KOMPUTASI PARALLEL



Oleh :

Muhammad Isa Hidayatullah

1210651073

FAKULTAS TEKNIK

TEKNIK INFORMATIKA

UNIVERSITAS MUHAMMADIYAH JEMBER

Komputasi paralel adalah salah satu teknik melakukan komputasi secara bersamaan dengan memanfaatkan beberapa komputer independen secara bersamaan. Ini umumnya diperlukan saat kapasitas yang diperlukan sangat besar, baik karena harus mengolah data dalam jumlah besar (di industri keuangan, bioinformatika, dll) ataupun karena tuntutan proses komputasi yang banyak. Kasus kedua umum ditemui di kalkulasi numerik untuk menyelesaikan persamaan matematis di bidang fisika (fisika komputasi), kimia (kimia komputasi) dll.

Untuk melakukan aneka jenis komputasi paralel ini diperlukan infrastruktur mesin paralel yang terdiri dari banyak komputer yang dihubungkan dengan jaringan dan mampu bekerja secara paralel untuk menyelesaikan satu masalah. Untuk itu diperlukan aneka perangkat lunak pendukung yang biasa disebut sebagai middleware yang berperan untuk mengatur distribusi pekerjaan antar node dalam satu mesin paralel. Selanjutnya pemakai harus membuat pemrograman paralel untuk merealisasikan komputasi. Tidak berarti dengan mesin paralel semua program yang dijalankan diatasnya otomatis akan diolah secara parallel.

Di dalam komputasi parallel ada yang dinamakan dengan pemrograman parallel. Pemrograman paralel adalah teknik pemrograman komputer yang memungkinkan eksekusi perintah/operasi secara bersamaan (komputasi paralel), baik dalam komputer dengan satu (prosesor tunggal) ataupun banyak (prosesor ganda dengan mesin paralel) CPU. Bila komputer yang digunakan secara bersamaan tersebut dilakukan oleh komputer-komputer terpisah yang terhubung dalam suatu jaringan komputer lebih sering istilah yang digunakan adalah sistem terdistribusi (distributed computing).

Tujuan utama dari pemrograman paralel adalah untuk meningkatkan performa komputasi. Semakin banyak hal yang bisa dilakukan secara bersamaan (dalam waktu yang sama), semakin banyak pekerjaan yang bisa diselesaikan. Analogi yang paling gampang adalah, bila anda dapat merebus air sambil memotong-motong bawang saat anda akan memasak, waktu yang anda butuhkan akan lebih sedikit dibandingkan bila anda mengerjakan hal tersebut secara berurutan (serial). Atau waktu yg anda butuhkan memotong bawang akan lebih sedikit jika anda kerjakan berdua.

Performa dalam pemrograman paralel diukur dari berapa banyak peningkatan kecepatan (speed up) yang diperoleh dalam menggunakan tehnik paralel. Secara informal, bila anda memotong bawang sendirian membutuhkan waktu 1 jam dan dengan bantuan teman, berdua anda bisa melakukannya dalam 1/2 jam maka anda memperoleh peningkatan kecepatan sebanyak 2 kali.

Parallel processing berbeda dengan multitasking, yaitu satu CPU mengeksekusi beberapa program sekaligus. Parallel processing disebut juga parallel computing. Pada system komputasi parallel terdiri dari beberapa unit prosesor dan beberapa unit memori. Ada dua teknik yang berbeda untuk mengakses data di unit memori, yaitu shared memory address dan message passing. Berdasarkan cara mengorganisasikan memori ini computer parallel dibedakan menjadi shared memory parallel machine dan distributed memory parallel machine.

**Arsitektur Komputer Paralel**

Menurut seorang Designer Processor, taksonomi Flynn, Arsitektur Komputer dibagi menjadi 4 baguan, yaitu :

1. SISD ( Single Instruction Single Data Stream )

Jenis Komputer yang hanya memiliki satu prosesor dan satu instruksi yang dieksekusi secara serial.

1. SIMD ( Single Instruction Multiple Data Stream )

Jenis komputer yang memiliki lebih dari satu prosesor, tetapi komputer ini hanya mengeksekusi satu instruksi secara paralel pada data yang berbeda pada level lock-step.

1. MISD ( Multiple Instruction Single Data Stream )

Jenis komputer yang memiliki satu prosesor dan mengeksekusi beberapa instruksi secara paralel tetapi di dalam praktiknya tidak ada komputer yang dibangun dengan arsitektur ini karena sistemnya tidak mudah dipahami, sampai saat ini pun belum ada komputer yang menggunakan arsitektur jenis ini

1. MIMD ( Multiple Instruction Multiple Data Stream )

Jenis komputer yang memiliki lebih dari satu prosesor dan mengeksekusi lebih dari satu instruksi secara paralel. Tipe komputer ini yang paling banyak digunakan untuk membangun komputer paralel, bahkan banyak supercomputer yang menerapkan arsitektur ini, karena model dan konsepnya yang tidak terlalu rumit untuk dipahami.

**Pengantar Thread Programming**

Sebuah thread di dalam pemrograman komputer adalah sebuah informasi terkait tentang penggunaan sebuah program tunggal yang dapat menangani beberapa pengguna secara bersamaan.Thread ini memungkinkan program untuk mengetahui bagaimana user masuk ke dalam program secara bergantian dan user akan masuk kembali menggunakan user yang berbeda. Multiple thread dapat berjalan bersamaan dengan proses lainnya membagi sumberdaya menjadi memori, disaat proses lain tidak membaginya.

**Pengantar Message Passing, Open MP**

Message Passing merupakan sebuah bentuk dari komunikasi yang digunakan di komputasi paralel, OOT (Object Oriented Programming) atau Pemrograman Berbasis Objek dan komunikasi interproses

**Pengantar Pemrograman CUDA GPU**

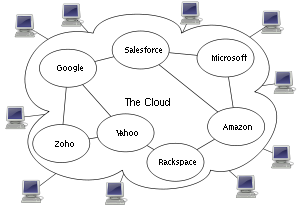
Sebelum saya membahas tentang CUDA, saya akan membahas GPU terlebih dahulu. GPU dalah sebuah processor khusus untuk memepercepat dan mengubah memori untuk mempercepat pemrosesan gambar. GPU ini sendiri biasanya berada di dalam graphic card komputer ataupun laptop

CUDA(Compute Unified Device Architecture) adalah suatu skema yang dibuat oleh NVIDIA agar NVIDIA selaku GPU (Graphic Processing Unit) mampu melakukan komputasi tidak hanya untuk pengolahan grafis namun juga untuk tujuan umum. Jadi dengan adanya CUDA kita dapat memanfaatkan banyak prosesor dari NVIDIA untuk melakukan proses perhitungan ataunpun komputasi yang banyak.

**Cloud Computing**

Cloud Computing adalah gabungan dari pemanfaatan teknologi (komputasi) dan pengembangan berbasis internet (awan). Cloud Computing merupakan sebuah metode penggabungan dari berbagai pemanfaatan beberapa teknologi dimana kemampuan TI disediakan sebagai layanan berbasis internet.

