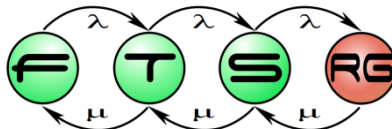


OSGi



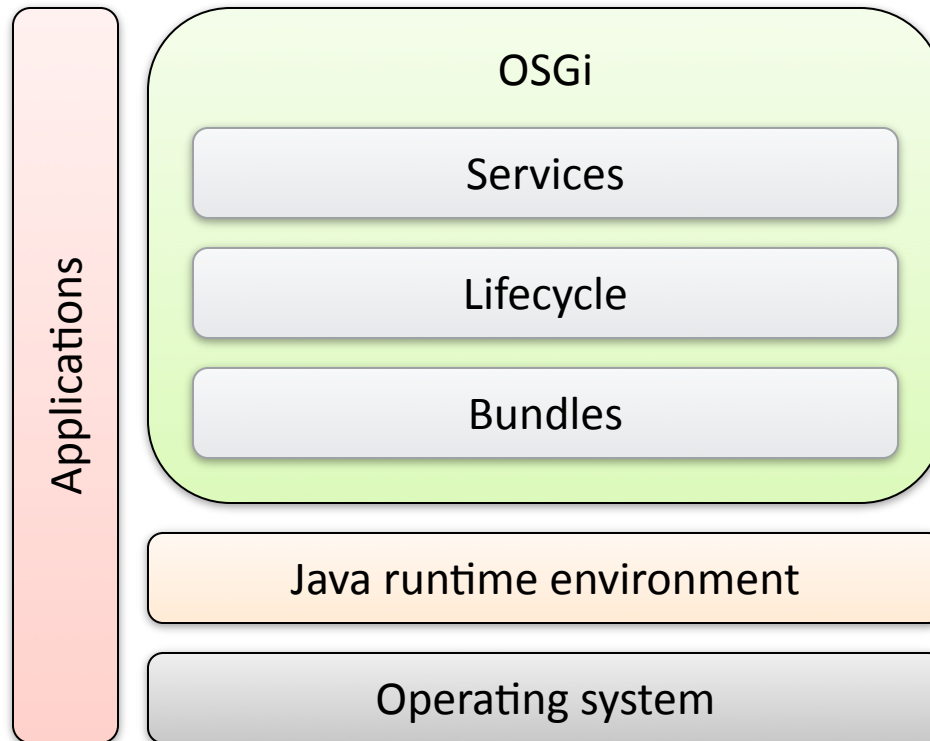
OSGi

- „A dynamic module system for Java”
- OSGi szövetség (www.osgi.org) → ~30 teljes tag (Nokia, IBM, NTT, Motorola. stb.)



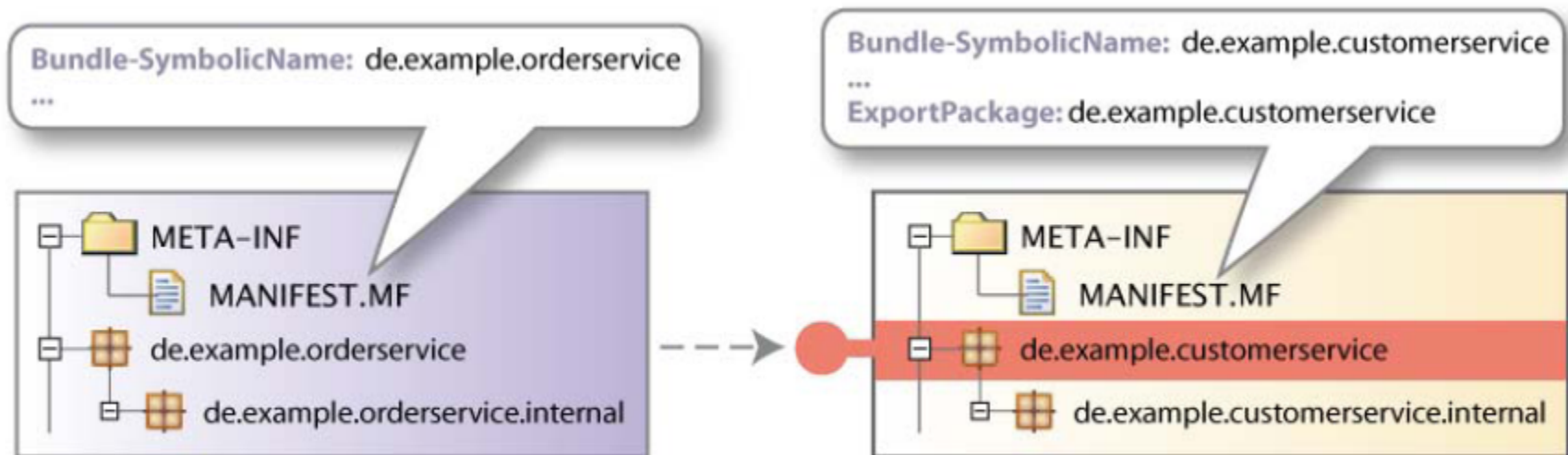
- Közös problémák (integráció, verziózás, élet ciklus)
- Megoldás → OSGi szabvány (specifikáció)
- Komponens alapú
- Közös integrációs „primitívekkel” (interfészek, leírók, stb.)

OSGi



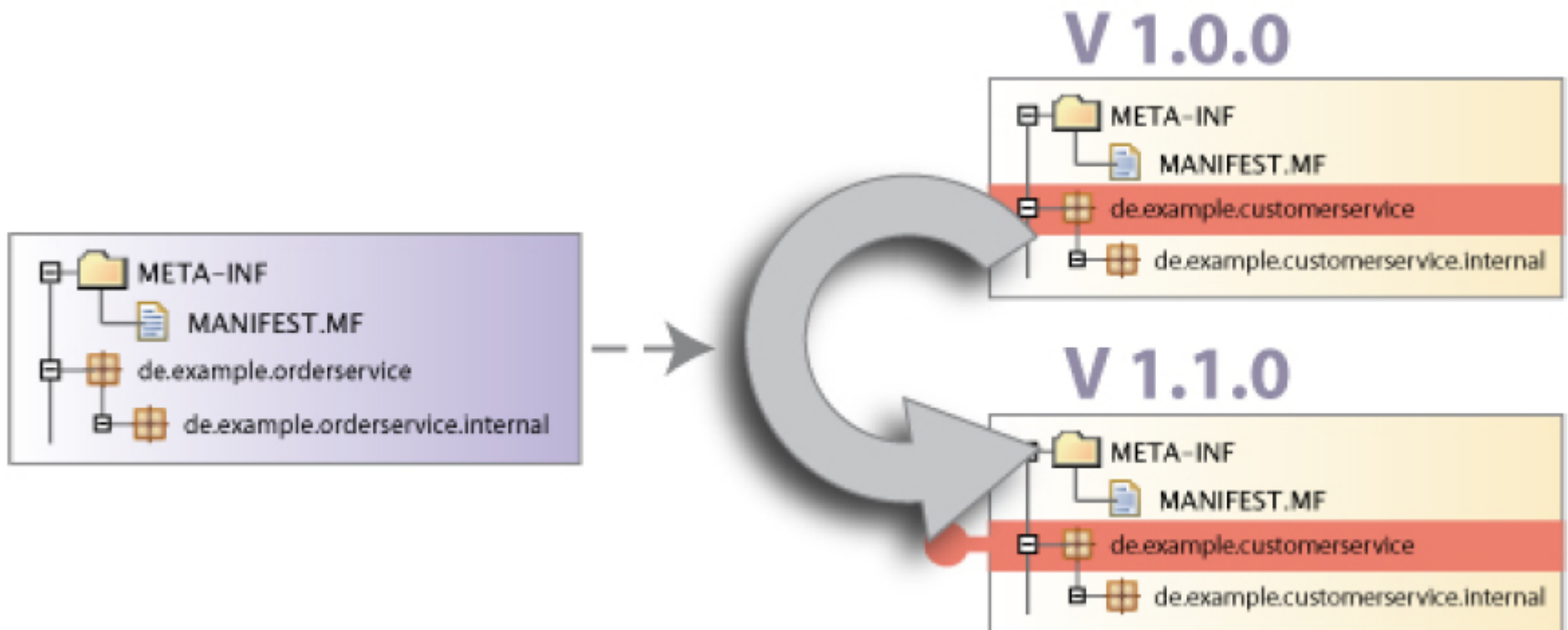
OSGi alapok: modulok

- Modulok (bundles)
 - Public és private API láthatósága
 - Függőség kezelés
 - Verziózás

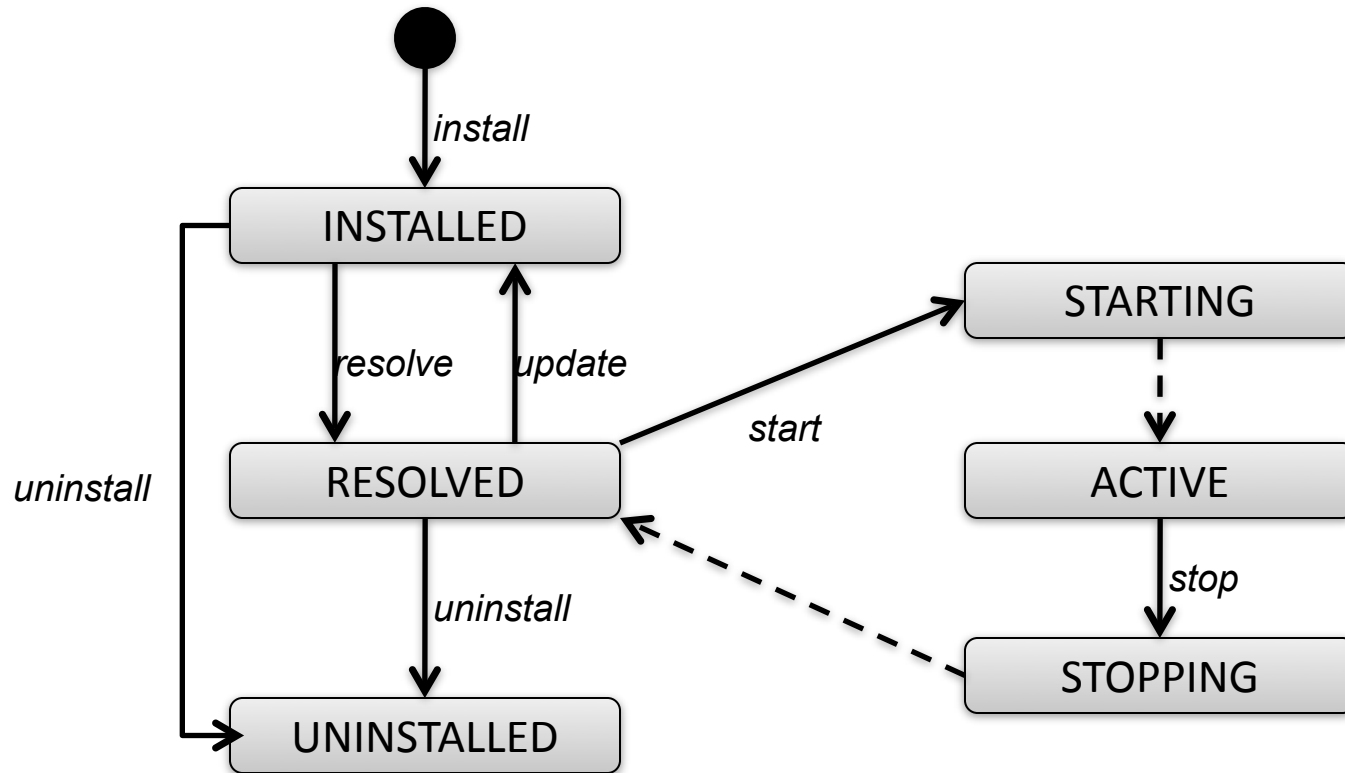


OSGi alapok: élelciklus

- Élelciklus (Life cycle)
- Dinamikus Bundle:
 - Betöltés (install)
 - Start/stop

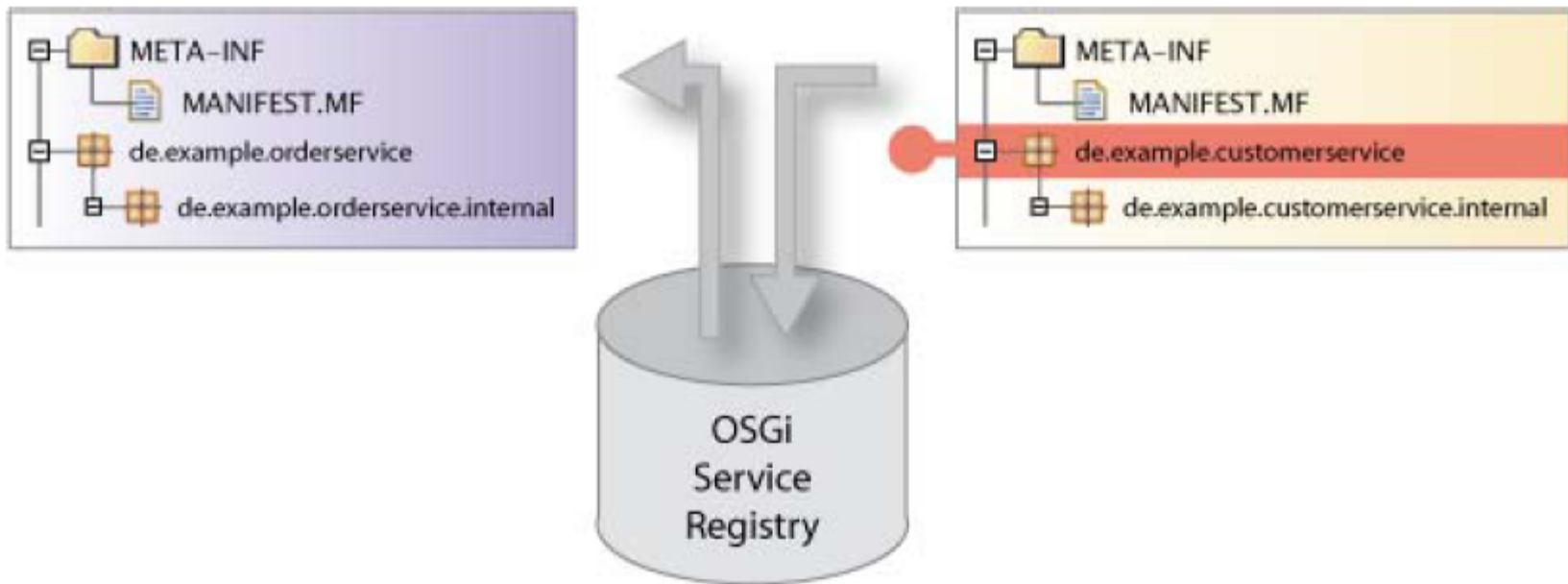


OSGi alapok: Bundle életciklus modell



OSGi alapok: szolgáltatások

- Szolgáltatás orientált (Services):
 - Bundle-ok kiejánlhatóak
 - Kereshetőek (service registry)
 - Futás idő közben megjelenhetnek(/eltűnhetnek)!



