Agent Communication

Tim Finin
University of Maryland, Baltimore County
finin@umbc.edu
http://umbc.edu/~finin

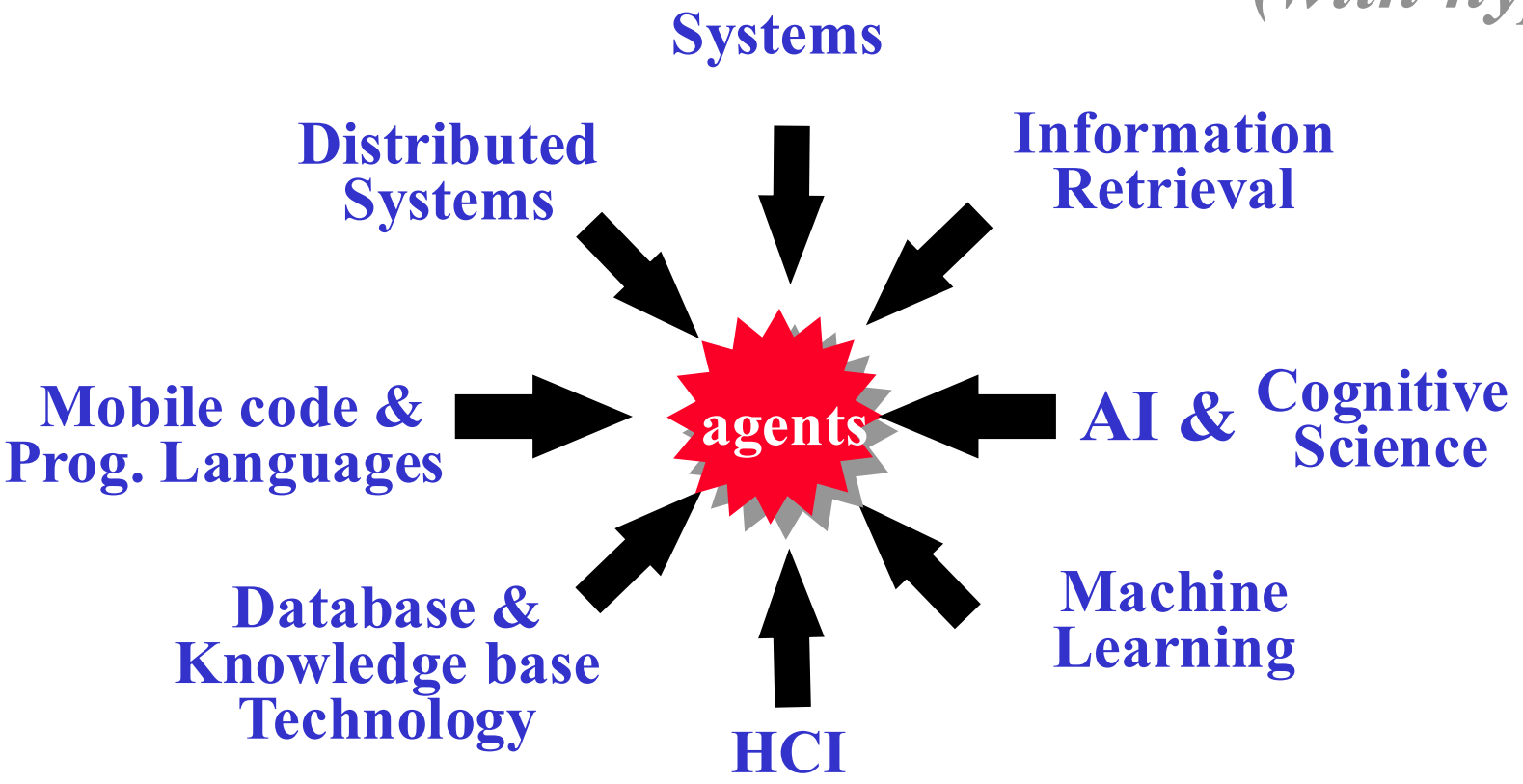
What is an agent?

An agent is a powerful and ubiquitous abstraction in computer science

- Daemons (e.g., ftp agent)
- User interface clients (e.g., mail agent)
- Physical agents (e.g., robotics)
- Believable agents (e.g., VR and graphics)
- (Intelligent) software agents
  - Mobile agents
  - User-facing agents
  - Personal (expert) assistants
  - Cooperating software agents

an emerging system-building paradigm

(with hype)



$$\frac{\text{agents}}{1997} = \frac{\text{objects}}{1982} = \frac{\text{structured programming}}{1975}$$

So, what's a software agent?

