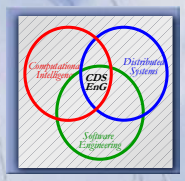
ES595a - Advanced Topics in Software & Systems Design
Cooperative Distributed Systems Engineering: Technologies &Applications



W05: *Agent-Orientation*

Western

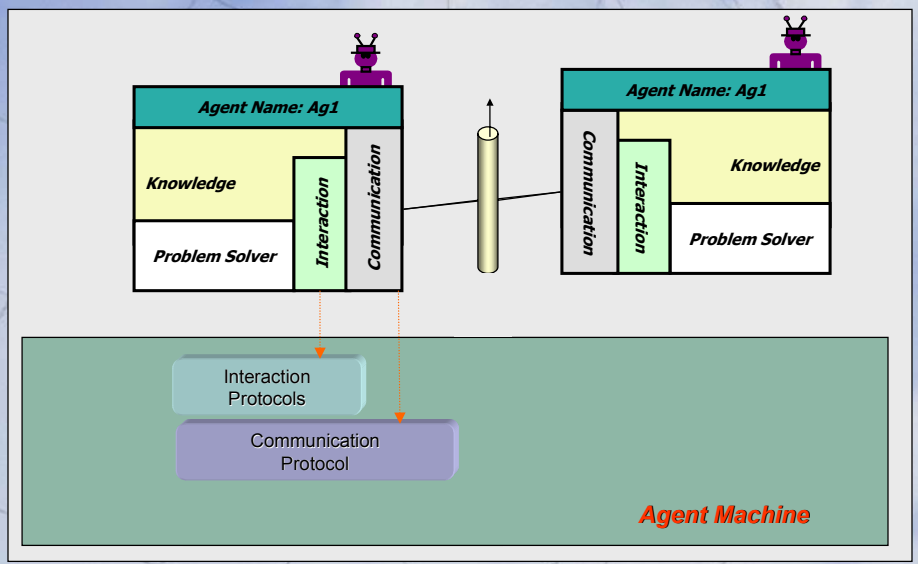
Communication vs. Interaction

- Interactions
  - occur when interdependencies between agents exist
    - resource contention, e.g., mobile robots bumping into each other
- Communication
  - occurs when agents exchange messages
    - with a view to influencing beliefs and intentions

Communication & the other 2 Cisters!

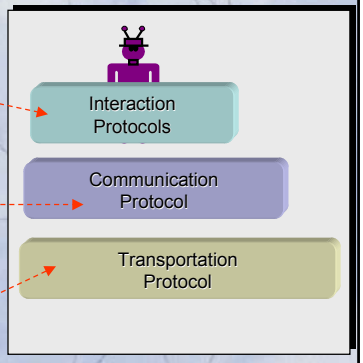
- Agents may coordinate without communication
  - Only if they have perfect models of the world
    - Otherwise, communication is essential for *coordination*
- To facilitate cooperation, agents often need to communicate their
  - intentions, goals, results, and state

Supporting Levels of Agent Machine



Supporting Levels of Agent Machine

- Interaction layer:
  - Refers to coordination-level strategy pursued by the agents and to the policy managing the structure of the inter-agent conversation
- Communication layer:
  - The medium through which attitudes regarding the content of the exchange are communicated
- Transportation layer:
  - The actual transport mechanism used for the networked communication



The Structure of Communication: Syntax, Semantics

- For message passing communication
  - Syntax
    - requires a common language
      - to represent information and queries,
    - languages that are inter-translatable
  - Semantics
    - requires a structured vocabulary and a shared framework of knowledge-shared ontology

ACL Requirements

- ACL will be valuable to the extent that it meets the following requirements
  1. Form
  2. Content
  3. Semantics
  4. Implementation
  5. Networking
  6. Environment
  7. Reliability

Features of ACLs *(Cont.)*

- Form
  - A CL should be
    - declarative, syntactically simple, and readable by people
    - concise, yet easy to parse and to generate
    - linear or should be easily translated into a linear form
    - ACL syntax should be extensible
- Content
  - ACL at least should
    - make a distinction between the
      - communication language, which expresses communicative acts
      - content language,which expresses aspects of the domain
    - commit to a well defined set of communicative acts (primitives)
      - a core of primitives that capture most of our intuitions about what constitutes a communicative act irrespective of application
      - the choice of the core set of primitives also related to the decision of whether to commit to a specific content language

