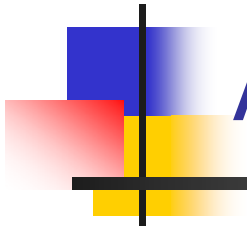
Universidad Zaragoza



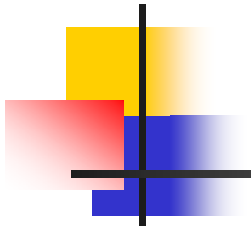
Agentes Móviles

Ingeniería del Software II

Curso 2010/2011

Sergio Ilarri Artigas

silarri@unizar.es



Índice

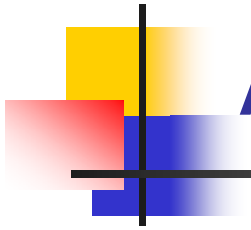
- Agentes vs. Objetos
- Caracterización de la Movilidad
- Agentes Móviles: definición, ventajas, aplicaciones
- Movilidad: Fuerte y Débil
- Plataformas de Agentes Móviles
- SPRINGS:
  - *Proxies* dinámicos
  - Problema de *Livelock*

¿Diferencia entre objeto y agente?

Agentes y Objetos (I)

Los objetos son pasivos y obedientes

¡Los agentes son autónomos!



Agentes y Objetos (II)

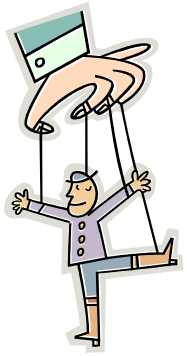
- Ambos encapsulan el estado
- Los agentes, además, encapsulan un comportamiento
  - Los objetos no, dado que no tienen ningún control sobre la ejecución de sus métodos
- Obsérvese la diferencia:
  - *Invocamos* métodos de objetos
  - *Pedimos* a los agentes que ejecuten acciones

Agentes y Objetos (III)

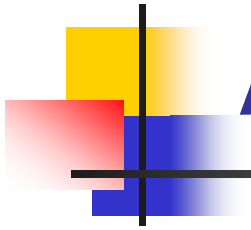
- Es decir:

  - En el caso de objetos el paso de mensajes representa simplemente invocaciones de métodos
  - Los agentes distinguen distintos tipos de mensajes (modelados normalmente como actos del habla –*speech acts*-), pueden usar protocolos complejos de negociación, analizan los mensajes y deciden qué ejecutar

Agentes y Objetos (IV)

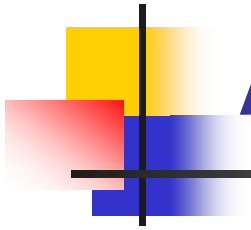


- Los objetos no son proactivos, no tienen capacidad de decisión sobre qué hacer en una situación concreta
- Los objetos no encapsulan el control de ejecución, se asume un control externo
- Los agentes pueden caracterizarse mediante su estado mental
  - Por ejemplo: Creencias, Deseos, Intenciones (BDI)



Agentes y Objetos (V)

- Los agentes también pueden tener una dimensión social
- Con objetos no hay noción de “entorno”
  - Los objetos perciben el mundo sólo a partir de referencias a otros objetos y sus propios atributos
- Debido a todas estas diferencias:
  - OOSE → AOSE
  - OOP → AOP



Agentes y Objetos (VI)

- Sin embargo, hoy en día los objetos pueden aproximarse a la visión de agentes:
  - Los objetos pueden ser activos e incluir uno o varios hilos de ejecución
  - Los agentes normalmente se implementan utilizando objetos activos
  - Podría decirse que ambas tecnologías están convergiendo desde el punto de vista de la Ingeniería del Software
    - Prueba de que la abstracción de agente es útil para desarrollar sistemas complejos

Agentes vs. Agentes Móviles

- Agentes:
  - Inteligencia artificial
  - Son “inteligentes”
  - Pueden moverse o no
- Agentes móviles:
  - Computación distribuida
  - Son móviles
  - Pueden ser inteligentes o no

