PEOPLE COME FIRST
INFORMATIKAI SZAKÉRTŐK EGYESÜLETE

# Oracle Junior Program
# 2021/22 őszi félév

## IT rendszerek és architektúrák üzemeltetése

# Féléves tematika

**2021.09.09.** Az adatbázisok architektúrája

**2021.09.16.** Az adatbáziskezelő rendszerek kötelezően elvárt, illetve opcionális képességei

**2021.09.23.** Vállalati informatikai rendszerek architektúrájának az áttekintése

**2021.09.30**. Middleware

**2021.10.07**. Alkalmazásinfrastruktúra

**2021.10.14**. Identity management

**2021.10.21**. Virtualizáció és cloud

**2021.10.28.** Őszi szünet

**2021.11.04**. Monitorozás, rendszerfelügyelet, hibaelhárítás

**2021.11.11.** Docker & Kubernetes

**2021.11.18.** Automatizált infrastruktúra

**2021.11.25**. Big data és elosztott adattárolás

**2021.12.02.** Elosztott adatfeldolgozás és adatbázisok

**2021.12.09.** Ismétlés, kitekintés az IT piacra

# Tudnivalók

**A tantárgy a félév során online, zoom konferencia keretén belül zajlik csütörtökönként 18:00 órától!**

- **A tárgy teljesítéséhez:**
  - Jelentkezés a kurzusra és a vizsgára a **login.pcf.hu** oldalon, kredites hallgatóknak is
  - Év végi vizsgát teljesíteni
- **Vizsga:**
  - 4 vizsgaalkalom
  - Online vizsgázás

**Online vizsgaalkalmak:**

1. December 16.
2. 2022. január 6.
3. 2022. január 13.
4. 2022. január 20.

# Bevezető

- Egy nagyvállalat megannyi rendszert (programot) üzemeltet
- A célszoftverek, vagyis alkalmazások lehetnek:
  - egy ügyfél számára fejlesztett programok,
  - egy általános szoftver testreszabása,
  - egy applikáció módosítás nélküli használata
- Az applikációs szoftverek működésének feltétele, hogy a gépeken alapból bizonyos rendszerszoftverek működjenek. Ilyenek pl. az:
  - Operációs rendszerek
  - Adatbáziskezelő rendszerek
  - Applikációs szerverek
  - Identity Management rendszerek
  - Messaging rendszerek
  - Sok-sok egyéb

# Miről szól ez a tárgy/előadássorozat?

- Az operációs rendszereket elég jól megtanítják az egyetemi oktatásban
- Az adatbáziskezelőkről már jóval kevesebb ismerettel rendelkeznek a friss diplomások
- A többi rendszerszoftvert lényegében nem oktatják.
- Célunk bemutatni ezeket
- Mivel az előadássorozatot az Oracle Corporation támogatja, ezért a gyakorlati példákat többnyire az Oracle termékeiből választjuk

# Miről fogunk beszélni a szemeszter folyamán?

- Adatbázisok architektúrája és funkcionalitása
- Alkalmazásszerverek
- Identity management rendszerek
- IT rendszerek monitorozása/rendszerfelügyelet
- Virtualizáció és Cloud
- Docker és Kubernetes
- Automatizált infrastruktúra
- Big Data és Hadoop
- NoSQL

# Miért érdemes erről tudnotok?

- A rendszergazdai és DevOps feladatok izgalmasak, kihívóak
- Nagy a hiány az ilyen szakértőkből
- Minden informatikusnak hasznos az ilyen alapismeret
- Téves az a vélemény, hogy az ilyen szakértők munkája nem elég kreatív

- A People Come First (PCF) egyesület úgy érzi, hogy hozzá kell járulnia e szakma népszerűsítéséhez
- Aki mélyebben szeretne ebben elmerülni, az keresse fel a PCF egyesület ZeroToHero honlapját ☺

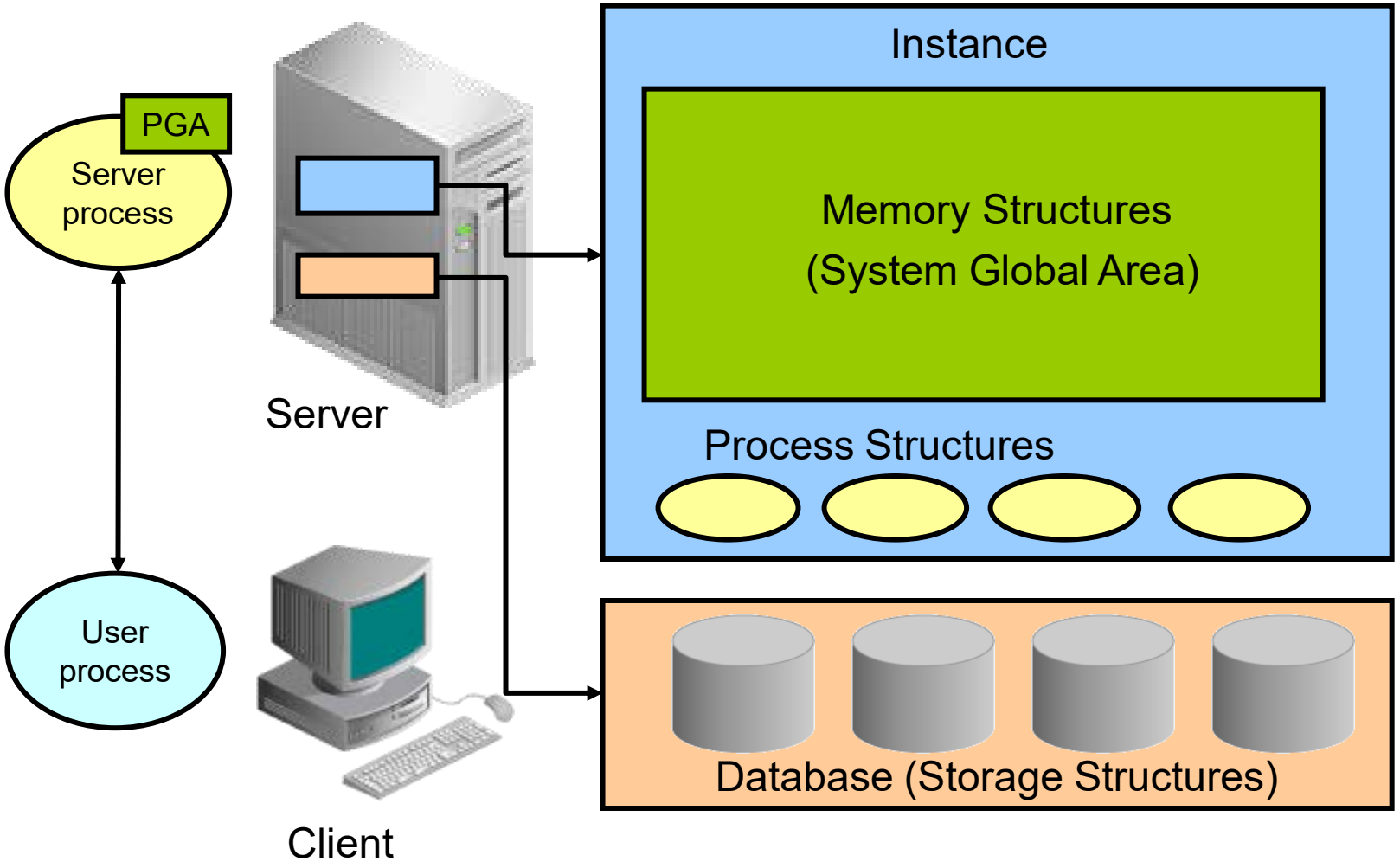
# Először az adatbázisokról beszélünk

- Ezen az előadáson egy tipikus adatbáziskezelő rendszer architektúrája következik
- A következő előadás az adatbáziskezelő rendszerek kötelezően elvárt és opcionális képességeiről szól majd

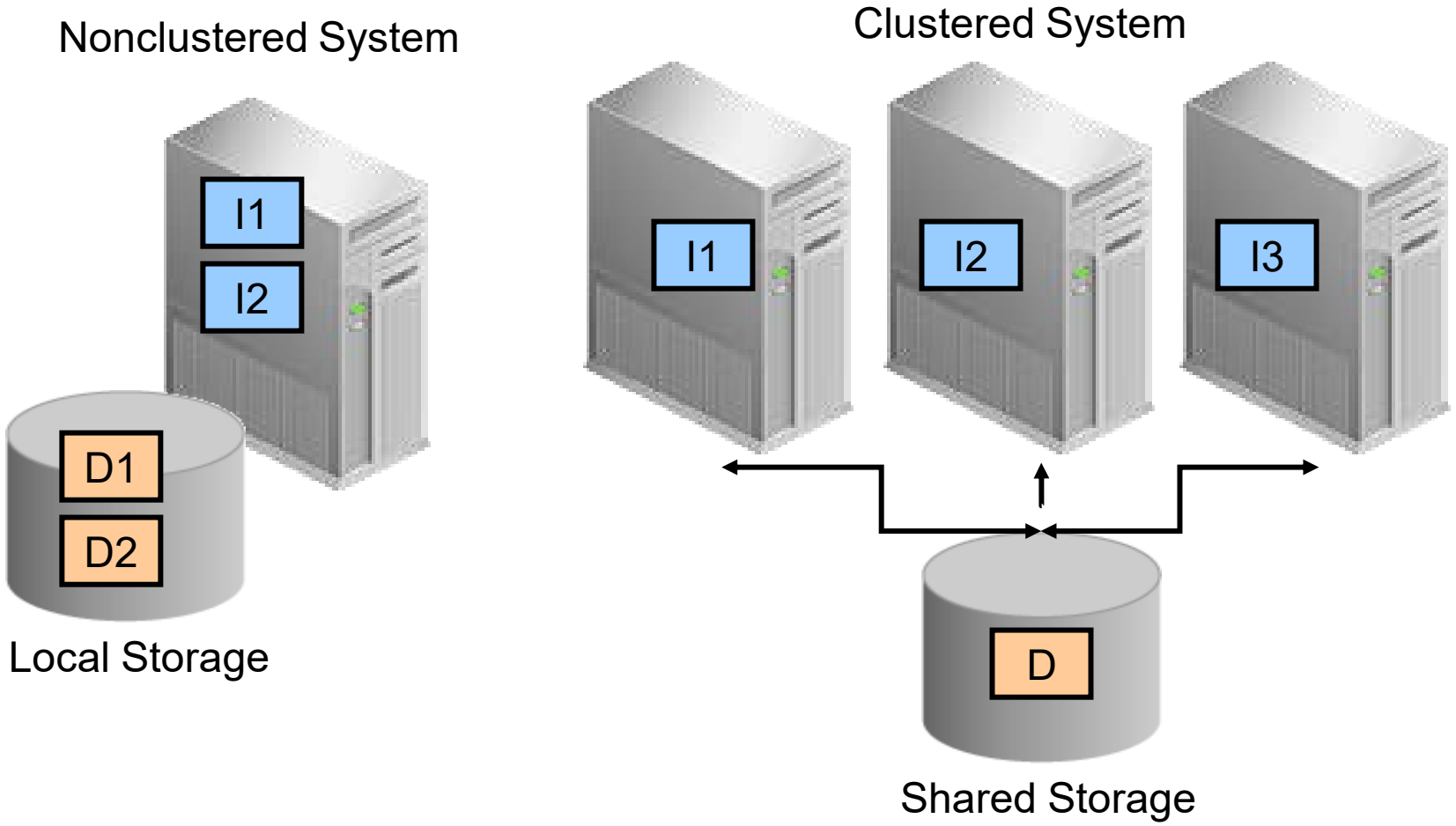
- Előadó: Kerepes Tamás, ügyvezető igazgató, Webváltó Kft.

# Exploring Oracle Database Architecture

# Oracle Database Server Architecture: Overview

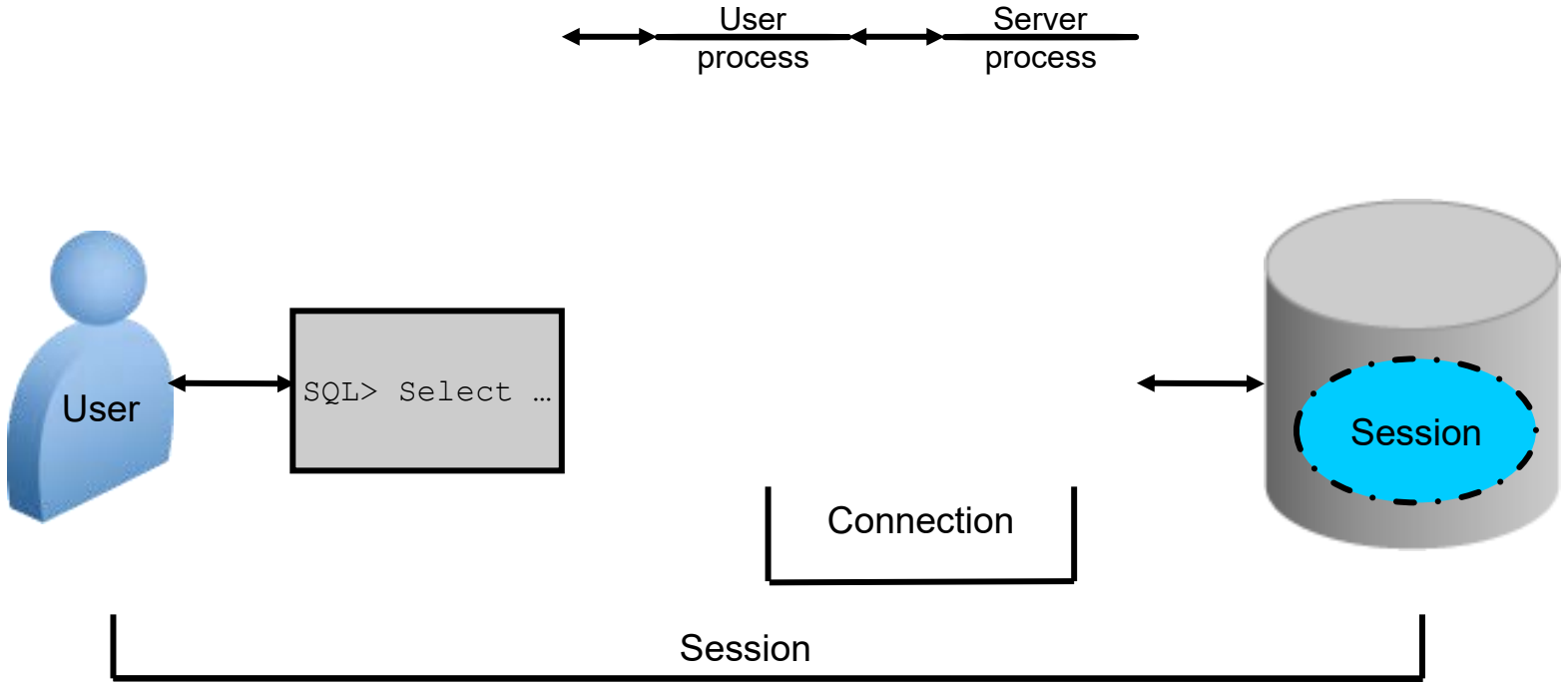


# Oracle Database Instance Configurations

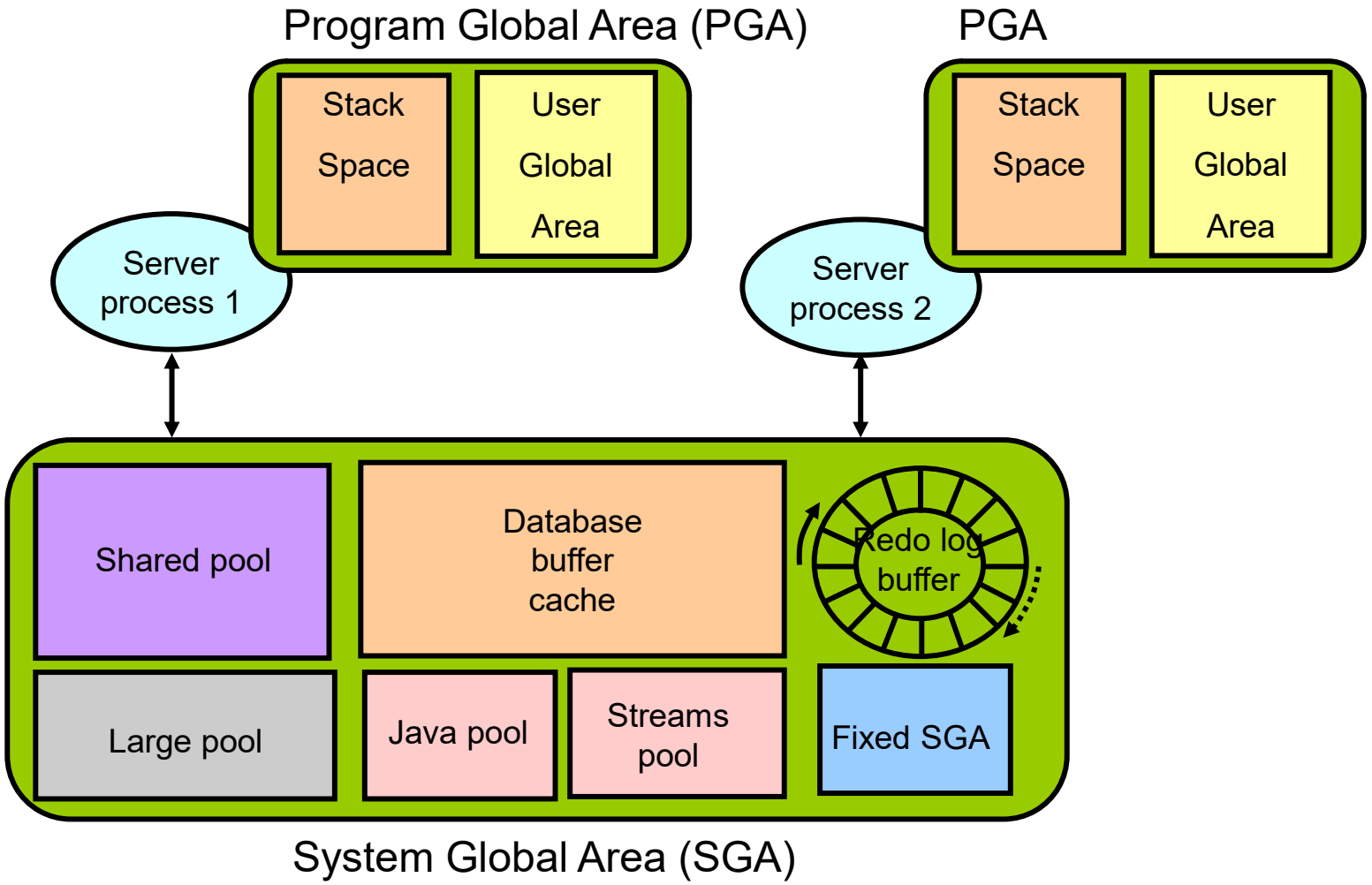


# Connecting to the Database Instance

- Connection: Communication between a user process and an instance
- Session: Specific connection of a user to an instance through a user process

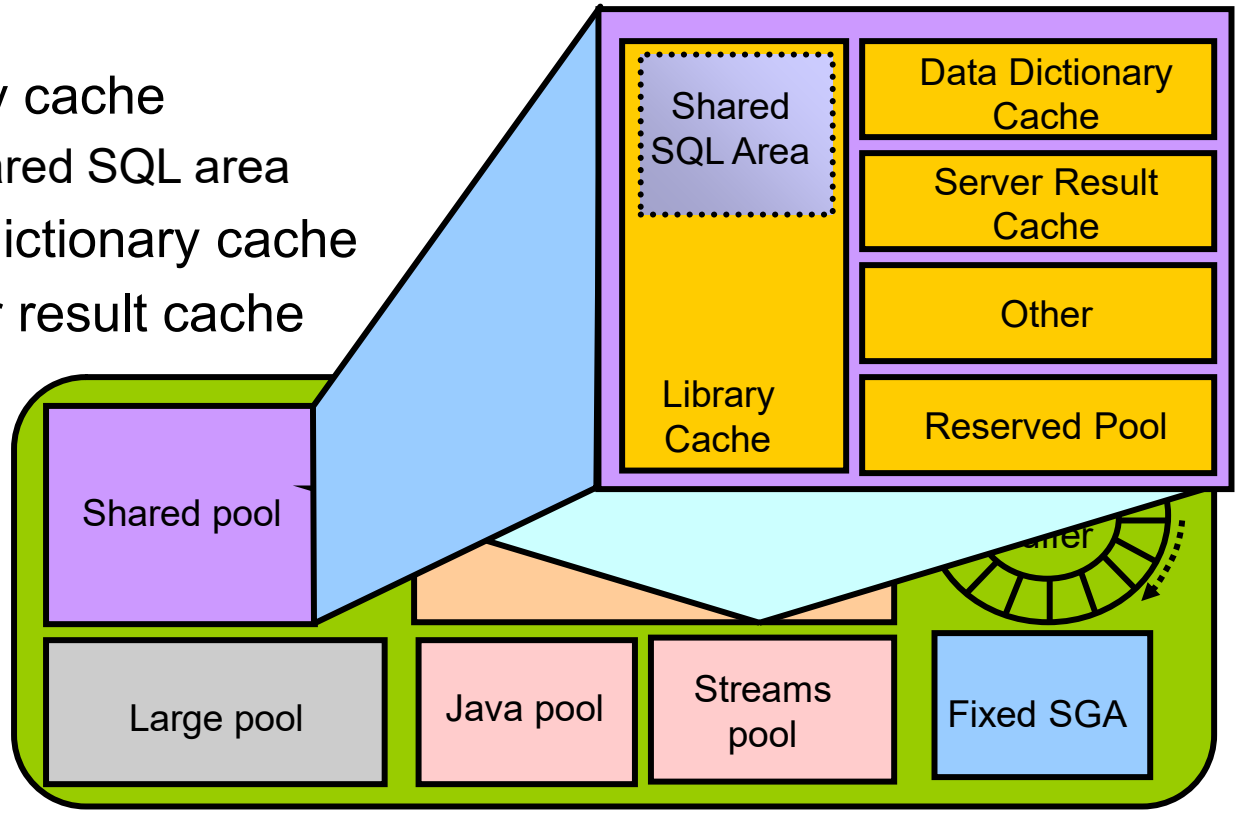


# Oracle Database Memory Structures



# Shared Pool

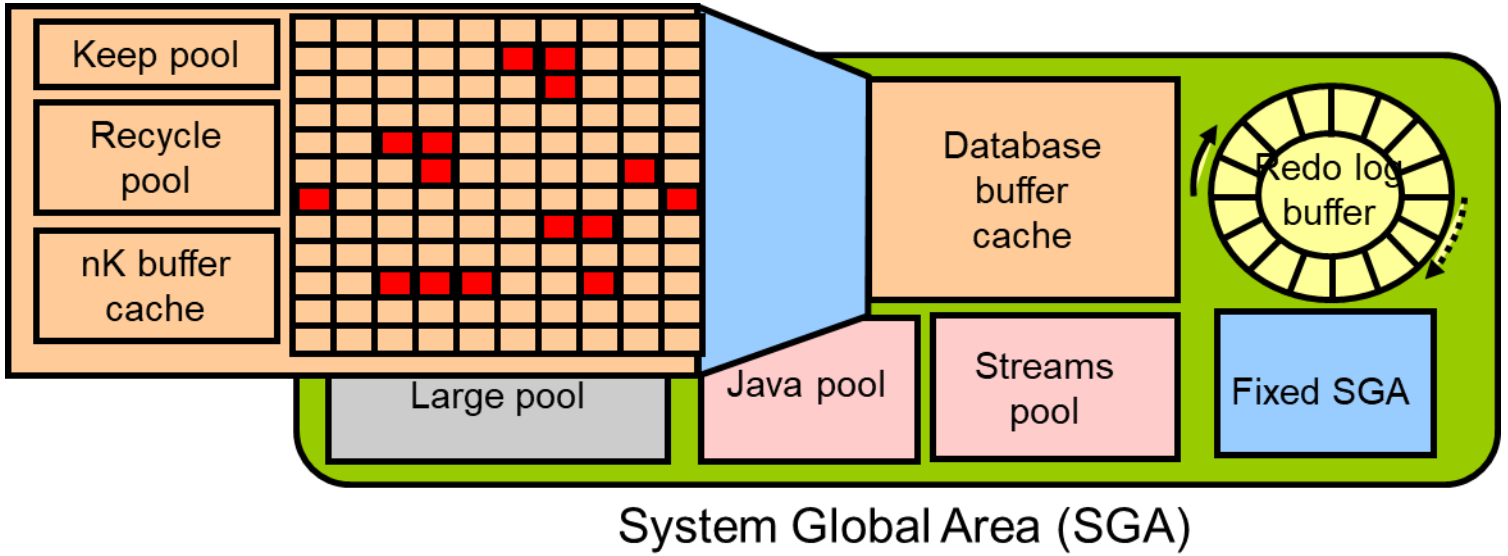
- Is a portion of the SGA
- Contains:
  - Library cache
    - Shared SQL area
  - Data dictionary cache
  - Server result cache




System Global Area (SGA)

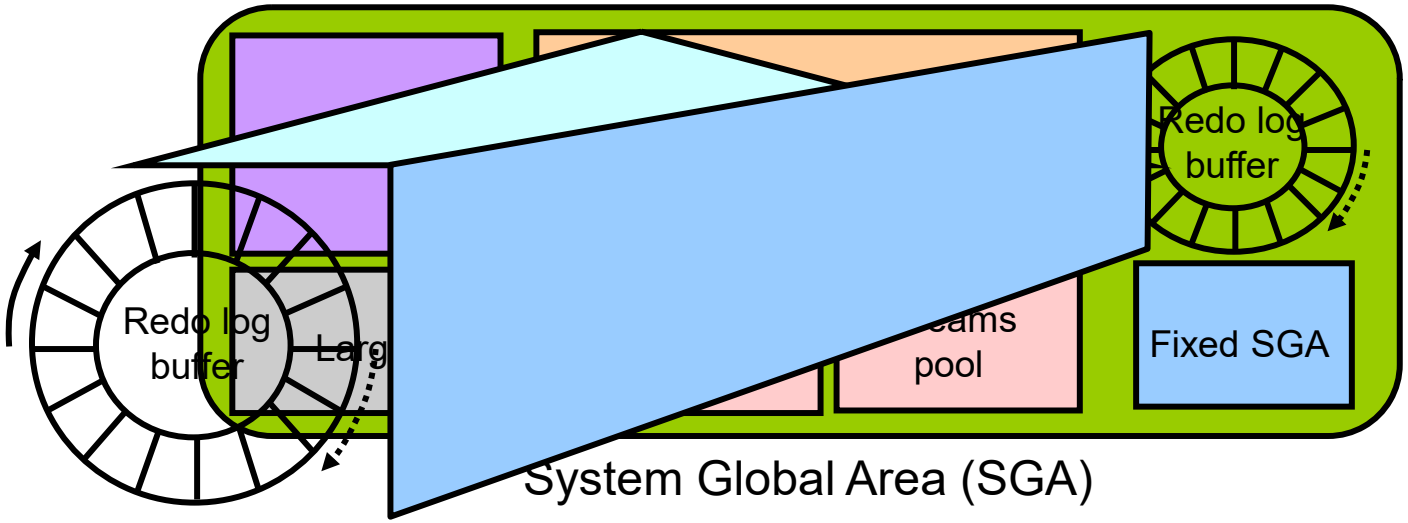
# Database Buffer Cache

- Is part of the SGA
- Holds copies of data blocks that are read from data files
- Is shared by all concurrent users



# Redo Log Buffer

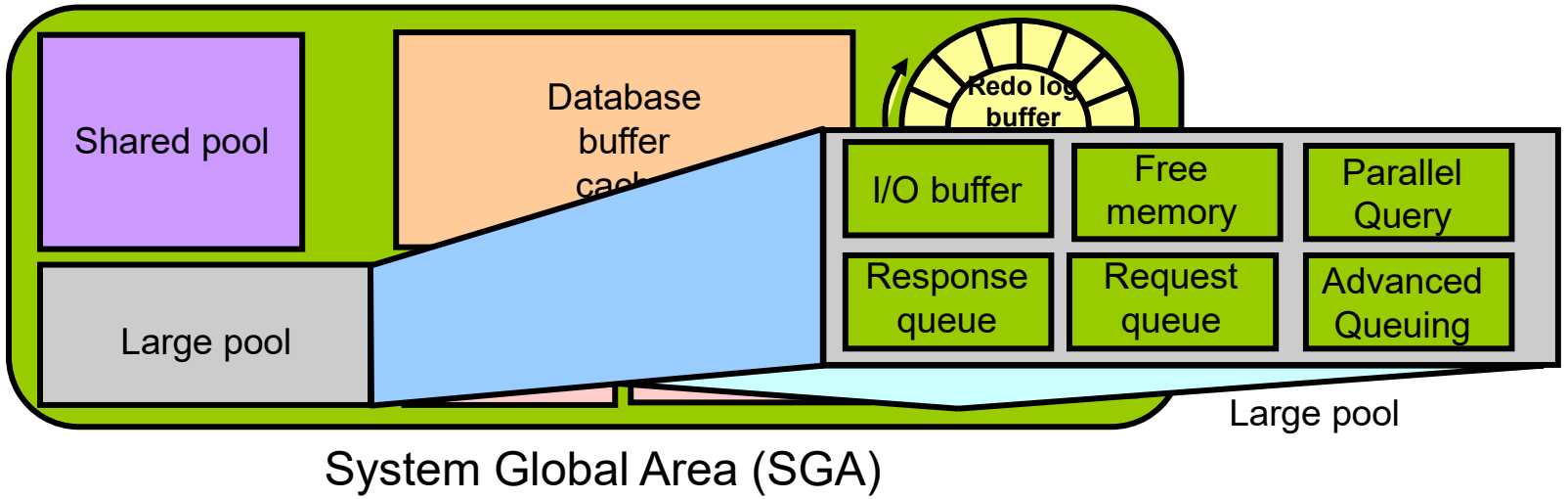
- Is a circular buffer in the SGA
- Holds information about changes made to the database
- Contains redo entries that have the information to redo changes made by operations such as DML and DDL



# Large Pool

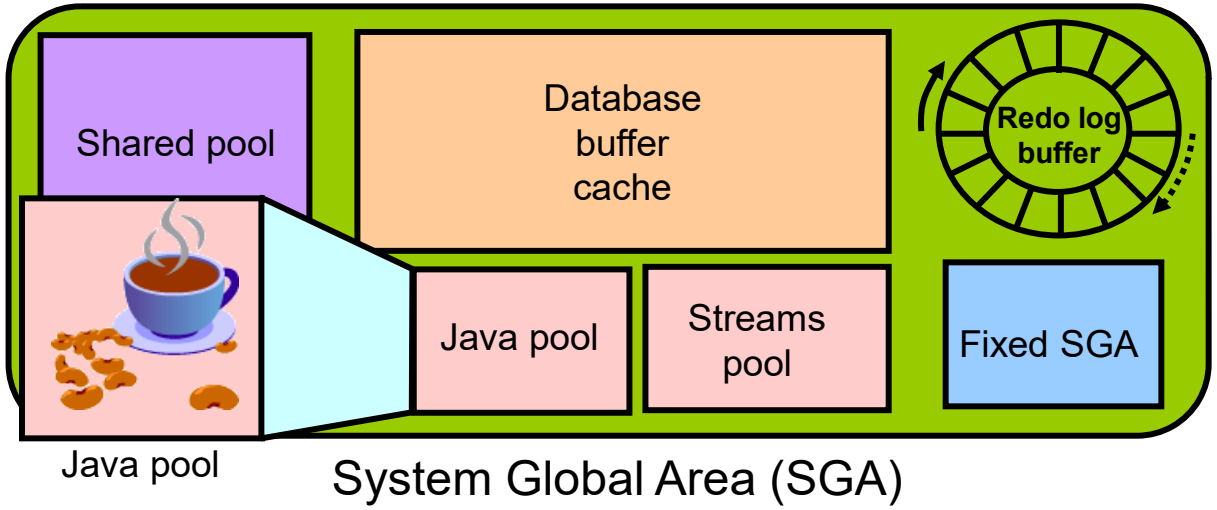
Provides large memory allocations for:

- Session memory for the shared server and the Oracle XA interface
- I/O server processes
- Oracle Database backup and restore operations



# Java Pool

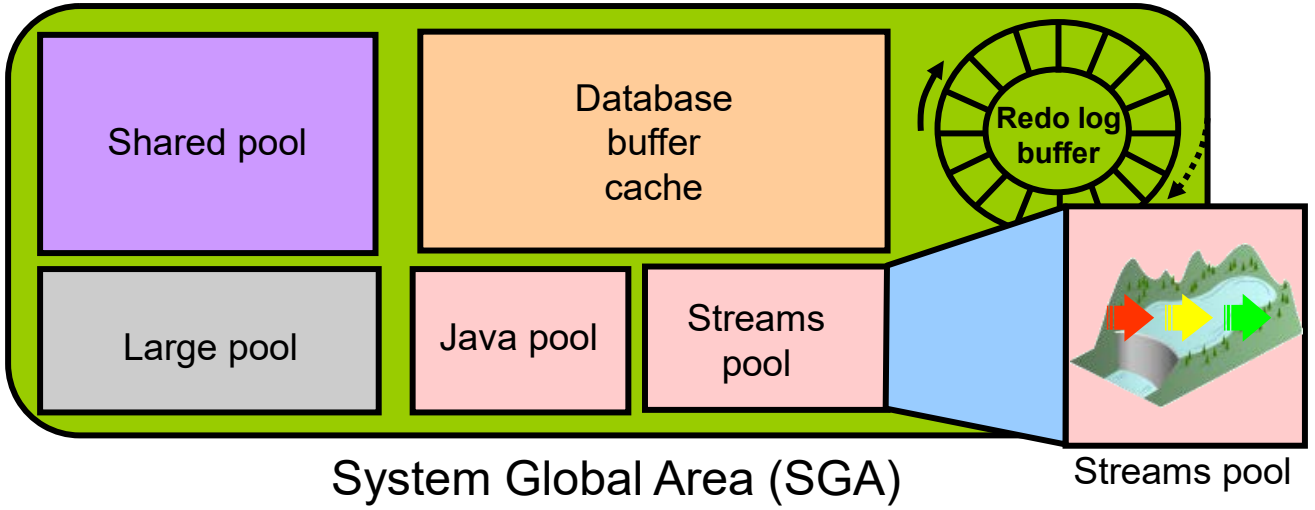
Java pool memory is used to store all session-specific Java code and data in the JVM.



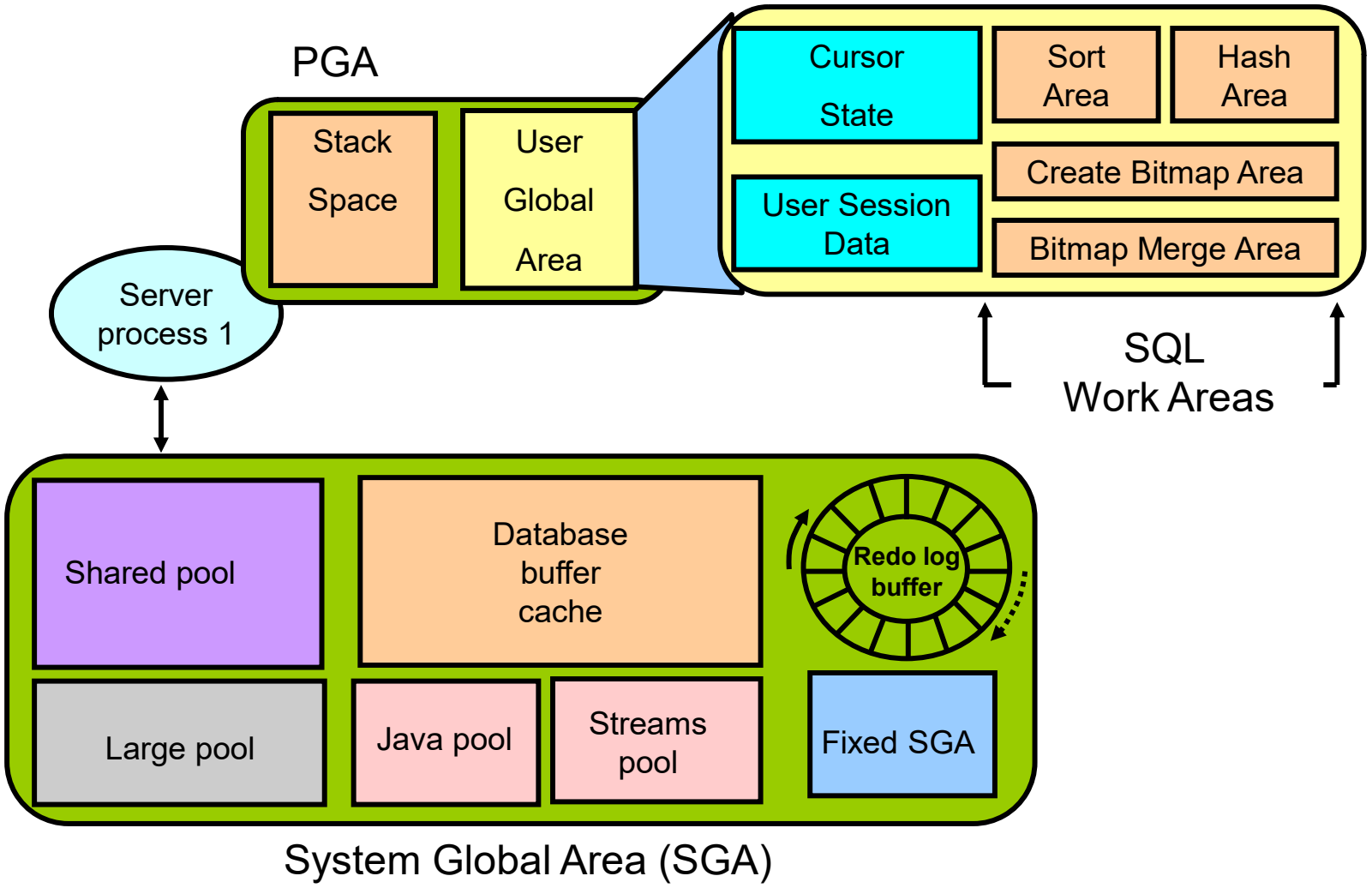
# Streams Pool

Streams pool memory is used exclusively by Oracle Streams to:

- Store buffered queue messages
- Provide memory for Oracle Streams processes



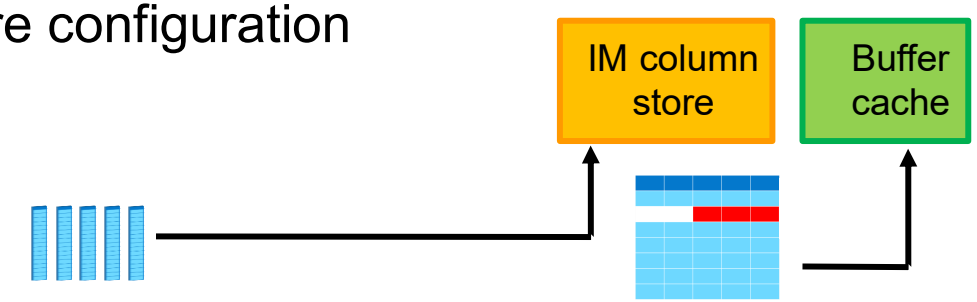
# Program Global Area (PGA)



12.1.0.2

# In-Memory Column Store: Introduction

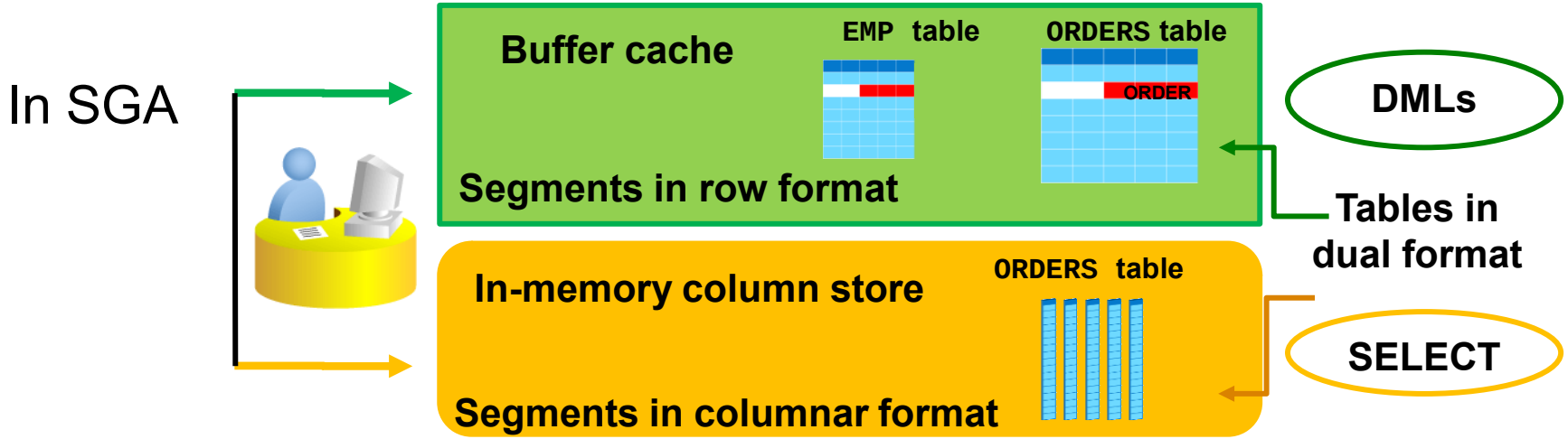
- Instant query response:
  - Faster queries on very large tables on any columns (100x)
  - Use of scans, joins, and aggregates
  - Without indexes
  - Best suited for analytics: few columns, many rows
- Faster DML: Removal of most analytics indexes (3 to 4x)
- Full application transparency
  ORACLE E-BUSINESS SUITE
  ORACLE FUSION APPLICATIONS
  ORACLE SIEBEL
  ORACLE JD EDWARDS
  ORACLE PEOPLESOFT
- Easy setup:
  - In-memory column store configuration
  - Segment attributes



12.1.0.2

# In-Memory Column Store: Overview

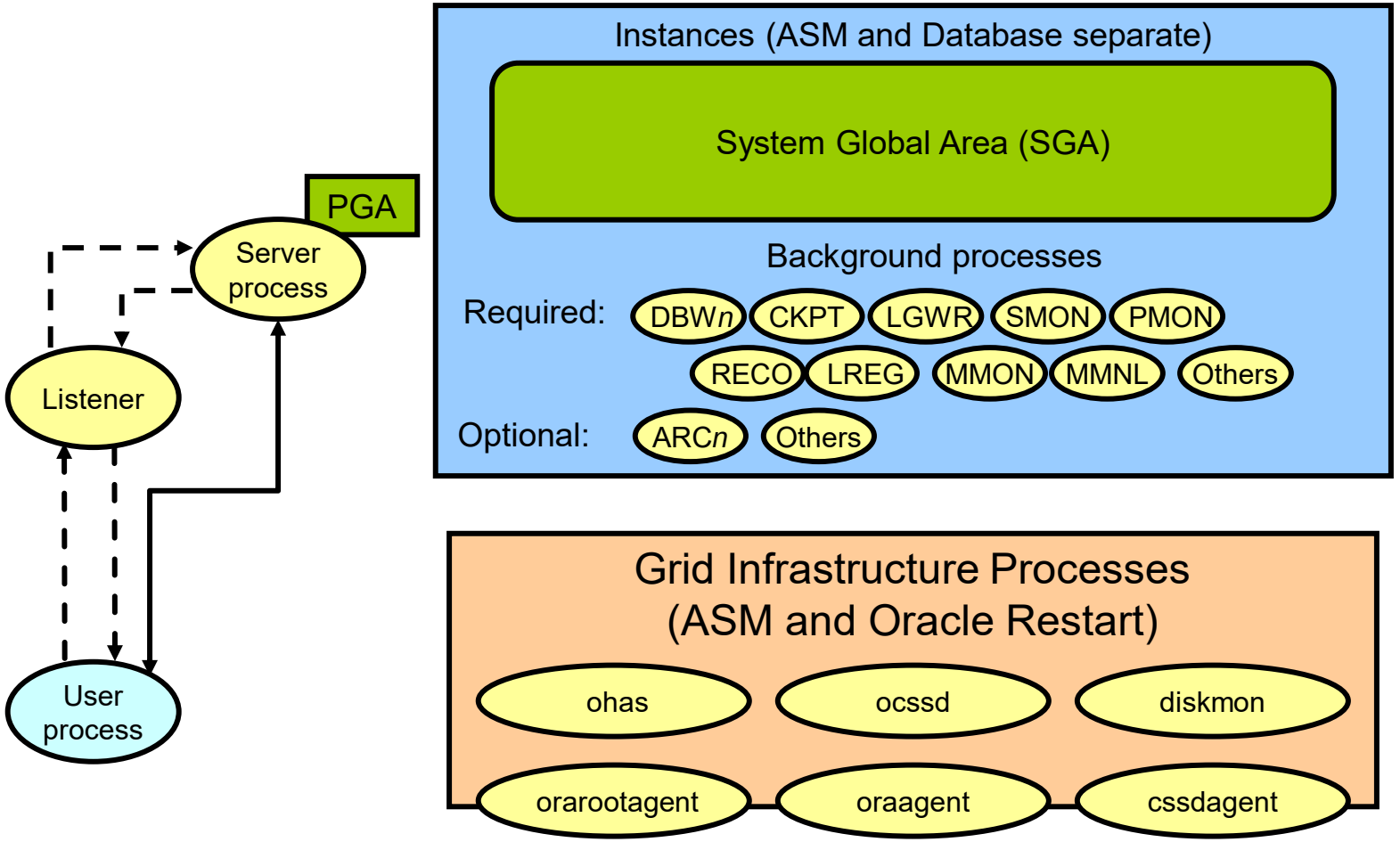
- A new pool in the SGA: In-Memory column store
  - Segments populated into the IM column store are converted into a columnar format.
  - In-Memory segments are transactionally consistent with the buffer cache.
- Only one segment on disk and in row format



# Process Architecture

- User process
  - Is the application or tool that connects to the Oracle database
- Database processes
  - Server process: Connects to the Oracle instance and is started when a user establishes a session
  - Background processes: Are started when an Oracle instance is started
- Daemon / Application processes
  - Networking listeners
  - Grid Infrastructure daemons

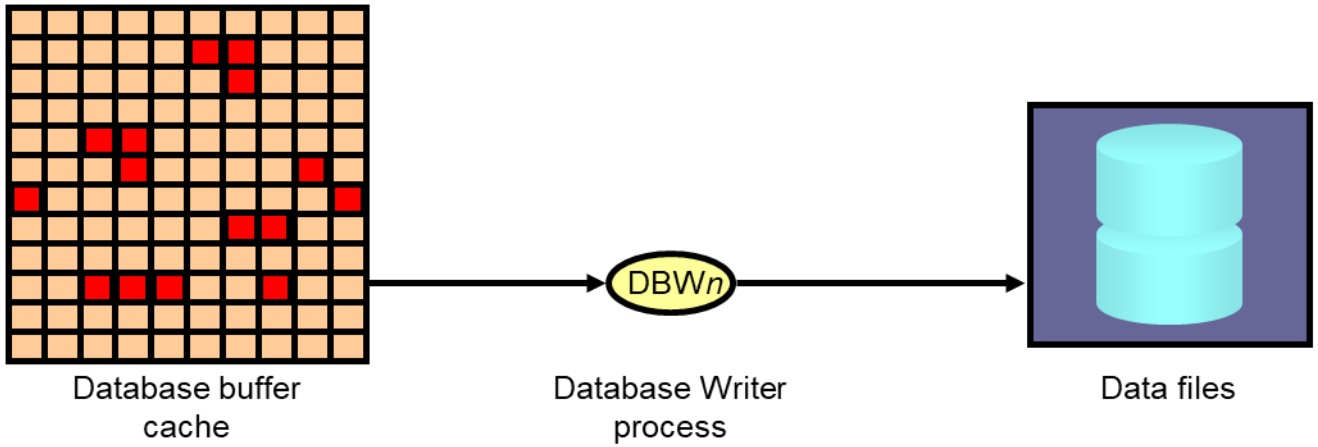
# Process Structures



# Database Writer Process (DBWn)

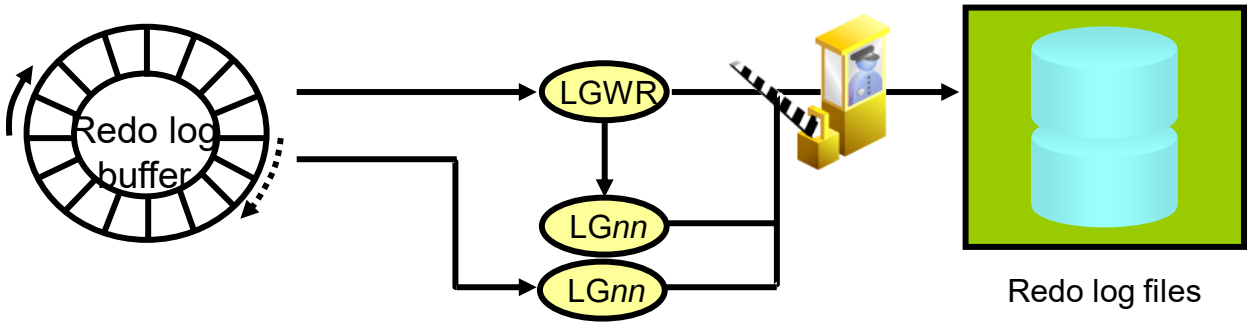
Writes modified (dirty) buffers in the database buffer cache to disk:

- Asynchronously while performing other processing
- To advance the checkpoint



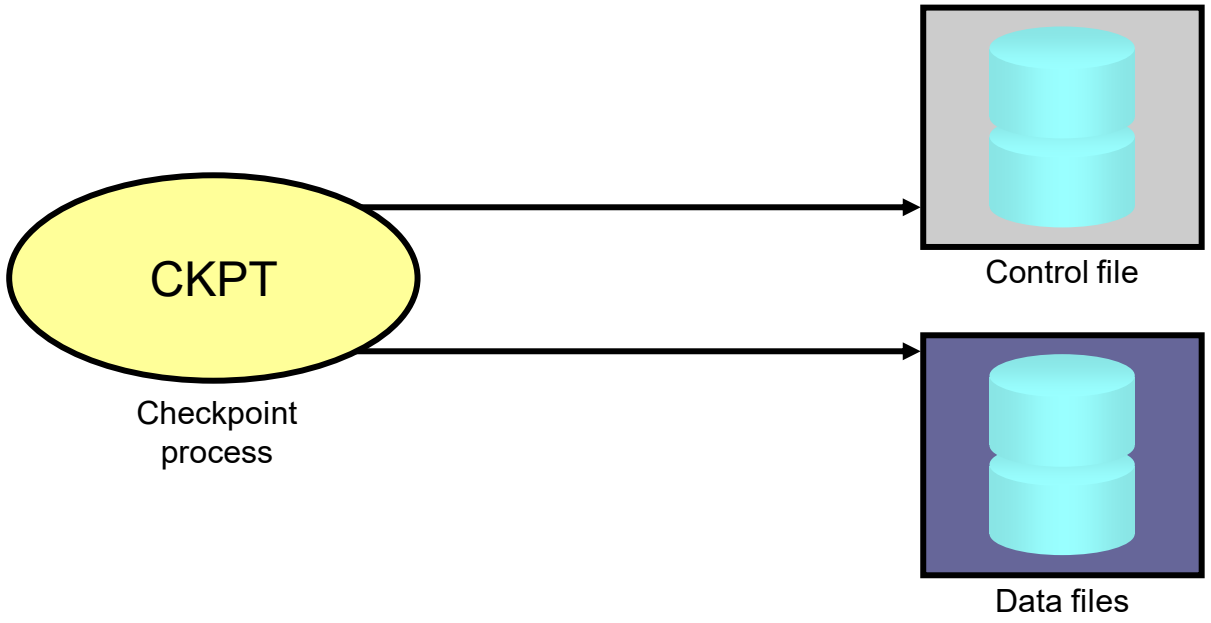
# Log Writer Process (LGWR)

- Writes the redo log buffer to a redo log file on disk
  - When a user process commits a transaction
  - When an online redo log switch occurs
  - When the redo log buffer is one-third full or contains 1 MB of buffered data
  - Before a DBW*n* process writes modified buffers to disk
  - When three seconds have passed since the last write
- Serves as coordinator of LG*nn* processes and ensures correct order for operations that must be ordered



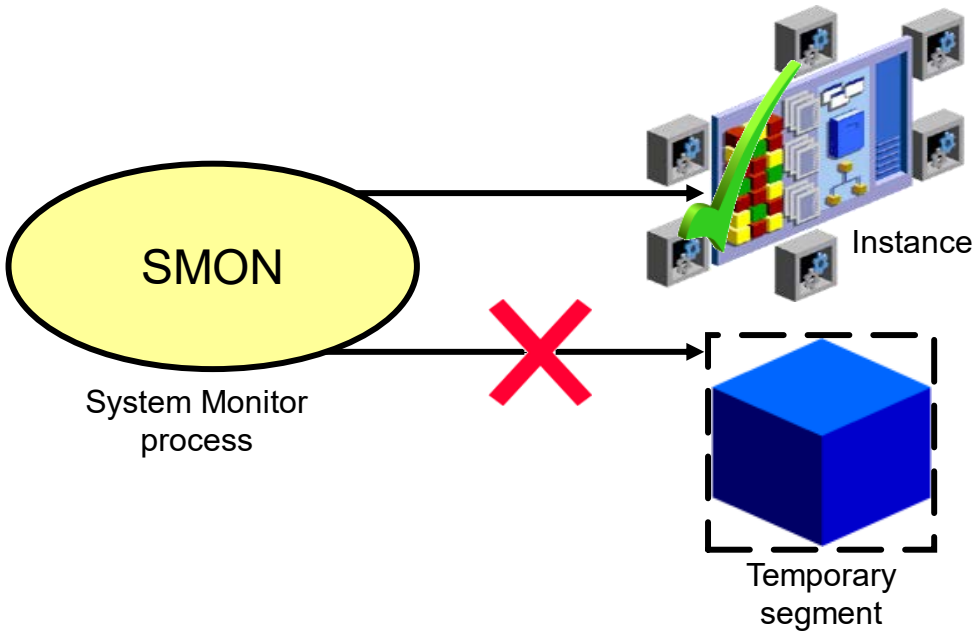
# Checkpoint Process (CKPT)

- Records checkpoint information in
  - Control file
  - Each data file header
- Signals DBW*n* to write blocks to disk



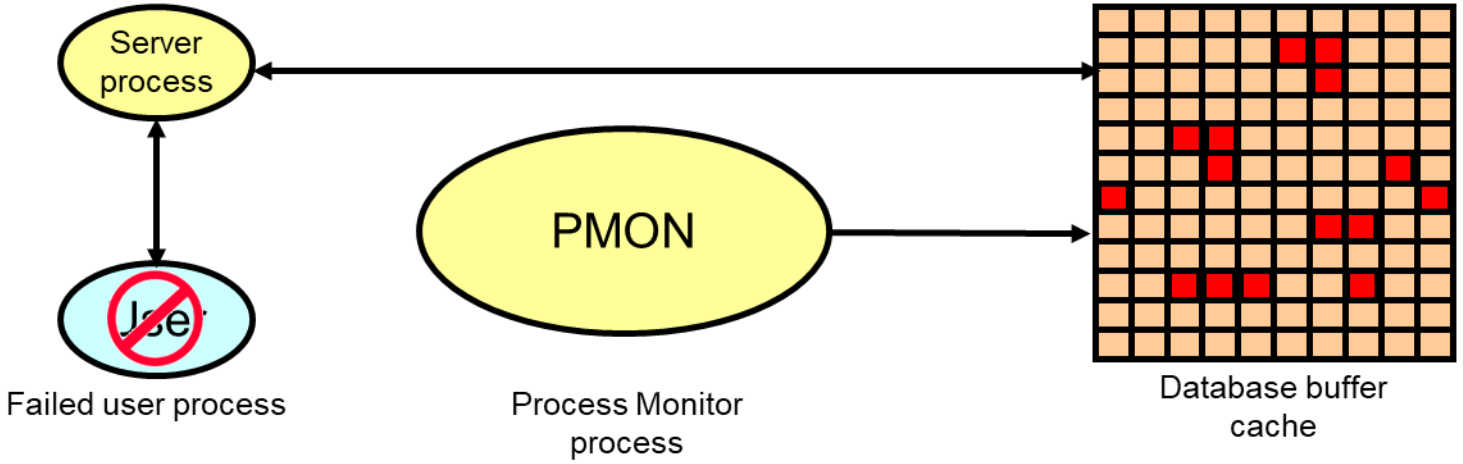
# System Monitor Process (SMON)

- Performs recovery at instance startup
- Cleans up unused temporary segments



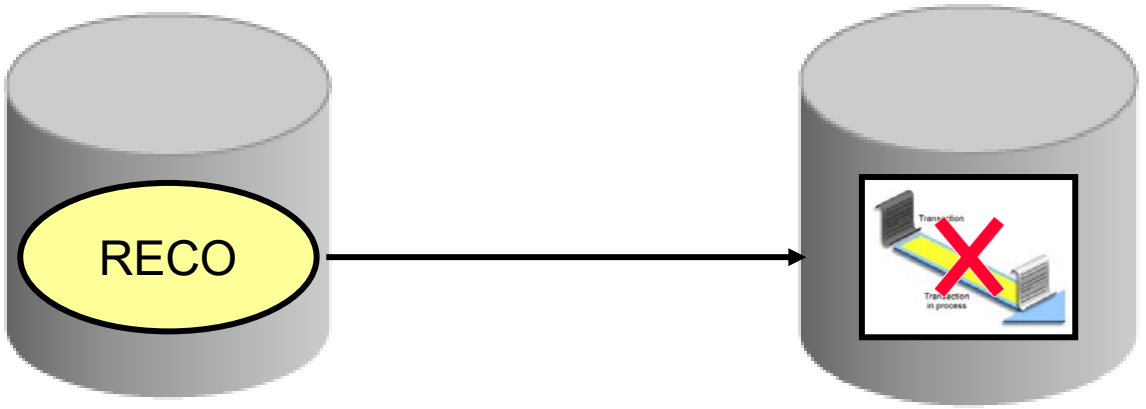
# Process Monitor Process (PMON)

- Performs process recovery when a user process fails
  - Cleans up the database buffer cache
  - Frees resources that are used by the user process
- Monitors sessions for idle session timeout



# Recoverer Process (RECO)

- Used with the distributed database configuration
- Automatically connects to other databases involved in in-doubt distributed transactions
- Automatically resolves all in-doubt transactions
- Removes any rows that correspond to in-doubt transactions

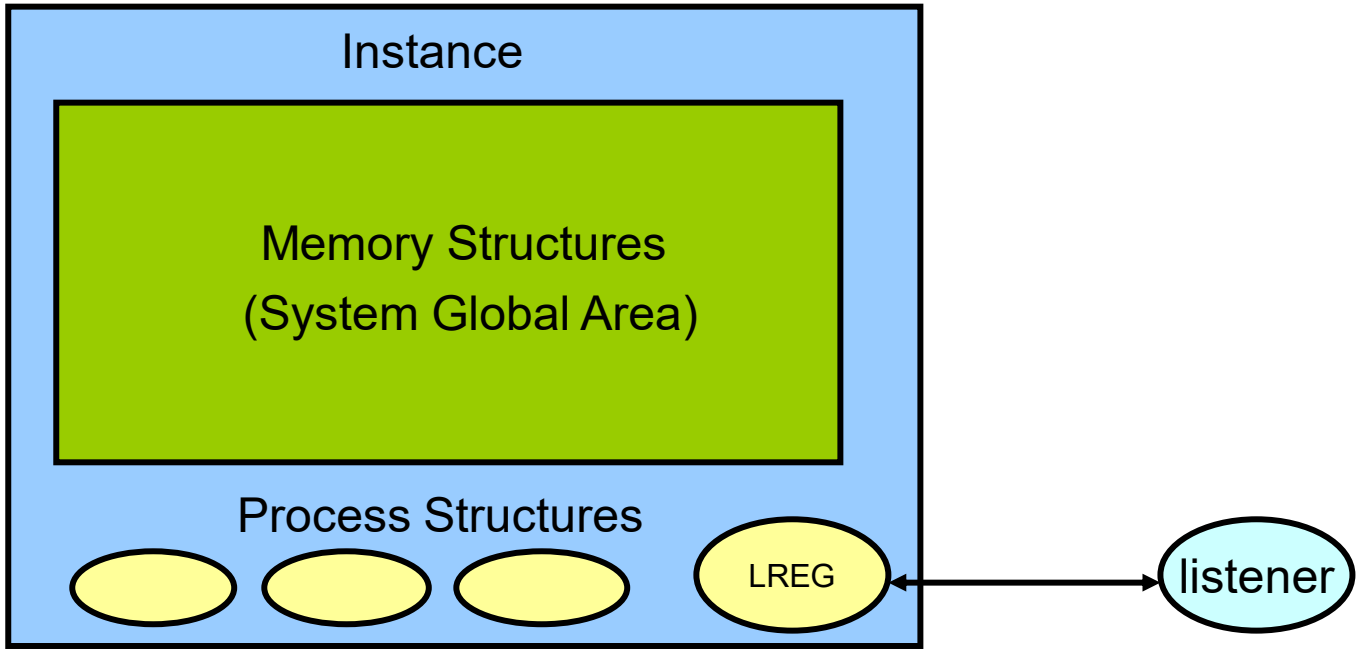



Recoverer process
in database A

In-doubt transaction
in database B

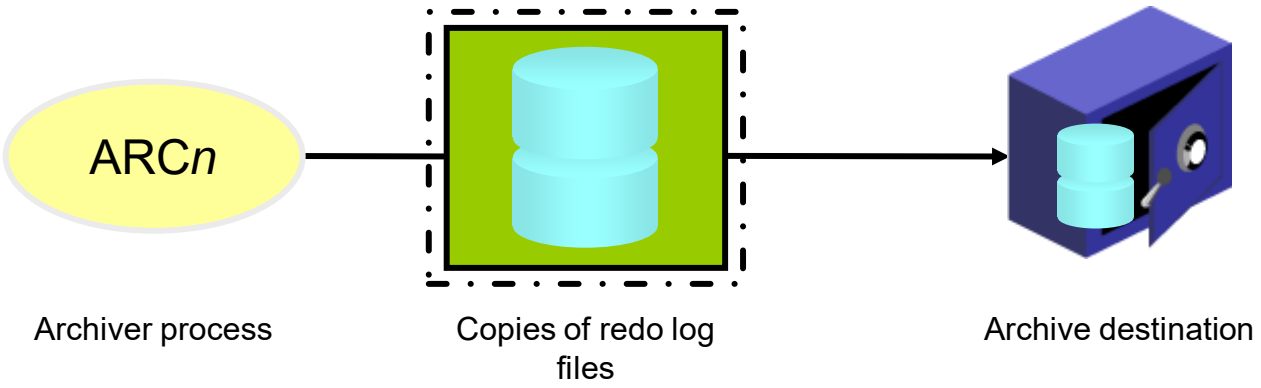
# Listener Registration Process (LREG)

Registers information about the database instance and dispatcher processes with the Oracle Net Listener

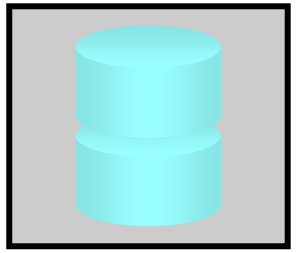


# Archiver Processes (ARC*n*)

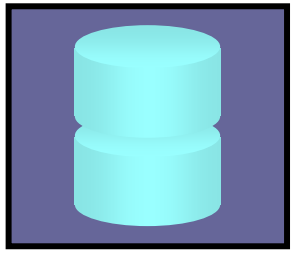
- Copy redo log files to a designated storage device after a log switch has occurred
- Can collect transaction redo data and transmit that data to standby destinations



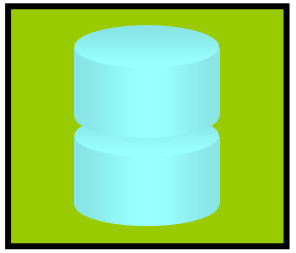
# Database Storage Architecture



Control files

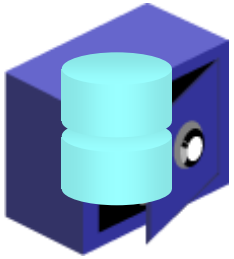


Data files

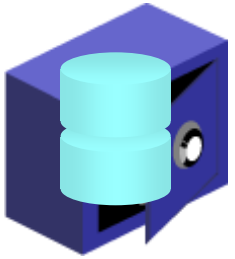


Online redo log files

Parameter file

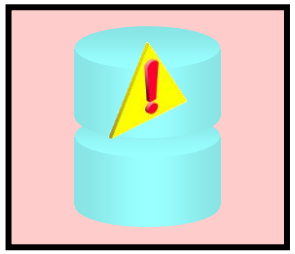


Backup files



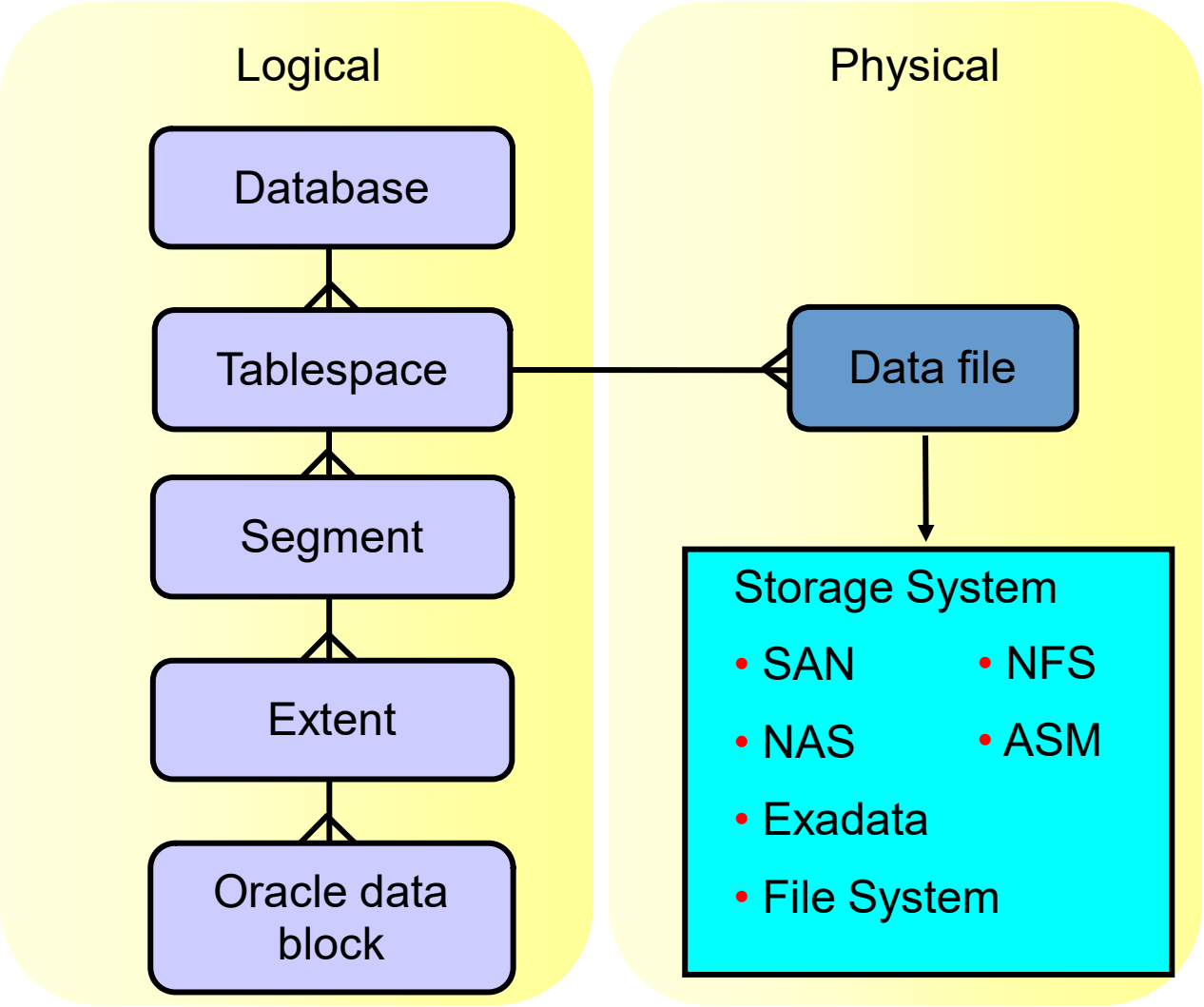
Archived redo log files

Password file



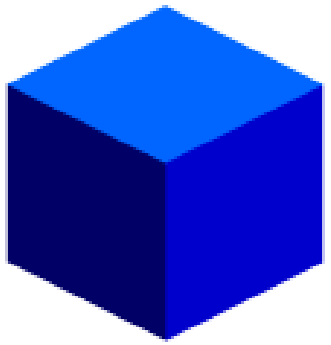
Alert log and trace files

# Logical and Physical Database Structures



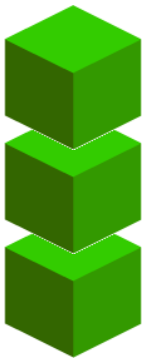
# Segments, Extents, and Blocks

- Segments exist in a tablespace.
- Segments are collections of extents.
- Extents are collections of data blocks.
- Data blocks are mapped to disk blocks.