OSGi Bundles

OSGi belülről (Manifest.MF)

- Identifikáció:
 - Bundle-SymbolicName: org.eclipse.equinox.registry
 - Bundle-Version: 3.2.100.v20060918
 - Bundle-Name: Eclipse Extension Registry
 - Bundle-Vendor: Eclipse.org
- ClassPath:
 - Bundle-ClassPath: ., someOtherJar.jar
- Életciklus:
 - Bundle-Activator: org.eclipse.core.internal.registry.osgi.Activator
- Függőségek:
 - Import-Package: javax.xml.parsers,
 - org.xml.sax,
 - org.osgi.framework;version=1.3
 - Require-Bundle: org.eclipse.equinox.common;bundle-version="[3.2.0,4.0.0)"
 - Bundle-RequiredExecutionEnvironment: CDC-1.0/Foundation-1.0,J2SE-1.3
- Kiajánlás (export)

Modul réteg

- A modulok indíthatók, leállíthatók
- A futó bundle-k szolgáltatásai kiajánlásra kerülnek
- Fontos manifeszt adatok
 - Bundle-activator: az életciklus menedzselését végző osztály neve
 - Bundle-classpath: a bundle-n belüli classpath-ok listája. A default értéke: . (a bundle root)
 - Bundle-name: olvasható név
 - Bundle-SymbolicName: azonosító (egyedi)
 - Bundle-UpdateLocation: URL, ahonnan a frissítések letölthetők
 - Export-Package: a kiajánlott java csomagok listája
 - Import-Package: importált csomagok listája
 - Require-Bundle: szükséges modulok listája (import + függőségek)

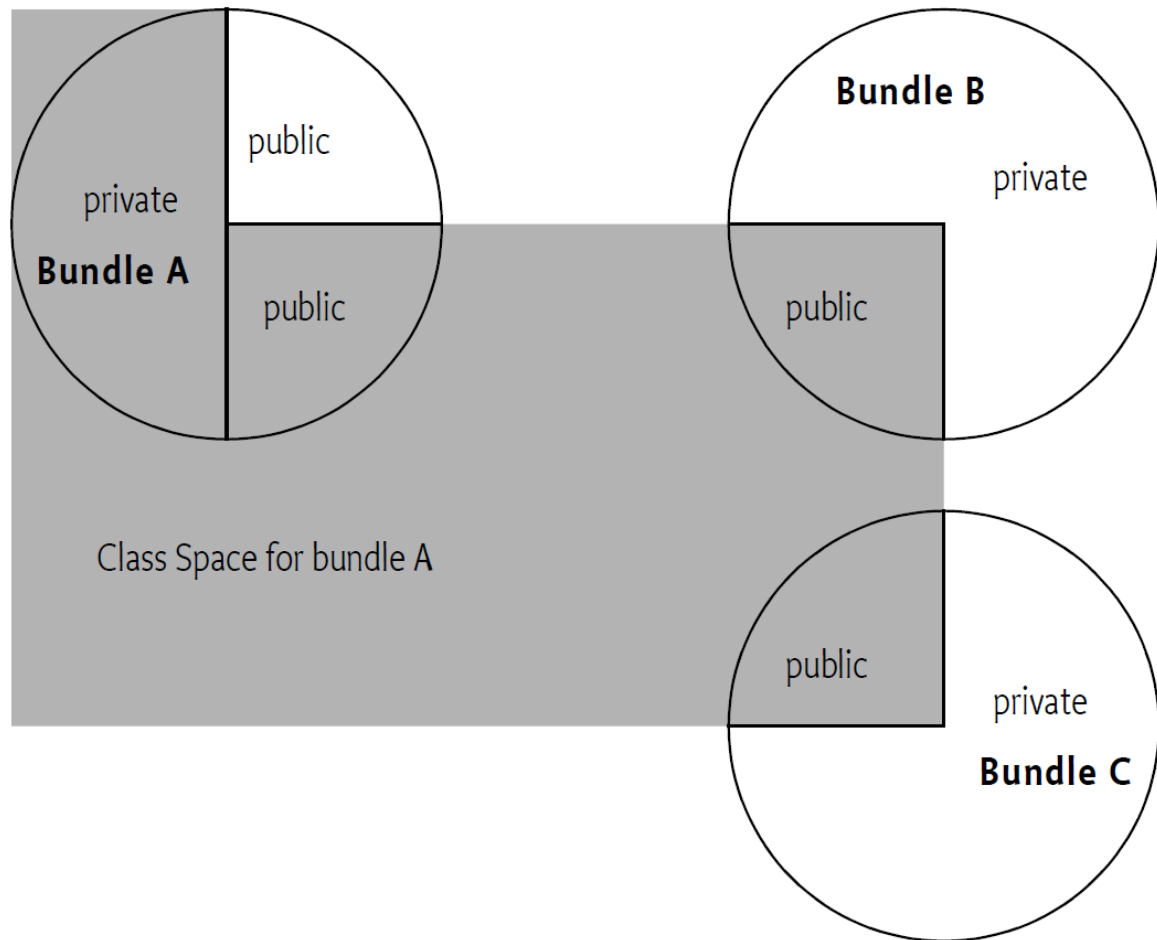
Class loading

- Minden bundle egy vm-en belül fut
- Minden bundle-nak saját class loadere van
 - 3 helyről tölthet be osztályokat/erőforrásokat
 - Boot class path: java.* csomagok és implementációik
 - Framework class path: a framework-nek saját class loadere van, amitől elkérhetőek az interfészek és implementáló osztályok
 - Bundle space: a bundle jar fájljai, valamint a hozzá kötődő egyéb jar-ok

Class loading – Class space

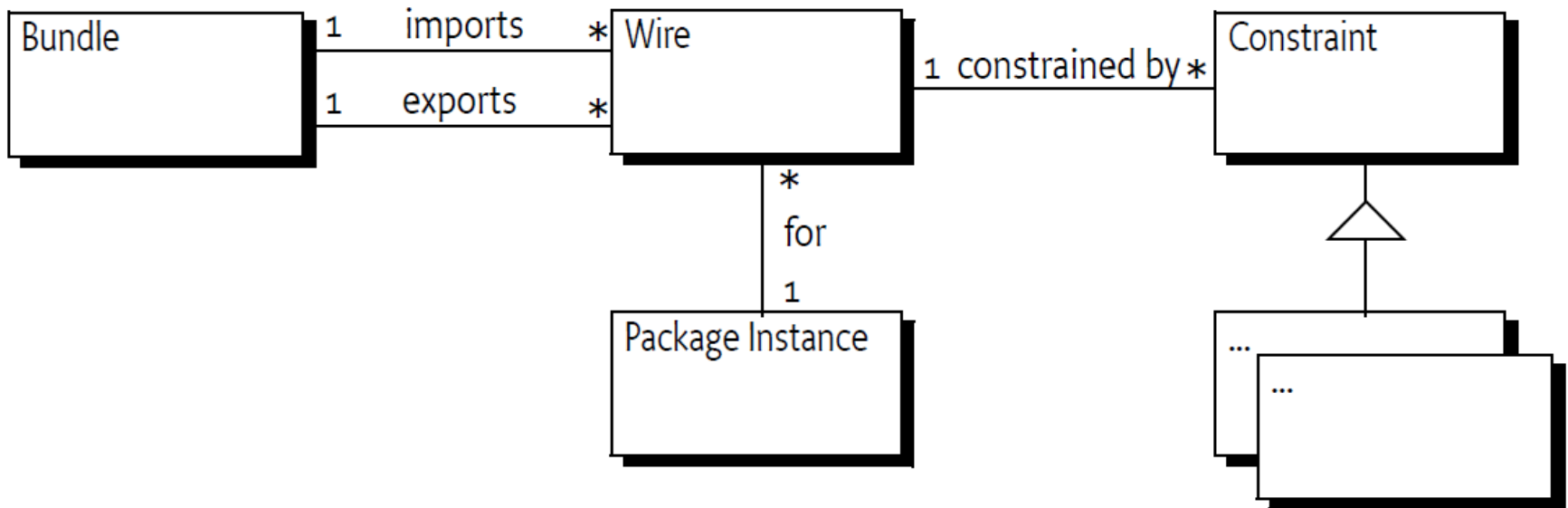
■ Egy bundle class space-e

- A szülő class loader-e (java.*)
- Importált csomagok
- Függőségek
- A bundle privát classpath-a
- Csatolt fragmensek



Bundle feloldás

- Feloldás: az importerek és exporterek összekötése
 - Kényszereknek megfelelően
- Vezeték (wire): összeköttetés importer és exporter között



Metaadat feloldás

■ Bundle-SymbolicName

- kötelező, egyedi azonosító
- Ha két egyező nevű és verziójú van, a második telepítése sikertelen
- Paraméterek
 - Singleton: csak egyetlen verziója lehet betöltve
 - Fragment-attached: definiálja, hogyan lehet fragmenseket hozzákapcsolni
 - Always: bármikor kapcsolódhat
 - Never: nem lehetséges
 - Resolve-time: csak a resolve fázisban

```
Bundle-SymbolicName: com.acme.foo;singleton:=true
```

○ Példa:

■ Bundle-Version

- Meghatározott formátum: major.minor.micro.qualifier
- Összehasonlítás hierarchikus
 - Numerikusan, illetve a qualifier esetén String.compareTo
 - Két verzió akkor azonos, ha minden szegmensük egyezik

○ Példa:

Metaadat feloldás

■ Imported-packages

- Importált csomagok listája
- Resolution – a csomagot fel kell oldani kötelező import esetében, ha ez sikertelen a bundle sem tölthető be
- Version – verzió intervallum a csomagot exportáló csomagra zárt [], nyitott (), pl. [1.0.0,2.0.0)
- Bundle-version: az exportáló bundle verziója
- Bundle-symbolic-name: az exportáló bundle neve
- Példa:

```
Import-Package: com.acme.foo;com.acme.bar;  
                version="[1.23,1.24]";  
                resolution:=mandatory
```

■ Exported-packages

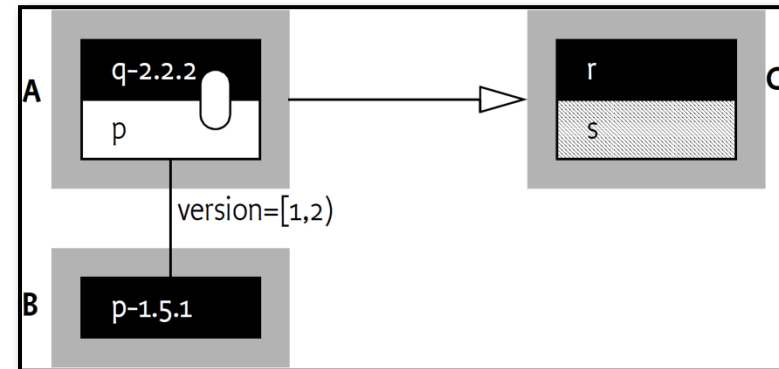
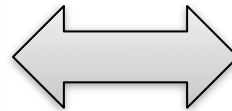
- Exportált csomagok listája
- Hasonlóan az Imported-packages-hez
- Példa:

```
Export-Package: com.acme.foo;com.acme.bar;version=1.23
```

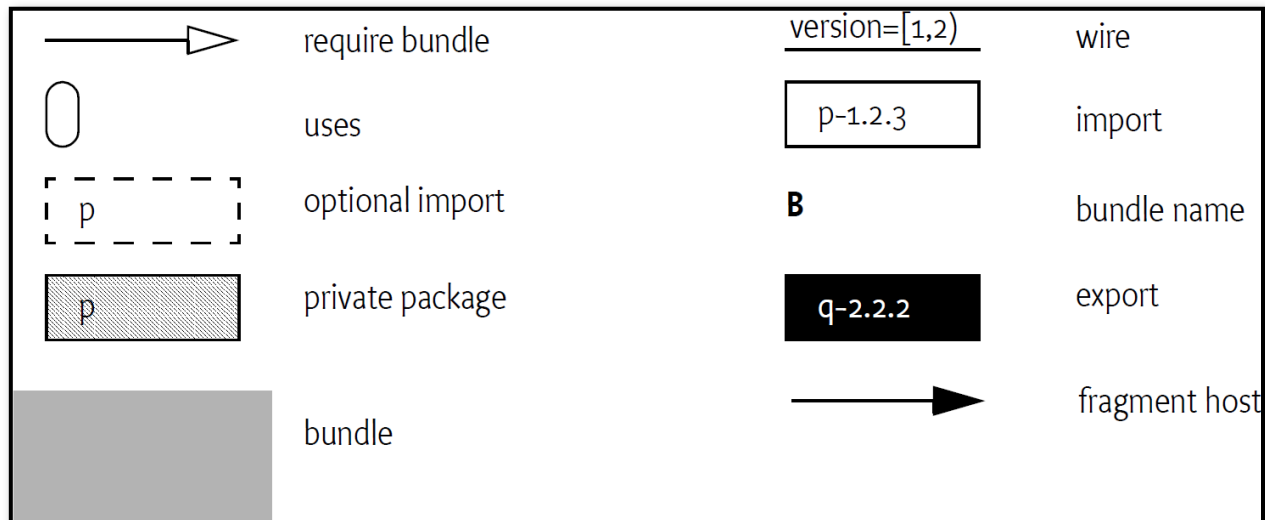
Bundle diagram

A: Import-Package: p; version="[1,2)"
 Export-Package: q; version=2.2.2; uses:=p
 Require-Bundle: C
 B: Export-Package: p; version=1.5.1
 C: Export-Package: r

