El estándar IEEE-FIPA
Foundation for Intelligent Physical Agents
Sistemas Multi-Agente y Sistemas Autónomos

Juan A. Botia

October 3, 2007

FIPA, primeras ideas

- Organización internacional sin ánimo de lucro
- Dedicada a establecer un marco común y genérico tanto para agentes y sistemas multi-agente
- Su objetivo es "*el permitir la construcción de sistemas que se integren con su entorno de computación particular, mientras que interoperan con sistemas de agentes que residen en entornos heterogeneos, todo con el mínimo esfuerzo*"

FIPA - Organización

Objetivo de las especificaciones

- facilitar la interacción entre agentes y sistemas de agentes a través de diferentes plataformas de agentes de diferentes fabricantes.

FIPA hace énfasis en el uso práctico de índole comercial e industrial de los sistemas de agentes sin embargo

- también se ocupan de agentes inteligentes (o cognitivos)

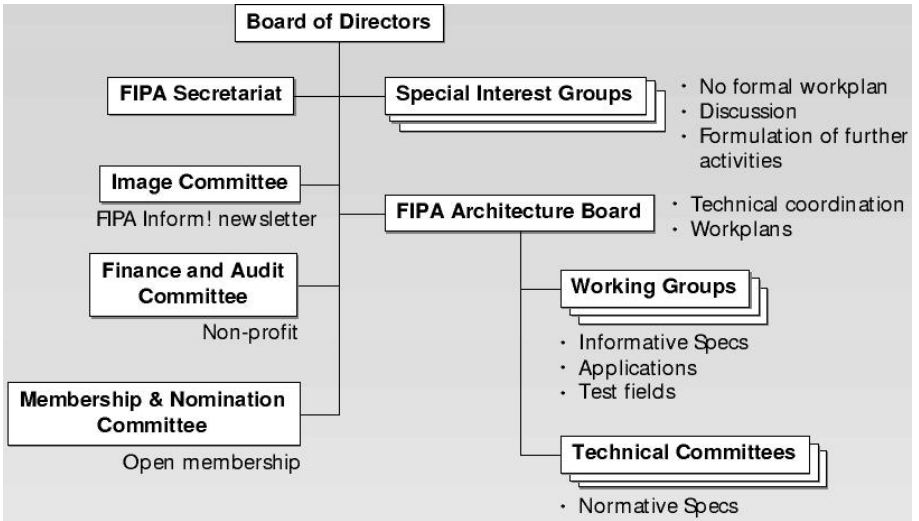
Planteamiento tecnológico básico

- a través de actos del habla, lógica de predicados y ontologías públicas, se puede conseguir interpretar la comunicación entre agentes de forma que se mantenga la intención y significado de la comunicación.

FIPA - Organización

Dos grandes grupos organizativos

1. Aquellos que se encargan de la especificación y mantenimiento de estándares para el desarrollo de agentes y SMAs
2. aquellos que se encargan del mantenimiento de la organización



FIPA - Estructura

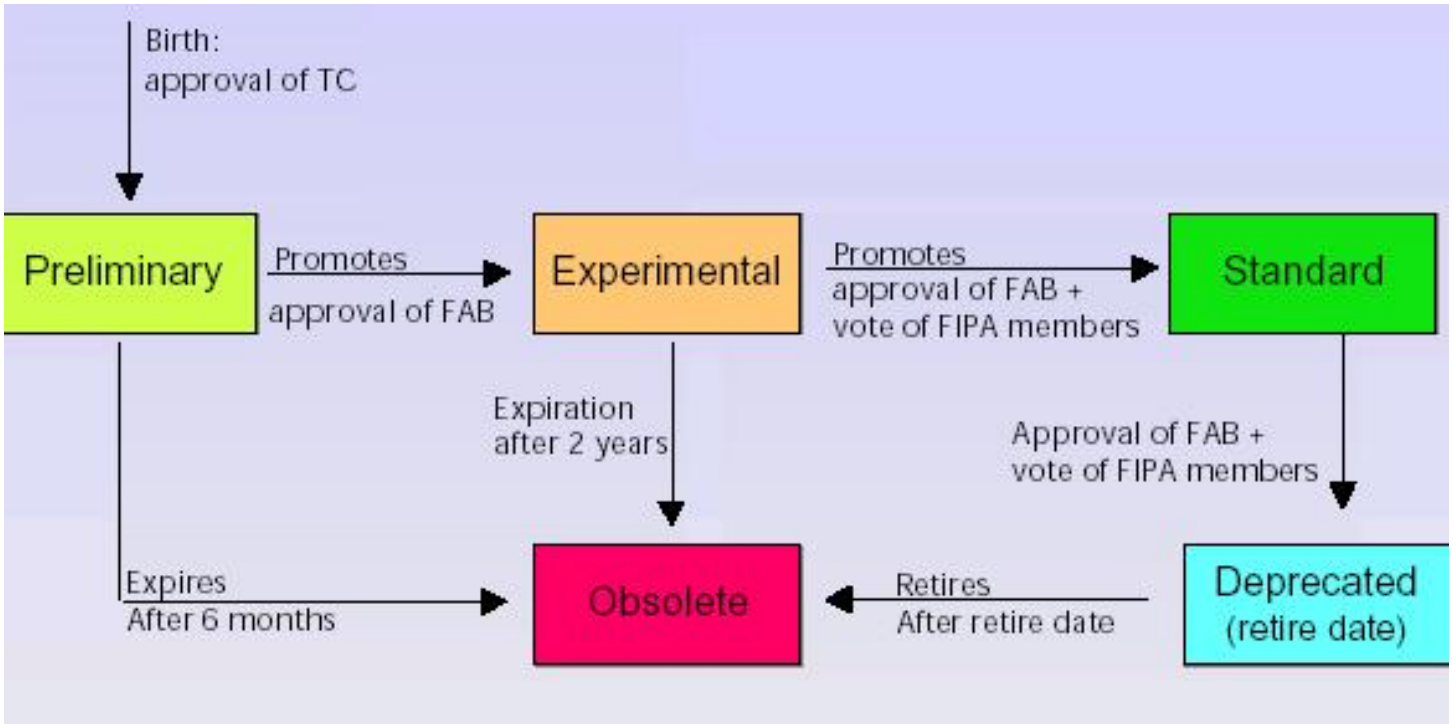
- SIGs: se dedican a tares muy concretas y de interés para la comunidad FIPA
  - SIG *Fipa for Business Applications* encargado de encontrar y definir nuevos segmentos de mercado en los que este tipo de aplicaciones software puedan resultar de interés
- TCs: desempeñan labores técnicas, generando especificaciones o modificando especificaciones ya producidas
  - comités para redes ad-hoc, metodologías, modelado, seguridad, semántica y servicios

  lo crea el *FIPA Architecture Board* en conjunción con el *Board of Directors*
- WGs: tareas que no necesariamente están relacionadas con la tecnología
- e.g. interacción directa con otras organizaciones productoras de estándares como la OMG, o el ITU

Ciclo de vida de las especificaciones

- Una especificación comienza a existir en forma de “Especificación Preliminar” generada por un comité técnico que lo ha generado a partir de un borrador
- Si el TC considera madura la especificación pasa al estatus de “Especificación Experimental”.
- Así se mantiene por un periodo de dos años en total
- Si se producen implementaciones y pruebas de campo con éxito se pasa a estándar
- Pasa a Deprecated cuando deja de ser útil, o es superada por otra especificación y después a Obsolete

Ciclo de vida de las especificaciones



Antecedentes a la organización

Idea principal: el poder de los agentes software tiene que venir de la interoperabilidad (diferentes SMAs comunicándose entre sí)

- KQML
  - A favor:como primera iniciativa, se ha convertido en uno de los lenguajes de comunicación más extendidos
  - En contra: no existe un consenso suficientemente grande para que sea un estándar de facto (múltiples dialectos)
- MASIF (*Mobile Agent System Interoperability Facility*)
  - A favor: generada por la OMG un año antes que FIPA, en 1995, se ocupa de estandarizar los aspectos de movilidad
  - En contra: MASIF no soporta o estandariza enteroperabilidad entre agentes que no sean móviles en plataformas de agentes diferentes (solo CORBA-based)

Otras razones

- Necesidad de buscar mercado y masa crítica de clientes, aplicaciones y productos a medio y largo plazo.
- En 1995 la investigación en sistemas multi-agente estaba ya madura, lista para estandarizar
- Los estándares estimulan la industria
- Ya existía la infraestructura necesaria para crear un estándar de agentes (CORBA e IIOP proporcionaban una infraestructura para el transporte de mensajes)

Un recorrido rápido por las especificaciones

Divididas en 5 grandes grupos

1. Aplicaciones
2. Arquitectura Abstracta
3. Lenguajes de comunicación
4. Gestión de agentes
5. Transporte de mensajes

Un recorrido rápido por las especificaciones

Aplicaciones

- FIPA Nomadic Application Support Specificacion
- FIPA Quality of Service Specificacion
- FIPA Personal Travel Assistance Specification
- FIPA Audio-Visual Entertainment and Broadcasting Specification
- FIPA Network Management and provisioning Specification
- FIPA Personal Assistant Specification
- FIPA Message Buffering Service Specification

se los cuales solamente las das primeras están consolidadas como estándares, siendo el resto experimentales.

Un recorrido rápido por las especificaciones

Arquitectura abstracta

- la especificación *FIPA Abstract Architecture Specification*, que tiene el estatus de estándar y
- *FIPA Domains and Policies Specification*, que ha caido en desuso aunque su estatus es aun Preliminar

Un recorrido rápido por las especificaciones

Lenguajes de comunicación

- Protocolos de interacción
- Actos comunicativos:
- Lenguajes de contenido:

Un recorrido rápido por las especificaciones

Lenguajes de comunicación

- Protocolos de interacción Los siguientes son estándares:
  - FIPA Request Interaction Protocol Specification
  - FIPA Query Interaction Protocol Specification
  - FIPA Request When Interaction Protocol Specification
  - FIPA Contract Net Interaction Protocol Specification
  - FIPA Iterated Contract Net Interaction Protocol Specification
  - FIPA Brokering Interaction Protocol Specification
  - FIPA Recruiting Interaction Protocol Specification
  - FIPA Subscribe Interaction Protocol Specification
  - FIPA Propose Interaction Protocol Specification

  y las subastas alemana e inglesa mantienen el estatus de experimental
  - FIPA English Auction Interaction Protocol Specification
  - FIPA Dutch Auction Interaction Protocol Specification
- Actos comunicativos:
- Lenguajes de contenido:

Un recorrido rápido por las especificaciones

Lenguajes de comunicación

- Protocolos de interacción
- Actos comunicativos: dentro de este grupo solamente existe una recomendación, la *FIPA Communicative Act Library Specification* que tiene el grado de estándar y que veremos a fondo.
- Lenguajes de contenido:

Un recorrido rápido por las especificaciones

Lenguajes de comunicación

- Protocolos de interacción
- Actos comunicativos:
- Lenguajes de contenido:
  - FIPA SL Content Language Specification
  - FIPA CCL Content Language Specification
  - FIPA KIF Content Language Specification
  - FIPA RDF Content Language Specification

  de las cuales, solamente la correspondiente al lenguaje SL tiene consideración de estándar

Un recorrido rápido por las especificaciones

Gestión de agentes

- FIPA Agent Management Specification
- FIPA Agent Discovery Service Specification
- FIPA JXTA Discovery Middleware Specification

de las cuales únicamente la primera es estándar (será tratada a fondo aquí) y el resto son preliminares aun.

