TACPD – 10.03.2005
Agregados (clusters) computacionais

1.Aspectos hardware e software de “clusters” de PCs

3.Sistemas de operação e ambientes de execução para “clusters”. A abordagem SSI (Single System Image)

Bibliografia

Disponível em http://asc.di.fct.unl.pt/~pm/TACPD-04-05/

- Mark Baker (editor), **Cluster Computing White Paper,** 2000
- Mark Baker, R. Buyya, **Cluster Computing at a Glance**, cap. 1 do livro *High Performance Cluster Computing – volume 1: Architectures and Systems*, Ed. R. Buyya, Prentice-Hall
- R. Buyya, T. Cortes, H. Jin, **Single System Image**, The International Journal of High Performance Computing Applications, Sage Publications, vol. 15, no.2, Summer 2001, pp 124-135

1- Cluster computacionais

- O que são e porque apareceram ?
- Hardware para “clusters”
- Software de sistema para “clusters”
  - –Comunicações
  - –Gestão de recursos

O que é um cluster de PCs?

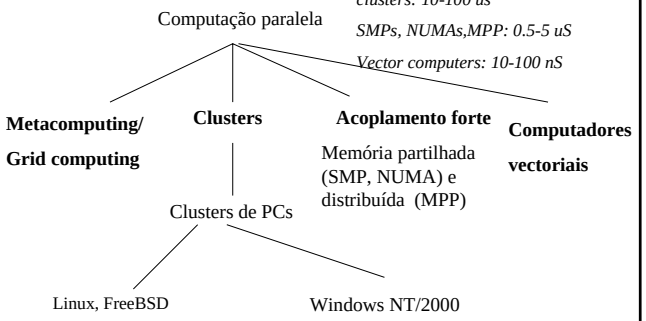
Classificação baseada na latência:

metacomputers: 10-100 ms

clusters: 10-100 us

SMPs, NUMAs,MPP: 0.5-5 uS

Vector computers: 10-100 nS



Quais as vantagens trazidas pelos clusters de PCs?

- Preço/desempenho – convergência da disponibilidade de hardware normalizado e de baixo custo e SO e outras ferramentas de domínio público
- Inclusão rápida dos avanços tecnológicos-COTS (component of the shelf)– os sistemas de gama baixa incorporam mais rapidamente os avanços em CPUs, RAM e hardware de rede
- Arquitectura hardware normalizada –protecção dos investimentos devido à existência de múltiplos fornecedores
- Disponibilidade de linguagens e APIs amplamente usadas – consequência da normalização.

Cluster de Computadores

Sistema de processamento paralelo/distribuído que é constituído por um conjunto de computadores autónomos interligados e que trabalham como um recurso único.

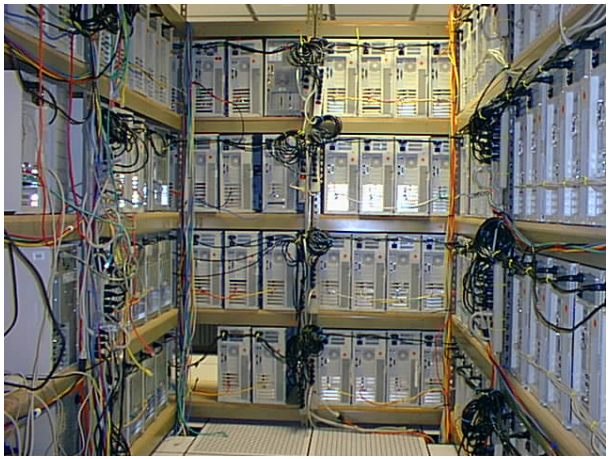
Características:

- **Relativamente local**
- **Sob um único domínio administrativo**
- **Rede de interligação dedicada**

Front end:
Interface humana
Monitorização
Serviços de directoria
Compilação e instalação de software
Segurança e routing

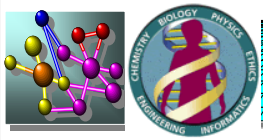
Cluster:
nós de computação

Outras necessidades:
- **Sincronização de relógios**
- **Gestor de recursos para escalonamento de processos nos nós**

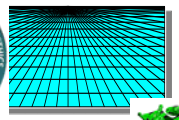


Aplicações de clusters

Resolução de problemas nas áreas de ciência e engenharia usando *modelação*, *simulação* e *análise* *usando clusters*



Ciências da Vida



Aeroespacial

Sistemas de Informação Geográfica

CAD/CAM

Biologia Digital

Aplicações Militares

Definição

- Um "commodity cluster" (agregado de computadores com nós baseados em hardware de uso geral) é um conjunto de nós (computadores) independentes interligados entre si
  - Localidade adminstrativa : todos os seus componentes constituem um domínio administrativo, gestão é feita como se se tratasse de uma única máquina
  - COTS (componentes off the shelf)
  - Localidade física: tipicamente reside numa sala

Objectivos dos clusters

- Melhoria do desempenho (performance @ baixo custo)
- Melhoria da disponibilidade (gestão de falhas)
- Single System Image ("look-and-feel" de um sistema único)
- Escalabilidade (hardware e aplicações)
- Comunicação rápida (redes & protocolos)
- Equilíbrio de carga (CPU, Rede, Memória, Disco)
- Segurança e Criptografia (clusters de clusters)
- Facilidade de administração
- Facilidade de programação (APIs simples)
- Aplicabilidade(suporte de aplicações cluster-aware e non-aware)

Nó de um cluster

- Computador autónomo com hardware completo e suporte de sistema para execução de programas e de comunicação com outros nós: de PCs desktop a SMPs
- Concentração em sistemas em que cada nó tem um hardware semelhante a um computador pessoal:
  - CPU: família do CPU (x86, Alpha, Sparc) , relógio, cache(s), RAM
  - discos: obrigatório nos nós de serviço (E/S), opcional nos outros
  - floppy, CD-ROM, monitor-teclado
  - rede

Interligação dos nós

Software de sistema:

- TCP/IP
- protocolos leves

Hardware:

- 100 Mbps Fast Ethernet (placa + switch pesam pouco)
- Gigabit Ethernet
- Myrinet ou outras: preço semelhante ao do nó

Hardware dos nós

• CPUs rápidos vs RAM lenta

• dados do programa devem residir em RAM, uso do disco explicitamente (out of core calculations) or implicitamente (memória virtual) tem custos

palpite: 1 byte de RAM por Flop:

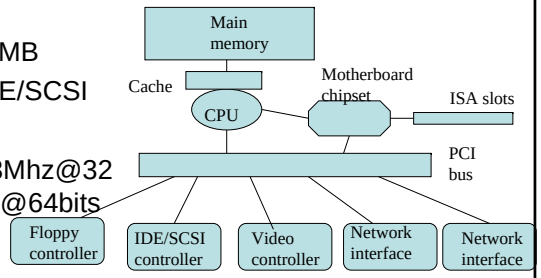
pentium 450 Mhz => 200 Mflops => 200 Mbytes de RAM

