Agent Standards

FIPA Agent Framework

Heimo Laamanen
01. 03. 2004
Content

- Introduction
- FIPA Organization
- FIPA Specifications
- FIPA Abstract Architecture
- FIPA Agent Management
- FIPA ACL and Content Languages
- FIPA Message Transport Service
- Example:
  - FIPA Nomadic Application Support
Introduction

To standardize or Not to standardize

01.03.2004

FIPA Agent Framework
Introduction

- To standardize
  - Interoperability

Semantic?

Language?

Transmission?

Culture?
Introduction

- Not to Standardize

  - Open competition - the best to win
  - Slow and bureaucratic process.
  - Conformance difficult and in some cases impossible.
  - Standard is always a compromise, not the best solution.
Introduction

FIPA

Foundation for Intelligent Physical Agents
www.fipa.org

OMG Agent PSIG

http://www.objs.com/agent/index.html
- ontology
- mobile agents / mobility in general

01.03.2004 FIPA Agent Framework
Content

- Introduction
- FIPA Organization
- FIPA Specifications
- FIPA Abstract Architecture
- FIPA Agent Management
- FIPA ACL and Content Languages
- FIPA Message Transport Service
- Example:
  - FIPA Nomadic Application Support
  - CRUMPET
FIPA Organization

Board of Directors

http://www.fipa.org/about/board.html

FIPA Architecture Board

http://www.fipa.org/about/fab.html

Technical Committees

http://www.fipa.org/activities/committees.html

01.03.2004  FIPA Agent Framework
FIPA Standardization Process

Workplan

http://www.fipa.org/docs/output/f-out-00010/

Call for Technology

Example:

http://www.fipa.org/docs/output/f-out-00062/

Specification lifecycle

http://www.fipa.org/specifications/lifecycle.html
Content

- Introduction
- FIPA Organization
- FIPA Specifications
- FIPA Abstract Architecture
- FIPA Agent Management
- FIPA ACL and Content Languages
- FIPA Message Transport Service
- Example:
  - FIPA Nomadic Application Support
  - CRUMPET
FIPA Specifications

Abstract Architecture

Applications

ACL

Agent Management

Agent Message Transport

01.03.2004 FIPA Agent Framework
FIPA Specifications

- Applications

- FIPA00014 FIPA Nomadic Application Support Specification
- FIPA00079 FIPA Agent Software Integration Specification
- FIPA00080 FIPA Personal Travel Assistance Specification
- FIPA00081 FIPA Audio-Visual Entertainment and Broadcasting Specification
- FIPA00082 FIPA Network Management and Provisioning Specification
- FIPA00083 FIPA Personal Assistant Specification
- FIPA00092 FIPA Message Buffering Service Specification
Fipa Specifications

- ACL representations
  - FIPA00069 FIPA ACL Message Representation in Bit-Efficient Specification
  - FIPA00070 FIPA ACL Message Representation in String Specification
  - FIPA00071 FIPA ACL Message Representation in XML Specification

- Communicative acts
  - FIPA00037 FIPA Communicative Act Library Specification

- Agent communications
  - FIPA00061 FIPA ACL Message Structure Specification
FIPA Specifications

- Content languages
  - FIPA00007 FIPA Content Languages Specification
  - FIPA00008 FIPA SL Content Language Specification
  - FIPA00009 FIPA CCL Content Language Specification
  - FIPA00010 FIPA KIF Content Language Specification
  - FIPA00011 FIPA RDF Content Language Specification
FIPA Specifications

- **Agent management**
  - FIPA00023 FIPA Agent Management Specification
  - FIPA00087 FIPA Agent Management Support for Mobility Specification

- **Message Transport**
  - FIPA00067 FIPA Agent Message Transport Service Specification
  - FIPA00093 FIPA Messaging Interoperability Service Specification
FIPA Specifications

- Interaction protocols
  - FIPA00025 FIPA Interaction Protocol Library Specification
  - FIPA00026 FIPA Request Interaction Protocol Specification
  - FIPA00027 FIPA Query Interaction Protocol Specification
  - FIPA00028 FIPA Request When Interaction Protocol Specification
  - FIPA00030 FIPA Iterated Contract Net Interaction Protocol Specification
  - FIPA00031 FIPA English Auction Interaction Protocol Specification
  - FIPA00032 FIPA Dutch Auction Interaction Protocol Specification
  - FIPA00033 FIPA Brokering Interaction Protocol Specification
  - FIPA00034 FIPA Recruiting Interaction Protocol Specification
  - FIPA00036 FIPA Propose Interaction Protocol Specification
Content