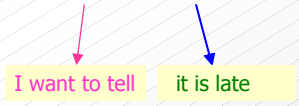
Features of ACLs *(Cont.)*

- Semantics
  - should be unambiguous, grounded in theory, and exhibit canonical form
    - i.e., similarity in meaning should lead to similarity in representation
  - should be supported by a formal description,
    - to support interaction among a diverse range of applications
  - In addition
    - provide a model of communication
      - which would be useful for performance modeling, etc.
    - address space and time carefully
      - to support interaction that extends over time amongst spatially dispersed applications

ACL: Theoretical Foundations

- Speech act theory

  A high level framework to account for human communication
  - Communication is viewed as actions
    - Communication Language= speech act + content



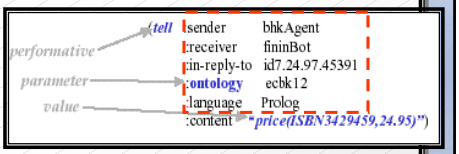
Speech Act Theory…

- considers three aspects of a message
  - Locution: physical utterance
    - e.g., "It is hot here"
  - Illocution: act of conveying speaker's intention
    using Speech acts
    - e.g., a request to turn on the cooler or an assertion about the temperature
  - Per-locution, actions that occur as a result of locutions
    Further acts resulting from the speech acts
    - e.g., turns on the cooler, opens the window, or ignores the speaker
- In agent communication, Illocution is the main aspect
  - Speech act theory classifies illocutions in various ways

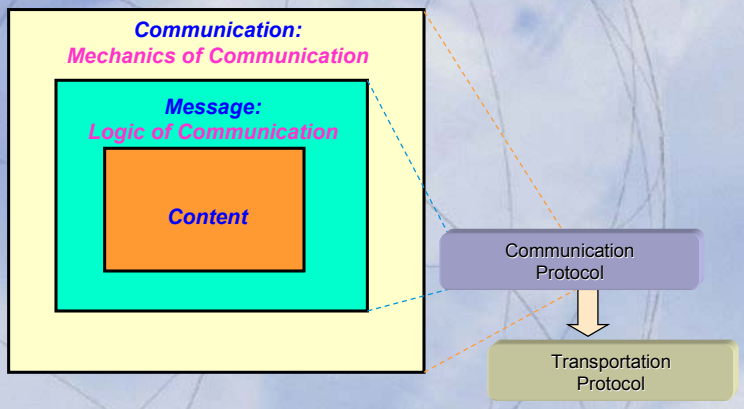
Speech Acts: Illocution

Illocution – communication of the speakers intention

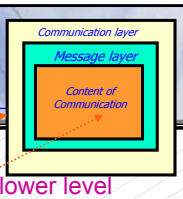
- An illocutionary act is composed from
  - Propositional content
    - Describes state of affairs (claim)
  - Context
    - Speaker, hearer, time, location, …
  - Illocutionary Force
    - 5 categories:
      - assertives, directives, commissives, expressives, declaratives



ACL is a Layered Language:
KQML



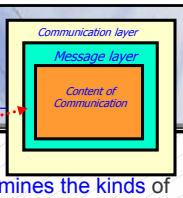
ACL Layers
KQML



- The communication layer
  - It encodes a set of message features which describe the lower level communication parameters, such as
    - the identity of the sender and recipient
    - a unique identifier associated with the communication
- The content layer
  - bears the actual content of the message in the desired representation language
    - KQML supports any representation language
      - including languages expressed as ASCII strings and those expressed using a binary notation
      - Some KQML-speaking agents (e.g., routers, very general brokers etc.) may ignore the content portion of the message except to determine where it ends

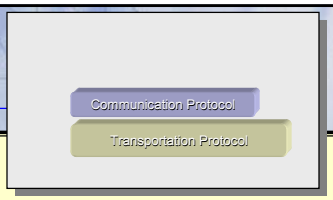
ACL Layers

KQML (Cont.)

