OWL-P: Processes = Protocols + Policies

Munindar P. Singh

(*Students:* Amit K. Chopra, Nirmit V. Desai, Ashok U. Mallya)

singh@ncsu.edu

Department of Computer Science

North Carolina State University

http://www.csc.ncsu.edu/faculty/mpsingh/

Problem, Solution, Approach

- *Problem:* Modeling and enacting open business processes
  - Traditional approaches can’t handle autonomy, heterogeneity, dynamism
  - Incorporating context is essential
- *Solution:* Interaction is the key
  - New way of thinking geared toward open systems
- *Approach:* Protocols capture interaction
  - Software engineering: refine, aggregate protocols
  - Agents: flexible enactment
  - Compliance in the face of flexibility

Accomplishments: Intellectual

- Protocols provide interaction-centric modeling, leaving policies to participants
- Commitment semantics yield flexible modeling and enactment
- Theory of protocols supports reusability, refinement, and aggregation of interactions
- Students
  - Three PhD dissertations being supported
  - PhD dissertation involving autonomic service selection based on OWL for QoS being defended on Dec 6.

Accomplishments: Implementational

On SemWebCentral

- OWL-P as an OWL ontology (using SWRL)
  - Roles
  - Messages: content as propositions and commitments
  - Rules to describe messages and roles
- Protégé plugin for OWL-P protocol editor
- Protocol composer and skeleton generator
- Agent-based architecture layered on FIPA
- Rule-based policies that help agents satisfy their protocol roles

Accomplishments: Evangelical

- Papers, tutorials, panels, invited talks
- Trying to reach the software engineering community: well-received at OOPSLA
- Contact with IBM and HP
- Beginning project jointly with IBM on autonomic and agent based business process management
- New book: *Service-Oriented Computing: Semantics, Processes, Agents*
- IEEE Internet Computing track (2005) on *Service-Oriented Computing*

