

MoonGrid: Monitoramento de Recursos em Ambientes de Grade

Daniel da Trindade Lemos Everton Stadolni Patrícia Kayser Vargas
Centro Universitário La Salle (UNILASALLE) Av. Victor Barreto, 2288 – Canoas, RS, Brasil
{daniel.mutley, stadolnieverton}@gmail.com, kayser@unilasalle.edu.br

Abstract

Obtaining and processing resource monitoring information in a grid computing environment presents challenges related to heterogeneity and scalability. Such information can be used mainly to help discovery, task scheduling, and management tools. Analyzing the available monitoring tools, we can observe that many of them obtain information concerning CPU and memory utilization, however, few obtain detailed software information. This paper presents an overview of MoonGrid model, which monitors hardware and software resources while deals with scalability and heterogeneity. A prototype was implemented for Windows platform, and it was tested in Banrisul labs showing promising results. This text also presents our ongoing works.

1. Introdução

A necessidade de que máquinas de instituições distintas compartilhem seus recursos computacionais através de uma rede de computadores integrada é a idéia chave por trás da Computação em Grade (*Grid Computing*) [3, 2]. Em grades existem potencialmente uma quantidade grande de usuários, recursos e aplicações sendo administradas por diferentes domínios administrativos.

Um dos serviços essenciais em uma infra-estrutura de grade é o monitoramento. Segundo Jain [7] um monitor é uma ferramenta utilizada para observar as atividades de um sistema. Em geral, os monitores observam o desempenho do sistema, coletam estatísticas de desempenho, analisam dados e mostram resultados. O monitoramento serve tanto para permitir uma melhor gerência do ambiente quanto para permitir a implementação de mecanismos de tolerância a falhas que precisam conhecer o estado do sistema. Em um ambiente de computação em grade, a coleta e o processamento de informações de monitoramento de recursos apresenta desafios relacionados a grande heterogeneidade e questões de escalabilidade. Existem muitos trabalhos relacionados a monitoramento de recursos e vários específicos para grade. Quase todos os *middlewares* mais re-

ferenciados possuem monitores associados. Assim, existem várias ferramentas disponíveis para monitoramento, como por exemplo Netlogger [11] e Gridbus [5], bem como para visualização de dados de monitoramento como o Monalisa [9]. Em particular, Ganglia [4], Hawkeye-Condor [6] e WebMDS-Globus [14] são ferramentas muito citadas na literatura. Pode-se observar nestes trabalhos alguns pontos que poderiam ser melhorados para a obtenção de uma ferramenta de uso geral, e que são a motivação deste trabalho.

Este texto apresenta um ambiente de monitoração denominado MoonGrid (*Monitoring on Grid resources*) [1, 8]. Este trabalho está sendo desenvolvido dentro do contexto do modelo GRAND [12]. O GRAND é um modelo de gerenciamento hierárquico de aplicações em ambiente de grade, cujo protótipo AppMan tem mostrado resultados promissores [13]. Um dos objetivos principais do MoonGrid é obter e publicar o máximo possível de informações de hardware, bem como disponibilizar informações de software, de modo que esses diferentes tipos de informações e seus detalhes permitam um uso geral. Com isso deseja-se permitir que tanto que escalonadores, e outros serviços da infra-estrutura de grade, quanto administradores de rede e gerentes possam utilizar essas informações. Também é objetivo que o serviço contemple monitoração tanto de aspectos estáticos quanto dinâmicos de hardware e software para um ambiente de grade, também para auxiliar escalonadores de recursos e administradores de ambiente em grade. O restante deste texto apresenta alguns detalhes do MoonGrid (Seção 2) e as considerações finais (Seção 3).

2. MoonGrid: Visão Geral

O MoonGrid é um modelo de monitoramento hierárquico em três níveis. Em cada máquina um **Sensor** é instalado para coletar os dados do recurso. A partir do momento que a máquina é ligada, o Sensor começa a monitorar os softwares usados e a coleta dos dados referentes a hardware. Muitas destas informações são estáticas, ou seja, não alteram sem ter que desligar a máquina e portanto são coletadas uma única vez. Pode-se identificar três etapas do envio de informações, relacionadas a cada um dos componentes do MoonGrid: (1) coleta da informações de soft-

ware e hardware localmente pelo componente **Sensor** e posterior envio para os Coletores Locais; (2) dados coletados nos **Coletores Locais** são replicados em um servidor de banco de dados, associado a um Coletor Global que processa e guarda as informações globais; (3) requisição de dados armazenados nos **Coletores Globais**, através de consultas dinâmicas.

Os dados monitorados podem ser divididos em duas grandes categorias: os relacionados a hardware e os relacionados a software. Os dados sobre hardware tendem a ser mais estáticos enquanto os de software mais dinâmicos.

