LUENTO 19

Hajautetut oliot, väliohjelmisto, rypäät

Stallings, Luvut 14.2-14.8+
Appendix B

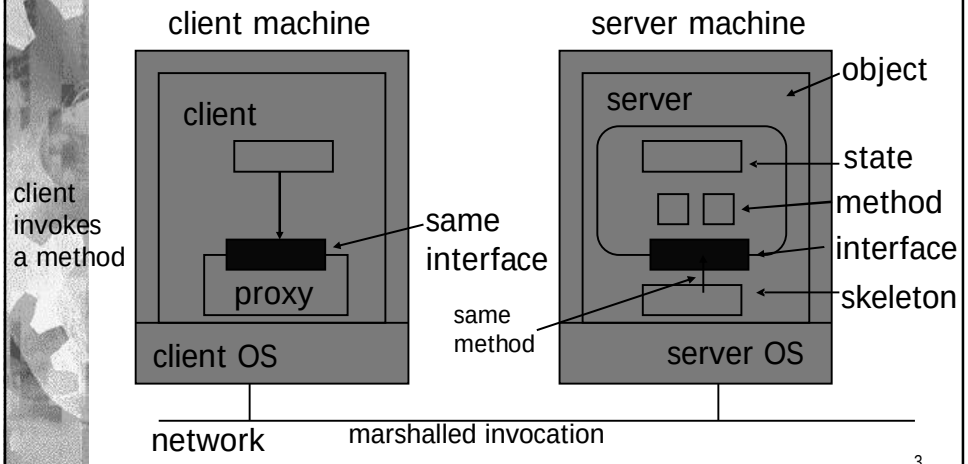
1

Sisältöä luento 19

- Hajautettu olio
- Väliohjelmisto ja CORBA
- Rypäät / Klusterit
  - Windows Cluster ja Sun cluster
  - Beowulf

2

Hajautettuja olioita (Distributed objects)



3

Hajautettu olio

- Käyttö: etäkutsu esim. RMI (Remote Method Invocation) ~ RPC
- Hajautettu rajapinta (distributed interface)
  - Sidonta (binding): haetaan rajapinta (jostain) asiakasohjelmaan => proxy
  - "server stub" ~ skeleton
- Olio itse
  - on yhdessä koneessa (mahdollinen hajautus on piilotettu, sijaintia ei tarvitse tietää)
  - Käyttö kuten aiemminkin olioviitteen avulla
  - Olio voi olla pysyvä (persistent) tai tilapäinen (transient)
- Olioviite
  - Yleensä yksilöivä koko hajautetussa järjestelmässä
- Esimerkkiä olioympäristöistä: CORBA, DCOM, Java RMI

4

Sidonta asiakkaan ja olion välillä

Sidontaa varten asiakkaalla pitää olla jo olioviite!!

- Implisiittinen: käytetään olioviitettä suoraan

```
Distr_object* obj_ref;          //Declare a systemwide object reference
obj_ref = …;                    // Initialize the reference to a distributed object
obj_ref-> do_something();       // Implicitly bind and invoke a method
```

- Eksplisiittinen: Sidonta ensin, käyttö sitten

```
Distr_object objPref;           //Declare a systemwide object reference
Local_object* obj_ptr;          //Declare a pointer to local objects
obj_ref = …;                    //Initialize the reference to a distributed object
obj_ptr = bind(obj_ref);        //Explicitly bind and obtain a pointer to  … the local proxy
obj_ptr -> do_something();      //Invoke a method on the local proxy
```

5

Parametrien välitys

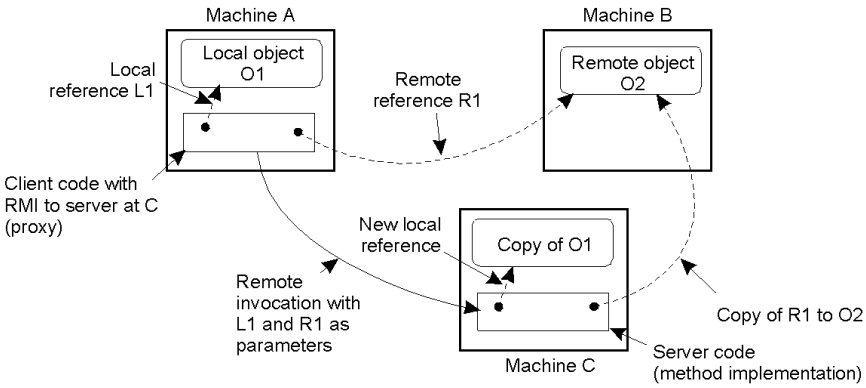


Fig. 2-18.The situation when passing an object by reference or by value.

Copying must not be hidden! ***Why?***

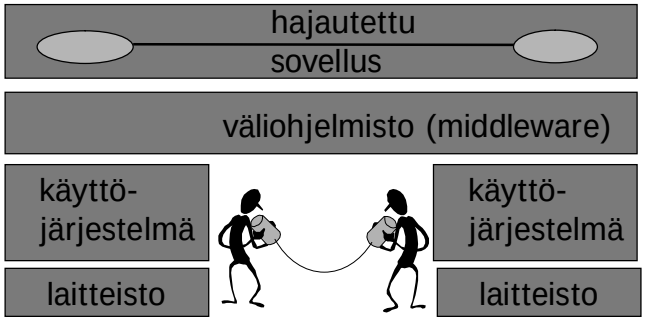
6

Väliohjelmisto (middleware)

7

Mitä väliohjelmistolla tarkoitetaan?

Vrt. kuva 14.9.



