LECTURE 5: Ontologies & Communication

Introduction to Multi-Agent Systems (MESIIA, MIA)

URV

Social Behaviour

- Previously we looked at:
  - Deductive Agents
  - Practical Reasoning Agents
  - Reactive and Hybrid Agents

- We said:
  - An intelligent agent is a computer system capable of *flexible* autonomous action in some environment.
  - Where by flexible, we mean:
    - Reactive
    - Pro-active
    - Social

- This is where we deal with the "*social*" bit, showing how agents communicate and share information.

Agent Communication

- In this lecture, we cover macro-aspects of intelligent agent technology, and those issues relating to the agent society, rather than the individual:
    - Communication:
        - speech acts; KQML & KIF; FIPA ACL
    - Ontologies:
        - the role of ontologies in communication
        - aligning ontologies
        - OWL
- There are some limited things that one can do without communication, but they are..., well..., *limited*!!!
    - Most work on multiagent systems assumes communication.
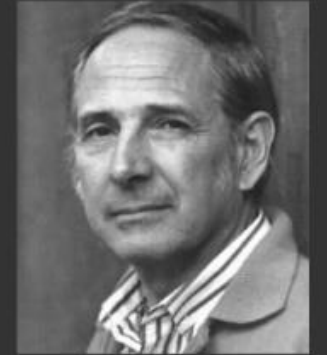
Speech Acts

- Austin's 1962 book "How to Do Things with Words" is usually taken to be the origin of speech acts

- Speech act theories are *pragmatic* theories of language, that is theories of how language *use*:
  - they attempt to account for how language is used by people every day to achieve their goals and intentions.

- Most treatments of communication in (multi-)agent systems borrow their inspiration from speech act theory…
  - …doubtless because the "action" part can be tied closely to existing ideas about how to model action.

- Austin noticed that some utterances are rather like 'physical actions' that appear to *change the state of the world*.
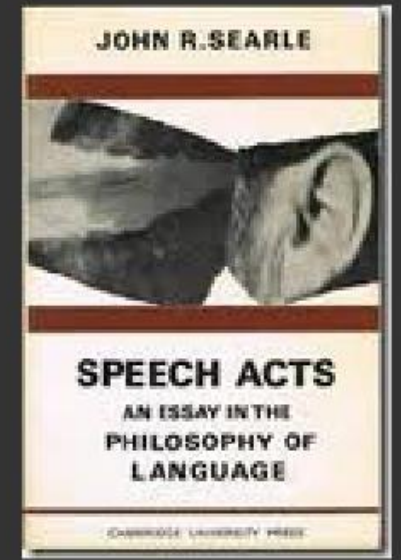
Speech Acts

- Paradigm examples are:
  - Declaring war ☹;
  - Naming a child;
  - "I now pronounce you man and wife" ☺

- But more generally, everything, we utter is uttered with the intention of satisfying some goal or intention.

- A theory of how utterances are used to achieve intentions is a speech act theory.
  - Proposed by John Searle, 1969.

Speech Acts: Searle

- In his 1969 book Speech Acts: an Essay in the Philosophy of Language he identified:
  - Representatives:
    - such as informing, 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 naming.

Speech Acts: Searle

- There is some debate about whether this (or any!) typology of speech acts is appropriate.

- In general, a speech act can be seen to have two components:
  - A performative verb:
    - E.g., request, inform, …etc
  - Propositional content:
    - E.g., 'the door is closed'

The door is closed…

request
    speech act = *"please close the door"*

inform
    speech act = *"the door is closed!"*

inquire
    speech act = *"is the door closed?"*

Each of the above speech acts result from the same propositional content (*"…the door is closed…"*), but with different performatives

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.
  - Just like STRIPS planner
- ## Note that a speaker cannot (generally) force a hearer to accept some desired mental state.

The semantics for "request"

$request(s, h, \varphi)$

precondition:

- s believes h can do φ

  (you don't ask someone to do something unless you think they can do it)

- s believe h believe h can do φ

  (you don't ask someone unless they believe they can do it)

- s believe s want φ

  (you don't ask someone unless you want it!)

post-condition:

- h believe s believe s want φ

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

KQML and KIF

- We now consider agent communication languages (ACLs)
  - ACLs are standard formats for the exchange of messages

> "… [developing] protocols for the exchange of represented knowledge between autonomous information systems…"
>
> Tim Finin, 1993

- One well known ACL is KQML, developed by the DARPA-funded Knowledge Sharing Effort (KSE).
  - The ACL proposed by KSE was comprised of two parts:
    - the message itself: the Knowledge Query and Manipulation Language (KQML); and
    - the body of the message: the Knowledge interchange format (KIF).

KQML and KIF

- KQML is an 'outer' language, that defines various acceptable 'communicative verbs', or performatives.
  - Example performatives:
    - ask-if ('is it true that. . . ')
    - perform ('please perform the following action. . . ')
    - tell ('it is true that. . . ')
    - reply ('the answer is . . . ')

- KIF is a language for expressing message content, or domain

- knowledge.
  - It can be used to writing down ontologies.
  - KIF is based on first-order logic.

KQML & Ontologies

- In order to be able to communicate, agents need to agree on the words (terms) they use to describe a domain.
  - Always a problem where multiple languages are concerned.

- A formal specification of a set of terms is known as a ontology.
  - The DARPA Knowledge Sharing Effort project has associated with it a large effort at defining common ontologies
    - software tools like ontolingua, etc, for this purpose.

