

# Contenidos

- ⇒ Importancia de la comunicación entre agentes
- ⇒ Modelos de comunicación
- ⇒ Teoría de los actos del habla
- ⇒ Organizaciones de estandarización
  - ✗ OMG, KSE y FIPA
- ⇒ Lenguajes de comunicación de agentes (ACLs)
  - ✗ KQML y FIPA-ACL
- ⇒ Protocolos de Interacción
- ⇒ Desarrollo de Ontologías
- ⇒ Práctica 2002-2003
  - ✗ ACL, Protocolo de Interacción, Ontología y Sockets en Java

# Comunicación entre Agentes

- ⇒ **La comunicación entre agentes es la llave para obtener todo el potencial del paradigma de agentes**
  - ✖ Al igual que el desarrollo del lenguaje humano fue la llave para el desarrollo de la inteligencia y de la sociedad
- ⇒ Los agentes emplean un lenguaje de comunicación (*ACL - Agent Communication Language*) para comunicar información y conocimiento
- ⇒ Las sociedades de agentes pueden realizar tareas que los agentes individuales no pueden
- ⇒ Los sistemas multiagente (*MAS*) están orientados a la resolución distribuida de problemas
  - ✖ Sólo es posible si los agentes tienen la capacidad de comunicación sobre la que establecer estrategias de cooperación

