The Evolution of IT

- **Applications:** Control of computations hidden in code; integration a nightmare
- **Workflows:** Control abstracted out; integration still difficult
- **Standards-driven orchestration:** Integration improved; limited support for autonomy
- **Messaging:** Integration simplified by MoM and transformations; limited support for autonomy
- **Choreography:** Model conversations over messages; limited support for autonomy
- **Governance:** Administer resources via interactions among autonomous parties

Technical Service

- Generally, an abstraction of a computational object
  - Traditional, as in web or grid services
  - Improved: Abstraction of a “capability”
- Well encapsulated, i.e., a black box
- Interface defined at the level of methods or messages

Service Engagement

An aggregation of business relationships

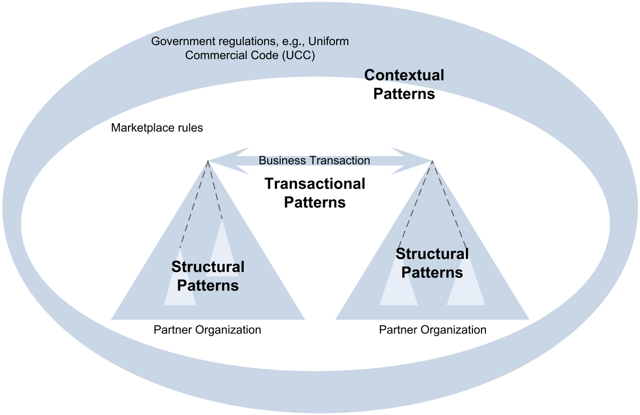
- Trillions of dollars worth of commerce conducted every year
- Characterized by
  - Independence of business partners (autonomy, heterogeneity)
  - Coproduction
    - Participation by all, though not at the same level
    - Symmetric relationships: complementary capabilities and goals
    - Produced on demand
  - Complex contracts among the partners
  - Participants are not black boxes

Business Service

Participant in a service engagement

- Characterized by transfer of (stakeholder) value, not bits
- Typically long-lived with on demand enactments
- Instantiated on the fly
    - Unlike a product
    - Though may be
        - About a product
        - Constructed using products

Conceptual Elements of a Service Engagement



- Transactional: main purpose and enactment, specifying the stakeholder value exchanged
- Structural: partnerships and contracts
- Contextual: setting of the engagement

Traditional Technical Approaches

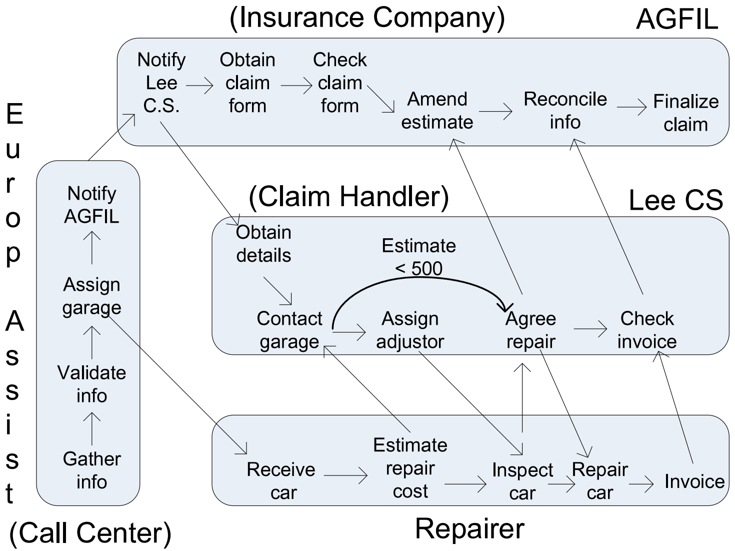
Quite unlike a real-life service engagement

- Take participants' internal control and data flows (e.g., in BPEL, BPMN) as units of abstraction
  - *Mix* private policies and public interactions
  - Proprietary: may not be available for reuse
  - Context-laden: even when available, cannot be readily reused
- Focus on low-level (e.g., WS-CDL) or data-level meanings (e.g., OWL)
  - *Ignore* business-level significance of messages
  - Ambiguous; not verifiable

BPEL, BPMN, WS-CDL, OWL are well-known standards

A Real-Life Service Engagement

Operationally over-specified as interacting flows



Sequence Diagrams

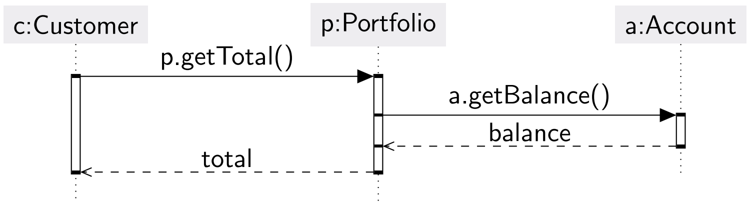
Well-known specification approach

- Originally used for object-oriented programming
- Our needs: closest to message sequence charts
- An intuitive way to express interactions
  - Expresses global view consolidating local perspectives
  - Excellent for describing possible interaction instances
  - But beware the pitfalls . . .
- Support (potential) validation checks
  - Formalizing semantics is not obvious: multiple approaches
- Standardized in UML 2.0 as Sequence Diagrams
  - Caveat: Arrowheads and other details of these notes don't necessarily match UML

Method Invocation in Object-Oriented Programming

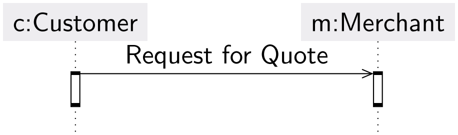
Only one thread of control

Objects exchange messages but in a fixed, blocking manner



Message Emission and Reception

Independent threads of control; autonomous parties exchange messages, asynchronously sending and receiving


c:Customer
m:Merchant
Request for Quote