Segment

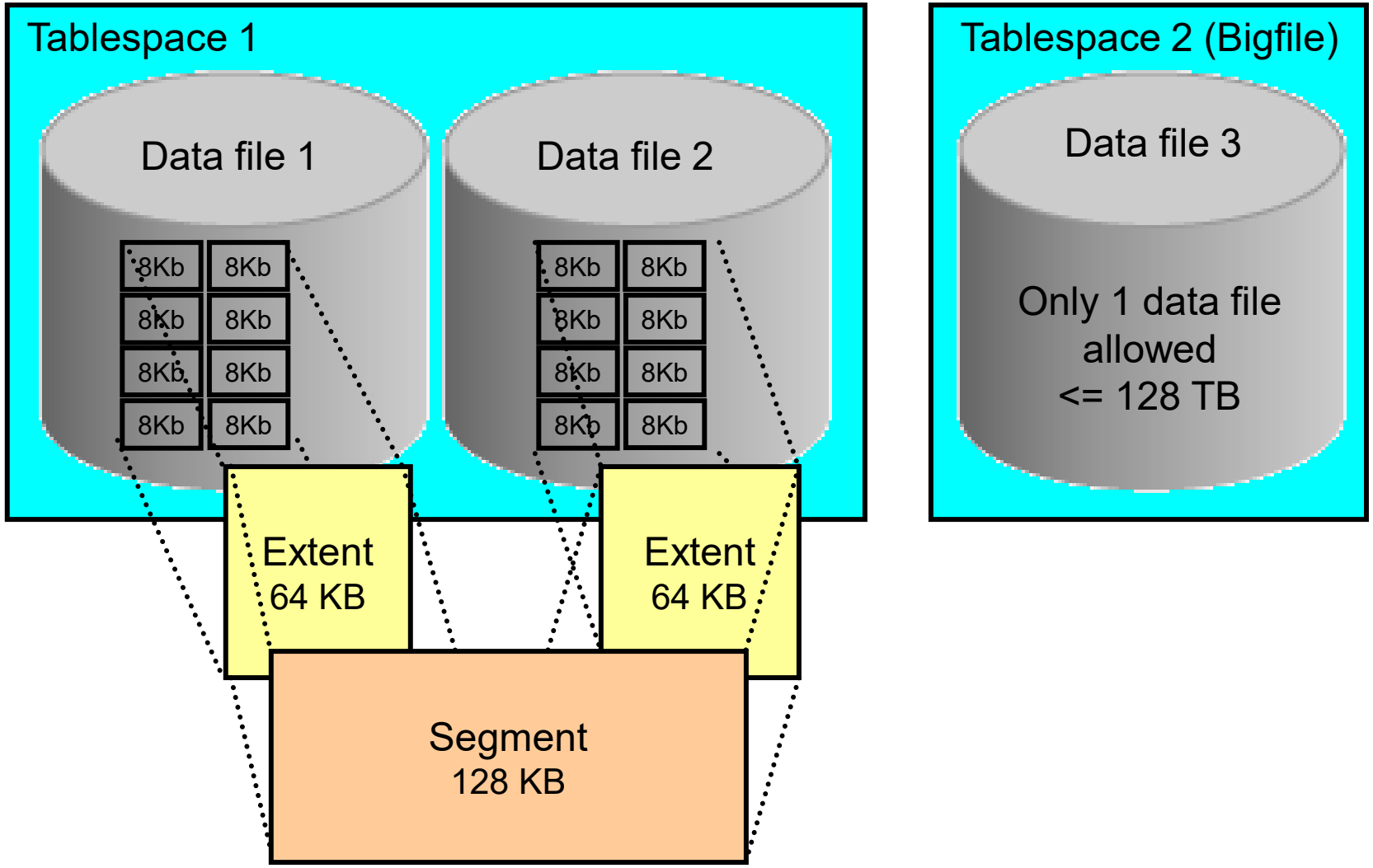
Extents



Data blocks

Disk blocks (File System Storage)

# Tablespaces and Data Files



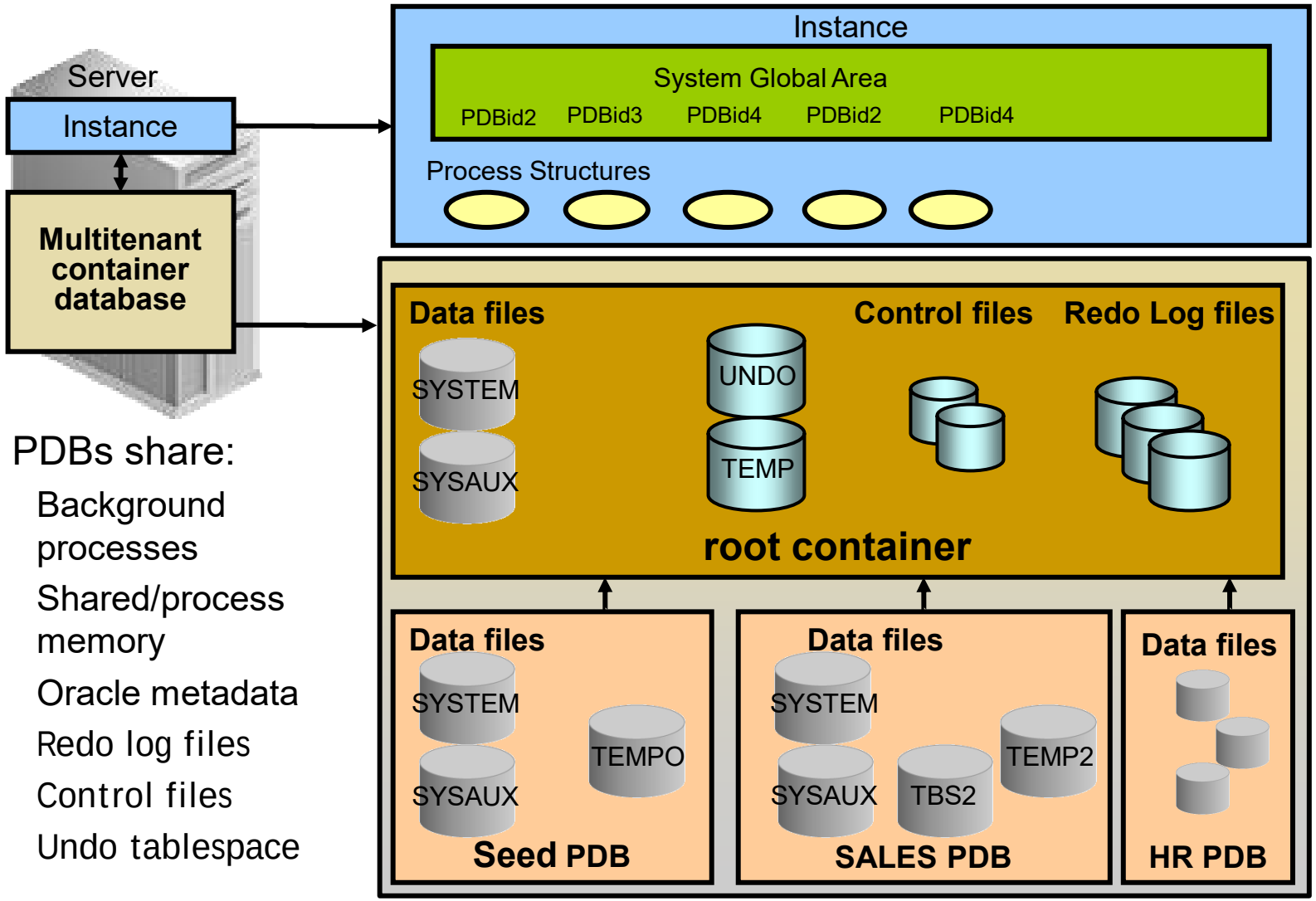
# SYSTEM and SYSAUX Tablespaces

- The SYSTEM and SYSAUX tablespaces are mandatory tablespaces that are created at the time of database creation. They must be online.
- The SYSTEM tablespace is used for core functionality (for example, data dictionary tables).
- The auxiliary SYSAUX tablespace is used for additional database components.
- The SYSTEM and SYSAUX tablespaces should not be used for application data.

# Oracle Container Database: Introduction

- *Pluggable database*: Is a set of database schemas that appears logically to users and applications as a separate database
- *Multitenant container database*: Has a database instance and database files at the physical level
- All pluggable databases share:
  - Background processes
  - Shared/process memory
  - Oracle metadata

# Multitenant Architecture

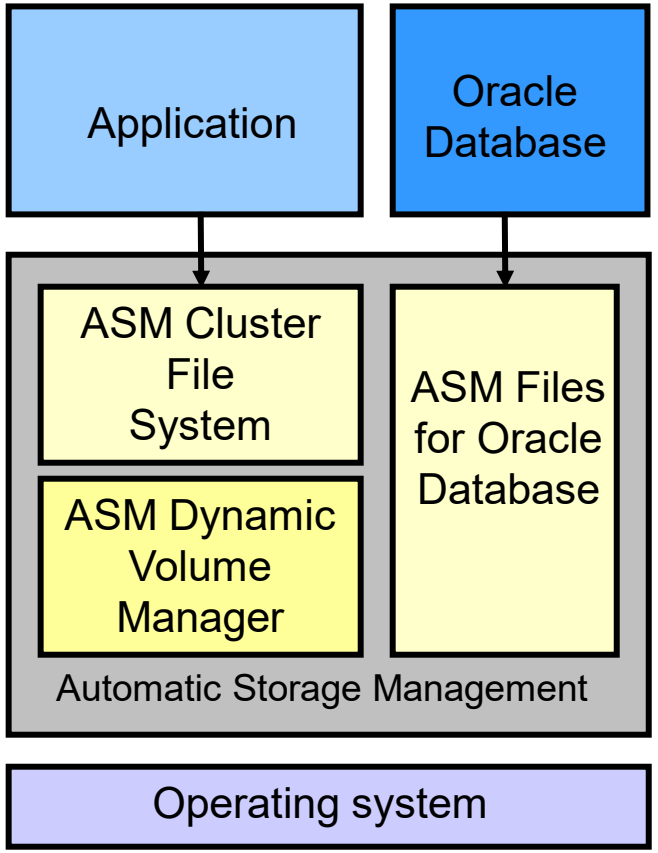



All PDBs share:

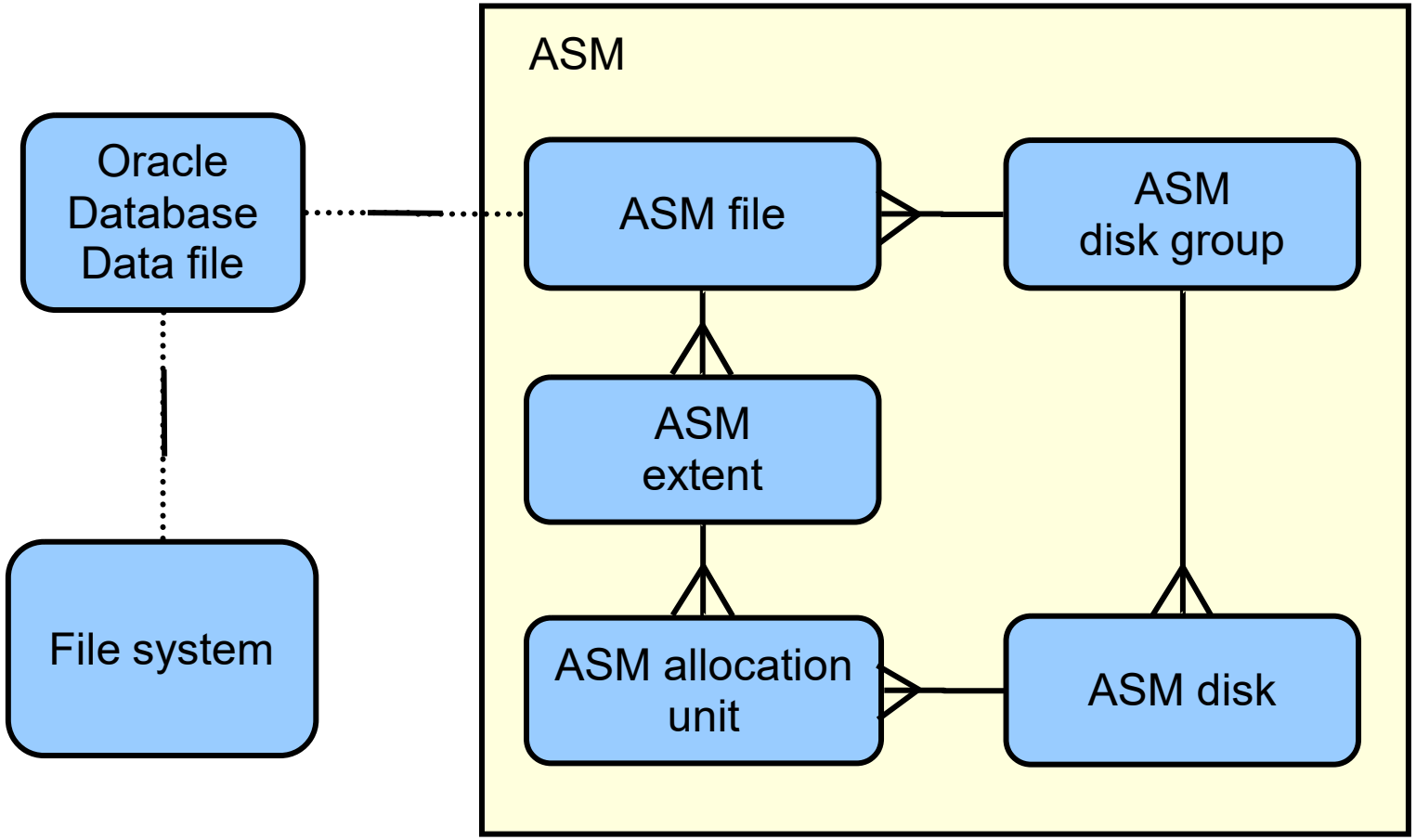
- Background processes
- Shared/process memory
- Oracle metadata
- Redo log files
- Control files
- Undo tablespace

# Automatic Storage Management

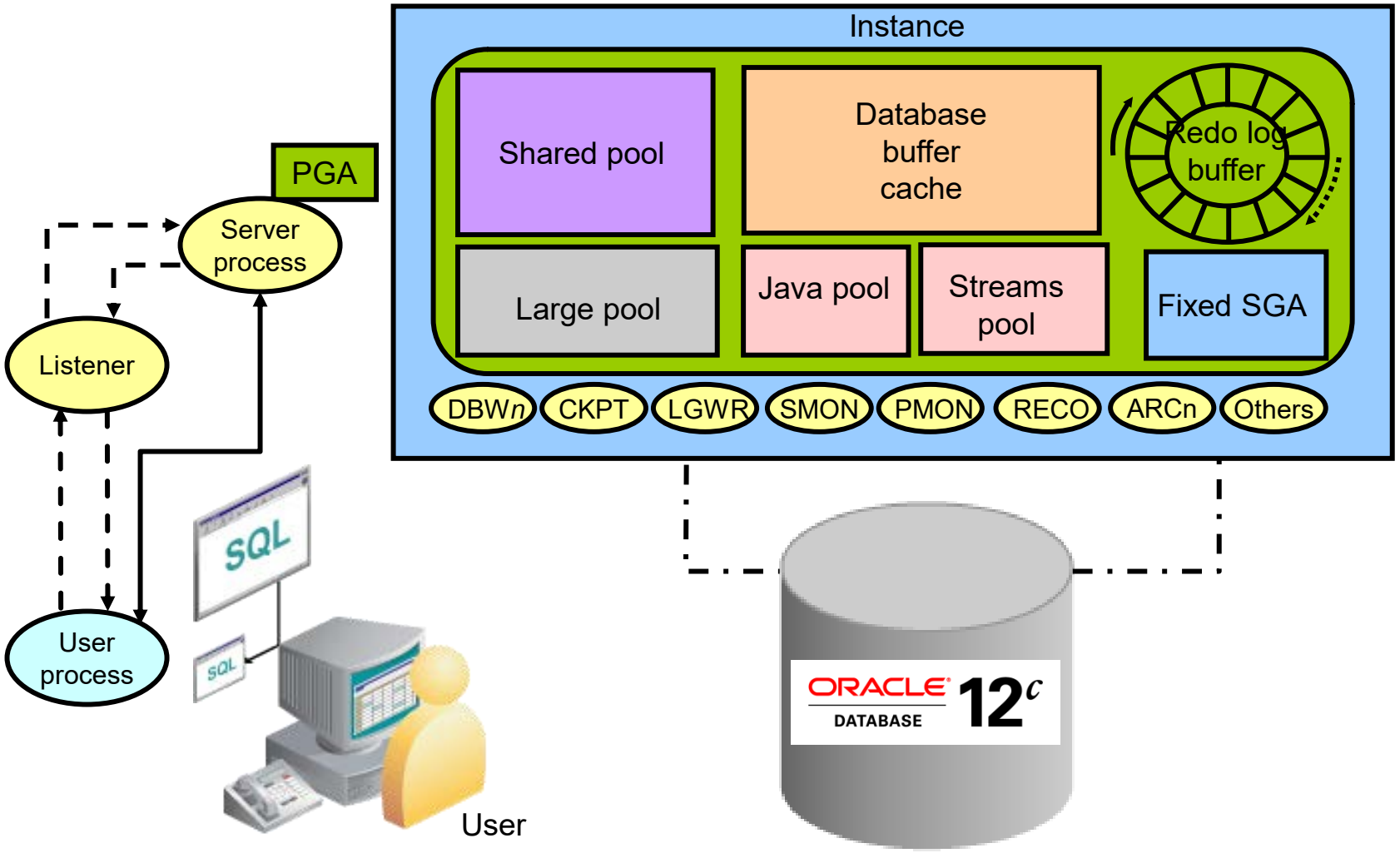
- Is a portable and high-performance cluster file system
- Manages Oracle database files
- Manages application files with ASM Cluster File System (ACFS)
- Spreads data across disks to balance load
- Mirrors data in case of failures
- Solves storage management challenges



# ASM Storage Components



# Interacting with an Oracle Database: Memory, Processes, and Storage



# Summary

In this lesson, you should have learned how to:

- List the major architectural components of Oracle Database
- Explain memory structures
- Describe background processes
- Correlate logical and physical storage structures
- Describe pluggable databases
- Describe ASM storage components
- Describe Oracle Multitenant Container Database technology

PEOPLE COME FIRST
INFORMATIKAI SZAKÉRTŐK EGYESÜLETE

# Mi alapján választunk a relációs adatbázisok közül?

# A relációs adatbázisok kötelezően elvárható és opcionális képességei

# Mi alapján választunk a relációs adatbázisok közül?

- A szoftver ára és a működési sebesség (performancia) messze nem az egyedüli kritériumok.
- És még ezek a kritériumok is igen összetettek – sok különböző aspektusuk és összetevőjük van.
- 1-1 ilyen választásnak jellemzően évtizedes igen széleskörű következményei vannak egy szervezet (pl. egy cég) működésére.
- Befolyásolja majd ez a személyzeti kérdéseket is, meg a szóba jöhető (kiválasztható) szoftver-rendszerek halmazát is.
- A kérdés azért is rendkívüli fontosságú, mert gyakran az informatikára elköltött pénzek legnagyobb részét is ez képezi.

# 1. Azok a képességek, amelyekkel mindegyik relációs adatábiskezelőnek rendelkeznie kell

- Táblák, nézetek
- SQL nyelv: DDL, DML, Query
- Tranzakciókezelés
- ACID képességek:
  - Atomi tranzakciók (Atomicity)
  - Konzisztencia (Consistency)
  - Izoláció (Isolation)
  - Tartósság (Durability)
- Kényszerek
- Adatszótár
- Optimalizáló (automatikus végrehajtási terv generátor)
- …

# 2. Az adatbáziskezelő rendszer költsége

- Licenszköltség: jellemzően a felhasználók számától vagy a hardver méretétől függ.
- A licenszben különböző szoftverkiszerelések létezhetnek és különböző opcionális elemek.
- Szoftveres támogatás (support) éves díja
- Saját gépeken való használat licenszdíja, vagy felhőből bérelt szolgáltatás bérleti díja
- Esetleges adatvesztésből fakadó károk: kicsiny az esélye (de nem nulla), viszont hatalmas az üzleti kár
- Tervezett és tervezetlen állásidőből fakadó üzleti károk
- A szükséges hardver költsége
- Az üzemeltető személyzet költségei (bérek, képzés...)

# 3. Kiforottság, megbízhatóság

- Sajnos minden szoftver bug-os
- Kockázatos olyan adatbáziskezelőt választani, amelyben mi fogjuk először megtapasztalni a hibát.
- A jó szoftver az, amelyet már mások is alkalmaztak ugyanolyam célokra, ugyanolyam körülmények között.
- Ha bármi kételyünk van, az interneten találunk róla valami leírást.
- Olyan az adatbáziskezelő, mint a jó bor: minél öregebb, annál jobb.

# 4. Elterjedtség

- A jó adatbáziskezelőt rajtunk kívül sok száz másik munkahelyen használják (a szomszédban is ez van).
- Ha szükség van egy új, de tapasztalt munkatársra, könnyen találunk.
- Ha szükség van egy konzultánsra, megtaláljuk.
- Könnyű, szinte folyamatos a tapasztalatcsere.
- Ha szakmai segítségre, esetleg csak tanácsra van szükségünk, azt több helyről is megkaphatjuk.

# 5. Szabványok betartása

- A relációs adatbázisok de-facto szabványa az SQL.
- Ezen belül is előny, ha minél inkább betartja a gyártó az SQL szabvány részleteit is.
- A szabványtól eltérő (azon túlmutató) utasítások ugyan hasznosak lehetnek, de kockázatosak is:
  - Később nehezebb lesz adatbáziskezelőt váltani
  - Később megszűnhet a támogatásuk

# 6. A támogatott programozási nyelvek

- A jó adatbáziskezelő rendszernek minél több nyelvet kell támogatnia
- Java nyelvhez: JDBC
- C, C#, .NET támogatás szinte kötelező.
- PHP támogatás is előny.
- Cobol, Fortran, egyéb klasszikus programozási nyelvek támogatása: beágyazott SQL („embedded SQL”) technológiával.

# 7. Hardver és operációs rendszer támogatása

- Sok architektúrát támogat: Intel, SUN SPARC, HP PA-RISC, és különböző IBM hardverarchitektúrákon is működik.
- Támogatja a virtualizációt elméletben és árazásban is.
- Előny, ha működik Linux, Windows, Solaris, AIX, HP-UX, esetleg VMS operációs rendszereken is.
- A későbbi platformváltások legyenek könnyűek.
- „Embedded Database”: egybeolvad a felhasználói program és az adatbáziskezelő rendszer. Jellemzően nincs külön adatbázis, amelyet felügyelnünk kellene.
- „Mobile Database”: okostelefonokon, PDA-kon futó rendszerek.

# 8. Saját gépterem, vagy felhő

- Idővel egyre jelentősebb előnnyé válik majd az, hogy nemcsak a saját géptermünkben működtethetjük, hanem bérelhetünk a felhőben is ilyen szolgáltatást: „Platform as a Service”, vagyis PaaS.
- A PaaS nemcsak elméleti lehetőség kell hogy legyen, hanem a gyakorlatban is olajozottan kell működnie.

# 9. Adattípusok széleskörű támogatása

- Az adatbázisokban manapság már nemcsak szövegeket, számokat és dátumokat tárolunk.
- Képek, tetszőleges szöveg, térinformatikai adatok, stb.
- XMLType, JSON
- User Defined Data Type
- Új objektumtípusok és objektumok tárolása (attribútumokkal és metódusokkal).

# 10. Karakterkészlet

- A Unicode támogatása szinte kötelező.
- A Unicode legújabb szabványa 10.0. Ebben 136.000 jelből áll a „repertoár” (régen ezt „Character Set”-nek nevezték volna, de ez most nem polkorrekt).
- A kódolást vagy UTF-8, vagy UTF-16 (ennek a régebbi, lebutított változata az UCS-2).
- A jó adatbáziskezelő rendszerben lehetséges a kódkonverzió az egyéb (régi) karakterkészletekről Unicode-ra.
- A jó adatbáziskezelőben ez a konverzió állásidő nélkül vagy rövid állásidővel elvégezhető.
- A jó adatbáziskezelő rendszerben nem nő számottevően a helyigény a Unicode miatt (tehát UTF-8 választható).

# 11. Elosztott tranzakciókezelés

- Nemcsak egy adatbázison belül kell megbízhatóan kezelnie a tranzakciókat, hanem két adatbáziskezelő rendszer között is.
- A két adatbázis közötti kommunikáció rendszerint két külön kategóriát is jelent:
  - Két azonos adatbáziskezelő rendszer közötti megbízható adatátvitel (pl. Oracle esetén az adatbázis link és a kétfázisú jóváhagyási mechanizmus teszi ezt lehetővé)
  - Két különböző gyártó rendszere között: XA tranzakciók támogatása
- Szükséges a megbízható tranzakciókezelés egy adatbáziskezelő rendszer és egy másik szoftver – mondjuk egy üzenetküldő rendszer („Messaging System”) – között is: rendszerint XA protokoll.

# 12. Procedurális lehetőségek

- Az SQL nyelv nagyszerű, de mégis akadnak esetek, amikor más kéne.
- Szükségünk lehet algoritmusokra is, és ezek adatbázison belüli tárolására.
- Így a feldolgozás az adatok „közelében” történhet.
- Így kiterjeszthető az adatbáziskezelő rendszer funkcionalitása:
  - Maga a gyártó is fejleszthet ilyen kiterjesztéseket
  - A felhasználó kifejlesztheti a saját kiterjesztéseit
  - A kényszerek halmaza is kibővíthető így
  - Az objektumorientált metódusok is ebben a nyelvben íródnak
- Pl. Oracle esetén két ilyen nyelv is rendelkezésre áll: PL/SQL és Java.

# 13. Adatbázis triggerek

- Igen hasznos egy olyan képesség, hogy bizonyos programok maguktól végrehajtódnak (elsülnek) egyes események bekövetkeztekor:
  - Adatok beszúrása előtt (Pre-INSERT triggerek). Ezeket jellemzően a beszúrás ellenőrzéseként használjuk.
  - Adatok beszúrása után (Post-INSERT triggerek). Ezeket gyakran a művelet naplózása érdekében alkalmazzuk.
  - INSERT mellett DELETE és UPDATE esetére is hasznosak a triggerek.
  - Nemcsak DML triggerek létezhetnek, hanem pl. olyanok, amelyek bejelentkezéskor, vagy pl. a teljes rendszer elindulása vagy leállása esetén futnak le.

# 14. A funkcionalitás kiterjesztése SQL-en túlra

- A többmillió adatbázis-felhasználó cégnek (esetleg embernek) gyakran van olyan közös igénye, amely túlmutat az SQL-en, de mégis praktikus. Ilyen pl:
  - Ütemezett feladatok végrehajtása
  - Email-ek küldése
  - Fájlok olvasása/írása
  - Üzenetek küldése és fogadása
  - …
- Amennyiben létezik procedurális lehetőség az adatbázison belül, akkor rendszerint a gyártó biztosít ilyen funkcionális kiterjesztéseket.
- Nyílt forráskódú adatbáziskezelő rendszer esetén ez lehet közösségi fejlesztés is.

# 15. Magas rendelkezésre állás

- A „High Availability” egyre fontosabbá válik.
- Egyre gyakrabban van szükségünk 7*24 órás rendelkezésre állásra. Persze valamiféle állásidőnek manapság még ilyenkor is mindenképpen lennie kell.
- Az állásidőnek két fajtája van:
  - Betervezett (előre bejelentett)
  - Nem betervezett (valamilyen hiba miatt)
- Pl. az „5 kilences” rendelkezésre állás (vagyis a 99.999%-os rendelkezésre állás) azt jelenti, hogy évente kb. 5-6 percet állunk csupán. Ez nagyon nehezen teljesíthető, de nem lehetetlen.
- Oracle esetén pl. a RAC opció a legfőbb magas rendelkezésre állási képesség.
- Problémás a szoftverek frissítése: oprendszer, adatbáziskezelő rendszer, az adatbázis adatszótára…

# 16. Helyreállíthatóság

- Előbb utóbb nemcsak összeomlik egy szoftver, hanem adatvesztés is történhet (pl. lemezhiba miatt).
- Az adatbáziskezelő rendszernek olyan mentési mechanizmus kell, amely biztosítja nemcsak a régi mentésre való visszaállást („Restore”), hanem a legfrissebb állapotba való helyreállást is („Recovery).
- A helyreállítás ideje is kritikus:
  - a jó eset manapság néhány perc
  - a nem túl jó eset több óra
  - Az elfogadhatatlanul lassú helyreállítási idő manapság a több nap
- Fontos, hogy a helyreállítás könnyű és megbízhatóan végrehajtható feladat legyen.
- Azért mindenre kiterjedő abszolút biztonság  nem létezik.

# 17. Katasztrófatűrés

- Nemcsak a ténylegesen várható veszélyekre illik felkészülni, hanem olyan katasztrófa-helyzetekre is, amelyek szinte kizártnak tűnnek:
  - Tűzvész, árvíz, földrengés
  - Terrortámadás
- Aki ilyen ellen is védekezni akar (egyre többen), azok távoli adatbázis-másolatokat szeretnének működtetni.
- Ezt megoldhatja valamilyen hardveres vagy szoftveres távoli tükrözés is, de lehet ez akár az adatbáziskezelő rendszer extra képessége.
- Oracle esetén a DataGuard a katasztrófatűrő megoldás.

# 18. Hatékonyság (gyors működés)

- Szándékosan nem az elsők között került említésre. Sajnos a korai fázisban sokan ez alapján választanak. Egy erősebb hardver ellensúlyozhatja a szoftver hatékonyságát.
- Leginkább a funkcionalitás rovására válik 1-1 adatbáziskezelő rendszer gyorssá.
- Máskor azért gyors mert memóriában dolgozik. Ez persze sérülékenyebbé teszi.
- Ha a sebesség kiemelkedően fontos, akkor is csak a releváns terheléseket vegyük figyelembe.
- Szinte mindegyik adatbáziskezelő rendszer gyártója kozmetikázza a hatékonysági mutatók értékeit.
- Minden újabb verzióról azt mondják, hogy gyorsabb mint a megelőző verzió, közben rendszerint lassabb.

# 19. Skálázhatóság

- A sebesség mellett legalább olyan fontos kérdés, hogy ha növekszik a terhelés, tudjuk-e ehhez igazodva növelni a teljesítményt.
- Ideális a lineáris skálázhatóság lenne: kétszer annyi hardverrel kétszer annyi munka elvégzése.
- A lineáris skálázhatóság szinte csak álom
- Lehet egyre több processzor egy gépen belül, vagy egyre több számítógép egy „cluster”-ben.
- Oracle esetén a RAC lenne a válasz a skálázhatóságra is.

# 20. Adatok titkosítása

- A merevlemez-alapú adatbázisoknál az adatok a merevlemezen fájlokban tárolódnak
- Komoly kockázat, hogy ezeket a fájlokat esetleg ellopja valaki és így jut hozzá az adatokhoz
- Ezt a komoly adatbáziskezelők úgy védik ki, hogy a fájlban már titkosított módon tárolhatnak adatot
- Ilyenkor az INSERT utasítás „titkosítva ír”, és a SELECT utasítás fejti azt vissza.
- Aki tehát SQL művelettel fér hozzá az adatokhoz, azt ez a titkosítás „nem érinti”.
- Ugyanilyen titkosítás létezhet a kliens-szervet kommunikáció során is.
- Oracle esetén ezt „Advanced Encryption Option”-nek nevezik. És sajnos feláras lehetőség.

# 21. A használat auditálása

- Az adatbáziskezelő rendszerek zsargonjában auditálásnak nevezik az adatbázis-használat figyelését biztonsági célokból.
- Azt kell tudnunk, hogy melyik felhasználó mikor milyen műveleteket hajtott végre.
- Az auditálás konfigurálható kell hogy legyen:
  - Opcionálisan eldönthető, hogy legyen-e audit
  - Ha van audit, akkor konfigurálható kell hogy legyen annak a részletessége
- Az auditálás működtetése nem lassíthatja számottevően az adatbáziskezelő rendszert.
- Hatékony elemzési lehetőségek kellenek az „Audit Record”-ok felett.
- Figyelem: ez nemcsak van/nincs kérdés. Lehet auditálni úgy is, hogy az hasznavehetetlen.

# 22. A mentések sokszínűsége

- A jó adatbáziskezelő rendszernek saját mentési program/módszer kell, de legyen alternatíva is.
- Ezzel nemcsak lementhető, hanem egyúttal logikailag ellenőrizhető is az adatok tartalma.
- Előny az, ha ez a program az adatbáziskezelő rendszer belsejébe van beleépítve.
- Hatékony (valószínűleg párhuzamosítható) mentési módszer kell.
- Teljes és inkrementális mentésre is legyen lehetőség.
- A mentés elvégezhető legyen online.
- A mentési katalógus is nagyon hasznos.
- Lemezre és szalagos mentőegységre is lehessen menteni.
- Harmadik fél eszközeivel (módszereivel) is lehessen menteni.

# 23. Milyen segédprogramok léteznek

- A gyakorlatban felmerül az igény különböző nem-SQL feladatok egyszerű és hatékony elvégzésére.
- Ilyenek pl.:
  - Adatok betöltése az adatbázistáblákba:
    - szöveges fix formátumú állományokból
    - CSV („Comma Separated Values”) fájlokból
    - Excell táblákból
    - XML vagy JSON állományokból
  - Adatok igény szerinti áthordozása adatbázistáblákból bináris állományokba és később ezek visszatöltése ugyanabba, vagy másik adatbázisba
- Pl. Oracle esetén ezek az eszközök az SQL*Loader és az Oracle DataPump Export és DataPump Import.

# 24. Szakember általi monitorozás, hangolás lehetősége

- Az adatbázis sebessége ugyan nem a legfontosabb kiválasztási kritérium, de annak mégis hatalmas jelentősége van, ha egy rendszer nagyon precízen monitorozható, és kideríthető róla, hogy hol van a szűk-keresztmetszete
- Ha ezután még át is paraméterezhető a működés, úgy, hogy a szűkkeresztmetszet eltűnjön, vagy csupán enyhüljön, az már főnyeremény.
- Nagyon fontos, hogy ez a monitorozhatóság SELECT utasításokkal történjék. Ezáltal ugyanis harmadik fél is gyártani tud monitoring eszközt (nyílt rendszerek).

# 25. Automatikus monitorozás és hangolás

- Egyre több rendszert működtetünk. Egyes cégeknél többszáz adatbázis van napi használatban. Nem érkeznek a rendszergazdák mindegyikre folyamatosan odafigyelni.
- Nő a „monitoring”, mint tevékenységi kör jelentősége.
- A jó rendszereket nem elsősorban kívülről figyelik („polling”), hanem azok saját magukat figyelik belülről, és riasztanak minket, ha baj van. Erre még nincs egységes szóhasználat, de gyakran „Server Generated Alert System” a neve.
- Az igazán jó rendszerek nemcsak riasztanak, hanem megoldást is javasolnak, sőt esetleg a javasolt változtatást meg is teszik automatikusan.

# 26. A szoftver fejlődésének a képessége és üteme

- Az igények folyamatosan bővülnek és változnak.
- Azt a szoftvert el kell kerülni, amelynek a gyártója nem fejleszti a terméket nagy tempóban.
- Egyes szoftverek nagyon gyenge architektúrális alapokra épültek, és ezért nem tudnak továbbfejlődni.
- Másik probléma lehet, ha a kód már ősrégi, és most már csak nehezen karbantartható.
- Fontos kérdés, hogy milyen programozási nyelvet használtak az adatbáziskezelő rendszer gyártói.
- E téren pl. az Oracle C-ben íródott, ami ma már inkább rossz tulajdonság, mint jó, és a kód jelentős része igen „dohos”. Ezt próbálják ellensúlyozni azzal, hogy rengetegen fejlesztik.

# 27. A szoftvertámogatás minősége

- „Bugs are facts of life” – úgy tűnik, hogy sajnos elkerülhetetlenek.
- Ha már vannak hibák, kritikus kérdés, hogy van-e aki javítsa őket?
- Hányan és hány problémát oldanak meg? A mi problémánkat jellemzően megoldják-e, és ha igen, mennyi idő alatt?
- A megoldási módszer is lényeges, hiszen érzékeny adatokat tárolunk. Ki férhet hozzá a hiba felderítése során?
- A „support” általában borsos áron történik.
- Jellemzően nemcsak a hibajavításokat, hanem az új verziókat is fedi a support-díj.
- A támogatás nemcsak a téves kódra, hanem a téves használatra is kiterjed-e vajon?

# 28. Feltörési lehetőségek, adatlopások

- Minden adatlopás egy tragédia. Sok esetben a következmény a cég megszűnése.
- Nem létezik 100%-os biztonság.
- Mégis érdemes felmérni a választás előtt, hogy az adott szoftvert milyen gyakran törik fel? Hány incidens került nyilvánosságra az elmúlt években?
- Hogyan reagál a szoftvergyártó, ha kiderül 1-1 sebezhetőség?
- Megtesz-e a gyártó minden tőle telhetőt ahhoz, hogy megelőzze az újabb sebezhetőségek kialakulását.
- Csupán „Denial-Of-Service” típusú sérülékenységekről beszélünk, vagy komoly adatlopások, adatmódosítások is történhetnek (pl. „SQL Injection”)?

# 29. Mennyire védettek az adatok a DBA-tól

- Egy komoly és eldöntendő kérdés, hogy veszélyforrásként tekint-e a cég a rendszergazdákra, vagy nem.
- Ha félünk a DBA-k és az operációs rendszerek rendszergazdáinak a jogosultságaitól, akkor vajon létezik-e olyan szoftververzió, ahol a DBA és a rendszergazda sem férhetnek hozzá az adatokhoz?
- Pl. az Oracle adatbázis esetén ezt a megoldást Oracle Database Vaultnak nevezzük.

# 30. És még sok egyéb szempont is felmerülhet

- Az eddigiekben felsorolt kritériumok nem egy végleges listát alkotnak.
- Az évek során és a technológia fejlődésével állandóan új szempontok merülnek fel.
- Az látszik, hogy a „melyik a jobb adatbátiskezelő rendszer” kérdés nagyon sokrétű, komplex.

PEOPLE COME FIRST
INFORMATIKAI SZAKÉRTŐK EGYESÜLETE

# Nagyvállalati rendszerek

## Vállalati informatikai rendszerek architektúrájának áttekintése

# Informatika szerepe egy vállalat életében

- Értékteremtés
  - Saját fejlesztések
  - Gyártás vezérlés
- Támogatás
  - Levelező rendszer
  - Címtárak
  - Automatikus riportok

# Kinek vannak igényei?

- Felhasználók
- Üzlet
- Biztonsági szakértők
- Jogi osztály
- Üzemeltetők

# Teljesítmény

- “Legyen gyors”
  - Mikortól számít jónak/rossznak?
- Service Level Agreement (SLA)
  - 99.999% (“öt kilences”)
- Mérés és riasztás
  - Automatikus beavatkozás, ha lehetséges

# Robosztusság/hibatűrés

- Single point of Failure (SPF)
- Redundancia
  - Áramellátás (UPS)
  - Hálózati eszközök (bonding)
  - Komponensek (cluster)
- Hibakezelés
- Mentés és visszaállítás

# Magas rendelkezésre állás

- Nem csak hiba miatt lehet leállás
  - Karbantartás (külső, szoftveres, hardveres)
  - Verzió frissítés
- Megoldás: redundancia
- SLA itt is használandó
  - 99.999% rendelkezésre állás→ 5 perc/év kiesés
- Példa: RAC, WebLogic Cluster

# Változásra képes

- Minden projektvezető rémálma: scope bővülés
- Megoldás: absztrakció

  “X szolgátatás-t kell meghívni”

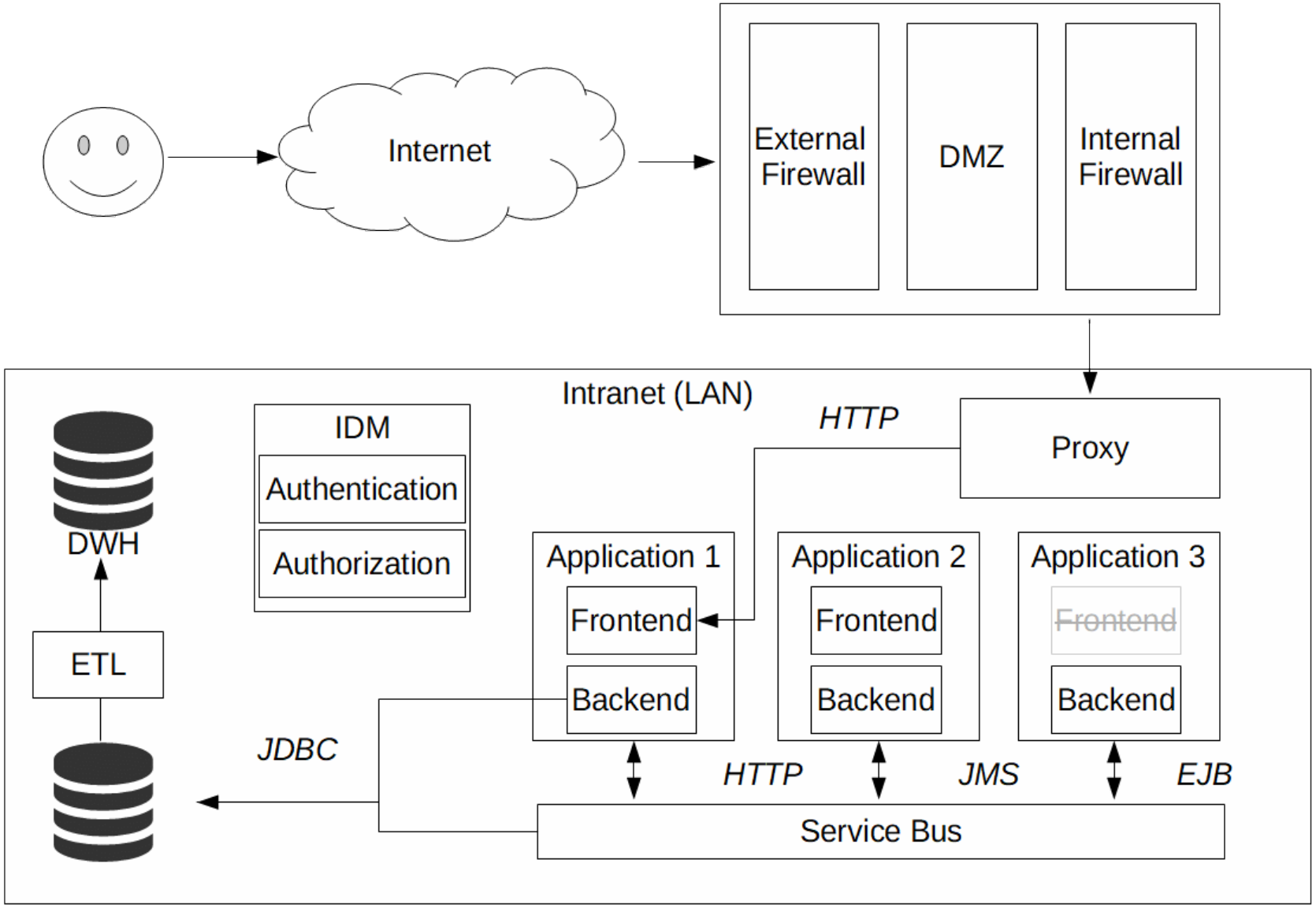
  “A https://192.168.60.13:8443/myService címen található szolgáltatást kell meghívni”
- Érdemes haladni a korral
  - Technical debt

# Biztonság

- Senki sem férhet hozzá mindenhez
- A hozzáférések központi rendszer(ek)en keresztül kaphatók
  - Szerepkörök
- “Közös”, illetve “service account”-ok kerülése
- Események auditálása
- Gyakori, ütemezett biztonsági frissítések

# Megismételhetőség

- Biztos működni fog? Biztos “úgy” fog működni?
- Változtatások, fejlesztések előzetes tesztelése
- Több célhoz több környezet szükséges
  - Funkcionális tesztelés
  - Teljesítmény tesztelés
  - Fejlesztés támogatás
- Automatizálással érdemes támogatni


Internet
External Firewall
DMZ
Internal Firewall
Intranet (LAN)
HTTP
Proxy
IDM
Authentication
Authorization
DWH
ETL
Application 1
Frontend
Backend
Application 2
Frontend
Backend
Application 3
Frontend
Backend
JDBC
HTTP
JMS
EJB
Service Bus

# Szerverek, virtuális gépek

- Erőforrások ezeken keresztül biztosíthatók
  - CPU, RAM stb.
- Jellemzően ez az üzemeltetési egység
- Főleg virtuális gépek (VM)

# Storage

- Jellemzően Network Attached Storage (NAS)
- Redundancia támogatás (RAID)
- Adatbázisok esetén komoly teljesítmény tényező

# Tűzfal

- Hálózati komponens
- ISO/OSI Layer 3/Layer 4 (Network + Transport)
- Célok:
  - Forgalom szűrés (filtering)
  - Network Address Translation (NAT)
- Hardveres és szoftveres (iptables) is

# (Reverse) Proxy

- Hálózati vagy Middleware komponens
- Jellemzően HTTP szolgáltatásokhoz
- Apache, nginx, TinyProxy
- Gyarkan SSL végződési pont
- Terheléselosztóként (load balancer) is használatos
  - WebLogic Cluster-hez szükséges
- VirtualHost
  - app1.example.com vs. app2.example.com

# Alkalmazás szerver

- Jellemzően JavaEE (de lehet .NET stb.)
- Absztarkciós réteg az alkalmazás számára
  - pl. adatbázis kapcsolatok
- Clusterezési képesség
- Több komponens futtatását is lehetővé teszi
  - Jobban menedzselhető
  - Támogatja a modularitást

# Frontend vs Backend

- Érdemes az alkalmazásokat szétbontani
  - Komponens szinten (külön EAR/WAR/stb.)
  - Infrastruktúra szinten (külön JVM/szerver/stb.)
- Szabványos protokolok a komponensek között
  - HTTP, SOAP, JMS, EJB ...

# Integrációs réteg (Service Bus)

- Szolgáltatások hívása egy közös rétegen keresztül
- Nem csupán proxy
  - Kompozit szolgáltatások
  - Protokol váltás
  - Throttling (ráta korlátozás)
- Logikai komponens
  - Alkalmazás szervern fut ez is

# Perzisztencia

- Nem minden szolgáltatás igényli
- Jellemzően adatbázis alapú (de: PDF generálás)
- Alkalmazás szerver oldali támogatás
  - Adatbázis kapcsolatok
  - Persistent store-ok

# Identity Management

- Címtár szolgáltatások
  - Felhasználók (authentication)
  - Csoportok (authorization)
- Jellemzően LDAP, SAML technológiákkal
- Jogosultságok automatikus kiosztása
  - Folyamatok alapján (igény, elfogadás, kiosztás)

# Monitoring

- Két megközelítés
  - Push/ágens alapú architektúra (pl. OEM)
  - Pull architektúra (Nagios)
- Előre meghatározott megtrikák gyűjtése
- Metrika határok alapján történő riasztás
  - Akár trendek alapján (historikus adatokkal)
- Nehézség: alkalmazás specifikus metrikák

# Logolás és Audit

- Érdemes központi log gyűjtést alkalmazni
- Nem jó ha:
  - Nagyon kevés bejegyzés van
  - Túl sok bejegyzés van
- Auditálás: komponens függő
  - Oracle adatbázisban beépített
- Monitoring eszközzel jól integrálható

PEOPLE COME FIRST
INFORMATIKAI SZAKÉRTŐK EGYESÜLETE

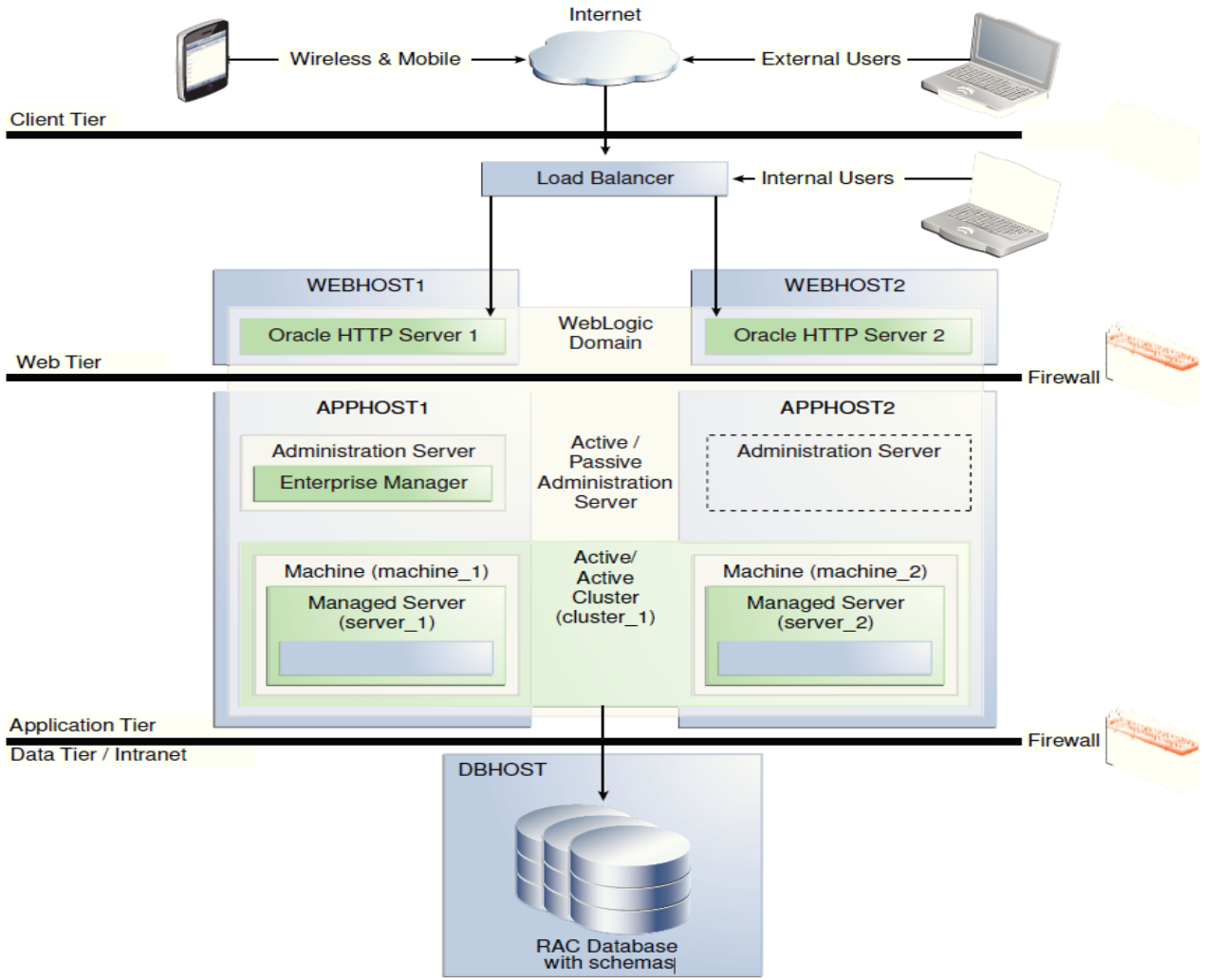
# Oracle Fusion Middleware

# Agenda

- Overview of Oracle Fusion Middleware
- About Key Oracle Fusion Middleware Concepts
- Overview of Oracle Fusion Middleware Components
- Using Oracle Fusion Middleware Tools

# Overview of Oracle Fusion Middleware

# Overview of Oracle Fusion Middleware



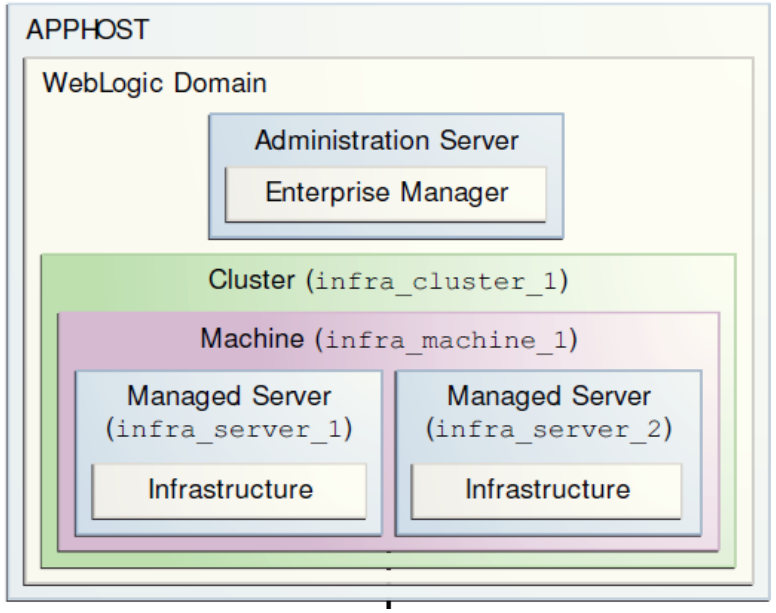
# About Key Oracle Fusion Middleware Concepts

# About Key Oracle Fusion Middleware Concepts

- What Is an Oracle WebLogic Server Domain?
- What Is the Administration Server?
- Overview of Managed Servers and Managed Server Clusters
- What Is a Java Component?
- What Is a System Component?
- What Is Node Manager?
- What Are the Key Oracle Fusion Middleware Directories?
- What Is the WebLogic Management Framework?
- What Is the Metadata Repository?

# What Is an Oracle WebLogic Server Domain?

- An Oracle WebLogic Server **domain** is a logically related group of Java components.
- A domain includes a special WebLogic Server instance called the **Administration Server**, which is the central point from which you configure and manage all resources in the domain.



# What Is the Administration Server?

- The **Administration Server** operates as the central control entity for the configuration of the entire domain.
- It maintains the domain's configuration documents and distributes changes in the configuration documents to Managed Servers.
- The Administration Server serves as a central location from which to manage and monitor all resources in a domain.
- Each domain must have one server instance that acts as the Administration Server.

# Overview of Managed Servers and Managed Server Clusters

- **Managed Servers** host business applications, application components, Web services, and their associated resources.
- To optimize performance, Managed Servers maintain a read-only copy of the domain's configuration document.
- When a Managed Server starts, it connects to the domain's Administration Server to synchronize its configuration document with the document that the Administration Server maintains.
- For production environments that require increased application performance, throughput, or high availability, you can configure two or more Managed Servers to operate as a cluster.
- A **cluster** is a collection of multiple WebLogic Server instances running simultaneously and working together to provide increased scalability and reliability.

# What Is a Java Component?

- A **Java component** is an Oracle Fusion Middleware component that is deployed as one or more Java EE applications and a set of resources.
- Java components are deployed to an Oracle WebLogic Server domain as part of a domain template.
- Oracle WebLogic Server and Oracle Coherence are examples of Java components.

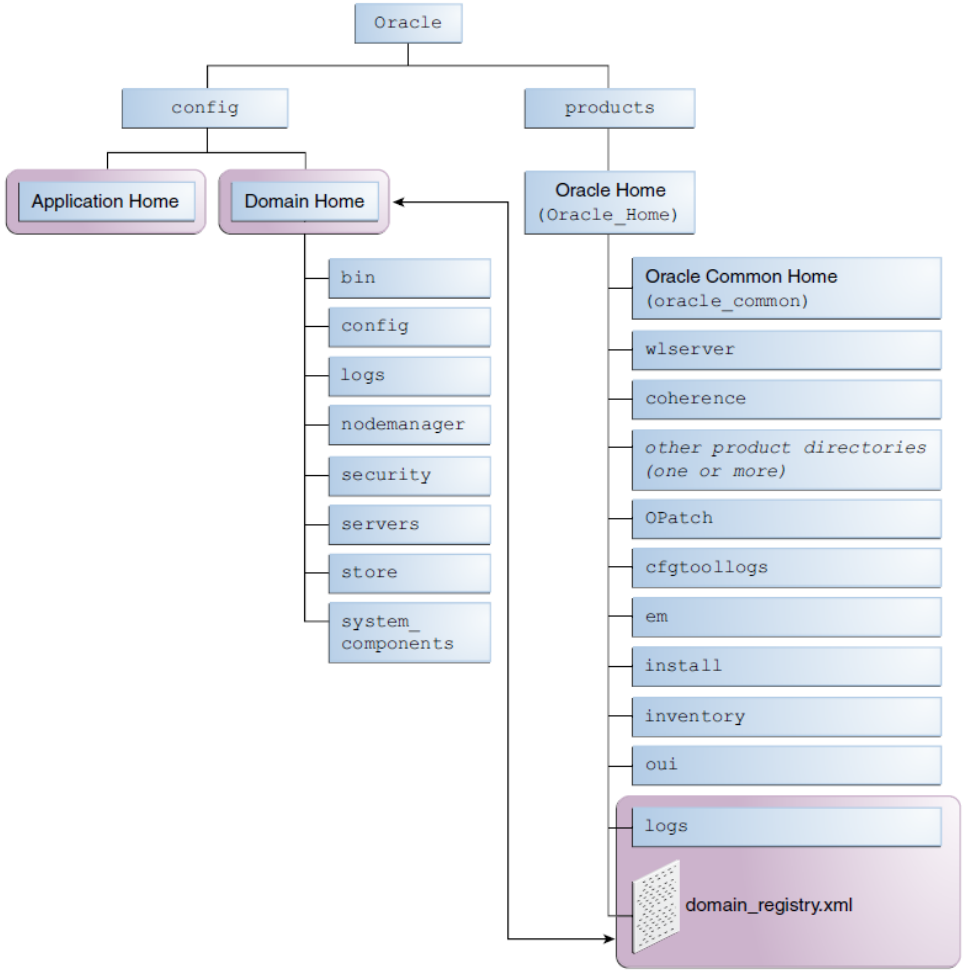
# What Is a System Component?

- A **system component** is a manageable process that is not deployed in a Java application container. Oracle HTTP Server is an example of a system component.
- System components can be deployed in a WebLogic Server domain or in a standalone domain and managed by the Weblogic Management Framework.

# What Is Node Manager?

- **Node Manager** is a Java utility that runs as a separate process from Oracle WebLogic Server and allows you to perform common operations for a Managed Server, regardless of its location with respect to its Administration Server, and for system components.
- If you run Node Manager on a computer that hosts Managed Servers, you can start and stop the Managed Servers remotely using the Administration Console, Fusion Middleware Control, or the command line.
- Node Manager can also automatically restart a Managed Server after an unexpected failure.

# What Are the Key Oracle Fusion Middleware Directories?



# What Is the WebLogic Management Framework?

- Oracle Fusion Middleware provides the **WebLogic Management Framework**, which provides heterogeneous management capabilities for Oracle Fusion Middleware products that require basic administrative capabilities.
- Its capabilities include start, stop, configuration settings, and other such basic product lifecycle operations through a common command line, API and user interface.

# What Is the Metadata Repository?

- The metadata repository contains metadata for Oracle Fusion Middleware components, such as Oracle Application Development Framework and Oracle SOA Suite.
- It can also contain metadata about the configuration of Oracle Fusion Middleware and metadata for enterprise applications.
- A metadata repository can be database-based or file-based.
- If it is database-based, the repository can be installed into an existing database using the Repository Creation Utility (RCU).
- With RCU, you can create schemas for Oracle Fusion Middleware components.

# Overview of Oracle Fusion Middleware Components

# Overview of Oracle Fusion Middleware Components

- Oracle Fusion Middleware Infrastructure
- Oracle WebLogic Server
- Oracle Coherence
- Oracle HTTP Server
- Oracle Identity Management
- Oracle Data Integrator
- Oracle SOA Suite
- Oracle Business Process Management Suite
- Oracle Traffic Director
- Oracle WebCenter Content
- Oracle WebCenter Portal
- Oracle BI Enterprise Edition
- Oracle Forms Services

# Oracle Fusion Middleware Infrastructure

- Oracle Fusion Middleware Infrastructure is an Oracle Fusion Middleware distribution that provides Oracle WebLogic Server, Oracle Coherence, and the Oracle JRF infrastructure services.
- The Oracle JRF infrastructure services include:
  - • Oracle Application Development Framework• Oracle Metadata Services
  - • Oracle Platform Security Services
  - • Oracle Web Services Manager
  - • Oracle Enterprise Manager Fusion Middleware Control
- These services are installed into the Oracle common directory inside the Oracle Fusion Middleware 12c Oracle home by the Infrastructure installer.

# Oracle WebLogic Server

- The Oracle WebLogic Server is a scalable, enterprise-ready Java Platform, Enterprise Edition (Java EE) application server.
- The Oracle WebLogic Server infrastructure supports the deployment of many types of distributed applications and is an ideal foundation for building applications based on SOA.
- Java EE is a widely used platform for server programming in the Java programming language.
- The Java EE Platform differs from the standard edition of Java in that it adds libraries which provide functionality to deploy fault-tolerant, distributed, multi-tier Java software, based largely on modular components running on an application server.

# Oracle Coherence

- Oracle Coherence provides clustered data management with a fully coherent, single system image (SSI), scalability for both read and write access, fast, transparent failover and failback, linear scalability for storage and processing, no Single-Points-of-Failure (SPOFs) and cluster-wide locking and transactions.
- Built on top of this foundation are the various services that Coherence provides, including database caching, HTTP session management, grid agent invocation and distributed queries.

# Oracle HTTP Server

- Oracle HTTP Server is based on Apache HTTP Server infrastructure, and includes modules developed specifically by Oracle.
- The features of single sign-on, clustered deployment, and high availability enhance the operation of the Oracle HTTP Server.
- Oracle HTTP Server has the following components to handle client requests:
  - HTTP listener, to handle incoming requests and route them to the appropriate processing utility.
  - Modules (mods), to implement and extend the basic functionality of Oracle HTTP Server.

# Oracle Identity Management

- Oracle Identity Management enables organizations to effectively manage the end-to-end lifecycle of user identities across all enterprise resources, both within and beyond the firewall and into the cloud.
- The Oracle Identity Management platform delivers scalable solutions for identity governance, access management and directory services.
- This modern platform helps organizations strengthen security, simplify compliance and capture business opportunities around mobile and social access.

# Oracle Data Integrator

- Oracle Data Integrator provides a fully unified solution for building, deploying, and managing complex data warehouses or as part of data-centric architectures in a SOA or business intelligence environment.
- In addition, it combines all the elements of data integration — data movement, data synchronization, data quality, data management, and data services—to ensure that information is timely, accurate, and consistent across complex systems.
- Oracle Data Integrator features an active integration platform that includes all styles of data integration: data-based, event-based and service-based.
- ODI unifies silos of integration by transforming large volumes of data efficiently, processing events in real time through its advanced Changed Data Capture (CDC) capability, and providing data services to the Oracle SOA Suite.
- It also provides robust data integrity control features, assuring the consistency and correctness of data.
- With powerful core differentiators — heterogeneous E-LT, Declarative Design and Knowledge Modules — Oracle Data Integrator meets the performance, flexibility, productivity, modularity and hot-pluggability requirements of an integration platform.

# Oracle SOA Suite

- Service Oriented Architecture (SOA) provides an enterprise architecture that supports building connected enterprise applications to provide solutions to business problems.
- SOA facilitates the development of enterprise applications as modular business web services that can be easily integrated and reused, creating a truly flexible, adaptable IT infrastructure.
- Oracle SOA Suite provides a complete set of service infrastructure components for designing, deploying, and managing composite applications.
- Oracle SOA Suite enables services to be created, managed, and orchestrated into composite applications and business processes.
- Composites enable you to easily assemble multiple technology components into one SOA composite application.
- Oracle SOA Suite plugs into heterogeneous IT infrastructures and enables enterprises to incrementally adopt SOA.

# Oracle Business Process Management Suite

- Oracle BPM Suite provides an integrated environment for developing, administering, and using business applications centered around business processes.
- Oracle BPM Suite provides a seamless integration of all stages of the application development lifecycle from design-time and implementation to runtime and application management.
- Oracle BPM Suite is layered on Oracle SOA Suite and shares many of the same product components.

# Oracle Traffic Director

- Oracle Traffic Director is a fast, reliable, and scalable layer-7 software load balancer.
- You can set up Oracle Traffic Director to serve as the reliable entry point for all HTTP, HTTPS and TCP traffic to application servers and web servers in the back end.
- Oracle Traffic Director distributes the requests that it receives from clients to servers in the back end based on the specified load-balancing method, routes the requests based on specified rules, caches frequently accessed data, prioritizes traffic, and controls the quality of service.
- The architecture of Oracle Traffic Director enables it to handle large volumes of application traffic with low latency.
- The product is optimized for use in Oracle Exalogic Elastic Cloud and Oracle SuperCluster.

