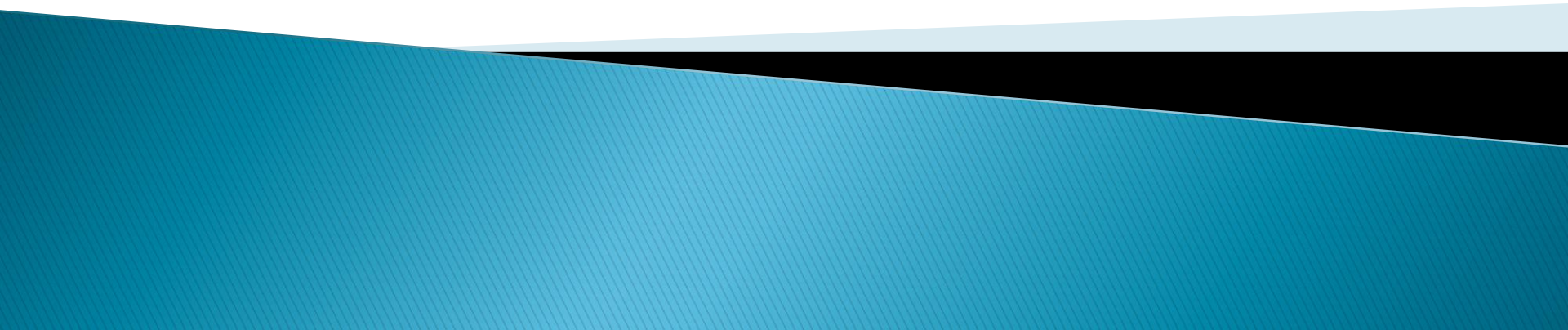


Arsitektur Sistem Terdistribusi

Pertemuan 2



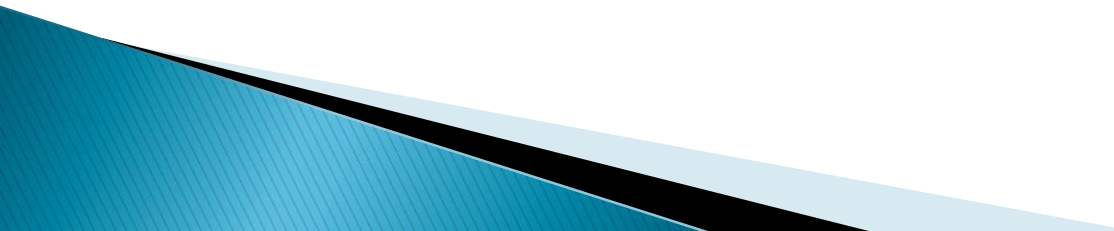
Objectives

- ▶ Pemahaman mengenai Model arsitektur SisTer
- ▶ Mengetahui Sudut pandang logis Arsitektur Sistem Tersebar
 - Layered architectures, Object-base architectures, Data-center architectures, Event-base architectures
- ▶ Memahami model Arsitektur sistem
 - Centralized architecture, Decentralized architecture, Hybrid
- ▶ Architecture Versus Middleware

Pengantar

- ▶ Sistem tersebar merupakan bagian dari sistem yang kompleks yang menghubungkan beberapa mesin.
- ▶ Untuk hal ini dibutuhkan **pengorganisasian sistem** yang **baik**.
- ▶ Ada dua hal dalam melihat pengorganisasian sistem tersebar, organisasi **secara logis** sebagai kumpulan perangkat lunak dan **secara fisik**.

Masalah Membangun SisTer

- ▶ Berhubungan dengan peletakan **komponen-komponen** dan juga relasi antar komponen
 - ▶ Memastikan struktur arsitektur bisa memenuhi kebutuhan dan membuat sistem **reliable, manageable, adaptable, dan cost-effective**
 - ▶ Sistem harus melakukan **klasifikasi** terhadap proses yang terjadi pada server, client, maupun peer
 - ▶ Sister memiliki **banyak sekali variasi**, tergantung dari jaringan komputer, performa, reliabilitas, keamanan, dan biaya
- 

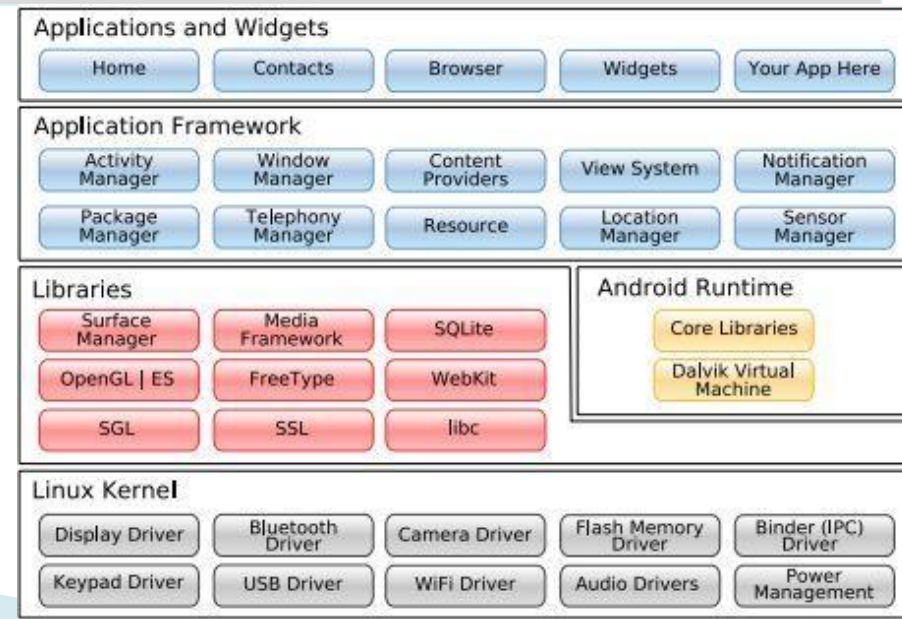
Kesulitan–kesulitan peletakan

- ▶ Dari sisi mode pemakaian
 - Variasi yang beragam terhadap karakteristik pemakaian sistem
 - Berapa kali suatu halaman dikunjungi?
 - Seberapa sibuk suatu server?
- ▶ Dari sisi masalah Internal
 - Masalah konkurensi akses
- ▶ Dari sisi masalah lingkungan sistem
 - Masalah heterogenitas: hardware, sistem operasi dan jaringan
- ▶ Dari sisi masalah ancaman eksternal
 - Masalah keamanan data

Arsitektur

Definisi:

Suatu rancangan untuk **penyusunan komponen-komponen** suatu sistem, dimana rancangan tersebut mengidentifikasi komponen serta fungsi masing-masing komponen, konektifitas antar komponen serta pemetaan fungsionalitas komponen.