Szöveges leírás



Grafikus ábrázolás



Jelölés

OSGi Services

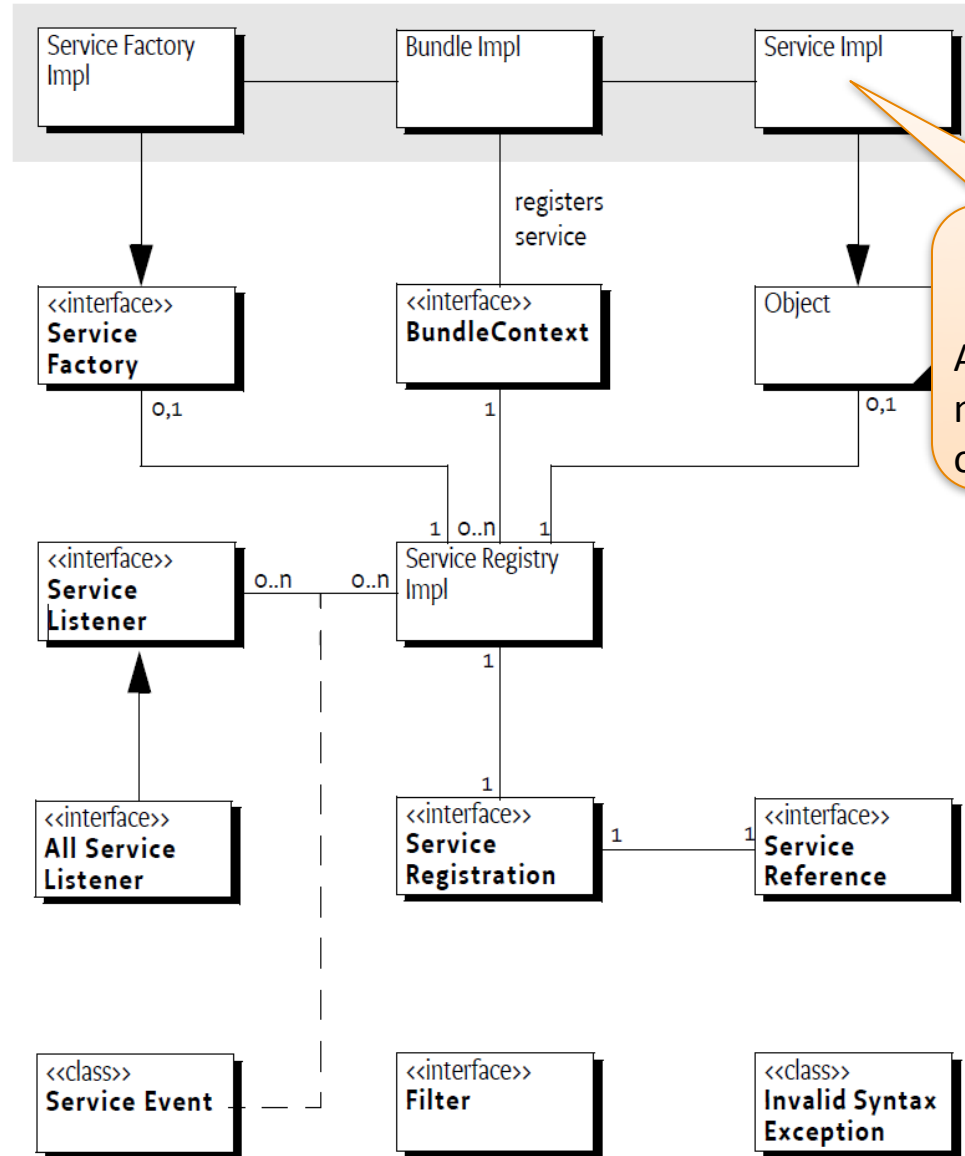
Szolgáltatási réteg

- Definiálja az együttműködési modellt
 - „Publish, find, and bind”
 - A szolgáltatás egy normál java objektum
 - Regisztrálódik egy vagy több java interfész alatt
- A bundle-k
 - Regisztrálhatnak
 - Kereshetnek
 - Használhatnak szolgáltatásokat
 - Illetve, ezekkel kapcsolatban eseményeket kezelhetnek

Szolgáltatási réteg - alaptulajdonságok

- Kollaboratív: bundle-k közötti együttműködés megvalósítása
- Dinamikus: futásidejű változások
 - Új szolgáltatások megjelenése
 - Régi szolgáltatások megszűnése
- Biztonságos: hozzáférés korlátozható
- Reflektív: teljes hozzáférés a réteg belső állapotához
- Verziókezelés: a szolgáltatások frissülhetnek
- Perzisztens id: framework indítások között is lehet a szolgáltatásokat követni

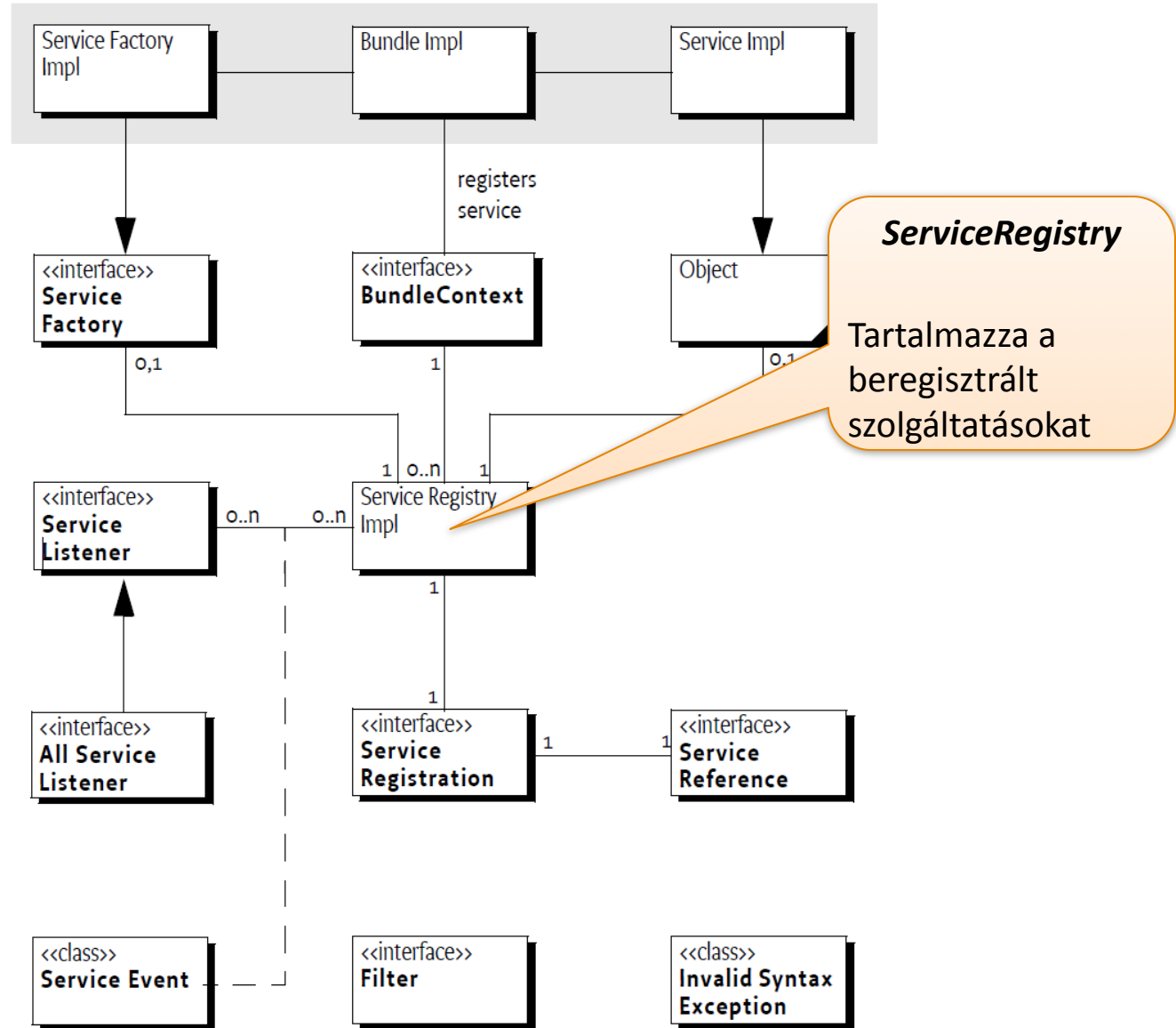
Szolgáltatási réteg elemei



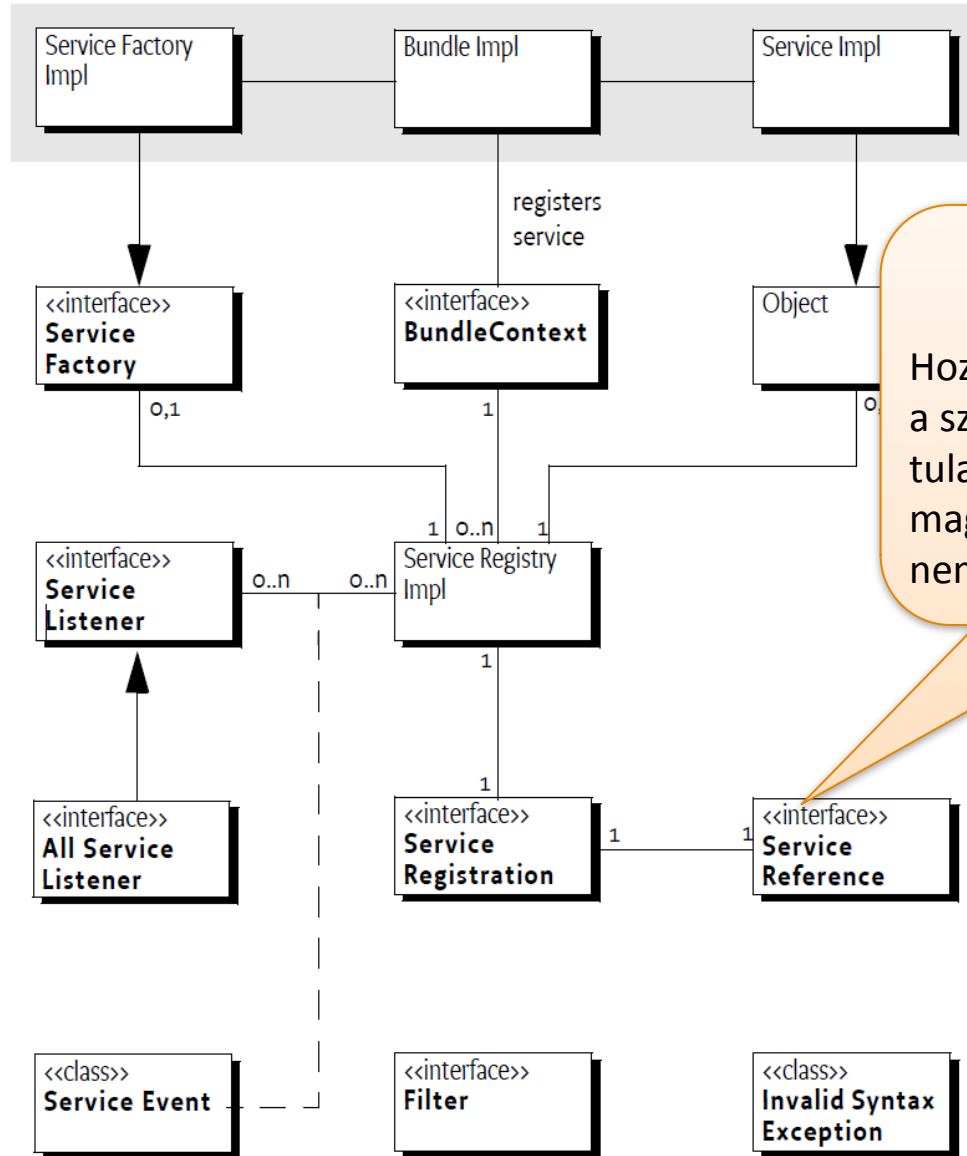
Service

A szolgáltatást megvalósító objektum

Szolgáltatási réteg elemei



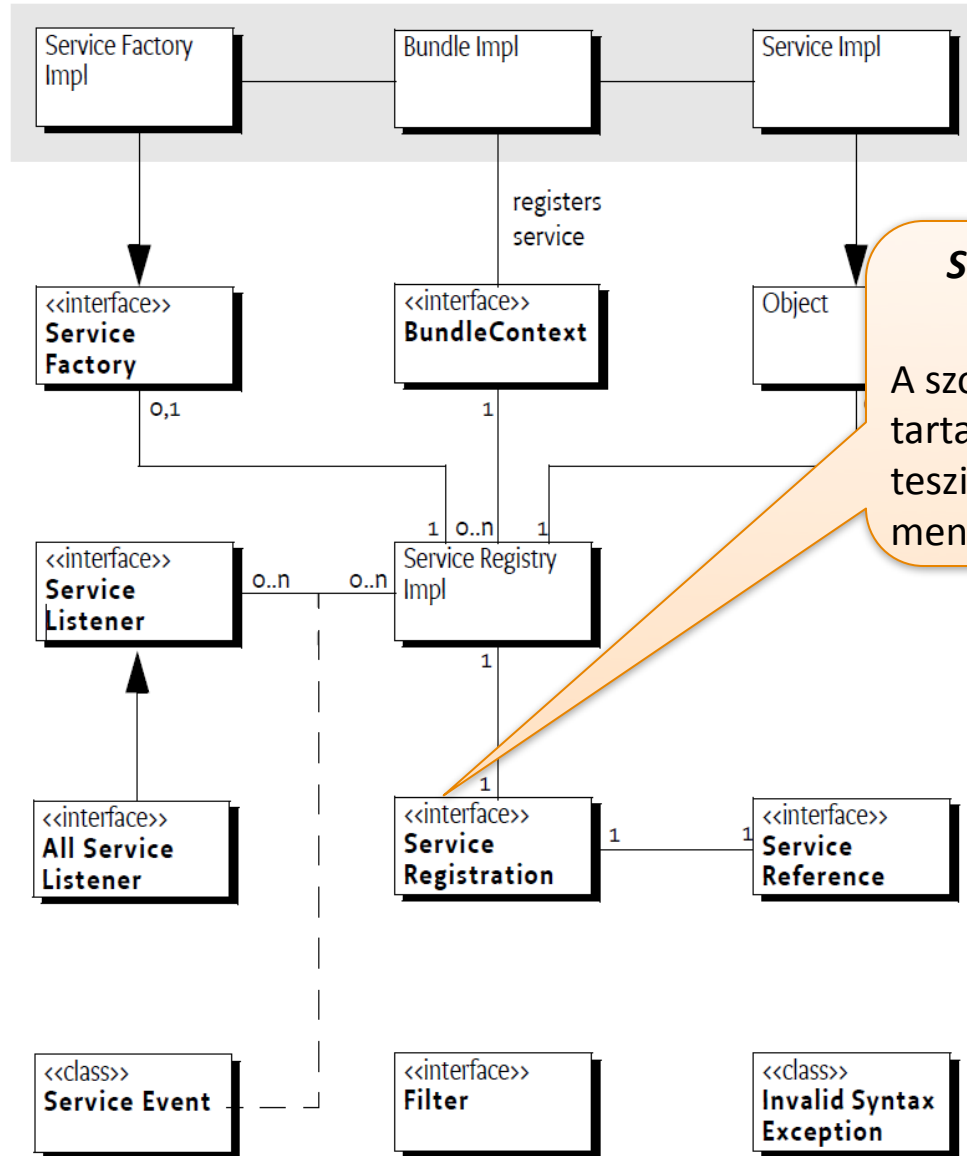
Szolgáltatási réteg elemei



ServiceReference

Hozzáférést biztosít a szolgáltatás tulajdonságaihoz, de magához az objektumhoz nem.

