Multi-agent Semantic Web Systems:
Agent Communication Overview

Jacques Fleuriot
1 Agents Overview

2 Agent Architectures
   - Middle Agents
   - Network Architectures

3 Agent Communication

4 Interaction Models

5 Summary
What are Agents?

- ‘Intelligent’ characteristics of agents:
  - autonomy
  - reasoning ability
  - learning ability
  - mobility
  - sociability
  - cooperation
  - negotiation

Unlike standard WS, an agent can reason about:

- how to handle external requests,
- what order to carry them out.
What are Agents?

- ‘Intelligent’ characteristics of agents:
  - autonomy
What are Agents?

- ‘Intelligent’ characteristics of agents:
  - autonomy
  - reasoning ability
What are Agents?

- ‘Intelligent’ characteristics of agents:
  - autonomy
  - reasoning ability
  - learning ability
‘Intelligent’ characteristics of agents:

- autonomy
- reasoning ability
- learning ability
- mobility
What are Agents?

‘Intelligent’ characteristics of agents:

- autonomy
- reasoning ability
- learning ability
- mobility
- sociability
What are Agents?

‘Intelligent’ characteristics of agents:

- autonomy
- reasoning ability
- learning ability
- mobility
- sociability
- cooperation
What are Agents?

‘Intelligent’ characteristics of agents:

- autonomy
- reasoning ability
- learning ability
- mobility
- sociability
- cooperation
- negotiation
What are Agents?

- ‘Intelligent’ characteristics of agents:
  - autonomy
  - reasoning ability
  - learning ability
  - mobility
  - sociability
  - cooperation
  - negotiation

- Unlike standard WS, an agent can *reason* about:
What are Agents?

- ‘Intelligent’ characteristics of agents:
  - autonomy
  - reasoning ability
  - learning ability
  - mobility
  - sociability
  - cooperation
  - negotiation

- Unlike standard WS, an agent can reason about:
  - how to handle external requests,
What are Agents?

- ‘Intelligent’ characteristics of agents:
  - autonomy
  - reasoning ability
  - learning ability
  - mobility
  - sociability
  - cooperation
  - negotiation

- Unlike standard WS, an agent can reason about:
  - how to handle external requests,
  - what order to carry them out.
Multi-Agents Systems (MAS)

- Distributed system which incorporates independent agents.
Multi-Agents Systems (MAS)

- Distributed system which incorporates independent agents.
- The collection of agents *interact*, and
Multi-Agents Systems (MAS)

- Distributed system which incorporates independent agents.
- The collection of agents interact, and
- solve problems that are outside their individual capacities.
Multi-Agents Systems (MAS)

- Distributed system which incorporates independent agents.
- The collection of agents interact, and
- solve problems that are outside their individual capacities.
- Focus on properties that emerge from cooperation (vs. capabilities of individual agents)
Multi-Agents Systems (MAS)

- Distributed system which incorporates independent agents.
- The collection of agents interact, and
- solve problems that are outside their individual capacities.
- Focus on properties that emerge from cooperation (vs. capabilities of individual agents)
- ‘standard’ distributed system: coordination must be specified in advance
Multi-Agents Systems (MAS)

- Distributed system which incorporates independent agents.
- The collection of agents interact, and
- solve problems that are outside their individual capacities.
- Focus on properties that emerge from cooperation (vs. capabilities of individual agents)
- ‘standard’ distributed system: coordination must be specified in advance
- Multi-Agent system: (some aspects of) coordination achieved dynamically at run-time
Multi-Agents Systems (MAS)

- Distributed system which incorporates independent agents.
- The collection of agents interact, and
- solve problems that are outside their individual capacities.
- Focus on properties that emerge from cooperation (vs. capabilities of individual agents)
- ‘standard’ distributed system: coordination must be specified in advance
- Multi-Agent system: (some aspects of) coordination achieved dynamically at run-time
- Seen as complementary to existing Semantic Web/Grid technologies.
Coordination Problem

- Simplistic WS Model:
Coordination Problem

- Simplistic WS Model:
  - “one-shot” interactions:
Coordination Problem

- Simplistic WS Model:
  - “one-shot” interactions:
  - client sends a request message to a single service operation and receives a response message.
Coordination Problem

- Simplistic WS Model:
  - “one-shot” interactions:
  - client sends a request message to a single service operation and receives a response message.

