DIS - Netværk 1 - Physical, Architectural, fundamental

Architectural Models
Side 40

Applications, Services
Middelware
Operating System
Computer and network hardware
(Sidste 2 - Platform)

Platformsuafhængigt

2-tier og 3-tier model
Den eneste måde vi kan bygge en 2-tier er ikke at have en database. Har vi en database under eller ikke under. Det er forskellen på 2-tier og 3-tier modellen. **Trelagsmodellen** er datalaget, logiklaget og præsentationslaget

Tyk vs Tynd klient

Tynd klient
En tynd klient håndterer ikke selv logik, men står udelukkende for kommunikation med en central service - som har alt logik liggende.

Tyk klient
Logikken og dermed beregningerne bliver håndteret af klienten, som kommunikerer resultaterne tilbage til serveren - eller andre klienter.

Det behøver ikke være defineret som online/offline, idet vi nogle gange i en online klient vil have logik på klienten af hensyn til performance.

Architectural Elements

Communicating entities

System-oriented perspective
- Nodes
- Threads

Problem-oriented perspective
- Objects
- Components
- Web services

Communication paradigms

Interprocess communication

Remote Invocation
- **Request-reply protocols**
- **Remote Procedure Calls (RPC)**
- **Remote Method Invocation (RMI**

Indirect Communication
- **Space uncoupling**
  Senders do not need to know who they are sending to
- **Time uncoupling**
  Senders and receivers do not need to exist at the same time
- **Distributed shared memory**
- **Publish-subscribe systems**
- **Group Communication**
- **Message queues**
- **Tuple spaces**
- **AJAX**

Placement
- **Mapping of services to multiple servers**
- **Proxy**
- **Caching**
- **Mobile code**
- **Mobile agents**

Physical Models
Side 39

Server
En server er vel bare en computer, der i en given situation indeholder data, en anden computer skal tilgå.

Klient
Enhver form for programmel, som ikke selv ejer sine ressourcer.

Proxy Server / Caching

Proxy
Al trafik kører igennem proxien. Man kan bare fjerne facebook fra internettet fx

Load balancer

Cache server
EB har en proxy server – når vi spørger ekstrabladet, så spørger vi direkte fra proxyserveren som er en cacheserver. Fx behøver jeg ikke at indberegne hvilke 10 artikler der er læst nu. Fx regner den det ud hvert 10. minut – så sparer man kræfter over på serveren. Den gemmer det HTTP request. Når vi sender den, så kan proxyen bare returnere et svar

fra test
En proxy server skal ses som et ekstra led imellem et lokalnetværk og internettet. Proxy serveren kan have mange funktioner, som bl.a. virke som et ekstra sikkerhedsled. Proxy-serveren kan ydermere regulere al aktiviteten der skal foregå mellem lokalnetværket og internettet. Det kan også være en cache server og kunne andre ting

Peer to Peer

Har man nogen central server i peer to peer – bittorrent – fungerer ved at man henter en eller anden fil – en tracker – som indeholder en liste med hvor vi kan finde en liste med hvor filen findes.

Trackeren indeholder altså filen – det er "serveren" - typisk være for simpelthen at holde styr på hvem kan man snakke med.

Fundamental models
Side 61

Interaction Models
- **Tidshåndtering**
- **Lazy-load**
- **Synkront (blocking) interaktion**
  Synchronous distributed systems
- **Asynkront (non-blocking) interaktion**
  Asynchronous distributed systems
- **Client/Server**
- **Peer to Peer**
- **Performance of communication channels**
  - **Latency**
  - **Bandwith**
  - **Jitter**
- **Event ordering**
  - Real-time ordering of events
  - Logical time

Failure Models
- **Fail-stop** - Process
  Process halts and remains halted. Other processes may detect this state
- **Crash** - Process
  Process halts and remains halted. Other process may not be able to detect this state
- **Omission** - Channel
  A message inserted in an outgoing message buffer never arrives at the other end's incoming message buffer
- **Send-omission** - Process
  A process completes a send operation but the message is not put in the outgoing message buffer
- **Receive-omission** - Process
  A message is put in a process's incoming message buffer, but that process does not receive it.
- **Arbitrary** - Process or Channel
  Process/Channel exhibits arbitrary behaviour. It may send/transmit arbitrary messages at arbitrary times or commit omissions; a process may stop or take an incorrect step
- **Reliability of one-to-one commucation**
  - **Vailidty**
    Any message in the outgoing message buffer is eventually delivered to the incoming message buffer.
  - **Integrity**
    The message received is identical to the one sent, and no message are delivered twice

Security Model

The security of a distributed system can be achieved by securing the **processes** and the **channels** used for their interaction and by protecting the objects that they encapsulate against unauthorized access

- **Cryptography and shared secrets**
- **Authentication**
- **Secure channels**
- **DOS and Mobile code**
  - Runnable .exe files

Threads to processes

Servers: Since a server can receive invocations from many different clients, it cannot necessarily determine the identity of the principal behind any particular invocation. Even if a server requires the inclusion of the principal's identity in each invocation, an enemy might generate an invocation with a false identity. Without reliable knowledge of the sender's identity, a server cannot tell whether to perform the operation or to reject it

Clients: When a client receives the result of an invocation from a server, it cannot necessarily tell whether the source of the result message is from the intended server or from an enemy, perhaps 'spoofing' the mail server. Thus the client could receive a result that was unrelated to the original invocation, such as a false mail item (one that is not in the user's mailbox).

Threads to channels

An enemy can copy, alter or inject messages as they travel across the network and its intervening gateways. Such attacks present a threat to the privacy and integrity of information as it travels over the network and to the integrity of the system