
CmpE 593

Multiagent Systems

Pınar Yolum
pinar.yolum@boun.edu.tr

Department of
Computer Engineering
Boğaziçi University

Multiagent Systems

Based largely on

Service-Oriented Computing: Semantics, Processes, Agents

– Munindar P. Singh and Michael N. Huhns, Wiley, 2004

Multiagent Applications (1)

- Web Intelligence
 - Personal assistant that searches travel itineraries for you, picks a good one for your needs, and pays with your credit card
 - An agent-based peer in the P2P networks that models your taste in music, gets recommendations, connects to other peers, and downloads MP3s

Multiagent Applications (2)

- Ambient Intelligence
 - Agent in your refrigerator keeps track of your groceries (ex. LG Internet refrigerator).
 - Agent in your cell-phone keeps track of your location.
 - When you are passing in front of a Migros, your agent in the cell phone asks your refrigerator agent if you need groceries
 - Cell phone agent alerts you to stop at the market

Multiagent Applications (3)

- Business Intelligence
 - Agent carries out the transactions for a company
 - Negotiates with the customer for B2C transactions and with supporting service providers for B2B transactions
 - Negotiation in money, value, goods

Multiagent Applications (4)

- **Business Intelligence**
 - A customer agent participates in an auction for its user
 - Bids for individual goods or combination of goods in auctions
 - Sells some of its goods to other agents
- **Trading Agent Competition**

Multiagent Applications (5)

- Intelligent Healthcare
 - Agent runs on the Alzheimer patient's handheld.
 - Using a GPS, the agent keeps track of and learns where the patient goes in her everyday routine.
 - If she starts going places out of the ordinary, the agent locates the agent of a relative
 - Communicates the situation to the agent
- Usability concerns

Attributes of Multiagent Systems

- Decentralization
 - No central control
- Complex components
 - Each agent autonomously decides on actions
- Adaptive behavior
 - Agents learn certain behavior over time
- Complex interactions
 - Not rigid like typical distributed systems
- Coordination
- Emergent, aggregate behaviors
 - The interactions bring out interesting properties about structure or behavior

Dimensions of MAS: System

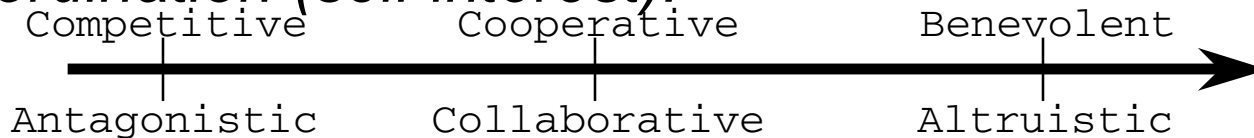
Scale (the number of agents):



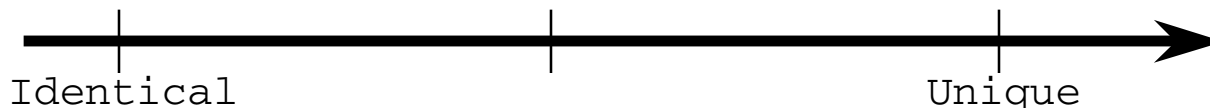
Interactions:



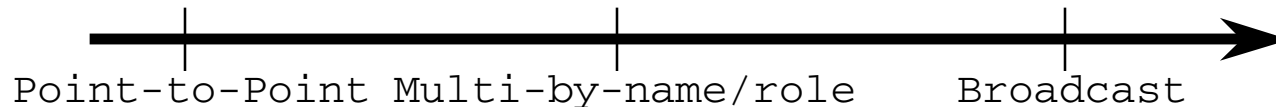
Coordination (self interest):



Agent Heterogeneity:



Communication Paradigm:



