Commitments for Business Processes:

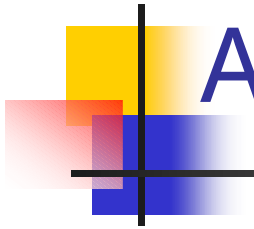
Overview of Concepts, Protocols, Machines, Compliance

Munindar P. Singh
Department of Computer Science
North Carolina State University

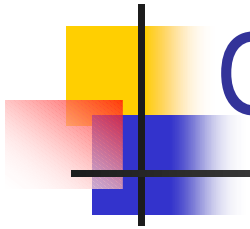
Essential Properties of Business Processes

- Autonomy.
- Heterogeneity.
- Exceptions.
- Opportunism.

Agent and Multiagent Systems

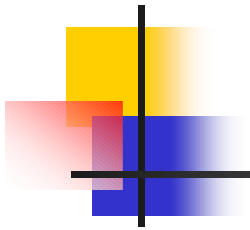


- *Agents* cover a wide range of behavior and functionality.
- An agent is an active computational entity that
  - Has a persistent identity.
  - Perceives, reasons about, and initiates activities in its environment.
  - Communicates with other agents.
  - Enters into complex relationships with other agents.
- These features enable agents to participate in open systems as service providers and consumers.



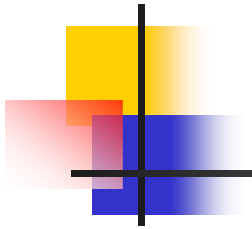
Commitments

- An agent's commitment to another agent:
  - Is a directed obligation.
  - Arises within a well-defined scope or *context*, which is itself a MAS.
  - Manipulable (including revocable) with restrictions.
- Enable coherent agent interactions by capturing the meaning behind the interactions.
- *Claim:* 90% of all business data reflects commitments [statistics made up **J**].



Outline

- History, briefly:
  - AI, communication, mentalism.
  - Introduced 1991, applied late 1990s.
- Commitments and spheres thereof.
- Contracts.
- Commitment protocols and machines.
- Compliance.
- Directions.



Dynamic Organizations

Whenever agents come together dynamically and have structure to their interactions.

- *Abstractly*, organizations
  - Consist of roles
    - Requiring certain capabilities and commitments.
    - Offering certain authorities.
  - Require commitments among the roles.
  - Support commitments among the roles.
- *Concretely*, organizations
  - Consist of agents.
  - Acting coherently.

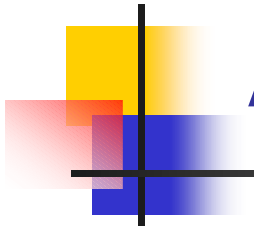
Sphere of Commitments

SoCom: an organization that provides the context or scope of commitments among agents.

- Conceptually, the SoCom
  - Serves as a witness or adjudicator for the commitment.
  - Helps validate commitments and test for compliance.
  - Offers compensations to undo members' actions.