- In practice, we want to allow more complex kinds of interaction:
Coordination Problem

- Simplistic WS Model:
  - “one-shot” interactions:
  - client sends a request message to a single service operation and receives a response message.

- In practice, we want to allow more complex kinds of interaction:
  - multiple operations,
Coordination Problem

- Simplistic WS Model:
  - “one-shot” interactions:
    - client sends a request message to a single service operation and receives a response message.
  - In practice, we want to allow more complex kinds of interaction:
    - multiple operations,
    - multiple messages exchanged,
Coordination Problem

- Simplistic WS Model:
  - “one-shot” interactions:
    - client sends a request message to a single service operation and receives a response message.
  - In practice, we want to allow more complex kinds of interaction:
    - multiple operations,
    - multiple messages exchanged,
    - messages sequenced in a particular order,
Coordination Problem

- Simplistic WS Model:
  - “one-shot” interactions:
    - client sends a request message to a single service operation and receives a response message.
  - In practice, we want to allow more complex kinds of interaction:
    - multiple operations,
    - multiple messages exchanged,
    - messages sequenced in a particular order,
    - multiple parties involved in the interaction.
Coordination Problem

- Simplistic WS Model:
  - “one-shot” interactions:
  - client sends a request message to a single service operation and receives a response message.

- In practice, we want to allow more complex kinds of interaction:
  - multiple operations,
  - multiple messages exchanged,
  - messages sequenced in a particular order,
  - multiple parties involved in the interaction.

- How do we ensure that such interactions are
Coordination Problem

- Simplistic WS Model:
  - “one-shot” interactions:
    - client sends a request message to a single service operation and receives a response message.
  - In practice, we want to allow more complex kinds of interaction:
    - multiple operations,
    - multiple messages exchanged,
    - messages sequenced in a particular order,
    - multiple parties involved in the interaction.
  - How do we ensure that such interactions are coordinated?
Coordination Problem

- Simplistic WS Model:
  - “one-shot” interactions:
  - client sends a request message to a single service operation and receives a response message.

- In practice, we want to allow more complex kinds of interaction:
  - multiple operations,
  - multiple messages exchanged,
  - messages sequenced in a particular order,
  - multiple parties involved in the interaction.

- How do we ensure that such interactions are
  - coordinated?
  - correct?
Coordination Problem

- Simplistic WS Model:
  - “one-shot” interactions:
    - client sends a request message to a single service operation and receives a response message.
  - In practice, we want to allow more complex kinds of interaction:
    - multiple operations,
    - multiple messages exchanged,
    - messages sequenced in a particular order,
    - multiple parties involved in the interaction.

- How do we ensure that such interactions are
  - coordinated?
  - correct?
  - robust to failures?
Interaction Example

Outline
- Agents Overview
- Agent Architectures
- Middle Agents
- Network Architectures
- Agent Communication
- Interaction Models
- Summary
Types of Middle Agent

- Middle-agents:
  - Matchmaker (yellow pages agents): receive advertisements and match with requests.
  - Broker: like matchmaker, but also processes the requests.
Types of Middle Agent

- **Middle-agents:**
  - assist in locating service providers, and
Types of Middle Agent

- **Middle-agents:**
  - assist in locating service providers, and
  - connecting service providers with service requesters.
Types of Middle Agent

- **Middle-agents:**
  - assist in locating service providers, and
  - connecting service providers with service requesters.

- Two important types of middle-agent
  ([http://www.cs.cmu.edu/~softagents/middle.html](http://www.cs.cmu.edu/~softagents/middle.html))
Types of Middle Agent

- **Middle-agents:**
  - assist in locating service providers, and
  - connecting service providers with service requesters.

- **Two important types of middle-agent**
  ([http://www.cs.cmu.edu/~softagents/middle.html](http://www.cs.cmu.edu/~softagents/middle.html))

  **Matchmaker (yellow pages agents):** receives
  advertisements and matches with requests.
Types of Middle Agent

- **Middle-agents:**
  - assist in locating service providers, and
  - connecting service providers with service requesters.

- **Two important types of middle-agent**
  
  **Matchmaker (yellow pages agents):** receives advertisements and matches with requests.
  
  **Broker:** like matchmaker, but also processes the requests.

Two important types of middle-agent
(http://www.cs.cmu.edu/~softagents/middle.html)
Service Matchmaking
Service Matchmaking

Requestor → Matchmaker → Provider
- Request for Service
- Advertise Service
- Reply Provider Agents' Names
- Reply Result of Service
- Request for Service
Service Brokering

Outline

Agents Overview
Agent Architectures
Middle Agents
Network Architectures
Agent Communication
Interaction Models
Summary
So far, mainly assumed some kind of centralized client/server architecture.
So far, mainly assumed some kind of centralized client/server architecture.