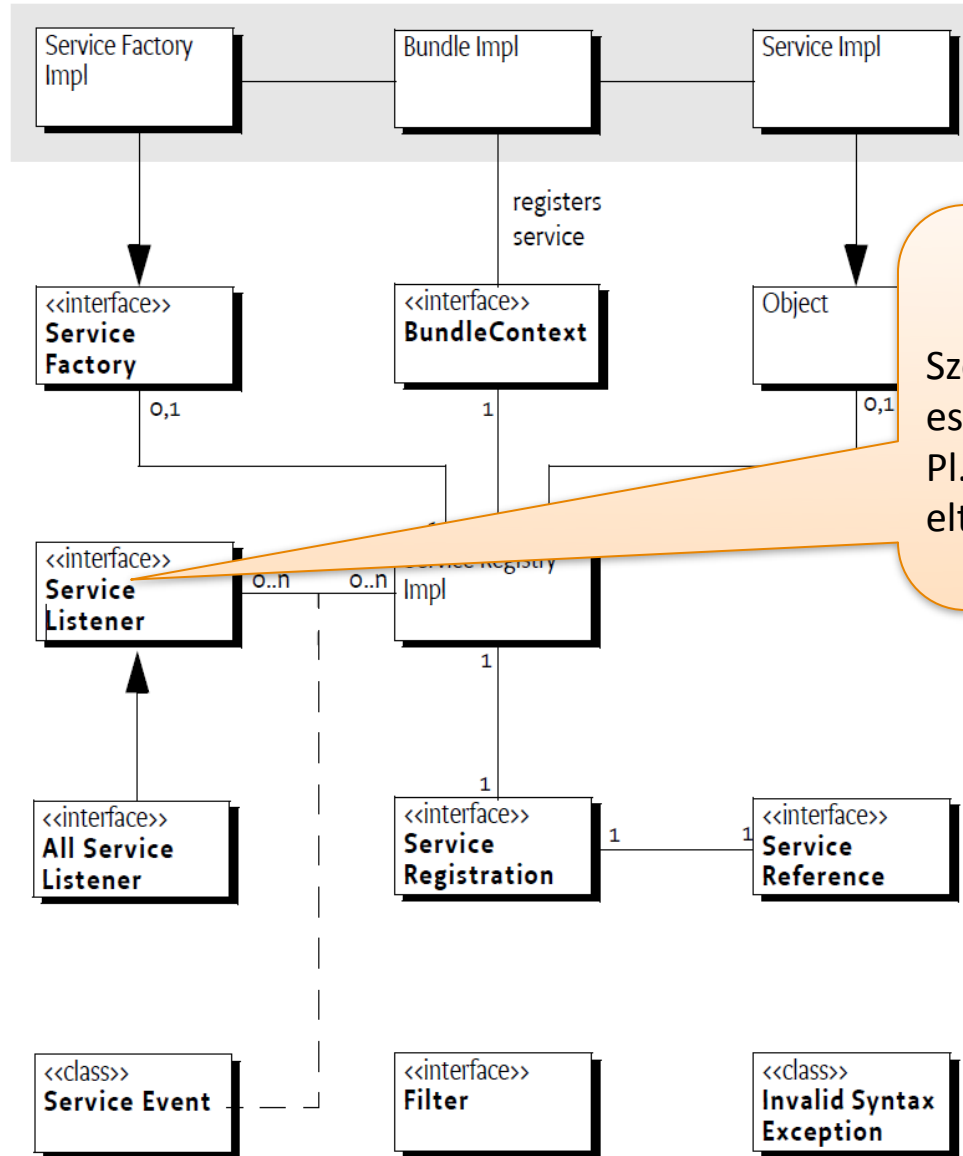
Szolgáltatási réteg elemei



ServiceRegistration

A szolgáltatás adatait tartalmazza, lehetővé teszi a regisztráció menedzselését

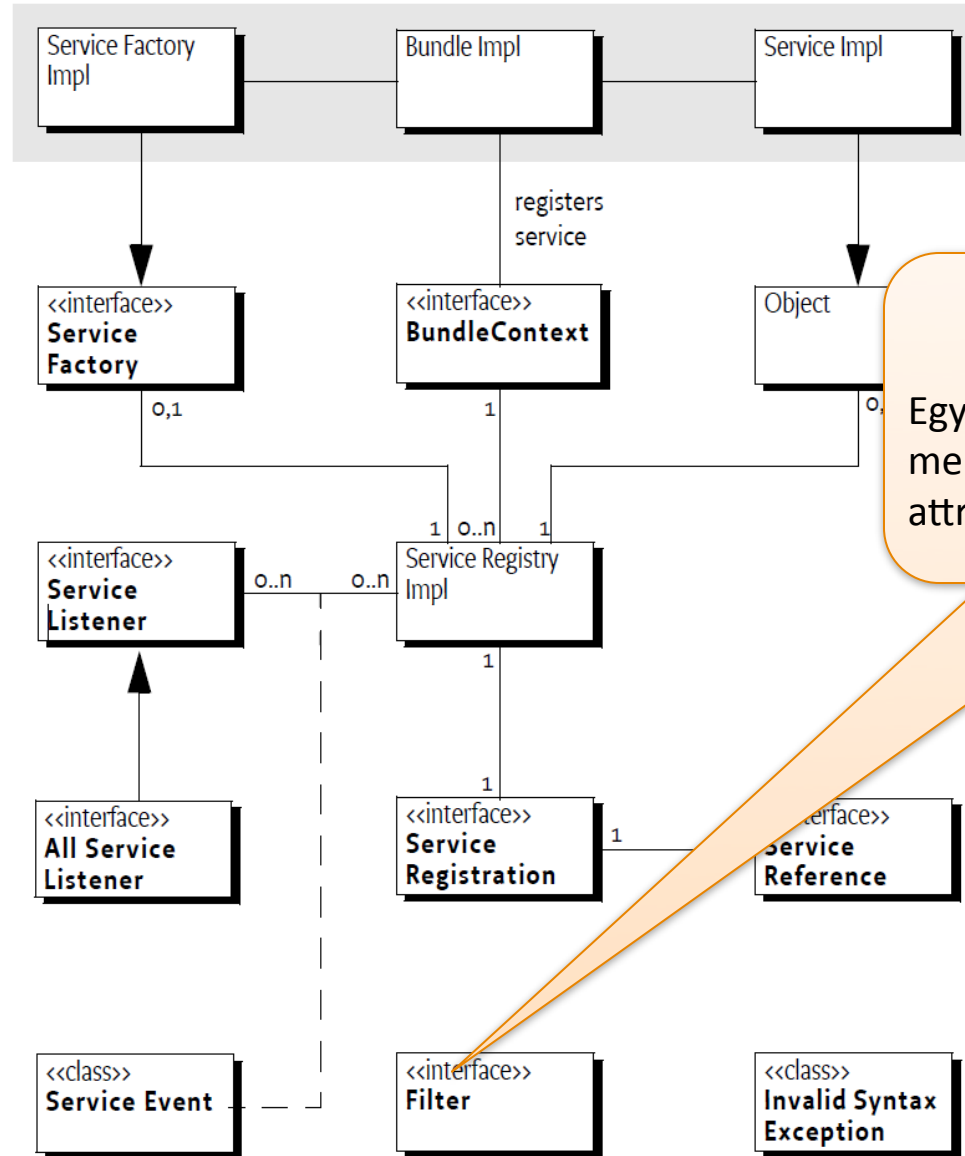
Szolgáltatási réteg elemei



ServiceListener

Szolgáltatással kapcsolatos eseményeket figyel.
Pl.: szolgáltatás megjelenés / eltűnés

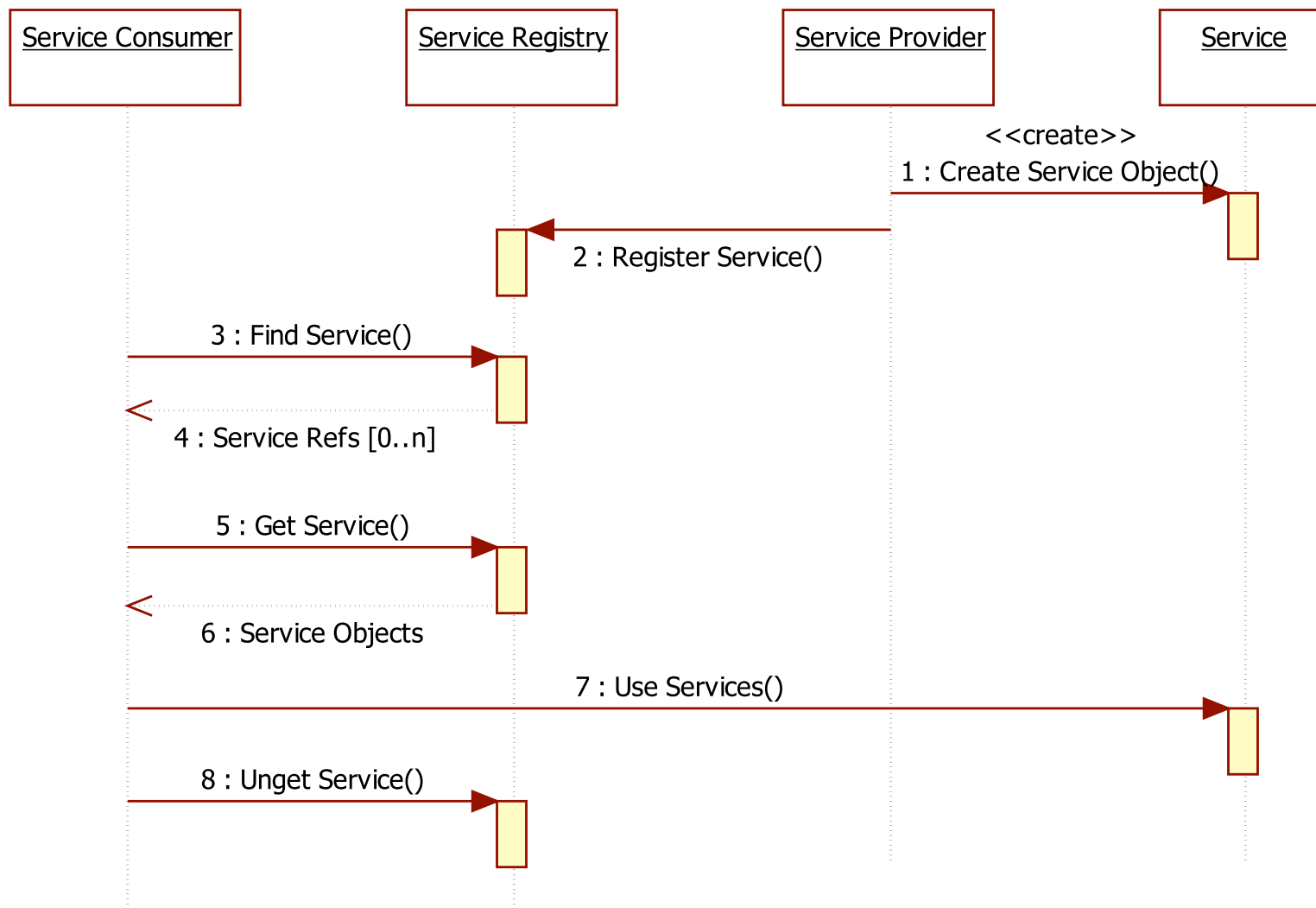
Szolgáltatási réteg elemei



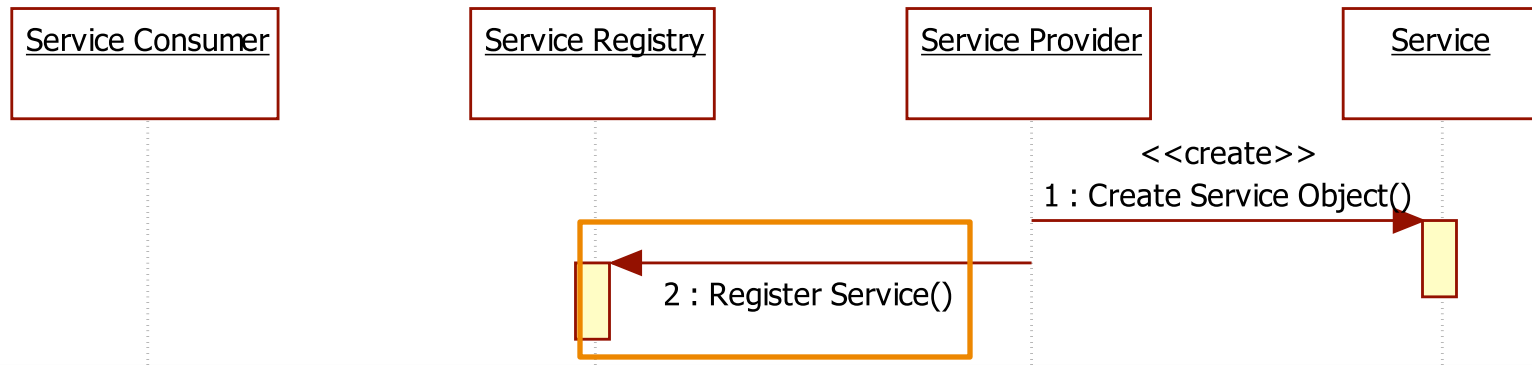
Filter

Egyszerű szűrő nyelvet ad meg, mely a szolgáltatások attribútumaira szűr

Szolgáltatások „életciklusa”