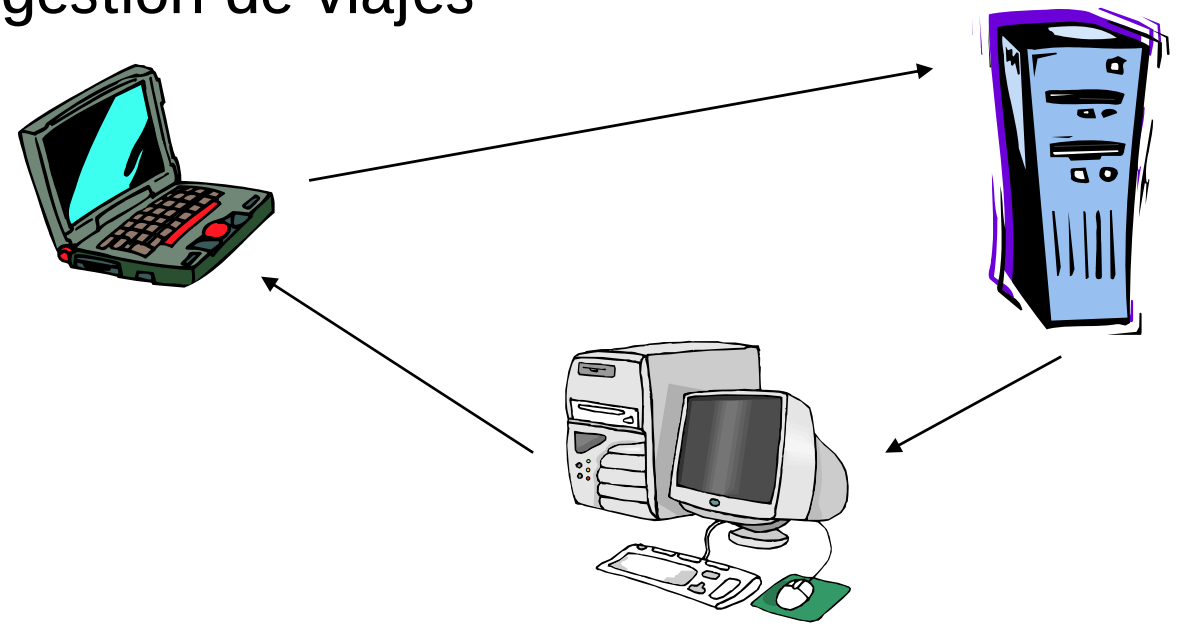
Conflicto y esperada reconciliación

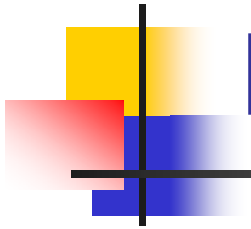
Agentes Móviles: Definición

Agentes software:

- Se mueven de ordenador a ordenador
- A petición del usuario, autónomamente
- Ejemplo: gestión de viajes

Soportan usuarios móviles (asíncronos)





Los Ancestros...

- Frente al paradigma clásico cliente/servidor, surge la idea de movilidad de código

- Código móvil:
  - Evaluación remota
  - Código bajo demanda
  - Procesos móviles

Cinco Formas de Hacer una Tarea

1. Hazla tú mismo, solo

Cinco Formas de Hacer una Tarea

1. Hazla tú mismo, solo
2. Pide a otro que la haga: cliente/servidor

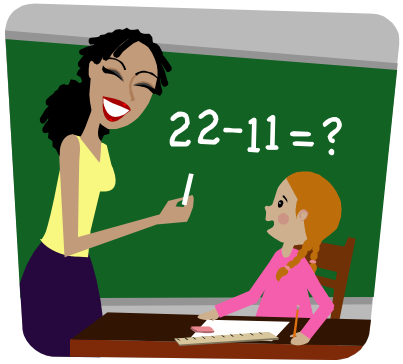
Cinco Formas de Hacer una Tarea

1. Hazla tú mismo, solo
2. Pide a otro que la haga: cliente/servidor
3. Díle a otro la secuencia de pasos a ejecutar: evaluación remota

Sergio Ilarri

Cinco Formas de Hacer una Tarea

1. Hazla tú mismo, solo
2. Pide a otro que la haga: cliente/servidor
3. Dile a otro la secuencia de pasos a ejecutar: evaluación remota
4. Pregúntale a otro cómo se hace: código bajo demanda



Cinco Formas de Hacer una Tarea

1. Hazla tú mismo, solo
2. Pide a otro que la haga: cliente/servidor
3. Dile a otro la secuencia de pasos a ejecutar: evaluación remota
4. Pregúntale a otro cómo se hace: código bajo demanda
5. ¡Voy para allá! ¡Ten todo listo!: agente móvil