But Service Oriented Architectures can equally well be decentralized
Centralized: Client/Server
Decentralized: Peer-to-Peer
Peer-to-Peer Architectures

P2P is...

a self-organizing system of equal, autonomous entities (peers) [which] aims for the shared usage of distributed resources in a networked environment avoiding central services.
Peer-to-Peer Architectures

P2P is...

a self-organizing system of equal, autonomous entities (peers) [which] aims for the shared usage of distributed resources in a networked environment avoiding central services.

- Peers interact directly with each other, usually without central coordination.
Peer-to-Peer Architectures

P2P is...

a self-organizing system of equal, autonomous entities (peers) [which] aims for the shared usage of distributed resources in a networked environment avoiding central services.

- Peers interact directly with each other, usually without central coordination.
- Each peer has autonomy over its own resources.
Peer-to-Peer Architectures

P2P is...

- a self-organizing system of equal, autonomous entities (peers) [which] aims for the shared usage of distributed resources in a networked environment avoiding central services.

- Peers interact directly with each other, usually without central coordination.

- Each peer has autonomy over its own resources.

- Within a set of peers, each uses resources provided by other peers.
Peer-to-Peer Architectures

P2P is...

a self-organizing system of equal, autonomous entities (peers) [which] aims for the shared usage of distributed resources in a networked environment avoiding central services.

- Peers interact directly with each other, usually without central coordination.
- Each peer has autonomy over its own resources.
- Within a set of peers, each uses resources provided by other peers.
- Peers can act as both clients and servers; i.e., no intrinsic asymmetry of role.
Peer-to-Peer Architectures

P2P is...

a self-organizing system of equal, autonomous entities (peers) [which] aims for the shared usage of distributed resources in a networked environment avoiding central services.

- Peers interact directly with each other, usually without central coordination.
- Each peer has autonomy over its own resources.
- Within a set of peers, each uses resources provided by other peers.
- Peers can act as both clients and servers; i.e., no intrinsic asymmetry of role.
- Performance considerations may dictate some centralized elements in P2P systems — leads to hybrid P2P systems.
Hybrid: Peers and Super-Peers
Agent Architectures

MAS can have any of the above architectures.
Agent Architectures

- MAS can have any of the above architectures.
- Middle-agent will play role of coordinator in a centralized architecture.
Agent Architectures

- MAS can have any of the above architectures.
- Middle-agent will play role of coordinator in a centralized architecture.
- Increasing interest in achieving coordination in pure P2P systems.
Communication involves sharing at various levels, including
Communication involves sharing at various levels, including
- how to describe states of affairs (vocabulary, grammar),
Communication involves sharing at various levels, including

- how to describe states of affairs (vocabulary, grammar),
- how to carry out speech acts (performative verbs),
Communication involves sharing at various levels, including
- how to describe states of affairs (vocabulary, grammar),
- how to carry out speech acts (performative verbs),
- how to engage in dialogue.
DARPA Knowledge Sharing Effort (KSE)

DARPA Knowledge Sharing Effort (KSE)

- Project from early ’90s aimed at developing techniques, tools and re-usable resources to support building large-scale knowledge-based systems and knowledge bases.
DARPA Knowledge Sharing Effort (KSE)

- Project from early ’90s aimed at developing techniques, tools and re-usable resources to support building large-scale knowledge-based systems and knowledge bases.
- Results in the area of knowledge representation:
DARPA Knowledge Sharing Effort (KSE)

- **http://www-ksl.stanford.edu/knowledge-sharing/**
- Project from early ’90s aimed at developing techniques, tools and re-usable resources to support building large-scale knowledge-based systems and knowledge bases.
- Results in the area of knowledge representation:
  
  **Syntax** KIF — Knowledge Interchange Format
DARPA Knowledge Sharing Effort (KSE)

- Project from early ’90s aimed at developing techniques, tools and re-usable resources to support building large-scale knowledge-based systems and knowledge bases.
- Results in the area of knowledge representation:
  - **Syntax** KIF — Knowledge Interchange Format
  - **Semantics** Ontolingua — language for defining sharable ontologies
DARPA Knowledge Sharing Effort (KSE)