- Introduction
- FIPA Organization
- FIPA Specifications
- FIPA Abstract Architecture
- FIPA Agent Management
- FIPA ACL and Content Languages
- FIPA Message Transport Service
- Example:
  - FIPA Nomadic Application Support
  - CRUMPET
FIPA Abstract Architecture

Architecture Technical Committee

The mission of the Architecture Technical Committee is to construct abstract architectural specifications that provide a framework in which services necessary to support the end-to-end interoperability of agents.

http://www.fipa.org/activities/architecture.html
Why abstract architecture?

- The first purpose is to foster interoperability and reusability.
- To achieve this, it is necessary to identify the elements of the architecture that must be codified.
- Specifically, if two or more systems use different technologies to achieve some functional purpose, it is necessary to identify the common characteristics of the various approaches.
- This leads to the identification of architectural abstractions: abstract designs that can be formally related to every valid implementation.
FIPA Abstract Architecture

Why abstract architecture?

- By describing systems abstractly, one can explore the relationships between fundamental elements of these agent systems.
- By describing the relationships between these elements, it becomes clearer how agent systems can be created so that they are interoperable.
- From this set of architectural elements and relations one can derive a broad set of possible concrete architectures, which will interoperate because they share a common abstract design.
Why abstract architecture?

- Because the abstract architecture permits the creation of multiple concrete realizations, it must provide mechanisms to permit them to interoperate.
- This includes providing transformations for both transport and encodings, as well as integrating these elements with the basic elements of the environment.
FIPA Abstract Architecture

Abstract Architecture

Specifications

Implementations
FIPA Abstract Architecture

Scope of abstract architecture
- Message transport interoperability
- Supporting various forms of ACL representations.
- Supporting various forms of content languages.
- Supporting multiple directory services representations.

Not specified
- Agent mobility.
- Agent lifecycle and management.
- Domains.
- Conversational policy.
FIPA Abstract Architecture

- The Abstract Architecture focuses on core interoperability between agents.
- These include:
  - Managing multiple message transport schemes.
  - Managing message encoding schemes.
  - Locating agents and services via directory services.
FIPA Abstract Architecture

Agent:

An agent is a computational process that implements the autonomous, communicating functionality of an application.
FIPA Abstract Architecture

Agent:

Basic Agent Relationships

- AgentName: FipaAgentName
- Address: WrappedAddress

0..n

Associated With

0..n

Registers With

Agent Platform

Directory Service

Agent Description

- AgentName: FIPA Agent Name
- AgentDestination: FIPA Agent Destination
- Properties: List of Agent Properties

Agent Name

Destination?

01.03.2004 FIPA Agent Framework
Agent’s relationships to other elements

- **Agent** is associated with one or more **agent-platforms**.
- **Agent** may have one or more **agent-names**.
- **Agent** may be reachable via a **destination**, using the **transport** corresponding to one of the **addresses** in the **destination**.
- **Agent** may send a **FIPA-message** to one or more **agents**.
- **Agent** may register with one or more **directory-services**.
- **Agent** may modify its **agent-description** as registered by a **directory-service**.
- **Agent** may delete its **agent-description** from a **directory-service**.
- **Agent** may query for an **agent-description** registered within a **directory-service**.
Agents and services

**Agents** communicate by exchanging messages which represent speech acts, and which are encoded in an *agent-communication-language*.

**Services** provide support services for agents. This version of the Abstract Architecture defines two support services:

- directory-services and
- message-transport-services.

**Services** may be implemented either as agents or as software that is accessed via method invocation, etc.
Agent messages

In FIPA agent systems agents communicate with one another, by sending messages.