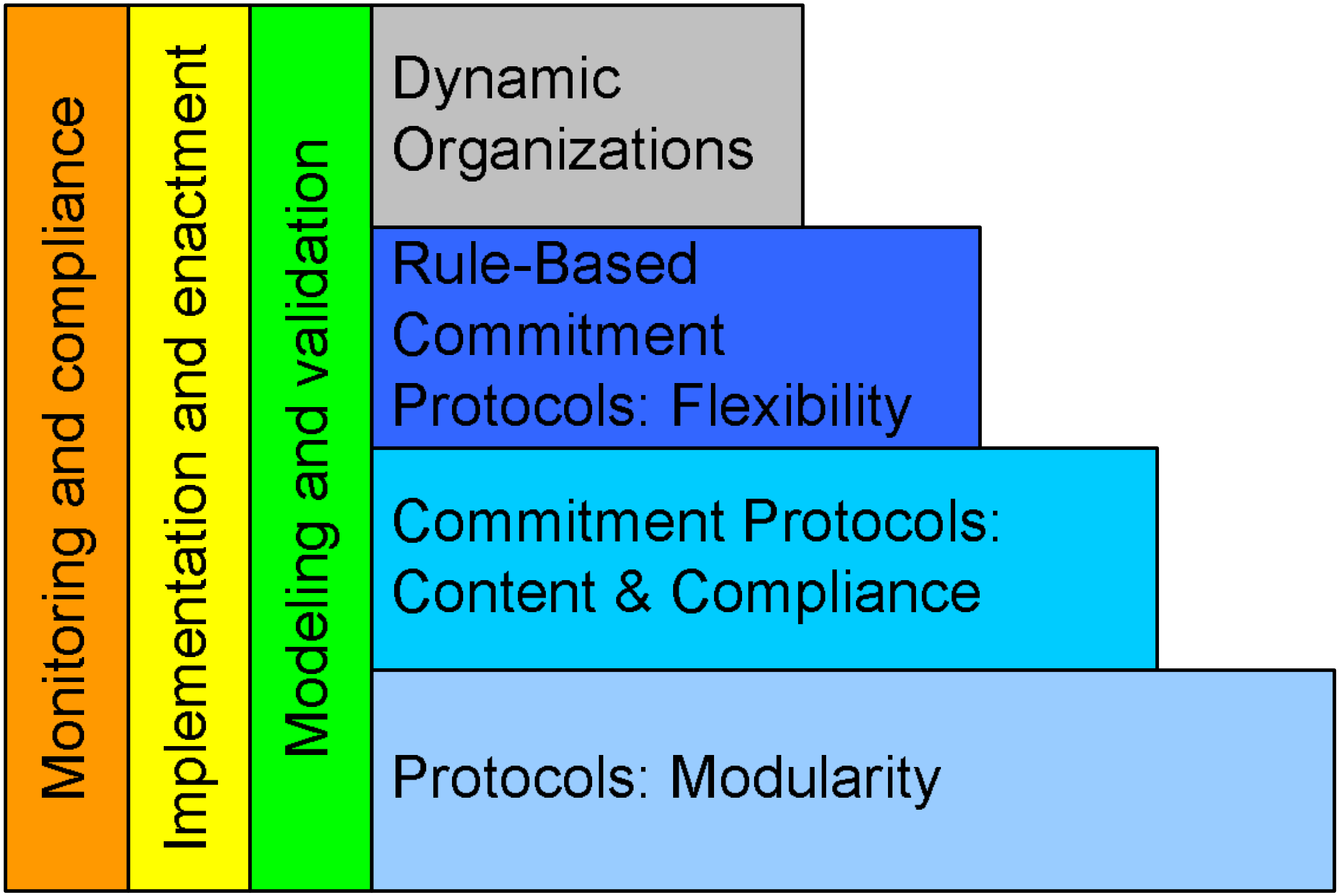
The Essential Tension

- *Reusability* requires
  - Context freedom
  - Encapsulation
- *Usability* (usefulness) requires
  - Context sensitivity
  - Varieties of context include organizations, laws, and the real world
- Main idea
  - Autonomy: components have a life of their own
  - Interactions are what matter

A Process is . . .

- *Orchestration:* a partial order of actions under the control of a central conductor
  - Akin to a workflow or flow in BPEL
- *Choreography:* an exchange of messages among participants
  - Akin to a conversation as described by WS-Chor
- *Collaboration:* a joint set of activities among business partners
  - Akin to real business; essential for SOAs

Emphases of Collaboration



Monitoring and compliance
Implementation and enactment
Modeling and validation
Dynamic Organizations
Rule-Based Commitment Protocols: Flexibility
Commitment Protocols: Content & Compliance
Protocols: Modularity

Innovations: 1

- *Protocols:* Conceptually decentralized, reusable, encapsulations of processes
- *Commitments:* Content for protocols
  - Support reuse via abstractions for *refinement* and *aggregation* of protocols
  - What the protocol should accomplish
  - What deviations are legitimate and what aren’t
  - Operational semantics for commitments

Innovations: 2

- *Rule-Based Reasoning:*
  - Expressing protocols flexibly
  - Accommodating context
  - Deciding specific actions by applying policies
- *Spheres of Commitment:*
  - Modeling organizations
  - Enacting protocols
  - Monitoring and verifying compliance
- Slogan: Processes $=$ Protocols $+$ Policies

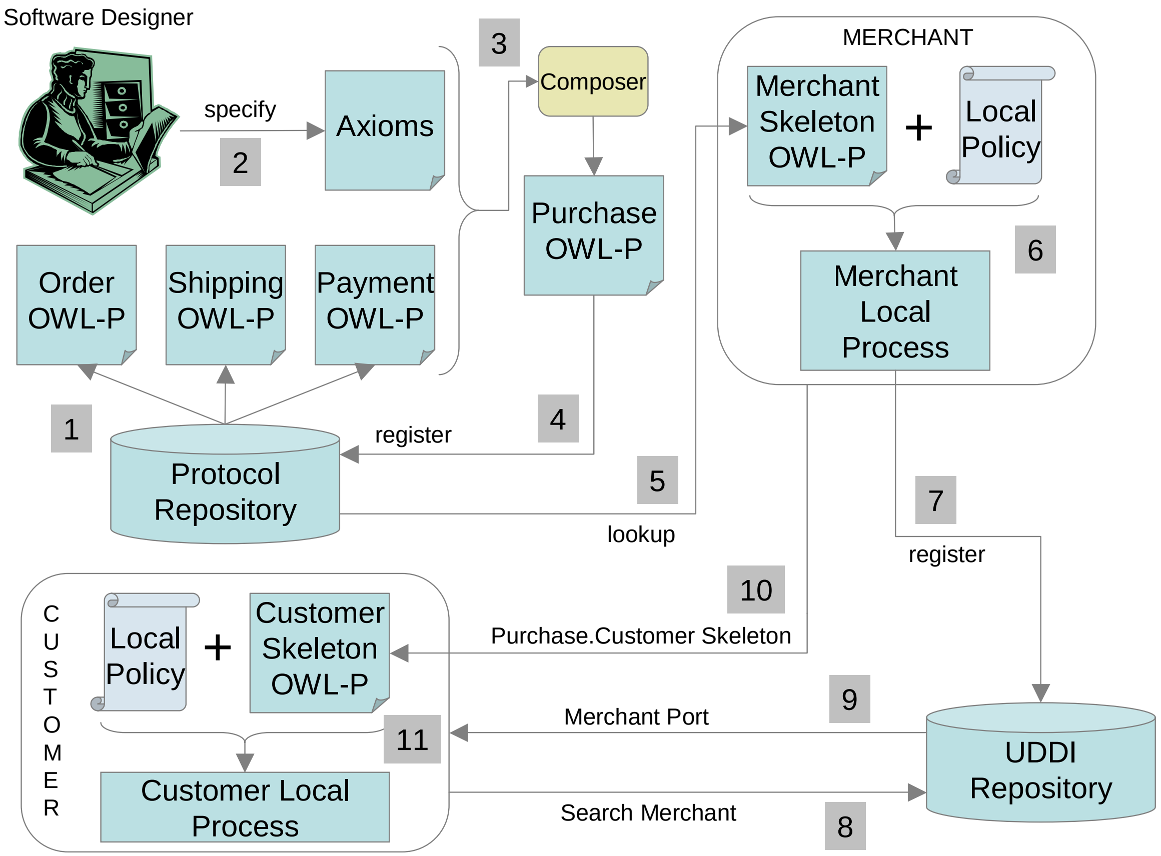
Trends and Assessment

- Increasing # of business protocols
  - IOTP, Escrow, SET, NetBill, . . .
  - RosettaNet: 107 Partner Interface Processes (PIPs)
  - ebXML Business Process Specification Schema (BPSS)
- Intended to be legally binding
- Generally highly limited: two party, request-response protocols
- No commitments; no formal semantics
- Limited support for modeling or enactment