Szolgáltatások regisztrációja



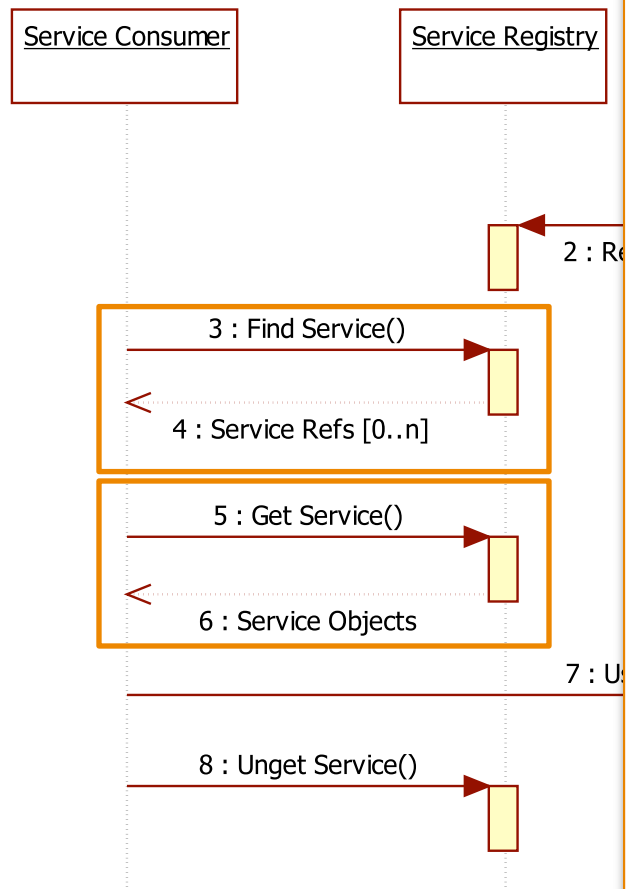
❖ BundleContext

- ❖ registerService(String, Object, Dictionary)
 - ⦿ Egy adott interfész név alá regisztrálja a szolgáltatást
- ⦿ registerService(String[], Object, Dictionary)
 - ⦿ Több interfész név alá regisztrálja a szolgáltatást

⦿ Példa

```
service = new HelloServiceImpl();  
// register the service  
context.registerService(HelloService.class.getName(),  
                        service, new Hashtable());
```

Szolgáltatások keresése



❖ BundleContext

- ❖ `getServiceReference(String)`
 - ❖ Ha több van, a ranking dönt
- ❖ `getServiceReferences(String,String)`
 - ❖ Az összes referencia az adott interfészhez és filterhez
- ❖ `getService(ServiceReference)`
 - ❖ A szolgáltatás objektumot adja vissza
- ❖ `ungetService(ServiceReference)`
 - ❖ Szolgáltatás „elengedése”

🕒 Példa

```
ServiceReference reference = context.getServiceReference(
    HelloService.class.getName());
HelloService service = (HelloService) context.    getService(
    reference);
```

Service Tracker

- Problémák a szolgáltatások közvetlen lekérésével
 - Nincs értesítés, ha egy szolgáltatás megszűnik/megjelenik
 - Alacsony szintű API
 - Minden alkalommal le kéne kérni a szolgáltatást, amikor használni akarjuk
 - Lehetőség még a ServiceListener használata, de könnyű hibázni
 - Körülményes, sok Java kód
- Megoldás: a ServiceTracker használata
 - Szolgáltatáshoz lehet regisztrálni
 - Jelez, ha
 - Megjelenik egy, az adott típusú szolgáltatás
 - Eltűnik egy, az adott típusú szolgáltatás
 - Módosul egy, az adott típusú szolgáltatás

OSGi compendium

OSGi compendium

- OSGi core specifikáció kiegészítése
- Szolgáltatások jegyzéke
- Fontosabb szolgáltatás csoportok
 - Declarative Services: Deklaratív modell szolgáltatások kezelésére
 - Http Service: Http alapú szolgáltatások
 - Remote Services: Szolgáltatások használat különböző framework-ök között (Version 1.0)
 - Log Service: általános célú logolásra alkalmas szolgáltatások
 - stb...

**OSGi Service Platform
Service Compendium**
The OSGi Alliance

Release 4, Version 4.2
August 2009



Declarative Services

- DS előzménye: Service Binder
 - Humberto Cervantes és Richard Hall fejlesztették ki
 - Szolgáltatások függőségeinek automatikus menedzselése
 - Fejlesztő a szolgáltatások írására koncentrálhat (POJO)
- DS a Compendium services része R4 óta
 - XML leírók (komponens leíró)
 - „Publish, find, and bind” deklaratív módon
 - Válasszuk szét a felelőségeket:
 - Szolgáltatás implementáció ← továbbra is bundle felelőssége
 - Szolgáltatás regisztráció ← Service Component Runtime (SRC)
 - Dinamikus, mint a szervizek
 - „On demand” betöltés, mint az Eclipse Extension-ök.

DS – komponens leíró

- XML alapú
- Deklaratív módja a szervizek regisztrálásának, kötésének
- OSGI-INF könyvtárban
- Több komponens egy bundle-ban
- MANIFEST.MF-ban összeset fel kell venni
 - Service-Component

Példa komponensleíróra

```
<scr:component xmlns:scr="http://www.osgi.org/
  xmlns/scr/v1.1.0"
  name="sample.component">
  <implementation
class="org.sample.HelloServiceImpl"/>
  <service>
    <provide interface="org.sample.HelloService"/>
  </service>
  <reference
    bind="setService"
    unbind="unsetService"
    cardinality="0..1"
    interface="org.sample.ServiceForHello" name=
„SERVICEFORHELLO"
    policy="dynamic"/>
</scr:component>
```

Példa komponensleíróra

```
<scr:component xmlns:scr="http://www.osgi.org/xmlns/scr/v1.1.0"
  name="sample.component">
  <implementation
class="org.sample.HelloServiceImpl"/>
  <service>
    <provide interface="org.sample.HelloService"/>
  </service>
  <reference
    bind="setService"
    unbind="unsetService"
    cardinality="0..1"
    interface="org.sample.ServiceForHello" name=
„SERVICEFORHELLO"
    policy="dynamic"/>
</scr:component>
```

Kiajánlott
szolgáltatás
definíciója

Példa komponensleíróra

```
<scr:component xmlns:scr="http://www.osgi.org/xmlns/scr/v1.1.0"
  name="sample.component">
  <implementation
class="org.sample.HelloServiceImpl"/>
  <service>
    <provide interface="org.sample.HelloService"/>
  </service>
  <reference
    bind="setService"
    unbind="unsetService"
    cardinality="0..1"
    interface="org.sample.ServiceForHello" name=
"SERVICEFORHELLO"
    policy="dynamic"/>
</scr:component>
```

Kiajánlott
szolgáltatás
definíciója

Felhasznált
szolgáltatás

DS – komponens leíró

- Eclipse támogatás: Declarative Service Tooling
 - Component Definition Editor

The screenshot shows the Eclipse IDE's Component Definition Editor with the 'Overview' tab selected. The window title is 'component.xml'. The 'Component' section contains fields for Name (sample.comonent), Class* (org.sample.HelloServiceImpl with a 'Browse...' button), Activate, Deactivate, and Modified. The 'Options' section includes Factory ID, Configuration Policy (a dropdown), and two checkboxes: 'This component is enabled when started' (checked) and 'This component is immediately activated'. The 'Properties (0)' section has a large empty text area and buttons for 'Add File...', 'Add Property...', 'Edit...', 'Remove', 'Up', and 'Down'. The bottom tab bar shows 'Overview', 'Services', and 'Source'.

The screenshot shows the Eclipse IDE's Component Definition Editor with the 'Services' tab selected. The window title is 'component.xml'. The 'Referenced Services (1)' section lists '0..1 SERVICEFORHELLO [setService,unsetService]' with buttons for 'Add...', 'Remove', 'Edit...', 'Up', and 'Down'. The 'Provided Services (1)' section lists 'org.sample.HelloService' with buttons for 'Add...', 'Remove', and 'Edit...'. The bottom tab bar shows 'Overview', 'Services', and 'Source'.

DS – komponensek összekapcsolása

- 3 dimenzió
 - Opcionális / Kötelező
 - Egy értékű / Több értékű
 - Statikus / Dinamikus
- Első két dimenziót adja a számosság (cardinality):
 - 0..1 → opcionális, egy értékű
 - 1..1 → kötelező, egy értékű
 - 1..n → kötelező, több értékű
 - 0..n → opcionális, több értékű
- Harmadik dimenziót pedig a policy
 - dynamic: a szolgáltatás menet közben „kicserélhető”
 - static: garantáltan egy szolgáltatás lesz végig (pl. állapottal rendelkező)

```
<reference
  bind="setService"
  unbind="unsetService"
  cardinality="0..1"
  interface="org.sample.ServiceForHello" name="SERVICEFORHELLO"
  policy="dynamic"/>
```

DS – komponensek

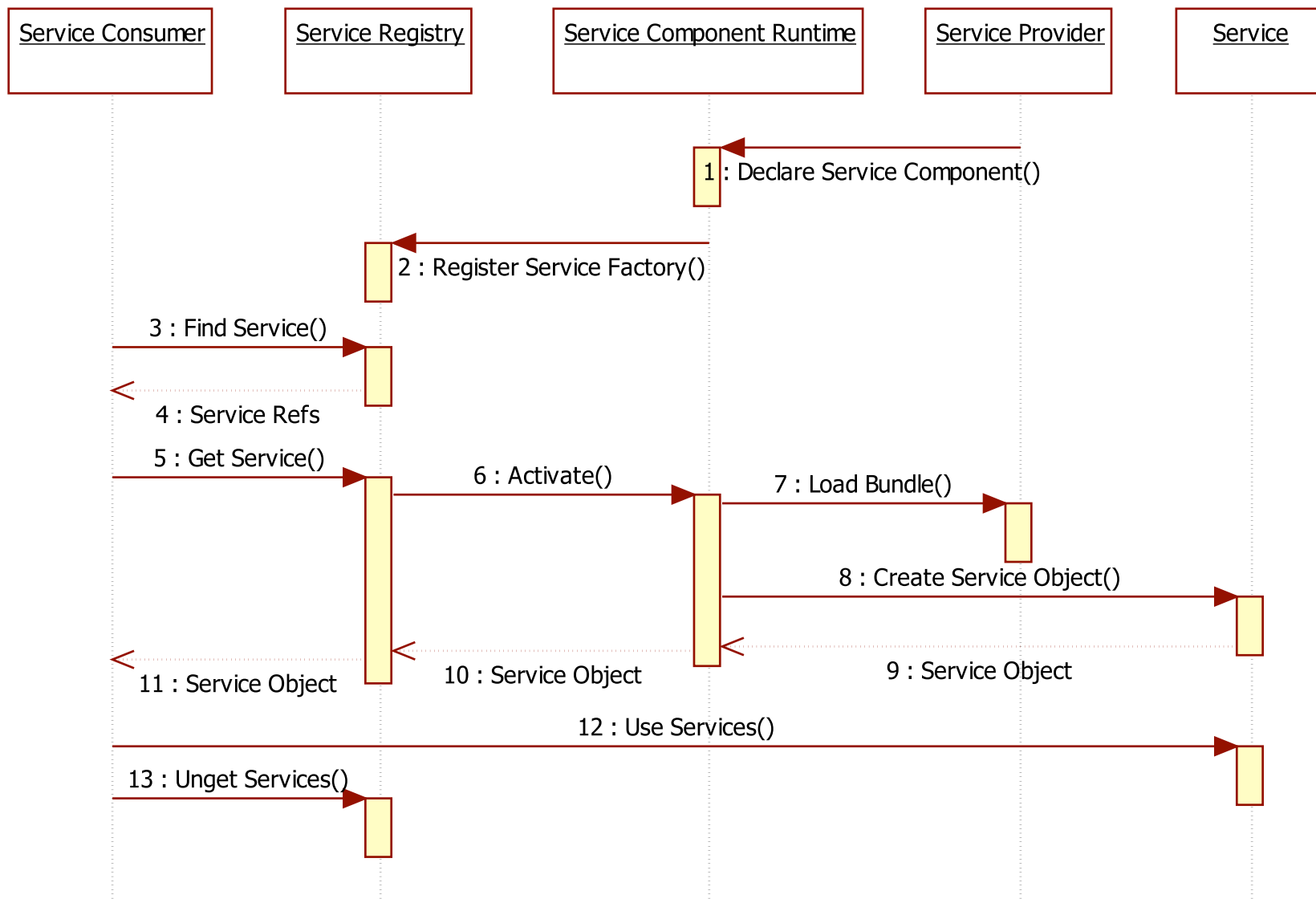
- 3 dimenzió
 - Opcionális / Kötelező
 - Egy értékű / Több értékű
 - Statikus / Dinamikus
- Első két dimenziót adja a számosság (**cardinality**):
 - 0..1 → opcionális, egy értékű
 - 1..1 → kötelező, egy értékű
 - 1..n → kötelező, több értékű
 - 0..n → opcionális, több értékű
- Harmadik dimenziót pedig a **policy**
 - dynamic: a szolgáltatás menet közben „kicsúszhat”
 - static: garantáltan egy szolgáltatás lesz végig (pl. állanottal rendelkező szolgáltatás)

