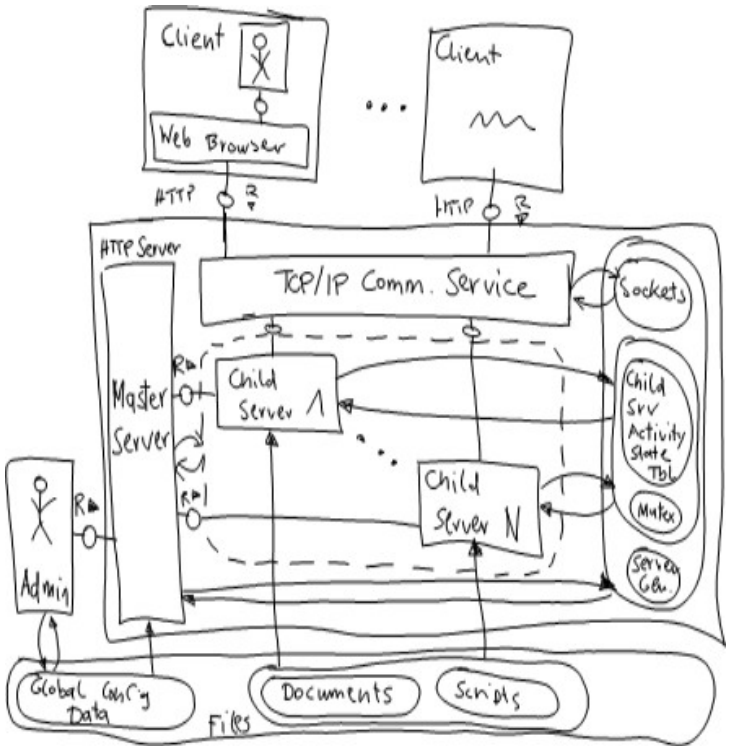
The Unified Modeling Language (UML)

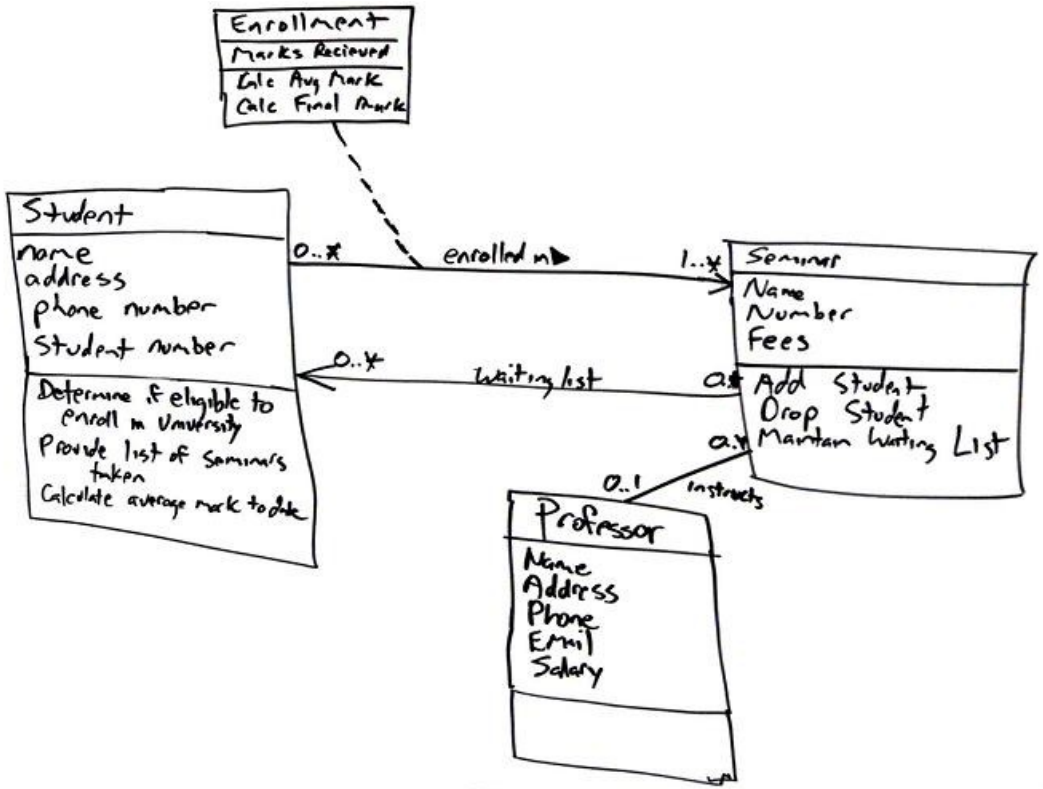
October 17, 2017

Byung-Gon Chun

(Slide credits: George Candea, EPFL)

