Distributed Artificial Intelligence and Intelligent Agents

Agent Communication

Mihhail Matskin

Agent Communication

"Language is a very difficult thing to put into words"

Voltaire

- Approaches to software interoperation
- Basic components and theory
- Knowledge Sharing Effort approach
  - KIF
  - KQML
  - implementation
- FIPA approach

References - Curriculum

- Wooldridge: ”Introduction to MAS”,
  - Chapter 7 (Chapter 6 is optional)

Software interoperation

Interoperation - exchange of information and services with other programs, thereby solving problems that cannot be solved alone

Main problems:

- heterogeneity
  - different people
  - different time
  - different languages

Approaches to software interoperation

Should be provided

- standard communication languages
- common libraries
- run-time support

Questions to be answered:

- What is an appropriate agent communication language?
- How do we build agents capable of communicating in this language?
- What communication "architectures" are conductive to cooperation?

Components of a system for effective interaction and interoperability

- common language
- common understanding of the knowledge exchanged
- ability to exchange whatever is included in the previous items

Agent Communication

- Agents have **conversations** (as opposed to exchange single messages), e.g.
  - Task-oriented
  - Negotiations
- Communication behavior should allow to express agents strategies, intentions, roles etc

Three Important Aspects

Syntax	1.How the symbols of communication are structured.
Semantics	2. What the symbols denote.
Pragmatics	3. How the symbols are interpreted.

(Meaning is a combination of semantics and pragmatics.)

Requirements for an ACL

Syntactic

- **Should allow syntactic translation** between languages

Meaning

- **Should allow meaning content preservation** among applications
  - The concept must have a uniform meaning across applications.

Communication

- **Should be able communicate complex attitudes** about their information and knowledge.
  - Agents need to question, request, etc.
  - Not about transporting bits and bytes.

What distinguishes ACLs?

- Semantic complexity
- ACLs can handle propositions, rules and actions instead of simple objects with no semantics associated with them.
- An ACL message describes a desired state in a declarative language, rather than a procedure or a method.

Speech Acts 1

- Communication in MAS is inspired by **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.
- The origin of speech act theories are usually traced to the work of the philosopher John Austin (1962) then developed by John Searl (1969)

Speech Acts 2 - Austin

- Austin noticed that some utterances are like ”physical actions” that appear to change the state of the world. e.g.
  - Declaring war
  - ”I now pronounce you man and wife”
- In general, everything we utter is uttered with the purpose of satisfying some goal or intention.
- Humans communicate by uttering sentences expressing their intentions

➢ A theory of how utterances are used to achieve intentions is a **speech act theory**.

Speech Acts 3 - Austin

- Austin distinguished 3 different aspects of speech acts:

1. Locutionary act - act of making an utterance
    - e.g. saying ”please make some tea” ’
2. Illocutionary act – action performed in saying something
    - e.g. he requested me to make some tea
3. Perlocution – effect of the act
    - e.g. he got me to make tea

Speech Acts 4 - Austin

We consider illocutionary act only.

An illocutionary act F(P) is composed from

- propositional content P
- context
- illocutionary force F

Illocutionary force divides speech acts into categories described by

- illocutionary point
- direction of fit
- sincerity conditions

Speech Acts - Searle

Illocutionary act	**Illocutionary point**	**Direction of fit**	**Sincerity condition**
Assertives/ Representatives	"It rains“ Commits speaker to truth of utterance	world-to-word (describe world)	Speaker believes utterance
Directives	"Close the window“ Speaker tries to make hearer do something	word-to-world (change world)	Speaker wants Hearer to establish the truth of utterance
Commisives	"I will“ Commits Speaker to future action	word-to-world (change world)	Speaker intends to act such that the truth of the utterance is established
Expressives	"Excuse me", "Congratulations" Express psychological state	None	Several possibilities
Declaratives	"I name this door the Golden Gate“ Establish correspondence between utterance and world	world-to-word and word-to-world	None

Some additional speech acts

- permissives:
  - "you may shut the door"

- prohibitives:
  - "you may not shut the door"

Communication Levels

Semantics+Pragmatics

Meaning of the interaction

Syntax

Format of information being transferred

Communication

Method of interconnection

Speech Acts 5 - Components

- In general, speech acts can be seen to have 2 components:
  1. An illocutionary force represented by **performative verb**
     - e.g. Request, inform
  2. **Propositional content**
     - e.g. ”the door is closed”

Speech Act	Please close the door	The door is closed	Is the door closed?
Performative	request	inform	inquire
Content	the door is closed	the door is closed	the door is closed

Speech Acts 6 – Plan-based Theory

- How does one define the semantics of speech acts? When can one say someone has uttered, e.g. a *request* or an *inform*?
- How can the properties of speech acts be represented such that planning systems could reason about them?

➢ **Speech acts are treated as physical actions**.

➢ Actions are characterised via **preconditions and postconditions**.

Speech Acts 7
Plan-based Theory Example

- Semantics for request: ***request*(*s*,*h*,α)**
- Pre:
  - ***s* believes (*h* can do α)**

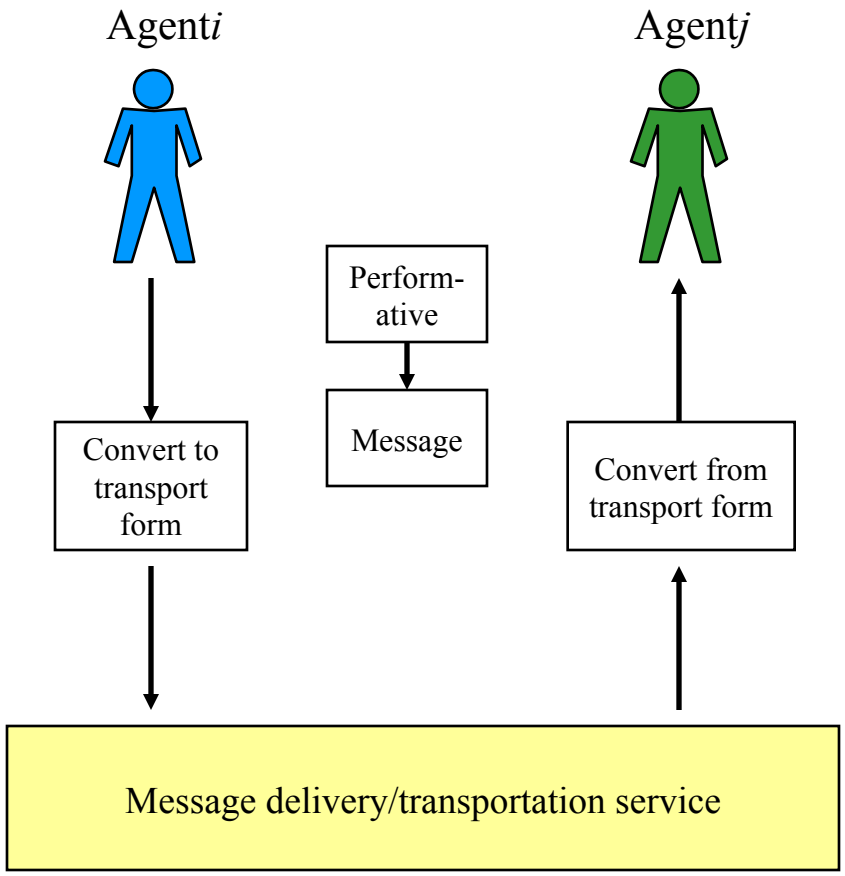
    (you don't ask someone to do something unless you think that they can do it)
  - ***s* believes (*h* believes (*h* can do α))**

    (you don't ask someone unless they believe they can do it)
  - ***s* believes (*s* wants α)**

    (you don't ask someone unless you want it!)
- Post:
  - ***h* believes (*s* believes (*s* wants α))**

    (the effect is to make them aware of your desire)

Agent Communication Language (ACL)



ACLs allow agents to effectively communicate and exchange knowledge with other agents.

Origins of ACLs

Knowledge Sharing Effort (KSE)

Basic assumption

Software agents are applications for which ability to communicate with other applications and share knowledge is of primary importance

Basic components

- Communication
- Representation
- Supporting components

Knowledge Sharing Effort (KSE) cont.

