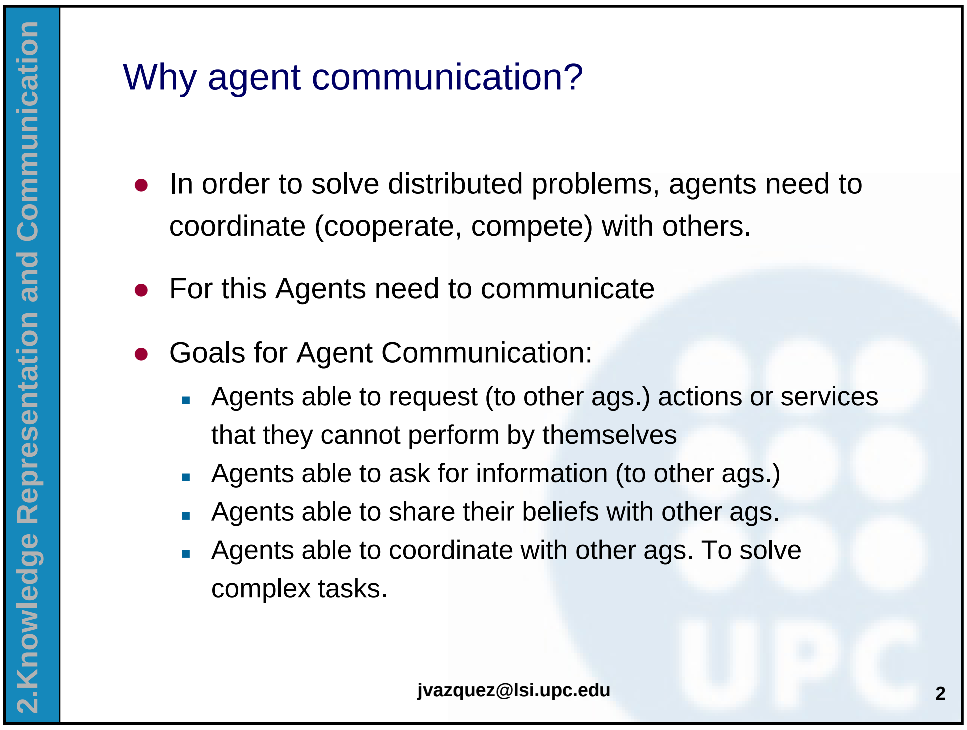
2. Knowledge Representation and Communication
Part 2:
Agent Communication

Javier Vázquez-Salceda
SMA-UPC

Knowledge Engineering and Machine Learning Group
UNIVERSITAT POLITÈCNICA DE CATALUNYA
https://kemlg.upc.edu



Why agent communication?

- In order to solve distributed problems, agents need to coordinate (cooperate, compete) with others.
- For this Agents need to communicate
- Goals for Agent Communication:
  - Agents able to request (to other ags.) actions or services that they cannot perform by themselves
  - Agents able to ask for information (to other ags.)
  - Agents able to share their beliefs with other ags.
  - Agents able to coordinate with other ags. To solve complex tasks.

2.Knowledge Representation and Communication

Levels in Agent Communication

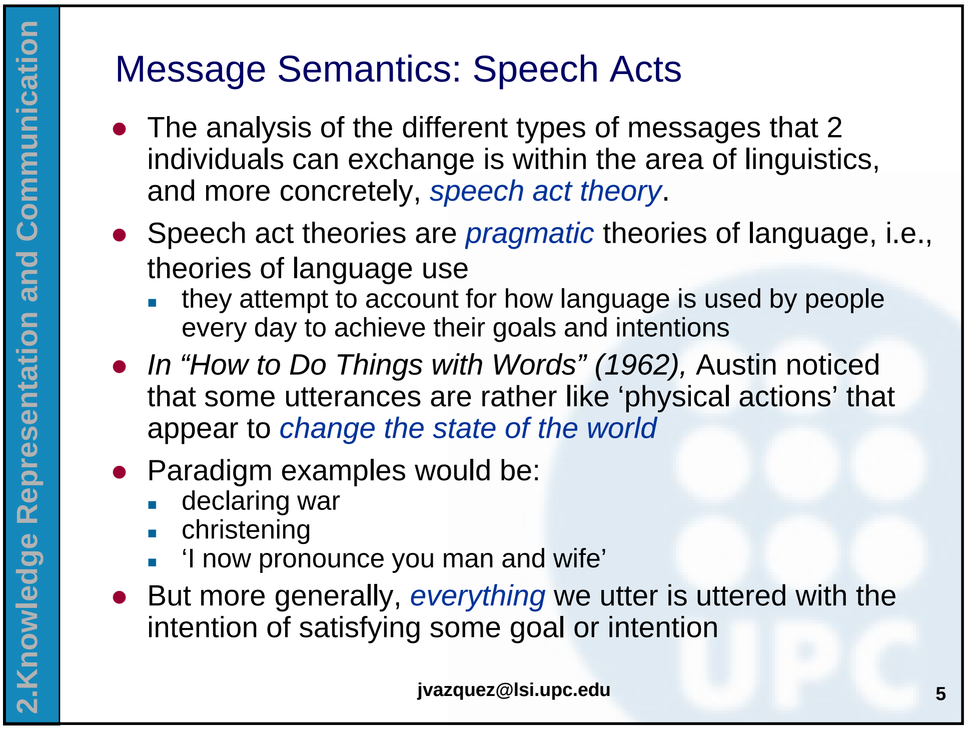
- Four levels in communication:
  - **Message Semantics**
    - What does each message means?
    - 3 components
      - **Message type**: gives intensionality
      - **Message content**: contains the information
      - **Ontology** (the message refers to)
  - **Message Sintaxis**
    - How each message is expressed?
    - 2 components
      - Message structure: **Agent Communication Language**
      - Content codification: **Content Language**
  - **Interaction protocol**
    - How are conversations/dialogues structured?
      - **Agent Protocols**
  - **Transport protocol**
    - How messages are actually sent and received by agents?

