Pertemuan 1
Pengenalan Sistem Terdistribusi

Pokok Bahasan

- Konsep dasar sistem terdistribusi
- Cara kerja Sistem terdistribusi
- Middleware
- Tujuan Middleware
- Memisahkan kebijakan dari mekanisme

KONSEP SISTEM DISTRIBUSI

Sistem terdistribusi adalah kumpulan komputer independen yang muncul bagi penggunanya sebagai sistem koheren tunggal

Definisi ini memiliki beberapa aspek penting.

1. Sistem terdistribusi terdiri dari komponen (mis., Komputer) yang otonom.
2. Pengguna (karena orang atau program) berpikir mereka berurusan dengan satu sistem. Ini berarti bahwa satu atau lain cara komponen otonom perlu berkolaborasi. Bagaimana membangun kolaborasi ini terletak di jantung pengembangan sistem terdistribusi. Perhatikan bahwa tidak ada asumsi yang dibuat mengenai jenis komputer.

Pada prinsipnya, bahkan dalam satu sistem tunggal, mereka dapat berkisar dari komputer mainframe berkinerja tinggi hingga node kecil di jaringan sensor. Demikian juga, tidak ada asumsi yang dibuat tentang cara komputer saling berhubungan.

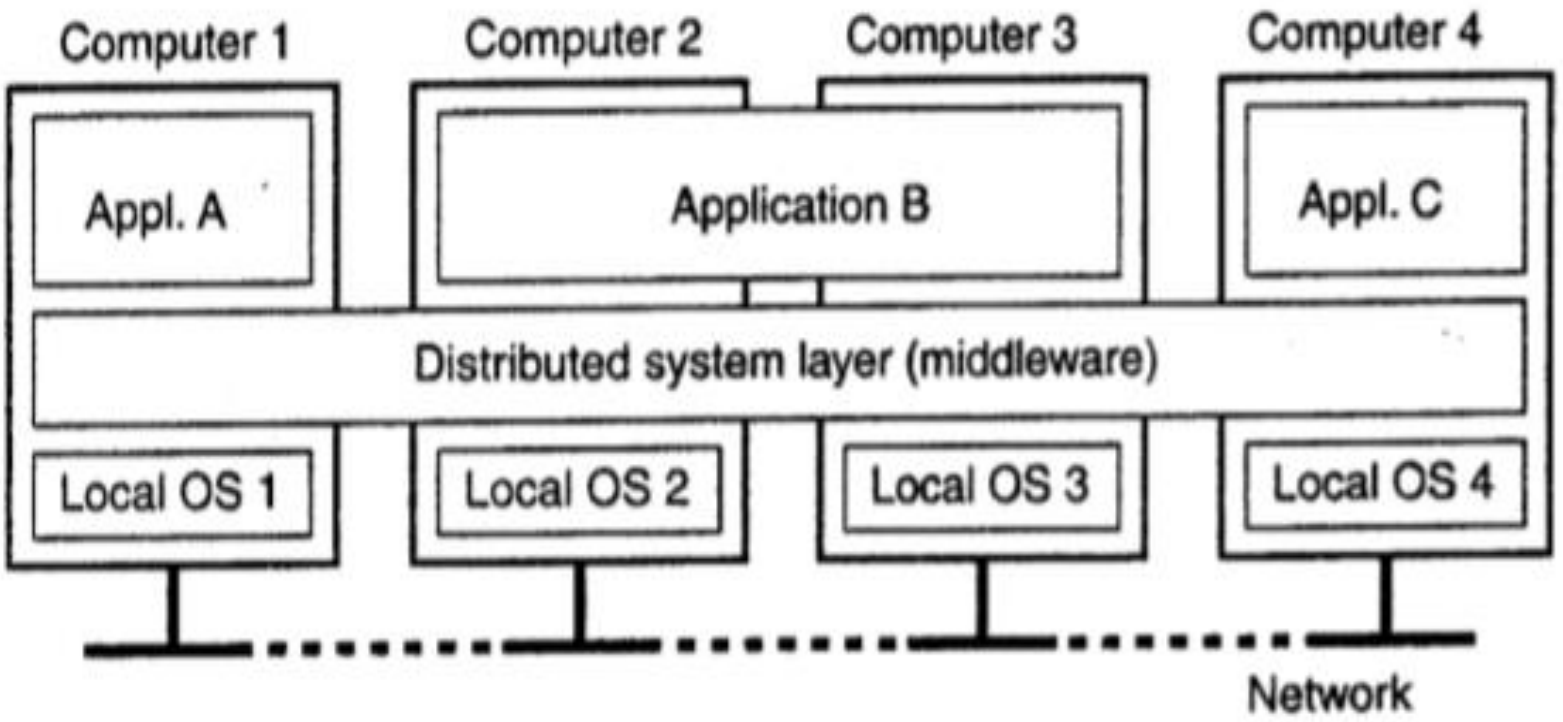
Karakteristik penting lainnya adalah bahwa pengguna dan aplikasi dapat berinteraksi dengan sistem terdistribusi secara konsisten dan seragam, terlepas dari di mana dan kapan interaksi berlangsung.