Communication

- interaction protocol
- communication language
- transport protocol

Representation

- Ontologies
- Knowledge bases

Supporting components

- Planning
- Modeling
- Meta-Knowledge
- Reasoning

Origins of ACLs

- Main results of KSE

➢ **KIF**: Knowledge Interchange Format (Chapter 6.2.3)

– Langauge for expressing message content.

➢ **Ontolingua:**

- Ontology presentation

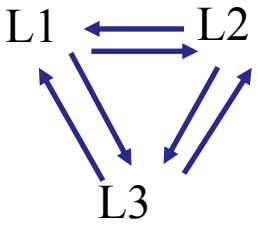
➢ **KQML**: Knowledge Query and Manipulation Language (Chapter 7.2.1)

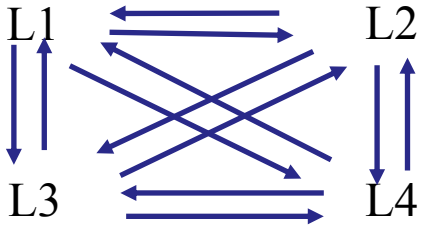
– Language for both message formatting and message handling protocols.

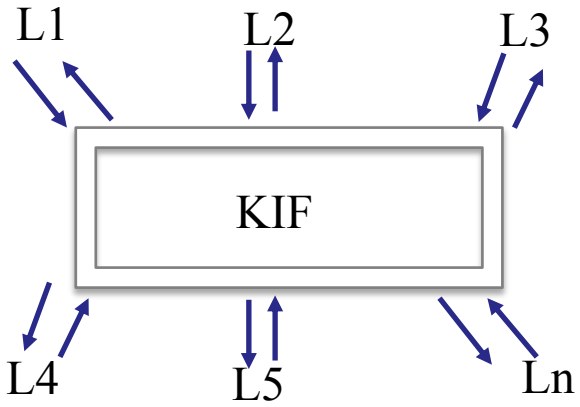
Knowledge Interchange Format (KIF) Motivations

- creation of a language for development of intelligent applications
- creation of common interchange format
- Intented to express contents of a message; not the message itself.
- KIF is not intended for interaction with human user
- KIF is not intended to be internal representation for knowledge within computer programs

Interchange format







KIF

KIF is a prefix version of first order predicate calculus

a+b + a b

- KIF has declarative semantics
- KIF is logically comprehensive
- KIF provides for representation of knowledge about representation of knowledge (meta-level)

Additional features

- Translatability
- Readability
- Usability as a representation language

KIF

- Using KIF, it's possible to express:
  - **Properties** of things in a domain
    - e.g. *Mike is a vegetarian* – Mike has the property of being a vegetarian
  - **Relationships** between things in a domain
    - e.g. *Mike and Janine are married* – the relationship of marriage exists between Mike and Janine.
  - **General properties** of a domain
    - e.g. *Everybody has a mother*.

KIF

- Variables:
  - ?x, ?res, @seq
- Operators
  - term operators: listof, if,...
  - sentence operators: not, and, /=, ...
  - rule operators: =>>, ...
  - definition operators: defobject, deffunction,...
- Constants
  - object constants
  - function constants
  - relational constants
  - logical constants
- Expressions/Sentences
  - word or a finite sequence of words

KIF - Example

- Terms:
  - `(size chip1)`
  - `(+ (sin theta)(cos theta))`
  - `(if (> theta 0) theta (-theta))`
- Sentences:
  - *They are build from terms*
  - *They are used to express facts about the world*
  - `(> (sin theta )(cos phi))`
  - `(prime 565762761)` – *prime number*
  - `(=> (> ?x 0)(positive ?x))` – *implication*

KIF - Example

- Definition of new concept:
  - *An object is a bachelor if this object is a man and not married:*

```
(defrelation bachelor (?x) :=
    (and (man ?x)
         (not (married ?x))))
```

- Relationship between individuals in the domain:
  - *Any person with the property of being a person also has the property of being a mammal:*

```
(defrelation person (?x) :=> (mammal ?x))
```

KIF

- Metaknowledge

  (believes John '(material moon jarlsberg))

  (=> (believes John ?p) (believes Mary ?p))

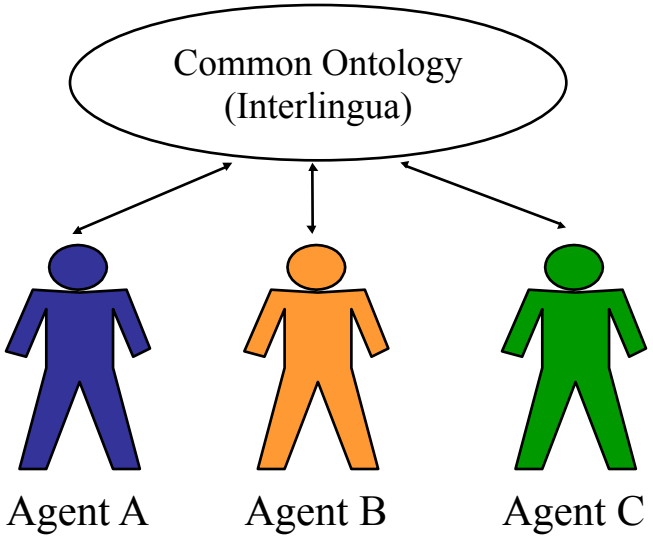
- Rules

  (<<= (flies ?x)(bird ?x))

  (<<= (flies ?x)(bird ?x)(consis (flies ?x)))

Ontologies

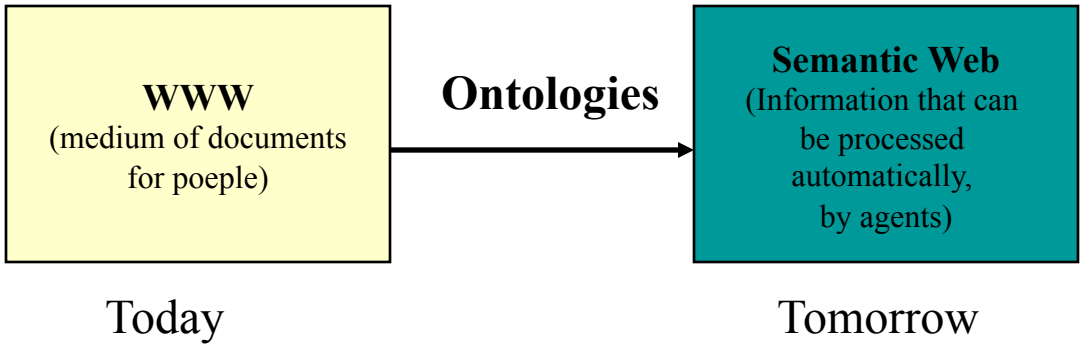
- "An ontology is a formal explicit specification of a shared conceptualization." (Gruber 1993)
- An ontology is a description of the concepts and relationships that can exist for an agent or a community of agents.
- Why do we need an ontology?
  - To agree on a terminology to describe a domain.



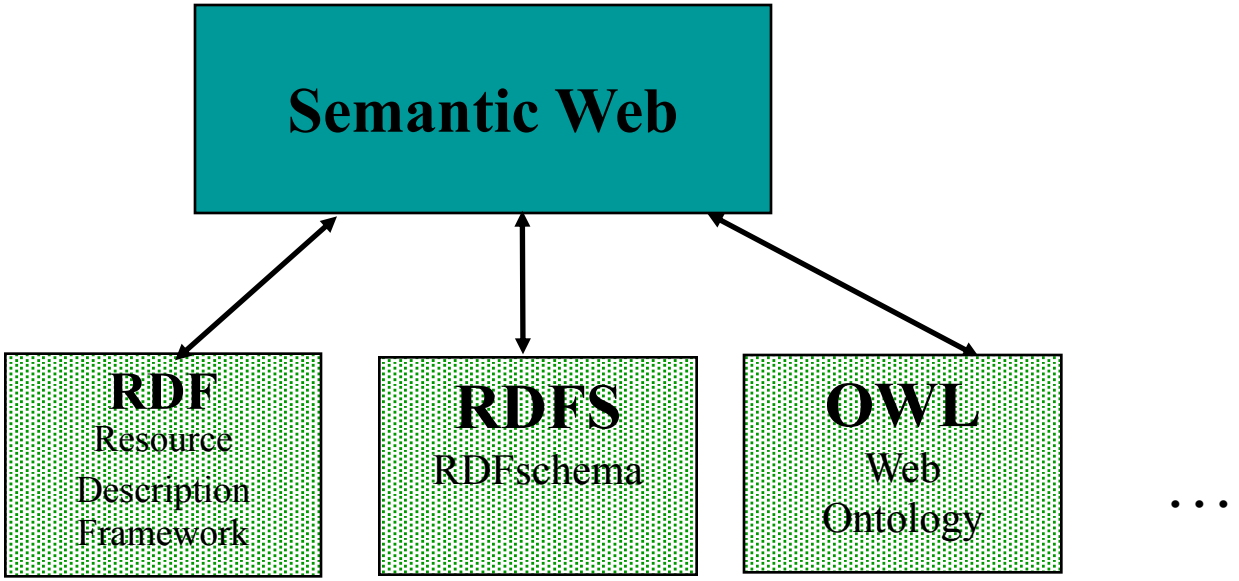
- **Ontolingua** is one of the first ontologies developed using KIF.
- A web-based service intended to provide a common platform in which ontologies developed by different groups can be shared.