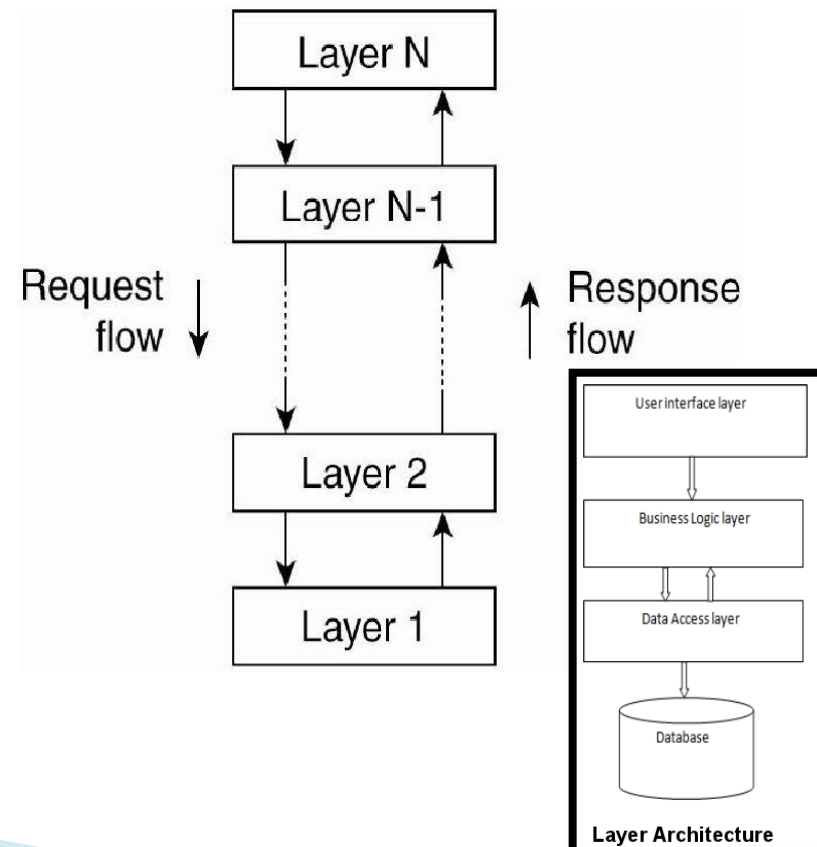


Model Arsitektur Sister

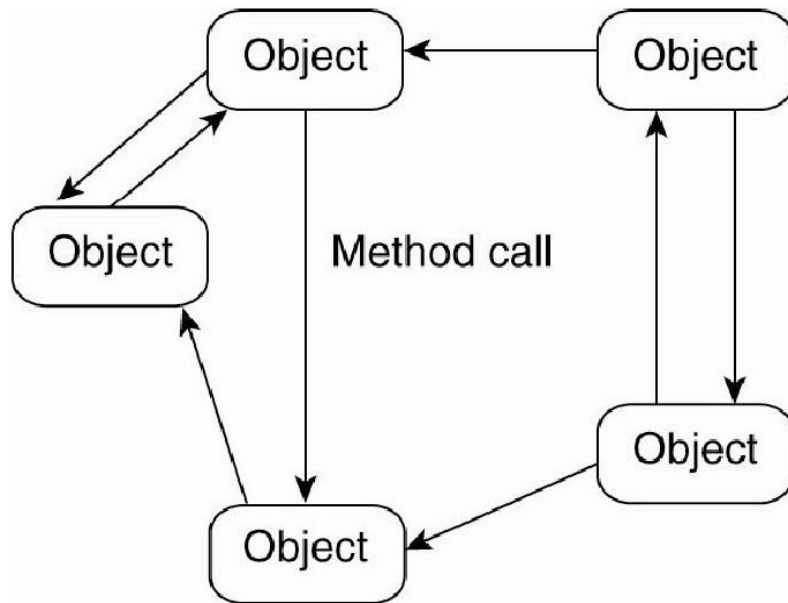
- ▶ Arsitektur Logis (*Software architecture*)
 - Organisasi **logika** dari komponen-komponen perangkat lunak
 - **Komponen** yang dimaksud berupa **unit modular** berupa **interface** yang dapat diproses di sistem yang **berbeda**
 - *RPC (remote procedure call), message passing*
 - Jenis **Model** arsitektur logis (style)
 - Layered architectures
 - Object-base architectures
 - Data-Center architectures
 - Event-based architectures
- ▶ Arsitektur Fisik (*System architecture*)
 - Peletakan mesin
 - Peletakan komponen perangkat lunak pada mesin sesungguhnya

Layered Architectures

- ▶ Komponen-komponen pada Layered architectures diorganisasi dalam bentuk **lapisan-lapisan (layer) fungsi dan service**
- ▶ Contoh:
 - Operating system (windows, linux)
 - Network Protocol (OSI, TCP/IP)

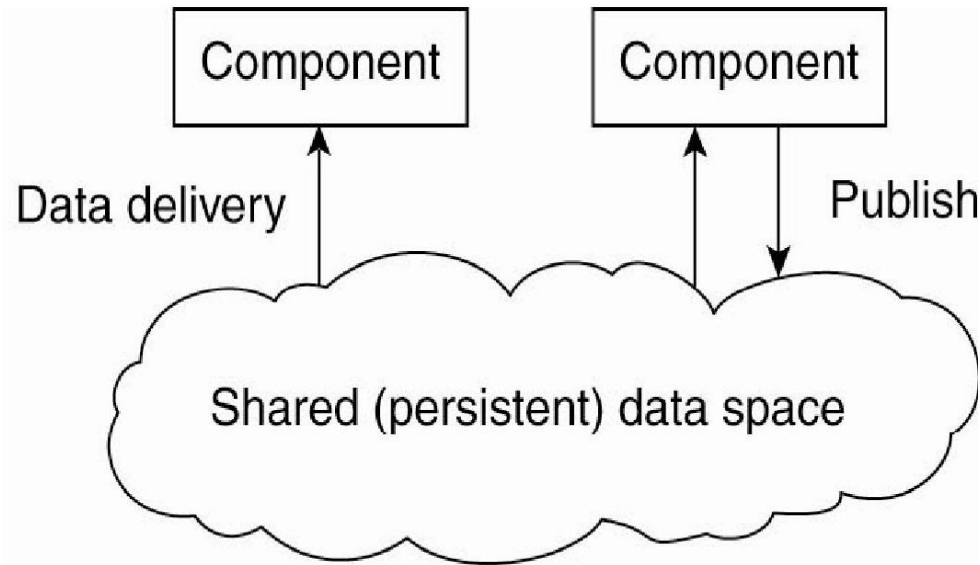


Object-base Architectures



- ▶ Object-base architectures menggambarkan setiap objek melakukan koresponden dengan komponen, dan komponen ini terkoneksi melalui mekanisme procedure call.
- ▶ Bentuk sistem OA ini digunakan aplikasi perangkat lunak dalam skala besar.