Vision

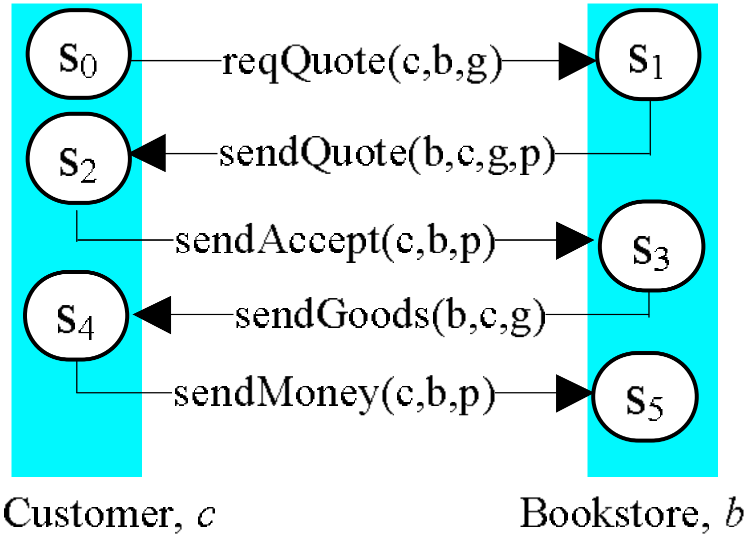
- *Target Audience:* Practitioners
- Formalization in the background
- *Engineering:* not full automation, but tools for
  - Modeling and validation of protocols
  - Modeling and validation of processes
  - Generation of software components
  - Enactment via Spheres of Commitment
  - Monitoring and compliance

