Sisteme distribuite.
Definitie si caracteristici

Capitolul 1

2013

Definirea sistemelor distribuite

- Sistem distribuit (SD): sistem ale carui componente se afla pe calculatoare interconectate in retea, comunica si se coordoneaza prin transfer de mesaje.
- Caracteristici: *concurenta componentelor*, *lipsa unui ceas global*, *erori* (defecte, caderi, engl.*failures*) *independente ale componentelor*.
- Andrew Tanenbaum: "*A distributed system is a collection of independent computers that appears to its users as a single coherent system*"
- Exemple:
  - Internet, Intranet, Web
  - Sistem *workflow* care coordoneaza activitatea angajatilor intr-o organizatie virtuala
  - Sisteme *grid* sau *cloud*
- Motivul principal al introducerii SD: partajarea resurselor. Resursele se incapsuleaza in servere sau obiecte accesate apoi de diversi clienti.
- Probleme ale proiectarii si construirii SD: *eterogenitatea componentelor*, *deschiderea/extensibilitatea* (engl.*openness*), *scalabilitatea*, *securitatea*, *tratarea erorilor*, *concurenta*, *transparenta*.

2013

Consecinte ale definitiei SD

- SD = *sistem ale carui componente se afla pe calculatoare interconectate in retea, comunica si isi coordoneaza actiunile prin transfer de mesaje.*
- Definitia acopera o gama foarte larga de sisteme ce se pot studia folosind conceptul de SD: retele de telefonie mobila, retele ale unor corporatii, retele industriale, retele pentru controlul unui automobil, etc.
- Concurenta = executia concurenta a programelor intr-un SD. O problema este coordonarea programelor concurente ce acceseaza resurse partajate.
- Lipsa unui ceas global: Este o consecinta a faptului ca unicul mecanism de comunicare intre componente este transferul de mesaje.
- Erori independente: Orice componenta a unui SD poate fi afectata de erori si proiectantul trebuie sa aiba in vedere acest lucru. Exemple de erori: deconectarea sau defectarea unui calculator din retea, terminarea neasteptata a unui program (engl.*crash*).
- Alte motive ale introducerii SD: partajarea si virtualizarea resurselor de calcul (de exemplu in *grid & cloud computing*), toleranta la defecte (engl.*fault tolerance*).

Sub-sistemul hardware al SD

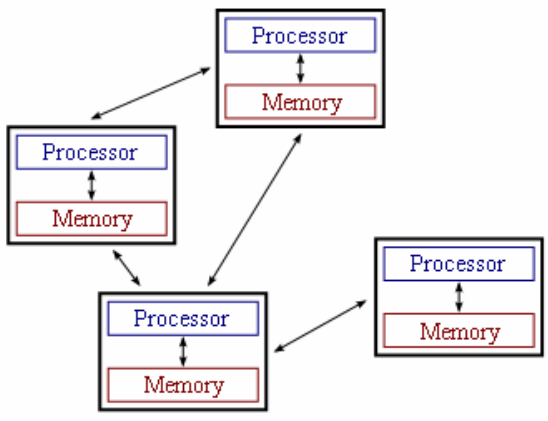
- La baza unui SD se afla o multime de CPU-uri interconectate. Exista dpdv al memoriei doua mari categorii:
  - Interconectare prin memorie comuna – *sisteme multiprocesor* (engl. *multiprocessors*)
  - Interconectare fara memorie comuna – *sisteme multicalculator* (engl. *multicomputers*)
- In functie de modul de interconectare (se aplica ambelor scheme de mai sus) avem:
  - Interconectare prin *magistrala comuna* (engl. *bus*) – de exemplu reteaua de televiziune prin cablu
  - Interconectare prin *retea de comutare* (engl. *switch*) – reteaua telefonica clasica
- Pentru sistemele multicalculator este relevanta distinctia intre:
  - Sisteme omogene
  - Sisteme eterogene

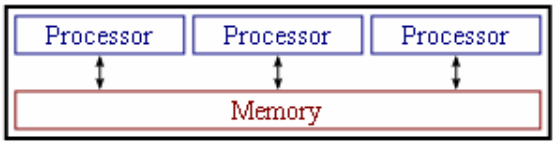
Sub-sistemul software al SD

- Software-ul SD are doua functii majore asemanatoare functiilor unui sistem de operare:
  - Functia de gestionare a resurselor
  - Functia de masina virtuala
- Sistemele de operare pentru SD se impart in doua mari categorii (distinctia este legata si de clasificarea sub-sistemului hardware):
  - Sisteme cu cuplaj strans
  - Sisteme cu cuplaj slab
- Sistemele cu cuplaj strans se numesc si *sisteme de operare distribuite*. Ele sunt folosite pentru gestionarea sistemelor multiprocesor sau sistemelor multicalculator omogene.
- Sistemele cu cuplaj slab se numesc si *sisteme de operare de retea*. Ele se utilizeaza pentru gestiunea sistemelor multicalculator eterogene. Pentru a putea deservi un SD, serviciile unui sistem de operare de retea trebuie extinse cu un nivel suplimentar numit *middleware*.

Sisteme paralele si concurente

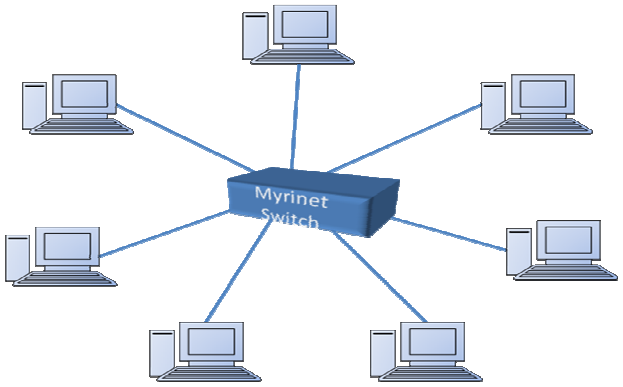
- Exista suprapuneri intre conceptul de SD si alte concepte ca *sisteme concurente* SC si/sau *sisteme paralele* SP.
  - SD se poate descrie ca SP deoarece procesele unui SD ruleaza paralel.
  - SP poate fi descris ca SD cu cuplaj strans; SD poate fi descris ca SP cu cuplaj slab.
- Se accepta tacit urmatoarele deosebiri:
  - In SP toate procesele au acces si comunica printr-o memorie partajata
  - In SD toate procesele au doar acces la o memorie privata si informatia este interschimbata exclusiv prin mesaje.