Multiagent Systems (SMA-UPC)

Message Semantics

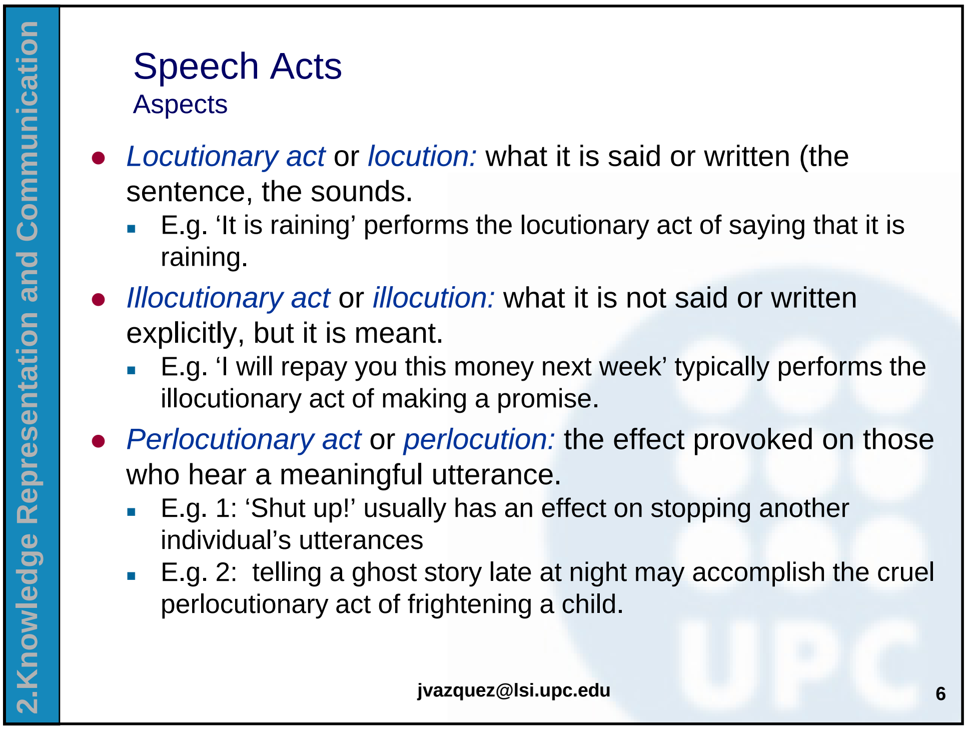
- **Speech Act Theory**

Knowledge Engineering and Machine Learning Group
UNIVERSITAT POLITÈCNICA DE CATALUNYA
https://kemlg.upc.edu



Message Semantics: Speech Acts

- The analysis of the different types of messages that 2 individuals can exchange is within the area of linguistics, and more concretely, *speech act theory*.
- Speech act theories are *pragmatic* theories of language, i.e., theories of language use
  - they attempt to account for how language is used by people every day to achieve their goals and intentions
- *In “How to Do Things with Words” (1962),* Austin noticed that some utterances are rather like ‘physical actions’ that appear to *change the state of the world*
- Paradigm examples would be:
  - declaring war
  - christening
  - ‘I now pronounce you man and wife’
- But more generally, *everything* we utter is uttered with the intention of satisfying some goal or intention



Speech Acts

Aspects

- *Locutionary act* or *locution:* what it is said or written (the sentence, the sounds.
  - E.g. ‘It is raining’ performs the locutionary act of saying that it is raining.
- *Illocutionary act* or *illocution:* what it is not said or written explicitly, but it is meant.
  - E.g. ‘I will repay you this money next week’ typically performs the illocutionary act of making a promise.
- *Perlocutionary act* or *perlocution:* the effect provoked on those who hear a meaningful utterance.
  - E.g. 1: ‘Shut up!’ usually has an effect on stopping another individual’s utterances
  - E.g. 2: telling a ghost story late at night may accomplish the cruel perlocutionary act of frightening a child.

Speech Acts

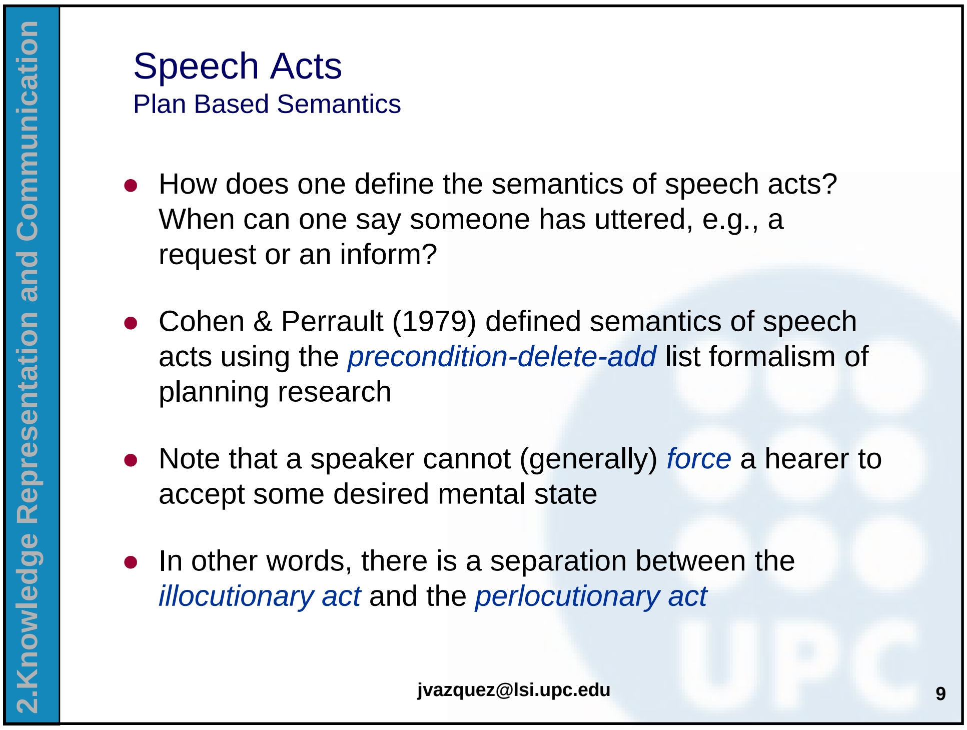
Types

- Searle (1969) identified various different types of speech act:
  - *representatives:*
    such as *informin*g, e.g., ‘It is raining’
  - *directives:*
    attempts to get the hearer to do something
    e.g., ‘please make the tea’
  - *commisives:*
    which commit the speaker to doing something,
    e.g., ‘I promise to… ’
  - *expressives:*
    whereby a speaker expresses a mental state,
    e.g., ‘thank you!’
  - *declarations:*
    such as declaring war or christening

