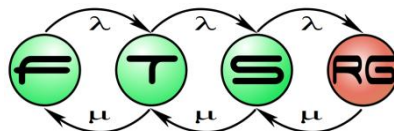


Kommunikációs middleware megoldások

Szolgáltatásintegráció előadás

Gönczy László, Huszerl Gábor (BME MIT)



Kérdések

- Hogyan kommunikálhat két alkalmazás?
 - Szinkron és aszinkron megoldások
 - Ilyet én is tudok írni. Sőt, jobbat is!
- Melyik megoldás hogyan működik?
 - Melyik mire jó?
 - Mi a különbség?
 - Hol találni ilyet?
- Aszinkron megoldások olcsón?

Middleware

- Hol van az a középben?
 - Operációs rendszer felett, alkalmazás alatt
 - Alkalmazásokból „alászálló” funkciók
 - néha tovább az OS-be
- Mit csinálnak?
 - Kommunikáció, HA, UI, skálázás, grafika, játék,
 - A továbbiakban itt mindig kommunikációs MW
- Minek?
 - Alkalmazás integráció, szolgáltatás integráció
 - Komponensközi kommunikáció

Preklasszikus middleware fajták

- Adatcsere az alkalmazások között, ahogy lehet
 - Fájl átvitel
 - Adatbázisok
 - Elektronikus levelezés
 - Weboldalak
 - Sockets, Pipes
 - Házilagos megoldások
 - alkalmazásokból kiemelve

Korszerű middleware technikák

- Házilagos megoldások
 - Házon belüli fejlesztés házon belüli igényekhez
- Távoli eljáráshívás (RPC, ORB) - 1988
 - Szinkron elosztott alkalmazásokhoz
- Üzenetsorok (MQ) -1994
 - Üzenet alapú, nagy megbízhatóságú komm.-hoz
- Publish-Subscribe (P/S) -1998
 - Üzenet alapú, valós idejű kommunikációhoz
- Egyéb aszinkron technikák
 - Üzenet alapú, olcsóbb
 - Rövid érvényességű információhoz

Middleware-ek feladatai

- Kliens-szerver kapcsolat (tág értelemben)
 - Komponensközi kapcsolat, hívások/üzenetek
- Platformfüggetlenség (HW-től és OS-től)
- Hálózatfüggetlenség (hálózati protokolltól is)
- Publikus API (Mennyire publikus?)
- Nyelvfüggetlenség (programozási nyelvtől)
- Adattárolás függetlenség
 - Adatbázis rendszerektől
 - Üzenettárolásnál, jogosultságoknál, címeknél is
- Egységes alkalmazásfejlesztői platform

Extra MW feladatok 1.

- Közös felhasználó azonosítás
 - Egységes felhasználói jogosultságok
 - Egyszeri belépés (single sign on) (különböznek!)
- Tranzakció azonosítás (összetartozó üzenetek)
- Biztonság
 - Titkosított adat- és vezérlési forgalom
 - Komm. titkosítása + üzenetek titkosítása
- Elhelyezkedés-függetlenség
 - Hol fut a másik?
 - Ha „mozog” a másik

Extra MW feladatok 2.

- Adatbázis-orientált szolgáltatások
 - Elosztott lekérdezések/beillesztések/törlések
 - RDBMS szolgáltatások
- Alkalmazás-orientált szolgáltatások
 - Bármilyen, ami ott éppen kell
 - Pl. atomi tranzakciók, óra szinkron, ...
- Menedzsment szolgáltatások
 - MW felügyelete (SNMP, Unicenter, Tivoli, ... ágens)
 - Konfigurációs eszközök
- ... és még sokan mások

Korszerű middleware technikák

- Házilag megoldások
- Távoli eljáráshívás (RPC, ORB)
- Üzenetsorok (MQ)
- Publish-Subscribe (P/S)
- Egyéb aszinkron technikák

Házilagos MW megoldások

- Zöld mezős, házon belüli megoldások
 - Házon belüli célokra (fejlesztendő alkalmazás(család)hoz)
- Fejleszteni kell hozzá
 - Mindenféle fejlesztő és tesztelő eszköz
 - Saját hálózati protokoll
 - Saját API, belső működés, technológiák
- Szakember igény (kereskedelmi termék adoptálásához kevesebb/gyengébb is elég)
 - Rendszermérnök, programozó, tesztelő
 - Hálózati mérnök, hálózati adminisztrátor
 - Projekt menedzser

Házilagos MW megoldások

- A várható igények felmérése (rendszerfejlesztők)
- Meghozandó fejlesztői döntések
 - Kommunikáció (szinkron, aszinkron)
 - Hálózat (TCP, UDP, IPX, ...)
 - Információáramlás (üzenetek/hívások, egyirányú/kétirányú, 1-to-1/1-to-n/n-to-n)
 - Technológiák (C, C++, Java, C#, XML, címzés, ...)
 - Teljesítmény (sávszélesség, késleltetés, kliensek száma, üzenetek száma, ...)
 - Megbízhatóság, biztonság, ...

Házilagos MW megoldások

- Előnyök
 - Jobb testre szabhatóság,
kritikus paraméterei jobbak lehetnek
 - Extrém körülmények között megoldást nyújt
- Hátrányok
 - Fejlesztés költsége, ideje, szakember igénye
 - Folyamatos karbantartás és fejlesztés
 - Támogatás később is csak házon belülről
 - cég erős függése alkalmazottaitól, kivéve ha
→ nyílt forráskódú projekt (ld. Később)

Házilagos MW megoldások

- Tipikus alkalmazási területek
 - Valós idejű kómm. speciális igényekkel
 - pl. TTTech
 - Speciális hálózati protokollok felett
 - kevésbé támogatott technológiákhoz
 - Örökölt, kritikus, nehezen integrálható alkalmazásokhoz
 - ha már a kereskedelmi MW házilagos adaptere sem megoldás
- „Csináld magad!”

Korszerű middleware technikák

- Házilagos megoldások
- Távoli eljáráshívás (RPC, ORB)
- Üzenetsorok (MQ)
- Publish-Subscribe (P/S)
- Egyéb aszinkron technikák

Két rokon middleware technika

- RPC (Remote Procedure Call)
 - Tradicionális progr. techn. mintájára
 - Szabványos eljáráshívás szerint
- ORB (Object Request Broker)
 - OO technológia mintájára
 - Metódushívás „kiterjesztése”
- Mindkettő
 - Request/reply szinkron kommunikáció
 - Hely és platform transzparens
 - interoperabilitás (gép, nyelv, OS között)
 - heterogén elosztott rendszerek
 - skálázhatóság (egy → több gép, kis → nagy gép)

RPC/ORB

- Feladatai
 - Hívás elkapása, hívott fél megkeresése
 - Paraméterek átvitele
 - Szerver eljárásának/metódusának meghívása
 - Eredmény visszajuttatása (vezérlés visszaadása)
- Technológiák
 - RPC: régi, tisztán nem fordul elő már
 - ORB: különböző technológiák
 - CORBA (OMG szabvány)
 - DCOM (eredetileg Windows alá)
 - RMI (csak Java alá)

RPC/ORB

□ Előnyök

- Szabványos technológiák elosztott rendszerekhez
- Alkalmazások központosítható menedzselése
- Tipikusan objektum-orientált megközelítés
- Hagyományos alkalmazások „webesíthetőek”

□ Hátrányok

- Igazán nagyra rosszul skálázódik
- Gyakran szakértőt igénylő architektúrák
- Inkompatibilis ORB implementációk
- Nehéz hibakeresés és adminisztráció