Usage Scenario

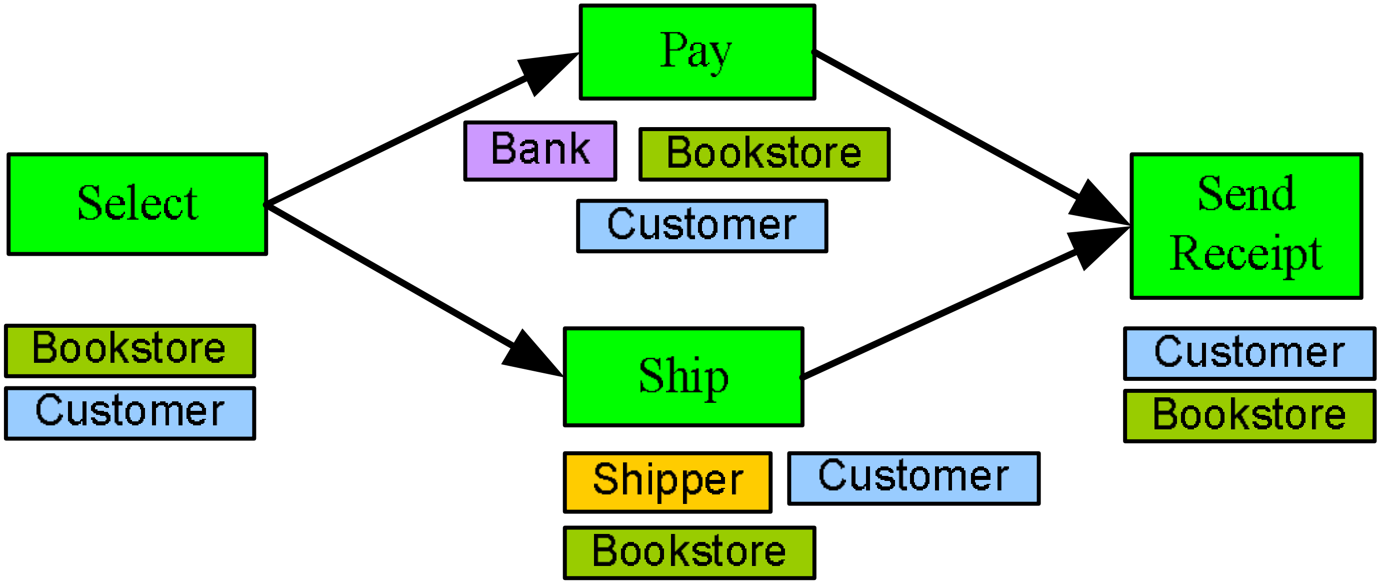


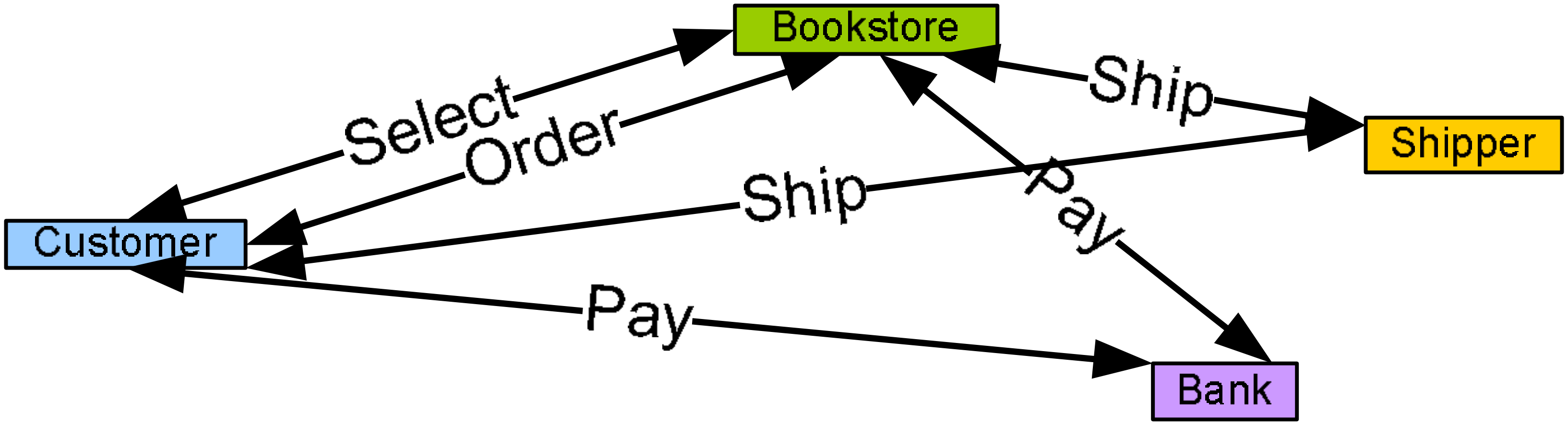
Simple Scenario and Example Run

- A customer (C) looks up a book at a vendor (B) and is quoted price and availability
- C orders the book from B
- B ships to C
- C pays B



