Service Specification and Business Process Design

Paul C. Brown

Topics

- ***Why focus on business processes?***
- **Specification: observable behavior**
  - Functional requirements
  - Coordination
  - Constraints and dependencies
  - Non-functional requirements
  - Service architecture
- **Summary**

Why Focus on Business Processes?

- **Business processes are the source of business value**
  - IT exists to make business processes work
- **Services provide value only when they are part of a business process**
  - No use = no value
- **The value of a service depends on its ability to fit one or more business processes**
  - Lack of fit = wasted investment

Services must be designed to fit business processes!

Service Benefits Depend on Business Process Design

- **Service Reuse**
  - The service can be used in multiple business processes
  - The service interface and observable behavior can support multiple business processes
- **Isolation of service consumer and provider**
  - The service interface and observable behavior remain stable as the business process evolves

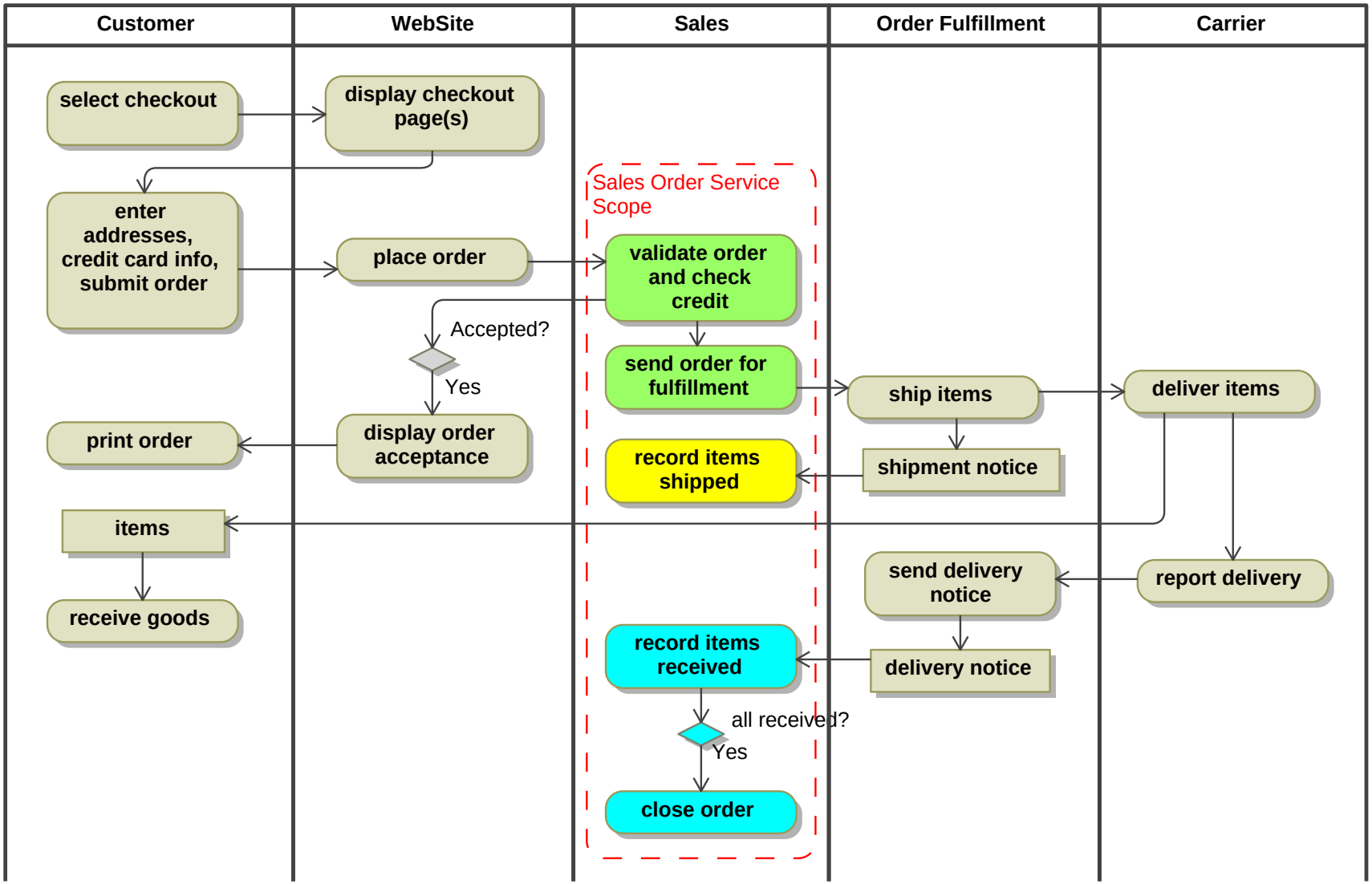
A focus on business processes is the key!