RPC/ORB

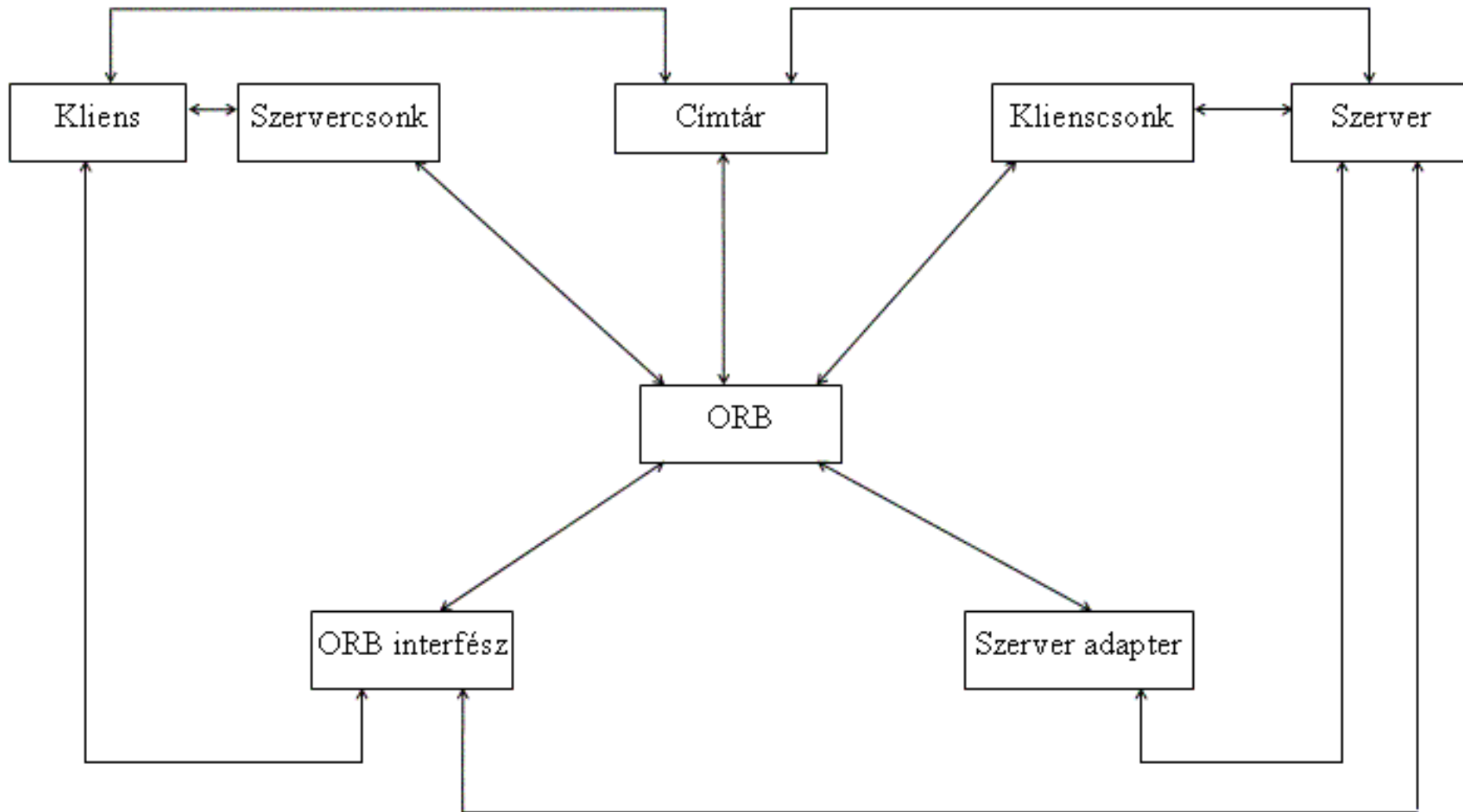
- Tipikus alkalmazási területek
 - Help desk alkalmazások
 - Számla lekérdező rendszer
 - Hagyományos szerverek webes felülete

- Jellemző: A kliens ügyis kénytelen megvárni

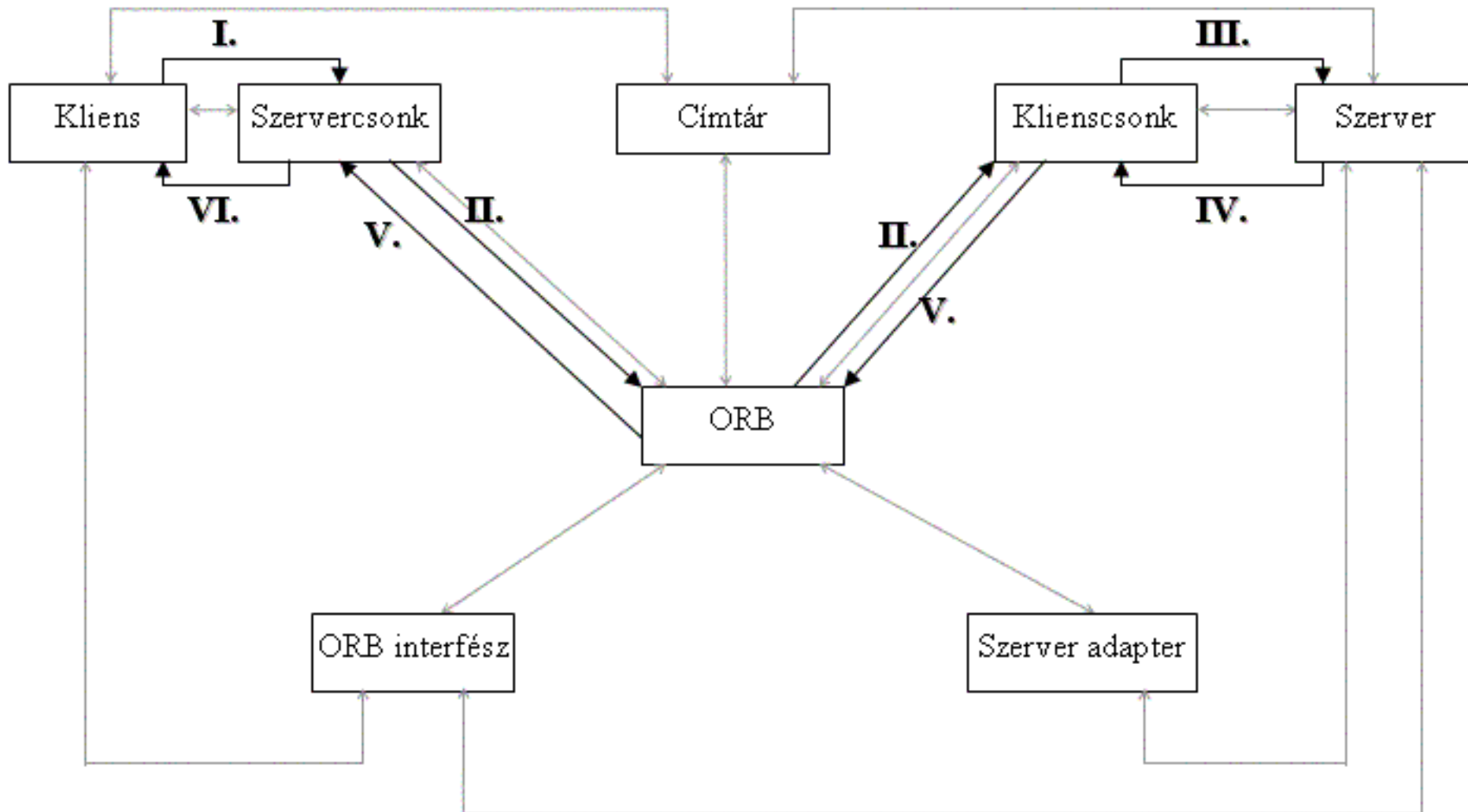
ORB architektúra modellek

- Szabó Péter: Távoli eljáráshívás alapú middleware rendszerek modellezése (Diplomaterv, BME MIT, 2003)
- Általános modell
- CORBA
- DCOM
- RMI