Two fundamental aspects of message communication between agents are

- the message structure and
- the message transport.
FIPA Abstract Architecture

- Agent messages

- The structure of a message is a **key-value-tuple**
  - It is written in an **agent-communication-language**, such as FIPA ACL.
  - The **content** of the message is expressed in a **content-language**, such as KIF or SL.
  - The **content-language** may reference an **ontology**, which grounds the concepts being discussed in the **content**.
  - The messages also contain the sender and receiver names, expressed as **agent-names**.
    - **Agent-names** are globally unique name identifiers for an agent.
  - **Messages** can recursively contain other messages.
Agent name

- An **agent name** is a means to identify an **agent** to other agents and to platform services.
- The primary role of an agent name is to allow agents to correlate propositional attitudes with the particular agents that are believed to hold those attitudes.
- An **agent-name** is an identifier (a.k.a. GUID, globally unique identifier) that is associated with the agent when the agent is created or when it registers for the first time.
Agent description

- An **agent-description** is a composite entity containing the **name**, **destination**, and other attributes of an **agent**.

- **Agent-description** contains the **agent-name** of the **agent** to which it refers.

- **Agent-description** contains the **destination** of the **agent** to which it refers.

- **Agent-description** is available from a **directory-service**.

- **Agent-description** contains **agent-attributes**.
FIPA Abstract Architecture

- Agent platform
  - An abstract **agent platform** is a collection of services that are closely coupled.
  - **Agent-platform** can host one or more **agents**.
  - **Agent-platform** provides a **directory-service**.
  - **Agent-platform** provides a **message-transport-service**.
Directory service

- The basic role of the directory-service function is to provide a location where agents register directory-entries.
- Other agents can search the directory-entries to find agents with which they wish to interact.
- The directory-entry is a key-value-tuple consisting of at least the following two key-value-pairs:
  - **Agent-name**
    - A globally unique name for the agent.
  - **Locator**
    - One or more transport-descriptors that describe the transport-type and the transport-specific-address to use to communicate with that agent.
A directory-service is a shared information repository in which agents may publish their agent-descriptions and in which they may search for agent-descriptions of interest.
Directory service

- Agent may register its agent-description with a directory-service.
- Agent may modify its agent-description as registered by a directory-service.
- Agent may de-register, or delete its agent-description from a directory-service.
- Agent may search for an agent-description registered within a directory-service.
FIPA Abstract Architecture

- Directory service
  - Variety of directory services
  - Abstract directory elements
    - agent-name, agent-address, agent-type, agent-services, interaction-protocols, ontology, ownership, fixed-properties, negotiable-properties, transport-properties, df-state.
FIPA Abstract Architecture

- Interoperability
  - Homogeneous
    - for example, same or same type of agent platform.
  - Via a gateway
    - gateway is connected to two different platforms.
FIPA Abstract Architecture

Gateway

A gateway is an element of an agent system that is capable of mediating some form of interoperability with another agent system.

Within the abstract architecture, there are two services that may be provided by a gateway:

- A gateway may provide message transfer between two agents that use different transports or message-encoding-representations.
- A gateway may support agent advertisement and discovery by translating between different directory-services.
FIPA Abstract Architecture

- Message transport

  - The key abstractions for all concrete architectures.
  - Variety of transports.
  - Support for alternative transports.
  - Transport agnosticism.
  - QoS.
FIPA Abstract Architecture

FIPA message

A FIPA-message is an individual unit of communication between two or more agents.

- FIPA-message is written in an agent-communication-language.
- FIPA-message has content.
- FIPA-message has an ontology.
- FIPA-message includes an agent-name corresponding to the sender of the message.
- FIPA-message includes one or more agent-name corresponding to the receiver or receivers of the message.
- FIPA-message is sent by an agent.
- FIPA-message is received by one or more agents.
- FIPA-message is transmitted as the payload of a transport-message.
FIPA Abstract Architecture

FIPA message to transport message

01.03.2004
FIPA Agent Framework
Questions?
Comments.
Objections!
Content

- Introduction
- FIPA Organization
- FIPA Specifications
- FIPA Abstract Architecture
- **FIPA Agent Management**
- FIPA ACL and Content Languages
- FIPA Message Transport Service
- Example:
  - FIPA Nomadic Application Support
  - CRUMPET
FIPA Agent Management

Abstract Architecture

Specifications

Implementations
Overview to agent platform

- Normative framework within which FIPA agents exist and operate.
- The creation, registration, location, communication, migration, and retirement of agents.
FIPA Agent Management