Pada prinsipnya, sistem terdistribusi juga harus relatif mudah diperluas atau dikembangkan. Karakteristik ini merupakan konsekuensi langsung dari memiliki komputer independen, tetapi pada saat yang sama, menyembunyikan bagaimana komputer ini benar-benar mengambil bagian dalam sistem secara keseluruhan. Sistem terdistribusi biasanya akan terus tersedia, walaupun mungkin beberapa bagian mungkin rusak untuk sementara waktu

Cara kerja Sistem terdistribusi

Untuk mendukung komputer dan jaringan yang heterogen sambil menawarkan tampilan sistem tunggal, sistem terdistribusi sering diatur dengan menggunakan perangkat lunak - yaitu, secara logis ditempatkan di antara lapisan tingkat yang lebih tinggi yang terdiri dari pengguna dan aplikasi, dan lapisan di bawahnya terdiri dari operasi sistem dan fasilitas komunikasi dasar, seperti yang ditunjukkan pada Slide selanjutnya dengan demikian, sistem terdistribusi semacam itu kadang-kadang disebut middleware

Middleware



Gambar 1. Midleware

Gambar pada slide sebelumnya menunjukkan empat komputer jaringan dan tiga aplikasi, di mana aplikasi B didistribusikan di seluruh komputer 2 dan 3. Setiap aplikasi ditawarkan antarmuka yang sama. Sistem terdistribusi menyediakan sarana untuk komponen aplikasi tunggal yang didistribusikan untuk berkomunikasi satu sama lain, tetapi juga untuk memungkinkan aplikasi yang berbeda berkomunikasi. Pada saat yang sama, ia menyembunyikan, sebaik dan seakurat mungkin, perbedaan dalam perangkat keras dan sistem operasi dari setiap aplikasi.

Tujuan

Hanya karena dimungkinkan untuk membangun sistem terdistribusi tidak berarti bahwa itu adalah ide yang baik. Lagi pula, dengan teknologi saat ini juga memungkinkan untuk menempatkan empat floppy disk drive atau fasilitas media penyimpanan lainnya pada komputer pribadi.

Pada bagian ini dibahas empat tujuan penting yang harus dipenuhi untuk membuat membangun sistem terdistribusi yang sepadan dengan usaha. Sistem terdistribusi harus membuat sumber daya mudah diakses; ia seharusnya menyembunyikan fakta bahwa sumber daya didistribusikan di seluruh jaringan; itu harus dibuka, dan itu harus ditingkatkan.

1. Tujuan utama sistem terdistribusi adalah memudahkan pengguna (dan aplikasi) untuk mengakses sumber daya jarak jauh, dan membagikannya dengan cara yang terkontrol dan efisien. Sumber bisa saja tentang apa saja, tetapi contoh-contoh tipikal termasuk hal-hal seperti printer, komputer, fasilitas penyimpanan, data, file, halaman Web, dan jaringan, hanya beberapa. Ada banyak alasan untuk ingin berbagi sumber daya. Salah satu alasan yang jelas adalah alasan ekonomi. Sebagai contoh, lebih murah untuk membiarkan printer dibagikan oleh beberapa pengguna di kantor kecil daripada harus membeli dan memelihara printer yang terpisah untuk setiap pengguna. Demikian juga, masuk akal secara ekonomi untuk berbagi sumber daya yang mahal seperti superkomputer, sistem penyimpanan berkinerja tinggi, imagesetters, dan periferal murah lainnya.