Un recorrido rápido por las especificaciones

Transporte de mensajes

- Representaciones de ACL
- Representaciones de envoltorios
- Protocolos de transporte

Un recorrido rápido por las especificaciones

Transporte de mensajes

- Representaciones de ACL
  - FIPA ACL Message Representation in Bit-Efficient Specification
  - FIPA ACL Message Representation in String Specification
  - FIPA ACL Message Representation in XML Specification
- Representaciones de envoltorios
- Protocolos de transporte

Un recorrido rápido por las especificaciones

Transporte de mensajes

- Representaciones de ACL
- Representaciones de envoltorios
  - FIPA Agent Message Transport Envelope Representation in XML Specification
  - FIPA Agent Message Transport Envelope Representation in Bit Efficient Specification

  las cuales son estándares ambas.
- Protocolos de transporte

Un recorrido rápido por las especificaciones

Transporte de mensajes

- Representaciones de ACL
- Representaciones de envoltorios
- Protocolos de transporte
  - FIPA Agent Message Transport Protocol for IIOP Specification
  - FIPA Agent Message Transport Protocol for HTTP Specification
  - FIPA Agent Message Transport Protocol for WAP Specification

  de las que son estándares las que tratan con IIOP y HTTP, siendo la de WAP experimental.

La arquitectura de referencia FIPA

La arquitectura de referencia FIPA se denomina “Arquitectura Abstracta” ¿por qué?

- definir los elementos comunes a diferentes sistemas subyacentes
- reunirlos en una especificación común
- instanciar esa arquitectura en forma de implementaciones diferentes pero interoperables !!

La arquitectura de referencia FIPA

Tópicos cubiertos

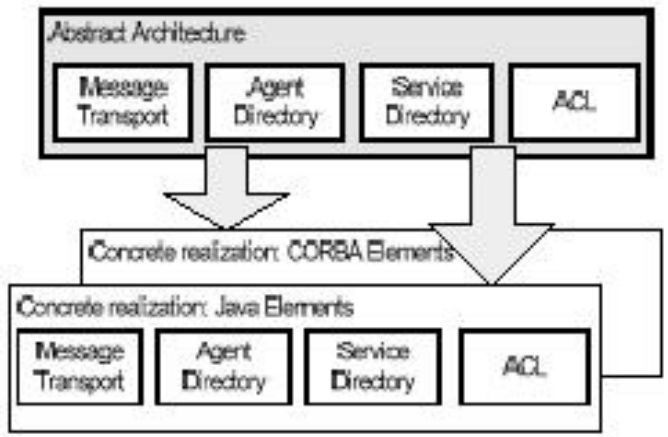
- Definición de un modelo abstracto para el descubrimiento de servicios disponibles para otros agentes y servicios.
- Interoperabilidad entre diferentes medios de transporte de mensajes.
- Soporte para el manejo de diferentes formas de representación de mensajes ACL
- Soporte para el manejo de diferentes lenguajes de contenido
- Soporte para el manejo de diferentes representaciones de servicios de directorio

Tópicos no cubiertos: por ejemplo, la gestión del ciclo de vida de un agente

Realización-materialización de la arquitectura abstracta

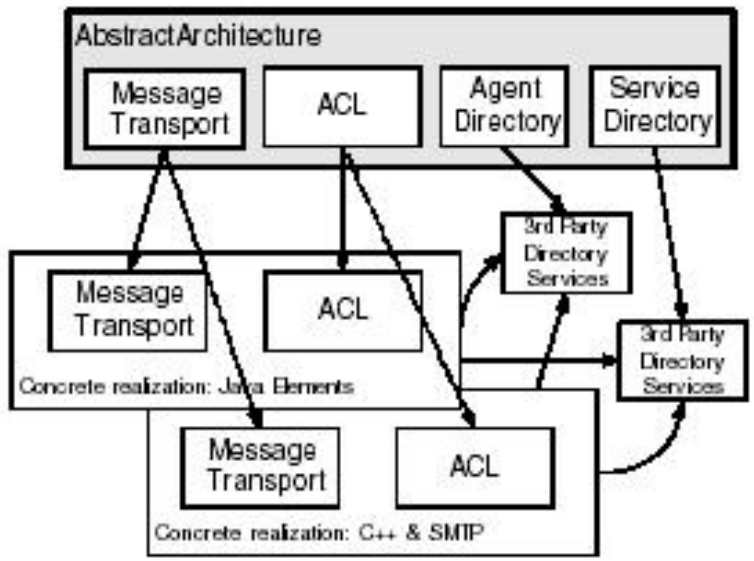
Una arquitectura abstracta no es directamente implementable

- Base para otras especificaciones más concretas (elementos software, lenguajes de programación, protocolos de red, etc)
- El desarrollador tiene libertad para materializar transporte de mensajes, directorio de agentes, directorio de servicios y ACL



Realización-materialización de la arquitectura abstracta

- Una arquitectura abstracta puede incluir elementos adicionales
- una realización puede ser, bien de la arquitectura entera pero también de un único elemento (e.g. directorio de agentes)

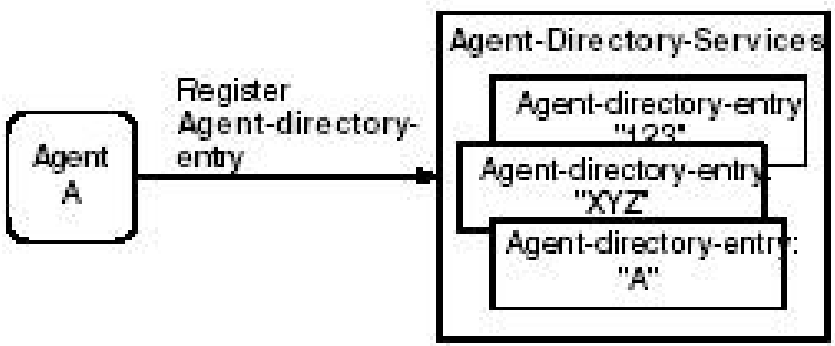


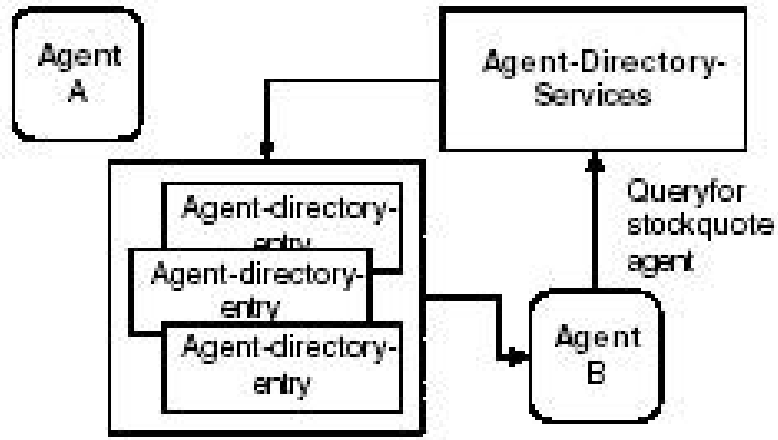
Localizando agentes y servicios en FIPA

- Proporcinar un lugar en donde los agentes registren sus descripciones/servicios de tal forma que otros agentes puedan utilizar ese medio para localizar agentes/servicios con los que deseen interactuar/invocar
- Datos mínimos de un agente
  - AID: nombre único globalmente
  - Localizador: una o más descripciones del transporte que, a su vez, contiene el tipo de transporte (i.e. el protocolo), la dirección de transporte para ese protocolo y cero o más propiedades adicionales
- Cómo anunciarse
  - Primero necesita estar accesible, y por lo tanto se engancha (i.e. hace un *bind*) a una o más direcciones de transporte
  - Luego se da a conocer
    1. crea una entrada de directorio con sus datos de agente
    2. registra la entrada con el servicio de directorio de agentes

Localizando agentes y servicios en FIPA II

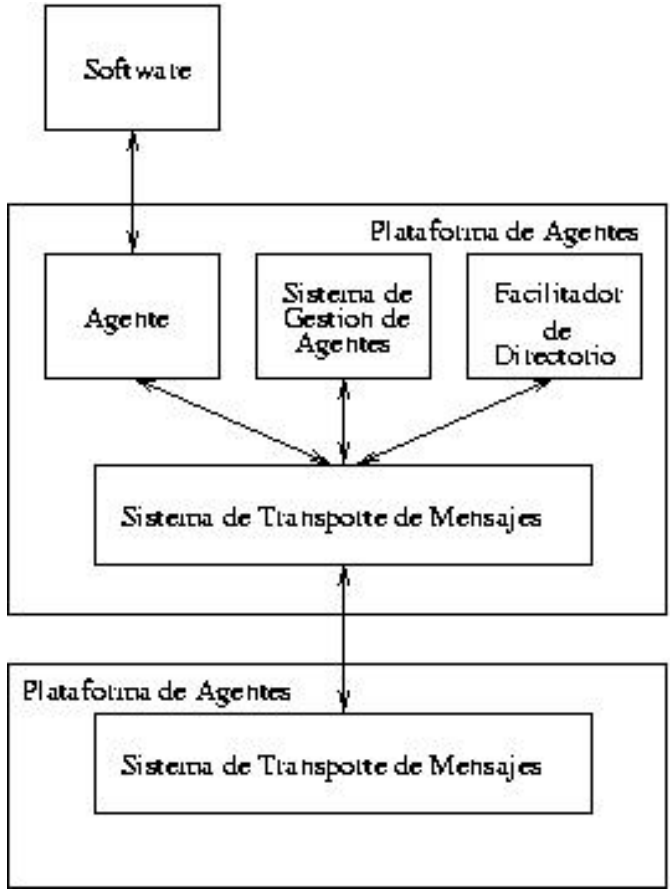
La búsqueda de los agentes es por matching simple de cadenas y pares atributo-valor





El registro y localización de servicios es análogo al de agentes

Panorama global



Elementos del entorno de ejecución

El AMS: habilita el acceso a la plataforma

- Un AMS para cada plataforma de agentes (AP)
- Gestiona la operación la plataforma con respecto a los agentes que viven en ella (ciclo de vida, registro de nombres)
- Operaciones en el directorio: `register`, `deregister`, `modify`, `search` y `get-description`
- Operaciones contra agentes: suspender, terminar, crear, finalizar ejecución, invocar y ejectuar agente

El agente como proceso software en la plataforma

Agente en la plataforma

proceso computacional que implementa la funcionalidad autónoma de comunicación para una aplicación concreta

Identificado mediante el AID (*Agent Identifier*)