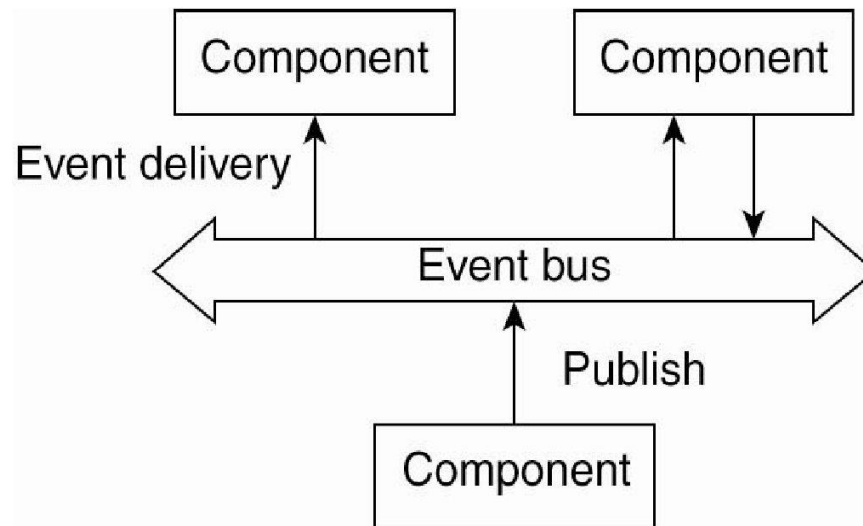
Data-center Architectures



- ▶ Data center dapat dipandang sebagai **gudang data (data warehouse)** yang berfungsi sebagai sistem pengelolaan data mulai dari **pengumpulan, pengolahan, penyimpanan** hingga penemuan kembali data, serta mampu pula memberikan dukungan dalam pengambilan keputusan.
- ▶ Sebagai contoh adalah sistem tersebar berbasis web.

Event-based Architectures

- ▶ Proses EBA pada dasarnya berdasarkan **propagasi** event. Proses mengeluarkan event setelah Middleware memberikan kepastian hanya proses itu saja yang bisa di subscribe untuk event yang diterima. Keuntungan EBA adalah proses bersifat loosely coupled.



System Architecture

A. Centralized Architectures (Client–Server)

- Application Layering
- Multi–tiered Architectures

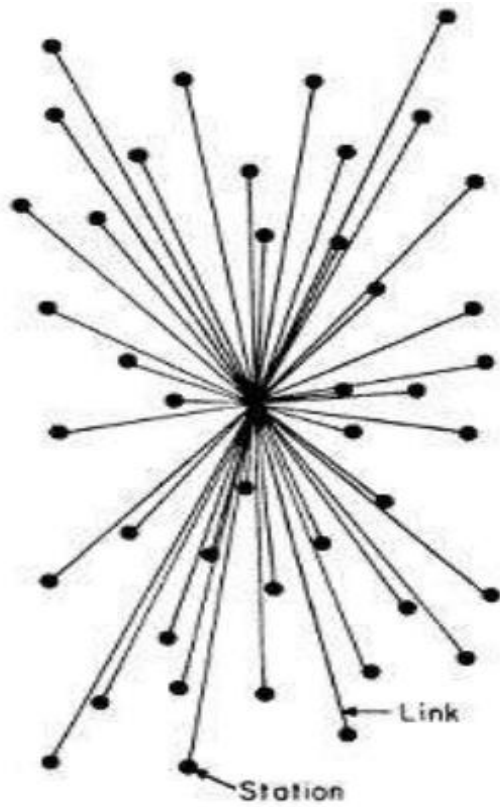
B. Decentralized Architectures

- Structured P2P (Peer–to–Peer) Architectures
- Unstructured P2P Architectures
- Topology Management of Overlay Networks
- Superpeers

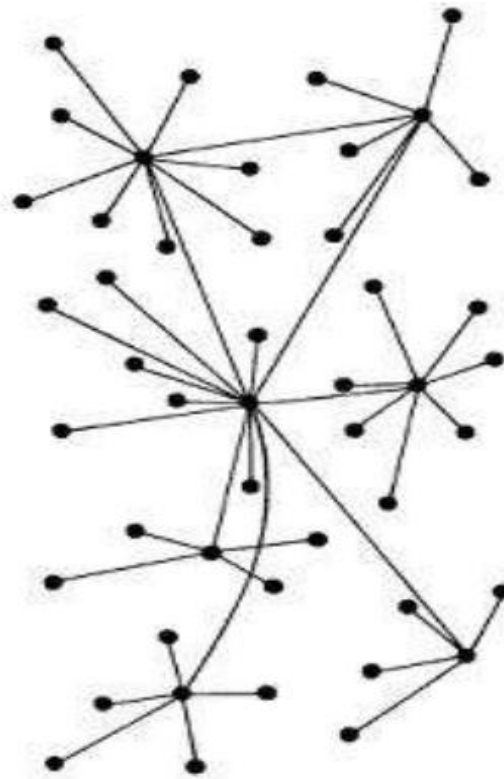
C. Hybrid Architectures

- Edge–Server Systems
 - Collaborative Distributed Systems
- 

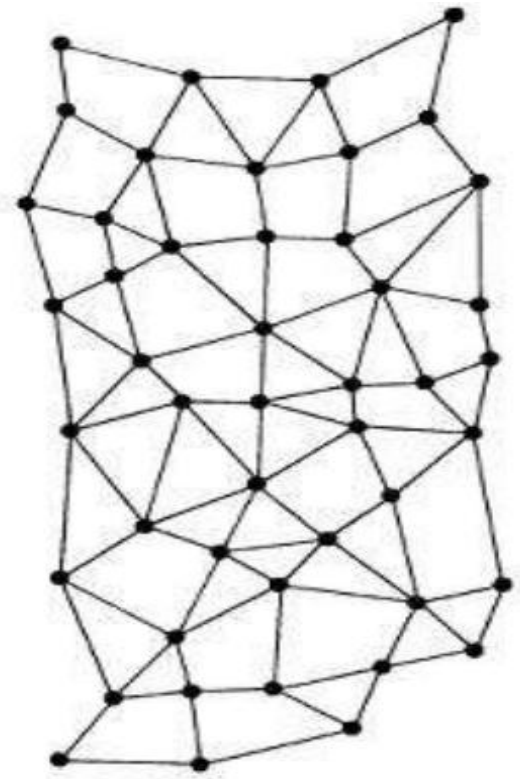
System Architecture



CENTRALIZED
(A)



DECENTRALIZED
(B)