Contoh Cloud Computing seperti Ymail atau Gmail. Anda tidak perlu software atau hardware (server) untuk menggunakannya. Semua Konsumen hanya perlu koneksi Internet dan mereka dapat mulai mengirimkan email. Software manajemen email dan server semuanya ada di cloud (internet) dan secara total di kelola oleh provider seperti Yahoo, Google, etc. Konsumen hanya perlu menggunakan software itu sendiri dan menikmati manfaatnya.

[](https://musthopz.files.wordpress.com/2013/10/11.png)

Perhatikan titik-titik komputer/server sebagai gabungan dari sumber daya yang akan dimanfaatkan. Lingkaran-lingkaran sebagai media aplikasi yang menjembatani sumber daya dan cloud-nya adalah internet. Semuanya tergabung menjadi satu kesatuan dan inilah yag dinamakan cloud computing.Analoginya adalah, “Jika anda membutuhkan Susu, anda tidak perlu membeli sapi. Karena yang anda butuhkan adalah Susunya (manfaatnya) bukan Sapinya (Software/hardware). Itu sama saja dengan anda menggunakan software atau hardware seperti mengirim email, dan lain-lain tanpa harus memiliki Aplikasi Penyedia layanannya.

Contoh Cloud Computing

Cloud computing saat ini sedang populer, namun apa contoh dari Cloud Computing itu, berikut adalah lima contoh dari sekian banyak Cloud Computing saat ini:

1) Email

Email adalah salah satu alat komunikasi yang sudah cukup lama ada. Dalam email tersimpan data-data pada saat kita mengirimkan atau menerima data. Yang berupa tulisan atau file. Data-data tersebut dapat kita akses sewaktu-waktu kita perlukan , tanpa kita harus menyimpan data tersebut pada komputer pribadi sendiri. Orang lain juga dapat mengakses data tersebut tapi tentunya yang hanya menerima email itu saja.

2) Data storage online

Penyimpanan data pada komputer personal tentu akan santa terbatas. Jika harus membeli server sendiri amat sangat mahal sekali. Oleh sebab itu saat ini kita dapat menyewa atau bahkan ada yang menyediakan server gratis yang dapat kita akses secara online. Contohnya adalah Humyo, ZumoDrive, Microsoft’s SkyDrive, S3 from Amazon, dan masih banyak yang lain.

3) Colaboration Tools

Penkolaborasian data sering kali diperlukan. Karena data yang ingin kita simpan bermacam-macam jenisnya dan fungsinya. ada banayk tools yang dapat digunakan. Conothnya adalah Spicebird, Mikogo, Stixy and Vyew t.

4) Virtual Office

Sering kita memerlukan office untuk memproses data-data. Saat ini kita dapat menggunakan office tidak hanya yang sudah terinstall namun kita juga dapat menggunakan office yang disediakan secara online. Contohnya antara lain Ajax13, ThinkFree and Microsoft’s Office Live.

5) Kekuatan ekstra processing

Bila membutuhkan kekuatan untuk memproses secara cepat tanpa perlu membeli perangkat tambahan maka salah satu solusinya adalah Amazon’s EC2 virtual computing ini juga dapat diatur sesuai dengan kebutuhan individu masing -masing orang. Contoh yang lain adalah AbiCloud, Elastichosts and NASA’s Nebula platform.

**Ciri – Ciri Cloud Computing**

Cloud computing saat ini sedang terkenal, namun apa ciri – ciri dari Cloud Computing itu, berikut adalah 7 ciri Cloud Computing yaitu:

1. Komputasi Otonom

Sistem komputer yang mampu mengatur atau me-manajemen sendiri secara otomatis.

1. Client-server model

Komputasi client-server mengacu secara luas untuk setiap aplikasi terdistribusi yang membedakan antara penyedia layanan (server) dan pengguna layanan (klien).

1. Grid computing

Suatu bentuk komputasi terdistribusi dan komputasi paralel, dimana sebuah ‘super komputer dan virtual’ terdiri dari sekelompok jaringan, komputer lebar ditambahkan serta bertindak untuk melakukan tugas yang demikian besar.

1. Mainframe komputer

Komputer luar biasa yang banyak digunakan oleh organisasi-organisasi besar untuk aplikasi kritis, biasanya proses transaksi pemrosesan data massal seperti sensus, statistik industri dan konsumen, perencanaan sumber daya perusahaan, dan keuangan.

1. Utility komputasi

Dengan kemasan sumber daya komputasi, seperti perhitungan dan penyimpanan.

1. Peer-to-peer

Arsitektur yang didistribusikan tanpa perlu koordinasi pusat, dengan peserta yang pada saat yang sama pihak pemasok dan konsumen sumber daya (berbeda dengan model client-server tradisional).

1. Berorientasi layanan komputasi

Cloud computing menyediakan layanan yang berkaitan dengan komputasi sementara, dengan cara timbal balik, komputasi berorientasi layanan terdiri dari teknik komputasi yang beroperasi pada perangkat lunak layanan.

Tingkat Layanan

Cloud computing mempunyai 3 tingkatan layanan yang diberikan kepada pengguna, yaitu:

1. Infrastructure as service

Hal ini meliputi Grid untuk virtualized server, storage & network. Contohnya seperti Amazon Elastic Compute Cloud dan Simple Storage Service.

1. Platform as a service

Hal ini memfokuskan pada aplikasi dimana dalam hal ini seorang developer tidak perlu memikirkan hardware dan tetap fokus pada pembuatan aplikasi tanpa harus mengkhawatirkan sistem operasi, infrastructure scaling, load balancing dan lain-lain. Contohnya yang sudah mengimplementasikan ini adalah Force.com dan Microsoft Azure investment.

1. Software as a service

Hal ini memfokuskan pada aplikasi dengan Web-based interface yang diakses melalui Web Service dan Web 2.0. Contohnya adalah Google Apps, SalesForce.comdan aplikasi jejaring sosial seperti FaceBook.

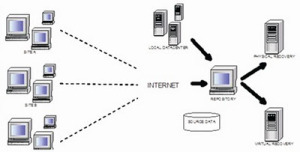
Para investor sedang mencoba untuk mengeksplorasi adopsi teknologi cloud computing untuk dijadikan bisnis seperti Google dan Amazon yang sudah punya penawaran khusus pada teknologi cloud. Dan juga Microsoft dan IBM tidak mau kalah dalam hal ini, mereka juga sudah menginvestasikan jutaan dolar untuk hal ini.

Bisa dipastikan ke depannya cloud computing ini akan menjadi sebuah trend, standar teknologi akan menjadi lebih sederhana karena ketersediaan dari layanan cloud.

**Cara Kerja Cloud Computing**

Secara garis besar, berikut merupakan cara kerja penyimpanan data dan replikasi data pada pemanfaatan teknologi Cloud Computing.

Ketika pengguna mengakses awan untuk sebuah website populer, banyak hal yang bisa terjadi. Pengguna IP misalnya dapat digunakan untuk menetapkan dimana pengguna berada (geolocation). DNS jasa kemudian dapat mengarahkan pengguna ke sebuah cluster server yang dekat dengan pengguna sehingga situs bisa diakses dengan cepat dan dalam bahasa lokal mereka. Pengguna tidak login ke server, tetapi mereka login ke layanan mereka menggunakan dengan mendapatkan id sesi dan / atau cookie yang disimpan dalam mereka browser .