- `name`
- `addresses`
- `resolvers`

El agente como proceso software en la plataforma

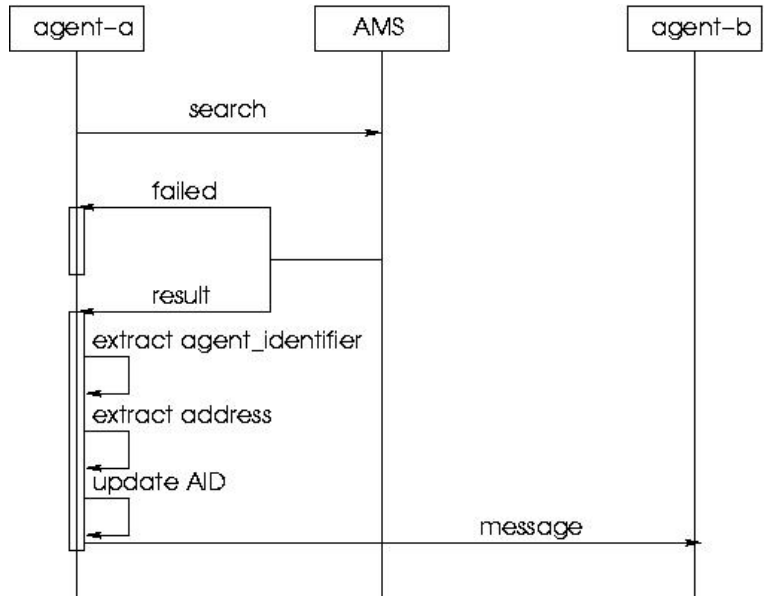
Ejemplo: supongamos que `agente-a` quiere mandar un mensaje a `agente-b` el cual tiene como AID el siguiente

```
(agent-identifier
  :name agent-b@bar.com
  :resolvers (sequence
    (agent-identifier
      :name ams@foo.com
      :addresses (sequence iiop://foo.com/acc))))
```

y `agente-a` necesita saber sus direcciones de transporte

Ejemplo, cont

La secuencia de comunicación será



Nótese que el AMS tiene un nombre reservado, como este

```
(agent-identifier
  :name ams@nombre_plataforma
  :addresses (sequence direcciones_transporte_plataforma))
```

Facilitador de directorio en la plataforma

Componente opcional

- implementa la funcionalidad de un directorio de servicios (i.e. páginas amarillas) de la arquitectura abstracta
- Si existe, tiene reservado el AID

```
(agent-identifier
   :name df@nombre_plataforma
   :addresses (sequence direcciones_transporte_plataforma))
```

- Posibilita subscripción para escuchar eventos relativos a operaciones (registro, desregistro y modificación) en el directorio de servicios mediante `fipa-subscribe`

Ciclo de vida del agente FIPA

Un agente físico (en términos de software) está sujeto a un ciclo de vida que define los estados en los que se puede encontrar y cómo se realizan los cambios de un estado a otro.
Características más importantes

- AP Bounded (conectado a la plataforma): un agente es gestionado físicamente dentro de una AP y el ciclo de vida de un agente estático está, por tanto, asociado a una determinada AP.
- Independiente de la aplicación
- Orientado a instancia: el agente que cumple el ciclo de vida es una instancia de una clase de agente, con un identificador único (identidad)
- Único: cada agente solamente tiene un ciclo de vida en todo momento y dentro de una única AP.

Envío de mensajes y ciclo de vida

El MTS es responsable de envío de mensajes.
Dependiendo del estado del agente receptor

- Activo: el mensaje se entrega al agente directamente.
- Iniciado/esperando/suspendido: o bien el MTS almacena el mensaje en el buffer correspondiente hasta que el agente está activo o bien entrega el mensaje en otra localización si antes se ha indicado un forward para ese agente.
- Tránsito: o bien el MTS almacena el mensaje en el buffer correspondiente hasta que el agente está activo o bien entrega el mensaje en otra localización si antes se ha indicado un forward para ese agente como en el caso anterior. Solamente los agentes móviles pueden entrar en este estado.
- Unknown: discreccional

El registro de un agente en una AP

Al comenzar la ejecución, se ha de registrar en la plataforma para que el resto de agentes puedan localizarlo y enviarle mensajes de manera apropiada
¿Cómo?

1. al crearse dentro de la plataforma
2. al llegar a la AP desde otra AP (agente móvil y APs que lo soportan)
3. de manera explícita

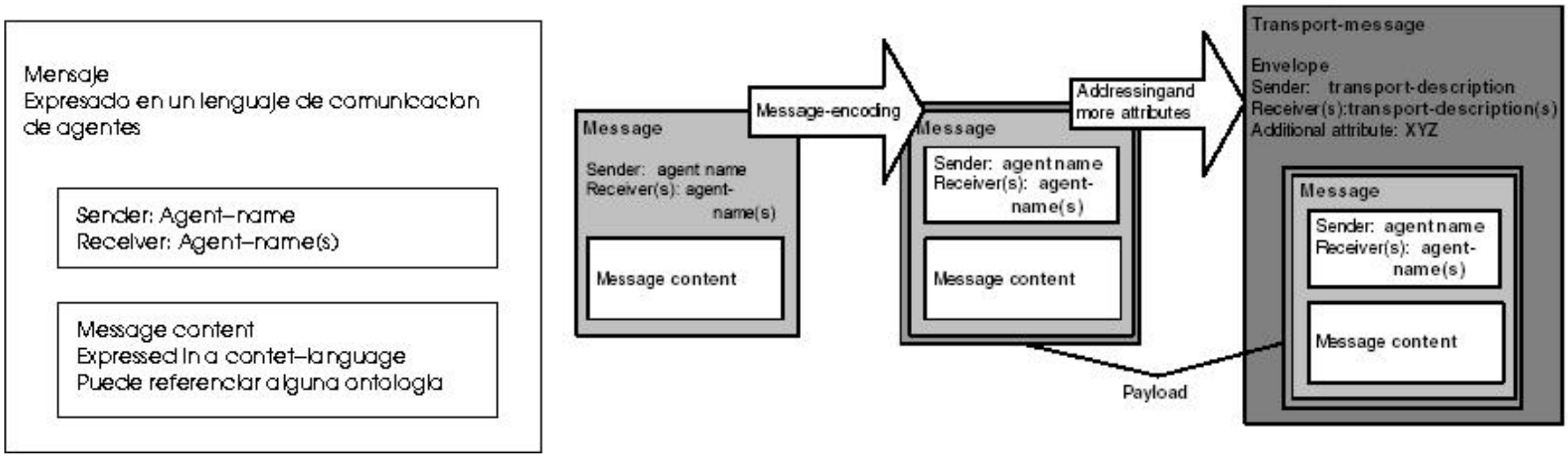
Ejemplo

Un agente localizado en AP `bar.com` se quiere registrar en la plataforma `foo.com`. Para ello enviará un mensaje `request` como el siguiente al correspondiente AMS:

```
(request
  :sender (agent-identifier
            :name discovery-agent@bar.com
            :addresses (sequence iiop://bar.com/acc))
  :receiver (set (agent-identifier
              :name ams@foo.com
              :addresses (sequence iiop://foo.com/acc)))
  :ontology fipa-agent-management :language fipa-sl0 :protocol fipa-request
  :content "((action
              (agent-identifier
                :name ams@foo.com :addresses (sequence iiop://foo.com/acc))
              (register
                (:ams-description
        :         name
                    (agent-identifier
                      :name discovery-agent@bar.com
                      :addresses (sequence iiop://bar.com/acc)) ...)))")...)
```

Elementos de los mensajes FIPA

Estructura, representación y transporte



Elementos de los mensajes FIPA

- La parte externa del mensaje corresponde al lenguaje de comunicación de agentes
- Si nos referimos a la estandarización ACL de FIPA, este va a consistir nuevamente en una lista de pares atributo-valor

Parameter	Category of Parameters
performative	Type of communicative acts
sender	Participant in communication
receiver	Participant in communication
reply-to	Participant in communication
content	Content of message
language	Description of Content
encoding	Description of Content
ontology	Description of Content
protocol	Control of conversation
conversation-id	Control of conversation
reply-with	Control of conversation
in-reply-to	Control of conversation
reply-by	Control of conversation

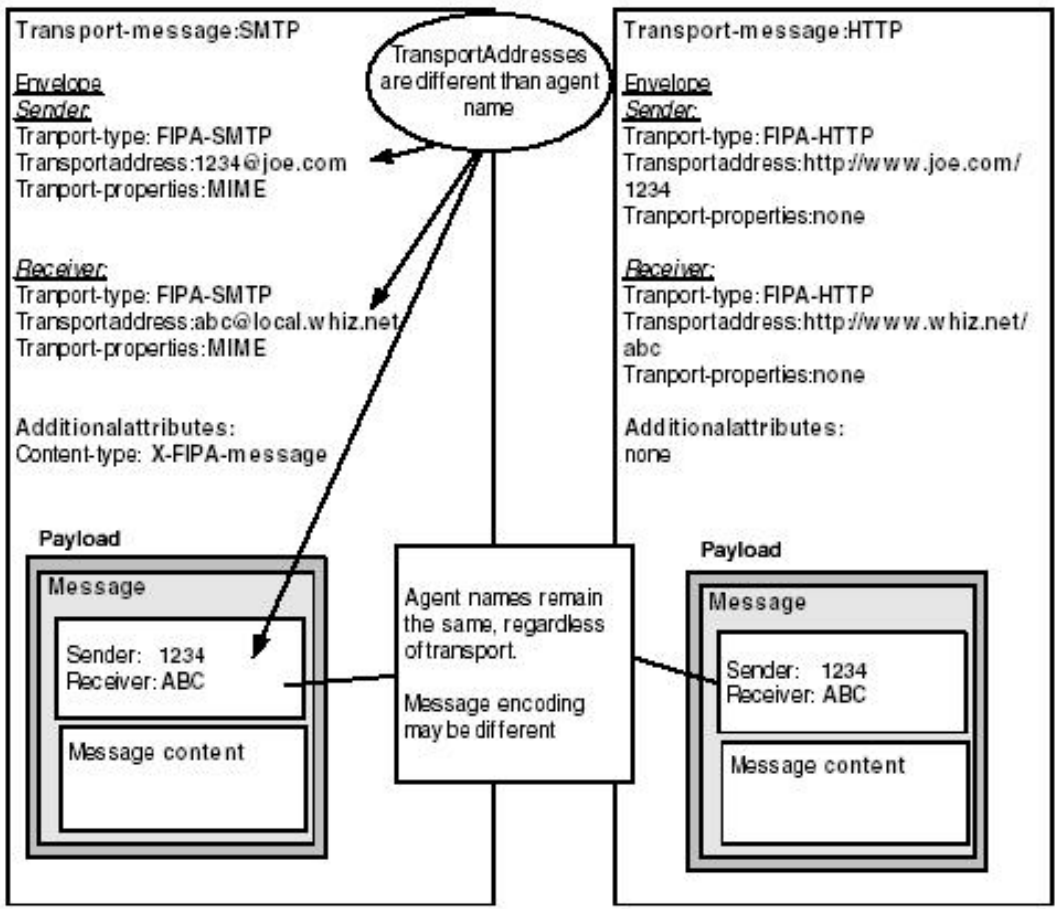
Más sobre mensajes

Ahora supongamos que en el AMS tenemos la entrada, en el directorio de páginas blancas correspondiente, siguiente

```
Agent-name:ABC
Agent-Locator:
  Transport-type    Transport-specific-address   Transport-specific-property
  HTTP              http://www.whiz.net/abc      <none>
  SMTP              abc@lowcal.whiz.net          <none>
Agent-attributes:
                    Attrib-1:yes
                    Attrib-2:yellow
                    Language: French, German, English
                    Preferred-negotiation: contract-net
```

Más sobre mensajes II

Cualquier otro agente podría contactar con ABC, a través del AMS y mandar mensajes de dos maneras distintas