Ontology and Semantic Web

- Tim Berners-Lee et al. considers ontologies as a critical part of the work on the **Semantic Web**.
- Semantic Web: ”an extension of the current WWW, in which information is given well-defined meaning, better enabling computers and people to work in cooperation.” *(Berners-Lee et. al., 2001)*



Ontology Languages for The Semantic Web



Knowledge Query Manipulation Language (KQML)

Message format and message-handling protocol

Basic features

- KQML message do not communicate sentences but rather communicate attitude about sentences
- The language primitives are performatives and they define permissible actions that agent may attempt in communicating
- KQML environment (may) contain facilitators which help to make communication protocol transparent

KQML - Performatives

- The idea of communication in KQML is to represent illocutionary acts.
- **Performatives** form the core of the language:
  - Determine the kinds of interactions one can have with KQML-speaking agents.
  - Identify the protocol to be used to deliver the message
  - Signify that the content is an assertion, a query, a command or another speech act.
  - Describe how the sender would like any reply to be delivered.

KQML
Categories of Performatives

Category	Performatives
Basic Query	evaluate, ask-if, ask-one, ask-all, ask-about
Multi-response Query	stream-about, stream-all
Response	reply, sorry
Generic informational	tell, achieve, cancel, untell
Generator	standby, ready, next, rest, discard, generator
Capability-definition	advertise, subscribe, monitor, import, export
Networking	register, unregister, forward, broadcast, route

Some examples

A to B: (tell (> 3 2))

A to B: (perform (print "Hello!" t))

B to A: (reply done)

A to B: (ask-if (> (size chip1) (size chip2)))

B to A: (reply true)

A to B: (subscribe (position ?x ?r ?c))

B to A: (tell (position chip1 8 10))

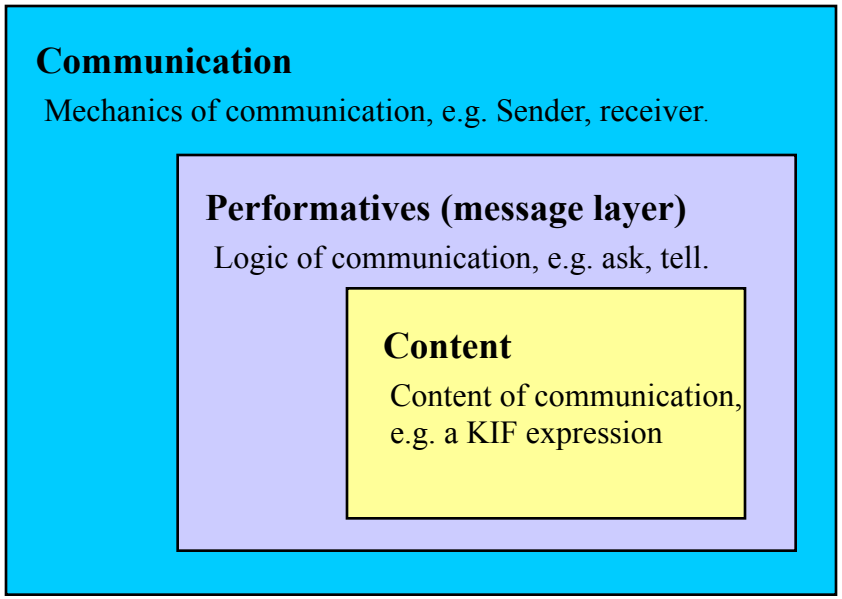
B to A: (tell (position chip2 8 46))

B to A: (tell (position chip3 8 64))

A to B: (unsubscribe (position ?x ?r ?c))

KQML

- An "outer language" that defines a set of performatives (communicative acts), such as *ask, reply*.



e.g.

(ask-if :sender *agenti*
:receiver *agentj*
:language *KIF*
:ontology *genealogy*
:content *"(spouse adam, eve)")*

KQML

KQML statement consists of

- a performative
- its associated arguments
  - message contents
  - optimal transport and context information

- General syntax:
  (KQML-performative
  :sender *word*
  :receiver *word*
  :language *word*
  :ontology *word*
  :content *expression*
  ...)

- :content, :language, :ontology - meaning of the message
- :sender, :receiver, :reply-with, :in-reply-to - parameters of message passing

KQML

```
(ask-one
  :content (geoloc ARN (?long ?lat))
  :ontology geo-model3)


(ask-all
  :content "price(IBM, [?price, ?time])"
  :receiver stock-server
  :language standard-prolog
  :ontology NYSE-TICS)
```

KQML

```
(standby
  :content(stream-all
                :content (price ?VL ?price)))

(subscribe
  :content(stream-all
                :content (price IBM ?price)))

(advertise
  :ontology NYSE-TICS
  :language LPROLOG
  :content (monitor :content (price ?x ?y)))
```

KQML

```
(tell
   :sender Agent1
   :receiver Agent2
   :language KIF
   :ontology Blocks-World
   :content (AND (Block A) (Block B) (On
   (Block A) (Block B)))

(forward
   :from Agent1
   :to Agent2
   :sender Agent1
   :receiver Agent3
   :language KQML
   :ontology kqml-ontology
   :content (tell
       :sender Agent1
       :receiver Agent2
       :language KIF
       :ontology Blocks-World
       :content (On (Block A) (Block B))))
```

KQML 4 - Examples

```
(evaluate
   :sender A
   :receiver B
   :language KIF
   :ontology motors
   :reply-with q1
   :content (val (torque m1)))

(reply
   :sender B
   :receiver A
   :language KIF
   :ontology motors
   :in-reply-to q1
   :content (=  (torque m1) (scalar 12 kgf)))
```

KQML 5 - Examples

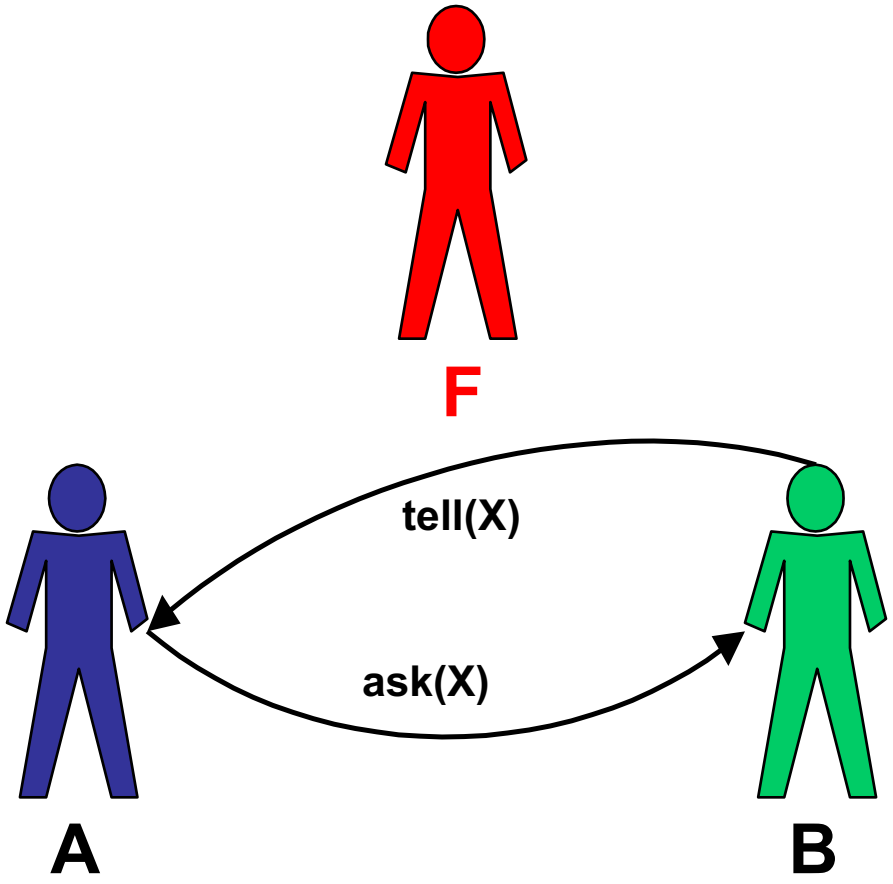
```
(stream-about
   :sender A
   :receiver B
   :language KIF
   :ontology motors
   :reply-with q1
   :content (m1)

(tell
   :sender B : receiver A
   :in-reply-to q1
   :content (=  (torque m1) (scalar 12 kgf)))
(tell
   :sender B : receiver A
   :in-reply-to q1
   :content (=  (status m1) (normal)))
(eos
   :sender B : receiver A
   :in-reply-to q1)
```

