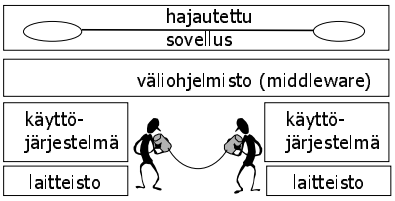
JOHDANTO

Johdanto - agenda

- Mitä väliohjelmistolla tarkoitetaan?
- Väliohjelmistojen rooli ja tehtävät
- Väliohjelmiston haasteet ja syntymekanismit
- Väliohjelmistojen kehitysvaiheet – haasteet ja trendit
- Väliohjelmistojen kategorisointia

Mitä väliohjelmistolla tarkoitetaan?



Middleware definition 1

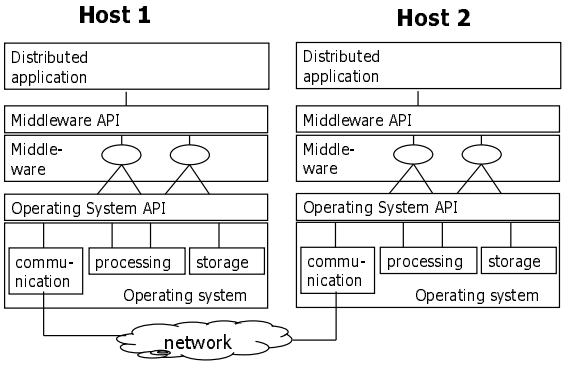
Software that is used to move information from one program to other program in a distributed environment, shielding the developer from dependencies on communication protocols, operating systems, and hardware platforms.

Campbell et.al.

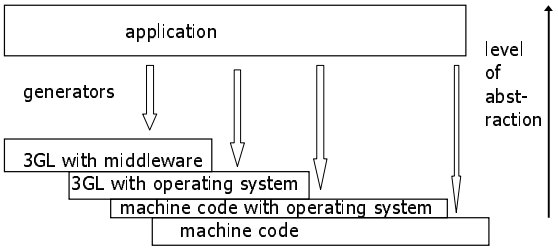
Middleware definition 2

Middleware is a class of software technologies designed to help manage the complexity and heterogeneity inherent in distributed systems. It is defined as a layer of software above the operating system but below the application program that provides a common programming abstraction across a distributed system.

Bakken 2001



Väliohjelmiston tehtävä -- abstraktiokuilun kurominen

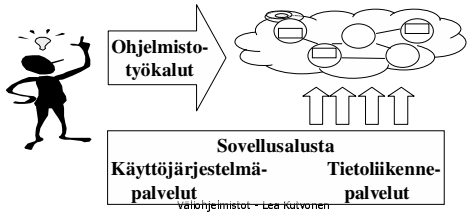


Väliohjelmiston rooli ja tehtävä

- Määritelmät kaksijakoisia
  - tarjoaa ohjelmoijalle käsitteet
  - tarjoaa mekanismin tiedon siirtoon
- Sovelluskehityksessä ohjelmointikieli & sovelluskehitin, ... tarjoaa abstraktit/deklaratiiviset, voimakkaat kommunikointiprimitiivit
  - Ohjelmointikäsitteet: objektit, komponentit, RPC, transaktio, operaatiokutsu
  - Kuvauskielet:IDL, UML; kuvauksista generoitavissa osittaisia toteutuksia
  - frameworks
- Suoritusaikana
  - käsitetasoa kohottava palvelukerros käyttöjärjestelmän ja tietoliikennepalvelujen päällä, kutsuttavaksi suorituskelpoisista sovelluksohjelmista
  - infrastruktuuripalvelut (vrt. globaali nimipalvelu yms)
  - application servers, ...

Väliohjelmistoihin siirtyy

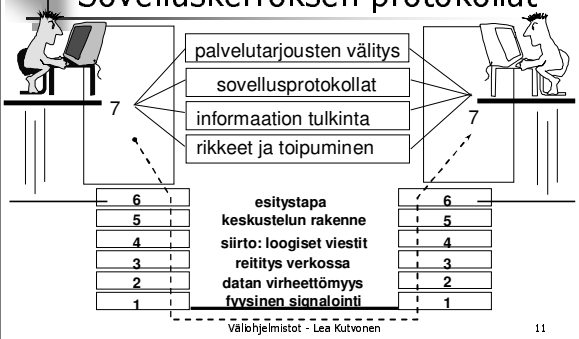
- toistuvia malleja sovelluksista, ohjelmointikehyksistä
- Yhteensulautuvia malleja väliohjelmistoista