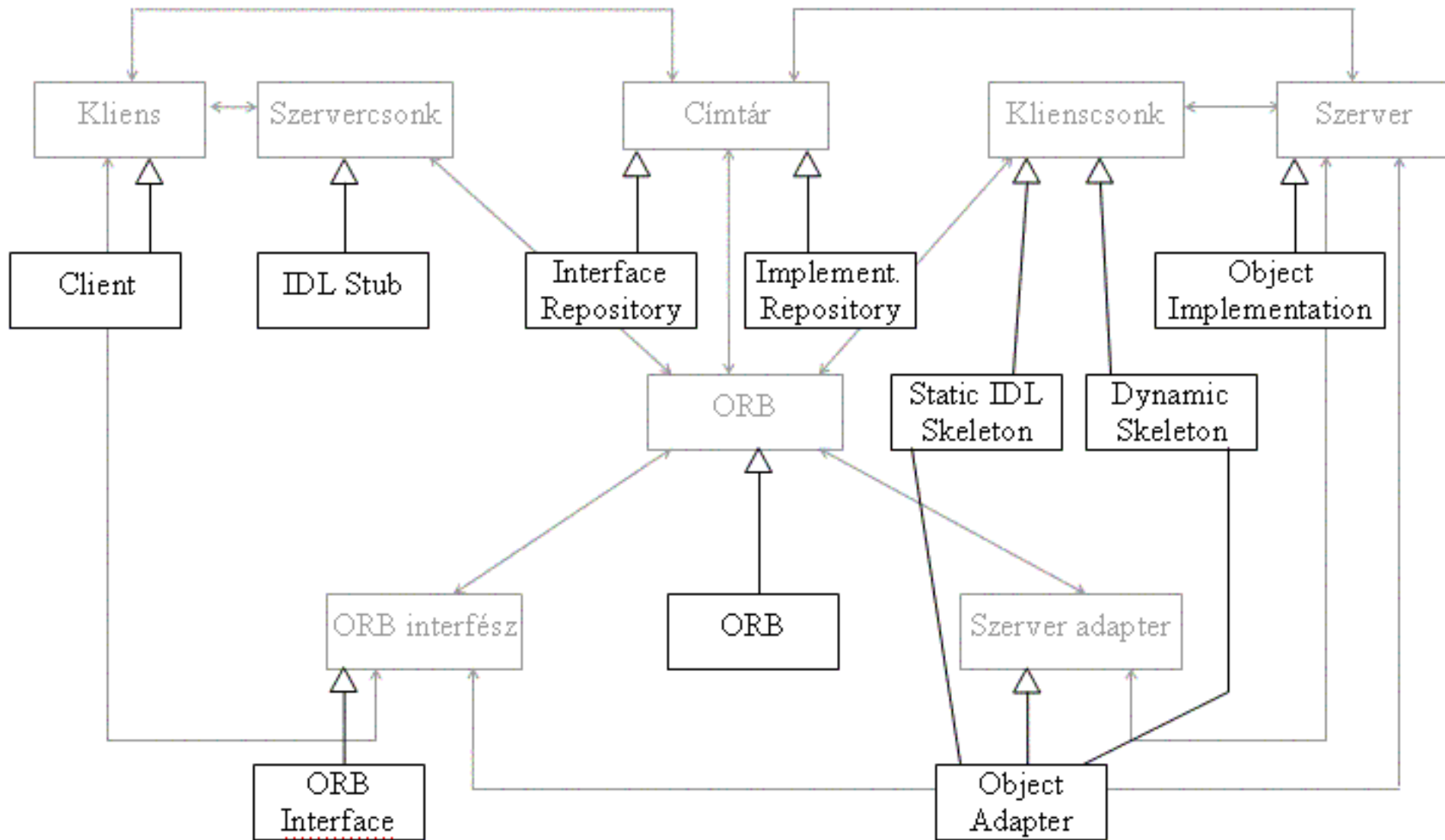
Általános ORB architektúra



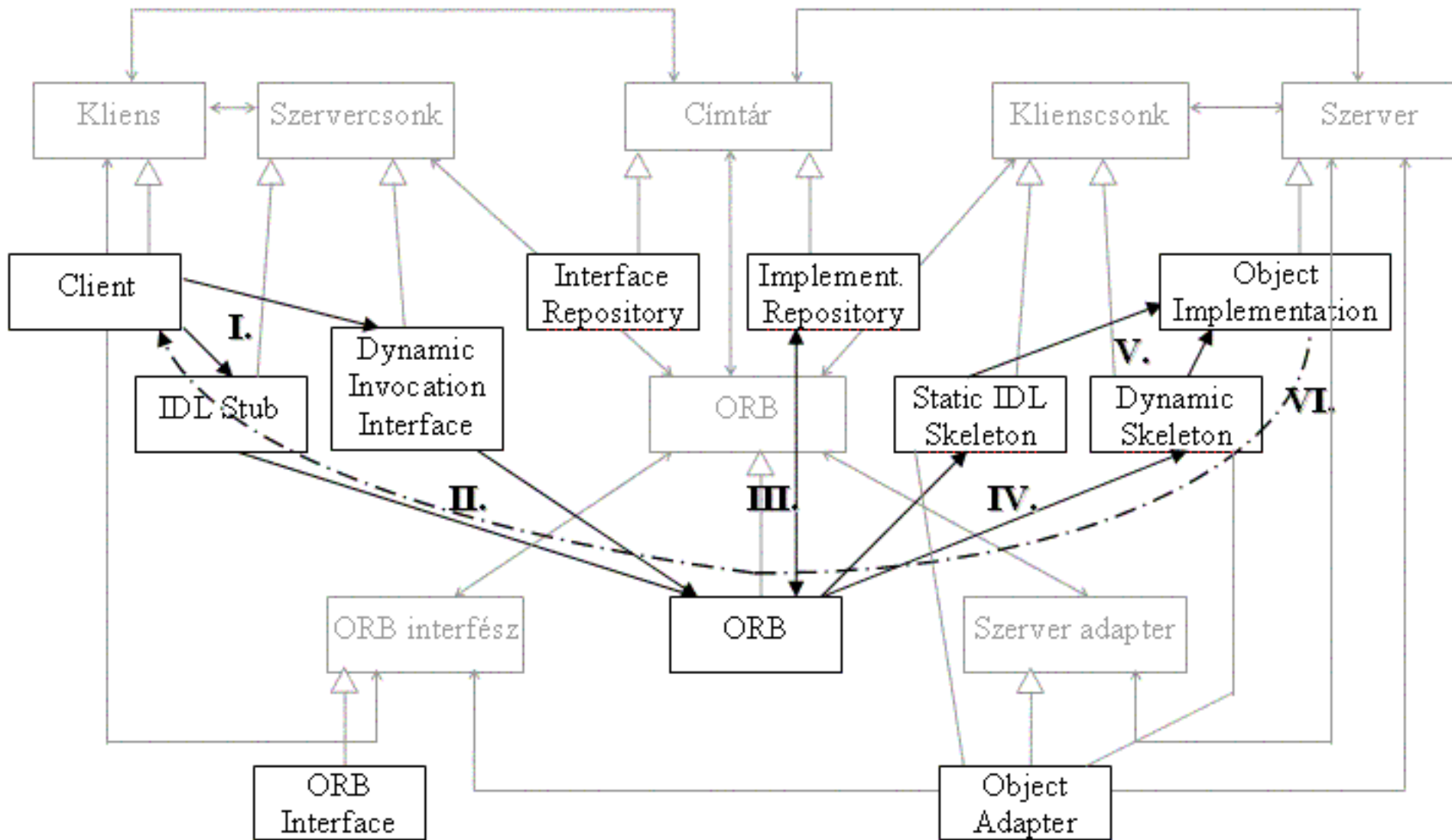
Általános ORB kommunikáció



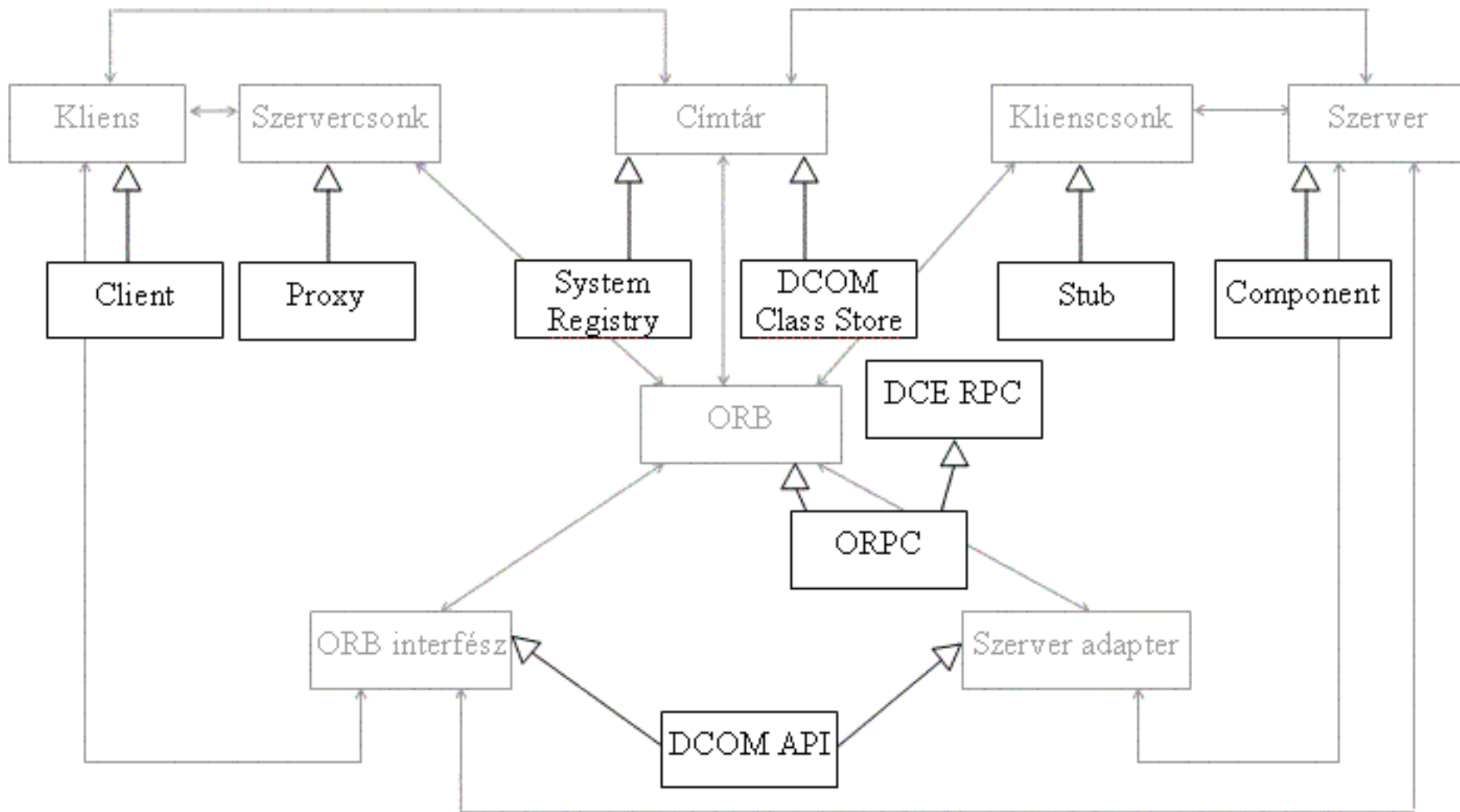
CORBA architektúra



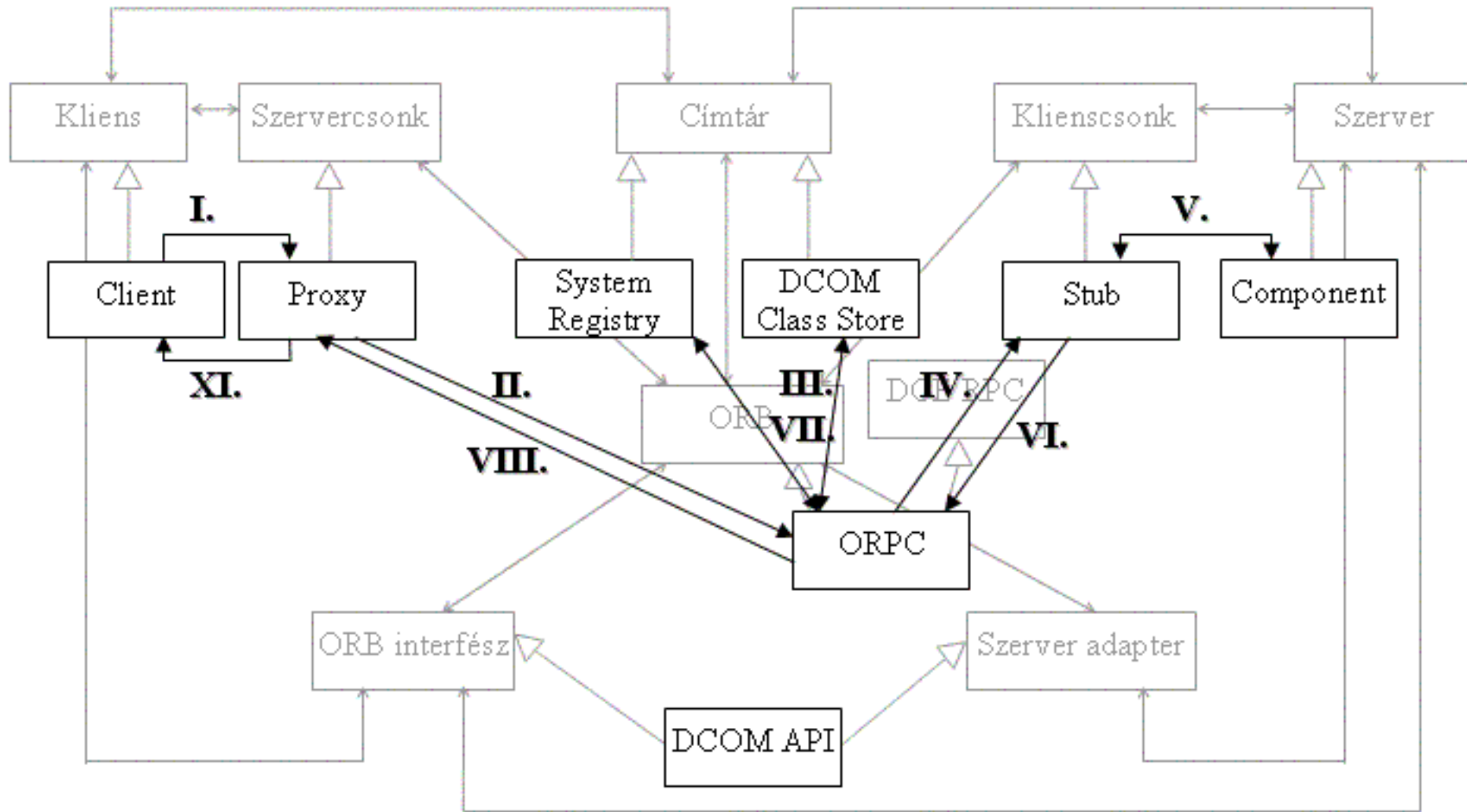
CORBA kommunikáció



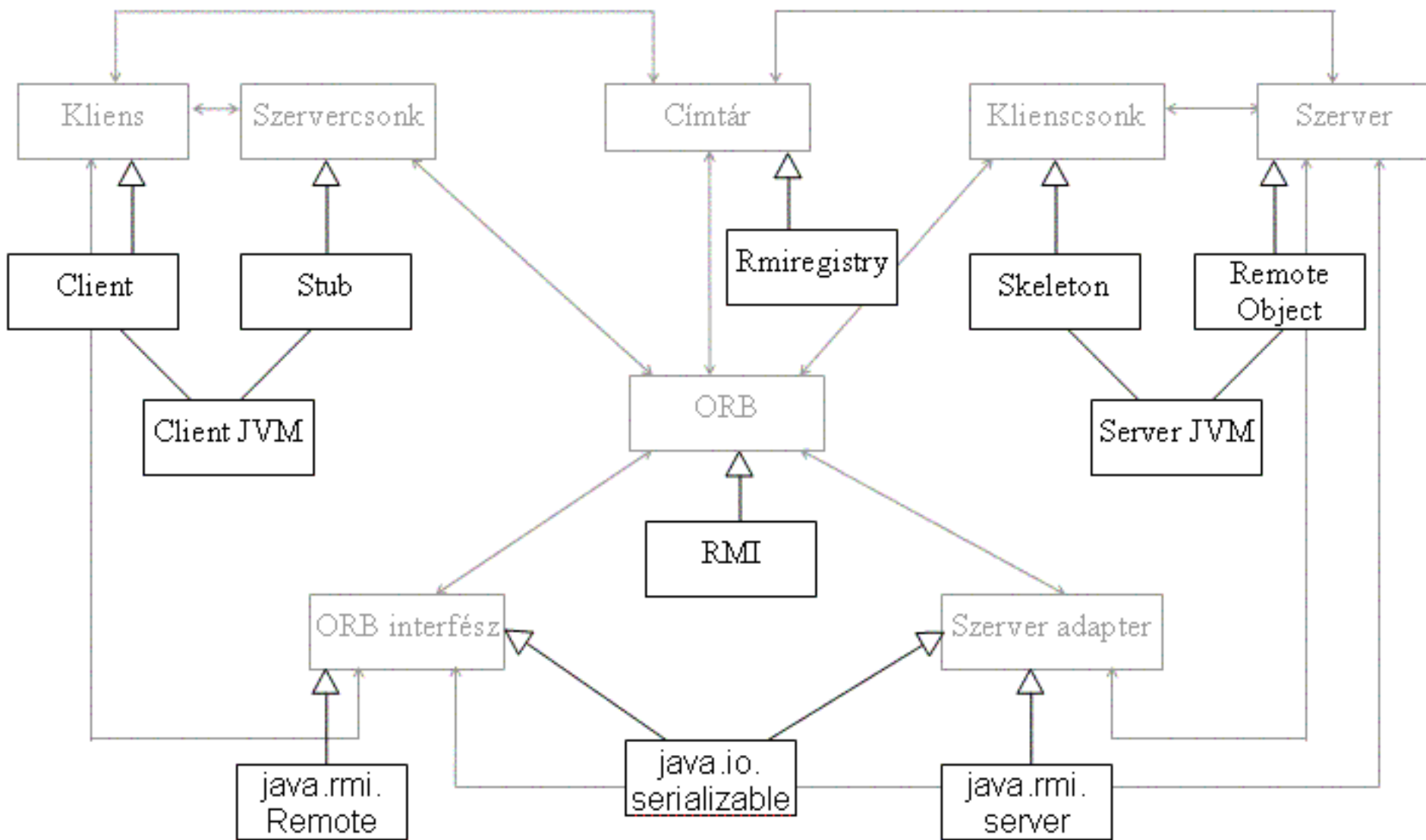
DCOM architektúra



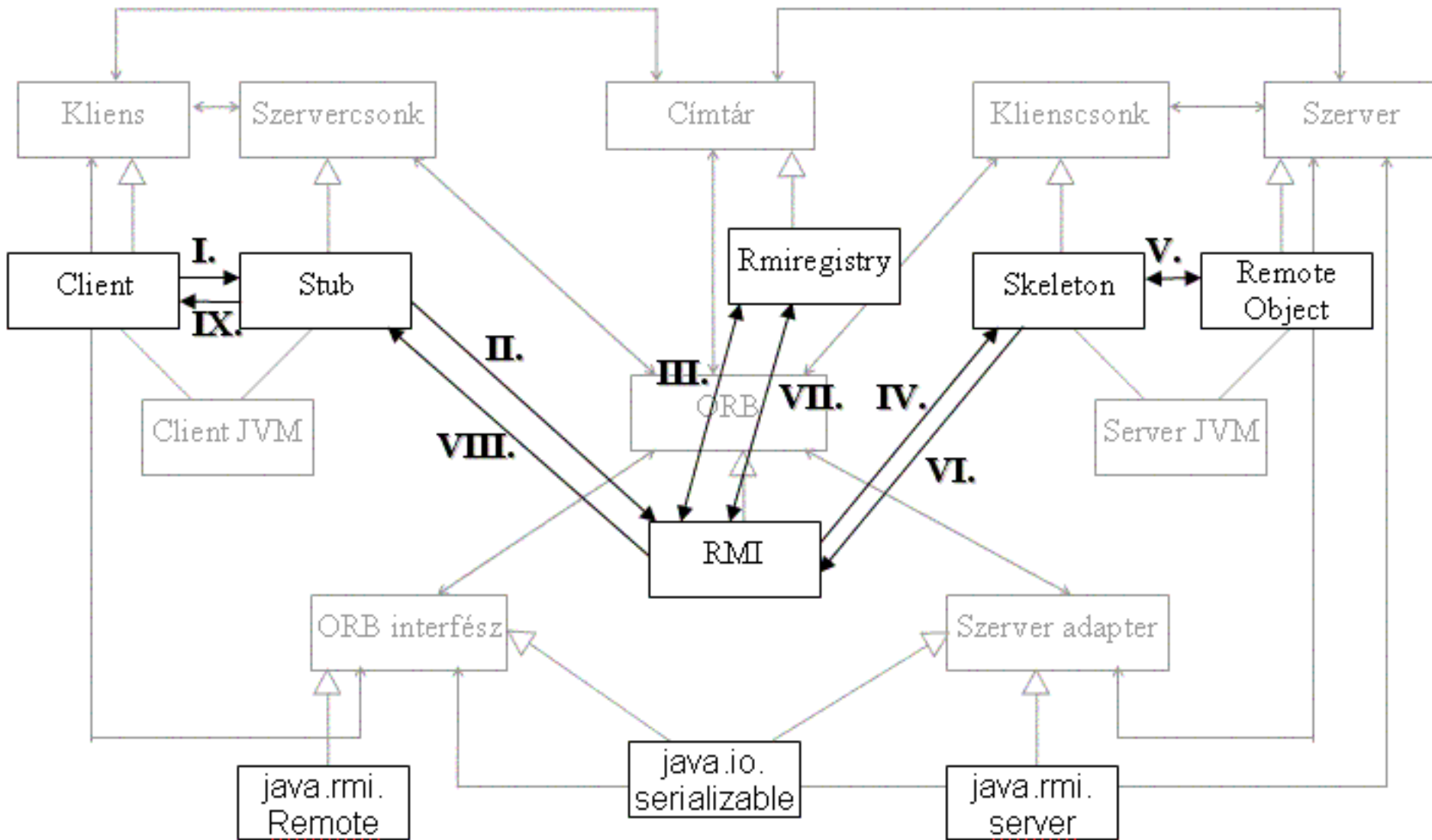
DCOM kommunikáció



RMI architektúra



RMI kommunikáció



Korszerű middleware technikák

- Házilagos megoldások
- Távoli eljáráshívás (RPC, ORB)
- Üzenetsorok (MQ)
- Publish-Subscribe (P/S)
- Egyéb aszinkron technikák

Üzenetsorok

□ Jellemzők

- Üzenet-orientált, aszinkron
- Inkább csak „... az egynek” kommunikáció
- Laza csatolás
 - nincs közvetlen kapcsolat
 - nem szinkronizálódnak
 - nem fogják vissza, nem rántják le egymást
- Nagy megbízhatóságú → nem gyors

□ Alapfogalmak

- Üzenetek: átküldeni szánt információ adag (msg.)
- Sorok: üzenetek elosztói, tárolói (queues)

Sorok feladatai

- Üzenetek fogadása a küldőtől
 - Aszinkronitás (vezérlés visszaadása a lehető leghamarabb)
 - Tárolás, míg címzett át nem veszi
- Postafiók rendszerű működés
 - Címzett változhat, ha a postafiók marad
- Üzenet nem veszhet el, amíg a sor él
 - Alkalmazásokat, MW-t futtató gépek leállása
- Üzenetek transzformációja
 - Interpretálhat, konvertálhat
 - Csak egyszerűbb átalakításokra van idő, kapacitás

QoS

- Garanciák (külön be kell állítani, ha lehet)
 - üzenet nem veszhet
 - üzenet nem duplikálódhat
 - sorrend nem cserélődhet fel