# Modelos de Comunicación (1)

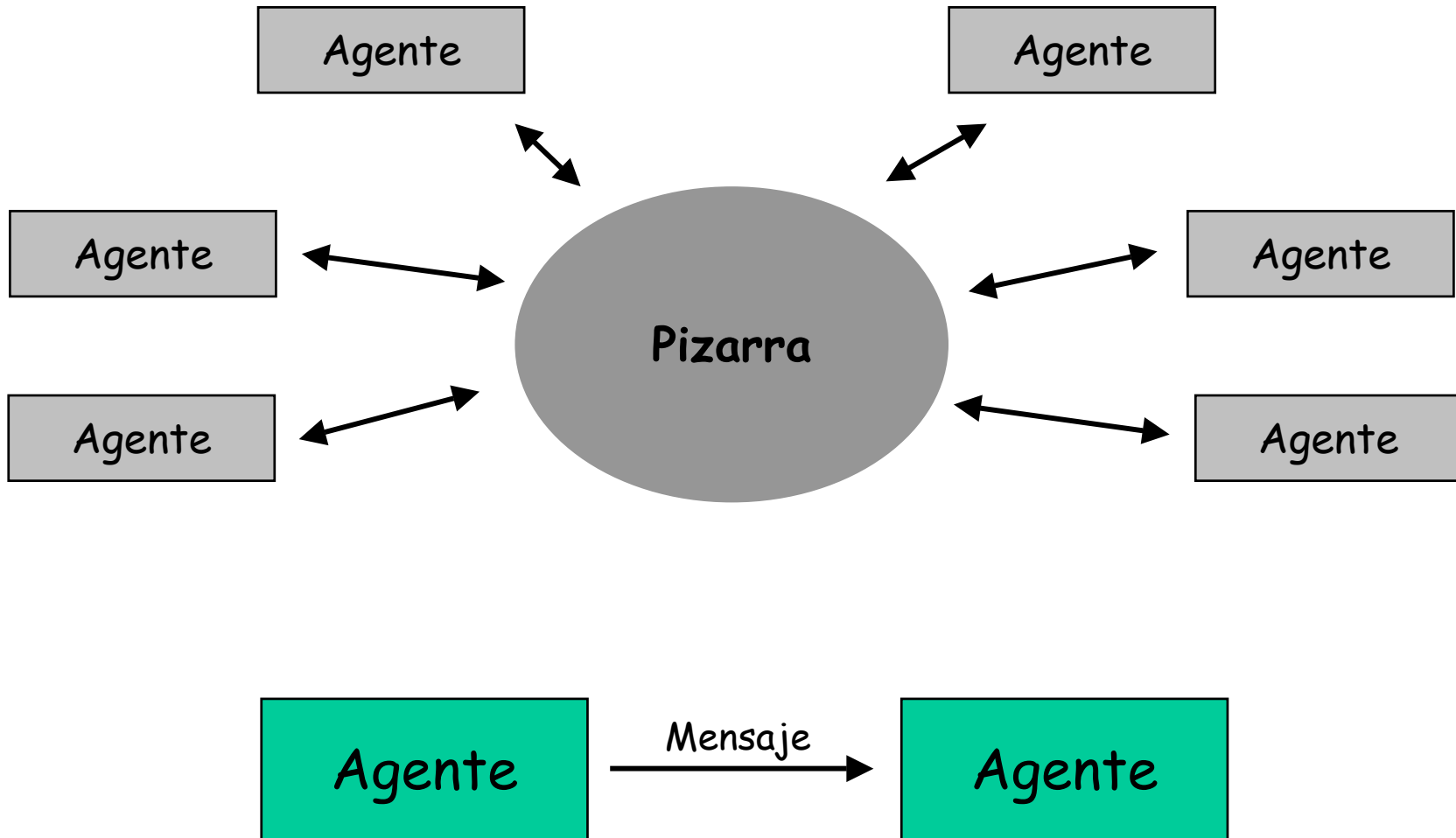
## ⇒ Arquitecturas de Pizarra

- ✗ Zona de trabajo común en la que se encuentra la información a compartir
- ✗ No hay comunicación directa entre los agentes
- ✗ Pueden existir agentes con tareas de control específicas
- ✗ Pueden existir varias pizarras
- ✗ Inconveniente: comunicación centralizada

## ⇒ Paso de Mensajes

- ✗ La comunicación se establece directamente entre dos agentes (emisor y receptor)
- ✗ Pueden existir agentes facilitadores
- ✗ Ventaja: flexibilidad

## Modelos de Comunicación (2)



## Comunicación tipo Vulcano (1)

- ⇒ Vulcano es el planeta de Spock (*Star Trek*)
- ⇒ Esta comunicación consiste en la transmisión de pensamientos o conocimiento de una mente a otra
  - ✘ "Cuando conectas tu cerebro al mío tu sabes lo que yo sé"
- ⇒ Utilizada en la comunicación máquina-máquina
  - ✘ Montar nuevos discos en una máquina
  - ✘ Formularios EDI permiten a un programa rellenar datos en las variables de otro programa
  - ✘ Las BD distribuidas

## Comunicación tipo Vulcano (2)

⇒ No sirve para modelar la comunicación humana por:

1. Mis pensamientos no son los tuyos

- × No te servirán de nada a menos que se expresen en un lenguaje común y todo lo que se asuma sea conocimiento compartido

2. Se ignora la naturaleza volitiva de los actos comunicativos

- × En cada momento particular seleccionaré las cosas que "deseo" comunicar.

3. No tengo por qué comunicarte mis pensamientos

- × Tengo que decirte lo que quiero que tu hagas con las proposiciones que yo expreso. Si no eres capaz de diferenciar entre conjeturas, informaciones o preguntas, mis expresiones no te servirán de nada.
- × "Una comunicación efectiva necesita un correcto reconocimiento de los actos del habla"

# Teoría de los Actos del Habla (1)

- ⇒ Está desarrollada por lingüistas para ayudar en la comprensión del lenguaje humano
- ⇒ Estudia el lenguaje como acción
- ⇒ Los hablantes no sólo expresan sentencias ciertas o falsas
- ⇒ Los hablantes realizan actos del habla
  - × Informan, preguntan, sugieren, prometen...
- ⇒ Cada expresión conlleva un acto del habla
- ⇒ Identificar el acto del habla es imprescindible para una correcta comunicación

## Teoría de los Actos del Habla (2)

⇒ Es la base teórica en la comunicación entre agentes

⇒ En un acto del habla se encuentran 3 acciones:

- × **Locución.** Contexto del hecho físico del acto del habla

- × **Ilocución.** Intención con la que se realiza

- × **Perlocución.** Acto que ocurre como resultado

⇒ Ejemplo: "Cierra la ventana"

- × **Locución:** identificación del emisor, del receptor, de la ventana...

