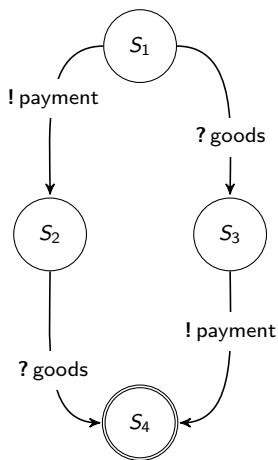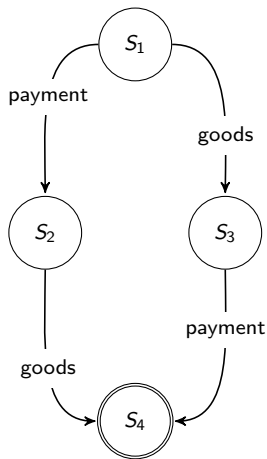# Protocols and Roles
Protocol: shared view; roles: each local view
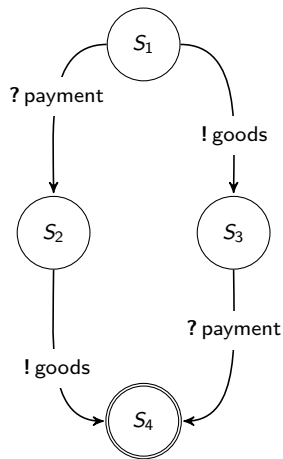

The Buyer Role

The Trade Protocol

The Seller Role

Communication Protocols
Protocols define how the agents ought to communicate with one another

- ▶ A protocol is a modular, potentially reusable specification of the interactions between two or more entities

- ▶ Defining a protocol helps ensure *interoperability*, i.e., being able to work together

- ▶ Communities of practice define appropriate protocols
    - ▶ RosettaNet: manufacturing
    - ▶ Foreign exchange transactions: TWIST
    - ▶ Health care: HL7

- ▶ What are the main requirements for protocol specifications?

- ▶ How can we specify a communication protocol?

Exercise: Identify Agents and Communications Protocols

Setting: healthcare service engagement—an annual physical

Engineering with Agent Communication

- ▶ Begin from a protocol
- ▶ Generate role skeletons (or endpoints) from the protocol
- ▶ For each role skeleton, implement one or more agents who realize ("flesh out") it
    - ▶ Map each skeleton to a set of incoming and outgoing messages and the changes each message induces in the local state
    - ▶ Implement methods to process each incoming message
    - ▶ Send messages allowed by the protocol
- ▶ Challenge: Generating role skeletons that ensure interoperation
    - ▶ Not trivial when a protocol involves more than two roles
    - ▶ The protocol must be such that such skeletons are derivable from it

Protocols Promote Autonomy
A protocol should not constrain an agent's interactions beyond what is essential for the application

▶ Each agent is free to act as it pleases
▶ Protocols specify allowed ordering and occurrence of interactions
  ▶ Should do so minimally
  ▶ Control flow specifications unnecessarily limit agent autonomy

Protocols Promote Heterogeneity

▶ A protocol enables interoperation by specifying
  ▶ Schemas of messages exchanged
  ▶ Meanings of messages, which determine the *state* of the interaction
▶ Correctness cannot depend upon the agents' internal reasoning
▶ Intelligence of the agents is irrelevant for a communication protocol
▶ Control flow specifications unnecessarily couple agent designs at a low level
▶ A protocol
  ▶ Becomes the standard to which agents are implemented
  ▶ Defines the extent of heterogeneity: the agents can be heterogeneous with regard to everything else

Traditional Software Engineering Approaches

- ▶ Don't emphasize autonomy and heterogeneity
- ▶ Emphasize operational details
  - ▶ Leave open the formulation of the message syntax (good)
  - ▶ Disregard the meanings of the messages (bad)
- ▶ Traditional representations capture occurrence and ordering of messages, mostly in procedural terms
  - ▶ Finite state machines (procedural)
  - ▶ State diagrams or statecharts (procedural); generalize FSMs
  - ▶ Sequence diagrams (procedural)
  - ▶ Petri nets (procedural)
  - ▶ Pi-calculus (procedural)
  - ▶ Temporal logic (declarative)
- ▶ Dependence upon low-level details leads to interoperation being fragile to irrelevant modifications