Seguridad en los mensajes

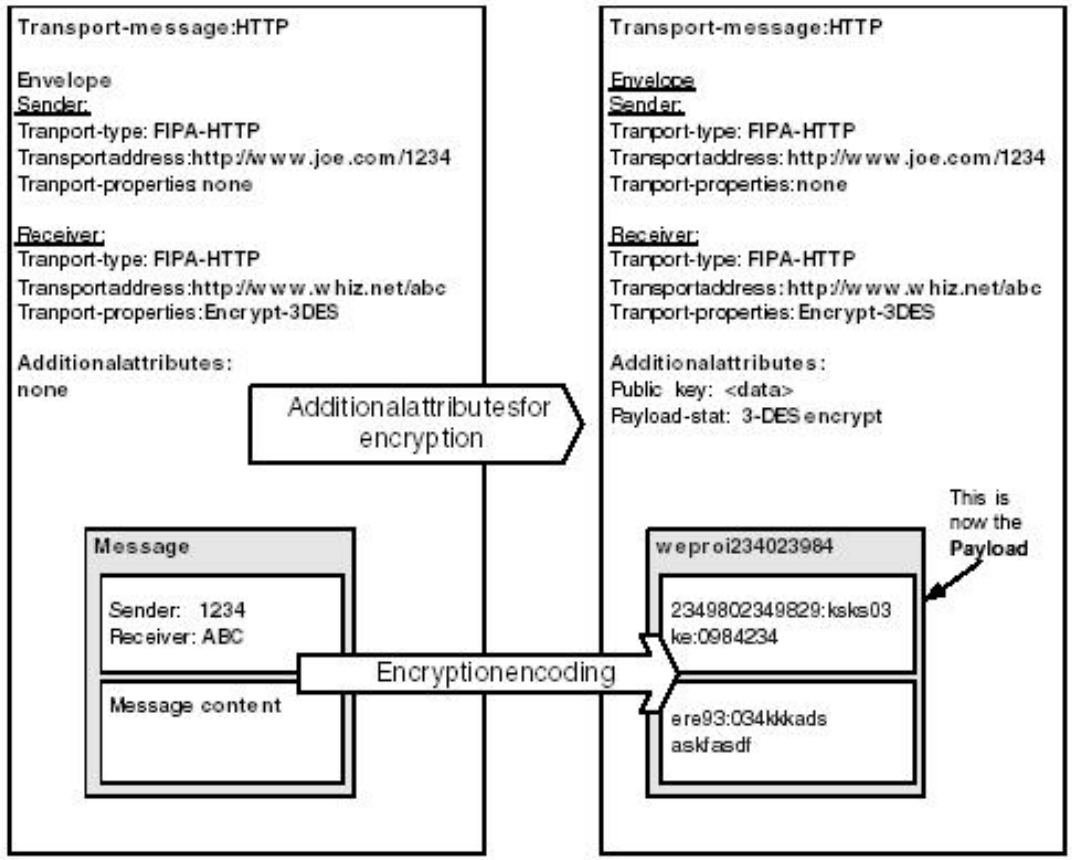
Se deben considerar

- Validez del mensaje
  - Toda modificación que se realice sobre el mismo sea detectable
- Encriptación
  - Poder enviar mensajes sin que terceros no autorizados sean capaces de leer su contenido

  La arquitectura abstracta propone soportar estas dos vertientes con
  - Atributos adicionales (envoltura)
  - Tipos de representación en la codificación

NO ESTANDARIZA!!!

Seguridad en los mensajes II



El ACL FIPA

- La interoperabilidad se consigue mediante un lenguaje de comunicación de agentes
  - bien definido
  - sin ambigüedades
  - con un aparato formal sólido
- La base de un ACL está compuesta por los actos comunicativos

Actos comunicativos

- Cada uno de las directivas FIPA está definida mediante
  - el resumen en donde se explica resumido el significado del mensaje
  - el contenido del mensaje en donde se detalla qué tipo de contenido debe llevar
  - la descripción que es una explicación detallada del acto comnicativo
  - el modelo formal que es una descripción en SL (*Semantic Language*)
  - un ejemplo de mensaje con el acto comunicativo

Modelo formal de mensajes ACL

- Para la semántica de los mensajes, haremos uso de tres elementos de los agentes (sea i un agente y p una proposición)
  - B para *belief*: $B_i p$ significa que el agente i cree p
  - U para *uncertainty*: $U_i p$ significa que el agente i manifiesta cierta incertidumbre sobre p pero cree que es más probable que $\neg p$
  - C para *choice*: $C_i p$ quiere decir que el agente i en este instante desea que p se cumpla

  Como se puede comprobar son actitudes mentales.
- Secuencia de eventos (acciones): Una secuencia puede estar formado por uno o varios eventos de entre los cuales podemos encontrar el evento vacío
- las acciones (i.e. eventos) se combinan
  - $a_1; a_2$ es una secuencia en donde a_2 sigue a a_1
  - $a_1 | a_2$ es una opción no determinista en la cual ocurre a_1 o a_2 pero no ambos

Más sobre el modelo formal de mensajes ACL

Nuevos predicados, como son *Feasible*, *Done* y *Agent*:

- $Feasible(a, p)$ significa que a puede ejecutarse y que si se ejecuta, entonces p se hará cierto inmediatamente después.
- $Done(a, p)$ significa que a se ejectó y que p se hizo cierto justamente después.
- $Agent(i, a)$ significa que el agente i es el único que bien ejecutó, está ejecutando o ejecutará las acciones que aparecen en a.
- $Single(a)$ significa que la acción a no es una secuencia. Cualquier acción individual es simple. El acto compuesto $a; b$ no es simple. La acción $a|b$ es simple si lo son a y b.

Objetivo persistente y elementos asociados a un AC

Un agente i tiene un objetivo persistente p si i tiene a p como objetivo y su compromo con él es del tipo *single minded*.

- Una intención es entonces un objetivo persistente que obliga al agente a actuar.
- Denotamos con $PG_i p$ que el agente i tiene a p como objetivo persistente y con $I_i p$ que el agente i se dispone a cumplir p.

Un acto comunicativo lleva asociados dos componentes importantes

1. el efecto racional (RE)
2. las precondiciones para la realización (FP)

Abreviaciones, como aparecen en el documento

- $Feasible(a) \equiv Feasible(a, true)$
- $Done(a) \equiv Done(a, true)$
- $Possible(\phi) \equiv (\exists a) Feasible(a, \phi)$
- $Bif_i \phi \equiv B_i \phi \vee B_i \neg \phi$ (es decir, que el agente i o bien cree ϕ o bien cree $\neg \phi$)

Propiedades *mentales* de un Agente FIPA

Intención es igual a acción

La intención de un agente por cumplir un cierto objetivo genera directamente la realización de un acto conocido por el agente. Más aun, el acto es tal que su RE corresponde con el objetivo del agente, y además el agente no tiene razón alguna para no llevarlo a cabo. Si lo formalizamos, sea a_k un acto tal que (1) $(\exists x)B_i a_k = x$ y (2) p es el RE de a_k y (3) $\neg C_i \neg Possible(Done(a_k))$, entonces la siguiente fórmula es válida

$$\models I_i p \rightarrow I_i Done(a_1 | \ldots | a_n),$$

en donde $a_1, \ldots, a_n$ son todos los actos de tipo a_k.

Propiedades *mentales* de un Agente FIPA

FPs generan intenciones

Un agente debe tener la intención de satisfacer todas sus FPs. siempre que un agente elija una acción. Si lo formalizamos, tenemos

$$\models I_i Done(a) \rightarrow B_i Feasible(a) \vee I_i B_i Feasible(a)$$

Propiedades *mentales* de un Agente FIPA

Acto comunicativo equivale a RE del acto

Si un agente tiene la intención de llevar a cabo un acto comunicativo, entonces necesariamente tiene la intención de cumplir el RE del acto. Formalizado

$$\models I_i Done(a) \rightarrow I_i RE(a)$$

Propiedades *mentales* de un Agente FIPA

ACs observados inducen a creer en REs perseguidos

Cuando un agente observa un acto comunicativo realizado por otro agente, el primero debería creer que el agente que realiza el acto tiene la intención de cumplir el RE correspondiente. Este efecto se denomina efecto intencional. Si lo formalizamos

$$\models B_i(Done(a) \wedge Agent(j, a) \rightarrow I_j B_i I_j RE(a))$$

Propiedades *mentales* de un Agente FIPA

Persistencia en las *Feasibility Preconditions*

Cuando se ha realizado un acto comunicativo, es obligado para un agente creer que las FP correspondientes a ese acto se siguen cumpliendo (siempre que no se refirieran a un instante de tiempo concreto). Formalizarlo es sencillo

$$\models B_i(Done(a) \rightarrow FP(a))$$

Notación para la definición de un AC

Un modelo de acto comunicativo (CA) se representará como sigue:

$$\begin{array}{l} < i, act(j, C) > \\ FP : \phi_1 \\ RE : \phi_2 \end{array},$$

en donde i es el agente que ejecuta el CA, j es el receptor, act es el nombre de la performativa, C se refiere al contenido del mensaje (i.e. relativo al dominio de aplicación) y ϕ_1 y ϕ_2 son proposiciones de la lógica. Obsérvese que el mensaje sería

```
(act
  :sender i
  :receiver j
  :content C)
```

El acto comunicativo `inform`

Semántica informal

- Un agente i es capaz de informar a un agente j de que una proposición p es verdad solo si i cree que p es verdad (i.e. $B_i p$)
- El agente i usará `inform` solo si cree que j no conoce p o su negación (i cree que j cree la negación de p, i debería mandarle un `disconfirm`, o un `confirm` si j no está seguro del valor de verdad de p)

La formalización queda como sigue:

$$< i, INFORM(j, \phi) >$$
$$FP : B_i\phi \wedge \neg B_i(Bif_j\phi \vee Uif_j\phi)$$
$$RE : B_j\phi$$

El acto comunicativo `inform`

Ejemplo: el ajente *i* informa al *j* que (es verdad) está lloviendo hoy

```
(inform
   :sender (agent-identifier :name i)
   :receiver (set (agent-identifier :name j))
   :content "weather (today, raining)"
   :language Prolog)
```

El acto comunicativo `request`

Semántica informal

- de entre las FP de la acción a, se cumplen las correspondientes a actitudes del agente i
- el agente i cree que el agente j es el adecuado para ello, y que no tiene a la acción a como objetivo persistente (si así fuera no habría que hacer nada)

Formalizado es:

$$< i, REQUEST(j, a) >$$
$$FP : PF(a)[i \backslash j] \wedge B_i Agent(j, a) \wedge B_i \neg PG_j Done(a)$$
$$RE : Done(a)$$

El acto comunicativo `request`

Ejemplo: el ajente *i* pide al *j* que abra un fichero

```
(request
  :sender (agent-identifier :name i)
  :receiver (set (agent-identifier :name j))
  :content "open \"db.txt\" for input"
  :language vb)
```

El acto comunicativo `inform-if`

Semántica informal

- Es una acción del tipo macro (se ha de ejecutar realmente un inform)
- Si el agente que recibe la acción cree que la sentencia es verdad, informará de manera afirmativa, si no indicará que es falsa

Formalizado tenemos:

$$
\begin{aligned}
&< i, INFORM - IF(j, \phi) > \equiv \\
&< i, INFORM(j, \phi) > \mid < i, INFORM(j, \neg\phi) > \\
&FP : Bif_i\phi \wedge \neg B_i(Bif_j\phi \vee Uif_j\phi) \\
&RE : Bif_j\phi
\end{aligned}
$$