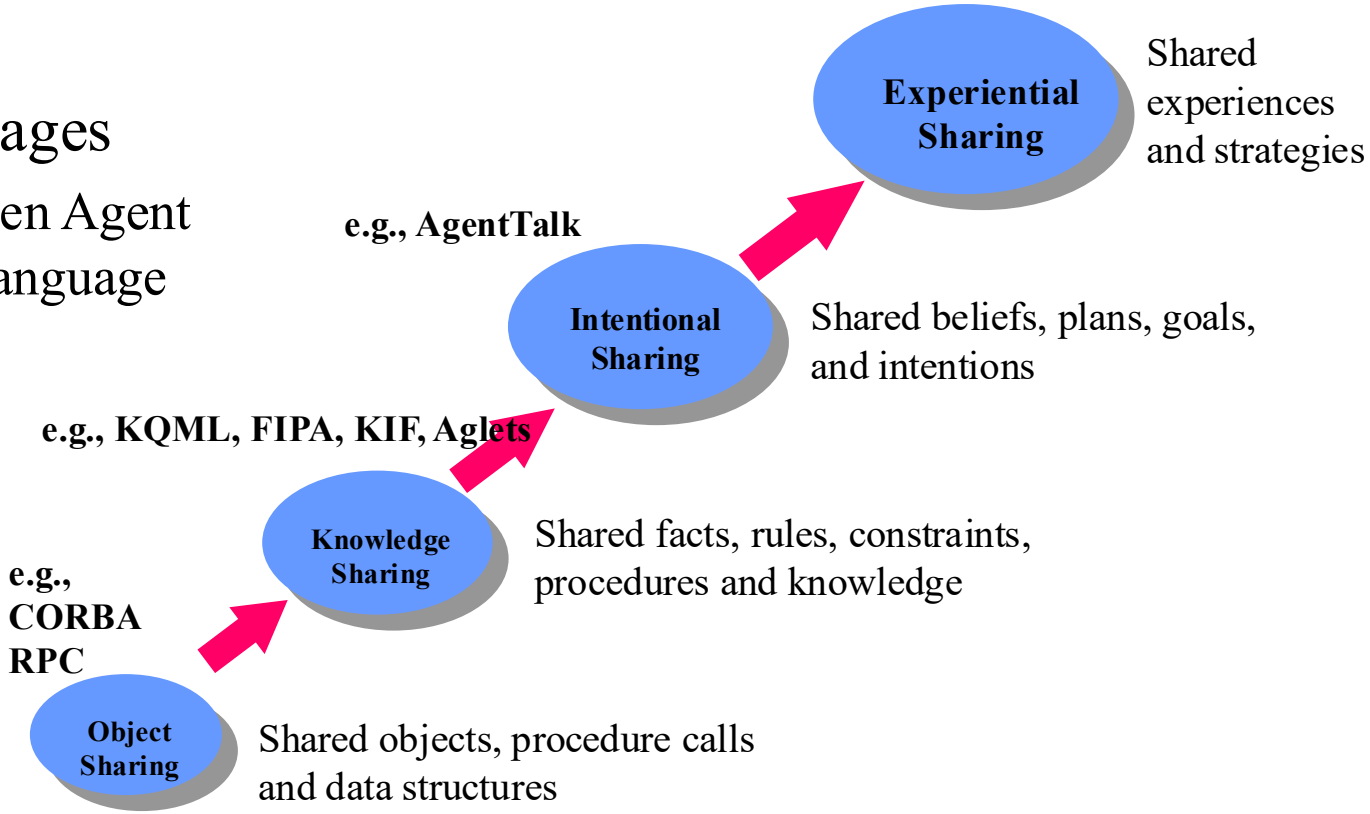
- There isn't a consensus yet, but several key concepts are important to this emerging paradigm.
- A software agent:
  - is an autonomous, goal-directed process
  - is situated in, is aware of, and reacts to its environment
  - cooperates with other agents (software or human) to accomplish its tasks

Agent Communication

- *Cooperation* requires communication
- **Opinion:** Agent-agent communication is the key to realizing the potential of the agent paradigm, just as the development of human language was a key to the development of human potential
- We assume that agents use an *Agent Communication Language* or ACL to communication information and knowledge.
- Genesereth (CACM, 92) defined a software agent as any system which uses an ACL to exchange information.

Some ACLs

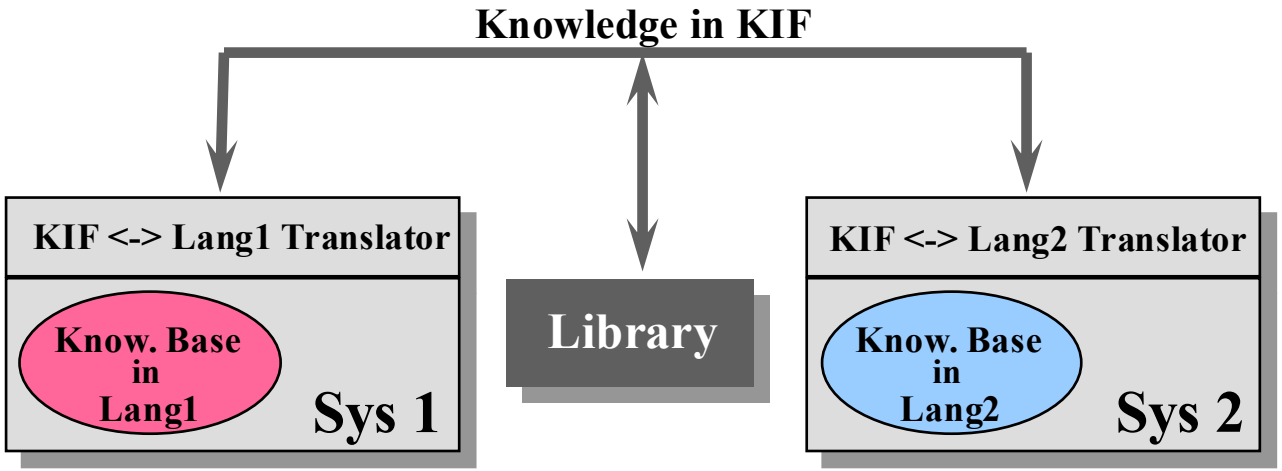
- Is CORBA an ACL?
- Knowledge sharing approach
  - KQML+KIF+Ontologies
- FIPA
- Ad hock languages
  - e.g., SGI's Open Agent Architecture language