Agent Platform (AP)

- The physical infrastructure in which agents can be deployed,
- Consists of
  - the machine(s), operating system,
  - agent support software,
  - FIPA agent management components
    - DF, AMS, MTS (Internal Platform Message Transport, ACC, …) and agents.
FIPA Agent Management

Agent Management Reference Model

- Software
- Agent Platform
  - Agent
  - Agent Management System
  - Directory Facilitator
- Message Transport System
  - Agent Platform
  - Message Transport System
Agent Life-Cycle

- Waiting
- Suspended
- Wake Up
- Resume
- Suspend
- Move
- Execute
- Initiated
- Transit
- Active

Unknown

Destroy
Quit
Create
FIPA Agent Management

Agent Life-Cycle

- **Create**: The creation (installation) of a new agent.
- **Invoke**: The invocation of a new agent.
- **Destroy**: The forceful termination of an agent.
- **Quit**: The graceful termination of an agent. This can be ignored by the agent.
- **Suspend**: Puts an agent in a suspended state.
- **Resume**: Brings the agent from a suspended state.
- **Wait**: Puts the agent in a waiting state. This can only be initiated by the agent.
- **Wake Up**: Brings the agent from a waiting state.
**Agent naming**

The FIPA agent naming reference model identifies an agent through an extensible collection of parameter-value pairs, called an Agent Identifier (AID). An AID comprises:

- A name.
- Other parameters, such as transport addresses, name resolution service addresses, and so on.

The `:name` parameter of an AID is a globally unique identifier that can be used as a unique referring expression of the agent.

One of the simplest mechanisms is to construct it from the actual name of the agent and its home agent platform address (HAP), separated by the `@` character.

The extensible nature of an AID allows it to be augmented to accommodate other requirements, such as social names, nick names, roles, etc. which can then be attached to services within the AP.
FIPA Agent Management

Agent naming

- The parameter values of an AID can be edited or modified by an agent, for example, to update the sequence of name resolution servers or transport addresses in an AID. However, the mandatory parameters can only be changed by the agent to whom the AID belongs.

- AIDs are primarily intended to be used to identify agents inside the envelope of a message, specifically within the :to and :from parameters.

- The name of an agent is immutable and cannot be changed during the lifetime of the agent; the other parameters in the AID of an agent can be changed.

- The HAP of an agent is the AP on which the agent was created.
FIPA Agent Management

Agent Management System (AMS)

- Mandatory component.
- Only one AMS in a single AP.
- Responsible for managing the operation of AP
  - creation of agents, deletion of agents, migration of agents.
- Index of all the agents
  - agent’s GUID.
- “White pages” services
FIPA Agent Management

Agent Management System (AMS)

- Default name

(agent-identifier
  :name ams@hap
  :addresses (sequence hap_transport_address))

- Actions
Register-agent

(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))
          ...))))
FIPA Agent Management

- Directory Facilitator (DF)
  - Optional element.
    - If it is present, it must be implemented as specified.
  - “Yellow pages” services.
  - Agents can register their services with DF.
  - Agents can query DF to find out services.
  - Maintains accurate, complete, and timely list of agents.
FIPA Agent Management

- Directory Facilitator (DF)

- All DFs have a default name

  (agent-identifier
   :name df@hap
   :addresses (sequence hap_transport_address))

- Management actions
  - register, search, deregister, modify.
- All agent management operations use fipa-request protocol.
FIPA Agent Management

Register

(request
 :sender
  (agent-identifier
   :name dummy@foo.com
   :addresses (sequence iiop://foo.com/acc))
 :receiver (set
  (agent-identifier
   :name df@foo.com
   :addresses (sequence iiop://foo.com/acc)))
 :language FIPA-SL0
 :protocol FIPA-Request
 :ontology FIPA-Agent-Management
 :content
  (action
   (agent-identifier
    :name df@foo.com
    :addresses (sequence iiop://foo.com/acc))
   (register
    (df-agent-description
     :name
      (agent-identifier
       :name dummy@foo.com
       :addresses (sequence iiop://foo.com/acc))
     :protocol (set FIPA-Request Application-Protocol)
     :ontology (set meeting-scheduler)
     :language (set FIPA-SL0 KIF)
     :services (set
      (service-description
       :name profiling
       :type user-profiling
       :ontology (set meeting-scheduler)
       :properties (set
        (property
         :name learning-algorithm
         :value BBN)
        (property
         :name max-nodes
         :value 10000000))))))))
)
Content

