Um modelo hierárquico para coleta de dados de monitoramento em sistemas distribuídos

Rafael Keller Tesser[1], **Philippe O. A. Navaux**[1]

[1]Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)
Caixa Postal 15.064 – 91.501-970 – Porto Alegre – RS – Brazil

{rktesser,navaux}@inf.ufrgs.br

1. Introdução

Através do monitoramento pode-se obter informações como o estado atual de componentes de um sistema distribuído, dados sobre a execução de programas no mesmo e sobre o funcionamento da rede de interconexão. Estas informações podem auxiliar na realização de uma grande variedade de tarefas. Algumas destas são: detecção de falhas, escalonamento, análise de desempenho e adaptação da execução de programas. Um tipo especial de sistema distribuído que pode também beneficiar-se do monitoramento é o *Grid* [Foster et al. 2001]. No entanto, uma ferramenta que proponha-se a realizar o monitoramento de tal sistema necessita lidar com suas características inerentes. Sua grande escala, dinamicidade quanto aos recursos disponíveis, heterogeneidade e distribuição geográfica devem ser levados em conta no projeto.

Uma dificuldade essencial do monitoramento de sistemas distribuídos é que as informações também estão distribuídas. Por isso é necessário um mecanismo para a coleta desses dados. Uma abordagem para resolver esse problema é a utilização de uma estrutura hierárquica formada por componentes que forneçam acesso a informações obtidas por um conjunto de componentes de nível hierárquico inferior. A ferramenta Ganglia [Massie et al. 2004] e o *Monitoring and Discovery Service* (MDS) [Czajkowski et al. 2001], que faz parte do *Globus Toolkit*, utilizam esse tipo de estrutura. O Ganglia utiliza o modelo *pull* (sob demanda) de recebimento de dados, utilizando *polling* para detectar a disponibilidade de novas informações. No entanto, para o propósito de acompanhar o comportamento de um conjunto de objetos por um período de tempo, é mais interessante utilizar o modelo *push*. Dessa forma os dados são enviados para os interessados, assim que tornam-se disponíveis ou que alguma condição for satisfeita, como a acumulação de uma certa quantidade de informação ou a passagem de um determinado intervalo de tempo desde o último envio. O MDS versão 4 suporta o modelo *push*, mas é mais otimizado para tarefas como descoberta ou envio de alertas do que para tarefas como o acompanhamento do comportamento de componentes ou da execução de programas.

2. Modelo hierárquico

Tendo em vista o problema abordado na seção anterior, propõe-se um modelo hierárquico de coleta de dados de monitoramento distribuídos, utilizando o modelo *push* de recebimento de dados. Esse modelo estende o DIMVisual [Schnorr et al. 2006], que é um modelo de integração de dados de monitoramento para visualização. O DIMVisual, que possui um protótipo implementado, recebe informações de monitoramento relacionadas a diversos objetos e coletados por diferentes ferramentas. Esses são unificados e padronizados, gerando dados em um formato de entrada de uma ferrramenta de visualização.

No DIMVisual original, o usuário precisa reunir localmente as informações de monitoramento, que estão espalhadas por todo um sistema distribuído, para que se possa fornecê-las ao modelo. A extensão aqui proposta visa automatizar esse processo. Se possível, também pretende-se adicionar ao DIMVisual a capacidade de realização de monitoramento *on-line*. O modelo proposto é composto por dois tipos principais de componentes chamados *coletores* e *agregadores* e por um cliente. O cliente passa a ser o responsável por realizar o processo de integração dos dados. Um coletor é um componente que obtém dados de monitoramento de uma fonte e os envia a um conjunto de agregadores. Os agregadores formam uma estrutura hierárquica e são responsáveis por receber informações de monitoramento de coletores e de agregadores de nível inferior e repassá-los a agregadores de nível superior ou a clientes. Para diminuir o envio de dados desnecessários aos clientes, é utilizado um mecanismo de subscrição. Cada cliente tem acesso a informações sobre os coletores acessíveis através de um agregador. Esses dados contém uma identificação única, e uma identificação do componente monitorado, para cada coletor. Para a o fornecimento desses dados, os coletores devem registrar-se nos agregadores antes de poderem enviar dados a eles. Essas informações são repassadas pelos agregadores, até chegarem ao topo da hierarquia. Nesse processo também é registrado qual agregador repassou o registro, a fim de poder-se percorrer o caminho até o coletor. Utilizando as informações de registro do coletor, o cliente faz solicitações de subscrição aos dados gerados pelos coletores de interesse. Essas são repassadas através da hierarquia até encontrar-se um agregador que já possua a subscrição ou então o próprio coletor. Quando possui dados a enviar, um coletor consulta suas subscrições e os envia aos agregadores subscritos. Um agregador, por sua vez, possui uma lista de subscrições para cada coletor. Dessa forma ele pode determinar para quem deve repassar os dados recebidos.

Está sendo implementado um protótipo do modelo proposto, utilizando a linguagem Objective-C. Nesse caso, os componentes são objetos distribuídos, e o envio de informações de monitoramento, solicitações de subscrição e informações registro é feito utilizando o envio de mensagens (chamadas de métodos) a esses objetos. Pretende-se realizar testes com esse protótipo para validar o modelo. Tem-se interesse em avaliar sua utilização em *Grid*.

Referências

Czajkowski, K., Kesselman, C., Fitzgerald, S., and Foster, I. (2001). Grid information services for distributed resource sharing. In *HPDC '01: Proceedings of the 10th IEEE International Symposium on High Performance Distributed Computing*, page 181, Washington, DC, USA. IEEE Computer Society.

Foster, I., Kesselman, C., and Tuecke, S. (2001). The anatomy of the Grid: Enabling scalable virtual organizations. *International Journal of Supercomputer Applications*, 15(3):200–222.

Massie, M. L., Chun, B. N., and Culler, D. E. (2004). The ganglia distributed monitoring system: Design, implementation, and experience. *Parallel Computing*, 30(7):817–840.

Schnorr, L. M., Navaux, P. O. A., and de Oliveira Stein, B. (2006). DIMVisual: Data integration model for visualization of parallel programs behavior. In *Proceedings of the 6th IEEE International Symposium on Cluster Computing and the Grid*, pages 473–480, Los Alamitos, CA, USA. IEEE Computer Society.