Paradigma Cliente/Servidor

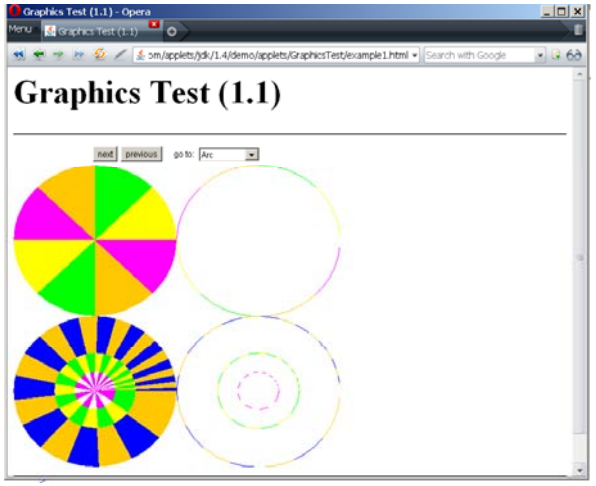
- Aproximación clásica en sistemas distribuidos
- Cliente y servidor estáticos
- El cliente le pide algo al servidor
- El servidor lo hace y le devuelve al cliente un resultado
- Conceptos de lenguajes de programación:
  - *RPC*
  - *RMI*
  - *Servicios web*
- Frente a *C/S* remoto, los agentes móviles
  - Paradigma de diseño alternativo
  - Paradigma de diseño complementario

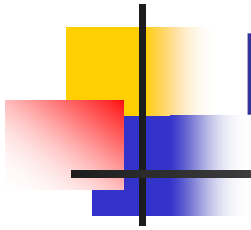
Paradigma de Evaluación Remota

- *Remote Evaluation* (*REV*)
- Extiende la idea de *RPC*
- El cliente envía código al servidor para que lo ejecute
- Ejemplos:
  - El lenguaje *Postscript* para impresoras
  - El envío remoto de trabajos por lotes
  - *NCL* (*Network Command Language)*
  - *SQL* remoto

Paradigma de Código Bajo Demanda

- *Code-on-demand*
- También se envía código… pero del servidor al cliente
- Por tanto, se usan recursos del cliente
- Ejemplo: *applets* de *Java*





Procesos Móviles

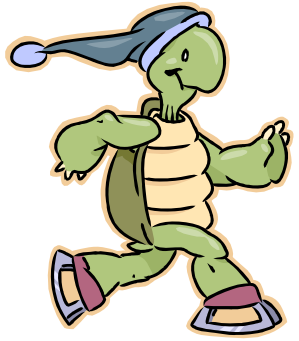
- Área de los sistemas operativos distribuidos (finales de los 80)
- Migración de procesos para balancear carga
- Ejemplo: *Sprite*
  - *Process Migration in the Sprite Operating System*, Frederick Douglis, Technical Report CSD-87-343, University of California at Berkeley (USA),1987.

Procesos Móviles

- Área de los sistemas operativos distribuidos (finales de los 80)
- Migración de procesos para balancear carga
- Ejemplo: *Sprite*
- Ejemplo de técnica de implementación: *checkpointing*
  - *Imágenes periódicas*
  - *Envío de la imagen*

Procesos vs. Agentes Móviles

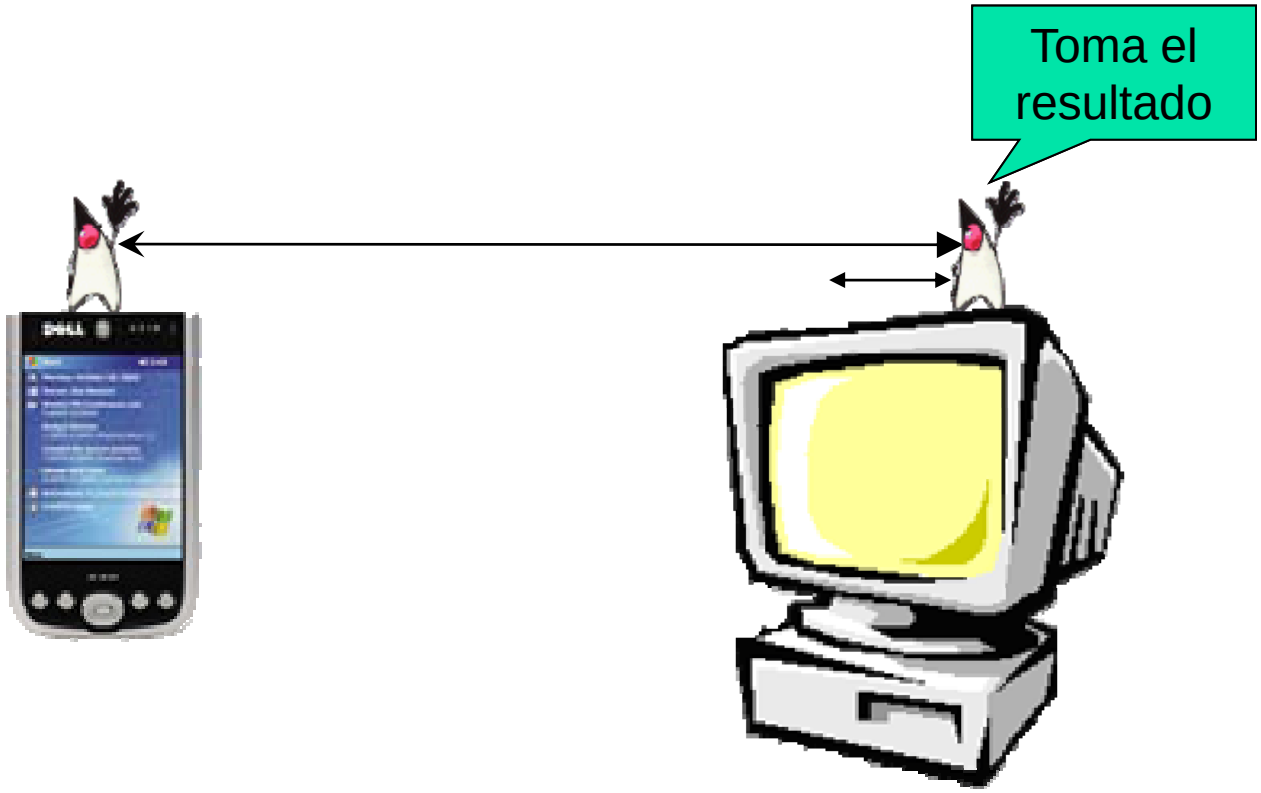
- Frente a los procesos móviles, los agentes móviles:
  - No sólo se mueven por balanceo de carga
    - Acceso a servicios
  - La migración no la inicia el *SO* o *middleware*
    - Agente autónomo
  - No sólo se mueven una vez
    - Migración multi-salto, itinerario