- Introduction
- FIPA Organization
- FIPA Specifications
- FIPA Abstract Architecture
- FIPA Agent Management
- **FIPA ACL and Content Languages**
- FIPA Message Transport Service
- Example:
  - FIPA Nomadic Application Support
  - CRUMPET
FIPA ACL

Introduction

Example:

- An agent A can tell agent B something only if A believes it and can establish that B does not believe it

Requires the agents to reason about each other’s beliefs and intentions and behave co-operatively and sincerely.
Components of a message

ACL message

(message
  :sender agent1
  :receiver hpl-auction-server
  :content
    (price (bid good02) 150)
  :in-reply-to round-4
  :reply-with bid04
  :language sl
  :ontology hpl-auction
)

Begin message structure
Communicative act type
Message parameter
Parameter expression
Message content expression
FIPA ACL

Communicative Acts

- Accept-proposal
- Agree
- Cancel
- Cfp
- Confirm
- Disconfirm
- Failure
- Inform
- Inform-if (macro act)
- Inform-ref (macro act)
- Not-understood

- Propagate
- Propose
- Proxy
- Query-if
- Query-ref
- Refuse
- Reject-proposal
- Request
- Request-when
- Request-whenever
- Subscribe
FIPA ACL

Parameters (message slots)

- :sender
- :receiver
- :content
- :reply-with
- :reply-by
- :in-reply-to
- :reply-to
- :encoding
- :language
- :ontology
- :protocol
- :conversation-id
FIPA ACL

- Request
  - The sender requests the receiver to perform some action
  - Example:

```prolog
(request
  :sender i
  :receiver j
  :content "open "db.txt" for input"
  :language vb)
```
FIPA ACL

- Semantics of request

\(<i, \text{request}(j, a)\)>

FP: \(FP(a) [i\backslash j] \land B_i \text{Agent}(j, a) \land \neg B_i \text{PG}_j \text{Done}(a)\)

RE: \(\text{Done}(a)\)

\(i = \text{sender}, j = \text{receiver}, a = \text{action to perform}\)

\(\text{Done}(a) = \text{action performed}\)

\(B_i \text{Agent}(j, a) = \text{agent } i \text{ believes that } j \text{ can perform } a\)

\(\neg B_i \text{PG}_j \text{Done}(a) = i \text{ does not believe that } \text{Done}(a) \text{ is } j \text{'s persistent goal}\)
**FIPA ACL**

- **not-understood**
  
  - The sender of the act (e.g. i) informs the receiver (e.g. j) that it perceived that j performed some action, but that i did not understand what j just did.
  
  - Example:

    ```
    (not-understood
      :sender i
      :receiver j
      :content ((query-if :sender j :receiver i ...)
                 (unknown (ontology www)))
      :language sl)
    ```
FIPA ACL

- Inform

  - The sender informs the receiver that a given proposition is true.
  - The sending agent:
    - holds that some proposition is true;
    - intends that the receiving agent also comes to believe that the proposition is true;
    - does not already believe that the receiver has any knowledge of the truth of the proposition.
FIPA ACL

- Inform

- Example:

```
(inform
  :sender i
  :receiver j
  :content "weather( today, raining )"
  :language Prolog)
```
FIPA ACL

- Semantics of inform

\[<i, \text{inform}( j, \phi)>\]

FP: \[B_i\phi \land \neg B_i( B_{i\phi} \lor U_{i\phi})\]

RE: \[B_{i\phi}\]

\(B_{i\phi}\) = i believes that \(\phi\) is true

\[\neg B_i( B_{i\phi} \lor U_{i\phi})\]

\(B_{i\phi} \equiv B_i\phi \lor B_i\neg\phi\)

\(U_{i\phi} \equiv U_i\phi \lor U_i\neg\phi\) (uncertain)

RE: \[B_{j\phi}\] (j believes that \(\phi\) is true)
FIPA Content Languages

- Content languages
  - FIPA-SL
    - subsets SL0, SL1, and SL2
  - FIPA-RDF0
    - RDF based content language
  - FIPA-CCL
    - Constraint Choice Language
  - KIF & FIPA-KIF
FIPA Content Languages