Väliohjelmistoihin siirtyy

- Käyttöjärjestelmä- ja tietoliikennepalveluja
  - Esitystapa- ja istuntokerrokset pois sovelluksista
  - Teknisen suorittamisen välineet: säikeet, aktivointi ja passivointi
- Prosessointiin liittyviä lisäpalveluja
  - Skaalautuvuustuki, esim kuormantasaus
  - Joustavuus, vikasietoisuus, siirtyvyys
  - Heterogeenisuus: laitteisto (little/big-endian), käyttöjärjestelmät, ohjelmointikielet
- Sovellustason toistuvia protokollia

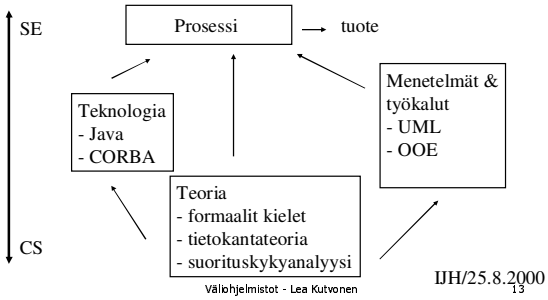
Sovelluskerroksen protokollat



Väliohjelmiston rakentaminen

- väliohjelmistoteknologian käyttökelpoisuuteen vaikuttaa käytetty tuotantoprosessi ja prosessia tukevat menetelmät
- kaikki kolme oltava teoreettisesti eheällä pohjalla
- väliohjelmistot ja niitä tukevat välineet rakennetaan primitiivisemmin, eri paradigmaa noudattavin välinein
- Sykli ideasta tuettuun prosessiin > 10 vuotta

Työkalut, sovellusalustat, sovellukset



IJH/25.8.2000

Työkalut, sovellusalustat, sovellukset

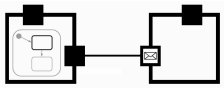
- Toteutus- ja suoritusaikaiset maailmat eivät erillisiä vaan tukevat toisiaan
- esimerkiksi nykyisissä adaptiivisissa sovelluksissa
  - Tavoite: Adaptiiviset sovellukset muuntuvat suoritusaikana
  - Edellytys: Komponenttien kyky tulla korvatuksi toisilla
  - Ohjelmointikäsite: Myöhäinen sidonta (late binding)
  - Infrastruktuuripalvelu: Palvelujen paikantaminen ominaisuuksien perusteella (meklaus)

Arkkitehtuurimallit virittämässä väliohjelmiston tukemia käsitteitä

- Architectural style - architectural *design pattern*
- system = *components* & *connectors*
  - Component: client, server, filters, layers, databases
  - Connector: procedure call, event broadcast, database, protocols, pipes
- some styles
  - Dataflow (Pipes and filters, Batch sequential)
  - Data-centered (Repository)
  - Virtual Machine (Interpreter, Rule-based system)
  - Call and Return (Main program and subroutine, Object-oriented, Layered)
  - Independent Components (Communicating processes, Client/server, Event systems with implicit or explicit invocation)

Arkkitehtuurimallit virittämässä väliohjelmiston tukemia käsitteitä

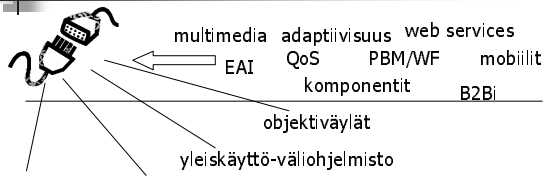
- Example: Event-based, implicit invocation
- Component: (active) object, capsule, module, ...
  - Can be an instance of a class, an active class, or simply a module (non-OO).
  - Have at their interface, methods and ports.
- Connector: "connector", channel, binding, callback.
  - Offers one-to-one, one-to-many, many-to-one connections;
  - Asynchronous event broadcast.
  - (Synchronous event post & await reply => call-and-return.)



Väliohjelmiston valinnasta

- Mikä on sovellusalueen kannalta luonteva arkkitehtuurityyli?
- Millainen kommunikointisemantiikka tarvitaan? Synkronisuus, transaktionaalisuus, tietovirta, monilähetys, globaali tietovarasto, portaali-taustajärjestelmä, ...
- Mitä järjestelmäpalveluita kommunikointisemantiikan tukemiseen on tarjolla? Entä komponentin pysyvyyden, rinnakkaisuuden, tilallisuuden yms tukemiseen?
- Mitä ohjelmointikielitason ilmaisuja voidaan käyttää?
- Mitä lisäpalveluita kääntäjistä, kirjastoista, kehittimistä, olio- tai komponenttikehyksistä on saatavissa?