Lea Kutvonen: väliohjelmistot kalvosarjasta 8

Väliohjelmisto (Middleware)

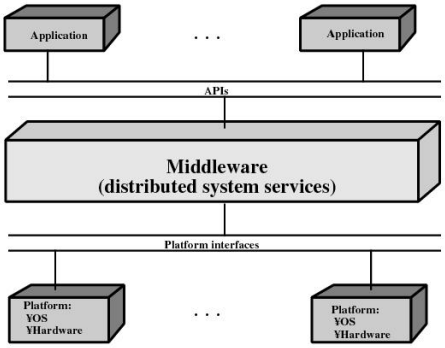
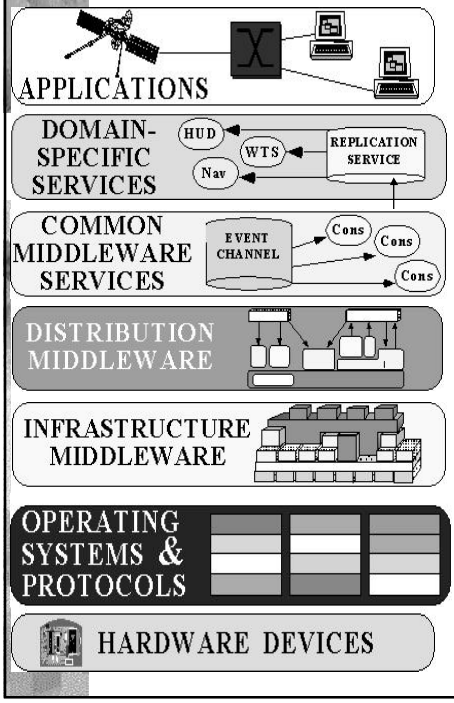


Figure 14.9 Logical View of Middleware

- Tavoitteet:
  - Tarjoaa yhtenäisen rajapinnan sovellukselle.
  - Piilottaa laitteistojen ja KJ:n heterogeenisyyden.
  - Huolehtii tietoliikenteestä ja jaetuista resursseista.

9



Väliohjelmistotasot

•sovellusalueen palveluja: lennon navigointialgoritmeja, potilastietokantamalleja

•yleispalveluja: ilmoitukset, turvallisuus, transaktiot, kuormantasaus, tietovirrat, vikasietoisuus

•objektien ja komponenttien välinen kommunikointi (RMI, CORBA)

•yhtenäinen näkemys käyttöjärjestelmä- ja kommunikointipalveluihin

CACM 45, 6 pp 45

10

Väliohjelmisto

- Väliohjelmiston tarjoamia toimintoja

  RMI, group communication, notification, replication, …

  (Sun RPC, CORBA, Java RMI, Microsoft DCOM, ...)
- Väliohjelmiston tarjoamia palveluja

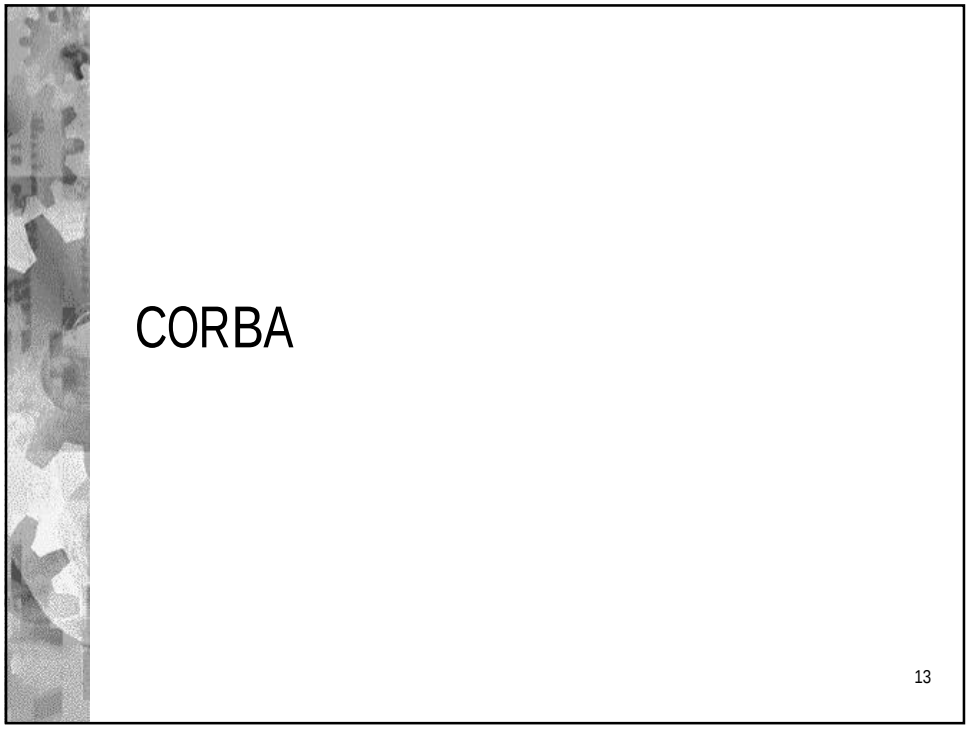
  naming, security, transactions, persistent storage, …
- Rajoitukset
  - Yleinen rajapinta ei aina tue kaikkia sovellusten erikoisia tarpeita.