Manipulating Commitments

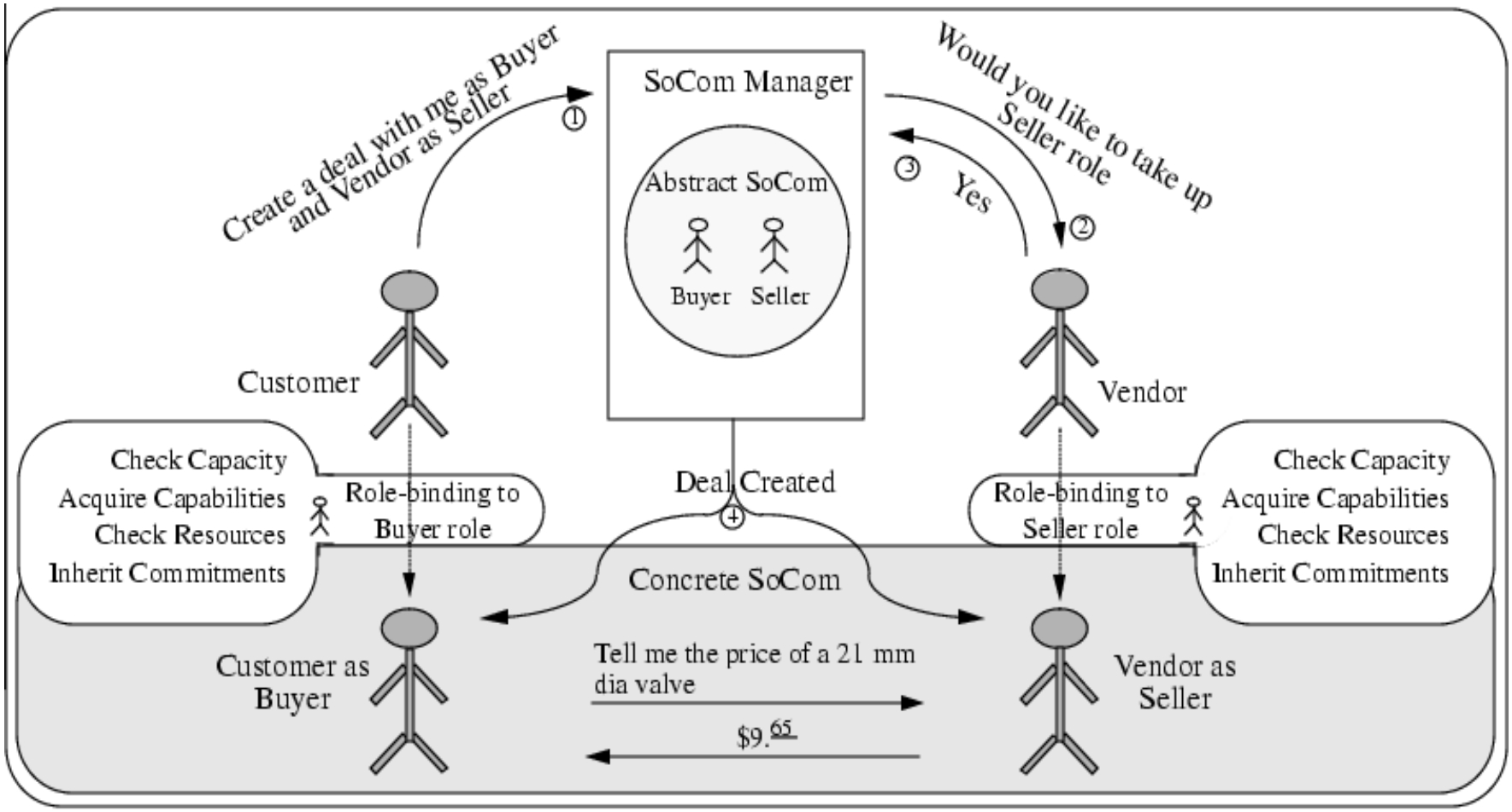
- Operations on commitments:
  - Create.
  - Discharge (satisfy).
  - Cancel.
  - Release (eliminate).
  - Delegate (change debtor).
  - Assign (change creditor).
- Metacommitments:
  - Constrain the manipulation of commitments.
  - Fall into a small number (dozen) of patterns for common business process scenarios.



Applying SoComs

- *Example:* buyer and seller roles with appropriate
  - Capabilities, e.g., requests they can honor.
  - Commitments, e.g., validity of their price quotes.
- To adopt a role, an agent must have the capabilities and adopt the commitments.
- System needs are architecture and tools for
  - Discovery.
  - Compliance.
  - Designing the right agents.