Ejemplo de uso del `inform-if`

El agente *i* pide al agente *j* que le informe de que si Lannion está en Normandía con el siguiente mensaje:

```
(request
:sender (agent-identifier :name i)
:receiver
  (set (agent-identifier :name j))
:content
  ‘‘((action
    (agent-identifier :name j)
    (inform-if
    :sender (agent-identifier :name j)
    :receiver (set
      (agent-identifier :name i))
    :content \"in(lannion,normandy)\"
    :language Prolog)))’’
:language fipa-sl)
```

y el agente *j* le responde que no

```
(inform
  :sender (agent-identifier :name j)
  :receiver (set
    (agent-identifier :name i))
  :content ‘‘\+ in(lannion, normandy)’’
  :language Prolog)
```

El acto comunicativo `refuse`

Semántica informal

- Se realiza cuando el agente emisor no puede cumplir todas las FP de las acciones que se le ha pedido que ejecute, e.g.
  - no se sabe por algo que se pregunta
  - no se tienen privilegios para ejecutar una acción
- Se acompaña de una explicación

Formalmente

$$
\begin{aligned}
&< i, REFUSE(j, < i, act >, \phi) > \equiv \\
&< i, DISCONFIRM(j, Feasible(i, < act >)) >; \\
&< i, INFORM(j, \phi \wedge \neg Done(< i, act >) \wedge \neg I_i Done(< i, act >)) > \\
&\quad FP : B_i \neg Feasible(< i, act >) \wedge B_i(B_j Feasible(< i, act >) \vee \\
&\qquad U_j Feasible(< i, act >)) \wedge B_i \alpha \wedge \neg B_i(Bif_j \alpha \vee Uif_j \alpha) \\
&\quad RE : B_j \neg Feasible(< i, act >) \wedge B_j \alpha
\end{aligned}
$$

siendo $\alpha = \phi \wedge \neg Done(< i, act >) \wedge \neg I_i Done(< i, act >)$

Recomendaciones FIPA para PIs

Protocolos de interacción Los siguientes son estándares:

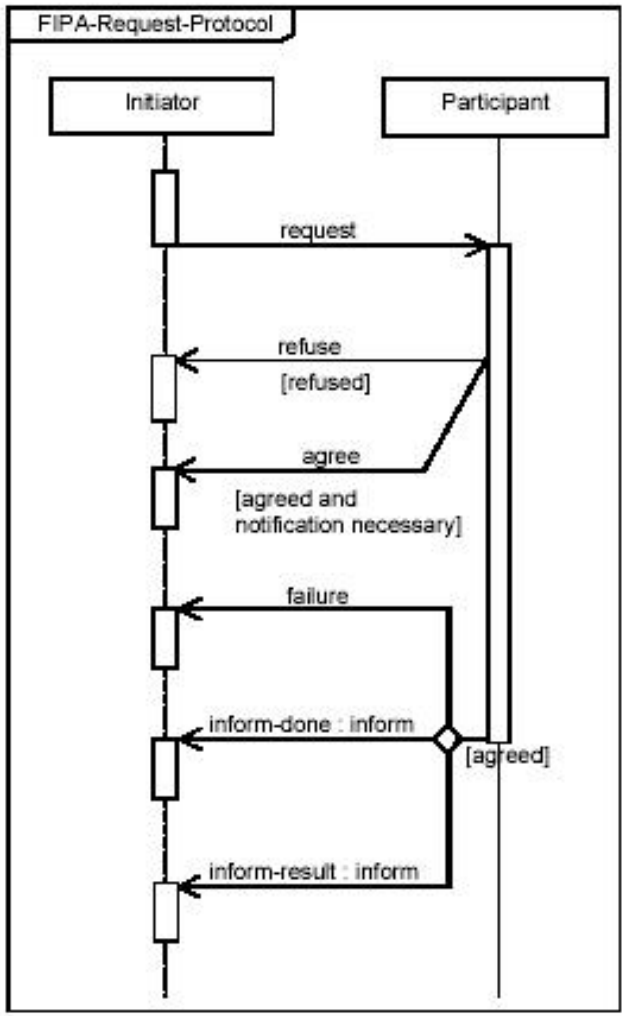
- FIPA Request Interaction Protocol Specification
- FIPA Query Interaction Protocol Specification
- FIPA Request When Interaction Protocol Specification
- FIPA Contract Net Interaction Protocol Specification
- FIPA Iterated Contract Net Interaction Protocol Specification
- FIPA Brokering Interaction Protocol Specification
- FIPA Recruiting Interaction Protocol Specification
- FIPA Subscribe Interaction Protocol Specification
- FIPA Propose Interaction Protocol Specification

y las subastas alemana e inglesa mantienen el estatus de experimental

- FIPA English Auction Interaction Protocol Specification
- FIPA Dutch Auction Interaction Protocol Specification

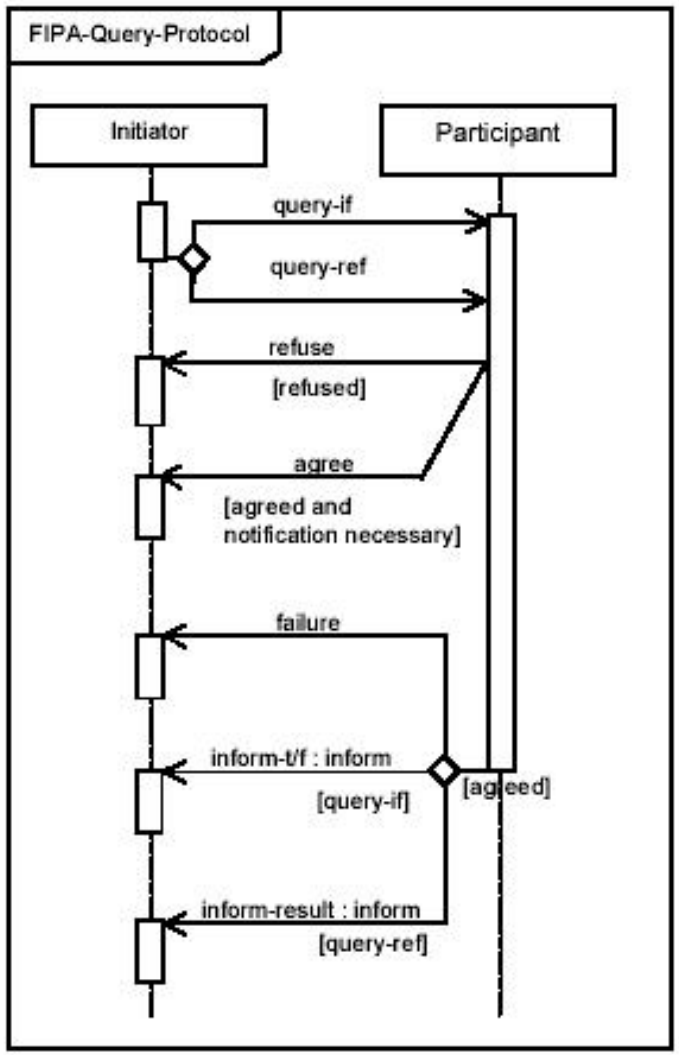
El PI `fipa-request`

El protocolo `fipa-request` permite a un agente pedir a otro que realice una determinada acción.



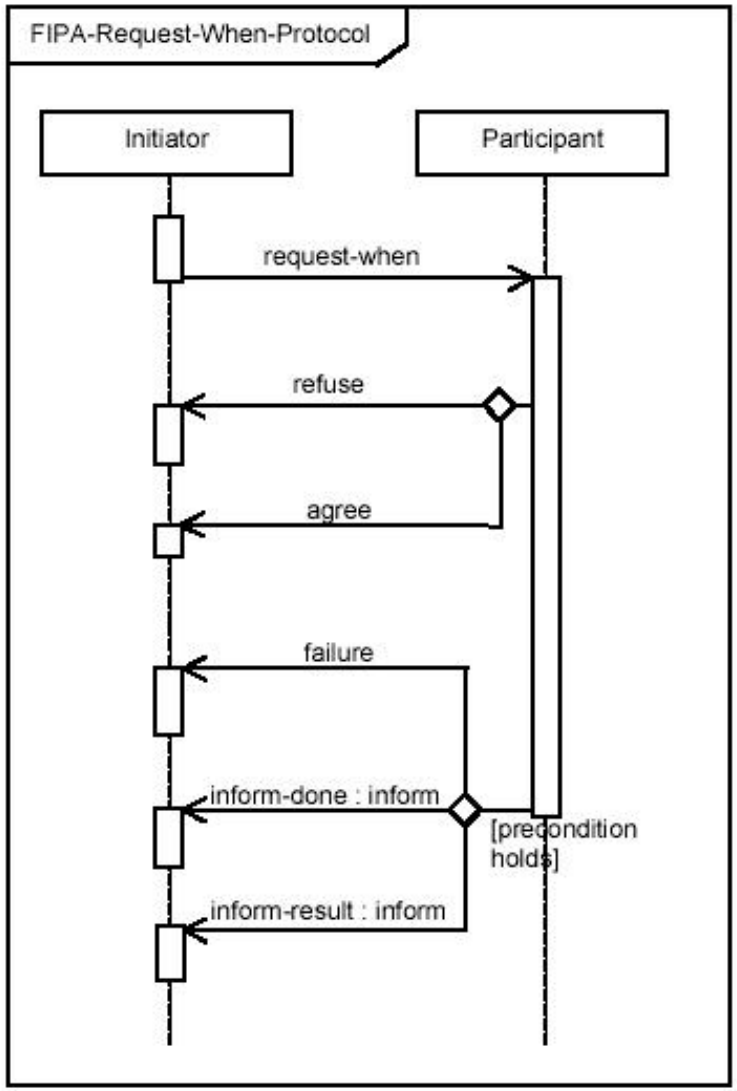
El PI `fipa-query`

El protocolo `fipa-query` es similar al anterior solo que la acción a ejecutar es de un tipo concreto: `inform` para informar sobre algo.



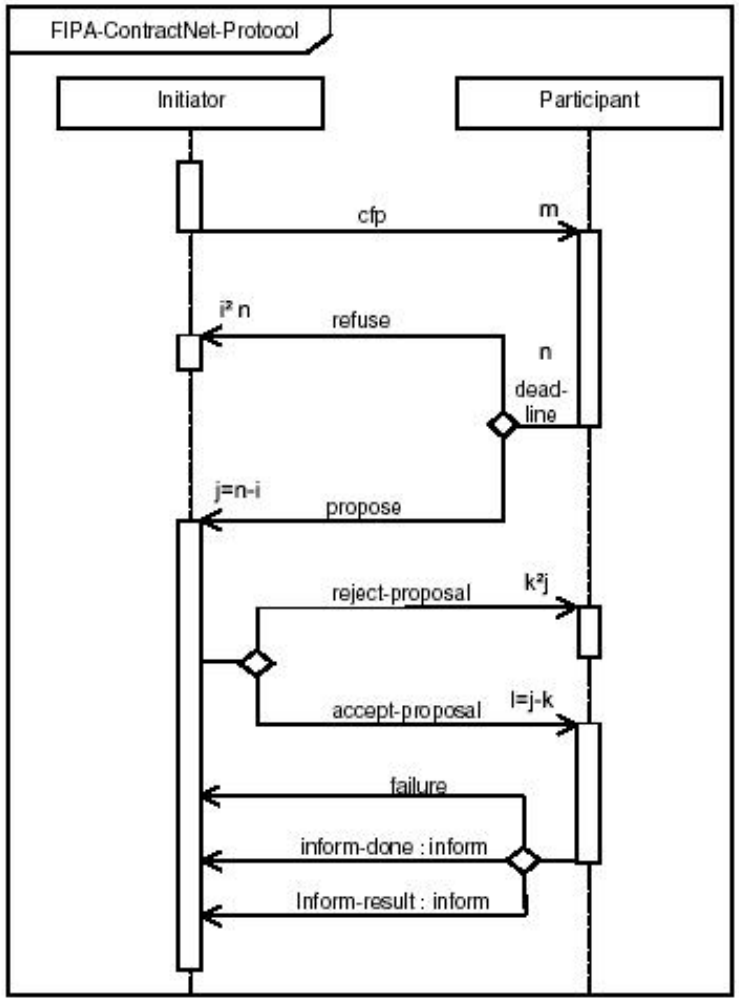
El PI `fipa-request-when`

Es posible indicar a un agente que ejecute una acción de manera diferida, cuando una determinada precondición se haga cierta. Para eso tenemos el protocolo de interacción `fipa-request-when`



El PI `fipa-contract-net`

El fipa-contract-net está inspirado en la propuesta de Smith solo que debido a la autonomía hay que incluir mensajes para confirmación y rechazo.