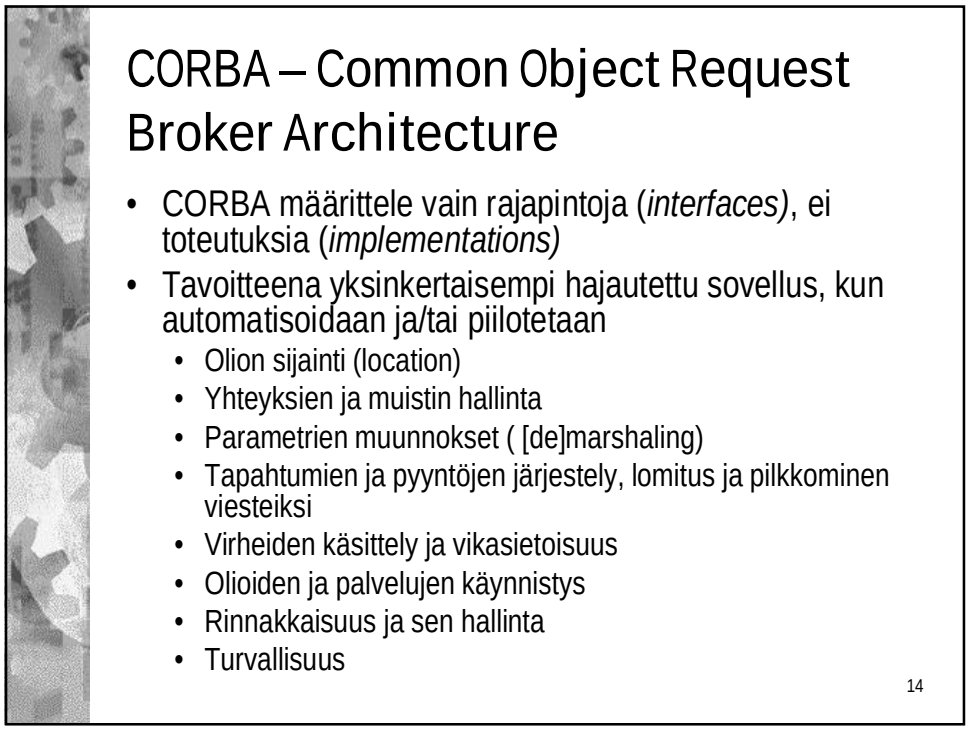
11

Väliohjelmisto

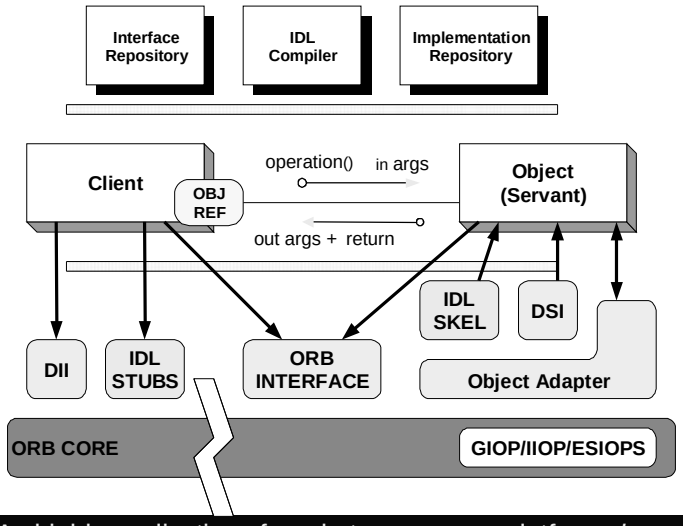
- Joukko työkaluja, jotka tarjoavat yhtenäisen rajapinnan järjestelmän kaikkien eri alustoilla olevien resurssien käyttöön.
- Työkalujen avulla ohjelmoija voi kirjoittaa ohjelmia, jotka eri alustoillakin näyttävät ja tuntuvat samalta.
- Ohjelmoija käyttää väliohjelmiston rajapintaa, jolloin kaikki viittaukset tietoihin ovat samanlaisia riippumatta tiedon sijainnista tai alustan tarjoamista rajapinnoista.

12


CORBA
13


CORBA – Common Object Request Broker Architecture
• CORBA määrittele vain rajapintoja (*interfaces)*, ei toteutuksia (*implementations)*
• Tavoitteena yksinkertaisempi hajautettu sovellus, kun automatisoidaan ja/tai piilotetaan
• Olion sijainti (location)
• Yhteyksien ja muistin hallinta
• Parametrien muunnokset ([de]marshaling)
• Tapahtumien ja pyyntöjen järjestely, lomitus ja pilkkominen viesteiksi
• Virheiden käsittely ja vikasietoisuus
• Olioiden ja palvelujen käynnistys
• Rinnakkaisuus ja sen hallinta
• Turvallisuus
14

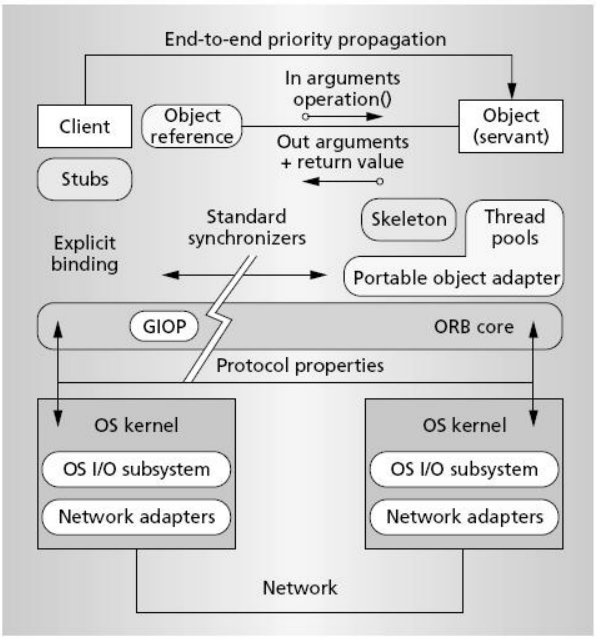
CORBA yleiskuva



- CORBA shields applications from heterogeneous platform *dependencies*
  - *e.g.*, languages, operating systems, networking protocols, hardware