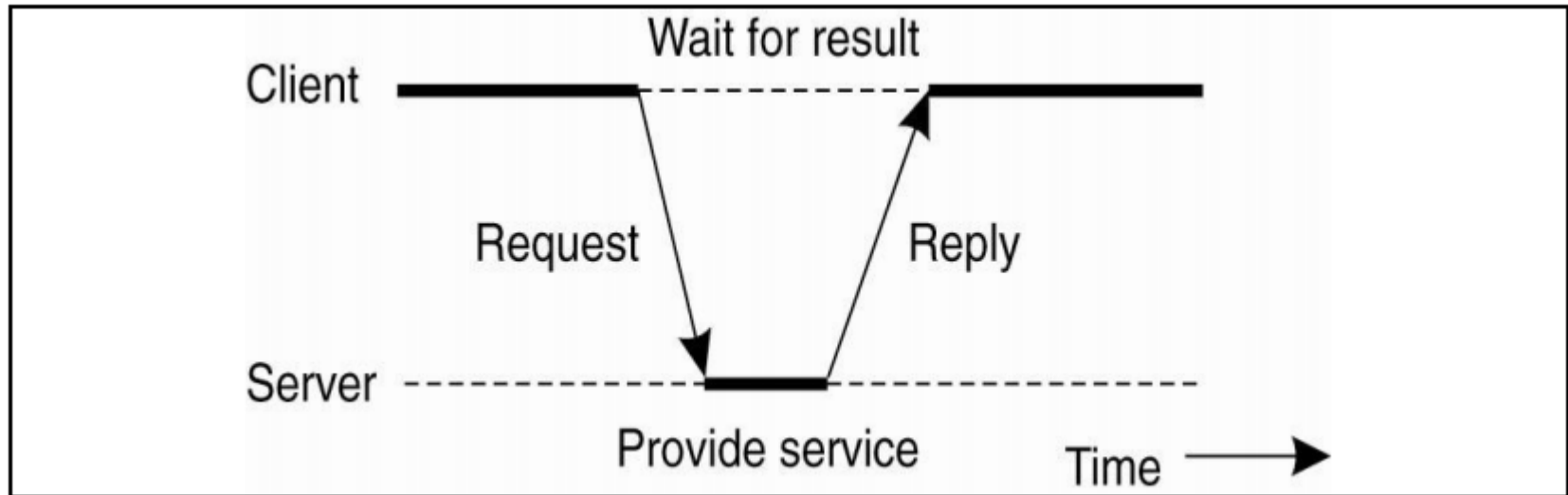


DISTRIBUTED
(C)

A. Centralized Architectures (Client–Server)

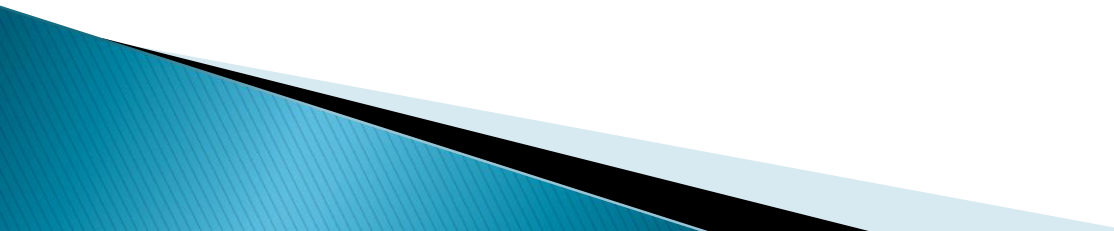
- Application Layering
- Multi-tiered Architectures

Centralized Architectures (Client-Server)

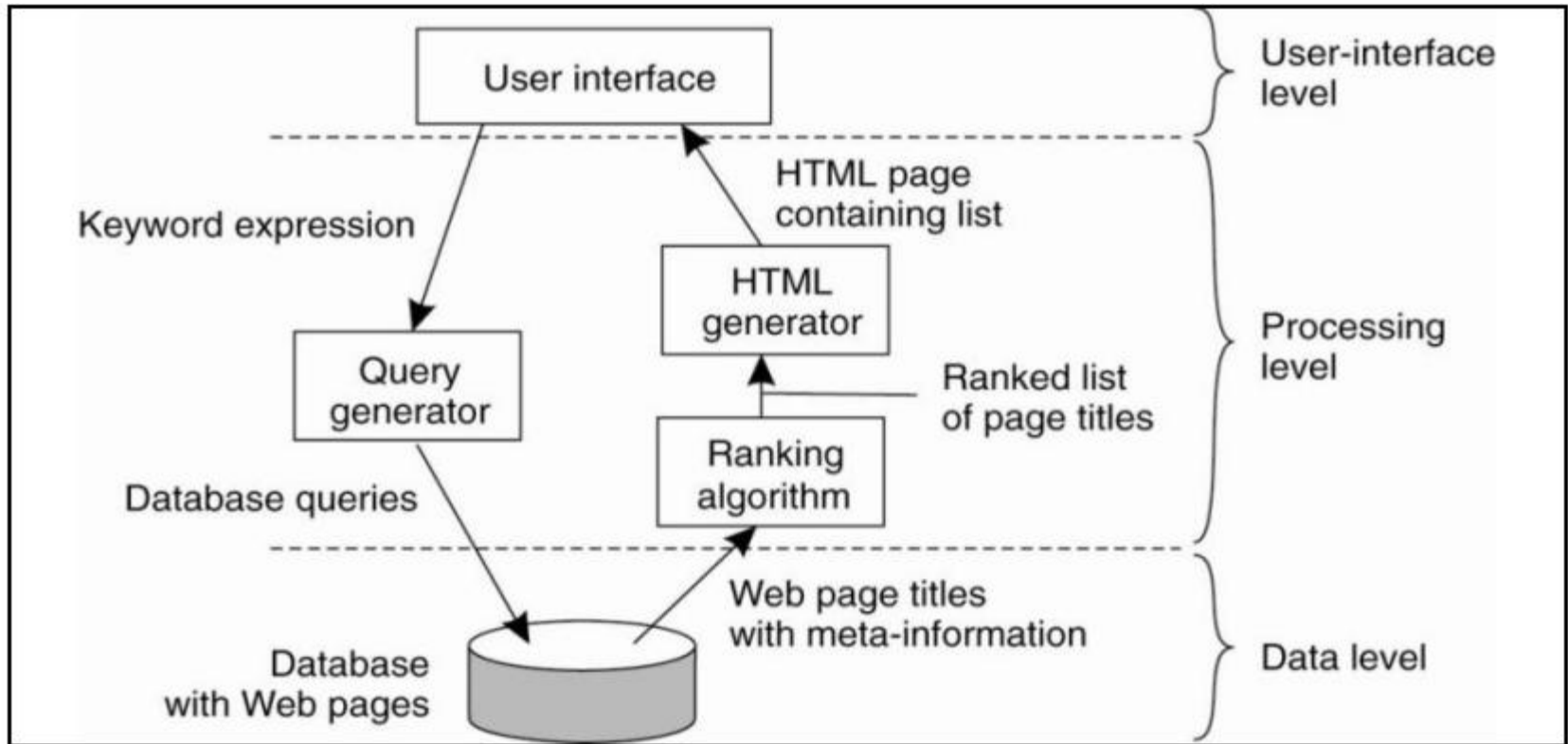


- ▶ *Client server* unggul dalam **kecepatan** dan mendukung **jaringan besar**.
- ▶ Kekurangan terdapat pada **sisi setup** yang cukup **komplek**, **biaya** tinggi dan membutuhkan **sumberdaya manusia** yang handal untuk mengelola.
- ▶ Pada model client server, terdapat perilaku yang biasa disebut ***request-reply behavior***

Application Layering

- ▶ Model client server seiring perkembangannya mengundang perdebatan mengenai perbedaan antara client dan server itu sendiri
 - ▶ Pada umumnya client server architecture ditujukan untuk keperluan user access ke database, maka dari itu layered architectural style dibagi menjadi:
 - user-interface level (display management)
 - processing level (applications)
 - data level (actual data that is being acted on)
- 

Application Layering



- ▶ The simplified organization of an Internet search engine into three different layers.