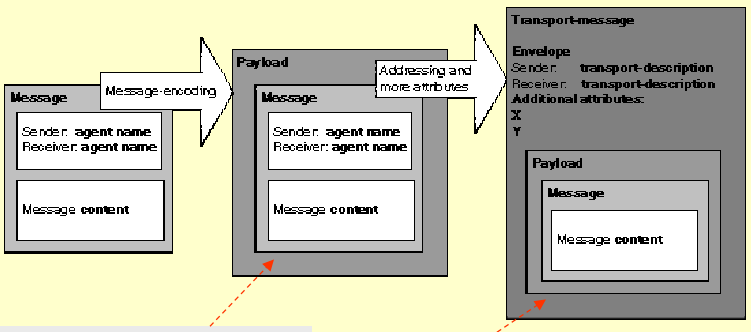


- The Message Layer: the core of KQML
  - to encode a message that need to be to transmitted, and determines the kinds of **protocol** one can have with a KQML-speaking agent
    - A primary function
      - to identify the protocol to be used to deliver the message and
      - to supply a speech act or performative attached to the content,
        - such as an assertion, a query, a command, etc.
  - In addition, since the content may be transparent to a KQML-speaking agent,
    - optional features to describe the content language, the ontology it assumes, and some type of description of the content
      (such as a, descriptor naming a topic within the ontology)
      - These features make it possible for KQML implementations to analyze route and properly deliver messages even though their content is inaccessible

Communication (Cont.)



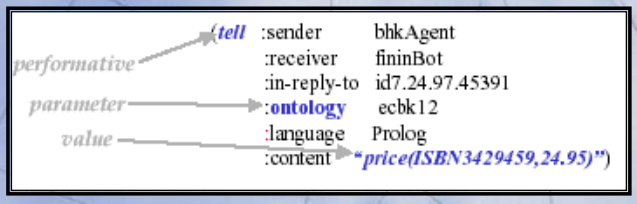
A Message Becomes a Transport-message



The **payload** is encoded using the **encoding-representation** appropriate for the transport

The **transport-message** is the **payload** plus the **envelope**. The **envelope** includes the sender and receiver **transport-descriptions**

A KQML Message



- Represents a single speech act (performative)
  - tell, ask, reply, subscribe, achieve, monitor, ...
  - ➢ an associated semantics and protocol
  - ➢ a list of attribute/value pairs (context)
    - :content, :language, :from, :in-reply-to

ACL: Performative

- Performatives are the core of the language; They
  - ➢ Used to describe the interactions with ACL-speaking agent
    - Identify protocol to be used and speech act attached to the content
      - Specify that content is an assertion, query, command, etc.

Example: KQML

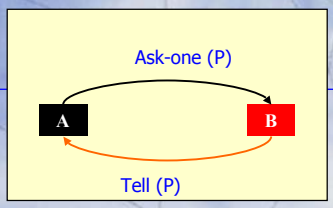
```
(ask-one
    :content <expression>
    :aspect <expression>
    :language <word>
    :ontology <word>
    :reply-with <expression>
    :sender <word>
    :receiver <word>)
```

```
(ask-one
        :content "price(MFC, [?price, ?time])"
        :receiver stock-server
        :language standard-prolog
        :ontology TSX-TICS)
```


ACL Semantics
KQML

- Each agent manages a virtual knowledge base (VKB)
  - Statements in a VKB can be classified into
    - Beliefs:
      - Encode information an agent has about itself and its environment
    - Goals:
      - Encode states of an agent's environment that it will act to achieve
- Agents use KQML to communicate
  - About the contents of their own and others' VKBs

Tell…



Ask-one

:sender <word>
:receiver <word>
:content <expression>
:language <word>
:ontology <word>

:sender whishes to know if :content matches any sentence in the :receiver VKB

Tell

:sender <word>
:receiver <word>
:content <expression>
:language <word>
:ontology <word>

:content sentence is in the :sender's VKB

Semantics for TELL

VKB: *bel(Ag1,X)*

Tell: Ag1 states to Ag2 that Ag1 believes X is true

(Tell Sender: Ag1; Receiver: Ag2; Content: S)