- Szintek (másik megközelítés)
 - 0 – „best effort”
 - 1 – kézbesítés legalább egyszer
 - 2 – kézbesítés pontosan egyszer

Üzenetsorok

- Előnyök
 - Nagy megbízhatóságú hálózati kommunikáció
 - Mobil (off-line) partnerek kommunikációja
 - Új és hagyományos rendszerek laza csatolása
- Hátrányok
 - Nehézkes inicializálás és adminisztráció
 - Lassú, ha a sor hosszú
 - „Sok soknak” komm. nehezen megvalósítható

Üzenetsorok

- Tipikus alkalmazási területek
 - Webes megrendelés felvétel
 - feldolgozás hagyományos alkalmazásokkal a háttérben, lassan, az ügyfelet elengedve
 - Ügynöki kiszolgáló rendszer
 - ügynökök off-line szakaszainak tolerálása
 - Tranzakciós rendszer felhasználói felülete
 - Felület omlása ne vigye magával a tranzakciót

Korszerű middleware technikák

- Házilagos megoldások
- Távoli eljáráshívás (RPC, ORB)
- Üzenetsorok (MQ)
- **Publish-Subscribe (P/S)**
- Egyéb aszinkron technikák

□ Jellemzők

- Üzenet-orientált, aszinkron
- Jó „... a soknak” kommunikáció
- Laza csatolás (mint az MQ-nál)
- Rugalmas komm. vegyes hálózati környezetben
- Terjesztő transzformálhatja az üzeneteket

□ Alapfogalmak

- Üzenetek: átküldeni szánt információ adag (msg.)
- Terjesztő: fogadó és elosztó hálózat
(publishing service, publishing network)
- Előfizetők: terjesztőnél regisztrált címzettek
(subscribers)

Terjesztő feladatai

- Üzenetek fogadása a küldőtől
 - Aszinkronitás (vezérlés visszaadása a lehető leghamarabb)
 - Címzettek azonosítása, útvonalválasztás
- Üzenetszórás a regisztrált címzettek felé