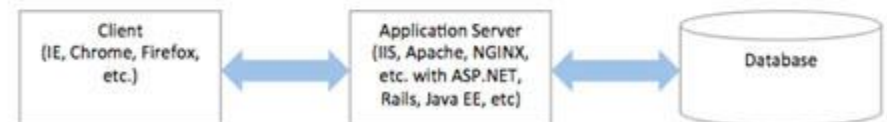
Multitier Architecture

- ▶ **2 tier** architecture
- ▶ Pengorganisasian paling simple dimana terdiri atas 2 type mesin
 - Client yang berisi implementasi program pada user-interface level
 - Server yang berisi implementasi program pada proses dan data level

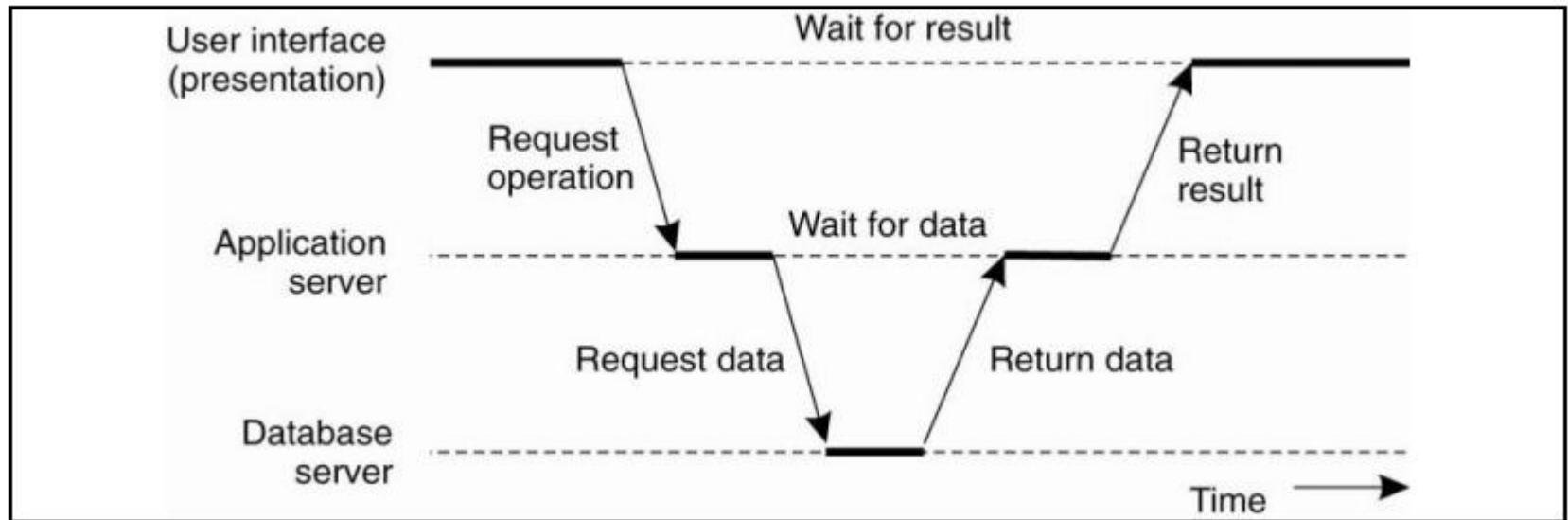
2-Tiered Architecture



3-Tiered Architecture

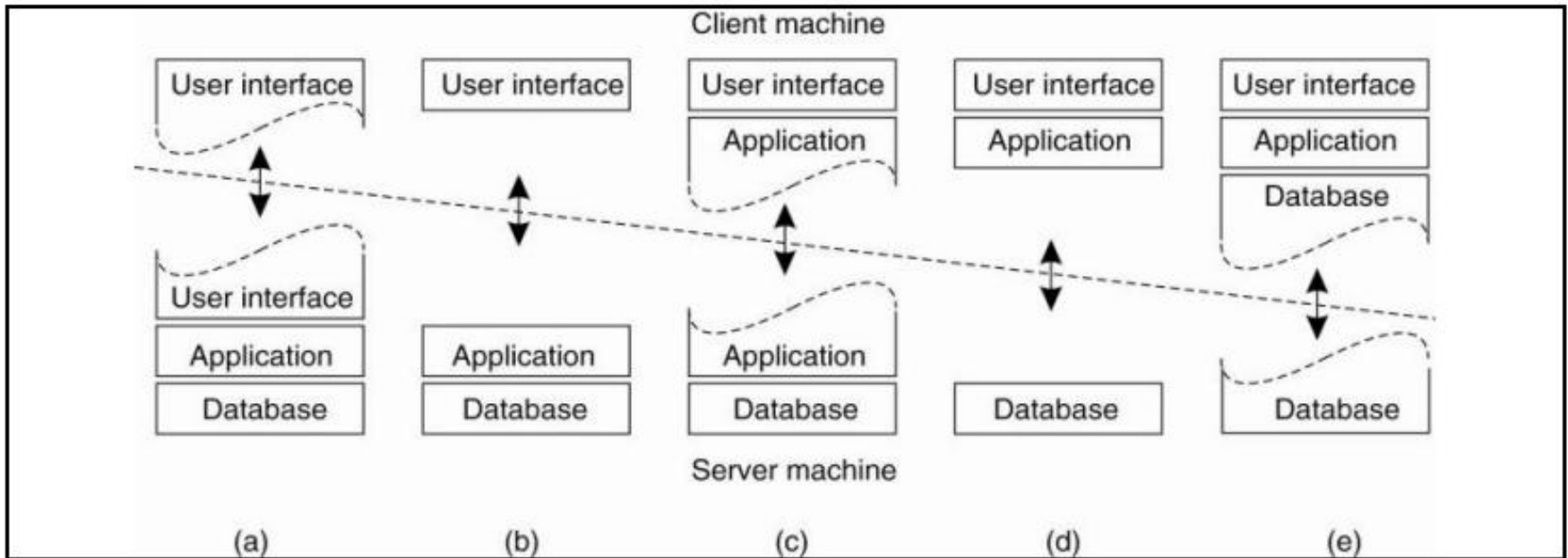


Multitier Architecture



- ▶ 3 tier architecture
- ▶ Pada keperluan khusus, kadang server juga perlu bertindak sebagai client
- ▶ Pada arsitektur ini, program pada processing level tidak hanya terdapat pada server yang terpisah, bahkan dapat terdistribusi pada client dan server mesin

