Interop Router

Alline Kobayashi[1], Bruno Melo[1], Henrique Baggio[1], Ivo Trivella[1], Raul Kist[1]

[1]Microsoft - Open Source Interoperability Lab
Instituto de Computação – Universidade Estadual de Campinas (Unicamp)

{allinekobayashi, brunoamelo, hnrqbaggio, emaildoivo, raulkist}@gmail.com

1. Introdução

Atualmente há um número cada vez maior de aplicações que necessitem de alto poder de processamento para executarem uma determinada tarefa. Tais aplicações só são viáveis quando processadas de forma distribuída. Então, o uso de programas paralelizados rodando em agregados de computadores - conhecidos como clusters - tornou-se uma ferramenta comum para computação de alto desempenho, não apenas para pesquisas acadêmicas, mas também para aplicações industriais [1].

Existem, hoje, diversos modelos de clusters, cada um implementando os conceitos de processamento distribuído da sua maneira e usando softwares e plataformas distintas [2]. Essa diversidade faz com que, às vezes, seja preciso que o usuário de aplicações paralelas aprenda detalhes internos do cluster que pretende usar.

Outro ponto importante neste cenário é a heterogeneidade da utilização dos recursos e a alternância de períodos de alta demanda e ociosidade. Gerando um desperdício de poder computacional.

O Interop Router foi concebido para integrar diversos clusters de diferentes plataforma, gerando interoperabilidade e aumento da escalabilidade e do poder computacional. Usando a Internet para interconectar clusters de diferentes arquiteruras e sistemas operacionais, dispersos geograficamente, os usuários podem submeter jobs paralelos e receber seus resultados através de uma interface simples e transparente. Ao mesmo tempo, ao levar em conta o poder de processamento total da rede de clusters, permite que os recursos presentes nos agregados sejam melhor utilizados.

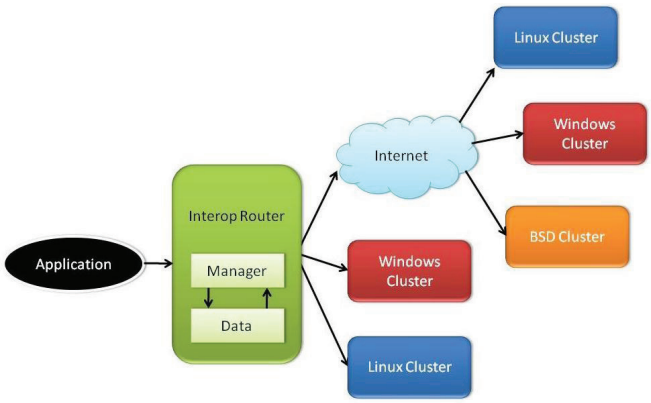

Figura 1: Visão geral da conexão entre o Interop Router e os clusters. As aplicações são submetidas ao IR e então distribuídas entre os diferentes clusters.

2. Estrutura

O Interop Router é organizado em camadas, facilitando a interoperabilidade e a portabilidade. Ele possui quatro camadas básicas: Guaardian (Interface), Banco de Dados (Armazenamento), Deemon (Processamento) e Aangel (Escalonamento).

Sua conexão com o usuário é feita pelo Guaardian, interface que possibilita o acesso a todas as ferramentas do Interop Router. Através dessa interface ocorre a submissão de aplicações e arquivos de dados, monitoramento dos estados dos jobs, download dos resultados dos projetos e informações sobre os clusters conectados ao Interop Router.

O Deemon (Processamento), presente em cada cluster, realiza a comunicação do cluster com o Banco de Dados que, por sua vez, comunica-se com o Aangel (Escalonamento). Este verifica informações de carga dos clusters e dos jobs a serem processados e os distribui de forma a otimizar o processamento.

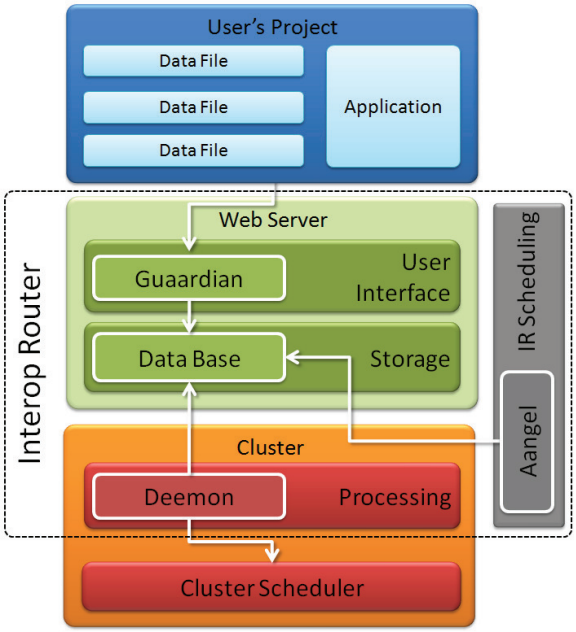

Figura 3: A organização do sistema

3. Projetos e Jobs

Dois conceitos foram adotados: projeto e job.

Os projetos são submetidos pelo usuário e consistem no código fonte da aplicação paralelizada, escrita em C e compatível com MPICH2, que será compilada pelo cluster, e um ou mais arquivos de dados que serão processados pelo programa no cluster.