[](https://musthopz.files.wordpress.com/2013/10/3.jpg)

Apa yang user lihat pada browser biasanya akan datang dari sekelompok web server. Webservers menjalankan perangkat lunak yang menyajikan pengguna dengan interface yang digunakan untuk mengumpulkan perintah atau instruksi dari pengguna (klik, mengetik, dll upload) Perintah-perintah ini kemudian diinterpretasikan oleh webservers atau diproses oleh server aplikasi. Informasi kemudian disimpan pada atau diambil dari database server atau file server dan pengguna kemudian disajikan dengan halaman diperbarui. Data di beberapa server disinkronisasikan di seluruh dunia untuk akses global cepat dan juga untuk mencegah kehilangan data.

Sebuah contoh sederhana dari komputasi awan email seperti Yahoo! dan GMAIL Anda tidak membutuhkan software atau server untuk menggunakannya. Semua konsumen akan membutuhkan hanya koneksi internet dan Anda dapat mulai mengirim email. Server dan perangkat lunak manajemen email adalah semua di atas awan (internet) dan benar-benar dikelola oleh operator selular awan Yahoo, Google konsumen bisa menggunakan perangkat lunak sendirian dan menikmati manfaat, di Analogi adalah “Jika Anda hanya perlu susu,apakah Anda akan membeli sapi?”.Semua pengguna atau konsumen butuhkan adalah untuk mendapatkan manfaat menggunakan perangkat lunak atau perangkat keras dari komputer seperti mengirim email dan sebagainya.

**Implementasi Cloud Computing**

Adopsi Teknologi IT di kalangan Usaha Kecil Dan menengah (UKM) memang terhitung masih cukup rendah. Di lain sisi penggunaan Teknologi IT dipandang memiliki peranan yang cukup besar bagi perkembangan UKM. Dengan memanfaatkan TI, UKM bisa menyamai akselerasi pertumbuhan usaha skala besar. Namun di sisi yang lain investasi di bidang IT bagi UKM memang terasa sangat berat, karena memerlukan biaya dan tenaga ahli bidang IT yang cukup mahal. Untuk membeli hardware sebagai pendukung penerapan TI bagi UKM saja sudah cukup besar, belum lagi software, aplikasi dan lain sebagainya.

Bagi kalangan UKM yang sudah mulai melirik TI masih saja ada kekhawatiran misalnya saja seperti belum tersedianya sumber daya yang mencukupi untuk membeli, memelihara serta mengamankan sistem informasi mereka sendiri. Investasi besar inilah yang masih menjadi momok bagi sebagian besar pelaku UKM untuk mau mengaplikasikan Teknologi Informasi bagi pengembangan bisnisnya.

Solusi murah dan efisien penerapan IT bagi UKM yang saat ini berkembang adalah Cloud Computing. Seberapa mahal layanan ini? Untuk skala bisnis, layanan cloud computing terbilang cukup murah karena layanan ini menggunakan mekanisme economies of scale, “Semakin banyak yang ikut menggunakan, semakin baik”. Telkom misalnya, dengan 4-5juta per bulan para pelaku UKM sudah dapat menikmati layanan yang mereka tawarkan, meliputi Infrastructure as a Service (IAAS), Platform as a Service (PAAS), hingga Software as a Service (SAAS).

Sebagai gambaran, beberapa contoh aplikasi cloud computing berbasis platform as as services (PAAS) di antaranya e-UKM, aplikasi untuk BPR (Bank Perkreditan Rakyat), aplikasi untuk pengelolaan koperasi, pendidikan, dan lainnya.

Nah, disinilah cloud computing itu berperan. Perusahaan Microsoft telah menyediakan aplikasi CRM yang dapat langsung digunakan oleh perusahaan yang membutuhkan tadi. Perusahaan yang membutuhkan itu tinggal menghubungi perusahaan Microsoft untuk menyambungkan perusahaannya (dalam hal ini melalui internet) dengan aplikasi CRM & tinggal memakainya. Dan pembayaran dilakukan per bulan, per triwulan, per semester, per tahun atau sesuai kontrak yang dibuat. Jadi, perusahaan yang membutuhkan aplikasi CRM tadi, tidak perlu melakukan investasi awal untuk pembelian hardware server dan tenaga ahli TI. Itulah salah satu manfaat dari cloud computing yang dapat menghemat anggaran suatu perusahaan.

Kelebihan dan kekurangan Cloud Computing

1. Menghemat biaya investasi awal untuk pembelian sumber daya.
2. Bisa menghemat waktu sehingga perusahaan bisa langsung fokus ke profit dan berkembang dengan cepat.
3. Membuat operasional dan manajemen lebih mudah karena sistem pribadi/perusahaan yang tersambung dalam satu cloud dapat dimonitor dan diatur dengan mudah.
4. Menjadikan kolaborasi yang terpercaya dan lebih ramping.
5. Mengehemat biaya operasional pada saat realibilitas ingin ditingkatkan dan kritikal sistem informasi yang dibangun.

Kekurangan Cloud Computing

Komputer akan menjadi lambat atau tidak bisa dipakai sama sekali jika internet bermasalah atau kelebihan beban. Dan juga perusahaan yang menyewa layanan dari cloud computing tidak punya akses langsung ke sumber daya. Jadi, semua tergantung dari kondisi vendor/penyedia layanan cloud computing. Jika server vendor rusak atau punya layanan backup yang buruk, maka perusahaan akan mengalami kerugian besar.

**Grid Computing**

**Pengertian Grid Computing**

Grid Computing atau Komputasi Grid merupakan salah satu dari tipe Komputasi Paralel, adalah penggunaan sumber daya yang melibatkan banyak komputer terpisah secara geografis namun tersambung via jalur komunikasi (termasuk Internet) untuk memecahkan persoalan komputasi skala besar. Semakin cepat jalur komunikasi terbuka, maka peluang untuk menggabungkan kinerja komputasi dari sumber-sumber komputasi yang terpisah menjadi semakin meningkat. Dengan demikian, skala komputasi terdistribusi dapat ditingkatkan secara geografis lebih jauh lagi, melintasi batas-batas domain administrasi yang ada.

Sebagai sistem terdistribusi dengan non-interaktif beban kerja yang melibatkan sejumlah besar file. Apa yang membedakan grid computing dari konvensional sistem komputasi kinerja tinggi seperti komputasi cluster adalah grid lebih cenderung longgar digabungkan, heterogen, dan geografis.

[](https://musthopz.files.wordpress.com/2013/10/2.png)

Grid computing merupakan salah satu jenis dari komputasi modern. Grid computing adalah arsitektur TI baru yang menghasilkan sistem informasi perusahaan yang berbiaya rendah dan lebih adaptif terhadap dinamika bisnis. Dengan grid computing, sejumlah komponen hardware dan software yang modular dan independen akan dapat dikoneksikan dan disatukan untuk memenuhi tuntutan kebutuhan bisnis. Lebih jauh, dari sisi ekonomi, implementasi grid computing berarti membangun pusat komputasi data yang tangguh dengan struktur biaya variatif yang bias disesuaikan dengan kebutuhan.