Multitier Architecture



Gambar : Alternatif organisasi Client Server

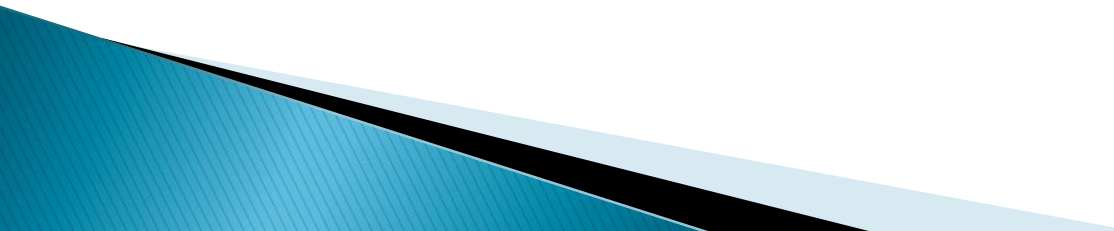
- Trend perkembangan model arsitektur

Multitier Architecture

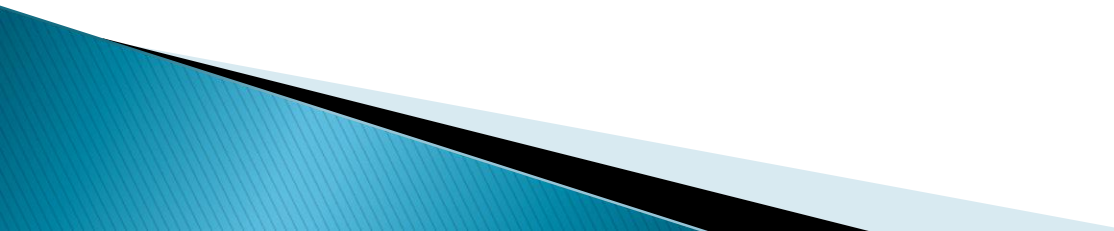
► Ringkasan

- Perbedaan tier berkaitan dengan aplikasi logis organisasi.
- Disebut sebagai vertical distribution dimana karakteristik tipe ini adalah menempatkan secara logis komponen yang berbeda pada mesin yang berbeda juga
- Memiliki vertical distribution dapat membagi secara logis maupun fisik dalam beberapa mesin yang berbeda, dimana masing-masing mesin dapat menjalankan fungsi khusus atau tergabung pada sebuah grup untuk fungsi tertentu

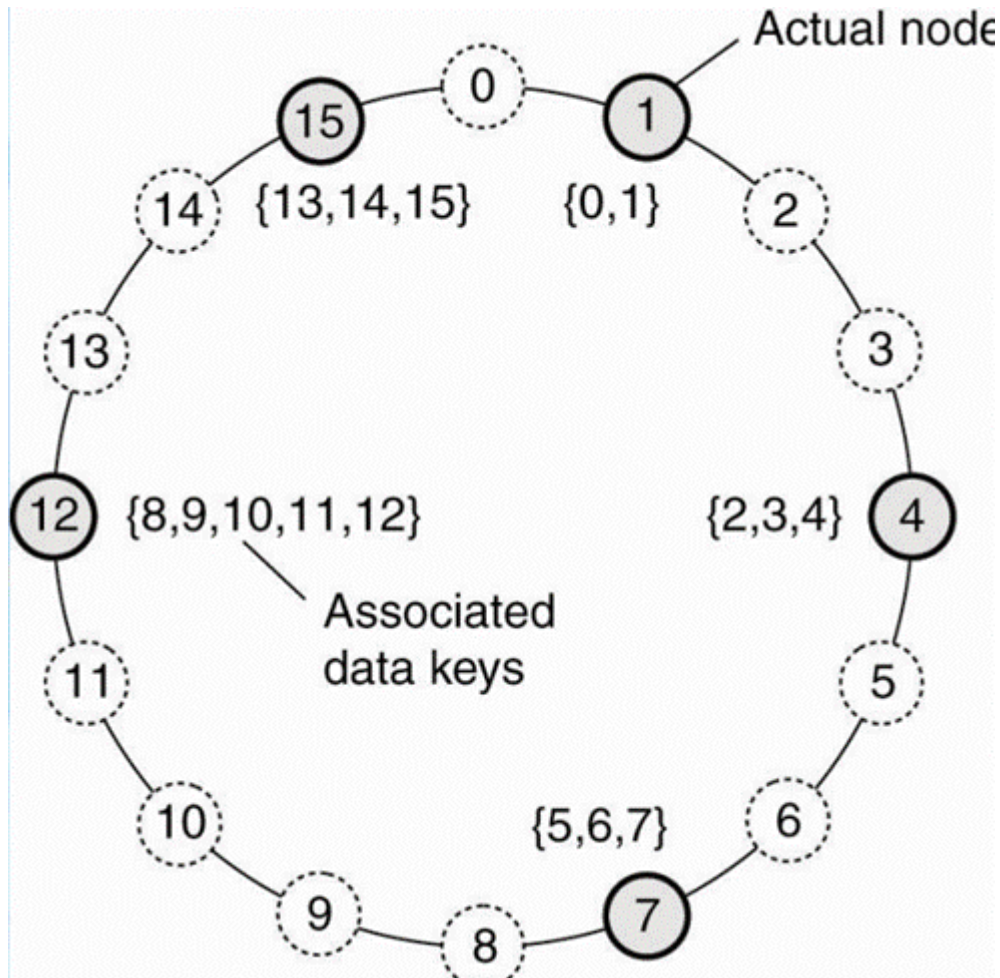
B. Decentralized Architecture

- Structured P2P (Peer-to-Peer) Architectures
 - Unstructured P2P Architectures
 - Topology Management of Overlay Networks
 - Superpeers
- 

Decentralized Architecture

- ▶ Disebut sebagai Horizontal distribution
 - ▶ Pada arsitektur ini, secara fisik terpisah namun secara logis memiliki fungsi level yang sama (equivalent), dimana setiap mesin memproses bagiannya sendiri kemudian melakukan *balancing* terhadap hasil proses.
 - ▶ Nama lain *Peer-to-peer architecture*
- 

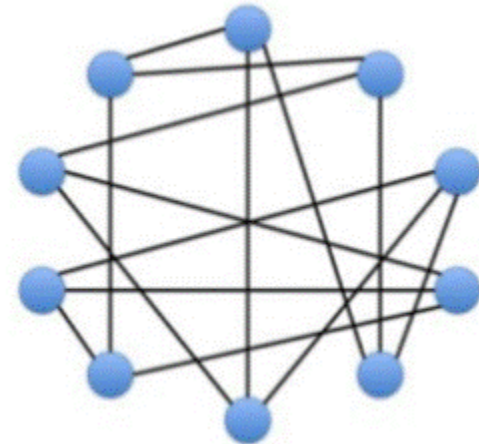
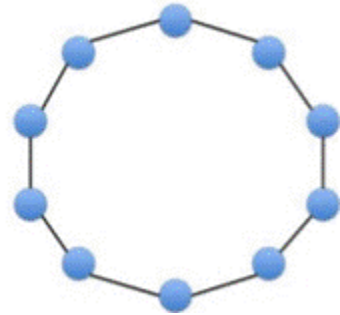
Peer-to-peer architecture



- ▶ Perkembangan P2P arsitektur tidak lepas dari pertanyaan
 - Bagaimana organisasi proses dalam jaringan
 - Sebuah proses tidak dapat berhubungan secara langsung dengan proses lain di jaringan
 - Diperlukan sebuah pesan khusus untuk komunikasi proses