• dispositivos de I/O devices: PCI bus

- disco 10 vezes o tamanho da RAM
- NIC

Nó típico de um cluster de PCs

- CPU > 500MHz
- Cache 16KL1,512K L2
- RAM >128MB
- Control. IDE/SCSI
- disco
- PCI bus 33Mhz@32 bits 66Mhz@64bits



Gargalo do sistema: bus PCI

- 132 Mbytes/s: 33Mhz : 32 bits
- 528 Mbytes/s: 66 MHz : 64 bits
- Transferências por bloco (burst)
- Evoluções recentes:
  - PCI-X
  - PCI-Express

Interconexões

- Normalizadas – Fast ou Gigabit Ethernet
- Mais exóticas – ver à frente

Componentes software

- Ambientes de desenvolvimento de programas
  - Ligados à configuração hardware sem memória partilhada – troca de mensagens ex: MPI
  - Depuradores ...
- Software para gestão dos recursos do cluster
  - Instalação e configuração remota
  - Atribuição de nós aos vários trabalhos
  - Administração corrente
  - Monitorização e diagnóstico
  - Acesso paralelo aos dados

2- Tecnologias de interligação

- Protocolos
  - TCP/UDP vs protocolos leves
  - Alternativas hardware

TCP/IP e sockets

- “Sockets” permitem acesso aos protocolos de transporte TCP e UDP
- As características da rede local UDP (baixíssima taxa de erros, troca de ordem, …) vs TCP (latência de início e fim da ligação, controlo de fluxo)
- A maior parte das aplicações emitem muitas mensagens relativamente curtas

Performance do TCP-IP em Linux

- Kernel 2.0.29, Pentium II 300 Mhz, 3C905B 100
- ping pong test usando TCP, sem outro tráfego, switch
- max.largura de banda: 12.5 MB/s; latência do hardware: 7uS
- Medidas:

  - largura de banda 10.8 MB/s, 6.25 MB/s com mensagens com 1750 bytes

  - latência: 70 uS

Razões para a diferença

- Atrasos introduzidos pelas chamadas ao sistema
- Cópia de buffers dentro do kernel e entre o espaço utilizador e o kernel
- Processamento desnecessários (checksums)
- Armazenamento de dados enquanto se espera pela recepção
- Não alinhamento de dados e cabeçalhos (palavra e página)
- Quanto maior é a "velocidade no fio" mais pesa o processamento feito em cada nó …

Protocolos leves de comunicação

- O desempenho depende da largura de banda e latência ao nível da aplicação
- Hardware tem largura de banda suficiente (> 1 Gbps)
- É preciso reduzir a latência :(pode-se tentar deixar isto ao cuidado do programador…)

  - executar os protocolos no NIC

  - modificar a interface do SO para permitir transferir dados directamente entre o NIC e a memória do utilizador

Protocolos leves

- Active messages (AM)(proj. NOW–U. Berkeley)
  - Msgs. Síncronas, pedido-resposta, emissor e receptor reservam espaço em RAM; zero cópias
- Fast Messages (U. Illinois)
  - Semelhante AM
- VMMC (proj. SHRIMP - Univ. Princeton)
  - Envio e recepção de mensagens correspondem a leituras/escritas em memória
- U-Net (U. Cornell)
  - Conceito de "virtual network interface" a nível utilizador
- BIP (U. Lyon)

Normas para comunicações leves

- VIA
- Infiniband

VIA Standard 1998

- Virtual Interface Architecture: promovida por Intel, Compaq and Microsoft
- Muito semelhante ao U-Net
- Implementações sobre Myrinet, SCI, …
- Deveria ser suportada nos NICs, mas parece que não …
- Norma de muito baixo nível (Exemplo: utilizador tem de registar zonas de memória …)

Infiniband

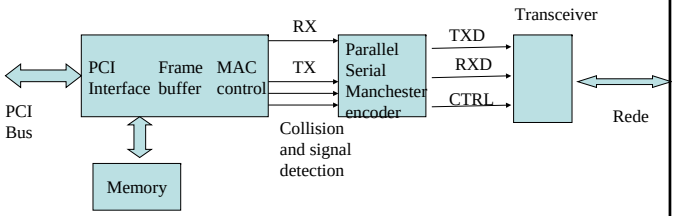
- Consórcio de fabricantes: HP/Compaq, IBM, Intel, Microsoft, Sun, Dell ...
- Objectivo inicial: substituir o bus PCI por um bus série de alva velocidade (2.5 Gbips): os periféricos constituiriam uma rede com “switches” ...
- Suporte de envio de pacotes e RDMA (Remote Direct Memory Access)

Formas de interligação

	Baseada em mensagens	Partilha de memória
Ligação ao bus de E/S	Forma mais comum: inclui o suportado pelo hardware de rede, TCP/IP, VIA	Partilha de discos
Ligação ao bus de memória	Habitualmente implementado em software sobre hardware de rede; SCI, Infiniband tem suporte de RDMA	Memória partilhada distribuída (SVM)

Hardware Fast Ethernet

- Muito baixo preço, Gigabit Ethernet vem baixando rapidamente
- NIC:



Hubs e switches

- UTP, ligação em estrela logicamente
- repetidores/hubs só amplificam os sinais (e o ruído)
- Switches: auto-negociação (velocidade,full/half duplex), ligação em árvore
- O número de comunicações simultâneas full-duplex connections está limitada pela capacidade agregadado do switch (1.2 a 40 Gbps)

Limitações da Fast Ethernet

- O aumento da velocidade dos CPUs e da largura de banda do bus PCI permitem a um PC saturar uma ligação a 100 Mbps
- “Channel bonding” : várias interfaces por nó; a prática mostra que 3 é um limite
- Gigabit ethernet é a solução
- Algumas limitações na sua forma standard: Jumbo frames

Outras tecnologias de alta velocidade

- **ATM** : 48 byte data cells, modelo por conexão, > 644 Mbps, hardware caro
- **FDDI**, 100 Mps, pacotes de 4500 byte
- **HIPPI**
- **Scalable Coherent Interface** (SCI)
- **Myrinet**
- **Fibre Channel**
- **Infiniband**

HIPPI (High Performance Parallel Interface)

- Desenhado originalmente para interligar supercomputadores com discos/bandas
- Também conhecido por GSN (Gigabit System Network)
- largura de banda: 800 Mbyte/s
- Bastante mais caro que outras tecnologias (cabos paralelo com 64 linhas de dados)

Scalable Coherent Interface(SCI)

- IEEE standard 1992
- Ligações ponto a ponto, série sobre fibra ou cobre
- Largura de banda: 400 a 1000 Mbyte/s
- latência: 2 uS
- A norma define as várias camadas do protocolo (físico, data link) e inclui protocolos para acesso a memória remota

SCI (cont.)