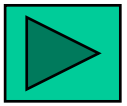
Knowledge Sharing Effort

- The KSE is a distributed research project involving over a dozen research groups aimed at developing techniques, methodologies and software tools for *knowledge sharing* and *knowledge reuse*.
- This requires a common language (syntax, semantics, pragmatics)
- Some existing components that can be used independently or together:
  - KIF -- knowledge interchange format *(syntax)*
  - Ontolingua - a language for defining sharable ontologies *(semantics)*
  - KQML - a high-level interaction language *(pragmatics)*

Knowledge Interchange Format



- KIF ~ First order logic with set theory
- An interlingua for encoded knowledge
  - Takes translation among n systems from $O(n^2)$ to $O(n)$
- Common language for reusable knowledge
  - Implementation independent semantics
  - Highly expressive -- can represent knowledge in typical application knowledge bases.
  - Translatable -- into and out of typical application languages



Common Semantics
Shared Ontologies and Ontolingua

Ontology : A common vocabulary and agreed upon meanings to describe a subject domain.

On*tol"o*gy (?), n. [Gr. the things which exist (pl.neut. of , , being, p.pr. of to be) + -logy: cf.F. ontologie.] That department of the science of metaphysics which investigates and explains the nature and essential properties and relations of all beings, as such, or the principles and causes of being.

Webster's Revised Unabridged Dictionary (G & C. Merriam Co., 1913, edited by Noah Porter)

Conceptual Schemas

This is not a profoundly new idea...

- Vocabulary specification
- Domain theory
- Conceptual schema (for a data base)
- Class-subclass taxonomy
- Object schema

139	74.50
140	77.60
…	…

Data Base:

Table: price
***stockNo: integer; cost: float**

Data Base Schema:

price(x, y) =>
∃ (x', y') [auto_part(x')
& stock_no(x') = x
& retail_price(x', y', Value-Inc)
& magnitude(y', US_dollars) = y]

Auto Product Ontology

Product Ontology

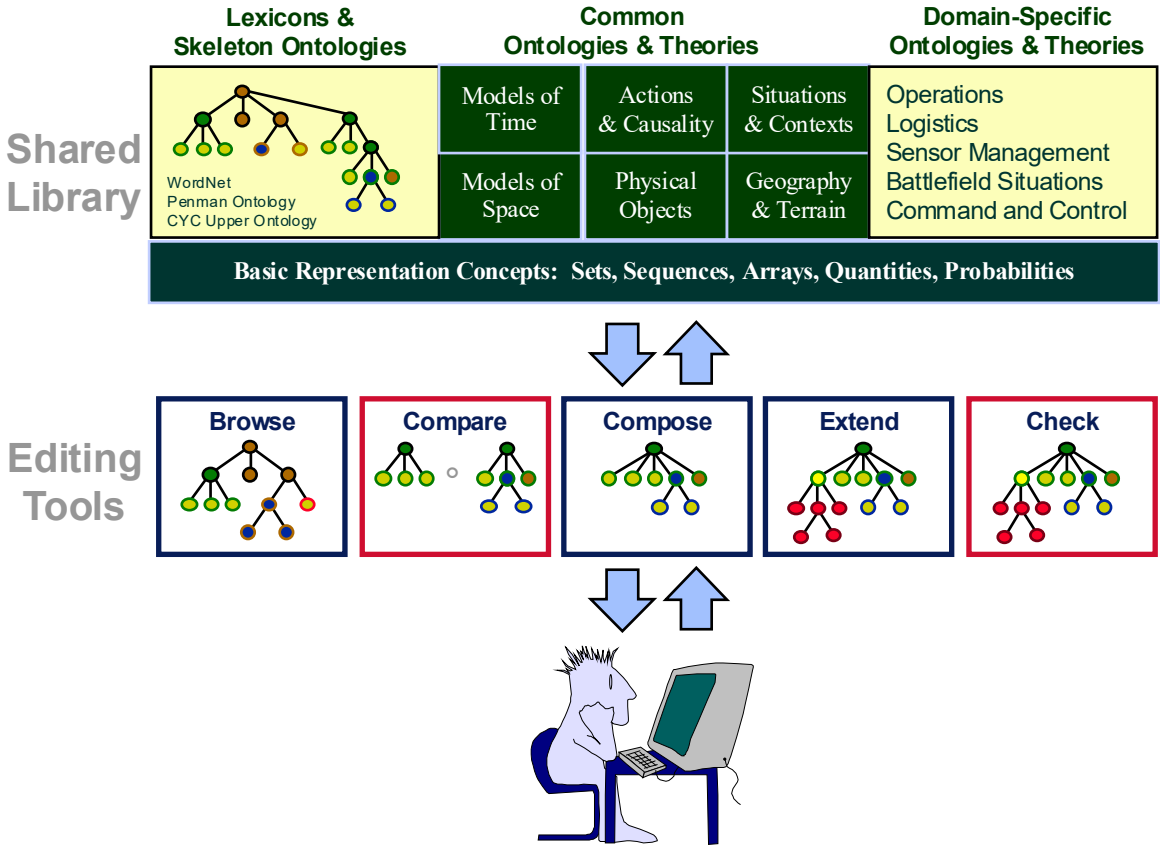
Units & Measures Ontology

Conceptual Schema:

