

Väliohjelmistojen tyypit

Tyypit

- Tapahtumaorientoitu
- Viestiperustainen
- Proseduuripohjainen
- Objekti- ja komponenttipohjainen

Teemat

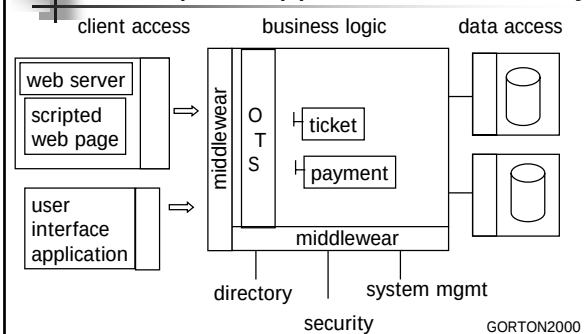
- Arkkitehtuurimallit
- Palvelut
- Sisärakenteet
- Laajennustrendit
- Realisaatioita

Tapahtumaorientoidut väliohjelmistot

Teemat

- Järjestelmämallin tarpeet
- Palvelut ja rajapinnat
- Sisärakenteet
 - networking and interoperability
 - server control
 - system administration infrastructure
- Sovellustilanteen arkkitehtuurimallikkeita
- Realisaatioita
- Laajennustrendit

Enterprise application – anatomy



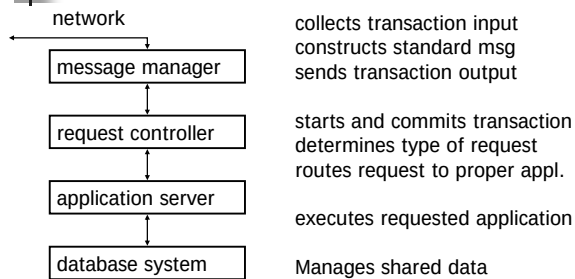
Enterprise application – anatomy

- requirements
 - high availability (99,9-99.999 %)
 - 24 * 7 * 365, downtime 8 h – 5 min /year
 - data integrity and consistency
 - scalability
 - security
- applications
 - customer billing
 - airport management system
 - loan processing system

Enterprise application – anatomy

- services
 - middlewear – message transport, "plumbing"
 - OTS = object transaction service
 - security service for access control and data encryption
 - directory service for locating services and applications in the enterprise system
 - management service: auditing, logging, configuration, alert handling

Traditional mainframe processing



BERNSTEIN198x?

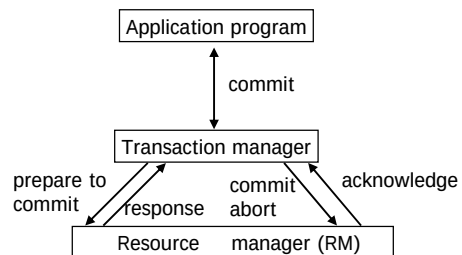
Anatomia

- tietokantaan kohdistuvia kyselyjä ja pyyntöjä
 - stabiili, ennustettava, korkea kuorma
 - muutampia sovellustyyppisiä
 - reaaliaikaisuus, dataintensiivisyys
 - transaktioiden perusominaisuudet ACID
- Services**
 - synchronous, asynchronous, store-and-forward, conversational communication
 - homogeneous cross-platform naming
 - scalability/high performance
 - high availability (recovery, rerouting)
 - workload management by priority management
 - access control

Palvelut: Tapahtumamonitori

- Transaction processing monitor
- Tukee ACID-ominaisuuksia
 - atomicity – kokonaan tai ei lainkaan
 - consistency – tieto vain hyvinmääritellyissä tiloissa
 - isolation – välitilanteet eivät näkyvissä, lukitus
 - durability – tapahtumien vaikutusten pysyvyys
- Hoidetaan
 - toipuvilla prosesseilla (recoverable processes)
 - 2PC – kaksivaiheinen sitoutumisprotokolla
- Palveluaspektit
 - sovelluskehitys
 - suoritusympäristö
 - hallintarajapinnat
 - yhteydet erilaisiin tietokantoihin

Kaksivaiheinen sitoutumisprotokolla Two-phase commit



Rajapinta?