Speech Acts

Components

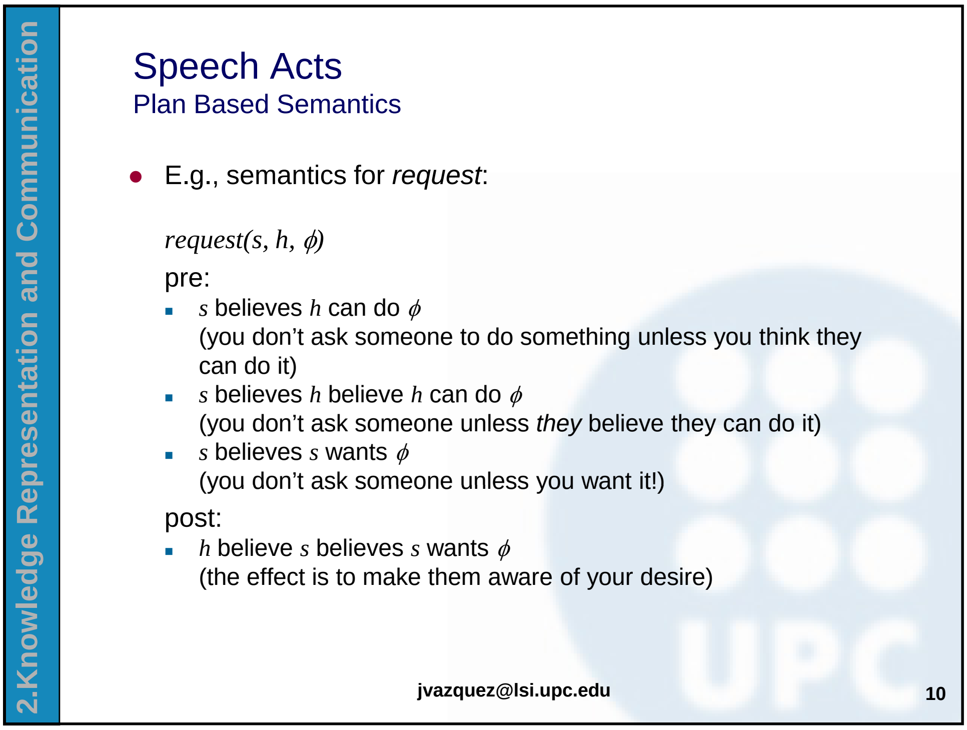
- In general, a speech act can be seen to have two components:
  - a *performative verb:*
    (e.g., request, inform, promise, …)
  - *propositional content:*
    (e.g., “the door is closed”)
- E.g.:
  - performative = request
    content = *“the door is closed”*
    speech act = *“please close the door”*
  - performative = inform
    content = *“the door is closed”*
    speech act = *“the door is closed!”*
  - performative = inquire
    content = *“the door is closed”*
    speech act = *“is the door closed?”*



Speech Acts

Plan Based Semantics

- How does one define the semantics of speech acts? When can one say someone has uttered, e.g., a request or an inform?
- Cohen & Perrault (1979) defined semantics of speech acts using the *precondition-delete-add* list formalism of planning research
- Note that a speaker cannot (generally) *force* a hearer to accept some desired mental state
- In other words, there is a separation between the *illocutionary act* and the *perlocutionary act*



Speech Acts

Plan Based Semantics

- E.g., semantics for *request*:

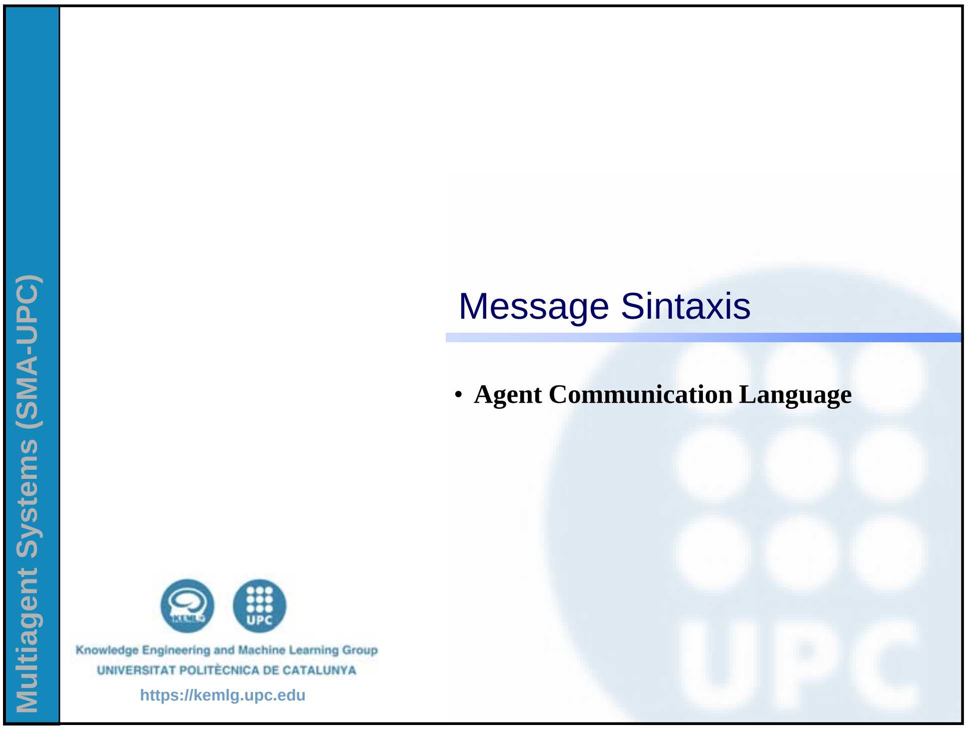
request(s, h, ϕ)

pre:

- *s* believes *h* can do ϕ
  (you don't ask someone to do something unless you think they can do it)
- *s* believes *h* believe *h* can do ϕ
  (you don't ask someone unless *they* believe they can do it)
- *s* believes *s* wants ϕ
  (you don't ask someone unless you want it!)

