WVM - Uma Ferramenta para Processamento Distribuído na WWW

Sousa, P.B.M.[1], Fiallos M.A.[2]

[1] Depto. de Computação, Universidade Federal do Ceará / CENAPAD-NE
Campos do Pici, Bloco 901, Fortaleza-CE, Brasil
{benicio@cenapadne.br}

[2] Depto. de Eng. Elétrica, Universidade Federal do Ceará / CENAPAD-NE
Campos do Pici, Bloco 901, Fortaleza-CE, Brasil
{mario@cenapadne.br}

Resumo—

Neste trabalho é descrita WVM (Web Virtual Machine), uma ferramenta portável, baseada nas tecnologias WWW, Java e Internet, para programação de aplicações paralelas-distribuídas que utilizam PVM, MPI ou Java.

WVM dá suporte às várias fases de desenvolvimento de aplicações, tais como conexão remota, edição de códigos, compilação, execução e monitoramento básico de programas.

Além disto, WVM permite que usuários ligados à WWW através de browser, acessem os aplicativos (e bibliotecas) de um sistema de PAD (Processamento de Alto Desempenho).

A ferramenta tem auxiliado no desenvolvimento de programas distribuídos em diferentes plataformas, tais como clusters de PCs, de estações de trabalho e computadores de grande porte.

Palavras-Chaves— **Processamento Distribuído, PVM, MPI, Java, WWW**

I. Introdução

Existem muitas diferenças entre a chamada programação internet e a programação distribuída para processamento de alto desempenho (PAD).

Consideremos, por exemplo, o caso da programação internet aplicada à World-Wide-Web. Os usuários são autorizados a acessar anonimamente processos e dados em lugares remotos, através de um script perl, por exemplo; ou ainda a fazer um "download" de um programa para execução local, como ocorre com uma applet Java. Em tais situações, algumas das principais questões envolvidas no projeto de software incluem aspectos de anonimato e segurança.

Por outro lado, na chamada programação distribuída de alto desempenho, os programas e as máquinas pertencem ao mesmo domínio internet. Embora a segurança seja um aspecto importante, os usuários acessam os recursos da máquina de uma forma explícita e não anonimamente. A programação distribuída de alto desempenho é feita com a ajuda de bibliotecas de passagem de mensagem ("message passing libraries"- MPLs), tais como PVM [GEI 93] e MPI [SNI 98].

A ferramenta aqui descrita tem como objetivo a conciliação de aspectos da programação internet com aspectos da programação distribuída de PAD. Mais especificamente, é proposta a integração das etapas envolvidas na programação paralela-distribuída, tais como edição, compilação, execução, conexão (telnet) e transferência de arquivos (ftp), num único ambiente de programação, baseado nas tecnologias Internet e WWW.

Visando a portabilidade, WVM foi desenvolvida totalmente em Java [GOS 96]. Isto permite sua utilização em ambientes (browsers, por exemplo), suportando a máquina virtual Java (JVM). Desta forma, os recursos disponíveis para processamento de alto desempenho são acessíveis a qualquer usuário que possua um computador ligado à Web.

Outras alternativas propostas para a utilização de Java em processamento paralelo-distribuído envolvem o desenvolvimento de classes para comunicação (Visper [ZHA 98], IceT [GRA 97]) ou a sugestão de arquiteturas para processamento na Web (JavaDC [CHE 97] e Javelin [CHR 97]).

O artigo está organizado da seguinte forma: na seção 2, são apresentadas as principais características funcionais de WVM. A seção 3 trata da organização interna da ferramenta, descrevendo os programas cliente e servidor que a compõem. Finalmente, as conclusões e os trabalhos futuros são apresentados na seção 4.

II. Modelo de Funcionamento de WVM

A. *Objetivos de WVM*

O projeto WVM tem como objetivos principais:

- Aproveitar a ampla divulgação e facilidades de utilização da Internet, do Java e da WWW para auxiliar na programação distribuída;
- Projetar e implementar um ambiente para a Internet e a WWW que permita a execução e o gerenciamento simples de aplicações distribuídas escritas em PVM, MPI ou Java executadas em clusters de computadores de tal forma que:

(a) Usuários de diferentes domínios possam acessar computadores de um outro domínio;
(b) Seja permitido aos usuários preparar o ambiente de execução e monitorar a execução das aplicações remotamente;
(c) Sejam integradas, num mesmo ambiente de trabalho facilidades gráficas que dispensem a necessidade de comandos para conexão remota (telnet), transferência de arquivos (ftp), visualização básica, compilação e edição de arquivos, bem como a configuração das máquinas;
(d) Seja totalmente portável.

