Análise de duas plataformas para HPC: Z-series e Grids

PAULO RICARDO RODRIGUES TREZENTOS

Seminário de Computação de Alto Desempenho

(Mestrado em Gestão de Sistemas de Informação)
24 de Junho de 2003

Resumo

Conteúdo

Lista de Figuras

1 HPC - High Performance Computing

Nesta secç˜ao aborda-se o conceito de *HPC* e o seu enquadramento estratégico.

1.1 Perspectiva histórica

O surgimento de uma tecnologia é geralmente motivado pela existência da necessidade que ela visa colmatar. Desde cedo que as comunidades científicas encontram, nas suas diferentes áreas, problemas que apenas podem ser solucionados com recurso a grandes capacidades de processamento. O mercado empresarial é simultaneamente outro cliente de recursos computacionais, pois tem também diferentes tipos de aplicaç˜oes exigentes.
Ao segmento do mercado cujo objectivo é oferecer soluç˜oes para grandes necessidades de processamento, é usual classificar como **High Performance Computing**.

Para responder a este tipo de necessidade, desde cedo que surgiram plataformas especificamente direccionadas para este tipo de problemas.
Essas plataformas poder˜ao dividir-se em dois grandes grupos: um grupo de **soluções proprietárias** e um conjunto de soluç˜oes montadas a partir de **computadores vulgares**.
Historicamente, as grandes empresas do mercado HPC comercializam soluç˜oes chave-na-m˜ao, proprietárias, com um suporte pago de qualidade e vulgarmente chamados *main frames*. Estas soluç˜oes s˜ao enquadráveis no primeiro grupo.
Por outro lado, centros de investigaç˜ao e organizaç˜oes que n˜ao disp˜oe de fundos desenvolveram plataformas baseadas em soluç˜oes mais baratas, mas naturalmente sem qualquer tipo de suporte comercial.

A IBM é a empresa com maior passado na disponibilizaç˜ao de soluç˜oes HPC, tendo uma gama de computadores especialmente destinadas a este mercado. À sua bem sucedida gama de servidores S/390 veio, em 2001, suceder-lhe a chamada Z-Series (ou série Z) objecto desta análise.
A aquisiç˜ao de sistemas de grande porte à IBM[1] é característica de grandes empresas financeiras (bancos, seguradoras,...), bem como de outras cujo volume de negócios é, regra geral, elevado.
Outras empresas oferecem soluç˜oes neste mercado como a Sun, HP/Compaq, SGI e, antes da aquisiç˜ao pela SGI, a Cray. Esta última era conhecida pelas suas soluç˜oes de cálculo vectorial.

Organizaç˜oes que n˜ao podiam adquirir as dispendiosas soluç˜oes comerciais, cedo começaram a montar plataformas menos dispendiosas e com capacidades de processamento bastantes elevadas.
Estas soluç˜oes eram baseadas em componentes de vulgares PCs (*commodity components*) que ligados por uma determinada técnica permitem que um *cluster* de máquinas seja visualizado como uma única máquina virtual [JG01].
A utilizaç˜ao de componentes largamente produzidos, como os da indústria PC, permite usufruir de economias de escala.
Estas soluç˜oes s˜ao designadas de *clusters beowulf*, NOW - *network of workstations* ou simplesmente *clusters*, consoante o contexto.
Os *clusters beowulf* s˜ao uma soluç˜ao utilizada por várias universidades e que consiste interligar vários PCs através de uma rede Ethernet, utilizando software como PVM ou LAM/MPICH.

Ultimamente, e partindo do trabalho desenvolvido em *clusters* foi sugerido por [FK99] o desenvolvimento de uma rede distribuída de computadores, baseado em computaç˜ao *peer-to-peer*. Essa rede, à semelhança de uma rede de distribuiç˜ao de energia eléctrica, poderia fornecer um recurso (capacidade computacional ou *data storage*) a um cliente sem que este tivesse de saber de onde viria essa energia. Essa analogia com a rede eléctrica, do termo anglo-saxónico *power grid*, veio dar origem à designaç˜ao de **Grid**.