post:

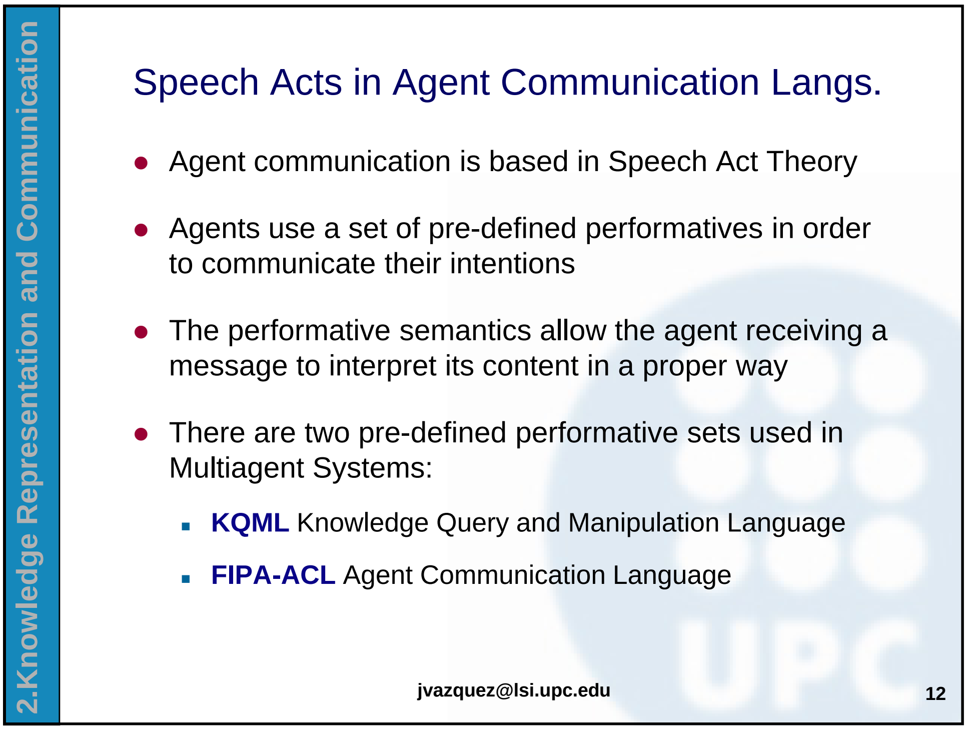
- *h* believe *s* believes *s* wants ϕ
  (the effect is to make them aware of your desire)



Message Sintaxis

- **Agent Communication Language**

Knowledge Engineering and Machine Learning Group
UNIVERSITAT POLITÈCNICA DE CATALUNYA
https://kemlg.upc.edu



Speech Acts in Agent Communication Langs.

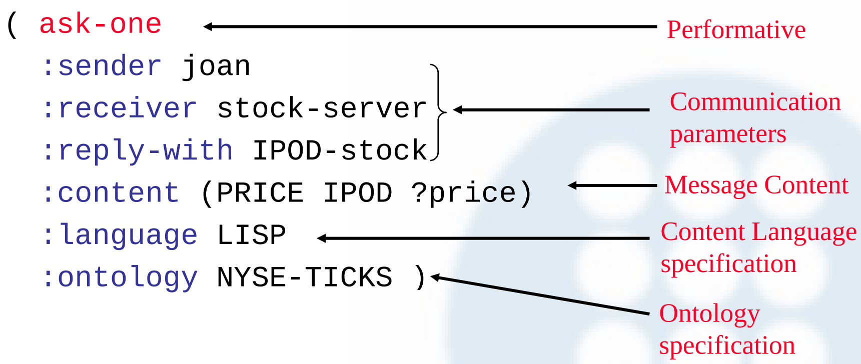
- Agent communication is based in Speech Act Theory
- Agents use a set of pre-defined performatives in order to communicate their intentions
- The performative semantics allow the agent receiving a message to interpret its content in a proper way
- There are two pre-defined performative sets used in Multiagent Systems:
  - **KQML** Knowledge Query and Manipulation Language
  - **FIPA-ACL** Agent Communication Language

KQML

- The first widely-spread ACL was KQML, developed by the ARPA knowledge sharing initiative
- KQML is comprised of two parts:
  - the knowledge query and manipulation language (KQML)
  - the content language (usually KIF)
- KQML is an ‘outer’ language, that defines a quite large set of acceptable ‘communicative verbs’, or *performatives* for :
  - Basic requests (`evaluate`, `ask-one`, `perform` …)
  - Multiagent requests (`stream-in`, …)
  - Responses (`reply`, `sorry`, …)
  - Information (`tell`, `achieve`, `cancel`, …)
  - Coordination (`stand-by`, `ready`, `next`, …)
  - Definition of capabilities (`advertise`, `subscribe`, …)
  - Networking (`register`, `forward`, `broadcast`, …)

KQML

Example



```
( ask-one
  :sender joan
  :receiver stock-server
  :reply-with IPOD-stock
  :content (PRICE IPOD ?price)
  :language LISP
  :ontology NYSE-TICKS )
```