Secara singkat, grid computing berarti menyatukan seluruh sumberdaya TI ke dalam sekumpulan layanan yang bisa digunakan secara bersama-sama untuk memenuhi kebutuhan komputing perusahaan. Infrastruktur gridcomputing secara kontinyu menganalisa permintaan terhadap sumberdaya dan mengatur suplai untuk disesuaikan terhadap permintaan tersebut. Dimana data disimpan atau computer mana yang memproses permintaan tidak perlu dipikirkan. Sebagaimana arus listrik; untuk memanfaatkannya, tempat pembangkit atau bagaimana pengabelan jaringan listrik tidak perlu diketahui.

Jadi Kesimpulanya

Grid computing adalah model generasi selanjutnya untuk komputasi perusahaan berbasis virtualisasi dan provisioning bagi setiap sumberdaya TI. Grid computing menjanjikan peningkatan utilitas dan fleksibilitas yang lebih besar untuk sumberdaya infrastruktur, aplikasi dan informasi. Oracle 10g telah berbasis grid computing, sehingga perusahaan yang menginginkan kemajuan dan perbaikan kinerja bisnis berbiaya rendah bagi aplikasi transaksional, business intelligence dan knowledge management dapat menggunakan solusi grid computingdari Oracle. Khusus bagi pelanggan Oracle sekarang ini, adopsi gridcomputing hanya berupa adopsi generasi selanjutnya dari software yang telah sukses dijalankan sebelumnya. IDC juga meyakini bahwa Oracle 10g cukup diperhitungkan oleh banyak perusahaan yang berkeinginan yang sama. Pelaku bisnis cukup mengadopsi teknologi grid dengan investasi minimal, kegagalan nol, dan ROI cepat.

**Contoh Grid Computing**

Oracle 10g

Untuk software Oracle 10g adalah singkatan dari grid. Fokus dari versi baru Oracle ini adalah untuk memudahkan perusahaan menyederhanakan proses implementasi grid computing di luar kerangka kerja komputasi akademik, teknik, riset dan saintifik. Oracle 10g meliputi:

* Oracle Database 10g
* Oracle Aplication Server 10g
* Oracle Enterprise Manaer 10g

**Konsep Grid Computing**

1. Sumberdaya Infrastruktur

Mencakup hardware seperti penyimpan, prosesor, memori, dan jaringan; juga software yang didisain untuk mengelola hardware ini, seperti database, manajemen penyimpan, manajemen sistem, server aplikasi dan system operasi.

2. Sumberdaya Aplikasi

Adalah perwujudan logika bisnis dan arus proses dalam software aplikasi. Sumberdaya yang dimaksud bisa berupa aplikasi paket atau aplikasi buatan, ditulis dalam bahasa pemrograman, dan merefleksikan tingkat kompleksitas. Sebagai contoh, software yang mengambil pesanan dari seorang pelanggan dan mengirimkan balasan, proses yang mencetak slip gaji, dan logika yang menghubungkan telepon dari pelanggan tertentu kepada pihak tertentu pula.

3. Sumberdaya Informasi

Saat ini, informasi cenderung terfragmentasi dalam perusahaan, sehingga sulit untuk memandang bisnis sebagai satu kesatuan. Sebaliknya, grid computing menganggap informasi adalah sumberdaya, mencakup keseluruhan data pada perusahaan dan metadata yang menjadikan data bisa bermakna. Data bias berbentuk terstruktur, semi-terstruktur, atau tidak terstruktur, tersimpan di lokasi manapun, seperti dalam database, sistem file lokal.

Prinsip Kerja Grid Computing

Dua prinsip kerja utama grid computing yang membedakannya dari arsitektur komputasi yang lain, semisal mainframe, klien-server, atau multi-tier: virtualisasi dan provisioning.

1. Virtualisasi

Setiap sumberdaya (semisal komputer, disk, komponen aplikasi dan sumber informasi) dikumpulkan bersama-sama menurut jenisnya, lalu disediakan bagi konsumen (semisal orang atau program software). Virtualisasi berarti meniadakan koneksi secara fisik antara penyedia dan konsumen sumberdaya, dan menyiapkan sumberdaya untuk memenuhi kebutuhan tanpa konsumen mengetahui bagaimana permintaannya bisa terlayani.

2. Provisioning

Ketika konsumen meminta sumberdaya melalui layer virtualisasi, sumberdaya tertentu di belakang layer didefinisikan untuk memenuhi permintaan tersebut, dan kemudian dialokasikan ke konsumen. Provisioning sebagai bagian dari grid computing berarti bahwa system menentukan bagaimana cara memenuhi kebutuhan konsumen seiring dengan mengoptimasi jalannya sistem secara keseluruhan.

**Cara Kerja Grid Computing**

Menurut tulisan singkat oleh Ian Foster ada check-list yang dapat digunakan untuk mengidentifikasi bahwa suatu sistem melakukan komputasi grid yaitu :

1. Sistem tersebut melakukan koordinasi terhadap sumberdaya komputasi yang tidak berada dibawah suatu kendali terpusat. Seandainya sumber daya yang digunakan berada dalam satu cakupan domain administratif, maka komputasi tersebut belum dapat dikatakan komputasi grid.

2. Sistem tersebut menggunakan standard dan protokol yang bersifat terbuka (tidak terpaut pada suatu implementasi atau produk tertentu). Komputasi grid disusun dari kesepakatan-kesepakatan terhadap masalah yang fundamental, dibutuhkan untuk mewujudkan komputasi bersama dalam skala besar. Kesepakatan dan standar yang dibutuhkan adalah dalam bidang autentikasi, otorisasi, pencarian sumberdaya, dan akses terhadap sumber daya.

3. Sistem tersebut berusaha untuk mencapai kualitas layanan yang canggih, (nontrivial quality of service) yang jauh diatas kualitas layanan komponen individu dari komputasi grid tersebut.

Keuntungan Grid Computing

Secara generik, keuntungan dasar dari penerapan komputasi Grid, yaitu:

1. Perkalian dari sumber daya: Resource pool dari CPU dan storage tersedia ketika idle

2. Lebih cepat dan lebih besar: Komputasi simulasi dan penyelesaian masalah apat berjalan lebih cepat dan mencakup domain yang lebih luas

3. Software dan aplikasi: Pool dari aplikasi dan pustaka standard, Akses terhadap model dan perangkat berbeda, Metodologi penelitian yang lebih baik

4. Data: Akses terhadap sumber data global, dan Hasil penelitian lebih baik

Indonesia sudah menggunakan sistem Grid dan diberi nama InGrid (Inherent Grid). Sistem komputasi grid mulai beroperasi pada bulam Maret 2007 dan terus dikembangkan sampai saat ini. InGrid ini menghubungkan beberapa perguruan tinggi negeri dan swasta yang tersebar di seluruh Indonesia dan beberapa instansi pemerintahan seperti Badan Meteorologi dan Geofisika.

