# Software Distribuït - T1 - Introducció a les aplicacions distribuides.

#### Eloi Puertas i Prats

Universitat de Barcelona Grau en Enginyeria Informàtica

14 de febrer de 2014



# Exemples sistemes distribuïts

... Digueu-hi la vostra!



# Exemples sistemes distribuïts

- Cercadors d'internet.
- Massively multiplayer online games (MMOGS)
- Casinos Online
- Skype
- Whatsapp





### Tendències en sistemes distribuïts

- Accés universal a Internet.
- Mobile & ubiquitous computing.
- Sistemes Multimedia Distribuïts.
- Software As A Service & hardware services! Cloud Computing





# Reptes (i necessitats) dels sistemes distribuïts

- Heterogeneïtat: Protocols, middleware, codi portable.
- Accessibilitat: publicació de protocols, APIs, documentació.
- Seguretat: Encriptació de dades sensibles.
- Escalabilitat: Recursos HW i SW, controlar pèrdua de rendiment.
- Tractament d'errors: Detectar, tolerar, recuperar-se d'errors.
- Oncurrència: Accés simultani a recursos compartits.
- Transparència: Accés i localització del sistema transparent a l'usuari.
- Qualitat del servei: Confiabilitat, seguretat i rendiment.



# Arquitectura dels sistemes distribuïts

**Sistema distribuït**: ordinadors independents interconnectats que col.laboren per a realitzar una mateixa tasca.

Arquitectures: rols i responsabilitats

- Model d'ordinador central amb terminals
- Model client/servidor
- Model Peer2Peer





### Model d'ordinador central amb terminals

- Arquitectura:
  - Ordinador central: dades + lògica
  - Terminals: presentació
- Problemes:
  - Saturació del servidor
  - Usuaris sense servei de transaccions



### Model client/servidor

#### Arquitectura:

- Servidor: dades + lògica
- Clients: presentació + lògica
- Connexió sempre comença pel client, que ha de conéixer al servidor
- Rols asimètrics.

Un servidor pot ser al seu torn, un client d'un altre servidor

#### Problemes:

Problema d'escalabilitat.



### Model Peer2Peer

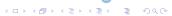
#### Arquitectura:

- Peer: Node de computació/comunicació.
- Rols simètrics, no hi ha diferència entre client i servidor

#### Problemes:

- Més complexitat: Lògica i dades totalment distribuïdes
- Localització del servei (Bootstrap problem)





#### Middleware

*Middleware:* Donar un nivell més alt d'abstracció per al desenvolupament d'aplicacions distribuïdes

- RPC
- Objectes Distribuïts RMI
- Components Distribuïts
- Serveis web



# Crides a procediments remots

#### RPC (Remote Procedure Calls)

- Programació no orientada a objectes:
  - crides a procediments que s'executen en altres màquines
- Programació Orientada a Objectes: [finals '80]
  - invocació de mètodes a objectes que es troben en un altre servidor
  - Objectes Distribuïts: tots els objectes són alhora clients i servidors



# Tecnologies d'objectes distribuïts

- RMI (Remote Method Invocation):
  - SO: independent
  - Llenguatge: dependent (Java)
- DCOM (Distributed Component Object Model):
  - SO: dependent
  - Llenguatge: independent (C, C++, VB...)
- CORBA (Common Object Request Broker Architecture):
  - SO: independent

11/19

Llenguatge: independent



### Components distribuïts

#### Finals '90: objectes es transformen en components

- Servidor d'aplicacions: contenidor de components
  - persistència automàtica
  - transaccions
  - cicle de vida
  - ...
- Plataformes de components distribuïts:
  - J2EE (Java Platform Enterprise Edition): JSP/Servlets
  - Windows DNA (Windows Distributed interNet Applications Architecture): ASP



#### Serveis web

#### Actualment:

#### Serveis web:

- Comunicacions: HTTP
- Crides remotes: SOAP o XML-RPC
- Tec.Obj.Distribuïts: WSDL (Web Services Description Language)
- Directori de serveis: UDDI (Universal Description Discovery and Integration)



# Patrons Arquitectures multi-capa

Pas d'aplicació *monolítica* a aplicació **modular** estructurada en **capes lògiques** de:

- usuari: presentació, navegació, interacció.
- negoci / lògica: gestió d'events sobre les dades, lògica del negoci, validació.
- dades: CRUD (Create Retrieve Update Delete), integritat ER.
- sistema: accés a fitxers, autentificació, errors, logs...



# Capes físiques: Single-Tiered

Single-Tiered: 1 màquina amb 1 aplicació

• -: poc escalable, poc flexible, mono-usuari



# Capes físiques: 2-Tier

#### 2-Tier: client/servidor

- +: multi-usuari, un sol missatge per invocar una operació.
- -: capa negoci dividida entre el client i el servidor.



# Capes físiques: 3-Tier

#### **3-Tier**: client / negoci / dades

- -: augment de missatges i complexitat
- +: cada màquina s'encarrega d'una capa



# Capes físiques: N-Tier

N-Tier: n-client / n-negoci / n-servidor

• +: molt escalable



# Tipus de client

- Gruixut (Thick client): 1/2 capes, bon maquinari ja que contenen molta funcionalitat.
- Prim (Thin client): capa lògica d'usuari, interfície senzilla d'introducció/extracció de dades (ex. navegador web).
- Intermig (*Pump Client*): capa lògica d'usuari + part capa de negoci (validacions simples).