Exemplu – cluster Myrinet

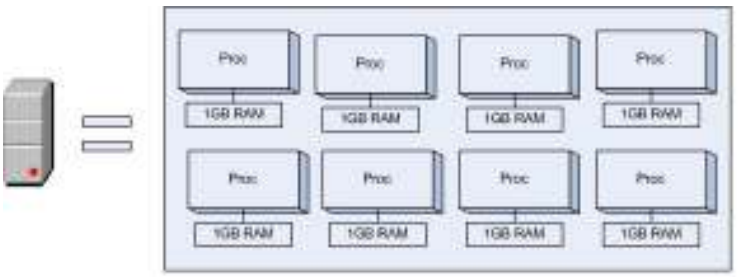
- 8 PC-uri conventionale
- High-speed interconnection network 2Gb/s
- Fibra optica
- Myrinet switch

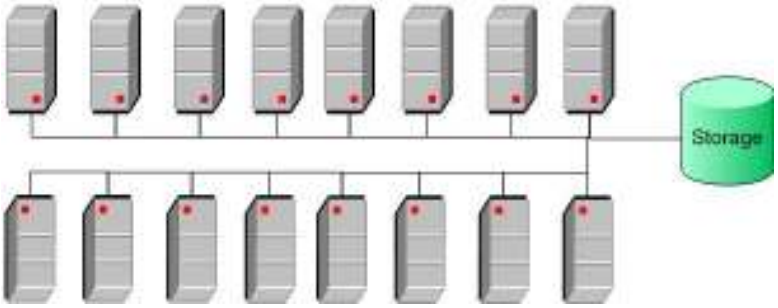


2013

Exemplu – cluster InfraGRID

- InfraGRID cluster
- 16 servers × 8 cores/server = 128 cores
- 1 GB RAM/core
- Linux
- Condor workload management system
  - job queueing mechanism,
  - scheduling policy,
  - priority scheme,
  - resource monitoring,
  - resource management

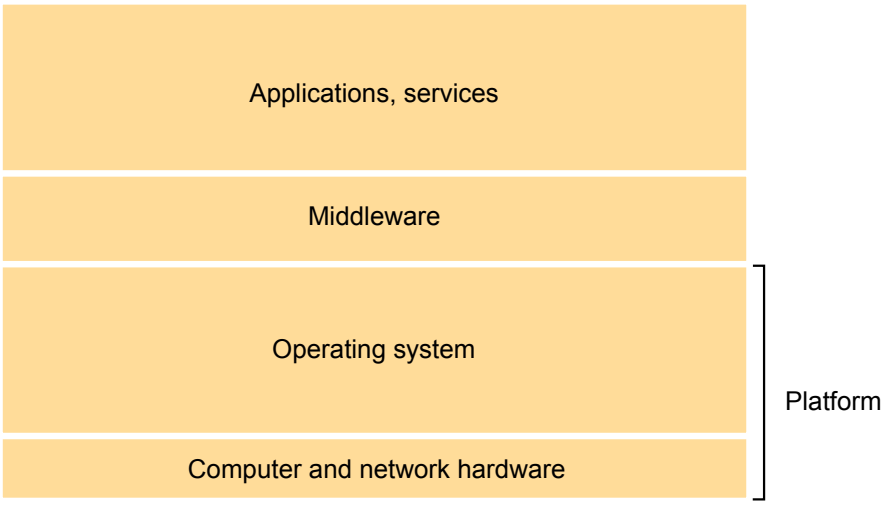




2013

Straturi ale software-ului SD

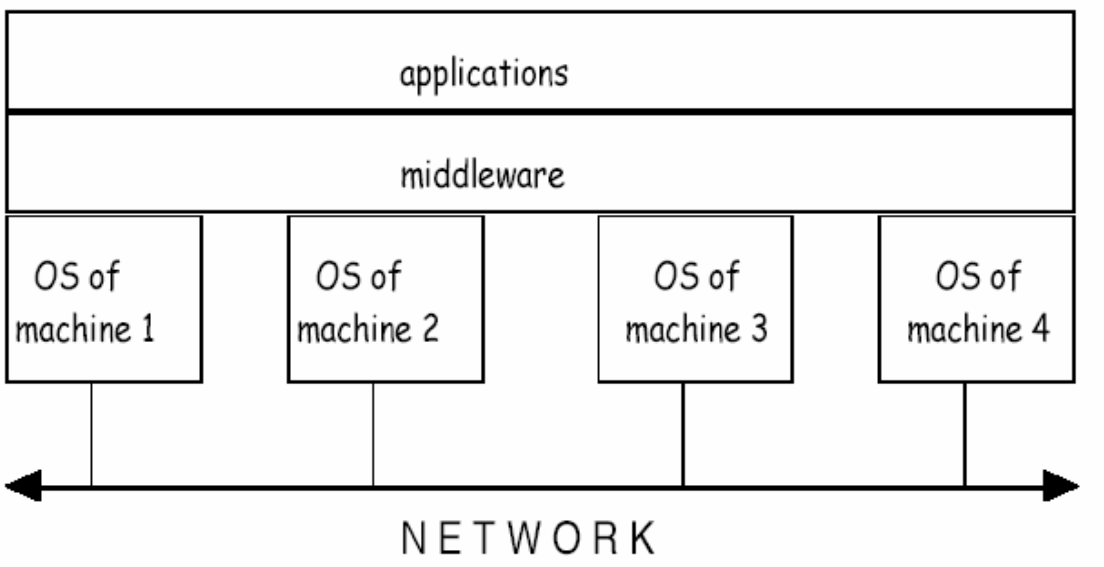
- Software-ul unui SD se structureaza intr-o ierarhie de straturi (engl.*layer*) sau module ofertante respectiv consumatoare de *servicii*. Serviciile sunt ofertate si consumate de procese aflate pe acelasi calculator sau pe calculatoare diferite.
- Viziunea orientata pe procese si servicii a unui SD se reprezinta printr-o ierarhie de servicii (engl.*service layers*). Un proces furnizor de servicii se numeste proces server si un proces consumator de servicii se numeste proces client.



Platforme si *middleware*

- Nivelul hardware si nivelul software imediat superior serviciilor sistemului de operare se numeste *platforma* a SD si a aplicatiilor acestuia. *Interfata de programare* (engl.*API*) a SD abstractizeaza nivelul comunicarii si coordonarii proceselor.
- Exemple de platforme: Intel x86/Windows, Sun SPARC/SunOS, Intel x86/Solaris, PowerPC/MacOS, Intel x86/Linux.
- *Nivel intermediar* (engl.*middleware*) = un nivel software al carui scop este mascarea eterogenitatii diverselor platforme si furnizarea unui model convenabil de programare a programelor de aplicatie. Nivelul *middleware* este compus din procese si obiecte alocate pe diverse calculatoare ce interactioneaza unele cu altele in vederea implementarii suportului de comunicare si partajare a resurselor unui SD.
- Exemple de *middleware*: Apel la distanta (engl.*Remote Procedure Call* – RPC); Sisteme de comunicare intr-un grup de procese; Common Object Request Broker Architecture – CORBA, propus de *Object Management Group* – OMG; Java Remote Method Invocation – RMI; Distributed Common Object Model – DCOM propus de Microsoft.

Middleware



- Majoritatea nivelelor *middleware* sunt implementate peste protocoalele Internet. Ele trateaza si diferentele intre sistemele de operare si hardware.
- *Middleware-ul* realizeaza *transparenta* = furnizarea unui model de calcul uniform pentru programatorii aplicatiilor distribuite, ascunzand retelei.