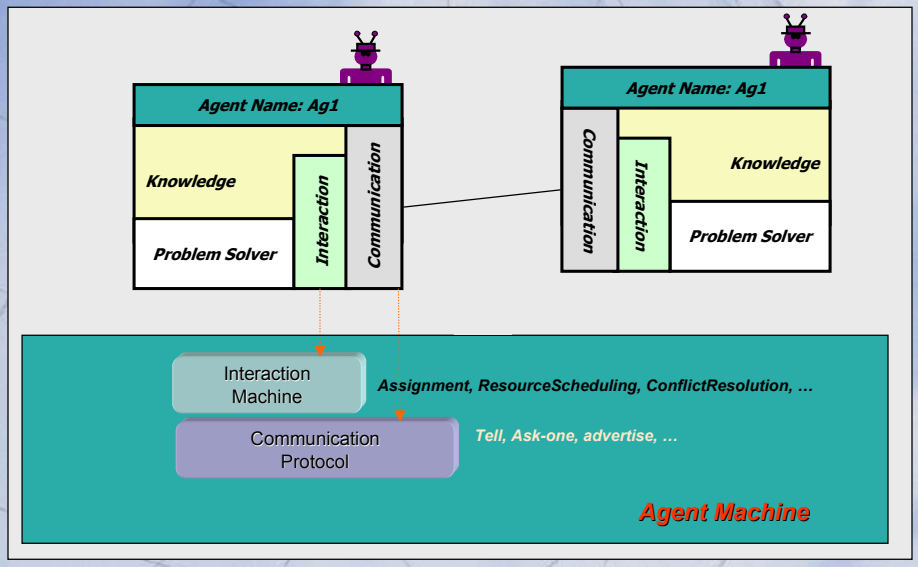
Tell-S <
Pre(Ag1): ***S*** *Λ know(Ag1,want(Ag2,know(Ag2,****S****)))*
Post(Ag1): know(Ag1,know(Ag2, ***S****)) >*

where ***S*** ≅bel(Ag1,X)

VKB: *bel(Ag1,X)*

Tell-R<
*Pre(Ag2): intend(Ag2,know(Ag2,****S****))*
*Post(Ag2): know(Ag2,****S****) >*