Specification: The Observable Behavior of the Service

- **How the service behaves from a consumer perspective**
- **Functional requirements**
  - What functionality does the business process require?
- **Coordination**
  - How the work of the service provider is coordinated with the work of the service consumer
- **Constraints and dependencies**
  - Dependencies between operations, constraints on their invocation
- **Non-functional requirements**
  - May differ from process to process
- **Service architecture**
  - Not entirely a black-box exercise

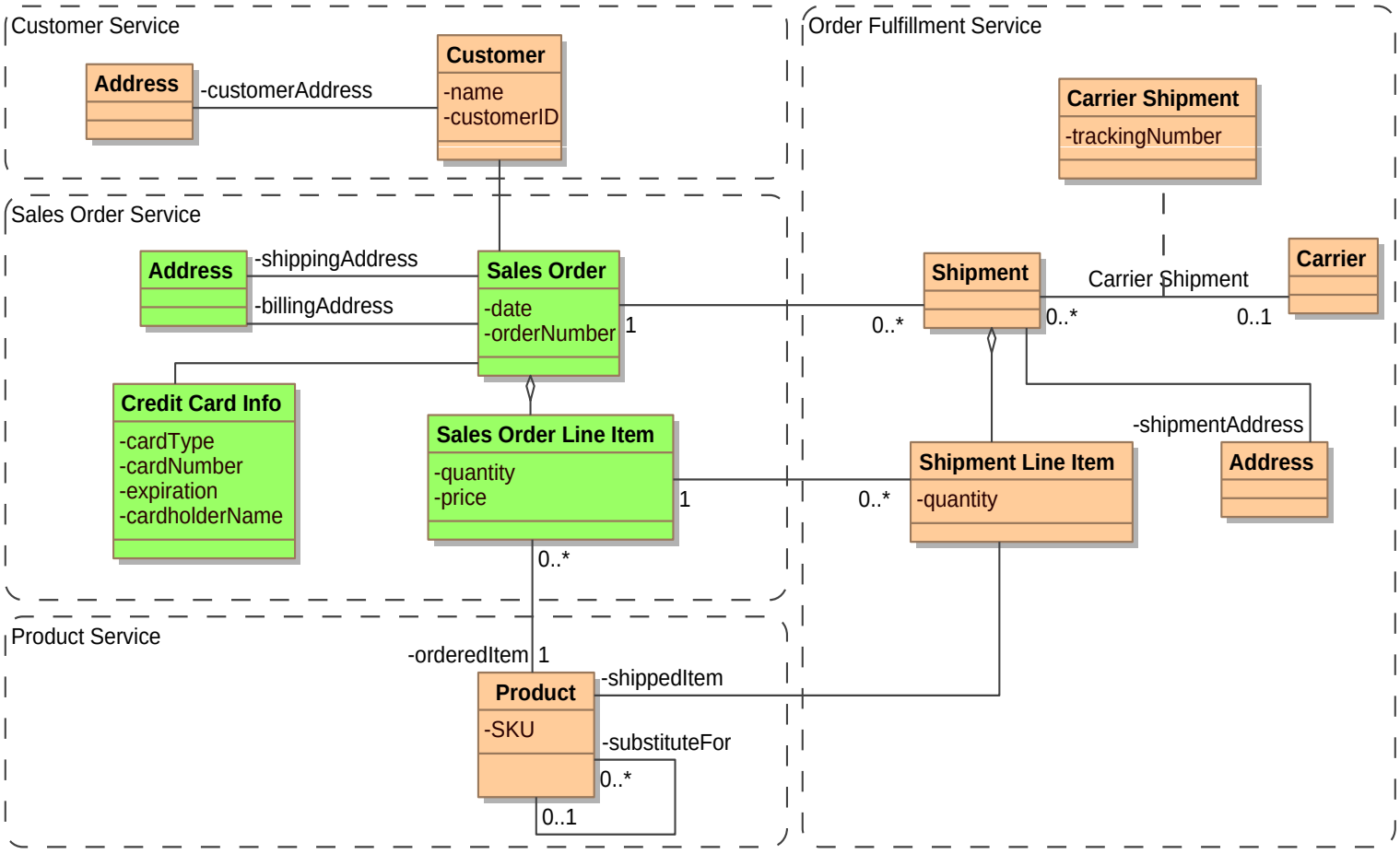
- ***Why focus on business processes?***
- **Specification: observable behavior**
  - ***Functional requirements***
  - Coordination
  - Constraints and dependencies
  - Non-functional requirements
  - Service architecture
- **Summary**

Business Process Context: How Does the Service Fit?



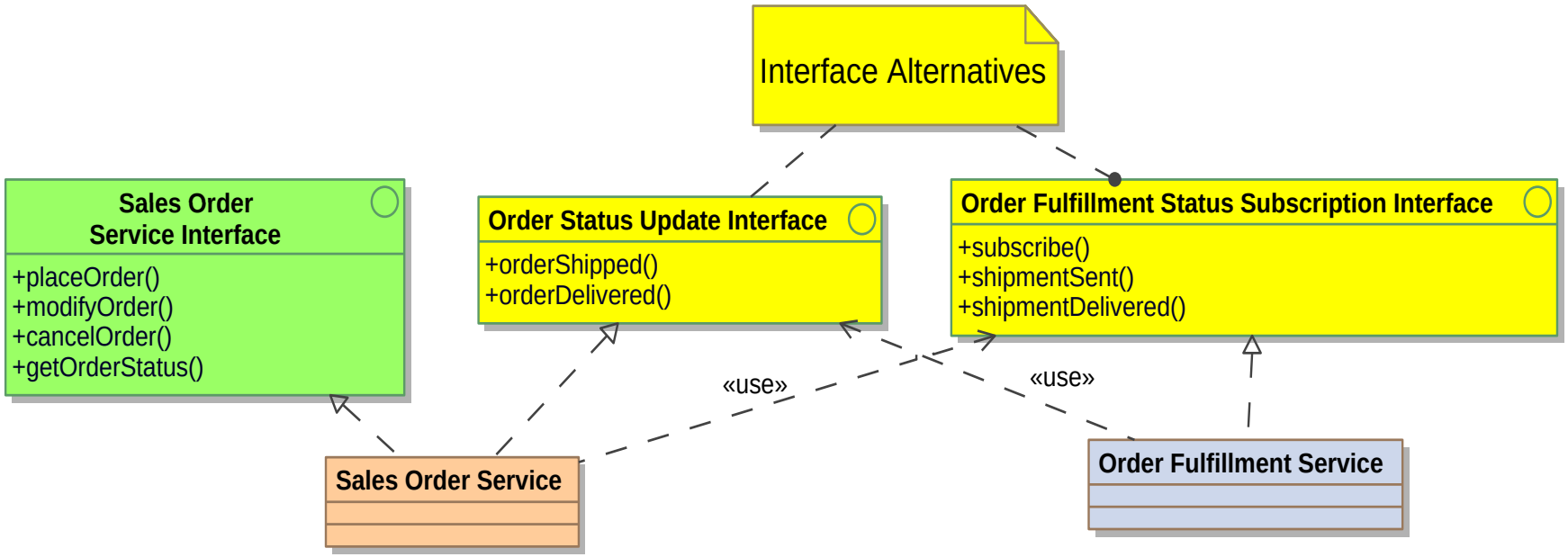
Information Requirements: Relationships to Other Services