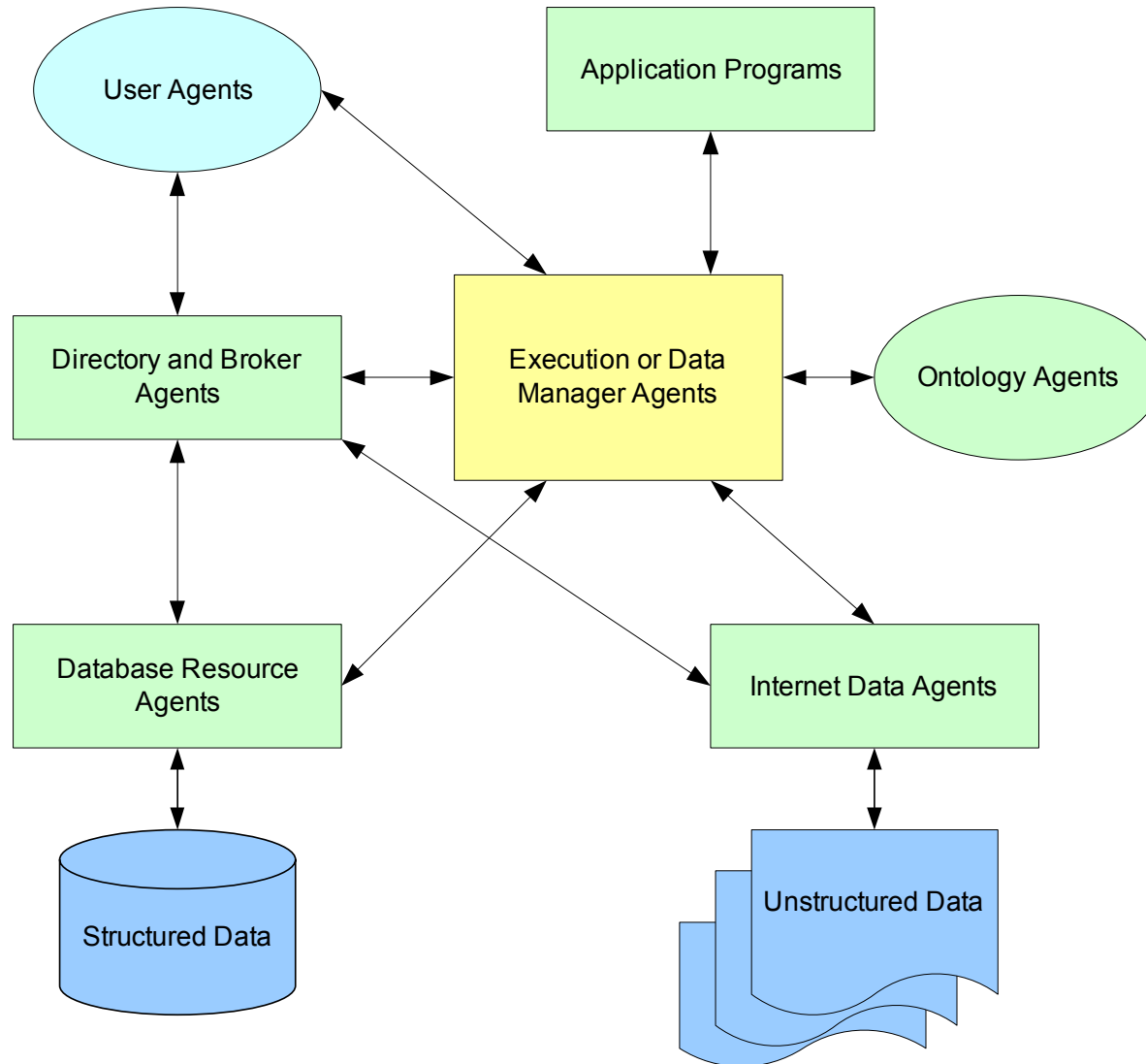
Challenges of MAS (1)

- Organization
 - Distribution of control among agents
 - Finding others (e.g., through directories)
 - Consistency maintenance, and reconciliation of conflicts among agents
 - Enforcing organizational rules
- Representation
 - Tasks, goals, etc.
 - Environment
 - Other agents
- Coordination
 - Description,
 - Decomposition,
 - Distribution of tasks among agents
 - Application: Business process modeling and enactment

Challenges of MAS (2)

- Communication
 - Agent communication languages
 - Interaction protocols
 - Semantics and pragmatics
- Service Selection
 - Whom to *trust*
 - Modeling others
 - Protocols for
- Methodologies for developing large systems
 - Abstractions

(de facto) Standard Agent Types



Name Service

- A multiagent architecture requires scalable, symbolic name resolution
- Alternative naming protocols
 - FIPA
 - LDAP
 - Jini
 - CORBA Naming Service
 - JNDI