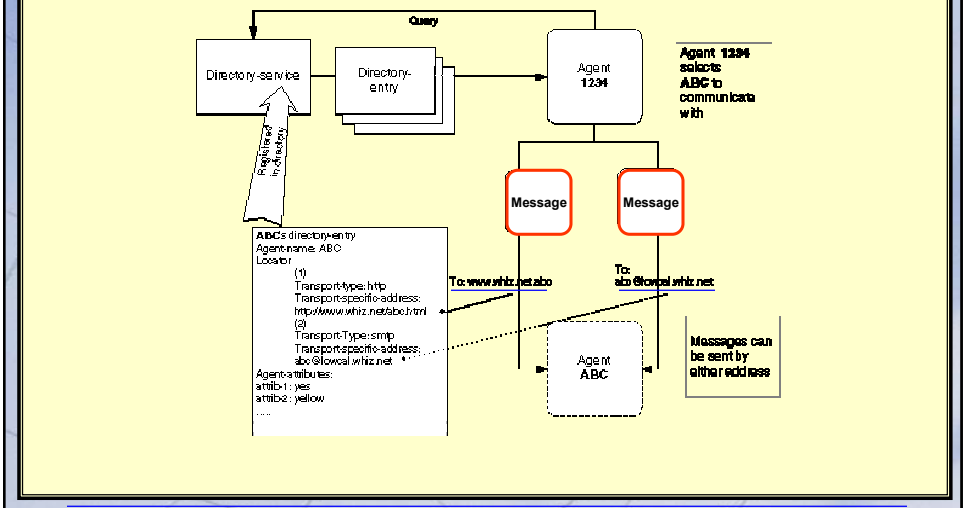
Supporting Levels of Agent Machine



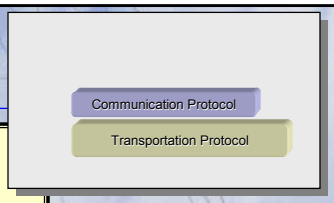
Agent Naming

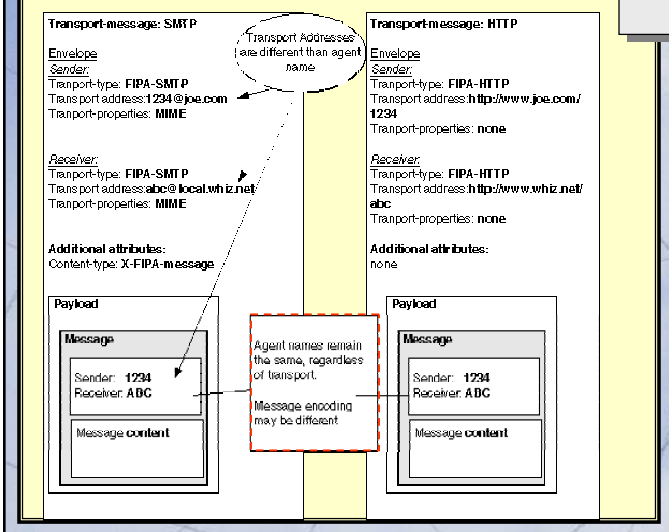
- The use of a central *Agent Name Server*,
  - an architecture used by most systems
- KQML assumes that names are *local*
  - Ag1 can register with Ag2 under the name Alice
  - Ag1 can register with Ag3 under the name Albert
- What gets registered under a name?
  - Contact information like:
    - name(Ghenniwa,smtp,[hghenniwa@uwo.ca)
    - name(UWO,http,[www.uwo.ca/)

Communication (Cont.)
Naming & Addressing…



Communication (Cont.)
Naming & Addressing…





Semantics Issues

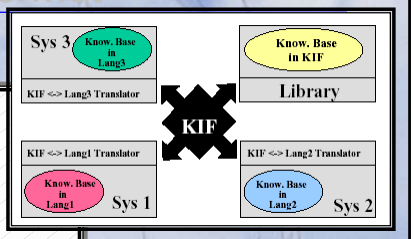
```
(ask-one
      :content <expression>
      :aspect <expression>
      :language <word>
      :ontology <word>
      :reply-with <expression>
      :sender <word>
      :receiver <word>)
```

- What if the agents have
  - different terms for the same concept?
  - same term for different concepts?
  - different class systems or schemas?
  - differences in depth and breadth of coverage?

Common Ontologies

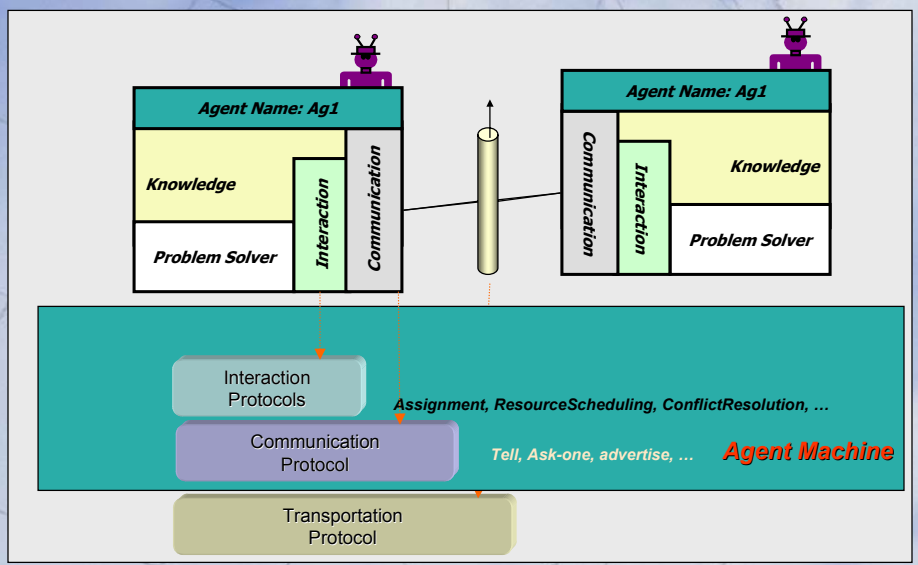
- A shared representation is essential
  - to successful communication and coordination
  - For computational agents
    - this is provided by a common ontology:
      - terms used in communication can be coherently defined
- interaction policies can be shared