Namun, dengan meningkatnya konektivitas dan berbagi, keamanan menjadi semakin penting. Dalam praktik saat ini, sistem memberikan sedikit perlindungan terhadap penyadapan atau intrusi komunikasi. Kata sandi dan informasi sensitif lainnya sering dikirim sebagai cleartext (mis., Tidak dienkripsi) melalui jaringan, atau disimpan di server yang hanya dapat diharapkan dapat dipercaya. Dalam hal ini, ada banyak ruang untuk perbaikan. Misalnya, saat ini dimungkinkan untuk memesan barang hanya dengan memberikan nomor kartu kredit. Jarang diperlukan bukti bahwa pelanggan memiliki kartu itu. Di masa mendatang, menempatkan pesanan dengan cara ini hanya mungkin dilakukan jika pelanggan benar-benar dapat membuktikan bahwa pelanggan secara fisik mengambil kartu dengan memasukkannya ke pembaca kartu.

2. Tujuan penting dari sistem terdistribusi adalah untuk menyembunyikan fakta bahwa proses dan sumber dayanya didistribusikan secara fisik ke banyak komputer. Sistem terdistribusi yang mampu menghadirkan dirinya kepada pengguna dan aplikasi seolah-olah hanya sistem komputer tunggal yang dikatakan transparan.
   Jenis transparansi apa yang ada dalam sistem terdistribusi ?.
   Apakah transparansi selalu diperlukan ?

Akses transparansi berkaitan dengan menyembunyikan perbedaan dalam representasi data dan akses ke sumber daya yang dapat diakses oleh pengguna. Pada tingkat dasar, perbedaan dalam arsitektur mesin disembunyikan, tetapi yang lebih penting adalah tercapainya kesepakatan tentang bagaimana data akan diwakili oleh mesin dan sistem operasi yang berbeda. Misalnya, sistem terdistribusi mungkin memiliki sistem komputer yang menjalankan sistem operasi yang berbeda, masing-masing memiliki konvensi penamaan file sendiri. Perbedaan dalam konvensi penamaan, serta bagaimana file dapat dimanipulasi, semua harus disembunyikan dari pengguna dan aplikasi.

3. Tujuan penting lain dari sistem terdistribusi adalah keterbukaan. Sistem terdistribusi terbuka adalah sistem yang menawarkan layanan sesuai dengan aturan standar yang menggambarkan sintaks dan semantik layanan tersebut. Misalnya, dalam jaringan komputer, aturan standar mengatur format, konten, dan makna pesan yang dikirim dan diterima. Aturan tersebut diformalkan dalam protokol. Dalam sistem terdistribusi, layanan umumnya ditentukan melalui antarmuka, yang sering dijelaskan dalam Bahasa Definisi Antarmuka (IDL). Definisi antarmuka yang ditulis dalam IDL hampir selalu hanya menangkap sintaks layanan.

Dengan kata lain, ditentukan secara tepat nama-nama fungsi yang tersedia bersama dengan jenis parameter, nilai pengembalian, kemungkinan pengecualian yang dapat dinaikkan, dan sebagainya. Bagian yang sulit adalah menentukan dengan tepat apa yang dilakukan oleh layanan tersebut, yaitu semantik antarmuka. Dalam praktiknya, spesifikasi seperti itu selalu diberikan secara informal dengan menggunakan bahasa alami

Memisahkan Kebijakan dari Mekanisme

Untuk mencapai fleksibilitas dalam sistem terdistribusi terbuka, sangat penting bahwa sistem disusun sebagai kumpulan komponen yang relatif kecil dan mudah diganti atau beradaptasi. Ini menyiratkan bahwa definisi harus diberikan tidak hanya untuk antarmuka tingkat tertinggi, yaitu yang dilihat oleh pengguna dan aplikasi, tetapi juga definisi untuk antarmuka ke bagian internal di sistem dan menggambarkan bagaimana bagian-bagian tersebut berinteraksi. Pendekatan ini relatif baru. Banyak sistem yang lebih tua dan bahkan kontemporer dibangun menggunakan pendekatan monolitik di mana komponen berada