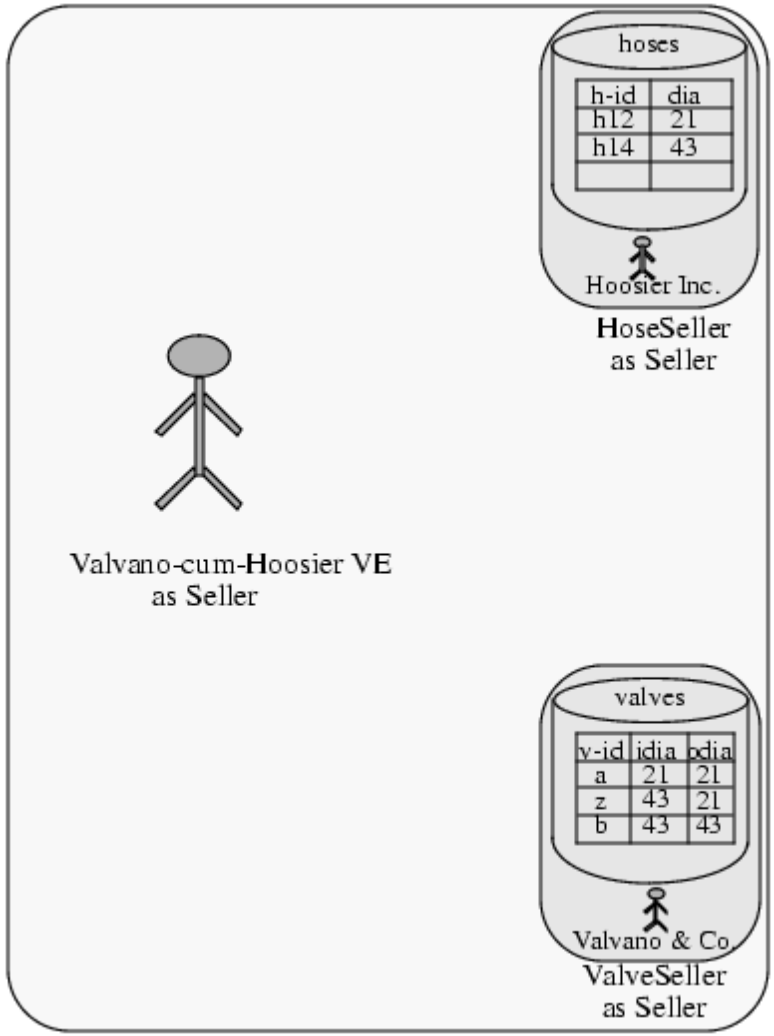
Binding Agents to Roles



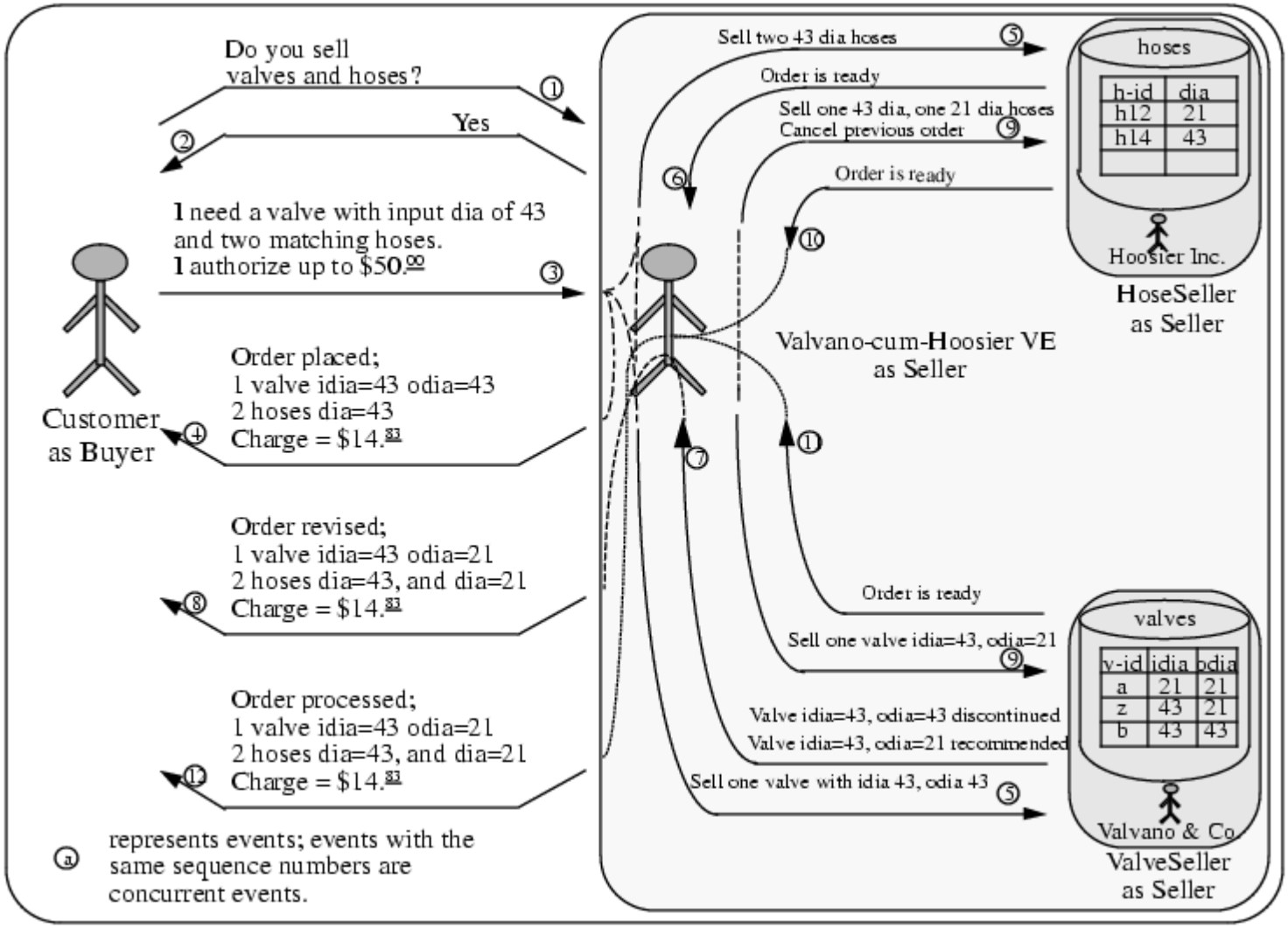
SoComs provide the context for concepts represented & communicated.

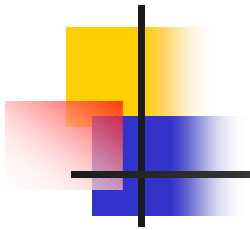
Virtual Enterprises (VE)

- Two sellers come together into a SoCom called VE (implemented, e.g., with a new proxy agent).
- Example of VE's commitments:
  - Notify on change.
  - Update orders.
  - Guarantee the price.
  - Guarantee delivery date.