- Project from early ’90s aimed at developing techniques, tools and re-usable resources to support building large-scale knowledge-based systems and knowledge bases.
- Results in the area of knowledge representation:
  - Syntax KIF — Knowledge Interchange Format
  - Semantics Ontolingua — language for defining sharable ontologies
  - Pragmatics KQML — high-level interaction language
KIF: First order logic with set theory.
Knowledge Interchange Format

- **KIF**: First order logic with set theory.
- Intended as interlingua for encoding declarative knowledge.
Knowledge Interchange Format

- **KIF**: First order logic with set theory.
- Intended as interlingua for encoding declarative knowledge.
- **Features:**
  - Semantics independent of implementation
  - Expressive
  - Non-decidable
  - Human readable

Current specification at [http://logic.stanford.edu/kif](http://logic.stanford.edu/kif)
Knowledge Interchange Format

- KIF: First order logic with set theory.
- Intended as interlingua for encoding declarative knowledge.
- Features:
  - semantics independent of implementation
Knowledge Interchange Format

- **KIF**: First order logic with set theory.
- Intended as interlingua for encoding declarative knowledge.
- Features:
  - semantics independent of implementation
  - expressive

Knowledge Interchange Format (KIF) is a format intended as an interlingua for encoding declarative knowledge. It is based on first-order logic with set theory and is designed to be semantically independent of implementation, making it expressive and human-readable. The current specification can be found at [http://logic.stanford.edu/kif](http://logic.stanford.edu/kif).
Knowledge Interchange Format

- **KIF**: First order logic with set theory.
- Intended as interlingua for encoding declarative knowledge.
- **Features**:
  - semantics independent of implementation
  - expressive
  - non-decidable

KIF: First order logic with set theory. Intended as interlingua for encoding declarative knowledge. Features:
- semantics independent of implementation
- expressive
- non-decidable
Knowledge Interchange Format

- KIF: First order logic with set theory.
- Intended as interlingua for encoding declarative knowledge.
- Features:
  - semantics independent of implementation
  - expressive
  - non-decidable
  - human readable
Knowledge Interchange Format

- KIF: First order logic with set theory.
- Intended as interlingua for encoding declarative knowledge.
- Features:
  - semantics independent of implementation
  - expressive
  - non-decidable
  - human readable
- Current specification at
  http://logic.stanford.edu/kif
- Knowledge Query and Manipulation Language (KQML).
Knowledge Query and Manipulation Language (KQML).
High-level, message-oriented language for information exchange.
KQML

- Knowledge Query and Manipulation Language (KQML).
- High-level, message-oriented language for information exchange.
- Makes no assumptions about:
KQML

- Knowledge Query and Manipulation Language (KQML).
- High-level, message-oriented language for information exchange.
- Makes no assumptions about:
  - transport mechanism;
KQML

- Knowledge Query and Manipulation Language (KQML).
- High-level, message-oriented language for information exchange.
- Makes no assumptions about:
  - transport mechanism;
  - content language;
Knowledge Query and Manipulation Language (KQML).
- High-level, message-oriented language for information exchange.
- Makes no assumptions about:
  - transport mechanism;
  - content language;
  - ontology used in message content.
KQML

- Knowledge Query and Manipulation Language (KQML).
- High-level, message-oriented language for information exchange.
- Makes no assumptions about:
  - transport mechanism;
  - content language;
  - ontology used in message content.
- Message types particularly oriented towards multi-agent communication.
Example Message

(tell :sender    amazon.com
     :receiver   info-agent0011
     :in-reply-to msid-7.24.97.45391
     :ontology  http://amazon.com/Books
     :language  prolog
     :content   ‘‘price(item567, gbp, 24.95)’’
)

performative parameter (an attribute/value pair)
declarative message content
Example Message

(tell :sender amazon.com
     :receiver info-agent0011
     :in-reply-to msid-7.24.97.45391
     :ontology http://amazon.com/Books
     :language prolog
     :content ‘‘price(item567, gbp, 24.95)’’
)

- performative
Example Message

(tell :sender amazon.com
    :receiver info-agent0011
    :in-reply-to msid-7.24.97.45391
    :ontology http://amazon.com/Books
    :language prolog
    :content ‘‘price(item567, gbp, 24.95)’’
)

- parameter (an attribute/value pair)
KQML Message

Example Message

(tell :sender amazon.com
    :receiver info-agt0011
    :in-reply-to msid-7.24.97.45391
    :ontology http://amazon.com/Books
    :language prolog
    :content ‘‘price(item567, gbp, 24.95)’’
)

■ declarative message content
FIPA (Foundation for Intelligent Physical Agents; http://www.fipa.org/)
- FIPA (Foundation for Intelligent Physical Agents; http://www.fipa.org/)

- Standards organization that promotes agent-based technology, based in Geneva.
FIPA

- FIPA (Foundation for Intelligent Physical Agents; http://www.fipa.org/)

- Standards organization that promotes agent-based technology, based in Geneva.