# Oracle WebCenter Content

- Oracle WebCenter Content can help a corporation unify, manage, and leverage all types of content across the entire enterprise.
- All corporate lines of business such as marketing, brand management, web presence, accounting, sales, human resources, and engineering can share the same tools and easily access the same information in a common repository.
- By centralizing information and allowing access to information by qualified users, Oracle WebCenter Content provides cost savings, eliminates redundancy and increases efficiency.
- At the same time, information is properly managed, tracked, and disposed of as needed.

# Oracle WebCenter Portal

- Oracle WebCenter Portal is an integrated set of components with which you can create social applications, enterprise portals, collaborative communities, and composite applications, built on a standards-based, service-oriented architecture.
- Oracle WebCenter Portal combines dynamic user interface technologies with which to develop rich internet applications, the flexibility and power of an integrated, multichannel portal framework, and a set of tools and services that provide content, collaboration, presence and social networking capabilities.
- Based on these components, Oracle WebCenter Portal also provides an out-of-the-box enterprise-ready customizable application called WebCenter Portal, with a configurable work environment that enables individuals and groups to work and collaborate more effectively.

# Oracle BI Enterprise Edition

- Oracle BI Enterprise Edition (sometimes simply referred to as Oracle Business Intelligence) provides a full range of business intelligence capabilities.
- These capabilities enable you to:
  - • Collect up-to-date data from your organization.
  - • Present the data in easy-to-understand formats (such as tables and graphs).
  - • Deliver data in a timely fashion to the employees in your organization.
- These capabilities enable your organization to make better decisions, take informed actions, and implement more-efficient business processes.

# Oracle Forms Services

- Oracle Forms is used to develop and deploy Forms applications.
- The Forms applications provide a user interface to access Oracle Database in an efficient and tightly-coupled way.
- The applications can be integrated with Java and web services to take advantage of service-oriented architecture (SOA).

# Using Oracle Fusion Middleware Tools

# Using Oracle Fusion Middleware Tools

- About the Installation and Configuration Tools
  - Oracle Universal Installer
  - Repository Configuration Utility
  - OPatch
  - Configuration Wizard
  - Reconfiguration Wizard
  - Upgrade Assistant
- About the Administration Tools
  - Oracle Enterprise Manager Fusion Middleware Control
  - Oracle WebLogic Server Administration Console
  - Oracle WebLogic Scripting Tool (WLST)
  - Oracle Enterprise Manager Cloud Control
- About the Development Tools
  - Oracle JDeveloper
  - Oracle Application Development Framework
  - Oracle TopLink

# Oracle Universal Installer

- Oracle Universal Installer (OUI) is a Java-based installer that enables you to install and deinstall Oracle components.
- It performs component-based installations as well as complex installations, such as integrated bundle and suite installations, and installations over the Web.
- One of the key features of OUI is that it provides a prerequisite checking tool to diagnose the readiness of an environment for installation.
- The prerequisite checks are run as part of the installation process, but can also be run as a separate application.

# Repository Configuration Utility

- Many of the Oracle Fusion Middleware components require the existence of schemas in a database prior to installation.
- These schemas are created and loaded in your database using the Repository Creation Utility (RCU).

# OPatch

- OPatch is a utility that assists you with the process of applying interim patches to Oracle's software.
- OPatch is a Java-based utility that can run on either OUI-based Oracle homes or standalone homes.
- It works on all operating systems for which Oracle releases software.

# Configuration Wizard

- The Configuration Wizard creates the appropriate directory structure for a WebLogic Server domain or a standalone domain, a domain configuration file, and scripts you can use to start the servers in the domain.
- The Configuration Wizard guides you through the process of creating or updating a domain for your target environment by selecting the product components to include in your domain, or by selecting template JAR files.
- If necessary, you can also customize the domain to suit your environment by adding and configuring Managed Servers, clusters, and machine definitions, or customizing predefined JDBC data sources and JMS file store directories.

# Reconfiguration Wizard

- The Reconfiguration Wizard reconfigures an existing WebLogic Server domain, which was created using a prior release.
- The following items are automatically updated, depending on the applications in the domain:
  - • WLS core infrastructure
  - • Domain version
- If your domain also includes Oracle Fusion Middleware products, other items may also be updated, depending on the particular product.
- This lets you take advantage of new features that are included with the newest version of Oracle Fusion Middleware.

# Upgrade Assistant

- The Upgrade Assistant automates the upgrade of many aspects of the Oracle Fusion Middleware environment.
- The Oracle Fusion Middleware upgrade Assistant guides you through the process of upgrading from previous versions.

# Oracle Enterprise Manager Fusion Middleware Control

- Oracle Enterprise Manager Fusion Middleware Control (Fusion Middleware Control) is a Web browser-based, graphical user interface that you can use to monitor and administer a domain.
- It can manage the Administration Server, Managed Servers, clusters, and the Oracle Fusion Middleware

# Oracle WebLogic Server Administration Console

- Oracle WebLogic Server Administration Console is a Web browser-based, graphical user interface that you use to manage an Oracle WebLogic Server domain.
- It is accessible from any supported Web browser with network access to the Administration Server.
- Use the Administration Console to:
  - Configure, start, and stop Oracle WebLogic Server instances
  - Configure Oracle WebLogic Server clusters
  - Configure Oracle WebLogic Server services, such as database connectivity (JDBC) and JMS messaging
  - Configure security parameters, including creating and managing users, groups, and roles
  - Configure and deploy Java EE applications

# Oracle WebLogic Scripting Tool (WLST)

- The Oracle WebLogic Scripting Tool (WLST) is a command-line scripting environment that you can use to create, manage, and monitor Oracle WebLogic Server domains and standalone domains.
- It is based on the Java scripting interpreter, Jython.
- In addition to supporting standard Jython features such as local variables, conditional variables, and flow control statements, WLST provides a set of scripting functions (commands) that are specific to Oracle WebLogic Server and Oracle Fusion Middleware components.
- You can extend the WebLogic scripting language to suit your needs by following the Jython language syntax.
- You can use any of the following techniques to invoke WLST commands:
  - • Interactively, on the command line
  - • In script mode, supplied in a file
  - • Embedded in Java code

# Oracle Enterprise Manager Cloud Control

- Oracle Enterprise Manager Cloud Control provides a comprehensive management solution for Oracle WebLogic Server, Oracle Fusion Middleware, non-Oracle middleware technology, and lifecycle management including configuration management, compliance management, provisioning, and patching.
- Cloud Control encompasses out-of-the-box availability and performance monitoring, robust diagnostics, configuration management, and lifecycle management across middleware software such as Oracle WebLogic Server, Oracle Coherence, and Oracle HTTP Server.

# Oracle JDeveloper

- Oracle JDeveloper is an integrated development environment (IDE) for building applications, such as SOA applications, using the latest standards for Java, XML, Web services, and SQL.
- It supports the complete development lifecycle with integrated features for modeling, coding, debugging, testing, profiling, tuning, and deploying applications.
- Oracle JDeveloper is the main development platform for the Oracle Fusion Middleware suite of products.
- It is a cross-platform IDE that runs on Windows, Linux, Mac OS X, and other UNIX-based systems.
- Oracle JDeveloper provides a visual and declarative development approach and works together with the Oracle ADF to simplify development.

# Oracle Application Development Framework

- Oracle ADF is an end-to-end application framework that builds on Java EE standards and open-source technologies to simplify and accelerate implementing service-oriented-applications.
- Oracle ADF is suitable for enterprise developers who want to create applications that search, display, create, modify, and validate data using Web, wireless, desktop, or Web services interfaces.
- Used in tandem, Oracle JDeveloper and Oracle ADF provide an environment that covers the full development lifecycle from design to deployment, with drag-and-drop data binding, visual UI design, and team development features built in.

# Oracle TopLink

- Oracle TopLink is an advanced, object-persistence and object-transformation framework that provides development tools and run time capabilities that reduce development and maintenance efforts, and increase enterprise application functionality.
- Oracle TopLink builds high-performance applications that store persistent object-oriented data in a relational database.
- It successfully transforms object-oriented data into either relational data, or Extensible Markup Language (XML) elements.

# Q/A?

- https://docs.oracle.com/en/middleware/lifecycle/12.2.1.3/ascon/toc.htm
- https://docs.oracle.com/en/middleware/
- https://tinyurl.com/webvalto

PEOPLE COME FIRST
INFORMATIKAI SZAKÉRTŐK EGYESÜLETE

# Application Infrastructure

# Objectives

- Standards, containers, APIs, and services
- Application component functionalities mapped to tiers and containers
  - Web container technologies
  - Business logic implementation technologies
  - Web service technologies
- Packaging and deployment
- Enterprise JavaBeans, managed beans, and CDI beans
  - Understanding lifecycle and memory scopes
- Linking components together with annotations, injections, and JNDI

# Requirements of Enterprise Applications

- The Java EE platform:
  - Is an architecture for implementing enterprise-class applications
  - Uses Java and Internet technology
  - Has a primary goal of simplifying development of enterprise-class applications through an application model that is:
    - Vendor-neutral
    - Component-based

Java™ COMPATIBLE ENTERPRISE EDITION

# Separation of Business Logic from Platform Services

Build from the ground up.

Use Application Component Server.

## Developer's Checklist

- [ ] Business services
- [ ] Persistence and Transaction management
- [ ] Multithreading
- [ ] Security management
- [ ] Networking

## Developer's Checklist

- [ ] Business services

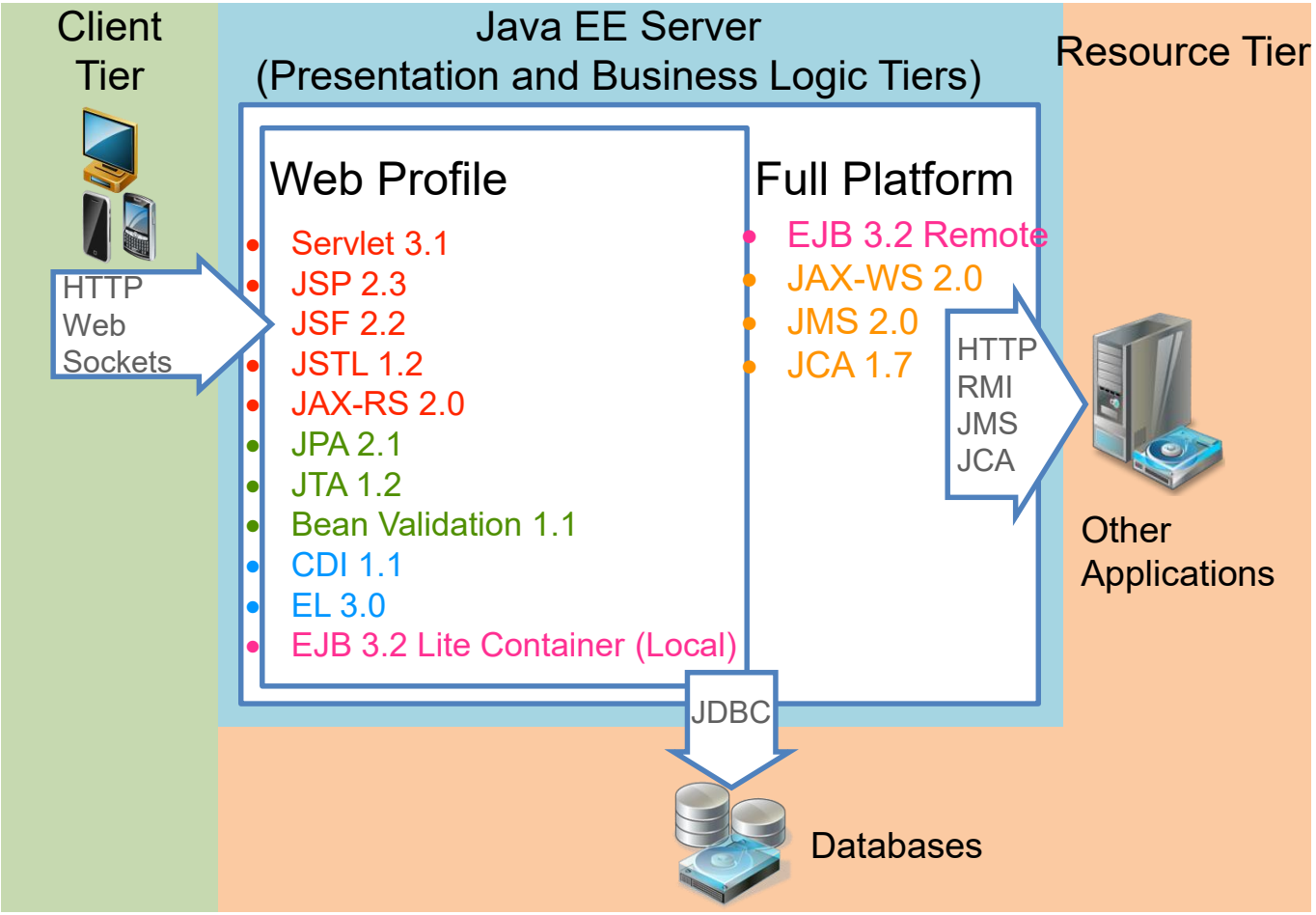
## Services Provided by Server

- [x] Persistence and Transaction management
- [x] Multithreading
- [x] Security management
- [x] Networking

# Structure and Purpose of Java EE 7 Server, Containers, and APIs

The Java EE platform describes Web and EJB containers and various APIs:

- → Web Container Technologies
- → Java SE Technologies
- → Technologies in all containers
- → EJB Container Technologies
- → Technologies supported with Full Platform server implementation



# EJB Lite and EJB Full Containers

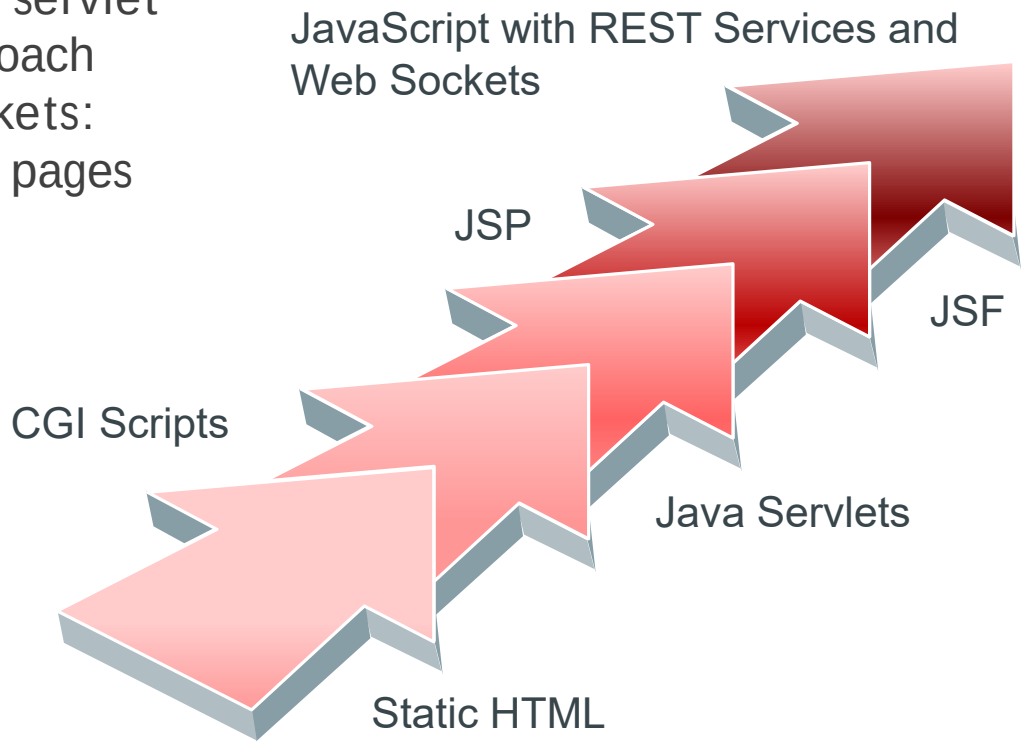
- EJB Lite features:
  - Required by the Web Profile
  - Session beans components:
    - Stateless
    - Stateful
    - Singleton
  - Support local clients
  - Method invocations:
    - Synchronous
    - Asynchronous
  - Transaction modes:
    - Container-managed
    - Bean-managed
  - Declarative and programmatic security
  - Automatically created EJB timers

EJB Full = EJB Lite + additional features:

- Required by Full Platform
- Message-driven beans
- Remote and local clients
- JAX-WS web service endpoints
- Persistent EJB timer service
- Support legacy services and EJB APIs

# Evolution of Web Design

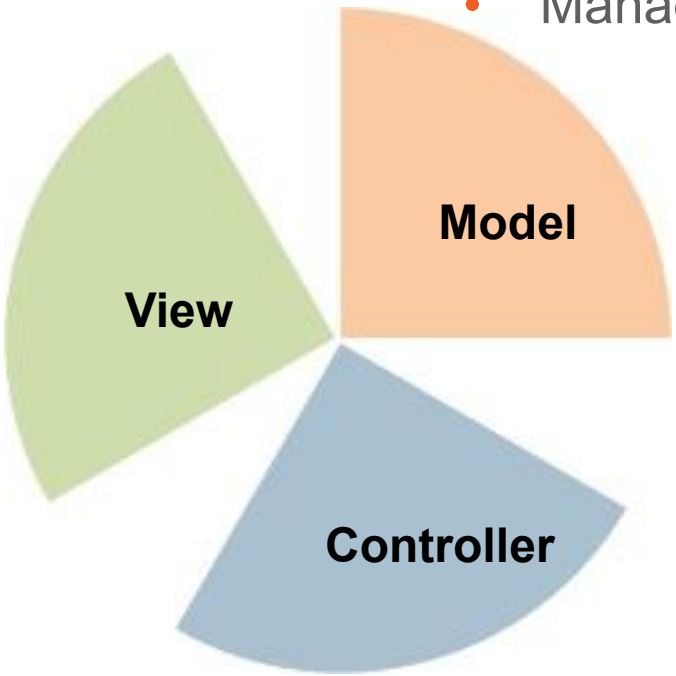
- Web: Started as static HTML documents
- CGI scripts: Introduced dynamically generated content
- Java servlets: Multithreaded and scalable solution
- Java Server Pages: Improved UI design of a servlet
- Java Server Faces: Implemented MVC approach
- JavaScript with REST services and web sockets: Added client-side UI and event handling to pages that are likely to be produced by using Servlet/JSP/JSF



# MVC (Model View Controller)

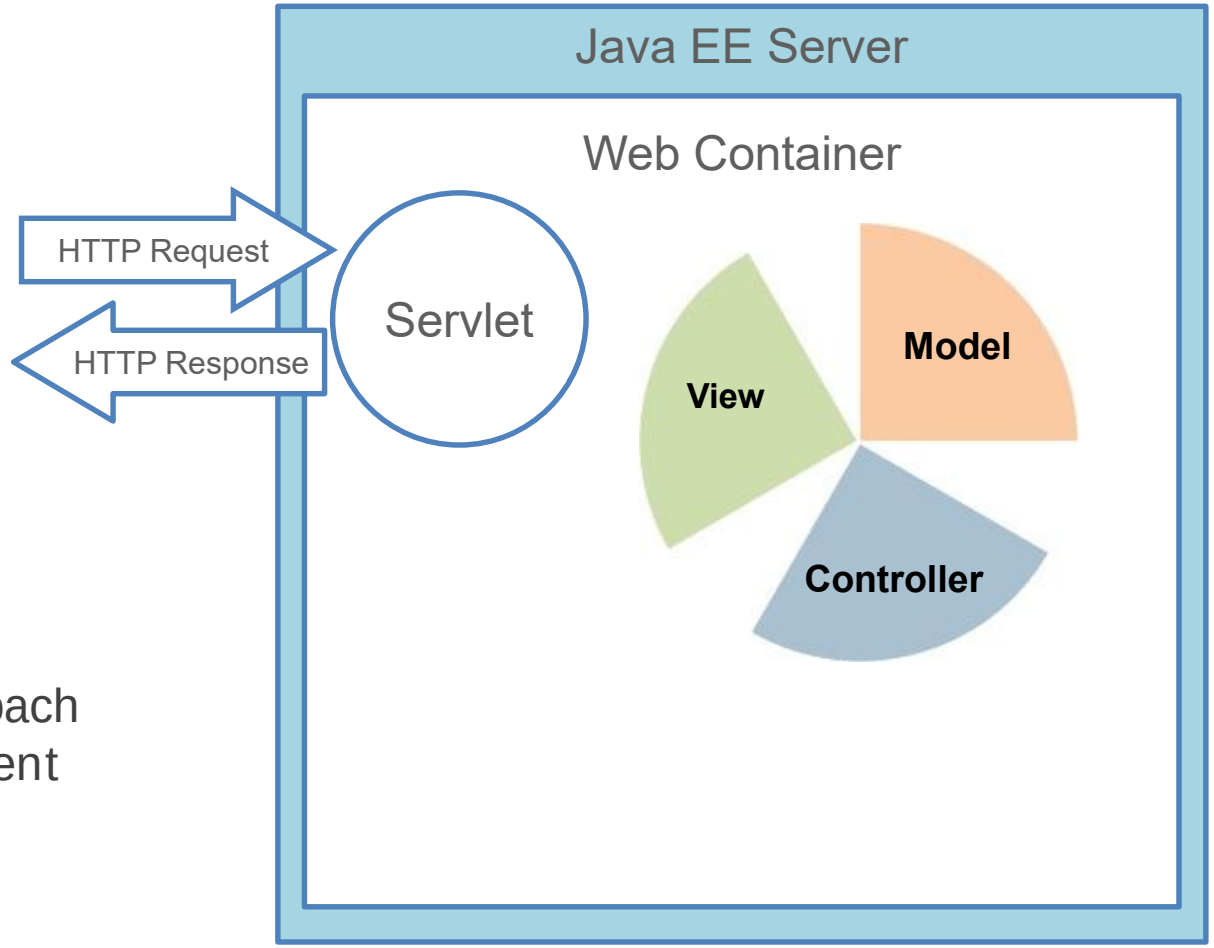
- Implement Business Logic
- Manage Data Persistency

- Produce User Interface
- Manage Presentation
- Generate Events



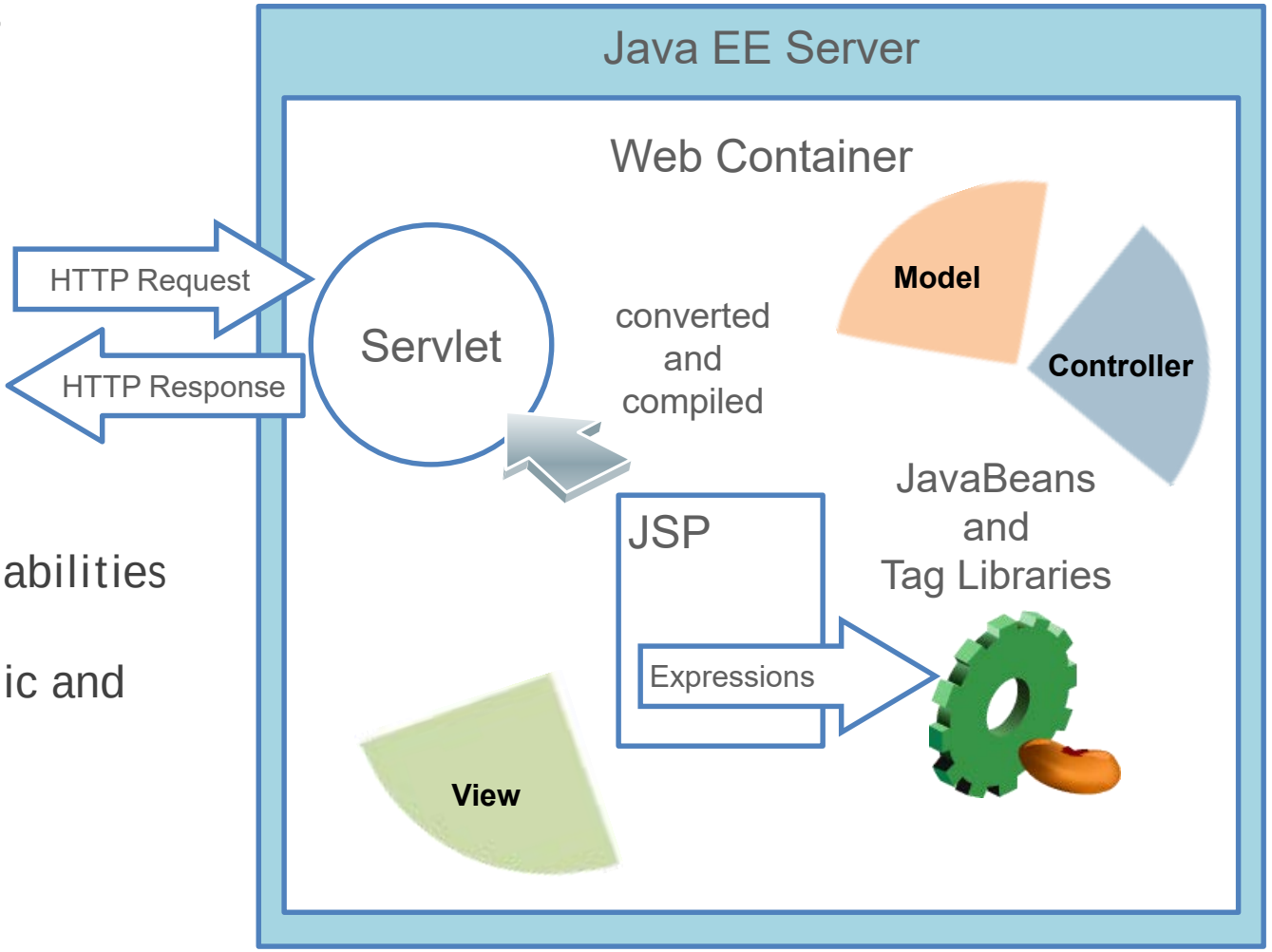

- Handle Events
- Send Commands to the Model
- Control View

# Java EE Web Container Components: Servlets



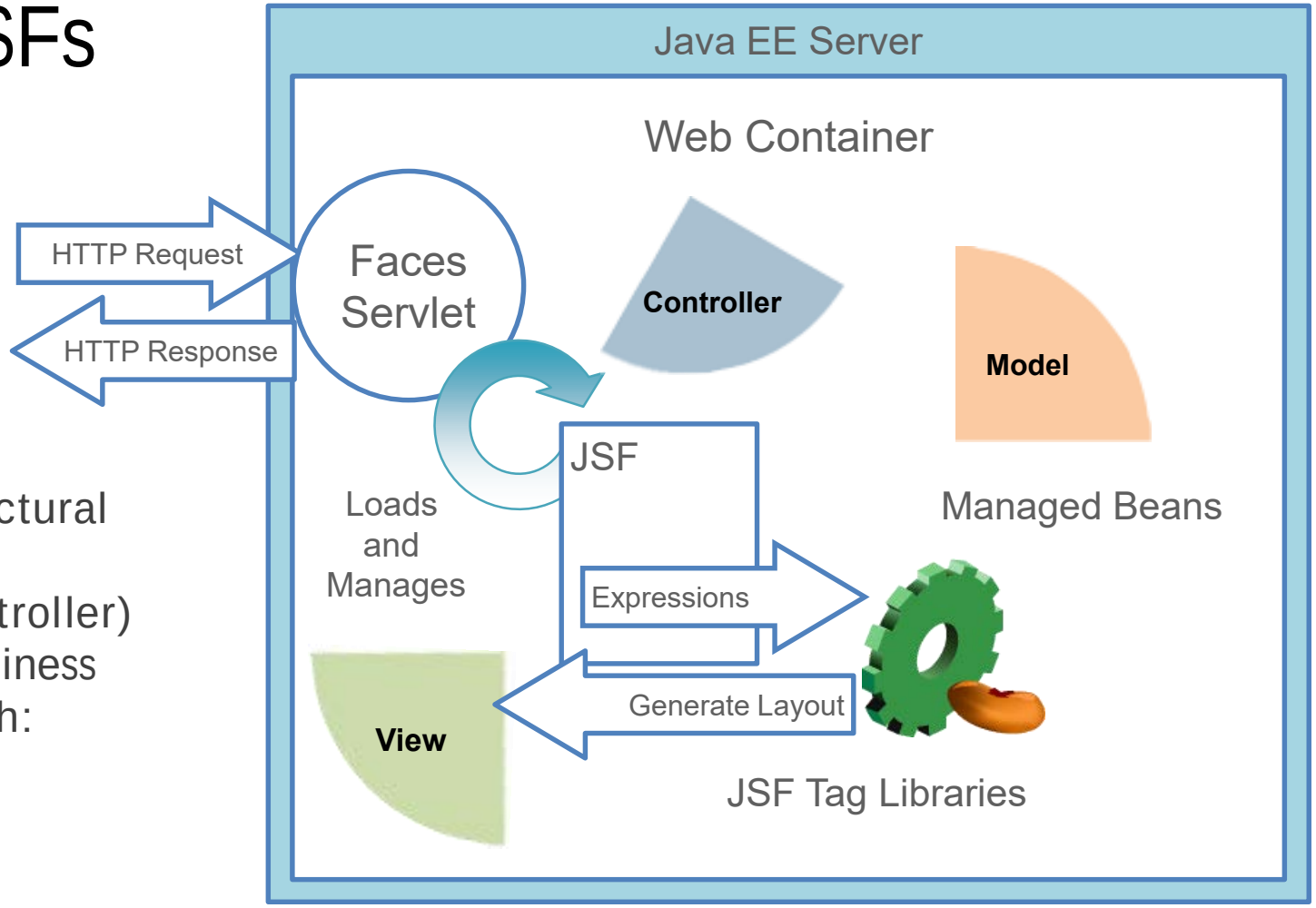

- Servlets:
  - Are Java classes mapped to URLs
  - Are typically invoked via HTTP
  - Utilize request-response architectural approach
  - Can mix business logic and layout management

# Java EE Web Container Components: JSPs



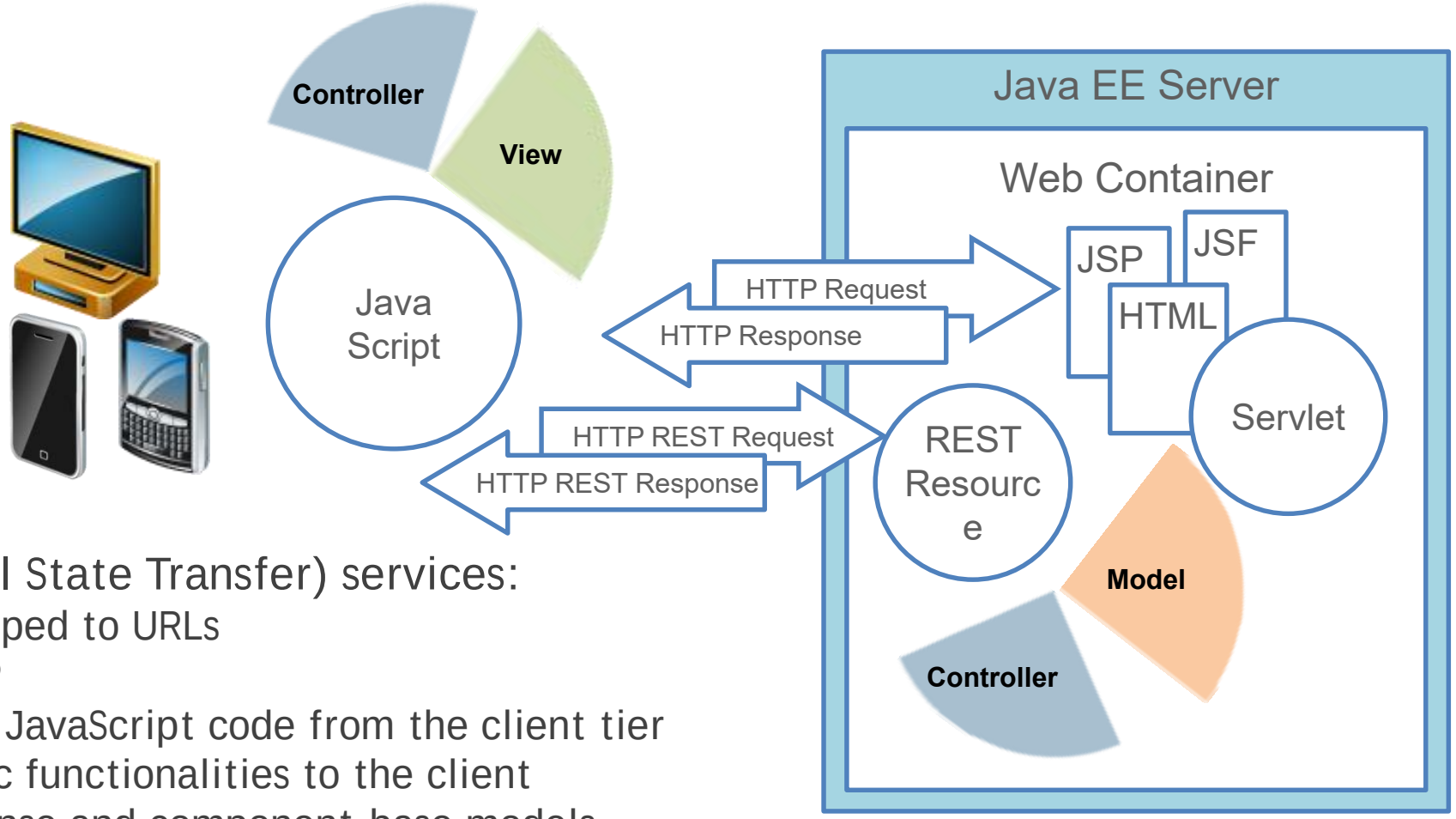

- JSPs (Java Server Pages):
  - Are turned into servlets (so all servlet abilities still apply)
  - Offer better separation of business logic and layout management with:
    - JavaBeans
    - Tag libraries
    - Expression Language

# Java EE Web Container Components: JSFs



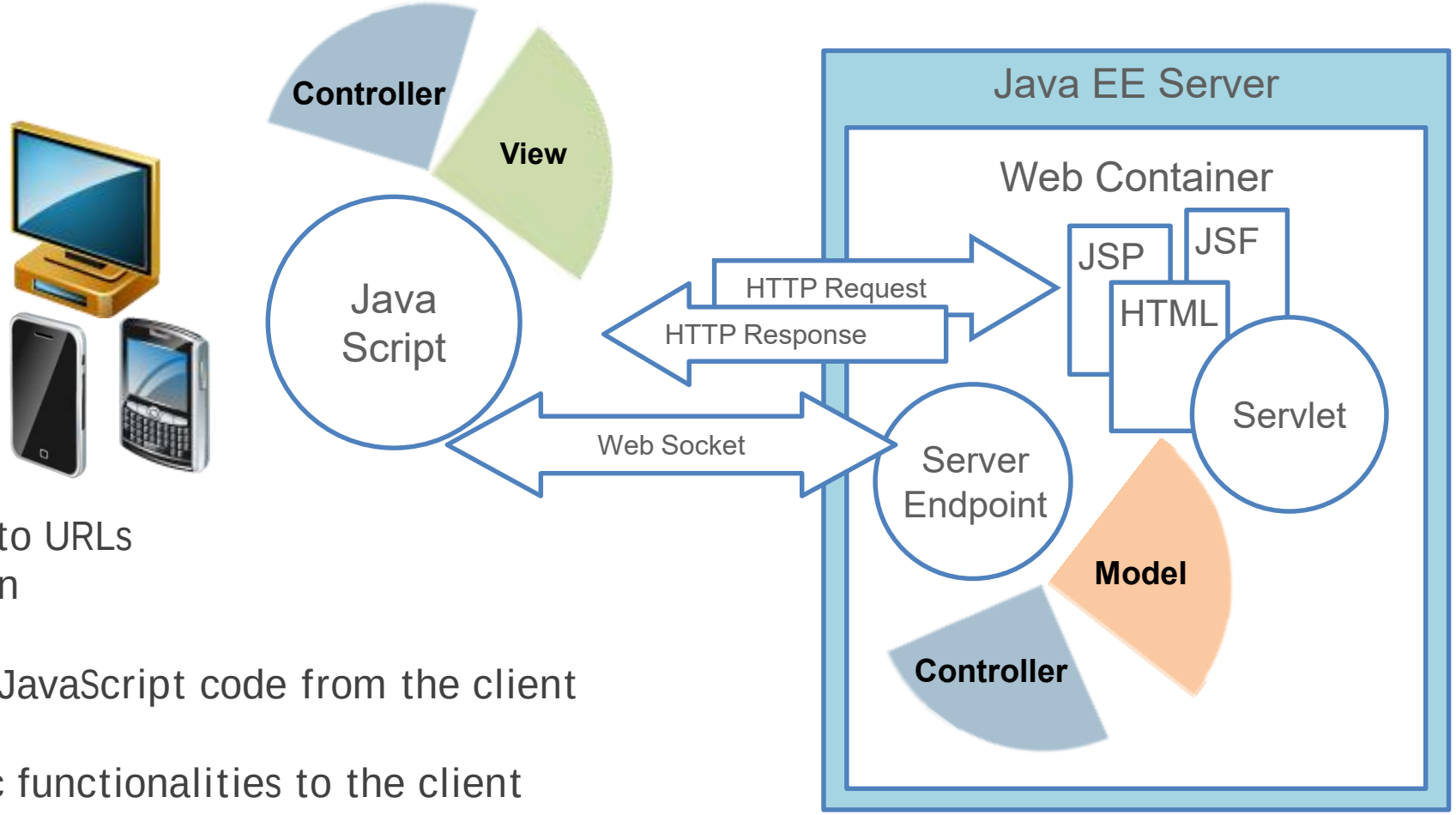

- JSFs (Java Server Faces):
  - Are interpreted by FacesServlet
  - Utilize a component-base architectural approach
  - Implement MVC (Model-View-Controller) architecture to fully separate business logic and layout management with:
    - JSF lifecycle management
    - Managed beans
    - Expression Language

# Java EE Web Container Components: REST Services




- REST (Representational State Transfer) services:
  - Are Java Classes mapped to URLs
  - Are invoked via HTTP
  - Are typically used by JavaScript code from the client tier
  - Present business-logic functionalities to the client
  - Utilize request-response and component-base models

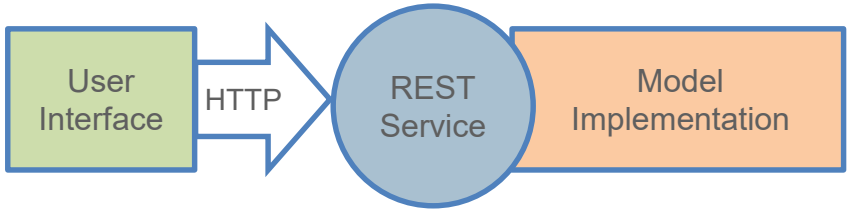
# Java EE Web Container Components: Web Sockets

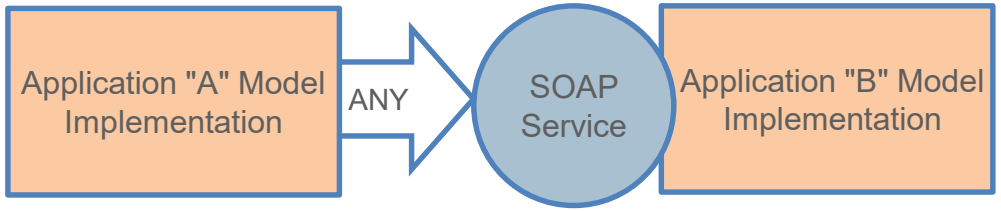



- Web sockets:
  - Java classes mapped to URLs
  - Full duplex connection
  - Can use server push
  - Are typically used by JavaScript code from the client tier
  - Present business-logic functionalities to the client
  - Utilize component-base models

# Java EE 7 Web Services

- All Web Services provide business functionalities in a way that disguises their implementation.




JAX-RS (REST services):

- Utilize HTTP protocol methods, such as:
  - GET
  - PUT
  - POST
  - DELETE
- Can transport any data, for example:
  - XML
  - JSON
- Typically used by browser and mobile UI

JAX-WS (SOAP services):

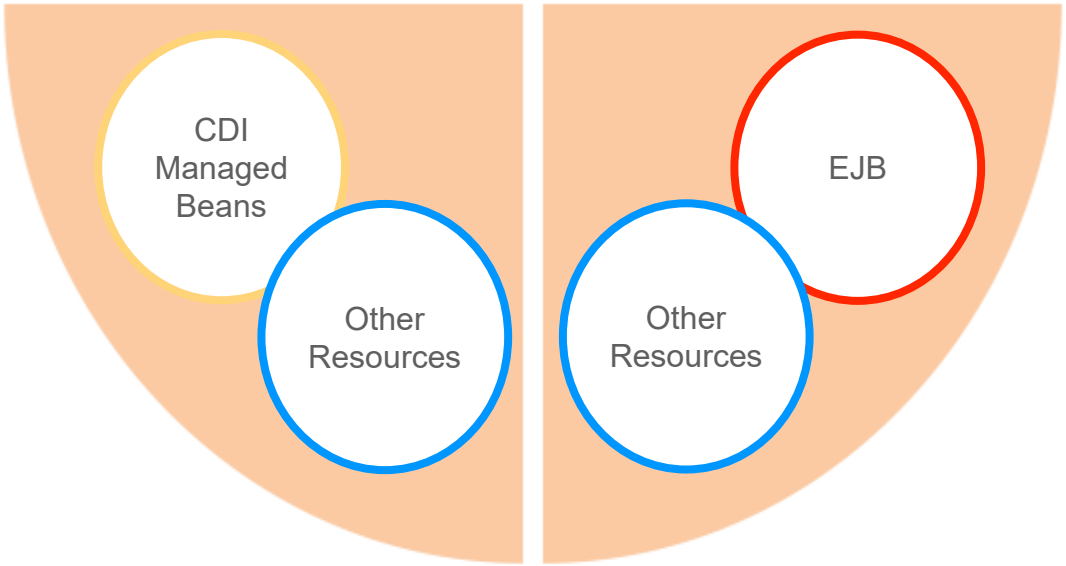
- Are transport protocol–independent
- Use standard WSDL descriptors
- Can transport XML described via XSD
- Provide a range of WS-* standard policies, such as:
  - WS-Security
  - WS-Reliability
  - WS-Addressing
  - WS-Transactions
- Typically used for system integration purposes and in SOA architecture

# Java EE 7 Business Logic Handling Technologies

- CDI managed bean:
  - Life cycle determined by memory scope context: Request, View, Session, Application, Dependent, Conversation, and so on
  - Can be invoked only locally
- EJB (Enterprise JavaBean):
  - Life cycle determined by the type of bean:
    - Session Beans: Stateless, Stateful, Singleton
    - Message Beans: Message-Driven
  - Can be invoked locally or remotely
- Other resources:
  - EntityManager, JMS Queue or Topic, DataSource, EJBContext and so on represent container-managed resources.

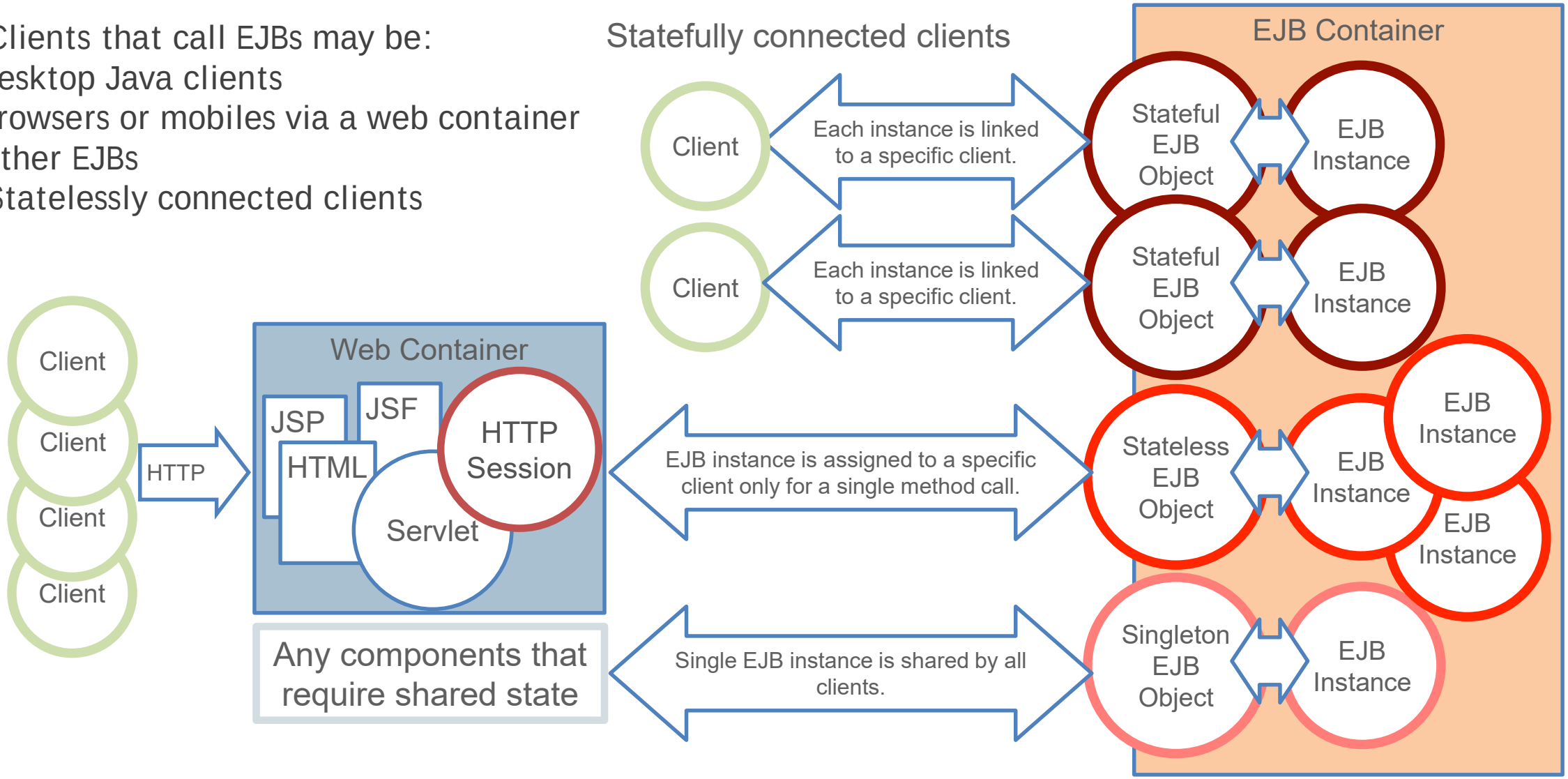
Model implementation:

- Contains your business logic
- Contained within CDI managed beans
- or Enterprise JavaBeans
- or both



# Session EJB Types

- Clients that call EJBs may be:
  - Desktop Java clients
  - Browsers or mobiles via a web container
  - Other EJBs
- Statelessly connected clients

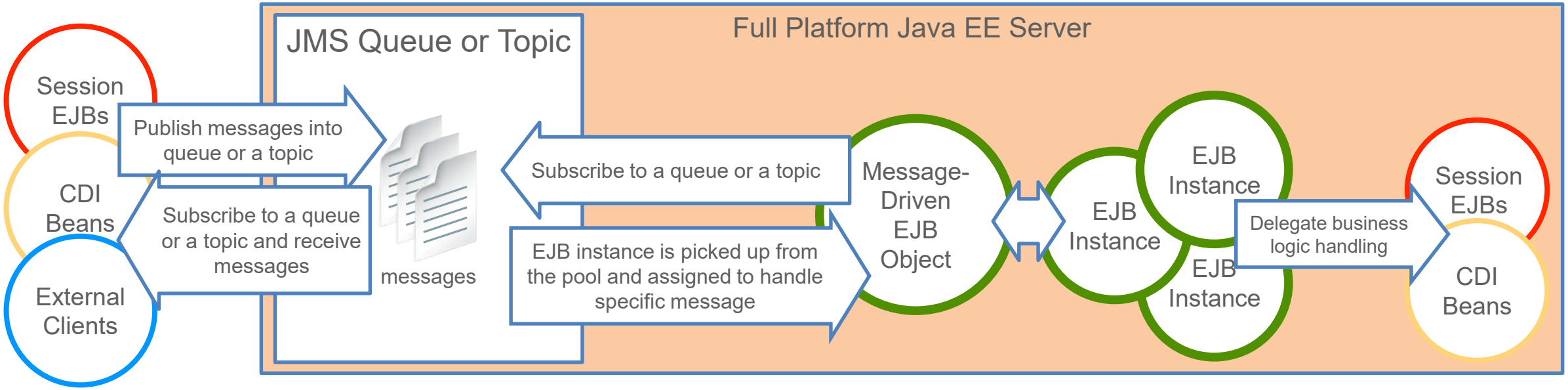


# Message-Driven EJB

- Message producers and consumers:
  - Java EE components
  - External clients

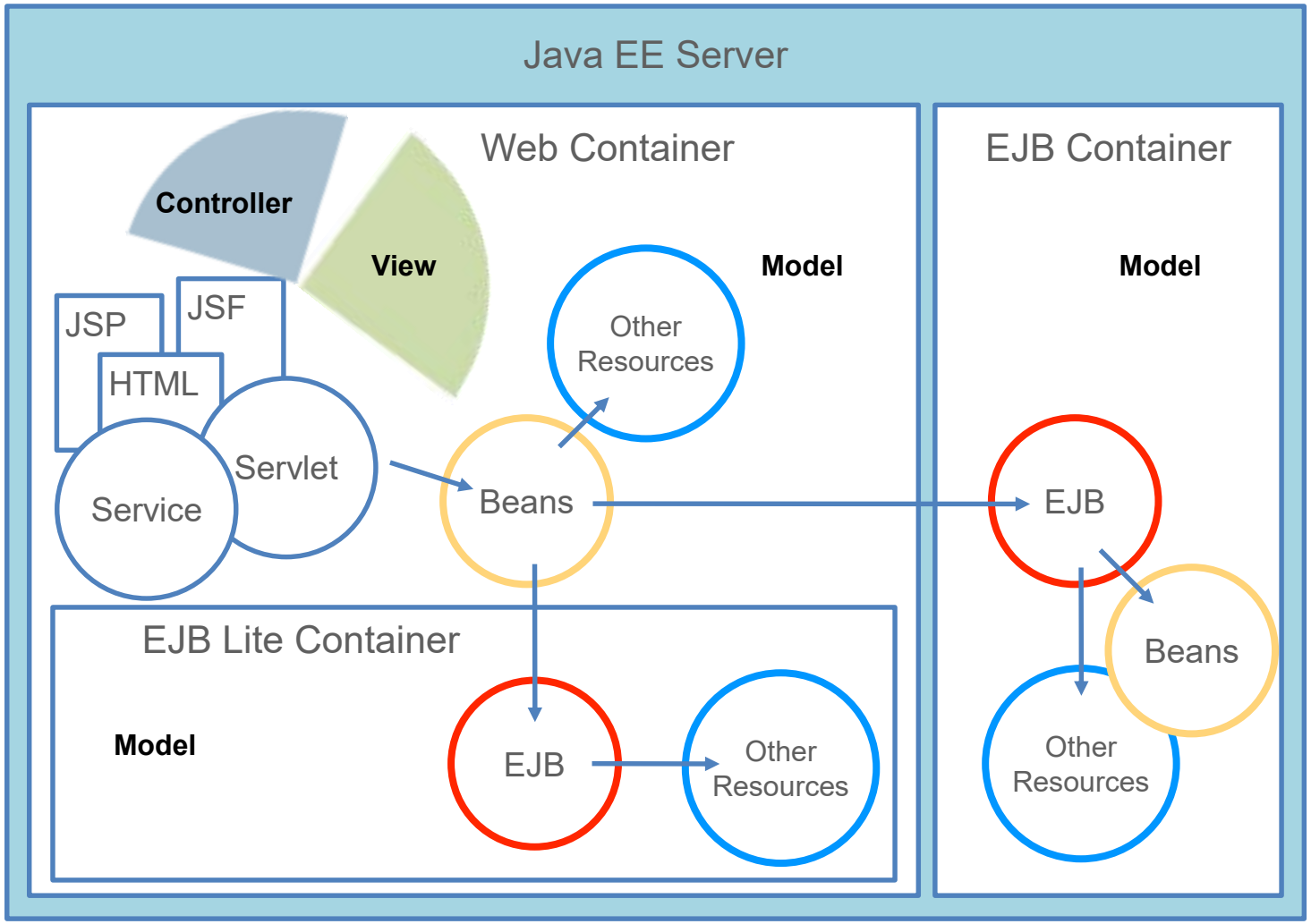
Message-driven bean:

- Stateless asynchronous message consumer
- Can be subscribed to receive messages from queues or topics
- Can never be directly invoked by a client
- Typically is responsible for acquiring, validating, and preparing messages before passing on the business logic handling classes



# Assembling Application Components with CDIs

- Context Dependency Injections and annotations are used to reference objects such as:
  - POJOs
  - JSF managed beans or CDI beans
  - EJBs
  - Other resources such as
    - EntityManager
    - JMS queues or topics
    - EJBContext
    - ServletContext
    - And so on



# JSF Managed Beans, CDI Beans, EJBs

- Enterprise JavaBeans (EJBs):
  - Described by the javax.ejb package
  - Can be used in the EJB Java EE container
  - Can be called locally and remotely
  - Can be stateful and be passivated
  - Can work with timers
  - Can be invoked asynchronously

CDI beans:

- Described by the javax.enterpise.context package
- Can be used in all Java EE Containers—not limited to JSF
- Support interceptors, events, and so on, and are more flexible than JSF Managed Beans
- In addition to annotations, may use beans.xml deployment descriptor

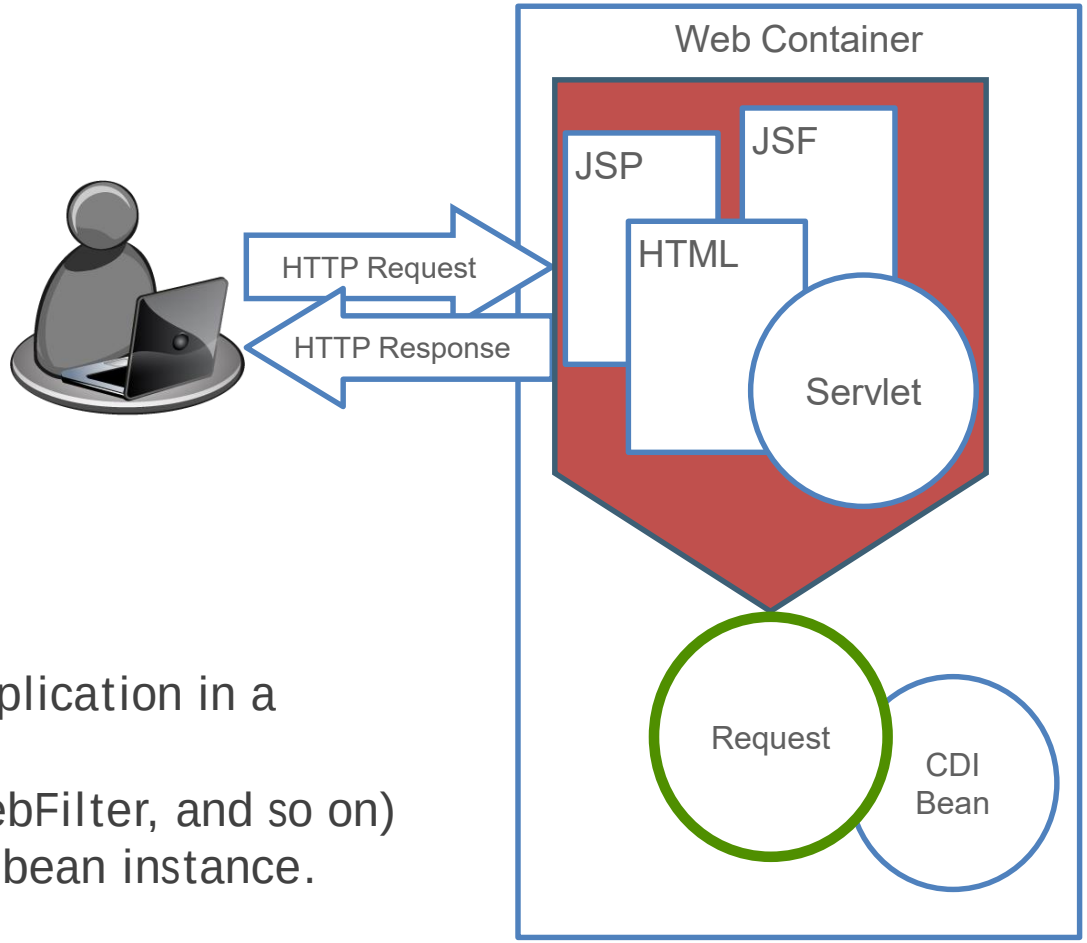
JSF managed beans:

- Described by the javax.faces.bean package
- Used in the web container by JSF Components
- CDI Beans "evolved" from JSF managed beans

❖ The following examples demonstrate the use of CDI beans in a context of a web container. However, they could also be used in EJB container.

# Request Scope

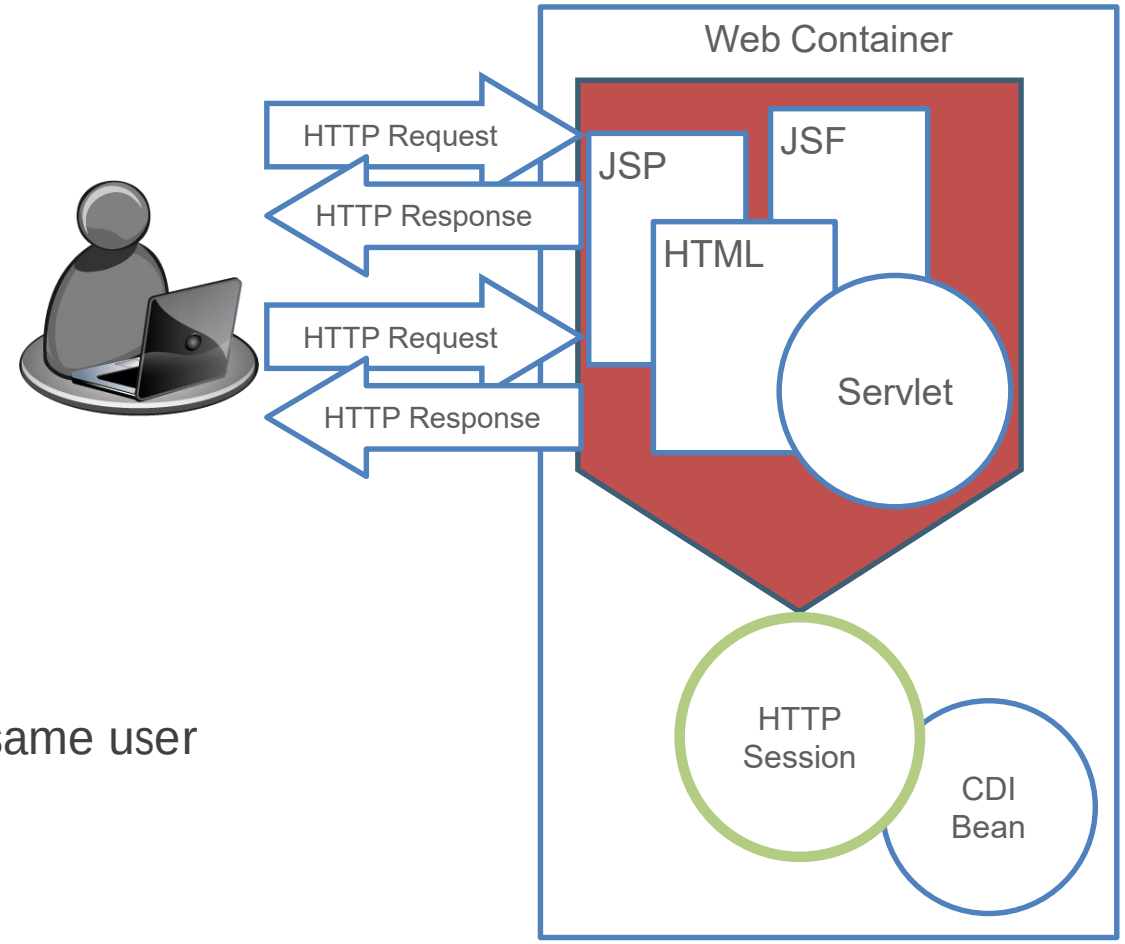
```
package demos;

import javax.enterprise.context.*;

@RequestScoped
public class MyBean {
   public void doSomething() {
      //...
   }
}
```




- Request scoped beans:
  - Instantiated once per user's interaction with a web application in a single HTTP request
  - More than one server component (servlet, JSP, JSF, WebFilter, and so on) can handle the same client request, sharing the same bean instance.

# Session Scope

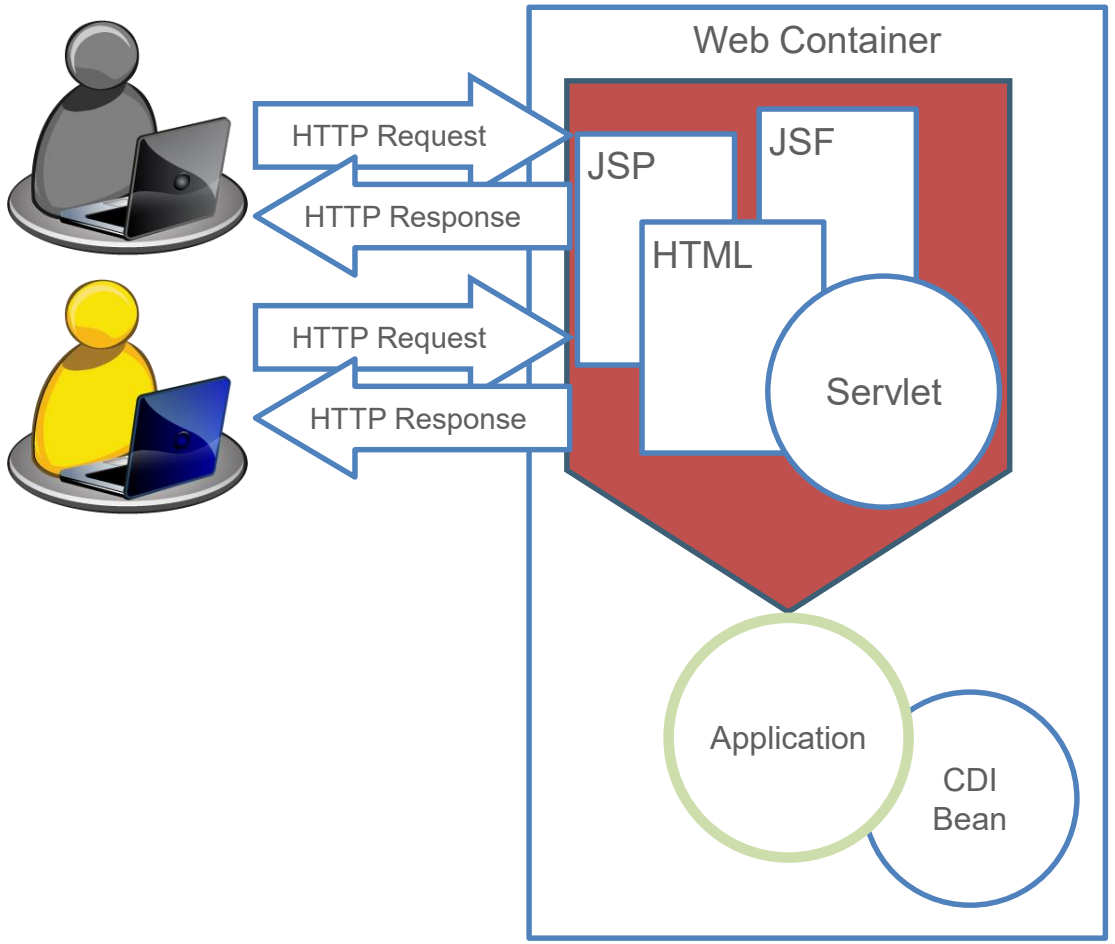
```
package demos;

import javax.enterprise.context.*;
import java.io.Serializable;

@SessionScoped
public class MyBean implements Serializable {
   public void doSomething()
   //...
   }
}
```




- Session Scoped Beans:
  - Instantiated once per user session
  - Shared across multiple HTTP requests from the same user

# Application Scope

```
package demos;

import javax.enterprise.context.*;

@ApplicationScoped
public class MyBean {
   public void doSomething() {
      //...
   }
}
```



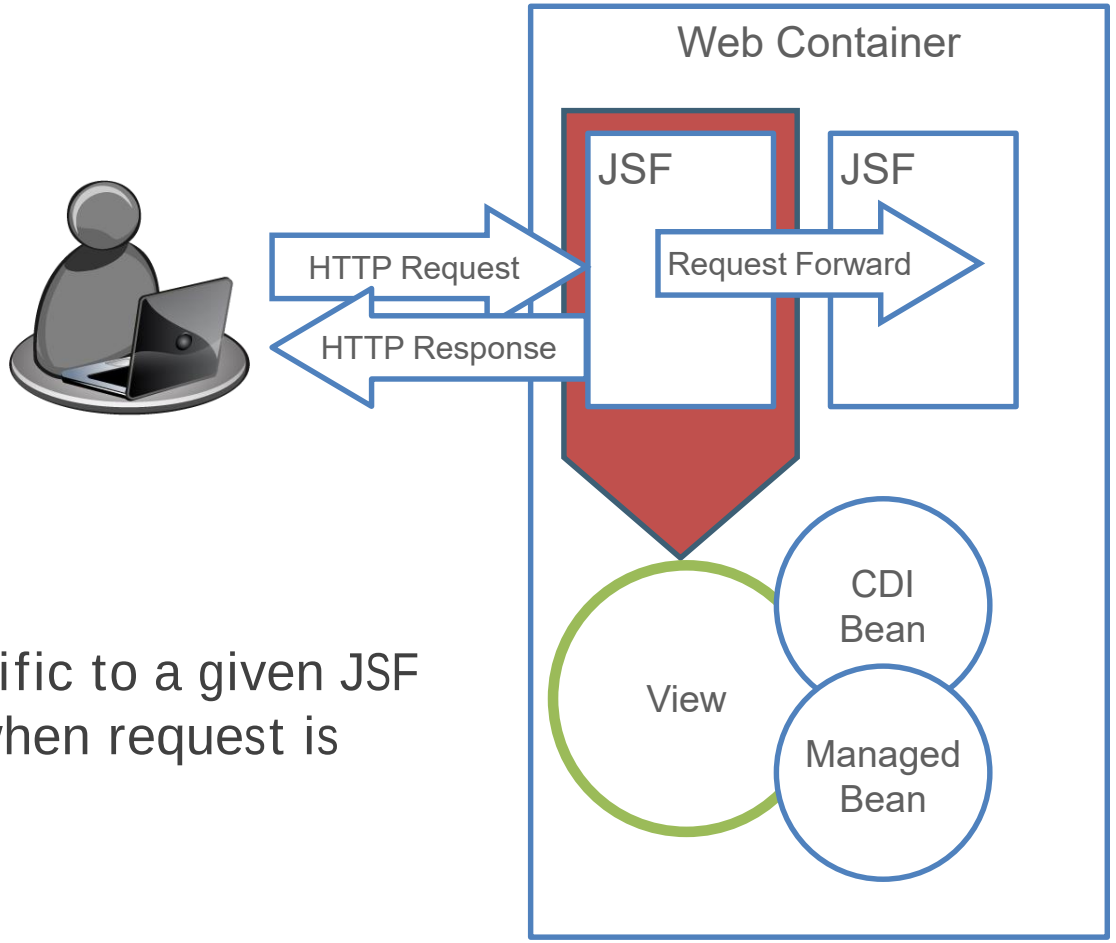

- Application Scoped Beans:
  - Instantiated once per application
  - Shared by all application users

# View Scope

```
package demos;
import javax.inject.*;
import javax.faces.view.*;
@Named("aBean")
@ViewScoped
public class MyCDIBean { ... }
```

OR

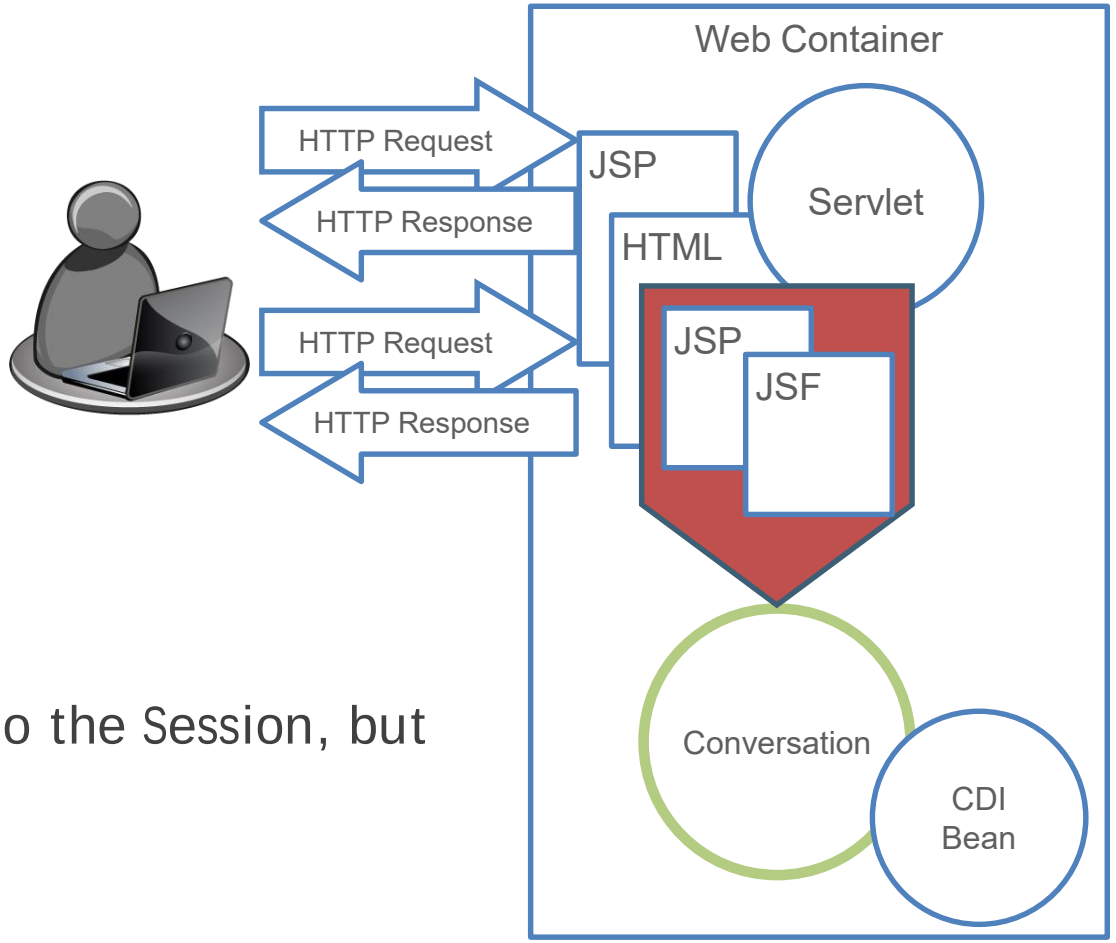
```
package demos;
import javax.faces.bean.*;
@ManagedBean("bBean")
@ViewScoped
public class MyManagedBean { ... }
```




- View Scoped is a JSF-specific scope that is specific to a given JSF page. It is not "shared" with another JSF page when request is forwarded to it during navigation.