- × **Ilocución:** el emisor quiere que el receptor cierre la ventana

- × **Perlocución:** el receptor cierra (o no) la ventana

# La hipótesis F(P)

⇒ Todo acto del habla consiste en una fuerza F aplicada a una proposición P

Ilocución	Sentencia
Información	La ventana está abierta
Pregunta	¿Está abierta la ventana?
Orden	Abre la ventana
Petición	Podrías abrir la ventana
Promesa	Te prometo que abriré la ventana
Oferta	Abriré la ventana si tu quieres
...	...

# Organizaciones de Estandarización

⇒ *OMG (Object Management Group)*

× <http://www.omg.org>

⇒ *KSE (Knowledge Sharing Effort)*

× <http://www.cs.umbc.edu/kse/>

⇒ *FIPA (Foundation for Intelligent and Physical Agents)*

× <http://www.fipa.org>

# OMG

⇒ Asociación de empresas e instituciones formada a finales de los 80

⇒ Objetivos

- ✗ Reutilización, portabilidad e interoperabilidad de sistemas distribuidos de componentes software orientados a objetos

⇒ Resultados

- ✗ UML (*Unified Modeling Language*)
- ✗ CORBA (*Common Object Request Broker Architecture*)
- ✗ MASIF (*Mobile Agent System Interoperability Facility*)

# KSE (1)

## ⇒ Iniciativa

- × ARPA (*Advanced Research Projects Agency*)
- × ASOFR (*Air Force Office of Scientific Research*)
- × NRI (*Corporation for National Research Initiative*)
- × NSF (*National Science Foundation*)

## ⇒ Objetivo

- × Facilitar la compartición y reutilización de bases y sistemas basados en conocimiento

## ⇒ Filosofía

- × La interacción eficaz de agentes software tres componentes fundamentales
  1. Un lenguaje común
  2. Una comprensión común del conocimiento intercambiado
  3. Una habilidad para intercambiar todo lo relativo a (1) y a (2)

# KSE (2)

⇒ El KSE está organizado entorno a tres grupos de trabajo

× *Interlingua Group*

- Encargado de desarrollar un lenguaje común para poder representar los contenidos de las bases de conocimiento

× *SRKB Group (Shared Reusable Knowledge Base)*

- Encargado de definir los conocimientos reutilizables para distintos dominios de aplicación

× *External Interfaces Group*

- Encargado de la interacción en tiempo de ejecución) entre sistemas basados en conocimiento

⇒ Resultados

× Sintaxis: KIF (*Knowledge Interchange Format*)

- Lenguaje para el intercambio de conocimiento <http://www.cs.umbc.edu/kse/kif/>

× Semántica: Ontolingua

- Lenguaje para la definición de Ontologías <http://www.cs.umbc.edu/kse/ontology/>

× Pragmática: KQML (*Knowledge Query and Manipulation Language*)

- Lenguaje para la comunicación entre agentes <http://www.cs.umbc.edu/kqml>
- Interoperabilidad entre agentes en un entorno distribuido

# KIF

- ⇒ Permite la representación de conocimiento
- ⇒ Basado en el cálculo de predicados de primer orden
- ⇒ Sintaxis similar a la empleada en LISP
- ⇒ Ejemplos de expresiones
  - × Datos
    - (salario 27843834k analista 30.000)
    - (salario 122333212s operador 15.000)
  - × Conocimiento
    - (> (salario empleado1) (salario empleado2))
    - (=> (and (num-real ?x) (num-par ?n)) (>(expt ?x ?n) 0))
    - (defrelation soltero(?x) := (and(hombre ?x) (not(casado ?x))))
  - × Inferencia
    - (interested Agente1 ` (salario ,?x ,?y ,?z))

# Ontolingua

- ⇒ Lenguaje para construir, publicar y compartir ontologías
- ⇒ Las ontologías pueden traducirse automáticamente a otros lenguajes de contenido (KIF, Prolog, CLIPS...)
- ⇒ Incluye primitivas para combinar ontologías
- ⇒ Este grupo ha definido un número considerable de ontologías sobre distintos dominios que pueden ser de interés para distintas aplicaciones