Ontology Library and Editing Tools

Ontolingua is a language for building, publishing, and sharing ontologies.

- A web interface to a browser/editor at *http://ontolingua.stanford.edu/* and mirror sites.
- Internal representation is KIF augmented with a frame language ontology
  - enables OO structuring of ontologies
- Ontologies can be translated into other content languages, including KIF, LOOM, Prolog, CLIPS, etc.
- Provides user with fully cross-referenced HTML documents

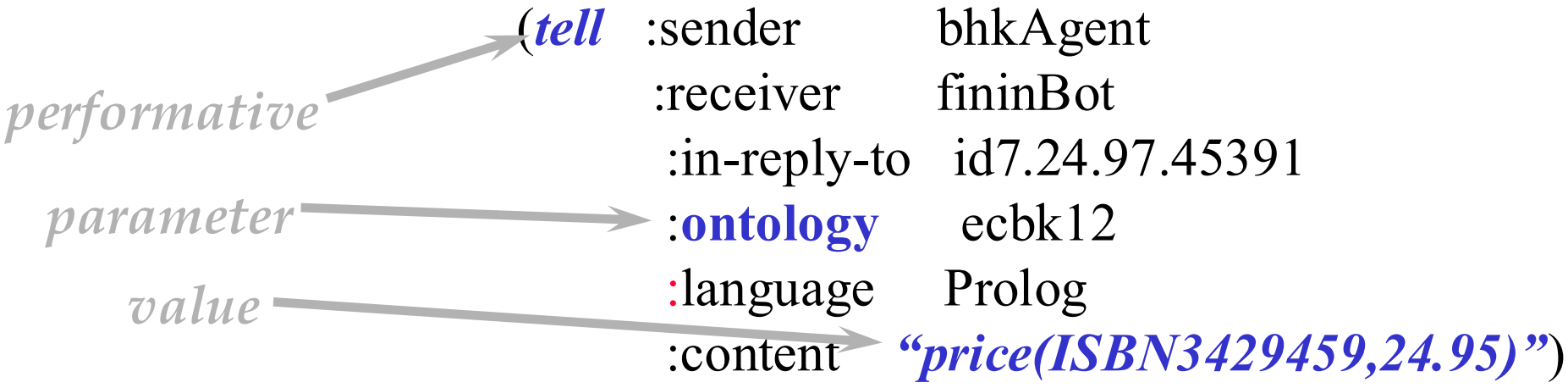


KQML

Knowledge Query and Manipulation Language

- KQML is a high-level, message-oriented, communication language and protocol for information exchange independent of content syntax and ontology.
- KQML is independent of
  - the transport mechanism (e.g., tcp/ip, email, corba objects, IIOP, etc.)
  - Independent of content language (e.g., KIF, SQL, STEP, Prolog, etc.)
  - Independent of the ontology assumed by the content.
- KQML includes primitive message types of particular interest to building interesting agent architectures (e.g., for mediators, facilitators, sharing of intentions, etc.)

A KQML Message



Represents a single *speech act* or *performative*

ask , tell, reply, subscribe, achieve, monitor, ...

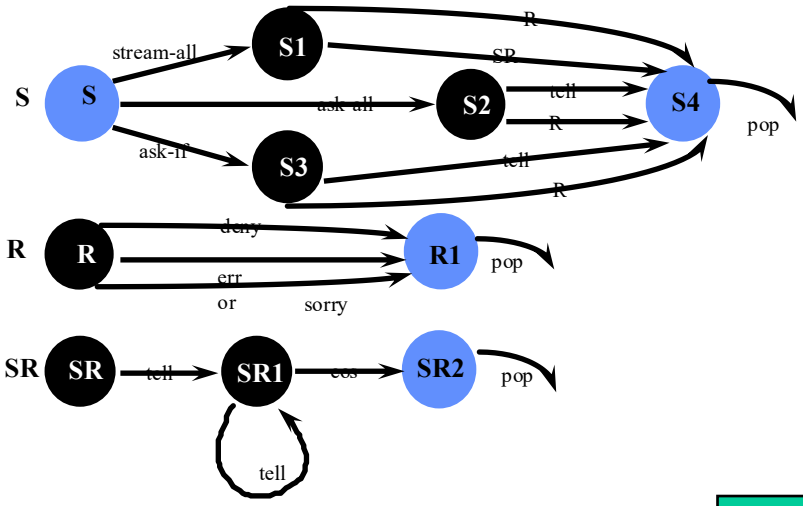
with an associated *semantics and protocol* and a list of *attribute/value pairs*

:content, :language, :from, :in-reply-to

Semantics for *tell(A,B,X)*

- bel(A,X), "A states to B that A believes X to be true (for A)."
- Pre(A): bel(A,X) AND know(A,want(B,know(B,S))) where S may be bel(A,X) or NOT(bel(A,X))
- Pre(B): intend(B,know(B,S))
- Post(A): know(A,know(B,bel(A,X)))
- Post(B): know(B,bel(A,X))
- Completion: know(B,bel(A,X))

Comment: The completion condition and postconditions hold unless a *sorry* or *error* suggests B's inability to properly acknowledge the *tell*.