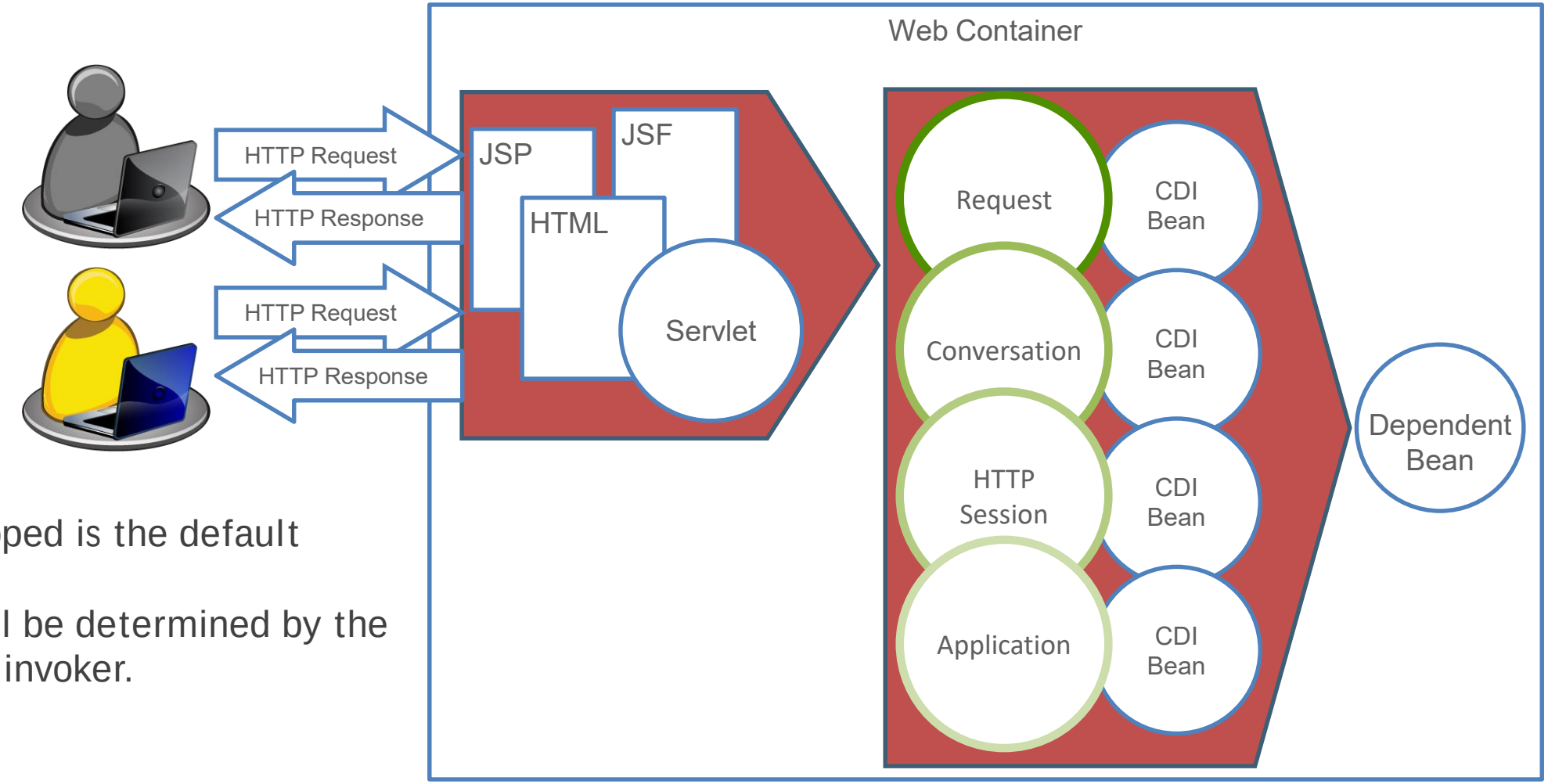
# Conversation Scope

```
package demos;

import javax.enterprise.context.*;
import java.io.Serializable;

@ConversationScoped
public class MyBean implements Serializable {
   @Inject
   private Conversation conversation;
   public void startConversation() {
      conversation.begin();
   }
   public void endConversation() {
      conversation.end();
   }
}
```




- Conversation Scoped is a custom scope similar to the Session, but can be limited to specific part of application.

# Dependent Scope




- Dependent Scoped is the default scope.
- Bean scope will be determined by the context of the invoker.

# Injecting Beans

ProductOrder bean definition with "Product" property, "placeOrder" operation, and optional alias "order":

```
package demos;
@Named("order")
@RequestScoped
public class ProductOrder {
  private String productName;
  public String getProduct() {
    return productName;
  }
  public void setProduct(String name) {
    productName = name;
  }
  public void placeOrder() {
    //...
  }
}
```

Injection of the ProductOrder bean into JSF page using "order" alias:

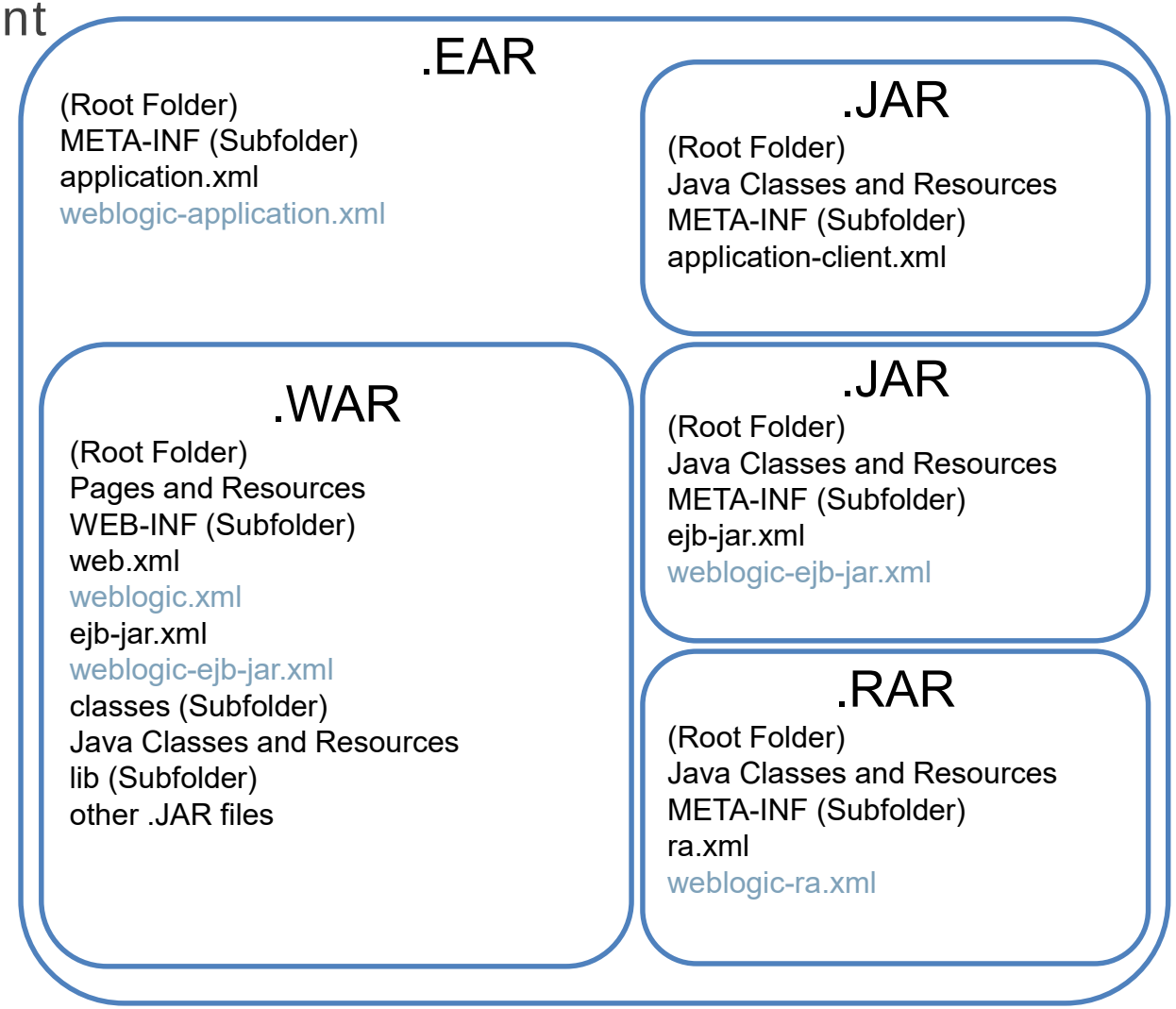
```
<h:form>
 <h:inputText id="name" value="#{order.product}"/>
 <h:commandButton value="Ok" action="#{order.placeOrder}"/>
</h:form>
```

Injection of the ProductOrder bean into OrderManagement class:

```
package demos;
public class OrderManagement {
  @Inject;
  private ProductOrder productOrder;
  public void handleOrder() {
    String name = productOrder.getProduct();
    productOrder.placeOrder();
  }
}
```

# Java EE Packaging and Deployment

- JSR 88: Java EE Application Deployment
  - Enterprise Archive
    - Packaged as EAR files
    - Contains other modules
  - Web Module - WAR files
    - Packaged as WAR files
    - Contains web content such as HTML, servlet, JSP, JSF etc..
    - May contain library or EJB JAR files
  - EJB Module
    - Packaged as JAR files
    - Contains Enterprise JavaBeans
  - Resource Adapter Module
    - Packaged as RAR files
    - Contains JCA adapters
  - Application Client Module
    - Packaged as JAR files
    - Contains Java client applications



# Annotations or Deployment Descriptors

- In Java EE 7, deployment descriptors are optional - a developer may use annotations instead.
- This example shows an EJB component described both ways.

Using Annotations:

```
package demos;

@Stateless(name="Orders")
public class OrderEntry {
   public void placeOrder() {
    //...
  }
}
```

These two options are available for all Java EE components.

Using ejb-jar.xml Deployment Descriptor:

```
<ejb-jar xmlns="http://xmlns.jcp.org/xml/ns/javaee"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
         http://xmlns.jcp.org/xml/ns/javaee/ejb-jar_3_2.xsd"
         version="3.2">
 <enterprise-beans>
    <session>
      <ejb-name>Orders</ejb-name>
      <ejb-class>demos.OrderEntry</ejb-class>
      <session-type>Stateless</session-type>
      <transaction-type>Container</transaction-type>
    </session>
  </enterprise-beans>
</ejb-jar>
```

# Annotations with Deployment Descriptors

- Deployment descriptors and annotation can be used together.
  - Annotations are convenient.
  - Descriptors are flexible.
  - Properties and behaviors of all
  - Java EE components can be
  - adjusted via XML descriptors
  - without changing annotations
  - in the source code.

```
package demos;

@Stateless(name="Orders")
public class OrderEntry {
  @Resource(name="mailhost")
  String mailServer;
  public void placeOrder() {
    //...
  }
}
```

```
<ejb-jar xmlns="http://xmlns.jcp.org/xml/ns/javaee"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
         http://xmlns.jcp.org/xml/ns/javaee/ejb-jar_3_2.xsd"
         version="3.2">
 <enterprise-beans>
    <session>
      <ejb-name>Orders</ejb-name>
      <ejb-class>demos.OrderEntry</ejb-class>
      <session-type>Stateless</session-type>
      <transaction-type>Container</transaction-type>
      <env-entry>
          <description>EMail server host</description>
          <env-entry-name>mailhost</env-entry-name>
          <env-entry-type>java.lang.String</env-entry-type>
          <env-entry-value>smtp.example.com</env-entry-value>
      </env-entry>
    </session>
  </enterprise-beans>
</ejb-jar>
```

# Java Naming Directory Interface Objects

- JNDI is used to catalog various types of objects, such as:
  - EJBs
  - JMS queues, topics, connection factories
  - Data sources
  - LDAP objects
  - etc…

```
@Stateless
public class OrderManagement {...}
```

```
@DataSourceDefinition (
      name="java:global/ProductsApp/productDB",
      className="org.apache.derby.jdbc.ClientDataSource",
      url="jdbc:derby://localhost:1527/ProductDB",
      user="...",
      password="...",
      databaseName="ProductDB"
   )
```

Global JNDI naming convention:

```
java:global/<application-name>/<module-name>/<bean-name>
                        java:app/<module-name>/<bean-name>
                                   java:module/<bean-name>
                                                <bean-name>
```

# Container-Managed Injections

- Components in Java EE environment use annotations to inject resources and dependencies.
- Injecting Resources:
  - Data sources
  - JMS queues, topics
  - etc…

```
@Resource(lookup="java:global/ProductsApp/productDB")
private DataSource myDB;
```

Defining Components:

- CDI beans
- Enterprise JavaBeans

```
@RequestScoped
public class ProductOrder {...}
```

```
@Stateless
public class OrderManagement {...}
```

Injecting Dependencies:

- CDI beans
- Enterprise JavaBeans

```
@Inject
private ProductOrder po;
```

```
@EJB
private OrderManagement om;
```

# JDNI Lookups

- Components outside of the Java EE container environment perform explicit JNDI lookups.
- Create Initial Context Object to reference JNDI:
  - Using Java code to set server context properties:

```
Hashtable env = new Hashtable();
env.put(Context.INITIAL_CONTEXT_FACTORY,
        "weblogic.jndi.WLInitialContextFactory");
env.put(Context.PROVIDER_URL,
        "t3://localhost:7001");
Context context = new InitialContext(env);
```

  - Or using the `jdni.properties` file:

```
java.naming.factory.initial=weblogic.jndi.WLInitialContextFactory
java.naming.provider.url=t3://localhost:7001
```

Perform lookups using JNDI object names:

```
Context context = new InitialContext();
```

```
DataSource ds = (DataSource)context.lookup("<data-source-name>");
OrderManager om = (OrderManager)context.lookup("<ejb-name>");
```

# Summary

- In this lesson, you studied the fundamentals of Java EE 7 Architecture and its components. After completing this lesson, you should have learned how to describe:
  - Standards, containers, APIs, and services
  - Application component functionalities mapped to tiers and containers
    - Web container technologies
    - Business logic implementation technologies
    - Web service technologies
  - Packaging and deployment
  - Enterprise JavaBeans, managed beans, and CDI beans
    - Understanding lifecycle and memory scopes
  - Linking components together with annotations, injections, and JNDI

# Agenda

- Tuning Web Applications
- Tuning Enterprise JavaBeans

# What Is a JavaServer Page?

- JavaServer Pages:
  - Are templates for dynamic content
  - Extend HTML with custom Java code
  - Are compiled into servlets by WebLogic Server (WLS)
  - Allow for the division of labor into content production and programming

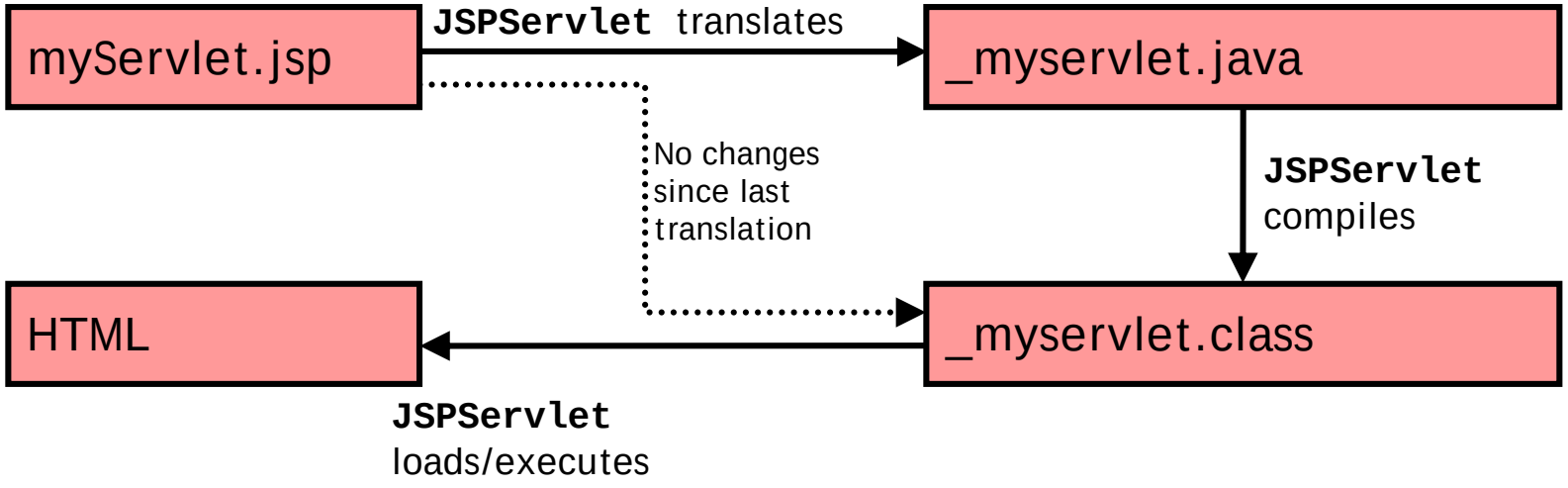
```
<H1>Today is:</H1> <%= new java.util.Date() %>
```

↓

```
Today is: Mon May 09 17:10:25 EDT 2006
```

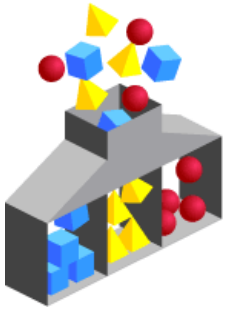
# JSP Request Processing

- Requests for JSPs are handled by a special WLS servlet called `JSPServlet`, which handles:
  - Translation of JSP into a servlet
  - Compilation of a resulting servlet into a class
  - Execution of JSP



# Precompiled JSP

- Precompiling JSPs save:
  - Initial time for the server translation and compilation
  - Memory required for noncompiled JSPs
  - Management related to noncompiled JSPs
- JSP precompilation can be done:
  - Manually, using WebLogic compilers such as `weblogic.appc`
  - Automatically, using the precompile option in `weblogic.xml`
  - If the precompile option is turned ON, recompilation
- of JSPs occurs each time the server restarts.



# The appc Application Compiler

- The appc compiler compiles EJBs and JSPs, and generates the classes needed to deploy to WebLogic Server.
- The appc compiler provides the following benefits:
  - Flexibility of compiling all modules of an application
  - Validation checks throughout an entire application as well as individual modules
  - Easier identification and correction of errors
  - Reduction in time and effort in repeated compilations—when deploying to multiple servers
- Syntax for using appc:

```
$> java weblogic.appc -options <application archive file or
   directory>
```

# Using the `precompile` Parameter

- You can configure WebLogic Server to precompile your JSPs when a web application is deployed or redeployed.
- Set the `precompile` parameter to true in the
- `<jsp-descriptor>` element of the `weblogic.xml` deployment descriptor.
- Snippet from `WEB-INF/weblogic.xml`:

In UNIX environments, the following parameter would help address mixed-case JSP names:

- `-Dweblogic.jsp.windows.caseSensitive=true`

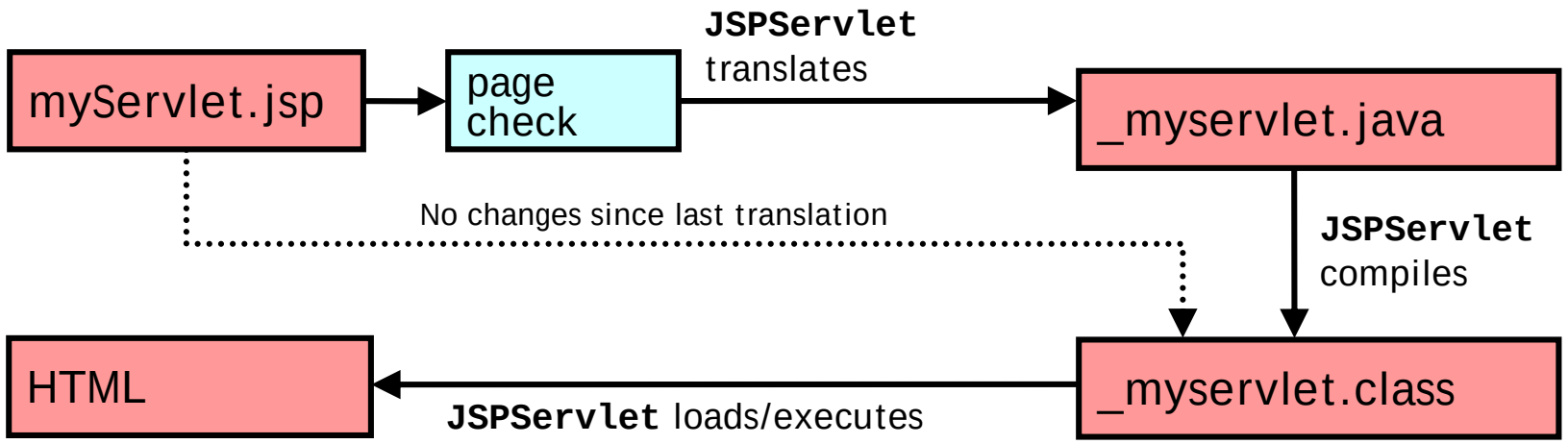
```
<jsp-descriptor>
        <precompile>true</precompile>
</jsp-descriptor>
```

# HttpSession Replication Tuning

- WebLogic Server replicates only the session's new/changed attributes.
- Keep session objects as small as possible.
- It is better to store lots of attributes rather than one.
- Put only Serializable objects in the session.
  - Failure to do so will prevent replication
- Monitor primary distributions across the cluster.
  - Uneven distributions likely means the Load Balancer/Proxy isn't properly distributing the load.

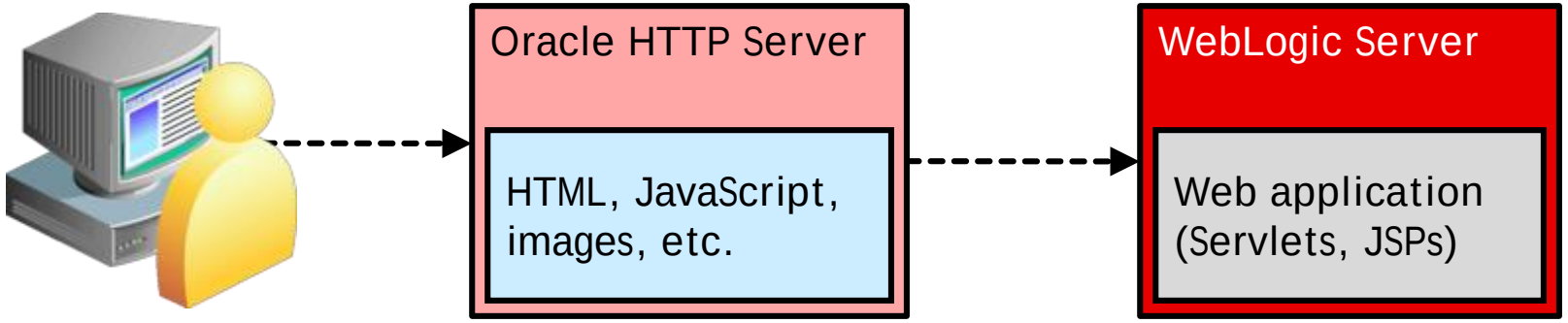
# Page Check Interval

- Whenever a JSP or a servlet have been modified, it must be recompiled by the server.
- The frequency at which a server checks for modifications to JSPs or servlets can affect performance.



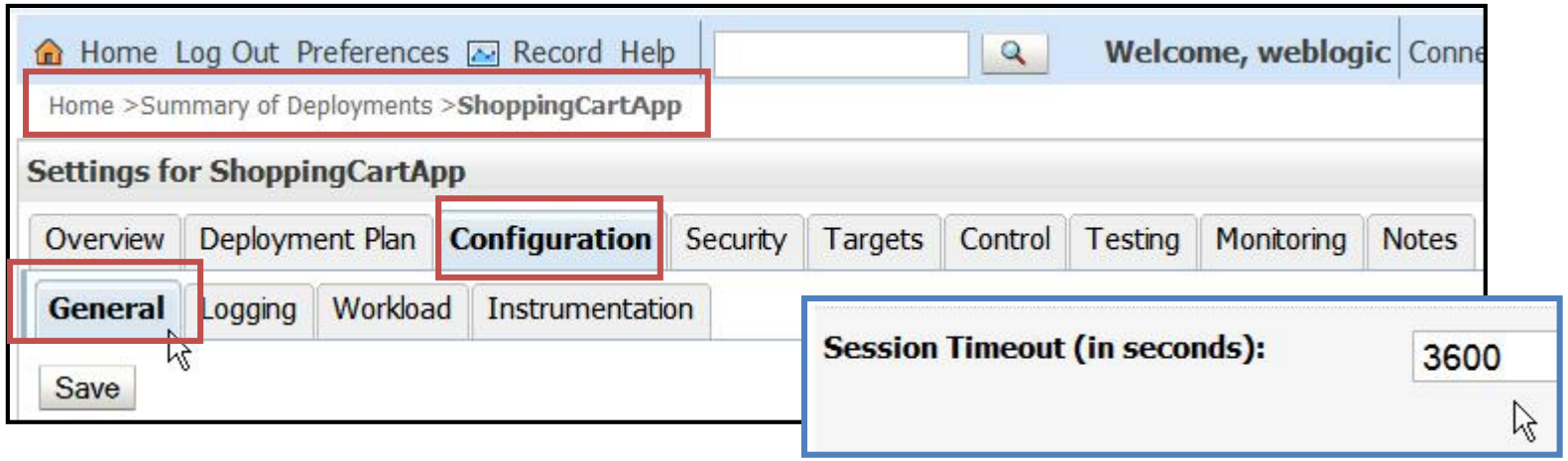
# Using Web Servers for Static Content

- Front application servers and clusters with a dedicated web tier to serve static content such as HTML and image files
- Proxy requests for servlets and JSPs to application servers
- Perform additional optimizations at the web tier, such as caching.



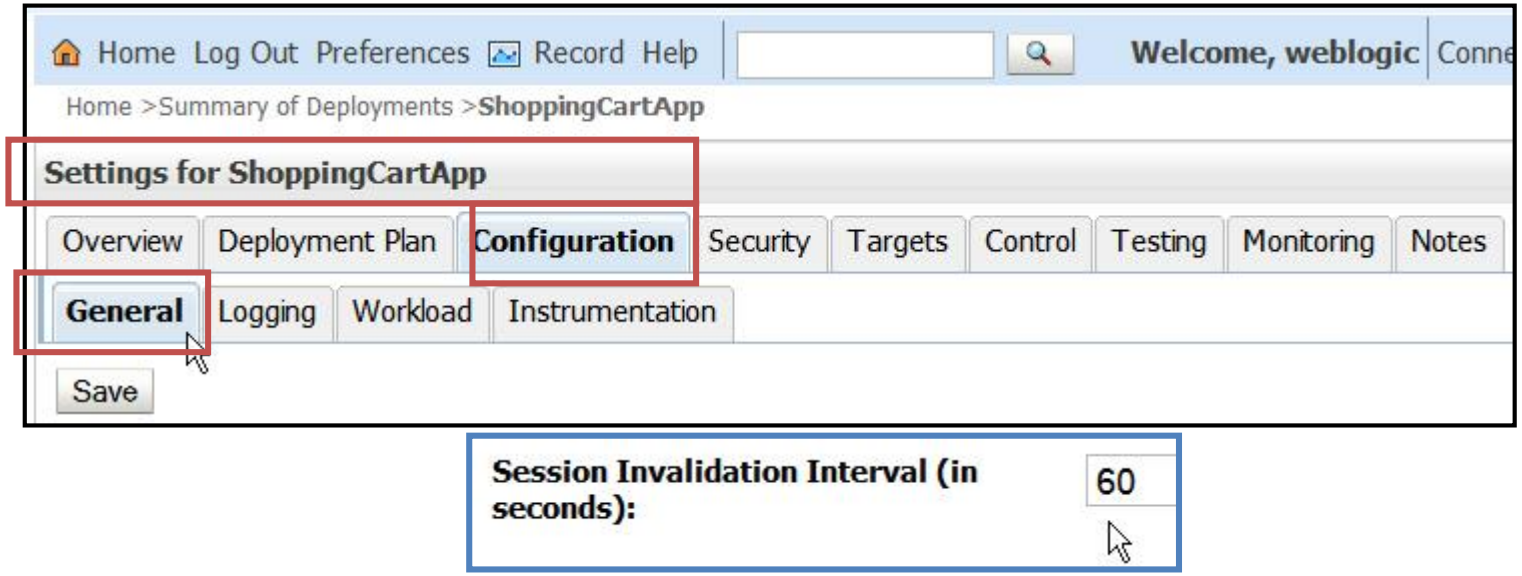
# Session Timeout

- Whenever an HTTP session object is created, it resides in memory until:
  - The session object is invalidated
  - The server removes it after a certain period of time (`session timeout`)
  - The server comes down
- You can set the session timeout parameter by using the Administration Console:



# Session Invalidation

- Set the `invalidation-interval-secs` parameter to an appropriate value.
- Tuning this parameter can improve the performance of applications that have high traffic.



# Using Custom JSP Tags

- Oracle provides three specialized JSP tags that you can use in your JSPs:
  - `cache`: Enables caching the work that is done within the body of the tag
  - `process`: Enables you to control the flow of query parameter-based JSPs
  - `repeat`: Enables you to iterate over many different types of sets, including Enumerations, Iterators, Collections, and Arrays of Objects

# Using the WebLogic cache Tag

- Caching data can boost performance significantly.
- The WebLogic cache tag enables caching the work that is done within the body of the <wl:cache> tag.
- The cache tag supports both output and input caching.
- The tag supports refreshing and flushing the cache at various scopes.

- Example of using the WebLogic cache tag:

```
<wl:cache name="holidaycache"
    key="parameter.holidaytable"scope="application">
  // Retrieve Holidays and output it to the page
</wl:cache>
```

# Web Application Tuning JSP and Servlet Output Buffer Tuning

- Web Container buffers output while building response:
  - Data flushed to client once buffer is full
  - Cannot forward request once data sent to client
    - Set buffer size accordingly
- **`ServletResponse.setBufferSize(int)`**
- **`<@page contentType="text/html; buffer="64kb" %>`**
  - WLS default buffer size is 8 kb.
  - WLS default buffer size on Exalogic is 64 kb.
- Avoid calling **`flush()`** and **`close()`** on **`ServletOutputStream`**

# Agenda

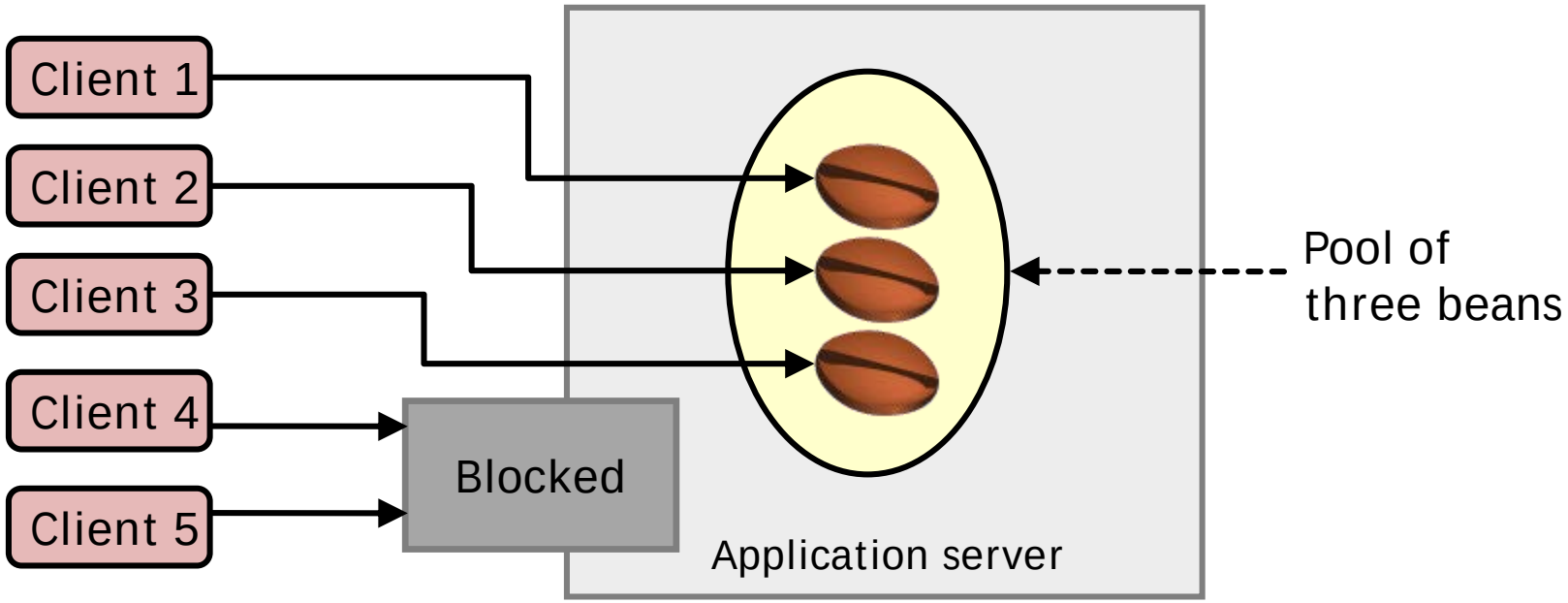
- Tuning Web Applications
- Tuning Enterprise JavaBeans

# Enterprise JavaBeans

- Enterprise JavaBeans (EJBs) standardize the development and deployment of server components built in Java.
- The EJB specification discusses four types of objects:
  - Stateless session beans
  - Stateful session beans
  - Singleton session beans
  - Message-driven beans
- Note that entity beans specifications are considered legacy and are not present in the current Java EE specifications.

# Stateless Session Beans

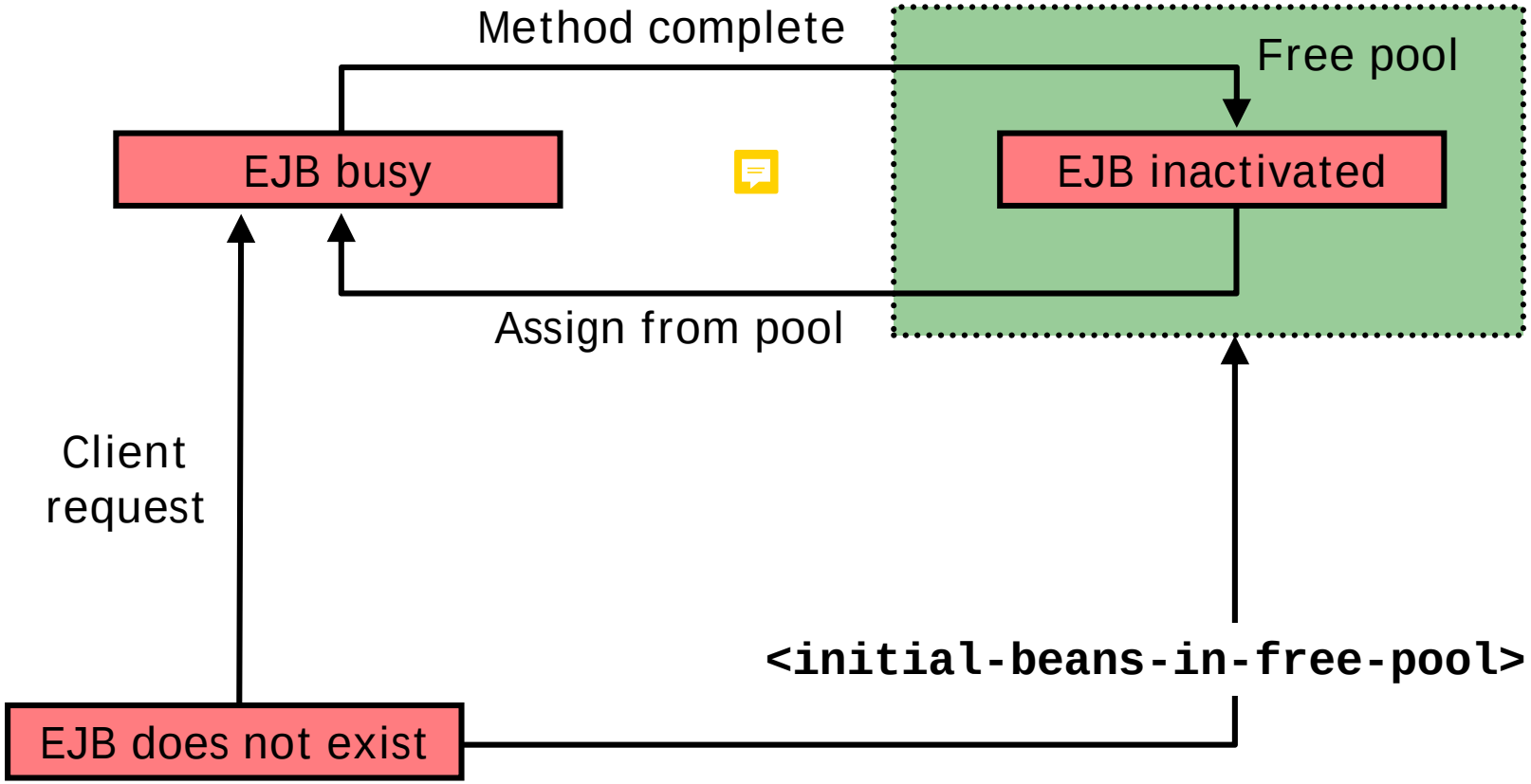
- Stateless session beans are maintained as a pool in memory.
- If the pool size is three, and if five clients attempt to use one EJB type, two clients are blocked.
- Stateless EJBs in a pool are identical. So a server can assign any *available* EJB to any client.



# Stateless Session Beans Tuning

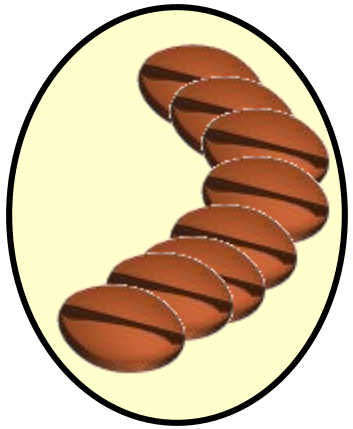
- The defaults are typically good enough.
  - **<initial-beans-in-free-pool>** = 0
  - **<max-beans-in-free-pool>** = 1000
  - There is a cost with creating a new instance in the pool. Thus, tuning the initial-beans-in-free-pool might be beneficial if the runtime cost is too much.
  - This may increase the time to start up, but reduces the runtime cost.

# Pool Management



# Determining the Pool Size

- You can improve performance by tuning the number of bean instances in this free pool.
- Factors to consider to determine how many EJBs should be in the pool include:
  - The number of threads set in WebLogic Server (WLS)
  - The number of concurrent clients
  - The number of dependent back-end resources (for example, database connections)



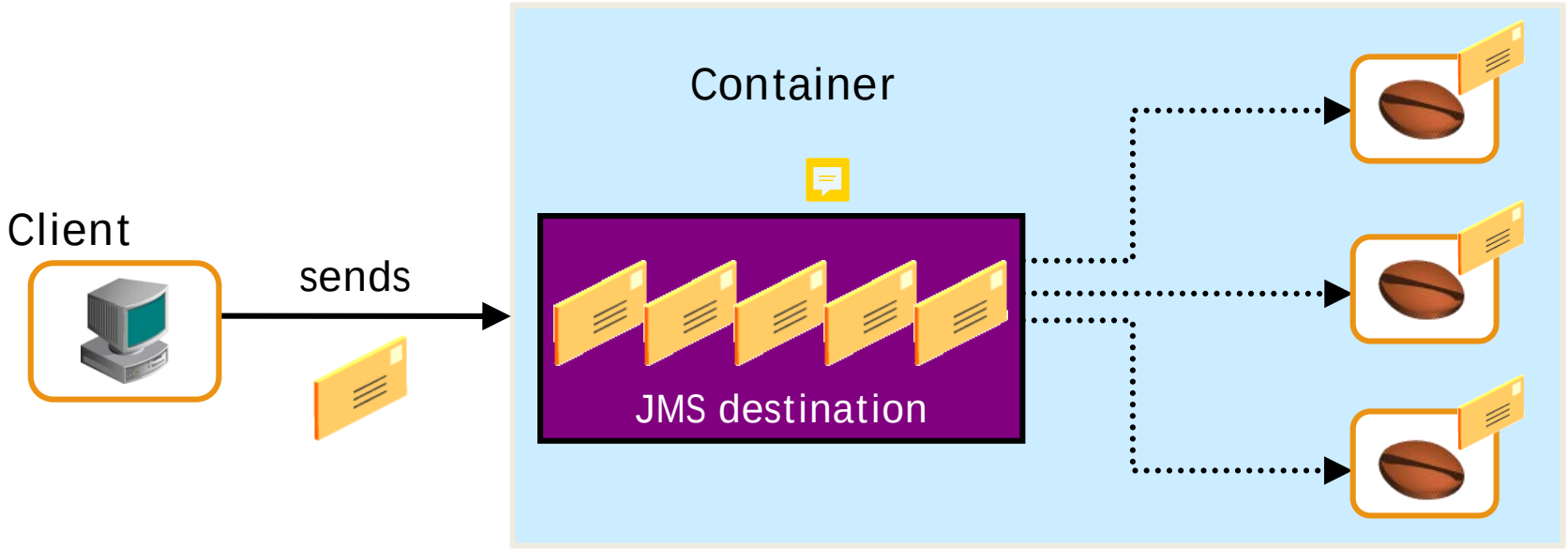
# Configuring a Stateless Session EJB Pool

- You can manage the stateless session EJB pool by:
  - Capping the number of instances
  - Setting an initial pool size

```
<weblogic-enterprise-bean>
    <ejb-name>InsuranceQuoteBean</ejb-name>
    <stateless-session-descriptor>
      <pool>
        <max-beans-in-free-pool>15</max-beans-in-free-pool>
        <initial-beans-in-free-pool>5</initial-beans-in-free-
    pool>
      </pool>
    </stateless-session-descriptor>
    ...
</weblogic-enterprise-bean>
OR
@Stateless(name="InsuranceQuoteBean", maxBeansInFreePool="15",
    initialBeansInFreePool="5")
```

# Message-Driven Beans

- Message-driven beans (MDBs) are:
  - Asynchronous stateless components
  - JMS message consumers
    - Clients do not interact directly with MDBs.



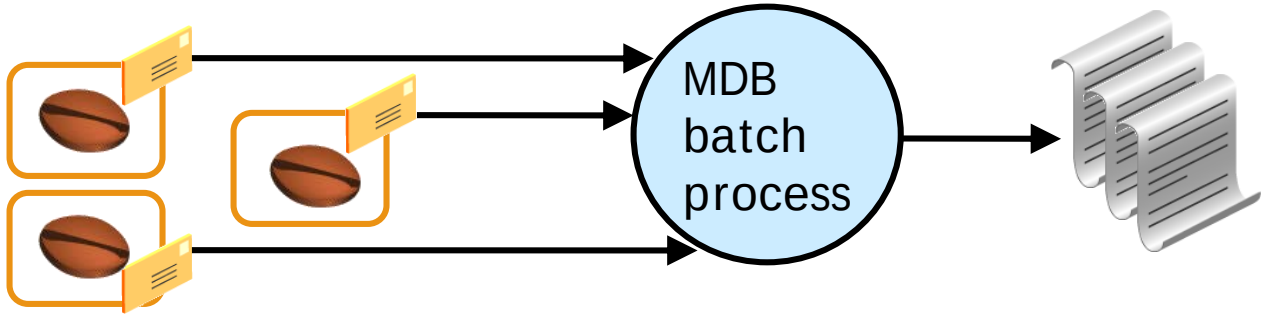
# Configuring an MDB Pool

- You can manage the message-driven EJB pool by setting:
  - The initial pool size
  - The maximum size
  - `idle-timeout-seconds`

```
<weblogic-enterprise-bean>
  <ejb-name>InsuranceQuoteBean</ejb-name>
   <message-driven-descriptor>
    <pool>
     <max-beans-in-free-pool>15</max-beans-in-free-pool>
     <initial-beans-in-free-pool>5</initial-beans-in-free-pool>
    </pool>
    </message-driven-descriptor>
</weblogic-enterprise-bean>
OR
@MessageDriven(name="InsuranceQuoteBean",
   maxBeansInFreePool="15", initialBeansInFreePool="5")
```

# Configuring to Use Batching with an MDB

- Group transactions to reduce writing transaction logs.
- You can configure these parameters in `weblogic-ejb-jar.xml` to optimize MDB processing:
  - `max-messages-in-transaction`
  - `trans-timeout-seconds`
  - Each MDB listening on topic uses a dedicated daemon-polling thread.
- Each MDB listening on queue uses at least one thread from the dispatch policy.

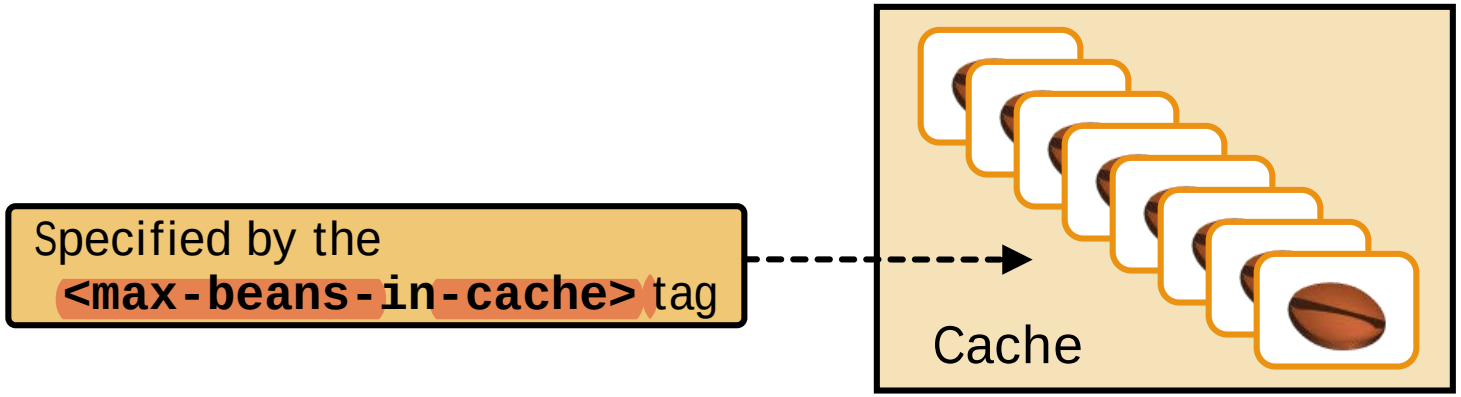


# Stateful Session Beans

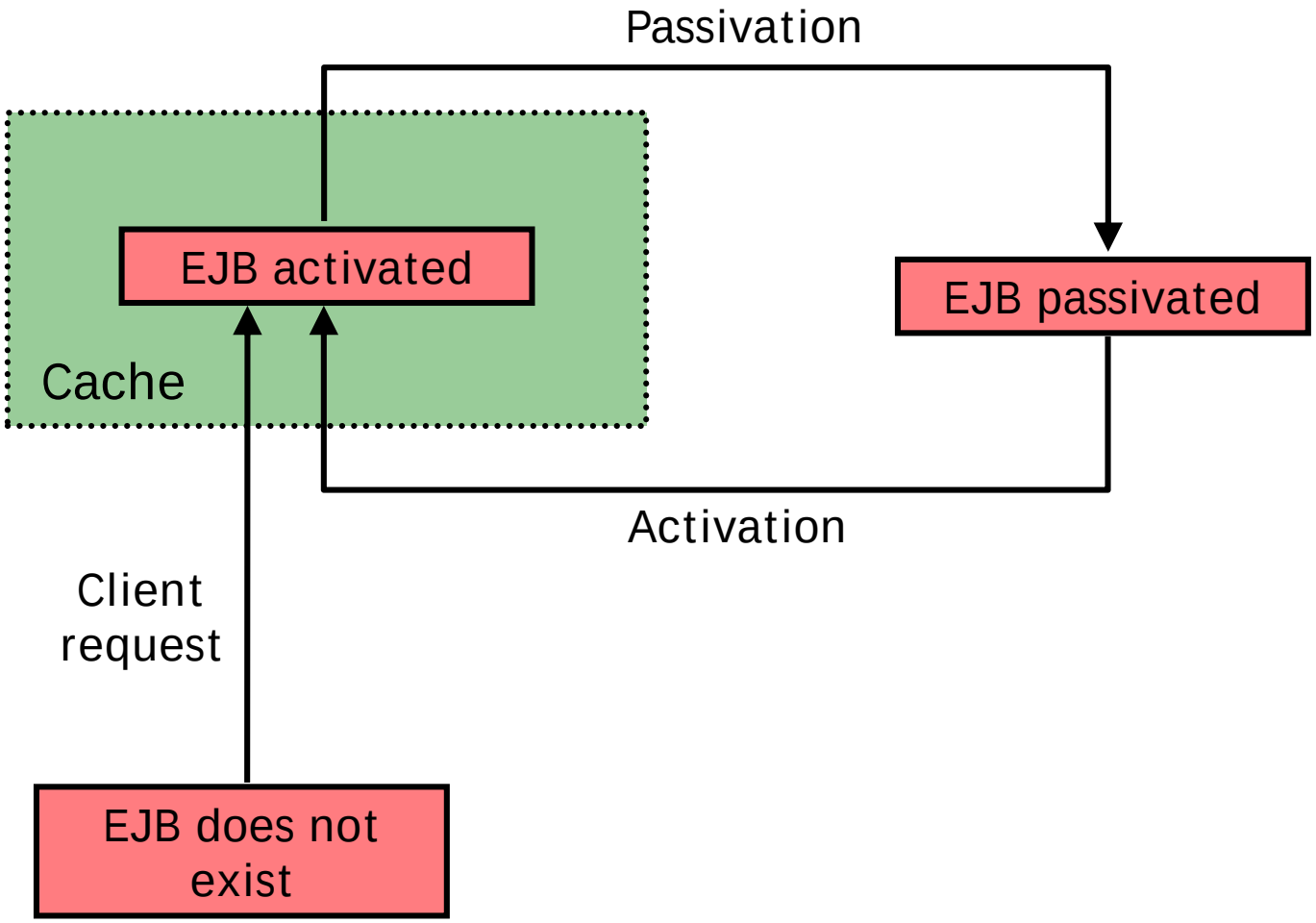
- Stateful session EJBs:
  - Provide conversational interaction
  - Store state on behalf of the client
  - Are associated with a single client
  - Are synchronous
  - Are maintained in memory
- Processing time includes passivation and activation.

# Cache Management

- You can specify the maximum number of stateful session EJB instances allowed in memory for each EJB class.



# Cache Management



# What Happens When the Cache Fills Up?

- WLS has two options to make additional memory available in the cache:
  - Destroy timed-out EJBs.
  - Passivate eligible EJBs to the hard drive.
- Passivation is the act of removing a bean instance from the cache and writing it to a temporary store.
- A passivated bean is activated when the requesting client comes back.
- Activation is the act of retrieving the bean instance from the temporary store to the cache.

# Determining the Cache Size

- Tuning this cache can have a significant impact on performance.
- The criteria that affect the size of a cache include:
  - The total amount of Java heap space available in the VM
  - The number of different stateful sessions
  - The amount of memory that one stateful session bean consumes while active
  - The number of concurrent clients

# Idle Timeout and Eligibility

- An EJB that exceeds its `<idle-timeout-seconds>` value can be destroyed by the WLS.
- An EJB is eligible for passivation if it:
  - Has not exceeded its `<idle-timeout-seconds>`
  - Is not currently executing a method
  - Is not participating in a transaction
- You can tune the `<idle-timeout-seconds>` value for optimal performance.

# Cache Type

- The cache type specifies the order in which EJBs are removed from the cache.
- It includes:
  - Not recently used (NRU), known as lazy passivation
  - Least recently used (LRU), known as eager passivation
- The `idle-timeout-seconds` and `max-beans-in-cache` elements also affect passivation and removal behaviors, based on the value of the cache type.

# Configuring a Stateful Session EJB Cache

- You can manage the stateful session EJB cache by:
  - Capping the number of instances
  - Setting an `idle-timeout` period
  - Setting the passivation strategy for the cached EJBs

```
<weblogic-enterprise-bean>
    <ejb-name>ShoppingCartBean</ejb-name>
    <stateful-session-descriptor>
      <stateful-session-cache>
       <max-beans-in-cache>1000</max-beans-in-cache>
       <idle-timeout-seconds>60</idle-timeout-seconds>
      </stateful-session-cache>
    </stateful-session-descriptor>
</weblogic-enterprise-bean>
OR
@Stateful(name="ShoppingCartBean",
    cacheType=Session.CacheType.N_R_U)
@CacheConfig (maxSize=100000, idleTimeoutSeconds=300,
    removalTimeoutSeconds=0)
```

# Using Filtering ClassLoaders

- Use a Filtering ClassLoader to reduce the number of JAR files WebLogic will search while trying to load a class or resource:
  - Loading class from the current application still has to search all the way up to the system classloader before working its way down to the application's classloader.
  - WLS has hundreds of jar files, making this a somewhat expensive operation.
  - A Filtering ClassLoader short circuits this by searching the Application ClassLoader for configured classes.
    - It will not search parent class loaders for filtered classes!
    - The same concept applies for resources that you know exist only in the application.

# General Application Performance Problems

- Most application performance issues are typically contention-related:
  - Overuse of synchronization
  - Insufficient resources (for example, database connections)
  - Overloaded back-end systems:
    - Load testing can identify most contention
    - Take thread dumps at regular intervals (5-10 secs)
    - Attach a profiler, such as HPROF or VisualVM

# EJBs: Coding for Performance

- Cache JNDI InitialContext, Data Source, JMS Connection Factory and Destination objects:
  - Use the Service Locator pattern to cache lookups.
  - Use Dependency Injection.
  - Use Coherence to cache frequently used data:
    - Can be used explicitly in your application
    - Can be used as a JPA Level 2 cache

Q/A?

PEOPLE COME FIRST
INFORMATIKAI SZAKÉRTŐK EGYESÜLETE

# Oracle Identity Management

# Az Identity Management rendszerek

- Az informatika világában az IDM (vagy IAM – Identity and Access Management) az a feladat, melynek kapcsán a szervezetekhez köthető felhasználókat (identitásokat), azok adatait és jogosultságait kezeljük
- Egy IDM rendszer lehetővé teszi a szervezeti menedzsereknek, hogy az egyes felhasználók információkhoz való hozzáférését hatékonyan irányíthassák
- Főbb feladatai egy ilyen rendszernek:
  - User lifecycle: A felhasználók „életciklusának” kezelése (létrehozás, menedzselés, törlés)
  - Authentication: Egységes belépési módszer biztosítása a szervezet alkalmazásaihoz, vagy akár egy több szervezetből álló domain alkalmazásaihoz is (Identity Federation)
  - Authorization: Jogosultságok nyilvántartása szerepkörökön (Roles) keresztül (ki? mit? mikor? miért?)