Process View: Flow or Protocol





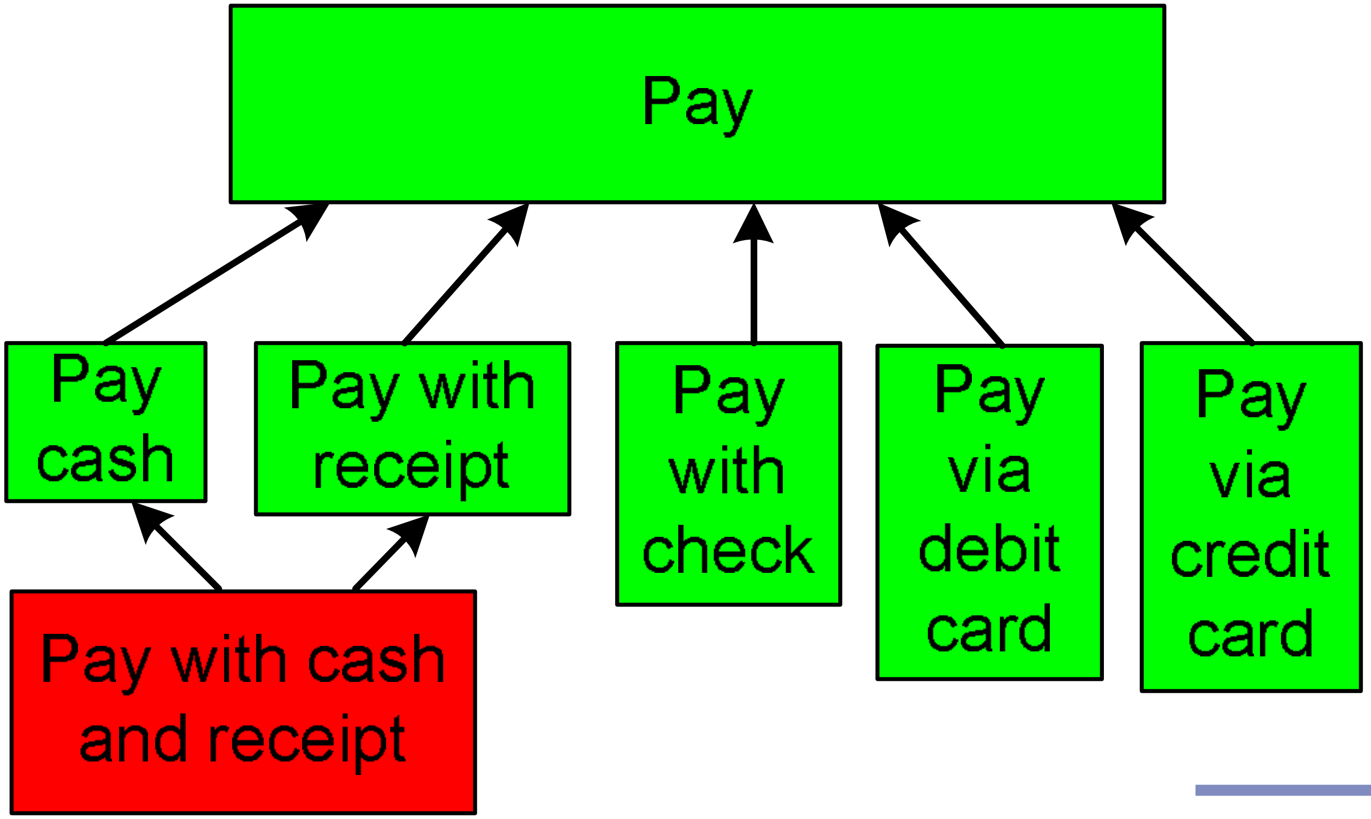
Challenges: Modeling

- *Refinement:* pay by credit card versus pay
- *Extensibility:* verify C's attributes, e.g., age
- *Adjustment:* receive payment before shipping; receive book before paying
- Alternative execution examples:
    - B arranges for a shipper (S) to deliver the book to C
    - C pays via bank (K)
    - Compose a process from the above