1.2 Enquadramento de Z-series e Grids

Como já foi referido, a **Z-series** é uma gama de servidores de grande porte comercializado pela IBM para empresas com grande necessidade de processamento.

[1] A IBM teve no ano de 2001 uma receito de 250 milh˜oes de EUROS, em Portugal.

A sua análise neste estudo é pertinente dado ser uma arquitectura recente que pode trazer algumas inovaç˜oes na área de arquitecturas avançadas.
Dado o seu custo, n˜ao é um forte candidato a uma utilizaç˜ao científica mas algumas das suas características podem ser incorporadas noutras plataformas.

As Grids s˜ao uma das arquitecturas que actualmente se encontram em maior desenvolvimento, verificável tanto pelo rápido desenvolvimento das soluç˜oes de software que a suportam, como pelas conferências em que é tema central.
Apesar deste tipo de soluç˜oes n˜ao ser muito bem suportado comercialmente, existe um conjunto de grandes organizaç˜oes, como a NASA e o CERN, que já a adoptaram como plataforma computacional. O seu suporte e desenvolvimento é realizado internamente por recursos dessas organizaç˜oes e n˜ao por *outsorcing*.

2 Grids

2.1 Conceito

Uma Grid é a partilha flexível, segura e coordenada de recursos entre grupos de indivíduos e instituiç˜oes [Fos01].
A analogia atrás referida com a rede eléctrica (*power grid*) é particularmente feliz se pensarmos que um utilizador de electricidade n˜ao sabe (nem precisa de saber) qual a proveniência da energia que consome quando liga a torradeira.
Idealmente, alguém que necessite de capacidade de computaç˜ao também n˜ao deveria necessitar saber de onde vem essa capacidade. Bastava submeter o seu trabalho à rede e esperar os resultados.
Na rede eléctrica isso acontece porque existe uma uniformizaç˜ao dos interfaces com a rede (fichas / tomadas), o que também é possível nas Grids com a *standardtização* do *middleware* que a comp˜oe.

Assim, permite a existência de uma comunidade ("organizaç˜ao virtual") que **partilha recursos** geograficamente distanciados à medida que perseguem objectivos comuns. Neste caso, a Grid pode prever a *ausência* de:

- **Localização central** - na Grid n˜ao é necessário haver uma centralizaç˜ao de um serviço em particular;
- **Controlo central** - a existência de um controlo central n˜ao é necessário já que o controlo é realizado de forma distribuído;
- **Omnisciência** - em nenhum momento, tem de existir uma entidade com o conhecimento de tudo o que se passa na Grid, derivado do ponto anterior;
- **Relação de confiança** - este é talvez um dos factores mais importantes dado que para duas entidades partilharem recursos entre elas numa Grid, n˜ao tem necessariamente de haver confiança entre as mesmas. Os mecanismos da Grid implementam a segurança necessária para estabelecer a confiança entre as mesmas.

O *aparecimento de Grids* apenas surge no final da década de 90 devido aos seguintes factores:

- O resultado prática da Lei de Moore é o surgimento de computadores de baixos custos, mas com capacidades computacionais muito elevadas
- A explos˜ao da Internet, e de tecnologias *wireless*, veio resultar na ligaç˜ao quase universal de todos os sistemas
- Hoje, o relacionamento entre instituiç˜oes de investigaç˜ao privilegia a cooperaç˜ao em detrimento da competiç˜ao
- A explos˜ao da capacidade das redes (cuja taxa de crescimento da largura de banda / latência é superior à lei de Moore) veio propiciar uma ligaç˜ao de qualidade em LANs e WANs

Em termos práticos, vejamos o seguinte exemplo:

"A organização A tem um centro de computação constituído por um conjunto de N computadores sobre Linux destinados à resolução de problemas de Monte Carlo. Uma segunda instituição, a B, tem um centro de computação constituído por N computadores AIX destinados a problemas de factorização de números compostos. A organização C tem uma rede de SGI destinado ao rendering de imagens."