- TM**
 - begin
 - commit
 - rollback
 - open
 - close
 - info
 - set-commit-return
 - set-transaction-timeout
- RM**
 - start
 - end
 - prepare
 - commit
 - rollback
 - recover

API

```

interface Current{
  void begin();
  raises(SubtrUnavail);
  void commit (in boolean
    report_heuristics);
  raises (notrans,
    heuristicmixed,
    heuristicchard);
  void rollback();
  raises (notrans);
  Status get_status();
  string get_transaction_name();
  void set_timeout (in long secs);
  Control get_control();
  control suspend();
  void resume (in Control which);
  raises (invalidcontrol);
};
  
```

"Pitkät" transaktiot

- tilanteissa joissa
 - käytetään verkon takana olevia hitaasti saatavia resursseja
 - suoritetaan loppukäsittelyä (tulostus tms)
- ei halua/tarvetta odottaa transaktion todellista päättymistä
 - ero esim pitkiin CAD-istuntoihin, joissa peruutusriski
 - ACID-ominaisuudet osittain vaarassa kun prosessointia jätetään suoritettavaksi transaktion ulkopuolella
- menettely
 - transaktio kirjoittaa varmistettuun jonoon tulokset kuten tietokantaan ikään
 - transaktio päättyy (commit), tulokset jonossa
 - erillinen sovellus käy purkamassa jonon (toivon mukaan transaktionaalisesti jonoa käsitellen)
- jononkäsittelyominaisuuksia
 - prioriteettijärjestetyt jonot, klusteroidut jonot kuormantasausta varten
 - non-fifo –käsittely, jonon selailu

Lukitseminen

- lukulukot ja kirjoituslukot
- lukkojen konfliktitaulukot
- lukkojen gradunarteetti -> skaalautuvuus, tiedon eheys
- 2PC-protokolla käsittelee myös lukot

Toipuvat prosessit

- TM ja RM ovat toipuvia: kykenevät palauttamaan oman tilansa häiriön jälkeen
 - TM kerää lokia "transaction log"
 - RM kirjaa transaktion tilan ja datamuutokset, joita transaktio sai aikaan
 - toipumisprosessissa toistetaan lokien tapahtumat, "replay"
 - optimointimahdollisuuksia ...

Suosituksia ja toteutuksia

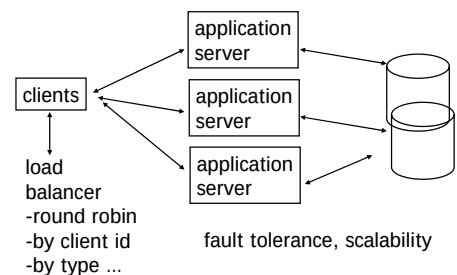
Tuotteita – nopeasti vanhenevaa asiaa

- OTMS
 - BEA WebLogic Enterprise
 - IBM Transarc Encina++
 - Iona OrbixOTM
 - Inprise Visibroker ITS
 - ...
- MTMS
 - samoilta ja miksei muiltakin
 - MOM & MQ
- RPC-varianttejakin ...

Suosituksia ja kehikoita – nämäkin vielä liikkuvia maaleja

- CORBA transaction service
- J2EE transactions
- .NET transactions
- TIP (Transactions over Internet protocols)?

Optimointia: kuormantasaus



Optimointia: tilallinen vai tilaton?