B. Modelo de Funcionamento de WVM

WVM baseia-se essencialmente no modelo cliente-servidor, cujo funcionamento é descrito na figura 1.

Inicialmente, ao requisitar uma determinada página (1), um cliente recebe informações de uma máquina central, onde reside o programa servidor, com o qual o usuário interage, fornecendo informações necessárias – como os tipos de aplicativos disponíveis (2). O cliente então atualiza os dados ou códigos existentes e prepara a execução de comandos para processamento remoto que são, então, enviados ao servidor (3). Este se encarregará de buscar as aplicações requisitadas no site (4), e executá-las no domínio de PAD (5). Após o processamento das mesmas (6), os resultados são então retornados ao programa servidor (7), que, finalmente, os repassará ao cliente (8).

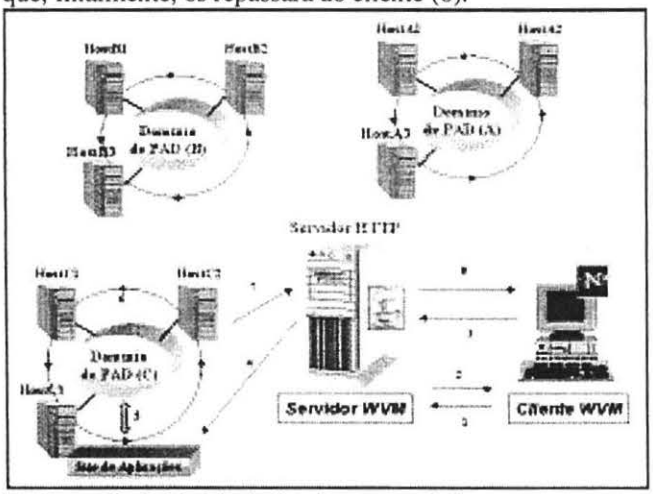

Fig.1. Modelo de funcionamento da ferramenta

Considerando o modelo apresentado, podemos, assim, identificar os seguintes elementos:

C.1. Clientes:

São applets Java ou programas "stand-alone" carregados a partir do servidor pelos usuários em qualquer local da rede através da Web. Cada cliente comunica-se com o servidor através de um canal de execução (thread) próprio.

C.2. Servidor:

Reside numa máquina com uma JVM ativa e conectada ao sistema de PAD. Ela deve possuir também um servidor Web, a fim de que as páginas HTML com código applet possam ser entregues aos clientes. O servidor é responsável pela captura das requisições dos clientes e pelo envio das saídas de erros (stderr) e de comandos (stdout) para cada um dos clientes atualmente ativos.

C.3. Domínio de Processamento e Site de Aplicações:

Correspondem ao conjunto de dados e programas onde residem as aplicações desenvolvidas pelos usuários, incluíndo códigos fontes, arquivos de configuração de máquinas, arquivos textos, etc.

Aqui, as aplicações desenvolvidas em MPLs são realmente processadas. Desta forma, é no domínio de PAD que os processos são exeutados, de acordo com as requisições feitas pelos usuários nos clientes.

C. Considerações sobre o Modelo

Para efeito de simplificação, integramos o site de aplicações e o domínio de processamento em uma mesma entidade na rede. No entanto, eles podem ser distintos. O servidor, deve possuir comunicação direta com tais ambientes ou mesmo ser uma das máquinas envolvidas no cluster.

A comunicação entre o cliente e o servidor se dá através de sockets TCP [HAR 97] e, portanto de forma segura. Para simplificar a transferência de dados, apenas um conjunto básico de comandos é necessário. No entanto, a interação entre o ambiente de processamento e o cliente remoto, pode requerer considerável tempo quando conteúdos de arquivos são transferidos ou massas de respostas são esperadas.