FIPA (Foundation for Intelligent Physical Agents; http://www.fipa.org/)

- Standards organization that promotes agent-based technology, based in Geneva.
- Has specifications in the areas of:
- FIPA (Foundation for Intelligent Physical Agents; http://www.fipa.org/)
- Standards organization that promotes agent-based technology, based in Geneva.
- Has specifications in the areas of;
  - agent applications
FIPA

- FIPA (Foundation for Intelligent Physical Agents; http://www.fipa.org/)
- Standards organization that promotes agent-based technology, based in Geneva.
- Has specifications in the areas of:
  - agent applications
  - abstract architecture
FIPA (Foundation for Intelligent Physical Agents; http://www.fipa.org/)

- Standards organization that promotes agent-based technology, based in Geneva.
- Has specifications in the areas of:
  - agent applications
  - abstract architecture
  - agent communication
FIPA

- FIPA (Foundation for Intelligent Physical Agents; http://www.fipa.org/)
- Standards organization that promotes agent-based technology, based in Geneva.
- Has specifications in the areas of:
  - agent applications
  - abstract architecture
  - agent communication
  - agent management
FIPA

- FIPA (Foundation for Intelligent Physical Agents; http://www.fipa.org/)
- Standards organization that promotes agent-based technology, based in Geneva.
- Has specifications in the areas of:
  - agent applications
  - abstract architecture
  - agent communication
  - agent management
  - agent message transport
FIPA

- FIPA (Foundation for Intelligent Physical Agents; http://www.fipa.org/)
- Standards organization that promotes agent-based technology, based in Geneva.
- Has specifications in the areas of:
  - agent applications
  - abstract architecture
  - agent communication
  - agent management
  - agent message transport
FIPA ACL (Agent Communication Language): a specification for inter-agent communication via message passing.
FIPA ACL

- FIPA ACL (Agent Communication Language): a specification for inter-agent communication via message passing.
  - Assumes that agents have Beliefs, Desires and Intentions (BDI model)
FIPA ACL

- FIPA ACL (Agent Communication Language): a specification for inter-agent communication via message passing.
  - Assumes that agents have Beliefs, Desires and Intentions (BDI model)
  - FIPA ACL is similar in design to KQML, but gives an explicit BDI-based semantics to the performatives.
FIPA ACL

- FIPA ACL (Agent Communication Language): a specification for inter-agent communication via message passing.
  - Assumes that agents have Beliefs, Desires and Intentions (BDI model)
  - FIPA ACL is similar in design to KQML, but gives an explicit BDI-based semantics to the performatives.

Semantics of inform

“S informs R that P” requires that:
FIPA ACL

- FIPA ACL (Agent Communication Language): a specification for inter-agent communication via message passing.
  - Assumes that agents have Beliefs, Desires and Intentions (BDI model)
  - FIPA ACL is similar in design to KQML, but gives an explicit BDI-based semantics to the performatives.

Semantics of inform

“S informs R that P” requires that:

- S believes that some proposition P is true,
FIPA ACL

- FIPA ACL (Agent Communication Language): a specification for inter-agent communication via message passing.
  - Assumes that agents have Beliefs, Desires and Intentions (BDI model)
  - FIPA ACL is similar in design to KQML, but gives an explicit BDI-based semantics to the performatives.

**Semantics of inform**

“S informs R that P” requires that:

- **S believes** that some proposition P is true,
- **S intends** that R also comes to **believe** that P is true, and
FIPA ACL

- FIPA ACL (Agent Communication Language): a specification for inter-agent communication via message passing.
  - Assumes that agents have Beliefs, Desires and Intentions (BDI model)
  - FIPA ACL is similar in design to KQML, but gives an explicit BDI-based semantics to the performatives.

