Balanceamento de Carga Autonômico em Sistemas Distribuídos

Josimar Sfreddo, Marcia Pasin

Programa de Pós-Graduação de Informática – Universidade Federal de Santa Maria
97.015-900 – Santa Maria – RS – Brasil

josimarbs@gmail.com, marcia@inf.ufsm.br

1. Introdução

Em 2001, um manifesto da IBM [Horn 2001] alertou sobre uma dificuldade de gerenciamento de sistemas complexos como um obstáculo para inovações em TI. Como forma de tratar esse problema, a IBM propôs o conceito de computação autônoma, onde a principal propriedade é o auto-gerenciamento. Auto-gerenciamento é desempenhado pelo sistema computacional através da garantia de execução de regras que implementam políticas de alto nível estabelecidas por administradores. Para isso, é necessário que o sistema implemente as propriedades *auto-configuração, auto-proteção, auto-otimização* e *auto-cura*, definidas anteriormente em [Kephart e Chess 2003]. Acreditamos que os sistemas computacionais podem tomar proveito dessas propriedades para melhorar o desempenho em relação a mudanças de cenário (caracterizadas por falhas, picos de carga, momentos de ociosidade, atualizações, etc.) de um sistema distribuído. Em direção à autonomia, este trabalho propõe um balanceador de carga adaptativo para aplicações distribuídas. A propriedade inicialmente enfocada nesta proposta contempla a *auto-otimização.*

2. Trabalhos relacionados

Uma visão interessante sobre computação autonômica e exemplos de sistemas é apresentado em [Corrêa e Cerqueira 2009]. Trabalhos recentes, que aplicam técnicas autonômicas para balanceamento de carga, enfocam diferentes propriedades do auto-gerenciamento. [Ewing e Menascé 2009] apresentam um projeto de um balanceador de carga autonômico usando uma página de leilão. O sistema utiliza *auto-configuração* para priorizar pedidos que sejam mais favoráveis na realização de lances e a *auto-otimização* para alocar mais recursos do sistema distribuído para o grupo de pessoas que estão executando mais lances durante o leilão. [Badonnel e Burgess 2008] aborda *auto-otimização*. Tarefas são dispostas em uma fila em um nodo centralizador, e através da técnica *pull-based*, nodos solicitam ao centralizador nova tarefa assim que a tarefa em execução é finalizada. [Rossi 2008] aborda *auto-otimização* através de uma modificação no escalonador de recursos do virtualizador *Xen* baseado em uma *data warehouse* que armazena informações geradas por diversos *benchmarks*.

3. Balanceamento de carga autonômico

Nossa proposta de balanceador de carga autonômico baseia-se na solução de [Badonnel e Burgess 2008], que usa a técnica *pull-based*, onde um centralizador divide tarefas entre os servidores. Acreditamos que este modelo pode obter melhores resultados, se o

centralizador for capaz de classificar e priorizar processos urgentes, como em [Ewing e Menascé 2009]. Durante o processamento, é possível armazenar informação no centralizador sobre a execução do serviço nos demais nodos. Essa informação, combinada com eventos coletados pelo sistema (ocupação da CPU, memória, ocorrência de falhas, etc), é analisada por um esquema de regras que implementam políticas de alto nível pré-definidas pelo administrador, utilizando a biblioteca *Java drools* [Drools 2010] ou ferramenta similar. Com suporte da correlação de eventos, a distribuição de tarefas pode ser implementada por um mecanismo adaptativo, que promove a *auto-otimização* e aloca tarefas de forma a aproveitar melhor os recursos disponíveis. Esse mecanismo busca evitar situações como a alocação uma tarefa complexa para um nodo com baixa capacidade de processamento (se um ambiente heterogêneo é considerado).

Otimização é a propriedade a ser implementada inicialmente. O acréscimo de outras propriedades como a *auto-configuração*, não está descartado. *Auto-configuração* possibilita que nodos possam ser incluídos ou removidos dinamicamente no sistema distribuído promovendo maior disponibilidade e escalabilidade.

4. Conclusão

Computação autonômica possibilita a adequação automática de sistemas computacionais a novas condições (ocupação da CPU, memória, recursos de rede, etc), com o objetivo de manter continua disponibilidade e alto desempenho. Trabalhos atuais não implementam todas as propriedades de auto-gerenciamento. A implementação dessas propriedades não é uma tarefa trivial.

A construção de sistemas autonômicos requer conhecimento sobre o escopo do serviço e infraestrutura computacional. Usando essas informações, é possível definir um conjunto de políticas, mapeadas em regras, que possam tornar o recurso autonômico mais eficaz.

Referências

Badonnel R., Burgess M. (2008) “Service Load Balancing with Autonomic Servers: Reversing the Decision Making Process”. In: Proceedings of AIMS 2008. LNCS 5127. p. 92-104.

Corrêa, S., Cerqueira, R. (2009) “Computação Autonômica: Conceitos, Infraestruturas e Soluções em Sistemas Distribuídos”, In: Capítulo 4 - Minicurso 27º SBRC.

Drools 5 - The Business Logic integration Platform (2010). http://www.jboss.org/drools (Acesso em dezembro de 2010)

Ewing, J. M., Menascé, D. A. (2009) Business-Oriented Autonomic Load Balancing for Multitiered Web Sites. In: Proceedings of MASCOTS 2009, London, UK.

Horn, P. (2001) “Autonomic Computing: IBM's Perspective on the State of Information Technology”; Technical Report, IBM Corporation, October 15, 2001.

Kephart, J. O., Chess, D. M. (2003) “The Vision of Autonomic Computing”. Cover Feature, IEEE Computer Society, January 2003.

Rossi, F. D. (2008) “Alocação Dinâmica de Recursos do XEN”. Dissertação de Mestrado. Pontifícia Universidade do Rio Grande do Sul. 70p.