Facilitators 1

- KQML environments (may) contain **facilitators** that help make the communication protocol transparent.
- Facilitators: a special class of **agents that perform useful communication** services such as:
  - Maintain registry of service names
  - Forward messages to named services
  - Routing messages based on content (:ontology, :language etc)
  - Provide matchmaking between information providers and requesters
  - Provide mediation and translation services
  - …

Facilitators 2

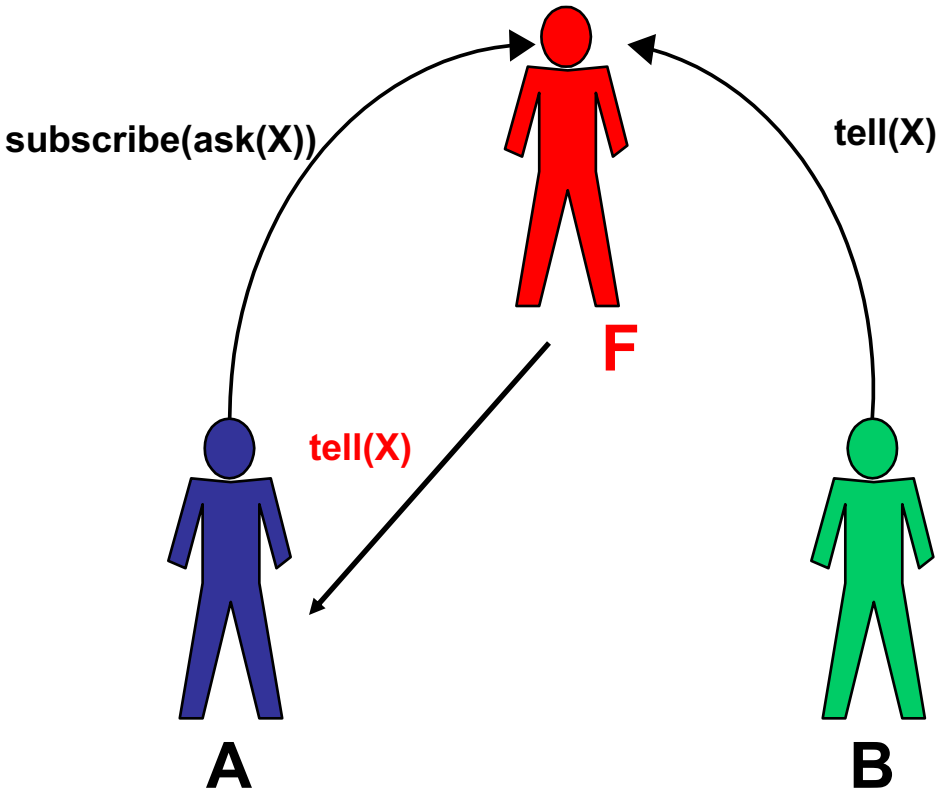


Point-to-point protocol

A is aware about B then it is appropriate to send a query about X to B

If A is not aware about B then there are several ways to achieve this via a Facilitator.

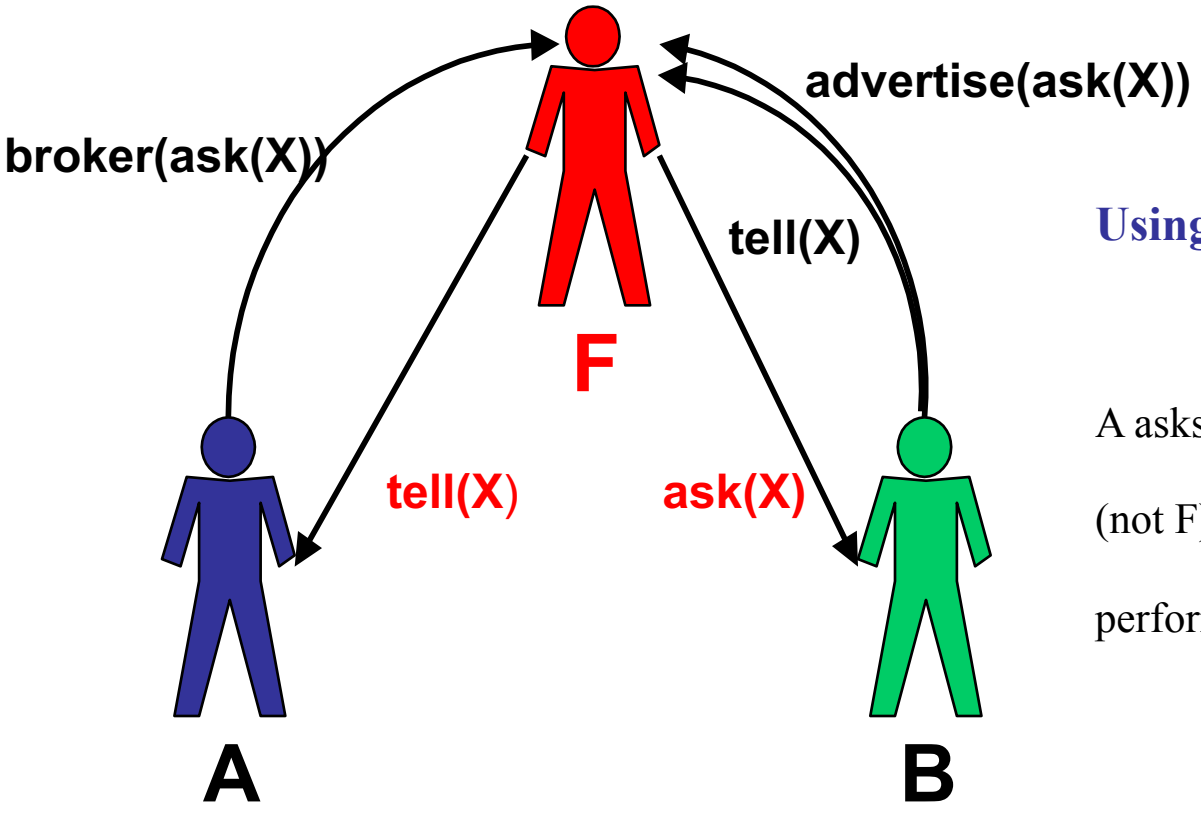
Facilitators 3



Using the *subscribe* performative

Request that Facilitator F helps find the truth of X. If B subsequently informs F that it believes X to be true, then F can in turn inform A.