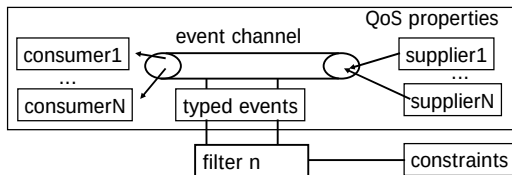
- mid-tier services
- stateless
 - easy to recover, program, scalable
- stateful
 - state not in client
 - clients easy to recover, program etc
 - no persistency services available in mid-tier -> clumsy
 - cache only read-only information

Optimointia: tulosten reititys

- monissa yrityssovelluksissa on lähetettävä tietoa suurelle joukolle vastaanottajia
- dynaamiset tapahtumien valvonnan ja odotuksen tarpeet
- tapahtumailmoituksen lähettäjän ja vastaanottajien ei tarvitse tuntea toisiaan
- julkaise-tilaa-arkkitehtuuri (publish-subscribe)
- vrt. message-queue-malli

CORBA Events & Notifications

- Event namespace (names and attributes)
- Typed events (header+body; fixed + other)
- Consumer event filtering, event batching, event priority, event expiration, logging, internationalization, flow control mechanism

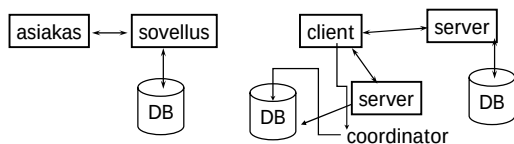


Publish-subscribe

- viestilaatikko yhteinen, kaikki voivat lähettää, tilaajilla suotimet vastaanotettaville viesteille
- taataan että kaikki tilaajat saavat kaikki asiaankuuluvat viestit
- mallit
 - Otsakepohjainen
 - Aiheperustainen
- Ongelmat
 - Skaalautuvuus: viestien vertailu tilausehtoihin
 - Viestien järjestys

Tapahtumaorientoituneet vo: Ongelmat ja laajennustrendit

- ongelmat:
 - heterogeeniset tietomallit
 - usein liian raskas sitoutumisprotokolla (pitkäkestoiset tai omnipotentit tapahtumat)
 - puutteellinen siirtosyntaksin tuki
- trendit
 - nested transactions
 - mukaantulo objektiiväliohtelmistoihin



Viestipohjaiset väliohtelmistot

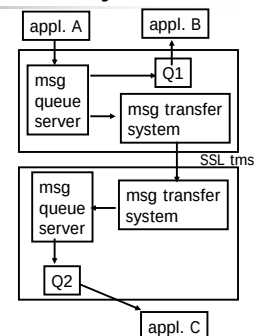
Teemat

- Järjestelmämallin tarpeet
- Palvelut ja rajapinnat
- Sisärakenteet
 - networking and interoperability
 - server control
 - system administration infrastructure
- Sovellustilanteen arkkitehtuurimallikkeita
- Realisaatioita
- Laajennustrendit

Viestipohjaiset väliohtelmistot

Arkkitehtuurimalli

- asynchronous messages
 - reliable, fault-tolerant
 - no loss, duplication, permutation, cluttering
- persistent subscriptions
- models supported
 - message queue
 - request-response
 - multicast
 - publish-subscribe



Viestipohjaiset väliohjelmistot

- MOM = message oriented middleware
- asynchronous messaging natural, synchronous communication cumbersome
- message queues support reliability of message transport
- violates access transparency, no support for data heterogeneity unless in programming language mapping, no support for transactions
- suitable for event notifications, publish/subscribe-based architectures
- persistent message queues support fault tolerance

Viestipohjaiset väliohjelmistot

- Perustila: client-server-malli, putki
 - vrt pistoketason ohjelmointi
- Topics for variation and development
 - persistent/transient msgs
 - FIFO/priority queues
 - translations of msgs
 - abstractions on msg ordering
 - multithreading, automatic load balancing
 - msg routing (source, cost, changes in topology etc)
 - secure transfer of msgs (at least between msg servers)

Viestipohjaiset väliohjelmistot

- Services (MQSeries)
 - 4 types of messages
 - Datagram: one way message, no reply
 - RequestMessage: query initiating message
 - ReplyMessage: response to RequestMessage
 - ReportMessage – "exception report"
 - API: 11 calls
 - Connect/disconnect queue manager, open/close queue, inquire/set (queue) attributes, get/put message from/to queue, put1 (open, put, close), commit/rollback