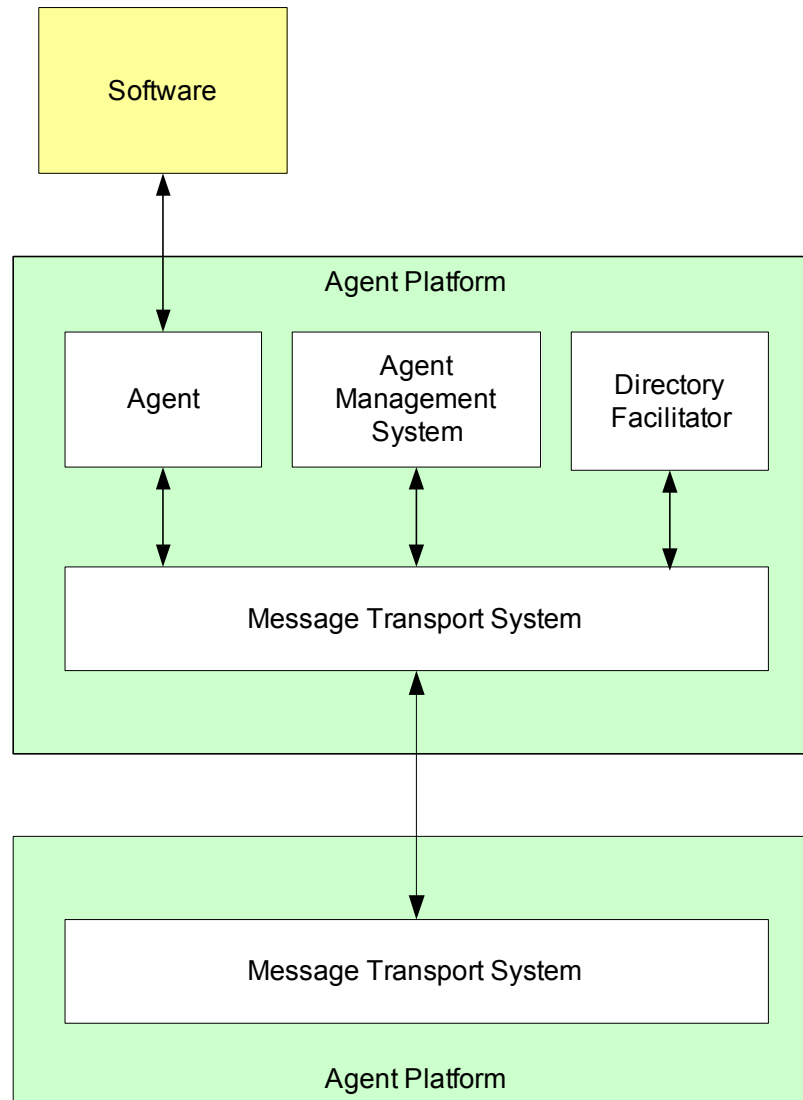
Directory Service

- Simple yellow-page service
- Registered agents advertise their services by providing their name, address, and service description
- Agents request recommendations for available services (provided by other registered agents or services)
- A simple database-like mechanism that allows agents to
 - insert descriptions of the services they offer
 - query for services offered by other agents
- Brokerage, recruitment and mediation services are not provided by Directory Service
- UDDI for Web services

Brokerage Service

- Cooperates with a Directory Service
- An agent requests the Brokerage Service to **recruit** one or more agents who can provide a service
- Brokerage Service uses knowledge about the requirements and capabilities of registered agents to
 - Determine the appropriate agents to which to forward a request for a service
 - Negotiates with the agents to determine a suitable set of service providers
 - Potentially learn about the properties of the responses
 - example: Brokerage agent determines that advertised results from agent X are incomplete and seeks a substitute for X

Agent Management System: 1



Agent Management System: 2

- Handles the creation, registration, location, communication, migration and retirement of agents.
- **White pages**
 - Agent location, naming and control access services
- **Yellow pages**
 - Service location and registration services, which are provided by the Directory Facilitator (DF)
- **Agent message transport services**