Knowledge Interchange Format



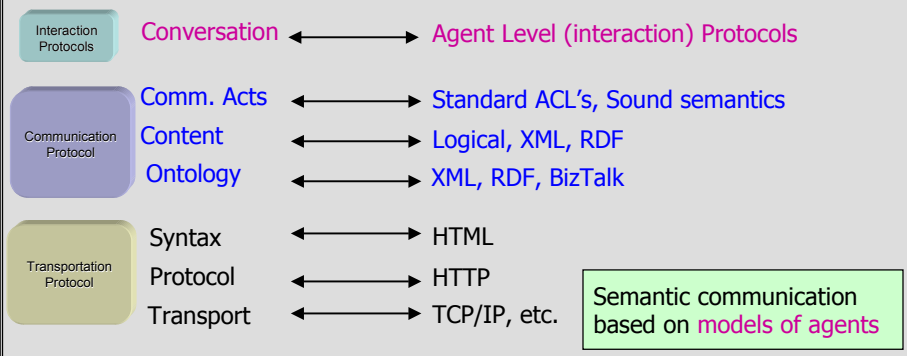
- KIF ~ First order logic, set theory
  - an interlingua
    - for encoded declarative knowledge
      - Takes translation among n
      - systems from $O(n^2)$ to **O(n)**
- Common language for reusable knowledge
  - Implementation independent semantics
  - Highly expressive –
    - can represent knowledge in typical application KBs
  - Translatable
    - into and out of typical application languages
  - Human readable
    - good for publishing reference models and ontologies

Supporting Levels of Agent Coordination



Agent-to-Agent Communication View

Systems ultimately need to communicate up at the semantic levels

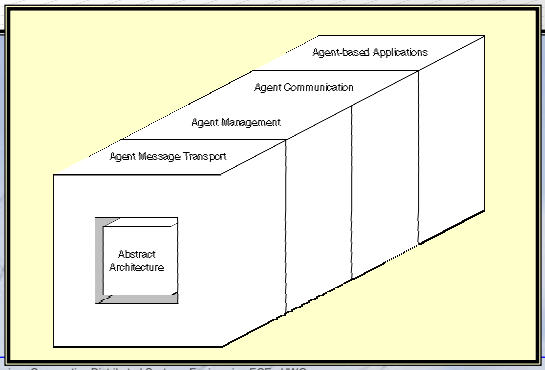


Environment

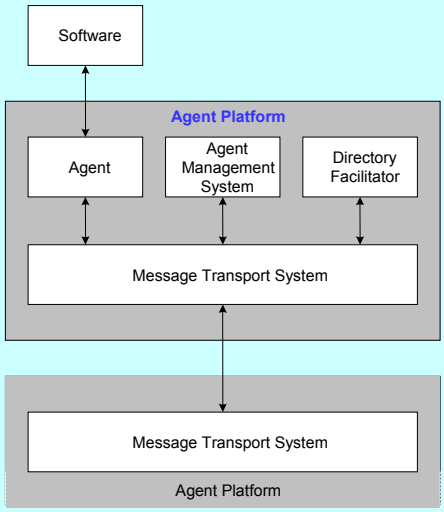
- Infrastructure
  - Agents require support to perform the tasks they are required to do
- Distributed computing
  - Linking different components to perform joint tasks
- Providing support services
  - To enable reuse of code
  - To abstract away from implementation level details
  - To provide uniform access to functions
- Middleware defines (part of…) the environment
  - Services available: communication, security, etc.
  - Agents live in a software environment – middleware plays a large part

FIPA

- Foundation for Intelligent Physical Agents
- Structure of standards specifications in the following main areas
  - Management services (AMS)
  - Directory services (DF)
  - Agent communication channel
    - Message transport



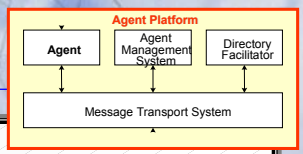
Agent Management Model



Agent platform (AP)

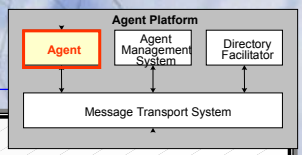
- Agents: Application
- Specialized Agents
  - Directory facilitator
  - Agent management system
- Message transport system
  - Agent communication channel (ACC)
    - The message transport service is the default communication method between agents on different APs
    - ACC supports the message transport system
    - The ACC also supports AP routing tasks