Partajarea resurselor (I)

- Termenul de resursa este general si abstract.
- Tipuri de resurse:
  - *Hardware* (de ex dispozitive de I/E). Se refera la echipamente care sunt partajate pentru reducerea costurilor.
  - *Software* (fisiere, motoare de cautare, baze de date). Sunt numite si resurse de "nivel inalt" reprezentate prin componente software incapsulate corespunzator. O categorie speciala o reprezinta *resursele informationale*
  - *Memoria si puterea de calcul* pot fi la randul lor considerate resurse din categoria utilitatilor.

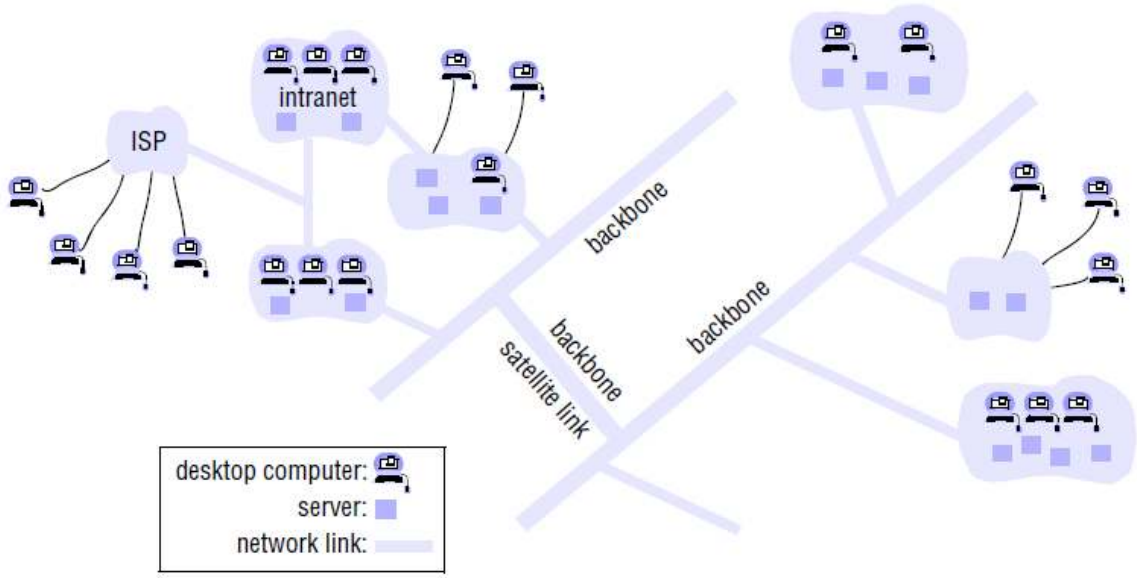
Partajarea resurselor (II)

- Serviciu = parte sau componenta a unui sistem de calcul ce gestioneaza o colectie de resurse corelate si le prezinta functiile catre diverse aplicatii sau utilizatori.
  - Exemple: servicii de fisiere, servicii de imprimare, servicii de plata electronica. Accesul la serviciu se face printr-un set restrans de operatii.
  - Exemplu de functii pentru un serviciu de fisiere: citire, scriere si stergere.
- Restrictia accesului la resurse printr-un set bine-definit de operatii reprezinta o practica standard in ingineria software, dar in acelasi timp corespunde si cu organizarea sistemelor distribuite. Resursele sunt "incapsulate" sau "atasate" in calculatoare si sunt accesibile doar prin subsistemele de comunicatie.
- Terminologia standard foloseste termenii de *server* (furnizor) si *client* (utilizator, consumator). Ei desemneaza rolurile jucate de doua programe (procese) intr-o interactiune de tip cerere-raspuns si corespunde foarte bine unui model obiectual.

Domenii ale aplicatiilor distribuite in retele

Comert si finante	Comert electronic: Amazon.com, eBay Tranzactii financiar-bancare: PayPal
Societatea informationala	Motoare de cautare: Google Enciclopedii online: Wikipedia Retele sociale
Creativitate si divertisment	Jocuri, filme si muzica *online* Continut generat de utilizator: albume foto, impresii turistice, etc.
Asistenta medicala	Monitorizarea si intretinerea sanatatii
Educatie	Invatamant, instructaj si tutorial *online*
Transport si logistica	Sisteme de gasire a rutelor cu GPS Servicii cartografice: Google Maps, Google Earth
Stiinta	Grid-ul ca tehnologie suport pentru colaborare stiintifica intre cercetatori
Mediu	Retele de senzori pentru monitorizarea dezastrelor naturale: poluare, inundatii, cutremure, incendii

Exemplu: Internet

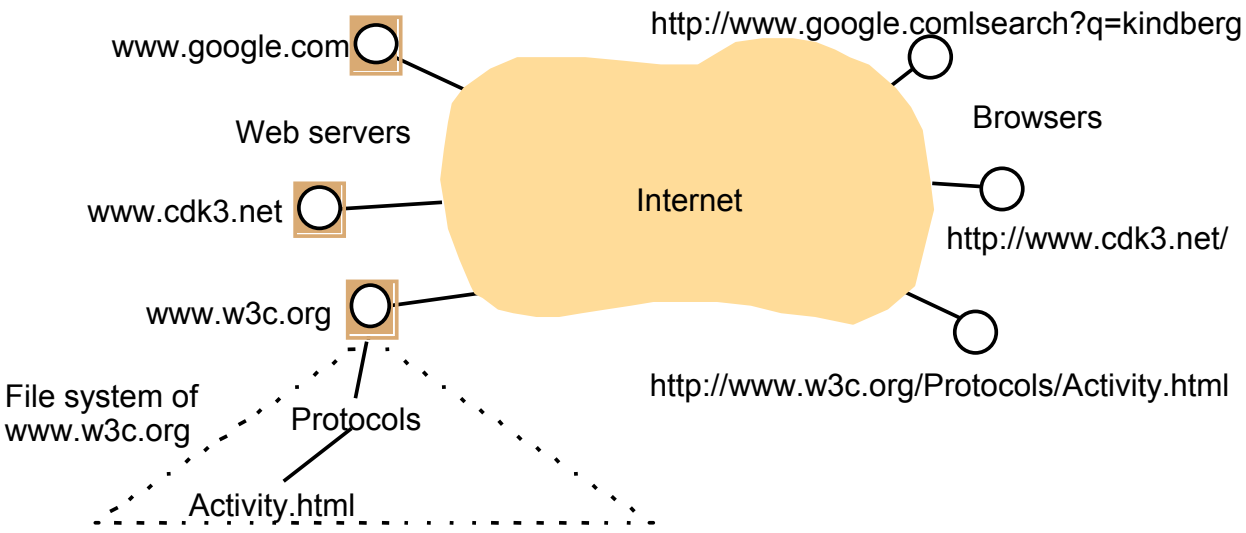


- Internet-ul este o retea de retele. Se bazeaza pe o suita de protocoale standard care permit comunicarea programelor prin mesaje.
- Servicii Internet: WWW, email, transfer de fisiere, fluxuri multimedia.
- Internet este o multime de intranet-uri interconectate. Exemplu: intranet-ul unui ISP.
- Intranet-urile sunt interconectate prin legaturi speciale de mare capacitate (engl.*backbone*). Acestea folosesc tehnologii ca: transmisie prin satelit, fibra optica, circuite speciale de banda larga (*high bandwidth circuits*), etc.