Refinement of Protocols

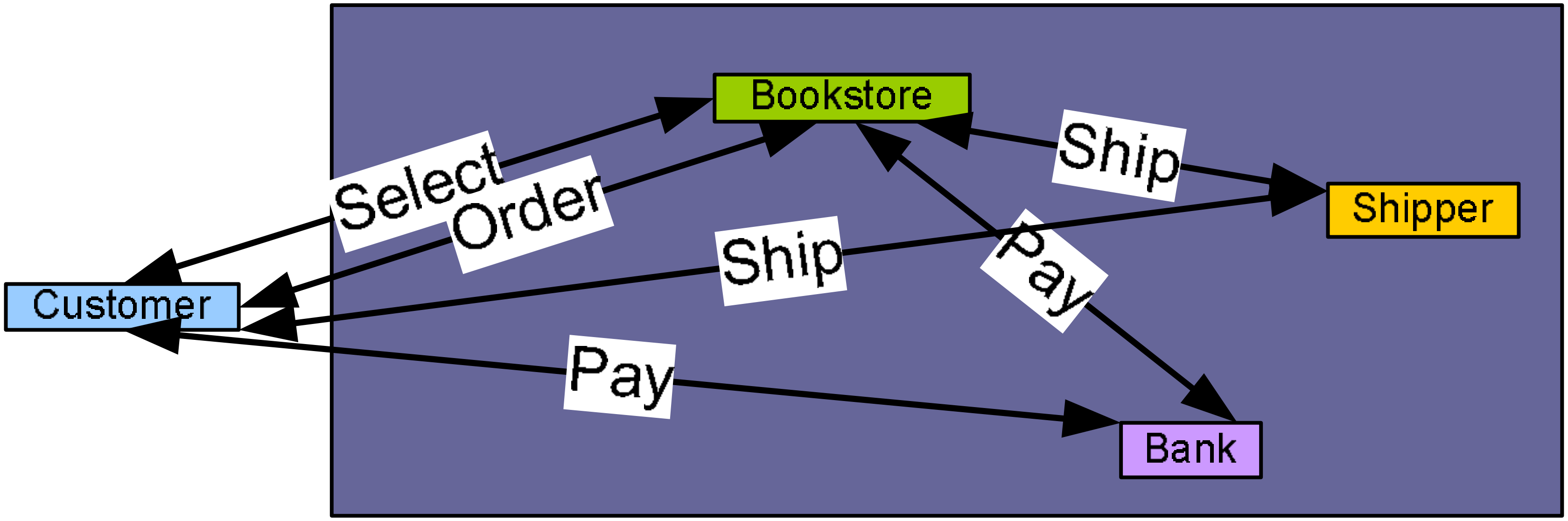
Selection criteria for protocols

- *Functional:* pay versus ship
- *Nonfunctional:* payer trusts payee or not

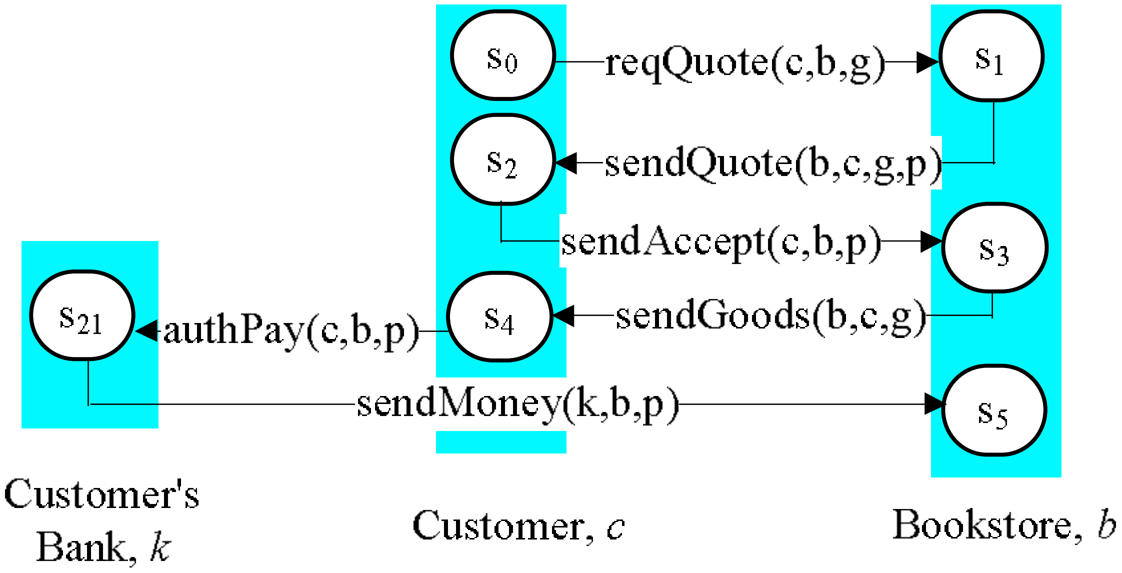


Aggregation of Protocols

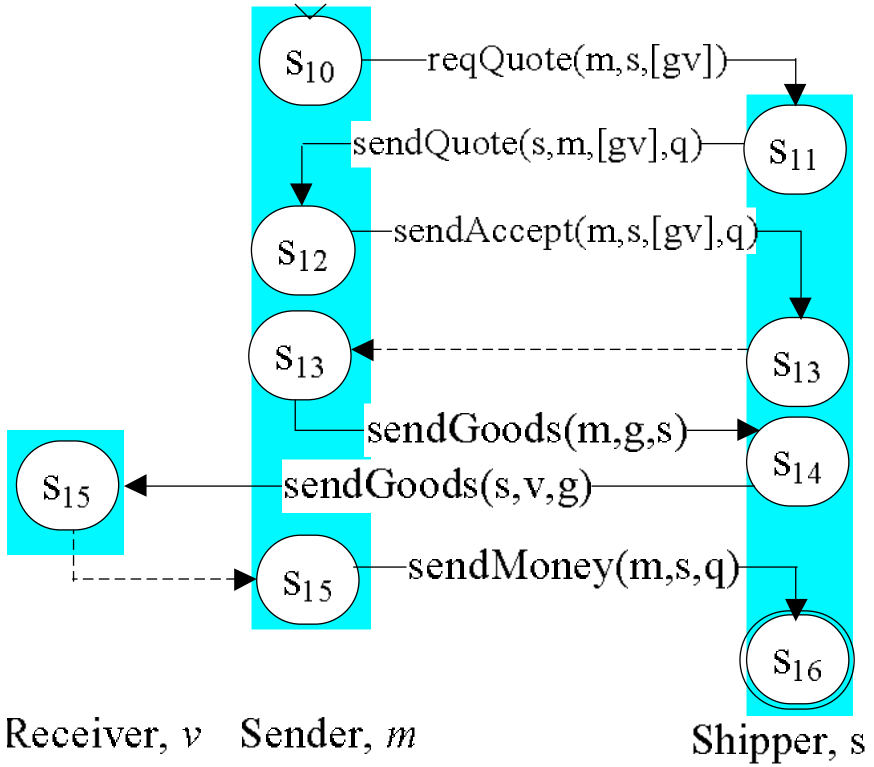
- A simplified protocol may be revealed to a give role
- Decisions could be taken internally but not exposed