Agentes Móviles

- “Programas” que se ejecutan en cierto conexto de ejecución o *place* y viajan de *place* a *place*
- Capaces de transportarse a sí mismos de un ordenador/dispositivo a otro
- Necesitan una infraestructura para funcionar (una plataforma de agentes móviles)
- Agentes móviles ≠ código móvil

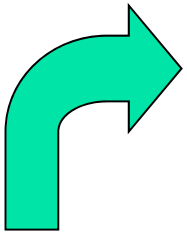
Una Alternativa a C/S



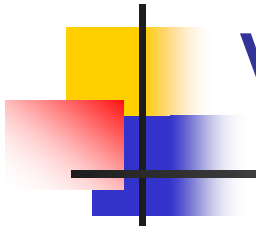
Los agentes móviles pueden reducir el uso de la red y la transferencia de datos

Características Básicas

- Algunas características interesantes de los agentes (móviles)
  - Autonomía: no/mínima interacción con el creador
  - Interoperabilidad: hardware, SO, etc.
  - Reactividad: cambios/eventos del entorno
  - Cooperación: objetivo común
  - "Inteligencia" (especialistas)
  - Movilidad (*code-shipping* vs. *data-shipping*)

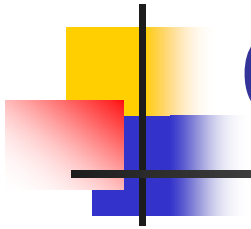


Agentes móviles en particular (sinergia)



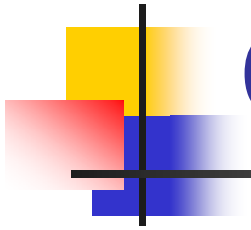
Ventajas

- Evitan instalaciones innecesarias
- “Salvan” la latencia de red: comunicación local
- Pueden encapsular protocolos (BD, etc.)
- Asíncronos/autónomos: perfectos para trabajar en modo desconectado
- Adaptativos
- Reaccionar entorno
- Pueden moverse: balanceado de carga, localidad datos
- Integración de sistemas heterogéneos
- Pueden programarse para que sean robustos / tolerantes a fallos
- Adaptación de *interfaces*



Campos de Aplicación (I)

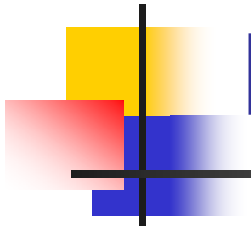
- Algunos entornos donde se han utilizado:
  - Recuperación de información distribuida
  - Procesamiento paralelo
  - Asistente personal
  - Diseminación de información
  - *E-commerce*
  - Gestión de red
    - heterogeneidad, monitorización, personalización, enrutamiento
  - Aplicaciones de *workflow*
  - *Brokering*
  - Entornos distribuidos: entornos móviles, ubicuos, inteligentes, P2P, ...
  - ...