Solução clássica
Cada uma das organizaç˜oes desenvolve software para resoluç˜ao dos seus problemas totalmente dependente da sua arquitectura.
Assim, a primeira organizaç˜ao desenvolvia sobre Linux utilizando uma biblioteca dependente deste sistema operativo, com comunicaç˜ao através de mensagens em formato proprietário da aplicaç˜ao. O mesmo aconteceria com a organizaç˜ao B e C.
A **vantagem** desta abordagem é que dado ser à medida da plataforma, pode ser altamente optimizada. Por outro lado, se no futuro desejar mudar de plataforma, requer um trabalho de migraç˜ao porventura complicado. Da mesma forma, que se desejar ter uma arquitectura híbrida com várias arquitecturas (diferente *hardware*, diferentes sistemas operativos), torna-se muito complicado.

Solução Beowulf Cluster
Nesta perspectiva, cada instituiç˜ao implementa os seus algoritmos sobre uma *camada que implementa a comunicação* entre nós do *cluster*. Assim, obtém-se independência em relaç˜ao à arquitectura pois é possível ter computadores de diferentes arquitecturas a comunicar entre eles.
A camada de comunicaç˜ao pode ser uma implementaç˜ao de MPI, como o LAM ou MPICH, ou PVM.
Nesta abordagem é vantajoso o facto de no mesmo centro de computaç˜ao poderem coexistir máquinas de diferentes arquitecturas.

Solução Grid
Nas anteriores soluç˜oes a capacidade média de computaç˜ao era:

$$CC_t = N * CC_n \tag{1}$$

(onde N = número de nós e CC_s é a capacidade computacional média de cada nó)

No exercício da actividade científica existem períodos onde a capacidade computacional é **totalmente absorvida** e outros em que a mesma **não é aproveitada**. Hiatos como estes s˜ao a raz˜ao da existência das Grids. Se se tornar possível partilhar uniformemente os recursos ent˜ao uma instituiç˜ao pode utilizar os hiatos computacionais da rede com outras organizaç˜oes, beneficiando dos hiatos das mesmas e obtendo dessa forma uma capacidade superior à sua capacidade unitária (CC_t).
Podíamos assim estabelecer a capacidade total agregada das três instituiç˜oes:

$$CC_{ta} = \sum_{i=A}^{C} N_i * CC_{ni} \tag{2}$$

(onde N = número de nós de i e CC_{ni} é a capacidade computacional média de cada nó em i)

Em (2) a capacidade total pode n˜ao coincidir com a capacidade disponível dado que a capacidade disponível é partilhada pelas três organizaç˜oes A,B e C[2].

Nesta secç˜ao foi enfatizado o **recurso partilhado** como sendo a capacidade computacional. Na verdade, a capacidade computacional e a capacidade de armazenamento (*data storage*) s˜ao os dois recursos mais populares, mas podemos pensar a partilha de recursos aplicada a sensores, redes ou outros elementos escassos.
Para implementar o conceito atrás enunciado tem vindo a ser delineado uma arquitectura que de seguida se apresenta.

[2] O facto de a capacidade m´edia dispon´ıvel beneficiar as organizaç˜oes que entraram para a Grid com capacidades computacionais mais baixas pode ser atenuada com pol´ıticas de *accounting* mais rigorosas

2.2 Arquitectura (Globus)

Para se poder precisar melhor a arquitectura, vamos focar agora numa implementaç˜ao do *middleware* necessário às Grids, o pacote *Globus*.
O *Globus* é a implementaç˜ao mais conhecida de Grids e está disponível sob uma licença *open-source*.

À semelhança dos *clusters*, a noç˜ao de Grid é conseguida através de um *middleware* que garante alguns dos principais serviços:

- *Gestor de recursos* - para facilitar a utilizaç˜ao de recursos remota
- *Gestor de dados* - para facilitar a distribuiç˜ao e acesso a dados
- *Segurança* - ao nível da autorizaç˜ao de acesso (em regime de *single login*, tipo Kerberos), autenticaç˜ao, encriptaç˜ao e n˜ao-repudio de identificaç˜ao
- *Contabilização* - de recursos utilizados com vista à manutenç˜ao da proporcionalidade de recursos investidos Vs recursos obtidos
- *Informação* - disponibilizaç˜ao de informaç˜ao sobre recursos disponíveis e suas características

A arquitectura idealizada emprega algumas das tecnologias já existentes e integra-as de uma forma consistente que permita obtermos os serviços atrás enunciados. A figura 1 apresenta-a numa perspectiva de alto-nível e a figura 2 em mais detalhe.