Semantics of inform

“S informs R that P” requires that:
- S believes that some proposition P is true,
- S intends that R also comes to believe that P is true, and
- S does not already believe that R has any knowledge of the truth of P.
Protocols in FIPA ACL

- Notion of ‘protocol’ refers to stereotyped pattern of conversation between agents.
Protocols in FIPA ACL

- Notion of ‘protocol’ refers to stereotyped pattern of conversation between agents.
- Available protocols are usually pre-specified by the agent designer.
Protocols in FIPA ACL

- Notion of ‘protocol’ refers to stereotyped pattern of conversation between agents.
- Available protocols are usually pre-specified by the agent designer.
- Agents somehow need to discover which protocol to follow.
 Protocols in FIPA ACL

- Notion of ‘protocol’ refers to stereotyped pattern of conversation between agents.
- Available protocols are usually **pre-specified** by the agent designer.
- Agents somehow need to discover which protocol to follow.
- Choice of protocol to follow could be negotiated,
Notion of ‘protocol’ refers to stereotyped pattern of conversation between agents.

Available protocols are usually pre-specified by the agent designer.

Agents somehow need to discover which protocol to follow.

Choice of protocol to follow could be negotiated,

but in FIPA ACL, convention is to place name of the protocol in the :protocol parameter of the message.
Multi-agent Semantic Web Systems: Agent Communication Overview

Jacques Fleuriot

Outline
Agents Overview
Agent Architectures
Middle Agents Network Architectures
Agent Communication
Interaction Models
Summary

FIPA-Query-Protocol

![Diagram of FIPA-Query-Protocol]

- Initiator
- Participant

Transactions:
- query-if
- query-ref
- refuse
- agree
- [agreed]
- failure
- inform-t/f
- [query-if]
- inform-result
- [query-ref]
Explanation of FIPA-Query-Protocol

- Initiator requests Participant to perform a **inform** action:
Initiator requests Participant to perform a `inform` action:
- `query-if`: whether proposition $P$ is true or false
Initiator requests Participant to perform a **inform** action:

- **query-if**: whether proposition $P$ is true or false
- **query-ref**: query about specified objects
Explanation of FIPA-Query-Protocol

- Initiator requests Participant to perform a *inform* action:
  - *query-if*: whether proposition $P$ is true or false
  - *query-ref*: query about specified objects
- Participant processes request and decides whether to accept or refuse.
Explanation of FIPA-Query-Protocol

- Initiator requests Participant to perform a `inform` action:
  - `query-if`: whether proposition $P$ is true or false
  - `query-ref`: query about specified objects
- Participant processes request and decides whether to accept or refuse.
- If decides to accept, “agreed” becomes true.
Initiator requests Participant to perform a inform action:
- **query-if**: whether proposition $P$ is true or false
- **query-ref**: query about specified objects

Participant processes request and decides whether to accept or refuse.

If decides to accept, “agreed” becomes true.

Participant uses inform-$t/f$ to assert whether $P$ is true or false;
Explanation of FIPA-Query-Protocol

- Initiator requests Participant to perform a `inform` action:
  - `query-if`: whether proposition $P$ is true or false
  - `query-ref`: query about specified objects
- Participant processes request and decides whether to accept or refuse.
- If decides to accept, “agreed” becomes true.
- Participant uses `inform-t/f` to assert whether $P$ is true or false;
- uses `inform-result` to refer to object that was queried about.
Protocols are often implemented as Finite State Machines.
Protocols are often implemented as Finite State Machines. Each state represents a stage in the conversation — sometimes thought of as an information state.
Protocols as FSMs

- Protocols are often implemented as Finite State Machines.
- Each state represents a stage in the conversation — sometimes thought of as an information state.
- Arcs represent the exchanged messages that allow transition to a new state.
Protocols are often implemented as Finite State Machines.

Each state represents a stage in the conversation — sometimes thought of as an information state.

Arcs represent the exchanged messages that allow transition to a new state.