4. Skalabilitas, Konektivitas di seluruh dunia melalui Internet dengan cepat menjadi hal biasa seperti mengirim kartu pos kepada siapa pun di seluruh dunia. Dengan pemikiran ini, skalabilitas adalah salah satu tujuan desain terpenting bagi pengembang sistem terdistribusi. Skalabilitas suatu sistem dapat diukur sepanjang setidaknya tiga dimensi yang berbeda (Neuman, 1994). Pertama, suatu sistem dapat scalable sehubungan dengan ukurannya, artinya dapat dengan mudah menambahkan lebih banyak pengguna dan sumber daya ke sistem. Kedua, sistem yang dapat diukur secara geografis adalah sistem di mana pengguna dan sumber daya mungkin terpisah jauh. Ketiga, suatu sistem dapat secara administratif dapat diukur, dan bahwa itu masih mudah untuk dikelola bahkan jika itu mencakup banyak organisasi administrasi independen.

Ketika suatu sistem perlu diukur, berbagai jenis masalah yang berbeda perlu dipecahkan. Skala dipertimbangkan terlebih dahulu dengan memperhatikan ukuran. Jika lebih banyak pengguna atau sumber daya perlu didukung, maka akan sering dihadapkan dengan keterbatasan terpusat. layanan, data, dan algoritma (lihat Gambar). Sebagai contoh, banyak layanan terpusat dalam arti bahwa mereka diimplementasikan dengan hanya menggunakan satu server yang berjalan pada mesin tertentu dalam sistem terdistribusi. Masalah dengan skema ini jelas: server dapat menjadi hambatan ketika jumlah pengguna dan aplikasi bertambah. Bahkan jika kapasitas pemrosesan dan penyimpanan yang dimiliki yang hampir tidak terbatas, komunikasi dengan server pada akhirnya akan melarang pertumbuhan lebih lanjut.

Concept	Example
Centralized services	A single server for all users
Centralized data	A single on-line telephone book
Centralized algorithms	Doing routing based on complete information

Gambar-2. Pembatasan Scalabilitas

Sayangnya. hanya menggunakan satu server terkadang tidak dapat dihindari. Bayangkan layanan yang dimiliki untuk mengelola informasi yang sangat rahasia seperti catatan medis, rekening bank. dan seterusnya. Dalam kasus seperti itu, mungkin lebih baik untuk mengimplementasikan layanan itu dengan menggunakan server tunggal di ruang terpisah yang sangat aman, dan dilindungi dari bagian lain dari sistem terdistribusi melalui komponen jaringan khusus.

Seburuk layanan terpusat adalah data terpusat. Bagaimana melacak nomor telepon dan alamat 50 juta orang? Misalkan setiap rekaman data dapat masuk ke dalam 50 karakter. Partisi disk 2,5 gigabyte tunggal akan menyediakan penyimpanan yang cukup. Tetapi di sini lagi, memiliki satu basis data tidak diragukan lagi akan memenuhi semua jalur komunikasi masuk dan keluar darinya. Demikian juga, bayangkan bagaimana Internet akan berfungsi jika Domain Name System (DNS)-nya masih diterapkan sebagai singletable. DNS menyimpan informasi tentang jutaan komputer di seluruh dunia dan membentuk layanan penting untuk mencari server Web. Jika setiap permintaan untuk menyelesaikan URL harus diteruskan ke satu-satunya server DNS, sudah pasti tidak ada yang akan menggunakan Web.

Akhirnya, algoritma terpusat juga merupakan ide yang buruk. Dalam sistem terdistribusi besar, sejumlah besar pesan harus dialihkan melalui banyak jalur. Dari sudut pandang teoritis, cara optimal untuk melakukan ini adalah mengumpulkan informasi lengkap tentang beban pada semua mesin dan jalur, dan kemudian jalankan algoritma untuk menghitung semua rute optimal. Informasi ini kemudian dapat disebarkan ke seluruh sistem untuk meningkatkan perute-an.