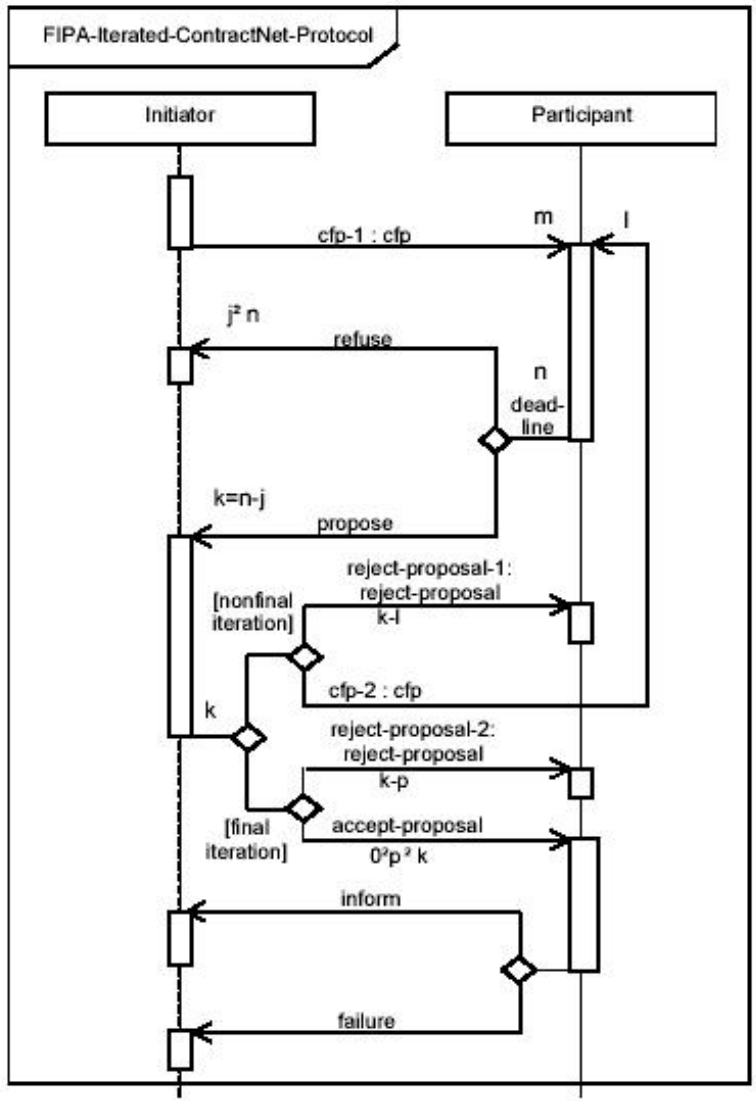


El PI `fipa-contract-net` (y II)

- Inicialmente, el manager, con el rol FIPA de iniciador, genera m mensajes del tipo `cfp` (*call for proposal*) y queda a la espera durante un determinado tiempo, después del cual no recibirá más mensajes (un total de n recibidos)
- Sea i el número de mensajes de tipo `refuse`, Tendremos entonces $j = n - i$ mensajes del tipo `propose`
- Para cada uno de los j mensajes, enviar posteriormente bien un `accept-proposal` o un `reject-proposal`
- Se informa del resultado con un `failure`, con un `inform` sin resultado o con el mismo acompañado del resultado.

El PI `fipa-iterated-contract-net`

- ligera variación del `fipa-contract-net` que permite el número de rondas que sea necesario
- tiene el propósito para el iniciador de la interacción de conseguir mejores ofertas de los interlocutores mediante argumentación o crítica de las ofertas anteriores



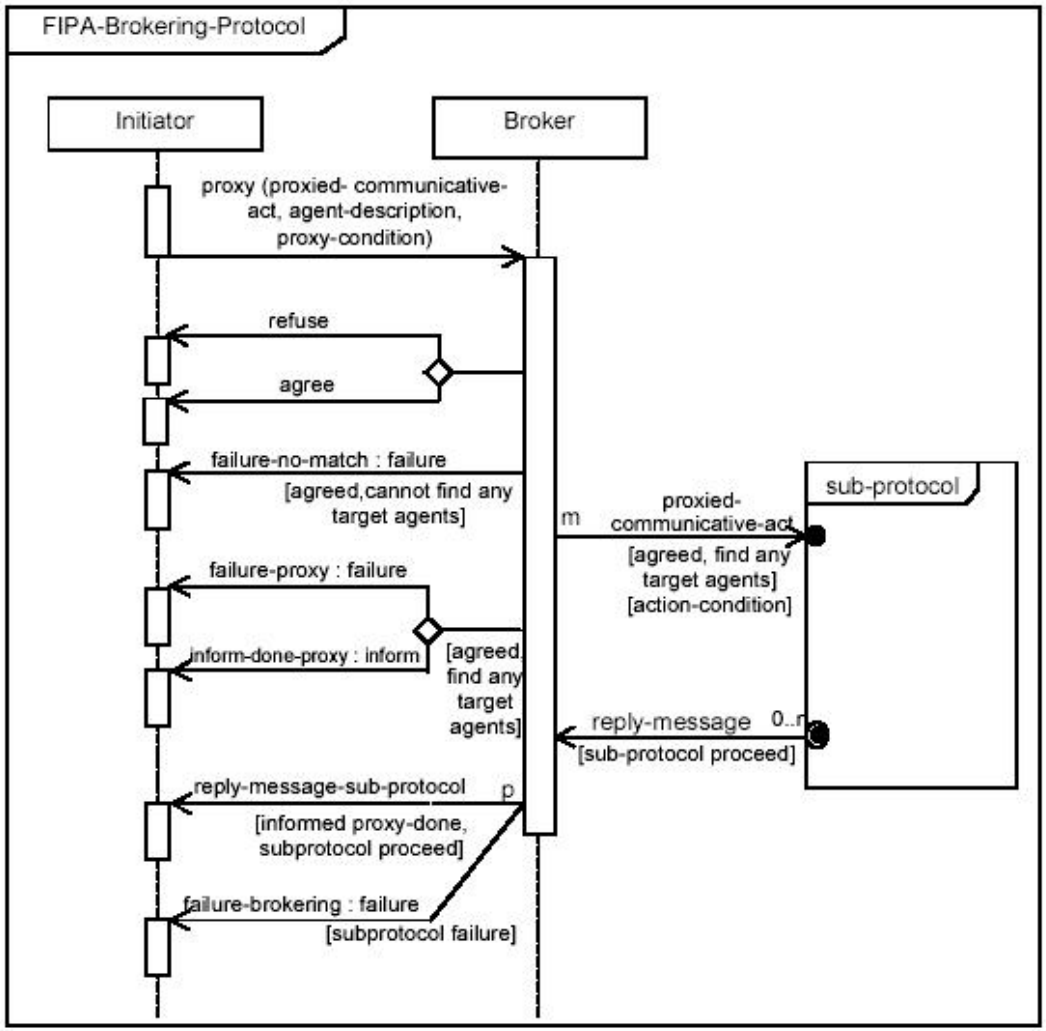
El PI `fipa-iterated-contract-net` (y II)

- Inicialmente, el agente con el rol de iniciador de la conversación genera m mensajes $\widehat{\text{cfp}}$
- Después de un deadline de espera, el iniciador recoge digamos n ofertas
- sean un total de j las que rechazan realizar la tarea mediante `refuse`
- Tenemos entonces $k = n - j$ ofertas de agentes que están dispuestos a realizar la tarea con `prospose`
- Si la iteración no es la última, de entre el total de k ofertas recibidas, se rechazarán algunas directamente, $k - l$, y se aceptarán otras tantas l.
- De entre las aceptadas, se elabora una contraoferta para cada agente y se envuelve en un nuevo `cfp`

El PI `fipa-brokering`

- Tiene como propósito permitir interaccionar con otros agentes a través de un mediador (el broker)
- `proxy` es una macro (incluye otro acto comunicativo que el broker debe hacer llegar al seleccionado o seleccionados)
- El broker devuelve los resultados mediante `reply-message-sub-protocol` (i.e. un `reply` con la respuesta en el cuerpo del mensaje)

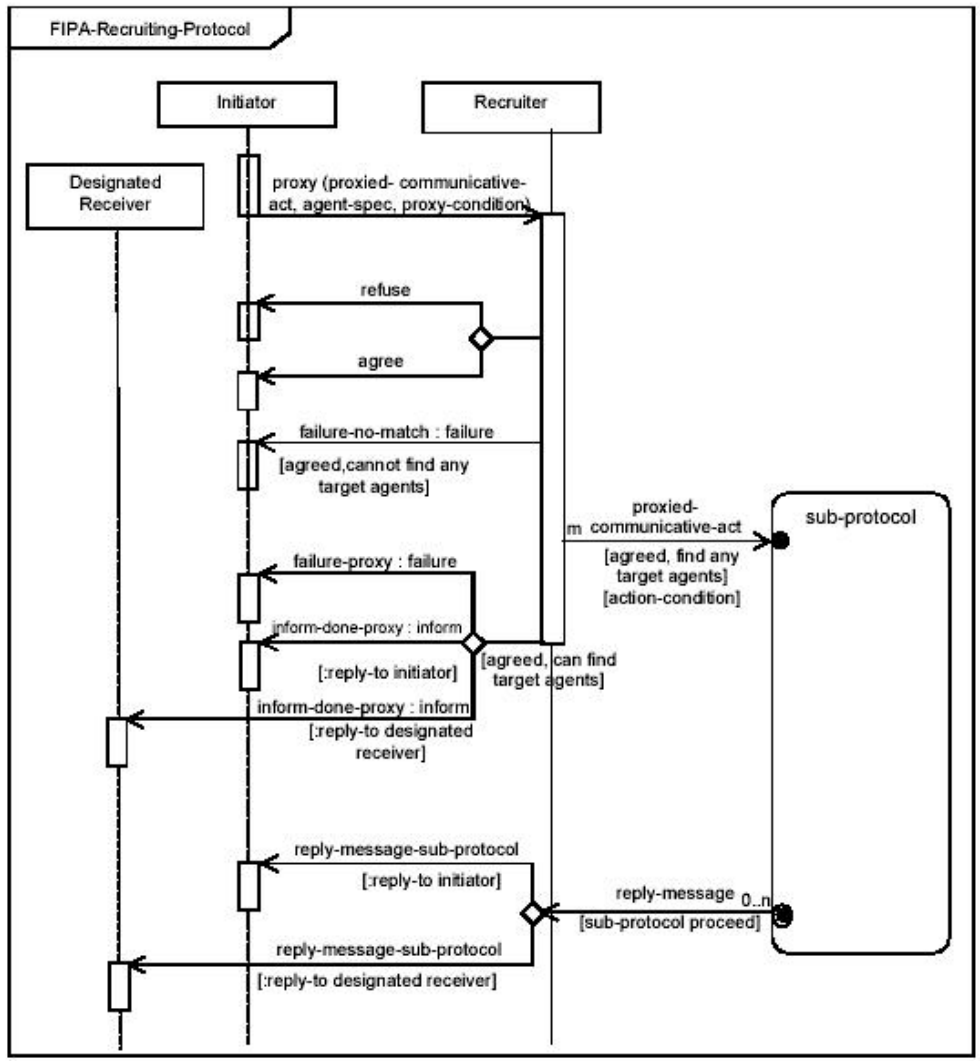
El PI `fipa-brokering` (y II)