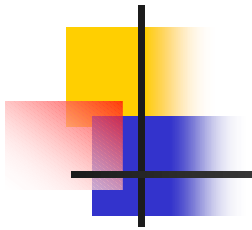
A Selling VE


Do you sell valves and hoses?
Yes
I need a valve with input dia of 43 and two matching hoses.
I authorize up to $50.00
Customer as Buyer
Order placed;
1 valve idia=43 odia=43
2 hoses dia=43
Charge = $14.83
Order revised;
1 valve idia=43 odia=21
2 hoses dia=43, and dia=21
Charge = $14.83
Order processed;
1 valve idia=43 odia=21
2 hoses dia=43, and dia=21
Charge = $14.83
a represents events; events with the same sequence numbers are concurrent events.
Sell two 43 dia hoses
Order is ready
Sell one 43 dia, one 21 dia hoses
Cancel previous order
Order is ready
Valvano-cum-Hoosier VE as Seller
Order is ready
Sell one valve idia=43, odia=21
Valve idia=43, odia=43 discontinued
Valve idia=43, odia=21 recommended
Sell one valve with idia 43, odia 43
hoses
h-id dia
h12 21
h14 43
Hoosier Inc.
HoseSeller as Seller
valves
v-id idia odia
a 21 21
z 43 21
b 43 43
Valvano & Co.
ValveSeller as Seller



Patterns

- Common patterns of commitments emerge, e.g.,
  - Policies to notify and renotify.
  - Policies to entertain requests, updates, from other roles.
- Patterns help design good systems.
- Agent skeletons can be generated from selected patterns that a role is expected to follow.



Outline

- History, briefly:
  - AI, communication, mentalism.
  - Introduced 1991, applied late 1990s.
- Commitments and spheres thereof.
- Contracts.
- Commitment protocols and machines.
- Compliance.
- Directions.

Hohfeldian Concepts: 1

Hohfeld discovered that “right” is used ambiguously and proposed a uniform terminology to distinguish its various uses.

- Sixteen concepts result:
  - Four main concepts.
  - Their correlates.
  - Their negations.
  - Their negations’ correlates.
- All two-party notions.

Hohfeldian Concepts: 2

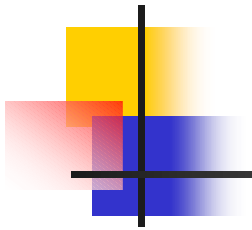
- *Claim-duty*: the claims a party has on another.
- *Privilege-exposure*: freedom from the claims of another agent – dual of claim.
- *Power-liability*: when an agent can change the claim-duty relationship of another agent – ability to create and manipulate commitments involving others.
- *Immunity-disability*: freedom from the power of another agent – dual of power.

Commitments for Contracts

Commitments express the Hohfeldian concepts.

Importantly, commitments are

- Public (unlike beliefs and intentions).
- Can be used as the basis for compliance.
- Contracts apply between parties, in a context.
- Other approaches are:
  - Single-agent focused, e.g., deontic logic.
  - Don't handle organizational aspects of contracts.
  - Don't accommodate manipulation of contracts.



Outline

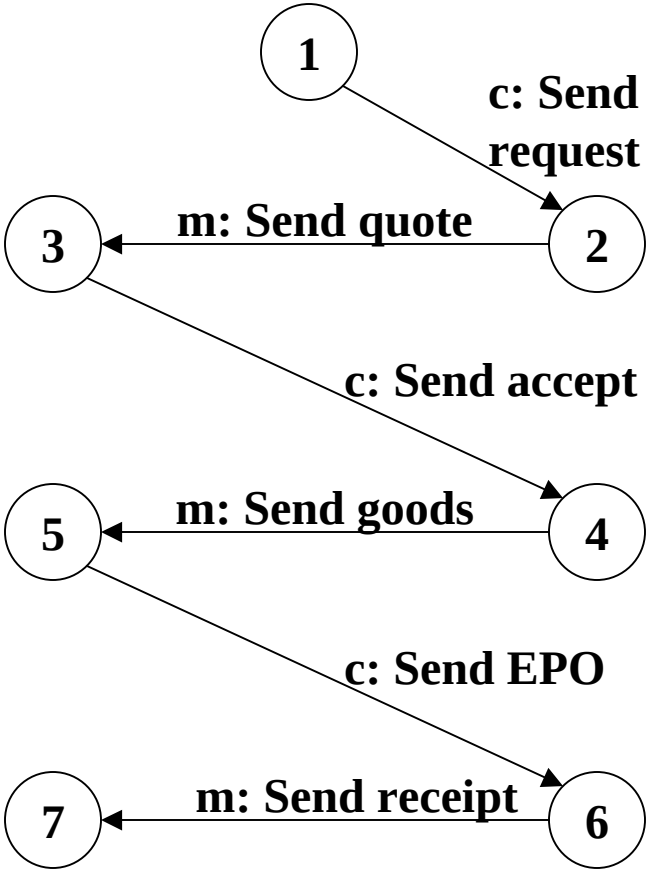
- History, briefly:
  - AI, communication, mentalism.
  - Introduced 1991, applied late 1990s.
- Commitments and spheres thereof.
- Contracts.
- Commitment protocols and machines.
- Compliance.
- Directions.