 - Terjesztő hálózat optimális kihasználása
- Regisztrációs lista folyamatos, központosított karbantartása
- Feliratkozási rendszerek (alap típusok)
 - Kategória alapú
 - Kulcsszavas
 - Mintaillesztős

- Előnyök
 - Komm. folyamatosan változó partnerekkel
 - Jól skálázódó „... soknak” kommunikáció
 - Időre érzékeny hagyományos rendszerek összekötése
- Hátrányok
 - Nehezen tranzakciósítható
 - Nehézkes üzemeltetés és hibakeresés
 - Robusztus, teljesítőképes hálózat kell alá

- Tipikus alkalmazási területek
 - Valós idejű árverező és tőzsdei rendszerek
 - bárki bármikor bármire fel-/leiratkozhat
 - bármelyik tag küldhet
 - Időjárás-jelentő rendszer
 - hírügynökségek feliratkoznak (terület, esemény)
 - jelentések folyamatosan mennek ki
 - jelentés készítése és előfizetés kezelése szétválik
 - Szolgáltatás-orientált rendszerek
 - „Nem tudom, kinek a dolga, de valaki csinálja meg”
 - bizonyos funkciókra mindig van előfizető
 - Hálózati riasztórendszer
 - hálózat gépei be-/kikapcsoláskor fel-/leiratkoznak

Korszerű middleware technikák

- Házilagos megoldások
- Távoli eljáráshívás (RPC, ORB)
- Üzenetsorok (MQ)
- Publish-Subscribe (P/S)
- **Egyéb aszinkron technikák**

Egyéb aszinkron megoldások

- Aszinkron kommunikáció sokszor hasznos
- MQ és P/S infrastruktúrája költséges
 - Időben és pénzben is
- Sokszor nem a nagy megbízhatóság a lényeg
- Sokszor fix, ismert a címzett

- Fire and forget (FF)
- Ajánlott üzenetek (Sync with server)
- Lekérdezés (Polling)
- Visszahívás (Call back)