# KQML

⇒ En la comunicación intervienen los siguientes elementos

- × El protocolo de interacción
  - Estrategia de alto nivel seguida por el agente software para controlar la interacción con otros agentes
  - Desde esquemas de negociación hasta esquemas tan simples como: "cada vez que desconozcas algo de tu interés busca a algún agente que lo sepa y pregúntale"
- × El lenguaje de comunicación
  - Es el medio a través del que se intercambian los actos de la comunicación
  - Indica si el contenido de la comunicación es una información, una respuesta o algún tipo de consulta
- × El protocolo de transporte
  - Mecanismo de transporte utilizado en la comunicación
  - TCP, SMTP, HTTP...

# KQML (capas)

⇒ El lenguaje KQML está dividido en 3 capas

× La capa de contenido

- Relacionada con el contenido del mensaje
- Puede expresarse en cualquier lenguaje
- KQML sólo tiene interés en identificar el inicio y el final del contenido

× La capa de mensaje

- Es el núcleo del lenguaje KQML
- Determina los tipos de interacción que puede realizar un agente
- Identifica el protocolo, la ontología y el acto del habla (*performative*)

× La capa de comunicación

- Identifica las características del mensaje que describen los parámetros de bajo nivel de la comunicación (emisor, receptor e identificador de mensaje)

# KQML (mensajes)

- ⇒ Un mensaje en KQML representa la intención de realizar alguna acción (*performative* o acto del habla)
- ⇒ Un mensaje KQML es una lista tipo LISP en la que
  - ✗ El primer elemento de la lista es el identificador del acto del habla
  - ✗ El resto de la lista son pares atributo-valor sin un orden predefinido

```
(ask-one
  :sender      agente1
  :receiver    servidor-telefonica
  :reply-with  accion-telefonica
  :content     (Precio Telefonica ?p)
  :language    prolog
  :ontology    IBEX-35
)
```

## KQML (atributos reservados)

Atributo	Significado
:content	La información
:sender	Emisor del mensaje
:receiver	Receptor del mensaje
:language	El nombre del lenguaje de representación empleado en el atributo :content
:ontology	El nombre de la ontología utilizada en el atributo :content
:reply-with	Etiqueta para la respuesta (si es que el emisor la espera)
:in-reply-to	La etiqueta esperada en la respuesta

## KQML (*performatives*)

Categoría	Nombre
Información	tell, untell
Consulta	evaluate, reply, ask-if, ask-about, ask-one, ask-all, sorry
Consulta con respuesta múltiple	stream-about, stream-all
Orden	achieve, unachieve
Control de respuestas	standby, ready, next, rest, discard
Capacidad	advertise
Notificación	subscribe, monitor
Gestión de red	register, unregister, forward, broadcast, pipe, break
Facilitación	broker-one, broker-all, recommend-one, recommend-all, recruit-one, recruit-all

# KQML (*performatives*)

Nombre	Significado
tell	E comunica una información que se encuentra en su BC
evaluate	E quiere que R evalúe la expresión
reply	E comunica a R una respuesta esperada
ask-if	E quiere saber si la expresión se encuentra en la BC de R
ask-about	E quiere conocer todo el conocimiento de R relacionado con la expresión
ask-one	E quiere que R conteste a una pregunta
stream-about	Versión en respuesta múltiple de ask-about
achieve	S quiere que R convierta en verdadera la expresión en su BC
standby	S quiere que R esté preparado para responder a una <i>performative</i>
ready	S está preparado para responder a R
next	S quiere que R le devuelva la siguiente respuesta (de una consulta múltiple)
advertise	S está preparado para responder a una determinada <i>performative</i>
register	S puede responder a <i>performatives</i> de un determinado agente
broadcast	S quiere que R envíe la <i>performative</i> a todas los agentes conectados
recommend-one	S quiere el nombre de un agente que pueda contestar a una <i>performative</i>
recruit-one	S quiere que R le proporcione el nombre de otro agente que conteste a la <i>performative</i>

# KQML (ejemplo 1)

A envía a B:

(ask-one :content (interested '(sobre, bloqueA, ?x))  
:language KIF :ontology bloques :reply-with q1)