Commitment Protocols

- Protocols enable open systems to be constructed.
- Interaction protocols expressed in terms of
  - Participants' commitments.
  - Actions for performing operations on commitments (to create and manipulate them).
  - Constraints on the above captured in temporal logic.

Example: NetBill

FSM Representation



Some variations:

- The merchant may start the protocol by sending a quote.
- The customer may send an accept prior to offer.
- The merchant may send the goods prior to accept.

These variations are not allowed by the FSM.

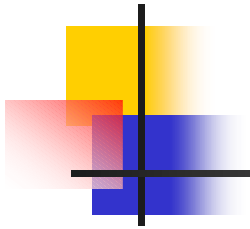
Capturing Meaning

Atomic propositions:

- *request*: the customer has requested a quote.
- *goods*: the merchant has delivered the goods.
- *pay*: the customer has paid the agreed amount.
- *receipt*: the merchant has delivered the receipt.

Metacommitments:

- *promiseGoods*: $C_m(\text{accept} \Rightarrow \text{goods})$
- *accept*: $C_c(\text{goods} \Rightarrow \text{pay})$
- *promiseReceipt*: $C_m(\text{pay} \Rightarrow \text{receipt})$
- *offer*: promiseGoods $\wedge$ promiseReceipt

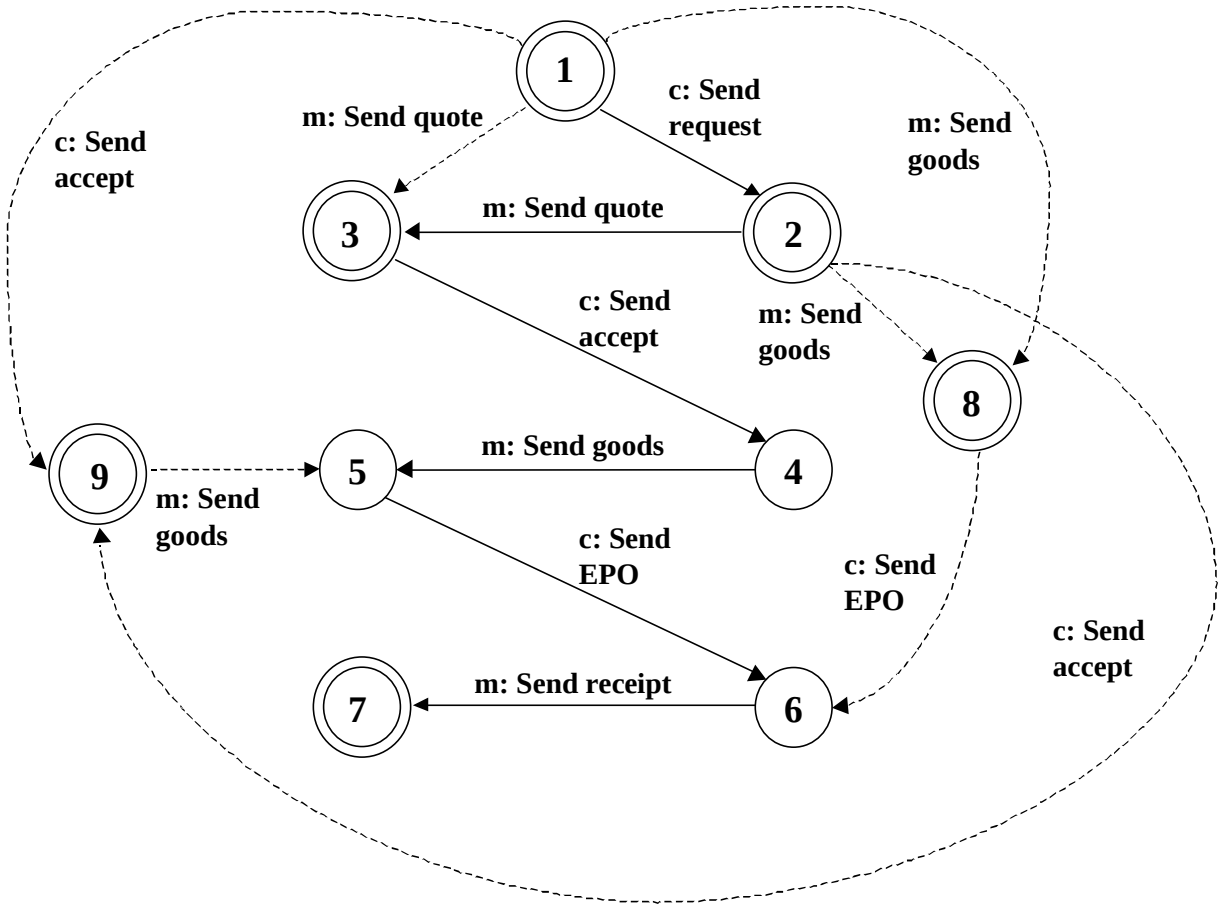


Reasoning

When we represent meaning, we can reason about how an agent should act given the protocols in which it is participating.