UML Sequence Diagrams
Used by FIPA (Foundation for Intelligent Physical Agents)

Combine constructs from Message Sequence Charts (MSCs) and FIPA

▶ Procedural constructs: sequencing (default), alternative, parallel, loop

▶ Highlights benefits of a protocol
  ▶ Clear roles
  ▶ Decouples agents from one another

▶ Ignores message meanings
  ▶ FIPA offers a semantics for message types
  ▶ But no application-specific meanings

FIPA Request Interaction Protocol
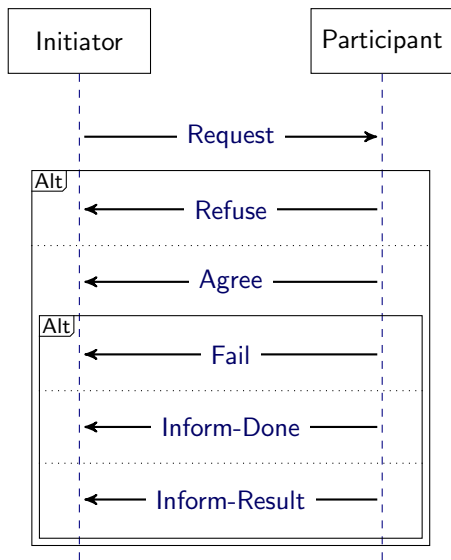
- ▶ Roles: INITIATOR and PARTICIPANT
- ▶ Messages
  - ▶ *request*, *agree*, *refuse*, *failure*, an *inform-done*, or an *inform-result*
- ▶ Ordering and occurrence
  - ▶ *refuse* or an *agree*
  - ▶ *agree* followed by a detailed response: *failure*, *inform-done*, or *inform-result*
  - ▶ *agree* is required only if the INITIATOR asked for a notification

Agent Programming for Protocols
Java Agent Development Framework or JADE is a leading platform

▶ *Behavior:* a specification of a role skeleton that characterizes important events such as the receipt of specified messages and the occurrence of timeouts

▶ Implement an agent according to a behavior by defining the methods it specifies as callbacks

  ▶ Define the handlers for any incoming methods

Role Conformance
Developing an agent that conforms to a role specification

- ▶ Produce a role skeleton from a protocol specification
- ▶ Publish role skeletons along with the protocol specification
- ▶ An agent who plays (and hence implements) a role fleshes out the skeleton
- ▶ Challenge: determine constraints on the messages an agent playing a role can receive and send and constraints on how the local representation of the social state should progress
- ▶ Software vendors produce agent implementations
- ▶ An agent vendor does not reveal internal details but specifies what roles the agent can play
- ▶ Conformance means that an agent can play a particular protocol role
- ▶ Challenge: identifying formal languages for specifying roles along with algorithms for checking conformance

Protocol Refinement and Aggregation
Apply traditional conceptual modeling relations to communication

- ▶ Refinement: how a concept refines another (*is-a* hierarchy)
- ▶ Aggregation: how concepts are put together into composites (*part-whole* hierarchy)
- ▶ Well-understood for traditional object-oriented design and supported by programming languages (as type checking)
- ▶ Nontrivial for communication protocols (especially, refinement)
- ▶ Challenge: produce a generalized theory and associated languages and tools for refinement and aggregation of *meaning-based* protocols (to be introduced)

Choreography

A specification of the message flow among the participants from a neutral perspective

- ▶ Decentralized nature
- ▶ Contrasts with *orchestration*, a description of how one party controls all others
- ▶ Somewhat like a sequence diagram written textually
- ▶ Proposed approaches: WS-CDL and ebBP
- ▶ Shortcomings
    - ▶ No encoding of the meaning
    - ▶ Focus on ordering and occurrence
    - ▶ Make private actions of agents visible
    - ▶ Lack support for composition of choreographies