- SCI foi usado para construção de multiprocessadores NUMA (HP, Sequent, Data General)
- SCI também usado para estender bus de E/S
- Permite configurações bastante grandes (ponto a ponto, anel, hierarquia de switches)
- custo por nó: ~ 1000 USD

Myrinet

- 1.28 Gbit/s full duplex
- Cabos com 8 bits de dados, Árvore de switches
- NICs têm um processador cujo código pode ser modificado pelo utilizador
- Muito popular em universidades, custo semelhante ao SCI

Giganet cLAN

- 1999 – Int. PCI 1.25 Gbps
- Suporte VIA em hardware

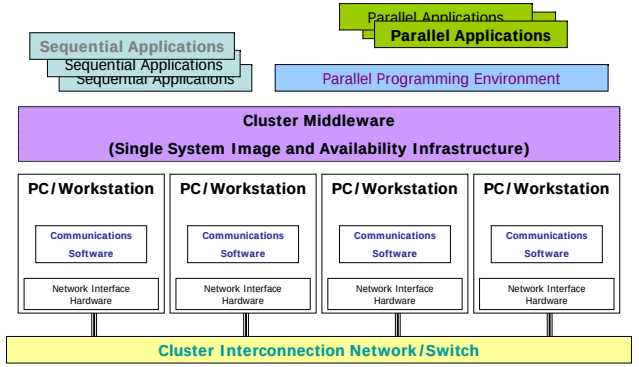
ServerNet 2

- Tandem -> Compaq -> HP
- Proprietário
- Suporte de agregação de “links”

Comparação

	Gigabit Ethernet	Giganet	Myrinet	SCI	ServerNet 2
Largura de banda MPI (Mbytes/s)	35-50	105	140	80	65
Latência MPI (uS)	100-200	20-40	18	6	20
Preço por interface (USD)	1500 (2000) 100 (2004)	1500	1500	1500	1500
Número de nós máximo	+1000	+1000	+1000	+1000	64K
Suporte VIA	Software	Software	Software	Software	hardware

2. Ambientes de administração, desenvolvimento e execução



Objectivos

- Transparência na Gestão de recursos
  - Permite ao utilizador usar o cluster facilmenee sem conhecer a estrutura deste
  - O utilizador tem uma visão global dos processos, rede e sistema de ficheiros
- Escalável quanto ao desempenho
  - Pode ser expandido facilmente , e o desempenho deve escalar da mesma forma
  - Para extrair o máximo desempenho, deve-se suportar equilíbrio de carga e distribuir Io trabalho pelos vários nós de forma equilibrada
- Melhoria da disponibilidade
  - Em qualquer altura , deve ser possível que aconteça uma falha sem que a aplicação do utilizador seja afectada
    - Tecnologias de tolerância a falhas: replicação, checkpointing …
  - Se houver dados replicados deve ser promovida a sua consistência

Sistemas de operação para clusters

- Grandes opções para suportar clusters:
  - Usar um SO normal e colocar o suporte do cluster a nível utilizador (middleware)
  - Modificações profundas a nível do kernel de sistemas existentes
  - SO desenhado de raiz

Software básico de um nó

- SO clássico
  - Linux; FreeBSD
  - Windows NT/2000
  - Linux é a plataforma de escolha da maioria dos grupos de investigação
- Suporte da comunicação
  - Sockets TCP/IP, RPC, Corba, Java RMI, MPI, PVM, …
  - Sistemas de ficheiros em rede: NFS, AFS, CIFS

Alguns aspectos interessantes do Linux

- Sistema de ficheiros /proc
- Módulos do kernel carregados dinamicamente
- Consolas virtuais
- pluggable authentication modules (PAM)
- Gestão de pacotes

SSI - Single System Image

- Esconde do administrador e/ou do utilizador a existência de múltiplos nós
- Transparência quanto à localização dos recursos
- SSI a nível de adminstração e escalonamento de trabalho ou SSI ao nível das chamadas ao sistema

Aspectos do SSI

- Suporte de heterogeneidade
- Acesso ao hardware de comunicações a nível utilizador (virtual NIC)
- E/S paralelas
  - E/S locais e remotas de forma transparente
  - Sistema de ficheiros paralelos
- Mais informação à frente ...

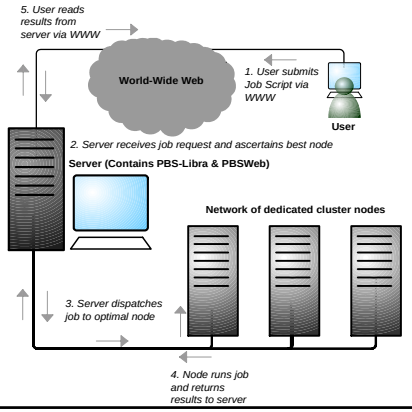
Alguns SOs para clusters

- Variantes do Linux
- MOSIX
- Solaris MC
- Versão AIX IBM SP; GPFS- General Parallel File Systems

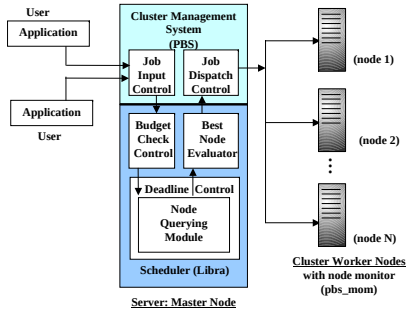
Gestão de recursos - Resource Management and Scheduling (RMS)

- RMS suporta a distribuição das aplicações entre os nós do cluster para maximizar o “throughput” (débito binário)
- Permite o uso efectivo e eficiente dos recursos disponíveis
- Componentes Software
  - Gestor de recursos (Resource manager)
    - Localização e reserva do recurso computacional, autenticação, criação de processos, migração de processos
  - Escalonador de recursos (Resource scheduler)
    - Filas de espera para aplicações, políticas de atribuição e reserva de recursos Indica ao “resource manager “ o que e quando fazer.
- Razões para usar RMS
  - Fornece às aplicações um aumento do seu “throughput” e aumenta a sua fiabilidade
  - Equilíbrio de carga (Load balancing)
  - Utilização de CPUs livres
  - Tolerância a falhas
- Arquitectura básica do RMS: sistema cliente-servidor

Exemplo de interacção com um RMS



Exemplo de um RMS



Serviços fornecidos por um RMS

- Migração de processos
  - Quando um recurso computacional tem demasiada carga
  - Preocupação com tolerância a falhas
- Checkpointing
- Procura de ciclos livres
  - 70% to 90% do tempo as workstations estão livres
- Tolerância a falhas
- Minimização do impacto nos utilizadores
- Equilíbio de carga
- Múltiplas Filas para Aplicações