- **Planning:** generate *protocol runs* that satisfy the given protocols
- **Opportunism**: Skip unnecessary states.
- **Composition**: Combine protocols through common commitment states.
- **Factoring**: Substitute a subprotocol for another (e.g., a sophisticated negotiation protocol for accepting quote) as long as both protocols produce semantically equivalent computations.

NetBill Enhanced by CMs



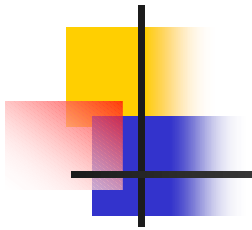
Meanings:

1. true
2. request
3. offer
4. $C_m$goods $\wedge$ accept $\wedge$ promiseReceipt
5. goods $\wedge$ $C_c$pay $\wedge$ promiseReceipt
6. goods $\wedge$ pay $\wedge$ $C_m$receipt
7. goods $\wedge$ pay $\wedge$ receipt

8. goods Ù promiseReceipt

9. accept

Final state: No open commitments remain.

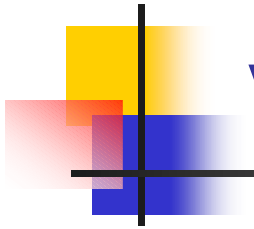


Outline

- History, briefly:
  - AI, communication, mentalism.
  - Introduced 1991, applied late 1990s.
- Commitments and spheres thereof.
- Commitment protocols and machines.
- Contracts.
- Compliance.
- Directions.

Compliance with Protocols

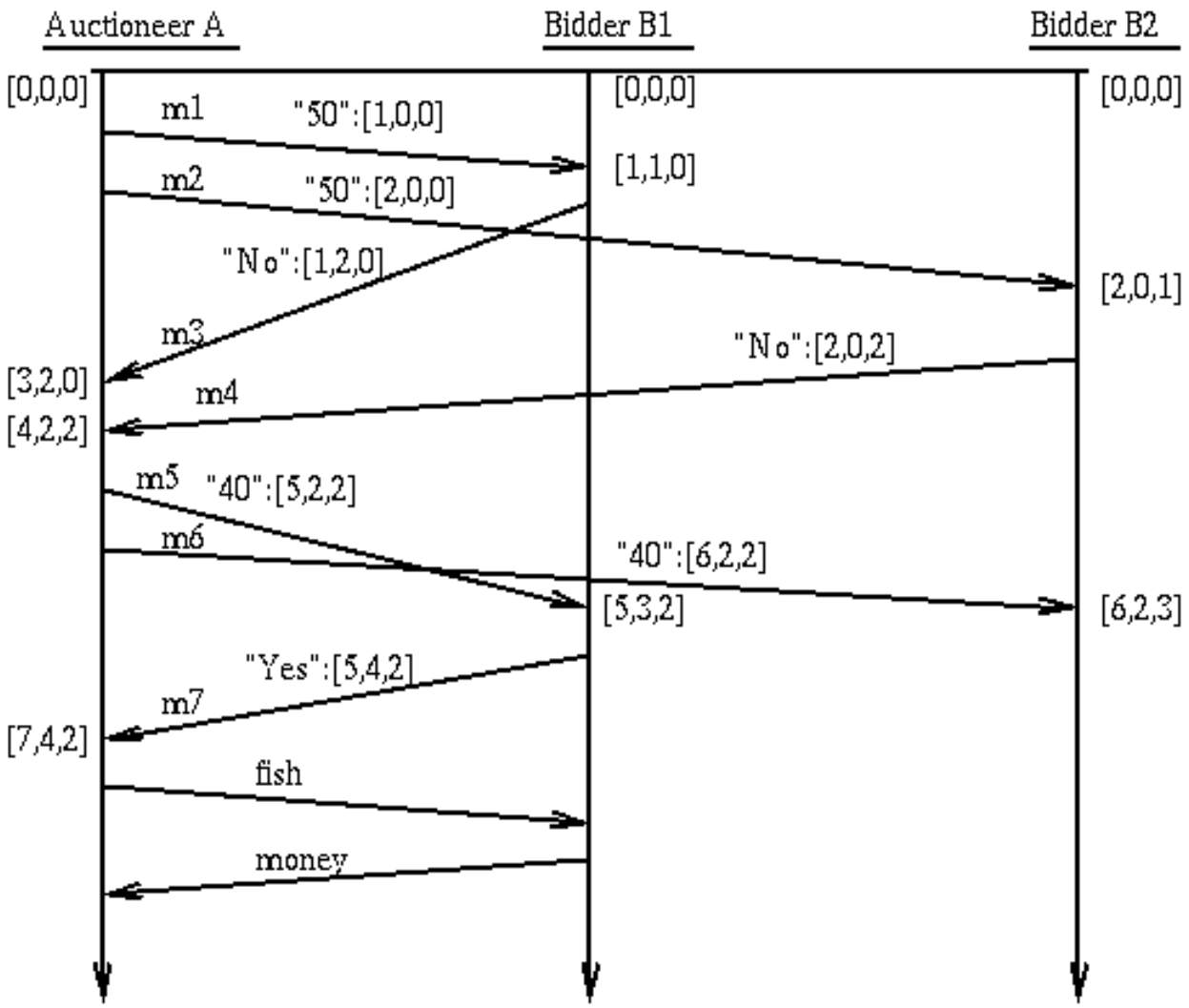
- Commitment machines deal with designing agents to obey protocols flexibly.
- But in open multiagent systems, agents are contributed by different vendors and serve different interests.
- How can we check if the agents *comply* with the specified protocols?
  - Coordination aspects: traditional techniques.
  - Commitment aspects: representations of the agents' commitments in TL.