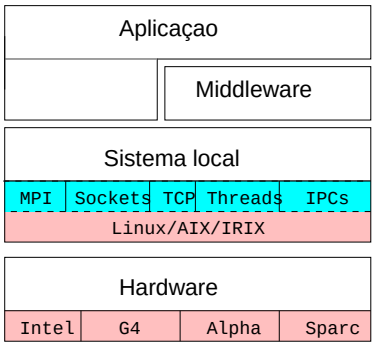

Figura 1: Arquitectura alto-nível de uma Grid

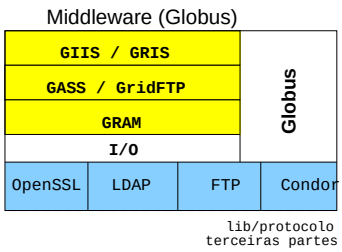

Figura 2: Arquitectura Grid em detalhe

Na figura 1 temos um nó da Grid representado em quatro camadas.
A superior, *Aplicação*, refere-se à implementaç˜ao do algoritmo que queremos executar, e que interage com a camada *Middleware (Globus)* através da chama a APIs.
Por sua vez, a camada *Middleware (Globus)* - que é concretizada através de servidores (*daemons*) presentes em cada nó - comunica com o sistema através de *system calls* e chamadas a bibliotecas. Note-se que as próprias aplicaç˜oes podem interagir com o sistema utilizando, por exemplo, MPICH ou PVM para comunicaç˜oes e *Condor* como *job scheduler*.

A camada *Middleware (Globus)* é pormenorizada em detalhe na figura 2 onde podemos observar que o Globus é constituído por 4 principais componentes: GIIS / GRIS, GASS / GridFTP, GRAM e I/O.
Na implementaç˜ao de cada um dos serviços atrás mencionado, o Globus recorre a tecnologias existentes (bibliotecas / protocolos) como OpenSSL, LDAP, etc...

Os serviços prestados pelo Globus podem ser acedidos utilizando ferramentas simples disponibilizadas pelo Globus ou, mais vulgarmente, acedidos directamente através de bibliotecas (APIs) .

Os serviços atrás mencionados têm os seguintes objectivos:

- *GRIS (Grid Resource Information Service)* - destina-se a disponibilizar informaç˜oes sobre um nó; a informaç˜ao encontra-se armazenada num servidor LDAP;
- *GIIS (Grid Index Information Service)* - o objectivo é fornecer informaç˜ao de uma forma agregado sobre toda a Grid. Por exemplo: *"Quantos nós têm um CPU >= 750 Mhz?"*. Recorre à GRIS de cada nó;
- *GASS (Global Access to Secondary Storage)* - permite a acesso e disponibilizaç˜ao de ficheiros dentro da Grid; é utilizado através da linguagem RSL (Resource Specification Language); é uma forma simples e multi-protocolo de transferir ficheiros;
- *GridFTP* - fornece a possibilidade de transferência de dados com alto desempenho e de forma fiável; aplicável a WANs e para ficheiros geralmente grandes;
- *GRAM (Grid Resource Allocation & Management)* - permite que programas sejam executados em recursos remotos, independentemente da sua localizaç˜ao;

Para além destes serviços existem outros, como *job dispatcher* ou comunicaç˜oes por mensagens, que podem ser fornecidos por terceiros devidamente adaptados a Grids. Concretamente, como *job dispatcher* temos o Condor-G (uma adaptaç˜ao do Condor para Grids) e como passagem de mensagens temos o MPICH-G2 (a adaptaç˜ao da implementaç˜ao de LAM, MPICH, mas adaptada a Grids).