Alguns RMS populares

Projecto	Sistemas comerciais - URL
LSF	http://www.platform.com/
SGE	http://www.sun.com/grid/
Easy-LL	http://www.tc.cornell.edu/UserDoc/SP/LL12/Easy/
NQE	http://www.cray.com/products/software/nqe/
	Sistemas de domínio público - URL
Condor	http://www.cs.wisc.edu/condor/
GNQS	http://www.gnqs.org/
DQS	http://www.scri.fsu.edu/~pasko/dqs.html
PBS	http://pbs.mrj.com/
Libra	http://www.buyya.com/libra or www.gridbus.org

Administração de um cluster

- Planeamento e manutenção da configuração
- Hardware
  - Número de CPUs e seu tipo, memória; com/sem disco
  - Número e tipo das interligações
- Software
  - Acesso remoto
  - Instalação remota
  - Monitorização e controlo remoto dos nós
  - Gestores e escalonadores de recursos (RMSs)

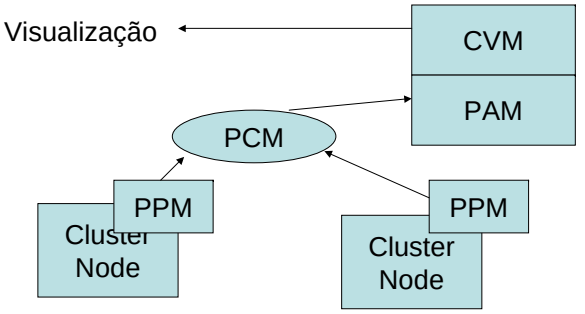
Acesso remoto

- Linux: telnet server, rshd server, ftpd, apache, X-windows “standards”
- Windows 2000: Services for Unix, ou 3rd party

Instalação remota

- Windows 2000: RIS, Ghost, Imagecast
- Linux:
  - Duplicação dos discos – exige configuração homogénea (...)
  - Utilização de processos de instalação associados a distribuições: Kickstart – RedHat
  - Distribuições dedicadas a “clusters” incluem ferramentas para isto:
    - OSCAR (Open Source Cluster Application Resources)
    - NPACI Rocks

Monitorização e controlo dos nós



PPM – módulo de recolha de informação
PCM – armazenamento da informação recolhida
PAM – biblioteca de acesso à informação recolhida
CVM – visualização da informação

Ambientes de programação para clusters

- Modelo de memória partilhada
  - DSM
  - Threads/OpenMP (modificado para clusters)
  - Java threads (HKU JESSICA, IBM cJVM)
- Modelo de troca de mensagens
  - PVM (alguns aspectos de RMS …)
  - MPI (mais fácil de optimizar para uma dada arquitectura)
- Compiladores com paralelização automática
- Bibliotecas paralelas e “Kernels” computacionais (e.g., NetSolve)

Clusters representativos

- Beowulf
- NOW
- HPVM

Beowulf

- Pioneiro na construção de clusters com PCs
  - Investigar o potencial dos clusters de PCs para efectuar cálculo científico
  - Registou o termo “Pile-of-PCs” (PoPC) para descrever um conjunto de PCs com acoplamento fraco
  - Pôs a tónica no uso de componentes de uso geral produzidos em massa, processadores dedicados, uso de redes de comunicação dedicadas
  - Atingir a melhor relação custo/desempenho para o cluster

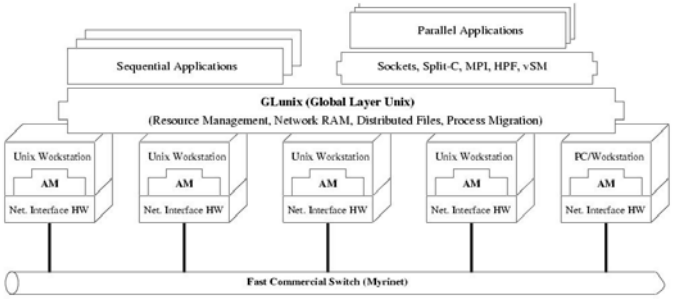
Beowulf (2)

- System Software
  - Grendel
    - Colecção de ferramentas
    - Gestão de recursos e suporte de aplicações distribuídas
  - Comunicação
    - Através de TCP/IP sobre uma Ethernet interna ao cluster
    - Emprega uma rede Ethernet múltipla em paralelo para satisfazer as necessidades de largura de banda para transferências intra-cluster
    - Uso de técnicas de ‘channel bonding’
  - Extensão do kernel do Linux para permitir a um conjunto de nós independentes participar em espaços de nomes globais
  - Dois esquemas de identificadores globais (GPID)
    - Independente de bibliotecas externas
    - GPID-PVM compatível com o formato do Task ID format ; usa o PVM internamente

NOW (I)

- Projecto Berkeley Network of Workstations (NOW)
  - Demonstrar a construção de um sistema paralelo de larga escala usando “workstations” comerciais e equipamento de comunicação “switched” topo de gama
  - Comunicação entre processos
    - Active Messages (AM)
      - Mecanismo de comunicação básica: RPC simplificado que pode ser implementado eficazmente numa larga gama de software de sistema/hardware
  - Global Layer Unix (GLUnix)
    - Uma camada sobre SO para fornecer execução remota transparente, suporte para “jobs” interactivos paralelos e sequenciais , equilíbrio de carga e “backward compatibility” para executáveis já existentes
    - Suporta um espaço de nomes “cluster-wide”: Network PIDs (NPIDs), e Virtual Node Numbers (VNNs)

Arquitectura do sistema NOW



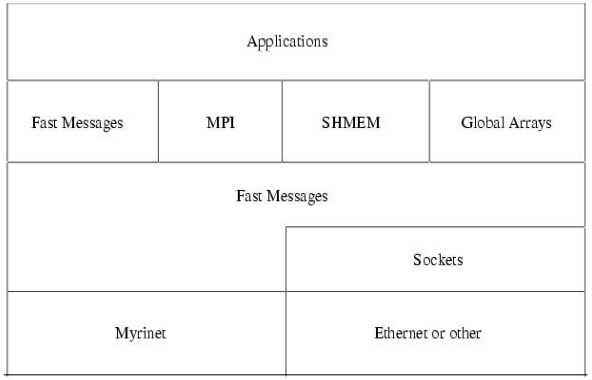
NOW - continuação

- Network RAM
  - Permite usar recursos livres em máquinas desocupadas como discos de paginação para máquinas ocupadas
  - Sem distinção cliente/servidor
    - Qualquer máquina pode ser um servidor quando está livre, ou cliente quando precisa de mais memória do que q fisicamente disponível.
- xFS: Serverless Network File System
  - Um sistema de ficheiros distribuído, sem servidores, que suporta baixa latência e alta largura de banda no acesso ao sistema de ficheiros, através da atribuição da funcionalidade de servidor entre os clientes.
  - A funcionalidade de localização de dados no xFS está distribuída entre clientes, sendo cada um responsável pelo atendimento dos pedidos para um dado subconjunto de ficheiros.
  - O conteúdo de um ficheiro está dividido (striped) entre vários clientes para fornecer alta largura de banda.

