº11, pg 74-80, Nov. 2007.

Chu, Q.; Shen, Y.; Jiang, Z. (2009). A Transaction Middleware Model for SCA Programming. In: First International Workshop on Education Technology and Computer Science, vol.3, pg 568-571.

Seinturier, L. et al. (2009). Reconfigurable SCA Applications with the FraSCAti Platform. In: *IEEE International Conference on Services Computing*, pp 268-275.