2.2.1 Globus e PVM/MPI

Dado o grande número de projectos que utilizam PVM (*Parallel Virtual Machine*) e MPI (*Message Passing Interface*) para paralelizar problemas, é pertinente analisar a relaç˜ao deles com o Globus.

Como atrás foi referido, o Globus é o *middleware* que nos permite que a nossa aplicaç˜ao seja executada de uma forma autenticada e facilitada na infra-estratura disponibilizada.
Assim, podemos pensar no Globus como o facilitador da colocaç˜ao de um programa que queremos pôr a *correr* sobre uma *grid*, preocupando-se com os aspectos da gest˜ao dos recursos disponíveis, transferência de ficheiros, directório de informaç˜ao sobre a Grid e segurança (autenticaç˜ao e encriptaç˜ao).
Contudo o Globus n˜ao endereça dois tipos de problema: a **comunicação e sincronização** entre instâncias do mesmo programa a correr simultâneamente em diferentes nós e a **gestão de trabalhos** (*Job Dispatcher*) submetidos à Grid.

Se pretender que o programa que está a ser executado na Grid estabeleça comunicaç˜ao entre as suas várias instâncias que correm em paralelo, ent˜ao será necessário recorrer ao PVM ou a uma implementaç˜ao do MPI (como o LAM ou MPICH). Esse *software* garantirá que a passagem de mensagens entre nós ou sincronizaç˜ao é realizada eficazmente.
Saliente-se que existe uma implementaç˜ao de MPI, o MPICH, que tem uma vers˜ao especialmente desenvolvida para *grids* (Globus). Essa vers˜ao chama-se *MPICH-G2*.

Se por outro lado, pretender que os trabalhos submetidos à Grid tenham prioridades e sigam determinadas políticas, entao precisará de um software de gest˜ao de tarefas (*job dispatcher*), como o Condor.
Existe uma vers˜ao do Condor (designada por Condor-G) especialmente desenvolvida para o Globus.

2.3 Exemplos de Grids

Várias organizaç˜oes utilizam Grids para resolver problemas concretos da sua actividade.
Para já, é um domínio em que predomina a investigaç˜ao em detrimento do mercado empresarial.

2.3.1 *National Technology Grid*

A *National Technology Grid* [3] pretende ser a infraestrutura Grid americana, tendo sido criado pela NPACI (*National Partnership for Advanced Computational Infrastructure*) e a NCSA (*National Computational Science Alliance*.

[3] https://hotpage.npaci.edu/

É mantida por fundos governamentais e liga um conjunto grande de laboratórios e centros de investigaç˜ao. Cada centro disponibiliza à Grid a sua capacidade computacional: Univ. Texas (Cray SV1 e TE3), SDSC (IBM Blue Horizon e Sun HPC 10000), Univ. Michigan (AMD Cluster e IBM SP), Caltech (HP V2500).
Este sistema está disponível para investigadores seniores que precisem de capacidade computacional. Para a utilizarem, apenas precisam de se candidatar. Consoante o projecto e o seu mérito, diferente recursos s˜ao alocados.

A título de exemplo, no primeiro trimestre de 2002 estes foram alguns dos projectos que utilizaram a *National Technology Grid*:

- *First-Principles Investigation of the Thermodynamic and Kinetic Parameters Governing the Self-Assembly of Quantum Dot Arrays* - Mark Asta, Northwestern Univ. Máquina: Michigan IBM SP: 25000
- *Parallel Stabilized Finite Element Methods for Simulation of Aero-, Hydro- and Hemodynamic Problems* - Marek Behr, Rice University. Máquina/horas: Caltech HP Exemplar: 2000; SDSC Sun HPC 10000: 2000; SDSC CRAY T3E: 26000
- *3d Numerical Simulation of Ultra High Density Magnetic Recording* - Neal Bertram, University of California-San Diego. Máquina/horas: SDSC IBM SP (Blue Horizon): 30000
- *Giant (Satellite-Forming) Impacts During Planet Formation: Parallel Tree Code Simulations Using Smooth Particle* - Peter Bodenheimer, University of California-Santa Cruz. Máquina/horas: Hydrodynamics; SDSC CRAY T3E: 40000