Tanto o cliente quanto o servidor, são compatíveis com as versões 1 e 2 do Java RunTime Environment (JRE) [SUN 97], onde residem as JVMs. Esta compatibilidade é necessária para assegurar que os browsers suportando applets, possam executar os bytecodes do Java [LIN 96] sem quaisquer problemas.

O servidor poderia ser implementado em alguma outra linguagem, desde que a manipulação das threads clientes seja permitida. No entanto, a opção por utilizar Java, permite que o mesmo seja facilmente portável para qualquer outra arquitetura, o que facilita a inclusão de novos domínios de PAD.

III. ARQUITETURA DE WVM

A. O Cliente WVM:

O Cliente WVM (fig. 2) é a interface responsável pela interação com o usuário, capturando seus comandos e enviando-os via socket para o servidor. Ela funciona como aplicação "stand-alone", ou através de uma applet.

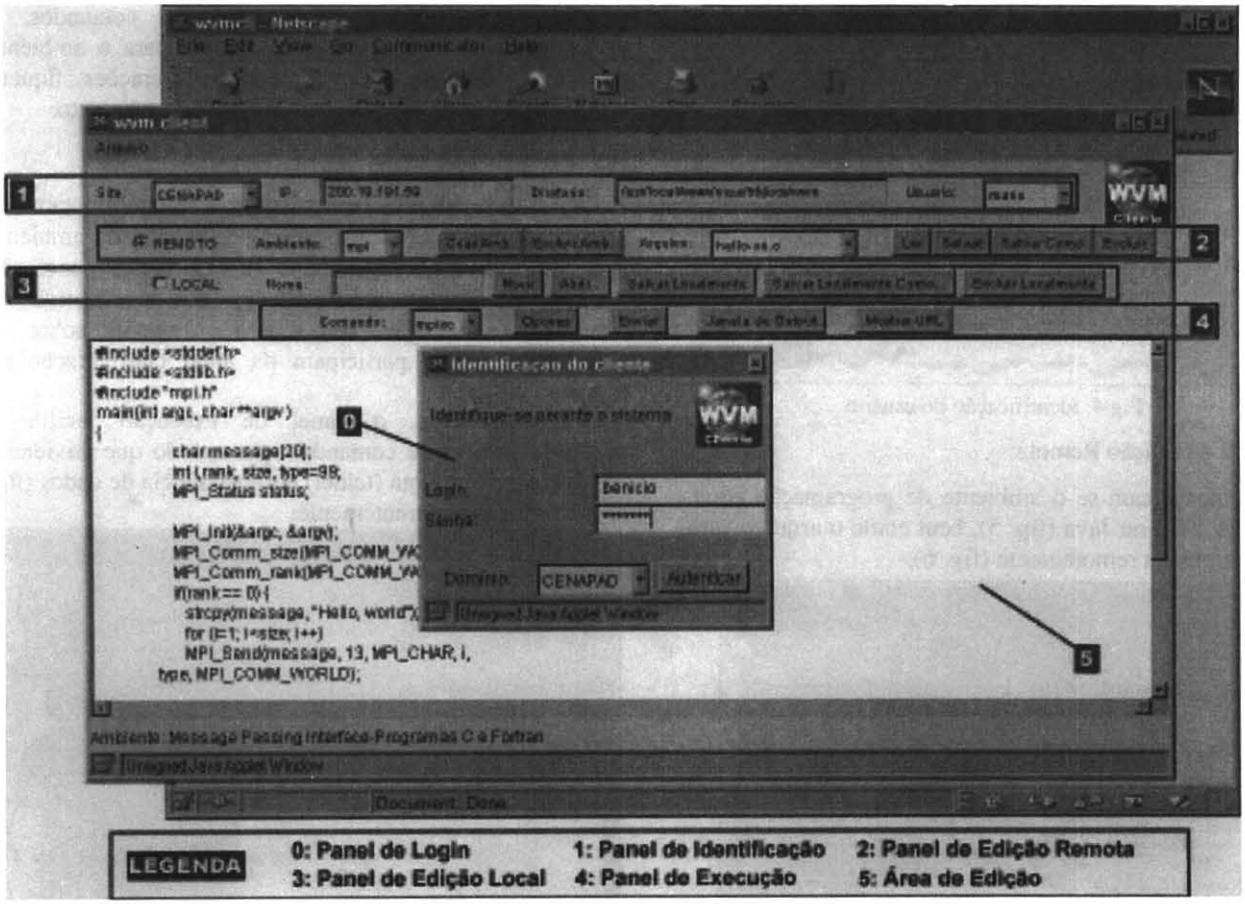

Fig.2. A interface do cliente

Para melhorar o tempo de carga da applet via rede (o que ainda hoje representa um gargalo em Java), apenas controles mínimos foram inseridos, todos compatíveis com a versão JDK 1.1. Isto otimiza o tamanho dos códigos, melhora a velocidade na transferência dos bytecodes pela rede e permite o seu uso em JVMs simples, tais como aquelas disponíveis via browsers.

Funcionalmente, ele apresenta todas as opções necessárias para a gerência de comandos a serem executados: edição de arquivos (locais ou remotos), configuração do ambiente virtual (tipo de MPL, número de máquinas, etc.), acompanhamento das execuções.

A interface subdivide-se em 6 partes, de acordo com suas funcionalidades:

A.1. Panel de Login:

No panel de login é que o usuário fornece as informações necessárias para conexão no ambiente de PAD, incluíndo nome e senha. Somente após a validação do login é que o cliente pode interagir com o servidor.

A.2. Panel de Identificação:

Aqui ficam as informações do ambiente de processamento ao qual o cliente está conectado, bem como sua identificação. O processo de login exige que somente usuários cadastrados possam interagir com o servidor. Vide figuras 3 e 4.

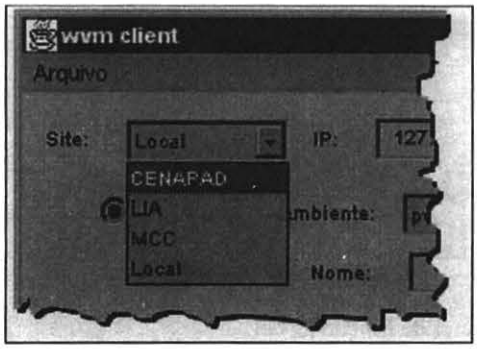

Fig.3. Identificando o domínio de processamento

Fig.4. Identificação do usuário

A.3. Panel de Edição Remota:

Aqui identificam-se o ambiente de programação atual, seja PVM, MPI ou Java (fig. 5), bem como o arquivo com o qual se trabalha remotamente (fig. 6).

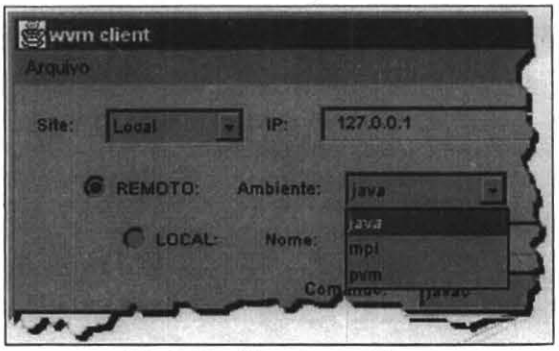

Fig.5. Identificando o ambiente

Na realidade, ao definirmos o arquivo, realizamos uma transferência de dados pela rede, de ou para o servidor, tal como se realizaria com comandos convencionais de ftp.

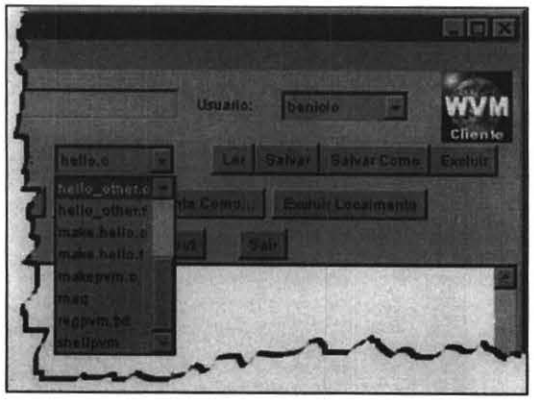

Fig.6. Lendo arquivos de um ambiente

A.4 Panel de Edição Local:

Permite que o arquivo remotamente lido possa ser localmente salvo para edição local. Isto melhora o processo de transferência de dados na rede, permitindo, ainda uma desconexão temporária com o servidor. Obviamente, quando for necessária a execução de comandos, é necessário uma transferência do mesmo para o ambiente remoto, a fim de que as últimas alterações fiquem adequadamente salvas no ambiente de processamento.