- We've previously discussed the use of ontologies and semantics...

Blocksworld

- The environment is represented by an ontology.

Recap:
The ontology specified the entities or the predicates we could use, and defined the actions and their meaning (semantics)

Blocksworld Ontology

$On(x,y)$	object x on top of object y
$OnTable(x)$	object x is on the table
$Clear(x)$	nothing is on top of object x
$Holding(x)$	arm is holding x

	$Stack(x,y)$
pre	$Clear(y) \wedge Holding(x)$
del	$Clear(y) \wedge Holding(x)$
add	$ArmEmpty \wedge On(x,y)$

	$UnStack(x,y)$
pre	$On(x,y) \wedge Clear(x) \wedge ArmEmpty$
del	$On(x,y) \wedge ArmEmpty$
add	$Holding(x) \wedge Clear(y)$

Ontologies

- The role of an ontology is to fix the meaning of the terms used by agents.

> "… An ontology is a formal definition of a body of knowledge. The most typical type of ontology used in building agents involves a structural component. Essentially a taxonomy of class and subclass relations coupled with definitions of the relationships between these things …"
>
> Jim Hendler

- How do we do this? Typically by defining new terms in terms of old ones.
  - Let's consider an example.

Ontologies

- Alice:
  - Did you read "Prey"?

- Bob:
  - No, what is it?

- Alice:
  - A science fiction novel. Well, it is also a bit of a horror novel. It is about multiagent systems going haywire.
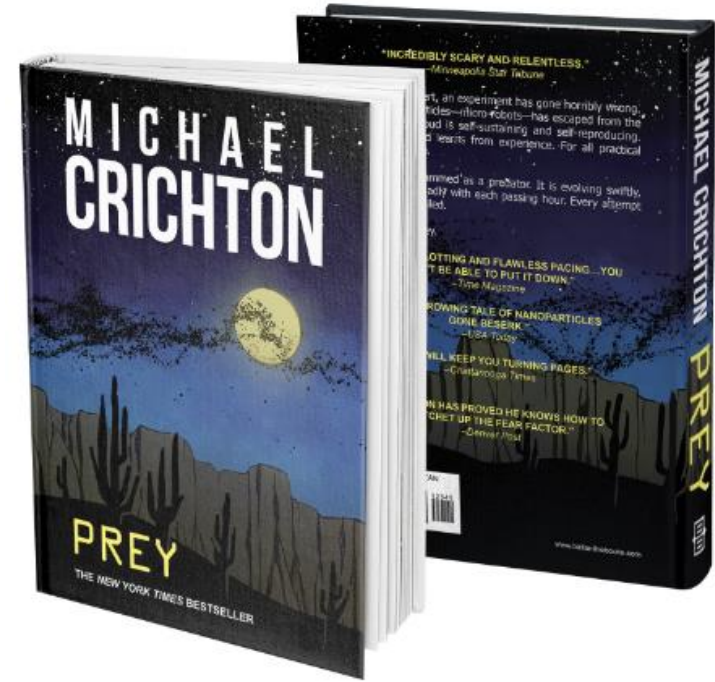
Ontologies

- What is being conveyed about "Prey" here?
    1. It is a novel
    2. It is a science fiction novel
    3. It is a horror novel
    4. It is about multiagent systems

- Alice assumes that Bob knows what a "novel" is, what "science fiction" is and what "horror" is.

- She thus defines a new term "Pery" in terms of ones that Bob already knows.

Types of objects

Classes
collections of things with similar properties

Instances
specific examples of classes

Relations
Describe the properties of objects and connect them together

Note that we also have these types of objects in languages such as Java, or modelling frameworks such as ER Diagrams. Such languages and frameworks also support inheritance.

Ontologies

- Part of the reason this interaction works is that Bob has knowledge that is relevant.
    - Bob knows that novels are fiction books
        - "novel" is a subclass of "fiction book"
    - Bob knows things about novels: they have
        - authors,
        - publishers,
        - publication dates, and so on.

- Because "Prey" is a novel, it *inherits* the properties of novels. It has an author, a publisher, a publication date.
   - Instances inherit attributes from their classes.

Ontology Inheritance

- Classes also inherit.

    - Classes inherit attributes from their super-classes

        - If "novel" is a subclass of "fiction book", then "fiction book" is a superclass of "novel"

- Fiction books are books.

    - Books are sold in bookstores.

   - Thus fiction books are sold in bookstores.

Ontologies

- A lot of knowledge can be captured using these notions.

  - We specify which class "is-a" sub-class of which other class.

  - We specify which classes have which attributes.

  - An axiomatic theory can also be included to support inference

    - if socrates is [an instance of a] man, and all men are mortal…

    - … we can infer that socrates is mortal!

- This structure over knowledge is called an ontology.

  - A knowledge base is an ontology with a set of instances.

- A huge number of ontologies have been constructed.