**Implementasi Grid Computing**

Indonesia sudah menggunakan sistem Grid dan diberi nama InGrid (Inherent Grid). Sistem komputasi grid mulai beroperasi pada bulan Maret 2007 dan terus dikembangkan sampai saat ini. InGrid ini menghubungkan beberapa perguruan tinggi negeri dan swasta yang tersebar di seluruh Indonesia dan beberapa instansi pemerintahan seperti Badan Meteorologi dan Geofisika.

**Komputasi Massasage Passing**

Cara Pemrograman MultiKomputer Message-Passing,

* DESIGN – Merancang Bahasa Pemrograman Paralel secara Khusus.
* DEVELOP – Mengembangkan SYNTAX atau Served Word pada Bahasa Pemrograman Sekuensial Tingkat Tingg yang telah ada untuk meng-Compile Message-Passing.
* USING – Menggunakan Bahasa Pemrograman Sekuensial Tingkat Tinggi yang telah ada dan Menyediakan sebuah Library mengenai prosedur Eksternal untuk MP.

Occam, adalah satu-satunya contoh Bahasa Pemrograman Paralel MP yang umum. Bahasa ini didesain untuk digunakan dengan Prosesor MP, yang disebut Transputer.

HPF (High Performance Fortran), adalah contoh Perluasan Bahasa dalam Pemrograman Paralel yang digunakan untuk Sistem Shared-Memory (Computer Parallel MultiProcessor).

Pada Pembahasan selanjutnya, kita akan focus menggunakan bahasa Pemrograman Tingkat Tinggi, Bahasa C, ditambah dengan pemanggilan Library MP (melakukan MP proses demi proses secara Langsung).

**Pembuatan Proses**

Istilah Proses akan digunakan pada Pemrograman Paralel ini. Proses ini akan menjelaskan bagaimana sebuah Program akan dieksekusi.

Lebih dari satu Proses dapat dipetakan ke suatu Prosesor tunggal. Akibatnya, hal itu tidak akan menghasilkan Eksekusi Berkecapatan Tinggi karena Prosesor harus membagi waktu antara proses-proses tersebut. Alasan lain menggunakan teknik ini ialah untuk menghilangkan beberapa Latency (Waktu yang diperlukan dalam pengiriman Pesan antara Prossesor) Jaringan.

Dua Metode Pembuatan Proses,

1. **Pembuatan Proses Statis (Static Process Creation)**

Semua Proses telah ditentukan / diidentifikasi sebelumnya, lalu system akan mengeksekusi beberapa Prosesor dengan jumlah yang TETAP menggunakan Perintah COMMAND-LINE..

1. **Pembuatan Proses Dinamis (Dynamic Process Creation)**

Proses bisa dibuat sementara system mengeksekusi proses yang lain. Library / System Calls digunakan untuk Membuat Process. Proses dapat dibuat dan dimusnahkan (Create and Destroy….). Jumlah Proses akan berubah-ubah saat eksekusi..

**Master Process dan Slave Process**

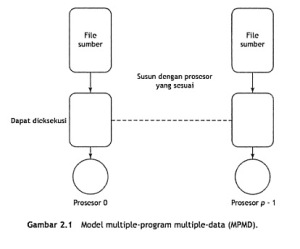
Master Process sebagai satu Proses Pengontrol dan Slave Process sebagai Proses (lainnya) yang Dikontrol. Istilah Slave-Master serupa dalam BENTUK, tapi BERBEDA pada Identifikasi Proses (ID).

**ID Proses**

* Untuk Memodifikasi aksi Proses, atau Menghitung tujuan Message yang Berbeda.
* Model Pemrograman (yang digunakan pada Komputasi MP)

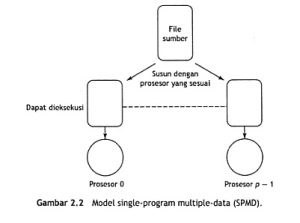
**Model MPMD (Multiple-Program Multiple Data)**

* Program yang BERBEDA ditulis untuk Prosesor yang BERBEDA.
* Minimal dua Program (cukup), sebuah Program Master dan Program Slave.
* Jika Program > 2, maka Program Slave akan SAMA dalam hal BENTUK, tetapi memiliki ID PROSES yang BERBEDA untuk menyesuaikan EKSEKUSInya.
* Contoh penggunaan ID Proses ialah digunakan untuk Menentukan TUJUAN dari Message-Message (Pesan) yang telah di GENERATE



**Model SPMD (Single-Program Multiple Data)**

* Khusus untuk Pembuatan Proses Statis.
* Program yang BERBEDA disatukan ke dalam SATU PROGRAM (Tunggal).
* Di dalam Program (Tunggal) tsb, ada Statement Kontrol yang akan MEMILIH Bagian-bagian yang BERBEDA dari TIAP PROSESOR.
* Setelah Program Source Dibentuk (Dilengkapi Statement Kontrol), kemudian Program tsb akan DISUSUN dalam KODE yang bisa dijalankan sesuai dengan tipe prosesornya.
* Model ini merupakan Pendekatan yang paling umum dari SISTEM MP, MPI (Message-Passing Interface).



Pembuatan Proses Dinamik

* Dua Program Berbeda dapat Ditulis, Program Master dan Slave. Program-program tsb disusun secara terpisah dan Siap untuk dieksekusi.
* Salah satu contoh Pemanggilan Library untuk Pembuatan Proses Dinamis, yaitu

Spawn (nama\_proses-prosesnya)

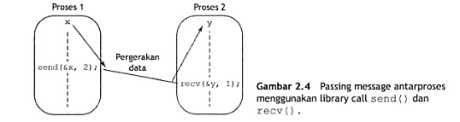
* Statement diatas akan mengawali proses2 lainnya.
* Proses yang Memanggil dan Dipanggil BERPROSES secara Bersamaan.
* Proses yang mengalami “SPAWN” hanyalah Ekseskusi Program yang telah disusun sebelumnya.

Rutin-Rutin Message-Passing

Rutin-rutin MP bekerja sebagai penentu operasi yang dilakukan antara satu proses dengan proses lainnya. Rutin-rutin ini dapat diperoleh dengan pemanggilan Library MP.

Rutin Dasar Send dan Receive

Rutin Send dan Receive ialah rutin yang sering digunakan pada Sistem Komputasi Message-Passing ini. Seperti yang telah dijelaskan sebelumnya, rutin-rutin yang berada di dalam Library MP ini menggunakan Bahasa Pemrograman (Sekuensial) Tingkat Tinggi, yaitu Bahasa C.Send n Receive



Rutin Send ( )

Karena menggunakan Implemtasi Bahasa C, maka nama (istilah) rutin / operasi merupakan istilah yang sering kita temukan dala m kehidupan sehari-hari.

Bentuk Pemanggilan Library untuk Rutin Send ( ), ialah Send (parameter\_list)

Rutin Send ( ) terletak dalam Proses Sumber. Rutin ini akan mengirimkan Message dari Proses Sumber ke proses Tujuan. Parameter yang digunakan dapat terdiri dari parameter yang sederhana hingga parameter-parameter yang kompleks.