Pegando num exemplo concreto, podemos analisar o caso do projecto *Lattice Gauge Theory on MIMD Parallel Computers* que utiliza a *National Technology Grid* para resolver soluç˜oes de equaç˜oes na área da física de partículas.
O projecto é muito exigente a nível computacional, estimando-se que requer 10^6 horas/ano de CPU. O código (em C e utilizando MPI) foi altamente optimizado para as máquinas da Grid, concretamente: Cray-T3E (SDSC), IBM SP (SDSC), Blue Horizon, SGI Origin 2000 e HP Exemplar.
A performance obtida é sintetizado no seguinte quadro (20^3 x 64 operaç˜oes em 8 CPUs)

- HP V Class: 60 Mflops/CPU
- Blue Horizon: 85 Mflops/CPU

2.3.2 *Projecto DataGrid*

Este é um projecto europeu financiado pelo V Quadro de apoio, em que vários parceiros avançam em algumas áreas das Grids. Tem um grupo de trabalho particularmente vocacionado para o *Data Storage* já que é uma das problemáticas mais importantes para o líder desse grupo: o CERN.

O CERN encontra-se a preparar o HLC esperado para 2005, com débitos à saída do acelerador de partículas na ordem dos Peta Bytes.
Esta informaç˜ao tem de ficar armazenada para poder ser acedida e produzidos os respectivos sumários que servir˜ao para consultas a serem realizadas pelos físicos. O armazenamento de uma forma distribuída é importante em duas vertentes. A primeira é poder armazenar-se localmente no CERN e ser acedido pelas aplicaç˜oes que fabricam os sumários. A segunda, é a possibilidade da informaç˜ao estar simultaneamente replicada e distribuída por centros regionais noutros países. A Grid pode fornecer ferramentas para estas funç˜oes como gestor de réplicas, meta-catálogos e ferramentas de transferência de alto débito.

2.3.3 Outros projectos

Outro exemplo é a rede, envolvendo o laboratório de Argonne, Univ. Michigan NCSA, UIUC e USC, destinada a simulaç˜oes de tremores de terra.

Esta rede é utilizada por equipas de engenheiros distribuídos por várias localizaç˜oes e que acedem a dados comuns e partilham capacidade computacional.

A Entropia, é uma empresa que disponibiliza recursos computacionais, e que neste caso se associou a dois laboratórios que fazem investigaç˜ao relacionada com o vírus HIV.
Tendo em média 1000 computadores domésticos ligados ao sistema, a Entropia cede essa capacidade aos laboratórios.
Este é um dos casos mais sintomáticos da utilizaç˜ao n˜ao exclusiva de computadores.
Apesar de mediaticamente ser uma das facetas mais citadas das Grids (utilizaç˜ao temporária de recursos cuja funç˜ao principal n˜ao é essa), acaba por n˜ao ser a filosofia empregue na maioria das Grids.

No caso da NASA, que a emprega para simulaç˜oes climáticas, os computadores da Grid s˜ao exclusivamente utilizados para esse fim. A justificaç˜ao é que logisticamente ficava mais caro administrar o parque informático instalado de estaç˜oes de trabalho de forma a que estivessem inseridos na Grid, dado a sua heterogeneidade, características dos utilizadores e dos sistemas, do que ter computadores (também sobre Intel) a servir a Grid.

3 Série Z

Nesta secç˜ao analisaremos a zSeries da IBM.

3.1 Conceito

A série Z é composta por duas gamas de servidores: o z800 e o z900.
N˜ao existem diferenças a nível de arquitectura entre ambos, podendo inclusive ser realizada a actualizaç˜ao de um z800 para um z900.
Assim, vamos basear a nossa análise no z900.