El PI `fipa-recruiting`

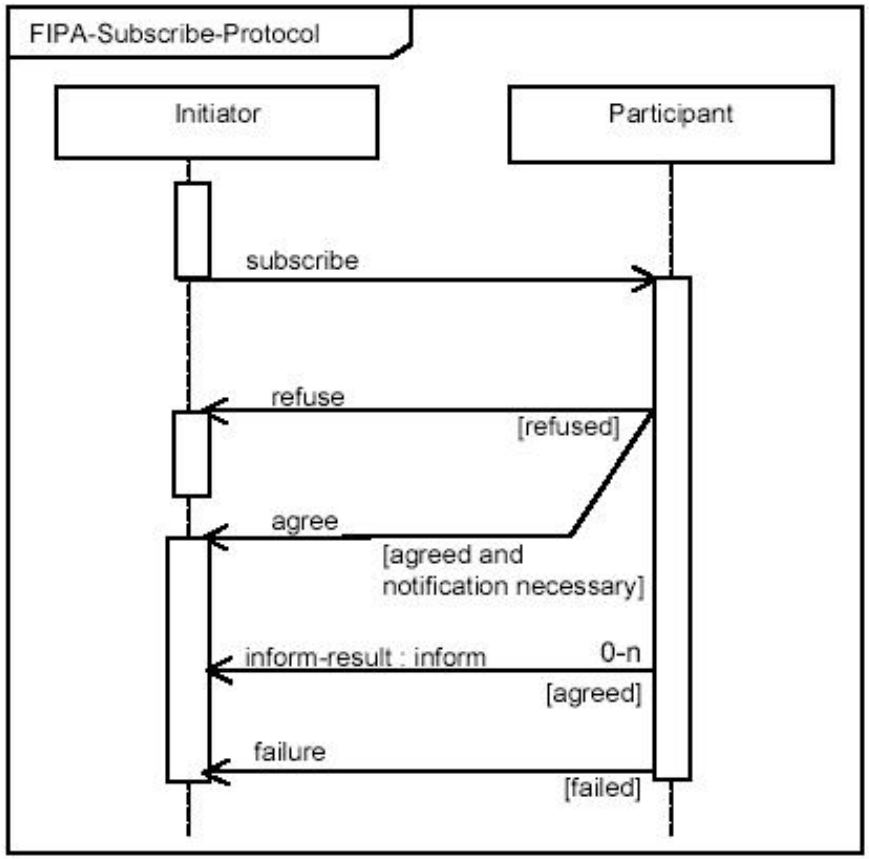
- Tiene como propósito reclutar agentes para interaccionar con ellos posteriormente
- Usado en entornos basados en mediadores
- Se diferencia del anterior en que ya no es el broker el que interactúa con los seleccionados, sino el receptro designado

El PI `fipa-recruiting` (y II)



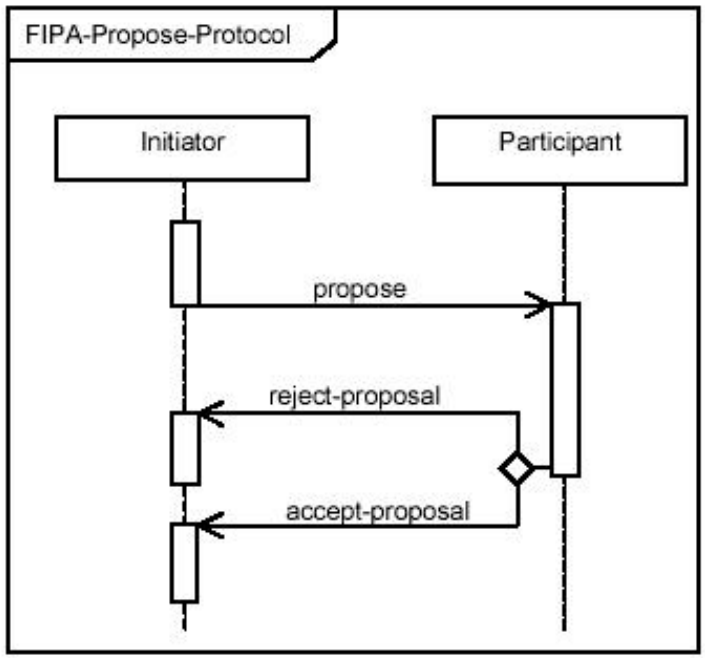
El PI `fipa-subscribe`

De utilidad para subscripción a un servicio de notificación de eventos. El agente participante se compromete (mediante la emisión de un `agree`) a informar mediante `inform` cada vez que el evento se produzca



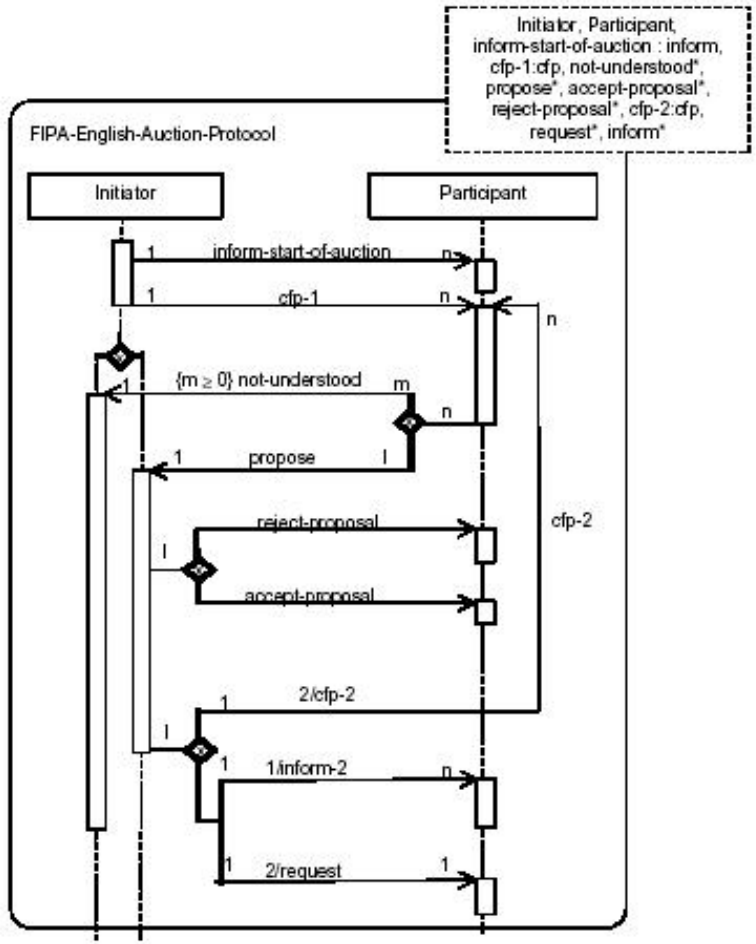
El PI `fipa-propose`

El agente iniciador propone al otro agente particante, que lo ha de supervisar, el realizar una acción



El PI `fipa-english-auction`

El iniciador ofrece un precio. Si ningún posible comprador realiza una oferta, se vuelve a ofrecer otro precio. Se selecciona el mejor de entre los participantes que emiten un `propose`.



Recordatorio de las especificaciones relacionadas

Lenguajes de contenido:

- FIPA SL Content Language Specification
  - El estándar FIPA
- FIPA CCL Content Language Specification
  - Para representar problemas de satisfacción de restricciones
- FIPA KIF Content Language Specification
  - No es posible representar otra cosa que no sea una expresión con valor de verdad o hechos
- FIPA RDF Content Language Specification
  - El más práctico y utilizado

de las cuales, solamente la correspondiente al lenguaje SL tiene consideración de estándar

El estándar SL

- Sintaxis muy parecida a la de LISP
- Sus f.b.f. pueden representar
  - Proposiciones, con un valor de verdad determinado en un contexto concreto. Las proposiciones se usan, por ejemplo, con el acto comunicativo `inform`.
  - Acciones que pueden ser ejecutadas. Estas acciones pueden ser simples o complejas usando para estas últimas operadores de secuenciación o alternancia. Por ejemplo, las acciones son el contenido de mensajes `request`.
  - Una expresión referencial de identificación (IRE), que identifica objetos en el dominio. Este es básicamente el operador referencial que tienen todos los lenguajes lógicos y que se utilizan, por ejemplo, en la performativa `inform-ref`.
- Ver recomendación

El experimental RDF

- RDF es una infraestructura que permite la publicación, codificación, intercambio y reusabilidad de metadatos (i.e. datos estructurados sobre los datos que describen)
- Reglas comunes para sintaxis, semántica y estructura (XML para serialización)
- Particularizado/extendido para cada dominio de aplicación (e.g. FIPA)
- Modelo para descripción de recursos mediante propiedades (y recursos)
- Una propiedad (*property*) representa una característica de interés de un recurso expresada con valores atómicos (i.e. cadenas de caracteres, número, valores de verdad)
- Aseveración (*statement*): recurso + propiedad + valor
  - Sujeto (recurso) + predicado (propiedad) + objeto (valor de la propiedad)

Notación gráfica en RDF

Si pensamos en las sentencias siguientes.

1. John Smith es el autor de Documento1
2. El autor de Documento1 es John Smith

Para nosotros estas dos sentencias tienen el mismo significado.