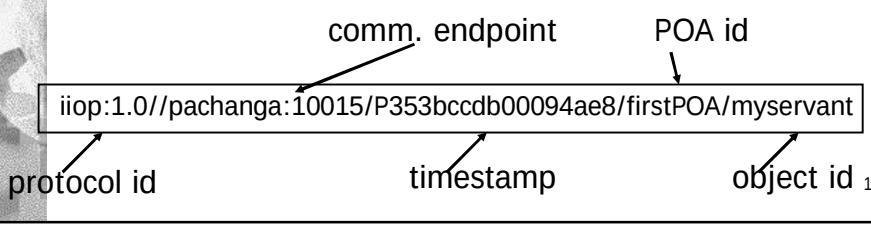
RT-CORBA

- Corba on siis kerros sovellusten ja KJ:n välissä.
- RT-ominaisuus tuo lisää prioriteetit ja eksplisiittisen sidonnan (älä välitä niistä tällä kurssilla)



CORBAn käsitteitä

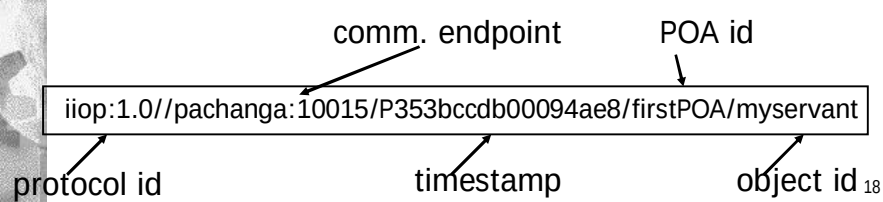
- Asiakas (Client)
  - Ohjelma, joka tekee operaatiopyyntöjä oliototeutuksille; ideaalitilanteessa pyynnöt ovat kuin ko. kielen metodikutsut yleensäkin
- Olio (Object)
  - CORBA:n entiteetti (entity), jolla on identiteetti (identity) ja toteutus (implementation), kutsutaan myös termillä servant



17

CORBAn käsitteitä

- Servant (koodipalanen, metodin toteutus)
  - An implementation programming language entity that defines the operations that support a CORBA IDL interface.
- Olioviite (Object reference)
  - Muuttumaton, läpinäkyvä viite olion instanssiin



18

CORBAn käsitteitä

- ORB rajapinta (interface) ja ORB ydin (core)
  - Abstrakti rajapintakuvaus, kommunikointistandardit, apupalveluja
    - Olioviitteiden käsittely
    - Sidonnat yleisiin palveluihin (esim. nimipalvelu)
    - Geneeriset kommunikointiprimitiivit
  - Piilottaa olion sijainnin, toteutuksen, tilan ja kommunikointimekanismit
  - IIOP, GIOP
    - Kommunikointiin Corba ympäristöjen välillä
    - request, reply, locaterequest, locatereply, cancelrequest, closeconnection, messageerror, fragment

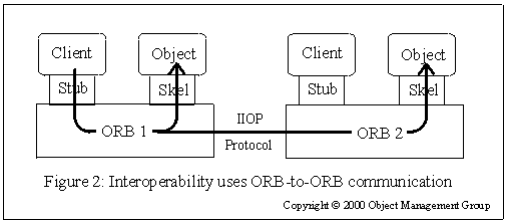
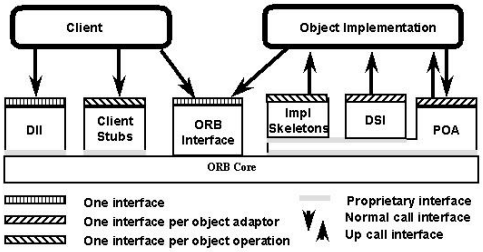


Figure 2: Interoperability uses ORB-to-ORB communication

19

CORBAn käsitteitä

- Stub (tynkä) ja skeleton (ranka)
  - Välittäjiä (proxies), jotka pakkaavat (marshal) ja purkavat (unmarshal) viestit kuljetettavaksi ORB:n kautta
  - ORB rajapinta on geneerinen tynkä.
- DII (dynamic invocation interface) ja DSI (dynamic skeleton interface)
  - Näitä käytetään, jos rajapintakuvaus ei ole käytettävissä käännöksessä



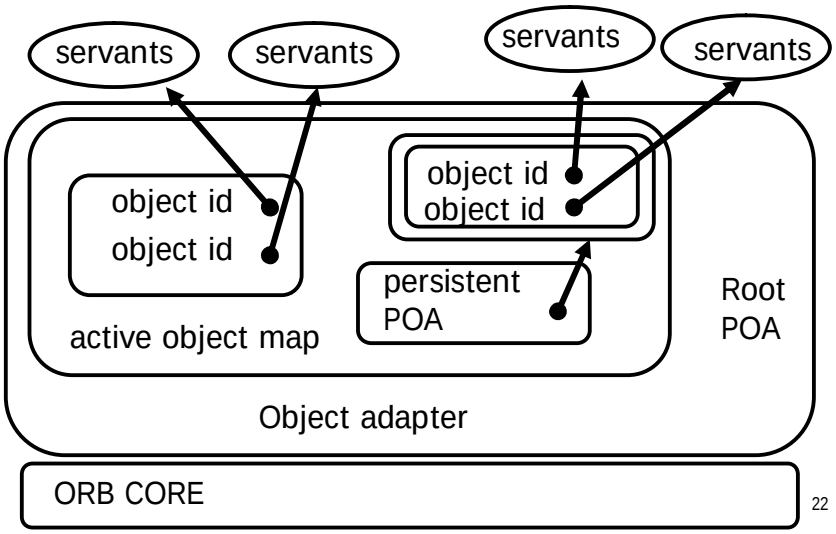
20

CORBAn käsitteitä

- Oliosovitin (object adapter)
  - Hallinnoi yhtä tai useampaa oliota
    - Ei tunne olioiden omia rajapintoja, toimii vain pyyntöjen jakelijana (dispatcher) edelleen oikeille olioille
  - Generoi olioviitteitä, rekisteröi hallitsemiaan olioita
  - Palvelinprosessien ja olioiden aktivointi
  - Oliokutsut (Object upcalls)
  - Eräs tällainen: Portable Object Adapter (POA)