B contesta a A:

(reply :content (sobre bloqueA bloqueB) :language KIF  
:ontology bloques :in-reply-to q1)

## KQML (ejemplo 2)

A envía a B:

(stream-about :content bloqueA :language KIF  
:ontology bloques :reply-with q1)

B contesta a A con la siguiente serie de performatives:

(tell :content (sobre bloqueA bloqueB) :language KIF  
:ontology bloques :in-reply-to q1)

(tell :content (sobre mesa bloqueA) :language KIF  
:ontology bloques :in-reply-to q1)

(sorry :in-reply-to q1)

## KQML (ejemplo 3)

A envía a B:

(achieve :content (sobre mesa bloqueB) :language KIF  
:ontology bloques :reply-with q1)

B después de llevar a cabo la tarea contesta a A:

(tell :language KIF :ontology bloques :in-reply-to q1  
:content (sobre mesa bloqueB))

## KQML (ejemplo 4)

A envía a B:

(standby :language KQML :ontology K10 :reply-with q1  
:content (stream-about :language KIF :ontology bloques  
:reply-with q3 :content bloqueA))

B responde con:

(ready :reply-with XA77 :in-reply-to q1)

A continúa con:

(next :in-reply-to XA77)

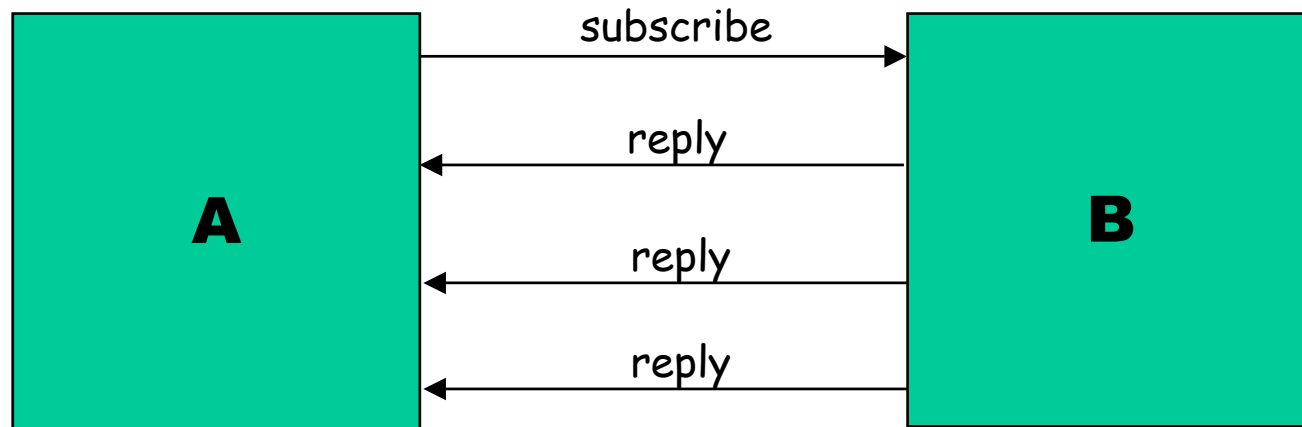
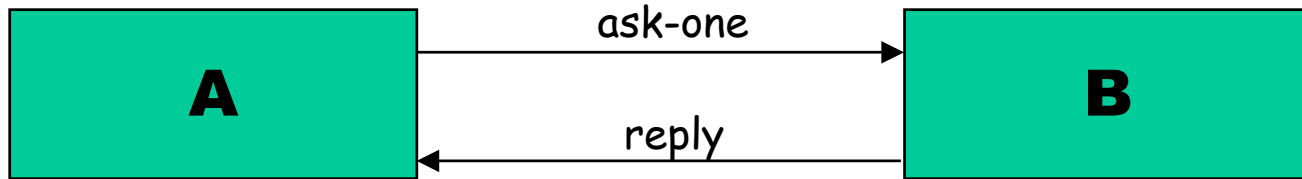
B responde con:

(tell :language KIF :ontology bloques :in-reply-to q3  
:content (sobre bloqueA bloqueB))