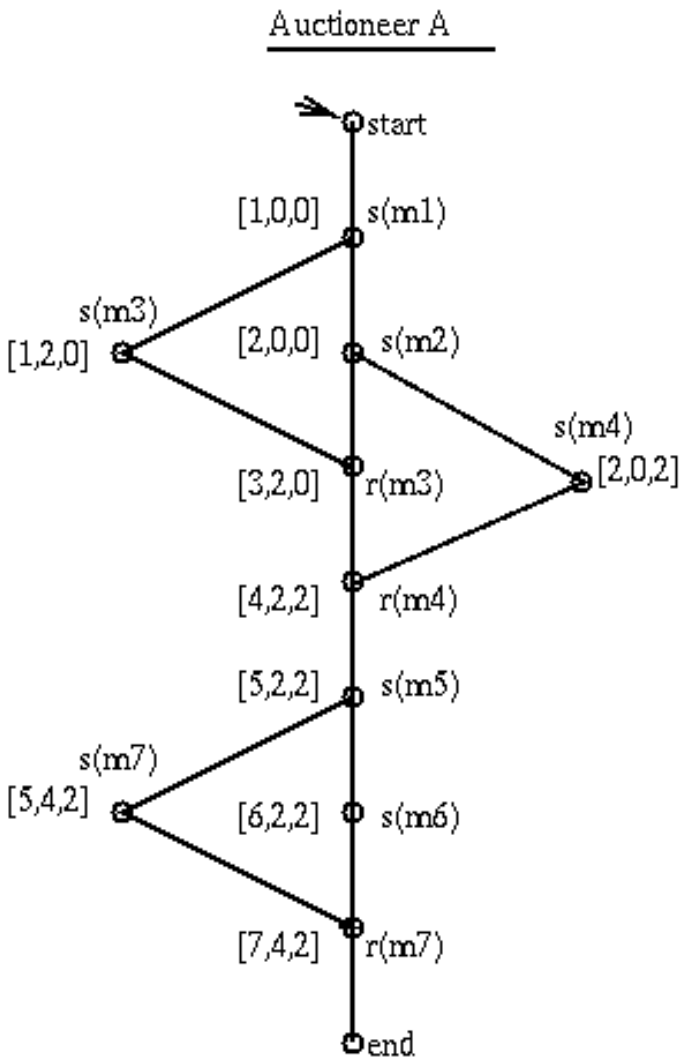
Verifying Compliance

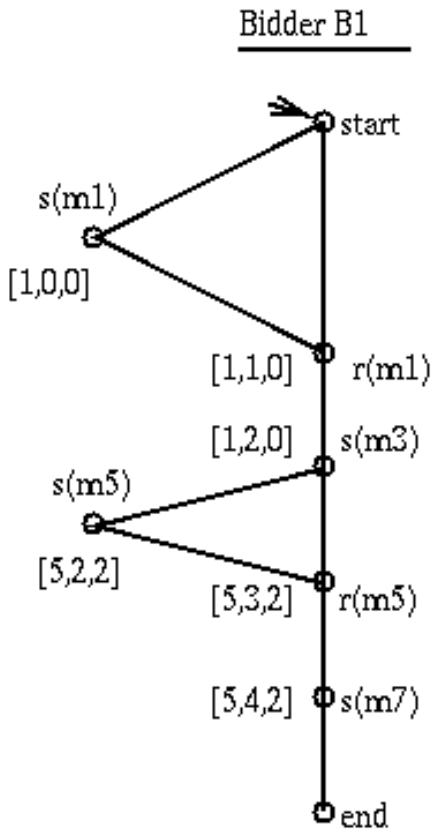
- Specification of commitment protocols:
  - Models based on *potential causality.*
  - Protocol:
    - Commitments based on branching-time TL.
    - Domain-specific propositions and actions
    - Skeletons of roles essential for coordination
- Run-time verification:
  - Respects design autonomy.
  - Uses TL model-checking.
  - Local verification based on observed messages.