Exemplu: World Wide Web

- WWW este un *sistem hipermedia distribuit*. Se bazeaza pe un model de structurare a documentelor ce foloseste trei concepte:
  - *Multimedia* – se refera la integrarea mai multor tipuri de media in cadrul aceluiasi model de document: text, grafica, imagine, video, etc.
  - *Hiperdocument* – se refera la crearea de legaturi intre documente, folosind un mecanism propriu modelului de document.
  - *Documente distribuite* – se refera la documente care contin legaturi la documente stocate pe alte calculatoare din cadrul unei retele.
- Se spune ca WWW foloseste un *model de documente hipermedia distribuite*. Termenul de hipermedia inglobeaza conceptele de multimedia si hiperdocument.
- Fiecare autor care creaza o resursa informationala (engl.*information resource*) in WWW o considera ca fiind un document separat. Insa, putem considera ca, la nivel global, multimea tuturor resurselor informationale din WWW formeaza un unic document hipermedia distribuit. Din acest punct de vedere, termenul de resursa informationala este mai potrivit decat cel de document.
- Exista trei nivele de distribuire a resurselor informationale in WWW: acelasi fisier, fisiere separate pe acelasi calculator, calculatoare diferite.

Exemplu: portiune din WWW



Scurt istoric al WWW

- WWW a aparut in lumea academica in 1989, la Laboratorul european de fizica particulelor (CERN) pt acces rapid si comod la articolele stiintifice.
- In 1990 proiectul s-a aprobat oficial, fiind denumit World Wide Web si s-a produs si prima implementare. In 1991 sursele prototipului au fost facute publice si totodata sistemul a fost instalat oficial in cadrul CERN.
- In 1992 numarul de servere WWW fiabile a ajuns la 26, incluzand locatii din toata lumea. In 1993 numarul de servere a crescut la 200.
- In 1993 a aparut primul navigator grafic, Mosaic, autor Marc Andresseen din cadrul National Center of Supercomputing Applications (NCSA). Apoi el a creat o firma Netscape.
- In 1994 a aparut consortiul WWW (W3C).
- In 1995 Microsoft a produs Internet Explorer declansandu-se astfel razboiul navigatoarelor WWW.
- Desi WWW a aparut in 1990, conceptele de baza au ramas aceleasi pana astazi: URL (denumirea unui document), HTTP (regasirea unui document) si HTML (descrierea continutului unui document).

Arhitectura WWW

- Este o arhitectura client/server tipica.
  - Un server WWW are sarcina de a gestiona o multime de documente din cadrul WWW. Aceste documente se numesc si *pagini WWW*.
  - Un client generic de WWW este un program care emite cereri catre un server WWW pentru accesarea paginilor WWW gestionate de acel server. Exemple de clienti sunt:
    - Un navigator WWW care permite regasirea si afisarea paginilor WWW in scopul vizualizarii continutului lor de catre un agent uman.
    - Un program de tip softbot care localizeaza diverse pagini WWW in scopul crearii unui index. Indexul poate fi utilizat ulterior de un motor de cautare.
- Conceptele pe care se bazeaza tehnologia WWW sunt:
  - Schema de denumire a resurselor (engl.*uniform resource locator*) URL
  - Protocolul de transfer al documentelor (engl.*hypertext transfer protocol*) HTTP
  - Limbajul de specificare a continutului paginilor WWW (engl.*hypertext markup language*) HTML

Identificarea resurselor in WWW

- Pentru identificarea resurselor in WWW se foloseste un URL (engl.*Uniform Resource Locator*). Un URL este un identificator simbolic al resursei si este compus din doua parti:
  - Schema, care indica modalitatea folosita pentru denumirea resurselor. Spre exemplu, schema poate fi numele unui proocol: *ftp*, *http*, etc
  - Partea specifica schemei, care indica cum se adreseaza resursa in cadrul schemei respective
- Sintaxa URL este *schema* ":" *parte-specifica-schemei*. Partea specifica schemei are sintaxa "//" [*utilizator* [":" *parola*] "@"] *gazda* [":" *port*] "/" *cale*
  - *utilizator* si *parola* sunt optionale si se aplica doar cu schemele care au sens (de exemplu *ftp*).
  - *gazda* indica numele calculatorului pe care se afla resursa, sau adresa de IP a acestuia.
  - *port* reprezinta numarul portului pe care se face conexiunea. Este optional, deoarece acest numar este predefinit pentru serviciile standard. Pentru HTTP portul predefinit este 80.
  - *cale* reprezinta calea de acces la resursa din cadrul calculatorului specificat. In general este o cale fizica existenta in sistemul de fisiere de pe calculatorul gazda.

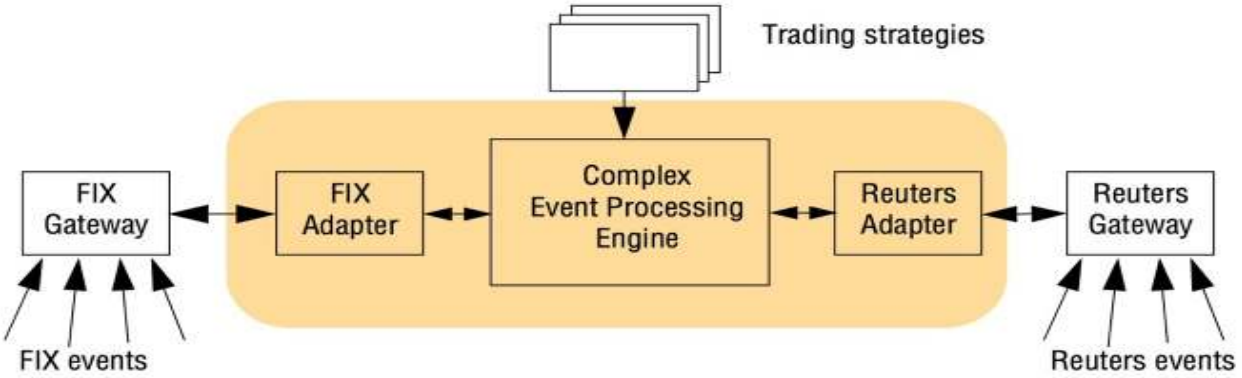
Cautare in WWW

- Nr cautari ~ 10 bilioane/luna
- Nr pagini Web ~ 63 bilioane
- Nr adrese Web unice ~ 1 trilion
- Problema indexarii acestor pagini este complexa
- Infrastructura unui *motor de cautare*:
  - Numar foarte mare de masini fizice interconectate si localizate in centre de date distribuite pe glob
  - Sistem distribuit de gestiune a fisierelor cu functii pentru:
    - Fisiere foarte mari
    - Acces rapid la fisiere
  - Infrastructura de stocare distribuita a unor seturi mari de date
  - Sistem distribuit efeicient pentru controlul accesului
  - Modele de programare distribuita corespunzator bazat pe calcul paralel si distribuit

Jocuri online cu numar foarte mare de utilizatori

- *Massively multiplayer online role-playing games* – MMO(RP)G
- Ofera utilizatorilor o experienta *imersiva*.
- Provocari:
  - *Arena de joc* extrem de complexa (EVE gestioneaza 5000 de “stele”)
  - *Numar foarte mare de jucatori prezenti simultan online* – 50000
  - *Numar total de jucatori* – 500000
  - *Timp de raspuns mic* pentru a nu afecta experienta utilizatorului
  - *Mentinerea consistentei lumii* partajate de utilizatori
- EVE foloseste un server de tip cluster, fiecare sistem solar fiind alocat unei submultimi de calculatoare din cluster
- EverQuest foloseste o partitonare a lumilor pe un server (posibil geografic) distribuit.