FIPA-Compliant Agent Frameworks

- FIPA is the Foundation for Intelligent Physical Agents, with website at www.fipa.org
- Specifies standards for heterogeneous, interoperating agent-based systems.
- Some of the popular, FIPA-compliant agent frameworks used for designing multiagent systems:
 - FIPA-OS, <http://fipa-os.sourceforge.net/>
 - JADE, <http://sharon.cse.it/projects/jade/>
 - Zeus, <http://zeus.enhydra.org/>

Consistency Maintenance across Services

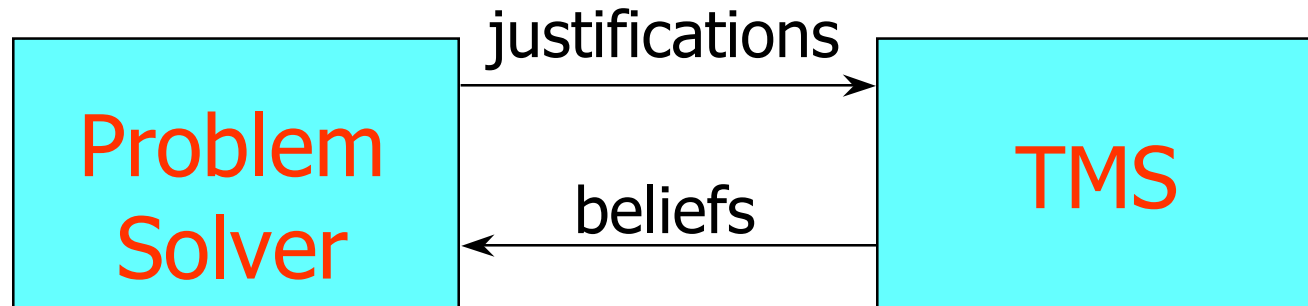
A truth maintenance system (TMS)

- performs a form of propositional deduction
- maintains justifications and explains the results of its deductions
- updates beliefs incrementally when data are added or removed
- uses its justifications to perform dependency-directed backtracking

TMSs are important because they

- deal with atomicity
- deal with the frame problem
- lead to efficient search

Architecture of TMS-Based Agent



- The problem solver represents domain knowledge in the form of rules, procedures, etc. and chooses what to focus on next
- The TMS keeps track of the current state of the search for a solution. It uses constraint satisfaction to maintain consistency in the inferences made by the problem solver

Knowledge Base Integrity

- *Stability*: believe everything justified validly; disbelieve everything justified invalidly
- *Well-Foundedness*: beliefs are not circular
- *Logical consistency*: logical contradictions do not exist
- *Completeness*: a system will find a consistent state if it exists, or report failure