KQML and KIF

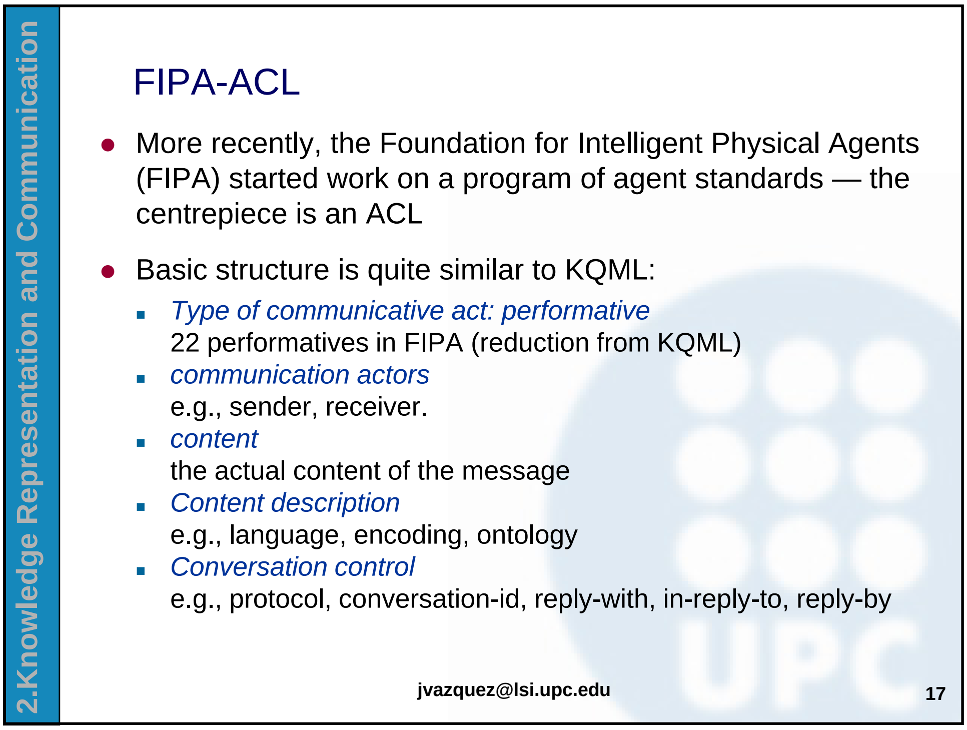
- KIF is a language for expressing message *content*
- E.g.,
  - “The temperature of m1 is 83 Celsius”:
    ```
    (= (temperature m1) (scalar 83 Celsius))
    ```
  - “An object is a bachelor if the object is a man and is not married”:
    ```
    (defrelation bachelor (?x) :=
    (and (man ?x) (not (married ?x))))
    ```
  - “Any individual with the property of being a person also has the property of being a mammal”:
    ```
    (defrelation person (?x) :=> (mammal ?x))
    ```

KQML and KIF

Example

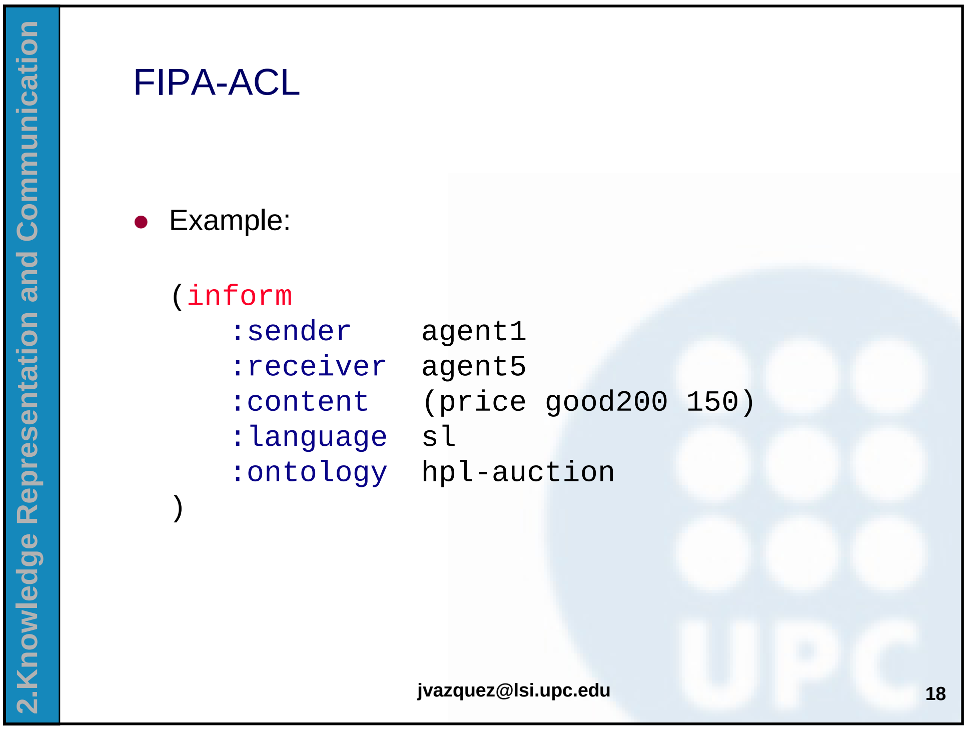
```
( tell
  :sender stock-server
  :receiver joan
  :content (= (price IPOD) (scalar 199 Euro))
  :language KIF
  :ontology NYSE-TICKS )
```

- In literature a short version of KQML/KIF messages is used to specify dialogues:
  ```
  A to B: (ask-if (> (size chip1) (size chip2)))
  B to A: (reply true)
  B to A: (inform (= (size chip1) 20))
  B to A: (inform (= (size chip2) 18))
  A to B: (perform (print “Hello!” t))
  B to A: (reply done)
  ```



FIPA-ACL

- More recently, the Foundation for Intelligent Physical Agents (FIPA) started work on a program of agent standards — the centrepiece is an ACL
- Basic structure is quite similar to KQML:
  - *Type of communicative act: performative*
    22 performatives in FIPA (reduction from KQML)
  - *communication actors*
    e.g., sender, receiver.
  - *content*
    the actual content of the message
  - *Content description*
    e.g., language, encoding, ontology
  - *Conversation control*
    e.g., protocol, conversation-id, reply-with, in-reply-to, reply-by



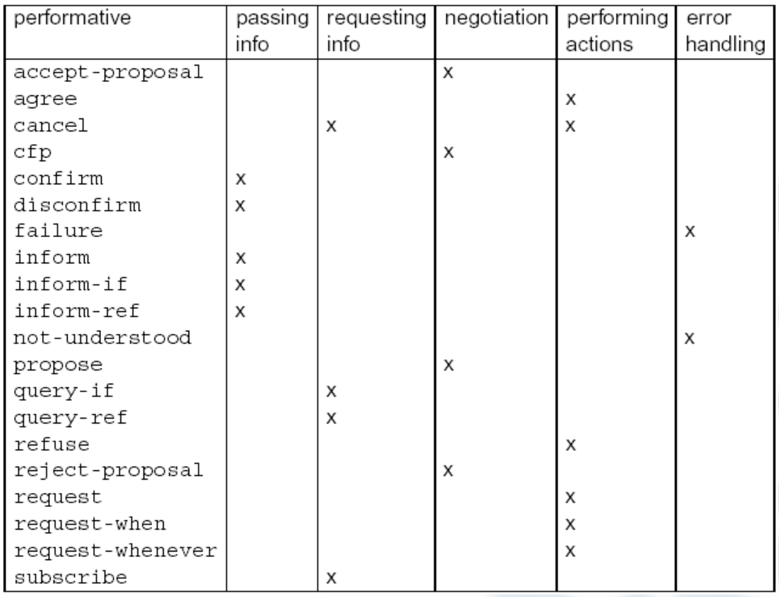
FIPA-ACL

- Example:

```
(inform
    :sender    agent1
    :receiver  agent5
    :content   (price good200 150)
    :language  sl
    :ontology  hpl-auction
)
```

