CSCI427/CSCI927
Service-Oriented Software Engineering

Service modeling

Acknowledgement: Materials from these slides are adapted from Aditya Ghose and "The Business Service Representation Language: A Preliminary Report"

Service Modeling: Key ideas

- What must a service description contain:
  - A description of service functionality?
  - Information on how to invoke the service, and how to interpret the outputs returned (data formats etc.) ?
  - Non-functional (Quality-of-Service or QoS) factors?

- Can we devise a service description language that seamlessly spans the spectrum from business services to web services?

Existing modeling languages and standards

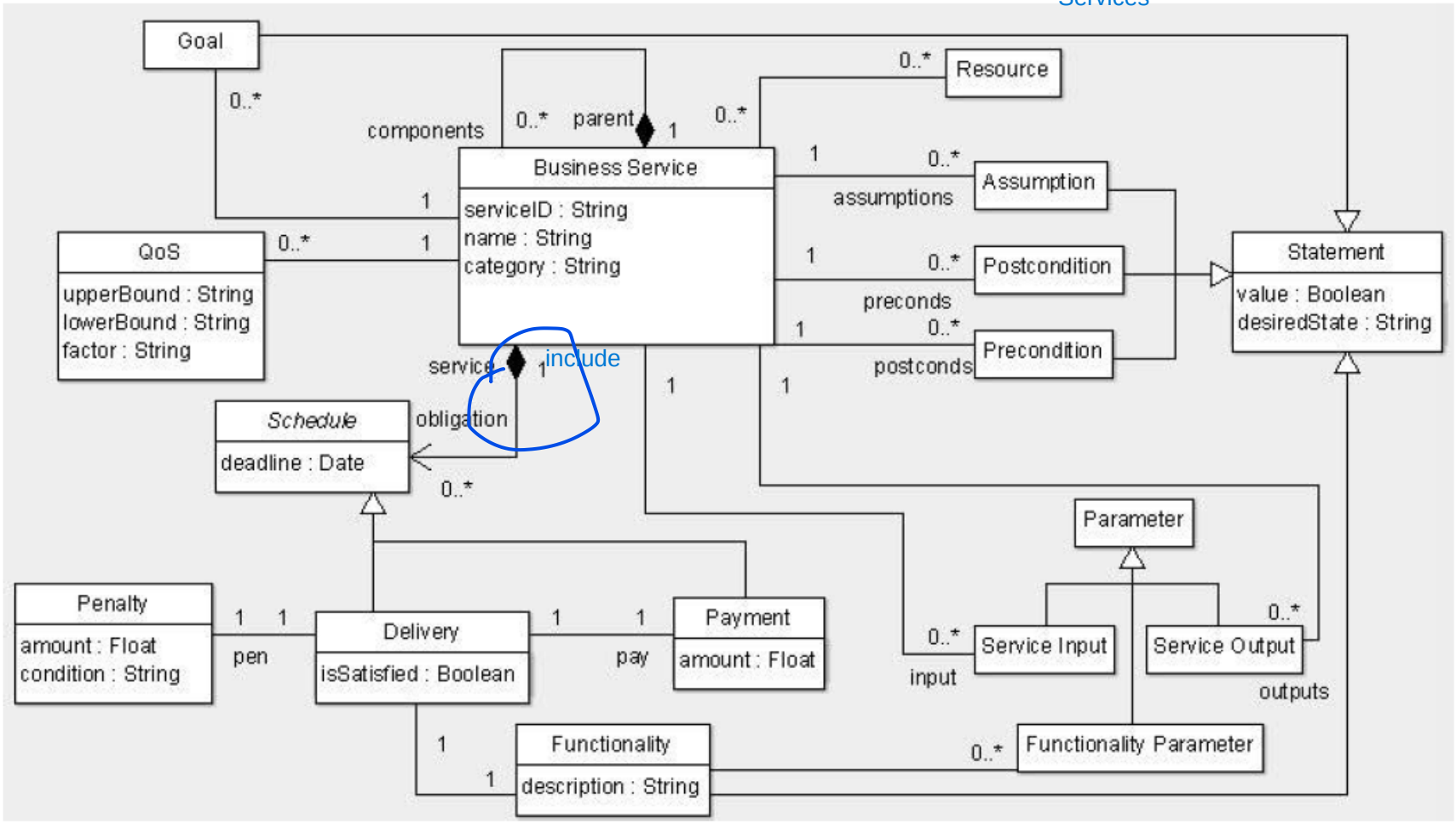
- Web service standards
  - Description: WSDL
  - Invocation: SOAP
  - Discovery: UDDI
- Semantic web service modeling languages
  - OWL-S
  - WSMO
  - WSDL-S
- Most of these follow the Input-Output-Precondition-Effect (IOPE) approach