High Performance Virtual Machine (HPVM)

- Fornecer desempenho de supercomputador num sistema COTS de baixo custo
- Esconde as complexidades de um sistema distribuído debaixo de uma interface simples
- Problemas tratados pelo HPVM
  - Comunicação de alto desempenho usando APIs normalizadas e de alto nível
  - Coordenação de gestão e escalonamento de recursos
  - Tratamento da heterogeneidade

Arquitectura em camadas do HPVM



HPVM (3)

- Fast Messages (FM)
  - Protocolo de comunicação de alta largura de banda e baixa latência baseado nas Active Messages de Berkeley
  - Funções para enviar mensagens longas e curtas e para extrair mensagens da rede
  - Optimizações levando em conta a hierarquia de memória
  - Garante entrega fiável e ordenada, bem como o controlo do escalonamento da realização das comunicações
  - Originalmente desenvolvido num Cray T3D e num cluster de SPARCstationsligadas por Myrinet
  - API de comunicação de baixo nível que garante desempenho próximo do que o hardware permite.
  - API de alto nível que fornece maior funcionalidade, facilidade de uso e portabilidade das aplicações

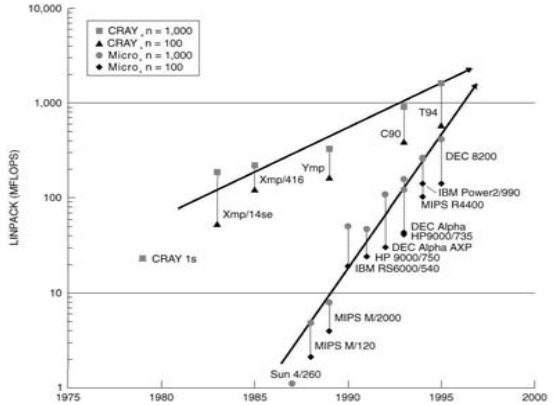
Comparação

Projecto	Platforma	Comunicações	SO	Modelos de programação
Beowulf	PCs	Múltiplas Ethernets com TCP/IP	Linux e Grendel	MPI/PVM. Sockets e HPF
Berkeley Now	PCs e workstations com Solaris	Myrinet e Active Messages	Solaris + GLUnix + xFS	AM, PVM, MPI, HPF, Split-C
HPVM	PCs	Myrinet com Fast Messages	NT ou Linux + global resource manager + LSF	Java-fronted, FM, Sockets, Global Arrays, SHEMEM e MPI

Tendências Hardware e Software

- Aumento da performance de rede de 10x quando se usa 100BaseT Ethernet com suporte full duplex. Idem Gigabit Ethernet?
- Disponilidade da tecnologia “switched” , incluindo comutadores com “full crossbar” para redes proprietárias switches como a Myrinet
- O desempenho das workstations e PCs aumenta significativamente
- O aumento de desempenho dos microprocessores levou ao aparecimento de PCs de secretária com o desempenho de “workstations” de baixo de gama a baixo custo
- A diferença de desempenho entre supercomputadores e clusters baseados em componentes de uso geral está a diminuir rapidamente (ver slide seguinte)
- Os supercomputadores paralelos são agora equipados com componentes COTS, especialmente microprocessadores
- Aumenta o uso de nós SMP com 2 a 4 CPUs
- O número médio de transístores cresce cerca de 40% ao ano
- A frequência de relógio crece cerca de 30% ao ano

Tendências tecnológicas



Cluster de SMPs (CLUMPS)

- Serão os supercomputadores do futuro
- Múltiplos SMPs cada um com diversas interfaces de rede podem ser ligados usando interfaces de rede dedicadas
- 2 vantagens
  - Beneficia-se do alto desempenho, facilidade de uso e programação dos SMPs com poucos CPUs
  - Os clusters podem ser construídos com menor esforço, a administração torna-se mais fácil, e há maior localidade dentro de um nó

Vantagens em usar Clusters baseado em COTS

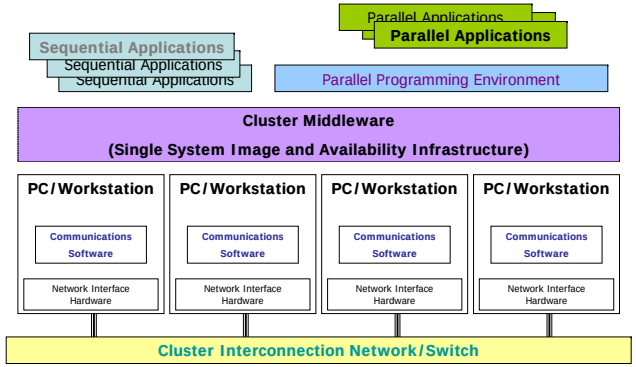
- Melhor relaçao preço/desempenho do que um supercomputador paralelo
- Permite crescimento continuado que se enquadra com os padrões de financiamento anual
- Melhor aplicabilidade a um maior número de domínios

Sistemas de operação SSI (Single System Image) para clusters

- Princípios gerais
- Estudo de casos

- Bibliografia disponível em http://asc.di.fct.unl.pt/~pm/clusters/
- - Mark Baker (editor), **Cluster Computing White Paper,** 2000

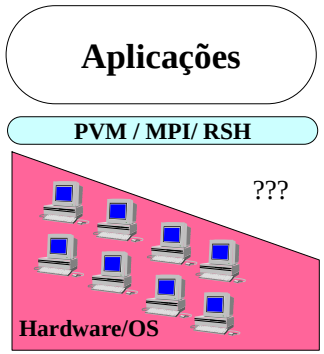
Ambientes de administração, desenvolvimento e execução



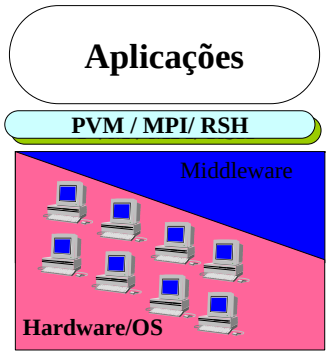
SSI - Single System Image

- SSI oferece a ideia que o cluster é uma máquina única
- Esconde do administrador e/ou do utilizador a existência de múltiplos nós
- Transparência quanto à localização dos recursos
- SSI pode existir:
  - Aplicação: oferece uma visão unificada ao utilizador
  - Middleware/Sistema de Operação: oferece uma visão unificada às aplicações
  - Hardware: Hardware DSM

Um ambiente típico de cluster



A ligação é assegurada pelo "middleware" do cluster



Objectivos do Middleware

- Transparência completa (Gestão):
  - Vê-se um sistema único
    - Ponto de entrada único para ftp, telnet, carregamento de software...
- Performance escalável:
  - Crescimento rápido do cluster
    - Não há mudança de API & distribuição automática da carga.
- Disponibildade acrescida:
  - Recuperação automática de falhas
    - Aplicação de checkpointing & técnicas de tolerância às falhas
  - Suportar a consistência dos dados no caso de haver replicação

O que é o Single System Image (SSI)?