An ontology of threats
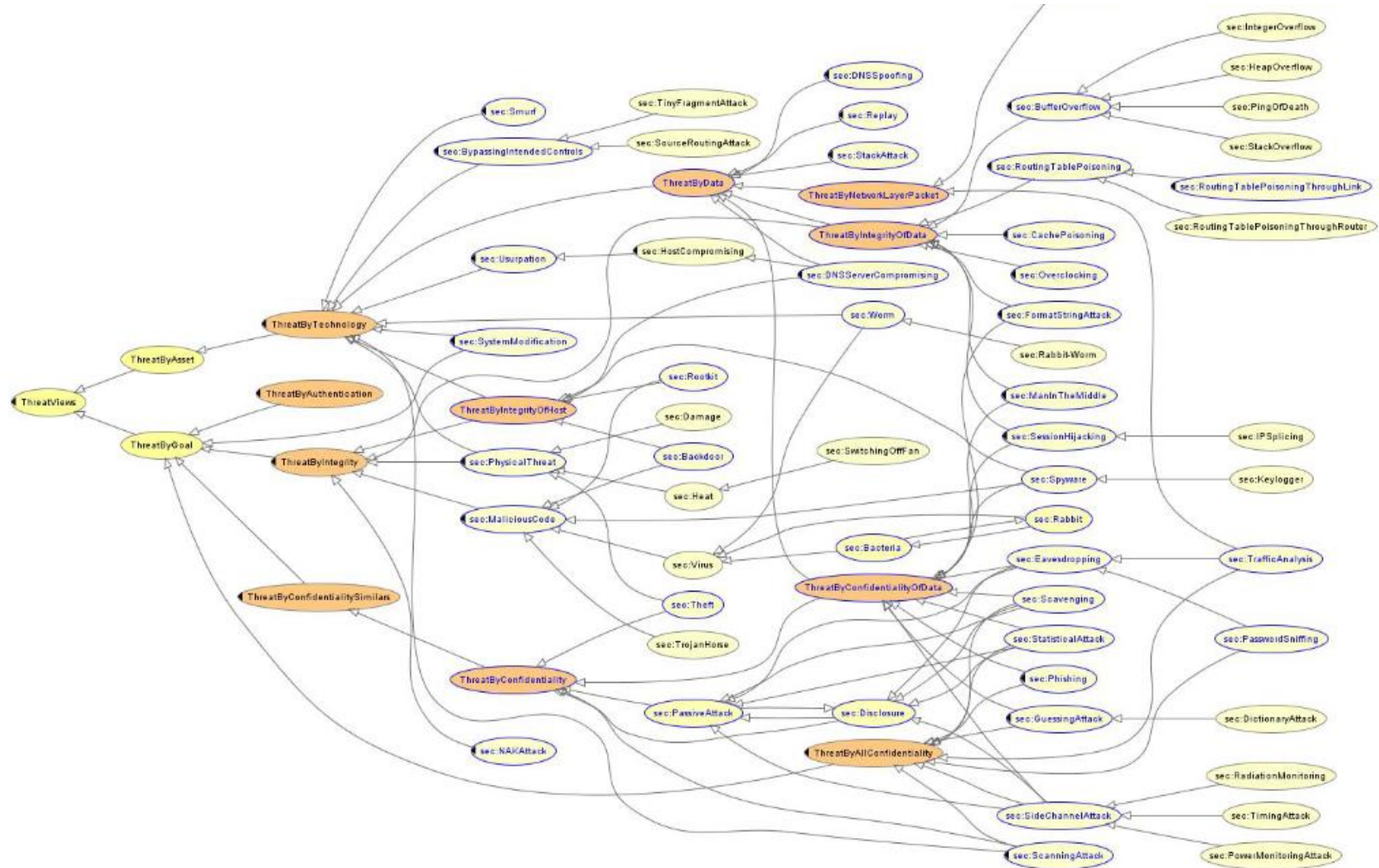
Ontologies

- In general, there are multiple ontologies at different levels of detail.

  - Application ontology

    - Like the threat ontology (see opposite)

  - Domain ontology

  - Upper ontology

    - Contains very general information about the world.

- The more specific an ontology, the less reusable it is.
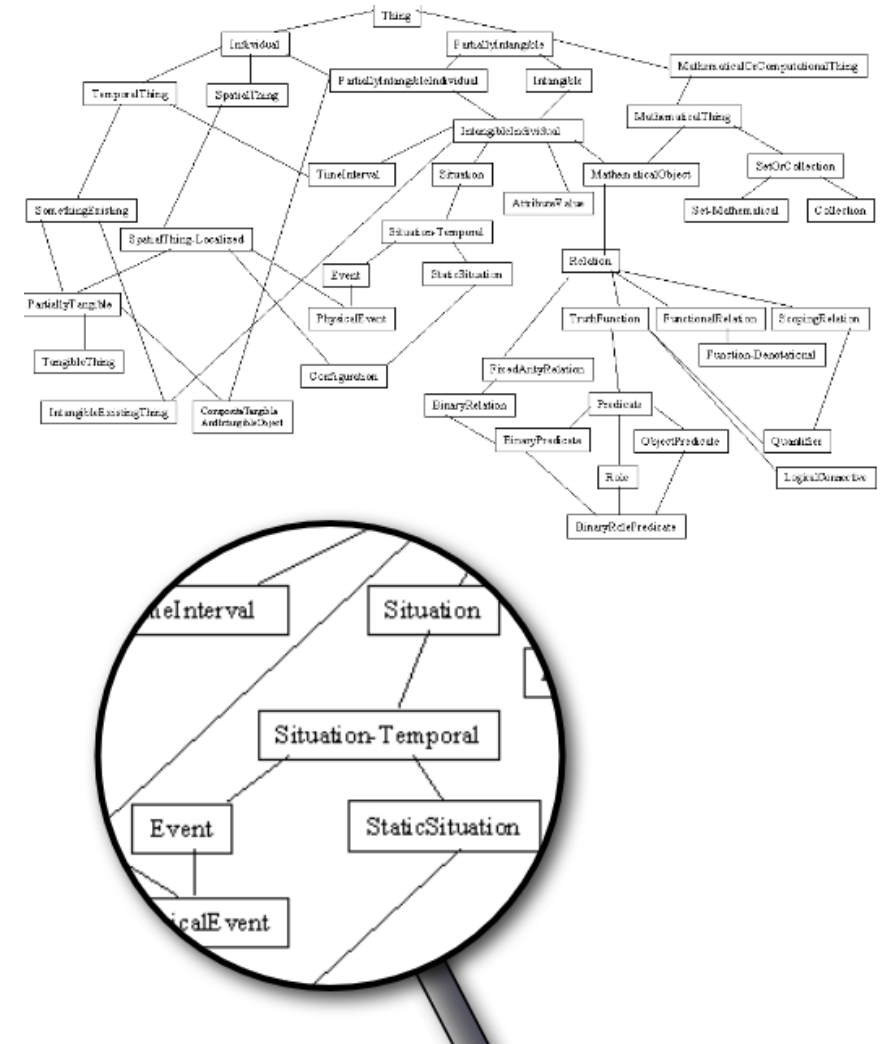
Multiple Ontologies