Os jobs são as partes nucleares de um projeto, formados a partir de cada arquivo de dados. Cada cluster responsável por no mínimo um job recebe do Banco de Dados o(s) arquivo(s) de dados e o código fonte da aplicação. No caso de vários jobs, o código fonte é compilado uma única vez.

Uma vez submetidos ao cluster pelo Deemon, os jobs são processados e os resultados são capturados pelo Deemon e enviado para o Bando de Dados. O usuário pode então ter acesso aos resultados através do Guaardian.

4. Escalonamento

Um dos objetivos do IR (Interop Router) é a otimização dos recursos dos clusters no grid. Isso é feito pelo Aangel, o escalonador do IR.

Sua função é os distribuir os jobs entre os diversos clusters da rede. Para isso, analisa as características estruturais (hardware, programas e bibliotecas instalados) de cada cluster e os recursos livres presentes nos mesmos.

Como o Deemon, Aangel é implementado em Python 2.5 [3], separada e externamente das outras estruturas do Interop Router, o que permite fáceis modificações e customizações. Dessa forma, é simples alterar e calibrar o algoritmo de escalonamento.

5. Interface com o Usuário

A interface consiste em um website. O foco do site é prover acesso aos usuários de uma forma simples e fácil. Para isso, ele é implementado em PHP, o que o mantém genérico e portável.

Nesse site, o usuário tem acesso fácil à informação (multilinguagem) com uma interface amigável. Ele pode acessar três abas: Projeto, Submissão e Recursos. Na aba de Projetos, há uma lista de todos os projetos do usuário com a hora de submissão e seu código fonte. Clicando no nome do projeto, é possível ver os detalhes dos jobs: o status, os arquivos de dados, arquivos de entrada e saída e mensagens de erro. Na aba de Submissão os usuários podem criar um projeto e fazer *upload* dos dados. A aba de Recursos informa detalhes sobre os clusters conectados ao Interop Router e dá uma curta descrição de sua infra-estrutura.

Usar a web como interface faz com que a operação do grid fique transparente e globalmente acessível. Deste modo, os usuários não precisam ter acesso físico aos clusters ou conhecer pormenores das operações de processamento em cada plataforma ou modelo de cluster.

6. Processamento

A interface entre o Interop Router e os clusters é feita pelo Deemon. Sua função é criar uma camada que seja capaz de traduzir as operações necessárias para o processamento dos jobs para os comandos equivalentes ao cluster no qual está instalado.

Uma vez que esteja rodando em um *head node* do cluster, ele se conecta ao Banco de Dados e captura os jobs escalonados para o cluster em questão. Quando o processamento do job termina, retorna os resultados para o Banco de Dados.

Durante esse processo, o Deemon cria uma *thread* que recebe, através de uma conexão SQL, o arquivo de dados e (caso seja o primeiro job de um projeto) o código fonte da aplicação do Banco de Dados. Então, compila o código fonte usando o compilador local do sistema operacional. Tal compilador é reconhecido quando o Deemon é iniciado. Depois disso, o job é submetido ao cluster e a thread fica esperando

que ele termine e gere os arquivos de saída. Uma vez capturados, os arquivos de saída são enviados para o Banco de Dados e a thread é reiniciada.

O uso de threads permite um melhor uso do balanceamento de carga do escalonador do cluster, possibilitando a execução de jobs simultâneos (caso seja suportada pelo cluster).

Outra função do Deemon é enviar à base de dados informações sobre a carga do cluster (quantidade de processadores livres). Esta é uma das informações usadas pelo Aangel ao distribuir jobs entre os clusters conectados ao Interop Router.

O Deemon é implementado em Python 2.5 [3], linguagem que suporta diversas plataformas, e pode facilmente ser portado para novos clusters ou web servers, usando diferentes bancos de dados, escalonadores, compiladores e/ou linguagens de programação, ou mesmo sistemas operacionais.

Atualmente, foram testados Deemon em clusters Linux Beowulf [4], usando PBS [5]; Windows Compute Cluster Server 2003 e Windows HPC Server 2008 [6].

7. Conclusão

Apesar de ainda estar em fase de desenvolvimento, o Interop Router já é uma resposta funcional para o que se propõe desde o começo: disponibilizar o poder de clusters de diferentes plataformas de forma transparente e otimizada.

Os próximos passos de desenvolvimento incluem um maior controle do processamento para administradores e novos níveis de acesso do usuário. Também há a possibilidade do envio de jobs para aplicações previamente instaladas no cluster, no caso de processamento de tarefas paralelas específicas.

Referências

[1] TOP500.Org (2008), "Segments Share Over Time", http://www.top500.org/overtime/list/32/segments, Dezembro.

[2] TOP500.Org (2008), "Operating System Share for 11/2008", http://www.top500.org/stats/list/32/os, Dezembro.

[3] Python Software Foundation (2008), "Python Programming Language", http://www.python.org/, Dezembro.

[4] Beowulf.org (2008), "Beowulf Oveview", http://www.beowulf.org/overview/index.html, Dezembro.

[5] PBS GridWorks (2008), "OpenPBS", http://www.pbsgridworks.com/PBSTemp1.3_2.aspx?top_nav_name=Products&item_name=OpenPBS&top_nav_str=1, Dezembro.

[6] Windows HPC (2008), "Windows HPC Server Technical Overview", http://go.microsoft.com/fwlink/?LinkID=121218, Dezembro.