- SSI é a ilusão criada pelo software e/ou hardware, que apresenta um conjunto de recursos de computação como sendo um recurso único.
- SSI faz o cluster aparecer como uma máquina única para o utilizador, as aplicações e a rede.

Serviços de um SSI

- Serviços essenciais
  - Ponto de acesso único
  - Hierarquia de sistema de ficheiros única
  - Espaço de E/S único
  - Ponto de gestão único
  - Acesso à rede único
  - Sistema de gestão de trabalhos e recursos unificado
  - Espaço de endereçamento dos processos único
  - Interface de utilização única
- Nem todos podem ser normalmente fornecidos

Benefícios do SSI

- Uso transparente dos recursos do sistema.
- Migração transparente de processos e equilíbrio de carga entre nós transparente.
- Fiablidade e disponibilidade melhorada.
- Melhoria do desempenho e do tempo de resposta
- Gestão do sistema simplificada.
- Redução do risco de erros de operação.
- Não é necessário ter conhecimentos sobre o hardware e software do cluster para o utilizar com eficiência.

Serviços de SSI

- Ponto de entrada único:
  - telnet cluster.di.fct.unl.pt
  - telnet node1.cluster.di.fct.unl.pt
- Hierarquia única do sistema de ficheiros: /Proc, NFS, xFS, AFS, etc.
- Ponto de controlo único: através de GUI -Like workstation/PC– pode usar tecnologia WEB
- Ponto de acesso à rede único
- Espaço de memória único - Network RAM/DSM
- Sistema de gestão de trabalhos unificado: Glunix, Codine, LSF

Funções que suportam a alta disponibilidade

- Espaço de I/O único:
  - Qualquer nó pode fazer acesso a qualquer periférico (ex. Disco) sem conhecer a sua localização física.
- Espaço de comunicação entre processo único:
  - Qualquer processo em qualquer nó pode criar processos em qualquer nó; os processos podem comunicar dentro do cluster através de "pipes" e sinais como se estivessem no mesmo.
- Checkpointing e migração de processos:
  - Pode guardar o estado do processo e resultados intermédios em memória no disco para suportar "rollback recovery" quando o nó falha. Isto também é útil para "Load balancing" feito pelo RMS...

Gestão dos processadores

- Escalonar trabalhos nos nós
  - A política de escalonamento deve considerar
    - Recursos pedidos vs. recursos necessários
      - Processadores
      - Memória
      - Entradas/saídas
    - Tempo limite de execução
    - Prioridades
  - Diferentes tipos de trabalhos
    - Sequenciais
    - Paralelos

Equilíbrio de carga (load balancing)

- Problema:
  - Um equilíbrio estático perfeito não é possível
    - O tempo de execução dos trabalhos é deconhecido
  - Os sistemas desiquilibrados são ineficientes
- Solução:
  - Migração de processos
    - Antes da execução
      - Granularidade deve ser pequena
    - Durante a execução
      - O custo tem de ser avaliado

Monitoração de um “cluster”

- Um cluster precisa de ferramentas para monitorização
  - Os administradores têm muitas coisas para verificar
  - O cluster deve ser visível de um ponto único
- Aspectos a monitorizar
  - Ambiente físico: temperatura, alimentação
  - Serviços lógicos: RPCs, NFS
  - Métricas de desempenho: activiadade de paginação, carga do CPU
- Monitorização em clusters heterogéneos
  - Diversos tipos de nós e diversos SOs
  - A ferramenta deve esconder as diferenças

Auto-administração

- Monitores sabem fazer diagnóstico, mas precisam de começar a fazer acções correctivas
  - Muito embrionários (ver Nagios –www.nagios.org)
- É um passo necessário
  - Muitos nós, muitos dispositivos, muita diversidade, grande probabilidade de erro

Tolerância às falhas

- Um grande “cluster” tem de ser tolerante às falhas
  - A probabilidade de falha é elevada
- Solução
  - Re-execução dos processos residentes no nó em falha
    - Nem sempre possível ou aceitável
  - “Checkpointing” e migração
    - Pode ter custos elevados
- Difícil com alguns tipos de aplicações
  - Aplicações que modificam o ambiente
    - Uma solução pode ser um comportamento transaccional

Alta disponibilidade (HA)

- Essencial para algumas aplicações
  - A disponibilidade 7dias/semana 24h/dia pode contrariar a escalibilidade
- Base em tecnologia já referida
  - SSI: esconder diferenças na configuração
  - Ferramentas de monitorização: detectar erros e eventualmente corrigi-los
  - Migração de processos: recomeçar/retomar computações nos nós sobreviventes

Alta disponibilidade: hardware e software

- Hardware
  - Cluster construído por componentes de baixo custo; é razoável ter hardware extra:
    - Mais nós do que os necessários
    - Material de rede redundante
- Software
  - "Watchdogs" e auto-teste
  - Recuperação
    - Recomeçar a aplicação – provavelmente a partir de um "checkpoint"
    - Migrar para outro nó

HA: Potencialidades e limitações

- Potencialidades
  - Acesso fácil a todos os recursos
  - Possibilidade de usar sistemas heterogéneos
  - Recursos que falham podem ser facilmente substituídos
- Limitação
  - Eficiência: muitas camadas de software, o paralelismo potencial nem sempre pode ser usado
  - Dificuldade em administrar

Gestão de sistemas heterogéneos

- Nós compatíveis mas com características diferentes
  - Torna-se um problema de equilíbrio de carga
- Nós não compatíveis
  - São necessários executáveis diferentes
  - Dados partilhados têm de estar em formato compatível
  - A migração é impossível

Sistemas de escalonamento

- A nível do kernel
  - Existem muito poucos que considerem o escalonamento a nível do "cluster"
- A nível utilizador
  - Distribuição do trabalho
  - Migração de processos
  - Equilíbrio de carga
  - Interacção com os utilizadores
  - Exemplos
    - CODINE, CONDOR, NQS, etc

Gestão de memória

- Objectivo
  - Usar toda a memória existente no cluster
- Aproximações possíveis
  - DSM (Distributed-Shared Memory) por software
    - Uso geral
  - Uso específico de memória remota livre
    - Fim específico
      - Paginação remota
      - Caches de sistemas de ficheiros ou “RAM disks”

DSM por software

- Camada de software
  - Permite a aplicações executando em nós diferentes partilhar regiões de memória
  - Relativamente transparente ao programador
- Estrutura do espaço de endereçamento
  - Espaço de endereçamento único
    - Completamente transparente para o programador
  - Áreas partilhadas
    - As aplicações têm de indicar que uma dada região é partilhada
    - Não é completamente transparente
    - Abordagem simples