- Application and domain ontologies will typically overlap
  - Illustrated by the challenges of facilitating interoperability between similar ontologies.
  - Different knowledge systems can be integrated to form merged knowledge bases.

- But in many cases, all that is needed is to be understood!

Ontologies as perspectives on a domain

A single domain may have an **intended representation** in the real world, that is not perfectly represented by any single formal ontology. Many separate ontologies then emerge, based on different contexts…

Ontology Alignment Evaluation Initiative

A test suite of similar ontologies used to test out alignment systems that link different ontologies representing the same domain.

The conference test suite consists of 21 ontology pairs.

Modelling and Context

- The problem with modelling ontologies is that different designers have different contexts, and requirements

The Architect

When modelling a bridge, important characteristics include:

tensile strength
weight
load
etc

*Pat Hayes, 2001 in conversation*

The Military

When modelling a bridge, important characteristics include:

what munitions are required to destroy it!

Ontologies, Alignments and Correspondences

- Different ontological models exist for overlapping domains
  - Modelled implicitly, or explicitly by defining entities (classes, roles etc), typically using some logical theory, i.e. an Ontology

- Alignment Systems align similar ontologies
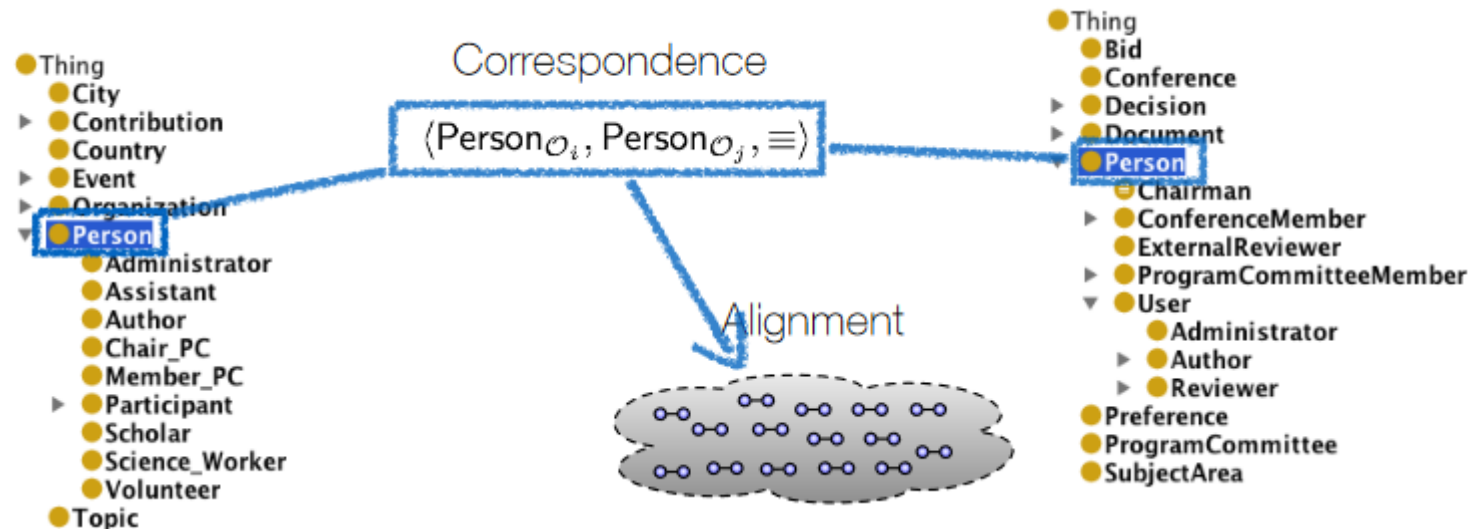
Aligning Agents' Ontologies

> "… as agents can differ in the ontologies they assume, the resulting semantic heterogeneity can impede meaningful communication. One solution is to align the ontologies; i.e. find correspondences between the ontological entities to resolve this semantic heterogeneity. However, this raises the question: how can agents align ontologies that they do not want to disclose?…"
>
> Terry Payne & Valentina Tamma, 2014

- Agent alignment is inherently *decentralised*!