- Semantic Language (SL)

- Formal language used to define the semantics of the FIPA ACL
- Logical propositions are expressed in a logic of mental attitudes and actions, formalised in a first order modal language with identity
FIPA Content Languages

- Content language
- FIPA-SL example

(query-ref
  :sender (agent-identifier :name B)
  :receiver (set (agent-identifier :name A))
  :content
    (iota ?x (UKPrimeMinister ?x))
  :language FIPA-SL
  :reply-with query2)

;inform
  :sender (agent-identifier :name A)
  :receiver (set (agent-identifier :name B))
  :content
    (= (iota ?x (UKPrimeMinister ?x))
        "Tony Blair")
  :language FIPA-SL
  :in-reply-to query2)
FIPA Content Languages

- Basis of the SL formalism
  - $p$, $p_1$, ... are taken to be closed formulas denoting propositions
  - $\phi$ and $\psi$ are formula schemas, which stand for any closed proposition
  - $i$ and $j$ are schematic variables which denote agents
  - $\models \phi$ means that $\phi$ is valid
FIPA Content Languages

- Basis of the SL formalism

- Modal operators $B$, $U$, and $C$
  - $B_i p$ “$i$ (implicitly) believes (that) $p$”
  - $U_i p$ “$i$ is uncertain about $p$ but thinks that $p$ is more likely than $\neg p$”
  - $C_i p$ “$i$ desires that $p$ currently holds”
Basis of the SL formalism

Operators *Feasible*, *Done* and *Agent* enable reasoning about actions:

- **Feasible**(*a*, *p*) = *a* can take place and if it does *p* will be true just after that

- **Done**(*a*, *p*) = *a* has just taken place and *p* was true just before that

- **Agent**(*i*, *a*) = *i* denotes the only agent performing, or that will be performing, the actions which appear in action expression *a*.

- **Single**(*a*) = *a* denotes an action expression that is not a sequence. Any individual action is Single. The composite act *a* ; *b* is not Single. The composite act *a* | *b* is Single iff both *a* and *b* are Single.
Basis of the SL formalism

A communicative act model is presented as follows:

\[ <i, \text{Act}(j, C)> \]

FP: \( \phi_1 \)
RE: \( \phi_2 \)

FP = \textit{feasibility preconditions}
RE = \textit{rational effect}
FIPA Content Languages

- FIPA-SL

- SL is used in most of the FIPA specs
  - mandatory for example for agent management

- SL0 (Minimal Subset of SL)

- SL1 (Proposition Form)

- SL2 (Restrictions for decidability)
FIPA Content Languages

SL Example

(action df@iiop://fipa.org:50/acc
  (register
    (:df-description
      (:agent-name peter@iiop://agentland.com:50/acc
        (:services
          (:service-description
            (:service-type video-on-demand)
            (:service-ontology itut-vod)
            (:service-name vod-1)
            (:fixed-properties (genre sport))
            (:interaction-protocols (fipa-request))
            (:ontology fipa-agent-management)
            (:address iiop://fipa.org/acc)
            (:ownership peter)
            (:df-state active)))))

01.03.2004 FIPA Agent Framework
FIPA Content Languages

- FIPA-RDF0 Example

```xml
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3c.org/TR/WD-rdf-syntax#"
         xmlns:s="http://description.org/schema/">
  <rdf:Description ID="TCP/IP Illustrated">
    <s:author>W. Richard Stevens</s:author>
  </rdf:Description>
</rdf:RDF>
```
FIPA Content Languages

- **FIPA-CCL**
  - FIPA-Constraint Choice Language
    - constraint satisfaction problems (CSPs)
    - multiple interrelated choices
  - FIPA-CCL supports
    - Problem representation
    - Information gathering
    - Information fusion
    - Access to problem solution techniques
FIPA Content Languages

CCL Example

(Object: CSP

(List Variables

{(Object: CSP-variable
  (Variable-name hotel-choice)
  (Variable-type hotel)
  (List Domain {“Hyatt” “Hilton” “HolidayInn”})
)
(Object: CSP-variable
  (Variable-name city-choice)
  (Variable-type city)
  (List Domain {“Paris” “New York” “Sydney”})
)
(Object: CSP-variable
  (Variable-name room-type-choice)
  (Variable-type hotel-room-type)
  (List Domain {“Suite” “Double” “Single”})
)
)