FIPA-ACL

performatives



performative	passing info	requesting info	negotiation	performing actions	error handling
accept-proposal			x		
agree				x	
cancel		x		x	
cfp			x		
confirm	x				
disconfirm	x				
failure					x
inform	x				
inform-if	x				
inform-ref	x				
not-understood					x
propose			x		
query-if		x			
query-ref		x			
refuse				x	
reject-proposal			x		
request				x	
request-when				x	
request-whenever				x	
subscribe		x			

FIPA-ACL

performatives for requests

- `request`, `request-when`, `request-whenever`: request for an action to be performed unconditionally/when a given condition holds/each time the condition holds
- `propose`: to propose an action to be performed when some given conditions hold
- `call-for-proposal`: request for proposals from other agents to perform actions under certain pre-conditions
- `inform-if`, `inform-ref`, `query-if`, `query-ref`: ask the receiver if he believes that a given condition is true or that for a referred element a given condition holds
- `propagate`, `proxy`: request another agent to forward a given message, either reading it and propagating it or propagating without reading
- `subscribe`: request to an agent to inform whenever a given expression/object changes its value

FIPA-ACL

performatives for responses

- `inform`: Informs that a given expression is true
- `accept-proposal`, `reject-proposal`: A proposal (for an action performance) is accepted or rejected
- `confirm, disconfirm`: A fact's truth value is communicated to an agent which has some uncertainty about it
- `agree`: An agreement about performing an action
- `refuse`: A refusal to perform an action (+ reason)
- `cancel`: Cancellation of an agreed action
- `failure`: Action could not be preformed properly
- `not-understood`: Last message has not been understood

FIPA-ACL

Content Language

- Almost any content language can be used with FIPA-ACL. Most used are KIF (ANSI-KIF, ISO-KIF), RDF, DAML, OWL and FIPA-SL
- Others can be used such as PROLOG, SQL, …
- **FIPA-SL** (Semantic Language)
  - Allows representation of asserts in modal
  - It is designed for agents with BDI architecture (Beliefs, Desires, Intentions)
  - Defines 3 types of content:
    - **Statements**: expressions which can be associated with a truth value
    - **Actions**: expressions defining an action that can be performed
    - **Reference expressions**: quantified formulae referring to domain objects which comply with that formulae

FIPA-SL

Elements

- Expressions in FIPA-SL are in prefix notation (such as in KIF)
- It includes connectives from First Order Logic
  - `not, and, or, implies, <=>, forall exist`
- BDI Operators
  - `(B <agent> <exp>)` Agent believes the expression
  - `(U <agent> <exp>)` Agent has some uncertainty about the expression
  - `(I <agent> <exp>)` Agent has as an intention the one in the expression
  - `(PG <agent> <exp>)` Agent has as an objective the one in the expression

FIPA-SL

Elements

- Temporal Logic operators
  - `(feasible <action> <exp>)`: Action can be performed when expression holds
  - `(done <action> <exp>)`: Action was performed before the expression held.
- Relational and list operators
  - `(=, >, <, member, contains)`
- Reference expressions (evaluated through a Knowledge Base)
  - `(iota <terms> <exp>)`: refers to the unique object which, instantiating the terms, makes the expressions true
  - `(any <terms> <exp>)`: refers to a/some objects which, instantiating the terms, make the expressions true
  - `(all <terms> <exp>)`: refers to all objects which, instantiating the terms, make the expressions true

FIPA-SL

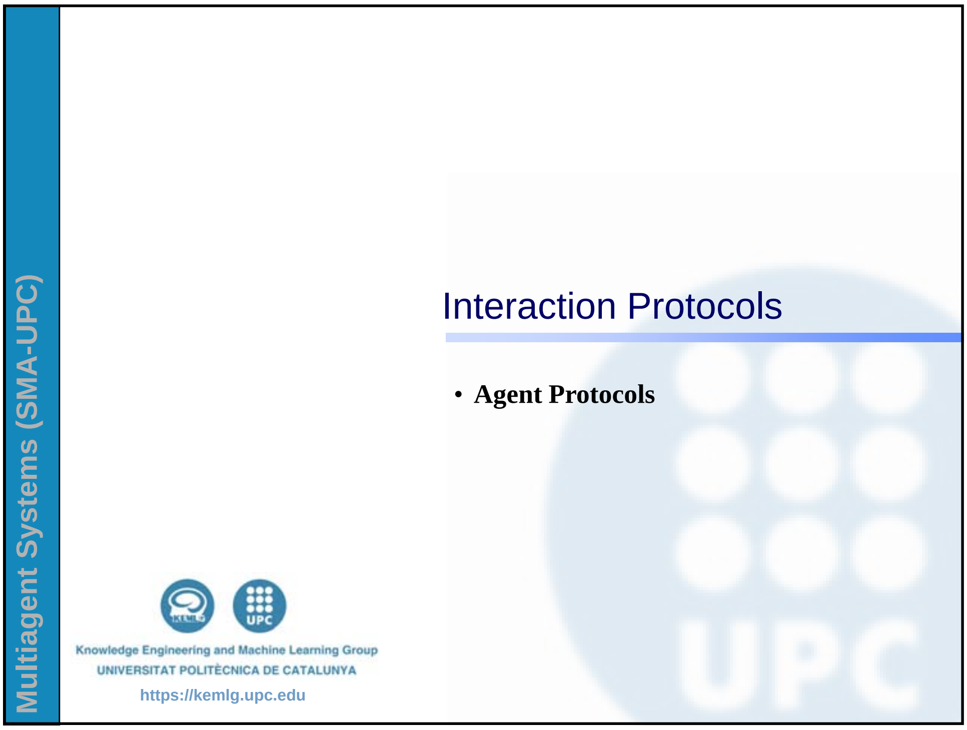
Elements

- Functional Terms (predicates): expressions which refer to an object through its functional relation with other objects (e.g., 3 = (+ 2 1)). There are two alternative expressions:
  - (<predicate> <value$_1$> … <value$_n$>), e.g. (person “Juan” 23)
  - (<predicate <prop$_1$> <value$_1$> … <prop$_n$> <value$_n$>) e.g., (person :name “Juan” :age 23)
- FIPASL has some pre-defined functional terms (arithmetic operators, set operators, list operators…)
- Predicates over actions and results
  - (action <agent> <exp>): we request the agent to perform the action expressed in the expression
  - (result <action> <exp>): informs about the result of a given action