21

Overview of POA architecture



22

CORBAn käsitteitä

- Toteutusvarasto (Implementation repository)
  - Kaikki mitä tarvitaan olioiden toteuttamiseen tai aktivointiin
  - Ei standardoitu, riippuu täysin
    - käyttöjärjestelmästä
    - ORB toimittajasta
    - Oliosovittimesta (object adapter)
  - Määrittää käynnistettävät palvelimet, porttinumerot, suoritettavat tiedostot; tietää myös, jos jo käynnissä ja missä

23

CORBA kommunikointivaihtoehdot

- Oliokutsu (object invocation); vaihtoehdot:
  - Synkroninen (kutsuja odottaa) with at-most-once
  - Yksi suuntainen (one-way) with best effort delivery
  - Viivästetty synkronointi (deferred syncronous) with at-most-once (kutsuja voi edetä, synkronointi vasta vastausviestillä)
- Tapahtumat (Signaling events)
  - Tapahtumapalvelu (event servide)
  - notification service
- Sanomajonot

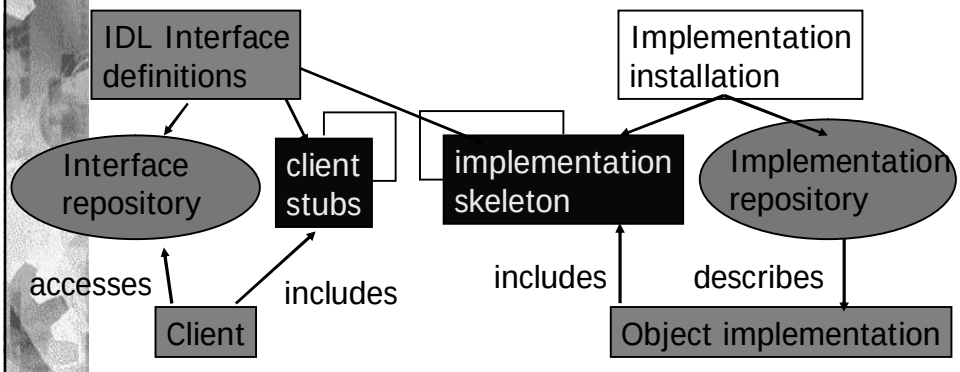
24

Some CORBA concepts - production

- CORBA IDL
  - IDL kääntäjä
  - Kuvaukset eri ohjelmointikielille
- Rajapintavarasto (Interface repository)
  - Tallentaa rajapintakuvaukset
  - relationships – replaceability

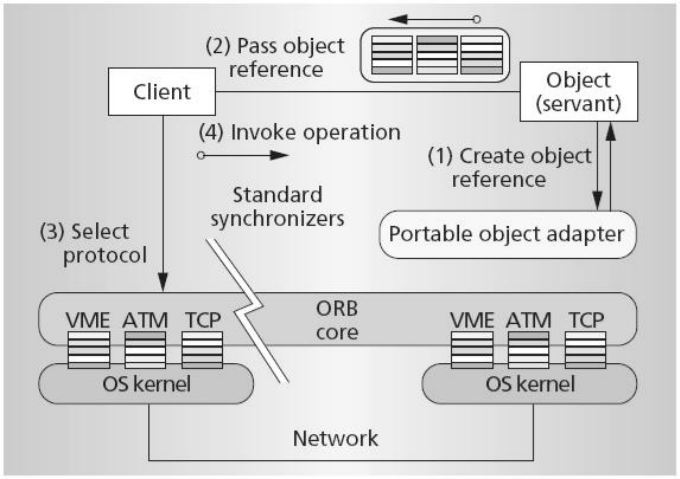
25

Sovelluksen teko



26

Toimintajärjestys



27

Hajautettu CORBA-sovellus: IDL

- IDL

```
module HelloApp{
        interface Hello {
                string sayHello();
        };
};
```

28

Hajautettu CORBA-sovellus: asiakas 1/2

```
Import HelloApp.*;
//contains generated stubs
Import org.omg.Cosnaming.*;
Import org.omg.CORBA*;
//must be in all CORBA applications

Public class HelloClient {
   public static void main(String args[]) {
   try {
            // create and initialize the ORB
            ORB orb = ORB.init(args, null);
            // get the root naming context
            org.omg.CORBA.object objRef =
              = orb.resolve_initial_references("NameService");
            NamingContext ncRef =
              NamingContextHelper.narrow(objRef);
```

29

Hajautettu CORBA-sovellus: asiakas 2/2

```
// Resolve the object reference in naming
   NameComponent nc = new NameComponent ("Hello", "");
   NameComponent path[] = {nc};
   // get the stub
   Hello HelloRef =
            HelloHelper.narrow(ncRef.resolve(path));
   // Call the Hello server object and print results
    String hello = helloRef.sayHello();
    system.out.println(hello);
} catch (Exception e) {
     System.out.println("Error : " + e);
     e.printStachTrace(System.out);
}
}
```

30

Hajautettu CORBA-sovellus: palveluolio 1/3

```
Import HelloApp.*;
// contains generated stubs
Import org.omg.Cosnaming.*;
Import omg.org.cORBA.*;

Public class HelloServer {
   public static void main(String args[]);
   //registration, entering server loop
   try{
      ORB orb = ORB.init(args, null);
```