Berikut parameter-parameter sederhana tersebut,

**ID tujuan dan Message di Rutin Send ( )**

send (&x , destination\_id);

**Rutin Receive ( )**

Bentuk Pemanggilan Library untuk rutin Receive ( ), ialah

Receive (parameter\_list);

Rutin Receive ( ) terletak dalam Proses Tujuan. Rutin ini akan Menerima Message dari Proses Sumber.

Berikut Parameter-parameter sederhana tersebur,

**ID Sumber dan Tempat menyimpan Message di Rutin Receive ( ).**

receive (&y, source\_id)

Beberapa Ketentuan untuk Rutin-rutin Send dan Receive,

1. Variable x dan y terlebih dahulu DIDEKLARASIKAN.
2. Tipe data x dan y harus SAMA.
3. Untuk data yang lebih kompleks (memiliki lebih dari satu elemen data), kita dapat menggunakan Formasi Message tertentu (akan dibahas pada bagian Penggunaan Komputer Cluster).

**Synchronous Message-PassingSynchronous**, atau dalam bahasa indonesianya berarti menyelaraskan, adalah istilah yang digunakan pada operasi send dan receive di library message passing. Istilah ini berarti bahwa beberapa rutin akan kembali ke tempatnya berasal setelah seluruh operasi pengiriman. Message telah selesai. Istilah kembali disini berarti bahwa Send () akan kembali ke Proses 1 dan Receive akan kembali ke Proses 2, di mana tempat tersebut merupakan tempat dipanggilnya rutin-rutin tersebut.Pada dasarnya, Synchronous akan melakukan dua pekerjaan (antara Send dan Receive), yaitu Mengirim data dan Menyelaraskan Data.

Rutin Send Synchronous : Rutin ini akan kembali ketika pengiriman Message telah diterima oleh Rutin Receive di proses 2.

Rutin Receive Synchronous : Rutin ini akan kembali ketika Message telah tiba di Receive dan kemudian disimpan di rutin Receive (variable y ).

Rendezvous, istilah yang digunakan untuk menjelaskan PERTEMUAN dan PENYELARASAN dua proses di dalam operasi Send/Receive.

Protokol Three-Way, ialah Protokol yang memiliki 3 jalur antara Send dan Receive.

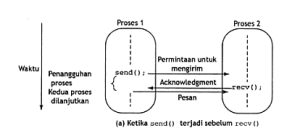
Jalur Pertama ialah ketika Proses sumber (proses 1) mengirimkan Message “Request to send” – Permintaan untuk mengirim pesan – ke Proses Tujuan (Proses 2).

Jalur kedua ialah ketika Proses Tujuan mengirim Sinyal Acknowledgement – Persetujuan request dari Jalur Pertama – ke Proses Sumber.

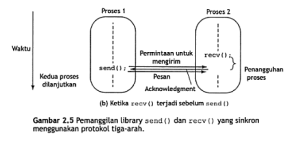
Lalu, Jalur Ketiga ialah ketika Proses Sumber Mengirimkan Message yang Sebenarnya.

Ada beberapa keadaan yang terjadi pada Protokol Three-Away di rutin Send-Receive,

1. Ketika proses 1 mencapai Send ( ) sebelum Proses 2 mencapai Receive ( ), Proses 1 beberapa saat akan ditunda hingga Proses 2 mencapai Receive. Message untuk sementara akan disimpan dalam Proses 1. Saat Receive telah tiba, maka Proses 2 akan membangkitkan sebuah “sinyal” ke proses 1. Akhirnya, Kedua Proses tersebut akan melakukan Transfer Message.



1. Ketika Proses 2 mencapai Receive ( ) sebelum Proses 1 mencapai Send ( ), Pada keadaan ini, proses 2 harus Ditangguhkan hingga Proses 1 mencapai rutin Send. Setelah Send tiba, kedua proses akan melakukan Transfer Message.



Message-Passing Blocking dan UnblockingIstilah Blocking dan Unblocking merupakan istilah yang digunakan untuk menjelaskan keadaan rutin-rutin, dalam hal ini rutin yang kita bahas ialah Rutin Send ( ) dan Rutin Receive ( ). (Locally) Blocking

Istilah ini digunakan untuk menjelaskan suatu Rutin yang KEMBALI Setelah Aksi Lokalnya selesai, meskipun Pengiriman Message Belum selesai / Belum Tiba di Proses Tujuan.

**Unblocking**

Istilah ini digunakan untuk menjelaskan suatu Rutin yang SEGERA KEMBALI, TIDAK PEDULI apa Messagenya DITERIMA atau TIDAK oleh proses Tujuan.

**Message Buffer**

Digunakan untuk PENYIMPANAN SEMENTARA Message yang Telah Dikirimkan oleh Rutin Send ( ) (Di Proses Sumber), tapi Belum diterima oleh Rutin Receive () ( di Proses Tujuan).

**Message Buffer**

Ada beberapa hal yang HARUS kita ingat mengenai bahasan Send dan Receive ini,

* Jika kita menginginkan Message dari Proses Sumber, maka Message tsb HARUS sampai pada Rutin Receive di Proses Tujuan.
* Jika Rutin Receive dicapai sebelum Rutin Send, maka Message Buffer akan KOSONG. Jika Rutin Send telah ber-AKSI di Proses Sumber, maka Message akan Langsung terkirim ke Rutin Receive dan Message Buffer hanya menjadi tempat lewat (tanpa disimpan) message-message.
* Pada Pembahasan selanjutnya, (Locally) Blocking dan Synchronous akan lebih banyak dibahas.

**Pemilihan Message**

Seperti pada pembahasan sebelumnya, Rutin Send dan Receive memiliki beberapa parameter dalam melakukan transfer. ID Tujuan diberikan sebagai parameter dari Rutin Send, dan ID Sumber diberikan sebagai parameter dari Rutin Receive. Analoginya, ID Sumber bertindak seperti sebuah ALAMAT dalamPENGIRIMAN SURAT POS. Jadi, Proses Sumber hanya akan mengirim Message pada Alamat Tertentu, dan juga Proses Tujuan hanya akan Menerima Message berdasarkan parameter ID Sumber yang Tertulis dalam Parameternya.

**Wild-Card**

Jadi, Bagaimana kita bisa Menerima Pesan dari Proses Sumber yang Berbeda?

Jawabannya ialah pada Istilah WILD CARD pada ID Sumber. WILD CARD pada ID Sumber = -1. Jadi, jika kita menginginkan Sebuah Rutin Receive ( ) menerima Semua Message dari Proses Sumber yang Berbeda, maka kita memberikan nilai Parameter di ID Sumber dengan nilai -1.

Contoh,

Send(&x, -1);

X (berbentuk pointer dari suatu tipe data yang tela h kita deklarasikan sebelumnya.

-1 = Wild Card.

Send( ) = Rutin untuk mengirmkan Message ke Proses Tujuan.

**Tag Message**

Agar kita bisa memilih Message yang Dikirim, maka kita dapat memasukkan Tag Message / Message yang dipilih ke Parameter milik Rutin Receive.