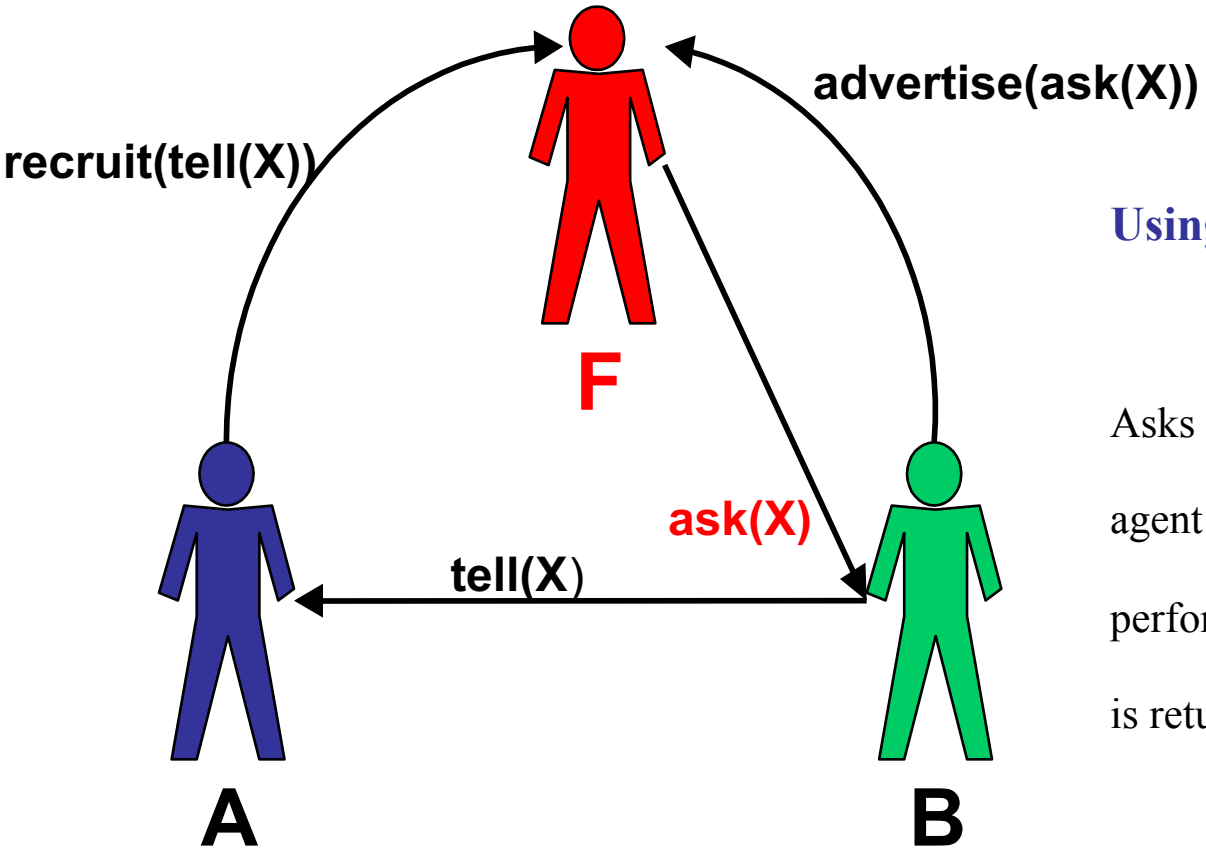
Facilitators 4



Using the *broker* performative

A asks Facilitator to find another agent (not F) which can process a given performative.

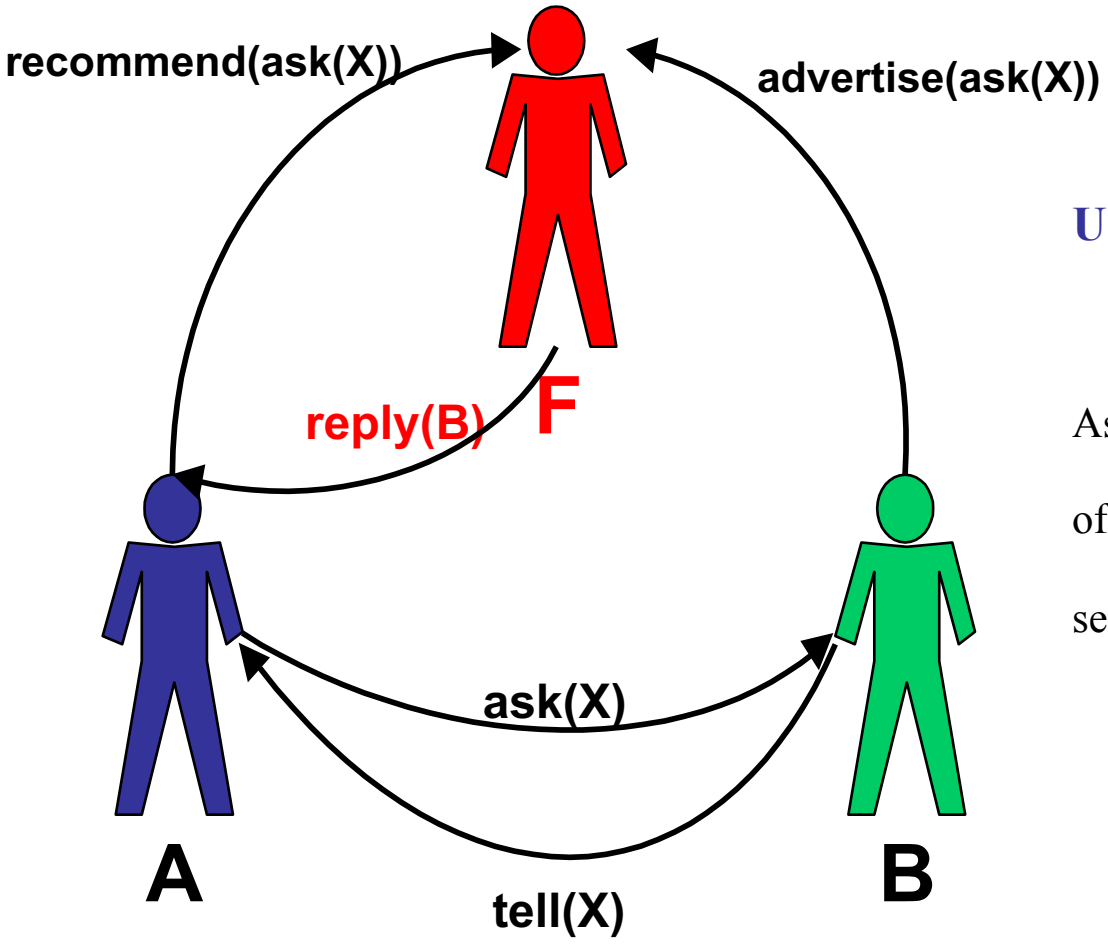
Facilitators 5



Using the *rercruit* performative

Asks Facilitator to find an appropriate agent to which an embedded performative can be forwarded. A reply is returned directly to the original agent.

Facilitators 6



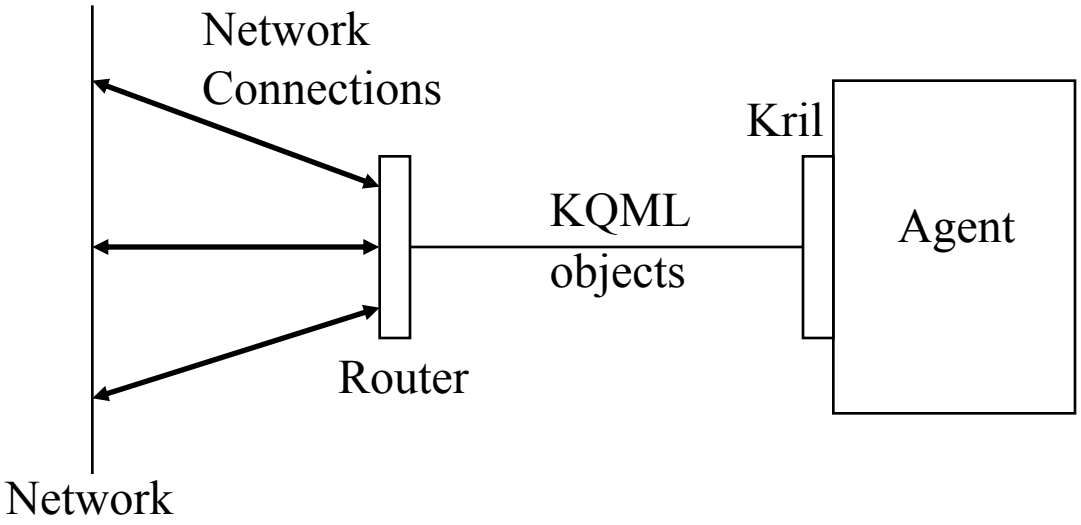
Using the *recommend* performative

Asks Facilitator to respond with the ”name” of another agent which is appropriate for sending a particular performative.