Szolgáltatás megjelenése
esetén hívandó metódus

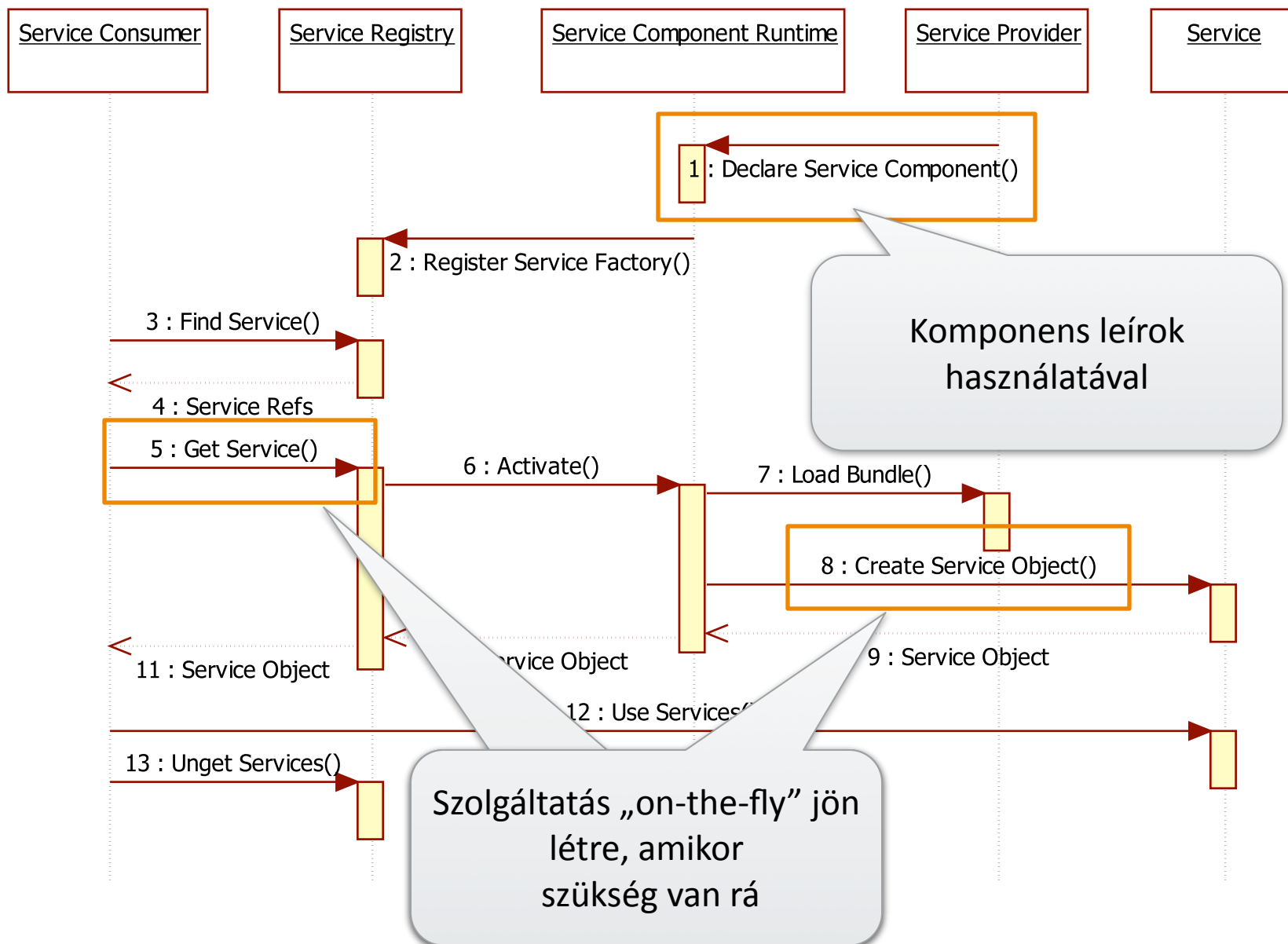
Szolgáltatás megszűnése
esetén hívandó metódus

```
<reference  
  bind="setService"  
  unbind="unsetService"  
  cardinality="0..1"  
  interface="org.sample.ServiceForHello" name="SERVICEFORHELLO"  
  policy="dynamic"/>
```

DS – Életciklus menedzselés



DS – Életciklus menedzselés



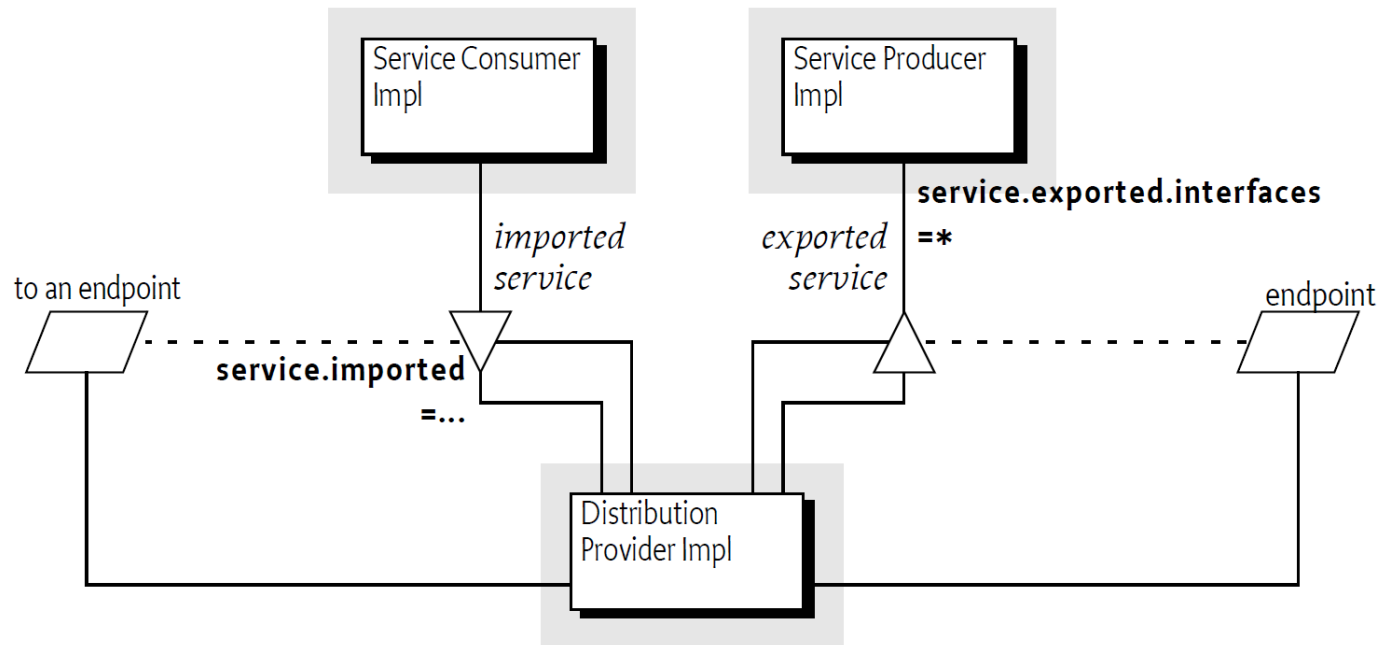
Http services

- Egyike a legrégebbi, legelterjedtebb szolgáltatásnak
- OSGi alapú webes komponensek fejlesztése
- Jelenleg támogatott komponensek
 - Servlet-ek regisztrálása
 - Servlet-ek regisztrálása on-the-fly
 - OSGi servlet-be csomagolva
 - Erőforrások regisztrálása (HTML fájlok, képek, stb...)
- Alkalmazások
 - Pl. Apache Felix Web Console: OSGi konténerek monitorozása

Remote Services

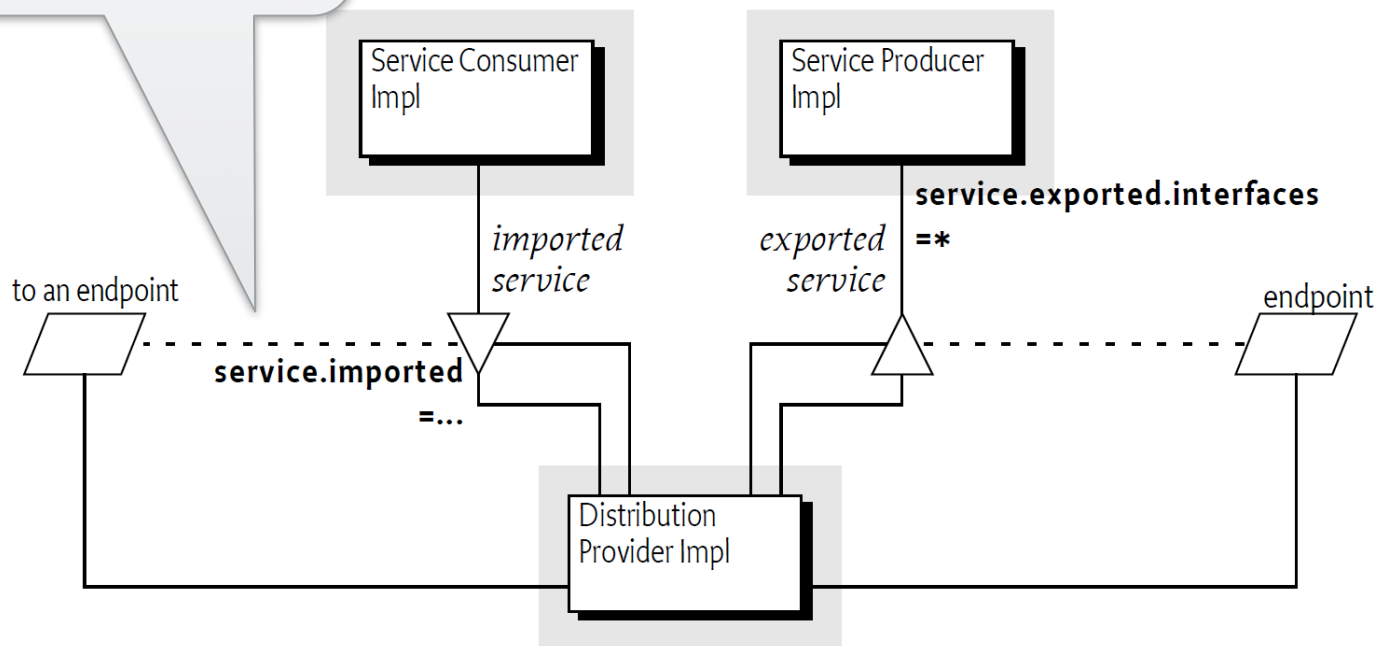
- Követelmények
 - Átlátszóság: nincs különbség helyi és távoli szolgáltatások között
 - Általános: Ne állítson korlátokat az elosztottsággal
 - Konzisztens viselkedés: helyi és távoli szerver ugyanúgy viselkedjen
- Elosztott alkalmazások készítése