Viestiperustaiset väliohjelmistot

- Konsortioita
 - MOMA – message oriented middleware association www.moma-inc.org ???
 - BQM – business quality messaging ???
- Välinekuvauksia
 - IBM WebSphere MQ (MQSeries), Sun JMS, Microsoft MSMQ, BEA MessageQ, TIB
 - <http://dsonline.computer.org/middleware/projects/MOM.html>
 - <http://www.ibm.com/websphere/developer/techjournal/>

Rajaavampi määritelmä

- The term "messaging" most often refers to electronic mail systems. MOM message movers, however, differ radically from mail systems, because they are high-speed, generally connectionless and are usually deployed for concurrently executing applications with a non-blocking sender. Message-queuing middleware (MQM) is a type of MOM that combines a high-speed message mover and a queuing (message storage) service.

Korhosen seminaarityö

Proseduraaliset väliohjelmistot

Teemat

- Järjestelmämallin tarpeet
- Palvelut ja rajapinnat
- Sisärakenteet
 - networking and interoperability
 - server control
 - system administration infrastructure
- Sovellustilanteen arkkitehtuurimallikkeita
- Realisaatioita
- Laajennustrendit

RPC-malli

- tavoitteena saantituntumaton proseduurikutsu
- muttei voi täysin täyttää tätä tavoitetta
 - Nimittä, vikaantumiset, suorituskyky paljastavat
 - Globaalit muuttujat, kontekstiriippuvat mjat, osoittimet
 - Call-by-reference vs. call-by-value
- Kutsusemantiikat
 - Maybe, at-least-once, at-most-once
 - Exception delivery
 - Asynchronous RPC
 - Multicast, broadcast
- Muut kutsujen ominaisuudet
 - Location transparency, migration transparency, ...
 - Concurrent processing

Request-Reply protocol

- Issues in marshalling of parameters and results
 - input, output
 - data representation
 - passing pointers
- Distributed object references

Handling failures

- Types of failure
 - client unable to locate server
 - request message lost
 - reply message lost
 - server crashes after receiving a request
 - client crashes after sending a request
- Reporting failures breaks transparency

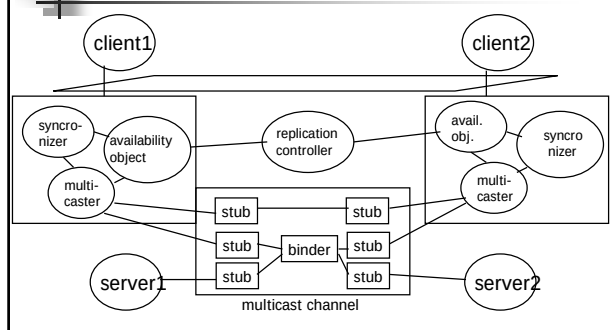
Handling failures

- Lost request message
 - retransmit a fixed number of times before throwing an exception
- Lost reply message
 - client resubmits request
 - server choises
 - re-execute procedure: service should be idempotent
 - filter duplicates: server should hold on to results until acknowledged
- Client crashes
 - orphan detection: reincarnations, expirations

Fault tolerance measures

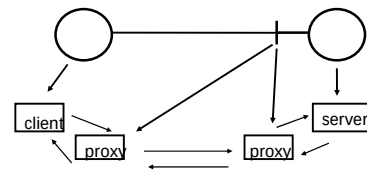
Retransmit request	Duplicate filtering	Re-execute/retransmit	invocation semantics
no	not appl	not appl	maybe
yes	no	re-execute	at-least-once
yes	yes	retransmit reply	at-most-once

"Replication transparent RCP"