KQML internal structure

Communication architecture is built around

- facilitators
- routers
- library of interfaces (KRIL)

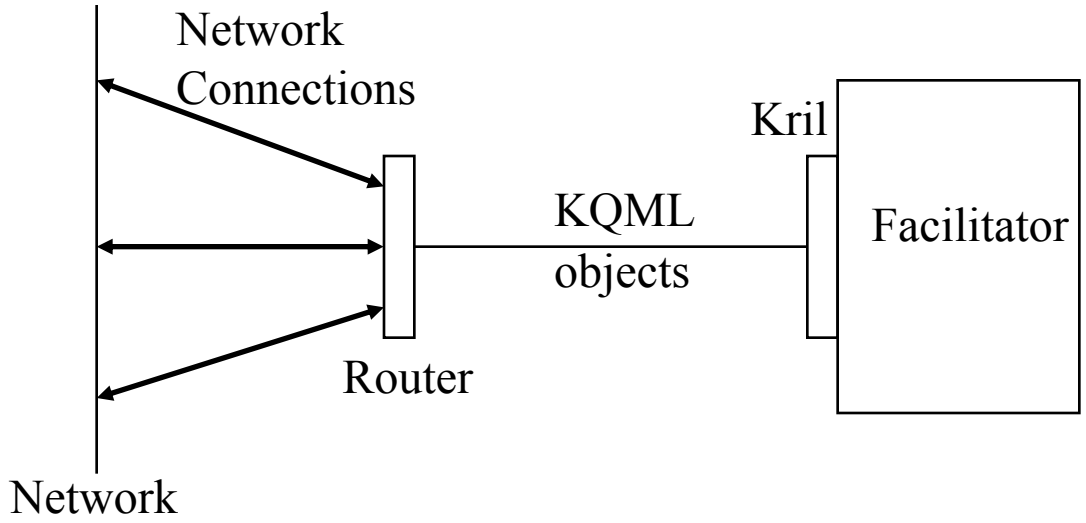


Routers

- content independent message routers
- each KQML agent is associated with its own separate router process
- router handles all KQML messages going to and from the associated agent
- can try to find Internet address for service and deliver message to it

How facilitators help in routing

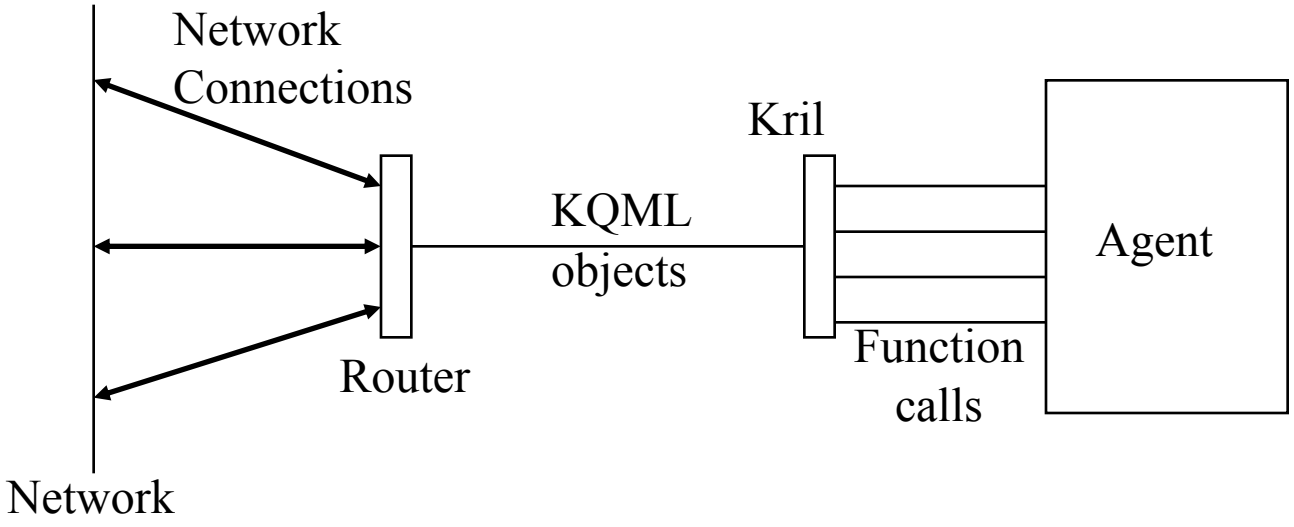
Facilitators are agents with own KQML routers



Typically one facilitator for each local group of agents

KQML Router Interface Library (Kril)

- a programming interface between router and agent
- it is embedded into application
- the main role of KRIL - to make access to the router as simple as possible for the programmer



KQML Criticism

- Weak semantics of performatives
  - Different implementations of KQML could not interoperate.
- Transportation mechanisms were not defined.
- Lacked the class of performatives: commissives
  - Difficult to implement multi-agent scenarios without commissives.
- Set of performatives was large and ad hoc.

➢ Recently, more efforts have been made to provide formal semantics in terms of preconditions, postconditions and completion conditions.

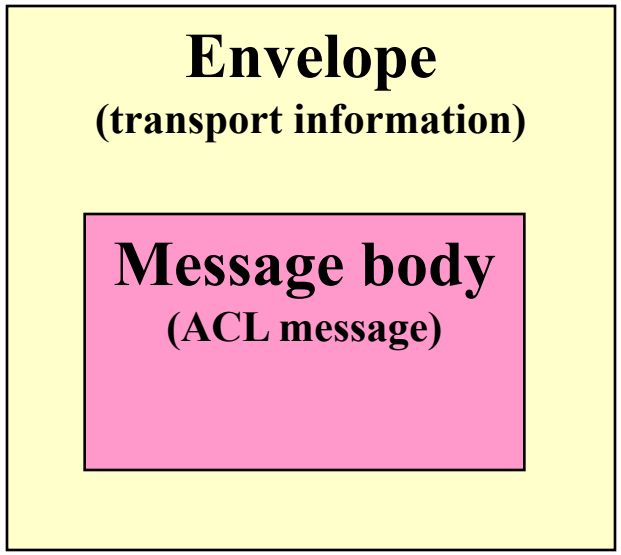
Who is FIPA?

- The Foundation for Intelligent Physical Agents (FIPA) is a non-profit association.
- FIPA's purpose is to promote the success of emerging agent-based applications, services and equipment.
- FIPA operates through the open international collaboration of member organizations: companies, universities and government organizations.

FIPA ACL 1

- Basic structure is quite similar to KQML:
  - Performative (communicative act)
    - 22 performatives in FIPA ACL
  - Housekeeping
    - e.g. Sender
  - Content
    - the actual content of the message

FIPA ACL 2
Message Structure



- Envelope:
  - Comprises of a collection of parameters
  - Contains atleast the mandatory *to* and *from* parameters

FIPA Envelope Parameter Semantics

- **to -** If no **intended-receiver** parameter is present, then the information in this parameter is used to generate **intended-receiver** field for the messages the Agent Communication Channel (ACC) subsequently forwards.
- **from -** If required, the ACC returns error and confirmation messages to the agent specified in this parameter.
- **comments**
- **acl-representation -** This information is intended for the final recipient of the message.
- **payload-length -** The ACC may use this information to improve parsing efficiency.
- **payload-encoding -** This information is intended for the final recipient of the message.
- **date -** This information is intended for the final recipient of the message.
- **intended-receiver -** An ACC uses this parameter to determine where this instance of a message should be sent. If this parameter is not provided, then the first ACC to receive the message should generate an **intended-receiver** parameter using the **to** parameter.
- **received -** A new received parameter is added to the envelope by each ACC that the message passes through. Each ACC handling a message must add a completed received parameter. If an ACC receives a message it has already stamped, it is free to discard the message without any need to generate an error message.
- **transport-behaviour -** reserved for future use.