Sketching Structure

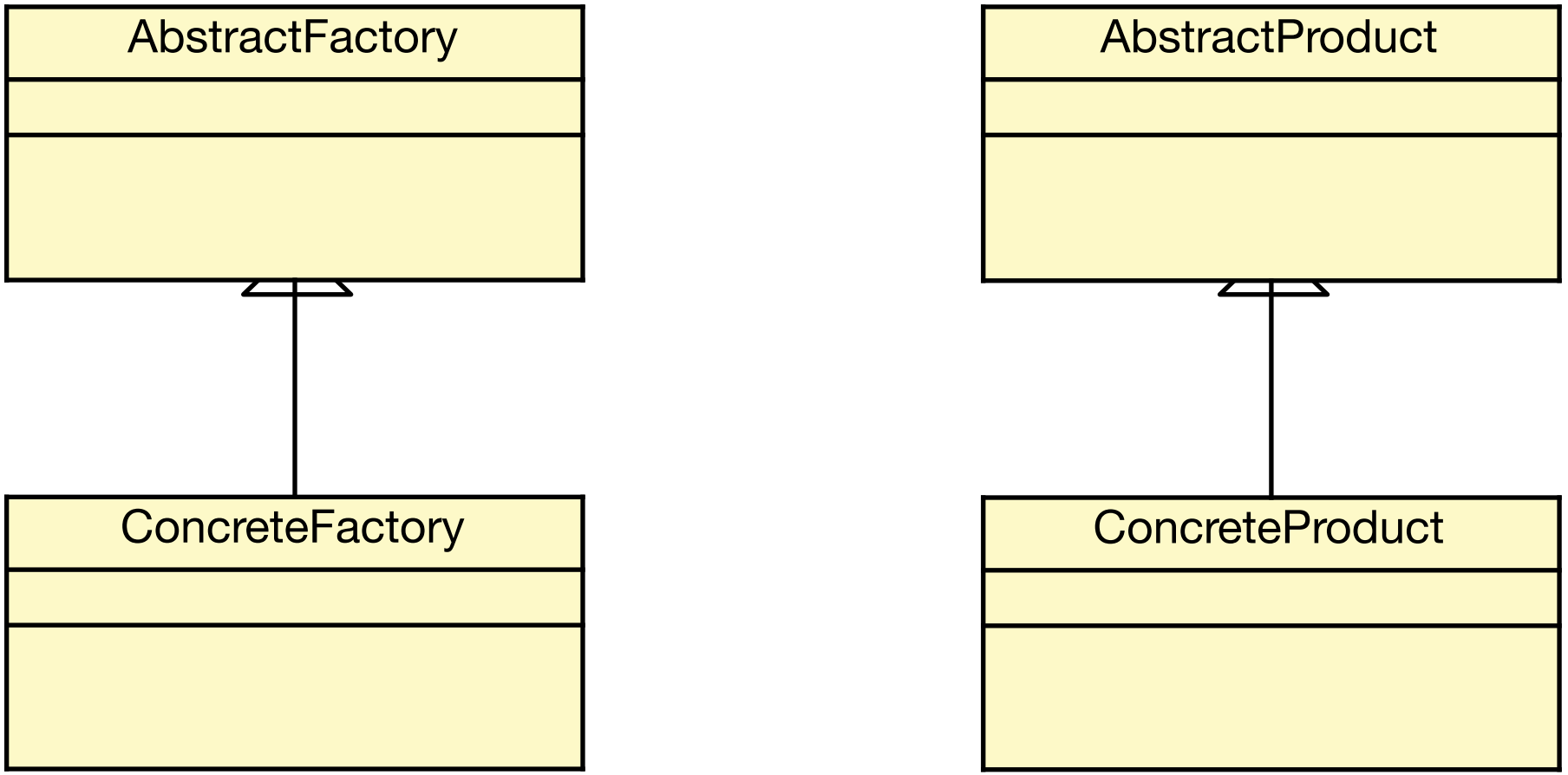




Modeling Languages

- Universally understood format for diagrams
- Format for non-code artifacts
- Document
  - *Requirements*
  - *Communication with customers*
  - *Software components*
  - *Hardware components*
  - *…*