Fire and forget

- Szerver nem ad visszatérési értéket
 - Pl. egyoldalú értesítések mennek
- Hibaüzenetek sincsenek
 - Pl. megismételt üzenet már nem lenne aktuális (külvilág nem állítható meg, pörgethető vissza)
- Üzenetvesztés elfogadható
 - Pl. nem kritikus a szolgáltatás vagy „úgyis csak frissen jó”
- Példák:
 - Loggolás
 - Model – View – Controller

Fire and forget

- Megoldás
 - Lokális csonttal szinkron kommunikáció
 - Csonk üzen távolra, de nem blokkol
 - új szálon fut,
vagy nem blokkoló kommunikációt használ

Ajánlott üzenetek

- Szerver nem ad visszatérési értéket
 - Pl. egyoldalú értesítések mennek
- Hibaüzenetek sincsenek, de visszaigazolás kell
 - Feldolgozás előtt, csak a kézbesítésről
- Megoldás:
 - Kliens oldali csomópont visszaigazolásig blokkol
 - Hálózati hibát detektál, szerver oldalit alig
 - Szerver oldali csomópont nem a feldolgozás szálán fut

Lekérdezés

- Aszinkron kommunikáció, de kell az eredmény
 - De nem kell azonnal
 - Szerverrel párhuzamosan dolgozó kliens
 - .pl. a kért ID generálása közben a kliens létrehoz, konfigurál, kitölt (amit ID nélkül is lehet)

- Megoldás:
 1. Kliens oldali csomópont pollozza a szerveret
 - kliens blokkolva, aszinkron ez?
 2. Kliens oldali csomópont blokkolva, kliens dolgozhat
 - kliens pollozza a csomópont egy szálát

- Hosszú pollozás → drága; rövid → szinkron jobb

Visszahívás

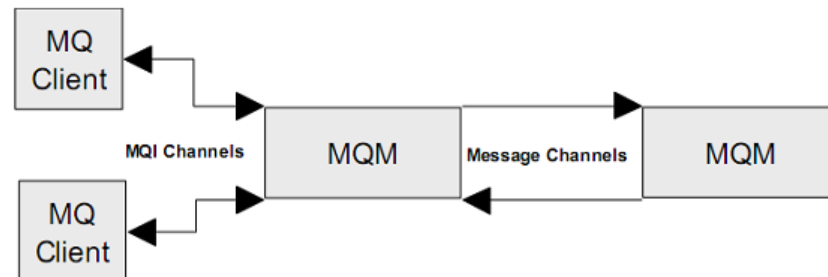
- Aszinkron kommunikáció, de kell az eredmény
 - Amikor pollozni hosszú lenne
- Megoldás:
 - Kliens oldali csonk blokkolt szála az eredménnyel visszahívja a klienst
 - kliens call-back interfésze? ennek címe?
 - Több szálas kliens kell
 - nem transzparens módszer

TECHNOLÓGIÁK

Message Queuing (MQ)

□ Queue manager

- Üzenetsorok kezelése



□ Lokális és távoli alkalmazások közti kommunikáció

- Banki, biztosítói, stb. rendszerekben elterjedt
- Tipikusan elfedik (pl. Message Broker)
- IBM Websphere MQ, Apache ActiveMQ, JBoss Messaging, RabbitMQ (Erlang)
- Szinkron/aszinkron kommunikáció
- Üzenetek perzisztens tárolása
- API több nyelvhez (C, C++, Java, COBOL)

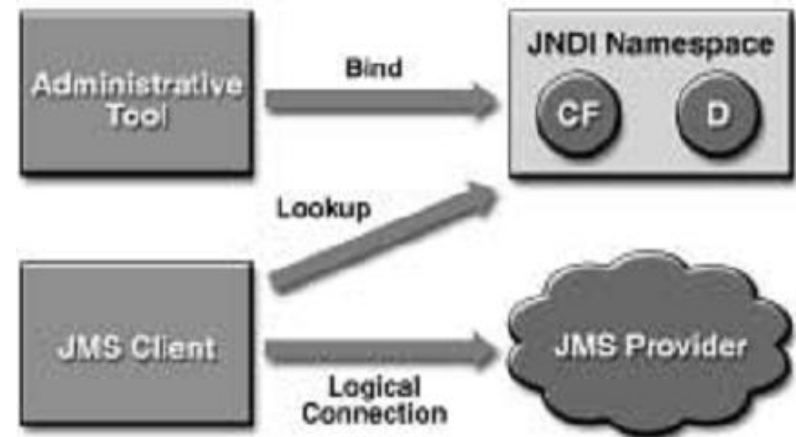
MQSeries Primer: <http://www.redbooks.ibm.com/redpapers/pdfs/redp0021.pdf>

MQ folytatás

- Üzenetek típusa többnyire programozón múlik
 - Stream, text, byte, map...
 - Nincs garantált „típushelyesség”
- Sorok (PTP) és „témák” (P/S) támogatása
- Célszerűen a kommunikációs kód leválasztandó
- MQ lehet átviteli közeg pl. webszolgáltatásokhoz
- Gyártóspecifikus megoldások...

Java Message Service (JMS)

- Java API üzenetküldéshez
- Adminisztráció: JMX
 - JNDI névtér
 - ConnectionFactory, Destination



- Java EE szabvány része
 - Kötelezően implementálandó alk. szerver oldalon
- Üzenetsor/téma („durable subscription”)
- Java EE: message-driven bean
- WS-* alatt...
- TIBCO, JBoss, IBM, Oracle, Fiorano....

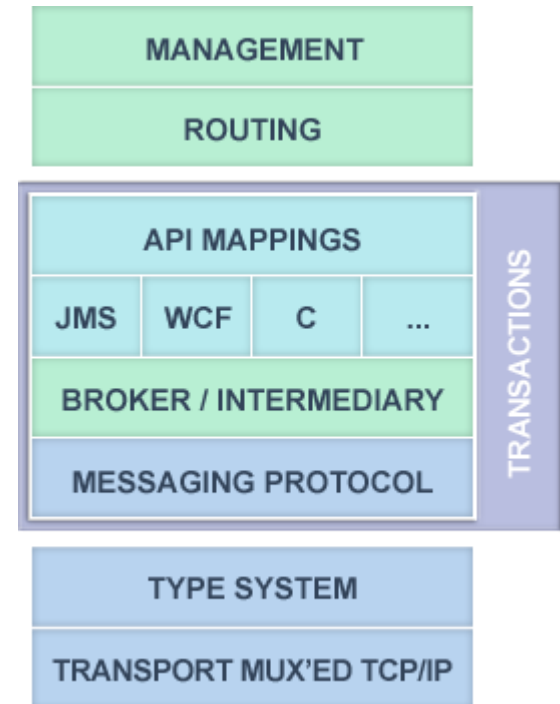
Java™ Message Service API Tutorial
by Kim Haase, Sun 2002

MQTT

- MQ Telemetry Transport
 - Várhatóan OASIS szabvány lesz
 - Erős IBM befolyás
- Kis protokoll overhead (2 byte), kis sáv szélesség
 - At-most once, at-least once, exactly once
- Gyors átvitel, megbízhatatlan hálózatra készítve
 - Max. 256 MB üzenet, TCP/IP fölött
 - Az üzenet tartalmáról nincs információ
- Szenzorok, mobil eszközök, stb.
 - MQTT-S: ZigBee
- Facebook (pl. mobil szinkronizáció)

AMQP

- Advanced Message Queuing Protocol
- Bináris szintű protokoll
 - Red Hat, Microsoft, VMWare...
 - Bank of America, JPMorgan...
- Nagy üzenetmennyiség kezelése
 - Prioritások az állapotjelzéseknek
 - Perzisztencia, biztonság, ...
- API mapping
 - JMS, WCF, Python...
 - Saját üzenatkódolás/XML/JSON



<http://www.amqp.org/product/architecture>

STOMP

- Simple Text Oriented Messaging Pr.
- Scriptnyelvek igényeihez fejlesztve
 - Ruby, Python, Perl, ...
 - HTTP fölött
 - Nincs közös üzenetküldési szemantika
 - Nyugtázás, tranzakcionalitás
 - Session kezelés
 - Header: kulcs-érték párok

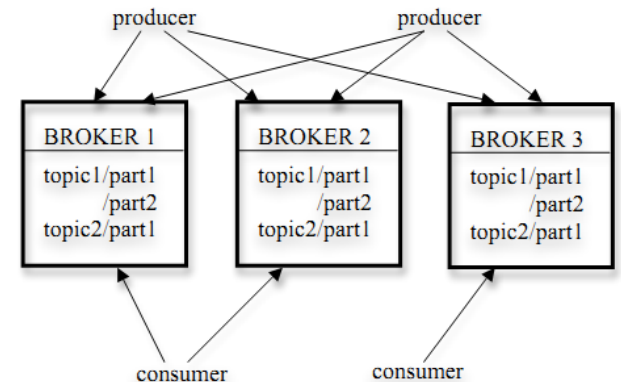


<http://blogs.vmware.com/vfabric/2013/02/choosing-your-messaging-protocol-amqp-mqtt-or-stomp.html>