Agent Communication Issues

Communication Issues

- Standards for agent technology
- What about communicating mobile agents?
- How to motivate this.
- Sounds good in theory, but...
- Language learning and language evolution

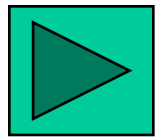
Issue

Agent standards

- Since interoperability is a defining characteristic of agents, this is important!
- What might be standardized?
  - Agent architecture
  - Agent communication language
  - Agent interaction protocols
  - Agent knowledge
  - Agent programming languages

Components

- **Agent architecture**
  - –Common computational environment and infrastructure in which our agents exist.
- **Agent communication language**
  - –High level language agents used among agents to communicate information and knowledge.
- **Agent interaction protocols**
  - –Rules agents follow in agent-agent interactions.
- **Agent knowledge**
  - –Common representation and reasoning languages, libraries of sharable ontologies and domain models used by agents.
- **Agent programming languages**
  - –Programming languages designed for or well suited to implementing agents (stationary or mobile).



Issue

Communicating mobile agents?

- To date no serious integration of the mobile agents and (rich) communicating agents paradigms.
- There don't seem to be any immediate deep issues
  - E.g., initial integration of TKQML and Agent-Tcl

  but there are some practical issues
  - Developing protocols for finding agents and/or forwarding messages
  - Getting out of the Java sandbox
- Might we develop a standard declarative language for communicating and sharing procedural knowledge?

Issue

How to motivate this

Q: What will these agents talk about?

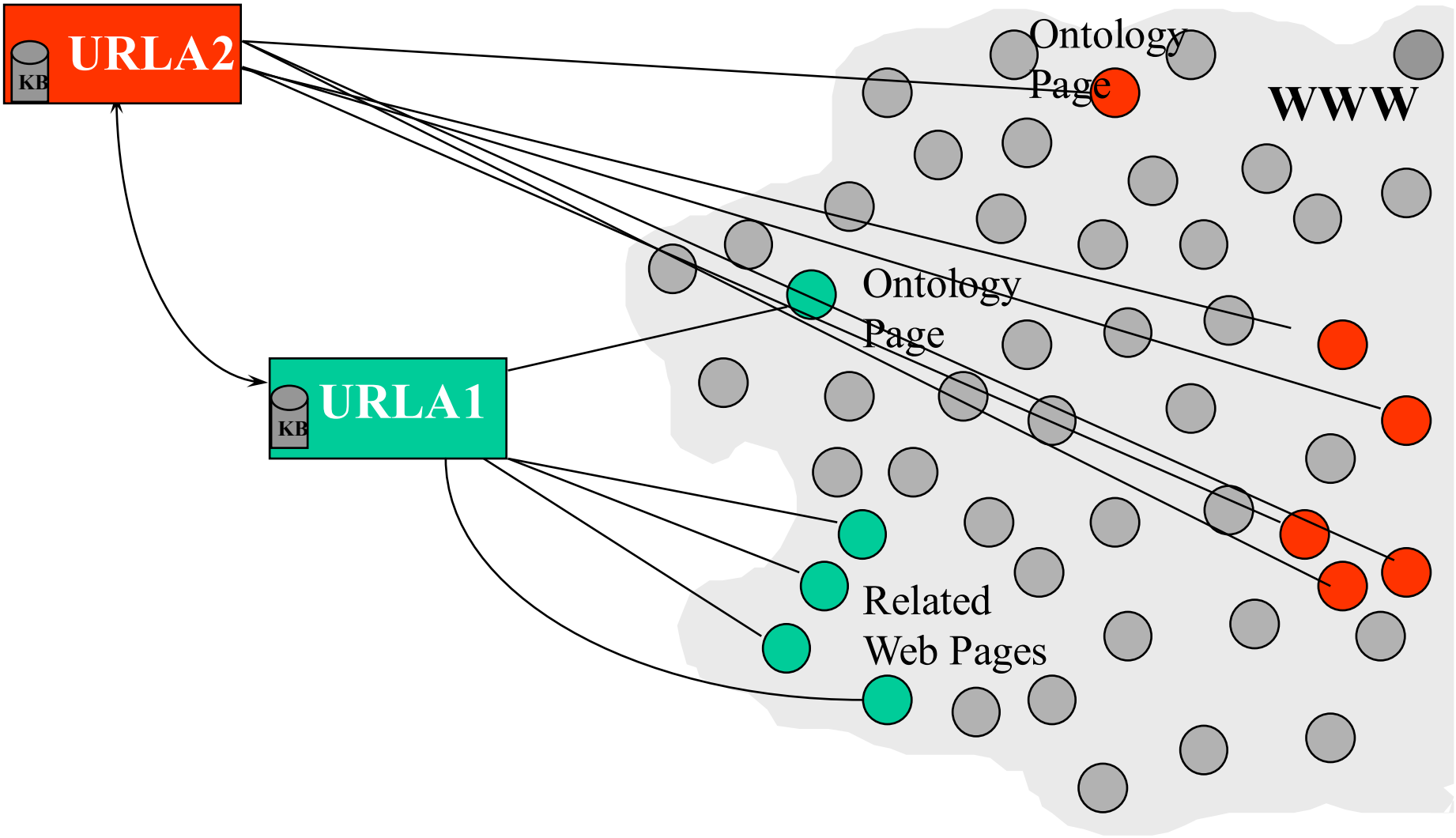
A: The web.