Correspondence Inclusion Dialogue

- ## Correspondence Inclusion Dialogue (CID)
  - Allows two agents to exchange knowledge about correspondences to agree upon a mutually acceptable final alignment AL.
  - This alignment aligns only those entities in each agents' working ontologies, without disclosing the ontologies, or all of the known correspondences.

- ## Assumptions
  1. Each agent knows about different correspondences from different sources
  2. This knowledge is partial, and possibly ambiguous; i.e. more than one correspondence exists for a given entity
  3. Agents associate a utility (Degree of Belief) κc to each unique correspondence
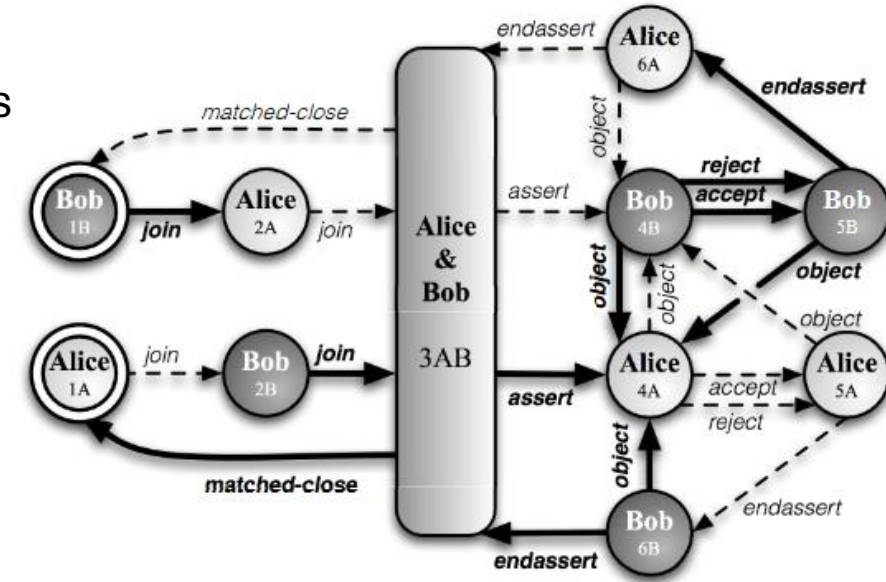
Picking the right correspondences

- Quality vs Quantity
  - Do we maximise coverage
    - Preferable when merging the whole ontology
  - Do we find the "best" mappings
    - Preferable when aligning specific signatures
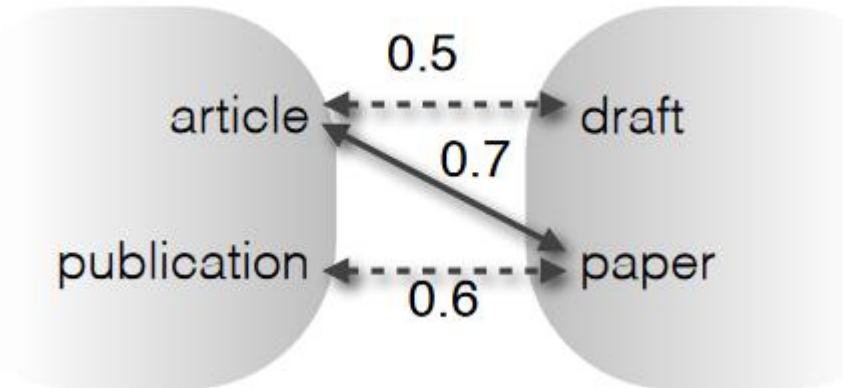
OWL - Web Ontology Language

- A general purpose family of ontology languages for describing knowledge
  - Originated from the DARPA Agent Markup Language Program
  - Followup to DARPA Control of Agent Based Systems (CoABS)

- Based on description logics
  - Various flavours with different expressivity / computability
  - Different syntaxes: XML, Turtle, Manchester Syntax…
  - Underpins the Semantic Web

OWL Example

```
NS1:geographicCoordinates rdf:nodeID='A179'/>
   <NS1:mapReferences>North America</NS1:mapReferences>
   <NS1:totalArea>9629091</NS1:totalArea>
   <NS1:landArea>9158960</NS1:landArea>
   <NS1:waterArea>470131</NS1:waterArea>
   <NS1:comparativeArea>about half the size of Russia; about three-tenths the size of Africa; about half the size of South America (or
slightly larger than Brazil); slightly larger than China; about two and a half times the size of Western Europe
   </NS1:comparativeArea>
   <NS1:landBoundaries>12034</NS1:landBoundaries>
   <NS1:coastline>19924</NS1:coastline>
   <NS1:contiguousZone>24</NS1:contiguousZone>
   <NS1:exclusiveEconomicZone>200</NS1:exclusiveEconomicZone>
   <NS1:territorialSea>12</NS1:territorialSea>
   <NS1:climate>mostly temperate, but tropical in Hawaii and Florida, arctic in Alaska, semiarid in the great plains west of the
Mississippi River, and arid in the Great Basin of the southwest; low winter temperatures in the northwest are ameliorated occasionally in
January and February by warm chinook winds from the eastern slopes of the Rocky Mountains
   </NS1:climate>
   <NS1:terrain>vast central plain, mountains in west, hills and low mountains in east; rugged mountains and broad river valleys in
Alaska; rugged, volcanic topography in Hawaii
   </NS1:terrain>
```