Problems arise when knowledge is distributed

Degrees of Logical Consistency

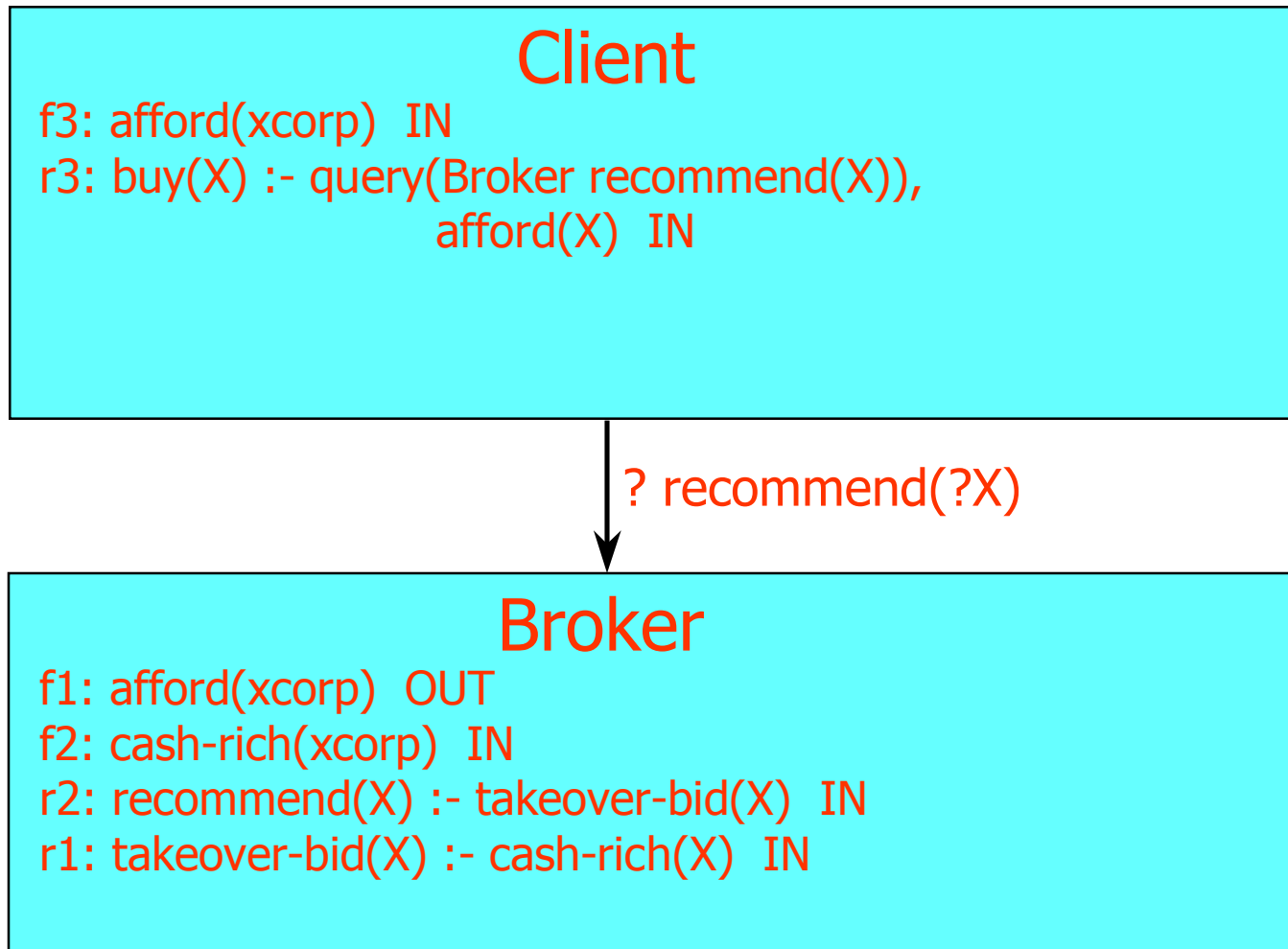
- *Inconsistency*: one or more agents are inconsistent
- *Local Consistency*: agents are locally consistent
- *Local-and-Shared Consistency*: agents are locally consistent and all agents agree about shared data
- *Global Consistency*: agents are globally consistent