Semantic web markup with SHOE

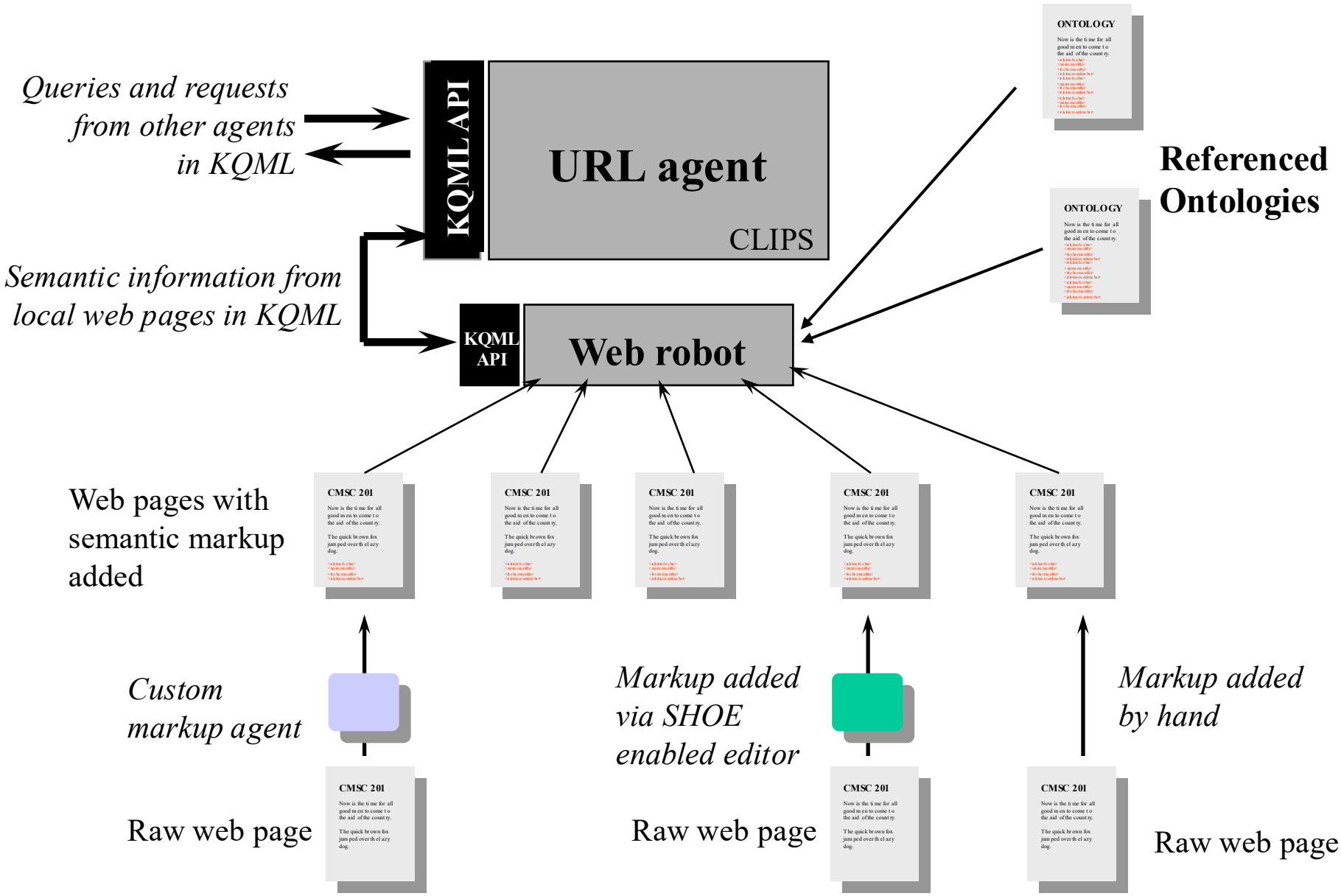
- *Simple HTML Ontology Extensions* developed by UMCP PLUS Group
- Ontological mark-up tags allow the web page author to add semantic knowledge of the page's content.
- Pages and sub-regions are the ground objects in the world.
- Definitions for classes, relations, attributes, axioms, etc. are also defined on web pages.
- *"URL agents"* are responsible for, and understand small groups of related web pages
  - Each URL has an agent which can think, speak, and act for it.