Exemplu: sistem de tranzactionare financiara

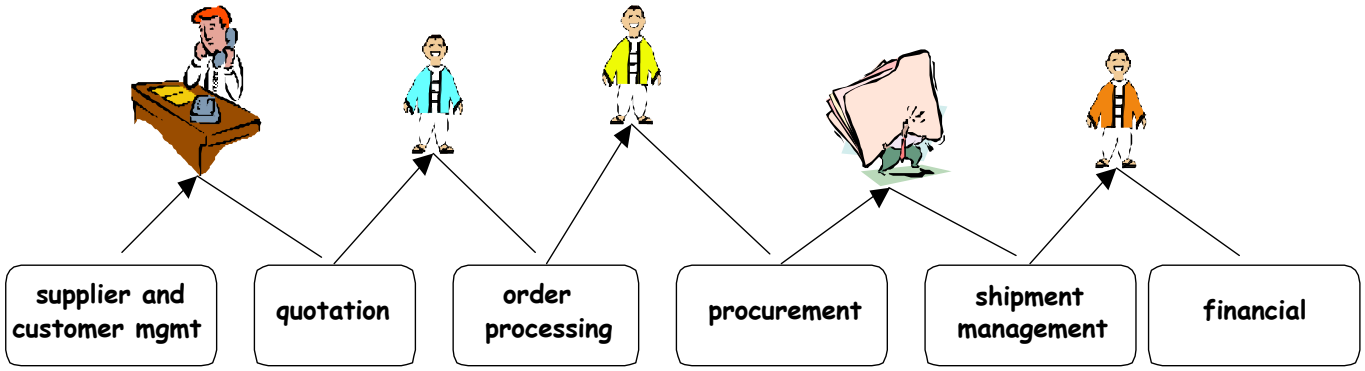


- Ofera accesul in timp real la o gama variata de surse informationale: cotatii bursiere, date de crestere economica, somaj, etc.
- Se cere transmiterea si procesarea eficienta a evenimentelor pentru un numar mare de clienti interesati => SD bazate pe evenimente (engl. *distributed event-based systems*).
- Evenimentele sunt generate din surse (engl. *feeds*) eterogene: Reuters, Financial Information eXchange – FIX, etc.).
- Sistemul trebuie sa proceseze o mare varietate de fluxuri de evenimente pentru detectarea rapida a unor sabloane ce pot indica oportunitati pentru investitii. Pentru aceasta se foloseste tehnologia *Complex Event Processing – CEP*.

Exemplu: sistem B2B (I)

- Pentru fiecare cerere de produse / servicii primita de la un client, o companie de deservire executa o multime de operatii:
  - Ofertare (engl. *quotation*). Presupune procesarea unei cereri de oferta (engl. *RFQ* sau *request for quote*) de la client
  - Prelucrare ordin (engl. *order processing*). In urma ofertei primita de la la companie clientul transmite un ordin de cumparare (engl. *purchase order*). Acesta este verificat si prelucrat, generandu-se un *ordin intern*.
  - Realizarea ordinului (engl. *order fulfilment*). Presupune realizarea ordinului intern si de obicei se parcurg urmatorii pasi:
    - Achizitie (engl. *procurement*). Presupune achizitia componentelor sau ingredientelor necesare de la furnizori si fabricarea (engl. *manufacturing*) produsului.
    - Expeditia (engl. *shipment*). Presupune livrarea produsului catre client.
    - Operatiuni financiare (engl. *financials*). Presupune facturarea clientului si plata furnizorilor.

Exemplu: sistem B2B (II)

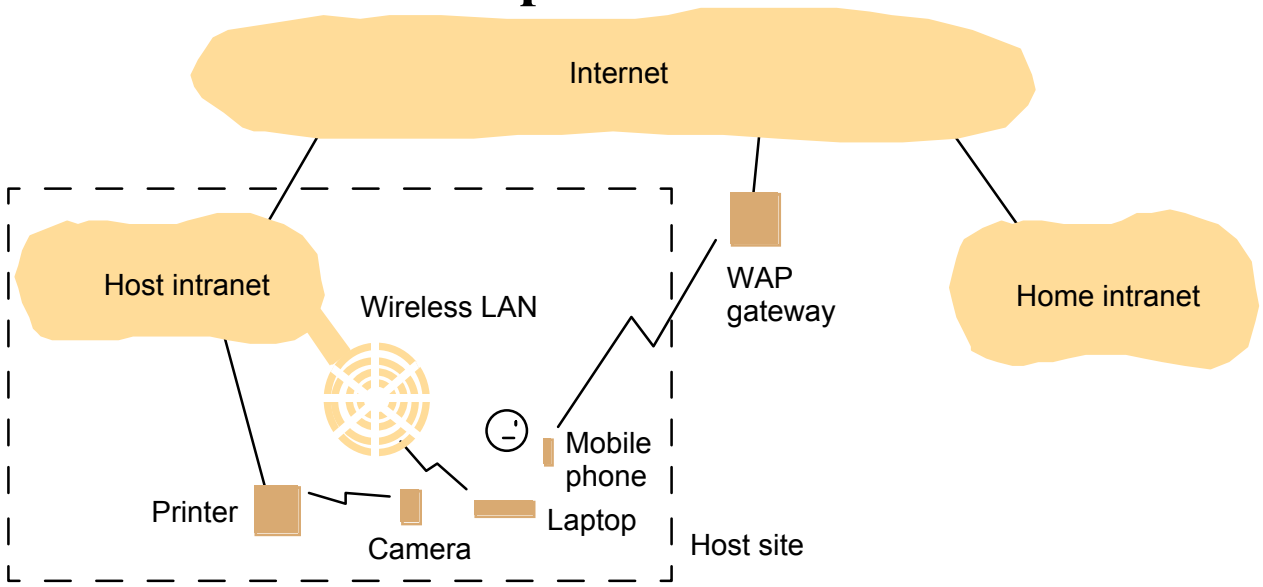


- Pentru automatizarea acestui flux de operatii (engl. *workflow*) sunt necesare tehnologii de integrare a aplicatiilor *Enterprise Application Integration – EAI*.
- Fiecare sistem / aplicatie poate fi parte a unui sistem informatic separat. Aceste sisteme informatice pot fi distribuite geografic si realizate cu tehnologii diferite.