The DTMS maintains local-and-shared consistency and well foundedness

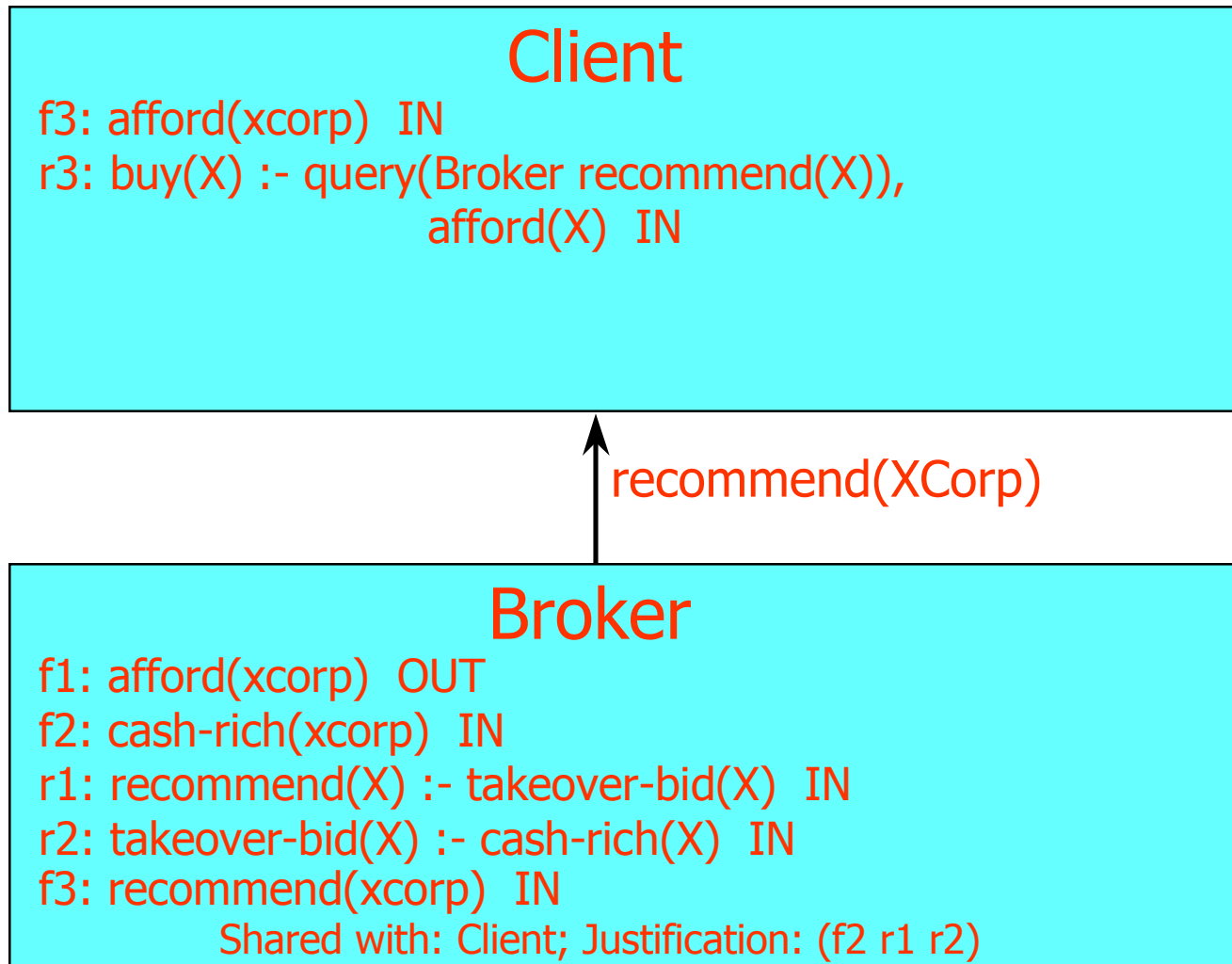
Distributed TMS

- Each agent has a justification-based TMS
- Each datum can have status OUT, IN (valid local justification), or EXTERNAL. A shared datum must be IN to one of the agents that shares it
- When a problem solver adds or removes a justification, the DTMS
 - Unlabels data based on the changed justification
 - Labels all unlabeled shared data
 - Chooses labels for remaining unlabeled data; if this fails, it backtracks by unlabeled additional data and iterating