Valid messages depend on the current state of the conversation — represented by outgoing arcs.
Protocols as FSMs

- Protocols are often implemented as Finite State Machines.
- Each state represents a stage in the conversation — sometimes thought of as an information state.
- Arcs represent the exchanged messages that allow transition to a new state.
- Valid messages depend on the current state of the conversation — represented by outgoing arcs.
- Final states represent completed conversations.
Example FSM for a Conversation

Multi-agent Semantic Web Systems: Agent Communication Overview

Outline
- Agents Overview
- Agent Architectures
  - Middle Agents
  - Network Architectures
- Agent Communication
- Interaction Models
- Summary
Protocols give us a way of specifying a class of legal interactions between agents.
Agent Roles

- Protocols give us a way of specifying a class of legal interactions between agents.
- However, we often want to have higher-level ways of describing agent behaviour.
Agent Roles

- Protocols give us a way of specifying a class of legal interactions between agents.
- However, we often want to have higher-level ways of describing agent behaviour.
- Key notion: role that is assigned to an agent.
Agent Roles

- Protocols give us a way of specifying a class of legal interactions between agents.
- However, we often want to have higher-level ways of describing agent behaviour.
- Key notion: role that is assigned to an agent.
- Roles determine rights, duties and opportunities.
Protocols give us a way of specifying a class of legal interactions between agents.

However, we often want to have higher-level ways of describing agent behaviour.

Key notion: role that is assigned to an agent.

Roles determine rights, duties and opportunities.

The role assumed by an agent limits its possible actions.
Agent Roles

- Protocols give us a way of specifying a class of legal interactions between agents.
- However, we often want to have higher-level ways of describing agent behaviour.
- Key notion: role that is assigned to an agent.
- Roles determine rights, duties and opportunities.
- The role assumed by an agent limits its possible actions.
- Example roles in interaction: seller, buyer, auctioneer
Protocols give us a way of specifying a class of legal interactions between agents.

However, we often want to have higher-level ways of describing agent behaviour.

Key notion: role that is assigned to an agent.

Roles determine rights, duties and opportunities.

The role assumed by an agent limits its possible actions.

Example roles in interaction: seller, buyer, auctioneer

- Seller must own goods before submitting them for sale.
Agent Roles

- Protocols give us a way of specifying a class of legal interactions between agents.
- However, we often want to have higher-level ways of describing agent behaviour.
- Key notion: **role** that is assigned to an agent.
- Roles determine rights, duties and opportunities.
- The role assumed by an agent limits its possible actions.
- Example roles in interaction: seller, buyer, auctioneer
  - Seller must own goods before submitting them for sale.
  - Buyer may submit bids if credit standing is good.
Agent Roles

- Protocols give us a way of specifying a class of legal interactions between agents.
- However, we often want to have higher-level ways of describing agent behaviour.
- Key notion: role that is assigned to an agent.
- Roles determine rights, duties and opportunities.
- The role assumed by an agent limits its possible actions.
- Example roles in interaction: seller, buyer, auctioneer
  - Seller must own goods before submitting them for sale.
  - Buyer may submit bids if credit standing is good.
  - Auctioneer may offer goods and accept bids.
- **Policies** provide high-level description of acceptable behaviour (or social norms).
Policies

- **Policies** provide high-level description of acceptable behaviour (or social norms).
- Often used for security.
Policies

- Policies provide high-level description of acceptable behaviour (or social norms).
- Often used for security.
- Conversation policies: do not specify types of messages that can be exchanged.
Policies

- **Policies** provide high-level description of acceptable behaviour (or social norms).
- Often used for security.
- Conversation policies: do **not** specify types of messages that can be exchanged.
- Rather, conversation policies place constraints on
- **Policies** provide high-level description of acceptable behaviour (or social norms).
- Often used for security.
- Conversation policies: do **not** specify types of messages that can be exchanged.
- Rather, conversation policies place constraints on who can participate in interaction;
Policies

- **Policies** provide high-level description of acceptable behaviour (or social norms).
- Often used for security.
- Conversation policies: do not specify types of messages that can be exchanged.
- Rather, conversation policies place constraints on
  - who can participate in interaction;
  - under what circumstances;
Policies

- Policies provide high-level description of acceptable behaviour (or social norms).
- Often used for security.
- Conversation policies: do not specify types of messages that can be exchanged.
- Rather, conversation policies place constraints on
  - who can participate in interaction;
  - under what circumstances;
  - whether sub-conversations can be initiated, etc.
Policies

- Policies provide high-level description of acceptable behaviour (or social norms).
- Often used for security.
- Conversation policies: do not specify types of messages that can be exchanged.
- Rather, conversation policies place constraints on
  - who can participate in interaction;
  - under what circumstances;
  - whether sub-conversations can be initiated, etc.
- Policy language: means of stating what agent can/cannot and should/shouldn’t do.
Policies

- **Policies** provide high-level description of acceptable behaviour (or social norms).
- Often used for security.
- Conversation policies: do not specify types of messages that can be exchanged.
- Rather, conversation policies place constraints on
  - who can participate in interaction;
  - under what circumstances;
  - whether sub-conversations can be initiated, etc.
- Policy language: means of stating what agent can/cannot and should/shouldn’t do.