<http://stomp.github.io/index.html>

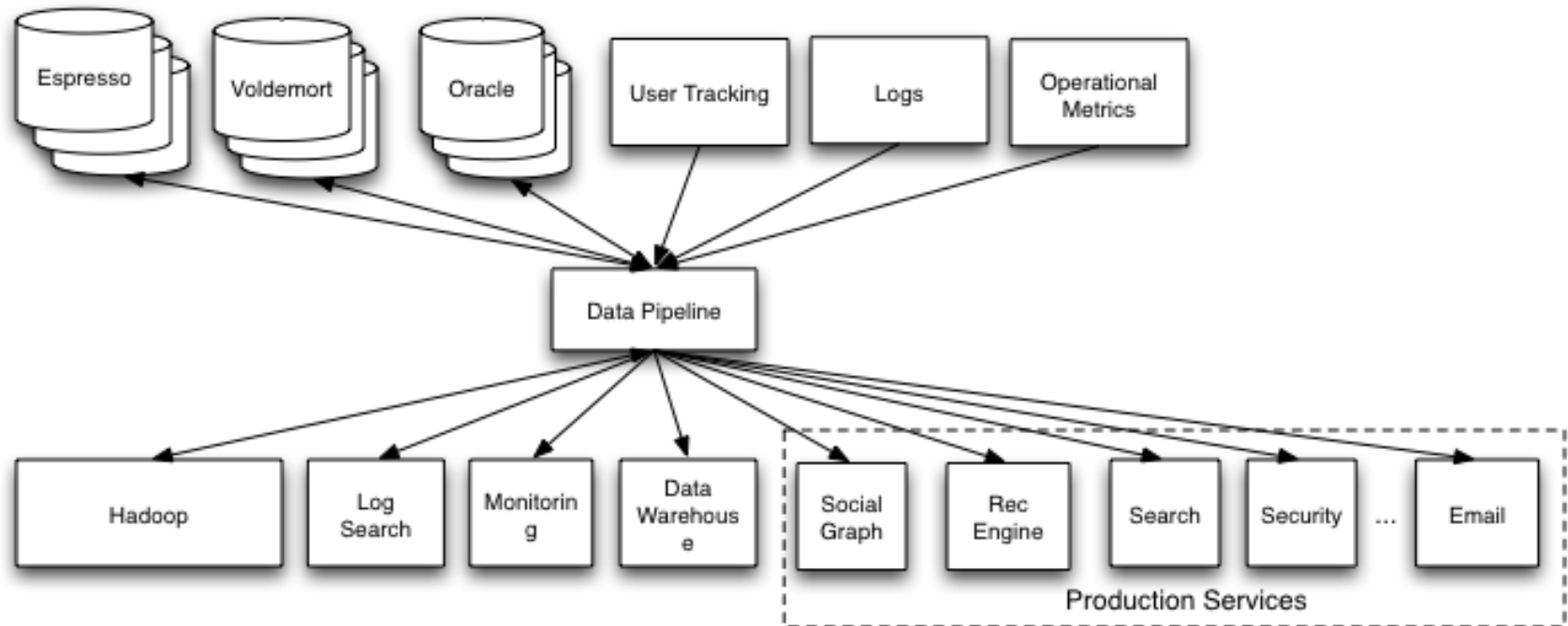
Apache Kafka

- Nyílt forráskódú P/S protokoll (2011)
- Eredetileg logfeldolgozásra
 - (LinkedIn)
- Elosztott architektúra
- Állapot a fogadó oldalán karbantartva
- Tipikusan at-least-once szemantika
- Hatékony üzenetfeldolgozás (pl. Storm, Hadoop...)



<http://research.microsoft.com/en-us/um/people/srikanth/netdb11/netdb11papers/netdb11-final12.pdf>

Kafka példa (LinkedIn)



<http://www.slideshare.net/chriscurtin/ajug-march-2013-kafka>

Legközelebb...

- Hogyan használjuk fel ezeket?
- Hogyan áll össze a „szolgáltatásintegráció”?

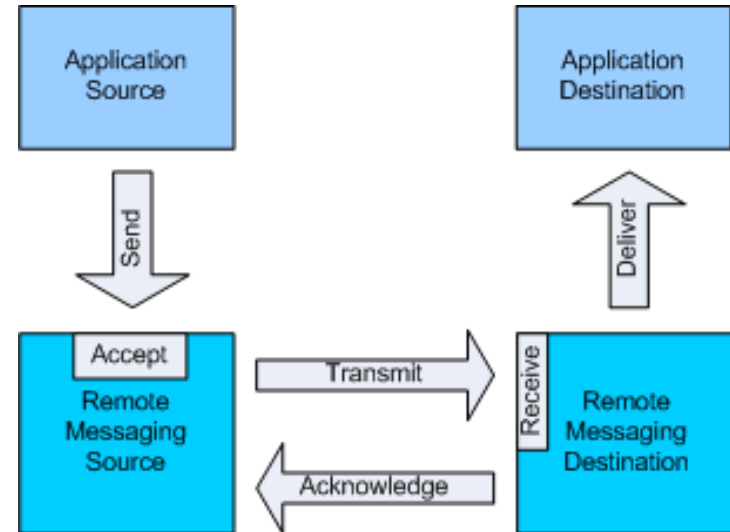
EMLÉKEZTETŐ....

Szolgáltatás konfigurációk ellenőrzése

- Teljesítőkéesség vizsgálata
 - „Performability = Performance + Reliability”
- Mi történik hibák esetén?
 - Pl. a megbízható üzenetküldést garantáló middleware újraküldi az elveszett üzenetet → megnőhet a garantált válaszidő
 - Pl. túl kicsi a beállított timeout → téves újraküldések
- Mi a megbízhatóság ára? (teljesítményben)
- Hogyan állítsuk be az SLA paramétereiket??

Felhasznált technológia

- Reliable messaging: WS-RM
 - „TCP protocol”
- Mit jelent?
 - Nyugtázás
 - üzenetsorrendezés
 - Duplikátumok szűrése
 - Garantált kézbesítés
- Üzenetküldési szemantika
 - At-least once
 - At-most-once
 - Exactly-once
- Több szabványból fejlődött ki (MS, IBM)
- implementációk
 - RAMP (IBM WebSphere Application Server)

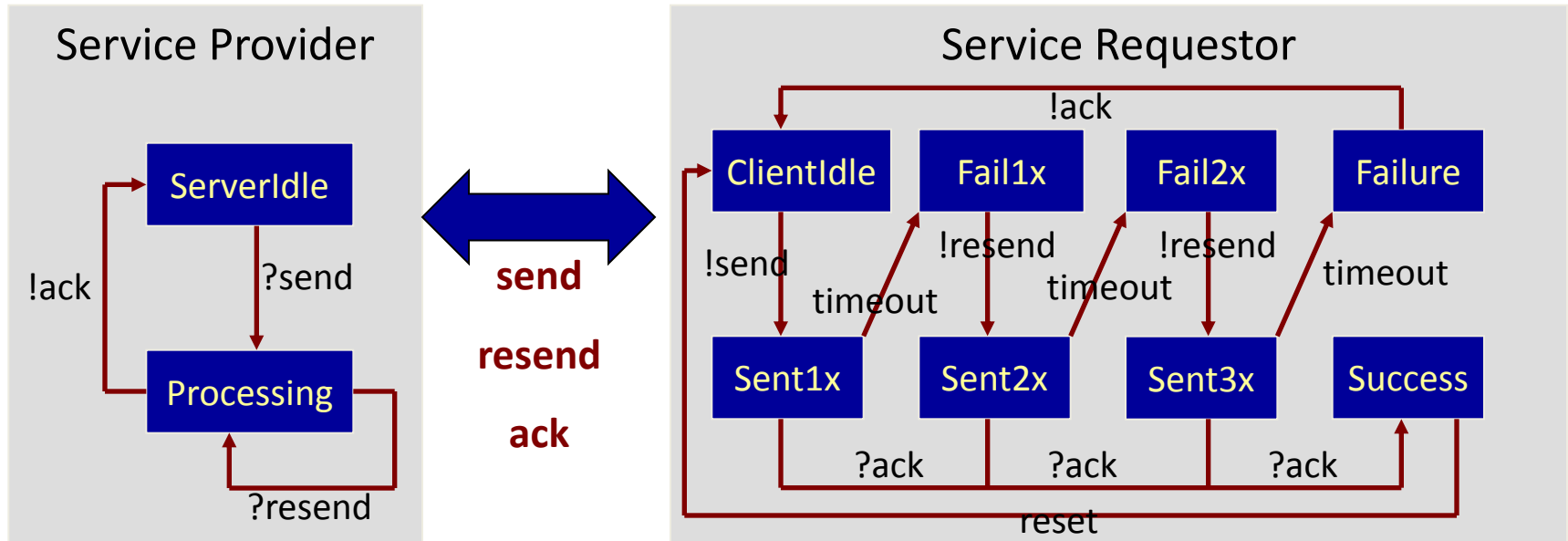


Mit modellezünk ebből?