Campos de Aplicación (II)

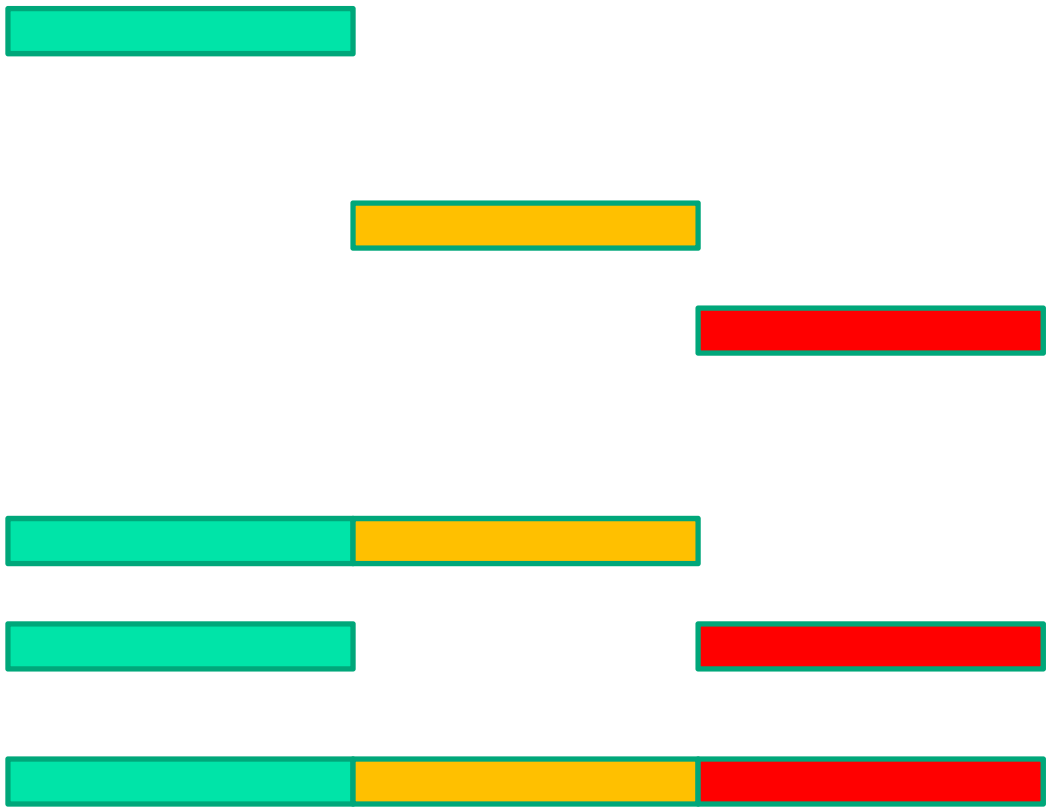
- ¿Y la *killer application*?
  - No hay una aplicación clave
  - ¿Cómo puede defenderse un lenguaje estructurado frente a un lenguaje ensamblador?

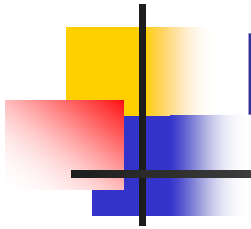
Agentes Móviles en Entornos Móviles



- Apropiados para computación inalámbrica:
  - Desconexiones:
    - Breve conexión: enviar agente a red fija
    - Antes de desconexión: coger agente de red fija
  - Descarga de trabajo del cliente
  - Contribuyen a limitar el uso de las comunicaciones inalámbricas:
    - Reducir los datos a intercambiar por el enlace inalámbrico
    - Evitar interacciones entre cliente y servidor
    - Sólo comunicar agente y resultado

Caracterización de la Movilidad





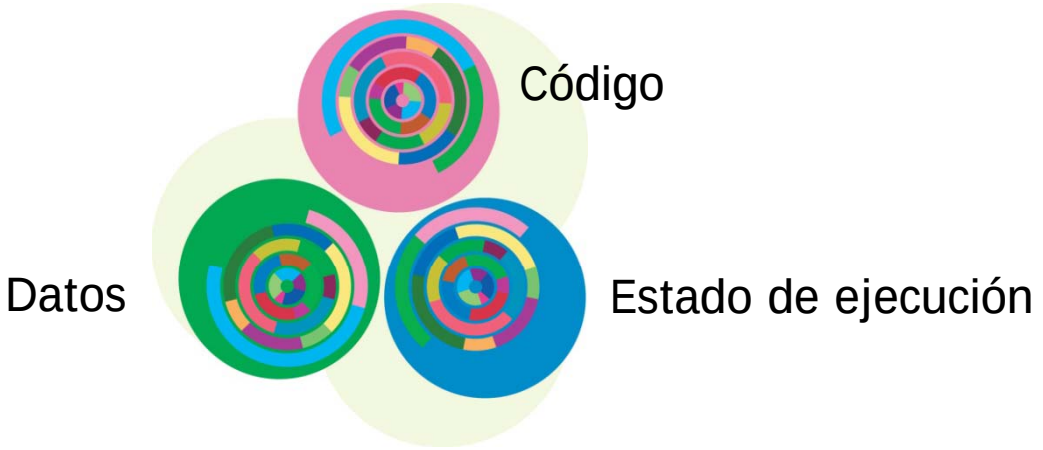
Movilidad: Cómo Funciona

- Los agentes móviles se crean en *places* y viajan entre *places*

- *moveTo(newHost)*
  - Se *interrumpe* la ejecución del *thread*
  - Se *serializa* el código, datos y (quizá) el estado de ejecución del agente (qué estaba ejecutando):
    - Movilidad fuerte y movilidad débil
  - El agente se reconstruye en el *place* destino y continúa su ejecución