# Miért előnyös az IDM egy nagyvállalatban?

- Automatizálja a manuális lépéseket a felhasználókezelésben
- Ezzel nagymértékben csökkenti az időt és költséget, amit a felhasználók kéréseinek teljesítésére kellene fordítani (jelszó módosítás, adatmódosítás, jogosultság kérés, stb.)
- A felhasználói élmény jelentősen javul a Self-Service portálok használatával
- Standard folyamatokat vezet be a szervezeti eljárások (házirend) betartására
- Az Auditálás (Auditing) és Riportolás (Reporting) egyszerűbbé válik

# Oracle Identity Management

- Az Oracle Identity and Access Management egy termékcsalád, ami lehetővé teszi a szervezetek számára a felhasználók egységes és széleskörű kezelését a teljes életciklusuk alatt
- A felhasználóknak lehetőséget ad a szervezeti erőforrások irányított eléréséhez, ehhez Self-Service felületet biztosít
- Teljes körű eszköztárt ad az Authentication és Authorization szervezeti megvalósítására
- Biztosítja a több rendszerek, vagy akár több szervezetek közötti egységes adatáramlást és adatellenőrzést (Provisioning, Reconciliation, Certification)
- Az első termék az Oracle Internet Directory (1999) volt, ami egy LDAP implementáció. Ehhez az Oracle fokozatosan adta hozzá az évek során a komponenseket, kiterjesztve az IDM lehetőségeit
- Az aktuális verzió a 12c

# Oracle IDM előnyei

- Az Oracle IDM közepes- és nagy-vállalatok számára ideális, ahol számítanak a növekedésre és komplex folyamatokkal rendelkeznek
- A termékcsalád egyes komponensei könnyen és támogatottan integrálhatók
- Igény esetén egyszerűen lehet új rendszereket bevonni az IDM folyamatokba
- Nagymértékben szabható a vállalati igényekhez
- A kezdeti fejlesztési költség magas lehet, ha a standard funkcionalitás nem elégséges a vállalat számára
- A dokumentáció széleskörű, de mélyre nyúló fejlesztések esetén konzultáns bevonására lehet szükség

# Komponensek - OIM

- Az „Oracle Identity Manager” a felhasználókezelő rendszer központi alkalmazásainak összesége
- Magába foglalja a felhasználói Self-Service felületet (/identity), a rendszer adminisztrációs felületét (/sysadmin) és a folyamatok működését biztosító backend alkalmazásokat
- Az Identity Weblogic Domain része, dedikált Managed Server példányokon fut

# Komponensek - SOA

- A Service-Oriented Architecture (Szolgáltatásorientált Architektúra) elve, hogy a hálózaton belül elérhető szolgáltatások egységes protokollon keresztül, egymástól függetlenül működve kommunikálhassanak egymással és az alkalmazásokkal
- Előre meghatározott kimenetelű üzleti folyamatokat definiálhatunk a BPEL (Business Process Execution Language) segítségével
- Az IDM-en belül a SOA komponens feladata a szervezet üzleti folyamatainak (Workflow) kezelése. Ilyen folyamat például a jogosultságok kérése, azok elfogadóinak meghatározása
- Az Identity Weblogic Domain része, dedikált Managed Server példányokon fut

# Komponensek - BIP

- Az Oracle Business Intelligence Publisher lehetővé teszi a szervezet felhasználói számára a riportok készítését, használatát és küldését
- Saját felhasználói felülettel rendelkezik (/xmlpserver)
- Az IDM Suite részeként a Standalone verzió áll rendelkezésre, extra licenc nélkül használható (ebből hiányzik sok extra funkcionalitás, például analitika, ami az ~~OBIEE~~ csomagban megtalálható)
- Az adatbázis vagy LDAP lekérdezések írásához adatmodell szerkesztő felületet biztosít, a riportok megjelenítéshez layout tervező áll rendelkezésre
- A riportokat generálhatjuk HTML, PDF, Excel, stb. formátumban
- A riportok generálását és küldését automatizálhatjuk az ütemező segítségével
- A jogosultságok kezelése egyszerűen kapcsolható az IDM-ben definiált szerepkörökhöz
- Az Identity Weblogic Domain része, dedikált Managed Server példányokon fut

# Komponensek - OAM

- Az Oracle Access Manager feladata az erőforráselérés szabályozása
- Két fő területe:
  - Authentication – A felhasználók bejelentkezési kísérletének hitelesítése
  - Authorization – A hitelesített felhasználók erőforrásigényének vizsgálata
- A vállalat szintű bejelentkezések kezelését az Single Sign-On (SSO) technológiával támogatja
- Az Oracle HTTP Server-el a WebGate-en keresztül kommunikál
- Saját felületet biztosít az OAM szolgáltatások konfiguráláshoz (/oamconsole)
- Az Access Weblogic Domain része, dedikált Managed Server példányokon fut
- Igény esetén a domain egyéb szolgáltatásokkal egészíthető ki: Mobile Security, Adaptive Access Manager, Policy Manager

# Komponensek - OHS

- Az Oracle HTTP Server egy Apache alapú webszerver
- Modulokkal egészítette ki az Oracle, amelyek kiterjesztik a funkcionalitását, például képessé teszik a Weblogic-cal való kommunikációra
- Képes terheléselosztásra (load-balance) több Weblogic példány között, vagy egy példány kiesésekor a kérések átirányítására (failover)
- IDM rendszerben „Reverse-proxy”-ként működik
- A WebGate plugin az OHS része, ezen keresztül kommunikál az Access Manager szerverekkel
- Ha van „SSL Termination”, akkor az lehet ezen a szinten, vagy a Load Balancer szintjén
- A Virtual Host-ok segítségével szeparáljuk az alkalmazásokat, például:
  - https://idm.mycompany.com/identity
  - https://oimadmin.mycompany.com/sysadmin

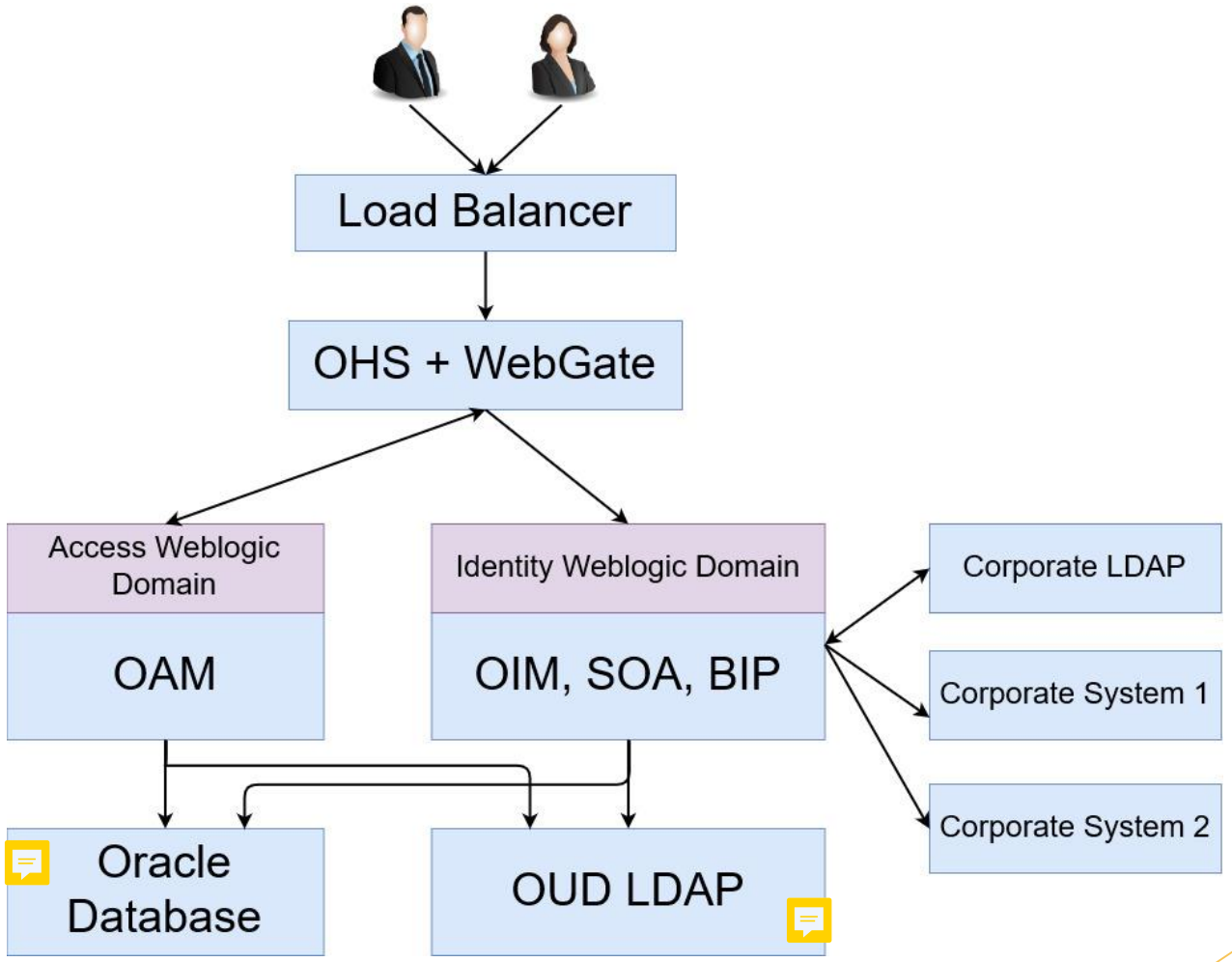
# Komponensek - Adatbázis

- Az IDM Suite telepítésének feltétele egy előre konfigurált Oracle adatbázis
- Minden komponens saját sémával rendelkezik
- A telepítő szoftver létrehozza a szükséges adatbázis felhasználókat, táblatereket és objektumokat
- Minden, a felhasználókhoz és a rendszer működéséhez köthető adat (OIM Metadata) itt van eltárolva

# Komponensek - OUD

- Oracle Unified Directory – újgenerációs LDAP (Lightweight Directory Access Protocol) kezelő szoftver, amit az Oracle a Sun OpenDS-ből alakított ki
- Az adatokat fájlokban tartja (szemben az Oracle Internet Directory-val, ami egy Oracle adatbázisban tárolja az adatokat)
- Képes proxy-ként működni a kliens és egy másik LDAP szerver között
- Szinkronizálhatja (replikálhatja) az adatokat más LDAP szerverek felé (vagy több OUD példány egymás között)
- Az IDM rendszerben az OUD tárolja a felhasználók legfontosabb adatait és jogosultságaikat. Identity Store-nak is hívjuk
- Bejelentkezéskor a felhasználót az OUD-ban tárolt adatok alapján hitelesíti az Access Manager
- Ez a kapcsolati pont az OIM és az OAM között

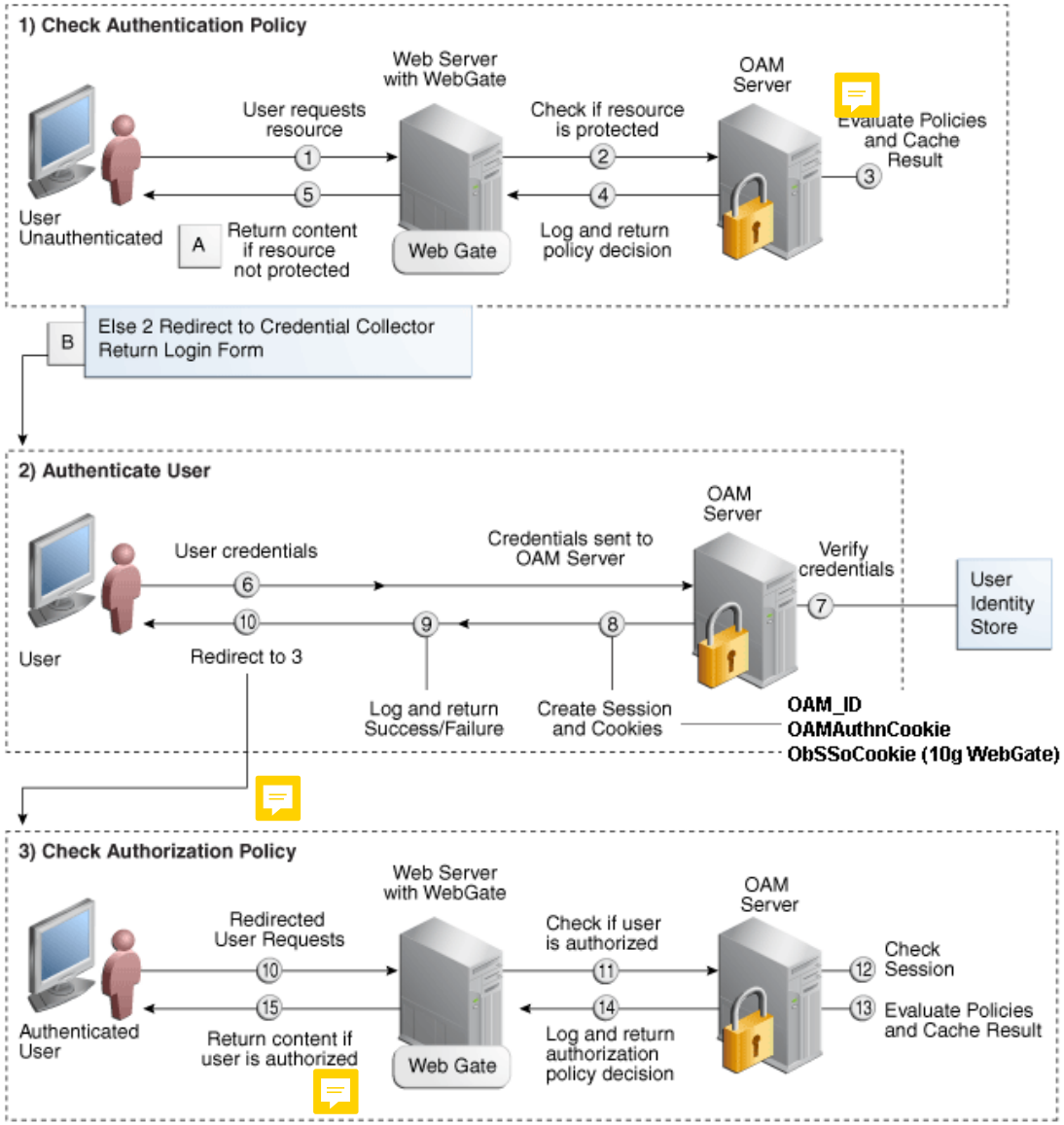
# IDM architektúra



# Az OIM Telepítése

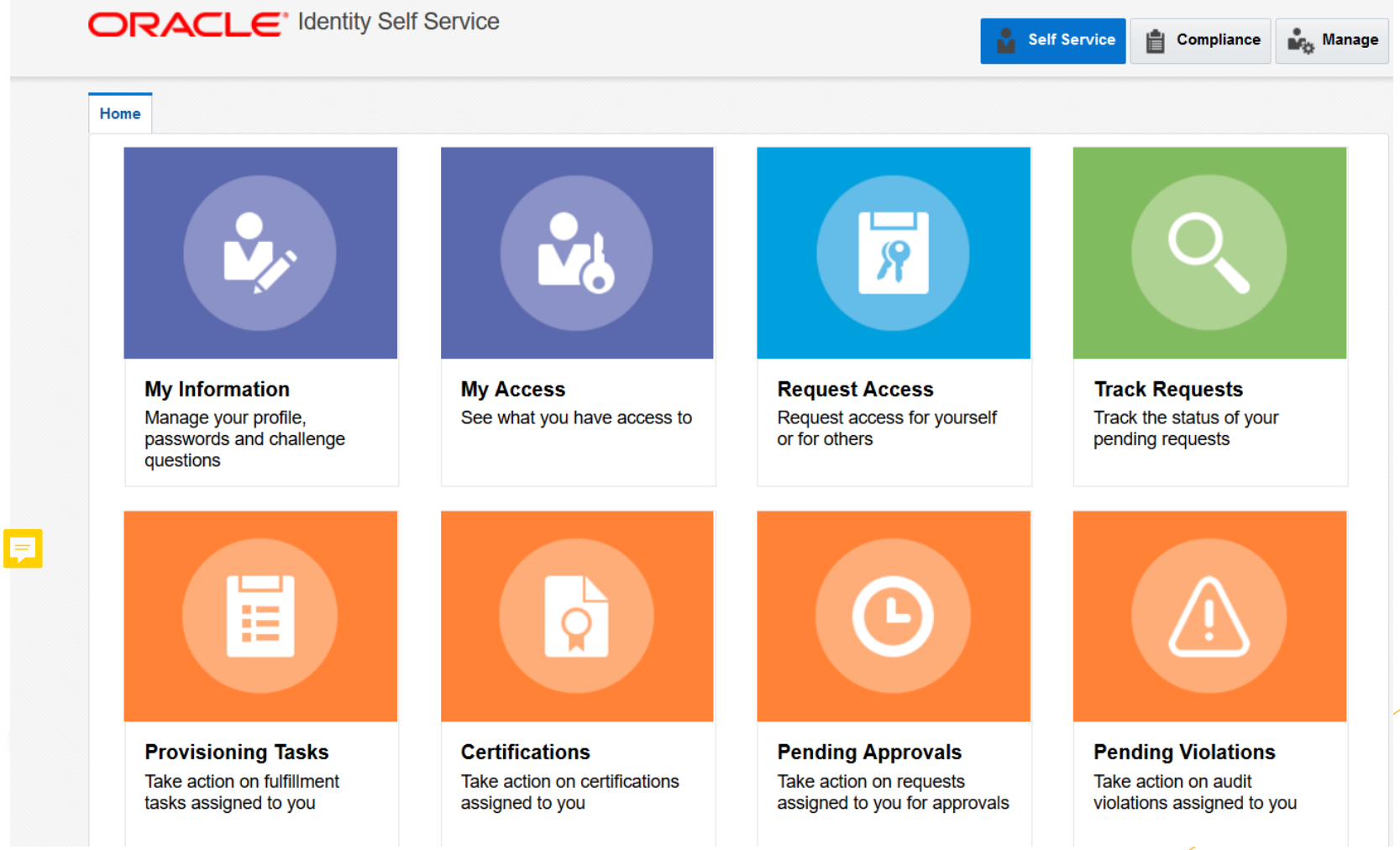
- Több lehetőség van:
  1. Kézzel telepítjük és konfiguráljuk az egyes komponenseket. A dokumentáció részletes, de időigényes munka, tele hibalehetőséggel
  2. 12c-től kezdve: Quick Installer
- Frissítések: az OPatch szoftverrel telepíthetjük a javításokat, minden komponenshez külön-külön
- Javítócsomagok (Bundle Patches) három havonta érkeznek
- Egy-egy hiba javítását one-off patch-nek hívjuk

# A bejelentkezés folyamata



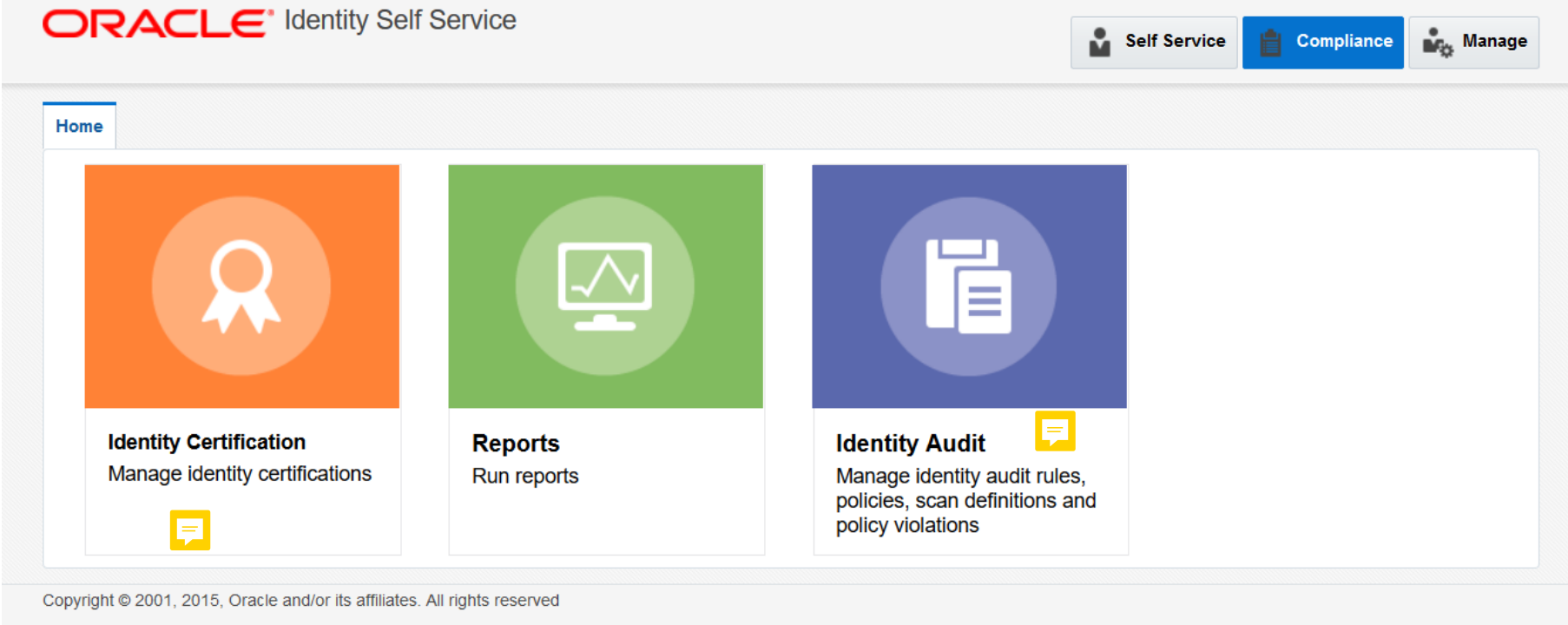
# Az OIM felülete I

- Az OIM Identity front-end alkalmazásban, a „Self Service” oldalon a felhasználó a hozzá kötődő adatokat tekintheti meg, illetve kezelheti őket



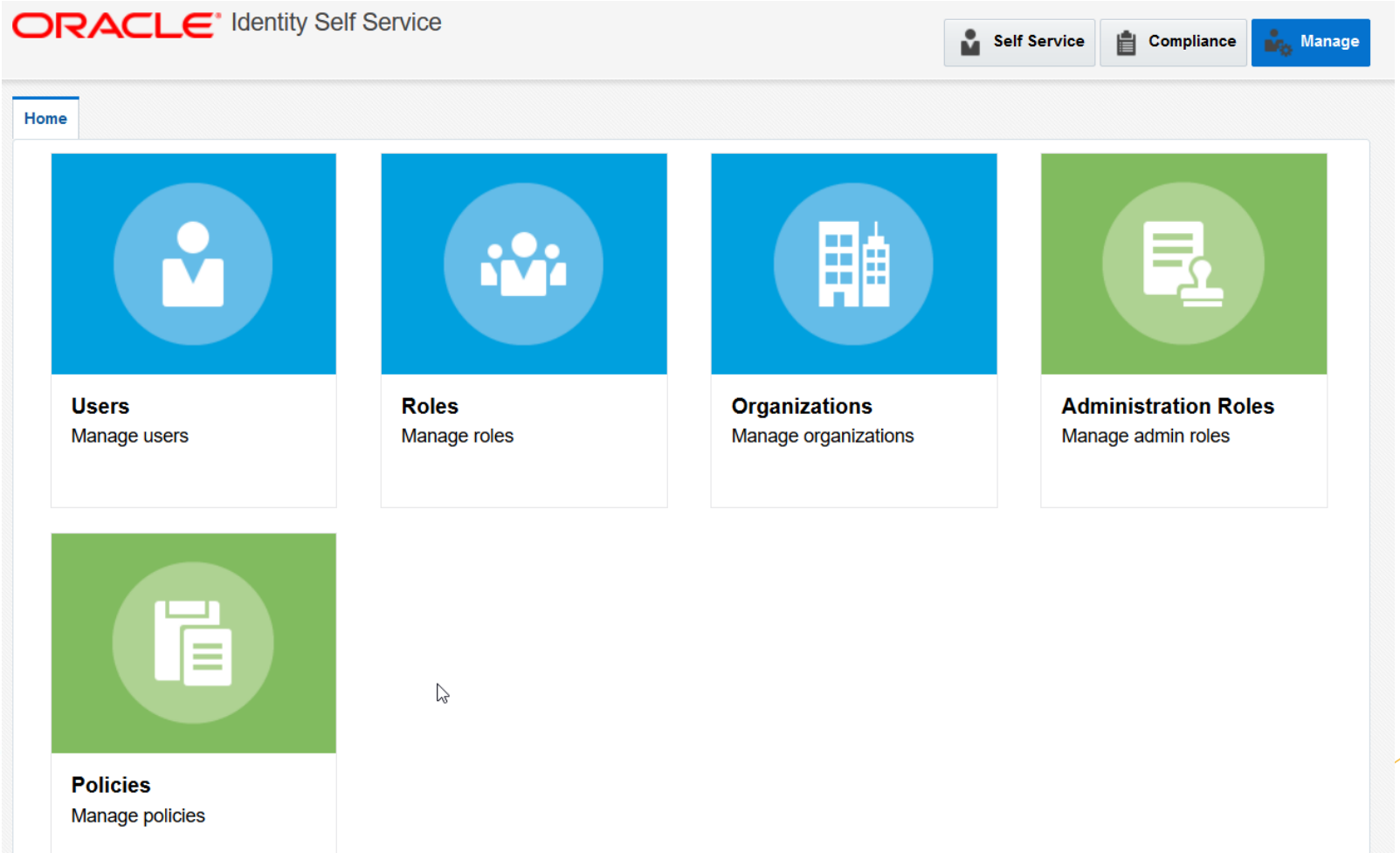
# Az OIM felülete II

- A „Compliance” oldal az üzleti adminisztrátorok számára biztosítja a minősítések (Certification) ellenőrzését

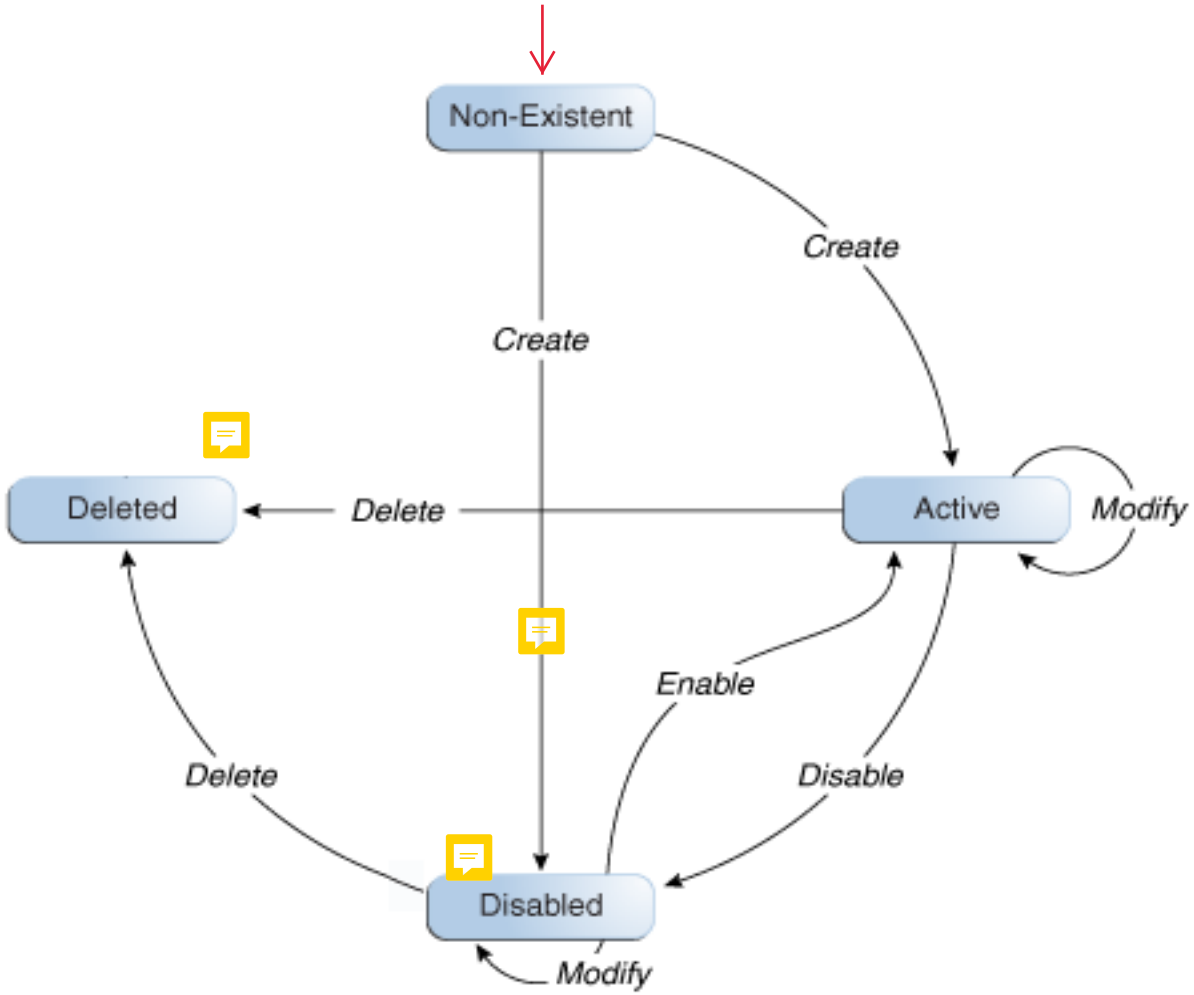


# Az OIM felülete III

- A „Manage” oldal az üzleti adminisztrátorok számára biztosítja a felhasználók, jogosultságok és szervezeti „házirend” kezelését



# A felhasználók életciklusa




- Minden felhasználónak egyedi azonosítója van (UserID)
- A felhasználók lehetnek még zárolt vagy nem zárolt állapotban is (Locked/Unlocked)
- Még törölve sem tűnnek el a rendszerből
- Minden felhasználó egy szervezet (Organization) alá van besorolva

# Jogosultságok

- Három féle jogosultságot különböztetünk meg az OIM-ben, a vállalat dönti el, hogy ezek közül melyiket használja
  - **Roles**: Szerepkörök, amelyek hierarchiába is rendezhetők. Ha van bekötve Identity Store (OUD), akkor ott is létrejönnek, mint egy LDAP csoport, a rendszer pedig a felhasználókat a csoportokhoz rendeli. Az LDAP-on keresztül ezeket a szerepköröket más Oracle komponensek is ellenőrizhetik.
  - **Entitlements:** jogosultságok, amelyek szinkronizálásra kerülnek más kapcsolódó rendszerekhez.
  - **Admin Roles**: Az Admin szerepkörökön keresztül adhatunk bizonyos felhasználóknak képességet (capability) az OIM funkcióinak elérésére. Ilyen lehet például a felhasználók létrehozása, törlése. Az OIM elrejti a funkciógombokat, amiket a felhasználó nem érhet el
- Felhasználói fiókok (Accounts): a kapcsolati pont a felhasználó más rendszerekben létező identitásai között. Rendszerenként lehet egy vagy több fiók is felhasználónként

# Jogosultság igények

- Minden felhasználó kérhet saját magának jogosultságokat, vagy akár mások számára is
- Az igényelni kívánt jogokat a katalógusból (Catalog) lehet kiválogatni
- A webáruházak „bevásárlókosár” elvén működik
- Ha az igény (Request) átesik az OIM oldali ellenőrzésen, és elküldésre kerül, akkor a SOA elindít egy folyamatot (Workflow) és ennek kapcsán meghatározza azt a felhasználócsoportot, akik a kérést elfogadhatják
- Ez több szinten is történhet (például „négy szem elv” esetén)

# Adatok szinkronizálása más rendszerekkel

- **Provisioning**: Az OIM által módosított felhasználói és szervezeti adatok küldése a kapcsolódó rendszereknek
- Tipikusan az OIM kezdeményezi a küldést (Push modell) egy módosítás hatására
- **Trusted Reconciliation**: Új felhasználók betöltése (létrehozása) az OIM-ben a kapcsolódó rendszerek adatai alapján
- **Target Reconciliation**: A felhasználók kapcsolódó rendszerekhez tartozó fiókjainak adatait tölti be a kapcsolódó rendszerben történt változások alapján
- A reconciliation lehet push vagy pull modell alapú. Pull modell esetén az OIM ütemezetten kéri le az adatokat

# Identity Connector Framework

- Az ICF egy egységes, köztes réteget képez az OIM és a célrendszerek között az adatcsere végrehajtásához (provisioning, reconciliation)
- A létrehozott Connector implementációk függetlenek az alkalmazástól, izoláltan tesztelhetők
- A keretrendszer két rétegből áll:
  - API (Application Programming Interface): Ezt a réteget hívják az alkalmazások. Független a tényleges Connector működéstől, elfedi azt
  - SPI (Service Provider Interface): A fejlesztők az SPI réteg azon interfészeit implementálják, amelyek szükségesek a célrendszer támogatásához
- Az Oracle sokféle Connector-t biztosít (Active Directory, SAP, Flat File, Office 365 stb.)
- Ha egy rendszerhez nincs Connector, akkor a keretrendszer segítségével fejleszteni kell egyet

# Identity Certification

- A Certifcation az a folyamat, amelynek kapcsán ellenőrzik a felhasználóknak kiosztott jogosultságok és hozzáférhetőségek helyességét
- A folyamat kimenetele lehet a jóváhagyás, vagy elutasítás, ami a kérdéses jogok elvételével jár
- Az OIM rendszerben indítható egyes felhasználókra (User Certification), vagy egyes jogosultságok birtoklóira (Entitlement- vagy Role-Certification).
- Minősítési feladatok létrejöhetnek ütemezetten, vagy Event Trigger hatására (például kiválthatja felhasználók módosítása)
- A minősítést végző felhasználók számára külön felületet biztosít

# Az OIM Ütemező

- Az OIM Scheduler felel az előre meghatározott feladatok (Jobs) ütemezett teljesítéséért
- Ezek típusai:
  - OIM specifikus feladatok, amelyek a rendszer karbantartásáért vagy funkcionalitásáért felelnek (például régi audit bejegyzések törlése vagy lejárt jelszavak ellenőrzése)
  - LDAP Reconciliation feladatok, amelyek az OUD-ban észlelt módosításokat betöltik az OIM rendszerbe (ilyen módosítás lehet például, amikor az Access Manager zárol egy felhasználót az elütött jelszavak miatt).
  - Saját fejlesztésű ütemezett feladatok, például a kapcsolódó rendszerek adatainak betöltéséhez

# Auditing

- Minden vállalat számára kritikus az IT rendszeren belül történt események rögzítése, naplózása
- Az OIM beépítetten képes felhasználók, jogosultságok és a katalógus auditálásra. Ezek szintjei (az audit részletessége) is állítható
- Ezen kívül az Oracle Fusion Middleware legtöbb komponenséhez beállítható auditálás:
  - Ezt a Weblogic kezeli és az Enterprise Manager-ben konfigurálható
  - Külön adatbázis sémát kell létrehozni az OIM és OAM audit számára, ide töltődnek be az adatok
  - Az Access Manager Auditing beállítása esetén például bejegyzések készülnek minden URL elérésekor, bejelentkezéskor vagy kilépéskor, felhasználónévvel és IP címmel ellátva

# Monitoring

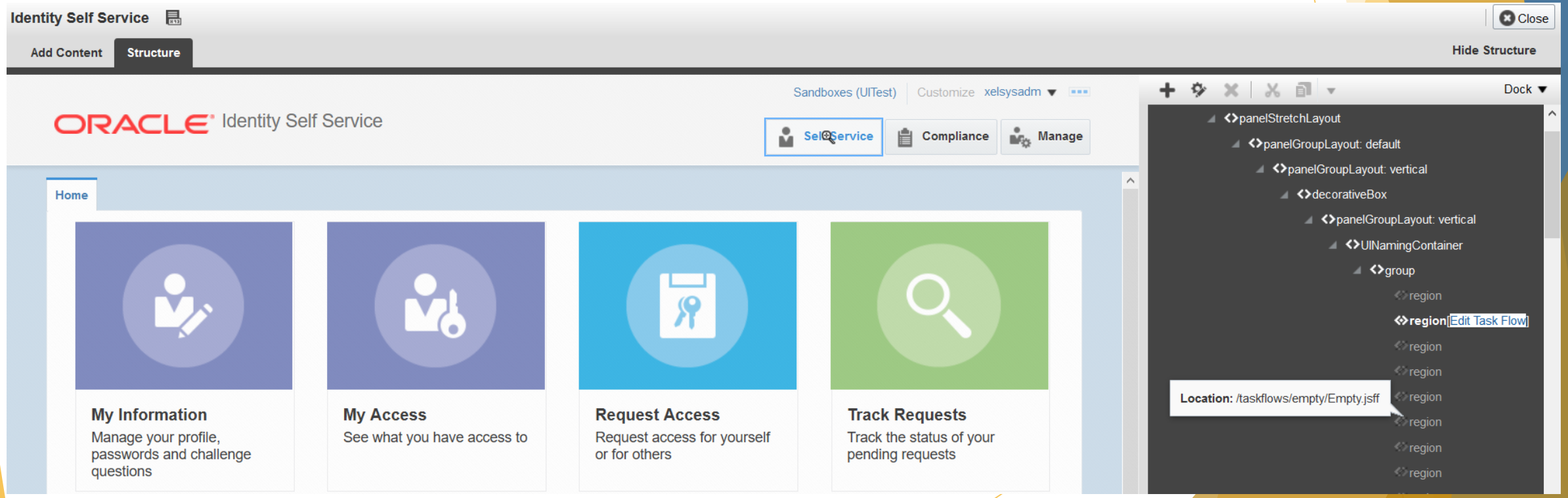
- Mind a rendszer komponenseinek működését, mind az alkalmazás funkciókat érdemes felügyelni
- Ellenőrizni kell az operációs rendszer erőforrásait
- A futó Weblogic példányok figyelésében segít az Enterprise Manager vagy írhatunk WLST szkripteket is
- Visszatérő hibák kiszűrésére érdemes lehet beütemezni szkripteket, amik az alkalmazás logokat olvassák
- Az OIM funckiók (hibás provisioning feladatok, beragadt jogosultság igények) felügyeletére érdemes BIP riportokat generálni, amik az adatbázis táblákból kérdeznek le (de az adminisztrátorok az OIM felületén is át tudják ezeket nézni)

# A rendszer konfigurálása

- Telepítés után alapesetben a xelsysadmin felhasználó használható az OIM teljeskörű adminisztrációjához
- A „System Administrator” Admin Role neki van kiosztva
- Beléphetünk vele a /sysadmin felületre, innen érhetjük el az ütemezőt, a rendszerbeállításokat, telepíthetünk Connector-okat, módosíthatjuk a meta-adatokat (export-import) stb.
- Több száz attribútumot állíthatunk, amelyek befolyásoljak a rendszer működését
- Az Enterprise Manager segítségével még részletesebb szoftverbeállításokat alkalmazhatunk
- Minden beállítás az adatbázisban van eltárolva

# Fejlesztés I - UI

- A felhasználó felület könnyen változtatható a böngészőben is
- A módosítások a Sandbox funkcionalitás segítségével menthetők el
- Saját kód hívásához: ADF alapú Java Managed Bean-ek létrehozása és ezek hívása az Expression Language segítségével. A fejlesztés JDeveloper környezetben történik



# Fejlesztés II - Sandbox

- Ahhoz, hogy a megjelenítési felülethez kötődő változtatásokat eszközöljünk, készítenünk kell egy Sandbox példányt
- Egyszerre mindig csak egy példány lehet aktív, ez rögzíti az elvégzett változtatásokat (módosításokat a metaadatokban)
- A Sandbox lényegében egy zip fájl könyvtárakkal és xml fájlokkal
- Az elkészített Sandbox exportálható (és importálható egy másik rendszerbe) vagy publikálható. Publikálás esetén minden módosítás rögzül a rendszerben
- Érdemes teljes metadata mentést készíteni publikálás elött
- Több Sandbox példány között gyakran van függőség: figyelni kell a sorrendre, ha például teszt rendszerből éles rendszerbe importáljuk őket. Könnyen felülírhatunk változásokat, ha nem a megfelelő sorrendben kerülnek ki a SandBox-ok

# Fejlesztés III - Backend

- Az OIM backend fejlesztése különféle feladatokból áll
  - SOA Workflow fejlesztés: a JDeveloper biztosítja a környezet a BPEL alapú logika előállításához
  - Connector implementációk létrehozása
  - Provisioning vagy Reconciliation feladatok elvégzéséért felelős objektumok létrehozása
  - Ezeken felül az OIM szinte minden működési területét át lehet alakítani vagy bővíteni lehet saját kód bekötésével

# Fejlesztés IV – Rest API

- Gyakori igény, hogy a távoli alkalmazások is képesek legyenek műveleteket végrehajtani az OIM rendszerben. Két lehetőség:
  - OIM Java API hívások: A távoli alkalmazás t3 protokollon keresztül, kliensként hívja az OIM interfészeket
  - OIM REST (Representation State Transfer) API hívások: SCIM (System for Cross-Domain Identity Management) protokoll alapú Webservice hívások HTTP-n keresztül
- Ha a kérések csak Load Balancer-en keresztül érkezhetnek, a direkt API hívások csak körülményesen oldhatók meg
- Az OIM REST szolgáltatásokon keresztül az OIM funkciók nagy részét elérjük: felhasználók, jogosultságok, szervezetek kezelése vagy akár a rendszer módosítása

# OIM Tuning

- Az optimalizálás főbb feladatai:
  - Adatbázis hangolása a megfelelő paraméterek beállításával
  - Az OIM szoftver beállításainak finomhangolása (pl.: object caching)
  - Weblogic Tuning: a JVM indító paramétereinek módosítása, Work Managerek és adatbázis kapcsolatok (Datasources) hangolása
- Iránymutatóként az Oracle készített tuning dokumentációkat

# Köszönöm a figyelmet!

- Elérhetőségem: peter.joo@webvalto.hu

# Felhő alapú szolgáltatások - SaaS - és ami mögötte van

Előadó: Szabó Gábor
Director, Hospitality Business Unit

ORACLE

Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

# Felhő alalpú szolgáltatások története

- 1950 es évek – óriási és drága számítógépek -> időmegosztás, eszközmegosztás
- 1960 as évek – ARPANET -> INTERNET
- 1970 es évek – IBM virtuális gépek megjelenése
- 1990 es évek – Internet forradalom
- 2000 es évek – Multinacioális cégek elkezdtek Felhő alapú szolgáltatást ajánlani ügyfeleiknek

ORACLE

Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

# Felhő definíció (NIST - National Institue of Standards and Technology)

A felhő alapú szolgáltatás az egy olyan informatikai szolgáltatási modell, amely lehetővé teszi azt hogy bárhol, kényelmesen, igény szerint nyújtson hálózaton keresztül hozzáférést megosztott számítástechnikai erőforrásokhoz, amelyeket gyorsan rendelkezésre lehet bocsátani úgy, hogy nagyon kevés adminisztratív tevékenysegre és minimális szolgáltatói interakcióra van esetleg szükség.

ORACLE

Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

# Felhő definíció (NIST - National Institue of Standards and Technology)

- A felhő alapú szolgáltatásoknak 5 alapvető jellemzője kell legyen:
  - Igény szerinti önkiszolgálás
  - Széleskörű hozzáférhetőség
  - Helyszíntől független erőforrás készlet
  - Teljes rugalmasság
  - Mérhető szolgáltatási mennyiség

ORACLE

Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

# Felhő típusok

## Nyilvános felhő

- Klasszikus felhő: szolgáltatások és infrastruktúra megosztása több ügyféllel
- Önkiszolgáló, interneten keresztül elérhető
- Biztonsági, irányítási és felügyeleti kérdések

## Magán felhő

- 1 ügyfélnak biztosít infrastruktúrát és szolgáltatást
- Több felügyelet és irányítási lehetőség az ügyfélnek
- Drágább mint a nyilvános felhő

## Közösségi felhő

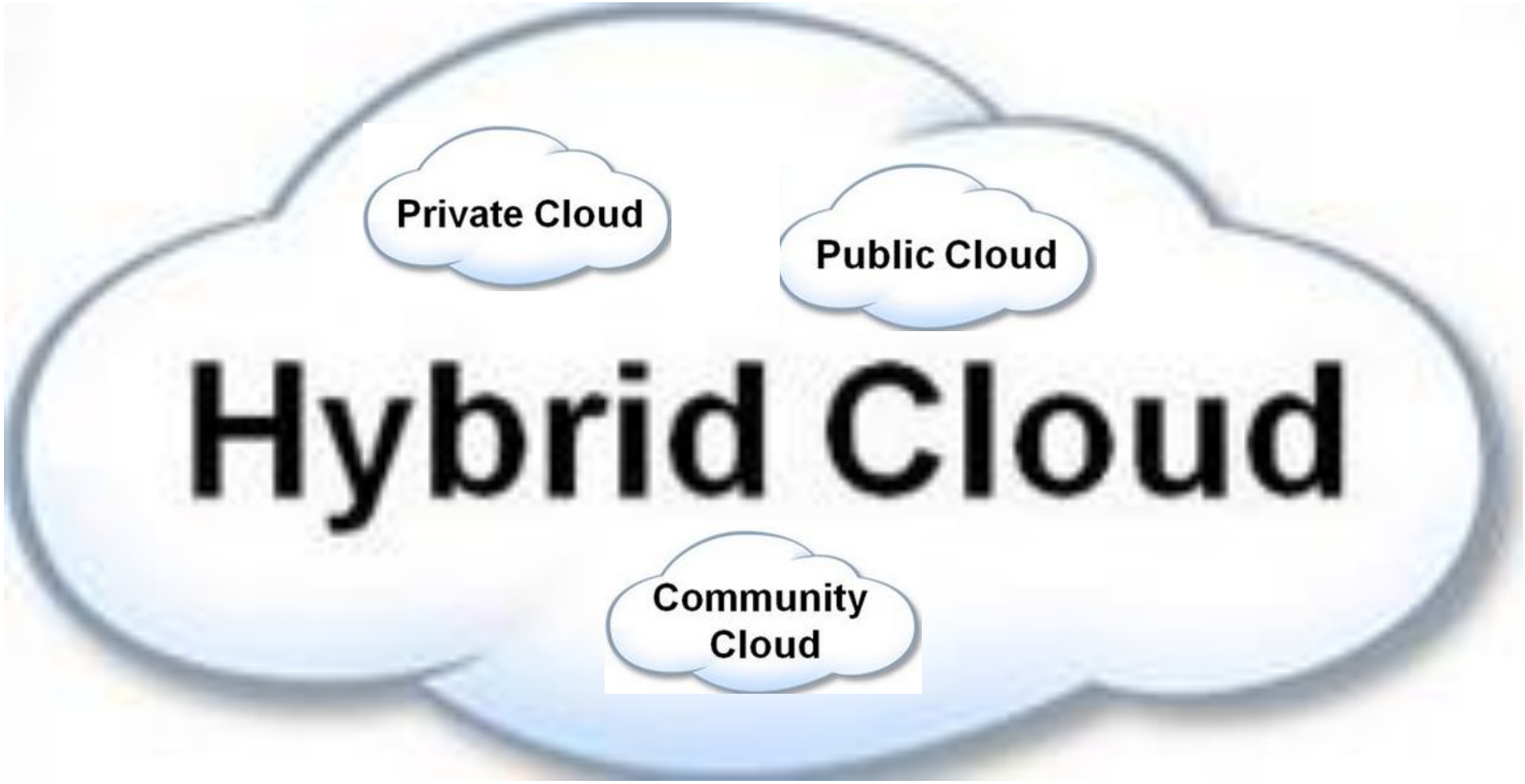
- Azonos tevékenységet folytató szervezetek között megosztott
- Egy közös cél
- Költségmegosztás

## Hibrid felhő

- Felhőtípusok kombinációja
- Bonyolultabb – biztonsági kérdések mikor nyilvános és magán felhőt kombinálunk akár több felhőszolgáltató is szerepet játszhat
- Skálázható, költséghatékony, átjárható

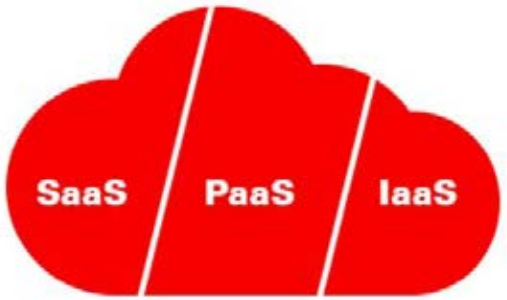
Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

# Felhő típusok




Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

# Felhő alapú szolgáltatások




- **Infrastrukturális számítási felhő (infrastructure as a service):** az ügyfelek számára hálózatot, tárhelyet, és minden olyan erőforrást biztosít a szolgáltató ami ahhoz kell, hogy a saját Operációs rendszerét, szoftvereit fejleszteni és futtatni tudja. Az ügyfél nem ismeri a felhő infrastruktúrát, viszont ő felel és felügyeli az operációs rendszert, a tárhely kapacitást, az alkalmazást ls néhény hálózati komponenst is., mint például a tűzfalat.
- **Platform alapú számítási felhő (platform as a pervice):** az ügyfél számára a szolgáltató biztosítja a lehetőséget, hogy a saját alkalmazását fejlesztheti, futtathatja a szolgáltató által a rendelkezésére bocsájtott eszközökkel. Ezáltal a felhasználó nem látja és nem felügyeli az infrastruktúrát (tárhely, Operációs rendszer, hálózat) csak és kizárólag a saját alkalmazását és maximum nehány háttér beállítást
- **Szoftver alapú számítási felhő (software as a service):** a szolgáltató lehetőséget nyújt az ügyfeleinek, hogy használja a szolgáltató alkalmazásait, amik elérhetőek különböző eszközökről, különböző módokon. Az ügyfél nem látja a felhő infrastruktúrát, még az alkalmzás architektúrát sem, csak talán néhány alkalmazáson belüli beállítást.
- Egyéb:

Database as a Service (DBaaS) Storage as a Service (STaaS) Hardware as a Service (HWaaS)

ORACLE

Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

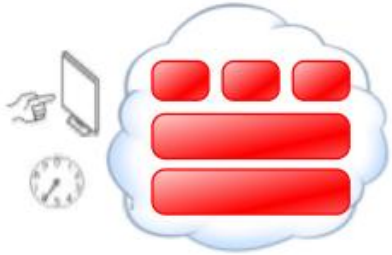
# Felhő evolúció

### Tradícionális

- Fizikai szerverek
- Dedikált rendszerek
- Heterogén rendszerek

### Konszolidált

- Virtuális szerverek
- Megosztott rendszerek
- Standardizált rendszerek



### Magán felhő

- Igény szerinti önkiszolgálás
- Automatikus erőforrás hozzárendelés
- Mérhetőség

### Nyilvános felhő

- Speicalizálhatóság alkalmazás szinten
- Megosztott rendszerek
- Standardizált platform

### Hibrid felhő

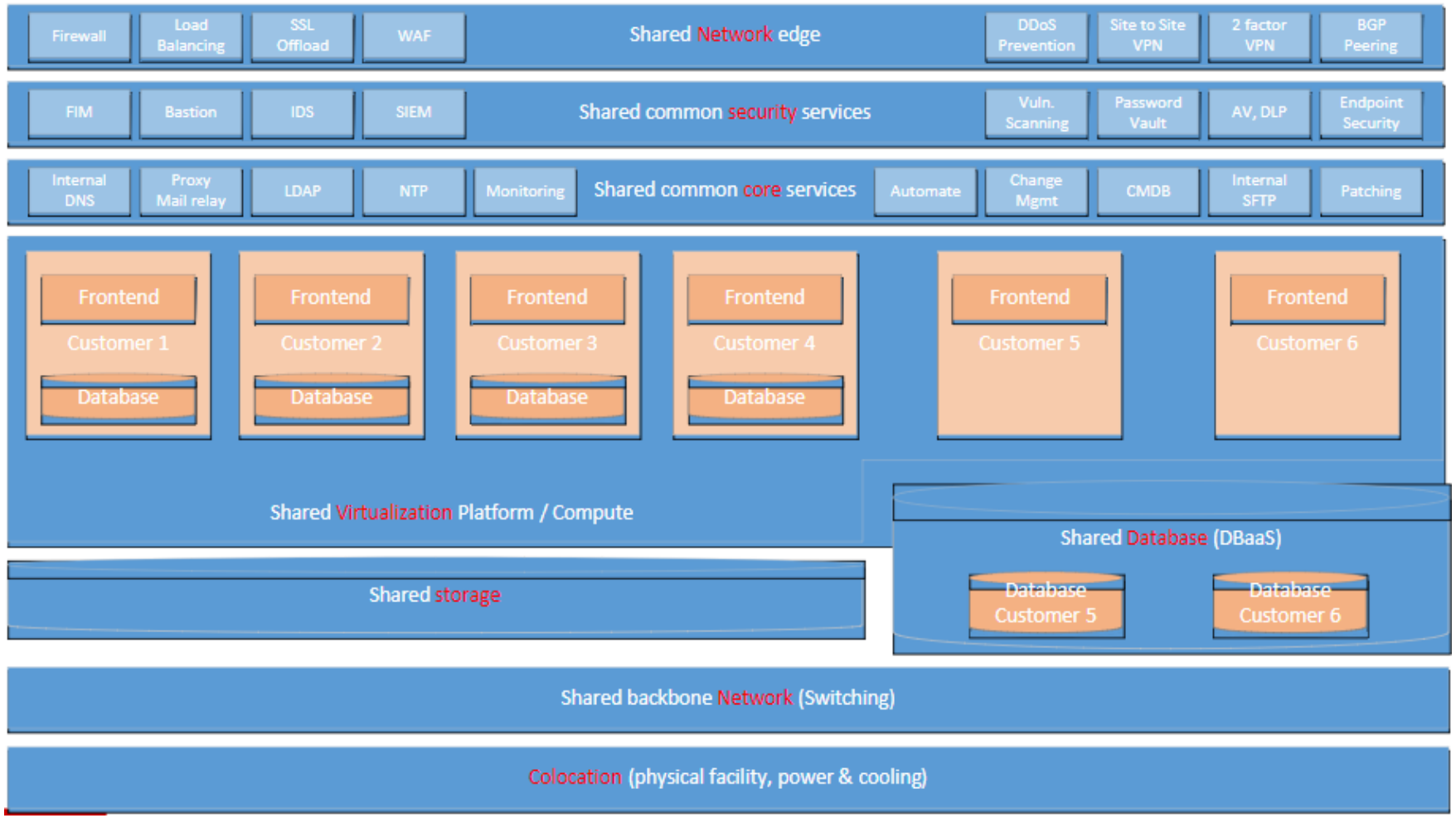
- Nyilvános és magán felhő együtt, azok minden előnyét kihasználva

ORACLE

Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

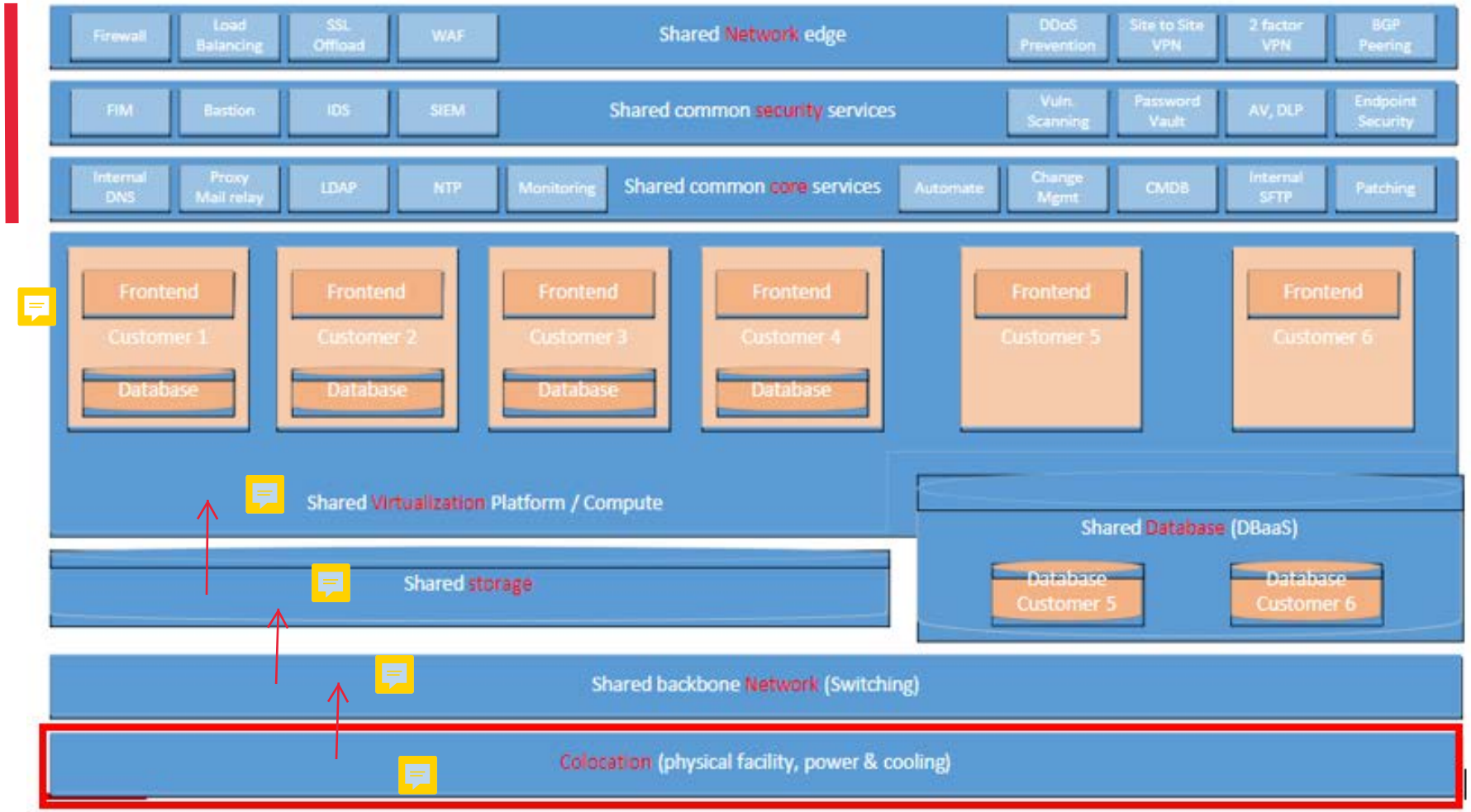
Forrás: Laczi Károly Oracle Junior program SaaS előadás 2016

# Felhő és ami mögötte van - felépítés




Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

# Felhő és ami mögötte van




Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

Forrás: https://www.youtube.com/watch?v=F1dAEuSlH0A

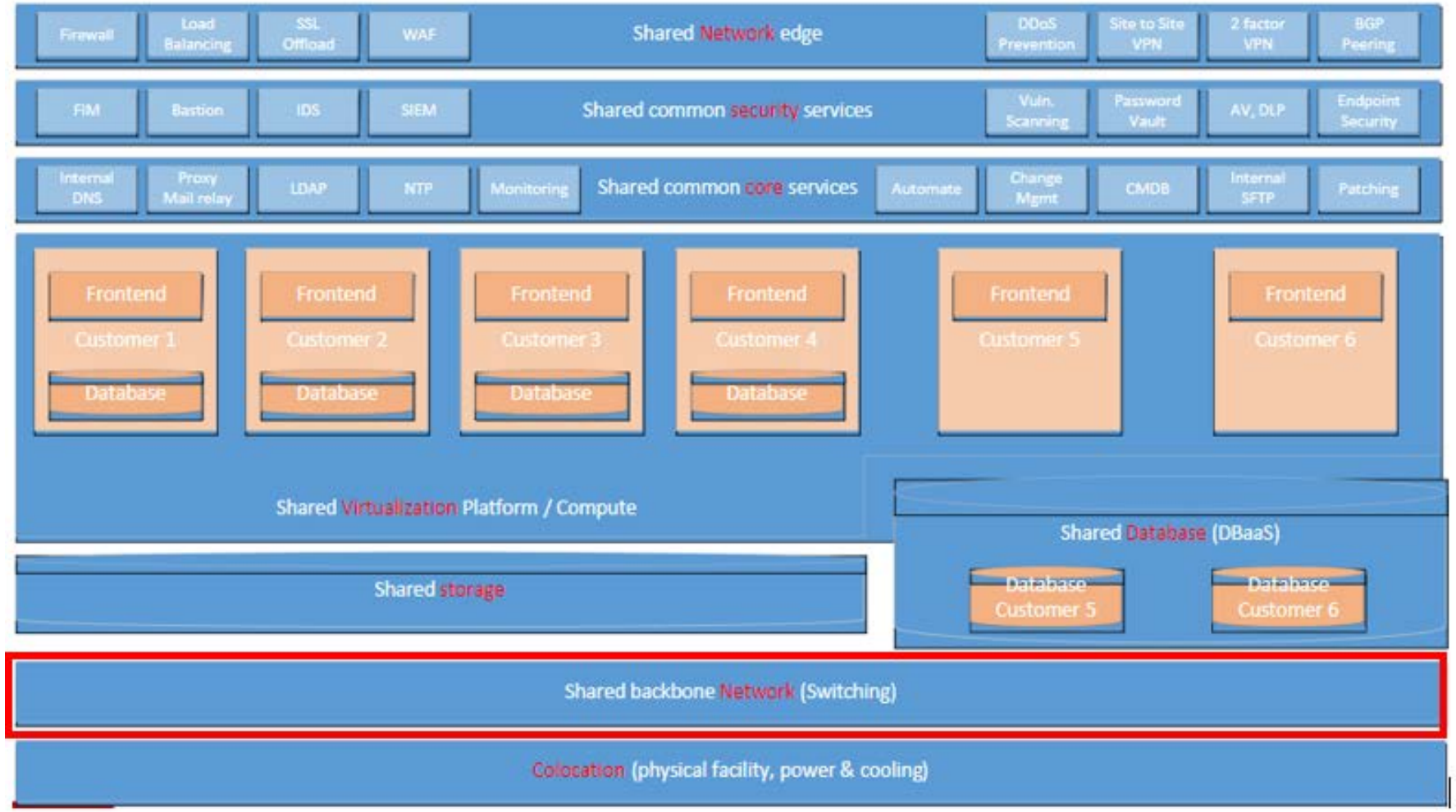
# Felhő és ami mögötte van – Fizikai infrastruktúra

- Fizikai hozzáférés és biztonság
- Tűzvédelmi rendszer
- Hőmérséklet és páratartalom monitoring:
  - Állandó hőmérséklet: 22 C° +/- 2 C°
  - Állandó páratartalom: 50% +/- 15%

- Redundáns áramszolgáltatás mellett:
  - UPS - szünetmentes tápegységek
  - Legalább 10 percig üzemképes akkumlátorok
  - Generátorok
  - Legalább 72 órára elég üzemanyag a generátorokhoz
  - Áramellátás monitoring

Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

# Felhő és ami mögötte van



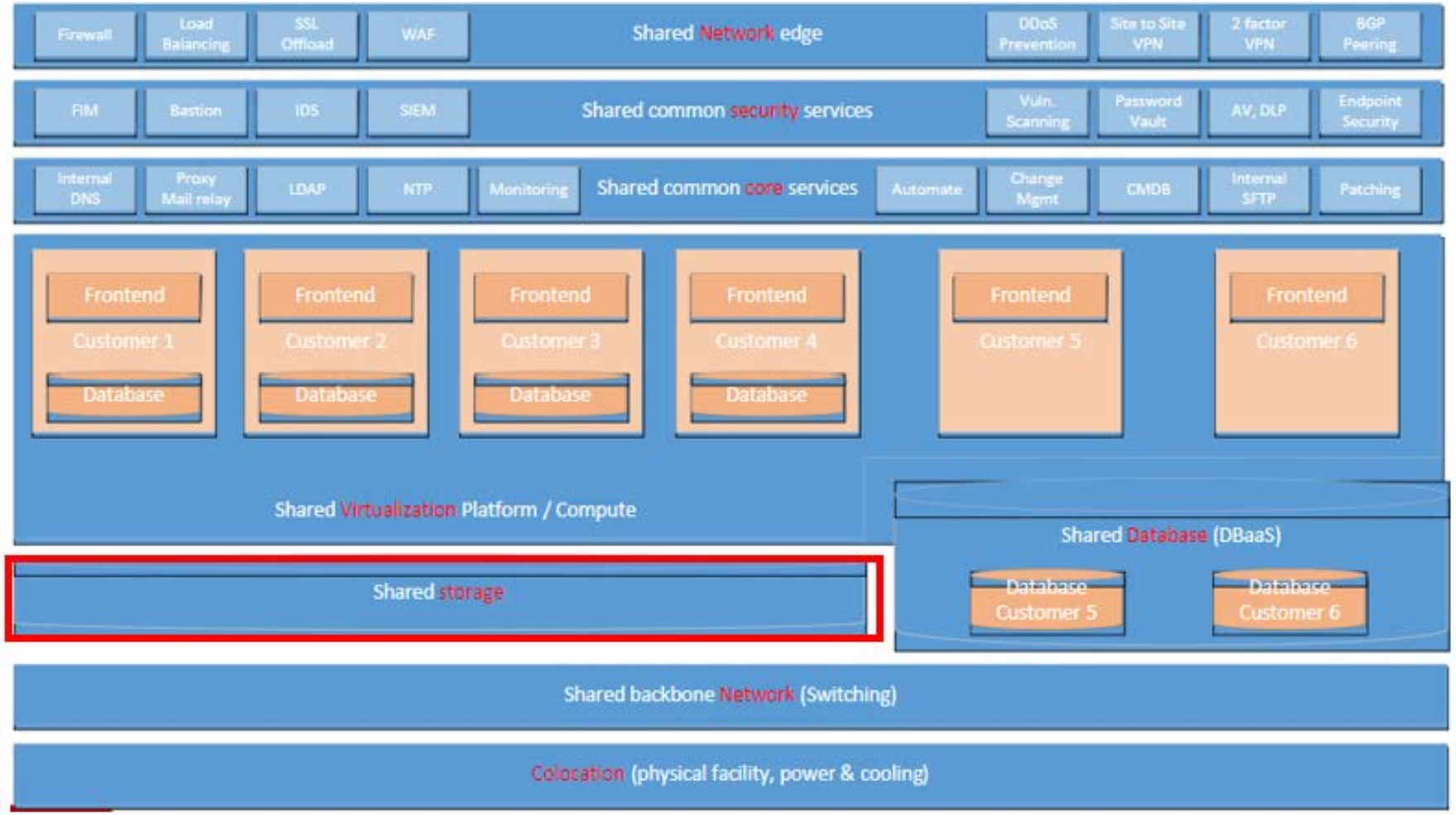

Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

# Felhő és ami mögötte van – Hálózat

- Redundáns:
  - Routerek
  - Switchek
  - Tűzfalak
  - Load balancerek
  - Behatolás érzékelő hálózati eszközök

Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

# Felhő és ami mögötte van



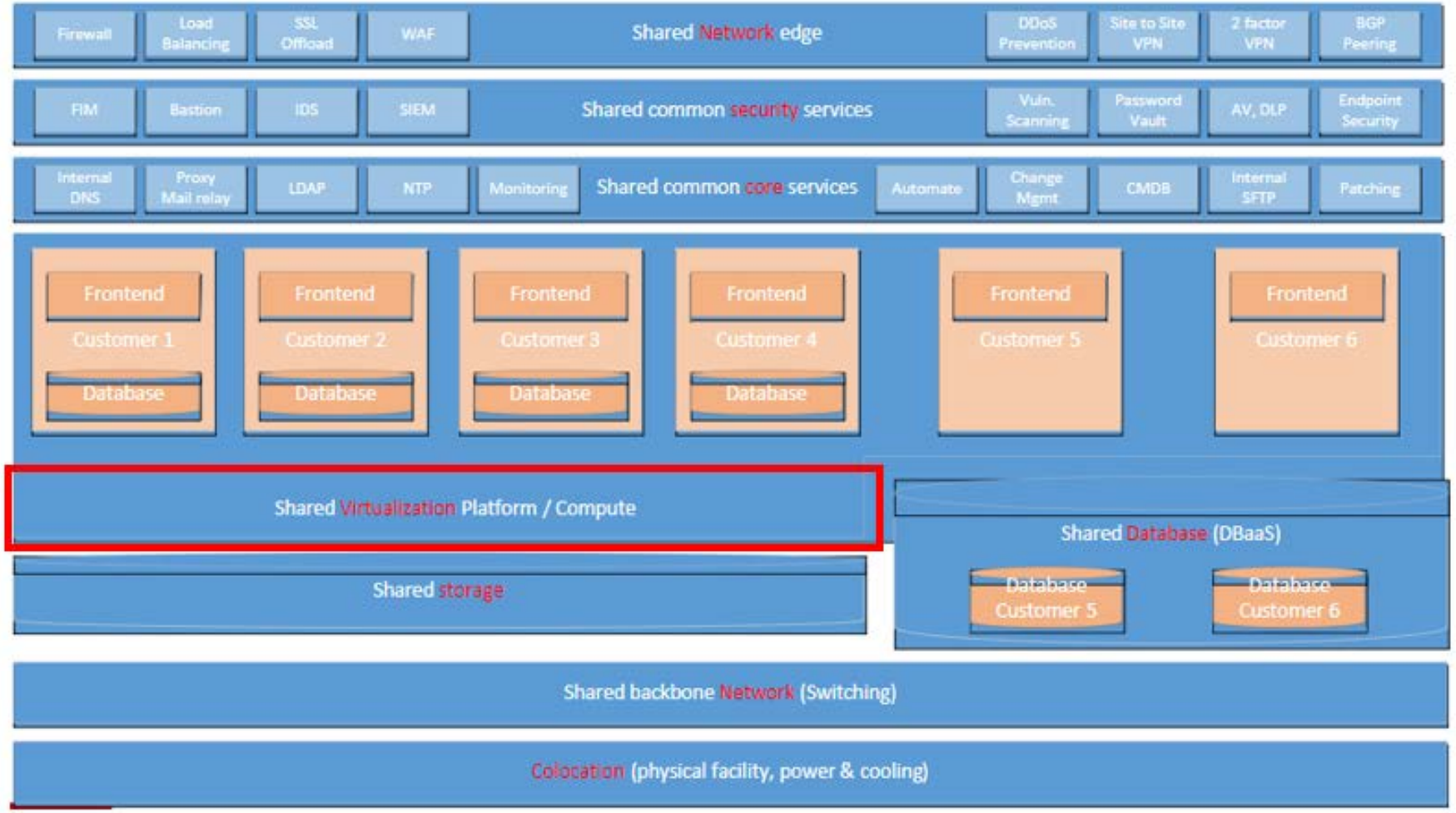

Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

# Felhő és ami mögötte van – Tárhely

- High Capacity
  - Nagyon sok tárhely kapacitás
  - Kevesebb redundancia
  - Lassabb elérés
  - Álatlában test rendszerek, nem kritikus rendszerek, mentések tárolására
- High availability
  - Rendkívül gyors elérés
  - Magas szintű redundancia – zero data loss
  - Kevesebb tárhely kapacitás
  - Éles rendszerel adataira

Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

# Felhő és ami mögötte van




Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

# Felhő és ami mögötte van – Fizikai szerverek

- Rendkívül nagy teljesítményű számítógépek
- Redundáns, legalább 2 tápegység
- Redundáns, legalább 2 hálózati kártya
- Ezek az alapjai a virtualizációnak
- Ezek adják az ugynevezett compute kapacitást:
  - Memória – akár 8TB
  - CPU – akár 512 CPU core

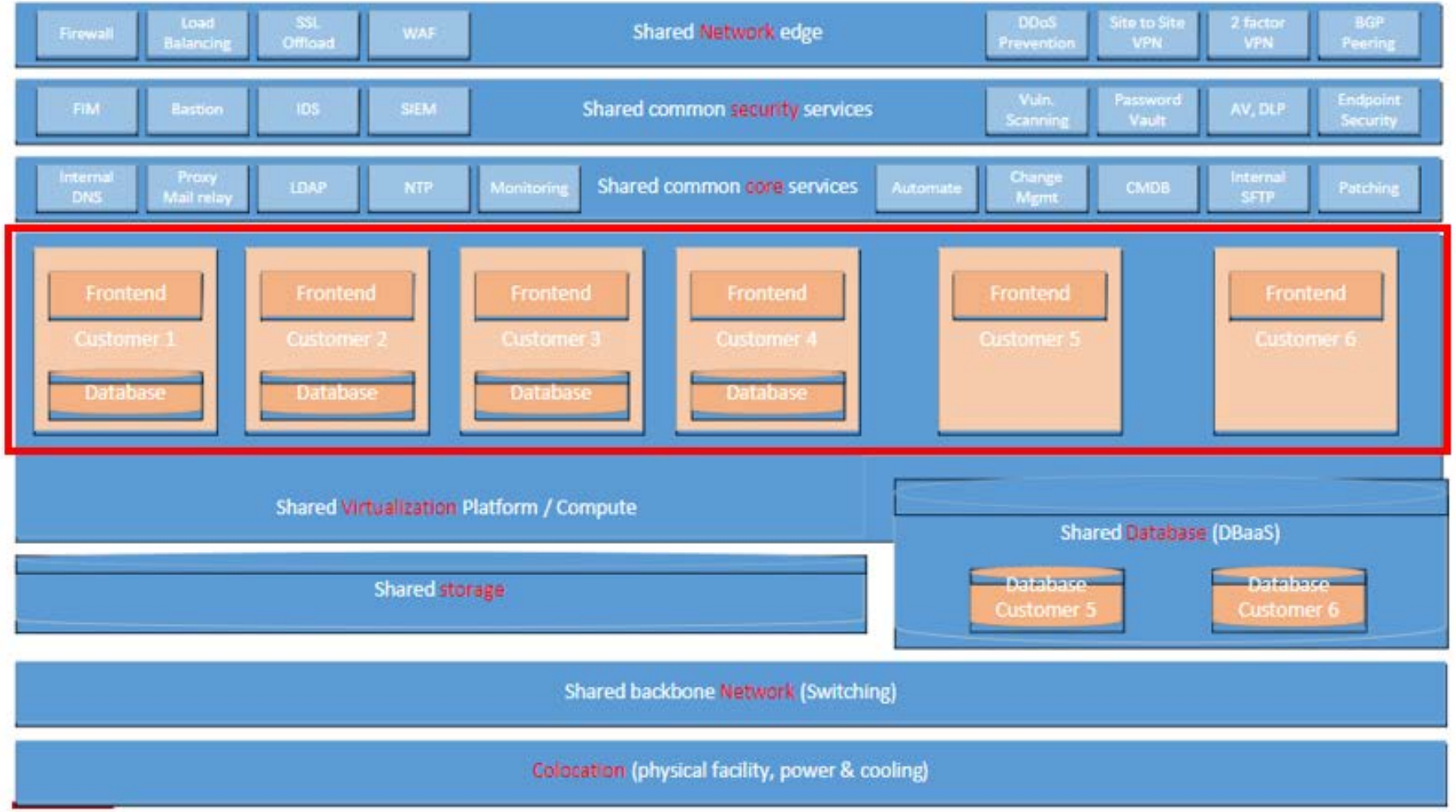
Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

# Felhő és ami mögötte van – Speciális hardware – EXADATA

- Arra lett kitalálva, hogy minél gyorsabb fusson rajta az Oracle adatbázis.
- Gyakorlatilag hardware szinten értelmezi az SQL utasításokat
- All in ONE adatbázis monstrum:
  - 4 TB DRAM
  - 22TB flash – akár 220TB tömörített adat – Összes aktív adat memóriában
  - 8 magos Intel Xeon E5-2690 processorok
  - Akér 1.5 millió disk olvasás 1 mp alatt
  - 500TB tárhely
  - Íráskor is a flash cache-t használja, így akár 1 millió írást is tud mpként
  - Oracle Enterprise linux
  - Oracle 12c adatbázis
  - 40X10Gb port

Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

# Felhő és ami mögötte van




Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

# Felhő és ami mögötte van – Vitualizáció

| Hagyományos virtualizáció | Cloud virtualizáció |
|---|---|
| Virtuális gépek létrehozása kézzel történik, nincs automatikus lehetőség | Virtuális eszközök, gépek listája, ami lehetővé teszi hogy gyorsan és kockázat mentesen hozzunk létre virtuális eszközöket |
| Nincs dinamikus erőforrás kezelés | Pool erőforrások automatikus és dinamikus kezelése |
| Fix kapacitású tárhely, fix hálózat | Dinamikus szerver pool, rugalmas network, megosztott, rugalmas tárhely kapacitás |
| Különböző csapatok bejövő jegyek és kérések alapján hozzál létre a gépeket: Rendszer adminok telepítik az Operációs rendszert, Storage mérnökök hozzák létre a kapcsolódó tárhelyeket, hálózati mérnökök konfigurálják a hálózatot, stb.. Ez szekvenciális és időigényes | Automatizációnak köszönhetően a virtuális gépek létrehozása gyors, és egyszerű, nem igényel emberi beavatkozást csak akkor, ha az erőforrások kimerülőben vannak, illetve ha az ügyfélnek speciális igényei vannak |

ORACLE

Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

# Cloud virtualizáció variánsok

Docker konténerek

Különböző Hypervisorok

Különböző OS

- Hardware virtualizáció

Mindent virtualizálunk az OS-sel együtt. A virtuális gépben futó operációs rendszer ideális esetben nem is tud arról, hogy nem fizikai hardveren fut, hanem egy virtuális gépben.