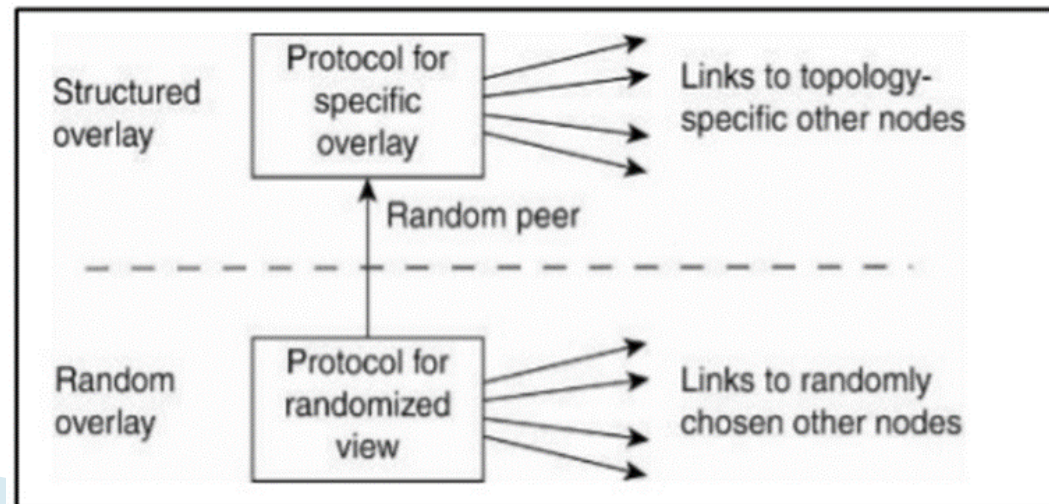
Peer-to-peer architecture

- ▶ Structured peer-to-peer architecture
 - Dalam struktur ini lapisan jaringan di bangun menggunakan deterministic procedure, seperti menggunakan distributes hash table (DHT).
- ▶ Unstructured peer-to-peer architecture
 - Dalam struktur ini menugaskan sebagian besar pada algoritma secara acak untuk membangun lapisan jaringan. Pada intinya setiap node mendata jaringan node neighbor, tetapi data node tersebut di tempuh dengan proses acak sederhana.



Peer-to-peer architecture

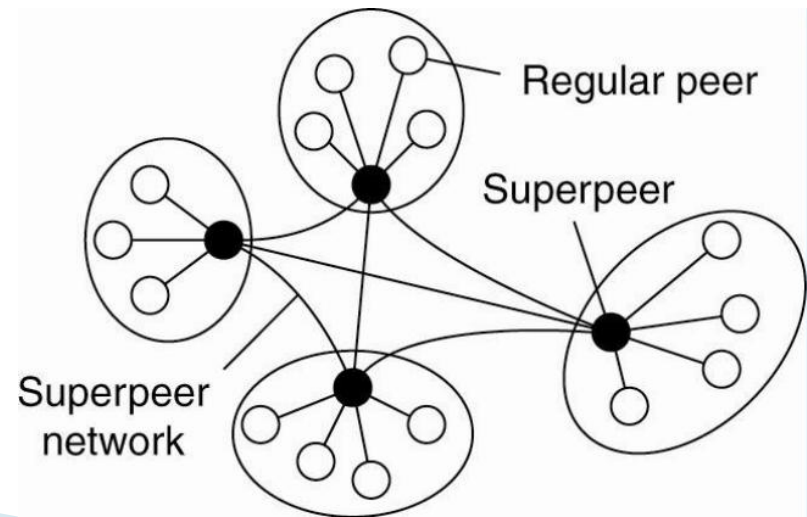
- ▶ Topology Management of Overlay Networks
 - Walaupun pada structured dan unstructured peer-to-peer System cukup jelas, namun dalam beberapa kasus masih belum lengkap. Satu kunci dari observasi adalah kehati-hatian dari proses pertukaran dan pemilihan entries dari pandangan parsial dimana topologi tertentu dapat dibangun dan dijaga konektivitasnya.
 - Pendekatan TMOOD in diperoleh dengan mengambil dua pendekatan Layering, yaitu seperti gambar berikut :



Peer-to-peer architecture

► Superpeers

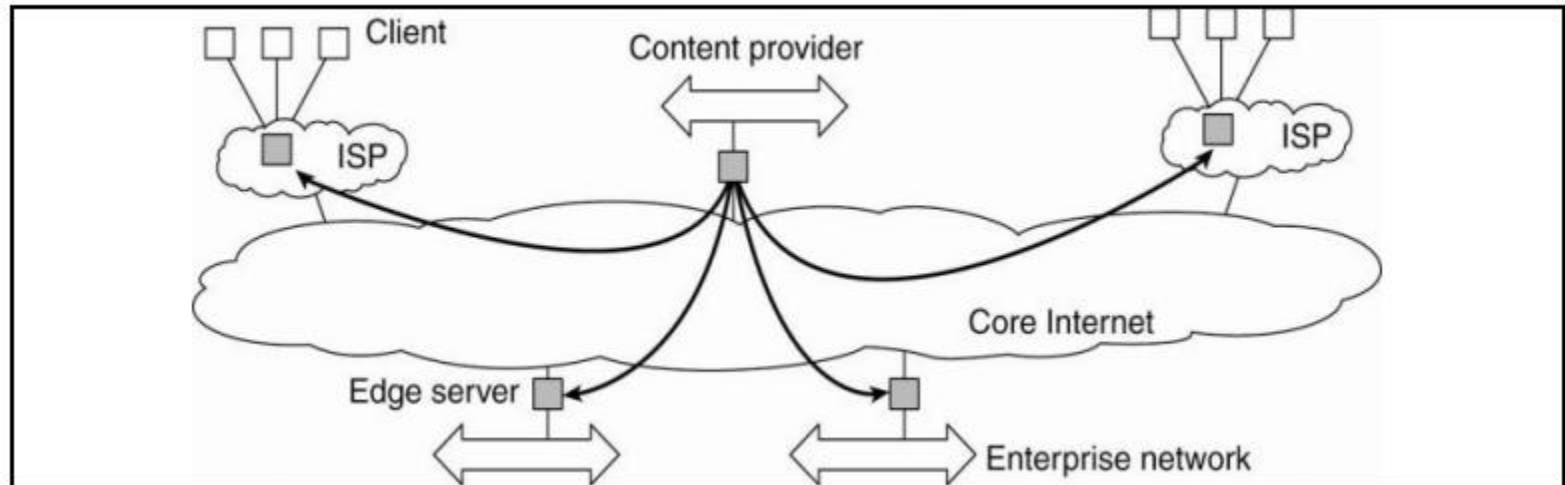
- Menangani masalah yang disebabkan penempatan item data ketika jaringan berkembang.
- Teknik Superspeers dapat menangani masalah yg terkait dengan scalability, karena dapat mempertahankan konektifitas terhadap item data.
- Umumnya Superspeers digunakan pada peer to peer network