A.5. Panel de Execução:

Aqui, são definidos os comandos a serem remotamente executados. As opções disponíveis dependem do ambiente e do tipo de arquivo tratados (fig.7). Neste panel são também permitidas as definições dos parâmetros de compilação, execução, bem como as opções sobre os processadores que participam da atual configuração da máquina virtual.

Funcionalmente, o panel de execução facilita a execução remota de comandos, permitindo que atividades como conexão remota (telnet) e transferência de dados (ftp) sejam feitas transparentemente.

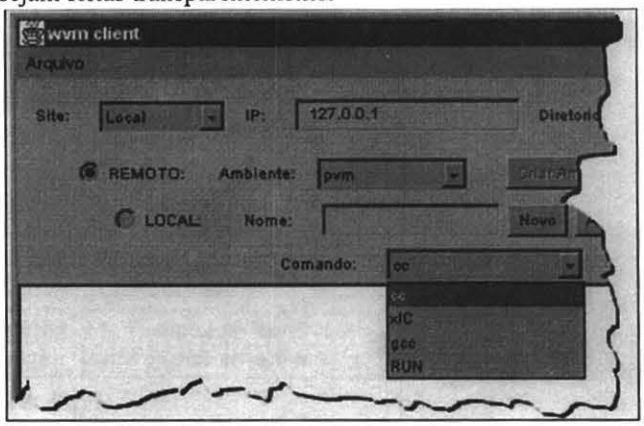

Fig.7. Identificando o ambiente

A.6. Área de Edição:

Principal área para a leitura, alteração e escrita dos códigos de programas, textos e informações de dados, tanto de arquivos locais quanto remotos.

Algumas restrições na implementação, como a adoção de um processador de texto não formatado, permitiu de um lado, uma simplificação do modelo e, de outro, uma redução no tamanho do código de bytes em Java – que constitui um fator crítico para a transmissão da applet via rede.

Ao lado destes painéis, existem janelas auxiliares como a tela de saída de comandos (que exibe para o usuário os resultados dos programas após sua execução) e as telas de mensagens do sistema.

B. O Servidor WVM:

É o servidor (fig. 8) o responsável pela transferência e conversão dos comandos de usuário (transferidos pelo cliente via socket), em chamadas diretas ao sistema distribuído de PAD.