Sisteme de calcul mobile si omniprezente

- Progresul tehnologic in miniaturizarea dispozitivelor, interconectarea fara cablu (engl.*wireless*) si raspandirea universala a Internet-ului (engl. *pervasive Internet*) au facut posibila integrarea dispozitivelor de calcul mici si portabile in SD - *wearable*. Acestea includ:
  - Calculatoare laptop, tablete, dispozitive *handheld*: *personal digital assistant* (PDA), telefoane mobile, pagere, camere video / foto digitale, dispozitove GPS
  - Dispozitive domotice (engl.*appliances*): masini de spalat, sisteme hi-fi, camere video
- Portabilitatea dispozitivelor si abilitatea de conectare facila la retea din diverse locatii au condus la *calculul mobil* (CM, engl.*mobile computing*). CM = efectuarea de sarcini de calcul in care utilizatorul se deplaseaza si viziteaza alte locuri decat mediul obisnuit de lucru, cu posibilitatea de a accesa resurse sau servicii locale prin dispozitivele portabile personale. Exploatarea resurselor din proximitate = *location-aware computing*.
- *Calculul ubicuu* sau *omniprezent* (CU) urmareste exploatarea diverselor dispozitive de calcul (mici si ieftine) din mediul fizic al utilizatorului: camin, birou, zone de cumparaturi, hotel, muzeu, spital, scoala, etc. CM si CU se suprapun, dar aceste concepte sunt in general diferite.

Exemplu: CM si CU



- Utilizatorul are trei forme de acces fara cablu: i) conectare la LAN-ul gazda folosind un laptop. Acesta retea furnizeaza sa zicem o acoperire la nivel de etaj sau cladire si se conecteaza la restul intranet-ului gazda printr-un *gateway*; ii) un telefon mobil cu conectare la Internet printr-un *gateway* folosind protocolul WAP. Astfel utilizatorul are acces la pagini textuale simple ce se pot vizualiza pe ecranul limitat al telefonului mobil; iii) o camera foto digitala care poate comunica prin legatura infra-rosu cu un alt dispozitiv, de exemplu o imprimanta.

2013

Exemplu: UbiLens

- UbiLens – un nou concept de interactiune directa cu un obiect domotic prin intermediul unui telefon mobil.
- Obiectele sunt augmentate cu servicii furnizoare de informatii.
- Localizarea obiectelor se realizeaza cu ajutorul recunoasterii imaginilor capturate cu un telefon mobil. Serviciile sunt furnizate utilizatorului printr-o interfata de realitate augmentata.

2013

Metacalcul, grid & cloud computing (I)

- Metacalcul (engl.*metacomputing*) = utilizarea unor resurse de calcul *disponibile transparent utilizatorului* printr-un mediu de retea. Termenul a fost introdus in 1987 de Larry Smarr.
- Necesitatea sa rezulta din cerintele de calcul potential infinite ale utilizatorilor, in timp ce resursele financiare ale acestora sunt finite.
- Este implementat printr-un *metacalculator*, adica un supercalculator virtual bazat pe o infrastructura de retea.
- Un metacalculator presupune:
  - Integrarea resurselor software si hardware intr-un mediu de retea
  - Implementarea unui *middleware* care sa ofere o imagine transparenta si uniforma a resurselor existente
  - Dezvoltarea si optimizarea aplicatiilor astfel incat acestea sa beneficieze in mod cat mai eficient de aceste resurse.

Metacalcul, grid & cloud computing (II)

- O grila de calcul (engl.*grid computing*) este o infrastructura hardware si software care ofera acces sigur, consistent si transparent la facilitati de calcul performant, in ciuda distributiei geografice atat a resurselor cat si a utilizatorilor.
- Termenul de *grid* a fost introdus de Foster si Kesselman in 1998. Pana atunci s-a folosit termenul de *metacomputing*.
- Grilele de calcul reprezinta o forma de metacalcul inspirata de utilitatile publice (apa, electricitate, gaz, sau telefonie). Se mai foloseste si termenul de utilitate de calcul (engl.*utility computing*). Analogia a fost facuta pentru prima oara in 1960 de John McCarthy – detinator al premiului Turing din 1971.
- Generarea resurselor de calcul este decuplata de utilizarea acestora, la fel ca in cazul utilitatilor. Ele sunt practic inchiriate nefiind necesara achizitia lor permanenta. Se preteaza modelelor *pay as you go* sau *subscription* din telefonia mobila.
- Standarde: Open Grid Forum – OGF: http://www.ogf.org/

Metacalcul, grid & cloud computing (III)

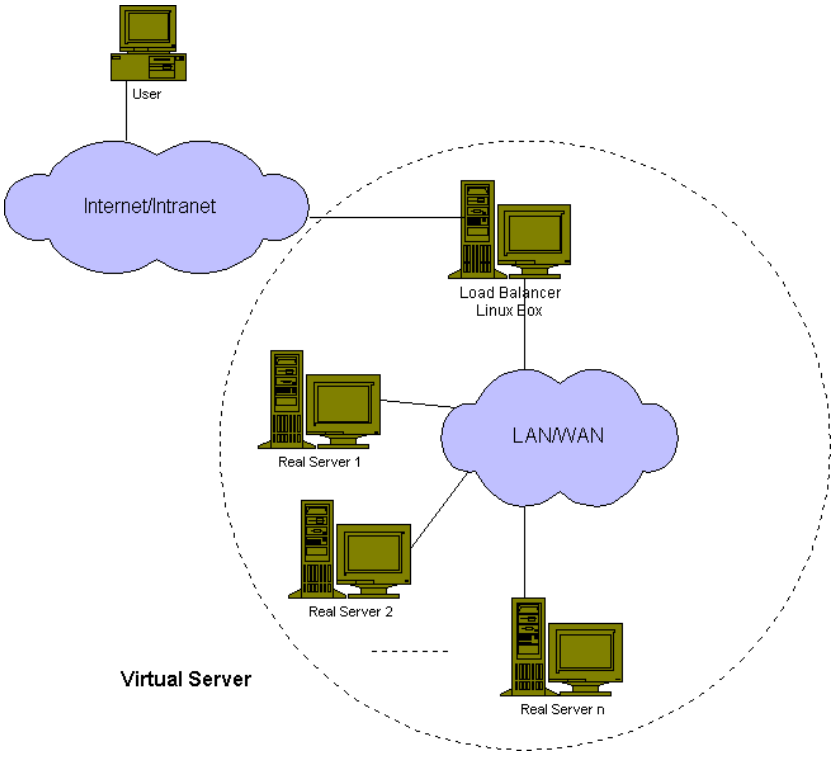
- *Cloud computing* este un model avansat de metacalcul pentru un acces convenabil, la cerere, si de la distanta prin intermediul Internet-ului la o multime configurabila de resurse partajate (retele, servere, memorie externa, aplicatii si servicii) ce pot fi furnizate rapid, cu efort administrativ minim si minima interactiune cu furnizorul (conform NIST: National Institute of Standards and Technology).
- Cloud computing abstractizeaza accesul la:
  - Infrastructura IT => *Infrastructure as a Service*, IaaS
  - Platforma (hardware + sistem de operare) => *Platform as a Service*, PaaS
  - Aplicatii => *Software as a Service*, SaaS

  prin intermediul serviciilor disponibile la cerere, contra cost in functie de volumul de utilizare.