FIPA-SL

3 subsets

- FIPA-SL defines 3 subsets of the language with different expressiveness, for computational reasons
  - **FIPA-SL0**: Allows predicates action, result, done, simple propositions, sets and sequences
  - **FIPA-SL1**: Adds boolean connectives in expressions
  - **FIPA-SL2**: Adds referential expressions and the modal/temporal operators, but with some restrictions to ensure that the demonstrations are decidable

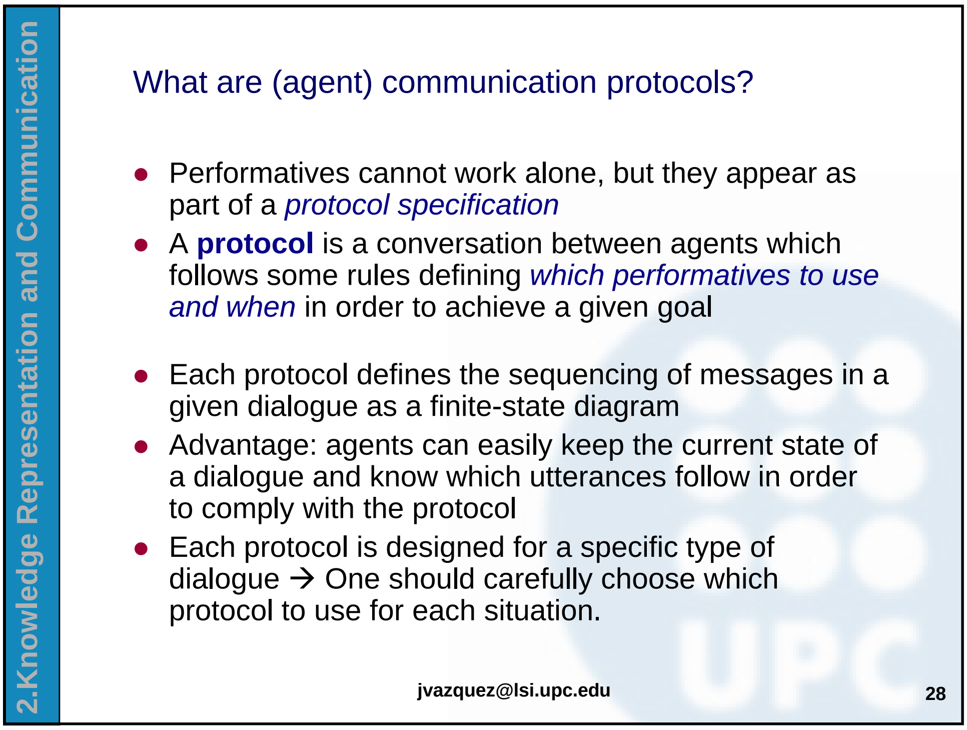


Multiagent Systems (SMA-UPC)

Interaction Protocols

- **Agent Protocols**

Knowledge Engineering and Machine Learning Group
UNIVERSITAT POLITÈCNICA DE CATALUNYA
https://kemlg.upc.edu



2.Knowledge Representation and Communication

What are (agent) communication protocols?

- Performatives cannot work alone, but they appear as part of a *protocol specification*
- A **protocol** is a conversation between agents which follows some rules defining *which performatives to use and when* in order to achieve a given goal
- Each protocol defines the sequencing of messages in a given dialogue as a finite-state diagram
- Advantage: agents can easily keep the current state of a dialogue and know which utterances follow in order to comply with the protocol
- Each protocol is designed for a specific type of dialogue → One should carefully choose which protocol to use for each situation.

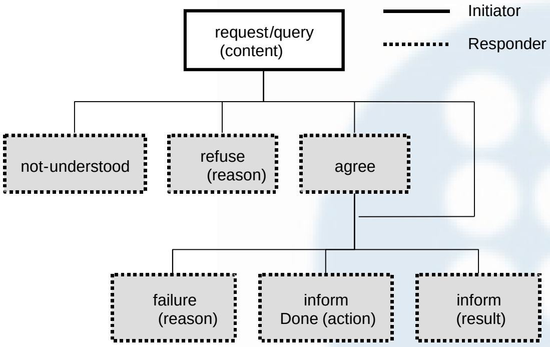
Protocols defined by FIPA

- They have two sides: *initiator* and *responder*.
- FIPA protocols: *Request, Query, Contract Net, Iterated Contract Net, Brokering, Recruiting, Subscribe, Propose*
- The most used are::
  - *Request*: dialogue to ask an agent for an action to be performed. The responder agent gives back the result, if possible
  - *Request-When*: dialogue to ask an agent for an action to be performed whenever some conditions hold
  - *Query*: dialogue to ask an agent if a given expression is true. The responder agent answers, if possible
  - *Propose*: dialogue to propose another agent to perform a given action under given conditions. The responder agent accepts or rejects the proposal
  - *Contract Net*: dialogue to request a group of agents to send back proposals for actions to solve a given task. The initiator agent selects the best proposals

FIPA protocols

Request-Response Protocols

- E.g. FIPA specification for *FIPA-Query* and *FIPA-Request*



request/query (content)