Remote Services



Remote Services

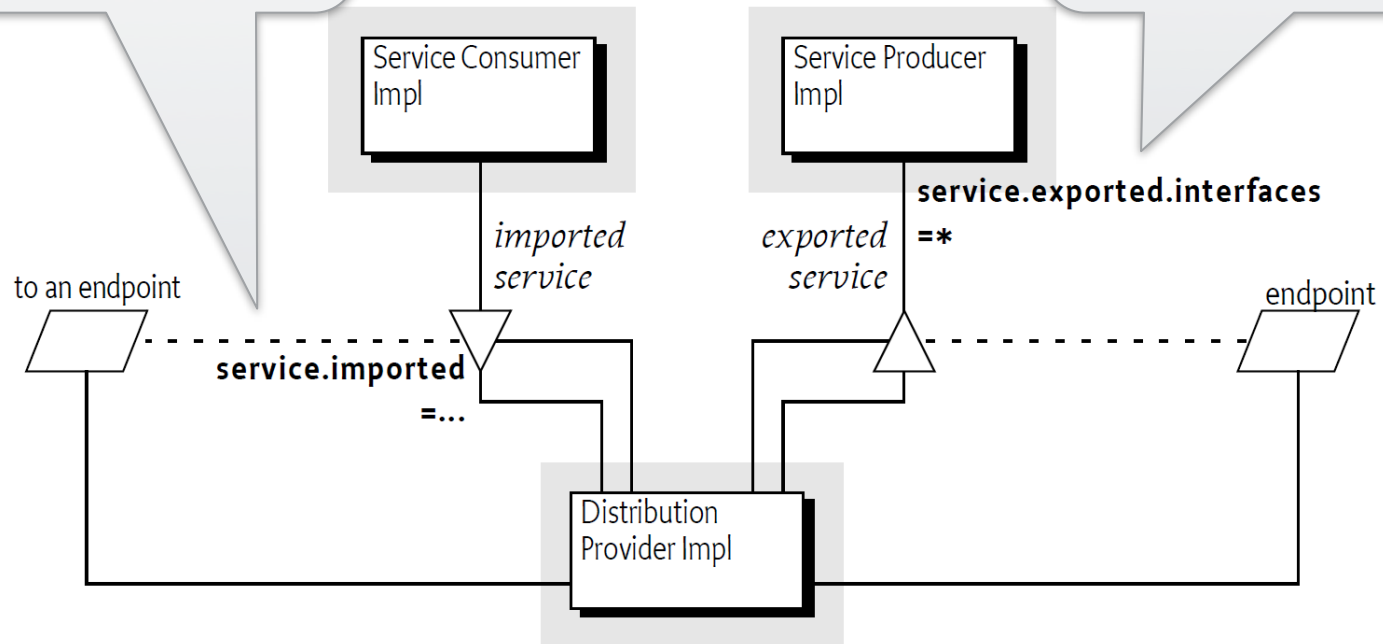
Property-ben a importált szolgáltatás



Remote Services

Property-ben a importált szolgáltatás

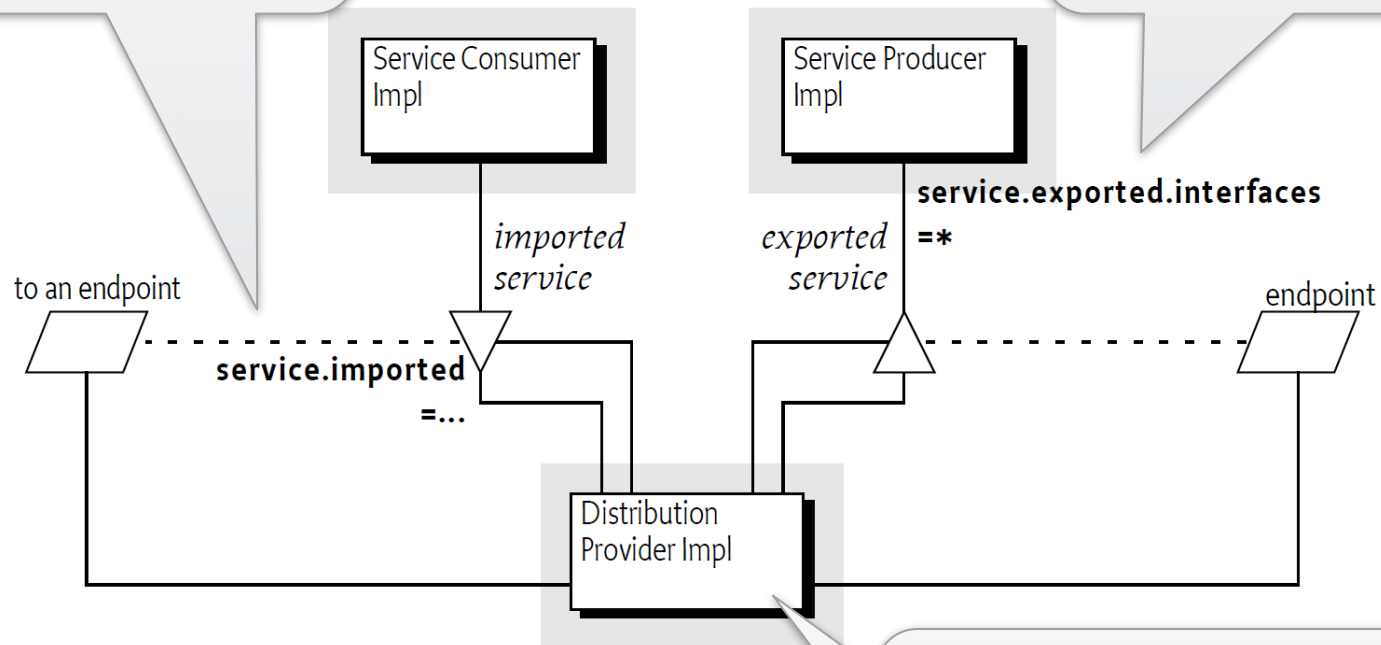
Property-ben a kiexportált szolgáltatás



Remote Services

Property-ben a importált szolgáltatás

Property-ben a kiexportált szolgáltatás



- Átlátszó módon
- Szolgáltatások proxy-ként

Eclipse Equinox

OSGi implementációk

- Open Source
 - **Eclipse Equinox** (<http://www.eclipse.org/equinox/>)
 - Apache Felix (<http://cwiki.apache.org/FELIX/index.html>)
 - Knopflerfish (<http://www.knopflerfish.org/>)
 - ProSyst mBedded Server Equinox Edition (http://www.prosyst.com/products/osgi_se_equi_ed.html)
- Fizetős:
 - ProSyst (<http://www.prosyst.com/>)
 - Knopflerfish Pro (<http://www.gatespacetelematics.com/>)

OSGi, Eclipse és Equinox viszonya

- OSGi szövetség
 - Nyílt szabvány
 - Komponens alapú leírások
 - Egyre szélesebb alkalmazási kör (mobil, szerver, desktop, vállalati, beágyazott)
- Eclipse
 - RCP használata nagyon megnőtt
 - Eclipse runtime lecserélése nyílt szabványra →
 - Eclipse 3.0 óta OSGi-ra épül
- Equinox
 - Eclipse OSGi implementációja (3.3 óta)
 - „Szerver oldali eclipse” → több ennél
 - OSGi 4.0 és 4.1 referencia implementáció

Equinox „Runtime”

- OSGi 4.1 ref implementáció + Eclipse Extension mechanizmusa!
- Több mint szerver oldali Eclipse! (eRCP, RCP, server side,...)
 - Extension és extension point definíciók
 - Erős support SDK oldalon
 - Add-on:
 - Admin
 - Security
 - Application container
 - ...

Hol használják

❖ Eclipse IDE és

eRCP
Nokia
Sprint

NASA
JPMorgan
Lotus
Jazz
SAS
Swiss Rail
Daimler

Rational Suite
Borland
BEA
Jazz

WAS
BEA
Jazz
Spring

Embedded

Rich Client

Tooling

Server

Equinox mint szerver

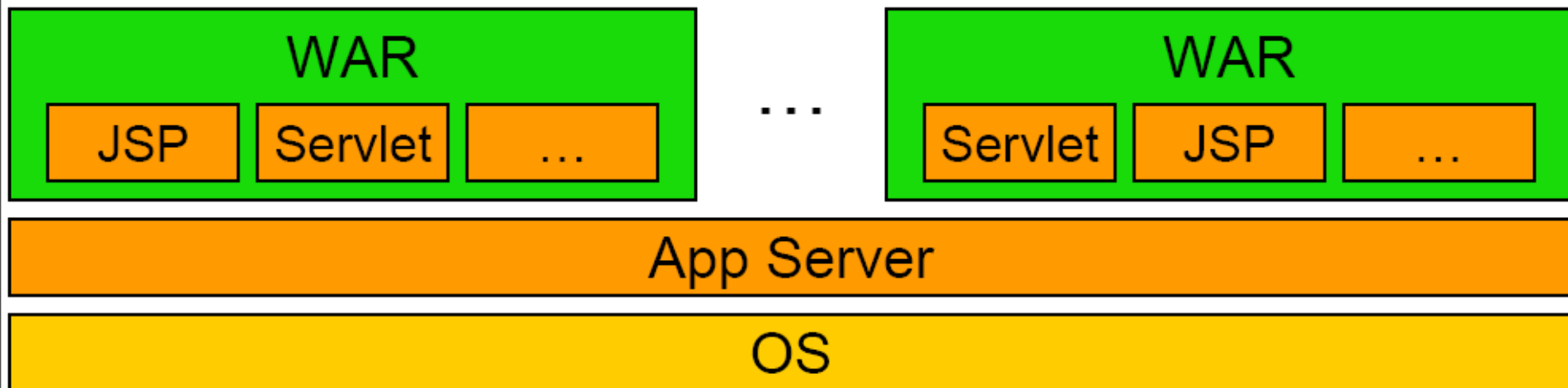
- Equinox runtime + szerver oldali add-ons
 - HTTP service/registry
 - Jetty: beépített pehelysúlyú webszerver
 - Integrációs bundle-ok (ServletBridge=Servlet/JSP, stb.)

Equinox Szerver oldali variánsok

- Tradicionális App szerver
- Equinox beágyazva egy App szerverbe
- Tiszta Equinox
- Equinox Equinoxba ágyazva
- App szerver Equinoxban

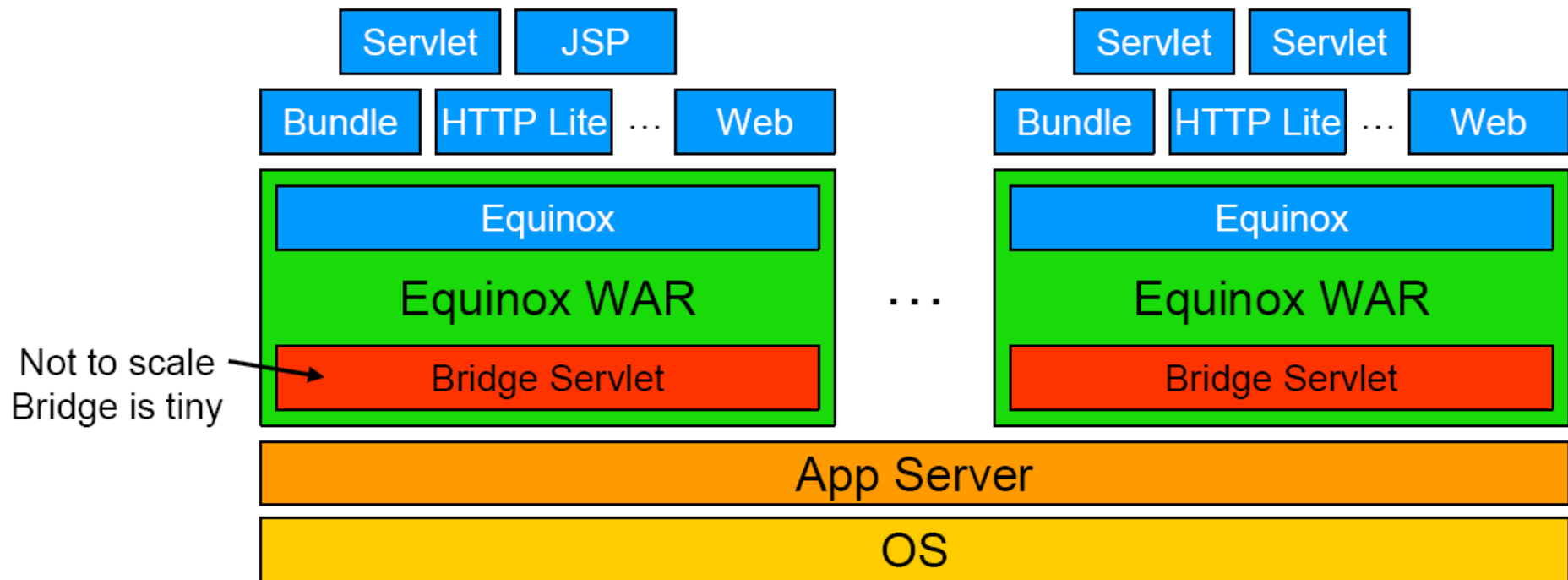
Tradicionális APP szerver

- Funkciók war-okba csomagolva
- App menedzsment teljes war cserével/updatettel/stb.
- Izolált applikációk



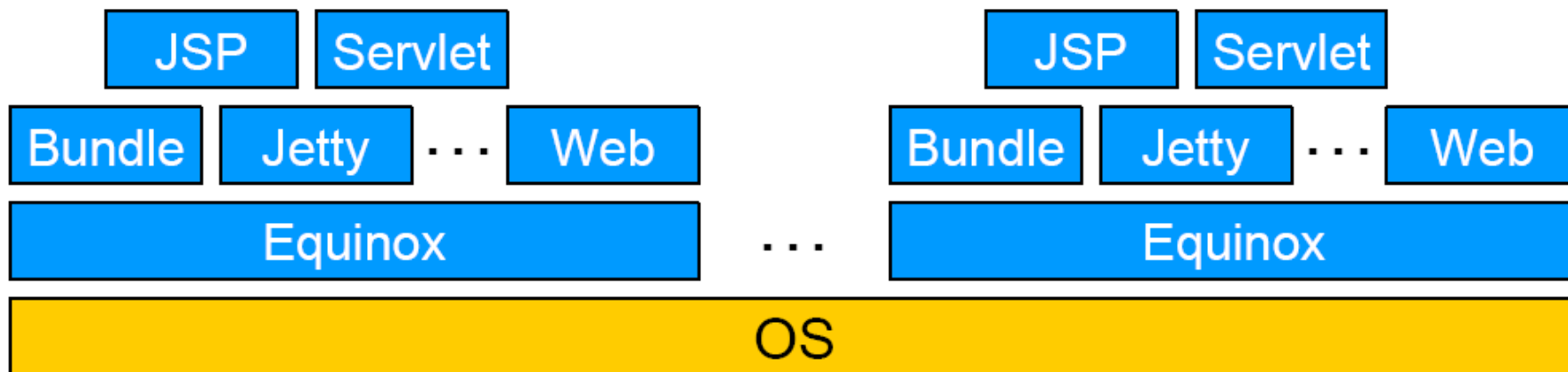
Equinox beágyazva egy App szerverbe

- „Bridge Servlet” integráció
- Izolált applikációk
- Eddigi infrastruktúra is felrakható



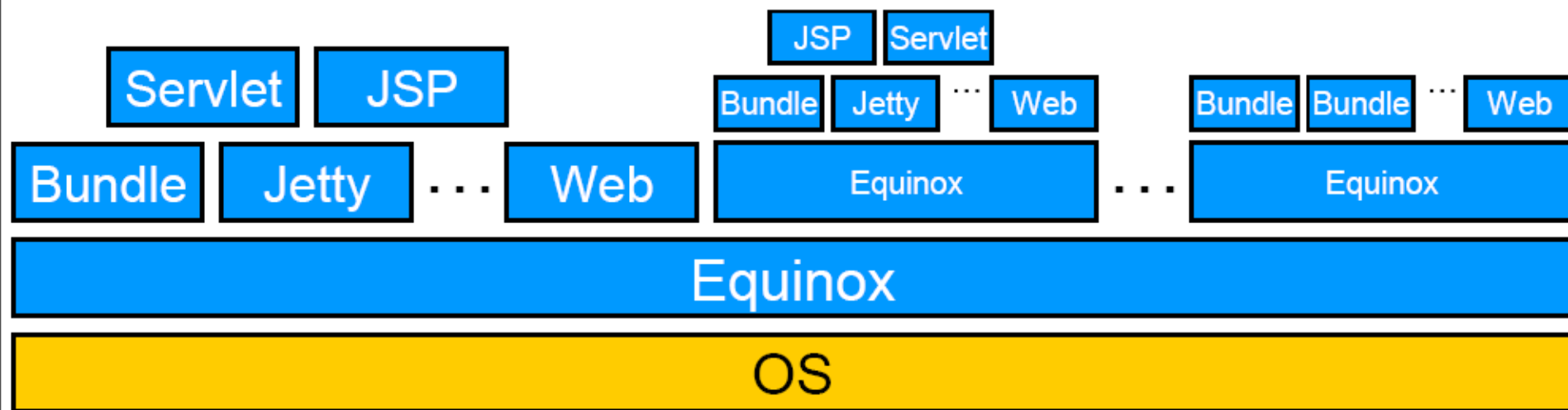
Tiszta Equinox

- Közvetlen Equinox futtatás
- Izolált Processzek!
- HTTP Jetty-n keresztül
- App install/update/... bundle kezelésen keresztül!