Movilidad Fuerte

- Movilidad fuerte
  - Se transfiere el estado de ejecución
  - Plataformas basadas en lenguajes de script
  - Plataformas basadas en Java:
    - Preprocesado para cambiar el código y capturar “a mano” el estado
    - Modificaciones a la máquina virtual de Java



Movilidad Fuerte (Idea)

Plataforma inventada

```
public class AgenteMovFuerte extends Agente
{
    public static void main(String[] args)
    {   String destino = "…";
        System.out.println("En ordenador origen");
        moveTo(destino);
        System.out.println("En ordenador destino");
    }
}
```

Dificultades:

-No es posible con Java estándar

-No todos los recursos son móviles: múltiples *threads*, ficheros abiertos, etc.

Movilidad Débil

- No se transfiere el estado de ejecución
- En el destino siempre se ejecuta un método predefinido o un método de callback
- El programador tiene que almacenar y restaurar el estado de ejecución

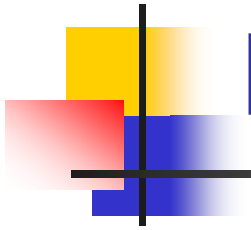
Movilidad Débil (Voyager)

Voyager

```
public class AgenteMovDebil extends Agent {
   public void metodoDestino (Object init) {
      System.out.println (“En ordenador destino”);
   }
   public void move(String destino) {
      Iagent proxy = Agent.of(this);
      System.out.println (“En ordenador origen”);
      proxy.moveTo(destino, “metodoDestino”);
   }
}

public static void main(String[] args) {
   Voyager.startup("8000");
   String serverClass = "AgenteMovDebil";
   AgenteMovDebil ag = new AgenteMovDebil();
   ag.move("tcp://fargo.sdsu.edu:8000");
}
```

Callback



Movilidad Fuerte vs. Débil

- Baumann [1995] señala que la movilidad fuerte es inútil
  - Un movimiento es un gran cambio en la vida del agente
  - El agente ejecuta “fases” y ninguna implica varios *places*
- Cabri et. al. [2000] indica que la movilidad fuerte lleva a un código más limpio
- Conclusión: la movilidad fuerte no es necesaria y es difícil de implementar (aunque podría ser útil)