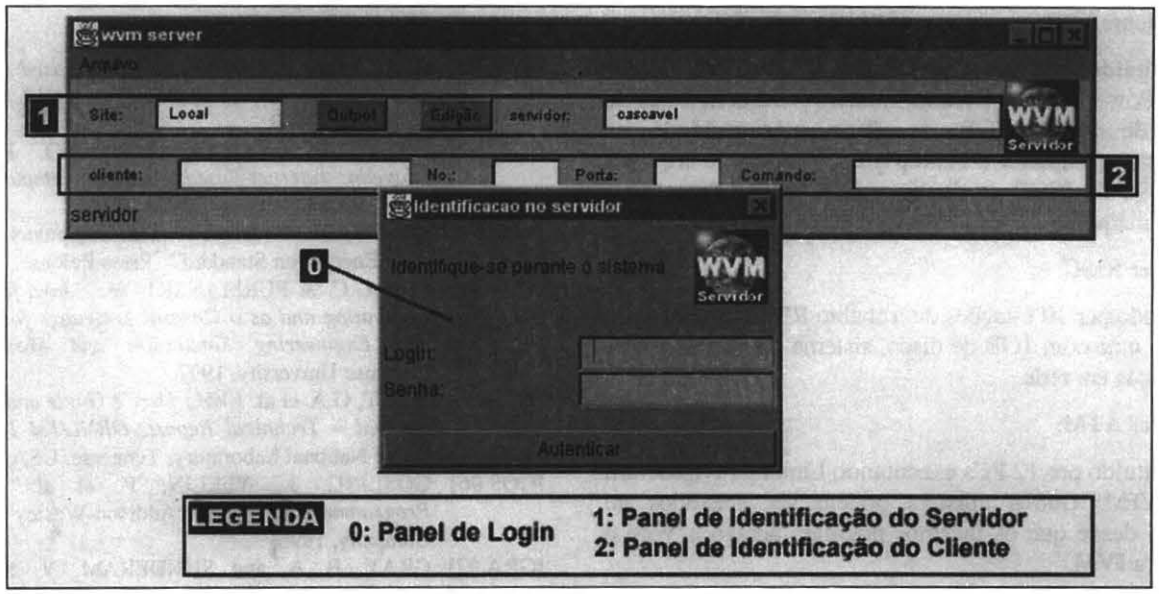

Fig.8. A interface do servidor

B.1. Características da Interface:

Por não participar da interface com o usuário e devido às restrições de segurança impostas pela JVM, tais como a necessidade de interação com o sistema e a escrita em arquivos, o servidor deve ser executado como aplicação stand-alone (isto é, fora de um browser). Apenas o administrador pode acessá-lo.

O servidor é responsável pelo monitoramento das "threads" geradas pelos clientes e sua identificação. Ele permite ainda a editoração de arquivos locais e a execução de comandos.

Em termos de componentes visuais, o servidor mantém uma organização parecida com a do cliente (fig. 8).

B.2. Funcionalidade:

Internamente, o servidor WVM realiza uma análise dos comandos enviados pelos clientes, a fim de executar as ações pretendidas pelos usuários já no ambiente de PAD de destino.

Por realizar constantes comunicações tanto com os clientes ativos, quanto com o ambiente de processamento, o servidor precisa minimizar as comunicações. Um exemplo disso é feito durante o processo de login do usuário: uma vez autenticado, são enviadas todas as informações relativas aos ambientes e arquivos existentes, tornando desnecessário um reenvio dos dados a menos que o sistema remoto passe por uma atualização (inclusão ou remoção de ambientes ou arquivos).

C. Ambiente de Processamento:

Os elementos externos apresentados no diagrama da arquitetura - Domínio de Computação Paralela e Site de Aplicações - (fig. 1), foram unificados no ambiente do Centro Nacional de Processamento de Alto Desempenho no Nordeste (CENAPAD-NE), onde existem diferentes recursos computacionais disponíveis para processamento (fig.9).

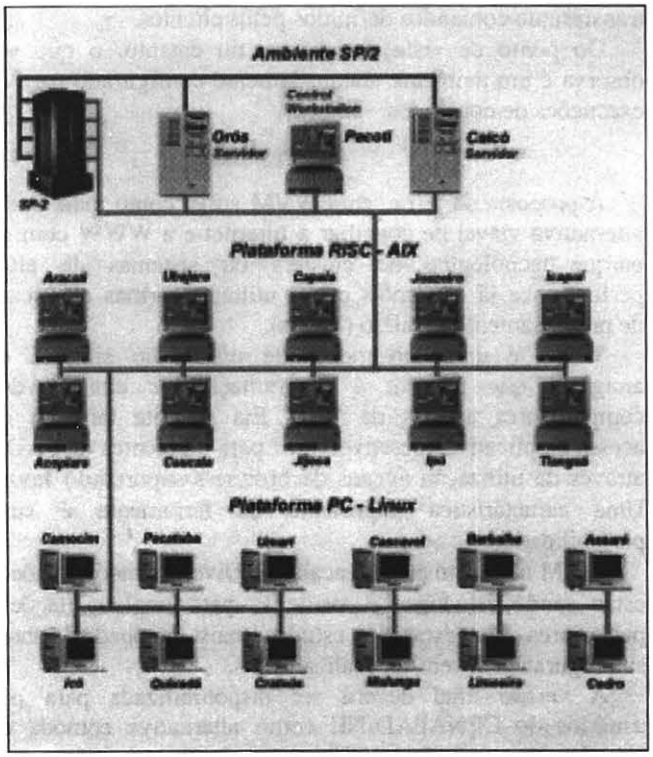

Fig.9. Ambiente de Processamento no CENAPAD/NE

C.1 Ambiente SP/2:

Constituído por um computador IBM SP/2 (Super Scalable Power Parallel System) modelo 9076 com 4 nós de 256MB de RAM, 2GB de disco e capacidade de processamento superior a 1Gflop e 2 servidoras de arquivos modelo 590 de 20GB de disco, com 128MB de memória. Todas as máquinas com sistema operacional AIX.