(List Constraints

(Object: CSP-constraint

(List Variables {room-type-choice hotel-choice})
(Relation (Extensional-nogoodList

{{(city-choice “New York”)
  (room-type-choice “Suite”)})
}})))

01.03.2004

FIPA Agent Framework
FIPA Interaction Protocols

- Interaction protocols
  - Specify agent communication patterns.
  - Example:

```
FIPA-query-Protocol

Initiator

query

refuse

not-understood

failure

inform

Participant
```

01.03.2004

FIPA Agent Framework
FIPA Interaction Protocols

- Protocol library
  - Request
  - Query
  - Request When
  - Contract Net
  - Iterated Contract Net
  - English Auction
  - Dutch Auction
  - Brokering
  - Subscribe
  - Brokering
  - Propose
FIPA Interaction Protocols

Requirement

- An ACL compliant agent need not implement any of the standard protocols, nor is it restricted from using other protocol names.
- However, if one of the standard protocol names is used, the agent must behave consistently with the FIPA’s protocol specification.
FIPA Interaction Protocols

Fipa-contract-net

Figure 1: FIPA Contract Net Interaction Protocol
FIPA Interaction Protocols

Fipa-contract-net

Agent

1) cfp

3) propose

5) reject-proposal

6) inform

4) propose

5) accept-proposal

01.03.2004

FIPA Agent Framework
Content

- Introduction
- FIPA Organization
- FIPA Specifications
- FIPA Abstract Architecture
- FIPA Agent Management
- FIPA ACL and Content Languages
- FIPA Message Transport Service
- Example:
  - FIPA Nomadic Application Support
  - CRUMPET
FIPA Message Transport Service

Message Transport Service

- The Message Transport Service (MTS) delivers messages between agents within an AP and to agents resident on other APs.
- All FIPA agents have access to at least one MTS and only messages addressed to an agent can be sent to the MTS.
- The ACL represents the content of the messages carried by both the MTS and MTP.
FIPA Message Transport Service

**Sending a message**

- An agent has three options when sending a message to another agent resident on a remote AP:
  - Agent A sends the message to its local ACC using a proprietary or standard interface. The ACC then takes care of sending the message to the correct remote ACC using a suitable MTP. The remote ACC which will eventually deliver the message.
  - Agent A sends the message directly to the ACC on the remote AP on which Agent B resides. This remote ACC then delivers the message to B. To use this method, Agent A must support access to one of the remote ACC’s MTP interfaces.
  - Agent A sends the message directly to Agent B, by using a direct communication mechanism. The message transfer, addressing, buffering of messages and any error messages must be handled by the sending and receiving agents. This communication mode is not covered by FIPA.
In its abstract form, a message is made up of two parts:

- a message envelope expressing transport information and
- a message body comprising the ACL message of the agent communication.
The following are general statements about the form of a message envelope:

- A message envelope comprises a collection of parameters.
  - A parameter is a name/value pair.
- A message envelope contains at least the mandatory :to, :from, :date and :acl-representation parameters.
- A message envelope can contain optional parameters.
Example:

```xml
<?xml version="1.0"?>
<envelope>
  <params index="1">
    <to>
      <agent-identifier>
        <name>receiver@foo.com</name>
        <addresses>
          <url>http://foo.com/acc</url>
        </addresses>
      </agent-identifier>
    </to>
    <from>
      <agent-identifier>
        <name>sender@bar.com</name>
        <addresses>
          <url>http://bar.com/acc</url>
        </addresses>
      </agent-identifier>
    </from>
    <acl-representation>fipa.acl.rep.xml.std</acl-representation>
    <date>20000508T042651481</date>
    <encrypted>no encryption</encrypted>
    <received>
      <received-by value="http://foo.com/acc"/>
      <received-date value="20000508T042651481"/>
      <received-id value="123456789"/>
    </received>
  </params>
</envelope>
```
FIPA Message Transport Service

- Agent Communication Channel (ACC)
  - Routes messages between agents within AP to agents resident on other APs.
  - *(Each ACC must support the FIPA baseline protocol.)*
  - Agents can request ACC to route message to target agent and ACC.
FIPA Communication Stack

- Conversation Layer
  - FIPA-query, FIPA-request, ...