  and communicate with other agents with similar pages

URL Agents speak for one or more related web pages



Architecture



Example

WWW text

Prerequisite: CMSC 202

<RELATION
"hasPrereq"
TO="http:// ....#CMSC-202">

Shoe tag

(tell :to urla42
:from spider387
:content "(prereq CS-341 CS-202)"
:ontology "http://…/courseontology.html"
:language KIF)

KQML message sent to URL agent

Course Ontology

```
<ONTOLOGY ”course-ontology" VERSION=”2.5">
<ONTOLOGY-EXTENDS ”meeting-ontology"
      VERSION="2.1" PREFIX=”mtg"
      URL="http://onto.org/mtg/2.1/">
<ONTDEF CATEGORY=”course" ISA=”mtg.periodic">
<ONTDEF RELATION=”frequency" ARGS=”course timeinterval">
<ONTDEF RELATION=”hasPrereq" ARGS=”course course">
 <ONTDEF INFERENCE>
    <ONTIF RELATION=“hasPrereq” 1=“a” 2=“b”>
    <ONTIF RELATION=“chainPrereq” 1=“b” 2=“c”>
    <ONTTHEN RELATION=“chainPrereq” 1=“a” 2=“c”>
</ONTDEF>
</ONTOLOGY>
```

Issue

Sounds good in theory, but...

- Ok, so there are there any good examples of this stuff outside the lab?
- Is it “too much”?
  - Too theoretical, too academic, too hard to market, etc.
- Can we build on emerging “acceptable” standards?
  - Such as CORBA, XML, ActiveX, MCF, ...
- What will Microsoft do?

Issue

Language learning and evolution

- There are lots of hard questions:
  - How can agents track changing ACLs?
  - How do we achieve consensus on ontologies?
  - How can agents resolve ontological differences?
  - How can agents track evolving ontologies?
  - How should knowledge spread through communities of agents?
- These are roughly solved in communities of human agents, although sometimes by war.

Conclusion

Some key ideas

- Software agents offer a new paradigm for very large scale distributed heterogeneous applications.
- The paradigm focuses on the interactions of autonomous, cooperating processes which can adapt to humans and other agents.
- Communication is of central importance and, in particular:
  - Establishing common agent communication languages.
  - Developing common ontologies for application domains.
  - Establishing common agent protocols.
- The paradigm is still forming.

References

- R. Scott Cost, T. Finin, J. Lakhani, E. Miller, C. K. Nicholas and I. Soboroff, *Developing communicating software agents in Tcl*, 4th International Workshop on Agent Theories, Architectures, and Languages, Providence, July, 1997.
- Y. Labrou and T. Finin, "Semantics and Conversations for an Agent Communication Language", in "Readings in Agents", M. Huhns and M. Singh (eds), Morgan Kaufmann, 1997. (also in Proc. IJCAI-97). <http://kqml.org/papers/ijcai97.ps>
- T. Finin, Y. Labrou, and J. Mayfield, "KQML as an agent communication language", invited chapter in J. Bradshaw (Ed.), ``Software Agents", MIT Press, Cambridge, (1997). <http://www.cs.umbc.edu/lait/papers/kqml-acl.ps>
- S. Luke, L. Spector, D. Rager, and J. Hendler, Ontology-based Web Agents, Proc. 1st Int. Conference on Autonomous Agents, 1997. <http://www.cs.umd.edu/projects/plus/SHOE/aa-paper.ps>
- KIF page: http://www.cs.umbc.edu/kif/
- KQML page: http://www.cs.umbc.edu/kif/
- Ontolingua: *http://ontolingua.stanford.edu/*

Backup

KIF Syntax and Semantics

- Extended version of first order predicate logic
- Simple list-based linear ASCII syntax

  E.g., (forall ?x (=> (P ?x) (Q ?x)))
- Model-theoretic semantics
- Axiomatic specification of large function and relation vocabulary
- Includes vocabulary for numbers, sets, and lists

KSE Methodology
for developing agent-based systems

- **Collect/construct necessary ontologies**
  - Use standard, published ontologies if possible
  - Develop (and publish) new components as needed
  - Use common tools, e.g. *Ontolingua*, *GFP*, ...
- **Choose common representation language(s)**
  - e.g., SQL or KBMS with *KIF* is a recommended default
- **Use an ACL like KQML as communication language**
  - extend with new performatives and protocols as needed
- **Identify and define new higher-level protocols**
  - e.g., for negotiation, purchasing, cataloging, etc.

Agent architecture

- E.G.: SRI's OO and KQML's infrastructure
- Should specify agents and protocols which provide standard services
  - Name servers, brokers, authenticators, certification, logging, visualization, ...
- Should be self configuring and dynamic
- Should work over a wide range of scales (10s to 1000s of agents)
- Should support interoperability among heterogeneous agents
  - e.g., different implementation languages, representation languages, reasoning algorithms, platforms, etc.
- Should encourage "online plug-and-play of agents & people"
  - Support human collaboration
- Should have lightweight implementations and work in a variety of environments
  - e.g., PCs, Workstations, PDAs, etc.
- Should be distributed, network-centric and multi-threaded

Agent communication language

- E.G.: KQML, ARCOL, SATP, AgentTalk, ...
- Should be a *lingua franca* among agents.
- Should support modeling of human interactions and be a target for the HCI (e.g., "logical form" meaning representation)
- Should be semantically well-founded
- Should be consistent with the underlying communications environment
- Should work with a variety of *content* languages
- Should be compatible with current programming languages and environments

Agent interaction protocols

- E.G.: ad hoc techniques based on rules or FSA's
- Are published protocols for agent-agent and agent-human interactions.
- Should be the specification for good agent "citizenship" in the architecture.
- Should have a sound theoretical basis, consistent with the agent communication language.
- Should support compilation into executable specifications.