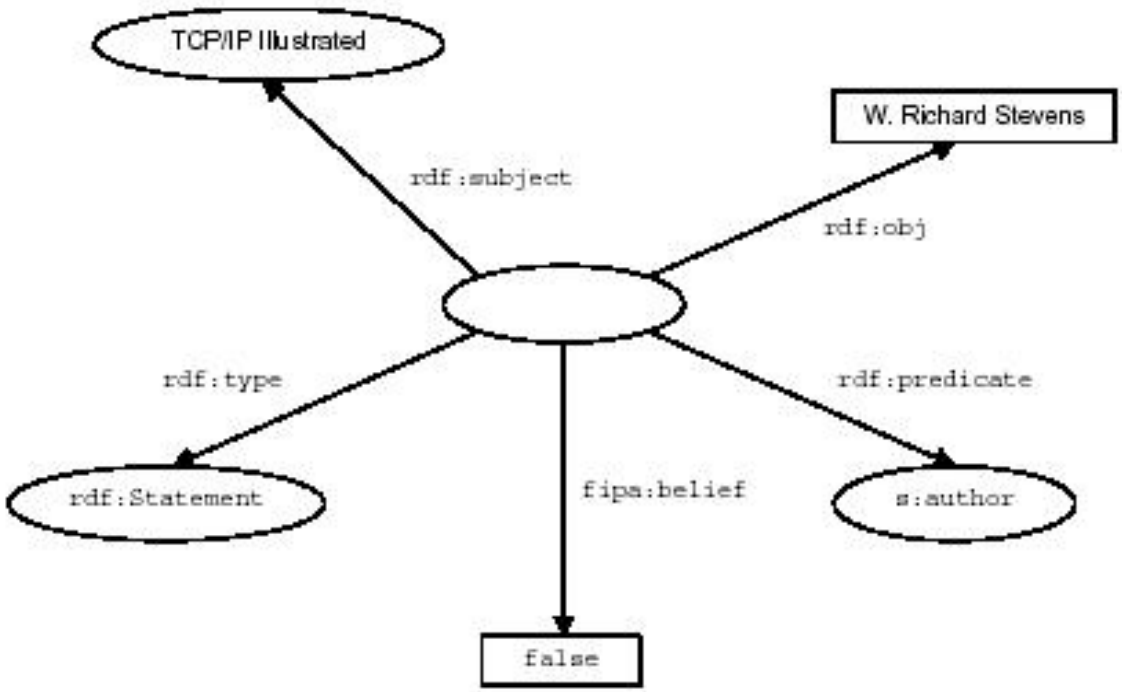
Si las representaramos tal cual mediante lógica de predicados, las computadoras las distinguirían.

Notación gráfica en RDF (y II)

RDF propone un modelo gráfico en donde se eliminan este tipo de ambiguedades mediante sujeto (óvalo), predicado (relación) y complemento (rectángulo)

Notación gráfica en RDF (y III)

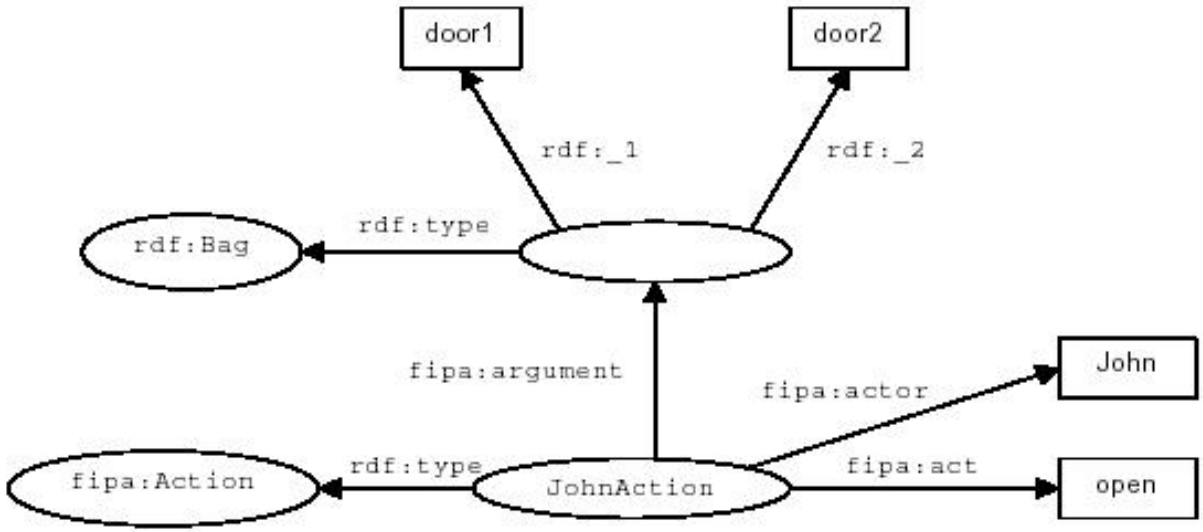
El complemento puede ser, a su vez, otro recurso (recursividad ilimitada)



Colecciones de recursos

- En ocasiones no es suficiente el referirnos a un único recurso sino a una colección de ellos
- RDF proporciona tres tipos
  1. `Bag`: conjunto de recursos o literales (i.e. valores tal cual para una propiedad), sin órden y con posibilidad de estar repetidos
  2. `Sequence`: como el primero (i.e. recursos y literales) salvo en que todos los elementos tienen un orden establecido
  3. `Alternative`: representa posibles alternativas de literales para una propiedad

Colecciones de recursos (y II)



Las propiedades que se refieren a cada uno de los recursos miembros de la colección se deminan simplemente con las etiquetas `rdf:_1`, `rdf:_2`, `rdf:_3`,etc.

XML para serialización

- La sintaxis de RDF se basa en XML aunque en la recomendación no es obligatorio usarlo para expresar RDF

```
<?xml version="1.0"?>
  <rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:s="http://description.org/schema/">
  <rdf:Description ID="TCP/IP Illustrated">
    <s:author>W. Richard Stevens</s:author>
  </rdf:Description> </rdf:RDF>
```

Extensiones de RDF para FIPA

En FIPA vamos a necesitar expresar, mediante RDF, los siguientes elementos:

1. Objetos: en FIPA vamos a poder usar los URI que designan recursos en RDF como referencias para los objetos (para resolver una referencia RDF podremos usar el acto comunicativo `query-ref`, seguido (como siempre) de un `inform` que describe el objeto)
2. Proposiciones: podemos ver una equivalencia entre sentecias RDF y proposiciones ACL (se añade un valor de verdad a las sentencias)
3. Accciones: Para describir una `fipa:Action` vamos a usar tres nuevas propiedades

Valor de verdad en sentencias RDF

Definimos una nueva propiedad, con etiqueta `belief`, en el espacio de nombres `http://www.fipa.org/schemas#`

```
<?xml vesion=''1.0''?>
<rdf:RDF xmlns:rdf=''http://www.w3.org/1999/02/22-rdf-syntax-ns#''
         xmlns:rdf=''http//www.w3.org/TR/1999/PR-rdf-schema-19990303#''>

  <rdfs:Class rdf:ID=''http://www.fipa.org/schemas#Proposition''>
    <rdfs:label xsml:lang=''en''>proposition</rdfs:label>
    <rdfs:label xsml:lang=''fr''>proposition</rdfs:label>
    <rdfs:comment>This describes the set of propositions</rdfs:comment>
    <rdfs:subclassOf rdf:resource=''http://www.w3.org/1999/02/22-rdf-syntax-ns#Statement''/>
  </rdfs:Class>

  <rdfs:ConstraintProperty rdf:ID=''http://www.fipa.org/schemas#belief''>
    <rdfs:label xml:lang=''en''>belief</rdfs:label>
    <rdfs:label xml:lang=''fr''>acte</rdf:label>
    <rdfs:domain rdf:resource=''#Proposition''/>
    <rdfs:range rdf:resource=''http://www.w3c.org/TR/1999/PR-rdf-schema-19990303#Literal''/>
  </rdfs:ConstraintProperty>
</rdf:RDF>
```

Valor de verdad en sentencias RDF (y II)

Y ahora podemos expresar

```
<?xml version=''1.0''?>
<rdf:rdf xmlns:rdf=''http://www.w3.org/RDF/RDF''
         xmlns:fipa=''http://www.fipa.org/schemas/fipa-rdf0#''>

  <fipa:Proposition>
    <rdf.subject>TCP/IP Illustrated</rdf:subject>
    <rdf:predicate rdf:resource=''http://description.org/schema#author''/>
    <rdf:object>W. Richard Stevens</rdf:object/>
    <fipa:belief>true</fipa:belief>
  </fipa:Proposition>
</rdf:RDF>
```

Especificando acciones con RDF en FIPA

Acción: actividad concreta llevada acabo por un objeto.

Vamos a extender un recursos con tres nuevas propiedades para describir `fipa:Action`

- Un acto: se refiere a la parte operativa. Podemos usar para este campo el tipo de la acción o una descripción. Para poder especificar tipos de acciónes simplemente especializaremos la clase `fipa:Action` con nuevas clases de acciones.
- Un actor: el identificador de la entidad que va a/debería/puede realizar la acción.
- un argumento: opcional, podemos usarlo para indicar los parámetros de la acción.

Especificando acciones con RDF en FIPA (y II)

Para expresar “Juan abre puerta1 y puerta2”

```
<’xml version=’’1.0’’?>
<rdf:rdf xmlns:rdf=’’http://www.w3.org/RDF/RDF’’
         xmlns:fipa=’’http://www.fipa.org/schemas/fipa-rdf0#’’>

  <fipa:Action rdf:ID=’’JuanAccion1’’>
    <fipa:actor>Juan</fipa:actor>
    <fipa:act>open</fipa:act>
    <fipa:argument>
      <rdf:bag>
        <rdf:li>Puerta1</rdf:li>
        <rdf:li>Puerta2</rdf:li>
      </rdf:bag>
    </fipa:argument>
  </fipa:Action>
</rdf:rdf>
```

Especificando acciones con RDF en FIPA (y III)

Pero, ¿cómo se representa el resultado de una acción en el correspondiente `inform` tras el `request` + `agree`?

- FIPA propone usar la clase RDF `Description`
- y añadir dos atributos `fipa:done` con dominio en los literales y tomando un valor de verdad
- `result` con una expresión que indique el resultado.

Especificando acciones con RDF en FIPA (y IV)

El Agente Maria le dice al agente Juan que cierre las puertas 1 y 2

```
(request
  :sender Maria
  :receiver Juan
  :content (
    <'xml version=''1.0''?>
    <rdf:rdf xmlns:rdf=''http://www.w3.org/RDF/RDF''
             xmlns:fipa=''http://www.fipa.org/schemas/fipa-rdf0#''>

      <fipa:Action rdf:ID=''JuanAccion1''>
        <fipa:actor>Juan</fipa:actor>
        <fipa:act>open</fipa:act>
        <fipa:argument>
          <rdf:bag>
            <rdf:li>Puerta1</rdf:li>
            <rdf:li>Puerta2</rdf:li>
          </rdf:bag>
        </fipa:argument>
      </fipa:Action>
    </rdf:RDF>)
  :language fipa-rdf0)
```

Especificando acciones con RDF en FIPA (y V)

El agente Juan le responde con el resultado

```
(inform
  :sender Juan
  :receiver Maria
  :content (
    <’xml version=’’1.0’’?>
    <rdf:rdf xmlns:rdf=’’http://www.w3.org/RDF/RDF’’
             xmlns:fipa=’’http://www.fipa.org/schemas/fipa-rdf0#’’>

      <rdf:Description about=’’#JuanAccion1’’>
        <fipa:done>true</fipa:done>
        <fipa:result>puertas abiertas</fipa:result>
      </fipa:Description>
    </rdf:RDF>)
  :language fipa-rdf0)
```

Conclusiones

- FIPA proporciona un conjunto de estándares para construcción de SMAs
- Estándares para
  - Comunicación entre agentes
  - Gestión de agentes en plataformas
  - Coordinación entre agentes
  - Expresión de conocimiento en cuerpo de mensajes
- Sencillez de conceptos
- Estándares concretos y no muy extensos (altamente testeables)