Plataformas de Agentes Móviles

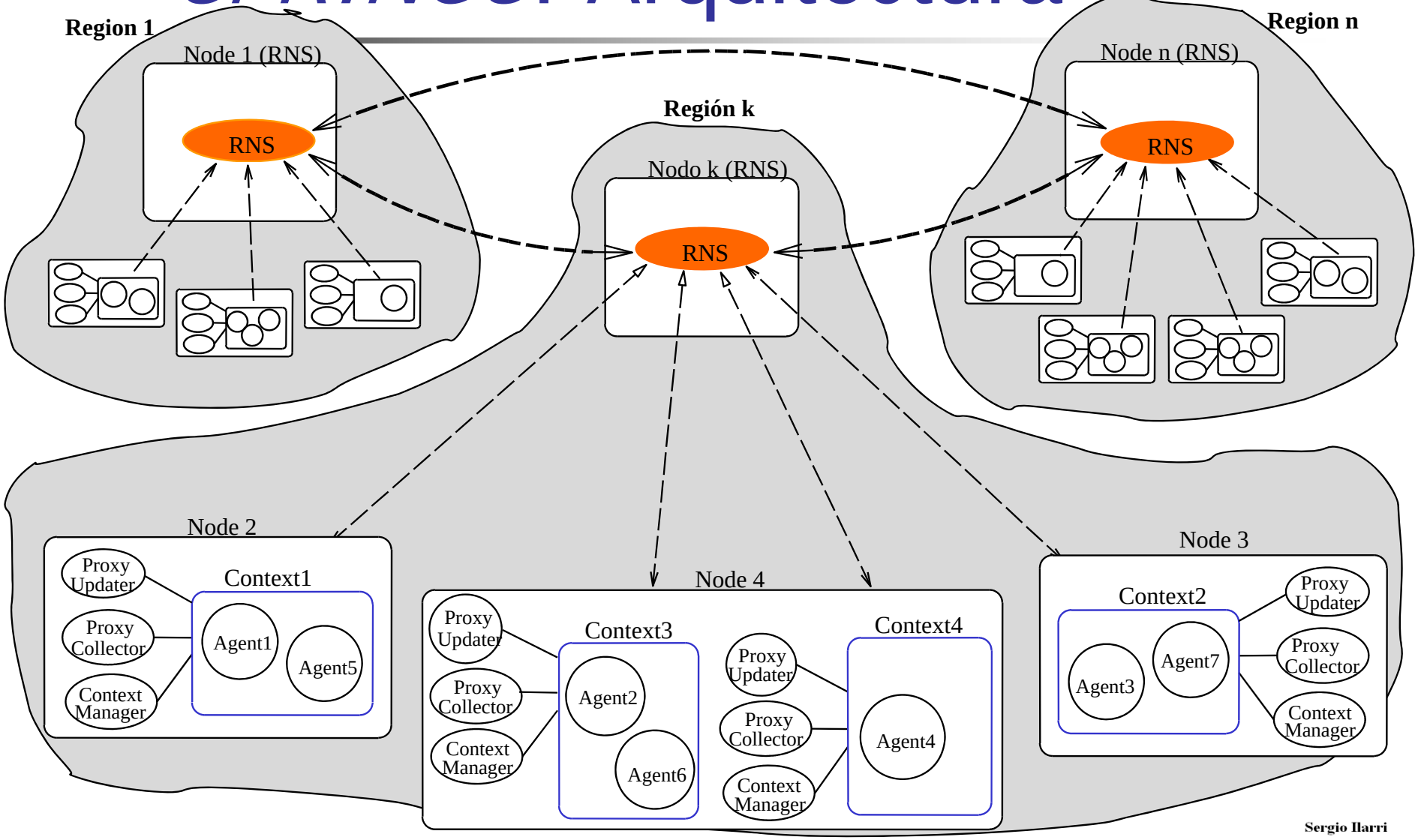
- *Aglets*
- *Voyager*
- *Jade*, *Tracy*, *Mole*, *SeMoa*, ...
- Viejas: *Grasshopper*, *Tryllian*, *...*
- Todas presentan problemas...
  - Por ejemplo, de escalabilidad y concurrencia
  - Por ello, desarrollamos SPRINGS

SPRINGS

- Escalable, mucho mejor rendimiento (3000 agentes en *test* exigente)
- Transparencia de localización:
  - Llamadas: proxies dinámicos
    - *callAgentMethod(“MovingAgentExample”,”go” , args);*
  - Movimientos: *callbacks*
    - *moveTo("C2","end");*
- Reintentos automáticos
- Prevención de *livelock*