FIPA ACL 3

- Example FIPA ACL message:

```
(inform
        :sender         agentA
        :receiver       agentB
        :content        (price good200 150)
        :language       KIF
        :ontology       hpl-auction
)
```

Pre-defined message parameters

Message Parameter:	**Meaning:**
:sender	Denotes the identity of the sender of the message
:receiver	Denotes the identity of the intended recipient of the message.
:content	Denotes the content of the message; equivalently denotes the object of the action.
:reply-with	Introduces an *expression* which will be used by the agent responding to this message to identify the original message.
:in-reply-to	Denotes an expression that references an earlier action to which this message is a reply.
:encoding	Denotes the specific encoding of the content language expression .
:language	Denotes the encoding scheme of the content of the action.
:ontology	Denotes the ontology which is used to give a meaning to the symbols in the content expression.
:reply-by	Denotes a time and/or date expression which indicates a guideline on the latest time by which the sending agent would like a reply.
:reply-to	In requesting messages, this parameter indicates that replying messages are to be sent to its value's agent name instead of the :sender of the received requesting message.
:protocol	Introduces an identifier which denotes the protocol which the sending agent is employing. The protocol serves to give additional context for the interpretation of the message.
:conversation-id	Introduces an expression which is used to identify an ongoing sequence of communicative acts which together form a conversation.

Catalogue of Communicative Acts

Communicative act	Information passing	Requesting information	Negotiation	Action performing	Error handling
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 4
Inform and Request

- **Inform** and **Request** are the two basic performatives in FIPA ACL.
- The meaning of inform and request are defined in 2 parts:

1. **Precondition**
   - What must be true in order for the speech act to succeed.
2. **Rational effect**
   - What the sender of the message hopes to bring about.

FIPA ACL 5
Inform and Request

	Inform	**Request**
Content	statement	action
Precondition	**•Holds that the content is true.** **•Intends that the recipient believes the content** **•Does not already believe that the recipient is aware whether content is true or not**	**•Intends action content to be performed** **•Believes recipient is capable of performing this action** **•Does not believe that sender already intends to perform action**

Inform

Summary:	The sender informs the receiver that a given proposition is true.
Message content:	A proposition.
Description:	inform indicates that the sending agent: · holds that some proposition is true, · intends that the receiving agent also comes to believe that the proposition is true, and, · does not already believe that the receiver has any knowledge of the truth of the proposition. The first two properties defined above are straightforward: the sending agent is sincere, and has (somehow) generated the intention that the receiver should know the proposition (perhaps it has been asked). The last property is concerned with the semantic soundness of the act. If an agent knows already that some state of the world holds (that the receiver knows proposition *p*), it cannot rationally adopt an intention to bring about that state of the world, that is, that the receiver comes to know *p* as a result of the inform act. Note that the property is not as strong as it perhaps appears. The sender *is not* required to establish whether the receiver knows *p*. It is only the case that, in the case that the sender already happens to know about the state of the receiver's beliefs; it should not adopt an intention to tell the receiver something it already knows. From the receiver's viewpoint, receiving an inform message entitles it to believe that: · the sender believes the proposition that is the content of the message, and, · the sender wishes the receiver to believe that proposition also. Whether or not the receiver does, indeed, adopt belief in the proposition will be a function of the receiver's trust in the sincerity and reliability of the sender.
Summary Formal Model	$\langle i, \text{inform}\ (j, f)\rangle$ FP: $B_i\,\varphi \wedge \neg B_i\,(\text{Bif}_j\,\varphi \vee \text{Uif}_j\,\varphi)$ RE: $B_j\,\varphi$
Example	Agent *i* informs agent *j* that (it is true that) it is raining today. (inform :sender (agent-identifier :name i) :receiver (set (agent-identifier :name j)) :content "weather (today, raining)" :language Prolog)

FIPA Agent Interaction Protocol 1

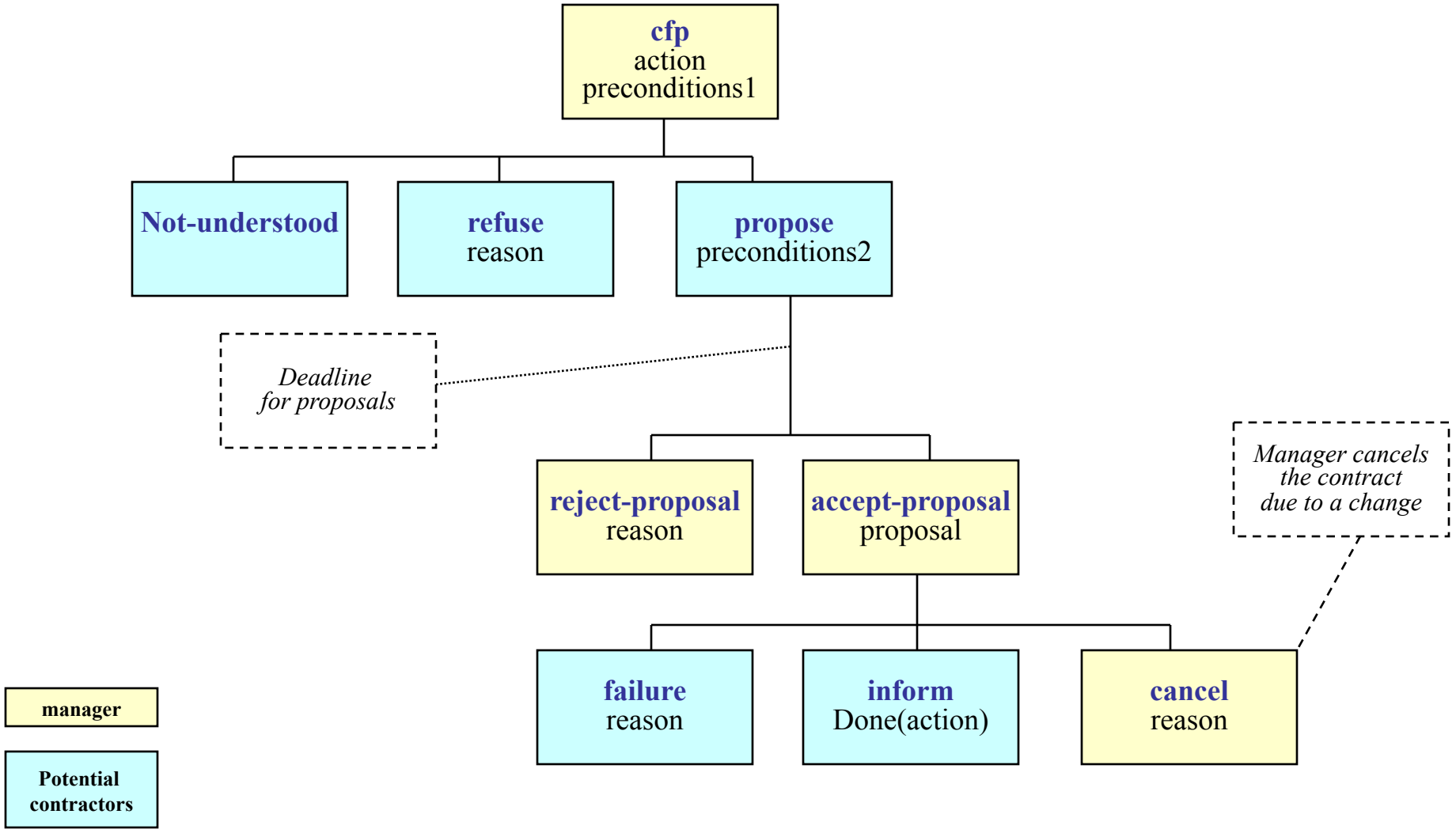
- Ongoing conversations between agents fall into typical patterns. In such cases, certain message sequences are expected, and at any point in the conversation, other messages are expected to follow.

➢ These **typical patterns of message exchange** are called ***protocols***.