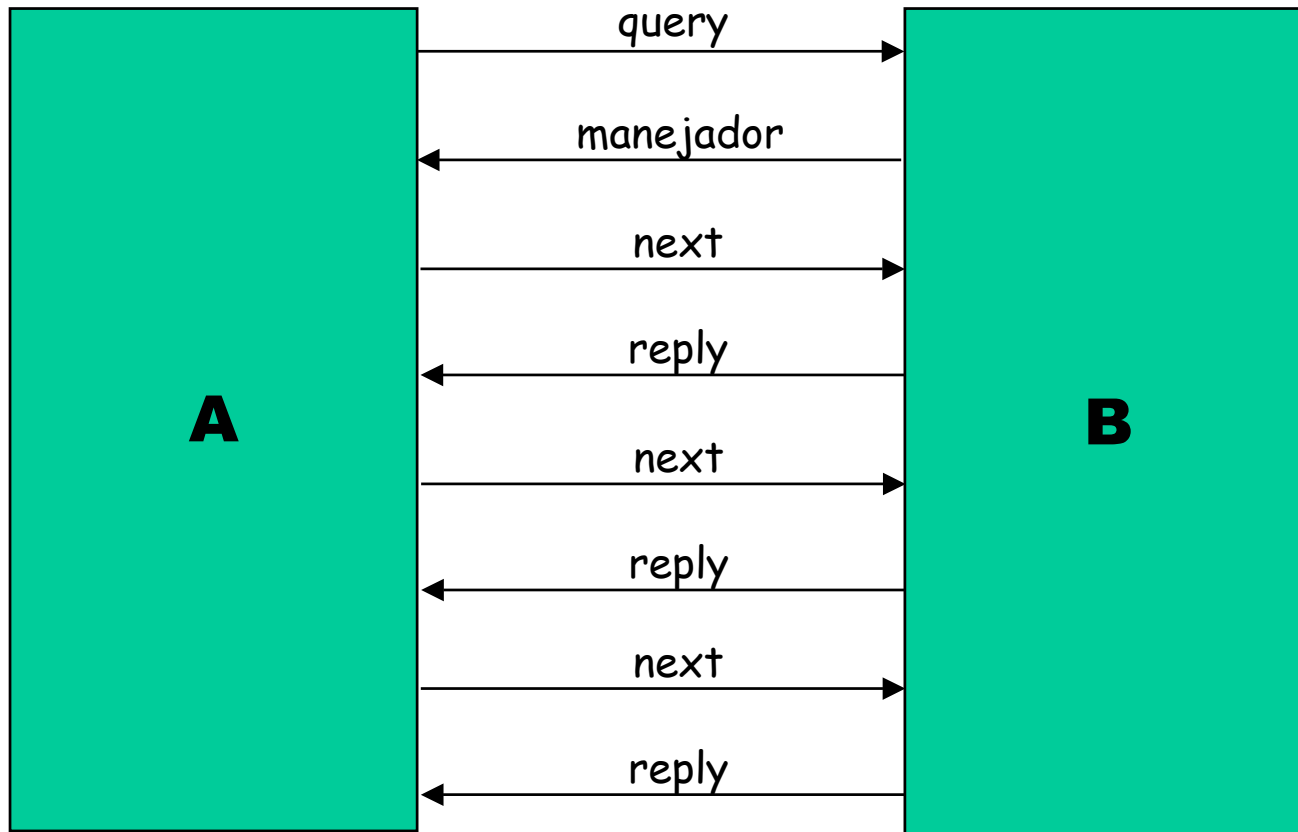
Así hasta que A envía:

(discard :in-reply-to XA77)