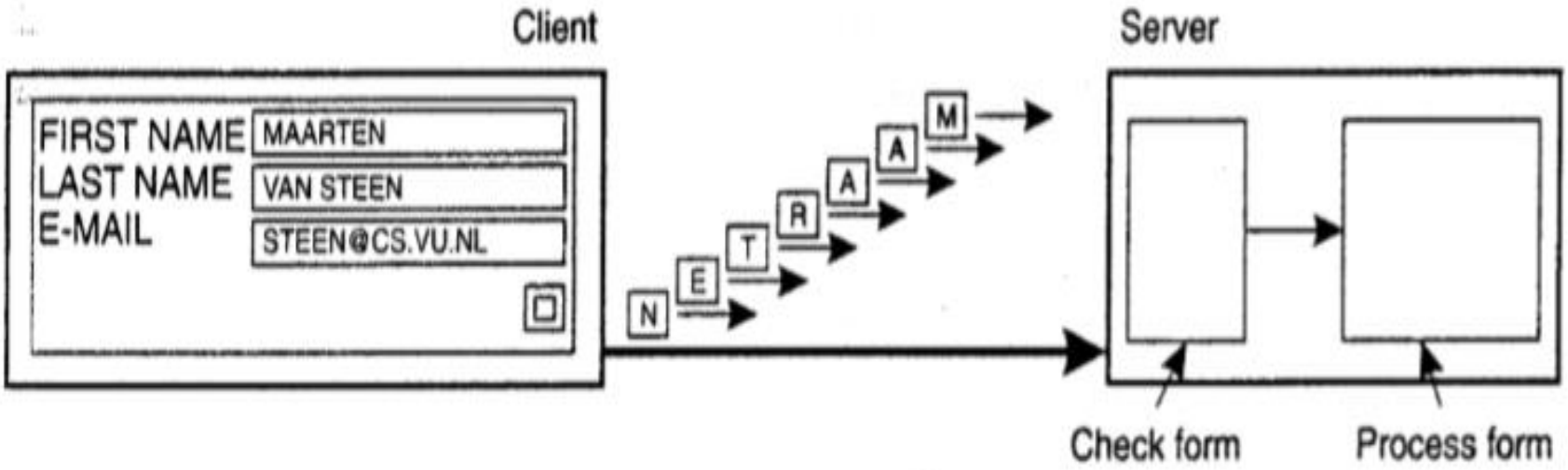
Masalahnya adalah bahwa mengumpulkan dan mengangkut semua informasi input dan output akan kembali menjadi ide yang buruk karena pesan-pesan ini akan membebani sebagian jaringan. Faktanya, setiap algoritma yang beroperasi dengan mengumpulkan informasi dari semua situs, mengirimkannya ke satu mesin untuk diproses, dan kemudian mendistribusikan hasil umumnya harus dihindari. Hanya algoritma desentralisasi yang harus digunakan. Algoritma ini umumnya memiliki karakteristik berikut, yang membedakannya dari algoritma terpusat.

1. Tidak ada mesin yang memiliki informasi lengkap tentang status sistem.
2. Mesin membuat keputusan hanya berdasarkan informasi lokal,
3. Kegagalan mesin tidak merusak algoritma.
4. Tidak ada asumsi sederhana bahwa global clock ada.

Komunikasi asinkron sering dapat digunakan dalam sistem pemrosesan batch dan aplikasi paralel, di mana tugas yang lebih atau kurang independen dapat dijadwalkan untuk dieksekusi sementara tugas lain sedang menunggu untuk menyelesaikan komunikasi. Atau, utas kontrol baru dapat dimulai untuk melakukan permintaan. Meskipun blokir menunggu balasan, utas lainnya dalam proses dapat dilanjutkan.