Hybrid Architecture

▶ Edge-Server Systems

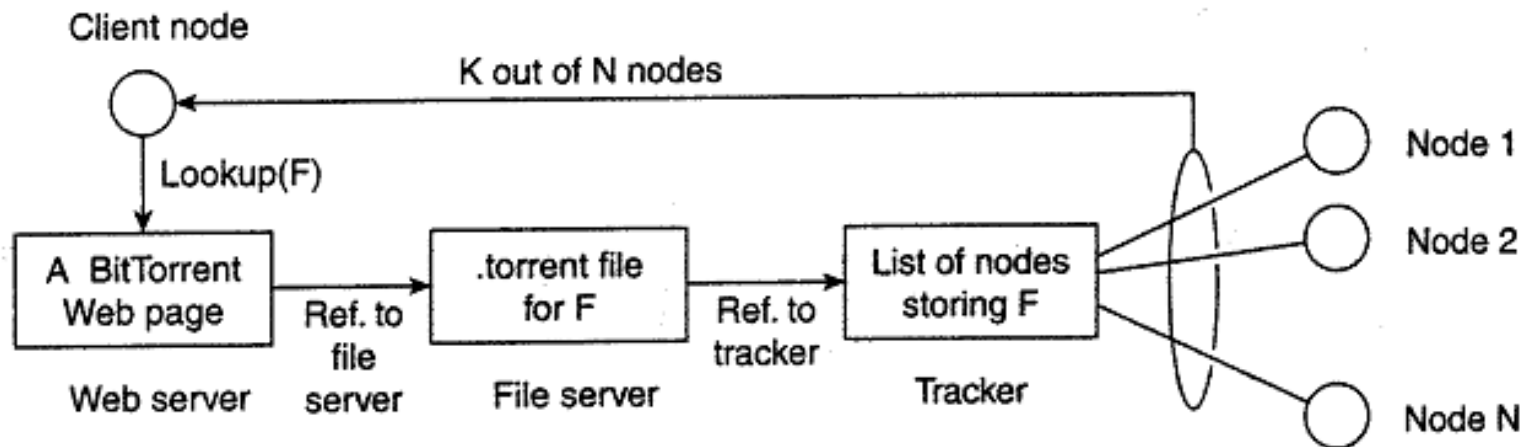
- Sistem ini dibangun di jaringan internet dimana server ditempatkan pada edge (tepi) dari jaringan.
- Tujuan Edge server adalah melayani content (isi), pada saat proses filtering dan fungsi transcoding



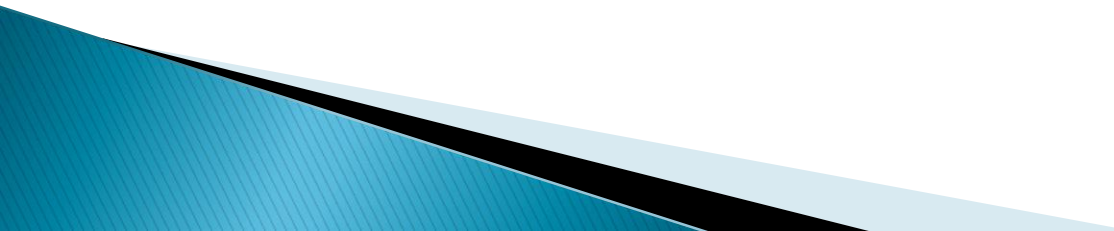
Hybrid Architecture

► Collaborative Distributed Systems

- Bentuk lainnya adalah CBS ini dibangun dari beberapa jaringan sistem tersebar yang ada.
- Konsep sama dengan BitTorrent file-sharing system
- Component dapat *redirect* client untuk akses server lain, analisa pola akses client, manage replication data

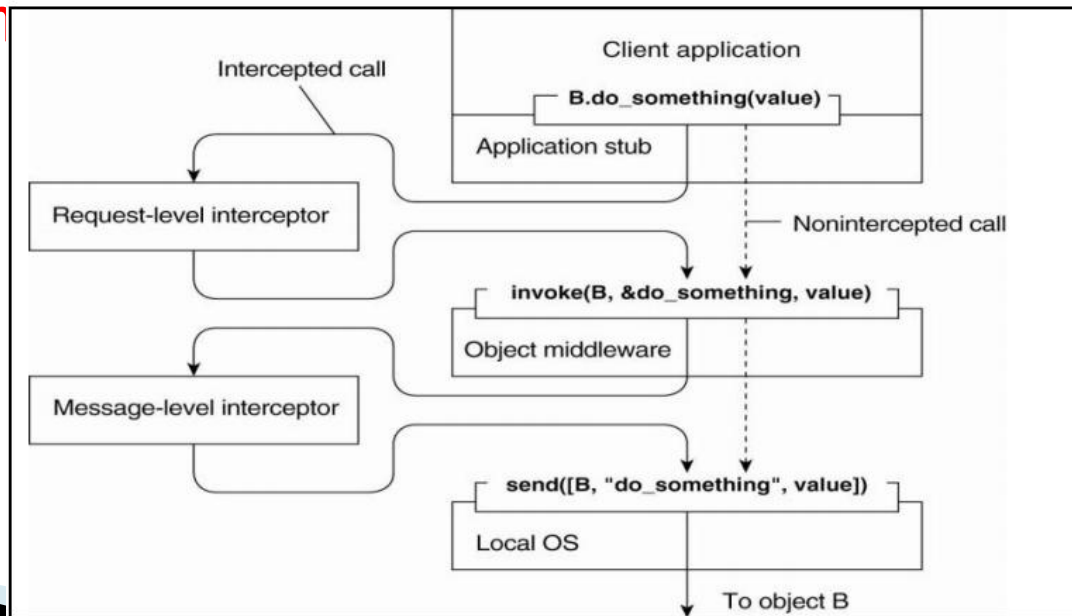


Architecture Versus Middleware

- ▶ Tidak membahas perbandingan Arsitektur dan Middleware
 - ▶ Middleware mengikuti bentuk arsitektur yang ada.
 - ▶ Middleware dan aplikasi menangani kebutuhan berbeda namun nantinya tetap dibutuhkan solusi dimana middleware mudah untuk di konfigurasi, disesuaikan dan di kostumisasi sesuai kebutuhan aplikasi.
- 

Interceptors

- ▶ **Interceptors** merupakan perangkat lunak yang memecah aliran pengendalian dan mengijinkan kode lain untuk di eksekusi/proses.
- ▶ Interceptors sangat baik untuk menyediakan proses *transparency* dari **Replication** dan **Performance**



General Approaches to Adaptive Software

- ▶ Kebutuhan akan penyesuaian terhadap **lingkungan aplikasi** di sistem tersebar adalah perubahan secara terus menerus.
- ▶ Perubahan ini sebagai hasil dari *mobility, quality-of-service networks, kerusakan hardware, dan battery drainage dll.*
- ▶ Konsep ini disebut sebagai **adaptive software**
- ▶ McKinley et al. (2004) membagi 3 teknik dasar menuju adaptive system
 - Separation of concerns:
 - Computational reflection
 - Component-based design (stand-alone)

Terima Kasih