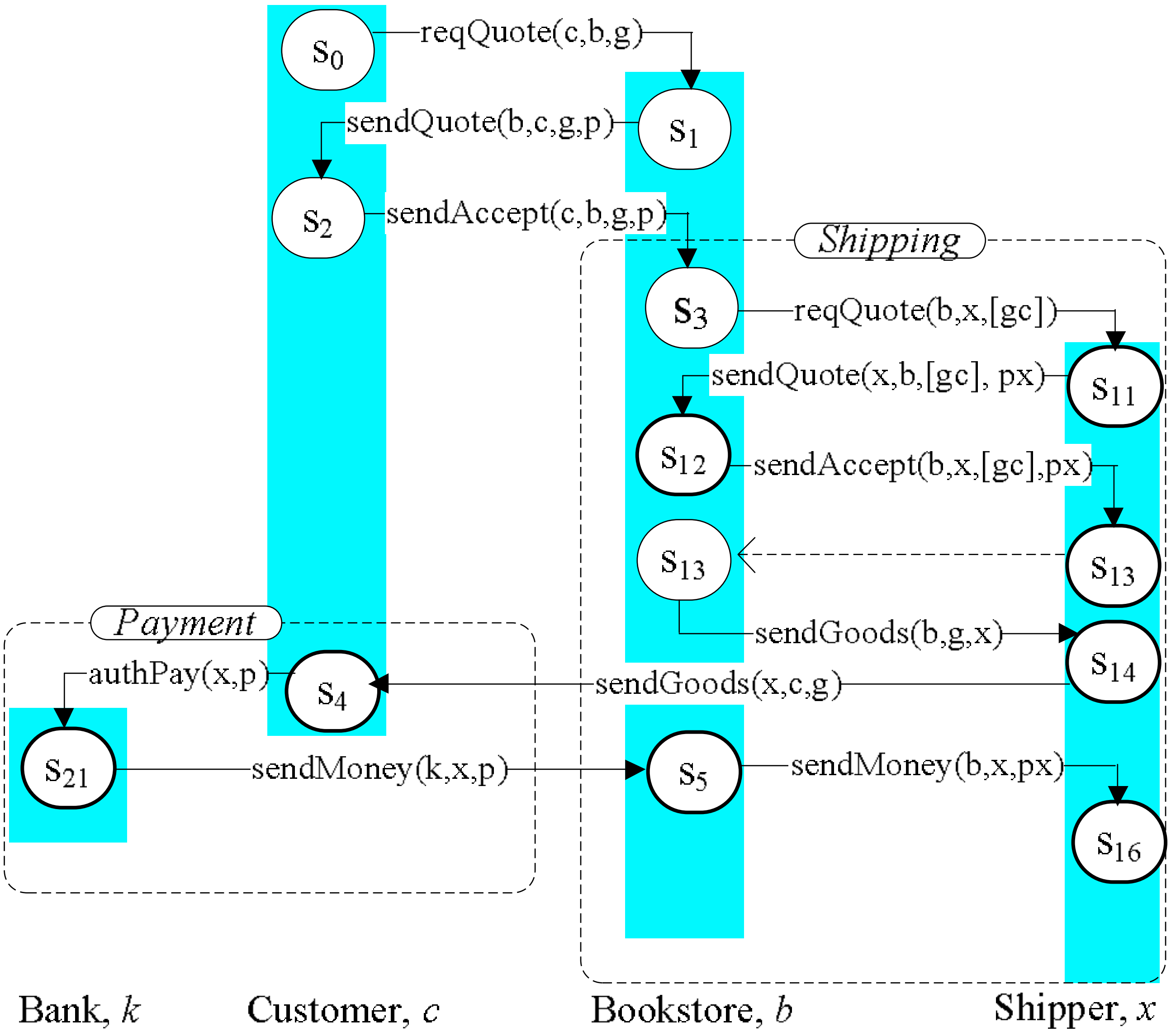
Example Run: Pay via Bank



Example Run: Shipper Protocol



Example Run: Composed Purchase

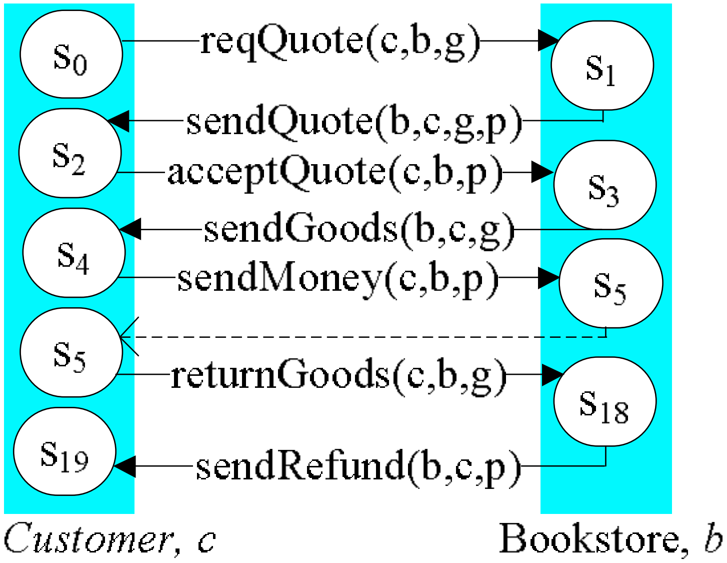


Challenges: Enactment

- *Behaving adaptively:* decide dynamically to ship before payment to trusted Cs
- *Handling exceptions*
  - External problems: cannot ship book
  - Context-sensitivity: not legal for kids
  - Detecting violations: no payment; book arrives damaged
  - Correcting violations: remind, complain, refund, . . .
- *Exploiting opportunities:* combine orders from same C

Example Run: Return and Refund

Example: Uniform Commercial Code (UCC) allows returns with refunds for goods that are received damaged



Processes = Protocols + Policies

- Operational patterns
  - Time outs, remind, garbage collect, . . .
  - Decisions to manipulate: delegate, assign, . . .
  - Enact protocols dynamically based on agent policies and context
- Transactional patterns
  - Induce transactional scopes
  - Apply retry, redo, undo (compensate) where appropriate
  - Enact via Spheres of Commitment

Remaining Work: Easy

Challenge	Remediation
Simplify protocol design for business	Libraries of composable protocols
Produce compliant agents	Refined methodology based on policies
Make up to date with Rules work and OWL-S	Freshen the work