- What information does the service manage?
- What information does the service use but not own? What information is cached? How is the cache maintained?
- Multiplicity: One or many shipments per order?

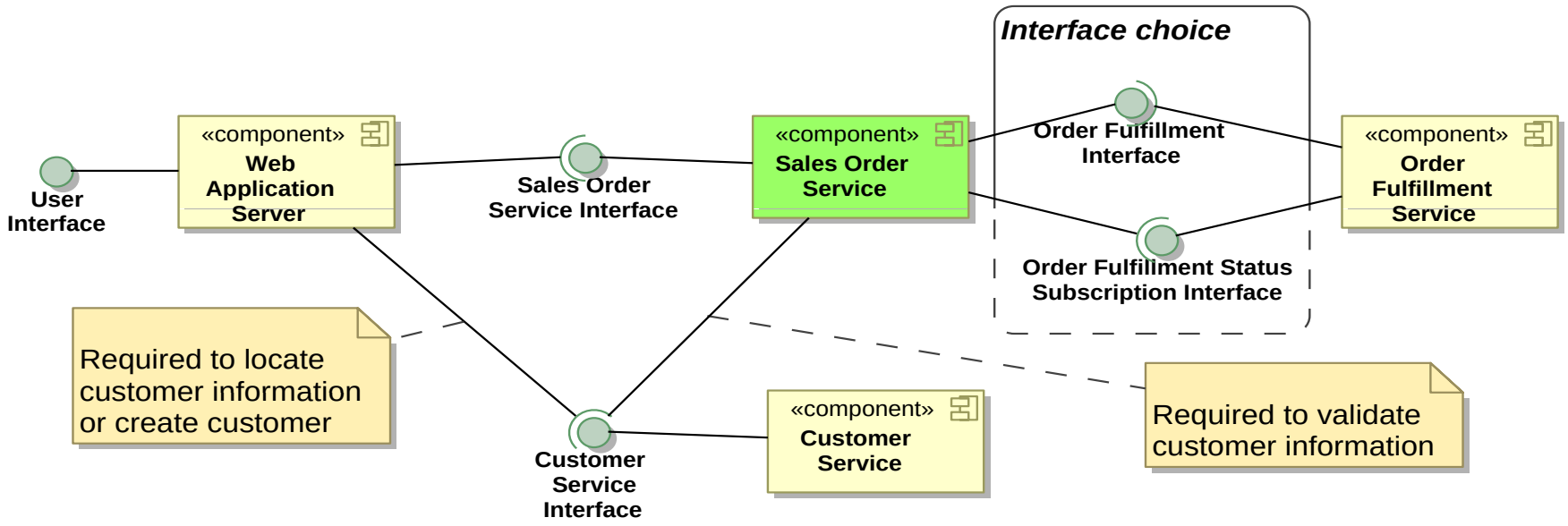


Capability Requirements

- Provided operations
- Operation sequencing
- Style of interaction

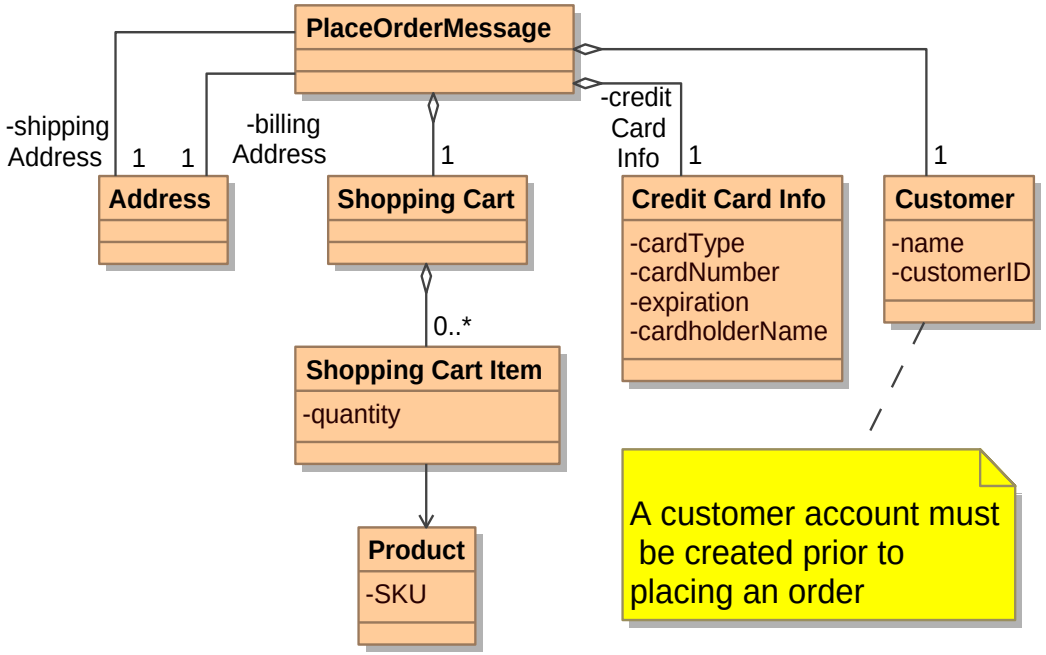


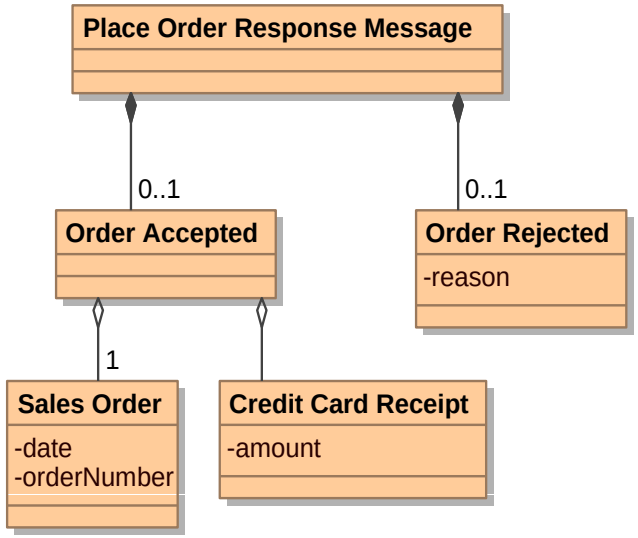
Partial Solution Architecture



Message Data Structures

- Information required
- Information returned
- Common data models
  - Whole messages?
  - Common sub-structures