DSM por software - problemas

- Consistência dos dados vs. Desempenho
  - Uma semântica estrita é muito ineficiente
- Localização dos dados
  - As soluções mais comuns baseiam-se num “nó dono” (owner node) que pode ser fixo ou variar dinamicamente
- Granularidade
  - Usualmente implementa-se um tamanho de bloco fixo
    - Restrições imposta pela MMU
    - Conduz a “falsa partilha”
    - Hipótese de granularidade variável

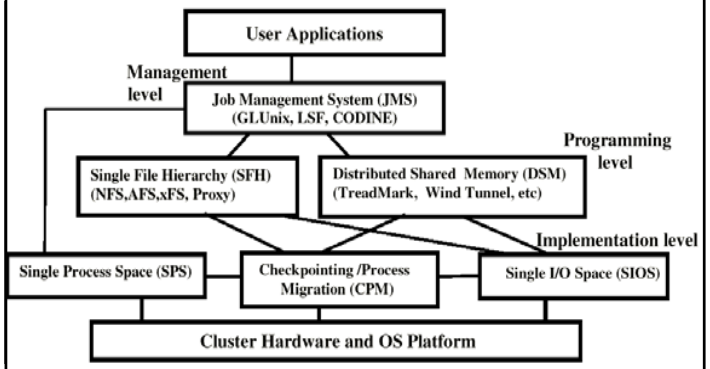
DSM - Problemas

- Sincronização
  - Mecanismos tipo “test and set” não podem ser usados
  - Soluções baseadas em semáforos (sobre troca de mensagens ...)
- Tolerância a falhas
  - Muito importante e raramente suportado
  - Mas há múltiplas cópias ...
- Heterogeneidade
  - Diferentes tamanhos de páginas
  - Representação de dados diferentes

Paginação remota

- Usar a memória livre como disco de paginação
  - Assume-se que muitos nós estão livres e que o acesso a disco é mais lento que o acesso a memória remota
  - Páginas alvo de substituição são enviadas para nós livres
  - Quando não existe memória livre utilizam-se os discos
  - Podem existir várias cópias – tolerância a falhas
- Exemplos
  - Global Memory Service (GMS) – Feeley 1995
  - Remote memory pager - Markatos 1996

Relações entre módulos de middleware



Níveis de SSI

- Níveis de abstracção do SSI:

Características do SSI

- Cada SSI tem uma fronteira (ou contexto).
- O suporte SI pode existir a diferentes níveis dentro do sistema, com alguns níveis suportados noutros.

SSI ao nível de aplicação e subsistema

Nível	Exemplos	Fronteira	Importância
Aplicação	Sistema batch e de gestão	Uma aplicação	O que o utilizador quer
Subsistema	Distributed DB, OSF DME, Lotus Notes, MPI, PVM	Um sub-sistema	SSI para todas as Aplicações do sub-sistema
Sistema de ficheiros	Sun NFS, OSF, DFS, NetWare, ...	Subárvore partilhada do sist. De ficheiros	Suporta implicitamente muitasaplicações e sub-sistemas
Toolkit	OSF DCE, Sun ONC+, Apollo Domain	Facilidades explícitas no toolkit: user, service name, time	Melhor nível de suporte para um sistema heterogéneo

SSI ao nível do kernel do SO

Nível	Exemplos	Fronteira	Importância
Kernel/ Camada OS	Solaris MC, Unixware MOSIX, Sprite, Amoeba /GLunix	Cada espaço de nomes: ficheiros, processes, pipes, devices, etc.	Suporte de Kernel Para aplicações e adm
Kernel interfaces	UNIX (Sun) vnode, Locus (IBM) vproc	Tipos de objectos no kernel: ficheiros, processos, etc.	Modulariza o código do SSI dentro do kernel
Memória virtual	Nenhum suporte para o kernel	Cada espaço de endereçamento	Pode simplificar a implementação de objectos no kernel
Microkernel	Mach, PAROS, Chorus, OSF/1AD, Amoeba	Serviços fora do microkernel	SSI implícito para todos os dispositivos de sistema

SSI ao nível Hardware

Nível	Exemplos	Fronteira	Importância
	Nível Aplicação e Susbsistema		
	Nível do Sistema de Operação		
memória	SCI, DASH	Espaço de memória	Melhor comunicação e sincronização
	Técnicas SCI e SMP	Memória e espaço de I/O	Menor overhead no I/O do cluster

(c) In search of clusters

Exemplos de sistemas e ferramentas SSI

- SSI ao nível do SO:
  - SCO NSC UnixWare;
  - Solaris-MC;
  - MOSIX, ….
- SSI ao nível do middleware:
  - PVM, TreadMarks (DSM), Glunix, Condor, Codine, Nimrod, ….
- SSI ao nível da aplicação:
  - PARMON, Parallel Oracle, ...

SSI por cima do SO

- 1. Construído como uma camada sobre um SO existente
  - Benefícios: torna o sistema facilmente transportável, segue as actualizações e upgrades do SO, e reduz o tempo de desenvolvimento.
  - i.e. novos sistemas podem ser construídos rapidamente mapeando os novos serviços em funcionalidades fornecidas pela camada inferior. e.g.: Glunix.
- 2. Construir o SSI a nível do kernel, “True Cluster OS”
  - Bom, mas **não se pode tirar partido dos melhoramentos do SO feitos pela equipa de desenvolvimento.**
  - E.g. Unixware, Solaris-MC, e MOSIX (1a. Versão, as últimas usam o Linux …) .

Sistemas de operação

- Pouco trabalho em SOs específicos para clusters.
- Tipicamente alguma forma de SSI integrado num SO convencional.
- Duas variantes:
  - Para fins de administração de sistema/ escalonamento de trabalhos - middleware que permite que cada nó disponibilize os serviços requeridos.
  - Nível-Kernel – por ex., acesso remoto a periféricos de forma transparente ou uso de armazenamento distribuído que é visto como um sistema de ficheiros único “clássico”.

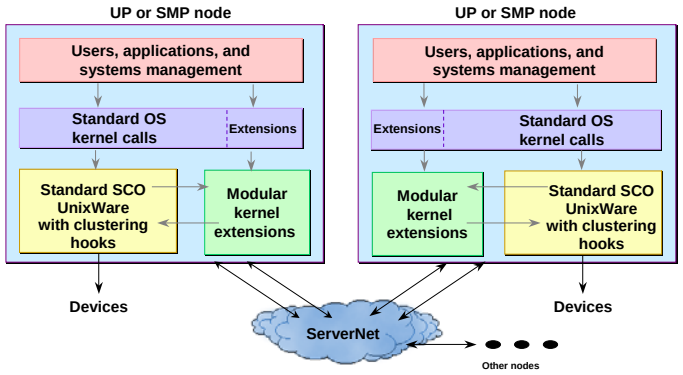
Exemplos – micro-kernels

- Outra abordagem é minimalista e usa micro-kernels - Exokernel é um destes sistemas.
- Desta forma, apenas o mínimo de funcionalidade é colocada no kernel – permite-se o carregamento de serviços a pedido.
  - Maximiza a memória física disponível através da remoção de funcionalidades desnecessárias;
  - O utilizador pode alterar as características do serviço: por exemplo, um escalonador específico para uma dada aplicação pode ser carregado, permitindo uma execução mais eficiente desta.