31

Hajautettu CORBA-sovellus: palveluolio 2/3

```
// create the servant and register with orb
   HelloServant helloRef = new HelloServant();
   orb.connect(helloRef);
//get root naming context
   omg.org.CORBA.object objRef =
      orb.resolve_initial_references("NameService"):
   NamingContext ncRef =
      NamingContextHelper.narrow(objRef);
//bind the object reference to name
   NameComponent nc = new NameComponent("Hello", "");
   NameComponent path[] = {nc};
   ncRef.rebind(path, helloRef);
```

32

Hajautettu CORBA-sovellus: palveluolio 3/3

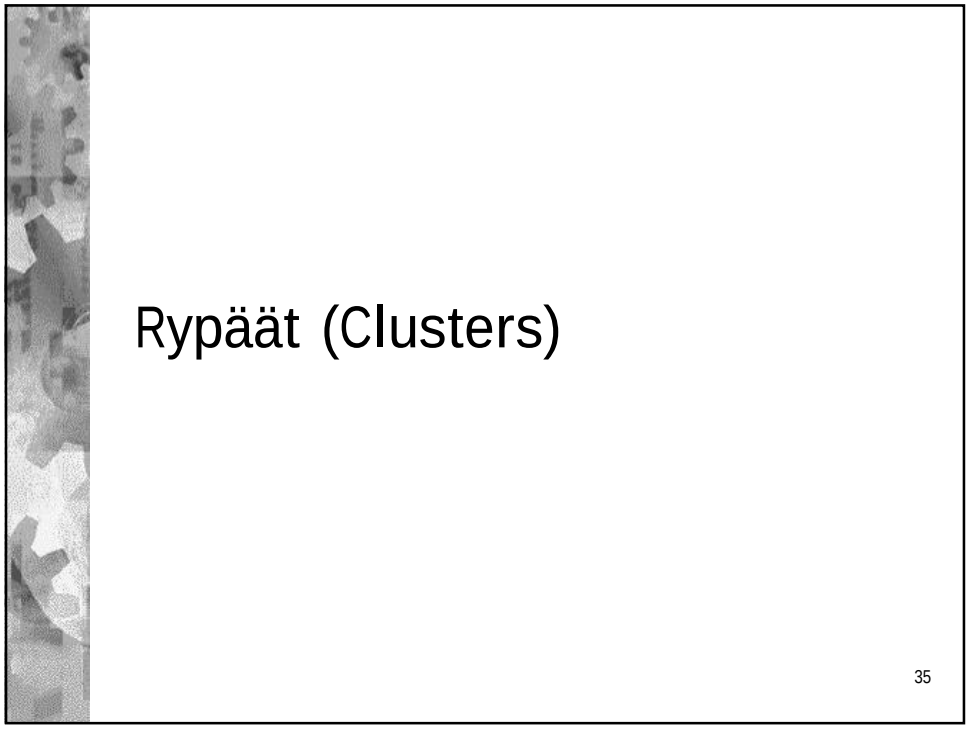
```
//wait for invocations from clients
java.lang.object sync = new java.lang.Object ();
syncronized (sync)  {
    sync(wait);
}
catch (Exection e) {
    system.err.println(”Error: ” + s);
    s.printStackTrace (System.out);}}

//the actual service
Class HelloServant extends _HelloImplBase {
   public string sayHello () {
   return “\nHello world!\n”;}}
```

33

CORBA lähteitä verkossa

- CORBA Tutorial
  - Schmidt, D., CORBA Tutorial. http://www.eng.uci.edu/~schmidt/PDF/corba4.pdf
- CORBA-sivustoja
  - http://www.corba.org/
  - http://www.cs.wustl.edu/~schmidt/corba.html
  - http://www.puder.org/corba/

34


Rypäät (Clusters)
35

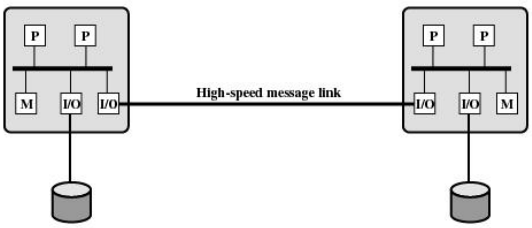

Rypäät / Klusterit
• Vaihtoehto moniprosessorikoneille
• Joukko yhdistettyjä itsenäisiä koneita, joita käytetään yhtenäisenä palveluna
• Yhden koneen illuusio (näyttävät yhdeltä)
• Jokainen solmu on itsenäinen ja voi tarvittaessa toimia ilman muita
• Tavoitteet:
• Laajennettavuus (scalability)
• Saatavuus (high availability)
• Hinta / suorituskyky -suhde tällä hetkellä usein parempi kuin superkoneilla
36

Luokittelua

Clustering Method	Description	Benefits	Limitations
Passive Standby	A secondary server takes over in case of primary server failure.	Easy to implement.	High cost because the secondary server is unavailable for other processing tasks.
Active Secondary:	The secondary server is also used for processing tasks.	Reduced cost because secondary servers can be used for processing.	Increased complexity.
- Separate Servers	Separate servers have their own disks. Data is continuously copied from primary to secondary server.	High availability.	High network and server overhead due to copying operations.
- Servers Connected to Disks	Servers are cabled to the same disks, but each server owns its disks. If one server fails, its disks are taken over by the other server.	Reduced network and server overhead due to elimination of copying operations.	Usually requires disk mirroring or RAID technology to compensate for risk of disk failure.
- Servers Share Disks	Multiple servers simultaneously share access to disks.	Low network and server overhead. Reduced risk of downtime caused by disk failure.	Requires lock manager software. Usually used with disk mirroring or RAID technology.