Agent knowledge

- Sharing information and knowledge will require common representation languages and ontologies
- Sharing can be facilitated by the use of common KR languages or via an interlingua (e.g., KIF).
- Shared ontologies and reference domain models are critical.

Do we really need semantics for an ACL?

- We have no complete semantic accounts for ACLs yet.
- KQML has been criticized for its lack of semantics.
- ACL Semantics should
  - Help with the process of standardization and agreement between implementations.
  - Provide a framework for defining new primitives.
  - License inferencing based on the message exchanges.

So, what do we mean by "semantics"?

Providing semantics is the process of ascribing meaning.

Claim: Describing the state of an agent before sending a particular message and after receiving it is a useful basis for ascribing meaning to the communication primitives.

So, the two-step process is:

- Which agents' states?
- Having a language to describe agents' states.

So, what constitutes the Semantic Description?

The semantic description is the sum of the following:

- Natural Language Description
- Formal Description of the NL description
- Preconditions
- Postconditions
- Completion Conditions
- Comments

How to describe agents' states?

Agent state descriptions refer to either

- **mental attitudes**
  knowledge, belief, intentions, …

- **actions**
  that have happened or will happen

How to describe agents' states?

- **bel(A,P)**
  P is an expression in A's Knowledge Base or Virtual KB
- **know(A,S)**
  knowledge that a mental state description (or action) holds (or has occurred)
- **want(A,S)**
  a desire for an action (or state) S to occur (to become true)
- **intend(A,S)**
  an intention , i.e., A has the intention of doing S and thus is committed towards achieving S in the future
- **proc(A,M)**
  refers to the action of A processing message M
- **sendMSG(A,B,M)**
  the action of A sending the message M to B

The arguments of the attitudes are either
- propositions (P), or,
- state descriptions (S).

Some explaining regarding the mental attitudes (part I)

- **bel(A,P)**

  Proposition P exists in the KB or can be proven true (some "turnstile")

- **know(A,S)**
  - an agent can know (know) that (not what) it believes (bel) a proposition, or
  - can know a state, or
  - something that has been communicated to it (a state or the occurrence of an action).

  So, **know(A,bel(A,P))** is fine but not **know(A,P)**

Some explaining regarding the mental attitudes (part II)

- **want(A,S)**
  - an agent can want to know some agent state, or
  - can want some action to occur
- **intend(A,S)**
  - an agent can intend to know some agent state, or
  - can intend for some action (of which it is the agent) to occur

Semantics for TELL

TELL(A,B,X)

- A states to B that A believes X to be true (for A).
- bel(A,X)
- Pre(A): bel(A,X) AND know(A,want(B,know(B,S)))
  where S may be bel(B,X) or NOT(bel(B,X))
  Pre(B): intend(B,know(B,S))
- Post(A): know(A,know(B,bel(A,X)))
  Post(B): know(B,bel(A,X))
- Completion: know(B,bel(A,X))
- The completion condition and postconditions hold unless a SORRY or ERROR suggests B's inability to properly acknowledge the TELL.

Semantics for the proactive-TELL

proactive-TELL(A,B,X)

- A states to B that A believes the content to be true.
- bel(A,X)
- Pre(A): bel(A,X)
  Pre(B): NONE
- Post(A): know(A,know(B,bel(A,X)))
  Post(B): know(B,bel(A,X))
- Completion: know(B,bel(A,X))
- The postconditions and completion condition hold unless a SORRY or ERROR suggests B's inability to properly acknowledge the TELL.

The case of KD45 semantics for BEL

If

TELL(A,B,P)

TELL(A,B,P implies Q)

Should A decide that BEL(B,Q) ?

Should B decide that BEL(A,Q) ?

Maybe , maybe not ...

Conversation Policies

- (intuitive) Where performatives "fit" in exchanges of KQML messages
- (formal) The minimum possible set of permissible KQML conversations, consistent with the semantic description.
- They are a formal specification, independent of implementation, but we've defined them as DCGs and as ATNs to provide "executable specifications" used to filter out inchoherent messages.

Other ACLs and semantic approaches

- Cohen & Lesveque
  - a few primitives only
  - heavily grounded on an agent theory
  - it is a moving target
- The FIPA version of an ACL
  - work in progress
  - there is an underlying agent theory
  - SL is the source of its power

KQML Semantics Conclusion

- We have developed a framework for providing the semantics of primitives for KQML-like language .
- We have tried to dissociate the ACL's semantic description from the agent theory of the conversing agents.
- We have provided semantics for the full set of KQML performatives (all 30-something of them).