O z900 é uma plataforma com diversas configuraç˜oes de *hardware* que partilham uma arquitectura comum (ver secç˜ao 3.2).
Algumas das características interessantes desta plataforma:

- *Virtualização* - funcionalidade que permite que um único z900 contenha milhares de imagens independentes funcionando como sistemas operativos autónomos. O exemplo mais frequente é que estas imagens (*partitions*) sejam sistemas Linux. O IRD (*Intelligent Resourde Director*) é o serviço que optimiza a carga e os recursos entre as diferentes LPAR's - *Logical PARtitions*, nome dado a cada imagem;

- *Clusterização* - é possível juntar vários z900 (e/ou S/390) num cluster através de uma tecnologia proprietária da IBM, *Parallel Sysplex Cluster Technology*. O cluster permite *load balancing*, tolerância a falhas e partilha de recursos (como robots de tapes e discos). A ligaç˜ao é realizada através de um equipamento adicional, o CF - *Coupling Facilities*. A conectividade é realizada através de fibra sobre protocolos como ISC3 e ICB-3 atingido picos de 1 GB/s[4];

- *Criptografia* - estes modelos trazem implementaç˜oes de algoritmos criptográficos como RSA, DES e triplo DES por hardware. De fábrica, o z900 tem dois co-processadores criptográficos CMOS, mas é passível de ser expandido com placas adicionais (PCICC/PCICA). Segundo a empresa, podem chegar a processar 4300 transacç˜oes SSL por segundo.

A virtualizaç˜ao tem algumas características particulares. A figura 3 apresenta a sua arquitectura.
Como podemos verificar, na camada inferior temos o *Processor Resource / Systems Manager* que permite a partilha de canais físicos às várias LPARs.
Sobre cada LPAR, pode ser instalado um sistema operativo (Linux, OS/390, VSE/ESA,..) mas estamos limitados a 15 imagens. Por outro lado, se numa LPAR for instalado o Z/VM (um produto à parte) é possível ter

[4] A IBM n˜ao avanç a com valores da latˆencia, mas deve ser compar´avel a Myrynet, i.e. 10 μs.

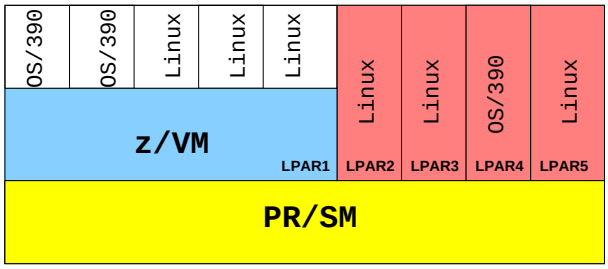

Figura 3: Arquitectura da virtualizaç˜ao de um z900

centenas de imagens Linux sobre um único z/VM com funcionalidades adicionais a nível de administraç˜ao de sistema: processos automáticos de criaç˜ao de imagens, controlo de recursos partilhados, QoS por recurso, etc... A IBM reclama em [IBM01] que a virtualizaç˜ao é uma grande mais valia para empresas que necessitam manter grandes parques de servidores Linux (como os ISPs) porque permite reduzir o TCO[5] e simultaneamente rentabilizar o CPU em templo *iddle*. Segundo o fabricante, já que os servidores têm picos de utilizaç˜ao em diferentes períodos consoante a sua utilizaç˜ao (mail, ficheiros, web,...) ent˜ao assim acaba-se por homogeneizar a sua utilizaç˜ao.

3.2 Arquitectura

Nativamente, o Z900 tem uma arquitectura de 64 bits. Isto traduz-se na utilizaç˜ao de registos gerais, instruç˜oes de código-máquina, endereçamento real, registos de controlo e barramentos a 64 bits.
A arquitectura é baseada num módulo MCM (*MultiChip Module*) que contém as unidades de processamento (PU - *Processor Units*), a estrutura de cache e a memória associada ao processador. O processador é, segundo dados da IBM, 1.3 a 1.4 vezes mais rápido que o G5 R16.
Na verdade, existem três tipos de MCM que variam consoante o número de processadores que incorporam (12 ou 20 PU).