OWL and Services

- OWL-S is an *upper level service ontology* developed to describe agent based (or semantic web) services

  - Profile used for discovery

    - Input/outputs, etc

  - Process model provided a planning formalism

  - Grounding linked to the syntactic messaging
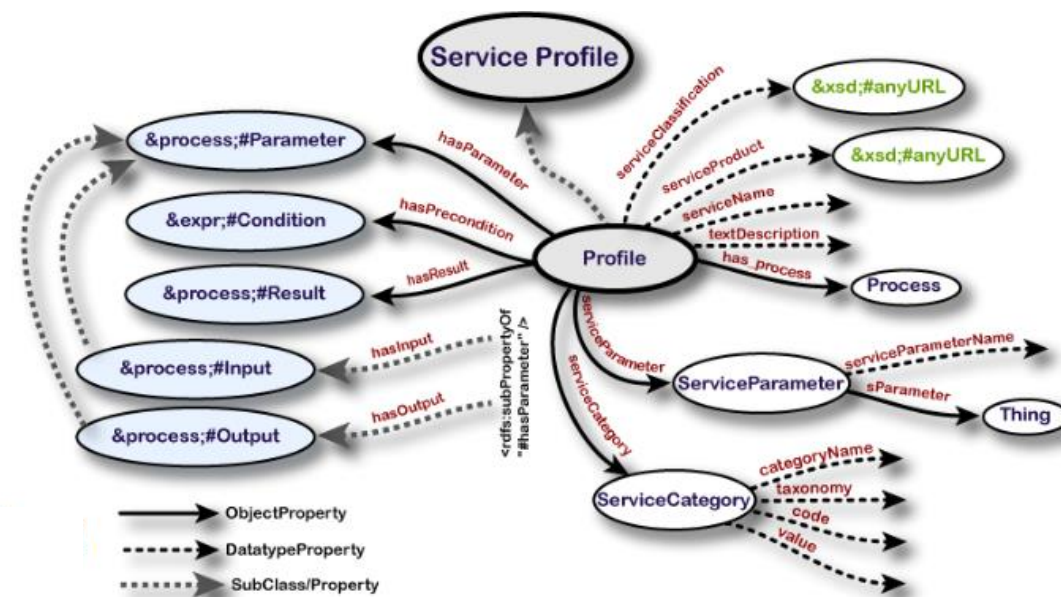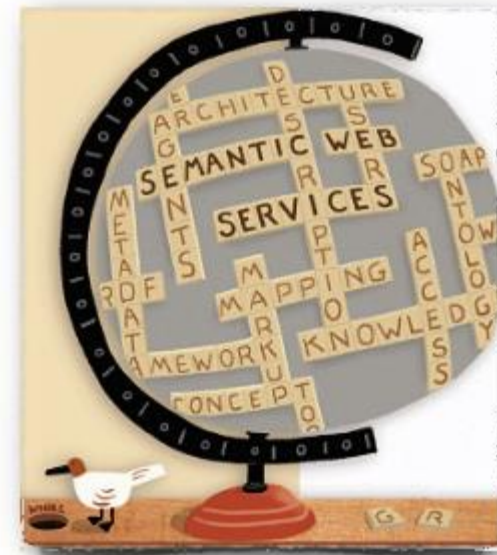
KQML / KIF

- After that digression, we can return to the KQML/KIF show.
  - KQML is an agent communication language. It provides a set of performatives for communication.

- KIF is a language for representing domain knowledge.
  - It can be used to writing down ontologies.
  - KIF is based on first-order logic.

- Given that, let's look at some examples.

KQML/KIF Example

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

A asks B for information about motor 1, using the ontology (represented in KIF) about motors.

```
(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
)
```

B responds to A's query q1. Two facts are sent:
1) that the torque of motor 1 is 12kgf; and
2) that the status of the motor is normal.

The ask stream is terminated using the eos performative.

Problems with KQML

- The basic KQML performative set was fluid

  - Different implementations were not *interoperable*

- Transport mechanisms for messages were not precisely defined

  - Again – *interoperability*

- Semantics of KQML were not rigorously defined

  - Ambiguity resulted in impairing *interoperability*

- There were no commissives in the language

  - Without the ability to *commit* to a task, how could agents *coordinate* behaviour

- The performative set was arguably ad-hoc and overly large

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:
  - The number of performatives was reduced to 20.
  - A formal semantics has been defined for the language, using the language SL
    - SL can represents beliefs, desires and uncertain beliefs, as well as actions

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			

"Inform" and "Request"

- This lecture has looked at two further kinds of agent:
  - Reactive agents; and
  - Hybrid agents.

- *Reactive agents* build complex behaviour from simple omponents.

- *Hybrid agents* try to combine the speed of reactive agents with the power of deliberative agents.

- Reading for this week:
  - M.Wooldridge: An introduction to MultiAgent Systems – Ch. 5 Reactive and Hybrid Architectures
  - A Robust Layered Control System for a Mobile Robot,", Rodney A. Brooks. IEEE Journal of Robotics and Automation, 2(1), March 1986, pp. 14–23. (also MIT AI Memo 864, September 1985)

"Inform" and "Request"

- "Inform" and "Request" are the two basic performatives in FIPA. Others are macro definitions, defined in terms of these.