Erilliset palvelimet (separate servers)

- Itsenäiset, erilliset solmut
- Ei jaettuja levyjä
- Hallinta- ja vuorotus-ohjelmisto
- Tietoa kopioitava solmusta toiseen

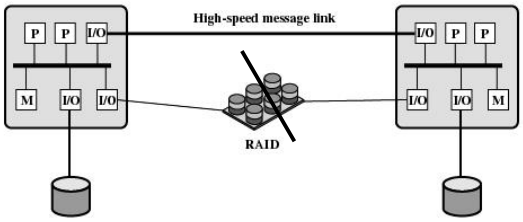


(a) Standby server with no shared disk

38

Jaettu levy, ei yhteiskäyttöä (connected to disk, shared nothing)

- Levyt jaettu osiin (volumes), joka osalla omistajasolmu
- Vain omistajalla pääsy ko. osioon
- Jos solmu kaatuu, omistajaa voidaan vaihtaa
- Tietoa ei tarvitse koko ajan kopioida solmusta toiseen

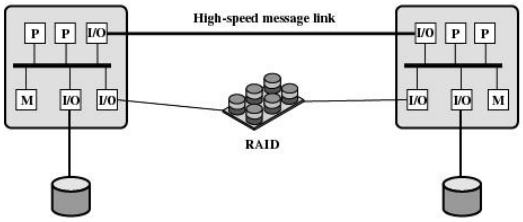


(b) Shared disk

39

Jaettu levy, yhteiskäyttö (shared disk)

- Useilla solmuilla pääsy samoille levyille samanaikaisesti
- Kaikki pääsevät kaikkialle
- Tarvitaan samanaikaisuudenhallintaa ja lukkoja



(b) Shared disk

40

Cluster Computer Architecture

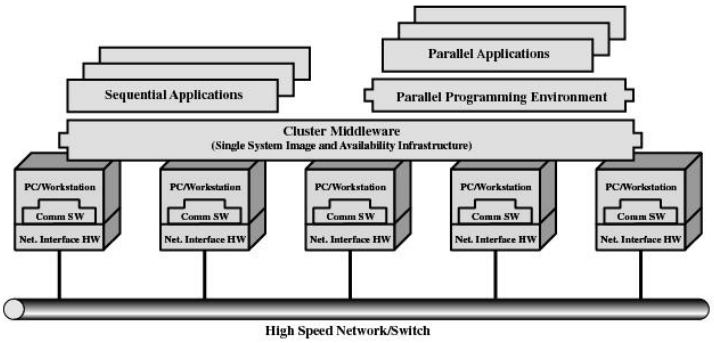


Figure 14.14 Cluster Computer Architecture [BUYY99a]

41

Cluster Middleware

- Single entry point
- Single file hierarchy
- Single control point
- Single virtual networking
- Single memory space
- Single job-management system
- Single user interface
- Single I/O space
- Single process space
- Checkpointing
- Process migration

42

(KJ:n) suunnitteluperiaatteita

- Häiriöiden (failure) hallinta
  - Saatavuuden kasvattaminen: lisätään todennäköisyyttä, että palvelu on saatavilla (*High-availability cluster*)
    - Häiriötilanteessa ei taata eheyttä
  - Vikasietoisuus: taataan, että palvelut ja resurssit ‘aina’ saatavilla (*Fault-tolerant cluster*)
    - Tieto pysyy eheänä
- Kuorman tasaus (Load balancing)
- Laskennan rinnakkaistaminen (Parallelizing)

43

Windows Cluster Service

- Cluster Service
  - Klusterin hallintaohjelmisto kussakin solmussa
- Resource
  - Asiat, joita klusterin hallintaohjelmisto valvoo
- Online
  - Resurssi on ‘Online’ tietyssä solmussa, kun se on ‘providing service’ juuri siinä solmussa
- Group
  - Resurssikokoelma, jota hallitaan yhtenä yksikkönä

44

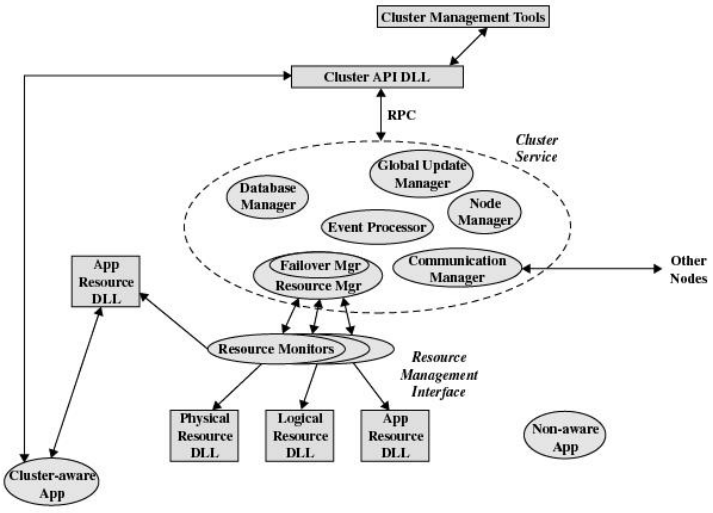


Figure 14.15 Windows Cluster Server Block Diagram [SHOR97]

45

Sun Cluster

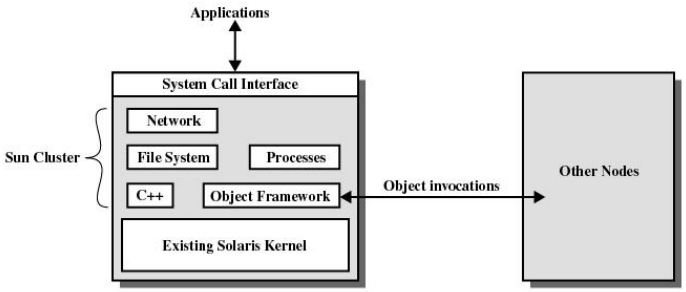


Figure 14.16 Su

- Tuki olioille ja niiden kommunikoinnille
- Prosessien hallinta
- Tietoliikenne
- Hajautettu tiedostojärjestelmä