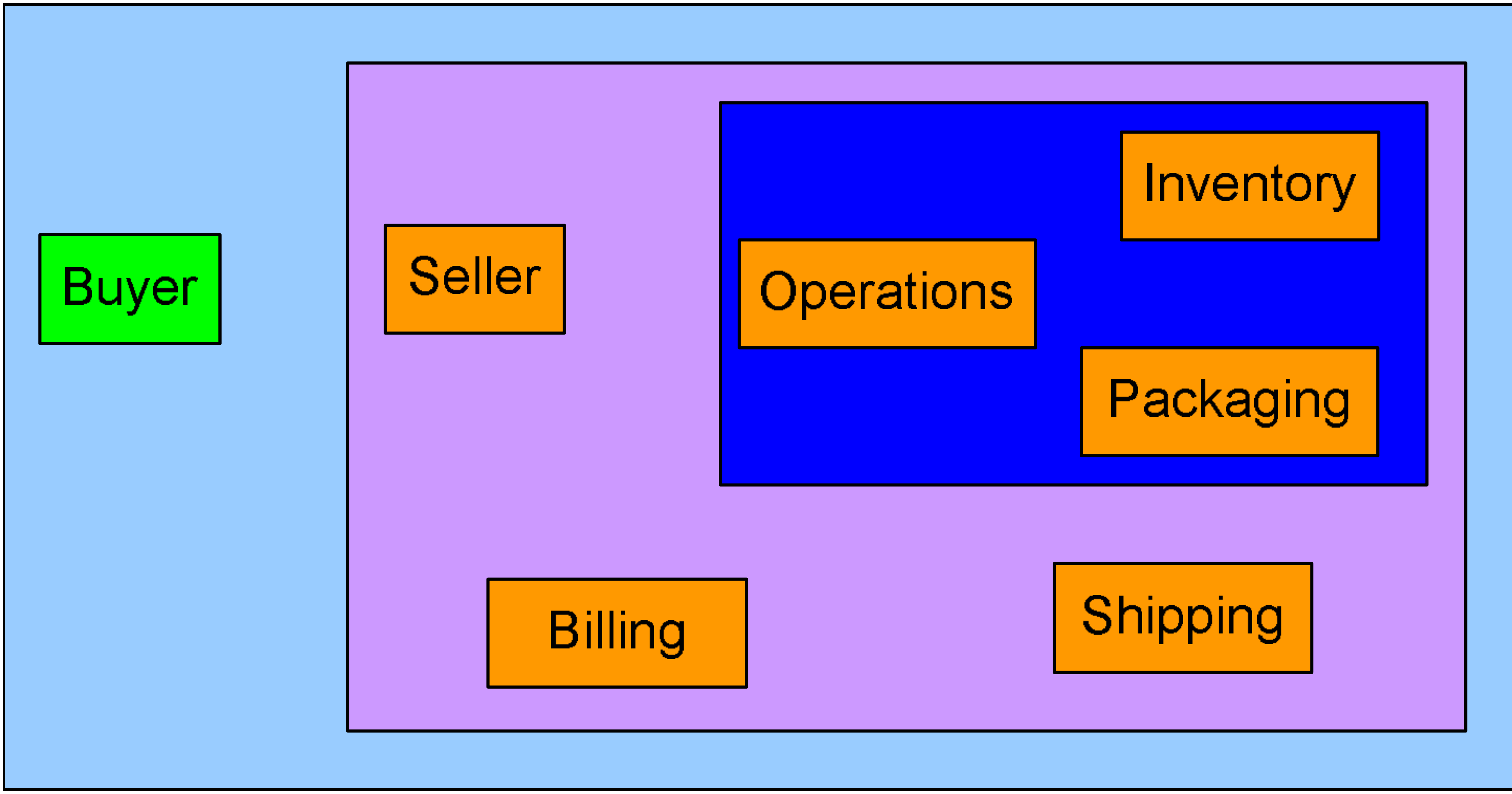
Remaining Work: Middlish

Challenge	Remediation
Formalize context	Develop operational semantics in π-calculus
Protocol compliance	Apply commitment semantics
Organizational, transactional exception modeling	Exploit Spheres of Commitment
Optimize role selection based on QoS	Incorporate service quality representations

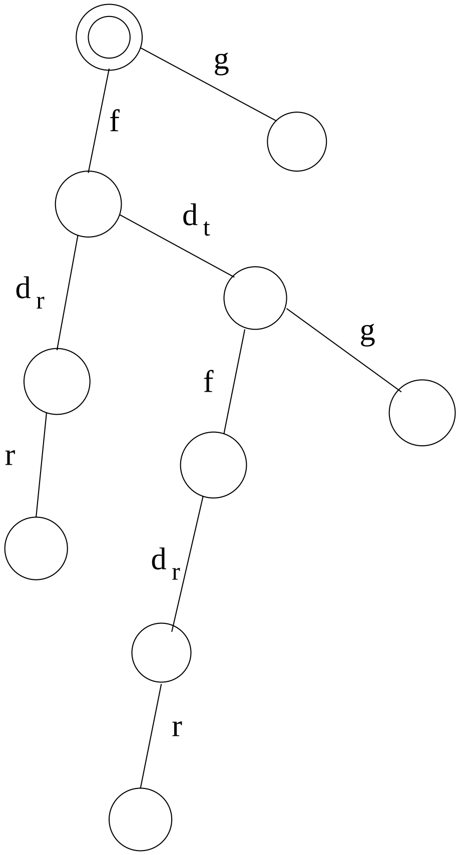
Papers on this Topic

- Recent papers in ICWS, AAMAS, OOPSLA, ICSOC address parts of the above vision
- Tutorials at WWW, AAMAS, OOPSLA
- Panels at WWW, AAMAS, ICWS
- "Agent Communication Languages: Rethinking the Principles." *IEEE Computer*, 31(12):40–47, Dec 1998
- "Reasoning About Commitments in the Event Calculus: An Approach for Specifying and Executing Protocols." *Annals Math & AI*, 42(1-3), 2004
- "Verifying Compliance with Commitment Protocols." *J. Autonomous Agents & MAS*, 2(3):217–236, Sep 1999

Spheres of Commitment



Contexts as Transformers


g
f
d t
d r
g
f
r
d r
r