- Content Language
  - FIPA SL, KIF, RDF, ...

- Agent Communication Language
  - FIPA ACL, KQML, ...

- Message Envelope
  - FIPA Message Envelope, ...

- Message Transport
  - GIOP, WAP, RMI, HTTP, ...

- Transport and Signaling Protocol
  - TCP/IP, WAP, SMS, ...

Network Infrastructure
Questions?
Comments.
Objections!
Content

- Introduction
- FIPA Organization
- FIPA specifications
- FIPA Abstract Architecture
- FIPA Agent Management
- FIPA ACL and Content Languages
- FIPA Message Transport Service
- Example:
  - FIPA Nomadic Application Support
FIPA implementations

- 16 implementations
- 5 open source implementations
- JCP called JAS
- Several related European projects
FIPA Nomadic Application Support

Nomadic Application

Features

- Wireless data communications
  - low throughput
  - highly variable QoS
  - unreliable

- Mobile computers
  - high power laptop
  - multimedia phone
FIPA Nomadic Application Support

Nomadic Application

Features

- today’s Internet
  - fixed network environment: unlimited resources
    - poor protocols
    - poor data representations
    - multimedia
  - very limited support for mobility
- seamless roaming
  - LAN - WLAN - GPRS - LAN - oooooo
FIPA Nomadic Application Support

What is required?

- Optimal data representation
- Good message transport protocols
- Selection of Message Transport Protocol and Message Transport Connection to be used for agent communication.
- Selection of an ACL representation and a content language to be used for agent communication.
- Provision of support for application agents to carry out adaptation of application data, such as still images, video and audio, XML, etc.
- Communication between agents performing adaptation.
FIPA Nomadic Application Support

- Nomadic Application Support specifies
  - *fipa-wap-std message transport protocol*
  - fipa-bitefficient-std ACL representation
  - An agent middleware to monitor and control a FIPA Message Transport Protocol and the underlying Message Transport Connection,
  - An ontology for the QoS of the Message Transport Service in the context of nomadic application support, and
  - Agent negotiation protocols for nomadic application support.
FIPA Nomadic Application Support

Monitoring and controlling the QoS
- The functions required to carry out monitoring and controlling:
  1. observing the QoS of MTP and MTC,
  2. measuring (if there are no other means to obtain the required information) the QoS of MTP and MTC,
  3. collecting information from the observing and measuring sources,
  4. analysing the information, and
  5. controlling the MTC and selecting the MTP.
FIPA Nomadic Application Support

- Agent middleware
  - Monitor Agent, which carries out the tasks 1 - 4, and
  - Control Agent, which carries out the task 5.
FIPA Nomadic Application Support

Agent middleware

Agent Platform A
- Agent
- AMS
- DF
- CA
- MA
- ACC
- IIOP
- WAP
- TCP/IP
- Wireline
- WLAN
- WWAN

Agent Platform B
- Agent
- AMS
- DF
- CA
- MA
- ACC
- IIOP
- WAP
- TCP/IP
- Wireline
- WLAN
- WWAN

01.03.2004 FIPA Agent Framework
Control Agent

Actions
- open-comm-channel
- close-comm-channel
- activate
- deactivate
- use
  - protocol: fipa-propose

Ontology: fipa-communication-management
Protocol: fipa-request
FIPA Nomadic Application Support

- **Monitor Agent**
  - **Actions**
    - request information
      - QoS information
      - protocol: query-ref
      - ontology: fipa-mtp-qos
    - subscribe notifications
      - protocol: fipa-subscribe
      - ontology: fipa-mtp-qos
    - inform changes in communication channel connectivity
      - protocol: fipa-subscribe
      - ontology: fipa-communication-management
FIPA Nomadic Application Support

- Ontologies
  - fipa-mtp-qos
  - fipa-communication-management
  - fipa-message-representation

- Protocols
  - fipa-subscribe
  - fipa-propose
FIPA Nomadic Application Support

- Negotiation of message transport protocol
- Negotiation of data representation
- Wireless message transport protocol
Questions?