C.2 Cluster RISC:

Formado por 10 estações de trabalho RISC IBM modelo 250, cada uma com 1GB de disco, sistema operacional AIX e conectadas em rede.

C.3 Cluster ATM:

Constituído por 12 PCs executando Linux através de um switch ATM. Outros clusters podem ser anexados ao ambiente, desde que os mesmos possuam acesso à Web e suportem a JVM.

D. O Ambiente:

A definição do ambiente configura o cluster de máquinas que participam do processamento. Cada ambiente define os comandos e parâmetros possíveis para a compilação e execução de programas previamente editados pelos usuários. Tais ajustes são feitos definindo-se variáveis de sistema, executando-se scripts (arquivos batch) ou ainda transferindo comandos definidos pelos clientes.

Do ponto de vista do cliente, no entanto, o que se observa é um ambiente adequadamente configurado para a execuções de comandos.

IV. Conclusão

A proposta da ferramenta WVM surge como mais uma alternativa viável de conciliar a Internet e a WWW com o parque tecnológico de clusters ou sistemas de alta performance já existentes e que utilizam formas clássicas de processamento paralelo (MPLs).

WVM é uma ferramenta de utillização simples e amigável que permite a programação de clusters de computadores através da Web. Ela permite também o acesso a aplicativos desenvolvidos para ambientes de PAD, através da utilização apenas de browsers suportando Java. Uma característica importante da ferramenta é sua portabilidade.

WVM não é um projeto acabado. Diversas modificações estão sendo estudadas e avaliadas para implementações posteriores. Por exemplo, estudos mais detalhados sobre sua segurança devem ser realizados.

A versão final deverá ser disponibilizada para os usuários do CENAPAD-NE como alternativa cômoda e viável para a submissão e/ou execução de programas via WWW.

Referências

[CHE 97] CHEN, Z, MALY, K. et all.: *"Web Based Framework for Distributed Computing"* – Old Dominion University, 1997.

[CHR 97] CHRISTIANSEN B.O, CAPPELLO, P. et all: *"Javelin: Internet-Based Parallel Computing Using Java"* – University of Californa, 1997

[DES 97] DESCHALL. "Internet-Linked Computers Challenge Data Encryption Standard" . Press Release. 1997.

[FOX 97] FOX G.C. & FURMANSKI, W.: *"Java for Parallel Computing and as a General Language for Scientific and Engineering Simulation and Modelling"* – Syracuse University, 1997.

[GEI 93] GEIST, G.A. et al. *PVM3 User's Guide and Reference Manual – Technical Report. ORNL/TM-12187*, Oak Ridge National Laboratory, Tennesse, USA, 1993.

[GOS 96] GOSLING, J., YELLIN, F. et al. *The Java Programming Language*. Addison-Wesley Publishing Company, 1996.

[GRA 97] GRAY, P. A. and SUNDERAM, V. S.: *"Ice-T: Distributed Computing and Java"* – Emory University, 1997.

[HAR 97] HAROLD, E.R.: *"Java Networking Programming"* – O'Really Associates, 1997, p.149-211.

[LIN 96] LINDHOLM, T. and YELLIN, F.: *The Java Virtual Machine Specification* – Java Series - Sun Microsystems, 1996.

[PHI 97] PHILIPPSEN, M.: *"Is Java ready for computationa science?"* – University of Karlsruhe, Germany – Draft, 1997.

[SNI 98] SNIR, M., GROPP, W. et al. MPI – The Complete Reference – MIT Press. 1998.

[SUN 97] Sun Microsystems Inc – *The Java Runtime Environment - Notes for Developers* – 1997.

[ZHA 98] ZHANG, N.S.K: *"Java and Parallel Processing on the Internet"* – Macquaric University, Austrália – Draft - 1998