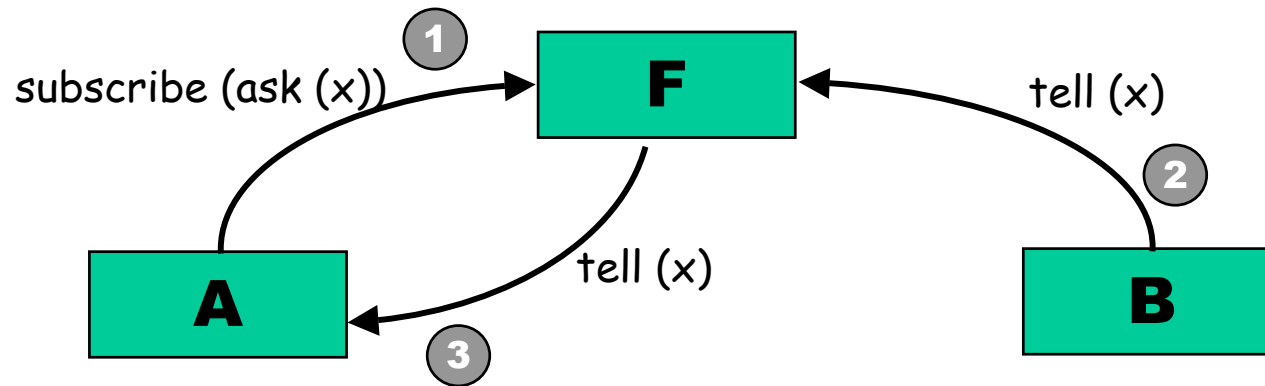
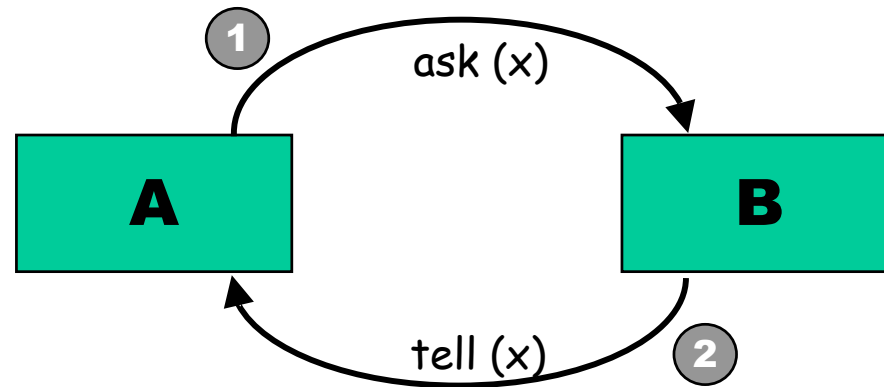
# KQML (protocolos básicos)



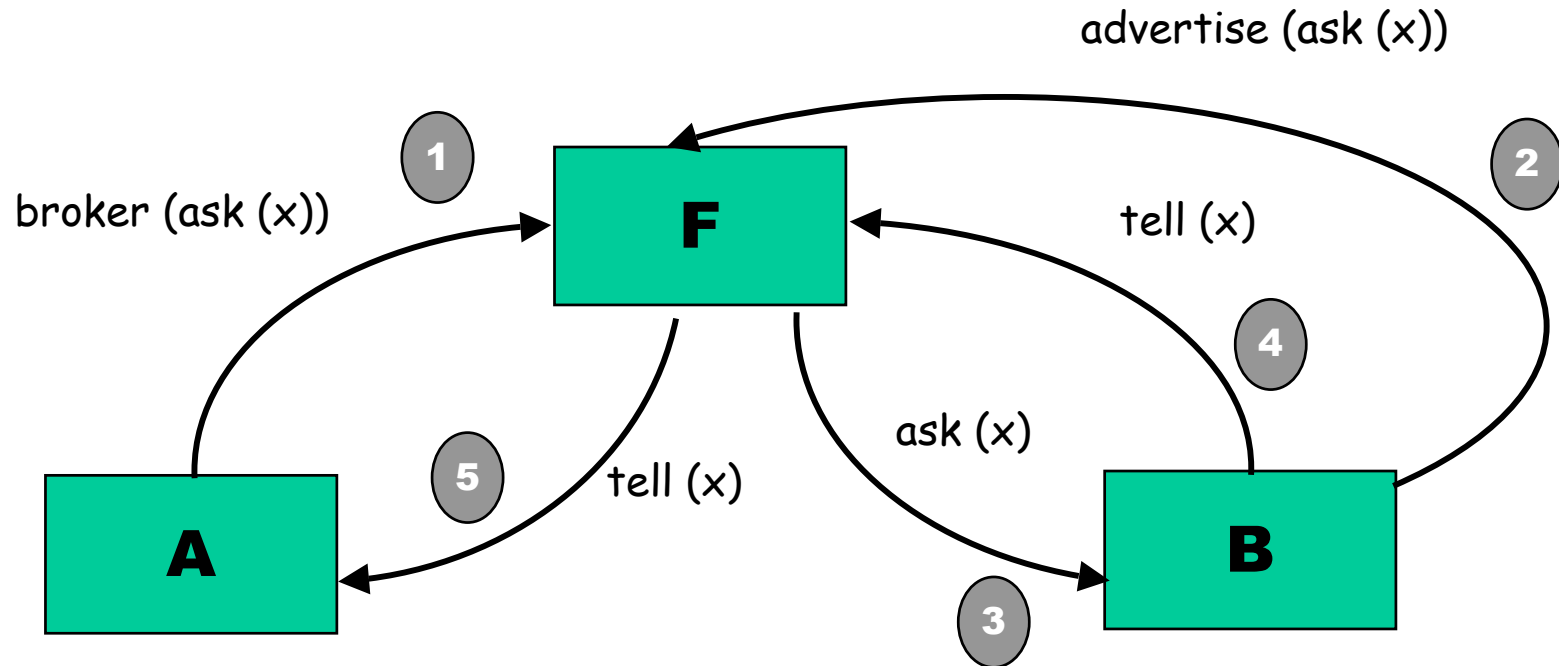
# KQML (protocolos básicos)