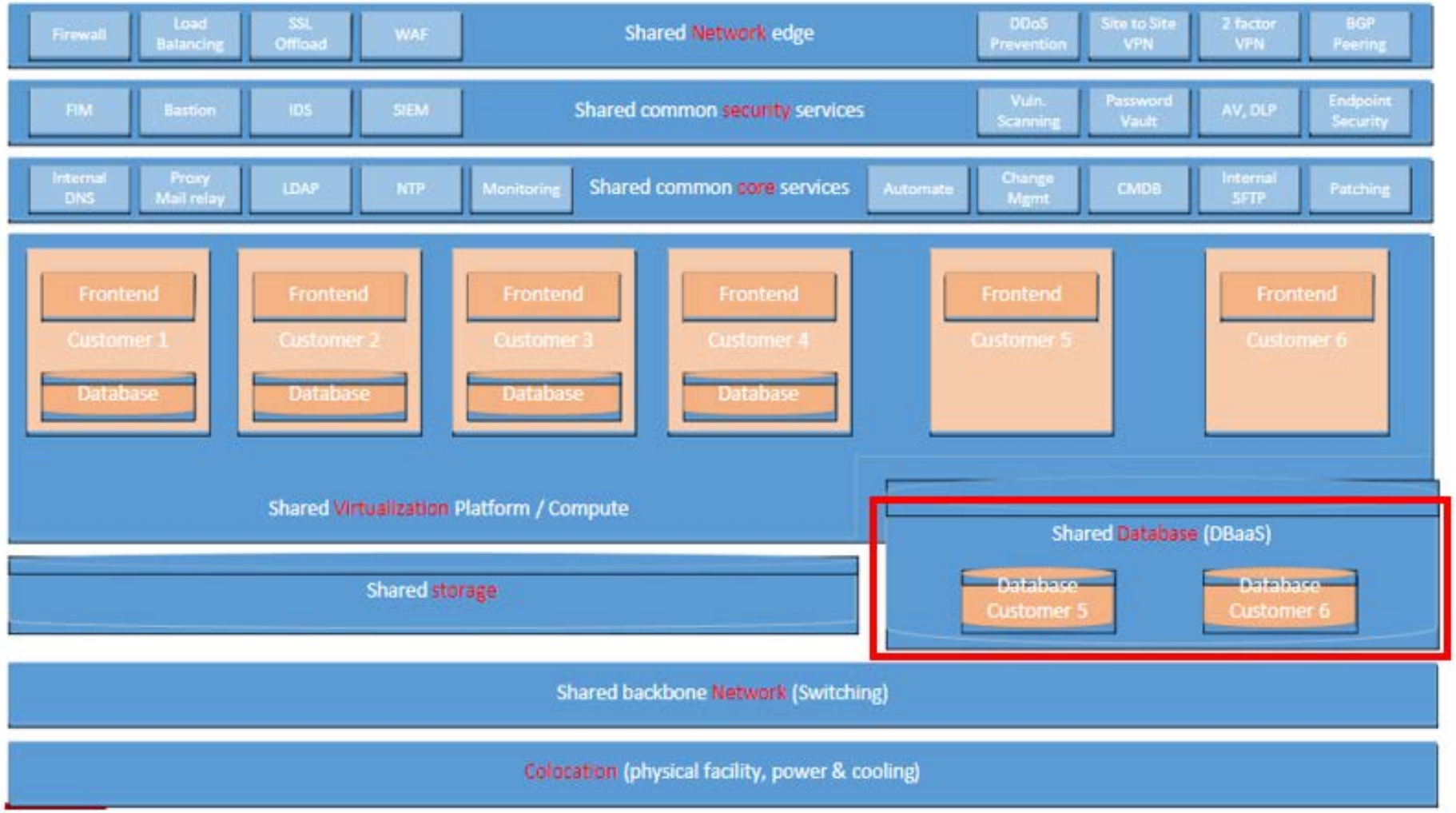
- Docker konténeres virtualizáció:

A konténerek szervezése szolgáltatás alapon történik, minden szolgáltatásnak konténert, saját környezetet definiálunk, amelyek között a célszerűségnek megfelelően linkeket képzünk. A konténerek többszörözése szükség esetén könnyen megtehető, így egy adott szolgáltatást pillanatok alatt több példányban futtathatunk.

ORACLE

Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

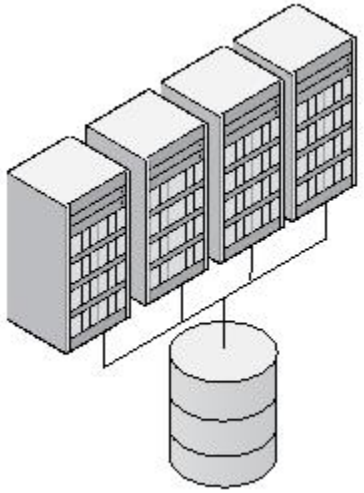
# Felhő és ami mögötte van




Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

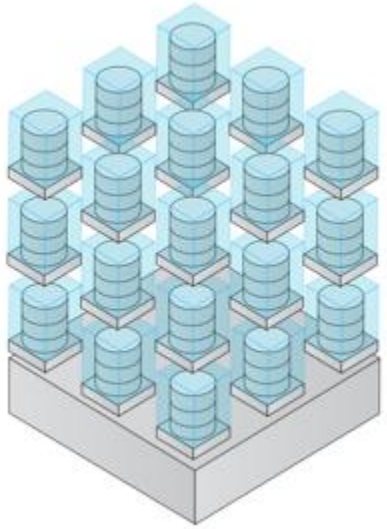
# Oracle Cloud – Database as a Service - Evolúció

Virtuális gépek



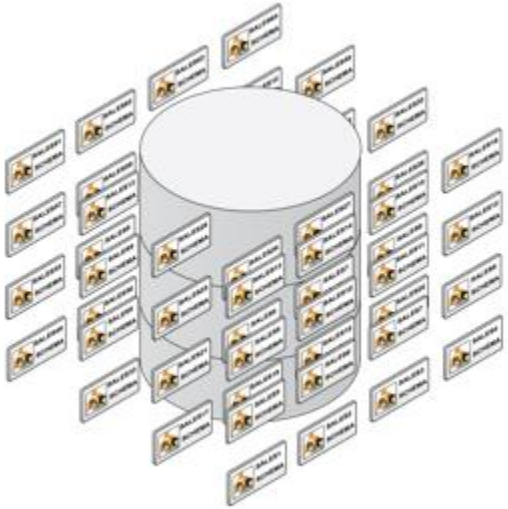
Megosztott hardware

Adatbázis példányok



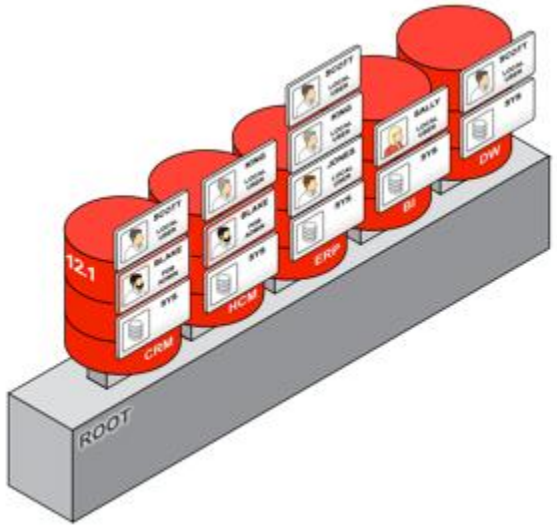
Megosztott hardware és OS

Séma konszolidáció



Megosztott hardware, OS és DB

Konténer adatbázisok

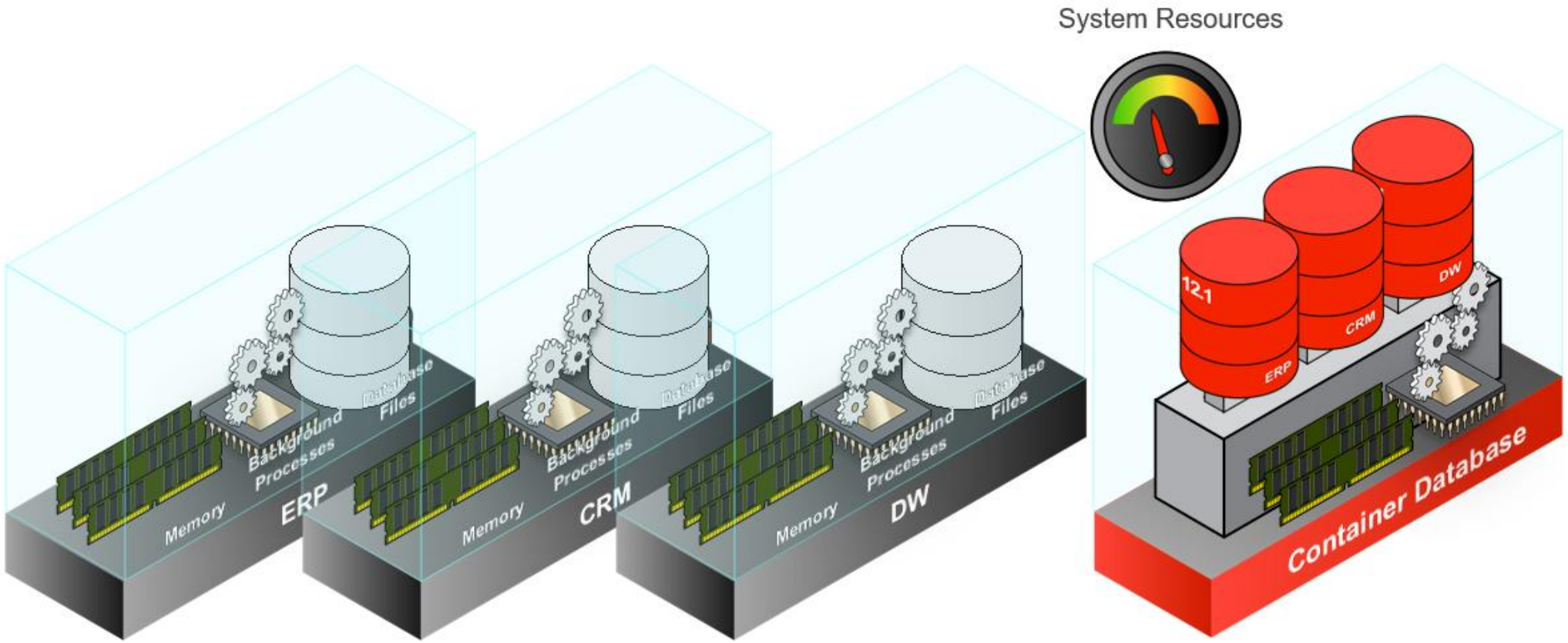



Megosztott hardware, OS és DB

EVOLÚCIÓ

Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

# Oracle Cloud – Database as a Service - Evolúció




Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

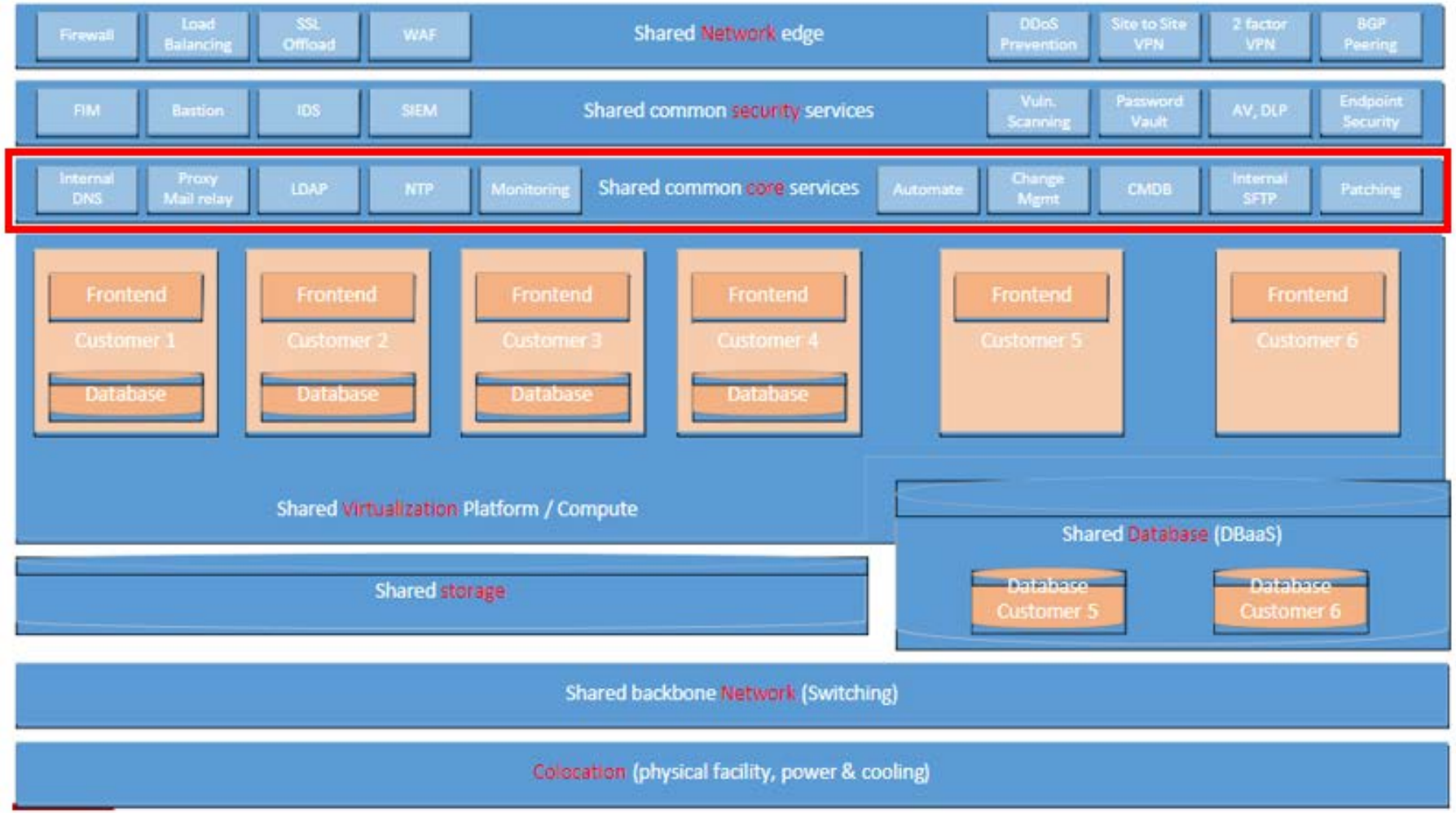
# Felhő és ami mögötte van – Cloud adatbázisok – 12c

- Max 4096 PDB egy konténer adatbázisba – gyakorlatilag még egy igazán nagy vállalatnak is elég 1 konténer adatbázis az összes alkalmazásának futtatására
  - Isolalt, mégis multitenant
  - Sokkal hatékonyabb erőforráskezelés
  - Sokkal hatékonyabb karbantartás
  - Nem kell alkalmazás kódot változtatni, működik az alkalmazás továbbra is
- Erőforrás szabályozás CPU, I/O és memória szinten
- Sharding – adatbázisokat több szerver között el lehet oszatni a párhuzamosítás, teljesítmény és megbízhatóság növelés érdekében
- Hot cloning – leállítás nélküli klónozás, migráció szerverek vagy éppen adatközpontok között
- Active Data Guardon is lehet lekérdezéseket futtatni – csökkentve a terhelést az éles rendszereken

ORACLE

Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

# Felhő és ami mögötte van




Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

# Felhő és ami mögötte van – Shared Core Services

- Internal DNS – belső és külső névfeloldás
- Mail servers – belső és külső levelezés
- LDAP – belső authentikáció, user access mgmt, stb...
- NTP – időszinkronizáció – minden szerver ugyanazt az időt mutassa
- Monitoring – Proaktív hiba felderítés és elhárítás
- Automation – automatikus szerver létrehozása, automatikus patchelés, stb...

ORACLE

Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

# Felhő és ami mögötte van – Shared Core Services

- ITIL alapú Change management – hogyan tervezzünk és hajtsunk végre bármilyen változtatást, úgy hogy minél biztonságosabb legyen a végrehajtás, és minimális legyen annak az esélye annak, hogy valami rosszul sül el.
- ITIL alapú Problem management – felelős folyamat hogy visszatérő incidensekre végső megoldást kínáljon, illetve hogy megakadályozza incidensek ujra előfordulását
- ITIL alapú Incident management - a cél, hogy olyan folyamatot alkossunk, hogy incidens során a leggyorsabban visszaállítsuk a teljes szolgáltatást, a lehető legkisebb hatással a többi ügyfélre illetve szolgáltatásra
- ITIL alapú Release management – folyamat a szoftver release-ekre: tervezés, tesztelés, ellenőrzött körülmények közötti kivitelezés.
- CMDB – egy olyan adatbázis, amiben tároljuk az összes, service-t leíró dolgot és ezek kapcsolatát

ORACLE

Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

# Felhő és ami mögötte van – Shared Core Services

## Host, Rendszer, Alkalmazás Monitoring.

## Konfiguráció kezelés, Ütemezési feladatok

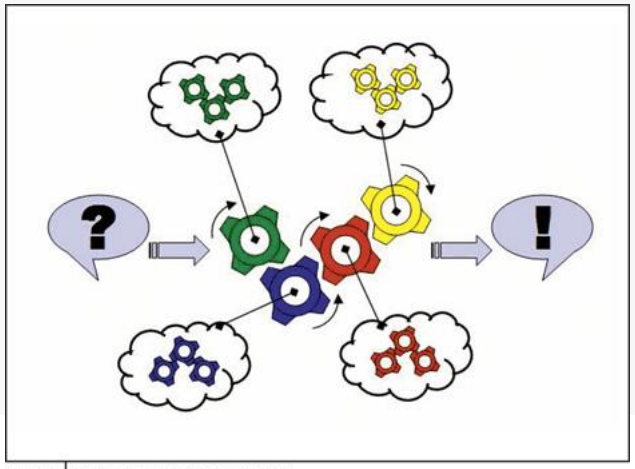
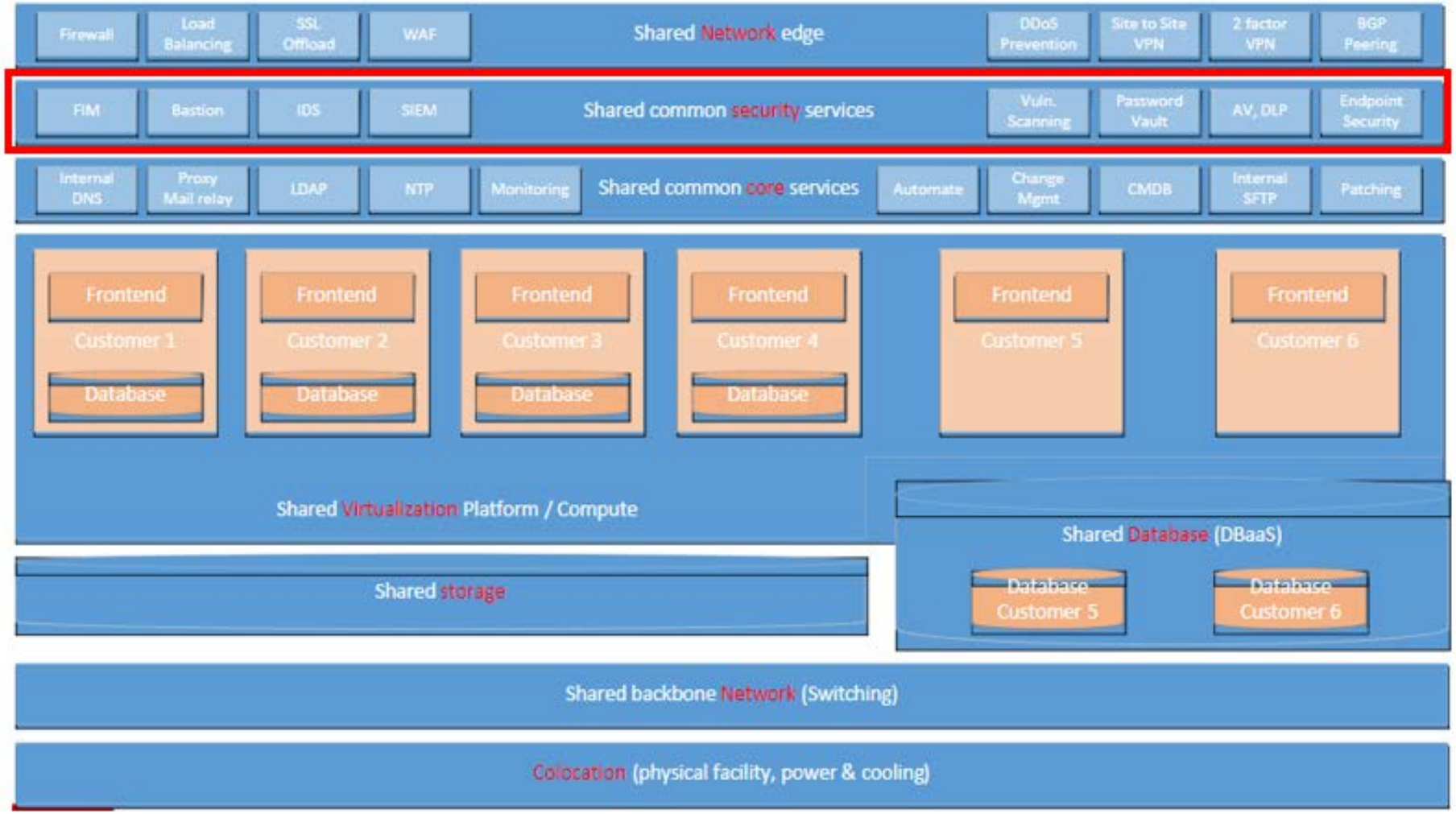



FIGURE 1 | Service composition business modeling

## Log és trend analízis

Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

# Felhő és ami mögötte van




Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

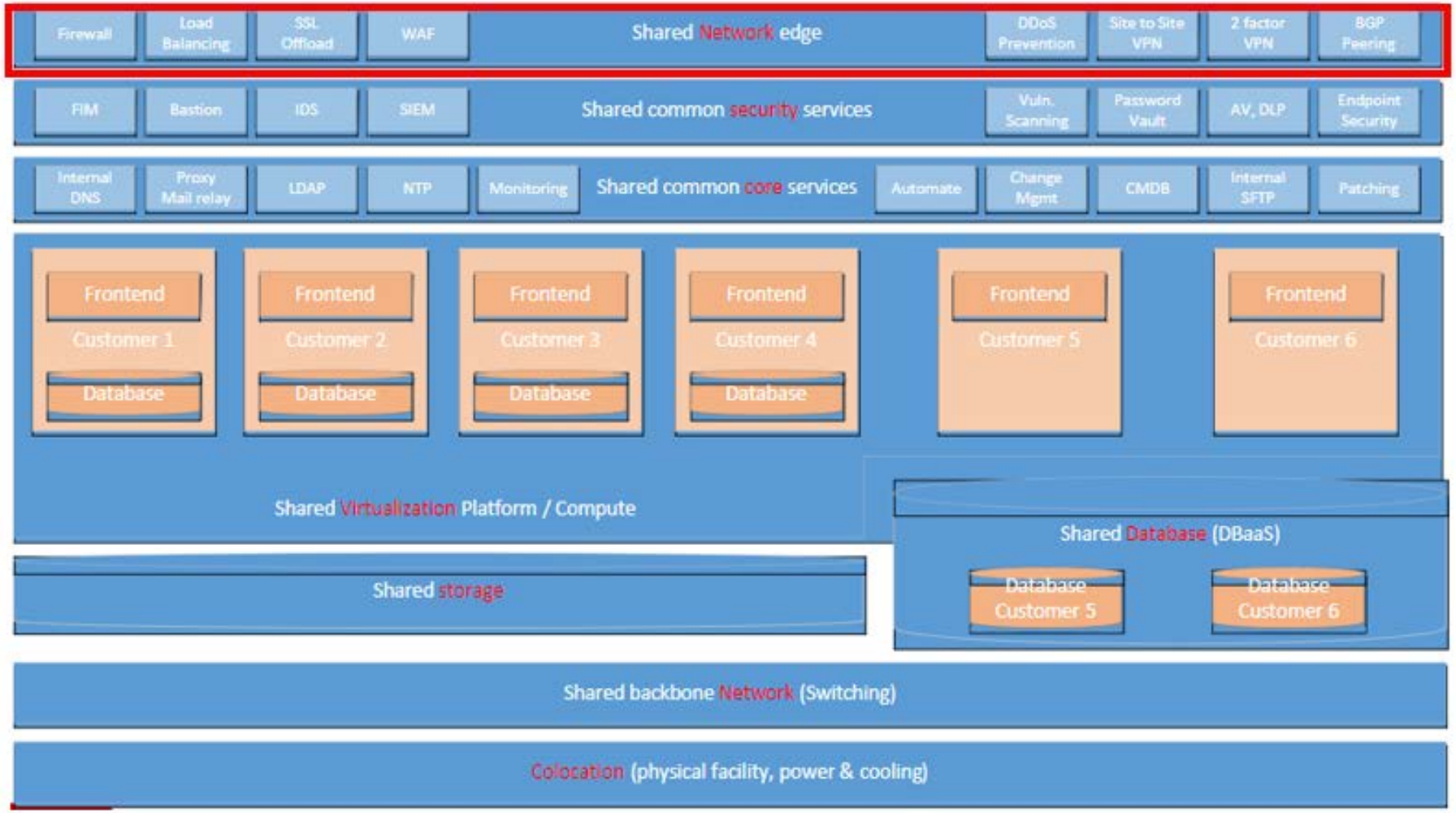
# Felhő és ami mögötte van – Security Services

- A felhők biztonsága az egyik legnagyobb kérdőjel az ügyfeleknél
- A felhő mégis biztonságosabb:
  - A felhőben mindenki egyforma – a cégvezetőnek, a dolgozóknak, az informatikusnak csak ahhoz van joga, ami a munkájához kell, és nem több. Saját rendszernél éppen a túl sok jog ad támadási felületet
  - A fizikai hozzáférés rendkívül szigorú: biztonsági szolgálat, ujjlenyomat, akár retina szkenner vagy arcfelismerés
  - Beléptetés mindig bonyolultabb és biztonságosabb
  - Vírus- és behatolásvédelem nagyon magas szintű
  - Külső és belső sebezhetőség szkennelése, gyenge pontok feltérképezése
  - Szabványok és tanúsítványok kötelezik a szolgáltatókat arra, hogy a szervereket és a szolgáltatásokat folyamatosan karban tartsák (pl. Windows patch) illetve az adatokat megvédjék – biztonsági mentések.
    Ezek betartását a tanúsítványt kiadó szervezetek folyamatosan ellenőrzik

ORACLE

Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

# Felhő és ami mögötte van




Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

# Felhő és ami mögötte van – Külső hálózat

- Kívülről érkező kapcsolatok kezelése:
  - VPN
  - Load balancing
- Kívülről érkező támadásokra:
  - Hardware tűzfalak
  - WAF – Web Application Firewall – pl. URL átírása és malware elrejtése benne
  - DDoS támadások detektálása és visszverése

ORACLE

Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

# Software as a Service (SaaS)

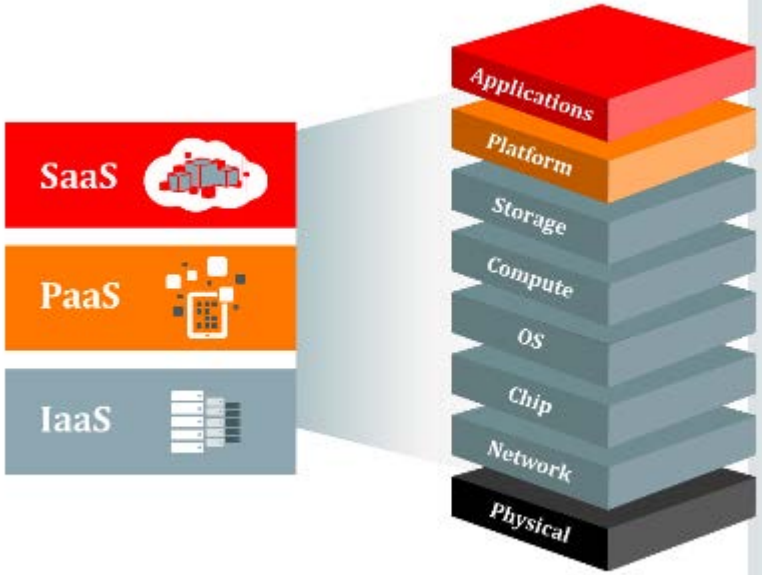
- A SaaS megoldás lehetővé teszi a felhasználóknak, hogy regisztrálják magukat a felhőbe és használják a felhő nyújtotta alkalmazásokat, legyen az vállalat vagy magánember.
- A modern SaaS rendszerek lehetőséget biztosítanak arra is, hogy a felhasználók magukra szabhassák a környezetet, és biztosítják a könnyű csatlakozást bárhonnan, bármilyen eszközről, biztonságosan.

ORACLE

Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

# Software as a Service (SaaS)

- SaaS szolgáltatás esetén a szoftver license előfizetés alapon működik és központilag szolgáltatunk.
- SaaS szolgáltatások túlnyomó többsége multitenant, ami annyit jelent, hogy egy adott alkalmazás egy adott beállítással működik minden ügyfélnek.
- A skálázhatóság és terheléselosztás miatt, ez a standard alkalmazás több gépre is telepítve van, és párhuzamosan futnak.
- SaaS szolgáltatásokat az ügyfelek minden eszközről elérhetik annélkül, hogy bármilyen változtatást kellene végrehajtani az eszközön, illetve bérmit is kellene telepíteni.




Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

# Software as a Service (SaaS)

- SaaS alkalmazásokat testre lehet szabni az ügyfél igényi szerint, de azzal számolni kell, hogy ezek költségeit az ügyfélnek kell állnia.
- A SaaS alkalmazásokat a tradícionális alkalmazásokkal szemben gyakrabban update-elik, frissítésket adhatnak ki akár hetente is.
- A SaaS alkalmazások általában nem rendelkeznek hozzáféréssel az ügyfél egyéb, nem SaaS rendszereihez, ezért a SaaS alkalmazások rendelkeznek biztonságos protocollokkal, interface-ekkel, amik segítségével mégis össze lehet kötni a 2 rendszert

ORACLE

Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

# Software as a Service (SaaS)

A SaaS modellnek 2 fő fajtája létezik:

- Vertikális SaaS

  Az a szoftver, ami egy ipari ágaztnak nyújt teljes szolgáltatást (pénzügy, ingatlan, mezőgazdaság)
- Horizontális SaaS

  Termékek, amik ezen ágazatokon átívelő szoftver kategóriák (marketing, sales, HR), tehát nem ágazat specifikus.

ORACLE

Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

# Software as a Service (SaaS) – Előnyök

- Üzleti szempontok
  - Üzemeltetési költségek csökkentése
  - Fejleszetési költségek csökkentése,
  - Skálázhatóság – olyan mértékben kell fizetni, amilyen mértékben növekszik a cég - kockázatmentes
- Technológia
  - Továbbfejlesztett WEB alapú technológiák alkalmazása
  - Továbbfejlesztrett SSL biztonsági proptokollok (HTTPs)
  - Nagy sebességű internet vonalak, fejlett mobilinternet technológiák
  - Sztéleskörű integrációs protokollok és interface-ek

ORACLE

Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

# Software as a Service (SaaS) - Kihívások

- Adatbiztonság
- Hálózati késleltetés
- Multi tenancy korlátozza a testreszabási lehetőségeket
- Külső nagy adathalmaz integrálása
- Alkalmazás fejlesztésének leállítása, alkalmazás kivezetése
- Szolgáltató betartja az uptime-ot?
- Szolgálató váltás

ORACLE

Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

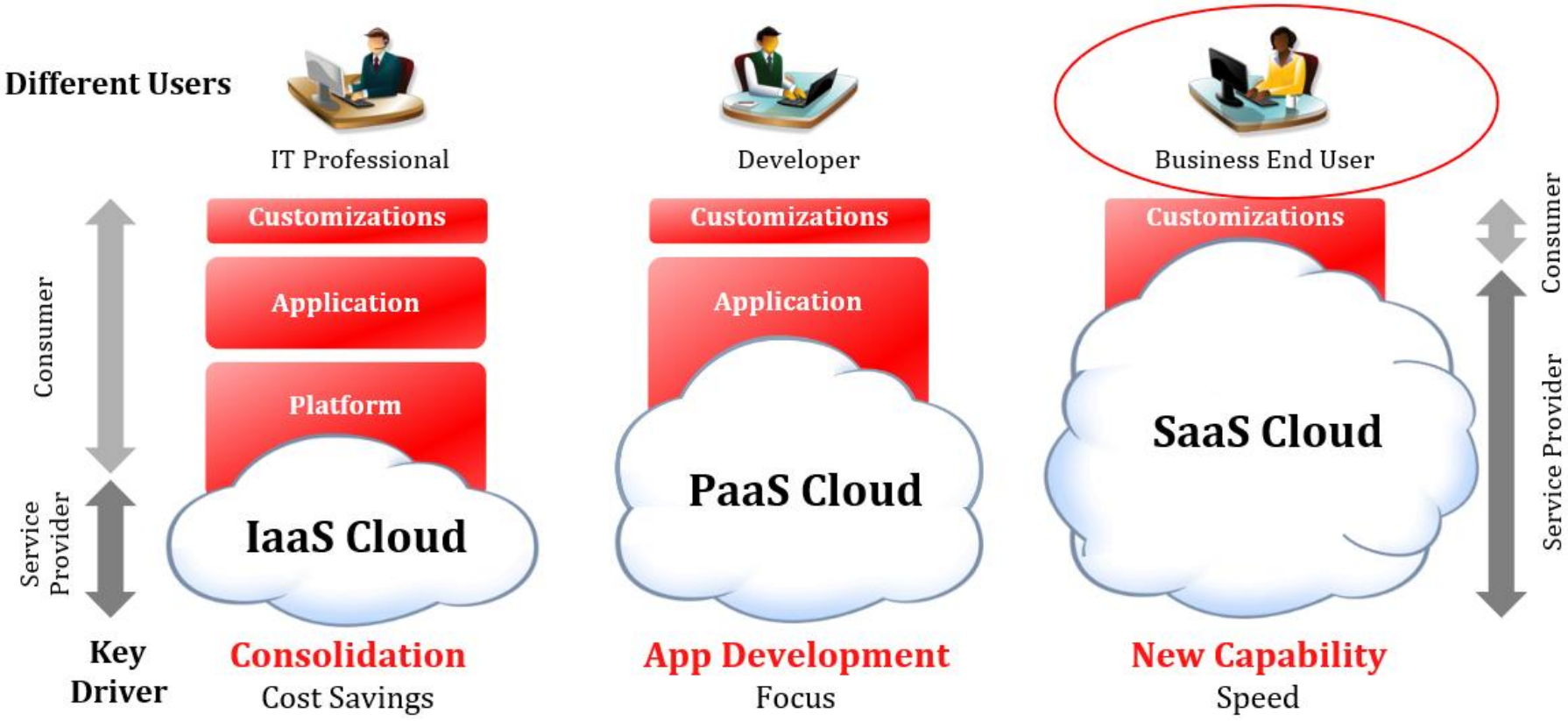
# Tradícionális modell és Saas modell

| Tradícionáls szoftver | Software as a Service |
|---|---|
| Dedikált telepítés | Multitenant telepítés |
| Skálázhatóság limitált | Skálázhatóság az egyik kulcs, dinamikus erőforráskezelés |
| Úgy tervezték, hogy mindenki sajét magának telepít | Interneten keresztül, telepítés nélkül használható |
| A verzió váltások és upgrade-ek az ügyfelek által kivitelezettek | A szolgáltató dönti el mikor és mit upgrade-el |
| Az upgrade-nek költségei vannak, amit az ügyfél visel | Ingeyn upgrade, a költségeket a szolgáltató állja |
| Testreszabott integrációs megoldások más rendszerekkel | Interface alapú integrációs megoldások - standardizált |

ORACLE

Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

# Felhő alapú szolgáltatások




Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

# Felhő alapú szolgáltatások

| Saját adatközpont | IaaS | PaaS | SaaS |
|---|---|---|---|
| Alkalmazások | Alkalmazások | Alkalmazások | Alkalmazások |
| Alkalmazás adatok | Alkalmazás adatok | Alkalmazás adatok | Alkalmazás adatok |
| Runtime | Runtime | Runtime | Runtime |
| Middleware | Middleware | Middleware | Middleware |
| Operációs rendszer | Operációs rendszer | Operációs rendszer | Operációs rendszer |
| Virtualizáció | Virtualizáció | Virtualizáció | Virtualizáció |
| Szerverek | Szerverek | Szerverek | Szerverek |
| Tárhely | Tárhely | Tárhely | Tárhely |
| Hálózat | Hálózat | Hálózat | Hálózat |

ORACLE

Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

# Pizza as a Service

| Saját adatközpont | IaaS | PaaS | SaaS |
|---|---|---|---|
| Mosogatás | Mosogatás | Mosogatás | Mosogatás |
| Terítés, üdítők | Terítés, üdítők | Terítés, üdítők | Terítés, üdítők |
| Sütés | Sütés | Sütés | Sütés |
| Összekészítés | Összekészítés | Összekészítés | Összekészítés |
| Tészta kelesztés | Tészta kelesztés | Tészta kelesztés | Tészta kelesztés |
| Szósz elkészítése | Szósz elkészítése | Szósz elkészítése | Szósz elkészítése |
| Tészta gyúrása | Tészta gyúrása | Tészta gyúrása | Tészta gyúrása |
| Hozzávalók beszerzése | Hozzávalók beszerzése | Hozzávalók beszerzése | Hozzávalók beszerzése |

Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

# Oracle mint szolgáltató

„Száz évvel ezelőtt a vállalatok megszüntették saját áramtermelő berendezéseiket, gépeiket, és csatlakoztak az elektromos hálózathoz… Napjainkban ugyanez a forradalom közeleg az informatika világában."

(Nicholas Carr - Harvard Business Review – főszerkesztő)

Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

# Támogatott Szegmensek és számok – Oracle mint szolgáltató

- Constructions and Engineering Üzletág – 8000 milliard dollár összértékű projectre több mint 100.000 cégnek nyújt segítséget
- Financial Üzletág – 1 milliárd számla, több mint 9000 pénzügyi szolgáltatónál, 1 milliárd kártyás tranzakció naponta
- Health Science Üzletág – TOP10 gyógyszergyártó cég szoftware-es támogatása
- Hospitality Üzletág – több mint 35000 szálloda és 200.000 étterem
- Retail – 6000 online webshop és weboldal
- Utilities Üzletág – 550 millió ügyfél energetikai adatainak elemzése évi 700 millárd leolvasással, és kb 1.1 milliárd dollár spórolással

ORACLE

Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

# Oracle mint Infrastruktúra szolgáltató

- Szoftveres virtualizáció
- Költséghatékony és rugalmas számítási, tárhely és hálózati kapacitás
- Standard környezet, az ügyfeleknek nem kell újraírniuk a meglévő applikációkat, és már automatizált folyamataikat
- Infrastruktúra monitoring
- Hardware támogatás
- Erős biztonság
- Oracle Bare Metal Cloud Services.

Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

# Oracle mint Platform szolgáltató

## Adatkezelés

## Alkalmazás fejlesztés

## Üzleti intelligencia

## Big Data

## Mobilitás

## Content & Social

## Enterprise Integráció

## Rendszer Felügyelet

ORACLE

Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

# Oracle mint szoftver szolgáltató

- A SaaS modell egyszerű és átlátható, azonnal bevezethető és a végfelhasználók által azonnal alkalmazható
- Mindent a szolgáltató ad, a felhasználó csupán némi konfigurálást hajt végre
- Internetes elérés, nem kell lokális telepítés
- Teljes körű felügyelet
- Licenszeket a szolgálató kezeli
- Szoftver karbantartás
- Verziókövetés
- Mérés és monitoring

Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

# Modern Oracle Cloud

**Complete** — Teljeskörű szolgáltatás a teljes üzletmenethez

**Data Driven** — Egységes és feldolgzásra kész, megbízható információk

**Personalized** — Testreszabható

**Connected** — Bárki, bárhonnan, bármit

**Secure** — Legmagasabb biztonság és teljeskörű felügyelet

Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

# Oracle SaaS

Egyszerű és gyors kiépítés, azonnal használatra kész

Legfejlettebb eszközök az alkalmazások kezeléséhez

Egyszerű alkalmazás fejlesztés, szofisztikált monitoring

Egyszerű, mégis minden igényt kielégítő alkalmazáskezelés

Magasfokú biztonság

Forradalmian új integrációs lehetőségek

ORACLE

Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

# Monitoring - Enterprise Manager Cloud Control 13*c*

- Teljeskörű adminisztrációs, support és monitoring rendszer
- Különböző gyártók hardware-eit, és különböző operációs rendszereket képes kezelni.
- Szoftverek és operációs rendszerek patch szintjének karbantartása
- Alkalmazások és hardware teljesítmény monitoring
- Képes cloud es nem cloud alkalmazások együttes monitoringozására és karbantartására
- Hibákra és monitoring jelzésekre automatikus választ képes adni
- Statisztikák, reporting

Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

# Monitoring - Enterprise Manager Cloud Control 13*c*

ORACLE Enterprise Manager Cloud Control 13c

Enterprise ▾ Targets ▾ SYSMAN ▾

## Enterprise Summary

Auto Refresh Off Page Refreshed 12-Nov-2016 12:37:01 CET

### Overview

View All Targets (90)

**Status**

Targets with Status 56

16%

84%

Up (47)
Down (9)

**Incidents**

Updated in the last 24 hours 10

Updated in last 7 days 11

Breakdown of incidents updated in the last 7 days

| Category | | | | |
|---|---|---|---|---|
| Availability | 8 | 2 | - | - |
| Performance | - | - | - | - |
| Security | - | - | - | - |
| Others | - | 1 | - | - |

**Problems**

Total Open 13

Updated in the last 24 hours 2

Without Service Request 13

**Jobs**

Suspended Executions 0

### Inventory and Usage

Show Hosts Details

14%

86%

Oracle Linux Server release 6.7
Enterprise Linux Enterprise Linux Server release 5.6 (Carthage)

### Compliance Summary

Standards Targets

Use Compliance Score Violations View Trends Target Type Host

| Target Name | Target Type | Average Compliance Score (%) |
|---|---|---|
| em12c.mycorpdomain.com | Host | 51 |
| ovm.mycorpdomain.com | Host | 51 |
| myserver4.mycorpdomain.com | Host | 100 |
| myserver2.mycorpdomain.com | Host | 100 |
| myserver1.mycorpdomain.com | Host | 100 |

### Patch Recommendations

View by Classification Target Type

Controlled
Other Recommendations 36
Security 37
Superseded

https://www.nloug.nl/downloads/ogh20161123_rob_zoeteweij.pdf

ORACLE

Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

# Monitoring - Enterprise Manager Cloud Control 13*c*

ORACLE Enterprise Manager Cloud Control 13c

SYSMAN

oemrep.mycorpdomain.com

em12c.mycorpdomain.com Auto Refresh Off

Oracle Database Performance Availability Security Schema Administration

Page Refreshed 12-Nov-2016 12:55:07 CET

12.1.0.2.0 Version

0 days, 3 hrs Up Time

5.63% Availability for Last 7 Days

N/A Last Backup

**Load and Capacity**

0.15 Average Active Sessions
11.57 Used Space (GB)

**Incidents and Compliance**

0 1 0 0
Compliance Not Configured

**Recommendations**

0 ADDM Findings
17 SQL Tuning

**High Availability**

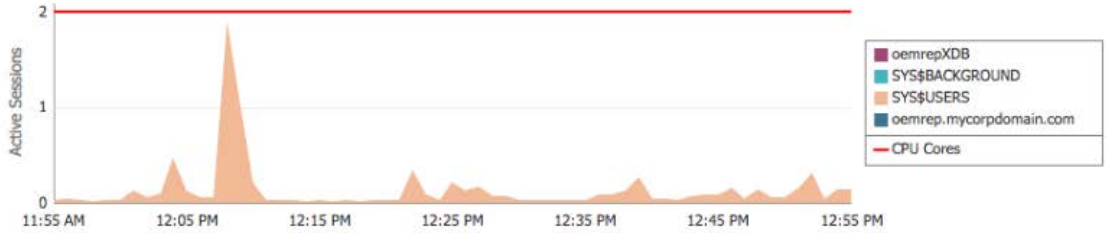
N/A Last Backup Status
Data Guard Not Configured

**Jobs**

0 Running

**Performance**

Activity Class | Services




**Resources**

**SQL Monitor - Last Hour**

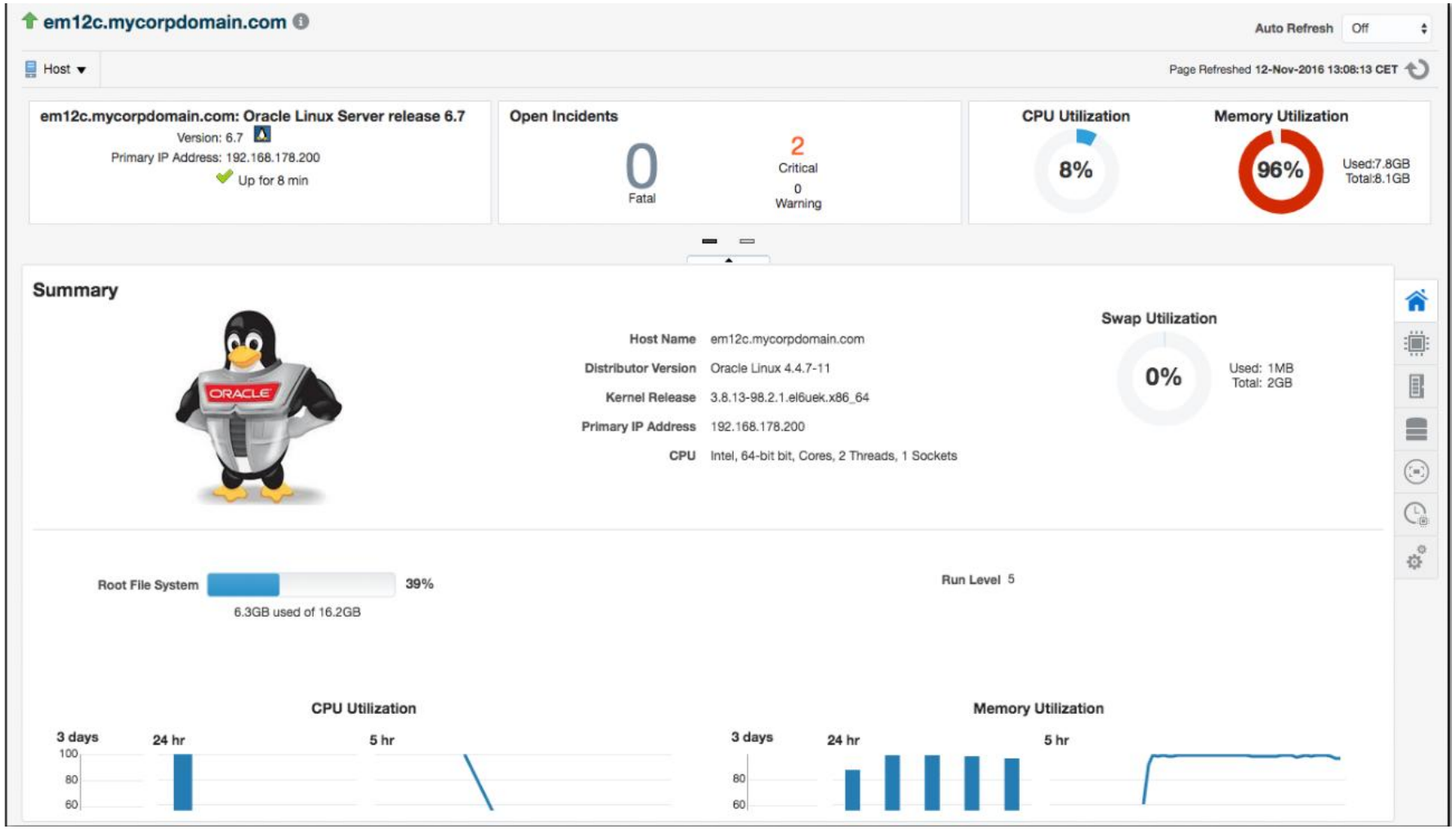
| Status | Duration | SQL ID | Session ID | Parallel | Database Time |
|---|---|---|---|---|---|
| ✓ | 6.00 s | 4dkz6cu1n9zmj | 274 | | 5.78 s |
| ✓ | 9.00 s | c3hjkha2s5q42 | 274 | | 8.95 s |
| ✓ | 6.00 s | 4dkz6cu1n9zmj | 75 | | 5.96 s |
| ✓ | 6.00 s | 4dkz6cu1n9zmj | 42 | | 5.96 s |
| ✓ | 22.00 s | 4dkz6cu1n9zmj | 302 | | 22.83 s |

https://www.nloug.nl/downloads/ogh20161123_rob_zoeteweij.pdf

ORACLE

Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

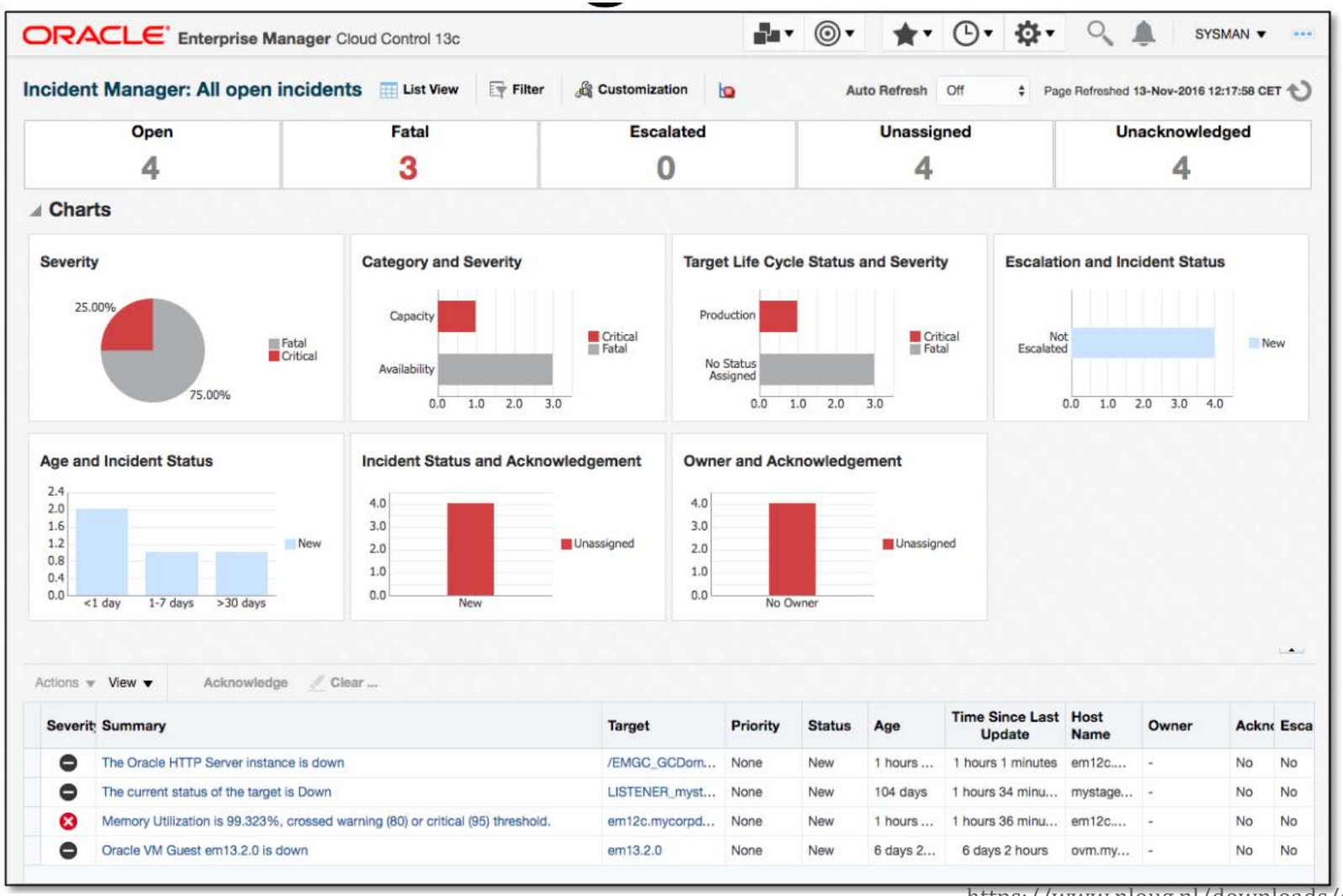
# Monitoring - Enterprise Manager Cloud Control 13*c*




https://www.nloug.nl/downloads/ogh20161123_rob_zoeteweij.pdf

Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

# Monitoring - Enterprise Manager Cloud Control 13*c*




https://www.nloug.nl/downloads/ogh20161123_rob_zoeteweij.pdf

Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

ORACLE®

# Security alert

NEMZETI KIBERVÉDELMI INTÉZET
GOVCERT

Nemzeti Kibervédelmi Intézet
Kormányzati Eseménykezelő Központ

*Ikt. sz.: 30700-0/2070-17/2018*

**TLP:WHITE**
**Szabadon terjeszthető!**

**Riasztás**

**Microsoft és Adobe termékeket érintő sérülékenységekről**

(2018.10.10.)

https://www.nloug.nl/downloads/ogh20161123_rob_zoeteweij.pdf

ORACLE

Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

# Wannacry

- 150 ország
- 200.000-nél is több fertőzött gép

- 327 fizetés történt $130,634.77 USD (kb 36.5 millió Ft) értékben.
- 4 nap után teljesen megállították a vírust

ORACLE

Copyright © 2014 Oracle and/or its affiliates. All rights reserved. |

PEOPLE COME FIRST
INFORMATIKAI SZAKÉRTŐK EGYESÜLETE

# Docker & Kubernetes

Konténer alapú virtualizáció

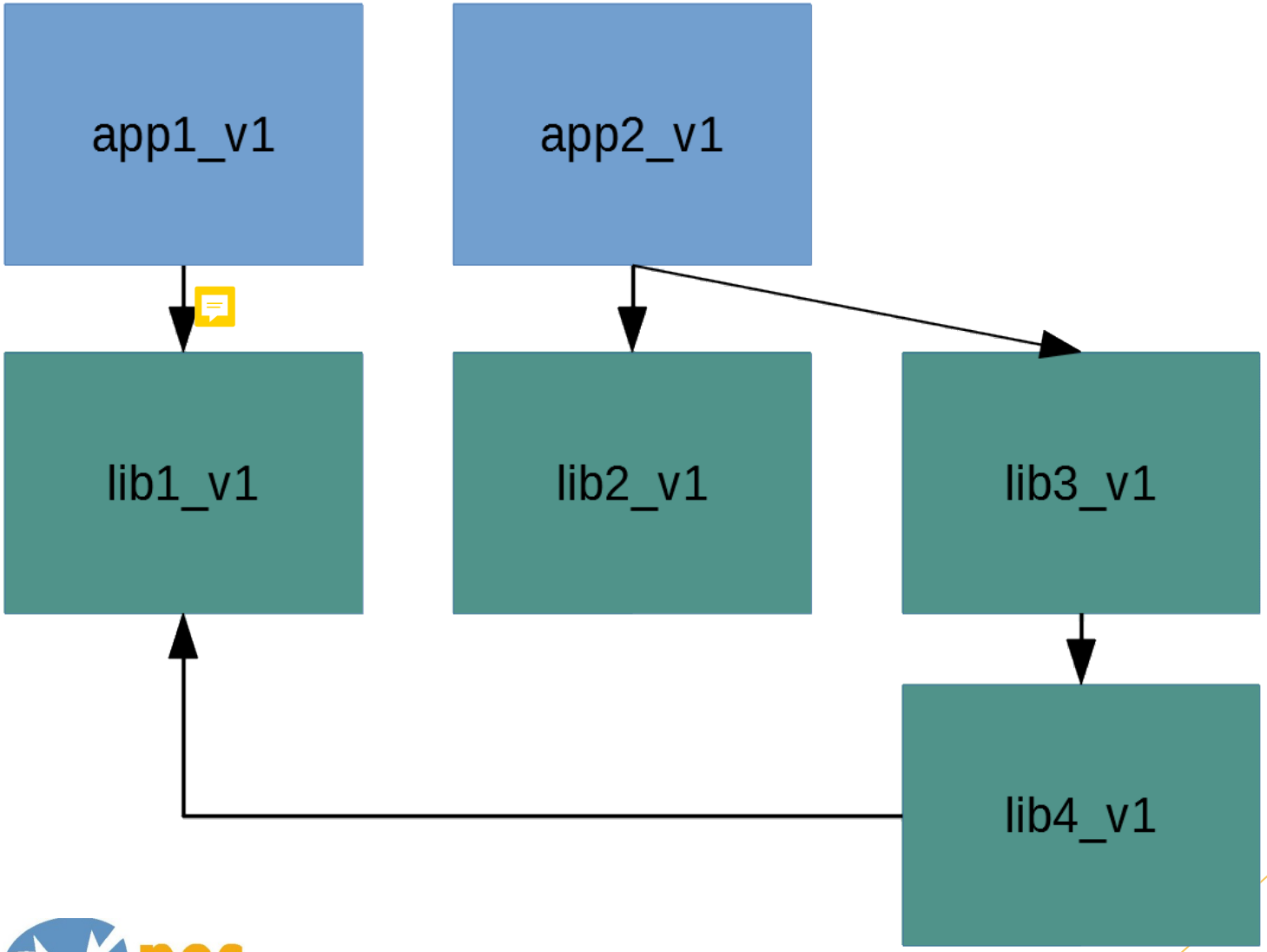
# Tematika

- Fizikai gépektől a konténerekig
- Linux konténerek – technológiai háttér
- Docker alapok és hiányosságok
- Elosztott konténerkezelés - Kubernetes

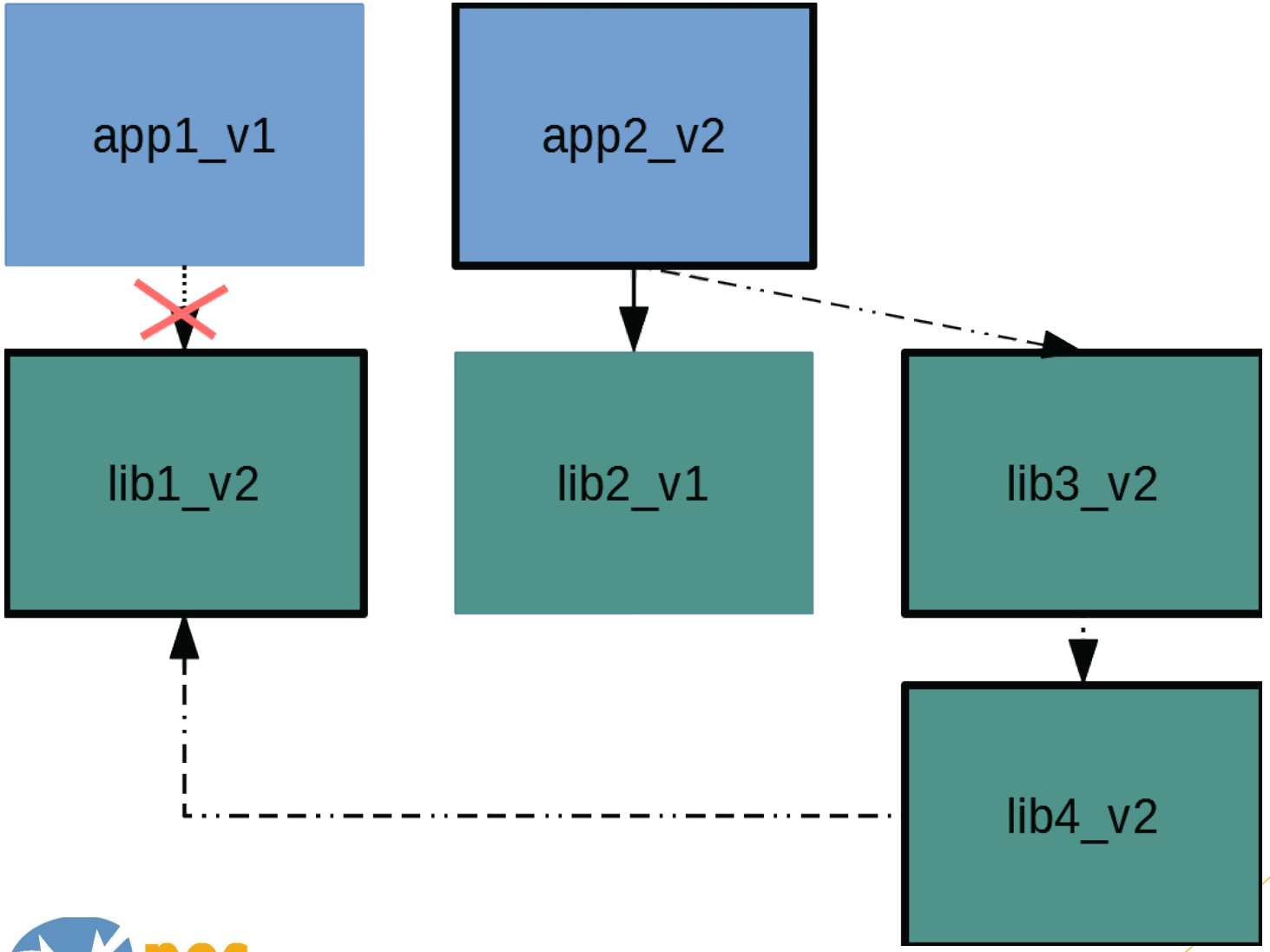
# Fizikai gép

- “Vas”
- Egy gépen több alkalmazás
- Szeparáció: külön OS userek, ~~virtualenv~~, stb.
- Például: web hosting
- Probléma: inkompatibilis csomag igények, erőforrás-kihasználtság

# Függőségek

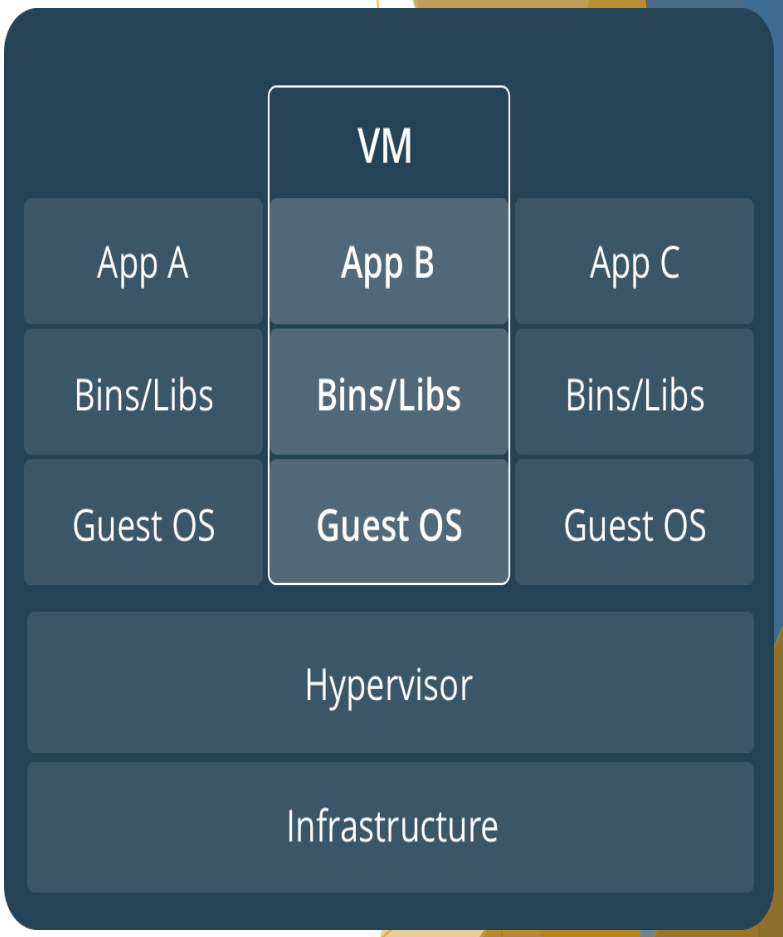


# Függőségek



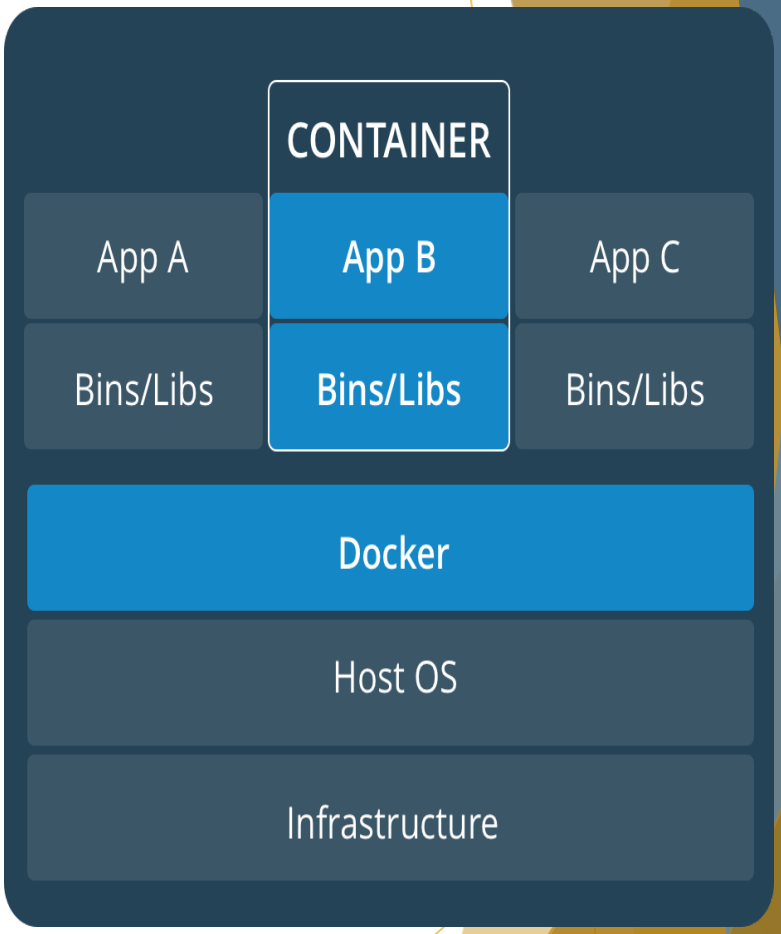
# Virtuális gép

- Egy fizikai gépen több
- Szeparáció: saját kernel
- Példa: Infrastucture as a Service (IaaS)
- Probléma: magas önköltség (1 app / VM?)