O Sensor faz a coleta dos dados localmente na máquina, após é feito o processamento antes do envio das informações ao Coletor Local. Para descentralizar o processamento das informações, a fim de diminuir o consumo de recursos de memória principal e secundária dos coletores, diminui-se o volume de informação a ser processada ao enviar apenas dados novos e/ou atualizados. Assim, o MoonGrid busca obter informações de software e hardware de forma pouco intrusiva e utilizando pouca banda, não sobrecarregando a rede no envio das informações coletadas localmente. Busca-se minimizar a intrusão pela obtenção seletiva de dados já citada. Para evitar a sobrecarga da rede, as informações serão enviadas em tempos distintos. Cada nó enviará em um determinado período para que auxilie e não entre em conflito com processos mais importantes. Além disto, as informações são enviadas por completo somente na primeira coleta, e posteriormente são enviadas somente as alterações ou atualizações. Mais detalhes sobre o modelo podem ser encontrados em [1].

Um protótipo foi desenvolvido com o Microsoft .Net Framework 1.1, da Microsoft, e a linguagem C#. O Sensor foi implementado como um serviço instalado localmente no nó. A partir do momento que a máquina é ligada o Sensor passará a monitorar o equipamento. Os dados são armazenados em arquivos XML para que no momento do envio eles possam ser comparados com os atuais antes do envio, é feita a comparação e atualizados o arquivo XML com as alterações e mantendo sempre atualizados com as últimas informações. Experimentos realizados em uma máquina Pentium 4 de 3,2 GHz com 512 Mbytes[8], indicam que o tempo levado para a coleta completa das informações no Sensor foi entre 2 e 4 segundos, e para uma coleta periódica de dados o tempo ficou entre 0 e 1 segundos.

3. Conclusões e Trabalhos Futuros

O modelo proposto foi implementado e um protótipo avaliado em um ambiente real: a rede de teste do Banrisul. Com estes experimentos, foi possível avaliar diversos aspectos conforme descrito em [1]. Os resultados foram pro-

missores e otimizações estão em andamento. Isto está estimulando diversos trabalhos em andamento.

Uma dessas atividades em andamento é o porte do protótipo para o ambiente Linux utilizando o *framework* Mono [10] e o desenvolvimento da classe para obtenção dos dados através do /proc. Isso possibilitará a avaliação do MoonGrid em um ambiente mais heterogêneo, tal como os tipicamente existentes em instituições de pesquisa. Outro trabalho a ser realizado é a utilização das informações obtidas pelo MoonGrid no protótipo AppMan. A integração das informações de monitoramento obtidas pelo MoonGrid no AppMan permitirá a realização de um escalonamento mais eficaz.

Alguns aspectos do modelo não estão sendo tratados neste momento e foram deixados para trabalhos futuros. Em primeiro lugar, seria útil uma interface para visualização dos dados monitorados, fornecendo relatórios por características ou permitindo que o usuário faça sua própria consulta.

Referências

- [1] D. da Trindade Lemos. *Monitoramento de Recursos em Ambientes de Grade*. Trabalho de conclusão de curso, Centro Universitário La Salle, Canoas, RS, Brasil, 2006.
- [2] I. Foster and C. Kesselman. *The Grid 2: Blueprint for a New Computing Infrastructure*. Morgan Kaufmann, San Francisco, California, USA, 2 edition, 2004.
- [3] I. Foster, C. Kesselman, and S. Tuecke. The anatomy of the grid: enabling scalable virtual organizations. *The International Journal of High Performance Computing Applications*, 15(3):200–222, 2001.
- [4] Ganglia Monitoring System, 2007. <http://ganglia.sourceforge.net/>.
- [5] GridBus, 2007. <http://www.gridbus.org/>.
- [6] Hawkeye, 2007. <http://www.cs.wisc.edu/condor/hawkeye/>.
- [7] R. Jain. *The art of computer systems performance analysis*. John Wiley & Sons, USA, 1991.
- [8] D. d. T. Lemos and P. K. Vargas. Monitoramento de recursos em ambiente de grade. In *Sessão de Pôsteres de Iniciação Científica em PAD*, Porto Alegre, RS, Brasil, 16 a 19 de janeiro 2007. Evento da ERAD2007.
- [9] MonaLisa, 2007. <http://monalisa.caltech.edu/>.
- [10] Projeto Mono Brasil, 2007. <http://monobrasil.softwarelivre.org>.
- [11] NetLogger toolkit, 2007. <http://www.didc.lbl.gov/NetLogger/>.
- [12] P. K. Vargas. *Um Modelo de Gerenciamento Hierárquico de Aplicações em Ambiente de Computação em Grade*. Teste de doutorado, COPPE/Sistemas, UFRJ, Rio de Janeiro, 2006.
- [13] P. K. Vargas et al. Hierarchical submission in a grid environment. In *3rd Intl Workshop on Middleware for Grid Computing*, Grenoble, France, November 28 – December 2 2005.
- [14] WebMDS, 2007. <http://www.globus.org/toolkit/docs/4.0/info/webmids/>.