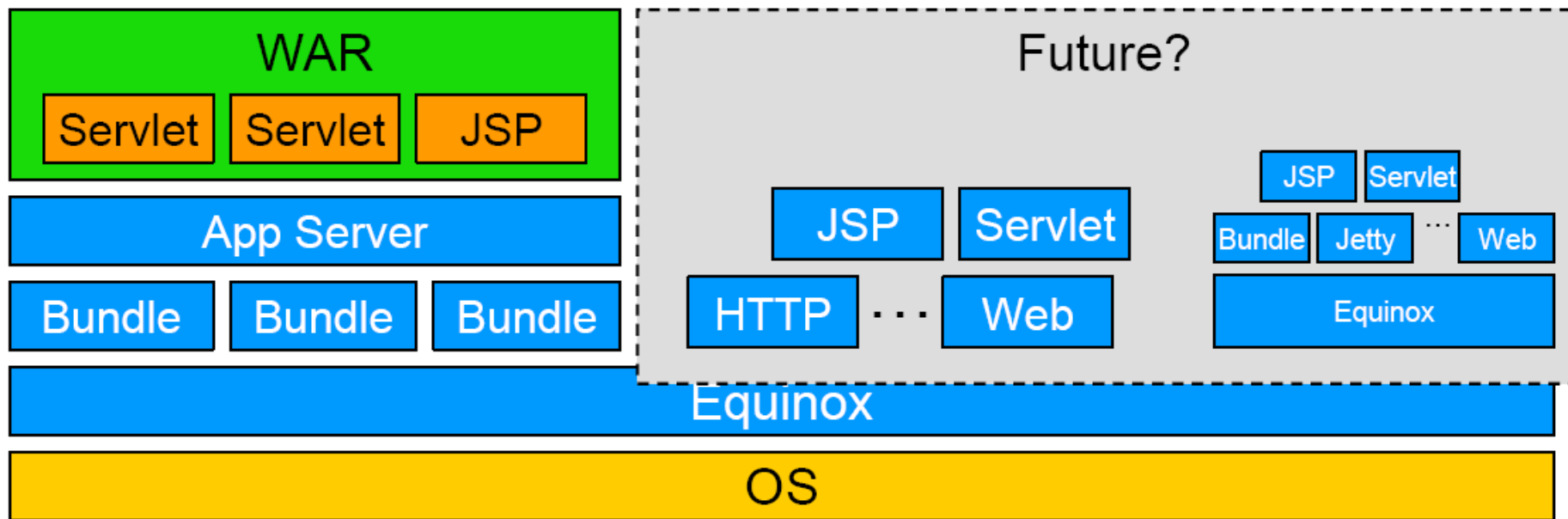
Equinox Equinoxba ágyazva

- Közvetlen Equinox futtatás ebbe ágyazva a többi Eq-t
- Beágyazott Equinox szintű izoláció
- App install/update/... bundle kezelésen keresztül
- Rekurzív infrastruktúra leképezés → óriási méretek kezelése



App szerver Equinoxban

- App szerver mint egy bundle
- Teljeskörű dinamikus szerver konfiguráció bundle kezelésen keresztül!
- Minden eddigi megoldást képes futtatni!



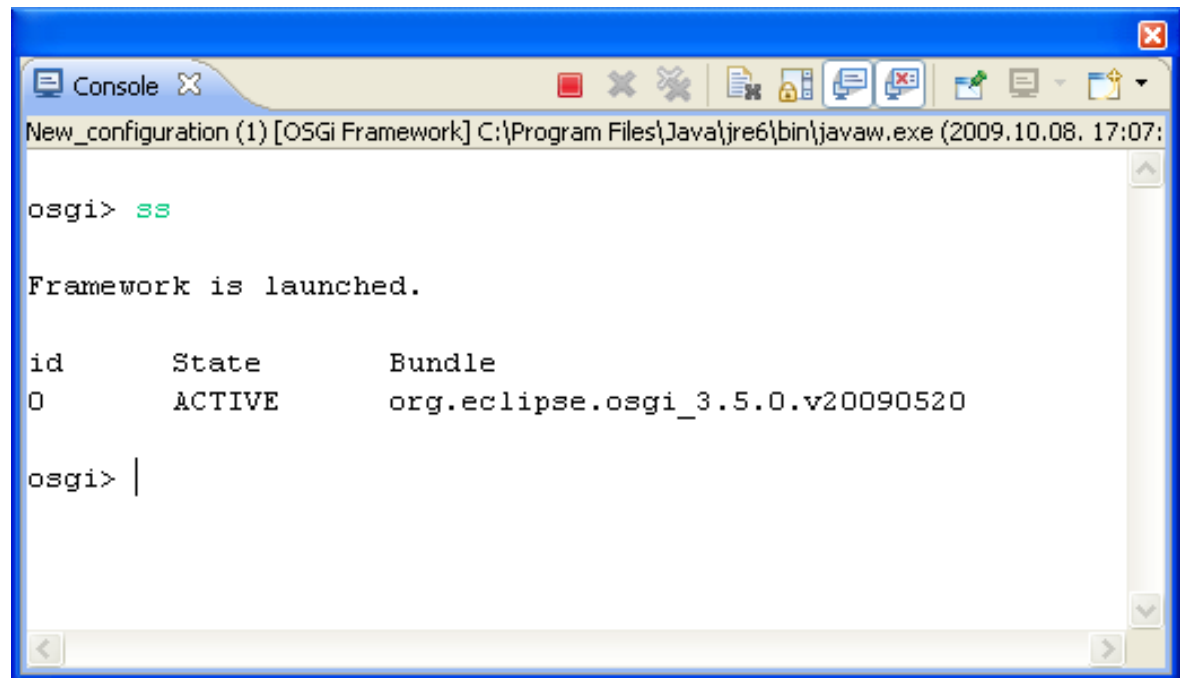
Előnyei

- Az egyes komponenseknek inkrementális frissítése
- Többszörös instance futtatás párhuzamosan
→ HA / nagy sebesség
- Management Külön-Külön és akár együttesen is (szinteken)
- Más-más igényekhez másként paraméterezett instancek
- Class loading teljesítmény növekedés
- Komponens megosztás server és kliens között
 - „Hálózat nélküli” mód támogatása

OSGi konzol

OSGi konzol

- OSGi prompt ~ Hasonló egy DOS v. Bash prompt-hoz
- Eclipse támogatás
 - Console view
 - Highlight



The screenshot shows the Eclipse IDE's Console view. The title bar reads 'Console'. The text inside the console window is as follows:

```
New_configuration (1) [OSGi Framework] C:\Program Files\Java\jre6\bin\javaw.exe (2009.10.08. 17:07:

osgi> ss

Framework is launched.

id      State      Bundle
0       ACTIVE    org.eclipse.osgi_3.5.0.v20090520

osgi> |
```

The 'ss' command has been executed, resulting in the framework being launched and a table of bundles being displayed. The bundle 'org.eclipse.osgi_3.5.0.v20090520' is shown with an 'ACTIVE' state.

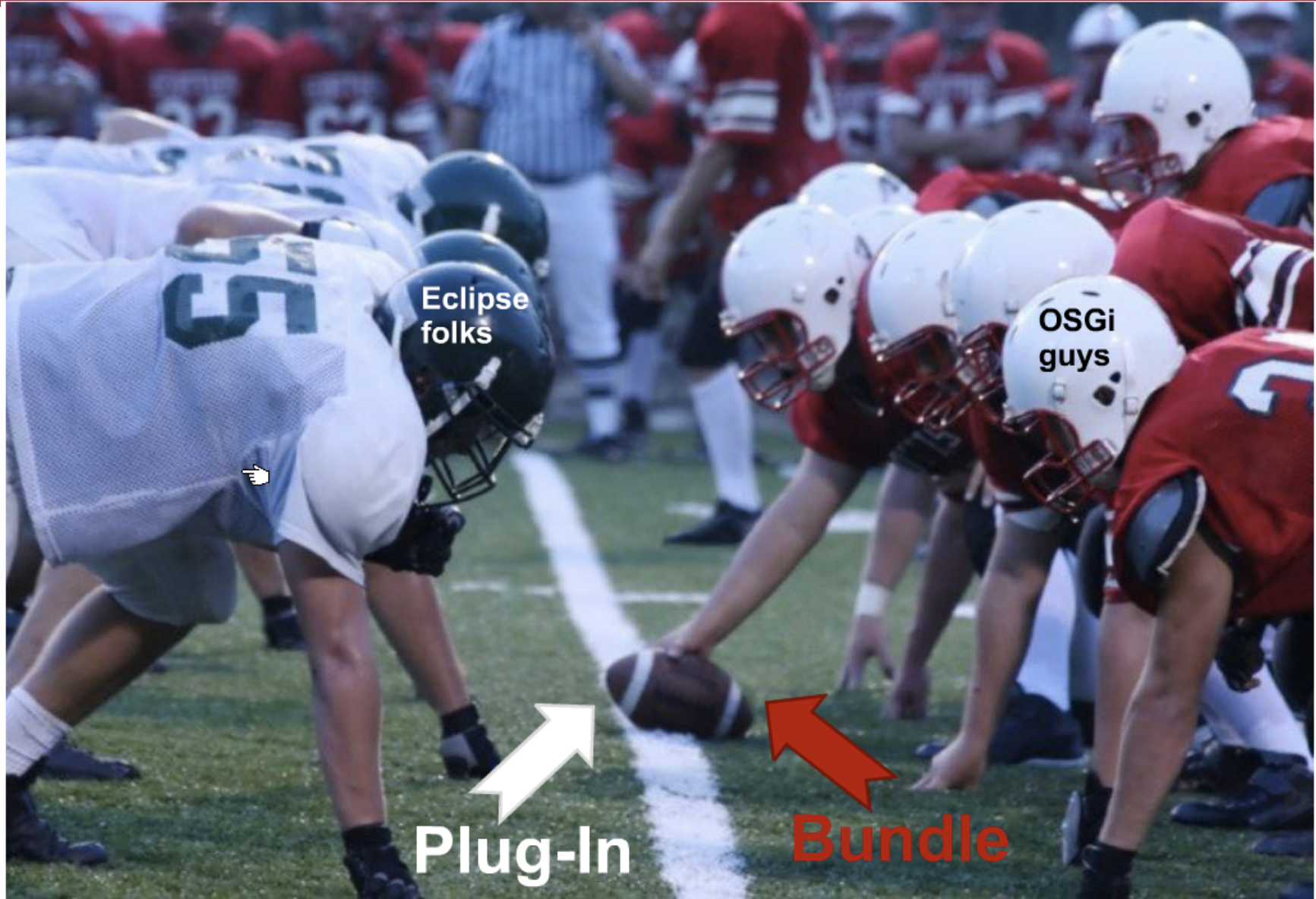
OSGi konzol - parancsok

■ Hasznos parancsok

- ss: kilistázza az telepített bundle-okat.
- start <id>: elindítja a megadott azonosítójú bundle-t
- stop <id>: leállítja a megadott azonosítójú bundle-t
- install file:<path>: telepíti a megadott „bundle”-t
- uninstall <id>: eltávolítja a megadott bundle-t
- update <id>: frissíti a megadott bundle-t
- services <filter>: kilistázza a futó szolgáltatásokat
pl.: `osgi> services (objectClass=*HelloService)`
- shutdown: a futó osgi framework leállítása
- close: shutdown és exit
- exit: ~ System.exit

Eclipse vs. OSGi

Bundle vs plug-in



Eclipse extensions vs. OSGi services

- Eclipse kiterjesztés
 - Egy kiterjesztési pont, amit bárki kiterjeszthet
 - UI kiegészítések (túl kicsik OSGi szolgáltatáshoz)
 - Nem kód jellegű kiegészítések
 - Témák megadása
 - Command framework jelentős része
- OSGi szolgáltatás
 - Szolgáltatásokat bárki definiálhat
 - Szolgáltatásokat bárki használhat
 - Nagyon dinamikus framework
 - Laza csatolás

Eclipse extensions vs. OSGi services

	Extensions	Services	(Declarative Services)
Mit regisztrálunk	XML deklarációk	Java Objektumok	Java Objektumok (Proxy az első tényleges használatig)
Hogyan regisztrálunk	A plugin.xml összes extension -je <i>automatikusan</i>	A BundleContext API-ját használva <i>manuálisan</i>	A Service-Component leírókban lévő összes szervíz <i>automatikusan</i>
Hogyan használjuk fel	Lekérhetőek az Extension point ID -vel	Lekérhetőek az interface nevével és property filterekkel.	Ua., mint services esetében, de a szolgáltatást az SCR állítja be a megadott metódusokkal.
Milyen a kapcsolat számossága	<i>One-to-many</i> : egy Extension point -nak több Extension -je is lehet, de minden Extension -höz pont egy Extension point tartozhat	<i>Many-to-many</i> : egy szolgáltatást többen használhatnak és egy felhasználó több szolgáltatást használhat	Ua., mint services
Mikor töltődik be	Az Extension deklarációk indulásnál töltődnek be, de az	A megvalósító osztályt a betöltés előtt létre kell	A proxy elemnek köszönhetően a szolgáltatás csak akkor

Required-Bundle vs. Import-Package

■ Required-Bundle

- Eclipse felhasználók
- Minden package-t beimportál
 - amit a bundle kiajánl
 - re-exported package-ket is
- Erősebb csatolás

■ Import-Package

- OSGi felhasználók
- Csak a megadott package-ket importálja be
- Lazább csatolás
- Kevesebb látszik