Agent Platform



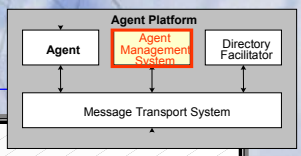
- AP is a logical (virtual) platform
  - Provides the physical infrastructure in which agents can be deployed
  - Consists of
    - The machine(s), operating system,
    - FIPA agent management components (DF, AMS and MTS)
    - Agents
  - The internal design of an AP is not a subject of standardization within FIPA
- FIPA
  - Envisages a variety of APs
    - From single processes containing lightweight agent threads
    - To fully distributed APs built around proprietary or open middleware standards
  - Concerned only with how communication is carried out between
    - Agents who are native to the AP
    - Agents outside the AP or agents who dynamically register with an AP

Agent



- The agent artifact
  - The fundamental element on an AP
    - Combines one or more functional capabilities into an integrated execution mode
  - May support several notions of identity
    - **Agent identifier** (AID)
      - Labels an agent so that it may be distinguished unambiguously within the agent universe
  - May be registered at a number of transport addresses at which it can be contacted

Agent Management System

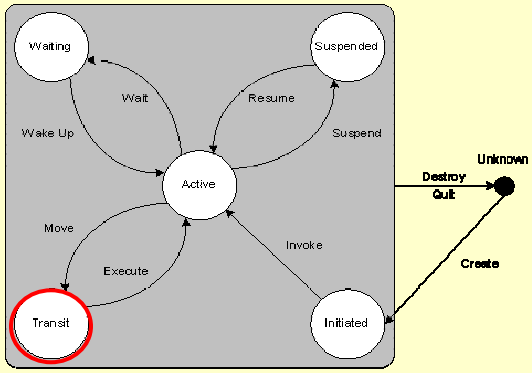


- A reference model that handles the lifecycle of an agent
  - creation, registration, location, communication, migration and retirement of agents
- The AMS
  - Exerts supervisory control over access to and use of the AP
  - A mandatory component of the AP
    - Only one AMS will exist in a single AP

Agent Lifecycle

- FIPA agents
  - As a software process, has a **physical lifecycle** that has to be **managed by the AP**
- Lifecycle assumptions
  - **AP bounded**
    - The lifecycle of a static agent is always bounded to a specific AP
  - **Application independent**
    - The lifecycle model is independent from any application system
      - It defines only the states and the transitions of the agent service in its lifecycle
  - **Instance-oriented**
    - The lifecycle describes is an instance of an agent with
      - Unique name and is executed independently
  - **Unique**
    - Each agent can only be in one state at any time and within only one AP

Lifecycle: States + Transitions



- The lifecycle states can be described in terms of
  - AMS responsibilities to message delivery in each state of the agent's lifecycle

Lifecycle: States

- Active
  - The MTS delivers messages to the agent as normal
- Initiated/waiting/suspended
  - The MTS
    - Either buffers messages until the agent returns to the active state
    - Or forwards messages to a new location (if a forward is set for the agent)
- Transit *(for mobile agent only)*
  - The MTS
    - Either forwards messages to a new location
    - Or, buffers messages until the agent becomes active
      - i.e., The move function failed on the original AP or the agent was successfully started on the destination AP
- Unknown
  - The MTS either buffers messages or rejects them
    - Depending upon the policy of the MTS and the transport requirements of the message

Lifecycle: State Transitions

- The state transitions of agents can be described as
  - Create
    - The creation or installation of a new **agent**
  - Invoke
    - The invocation of a new **agent**
  - Destroy
    - The forceful termination of an agent.
      - This can only be initiated by the **AMS**
      - Cannot be ignored by the **agent**
  - Quit
    - A graceful termination of an **agent**
      - This can be ignored by the **agent**

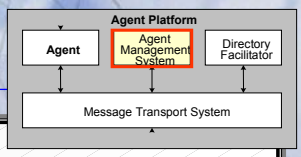
Lifecycle: State Transitions *(cont.)*

- Suspend
  - Puts an agent in a suspended state
    - This can be initiated by the **agent** or the **AMS**
- Resume
  - Brings the agent up from a suspended state
    - This can only be initiated by the **AMS**
- Wait
  - Puts an agent in a waiting state
    - This can only be initiated by an **agent**
- Wake up
  - Brings the agent up from a waiting state.
    - This can only be initiated by the **AMS**

Lifecycle: State Transitions (cont.)