Cooperative Service: 1



Cooperative Service: 2



Cooperative Service: 3

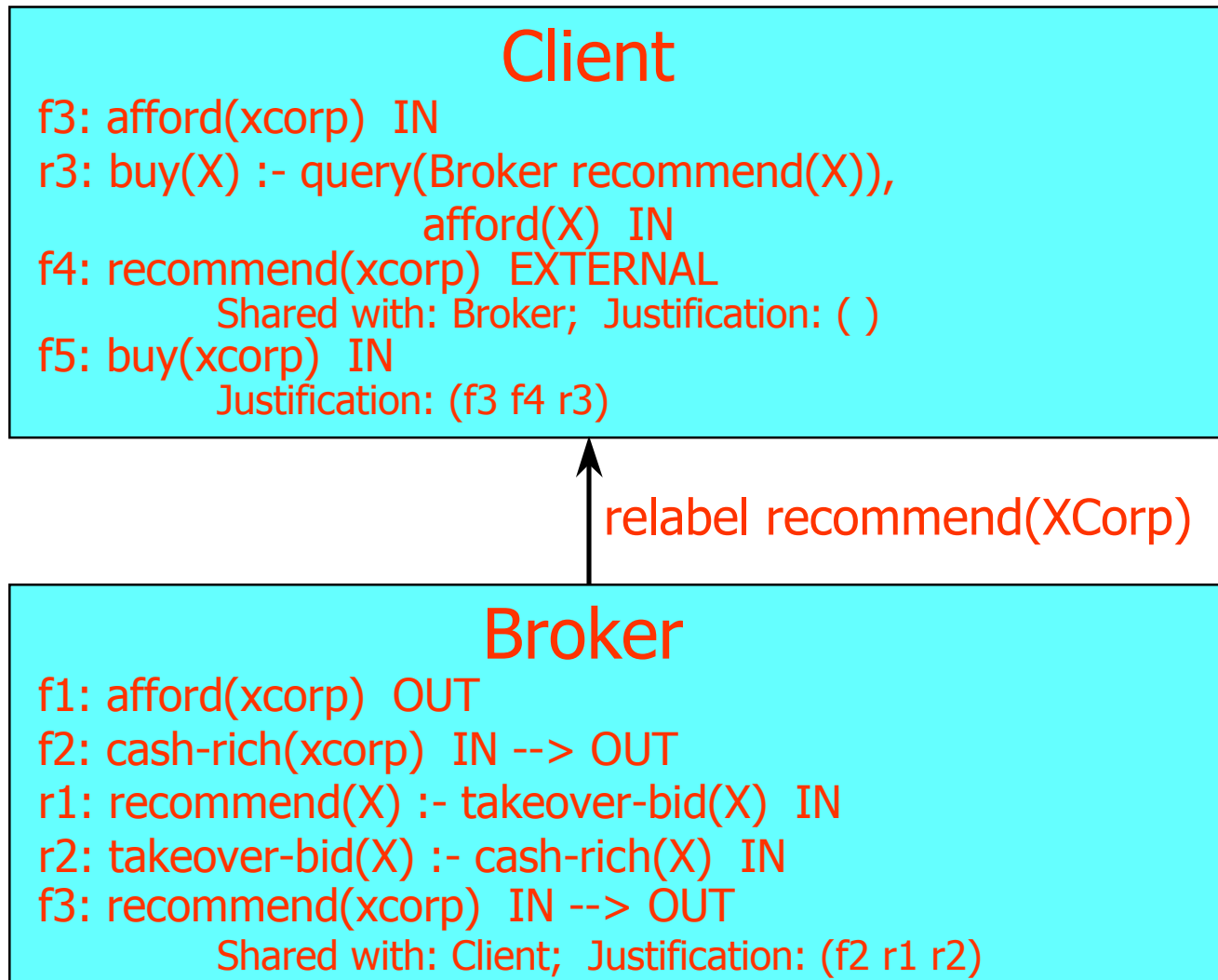
Client

f3: afford(xcorp) IN
r3: buy(X) :- query(Broker recommend(X)),
 afford(X) IN
f4: recommend(xcorp) EXTERNAL
 Shared with: Broker; Justification: ()
f5: buy(xcorp) IN
 Justification: (f3 f4 r3)

Broker

f1: afford(xcorp) OUT
f2: cash-rich(xcorp) IN
r1: recommend(X) :- takeover-bid(X) IN
r2: takeover-bid(X) :- cash-rich(X) IN
f3: recommend(xcorp) IN
 Shared with: Client; Justification: (f2 r1 r2)

Cooperative Service: 4



Cooperative Service: 5

Client

f3: afford(xcorp) IN
r3: buy(X) :- query(Broker recommend(X)),
 afford(X) IN
f4: recommend(xcorp) OUT
 Shared with: Broker; Justification: ()
f5: buy(xcorp) OUT
 Justification: (f3 f4 r3)

Broker

f1: afford(xcorp) OUT
f2: cash-rich(xcorp) OUT
r1: recommend(X) :- takeover-bid(X) IN
r2: takeover-bid(X) :- cash-rich(X) IN
f3: recommend(xcorp) OUT
 Shared with: Client; Justification: (f2 r1 r2)

Modeling Other Agents

- Becoming aware of each other
 - Exchanging messages
 - Monitoring the environment
 - Self-awareness
- Social conventions
 - Too many agents to be modeled individually
- Learning about the organization
 - Roles and their dynamics

Modeling Other Agents

- Model the other party like yourself
 - Change as differences arise
 - Deal with one kind of representation
 - Same inference mechanism
- Intentional stance
 - Ascribe cognitive concepts
 - Personal assistant
 - Models the user's intentions
 - Tasking: What to do rather than how to do it