Contoh, penggunaan Tag Message,

Untuk mengirim message x, dengan tag 5 (urutan pesan yang dipilih) dari proses sumber =1, ke proses tujuan = 2, dan diserahkan ke y,

Send(&x, 2, 5); >>> Pada Proses Sumber

Recv( ) >>> Pada Proses Tujuan

Hal yang HARUS diketahui tentang pembahsan ini,

* Tag Message dimuat dalam Message.
* Tag Message dapat digantikan dengan Wild Card sehingga recv( ) dapat sesuai dengan segala Rutin send ( ) yang berasal dari Proses Sumber.
* Programmer harus menyimpan track dari nomor tag message yang digunakan oleh message yang digunakan dalam program, dan program lainnya yang dsertakan.

**Broadcast, Gather, dan Scatter**

**Broadcast**

Broadcast (dalam istilah Parallel Programming) memiliki arti,

Pengiriman Message yang SAMA ke beberapa tujuan atau Grup tertentu.

Pengiriman Message yang SAMA ke beberapa Proses yang Berhubungan dengan suatu Masalah.

Bcast

Tahap-tahap Broadcast,

Identifikasi – Proses-proses yang terlibat dalam Broadcast terlebih dahulu diidentifikasi.

Group – Dibuat suatu Group Proses dengan nama tertentu.

Parameter – Memasukkan nama Group Proses tersebut ke Parameter Broadcast.

Misalnya, Proses 0 diidentifikasi sebagai Proses Root dalam Parameter Broadcast. Proses root adalah Proses APA SAJA di dalam Grup proses.

Berdasarkan gambar (contoh), Proses Root menyimpan data dalam Parameter buf untuk kemudian di Broadcast.

Aksi Broadcast Tidak akan terjadi hingga semua proses mengeksekusi rutin bcast( ). Hal itu berarti akan ada SINKRONISASI pada rutin Broadcast.

Hal penting tentang Rutin Broadcast,

Implementasi dari rutin Broadcast yang Efisien sangat penting dalam menyebarkan penggunaan Broadcast pada Program secara meluas.

**Gather**

Gather, adalah kebalikan dari Scatter. Rutin ini akan mengumpulkan nilai-nilai individu dari grup proses ke proses root (berdasarkan contoh dibawah, Proses 0). Gather biasanya dieksekusi setelah proses telah menyelesaikan beberapa komputasi.

Kenapa Gather merupakan kebalikan dari Scatter?

Karena data dari proses ke-i akan dikirim ke Proses Root (Proses 0) di lokasi array ke-i. Berdasarkan contoh di bawah, proses root juga mengumpulkan datanya ke array.

Scatter

Scatter, istilah yang menjelaskan tentang pengiriman suatu elemen array dari suatu data (pada root) ke proses-proses yang terpisah. Isi dari lokasi ke-I dari array, dikirim ke proses ke-i.

Tahap-tahap eksekusi Scatter,

* Identifikasi grup proses termasuk proses rootnya.
* Berdasarkan gambar dibawah, Proses root (proses 0) juga menerima data.
* Setiap proses (di dalam grup) mengeksekusi rutin Scatter( ) yang sama.

Hal yang perlu diperhatikan dalam Rutin Scatter ( ),

Pada model SPMD, proses eksekusinya akan lebih mudah.

Operasi Reduce

Operasi reduce merupakan operasi Gather yang dikombinasikan dengan Operasi Aritmatika / Logika. Caranya ialah, mengumpulkan nilai dari proses-proses lalu dijumlahkan oleh root. Operasi ini tidak mesti terjadi di Proses Root, melainkan dapat juga didistribusikan ke proses-proses lainnya. Hal ini bertujuan untuk menerapkan sekumpulan operasi yang umum seefektif mungkin.

Tool-Tool Perangkat Lunak

Berdasarkan apa yang telah kita bahas sebelumnya, Perangkat lunak (Software) bertugas untuk menerapkan Metode Message-Passing ke Komputasi Cluster yang biasanya tergabung dalam jaringan Workstation. Perangkat lunak yang banyak dipakai ialah, PVM (Parallel Virtual Machine).

PVM

PVM dibuat di Laboratorium Oak Ridge pada tahun 1980-an

Mulai dipakai secara meluas pada tahun 1990-an

PVM lebih banyak dipakai dibanding rutin library-library lainnya (mis. Buatan IBM dan vendor lainnya) karena PVM Gratis. (dapat diakses di, http://www.netlib.org/pvm3/)

Rutin library dari PVM dapat digunakan oleh user dengan menggunakan program C atau Fortran

Saat ini, PVM tersedia juga untuk Platform Windows.

PVM menggunakan pembuatan proses dinamis. Prinsipnya : ketika suatu proses sedang dibuat, maka dalam waktu bersamaan akan ada proses lain yang dieksekusi.

MPI (Message-Passing Interface)

Alasan dalam pembuatan MPI ialah keinginan untuk membuat MP yang Portabel dan Kemudahan dalam penggunaannya. MPI merupakan Standar Sistem Message Passing yang merupakan hasil kerja sama dari para akademisi dan Parther Industri. MPI menyediakan Rutin Library untuk MP dan Operasi-operasi yang berkaitan. Perlu diingat bahwa,

MPI = Standar ≠ Implementasi.

MPI melakukan perubahan guna memperbaiki kekurangan teknik pada system MP sebelumnya, yaitu PVM.

Versi MPI

Versi 1.0; diadakan pertemuan dan diskusi untuk membuatnya pada tahun 1992. Selesai dibuat pada Mei 1994.

Versi 1.2; selesai pada tahun1997. MPI 1.2 telah mendukung pembuatan Proses Dinamis, operasi satu-sisi, dan I/O Paralel.

**Gambaran scheduling parallel**

C:\Users\isaiye\Pictures\kom.png

Menggunakan 3 core processor membuat kinerja komputasi paralel lebih efektif dan dapat menghemat waktu untuk pemrosesan data yang banyak daripada komputasi tunggal.

Dari penjelasan-penjelasan di atas, kita bisa mendapatkan jawaban mengapa dan kapan kita perlu menggunakan komputasi paralel. Jawabannya adalah karena komputasi paralel jauh lebih menghemat waktu dan sangat efektif ketika kita harus mengolah data dalam jumlah yang besar. Namun keefektifan akan hilang ketika kita hanya mengolah data dalam jumlah yang kecil, karena data dengan jumlah kecil atau sedikit lebih efektif jika kita menggunakan komputasi tunggal.

**Java Parallel Programming Framework (JPPF)**

Fitur :

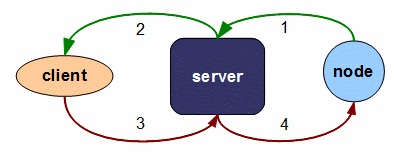
1. Mudah digunakan
2. Topologi yang fleksibel dan dinamis
3. Self-repair dan recovery
4. Pencocokan Beban Kerja dengan Grid
5. Tidak ada penyebaran
6. keamanan
7. Administrasi dan pemantauan
8. Load balancing
9. Integrasi Net dan J2EE
10. Kustomisasi

Cara Kerja :

Kerangka class loading didistribusikan di JPPF adalah mekanisme yang memungkinkan untuk mengeksekusi kode dalam simpul yang belum secara eksplisit dikerahkan untuk lingkungan node. Melalui ini, tugas JPPF yang kode (bytecode yang sebenarnya untuk mengeksekusi) hanya didefinisikan dalam aplikasi client JPPF, dapat dijalankan pada node terpencil tanpa pengembang aplikasi harus khawatir tentang bagaimana kode ini akan diangkut sana.