46

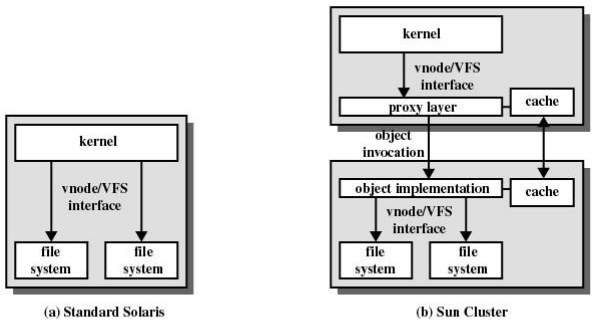


Figure 14.17 Sun Cluster File System Extensions

47

Beowulf

- Beowulf - Vanhin tunnettu englantilainen sankaritaru (suom: Beowulf - herooinen göötti)
- Pääpiirteet
  - www.beowulf.org
  - Kootaan yleisesti saatavista osista (tietokoneista)
  - Prosessorit kuuluvat järjestelmän hallintaan
  - Oma verkko prosessorien väliseen kommunikointiin
  - Easy replication from multiple vendors
  - Tyypillisesti rakennetaan Linuxin päälle

48

Beowulf ja Linux

- Kussakin solmussa
  - Oma Linux ydin, voi toimia myös itsenäisenä
  - Beowulf laajennoksia, mm.
    - Beowulf distributed process space (BPROC) - prosessitunnisteet koko ryvään laajuisina, etäkäynnistys toiseen solmuun
    - Beowulf Ethernet Channel Bonding – kokoaa virtuaalisen nopean verkon useammasta hitaasta todellisesta verkosta
    - Pvmsync – synkronointi ja jaettu data
    - EnFuzion – parametrisoitu laskenta

49

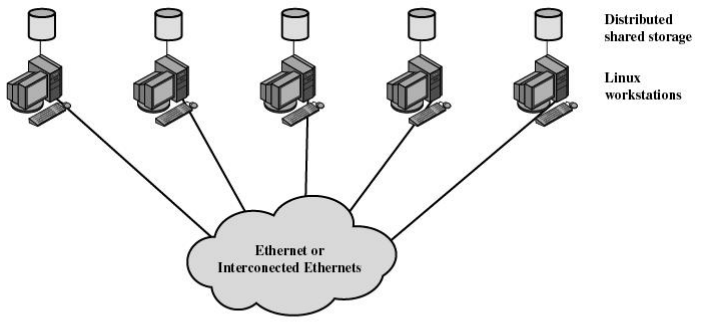


Figure 14.18 Generic Beowulf Configuration

50

Tieteen tietotekniikan keskus - CSC

- Tarjoaa tieteellisen laskennan palveluja: kuten fysiikan ja ilmatieteen mallien laskentaa yms.
- Vastaa korkeakoulujen tietoliikenneyhteyksistä, Funet-verkosta
- Laskentapalvelimia:

  - Louhi: Cray XT4
    - siinä on kuusi login-solmua (SuSe Linux) ja 1012 laskentasolmua (riisuttu Catamount-käyttöjärjestelmä), joista kussakin on kaksiytiminen AMD Opteron –suoritin ja muistia 1 GB/ ydin.
    - Ensimmäisessä vaiheessa Louhen huipputeho on 10,5 Tflop/s.
  - Murska: HP CP4000 BL ProLiant –superklusteri
    - Sen laskentasolmujen määrä on 512, joista kussakin on kaksi kaksiytimistä 2,6 GHz AMD Opteron –suoritinta. Murskan teoreettinen huipputeho on 10,6 Tflop/s ja todellinen laskentateho Linpack-testin mukaan 8,2 Tflop/s.
  - Corona: Sun Fire 25K (corona.csc.fi)
    - 2 kpl Sun Fire 25K –palvelinta, 96 UltraSPARC IV -prosessoria ja 384 gigatavua muistia. Coronan teoreettinen huipputeho on 460,8 Gflop/s.

  Sepeli: HP ProLiant DL145 klusteri (sepeli.csc.fi)
    - 256 kplHP ProLiant DL145 (128) ja DL145G2 (128) -laskentasolmusta ja HP ProLiant DL585 -edustasolmu. AMD Opteron-suorittimia laskentasolmuissa on yhteensä 512. Sepelin laskentasolmujen yhteinen teoreettinen huipputeho on siten 3,4 teraflop/s. Sen todellinen laskentateho mitattuna alalla yleisesti käytetyn Linpack-testin mukaan on noin 1990 Gflop/s.

51

CSC:n GRID-palvelut

- Tutkijalle grid on infrastruktuuri, virtuaalinen työympäristö, joka mahdollistaa hajallaan sijaitsevien tieto- ja laskentaresurssien käytön esimerkiksi Internetin kautta. Grid yhdistää erilaisia laskentaresursseja, tietovarantoja ja mittalaitteita, joita voidaan koordinoidusti, integroidusti ja joustavasti käyttää yhteisinä voimavaroina.
- Grid tarjoaa: organisaatiorajat ylittävästi resurssien, kuten laskentaprosessorien, tietovarastojen ja -aineistojen, verkkoyhteyksien, ja sovelluspalveluiden yhdistäminen, jakaminen ja hallinnointi
- Lisätietoja: http://www.csc.fi/tutkimus/Laskentapalvelut/gridymparisto

52