Business Service Representation Language (BSRL)

service description <service-id>
goal <description of functionality>
preconditions <description>
postconditions <description>
assumptions <description>
inputs
<input-item> <format>
.
.
.
<input-item> <format>
outputs
<output-item> <format>
.
.
.
<output-item> <format>
Resources <description>
Key steps: <Step 1>, <Step 2>……
QoS factors/Contractual factors
<QoS-factor> <inequality> <value>
.
.
.
<QoS-factor> <inequality> <value>

Business Service Representation Language (BSRL)

Business Serrvice consists of Business Services



Goals

- Goals represent states of affairs that we desire to achieve
- Why is a post-condition described separately from a service goal? Some post-conditions are desired, others are ?collateral?
- What goals can a service help achieve?
  - **Achieving** a condition
  - **Maintaining** a condition
  - **Avoiding** a condition
  - More complex goals: make the light blink every 5 minutes?
- How are the conditions written?
  - In natural language
  - In structure document formats: e.g. XML
  - In logic
    - First-order logic: descriptions of static worlds
    - Temporal logic: descriptions of dynamic worlds
    - Other: dynamic logic, non-monotonic logic etc.

Conditions and Assumptions

- *Pre-conditions:* Conditions (in the operating environment of the service) that must be true at the start of the service
- *Post-conditions (effects):* Conditions that are made true via the execution of a service
- *Assumptions:* Services sometimes rely on certain conditions being true that cannot be evaluated a-priori
  - Sometimes a service is invoked contingent on some assumptions being valid
  - If the assumption turns out to be incorrect, we need to abort and roll-back
  - How do we ensure that assumptions are properly managed in service composition?
- Ensure consistency of assumptions in services being composed
  - Assumption example: The FORCE MAJEURE clause in contracts

Quality of Services (QoS)

- Quality of service specifications provides a measure that describes the effectiveness of a business service
- QoS specications are constraints that describe operational aspects of service qualities.
- QoS factors can be described qualitatively or quantitatively.
- Examples of QoS Specications include:
  - Delivery in under 30 minutes

Delivery schedules and Payment schedules

- Delivery schedules
  - These are specified as a set of <functionality, deadline>
- Payment schedules
  - Similar to delivery schedules in representation.

Penalties

- Specified as condition and amount pairs
- Given a `condition' C, a penalty P is invoked as reparation for condition C becoming true
  - e.g. If paint is spilled on carpet then penalty is cost of cleaning the carpet.

Service Composition

- Example: A travel service
- Commonly occurring functionality in a travel service:
  - Airline ticket booking
  - Airport transfer booking
  - Hotel booking
  - Tour booking
  - Theatre ticket booking
- If each of these functionalities is available as a service (possibly atomic, or recursively composite), we can obtain a travel service by composing these existing services

Service Composition

- Functional requirements:
  - air-ticket-booked AND airport-transfer-booked AND hotel-booked AND tour-booked AND theatre-booked
- Non-functional requirements:
  - Σ cost < BUDGET
  - Σ time < DEADLINE
- What does the composite service look like?

What do we do with service models?

- Maintain a clear understanding of enterprise/system capabilities and know-how
  - Service catalogues
  - Intellectual property (IP) asset repositories
  - Enterprise architectures
- Dynamically generate functionality (e.g., to handle exceptional situations) by composing existing services
- Analyze compliance
- Strategic alignment:
  - Do we have services to realize all enterprise strategies?
  - Why do we support certain services (we answer the question by pointing to the strategies that these services support)?
  - Enterprise re-engineering/rationalization: What services are redundant?