Sementara mekanisme ini sepenuhnya transparan dari perspektif aplikasi klien, ia memiliki sejumlah implikasi dan kekhasan yang dapat mempengaruhi berbagai aspek JPPF tugas eksekusi, termasuk kinerja dan integrasi dengan perpustakaan eksternal.

Mari kita memiliki pandangan cepat dari jalur diikuti oleh permintaan class loading pada saat tugas JPPF dijalankan dalam sebuah node:



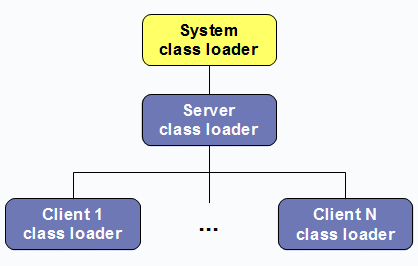
Kita bisa melihat bahwa ini permintaan class loading dilaksanakan dalam empat langkah:

* node mengirimkan permintaan jaringan ke server jauh untuk kelas
* server meneruskan permintaan ke klien remote diidentifikasi
* klien memberikan respon (bytecode kelas) ke server
* server meneruskan respon terhadap node

Setelah langkah ini dilakukan, node memegang bytecode kelas dan efektif dapat menentukan dan memuatnya sebagai untuk setiap kelas Java standar.

**Kelas loader hirarki di JPPF node**

Mekanisme JPPF classloader mengikuti hirarki didasarkan pada hubungan orangtua-anak antara contoh kelas loader, seperti digambarkan dalam gambar berikut:



Kelas sistem loader digunakan untuk memulai node JPPF. Dengan sebagian besar JVMs, itu akan menjadi contoh dari java.net.URLClassloader kelas dan penggunaan dan penciptaan ditangani oleh JVM.

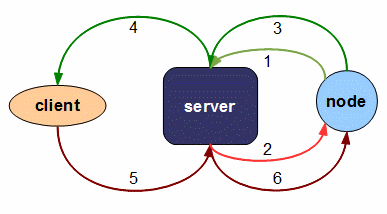
Kelas server loader adalah implementasi konkret dari AbstractJPPFClassloader kelas dan menyediakan akses remote ke kelas dan sumber daya dalam classpath server. Hal ini dibuat pada saat node membangun koneksi ke server. Hal ini juga dibuang ketika node terputus dari server. Induk dari loader kelas server kelas loader sistem. Harap dicatat bahwa AbstractJPPFClassLoader juga merupakan subclass dari URLClassLoader.

Loader kelas klien juga implementasi konkret dari AbstractJPPFClassloader dan menyediakan akses remote ke kelas dan sumber daya dalam classpath satu atau lebih klien. Setiap kelas loader klien dibuat pertama kali node menjalankan pekerjaan yang disampaikan oleh klien. Dengan demikian, node dapat memegang banyak loader kelas klien.

Penting untuk dicatat bahwa, dengan desain, node JPPF memegang koneksi jaringan tunggal untuk server, bersama dengan semua contoh AbstractJPPFClassLoader, termasuk server dan klien loader kelas. Desain ini menghindari banyak potensi kebingungan, inkonsistensi dan perangkap sinkronisasi saat melakukan beberapa permintaan class loading secara paralel.

Secara default, loader kelas JPPF mengikuti kebijakan delegasi standar induknya. Ini berarti bahwa, ketika kelas diminta dari loader kelas klien, pertama kali akan mendelegasikan ke induknya, server kelas loader, yang di akan berubah delegasi pertama class loader sistem. Jika kelas tidak ditemukan oleh orang tua, maka loader kelas akan mencarinya di classpath yang memiliki akses.

Dengan demikian, aliran permintaan, untuk kelas yang hanya jalan kelas klien, menjadi sedikit lebih kompleks:

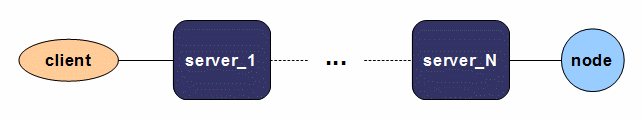


Di sini, dua langkah pertama yang diprakarsai oleh loader kelas server, sebagai akibat dari loader kelas klien mendelegasikan ke induknya. Apa yang hilang dari gambar ini adalah panggilan ke loader kelas sistem, karena mereka hanya bermakna jika kelas yang diminta dalam classpath lokal node.

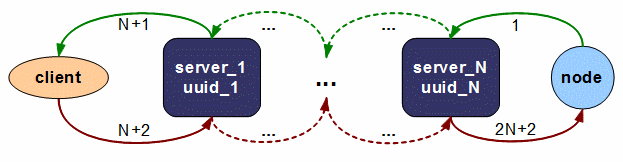
**Hubungan antara JPPF UUIDs dan loader kelas**

Kita telah melihat dalam Panduan Pengembangan bahwa setiap klien JPPF memiliki identifier sendiri, unik di seluruh JPPF jaringan. Hal ini juga berlaku untuk server dan node. Klien UUID adalah apa yang memungkinkan node JPPF untuk mengetahui klien loader kelas dikaitkan dengan, dan menggunakannya untuk rute permintaan class loading dari node ke klien yang disampaikan pekerjaan.

Jika node hanya mengetahui UUID klien, maka hanya akan mampu menangani routing permintaan class loading dalam topologi jaringan JPPF sederhana: topologi yang hanya memiliki satu server. Namun, ada mekanisme yang memungkinkan loading class untuk bekerja di jauh lebih kompleks topologi, seperti yang satu ini:



Untuk efek ini, setiap pekerjaan dilaksanakan pada node akan mengangkut, selain UUID klien yang berasal itu, UUID dari masing-masing server dalam rantai server yang harus dilalui untuk sampai ke node. Dalam terminologi JPPF, ini daftar memerintahkan UUIDs disebut jalur UUID. Dengan informasi ini diketahui, adalah mungkin untuk rute permintaan class loading melalui rantai server, seperti yang digambarkan pada gambar di bawah:



It is also important to note that this does not change anything to the class loader hierarchy within the node. In effect, there is still only one server class loader, which is associated with the server the node is directly connected to. This implies that the parent delegation model will not cause a class loading request to traverse the server chain multiple times.

Another implication of using client UUIDs is that it is possible to have multiple versions of the same code running within a node. Let's imagine a situation where two distinct JPPF clients, with separate UUIDs, submit the same tasks. From the node's point of view, the classes will be loaded by two distinct client class loaders, and therefore the classes from the first client will be different from those of the second client, even if they have the exact same bytecode and are downloaded from the same jar file.

The reverse situation may also happen, when two clients with the same UUID submit tasks that use different versions of the same classes. In this case, the tasks will be exposed to errors, especially at deserialization time, if the two versions are incompatible.

Contoh Hasil :