Service Operation Granularity

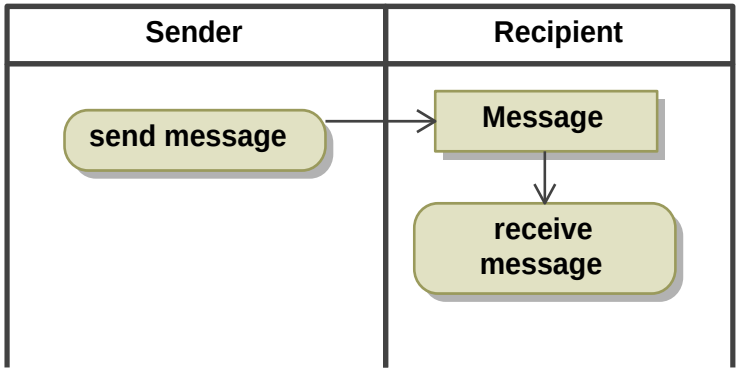
- **One or many items per order?**
- **One or many orders per request?**
  - Multiple shopping carts
  - Common shipping address?
  - Common billing information?
  - Common customer?
- **One or many shipments per order?**
  - Affects interfaces between Customer Order Service and Order Fulfillment Service

- ***Why focus on business processes?***
- **Specification: observable behavior**
  - Functional requirements
  - ***Coordination***
  - Constraints and dependencies
  - Non-functional requirements
  - Service architecture
- **Summary**