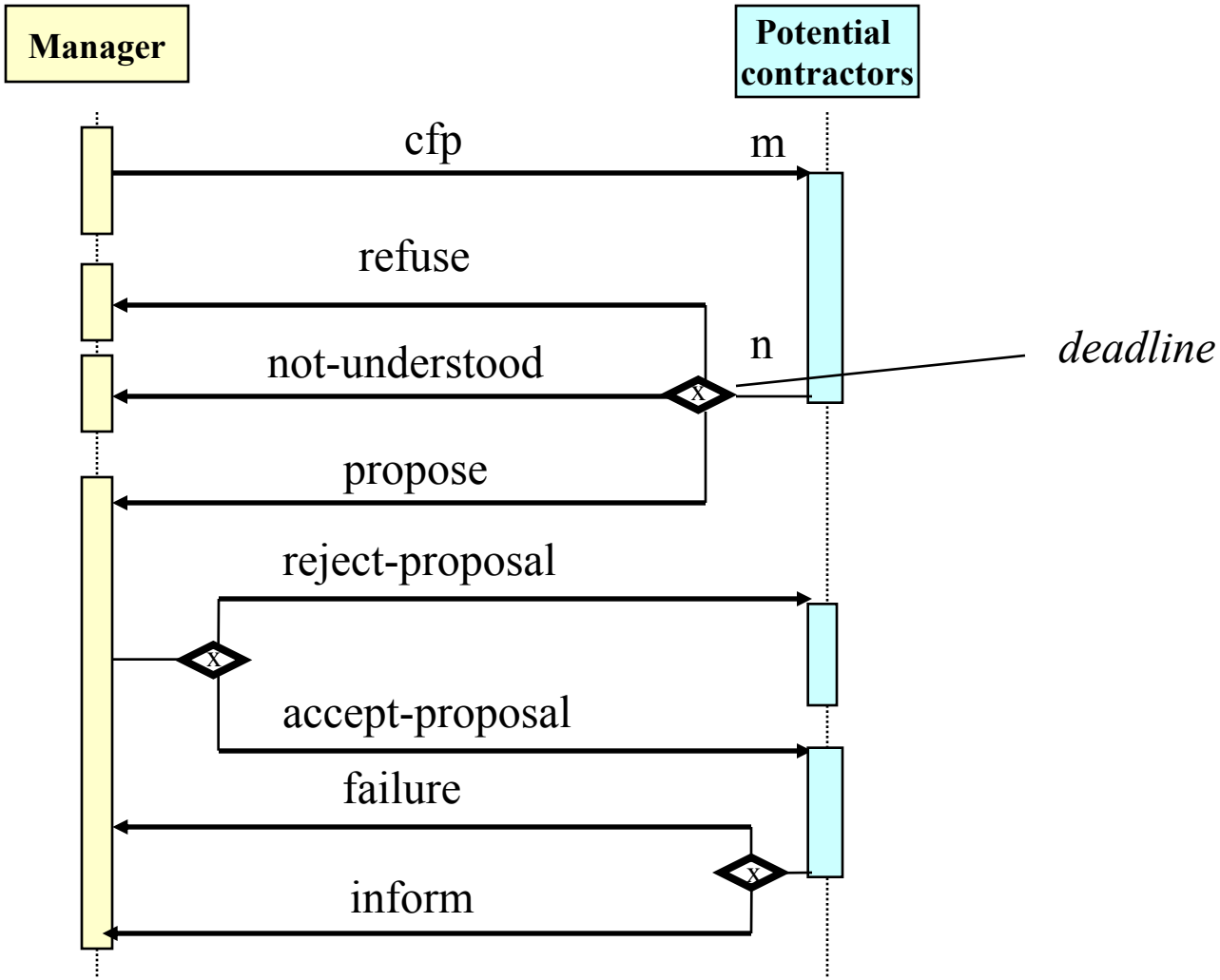
FIPA protocols

- **FIPA-request Protocol**
- **FIPA-query Protocol**
- **FIPA-request-when Protocol**
- **FIPA-contract-net Protocol**
- **FIPA-Iterated-Contract-Net Protocol**
- **FIPA-Auction-English Protocol**
- **FIPA-Auction-Dutch Protocol**
- …

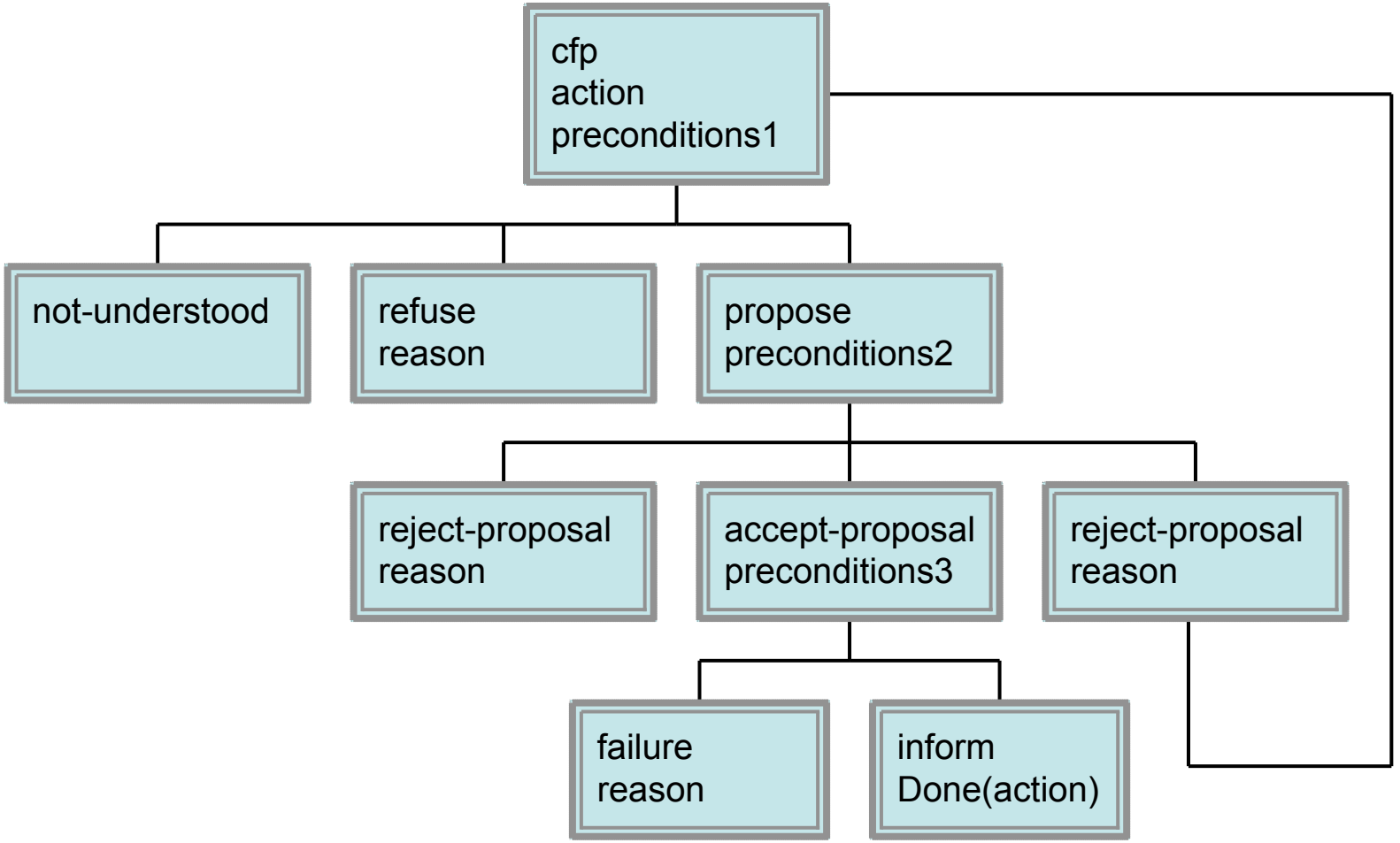
FIPA Contract Net Protocol 2



FIPA Contract Net Protocol 3



FIPA-iterated-contract-net protocol



Comparing KQML and FIPA ACL

- Similarities:
  - Separation of the outer language (performative) and the inner language (content).
  - Allows for any content language
  - Speech act based
- Differences
  - Communication primitives:
    - KQML – performative
    - FIPA ACL – communicative act
  - Different semantic frameworks – impossible to come up with an exact mapping or transformation between KQML and FIPA performatives.
  - FIPA has more formal basis and has means for describing interaction protocols
  - KQML provides facilitator services; FIPA ACL does not.

FIPA pretends to be standard for agent communication

Summary

- Speech Acts is a basis for ACLs
- Content and Ontologies
- Basic Agent Communication Languages
  - KQML
  - FIPA ACL

Next Lecture: Agent Coordination

Will be based on Text book:

Chapter 8