**Policy Example (Phillips & Link)**

Agents $A$ and $B$ are discussing arrangements for a party for $C$. Do they stop discussion when $C$ enters the room? This is a matter of policy.
Electronic Institutions, 1

- Agent counterpart of human organizations.
Electronic Institutions, 1

- Agent counterpart of human organizations.
- Specifies norms and rules to govern interaction.
Agent counterpart of human organizations.

- Specifies norms and rules to govern interaction.
- Conversation protocols are grouped into scenes.
Electronic Institutions, 1

- Agent counterpart of human organizations.
- Specifies norms and rules to govern interaction.
- Conversation protocols are grouped into scenes.
- Agents participate in scenes by virtue of a role — can play different roles in different scenes.
Electronic Institutions, 1

- Agent counterpart of human organizations.
- Specifies norms and rules to govern interaction.
- Conversation protocols are grouped into scenes.
- Agents participate in scenes by virtue of a role — can play different roles in different scenes.
- Example scenes (for auction):
Electronic Institutions, 1

- Agent counterpart of human organizations.
- Specifies norms and rules to govern interaction.
- Conversation protocols are grouped into scenes.
- Agents participate in scenes by virtue of a role — can play different roles in different scenes.
- Example scenes (for auction):
  - admit buyers
Electronic Institutions, 1

- Agent counterpart of human organizations.
- Specifies norms and rules to govern interaction.
- Conversation protocols are grouped into scenes.
- Agents participate in scenes by virtue of a role — can play different roles in different scenes.
- Example scenes (for auction):
  - admit buyers
  - admit sellers
Electronic Institutions, 1

- Agent counterpart of human organizations.
- Specifies norms and rules to govern interaction.
- Conversation protocols are grouped into **scenes**.
- Agents participate in scenes by virtue of a role — can play different roles in different scenes.
- Example scenes (for auction):
  - admit buyers
  - admit sellers
  - carry out auction
Electronic Institutions, 1

- Agent counterpart of human organizations.
- Specifies norms and rules to govern interaction.
- Conversation protocols are grouped into scenes.
- Agents participate in scenes by virtue of a role — can play different roles in different scenes.
- Example scenes (for auction):
  - admit buyers
  - admit sellers
  - carry out auction
  - settlement (i.e., paying for goods)
- Scenes are connected into a performative structure;
Scenes are connected into a performative structure;
latter governs how agents can move from one scene to another.
Electronic Institutions, 2

- Scenes are connected into a performative structure;
- latter governs how agents can move from one scene to another.
  - E.g., admit buyer precedes auction, auction precedes settlement
Electronic Institutions, 2

- Scenes are connected into a performative structure;
- latter governs how agents can move from one scene to another.
  - E.g., admit buyer precedes auction, auction precedes settlement
- Norms govern transitions between scenes.
Scenes are connected into a performative structure;
latter governs how agents can move from one scene to another.

- E.g., admit buyer precedes auction, auction precedes settlement

- Norms govern transitions between scenes.
  - E.g., a buyer agent that wins a bid is obliged to pay for the good.
Agent communication requires sharing at a number of levels:
Agent communication requires sharing at a number of levels:

- language for expressing propositional knowledge,
Agent communication requires sharing at a number of levels:
- language for expressing propositional knowledge,
- language for representing speech acts (performatives),
Summary

Agent communication requires sharing at a number of levels:

- language for expressing propositional knowledge,
- language for representing speech acts (performatives),
- protocols which determine possible messages and their sequencing.
Agent communication requires sharing at a number of levels:

- language for expressing propositional knowledge,
- language for representing speech acts (performatives),
- protocols which determine possible messages and their sequencing.

Higher level structures govern more abstract aspects of interaction.
Agent communication requires sharing at a number of levels:
- language for expressing propositional knowledge,
- language for representing speech acts (performatives),
- protocols which determine possible messages and their sequencing.

Higher level structures govern more abstract aspects of interaction.

Policy languages and electronic institutions: two ways of representing rights and obligations of agents.
Reading

- Wooldridge, esp Chaps 1, 2, 8.
Reading

- Wooldridge, esp Chaps 1, 2, 8.
- Passin, Chap 9.