Väliohjelmistojen kehitysvaiheet



Väliohjelmisto-termin synty

- termi syntyi 1980-luvun lopulla
- vielä 1990-luvun puolivälissä
  - väliohjelmisto-termi liitettiin etupäässä relaatiotietokantoihin
  - nykyiset väliohjelmistot kulkivat nimillä network operating systems, distributed operating systems, distributed computing environments
- merkittäviä järjestelmiä
  - Cronus, Clouds, Eden
  - RPC by Birel &Nelson -84, ONC (SUN), NCS (Apollo), DCE (X/Open)

Väliohjelmistojen alkuajat

- Hajautetun käyttöjärjestelmän tehtävänä tuottaa illuusio yhdestä tietokoneesta
  - Hajautettu prosessien hallinta
    - Prosessien elinkaari, allokointi,
    - prosessien välinen kommunikointi ja synkronointi
    - RPC, viestit
  - Hajautettu resurssien hallinta
    - resurssien varaus, suojaus
    - lukkiumien havaitseminen
  - Hajautetut palvelut
    - hajautettu tiedostojärjestelmä, muisti
    - hierarkkinen globaali nimentä

Haasteita ja ratkaisutapoja

- Haasteita järjestelmäarkkitehtuurille
  - Yrityksen tietojärjestelmän integrointi
- Haasteita järjestelmien toteuttajille
  - Ohjelmiston siirto uuteen järjestelmään
- Ratkaisumalli:
  - Standardoidut ohjelmointirajapinnat
    - Ohjelmointikielen tuettava peruspalveluita (kutsut jne)
    - Kielen ulkopuolella järjestelmärajapinnat (tietokannat, protokollien käyttö, jne)
  - Standardoidut protokollat

Väliohjelmistot vakiintuvat

- 1990-luvun loppupuolella väliohjelmisto erottuu muista ratkaisuista
  - yleiskäyttöinen (vrt. sovellusaluekohtainen framework)
  - tarjoaa hajautettuja palveluja
  - eri laitteisto-, kj- ja verkkoympäristöihin toteutettu
  - tukee standardoituja siirtoprotokollia ja tarjoaa standardi-API:n
- merkittäviä järjestelmiä
  - DCE, COM, CORBA

Haasteita ja ratkaisutapoja

- Haasteita järjestelmäarkkitehtuurille ja järjestelmien toteuttajille
  - Erillisten sovellusten yhteistoiminta yrityksen sisällä
  - asiakasportaalit
- Ratkaisumalli:
  - Standardoidut muodot tiedon esitystavalle
  - Standardoidut väliohjelmistorajapinnat
  - Standardoidut tavat esitellä sovelluksia käytettäväksi
  - Sovelluspalvelimet (application servers), "hotellit"

Perinteiset väliohjelmistot

- Perusmalleja
  - Tietokantojen hajautus, tapahtumamonitorit
  - Hajautettu kj tai verkkokj
  - Objektiväylät, objektikielten toteutus
- Sovellusalueita
  - Pankkijärjestelmät
  - Sairaalan tietojärjestelmä
  - Verkkopalvelut ja dokumenttien siirto

Väliohjelmistot nykyisin

- Vaativammat sovellusalueet
  - Multimediajärjestelmät
    - Tietovirtojen siirtäminen (streams) jatkuvaa valvontaa vaativaa
  - Reaaliaikajärjestelmät
    - Reaaliaikaskedulointi, sopimukset, resurssienvaraus
  - Telecom
    - Vikasietoisuus, järjestelmien suoritusaikainen päivittäminen
  - Sähköinen liiketoiminta
    - Järjestelmien globaalisuus
    - Turvallisuusnäkökohdat avoimessa verkossa
  - Adaptiiviset järjestelmät
    - Personoitavissa, paikkatietoiset, laitteistoon sopeutuvat, ubiquitos computing

Väliohjelmistot nykyisin

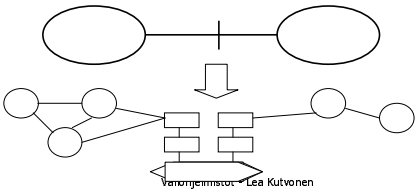
- Nykyaikainen väliohjelmisto tarjoaa standardoidut välineet käsitellä prosessointiyksiköitä ja niiden välistä vuorovaikutusta
- Aikaisempaa abstraktimpia ja voimakkaampia ohjelmointikäsitteitä
  - objektit ja komponentit
  - palvelun kuvaus ja toteuttaja erotetaan
  - metadatan käyttö
  - sijainti- ja pääsytuntumattomuus, muut tuntumattomuuspalvelut

Väliohjelmistot nykyisin

- Objektit ja komponentit
  - Loogiset vs. fyysiset
  - Miten kuvautuvat lopulta suorittaville prosesseille? Granulariteetti?
  - Kommunikointimuodot, rajapintojen lkm? Mallin odottama kommunikointituki?
  - Myöhäinen sidonta
  - Miten loogista objektia vastaavaa fyysistä konfiguraatiota hallitaan?