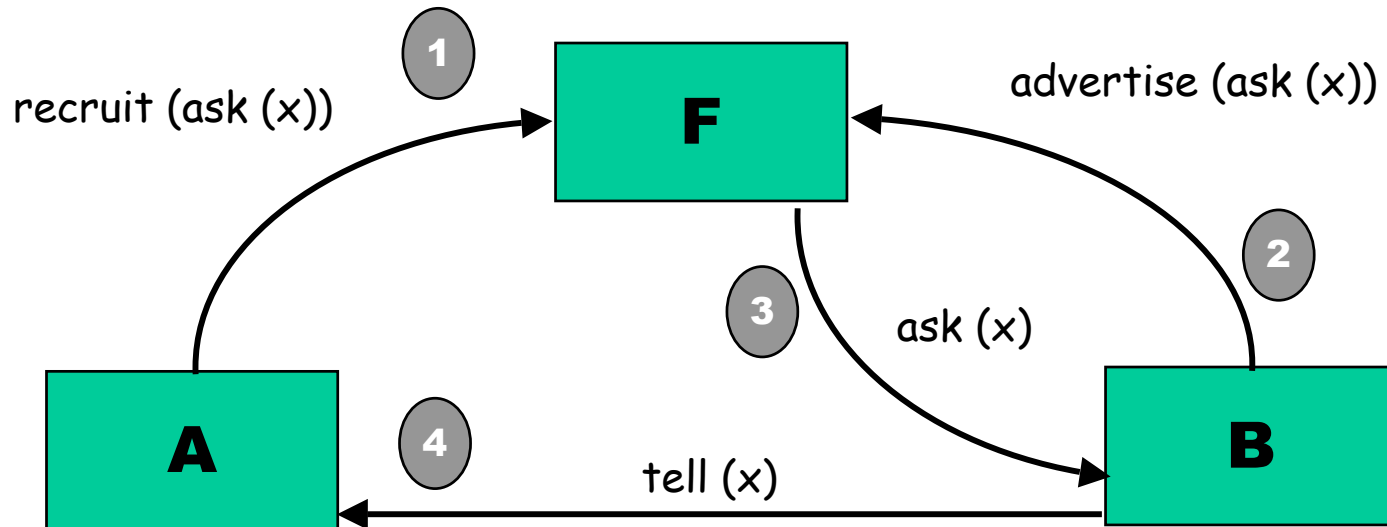
# KQML (agentes facilitadores)



# KQML (agentes facilitadores)



# KQML (agentes facilitadores)



# KQML (agentes facilitadores)