Simple Coordination Patterns

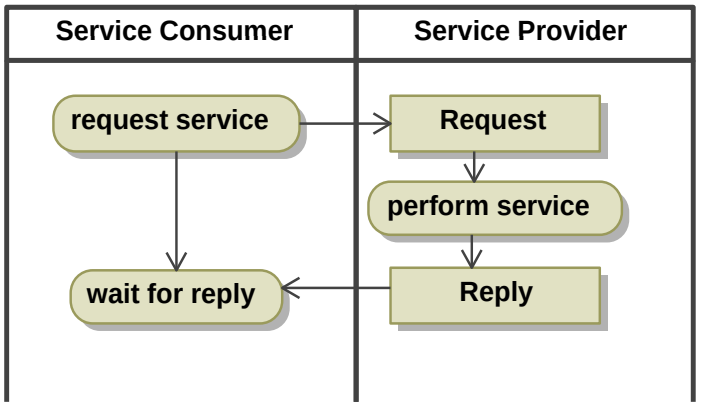
Fire-and-forget

- Message could be either a request or a notification
- No breakdown detection



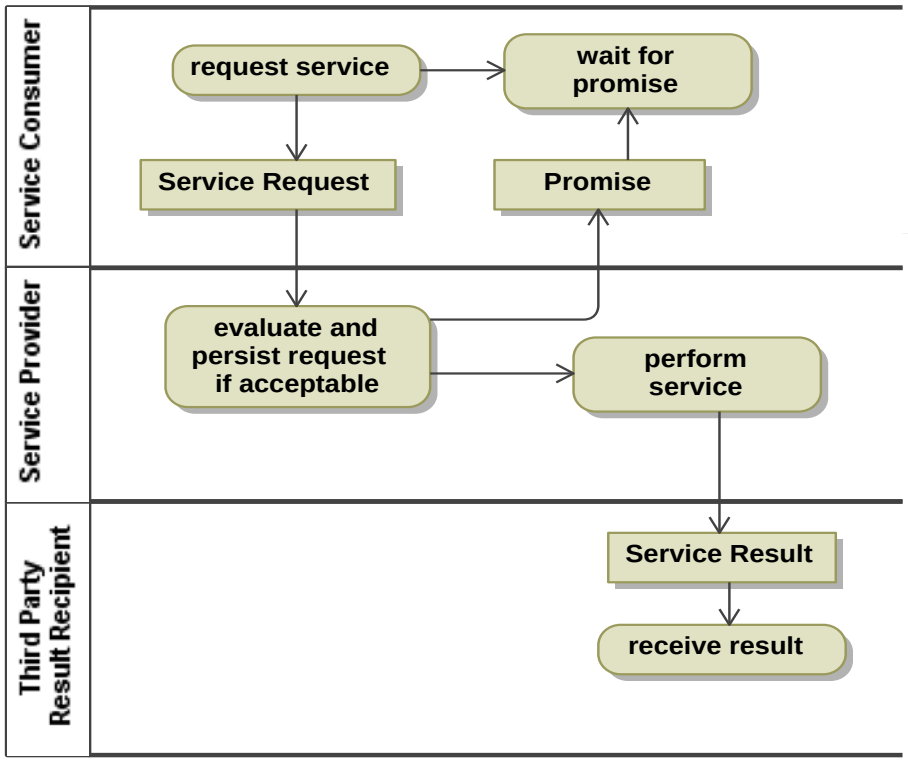
Request-Reply

- Message could be either a request or a notification
- Can detect breakdowns
  - Requires SLA for response



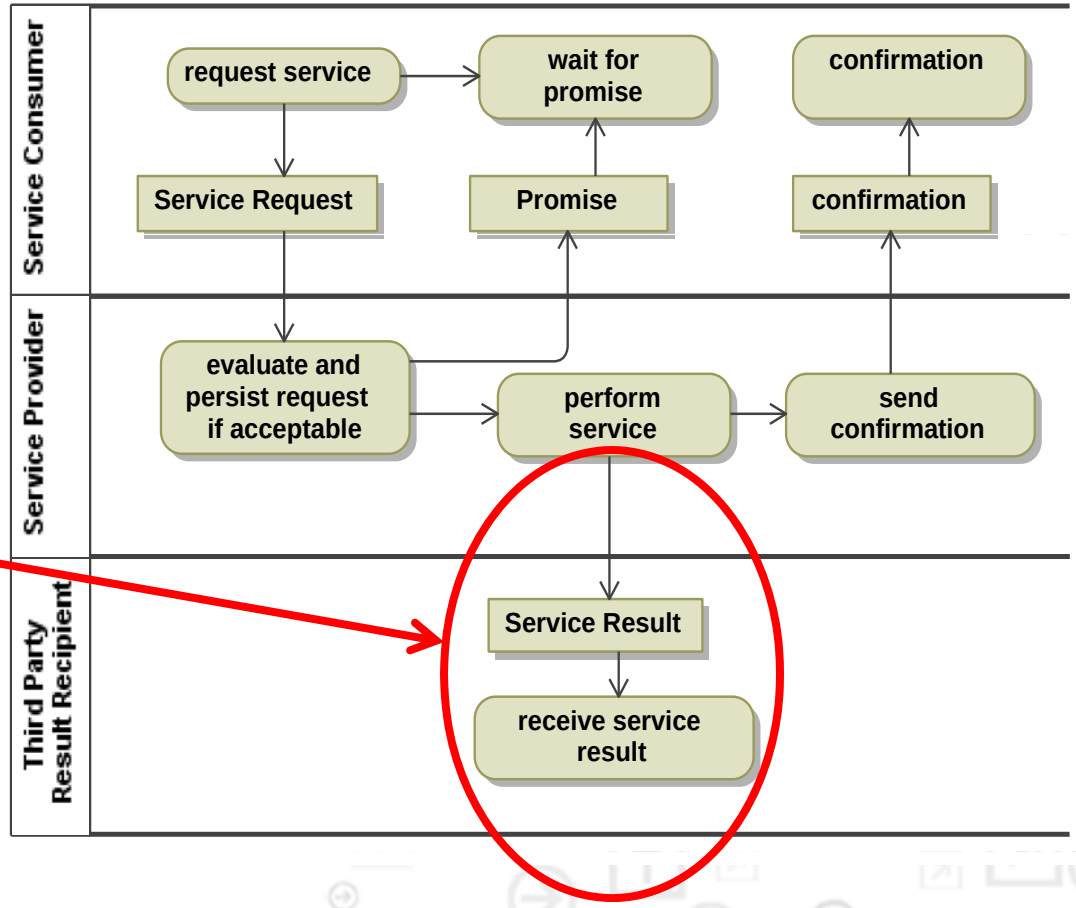
Delegation

- **Initial exchange returns promise**
  - Service provider agrees to work
  - Service provider should persist request
- **Can detect breakdown in work hand-off**
- **Cannot detect breakdown in work performance**