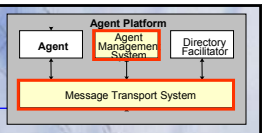
- The following two transitions are only used by mobile agents (see [FIPA00005])
  - Move
    - Puts the agent in a transitory state
      - This can only be initiated by the **agent**
  - Execute
    - Brings the agent up from a transitory state
      - This can only be initiated by the **AMS**

AMS…

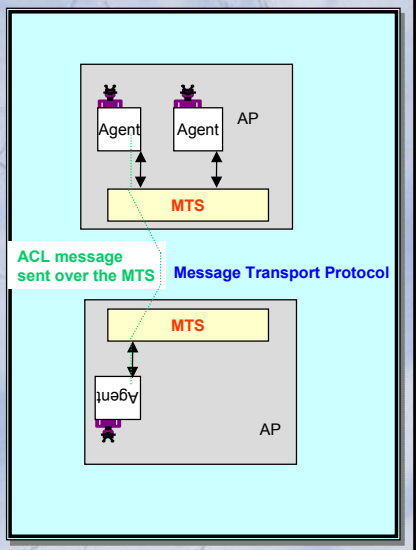


- It provides the following services:
  - maintains a directory of AIDs
    - represented by a flexible and extensible structure
      - which can support social names, transport addresses, name resolution services, amongst other things.
    - White pages
      - such as agent location, naming and control access services,
    - Directory Facilitator (DF) provides Yellow pages services, such as service location and registration services
  - Agent message transport services

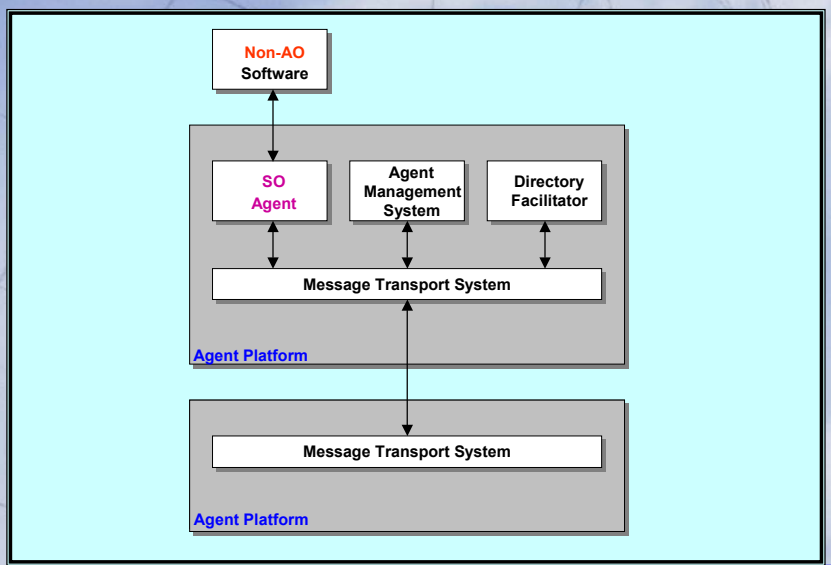
Message Transport Service



- MTS delivers messages
  - Between agents within an AP
  - To agents resident on other APs
  - All FIPA agents have access to at least one MTS
  - Only messages addressed to an agent can be sent to the MTS
- Agent message transport (AMT) defines a message as an envelope plus a body. They handle
  - FIPA ACL representations
    (e.g., String encoding, XML encoding, bit-efficient encoding)
  - Message envelope representation (e.g., XML for HTTP, bit-efficient for WAP)
  - Guidelines for various transport protocols (e.g., IIOP, http, WAP)



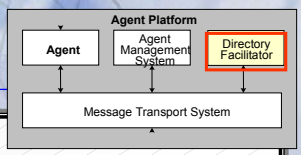
FIPA:
Agent Management Model



Agent Software Interoperation

- Non-AO Software
  - All non-agent (legacy) executable collections of instructions accessible through an agent
  - Agents may access software,
    - e.g., To add new services, acquire new communications protocols, acquire new security protocols/algorithms, acquire new negotiation protocols, access tools which support migration, etc.
- Use of wrappers to connect software with agents
- Agent resource broker (ARB) service
  - Management
  - Authentication
  - Permission

Directory Facilitator



- The DF provides yellow pages services to other agents
  - A DF is a mandatory component of the AP
- Agents may
  - Register their services with a DF
  - Query the DF to find out what services are offered by other agents
- Multiple DFs may exist within an AP and may be federated