Que partes do SO são necessárias?

- Que configuração do SO?– porque é que o SO do nó fornece mais serviços às aplicações do que aqueles que ele provavelmente vai usar?
- Por exemplo, um utilizador pode alterar a personalidade do SO local. "strip down" para um kernel minimalista kernel para maximizar a memória física disponível
- Mecanismos para conseguir isto podem variar :
  - Uso de um novo kernel;
  - Ligação dinâmica de módulos com serviços no kernel.

SCO Non-stop Cluster for UnixWare



Como funciona o NonStop Cluster?

- Extensões modulares e "Hooks" para fornecer:
  - Visão "Clusterwide" do sistema de ficheiros;
  - Acesso transparente "Clusterwide" a periféricos;
  - Partilha transparent do "swap space";
  - IPC transparente a nível do cluster;
  - Comunicação inter-nós de alta velocidade;
  - Designação de processos a nível do cluster, migração,etc.;
  - Gestão automática de recursos quando um nó falha;
  - Portas TCP-IP "Clusterwide" de forma transparente;
  - Disponibilidade para aplicações;
  - Gestão de "membership" a nível do cluster e sincronização dos relógios;
  - Administração de sistema do Cluster System;
  - Equilíbrio de carga.

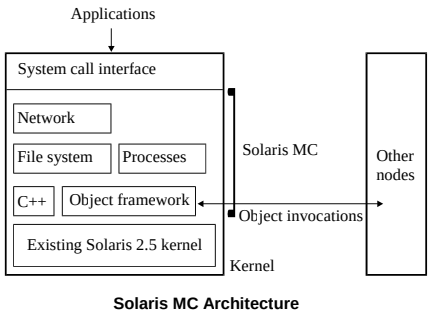
Exemplos – Solaris MC

- Uma versão do Sun Solaris para multi-computadores chamado Solaris MC.
- Incorpora tecnologia próprida da Sun, incluindo o uso de metodologias "object-oriented" no kernel - CORBA IDL.
- Consiste num pequeno conjunto de extensões ao kernel e uma biblioteca "middleware"- fornece serviços SSI ao nível dos periféricos:
  - Os processos executando num nó podem fazer acesso a dispositivos remotos como se fossem locais; também existe um sistema de ficheiros global e um espaço de processos.

Sun Solaris MC

- Artigo "Solaris MC: A High Performance Operating System for Clusters"
  - Arquitectura cluster com "high-speed interconnect"
  - Construído como uma camada de globalização sobre o kernel Solaris existente
  - Aspectos interessantes
    - estende o Solaris OS existente
    - Preserva a compatibilidade com a API/ABI do Solaris ABI/API
    - Suporte para alta disponibilidade
    - usa C++, IDL, CORBA no kernel
    - Tira partido da tecnologia desenvolvida no projecto Spring

Solaris-MC: Solaris for MultiComputers



Solaris MC Architecture

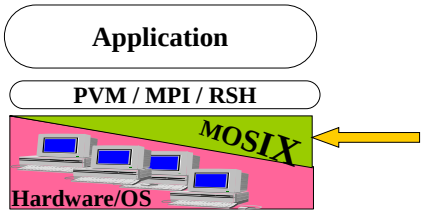
- Sistema de ficheiros global
- Gestão de processos a nível global globalized
- I/O e rede a nível global
- Suporte de comunicação entre objectos
- Suporte de High availability
- Sistema de ficheiros distribuído PXFS

http://www.sun.com/research/solaris-mc/

MOSIX: Multicomputer OS for UNIX

http://www.mosix.cs.huji.ac.il/

- **Uma camada (módulo) que fornece às aplicações a ilusão de executarem sobre um sistema único.**
- **Operações remotas são executadas como se fossem locais.**
- **Transparente para a aplicação – "user interface" não modificada.**



Ferramenta principal

Migração de processos com preempção →
qualquer processo, para qualquer lado, em qualquer altura

- **Supervisionada por algoritmos distribuídos** que respondem à disponibilidade de recursos – **transparentemente.**
- **Equilíbrio de carga** – migra processos de nós sobre-carregados para nós sub-carregados.
- **"Memory ushering"** - migrar processos de um nó que esgotou a sua memória, para evitar paginação e/ou swapping.

MOSIX for Linux no HUJI

- A configuração:
  - 50 Pentium-II 300 MHz
  - 38 Pentium-Pro 200 MHz (alguns são SMPs)
  - 16 Pentium-II 400 MHz (alguns são SMPs)
- 12 GB de RAM "cluster-wide"
- Ligação por the Myrinet 2.56 G.b/s
- **Red-Hat 6.0, Kernel 2.2.7**
- Upgrade: HW com Intel, SW com Linux
- Download MOSIX:
  - http://www.mosix.cs.huji.ac.il/

openSSI

- http://sourceforge.net/projects/ssi-linux
- Baseado no HP Non Stop Cluster for Unixware
- Objectivos
  - Disponibilidade, Escalabilidade e Facilidade de Gestão
- Componentes
  - Gestão de nós (entradas e saídas)
  - Single root/single init
  - Cluster file-system e Distributed Lock Manager (DLM)
  - Load balancing e equilíbrio de carga
  - Unificação do espaço de nomes de processos, dispositivos de IPC, periféricos e rede
- Usa resultados do projecto Cluster Infrastructure for Linux
  - http://ci-linux.sourceforge.net

Kerrighed

- http://www.kerrighed.org
- Modificação profunda do kernel Linux
- Suporte global dos recursos (processador, memória, disco)
- Checkpointing
- Suportado em
  - Shared virtual memory
  - Migração de processos leves
  - Sistema de ficheiros global

Linux Virtual Server

- http://www.LinuxVirtualServer.org
- Servidor escalável e tolerante a falhas construído a partir de um cluster
- Vocacionado para servidores da Internet
  - Extensão do stack TCP-IP: “load balancer” – o mundo vê a máquina como um servidor único
  - Algoritmos para seleccionar servidores dentro do cluster – “server pool” – WWW, FTP, Mail, DNS
  - “Back-end storage” – armazenamento partilhado para os servidores

Referências

- http://people.ac.upc.es/toni/CCTaskForce/SSI.html
- MOSIX
  - http://openmosix.sourceforge.net
- Kerrighed
  - www.kerrighed.org
- Open SSI
  - http://openssi.org
  - http://sourceforge.net/projects/ssic-linux/