Delegation with Confirmation

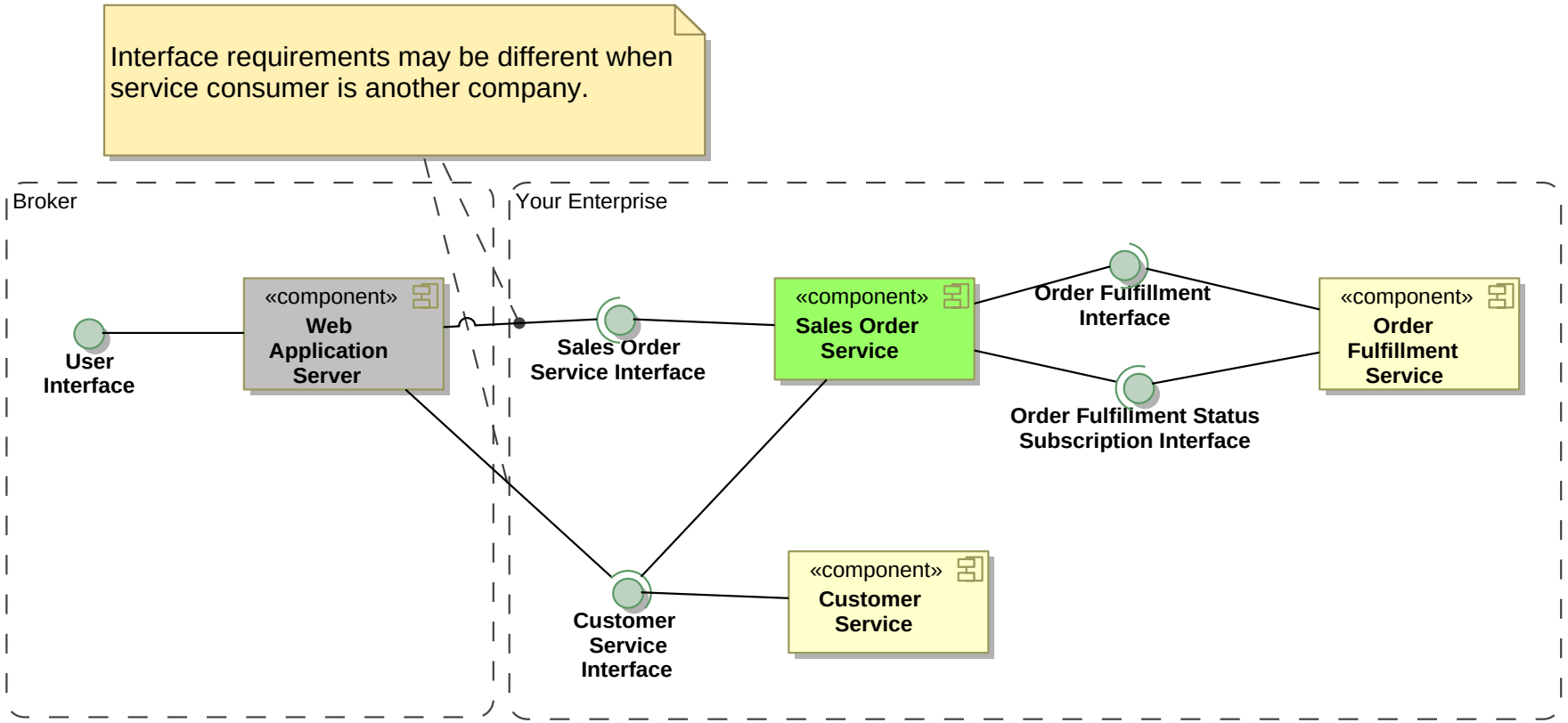
- Work performance confirmed in later asynchronous exchange
- Confirmation meaning depends on coordination pattern used by service provider



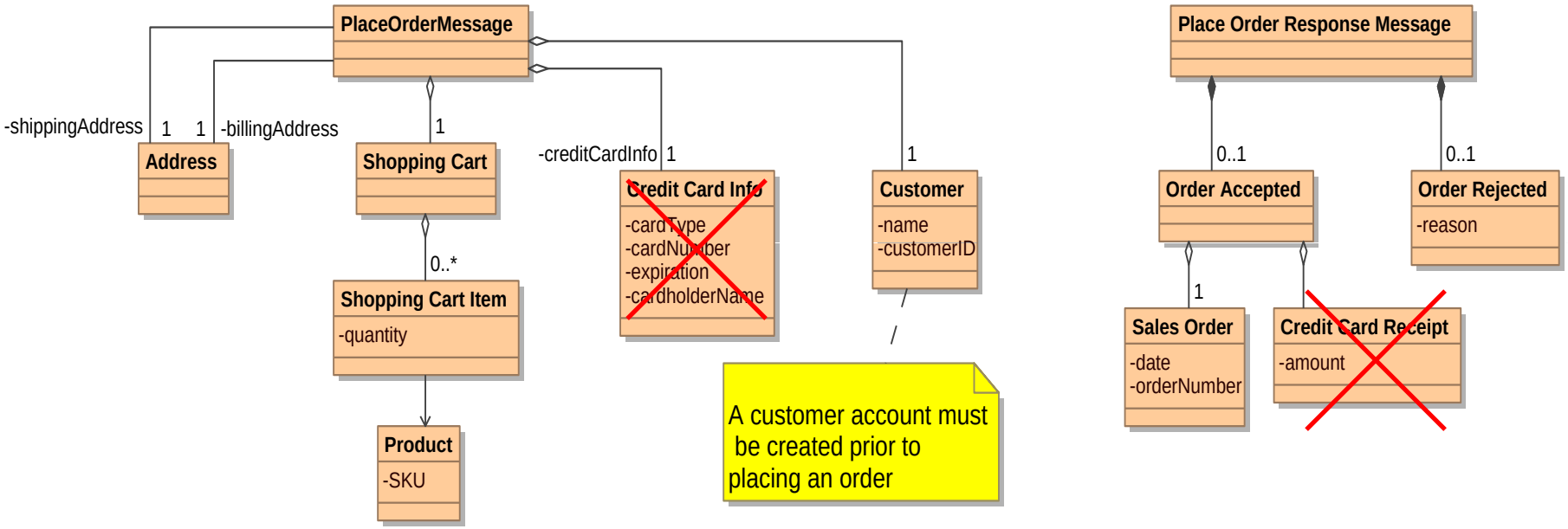
Different Channels May Require Different Coordination