Virtualizare

- *Virtualizarea* se refera la crearea unei versiuni virtuale a unei platforme hardware, sistem de operare, dispozitiv de stocare sau resursa de retea.
- Scopul virtualizarii il reprezinta centralizarea sarcinilor administrative, simultan cu cresterea scalabilitatii si nivelului de folosire a resurselor.
- Tipuri de virtualizare:
  - Hardware: crearea unei masini virtuale care functioneaza ca un calculator real, peste care se poate instala un sistem de operare: un hardware virtual de Mac ce poate rula pe un sistem cu MS Windows
  - Software: sistem de operare, spatiu de lucru sau aplicatie
  - Memorie: agregarea resurselor RAM dintr-o retea
  - Stocare: abstractizarea spatiului logic de stocare independent de spatiul fizic
  - Date: prezentarea datelor la un nivel abstract, independent de un anumit sistem de baze de date
  - Retea: retea virtualizata peste o multime de subretele sau conceptul de *thin client* ce permite separarea *desktop*-ului de un calculator fizic (concept intalnit in *network computers*).

Exemplu de virtualizare: Linux Virtual Server



- http://www.linuxvirtualserver.org
- LVS este un server scalabil pentru un *cluster* de calculatoare.
- Un *cluster* consta dintr-o multime de calculatoare interconectate printr-o retea de mare viteza. De obicei un cluster ruleaza pe langa sistemul de operare, un software special pentru gestiunea eficienta a executiei lucrarilor (engl. *job scheduling*) si echilibrarea incarcarii (engl. *load balancing*).
- Arhitectura unui cluster LVS face ca el sa apara din exterior ca unic server virtual de inalta performanta (scalabilitate si disponibilitate).

2013

Retele de senzori

- O *retea de senzori* (engl.*wireless sensor network*, *WSN*) consta dintr-o multime de senzori distribuiti spatial care coopereaza in vederea monitorizarii conditiilor fizice de mediu, cum ar fi: temperatura, sunet, vibratii, nivel de poluare, presiune, miscare, etc.
- Aplicatii in monitorizare, urmarire, control: armata, monitorizarea proceselor, monitorizarea mediului, controlul traficului, inteligenta ambientala, medicina, detectia incendiilor, etc.
- O retea de senzori este un tip de *retea ad-hoc fara fir* (engl.*wireless ad-hoc network*). Termenul *ad-hoc* inseamna ca ea nu se bazeaza pe o infrastructura preexistenta de rutere si puncte de acces. In schimb, fiecare nod din retea poate participa la rutare prin inaintarea mesajelor catre alte noduri folosind un *protocol de rutare multi-hop*. Aceste noduri se determina dinamic in functie de configuratia si conectivitatea retelei.
- Nodurile WSN cu facilitati de procesare se numesc si *mote* (nume metaforic, *fir de praf*). Un exemplu sunt WSN realizate cu tehnologia Sun SPOT (Sun Small Programmable Object Technology) ce pot rula o versiune speciala JVM http://en.wikipedia.org/wiki/Sun_SPOT.

2013

Provocari ale SD

- Eterogenitate (engl. *heterogeneity*)
- Deschidere / extensibilitate (engl. *openness*)
- Securitate (engl. *security*)
- Scalabilitate (engl. *scalability*)
- Tratarea erorilor (engl. *failure handling*)
- Concurenta (engl. *concurrency*)
- Transparenta (engl. *transparency*)
- Calitatea serviciului (engl. *quality of service*)

Eterogenitate

- Un SD trebuie sa permita rularea de aplicatii si accesul la servicii peste o colectie eterogena de calculatore si retele. Eterogenitatea se refera la: tehnologia retelelor, componente hardware (procesoare), sisteme de operare, limbaje de programare, implementari de la diversi dezvoltatori.
- Spre exemplu, indiferent de tipul de retea, calculatoarele conectate la Internet folosesc aceeasi suita de protocoale.
- Pot apare diferente in ordinea octetilor in reprezentarea datelor pe mai multi octeti (de ex.intregi).
- Desi sistemele de operare includ implementari ale protocoalelor Internet, accesul la ele se poate face prin API-uri diferite de la sistem la sistem.
- Limbajele de programare pot folosi diverse reprezentari ale structurilor de date (de ex,caracterele).
- Componentele software pot comunica intre ele numai prin folosirea unor standarde comune (ex.protocoale de nivel inalt ca HTTP sau reprezentari bazate pe XML).

Deschidere/extensibilitate

- Deschiderea sau extensibilitatea (engl.*openness*) este acea caracteristica a unui sistem care determina masura in care el poate fi extins sau reimplementat in diverse moduri. In cazul SD ea se refera in special la gradul in care se pot adauga noi resurse sau servicii la sistem.
- Nu se poate realiza decat prin publicarea pentru dezvoltatorii de software a specificatiilor si documentatiei interfetelor software esentiale ale componentelor sistemului = "*key interfaces are published*". Acest proces ocoleste procedurile formale de standardizare ce sunt de obicei complicate si decurg foarte incet (deoarece necesita consens).
- Exemple:
  - Protocoalele Internet sunt specificate in documente numite "*Request For Comments*" – RFC pe site-ul organizatiei de coordonare a dezvoltarii Internetului: www.ietf.org.
  - Specificatia CORBA exista intr-o serie de documente pe site-ul OMG www.omg.org.
  - FIPA este un set de standarde IEEE pentru sistemele middleware multi-agent: www.fipa.org
  - Standardele Web: www.w3c.org
- SD deschise pot fi construite din hardware si software eterogen, posibil de la furnizori diferiti.

Securitate

- Resursele informationale din SD au valoare pentru utilizatorii lor. De aceea securitatea lor este foarte importanta. Clientii depind de serviciile contractate, trebuie sa aiba incredere (engl. *trust*) in ele.
- Securitatea resurselor informationale are trei componente:
  - Confidentialitatea: protectia informatiei impotriva dezvaluirii sau divulgarii (engl.*disclosure*) unor persoane neautorizate.
  - Integritatea: protectia informatiei impotriva alterarii sau coruptiei.
  - Disponibilitatea: protectia impotriva interferentei cu mijloacele de acces la resurse.
- Probleme de securitate:
  - *Denial of service attack*: bombardarea unui serviciu cu un numar foarte mare de cereri astfel incat utilizatorii normali sa nu-l mai poata accesa.
  - Securitatea codului mobil: se refera la riscul pe care il presupune executarea de cod receptionat din exterior intr-un mod oarecare, de exemplu prin email, descarcare de pe Web, etc.
- Pentru rezolvarea problemelor de securitate se pot folosi diverse *tehnici criptografice* si *protocoale de securitate*.

Scalabilitate

- SD trebuie sa opereze corect si eficient indiferent de dimensiuni: simple intranet-uri sau intregul Internet.
- SD scalabil = un SD ce ramane efectiv in prezenta unei cresteri semnificative a numarului de resurse, componente si utilizatori.
- Spre exemplu, numarul de calculatoare si servicii din Internet a crescut dramatic.