- Absztrakt viselkedés
 - Szolgáltató
 - Kliens
- Üzenetkezelés paramétereit (származtatottak)
 - Üzenetkezelés módja
 - Újraküldések száma
 - **send, resend, ack** paraméterek
 - (exponenciális eloszlás)

Middleware modell

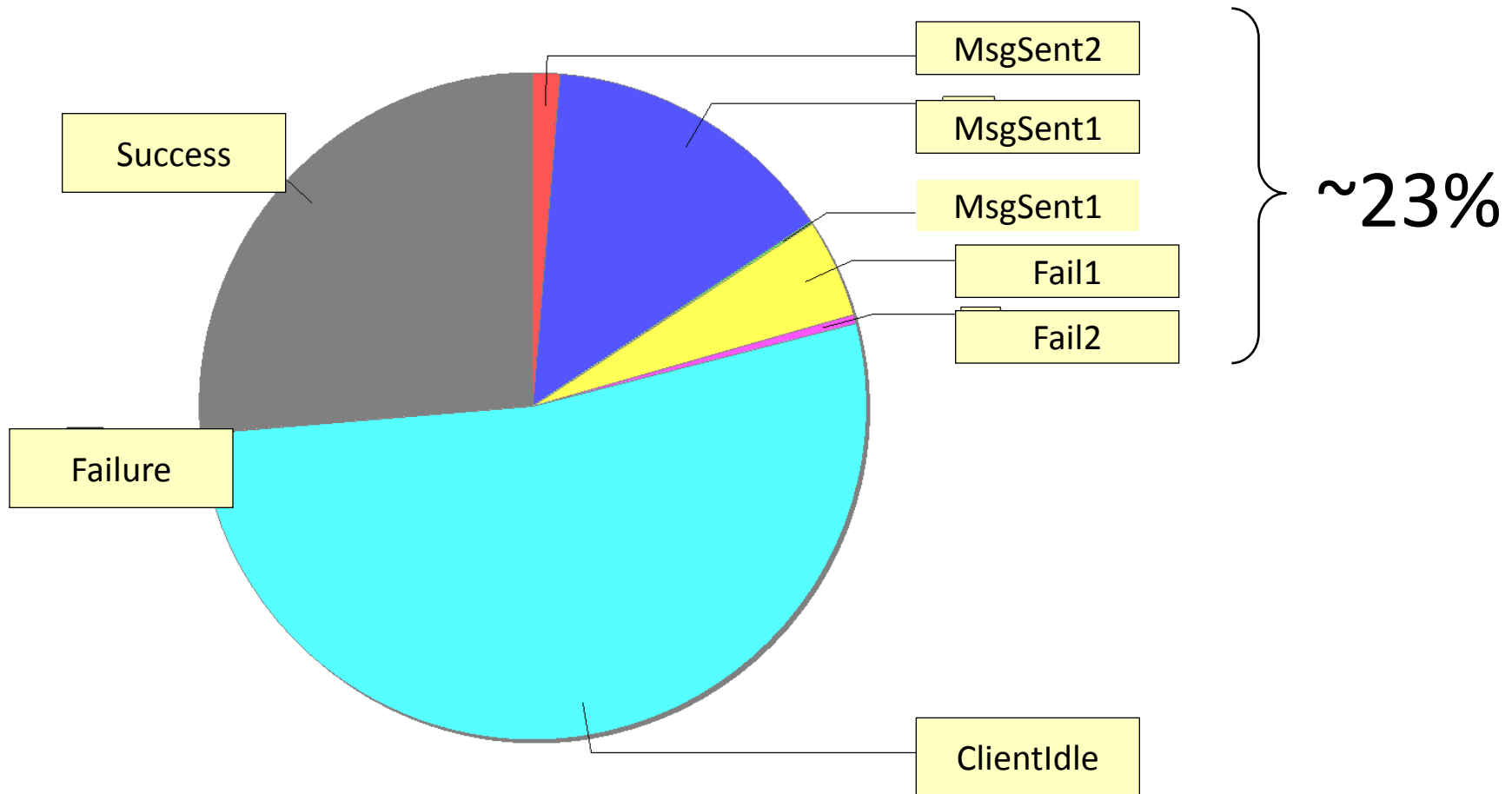
- Platform leíró
- A konkrét konfigurációs modell paraméterezi



Analízis eredmények: Kihasználtság

Állandósult állapot vizsgálata

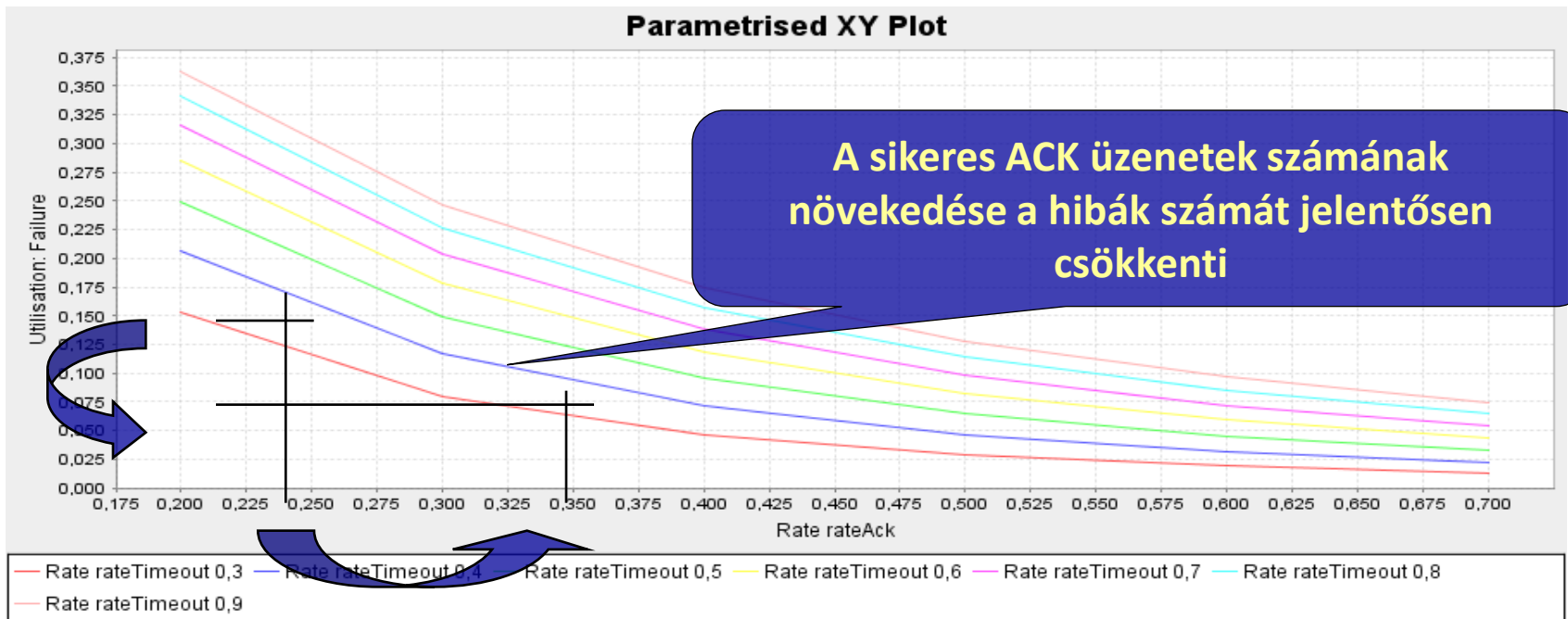
Az idő mekkora része telik a hibák kezelésével?



Analízis eredmények: érzékenységvizsgálat

Érzékenységvizsgálat: hol változtassunk?

Hogyan változik a rendszerszintű hiba valószínűsége a „resend” időzítés változásának hatására?



További referenciák

- <http://pavanz.blogspot.hu/2010/10/messaging-middleware-products.html>
- <http://www.docstoc.com/docs/799743/A-Comparison-of-Middleware-Products>
- <http://userpages.umbc.edu/~dgorin1/451/middleware/middleware.pdf>
- <http://www.b3websolutions.com/article/EAICompetitiveComparison.htm>
- <http://blog.cobia.net/cobiacom/2012/08/02/red-hat-enterprise-middleware-comparison-with-cloud-native-middleware/>
- <http://www.eaipatterns.com/ObserverJmsExample.html>
- <http://www.einfobuzz.com/2012/01/jms-design-considerations.html>