- **On-line direct ordering**
  - Immediate synchronous response
- **Email ordering**
  - Delayed asynchronous response
- **Phone ordering**
  - Operator requires immediate synchronous response
  - Secondary feedback channel may be required (e.g. email)
- **B2B ordering**
  - Batch – asynchronous response
  - Interactive – synchronous response
- **Agent ordering**
  - Includes customer payment information
  - Requires separate accounting for agent fees
- **Broker (reseller) ordering**
  - Does not include customer payment information
    - Broker handles customer payments
  - Broker pays for products at a discount

Partial Architecture with Broker



Place Order Message Structures with Broker



- An alternate mechanism for payments is required!

Exception Handling

- **Don’t use SOAP faults for expected conditions**
  - Expected conditions should return expected results
    - e.g. Bad credit card number on placeOrder() operation
  - May be difficult to continue business process flow with SOAP faults
- **Anticipate business process requirements for responding to exceptions**
  - Information required in expected results
    - e.g. primary keys, problem details
  - Additional service operations
    - e.g. query, edit, delete, hold, and resume operations
  - Options for resuming business process
    - e.g. engage customer service representative
    - Support must be provided for these activities

- ***Why focus on business processes?***
- **Specification: observable behavior**
  - Functional requirements
  - Coordination
  - ***Constraints and dependencies***
  - Non-functional requirements
  - Service architecture
- **Summary**

Operation Usage Constraints and Dependencies

- **Constraints: You can't:**
  - Call getOrderStatus(), modifyOrder(), or cancelOrder() an order that does not exist
  - Call modifyOrder() or cancelOrder() after the order has shipped
- **Dependencies**
  - In order to modify or cancel an order, you the need the order number created in the initial placeOrder()
  - Maybe you need a findOrder() operation?

- ***Why focus on business processes?***
- **Specification: observable behavior**
  - Functional requirements
  - Coordination
  - Constraints and dependencies
  - ***Non-functional requirements***
  - Service architecture
- **Summary**

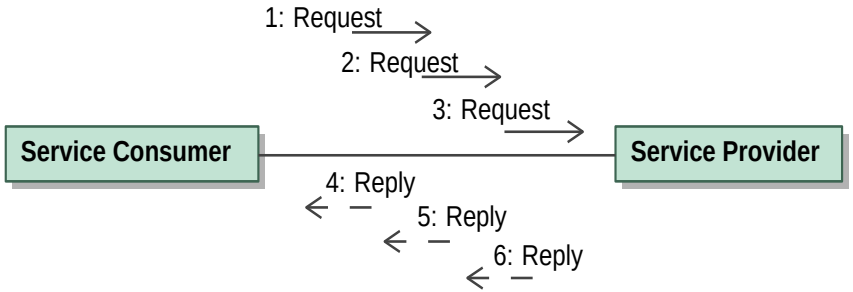
Non-Functional Requirements to Consider

- **Throughput**
- **Response time**
  - Synchronous and asynchronous
- **Availability**
  - Required availability during normal working hours
    - Maximum time to recover
  - Allowed outage periods
- **Security**
  - Authentication and authorization
    - Variations for different business processes
  - In-transit data encryption
  - At-rest data encryption
- **Service-level agreements (SLAs) for the above**

Most requirements come from the business processes!

High Throughput Considerations

- **Focus on true peak rate at which response time must be met**
- **Consider back-end performance limitations!**
- **High throughput does not necessarily require short response time**
  - Asynchronous responses allow multiple requests to be in progress



- **Is serialization of operations a requirement?**
  - E.g. create, delete, and update operations on the same order must be processed in the order received

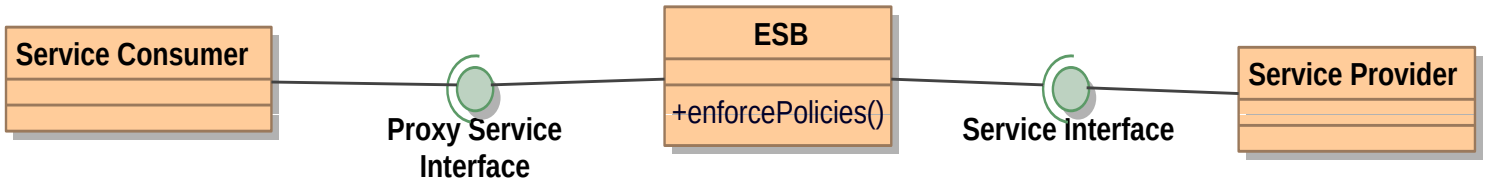
- ***Why focus on business processes?***
- **Specification: observable behavior**
  - Functional requirements
  - Coordination
  - Constraints and dependencies
  - Non-functional requirements
  - ***Service architecture***
- **Summary**