Date	*Computers*	*Web servers*	*Percentage*
1993, July	1,776,000	130	0.008
1995, July	6,642,000	23,500	0.4
1997, July	19,540,000	1,203,096	6
1999, July	56,218,000	6,598,697	12
2001, July	125,888,197	31,299,592	25
2003, July	~200,000,000	42,298,371	21
2005, July	353,284,187	67,571,581	19

Probleme de scalabilitate

- Controlul cresterii costului resurselor fizice
  - odata cu cresterea cererii pentru o resursa, trebuie sa fie posibil sa se extinda sistemul cu un cost rezonabil astfel incat aceasta cerere sa fie satisfacuta. In general, pentru ca un sistem cu n utilizatori sa fie scalabil trebuie ca resursele fizice necesare deservirii lor sa fie cel mult $O(n)$.
- Controlul pierderilor de performanta
  - se considera o structura de date cu dimensiunea proportionala cu numarul de resurse sau utilizatori (de ex. tabelele necesare sistemului DNS). Performantele pot sa scada datorita incetinirii accesului la aceste tabele. Daca pentru acest acces se folosesc de ex. arbori, accesul va consuma un timp $O(\log n)$.
- Prevenirea epuizarii resurselor software
  - de ex epuizarea numarului de adrese de IP. IPv4 32 de biti => IPv6 128 de biti
- Evitarea congestiilor de performanta (engl. *bottleneck*)
  - spre exemplu ar aparea probleme daca tabela DNS ar fi stocata intr-un singur loc. De aceea ea s-a partitionat si distribuit pe diverse servere din Internet.
- Solutii: replicarea datelor, *caching*, servere multiple, sarcini similare realizate concurent (replicare), algoritmi distribuiti (descentralizati), etc

Tratarea erorilor

- In SD exista posibilitatea defectarii partiale, adica anumite componente se pot defecta, iar altele functioneaza corect.
- Nu trebuie sa existe un unic *point of failure* !
- Obiectiv: mentinerea disponibilitatii (engl. *availability*) SD in prezenta erorilor prin:
  - Detectarea erorilor. Numai anumite defecte pot fi detectate (de ex, detectarea coruptiei datelor prin sume de control). Problema dificila este insa sa se gestioneze si prezenta erorilor ce nu pot fi detectate cu certitudine, ci doar suspectate.
  - Mascarea erorilor. Se refera la ascunderea defectelor care au fost detectate. De exemplu: i) mesajele eronate sau pierdute pot fi retransmise; ii) replicarea permite folosirea unei copii in cazul coruperii originalului.
  - Toleranta la erori. De exemplu in cazul deteriorarii sau indisponibilitatii unui serviciu Web, utilizatorul va fi instiintat de navigator, nefiind lasat sa astepte la infinit.
  - Recuperarea din erori. Presupune proiectarea SD a.i.starile eronate sa poata fi "derulate inapoi" (engl.*rolled back*). Ex.tranzactii.
  - Redundanta. Presupune realizarea tolerantei la defecte prin folosirea unor componente redundante. Exemple: i) existenta a cel putin doua cai intre doua rutere din Internet; ii) orice tabela de nume din DNS este replicata pe cel putin doua servere.

Concurenta

- Accesul la resursele SD se face in mod concurent. In concluzie orice obiect care reprezinta o resursa a SD este responsabil sa asigure o functionare corecta intr-un mediu concurent.
- Acest lucru se aplica atat serverelor cat si obiectelor din cadrul diverselor aplicatii.
- Asigurarea sigurantei unui obiect intr-un mediu concurent se face prin sincronizarea operatiilor de acces la obiectul respectiv. Sincronizarea va asigura mentinerea consistentei obiectului respectiv.
- Pentru aceasta se folosesc tehnici speciale de programare concurenta. Aceste tehnici trebuie uneori sa fie adaptate pentru a functiona intr-un mediu distribuit (de exemplu semafoare distribuite).

Transparenta I

- Transparenta SD reprezinta ascunderea pentru utilizator si programatorul de aplicatii a separatiei dintre diversele componente ale SD a.i. sistemul sa fie perceput ca un intreg, si nu ca o colectie de componente independente.
- S-au identificat 8 forme de transparenta:
  - Transparenta accesului: resursele locale si la distanta pot fi accesate prin operatori identici (ex.obiecte *proxy* cu interfete predefinite)
  - Transparenta locatiei: resursele sunt accesate fara a sti adresa lor (ex.acces prin nume)
  - Transparenta concurentei: procesele concurente pot opera cu resurse partajate fara interferente (corectitudine)
  - Transparenta replicarii: replicarea resurselor va conduce la cresterea performantei si fiabilitatii fara a o dezvalui utilizatorilor sau programatorilor de aplicatii
  - Transparenta erorilor: ascunderea erorilor astfel incat utilizatorii si programatorii de aplicatii sa-si poata realiza sarcinile in ciuda unor eventuale erori ale componentelor
  - Transparenta mobilitatii: permite migrarea resurselor si a clientilor unui sistem fara a afecta operarea altor utilizatori sau programe
  - Transparenta performantei: permite reconfigurarea sistemului astfel incat sa se imbunatateasca performantele in prezenta unor incarcari variabile
  - Transparenta scalarii: permite sistemelor si aplicatiilor sa creasca in dimensiune fara schimbari in structura sau algoritmi

Transparenta II

- Transparenta accesului + transparenta locatiei = transparenta retelei.
- Ex. transparenta accesului:
  - Afisarea continutului unui director de fisiere (locale sau la distanta) intr-un GUI
  - Acelasi API de acces la fisiere locale sau la distanta
- Ex. transparenta locatiei:
  - URL-urile sunt transparente dpdv al locatiei deoarece o parte a lor contine numele unui calculator, nu adresa IP a calculatorului
- Ex. transparenta retelei:
  - Adresele de email: destinatarul este specificat prin nume (nu adresa), iar modul de trsansmitere . Receptie a postei nu depinde de locatia destinatarului.
- Ex. transparenta erorilor:
  - Mesajele de email se retransmit automat in caz de eroare.
  - La fel comunicatia intre procese prin fluxuri TCP/IP mascheaza erorile.
- Ex. transparenta mobilitatii:
  - Comunicarea in telefonia mobila ascunde mobilitatea apelantului si apelatului.

Calitatea serviciului

- Orice functionalitate a SD care este furnizata ca serviciu se caracterizeaza prin proprietati nefunctionale ale calitatii serviciului, cum sunt:
  - Fiabilitate
  - Securitate
  - Disponibilitate
  - Performanta
  - Adaptabilitate
- Performanta
  - Latenta (engl. latency) este legata de reactie (engl. *responsiveness*)
  - Capacitatea (engl.*throughput*)
  - Termene (engl. *deadline*), de exemplu intarzierea maxima intre transmiterea, prelucrarea si afisarea a doua cadre video pentru a asigura afisarea rezonabila pentru utilizator
- QoS = cerinta ca SD ca dispuna si sa ofere resurse de calcul si de comunicatie corespunzatoare ce sa permita aplicatiilor sa-si indeplineasca sarcinile la timp (de exemplu afisarea unui flux video pentru utilizator).