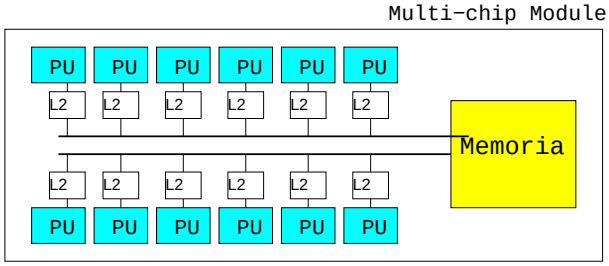

Figura 4: Previsivel MCM

A figura 4 apresenta um possível[6] *layout* da arquitectura MCM.

Comercialmente, a IBM apresenta os z900 em três segmentos:

- **z900 modelos 101-109** - modelos com MCM's de 12 PUs, dois bus de memória e capacidade de armazenamento de 5 a 32 GB. Um PU tem um ciclo de relógio de 1.3 ns.

[5] Total Cost of Ownership

[6] É poss´ı vel que a organizaç ˜ao n˜ao seja exactamente este j´a que a IBM n˜ao a disponibiliza, mas apenas os componentes que constituem o MCM.

- **z900 modelos 110-116 e 210-216** - modelos com MCM's de 20 PUs, quatro bus de memória e capacidade de armazenamento até 64 GB. Os modelos 110-116 têm PUs com um ciclo de relógio de 1.3 ns, mas existem uma variaç˜ao dos mesmos - chamada de "201-216", que tem um ciclo de relógio de 1.09 ns.
- **z900 modelos 1C1-1C9 e 2C1-2C9** - este segmento é semelhante ao segmento anterior mas está limitado a uma variaç˜ao de 1 a 9 processadores. A variaç˜ao com o primeiro segmento apresentado (*101-109*) é ter capacidade de expans˜ao até 16 processadores por software[7].

Curiosamente, se analisarmos os modelos 101-109, por exemplo, fisicamente s˜ao iguais. O que varia é a forma como saem parametrizados de fábrica em relaç˜ao à funç˜ao dos PU's. Como atrás foi dito, estas MCM's trazem 12 PU's que podem ser atribuídos às seguintes funç˜oes:

- *CP (Central Processors)* - s˜ao utilizados como processadores genéricos e s˜ao o que mais contribuem para o desempenho real da máquina. Quando se referencia um modelo por "4-way" refere-se ao facto de ter 4 PUs como CPs.
- *SAP (System Assist Processors)* - auxiliares para processamento I/O
- *ICF (Internal Coupling Facilities)* - auxiliares para a *clusterização*. Quem n˜ao quiser investir no equipamento atrás descrito (CF) pode alocar esta tarefa a um dos processadores
- *IFL (Internal Facility for Linux)* - processador auxiliar exclusivamente destinado a Linux

Cada modelo tem uma variaç˜ao na forma como os 12 PU's s˜ao utilizados. Assim, um 101 tem originalmente a seguinte configuraç˜ao: 1 CP, 2 SAP (+ 1 opcional) e, opcionalmente, até 8 ICF ou IFL. Depois sobre esta configuraç˜ao o cliente pode adicionar IFL, SAP ou ICFs.
O modelo 102 diverge porque, como indica o terceiro digito (2), tem dois CPs (*Central Processors*).

O z900 vem em duas caixas[8], chamadas de A-frame e Z-frame, acopladas e que integram espaço para *slots* de IO (nI/o cage) e para o CEC (onde est˜ao os processadores e a memória)

Referências

[FK99] I. Foster and C. Kesselman. *The Grid: Blueprint for a New Computing Infrastructure*. Morgan Kaufmamn, 1999.

[Fos01] Ian Foster. The anatomy of the Grid: Enabling scalable virtual organizations. *Lecture Notes in Computer Science*, 2150:1–??, 2001.

[IBM01] IBM. Ibm zseries 900 and z/os reference guide. Technical report, IBM, 2001.

[JG01] Paulo Trezentos José Guimar˜aes. Spino: A distributed architecture for massive text storage. *ICEIS*, 1:244–248, 2001.

[7] *Isto ´e, os processadores est˜ao sempre presentes, mas desactivados*
[8] Ou arm´arios, consoante a perspectiva