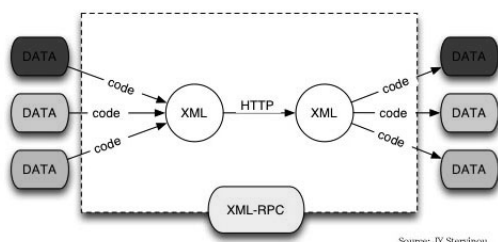


RPC middleware - model

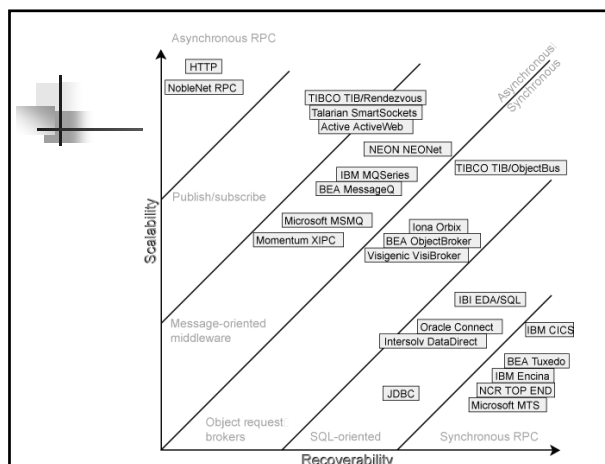
- First class - integrated with language, e.g. Java RMI
- Second class - language-independent IDL, e.g. CORBA



XML-RPC



Source: JV Stensin



RPC:n ongelmia

- Synkroninen vuorovaikutus
 - tiukka sidonta asiakkaan ja palvelijan välillä
 - voi olla odotustilassa pitkään
 - johtaa monisäikeiseen ohjelmointiin
- Hajautustuntumattomuus
 - ei voida peittää kaikkia ongelmia
- Palvelun käsite puuttuu
 - ohjelmoija ei ole kiinnostunut palvelijasta/palvelimesta
- Objektittomuus
 - kutsut funktio/proseduurikutsuja eivätkä metodi/operaatiokutsuja

Proseduraaliset väliohjelmistot

Teemat

- Järjestelmämallin tarpeet
- Palvelut ja rajapinnat
- Sisärakenteet
 - networking and interoperability
 - server control
 - system administration infrastructure
- Sovellustilanteen arkkitehtuurimallikkeita
- Realisaatioita
- Laajennustrendit

Objekti- ja komponentti- väliohejelmistoista

- Arkkitehtuurimalli
 - Hajautetut objektit (distributed objects)
 - Asiakas-palvelin
- Kehityskaari
 - Käyttöjärjestelmät
 - Laitteistoabstraktio, suoritusaikainen ympäristö, peruspalvelut sovelluksille
 - Objektijärjestelmät
 - Komponenttijärjestelmät

Objektijärjestelmät

- Objektialustat
 - Objektimalli
 - osoittaminen, rajapinnat, vuorovaikutustavat, objektien implementointi
 - Objektimallin toteutus ja suoritusaikainen ympäristö
 - Objektien luonti ja elinkaaripalvelut
 - Esimerkkejä: CORBA
 - Vertailu objektikielen ja CORBA-mallin mukaisesta ohjelmointitavasta
 - Schmidt, Vinoski: Object Interconnection. SIGS C++ Report Magazine, May 1995.

Objekti- ja komponenttimallit

- Eroihin johtaneita seikkoja
 - Kommunikointi -- siirrettävyys
 - Toiminnallisuus -- hallittavuus
 - Palvelevuus -- uudelleenkäytettävyys
- Objektimallit keskittyvät operaatioiden käynnistämiseen

"Objects are discrete software units – they contain data, and can manipulate it. There are four keys: encapsulation, inheritance, polymorphism, and instantiation."
- Komponenttimallit kuvaavat komponentin toimintaympäristöä

Tuumaustauko ...

- Onko MOM kuollut?
- Onko malli ehkä näkyvissä uudella tavalla?
 - http ja web? virukset?
 - sulautetut järjestelmät? auton jarrut ja lukitus?
 - miten ydinvoimalan säätö on järjestetty?
- Mitä tiedämme liikuteltavasta tiedosta?
 - MOM vs tapahtumaorient.