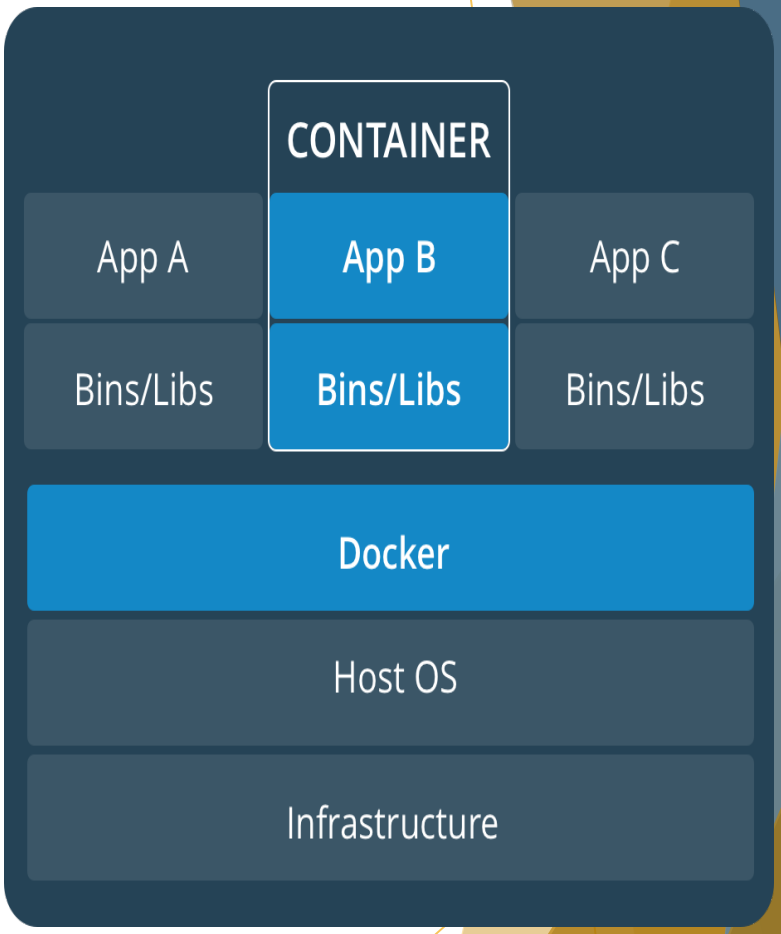
# Konténer

- Egy kernelben több
- Minimalista: csak az alklalmazás és annak függőségei
- Jobb erőforrás-kihasználtság



# Docker konténer

- "A container image is a lightweight, stand-alone, executable package of a piece of software that includes everything needed to run it: code, runtime, system tools, system libraries, settings."



# App vs. OS konténer

- OS konténerek
  - Linux Containers (LXC)
  - Solaris Zones
- Alkalmazás konténerek
  - Docker
  - Rocket (rkt)

# Docker technológiai háttér

- Google fejlesztések (saját konténer technológia)
- Linux kernel feature-ök:
  - cgroups (irányítás)
  - namespaces (szeparáció)
- “Kézzel hajtós” konténer: https://youtu.be/sK5i-N34im8

# Miért Docker?

- Elterjedt (6 milliárd letöltés 2016-ban)
- Könnyen használható (főleg fejlesztői oldalról)
- Hatékonyan automatizálható

# Mit ad a Docker?

- Egységes API
  - Konténer vezérlése (CLI, REST)
  - Konténer létrehozásra (Dockerfile)
- Konténerek csomagolása
  - Újrahasználhatóság
  - Verziókezelés
  - Disztibúció (registry)

# Docker Image

- A “becsomagolt” konténer
  - Image vs konténer ~ bináris fájl vs processz
- Base image: minimális OS eszközök
- Verziózás: tagek segítségével
- Rétegekből áll
- Előállítás: Dockerfile alapján

# Rétegek (layers)

```
FROM node:argon

# Create app directory
RUN mkdir -p /usr/src/app
WORKDIR /usr/src/app

# Install app dependencies
COPY package.json /usr/src/app/
RUN npm install

# Bundle app source
COPY . /usr/src/app

EXPOSE 8080
CMD [ "npm", "start" ]
```

```
IMAGE        ...   CREATED BY                          SIZE
fdd93d9c2c60 ... /bin/sh -c CMD ["npm" "start"]   0 B
e9539311a23e ... /bin/sh -c EXPOSE 8080/tcp       0 B
995a21532fce ... /bin/sh -c COPY dir:50ab47bff7   760 B
ecf7275feff3 ... /bin/sh -c npm install   3.439 MB
334d93a151e ... /bin/sh -c COPY file:551095e67    265 B
86c81d89b023 ... /bin/sh -c WORKDIR /usr/src/app   0 B
7184cc184ef8 ... /bin/sh -c mkdir -p /usr/src/app  0 B
530c750a346e ... /bin/sh -c CMD ["node"]          0 B
```

# Registry vs Hub

- Image-ek disztribúciójára szolgál
- Publikus image tároló: https://hub.docker.com/
- Image le- és feltöltés: pull, push
- Lokális tárolás: docker images
- Verziókezelés: tag-ek (latest, 1.0)
- Privát hub: registry

# Hogy indul egy konténer?

- Letöltésre kerül a megfelelő taggel ellátott image
- A konétner saját belső fájlstruktúrája leképzeződik a fájlrendszerre
  - Tipikusan: /var/lib/docker/…
- Minden változás ebbe a könyvtárba kerül (copy-on-write), az image NEM változik

# Perzisztencia

- Van lehetőség perzisztens tárolásra
- Példa: adatbázis konténer (mysql, postgres)
- Volume-ok csatolásával
  - Lokális fájlrendszer
  - Hálózati megosztás
  - Másik konténer

# Konténer konfiguráció

- Csak default értékeket érdemes image-be építeni
- Bemenetként környezeti változókkal
- Bemenetként környezeti változókkal, de valamilyen külső kulcs-érték tárolóból (pl. Redis)
- Konfigurációs könyvtár becsatolása volume-ként

# Konténerek összeállítása

- Alapvetően CLI-n keresztül (docker run …)
- Függőségek (más konténerek) kezelése nehézkes (scriptelési feladat)
- Megoldás: docker-compose (YAML)
- Deklaratív leírás (Infrastructure as Code)
- Újra felhasználható
- De: külső verziókezelés, tárolás szükséges

# Logolás

- Hibakereséshez, auditáláshoz szükséges
- Alkalmazások: alapértelmezetten fájlrendszerre
- Ajánlás: kivezetés STDOUT-ra/STDERR-re
- Ezek automatikusan gyűjtésre kerülnek
- Használható külső loggyűjtő rendszer is (pl. syslog)

# Docker hiányosságok

- Egy gépes megoldás (kernelen belüli)
- Monitorozás: van beépített, de fapados
- Online konténer frissítés: nincs
- Skálázás, terheléselosztás: nincs
- A használt portok karbantartása nehézkes

# Elosztott Docker

- Docker Swarm – 1.12-től beépített
  - A problémák jelentős részére megoldást jelent
  - Egyszerű, de fapados
- Kubernetes
  - Jól bejáratott technológia (Google)
  - Jól konfigurálható, de összetett

# Miért Kubernetes?

- Elterjedtség és támogatás
- Kiváló Docker integráció
- Nagyon jól testreszabható
- A nehézségek nagy részére már van megoldás
- Erős felhő támogatás (GKE stb.)

# Kubernetes alapok

- Moduláris felépítés (szinte minden plug-in)
- Elvárt állapot alapú megközelítés
- Állapot leírása: YAML
- Namespace alapú láthatóság

# Kubernetes Node

- Két node típus:
  - Worker: alkalmazás konténerek
  - Master: a cluster vezérléséhez szükséges komponensek
- Kubelet: a node-on futó komponens
  - Konténerek indítása/leállítása
  - Állapot jelentése a Master felé

# Master komponensek

- API szerver: komponens kommunikáció
- Etcd: állapot tárolása https://raft.github.io/
- Kube-scheduler: hol fusson a Pod?
- Kube-controller-manager: az elvárt állapot alapján az aktuális állapot módosítása

# Kubernetes networking

- Minden konténernek tudnia kell kommunikálnia bármely más konténerrel NAT használata nélkül
- Minden node-nak tudnia kell kommunikálnia bármely konténerrel (és vissza) NAT használata nélkül
- A konténer saját magát ugyanazon az címen látja, mint bármely másik konténer vagy Node

# Pod

- Ütemezési egység
- Akár több konténer is lehet egyben
- Saját IP cím
- Elérés: service-eken keresztül

# Perzisztencia

- Háttértárolók elérésére: volume
- Ha nem csak időlegesen kell: PersistentVolume
- További absztrakció: PersistentVolumeClaim
- Ezeket az objektumokat vagy a Pod definícióban kell hivatkozni (adott csatolási ponthoz)

# Service

- Pod-ok címzése és terheléselosztás
- Label-ek alapján
- Elérés
  - NodePort: a node-ok saját címén is elérhető
  - ClusterIP: saját IP cím
  - LoadBalancer: cloud környezetben

# ConfigMap és Secret

- Kulcs-érték párok tárolására
- A Pod-okban hivatkozhatóak
  - Érték alapján
  - Fájlként becsatolva
- Konfiguráció és alkalmazás szétválasztása

# Deployment

- Konténerek számának skálázására
- Lehet automatikus (pl. CPU használat alapján)
- Beépített, kiesés nélküli frissítés:
  - Új példányok indítása
  - Régi példányok leállítása

# Helm

- Összetett alkalmazás architektúra leírására
- Verziókezelés
- Sablon támogatás

# Kubernetes telepítés

- Összetett feladat (rengeteg komponens)
- Könnyű és megismételhető telepítés kubeadm segítségével
- Ajánlott “irodalom”:

  Kesley Hightower: Kubernetes The Hard Way

  https://github.com/kelseyhightower/kubernetes-the-hard-way

# Kubernetes telepítés

- Összetett feladat (rengeteg komponens)
- Könnyű és megismételhető telepítés kubeadm segítségével
- Ajánlott “irodalom”:

  Kesley Hightower: Kubernetes The Hard Way

  https://github.com/kelseyhightower/kubernetes-the-hard-way

Köszönöm a figyelmet!

PEOPLE COME FIRST
INFORMATIKAI SZAKÉRTŐK EGYESÜLETE

# Nagyvállalati rendszerek

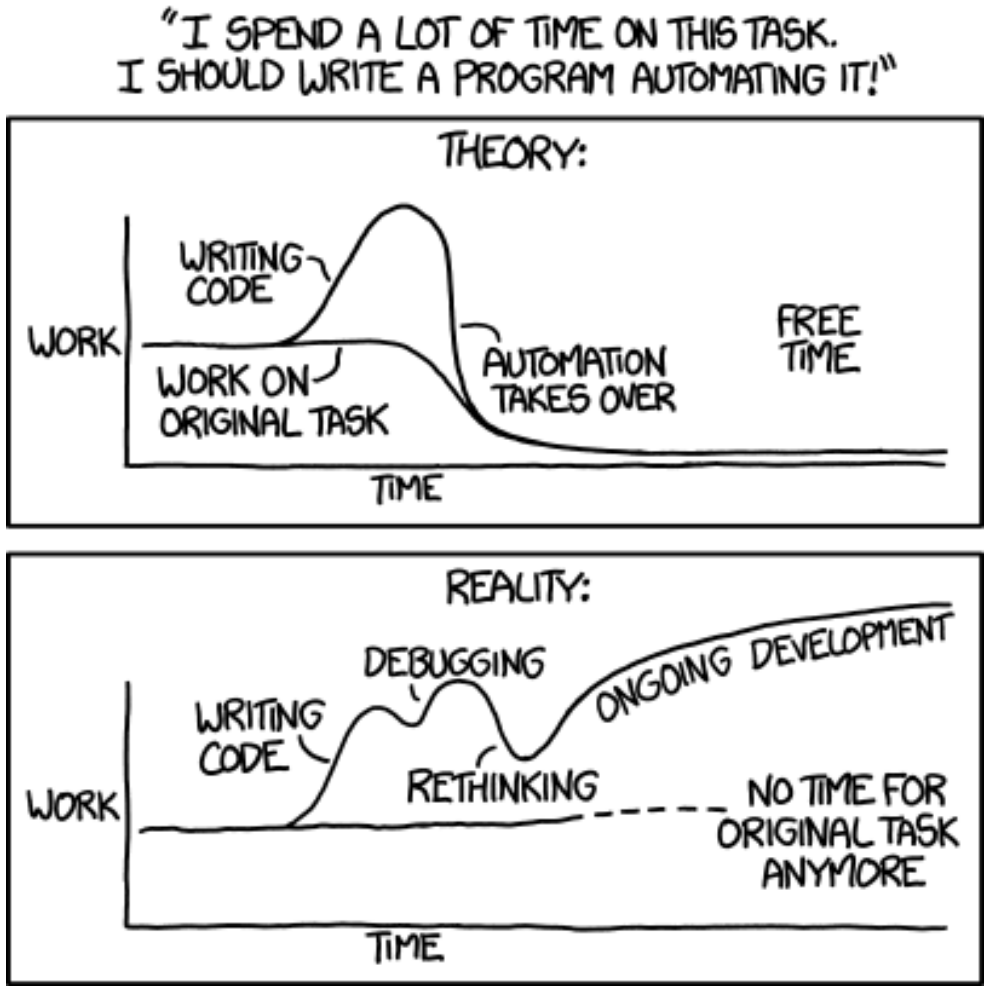
# Tematika

- Miért érdemes automatizálni?
- Mit érdemes automatizálni?
- Hogyan és mivel automatizáljunk
- Infrastruktúra automatizáló eszközök
- Saltstack

# Miért érdemes automatizálni?

- Munkaidőt lehet általa spórolni
  - Más feladatokkal többet lehet foglalkozni
- Gyorsabbá válhat tőle egy folyamat
  - Next, next, finish…
- Csökkenthető az emberi hiba előfordulási valószínűsége

# Automatizálás



# Mit érdemes automatizálni?

- Időigényes feladatokat
  - Adatbázis mentés
- Gyakori feladatokat
  - Log fájlok karbantartása
  - Frissítések telepítése
- Bonyolult feladatokat
  - Riportok
- Rutin feladatokat

# Automatizálási eszközök

- Legelterjedtebb: scriptek
- Monitorozó eszközök
- Workflow Engine-ek
- Alkalmazáson belüli képességek
- Ütemező szoftverek
  - Jellemzően scriptekkel kombinálva
- Dokumentáció

# Egy folyamat automatizálásának lépései

- Előfeltétel: a folyamat ismerete
- Első lépés: a folyamat dokumentálása
- 3 fázis: futtatási előtt, közben és után
  - Kell-e értesítés indulásról/hibáról stb.
- Legyen részfeladatokra bontva
- A feladathoz illeszkedő eszközt használjunk
- A lehető legkevesebb eszközt használjuk

# Infrastruktúra automatizálás

- Jellemzően hardware felett kezdődik
  - Cloud szolgáltatóknál van hardware-re is
- Operációs renszer telepítése
- Szoftverek telepítése és beállítása
- Felvétel monitorozásba

# OS telepítés automatizálása

- OS image-ek használata
  - Virtuális gépeknél VM sablon/klónozás
  - Fizikai gépeknél OS képfájl “terítése”
- PXE Network Boot
- Bizonyos operáció rendszerknél beépített mechanizmussal
  - CoreOS - Ingition

# Élet az OS telepítés után

- Jellemzően “A” telepítő script segítségével
- Probléma 1: különböző disztribúciók kezelése
  - httpd vs apache2
- Probléma 2: módosításra nem alkalmas
  - Csomag verzió váltásért ne kelljen OS újratelepítés

# Infrastructure as Code

- Nem csupán a szoftver, az infrastruktúra is “kódból” álljon elő
- Valamilyen deklaratív módon legyen leírva az infrastruktúra
- Legyenen eszközök arra, hogy ebből a deklaratív formátumból előállítható legyen a kívánt rendszer

# Megoldás: deklaratív eszközök

- Azt mondjuk meg, mit szeretnénk
  - “legyen feltelepítve X csomag”
- A keretrendszer kideríti, hogyan kell, és meg is csinálja
- Állapot alapú megközelítés
  - Csak akkor módosítsunk, ha szükséges
  - Kell mechanizmus az állapot ellenőrzésére

# Javasolt eszközök

- Puppet (Ruby)
- Chef (Ruby)
- Ansible (Python)
- SaltStack (Python)

# SaltStack

- Eredetileg távoli utasításvégrehajtásra
  - cmd.run modul
- YAML formátumban megadható állapotok
- Master/Minion komponensek
  - De van masterless és minionless mód is
- Könnyen bővíthető (Python)

# Salt Minion

- A célgépen fut
- A Mastertől megkapott állapotok/parancsok alapján dolgozik
- A feladatok eredményét visszaküldi a Masternek

# Salt Master

- Ide csatlakoznak be a Minionok
  - Titkosított forgalom (salt-key)
- Központi event bus
  - Minden Minion lájta
  - Scope: targeting alapján
- Állapotleírók jellemzően helyben
  - Van lehetőség pl. távoli git repository használatára

# SaltStack működési módok

- Master/Minion
- Masterless
  - Kézzel kell a leírókat hordozni, de nincs szükség service-ekre
- Minionless
  - SSH-n keresztül másolódik át a szükséges információ a célgépre
  - Jelentős átfedés az Ansible-el

# Salt Grains

- A gépről a Minion által szolgáltatott információk
- Lehet dinamikus (rendszeridő)
- Lehet statikus (disztribúció neve)
- Saját magunk is definiálhatunk Graineket
  - Tipikus felhasználás: gépek címkézése (csoportosítás)

# Salt Targeting

- A futtatott parancs “scope”-ját határozza meg (mely minionokra legyen érvényes)
- Mindenkire: “*”
- Adott gépre: “hostname.domainname”
- Listára: “host1,host2”
- Grain alapján:
  - Debian gépekre: “G@os:Debian”
  - x86-64-es gépekre: “G@cpuarch:x86_64”

# Salt State

- Elvárt állapot leírására szolgál
- YAML formátum
- Futtatás deklarációs sorrendben
  - De: függőségeket is tud kezelni
- Újra felhasználhatóak (include stb.)

# Salt State példa

```
/etc/httpd/extra/httpd-vhosts.conf:
  file.managed:
    - source: salt://webserver/httpd-vhosts.conf
apache:
  pkg.installed: []
  service.running:
    - watch:
    - file: /etc/httpd/extra/httpd-vhosts.conf
    - require:
      - pkg: apache
```

# Sablonok: Jinja2

- Fájlok tartalmának dinamikus előállítására
- YAML feldolgozás előtt fut
- State-ek generálására is használható
- Tipikus felhasználás: konfigurációs fájlok előállítása

# Salt Pillar

- Master oldali kulcs-érték tároló
- Fájl alapú (de: lehet pl. Redis-t használni)
- YAML formátum

# Jinja példa

```
motd:
  file.managed:
  {% if grains['os'] == 'FreeBSD' %}
    - name: /etc/motd
  {% elif grains['os'] == 'Debian' %}
    - name: /etc/motd.tail
  {% endif %}
    - source: salt://motd
```

```
motd:
  file.managed:
    - name: /etc/motd.tail
    - source: salt://motd
```

# Salt Formula

- Előre megírt State-ek, jelentős sablon használattal
- https://github.com/saltstack-formulas

# Salt Mine

- A Minionok csak a saját Grainjeiket érik el
  - Ha más Minionról származó adat kell: Salt Mine
- Master oldalon tárolódik, innen kérdezik le a Minionok
- A Master meghatározott időközönként frissíti az információt

# Salt Mine

- A Minionok csak a saját Grainjeiket érik el
  - Ha más Minionról származó adat kell: Salt Mine
- Master oldalon tárolódik, innen kérdezik le a Minionok
- A Master meghatározott időközönként frissíti az információt

PEOPLE COME FIRST
INFORMATIKAI SZAKÉRTŐK EGYESÜLETE

# IT rendszerek és architektúrák vállalati környezetben

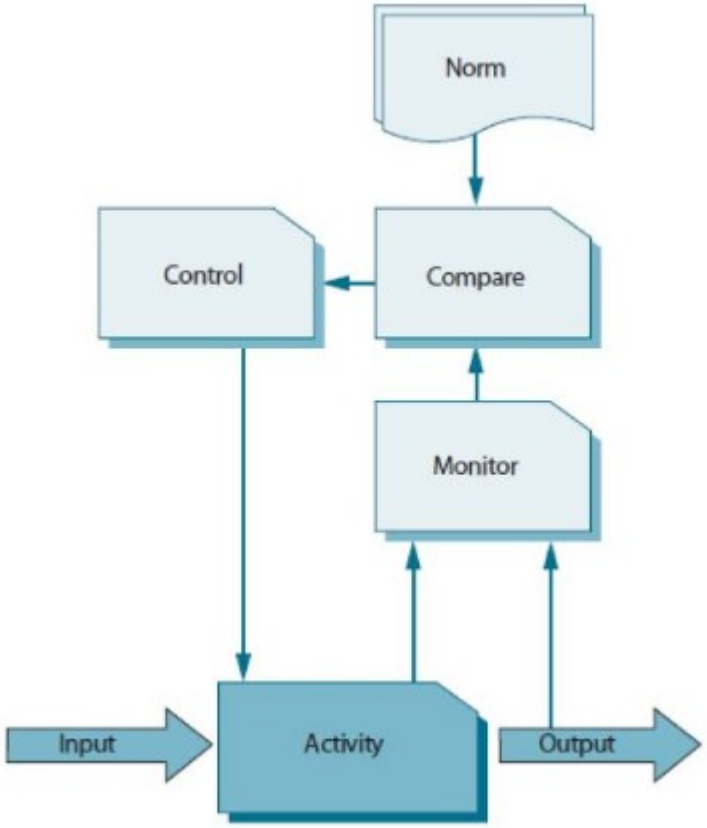
## Monitorozás, rendszerfelügyelet, hibaelhárítás

Előadó: Szabó Levente, Webváltó Kft.

# Monitorozás

- Monitorozás fogalma
- Folyamatos állapotkép
- Monitorozási ciklus
  - mérés
  - összehasonlítás (referencia érték)
  - vezérlés és visszacsatolás

# Monitorozási Ciklus (ITIL)



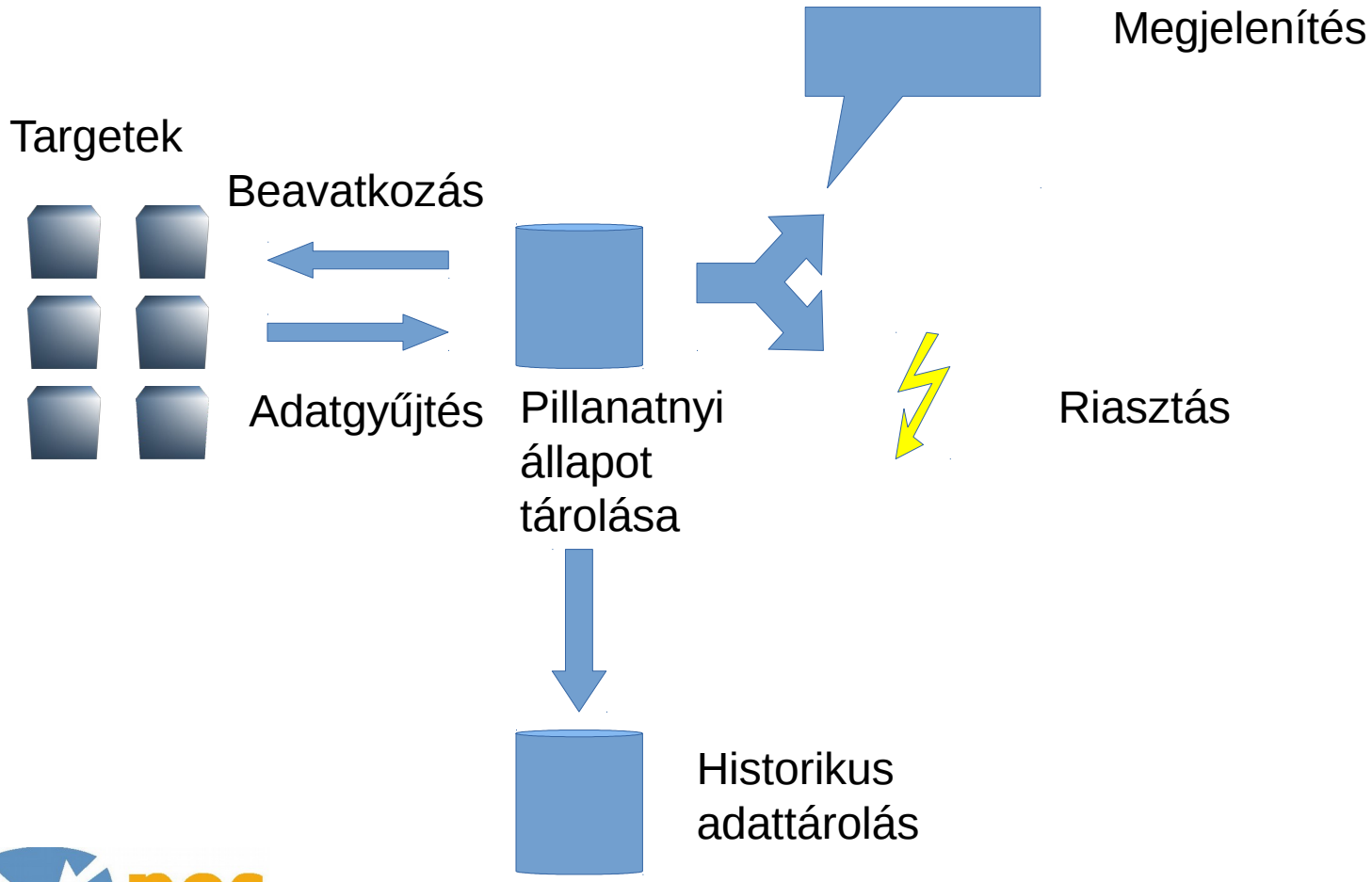
# Monitorozási Ciklus megjegyzések

- Ciklusok mért értékeinek historikus adattárolása, elemzés
- Referencia értéket többnyire nekünk kell meghatároznunk
- - környezet- / alkalmazás- / szituáció függő értékek
- - de nem feltétlenül statikus értékről van szó (pl. historikus átlag)
- Mekkora a ciklus optimális frekvenciája?

# Együttműködés más rendszerekkel

- Monitoring rendszerint nem csak kap, de adatokat is szolgáltat: nem kifejezetten monitorozási feladatokat végző rendszerek számára.
- Nagyobb rendszereknél a monitorozást nehéz különválasztani ezektől, pl:
- - esemény kezelés (riasztásból események generálódnak)
- - riporting, elemzés
- - kapacitás tervezés (erőforrások használatának időbeli tervezése)
- - configuration management
- - lifecycle management
- ...

# Általános Monitorozási Keretrendszer



# Monitorozási Megközelítések

- Beavatkozás alapján: Aktív / Passzív
- Incidens megelőzése: Reaktív / Proaktív
- Adatgyűjtés ciklusai szerint: Folyamatos / Kivétel alapú (Esemény vezérelt)

A valós életben általában ezek kombinációját alkalmazzák, pl:

- Adaptív monitorozó rendszerekben: változó frekvencia incidens alapján
-

# Adatgyűjtés megvalósítása

- Sokféle elemet szeretünk megfigyelni, ezeket a monitorozó keretrendszer meg is jeleníti:
  - Pillanatnyi értékek, skalár vagy diszkrét értékkészlettel
  - (memória kihasználtság, szerver fut/nem fut/ismeretlen)
  - Származtatott mérési adatok
    - kumulatív skalár és/vagy időablak átlag értékei
    - egyéb statisztikai jellemző
  - Napló bejegyzések, illeszkedő minták

# Agent / Agentless

- Agent: adatszolgáltatásra képes beépülő monitoring komponens
- Segítségével részletesebb adathalmazt figyelhetünk, beavatkozásra is képes lehet.

A dedikált agent és agent nélküli monitorozás között átmenetek is vannak:

Agentless ⟷ Dedikált Agent

Általános célú hozzáférés (SSH, Serial port, ping...)

Alapértelmezett (Syslog/SNMP)

Telepítést igényel (Nagios NRPE, JMX)

Dedikált hozzáférés (Oracle Management Agent Tivoli Monitoring Agent...)

# Agent / Agentless példák

- Agent nélkül is lekérdezhetjük pl. egy adatbázis kiszolgáló válaszidejét az adatbázishoz kapcsolódva.
- Nem integrált, de már a szerveren futó monitorozó Agent segítségével pl. szerver oldali jellemzőkhöz is hozzájuthatunk (CPU, Memória, fut-e a kiszolgáló háttérprocessz stb.)
- Integrált Agent, ami hozzáfér a kiszolgáló belső adatszerkezeteihez, sokkal részletesebb adatokat kaphatunk a monitorozni kívánt folyamatról.

# Lekérdezés iránya

- Pull: adatgyűjtő monitoring kezdeményezi a lekérdezést
- Push: agent kezdeményezi az adatszolgáltatást

Vigyázat, különböző megvalósítások pont fordítva használják!

# Speciális lekérdezések

- End-to-End monitoring, vagy szondázás
  - távolról kifejezetten a szolgáltatás/alkalmazás hozzáférési pontját használva kérdezünk a rendszer állapotáról (pl. alkalmazás szerver folyamatát kérdezzük)
  - előnye: biztosan észrevesszük ha a kiszolgálás sérül
  - hátránya: túl kicsi a “hiba felbontása” az okok pontos behatárolásához
  - pontos diagnosztizáláshoz több, nagyobb felbontású metrikára van szükség. Lásd következő pont: Diagnosztika

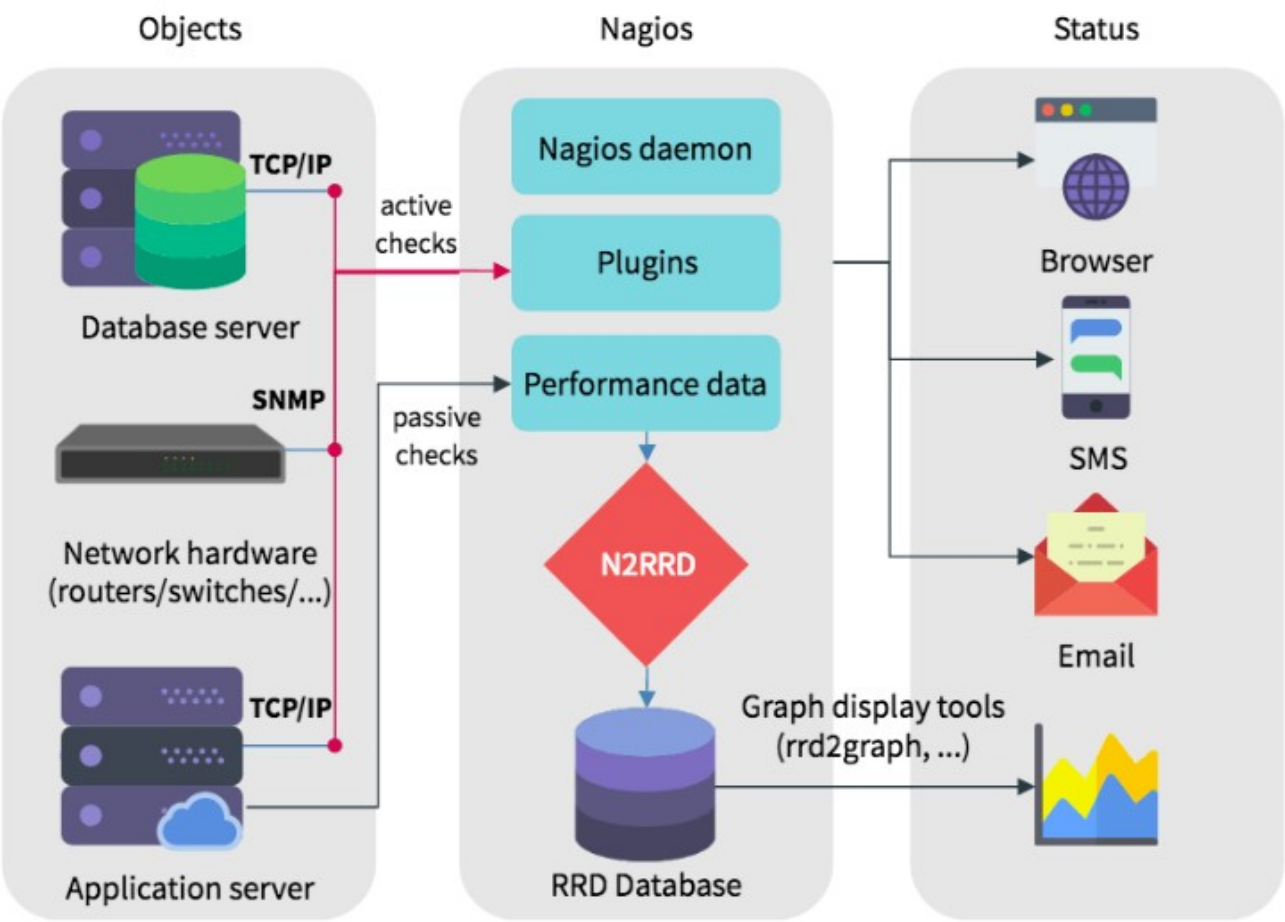
# Diagnosztika

- Detektálás: van-e hiba a rendszerben?
- Lokalizálás: hiba helyének meghatározása
  - milyen mélységig? (metrikák felbontása)
  - van-e ehhez elegendő metrikánk a monitoring rendszerünkben?
  - túl sok metrika is gond lehet!

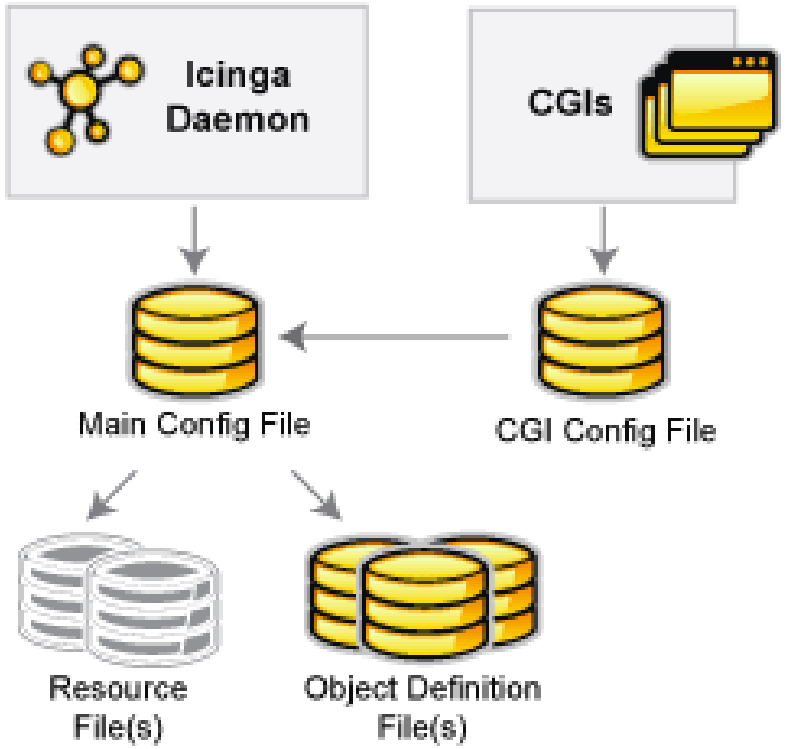
# Monitoring keretrendszerek, implementációk

- Nagios és klónjai (iparági standard)
- OMD / Check_MK
- Zabbix (széleskörben használt)
- ZenOSS
- Munin / Cacti / OpenNMS
- SCOM / Cloud Service Monitoring (Windows)
- Amazon CloudWatch
- Oracle Cloud Control (Enterprise Manager)

# NAGIOS


Objects
Nagios
Status
TCP/IP
Database server
SNMP
Network hardware
(routers/switches/...)
TCP/IP
Application server
active
checks
passive
checks
Nagios daemon
Plugins
Performance data
N2RRD
RRD Database
Graph display tools
(rrd2graph, ...)
Browser
SMS
Email

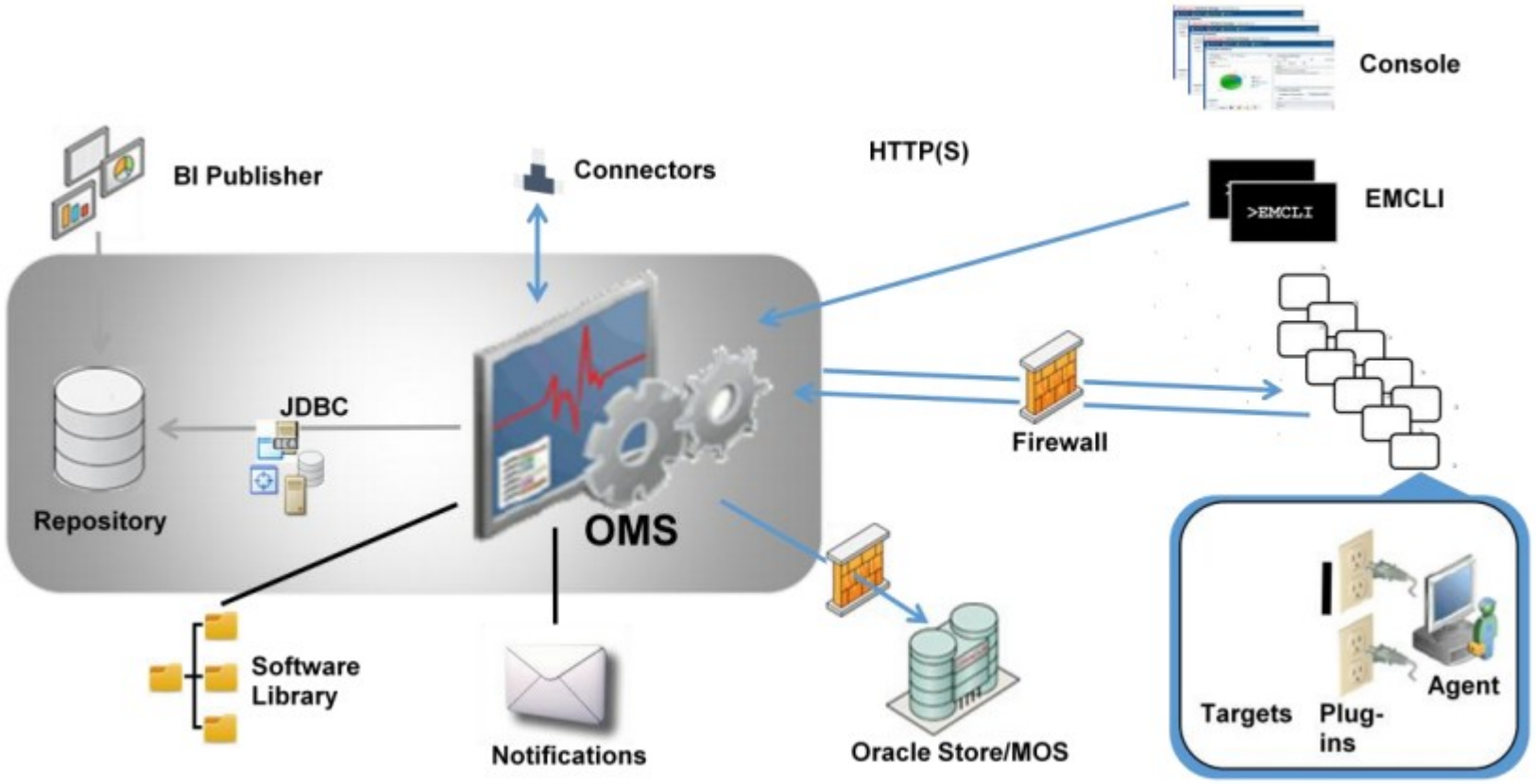
# Nagios/Icinga Config



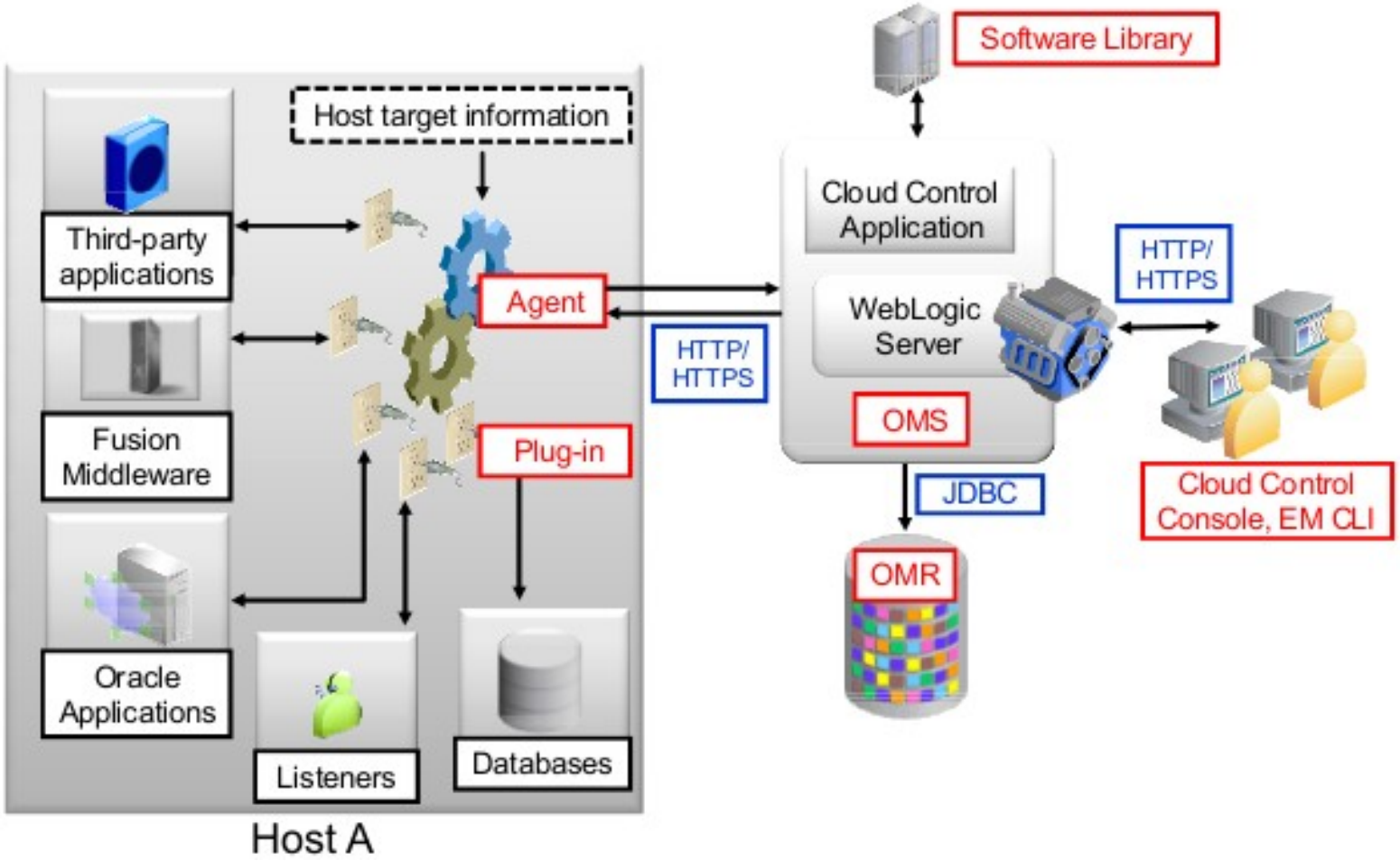
# Nagios/Icinga Web Interfész

- Néhány példa(kép) Nagios-ra
- Icinga demo page

# Oracle Cloud Control Architektúra



# Oracle Cloud Control Komponensek



# Oracle Cloud Control Web Console

- Áttekintés egy élő példán keresztül

PEOPLE COME FIRST
INFORMATIKAI SZAKÉRTŐK EGYESÜLETE

# Data Science and Big Data

Oracle Junior Program - IT technológiák és architektúrák nagyvállalati környezetben

**Szakal Ádám**

# Miről lesz ma szó

- Fogalmak tisztázása
  - DataScience
  - Mesterséges intelligencia
  - Machine learning
  - Deep learning
  - Big Data
- Példák

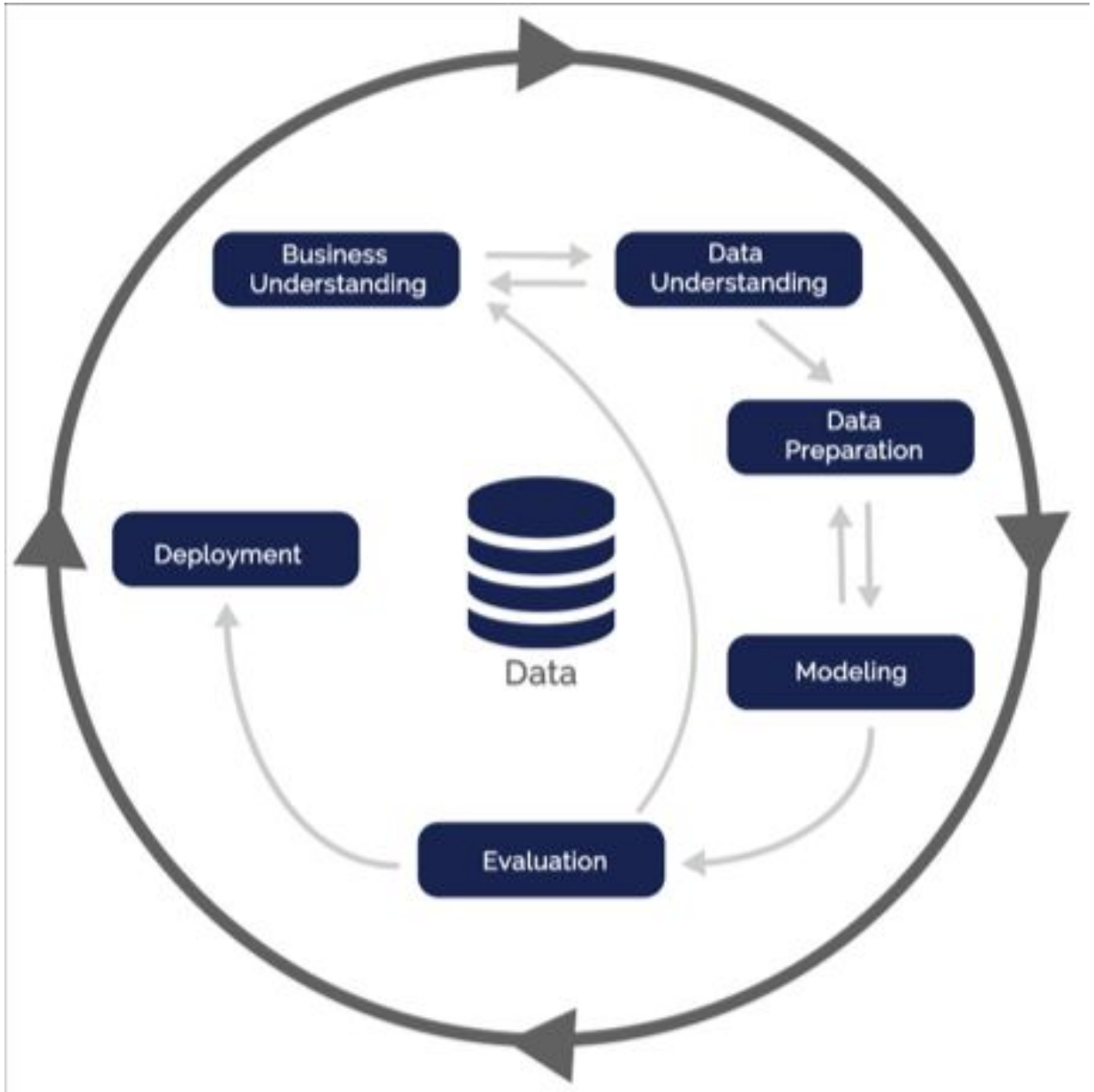
# Definíciók

- “Data science provides meaningful information based on large amounts of complex data or big data. Data science, or data-driven science, combines different fields of work in statistics and computation to interpret data for decision-making purposes.” - Investopedia
- “The use of scientific methods to obtain useful information from computer data, especially large amounts of data” - Cambridge dictionary
- Fancy name for statistics?

# Data Scientist

- 2020-ban is a “legszexibb” foglalkozás
- Statisztika/Matematika
- Programozás
- Domain tudás
- Machine learning algoritmusok ismerete
- Gyors tanulási képesség
- Egyetemeken info-s képzés szakirány

# Mit csinál egy Data Scientist



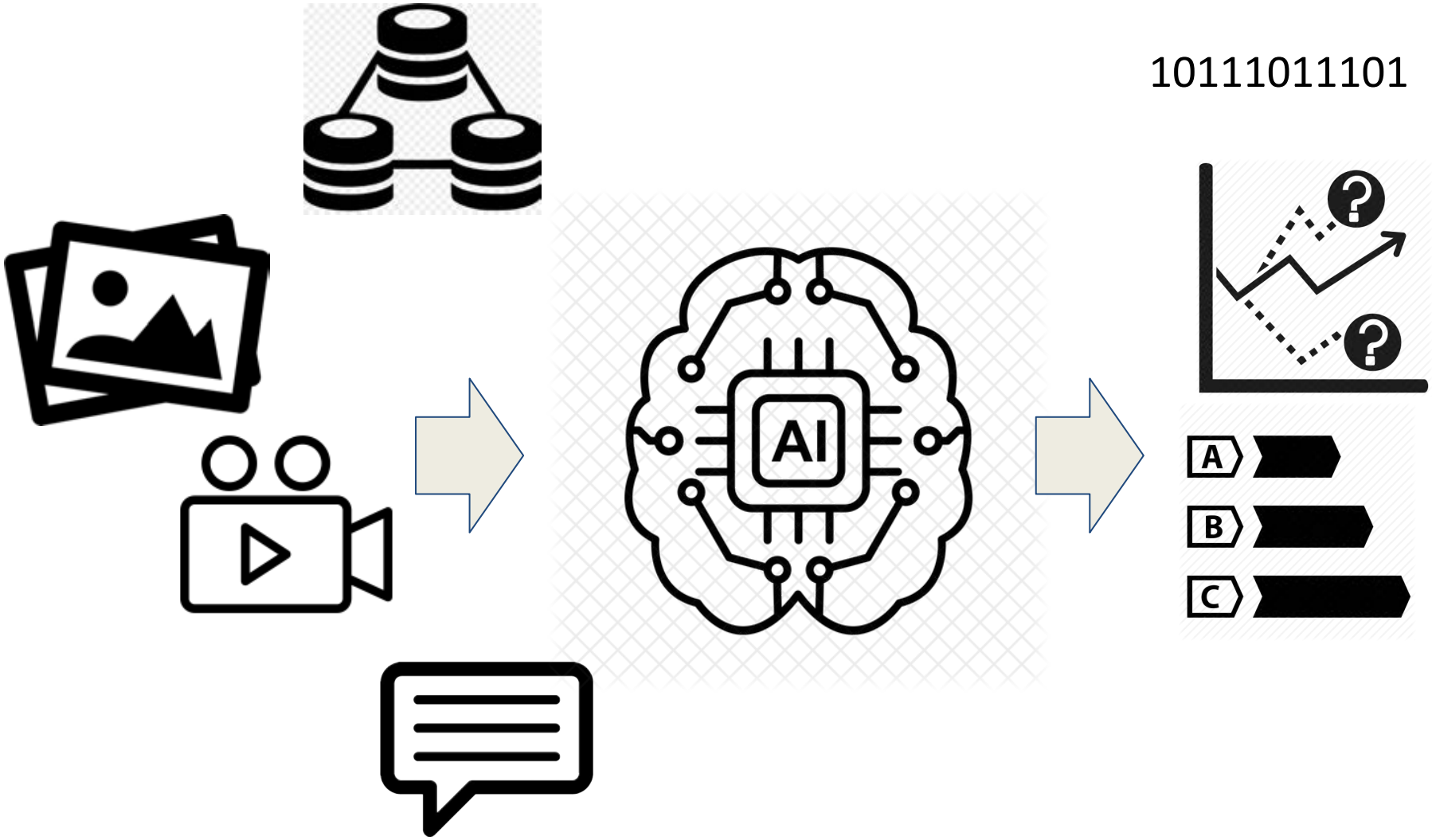
# Nehézségek

- Gyorsan fejlődő tudományág -> Folyamatos tanulást igényel
- Az adattisztítás időigényes és nem túl hálás feladat
- Az eredmények prezentáláshoz jó kommunikációs, tárgyaló és tanító készségek szükségesek
- Leleményesség és önálló munkavégzés
- Nem mindig állít elő kézzelfogható terméket

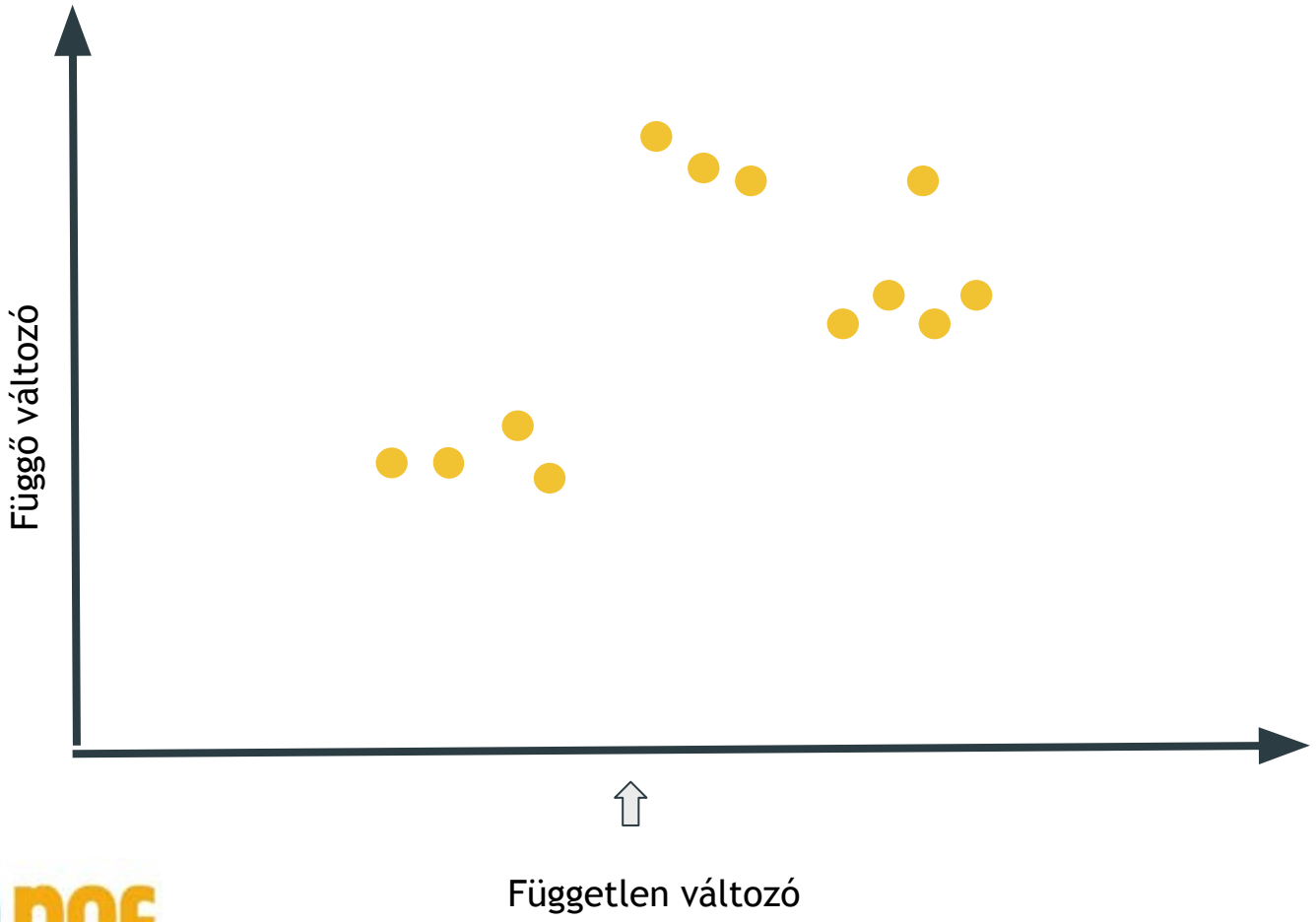
https://towardsdatascience.com/4-reasons-why-you-shouldnt-be-a-data-scientist-e3cc6c1d50e

# Mesterséges intelligencia(AI)

# Machine learning - Gépi tanulás



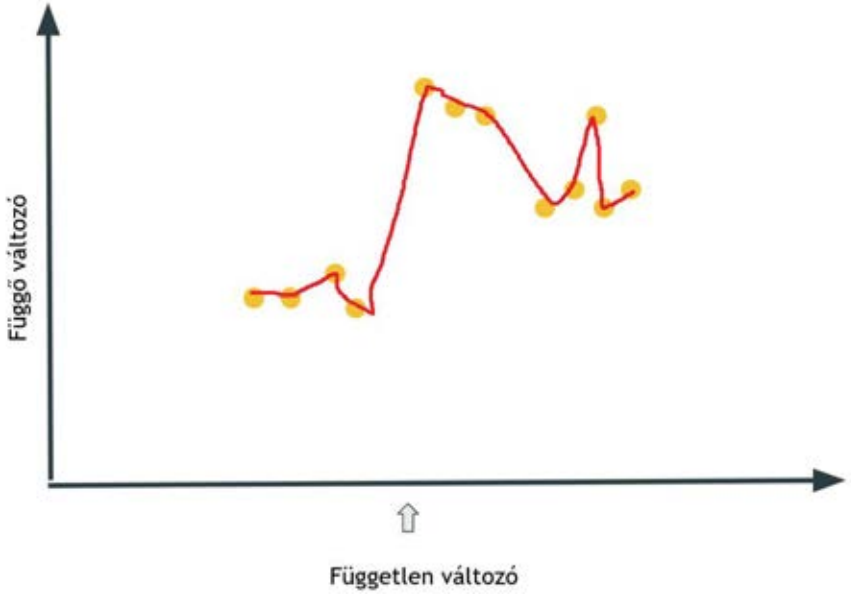
# Machine learning



# Machine learning - Bias and Variance

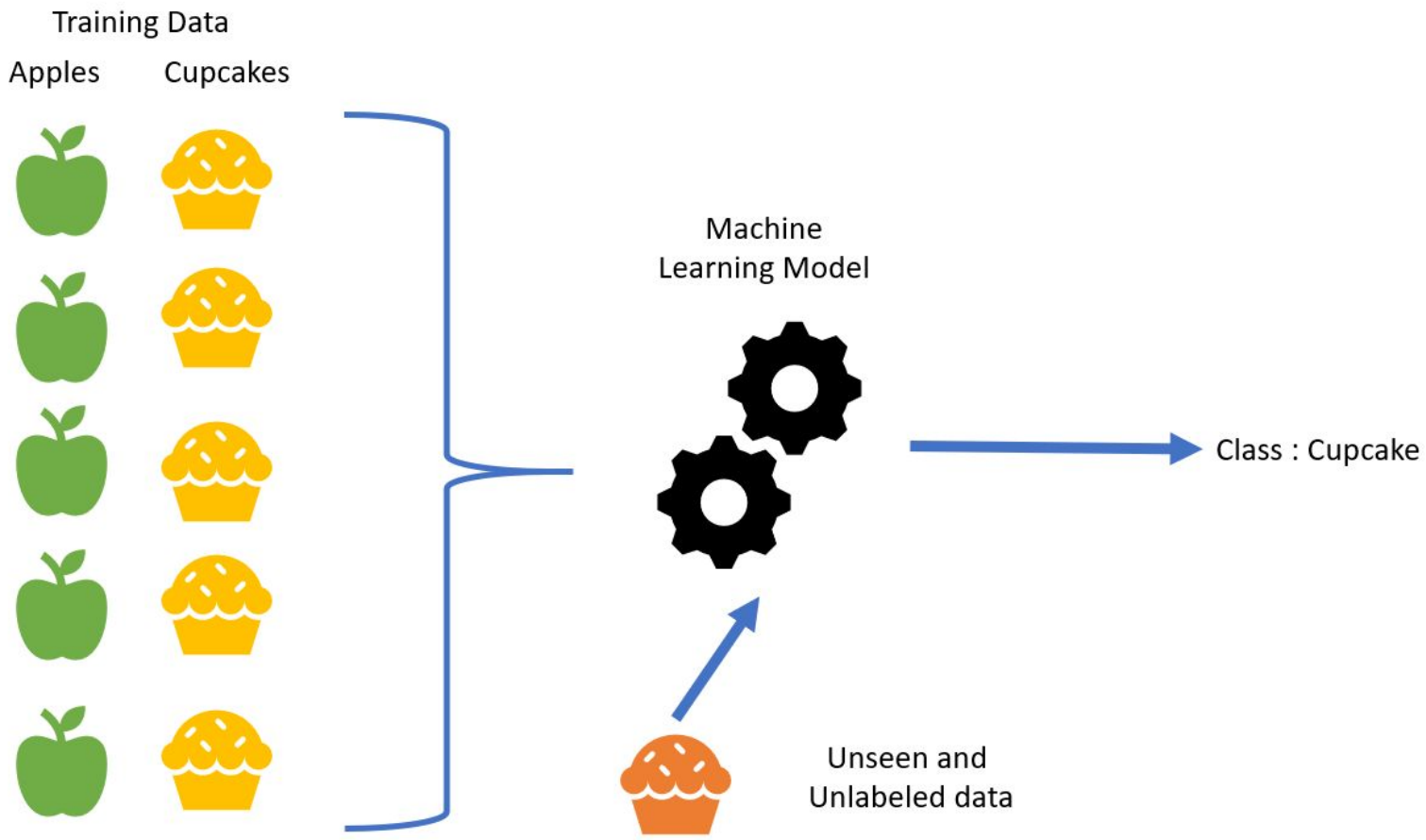
High bias

High Variance

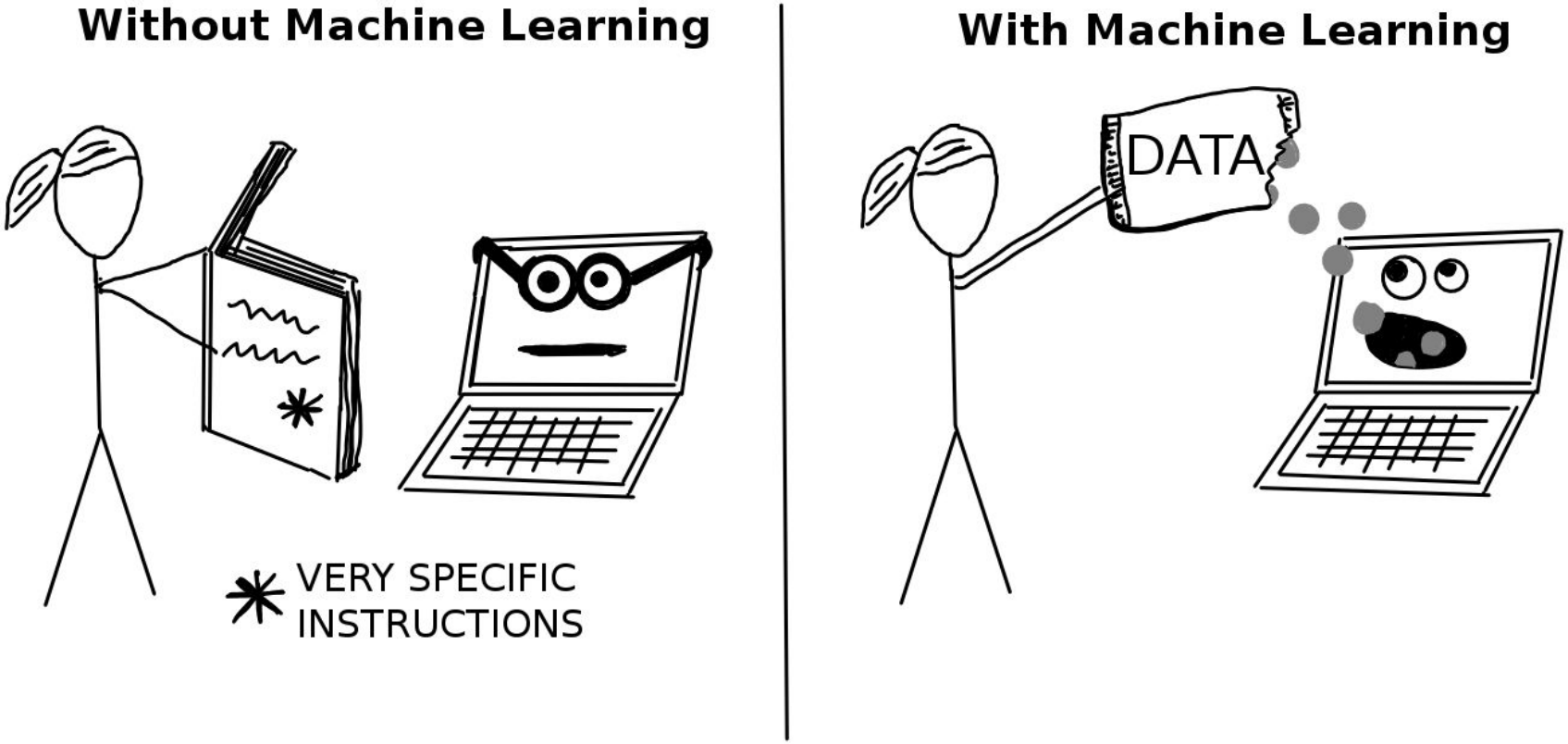


# Machine learning - Minimize error

# Machine learning



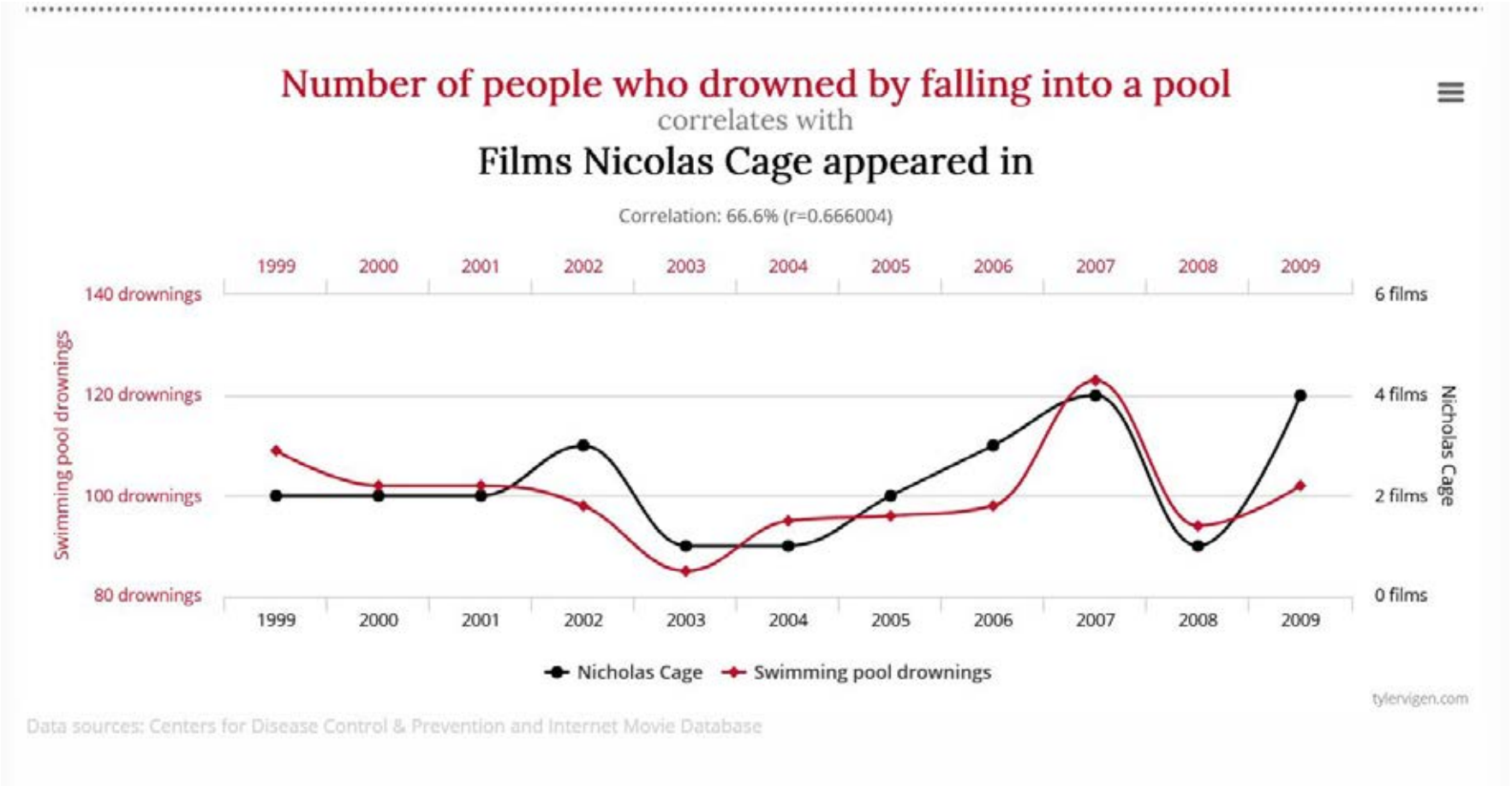
# Machine learning

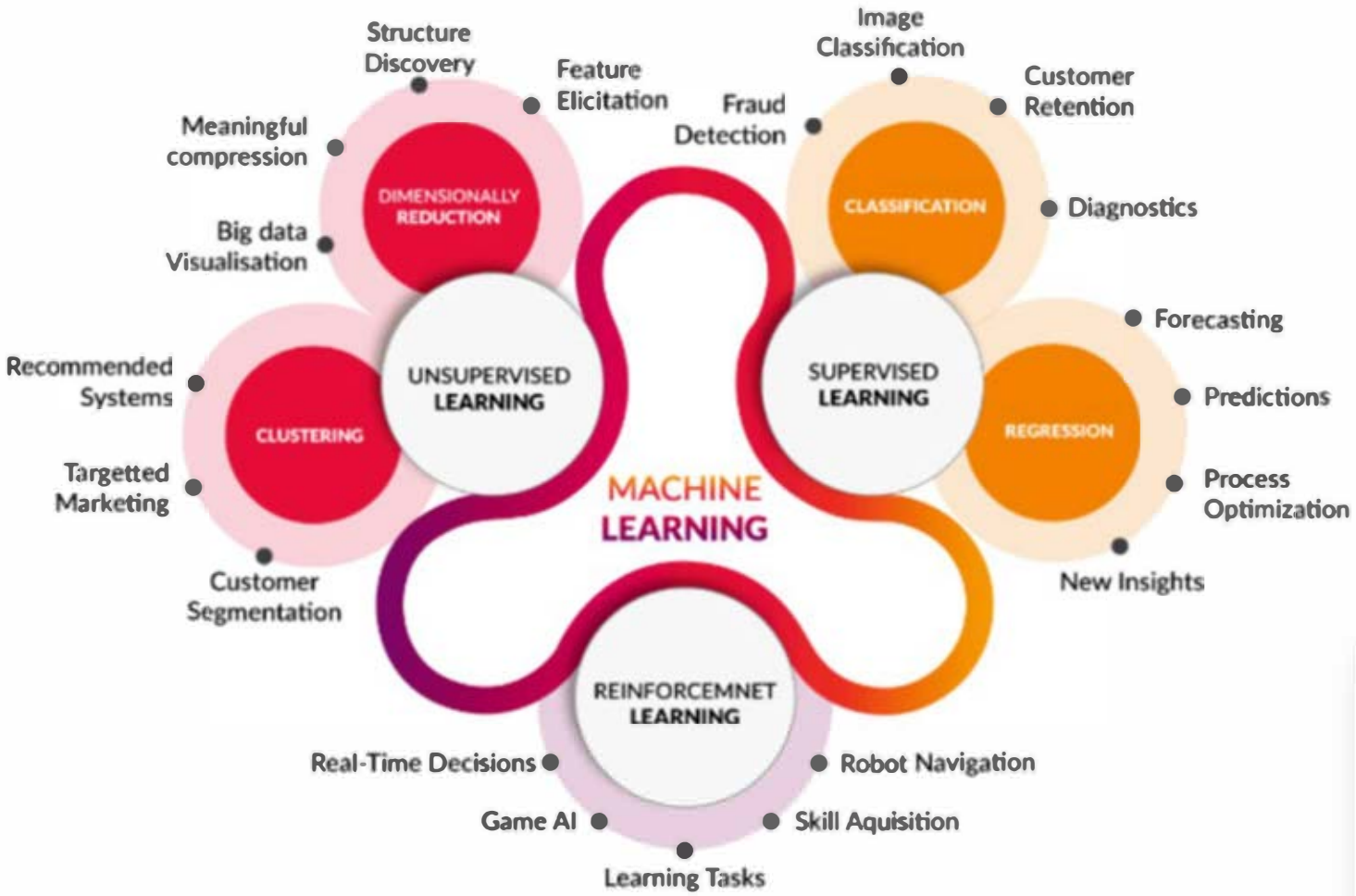


# Data Preparation

- Outlier filtering
- Handling missing values
- Dependency - correlation
- Multiple correlation
- Transform data