- The meaning of inform and request is defined in two parts:
  - pre-condition: what must be true in order for the speech act to succeed.
  - "rational effect": what the sender of the message hopes to bring about.

FIPA "Inform" Performative

The content is a *statement*. The pre-condition is that the sender:

- holds that the content is true;
- intends that the recipient believe the content;
- does not already believe that the recipient is aware of whether content is true or not

The speaker only has to believe that what he says is true.

FIPA "Request" Performative

The content is an *action*. The pre-condition is that the sender:

- intends action content to be performed;
- believes recipient is capable of performing this action;
- does not believe that recipient already intends to perform action.

The last of these conditions captures the fact that you don't speak if you don't need to.

Jade

- The FIPA ACL provides a language for writing messages down.
  - It says nothing about how they are passed between agents.

- Several software platforms have been developed to support ACL-based communication.
  - One of the most widely used is JADE.

- Provides transparent (from the perspective of the agent designer) transport of ACL messages
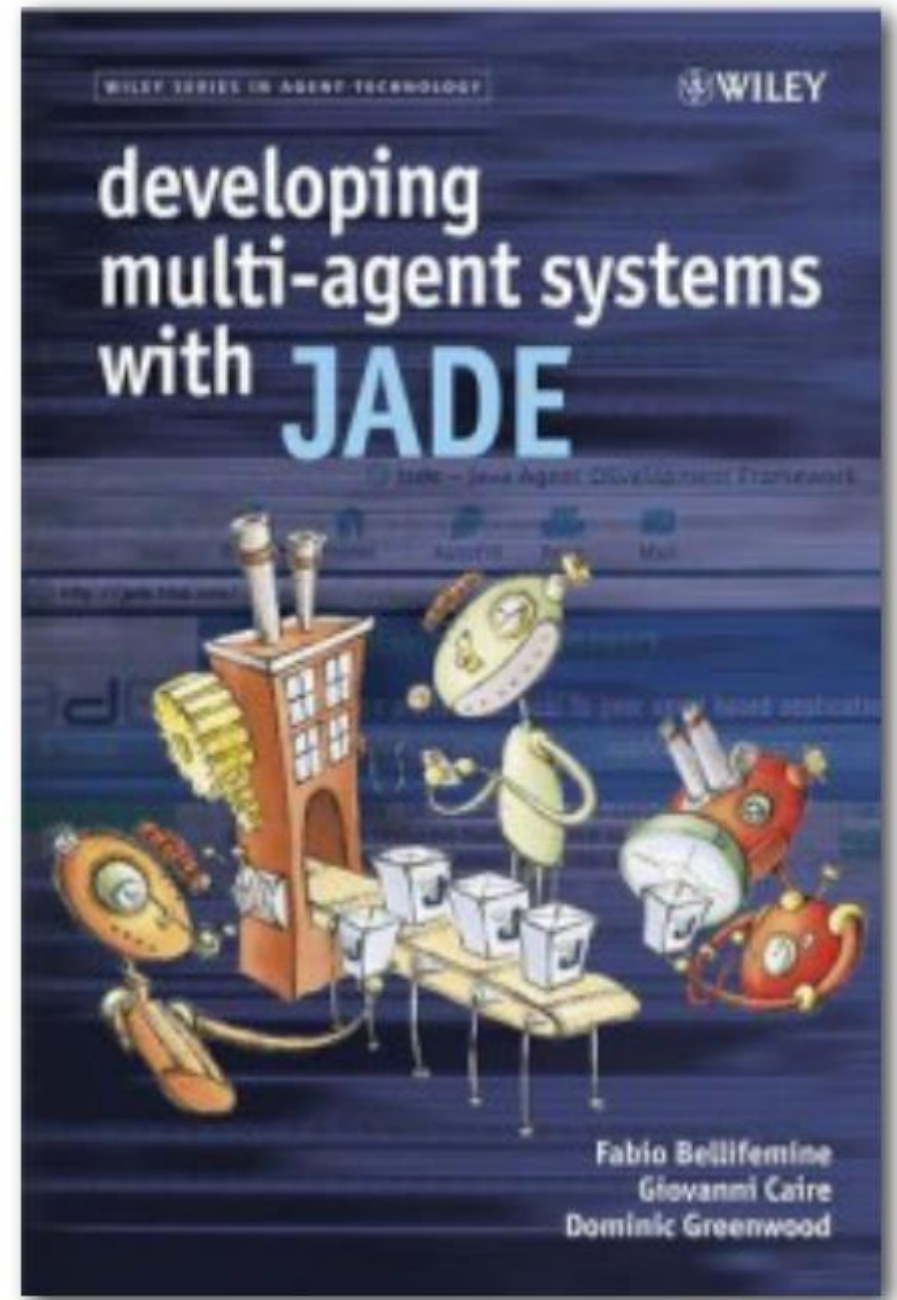
Jade