Fish-Market Sample Execution



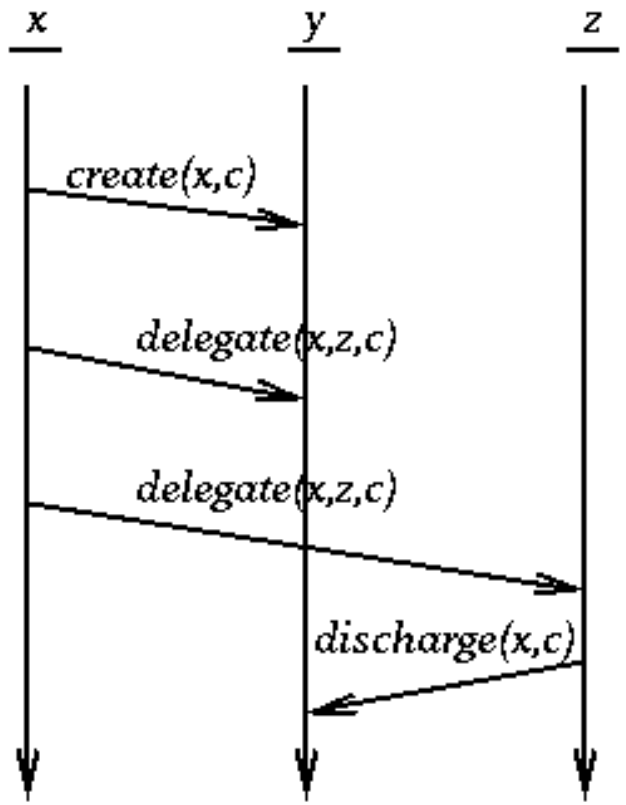
Fish-Market Local Observations

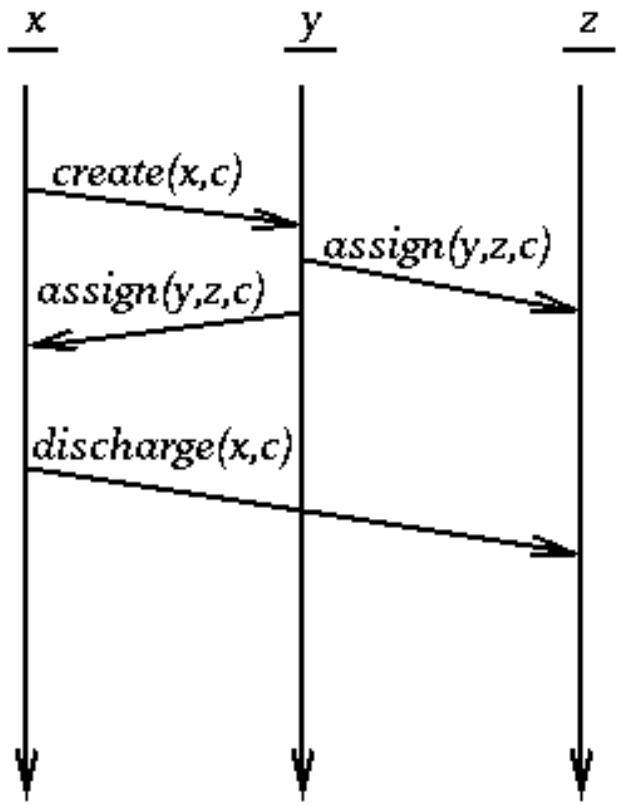




Message Patterns for Commitment Operations

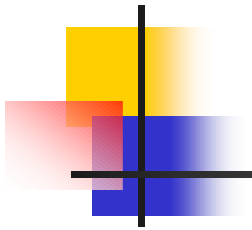
Ensure that information about commitment operations flows to the right parties, to enable local decisions.





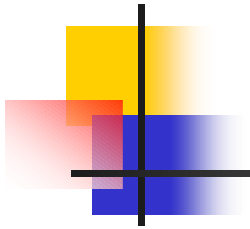
Run-Time Compliance Checking

- An agent can keep track of
  - Its pending commitments.
  - Commitments made by others that are not satisfied.
- It uses this local model to see if a commitment has been violated.
- An agent who benefits from a commitment can always determine if it was violated.



Outline

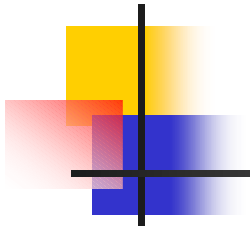
- History, briefly:
  - AI, communication, mentalism.
  - Introduced 1991, applied late 1990s.
- Commitments and spheres thereof.
- Commitment protocols and machines.
- Contracts.
- Compliance.
- Directions.



Directions

- Concepts and design:
  - Simplified commitment and policy capture.
  - Manipulation of commitments based on varying context groups.
- Protocols and machines:
  - Richer models of inference about commitments.
- Compliance:
  - Determination of compliance under different cases of system architecture and information flow.
  - Relationship to trust among participants.

Influence industry practice and standards.



Evaluation

- Control Flow: Excellent graph primitives, some with iteration
- Organizational abstraction: Not supported
- Conversations: Modeled as scripts (graphs), but not flexible
- Cooperation: Not supported
- Exception handling: Only low level, not semantic