Väliohjelmistot nykyisin

- palvelun kuvaus ja toteuttaja erotetaan
  - Tarpeita sekä suoritus- että toteutusaikana
- Kommunikoinnin laatu -> eriytetyt tuntumattomuuspalvelut



Väliohjelmistot nykyisin

- metadatan käyttö
  - Nimet yms. paikantamistavat
  - Palvelun laatu –sopimukset (QoS jne)
  - Informaation rakenne
  - Rajapinta-, komponentti- ja palvelukuvaukset
  - Ohjelmistoarkkitehtuuri, konfiguraatiot

Haasteita ja ratkaisutapoja

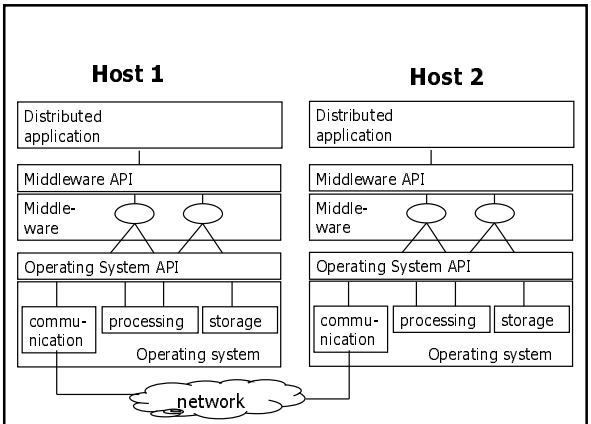
- Haasteita järjestelmien toteuttajille
  - Laajan ohjelmiston tuottaminen tai koostaminen
- Ratkaisumalli:
  - Informaation käsittely: data+metadata
  - Standardoitu, adaptiivinen väliohjelmistorajapinta
  - Ilmaisuvoimaiset välineet kuvata palveluja
  - Toteutusten osittainen generointi palvelukuvauksista

Väliohjelmistojen tulevaisuus

- Haasteita järjestelmäarkkitehtuureille
  - Organisaatiorajat ylittävä palvelujen integrointi
- Järjestelmien toteuttajille
  - Laajan ohjelmiston koostaminen
  - Palvelumarkkinoiden toteuttaminen
- Ratkaisumalli:
  - Standardoitujen metainformaatiopalveluiden käyttö
  - Globaali arkkitehtuurimalli?
    - Erilaiset arkkitehtuurityylit ja komponenttimallit yhteensovitettavissa?

Väliohjelmistotyyppejä

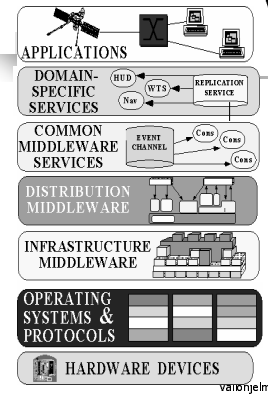
- Tapahtumaorientoitu vo. (Transactional mw)
  - IBM CICS, BEA Tuxedo, Transarc Encina
- Viestiperustainen vo. (Message oriented middleware, MOM)
  - IBM MQSeries, Sun Java Message Queue
- Proseduuripohjainen vo. (Procedural mw)
  - RPC kaikkine variaatioineen
- Objekti- ja komponenttipohjainen vo.
  - CORBA, DCOM, Java RMI, EJB, SOAP, .NET



Väliohjelmistotyyppejä

- eri tyypit ottavat eri aspektit huomioon
  - distributed tuples: communication, storage, limited in processing
  - MOM: communication, no processing, limited storage
  - RPC: communication, processing, no storage
  - distributed objects: communication, processing, and storage

Väliohjelmistotasot



- **sovellusalueen palveluja**: lennon navigointialgoritmeja, potilastietokantamalleja
- **yleispalveluja**: ilmoitukset, turvallisuus, transaktiot, kuormantasaus, tietovirrat, vikasietoisuus
- objektien ja komponenttien välinen **kommunikointi** (RMI, CORBA)
- **yhtenäinen** näkemys käyttöjärjestemä- ja kommunikointipalveluihin