Initiator

Responder

not-understood

refuse (reason)

agree

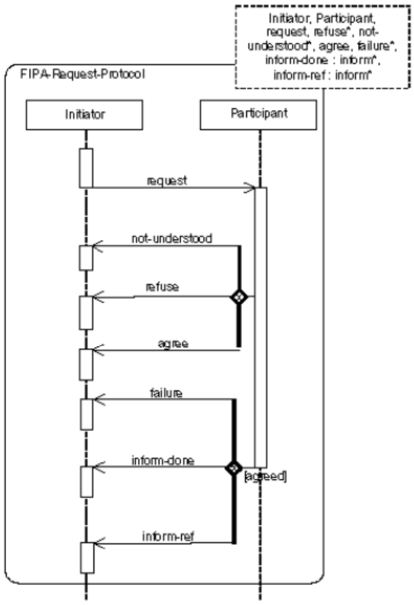
failure (reason)

inform Done (action)

inform (result)

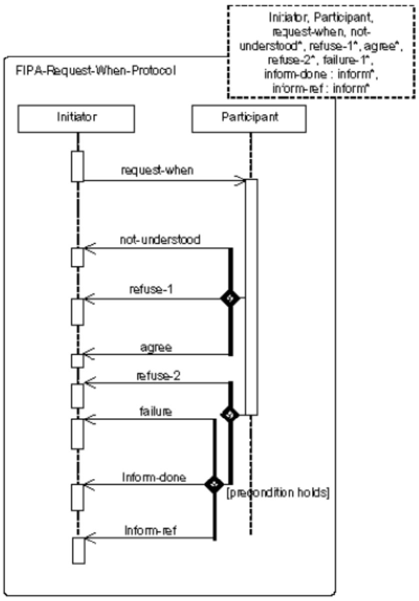
FIPA protocols

FIPA-Request



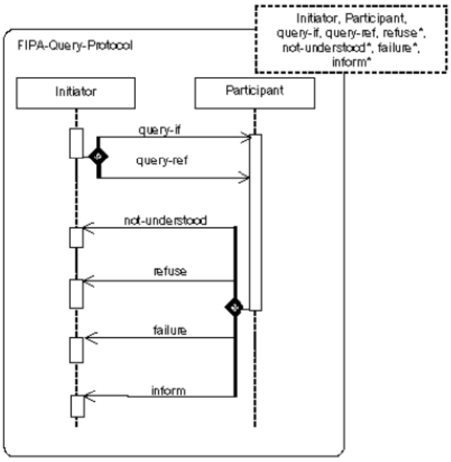
FIPA protocols

FIPA-Request-When



FIPA protocols

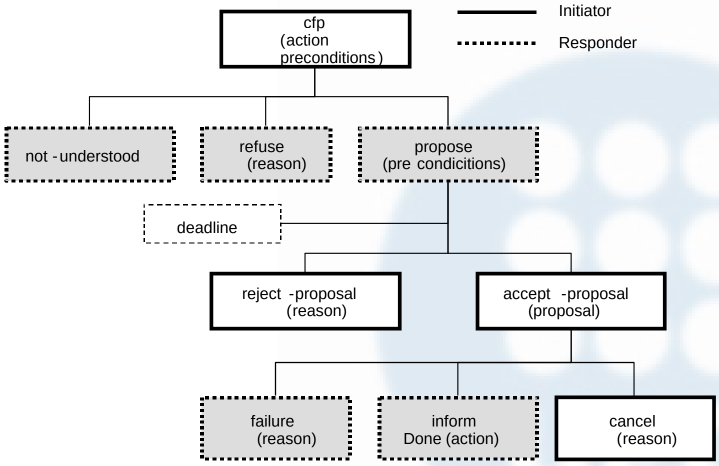
FIPA-Query



FIPA protocols

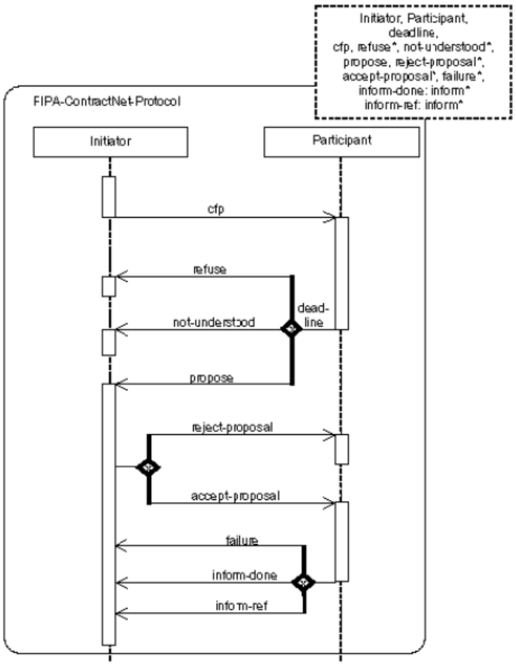
FIPA-Contract-Net (I)

- E.g. FIPA specification for *Contract Net*



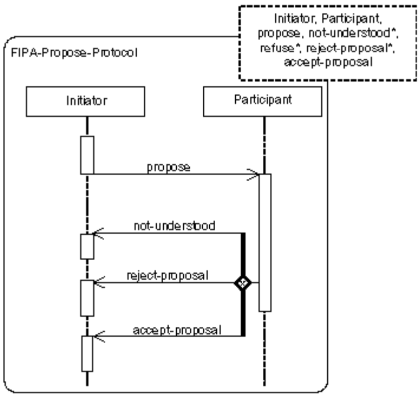
FIPA protocols

FIPA-Contract-Net (II)



FIPA protocols

FIPA-Propose



References

[1] Luck, M., McBurney, P., Shehory, Onn, Willmott, S. “Agent Technology: Computing as interaction. A Roadmap to Agent Based Computing”. Agentlink, 2005. ISBN 085432 845 9

[2] Wooldridge, M. “Introduction to Multiagent Systems”. John Wiley and Sons, 2002.

[3] FIPA Agent Communication specifications. http://www.fipa.org/repository/aclspecs.html

[4] Haddadi, A. “Communication and Cooperation in Agent Systems: A Pragmatic Theory” Lecture Notes in Artificial Intelligence #1056. Springer-Verlag. 1996. ISBN 3-540-61044-8

[5] Weiss, G. “Multiagent Systems: A modern Approach to Distributed Artificial Intelligence”. MIT Press. 1999. ISBN 0262-23203

These slides are based mainly in material from [2] and from J. Bejar, with some additions from material by U. Cortés and A. Moreno