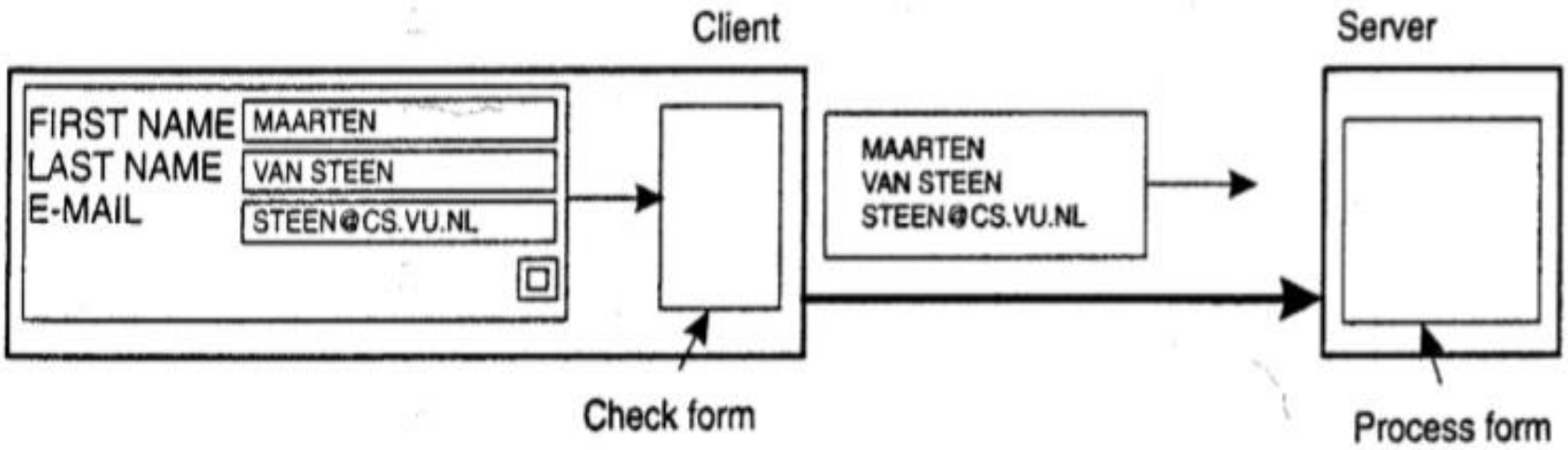
Namun, ada banyak aplikasi yang tidak dapat menggunakan komunikasi asinkron secara efektif. Misalnya, dalam aplikasi interaktif ketika pengguna mengirim permintaan, ia biasanya tidak akan melakukan apa pun selain menunggu jawabannya. Dalam kasus seperti itu, solusi yang jauh lebih baik adalah mengurangi komunikasi secara keseluruhan, misalnya, dengan memindahkan bagian dari perhitungan yang biasanya dilakukan di server ke proses klien yang meminta layanan. Kasus khas di mana pendekatan ini bekerja adalah mengakses database menggunakan formulir. Mengisi formulir dapat dilakukan dengan mengirim pesan terpisah untuk setiap bidang, dan menunggu pengakuan dari server.

Kasus khas di mana pendekatan ini bekerja adalah mengakses database menggunakan formulir. Mengisi formulir dapat dilakukan dengan mengirim pesan terpisah untuk setiap bidang, dan menunggu pengakuan dari server, seperti yang ditunjukkan pada Gambar-3



Gambar-3. Pengisian formulir.

server dapat memeriksa kesalahan sintaksis sebelum menerima entri. Solusi yang jauh lebih baik adalah mengirimkan kode untuk mengisi formulir, dan mungkin memeriksa entri, ke klien, dan meminta klien mengembalikan formulir yang sudah diisi, seperti yang ditunjukkan dalam Gambar-4.



Gambar 4. Permeriksaan sintaks

5. Perangkap.

Harus jelas dengan mengetahui bahwa mengembangkan sistem terdistribusi dapat menjadi tugas yang berat. Ada begitu banyak masalah untuk dipertimbangkan pada saat yang sama sehingga tampaknya hanya kompleksitas yang bisa menjadi hasilnya. Namun demikian, dengan mengikuti sejumlah prinsip desain, sistem terdistribusi dapat dikembangkan yang sangat sesuai dengan tujuan. Namun, sistem terdistribusi berbeda dari perangkat lunak tradisional karena komponen tersebar di jaringan. Tidak memperhitungkan dispersi ini selama waktu desain adalah apa yang membuat begitu banyak sistem menjadi rumit dan menghasilkan kesalahan yang perlu ditambal nanti.

Tidak memperhitungkan dispersi ini selama waktu desain adalah apa yang membuat begitu banyak sistem menjadi rumit dan menghasilkan kesalahan yang perlu ditambal nanti. Peter Deutsch, saat itu di Sun Microsystems, merumuskan kesalahan-kesalahan ini sebagai asumsi salah berikut yang dibuat setiap orang ketika mengembangkan aplikasi terdistribusi untuk pertama kalinya

1. Jaringan dapat diandalkan.
2. Jaringan aman.
3. Jaringannya homogen.
4. Topologi tidak berubah.
5. Latency adalah nol.
6. Bandwidth tidak terbatas.
7. Biaya transportasi nol.
8. Ada satu administrator.