Service Architecture

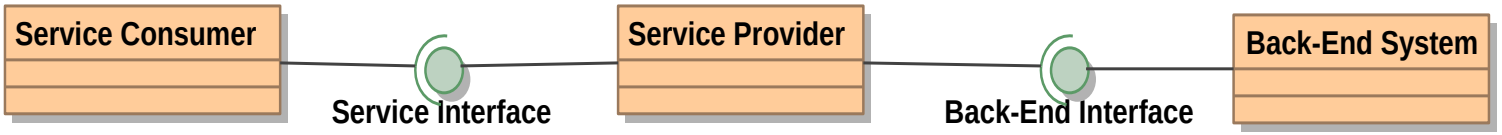
- **The architecture of the service implementation is relevant to the extent that it impacts:**
  - Observable behavior
  - Access mechanisms
  - Performance feasibility

Service Architecture Choices

- **Access mechanisms enabling policy-based access**
  - Security: authentication, authorization, encryption
  - Results filtering

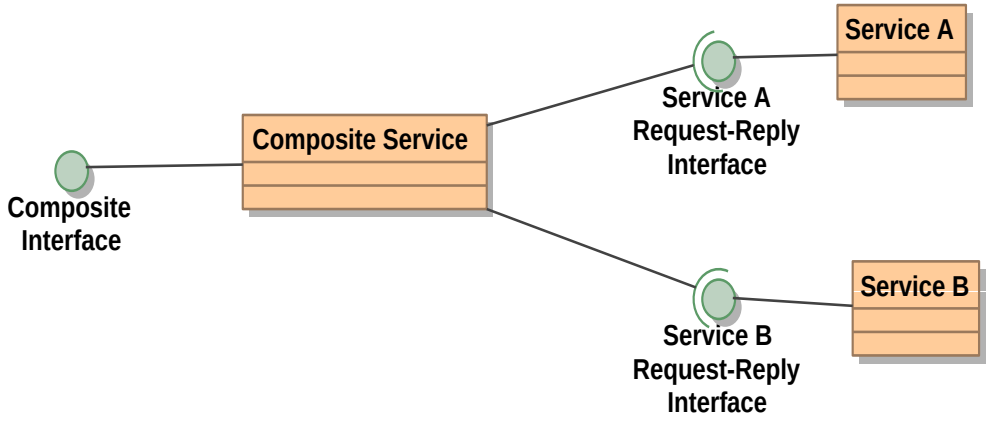


- **Performance for a back-end system wrapper**
  - How will scaling work?

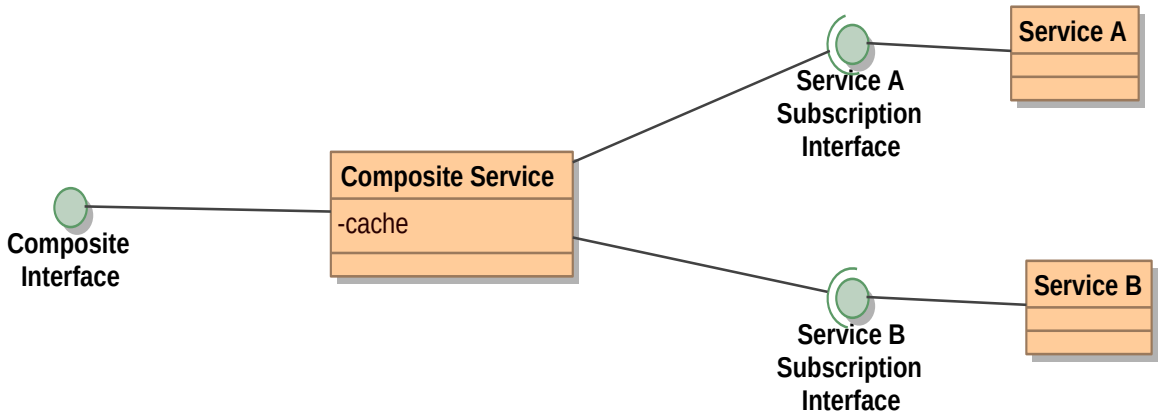


Composite Service Architecture Impacts Performance

Traditional architecture



Cache-based architecture



- ***Why focus on business processes?***
- **Specification: observable behavior**
  - Functional requirements
  - Coordination
  - Constraints and dependencies
  - Non-functional requirements
  - Service architecture
- ***Summary***

Summary

- **Business processes are the source of business value**
  - Behavior of services must fit into processes to provide value
    - Operation usage constraints and dependencies
  - Reuse depends on satisfying the requirements of more than one business process
- **Service interface stability is key to SOA ROI**
- **Most services manage information**
  - References to another service's information require additional interfaces to maintain consistency
- **Coordination of work is an important consideration**
  - Different consumers may require different coordination
- **Most non-functional requirements derive from business processes**
- **Service implementation architecture may impact feasibility**

Questions?