# Correlation is not causation

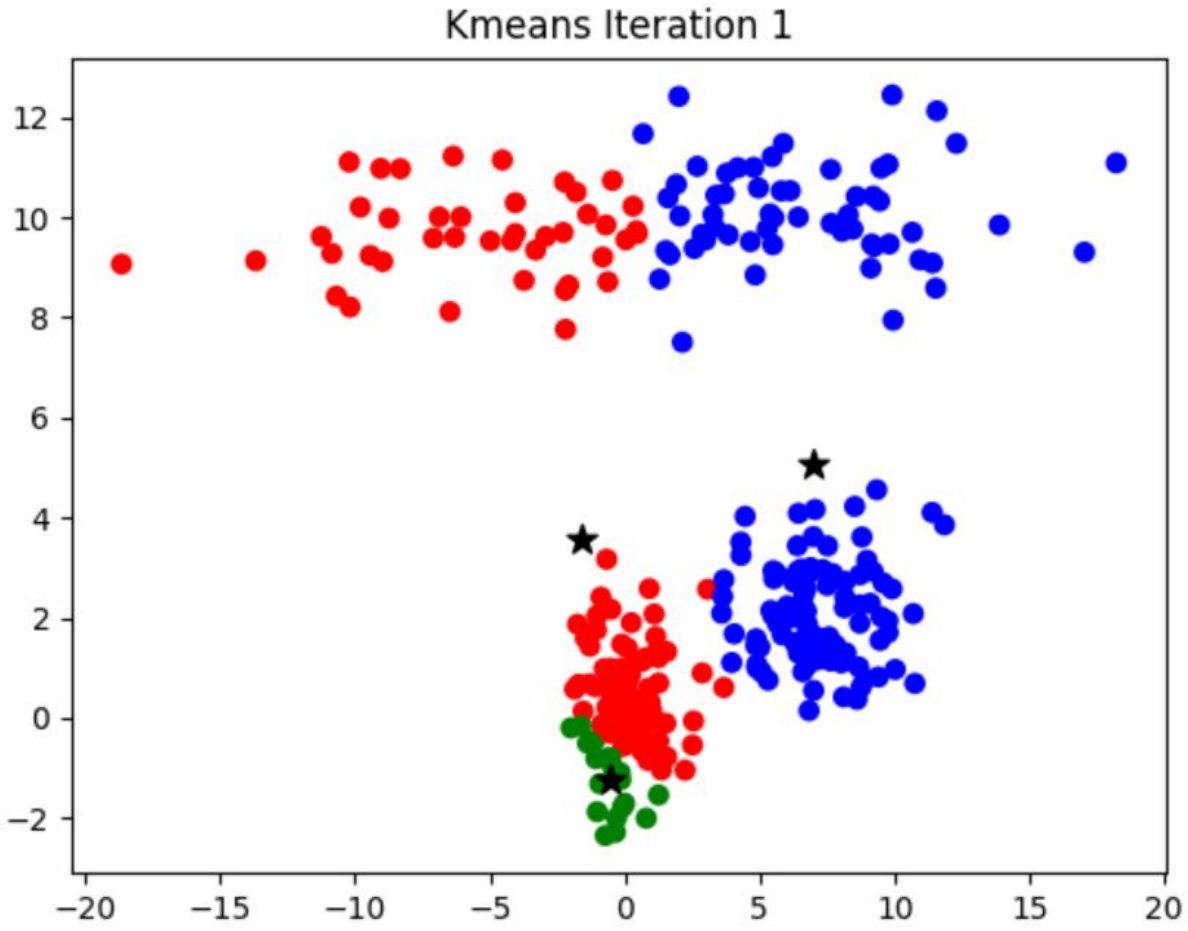



Structure Discovery
Feature Elicitation
Meaningful compression
Big data Visualisation
DIMENSIONALLY REDUCTION
Image Classification
Customer Retention
Fraud Detection
CLASSIFICATION
Diagnostics
Forecasting
Predictions
Process Optimization
New Insights
REGRESSION
Recommended Systems
Targetted Marketing
Customer Segmentation
CLUSTERING
UNSUPERVISED LEARNING
SUPERVISED LEARNING
MACHINE LEARNING
REINFORCEMNET LEARNING
Real-Time Decisions
Robot Navigation
Game AI
Skill Aquisition
Learning Tasks

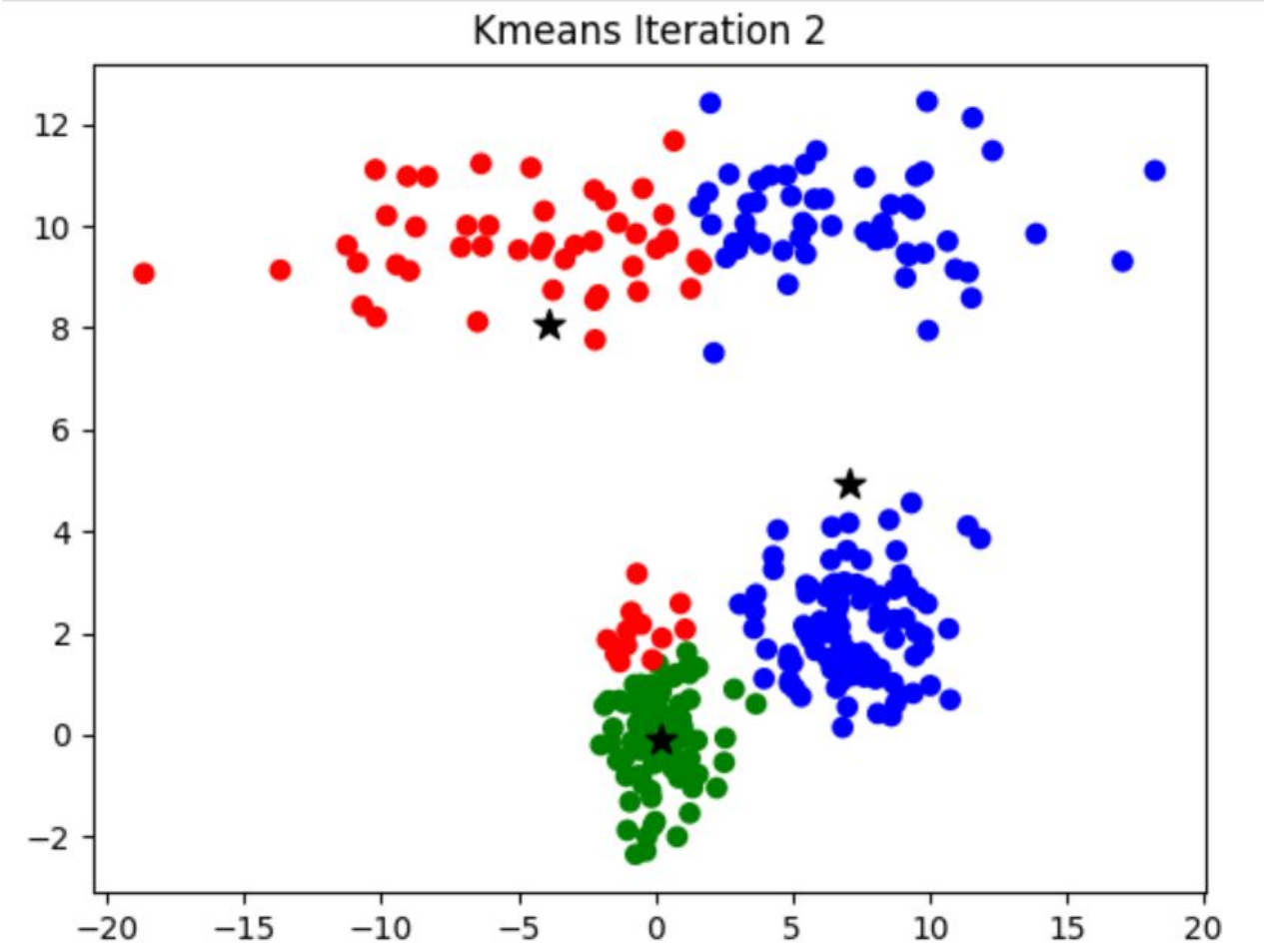
PC

# Reinforcement learning

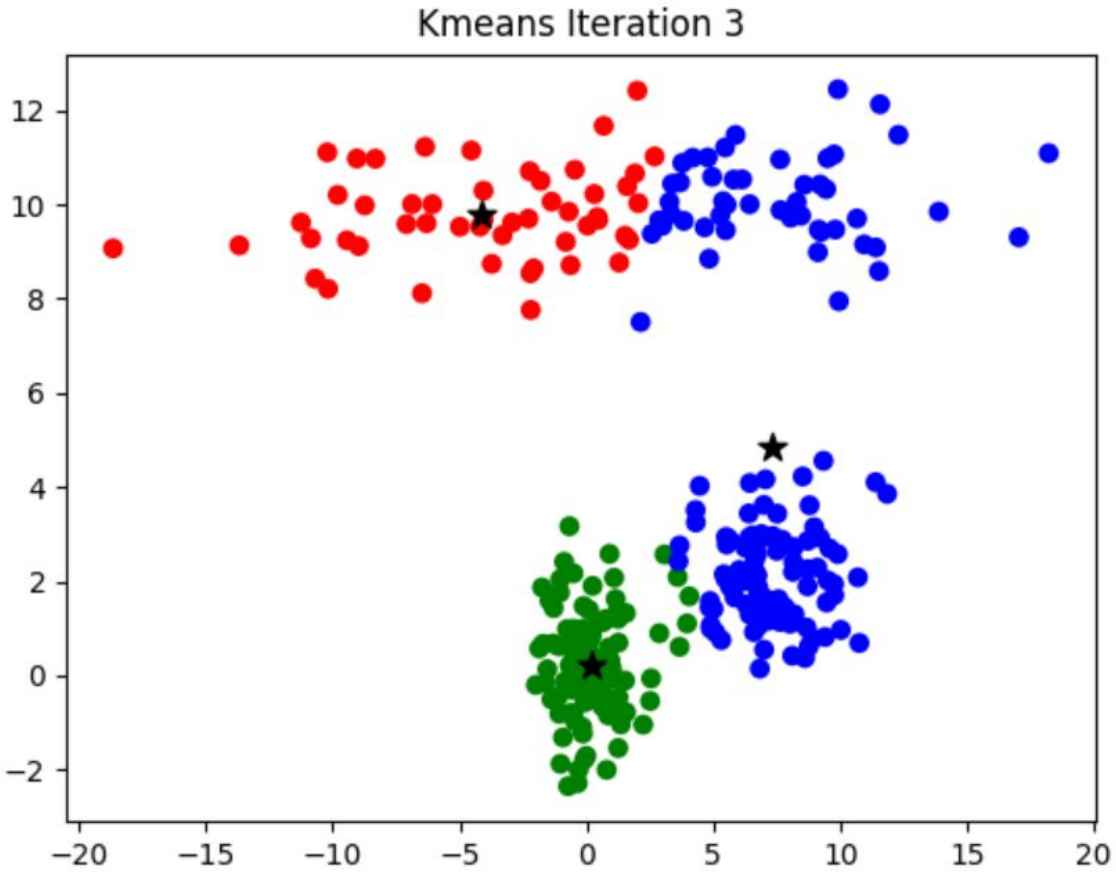
# Clustering - Kmeans



# Clustering Kmeans



# Clustering Kmeans



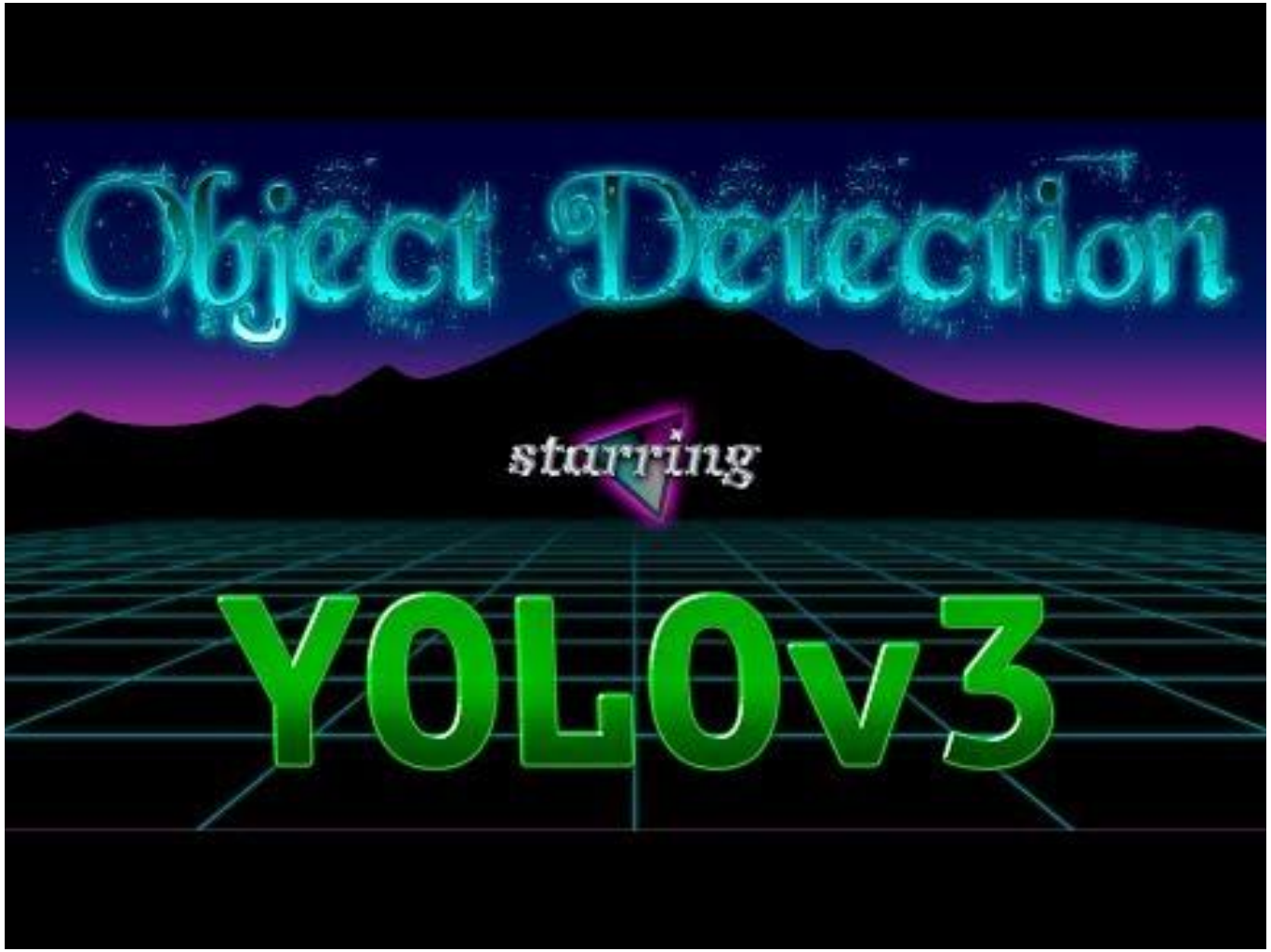
# Clustering Kmeans

Hogyan definiáljuk a távolságot?

Hány klasztert válasszunk?

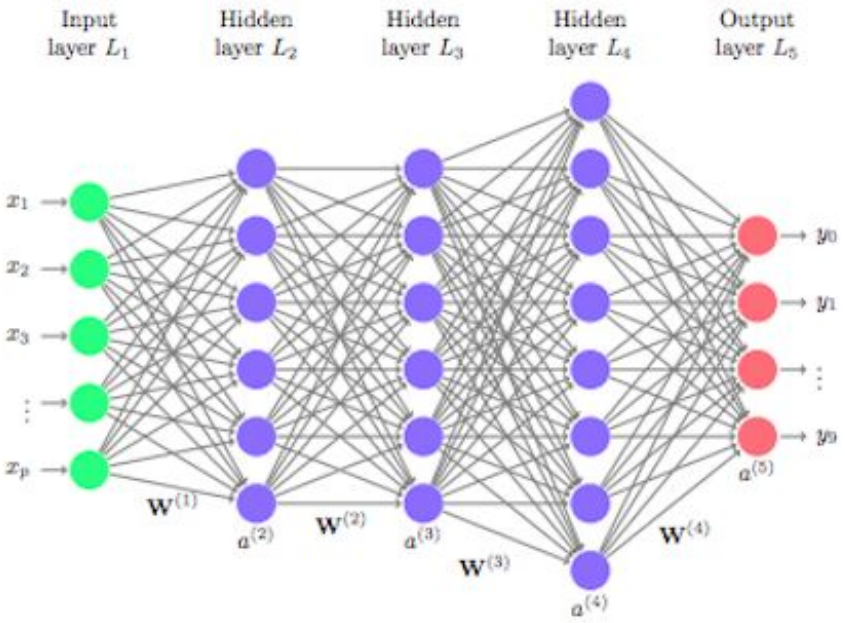
Mi a klaszterek jelentése?

# Classification



# Deep learning

- Fancy név a többszintű neurális hálóra
- Layerek lehetnek különböző típusúak
- Teljesítmény / bemenő adatok összefüggés jobban skálázódik



# Python

- Python is an open source script language
- High-level, general purpose
- Dynamic type system, automatic memory management
- Interpreted
- Fields of application
  - Numeric computation
  - Simulation
  - Data analysis
- It’s popular because:
  - It’s easy to learn and use
  - It’s fast (like really fast)
  - It’s resource-friendly
  - It’s almost limitless (there’s a Python library for almost everything)

# További lehetőségek

- R
- SAS
- Matlab
- Húzogatós programok
- PL/SQL

# Big data

**Egyszerűen megfogalmazva**

- „Olyan adatmennyiség, amitől az Excel már crash-el.”

**Oxford dictionary**

- „data sets that are too large and complex to manipulate or interrogate with standard methods or tools”

**Dictionary.com**

- „data sets, typically consisting of billions or trillions of records, that are so vast and complex that they require new and powerful computational resources to process

**O’Reilly Media (M. Loukides)**

- „As storage capacity continues to expand, today’s “big” is certainly tomorrow’s “medium” and next week’s “small.” The most meaningful definition I’ve heard: “big data” is when the size of the data itself becomes part of the problem.”

# Big data

**Gartner definíció (D. Laney):**

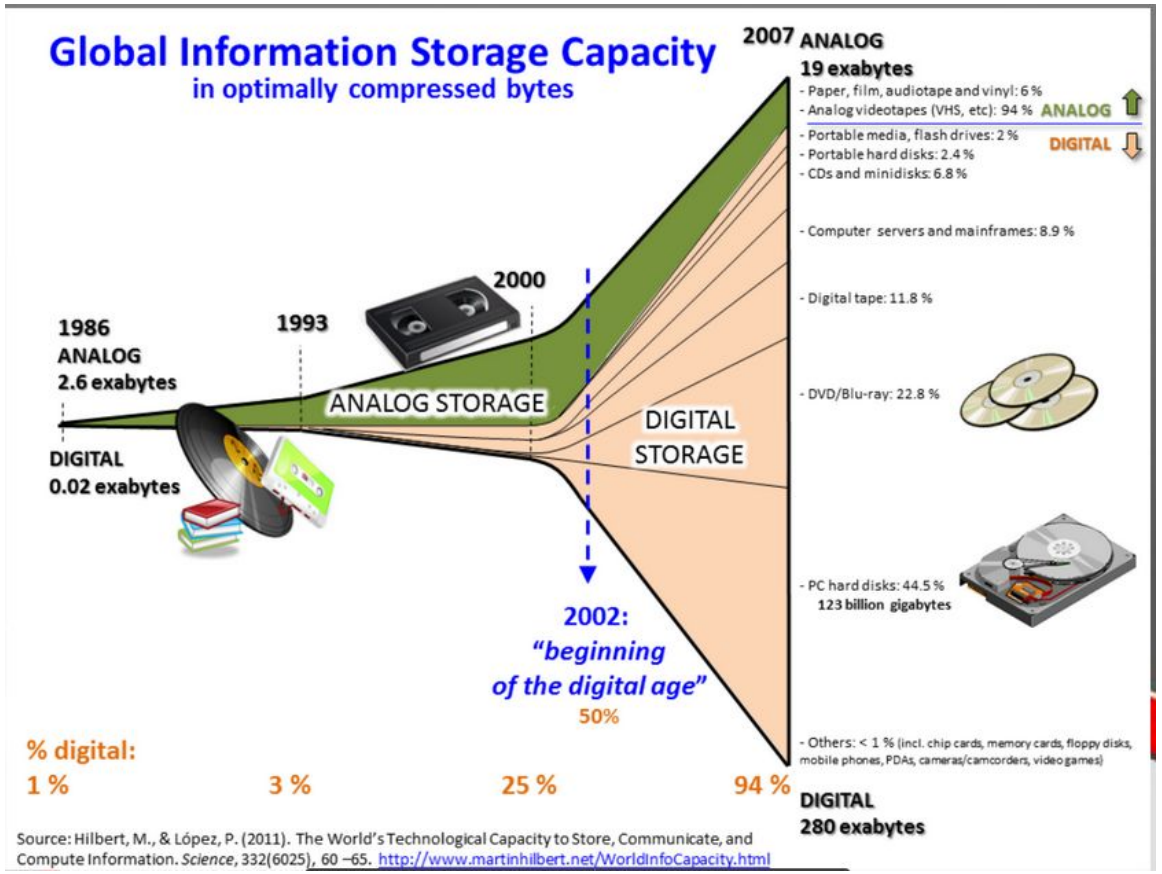
- „Big data are high **volume**, high **velocity**, and/or high **variety** information assets that require new forms of processing to enable enhanced decision making, insight discovery and process optimization.”

**Fentit szokás 3-10V-nek is nevezni:**

- Volume – méret
- Velocity – adatsebesség
- Variety – sokféle típusú adat
- …
  - Veracity – igazságtartalom, tisztaság; Validity – helyesség; Variability – egyre rugalmasabb struktúrák; Value – nagy értékű; Visualization – vizualizálhatóság; …

# Big data - Volume

- The capacity to store information doubles roughly every 3.5 years
- Roughly 2,5 exabytes of data are generated every day



# Big data - Variety

- Different data types are used: text, pictures,videos
- The capability to process important information is crucial
- Data come from different sources

# Big data Velocity

- So much so that the MetLife executive stressed that: “Velocity can be more important than volume because it can give us a bigger competitive advantage. Sometimes it’s better to have limited data in real time than lots of data at a low speed.”
- Standard relational databases are not capable of handling this amount of data
- Parallel computation
- Exmaples?

# Big data - Veracity

- Data veracity, in general, is how accurate or truthful a data set may be
- The actual data makes sense based on business needs
- Data source trustworthy
- The only measure that decreases over time

# Map reduce

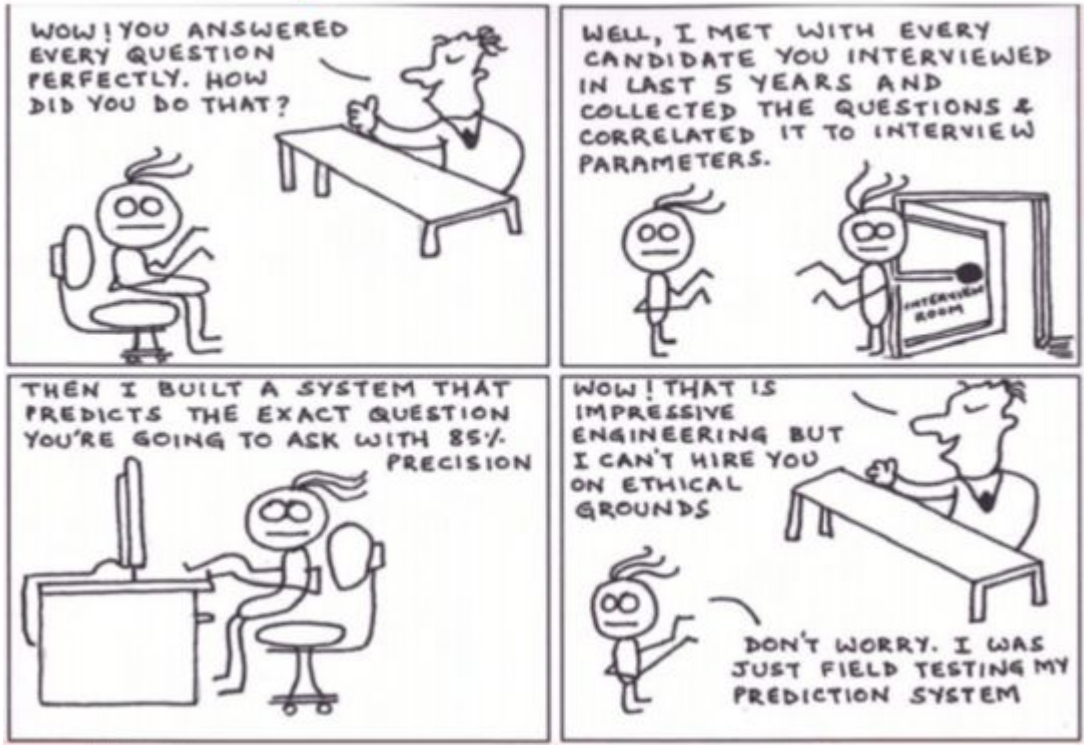
# Data Science akcióban

NETFLIX

Magyar Posta

PEOPLE COME FIRST
INFORMATIKAI SZAKÉRTŐK EGYESÜLETE

## When you interview a data scientist...




# Köszönöm a figyelmet!

PEOPLE COME FIRST
INFORMATIKAI SZAKÉRTŐK EGYESÜLETE

# Big data/NoSQL

Oracle Junior Program - IT technológiák és architektúrák nagyvállalati környezetben

**Szakal Ádám**
**(Webváltó)**

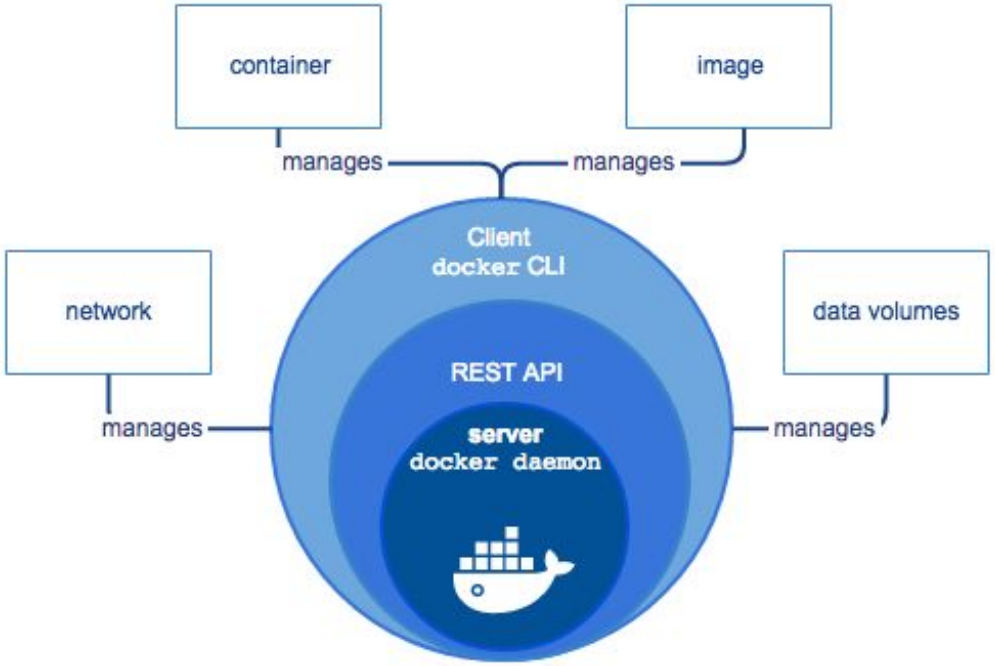
# Előző óráról

High bias

High Variance

# Kitekintés Docker

- NoSQL-Big data technológiák kipróbálásához



# Big data

**Egyszerűen megfogalmazva**

- „Olyan adatmennyiség, amitől az Excel már crash-el.”

**Oxford dictionary**

- „data sets that are too large and complex to manipulate or interrogate with standard methods or tools”

**Dictionary.com**

- „data sets, typically consisting of billions or trillions of records, that are so vast and complex that they require new and powerful computational resources to process

**O’Reilly Media (M. Loukides)**

- „As storage capacity continues to expand, today’s “big” is certainly tomorrow’s “medium” and next week’s “small.” The most meaningful definition I’ve heard: “big data” is when the size of the data itself becomes part of the problem.”

# Big data

**Gartner definíció (D. Laney):**

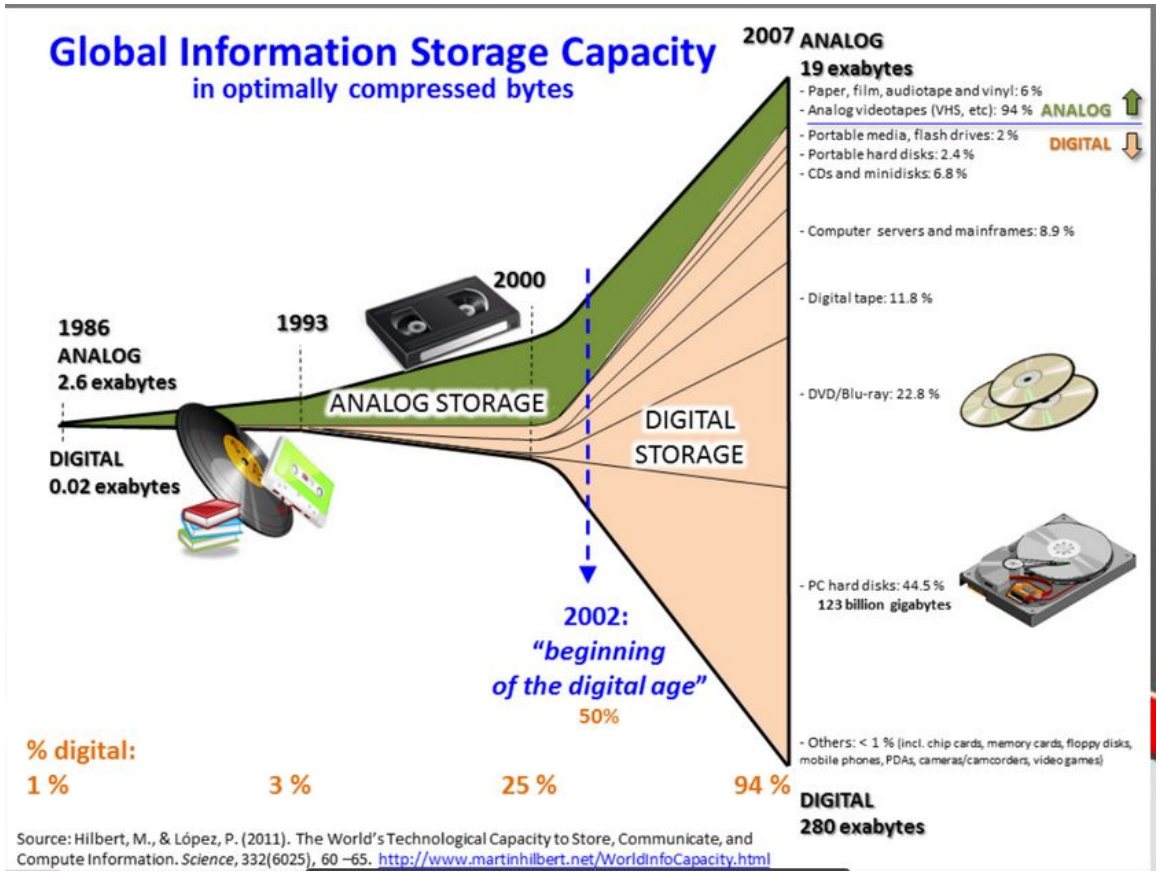
- „Big data are high **volume**, high **velocity**, and/or high **variety** information assets that require new forms of processing to enable enhanced decision making, insight discovery and process optimization.”

**Fentit szokás 3-10V-nek is nevezni:**

- Volume – méret
- Velocity – adatsebesség
- Variety – sokféle típusú adat
- …
  - Veracity – igazságtartalom, tisztaság; Validity – helyesség; Variability – egyre rugalmasabb struktúrák; Value – nagy értékű; Visualization – vizualizálhatóság; …

# Big data - Volume

- The capacity to store information doubles roughly every 3.5 years
- Roughly 2,5 exabytes of data are generated every day



# Big data Velocity

- So much so that the MetLife executive stressed that: “Velocity can be more important than volume because it can give us a bigger competitive advantage. Sometimes it’s better to have limited data in real time than lots of data at a low speed.”
- Standard relational databases are not capable of handling this amount of data
- Parallel computation

# Big data - Variety

- Different data types are used: text, pictures,videos
- The capability to process important information is crucial
- Data come from different sources

# Big data - Veracity

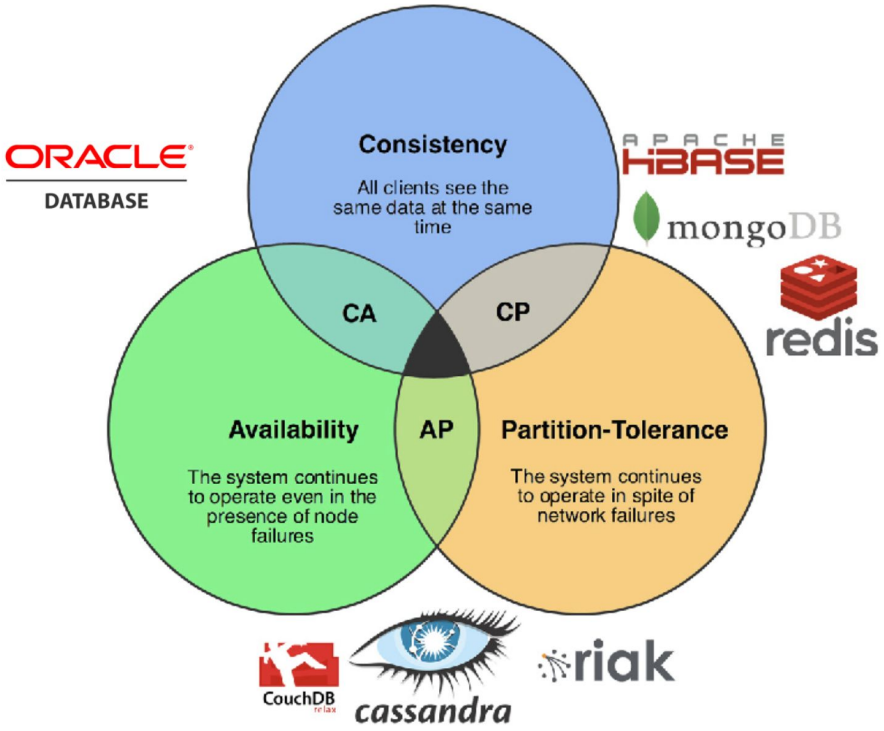
- Data source trustworthy
- The actual data makes sense based on business needs
- The only measure that decreases over time

# Distributed Data Storing and Processing

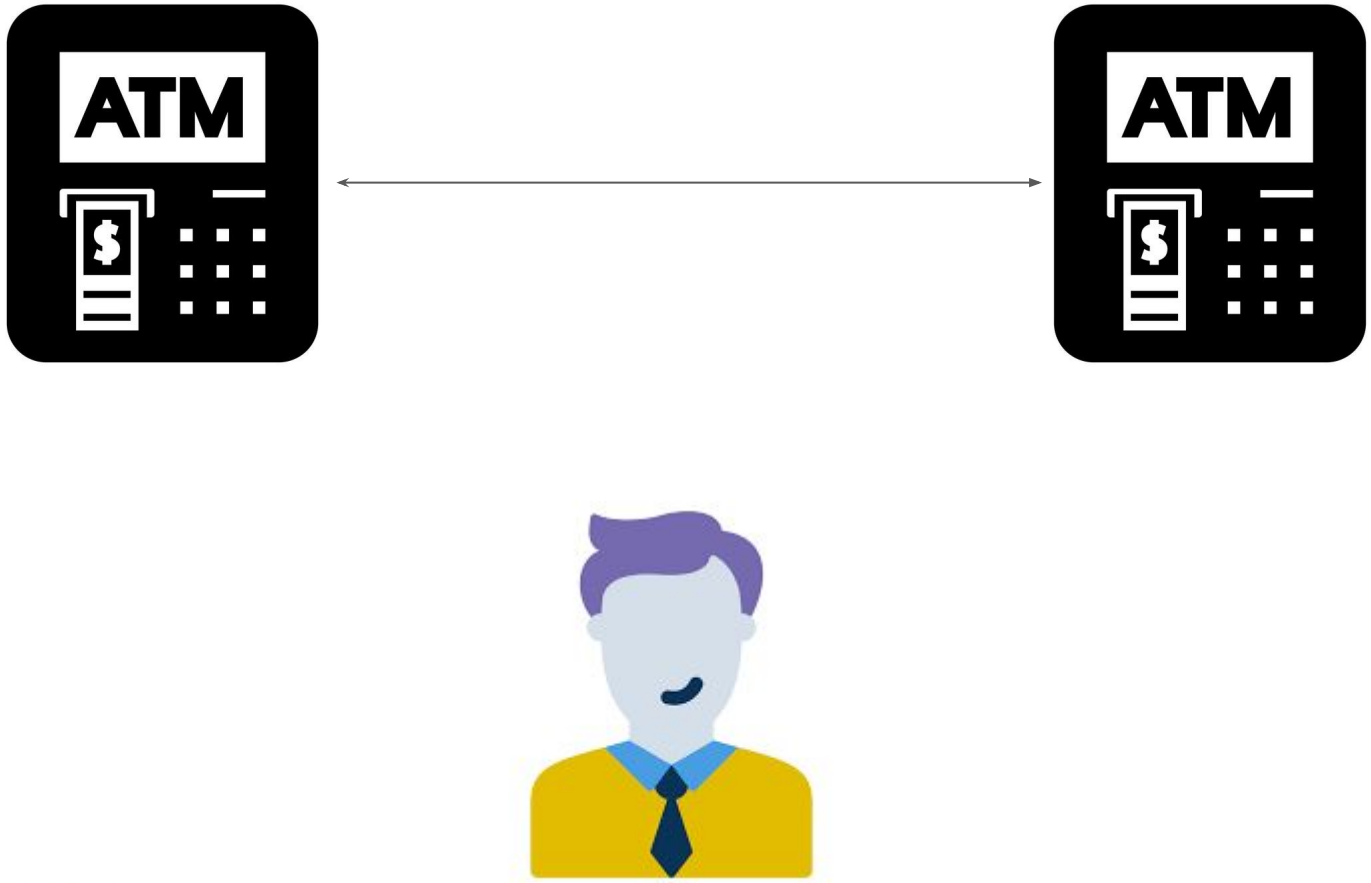
- CAP Theorem
- NoSQL database structures
- Distributed file systems
- Hadoop
  - HDFS
  - MapReduce

# CAP Theorem

Impossible for a distributed data store to simultaneously provide more than two out of the following three guarantees.



# CAP Theorem



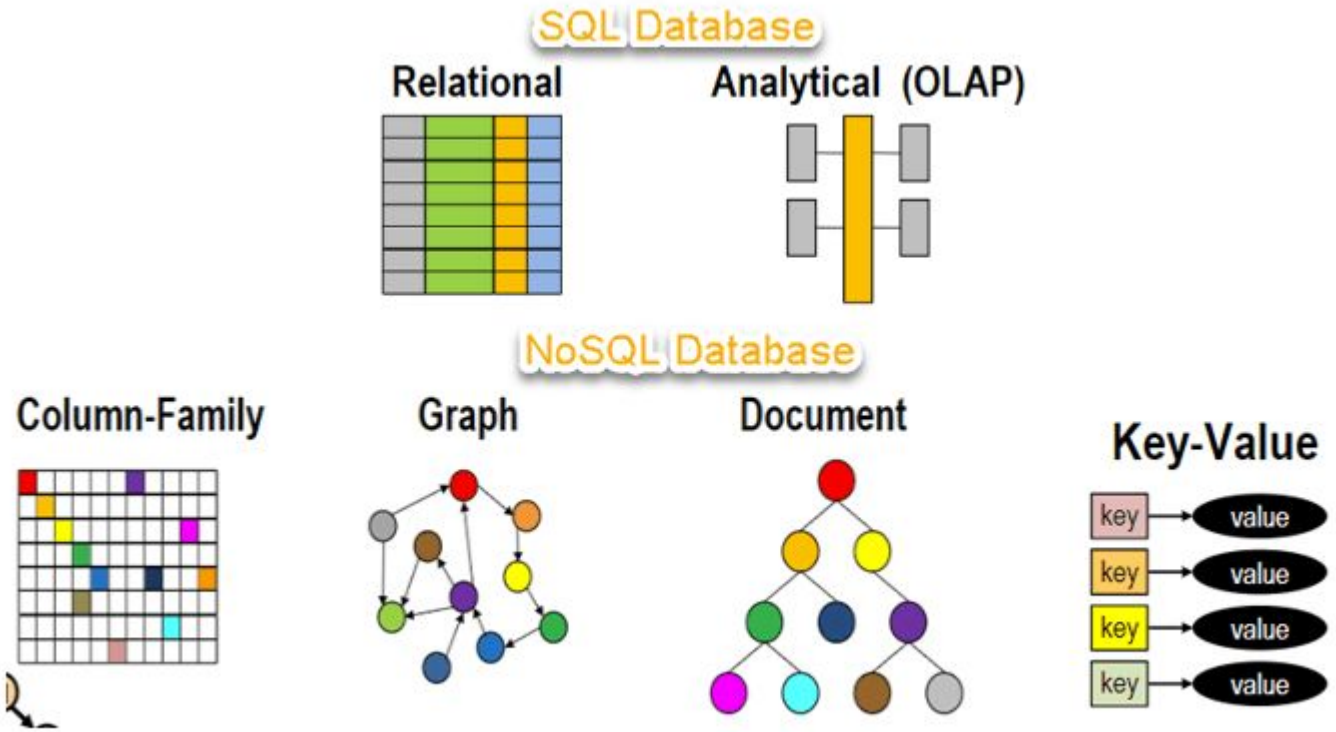
# Distributed data storing(BASE)

Eventually-consistent services are often classified as providing BASE (Basically Available, Soft state, Eventual consistency) semantics, in contrast to traditional ACID (Atomicity, Consistency, Isolation, Durability) guarantees. In chemistry BASE is opposite to ACID, which helps remembering the acronym. According to the same resource, these are the rough definitions of each term in BASE.

# Distributed data storing(BASE)

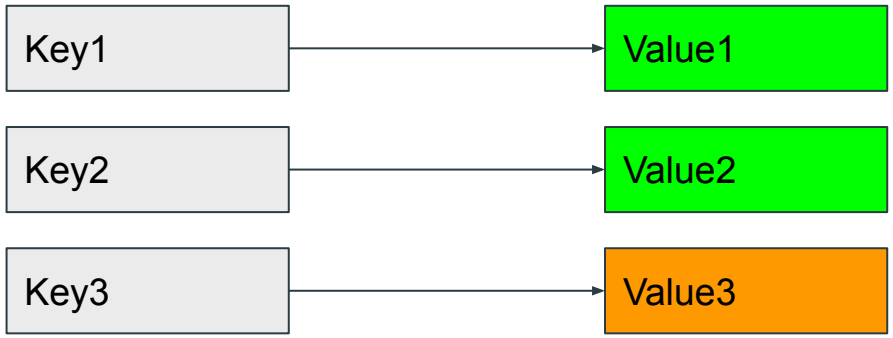
- (B)asically (A)vailable: basic reading and writing operations are available as much as possible (using all nodes of a database cluster), but without any kind of consistency guarantees (the write may not persist after conflicts are reconciled, the read may not get the latest write)
- (S)oft state: without consistency guarantees, after some amount of time, we only have some probability of knowing the state, since it may not yet have converged
- (E)ventually consistent: If the system is functioning and we wait long enough after any given set of inputs, we will eventually be able to know what the state of the database is, and so any further reads will be consistent with our expectations

# NoSQL


SQL Database
Relational
Analytical (OLAP)
NoSQL Database
Column-Family
Graph
Document
Key-Value
key
value
key
value
key
value
key
value

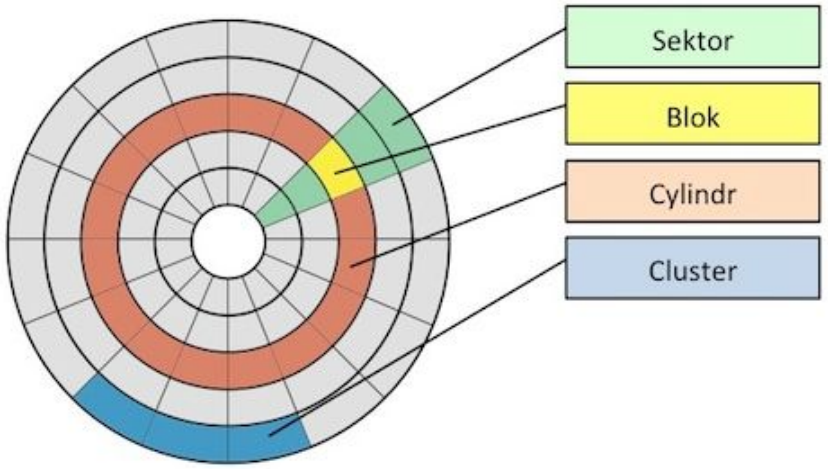
# Key-value database

- Key based search
- Hash function
- Value
  - Various type of objects
  - Don’t search in value
- Redis, Dynamo, Riak, Oracle NoSQL database



# Column oriented database

- Stores data tables by column rather than by row
- SQL syntax
- Data compression
- Fast column functions: SUM, COUNT, AVG, MIN



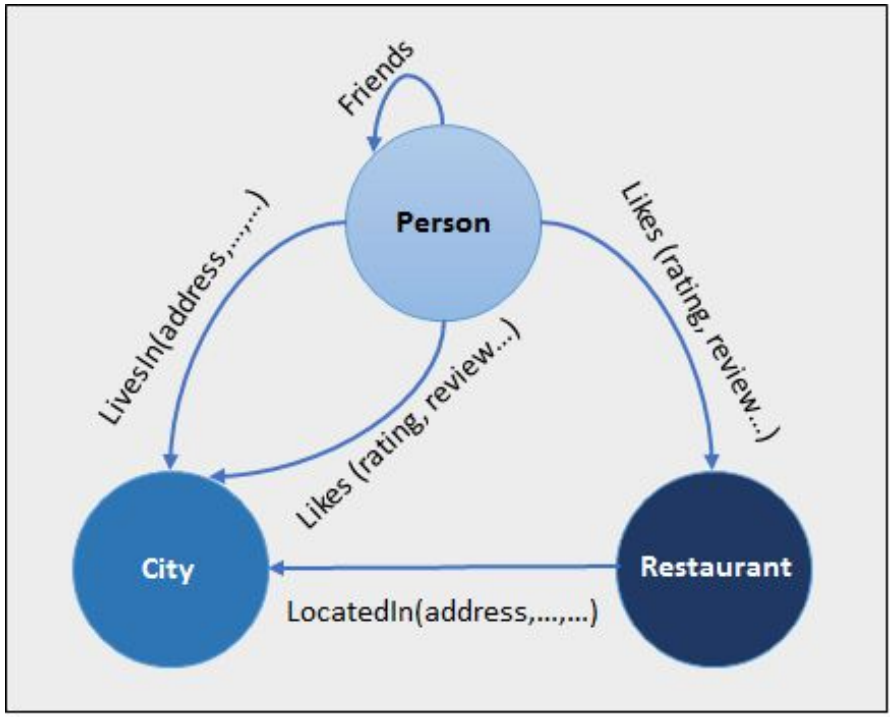
# Column family database

- Loose schema and various data types
- Google Bigtable, Apache Cassandra

| CustID | CustomerDetails | | | | | | | Relatives | | Accounts | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Name | Nickname | Address | Email | Mobile | Fax | Home | Wife | Sister | Checking | Savings | Business Checking |
| 101 | Adam | A-Man | 123 Main | Adam@email.com | 555-555-1234 | 555-555-2222 | 555-234-5325 | Debby | Kim | $1,500 | $25,000 | $8,250 |
| CustID | CustomerDetails | | | | | | | Relatives | | Accounts | | |
| | Name | Address | Email | Mobile | | | | | | Checking | | |
| 102 | Bob | 12 East St. | Bob@email.com | 555-562-1234 | | | | | | $250 | | |
| CustID | CustomerDetails | | | | | | | Relatives | | Accounts | | |
| | Name | Nickname | Address | Email | Home | | | Father | Mother | Savings | | |
| 103 | Christopher | Chris | 504 Rogers Road | Chris@email.com | 555-232-3332 | | | Thomas | Casey | $2,000 | | |

# Graph database

- Node: Data
- Edge: stored relation
- Fast join!
- Neo4J, Infinite Graph, OrientDB, FlockDB



# Documentum oriented database

- Like key-value database
- Document is organized structure (JSON)
- Amazon SimpleDB, CouchDB, MongoDB, Riak, Lotus Notes, MongoDB,

# NoSQL Advantages

- Can be used as Primary or Analytic Data Source
- **No Single Point of Failure**
- **Easy Replication**
- No Need for Separate Caching Layer
- It provides fast performance and horizontal scalability.
- Can handle structured, semi-structured, and unstructured data with equal effect
- Object-oriented programming which is easy to use and flexible
- **NoSQL databases don't need a dedicated high-performance server**
- Support Key Developer Languages and Platforms
- Simple to implement than using RDBMS
- It can serve as the primary data source for online applications.
- **Handles big data which manages data velocity, variety, volume, and complexity**
- Excels at distributed database and multi-data center operations
- Eliminates the need for a specific caching layer to store data
- Offers a flexible schema design which can easily be altered without downtime or service disruption

# NoSQL Disadvantages

- **No standardization rules**
- **Limited query capabilities**
- RDBMS databases and tools are comparatively mature
- It does not offer any traditional database capabilities, like **consistency** when multiple transactions are performed simultaneously.
- When the volume of data increases it is difficult to maintain unique values as keys become difficult
- Doesn't work as well with relational data
- The learning curve is stiff for new developers
- **Open source options so not so popular for enterprises.**

# Distributed file systems

- Uniform interface
  - Naming conventions
  - Mapping scheme
  - Can be reached like local files (mounting, unmounting)
  - Permission handling
- Clients should have a same view
- Authorization
- Goal maximization of local reads

# Distributed file systems

- Access transparency: clients are unaware that files are distributed and can access them in the same way as local files are accessed.
- Location transparency: a consistent namespace exists encompassing local as well as remote files. The name of a file does not give its location.
- Concurrency transparency: all clients have the same view of the state of the file system. This means that if one process is modifying a file, any other processes on the same system or remote systems that are accessing the files will see the modifications in a coherent manner.
- Failure transparency: the client and client programs should operate correctly after a server failure.
- Heterogeneity: file service should be provided across different hardware and operating system platforms.

# Distributed file systems

- Scalability: the file system should work well in small environments (1 machine, a dozen machines) and also scale gracefully to bigger ones (hundreds through tens of thousands of systems).
- Replication transparency: Clients should be unaware of the file replication performed across multiple servers to support scalability.
- Migration transparency: files should be able to move between different servers without the client's knowledge.

# Hadoop Distributed file systems (HDFS)

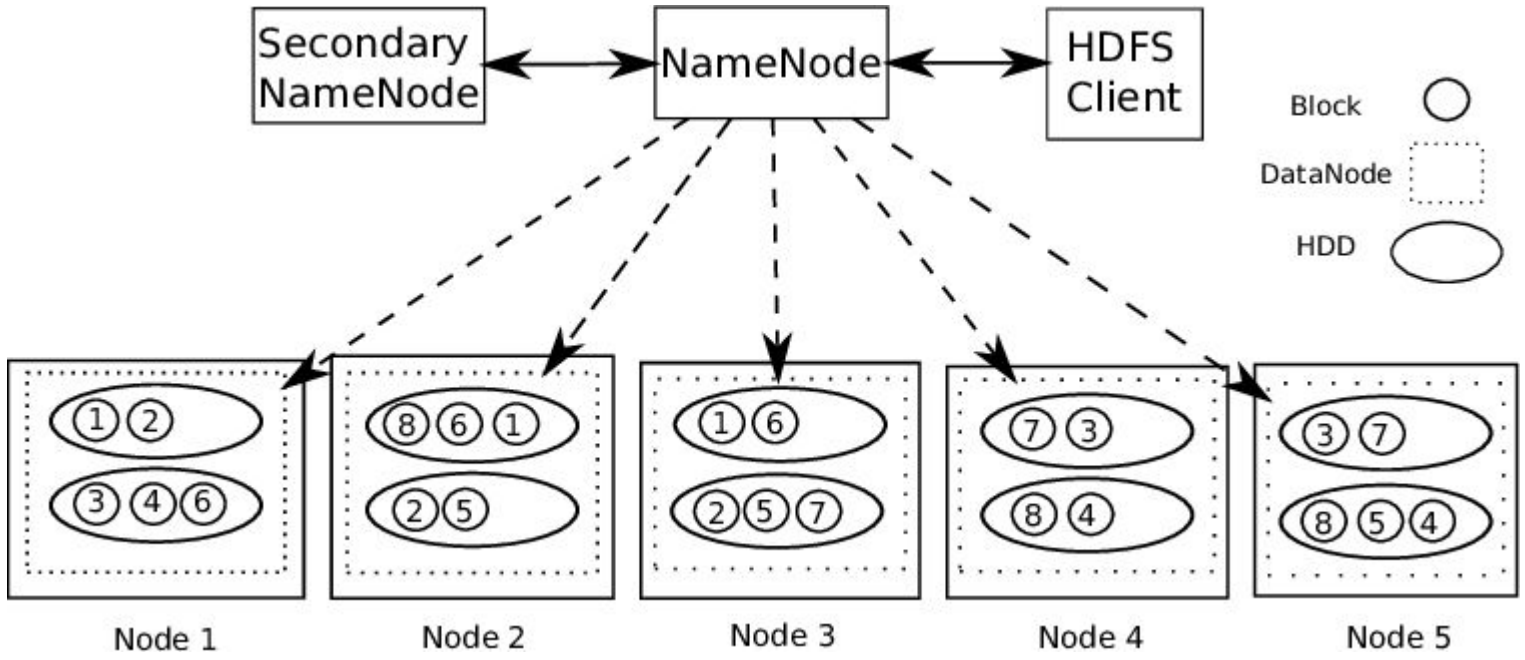
Written in JAVA

Lack of POXIS(Portable Operating System Interface) interface

Shell commands/JAVA API

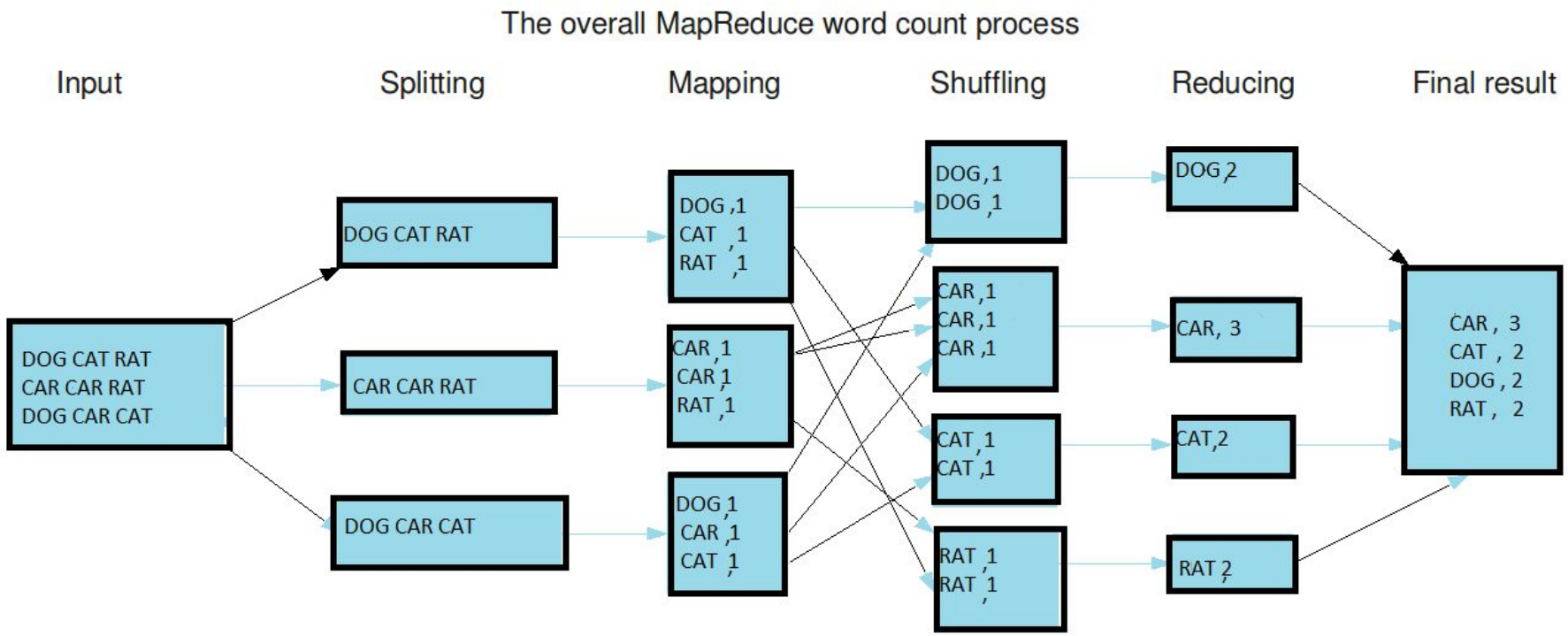
- Master elements
  - Name Node
  - Secondary Name Node
  - Job tracker
- Slave elements
  - Data Node
  - Task Tracker

# Hadoop Distributed file systems (HDFS)



# Hadoop mapreduce

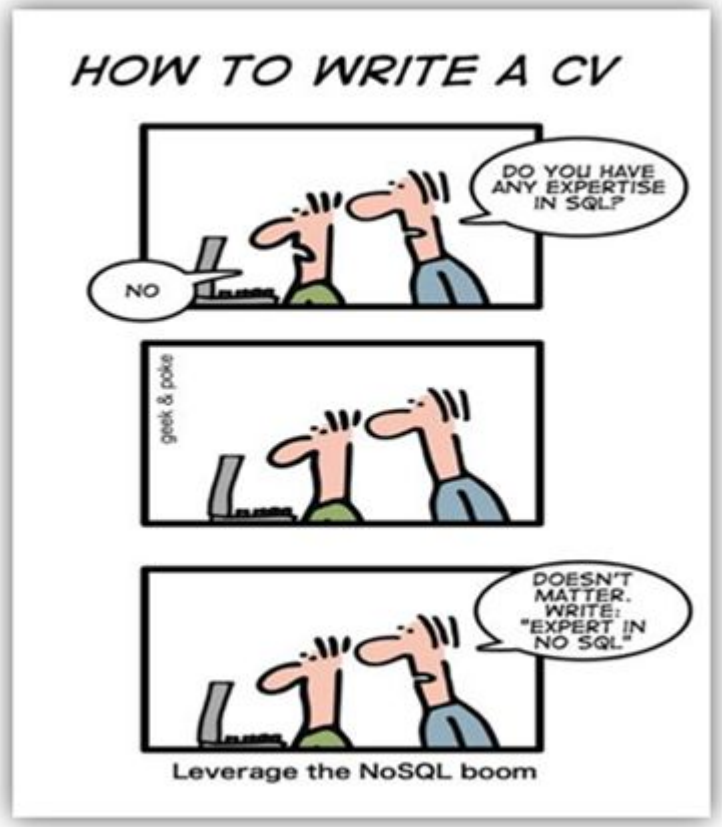
Done by Job and Task trackers



# More to observe

Apache Spark

Apache Casandra

Apache Kafka

# Köszönöm a figyelmet!