- In JADE, agents are Java threads running in a "container".
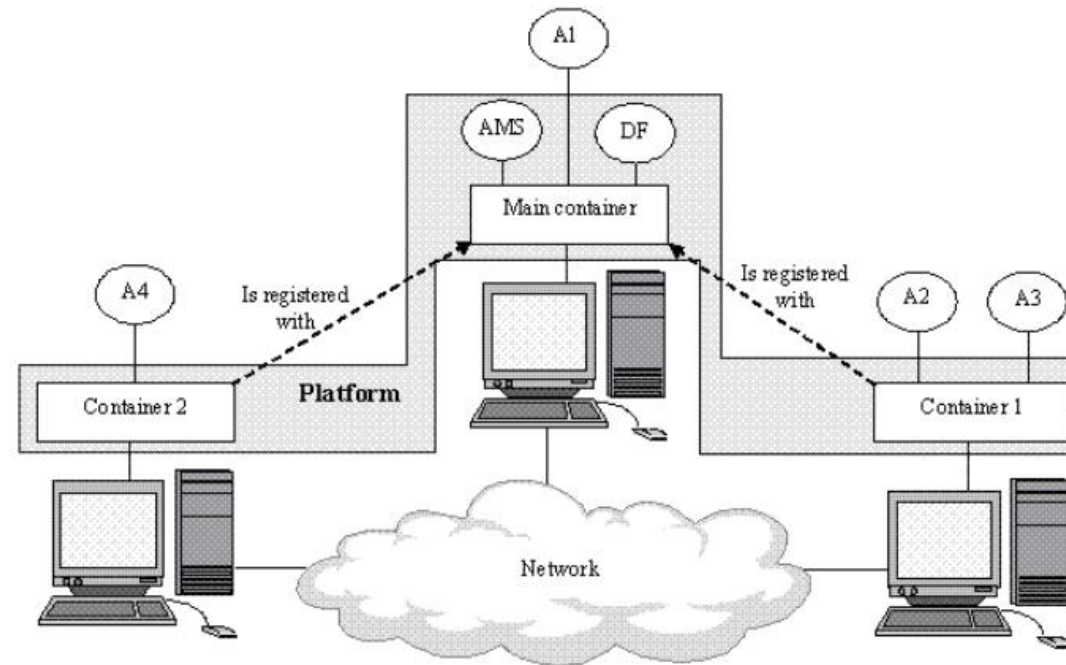
- All containers register with the main container

JADE Main Container

- The main container does the following:
  - Maintains the container table which lists all the containers and their contact information.
  - Maintains a list of all the agents in the system (including location and status).
  - Hosts the agent management system (AMS) which names agents as well as creating and destroying them.
  - Hosts the directory facilitator which provides a yellow pages allowing agents to be identified by the services they provide.

- See http://jade.tilab.com/ for more details.

AutoGen

- AutoGen is a framework that enables the development of LLM applications using multiple agents that can converse with each other to solve tasks. AutoGen agents are customizable, conversable, and seamlessly allow human participation. They can operate in various modes that employ combinations of LLMs, human inputs, and tools.

- https://github.com/microsoft/autogen
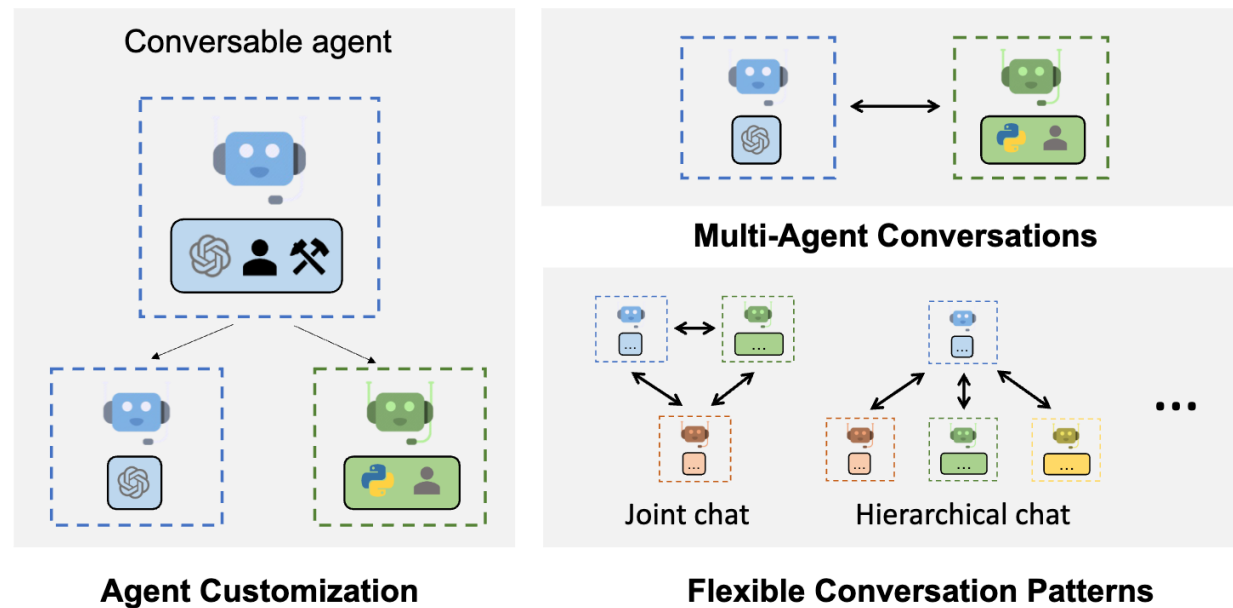
Class Reading

- M.Wooldridge: An introduction to MultiAgent Systems – Ch. 6 Understanding Each Other

- M.Wooldridge: An introduction to MultiAgent Systems – Ch. 7 Communicating