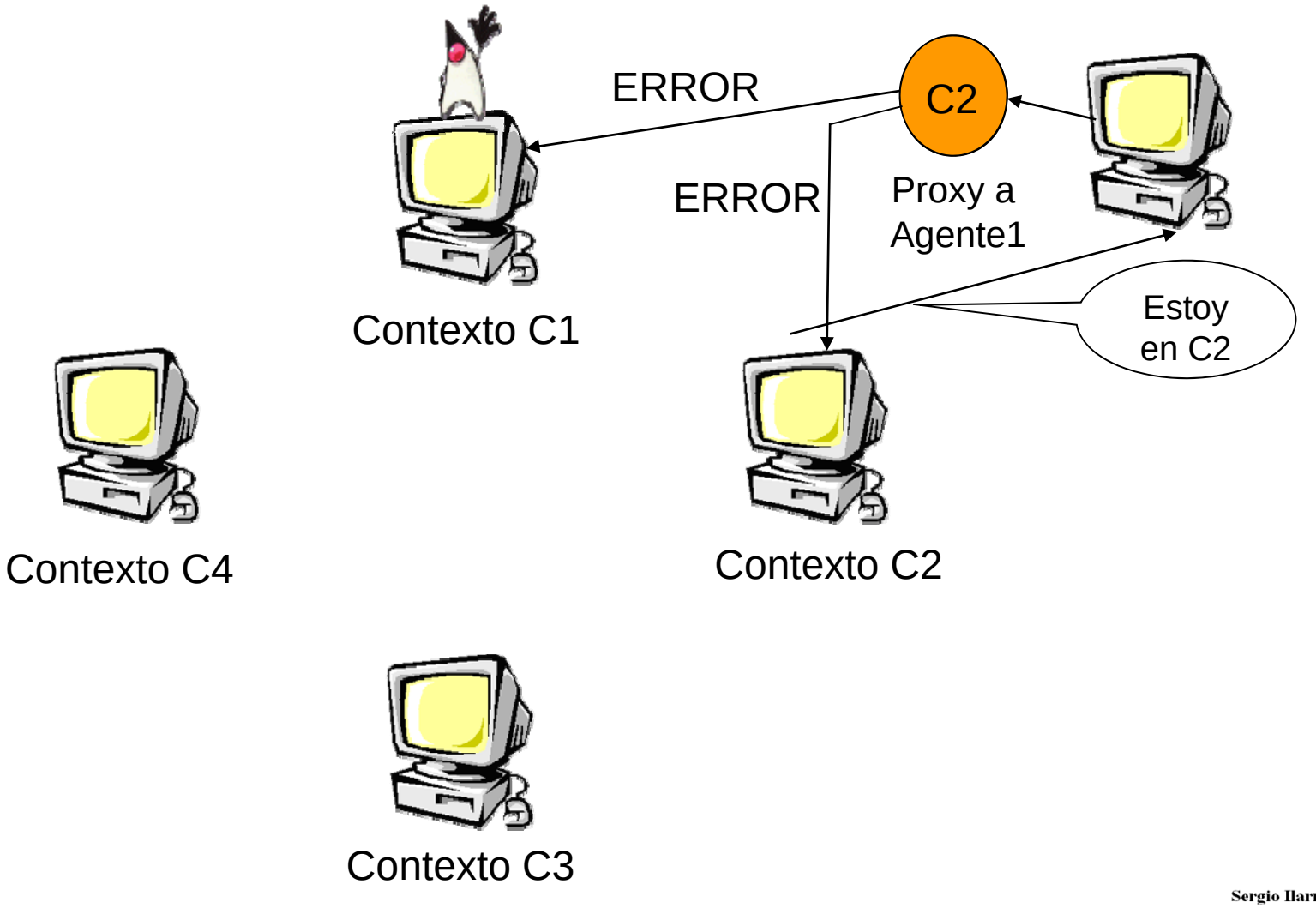
http://sid.cps.unizar.es/SPRINGS/

SPRINGS: Arquitectura

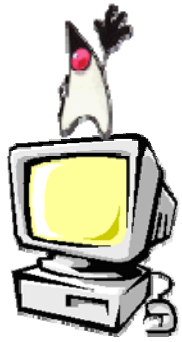


Sergio Ilarri

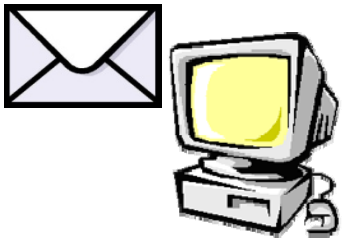
Problema de *Livelock*



Problema de *Livelock*



Contexto C1



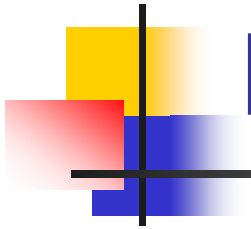
Contexto C4

Contexto C2

Contexto C3

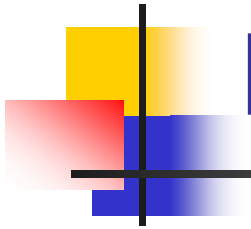
En SPRINGS

-Primero se actualizan los proxies, luego se renuda el agente
-Se retardan agentes muy rápidos



Referencias

- *Seven Good Reasons for Mobile Agents*, Danny B. Lange and Mitsuru Oshima, Communications of the ACM, Volume 42 , Issue 3 (March 1999), pp. 88-89, ACM Press, ISSN:0001-0782, 1999
- *Wireless Computational Models: Mobile Agents to the Rescue*, C. Spyrou, G. Samaras, E. Pitoura, and P. Evripidou, DEXA99 International Workshop on Mobility in Databases and Distributed Systems, August 1999, pp 428-433, IEEE Computer Society
- *SPRINGS: A Scalable Platform for Highly Mobile Agents in Distributed Computing Environments*, S. Ilarri, R. Trillo and E. Mena, 4th International WoWMoM 2006 workshop on Mobile Distributed Computing (MDC'06), Buffalo, New York (USA), IEEE Computer Society, ISBN 0-7695-2593-8, pp. 633-637, June 2006
- *Mobile Agents: Basic Concepts, Mobility Models, and the Tracy Toolkit*, Peter Braun and Wilhelm R. Rossak, Morgan Kaufmann Pub., 464 pages, ISBN *1558608176*
- *Comparison and Performance Evaluation of Mobile Agent Platforms*, R. Trillo, S. Ilarri and E. Mena, The Third International Conference on Autonomic and Autonomous Systems (ICAS'07), Athens, Greece, IEEE Computer Society, ISBN 978-0-7695-2859-5, June 2007.



Fin

Gracias por vuestra atención