You've seen UML

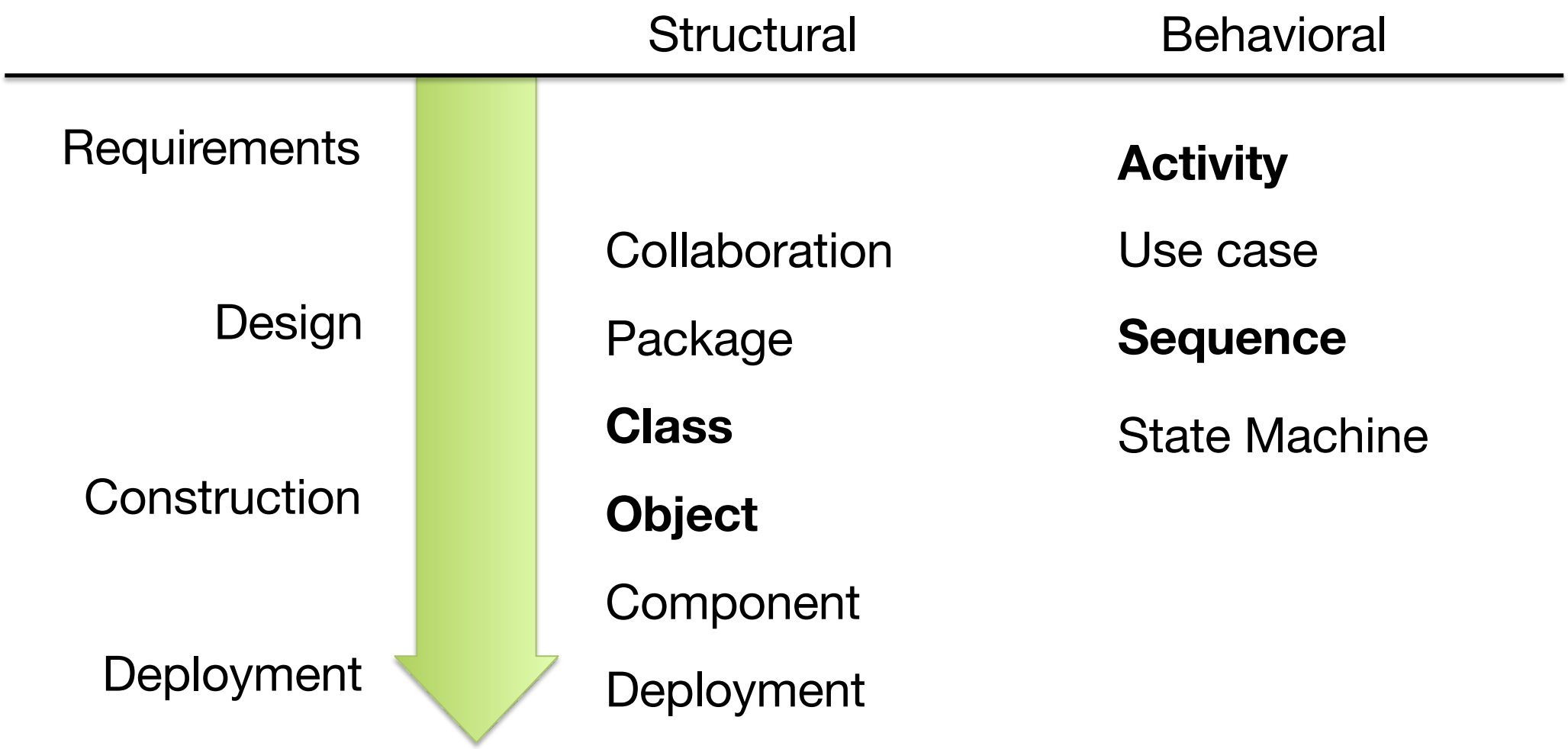


Brief History

- Defined in 1997 to integrate several modeling languages
- Three amigos at Rational
  - *James Rumbaugh, Grady Booch, and Ivar Jacobsen*
- Universal language for SW blueprints
  - *Different views for different aspects*
- Today's industry standard
  - *Lots of criticism, still widely adopted*

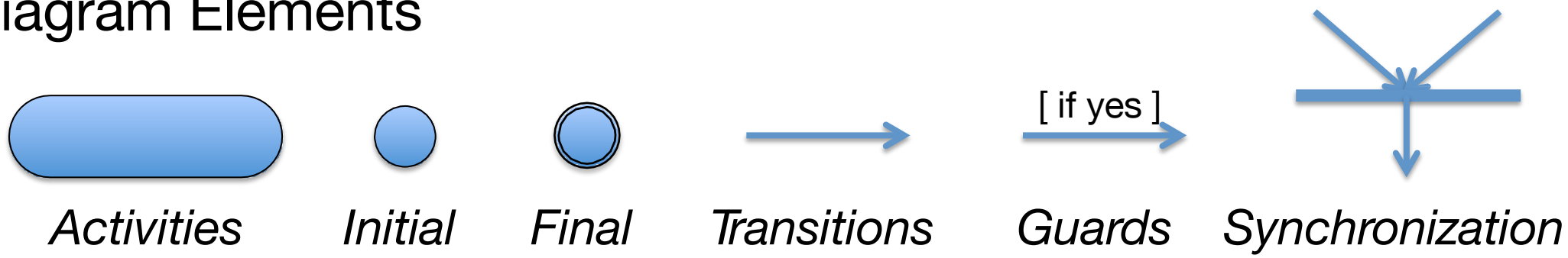


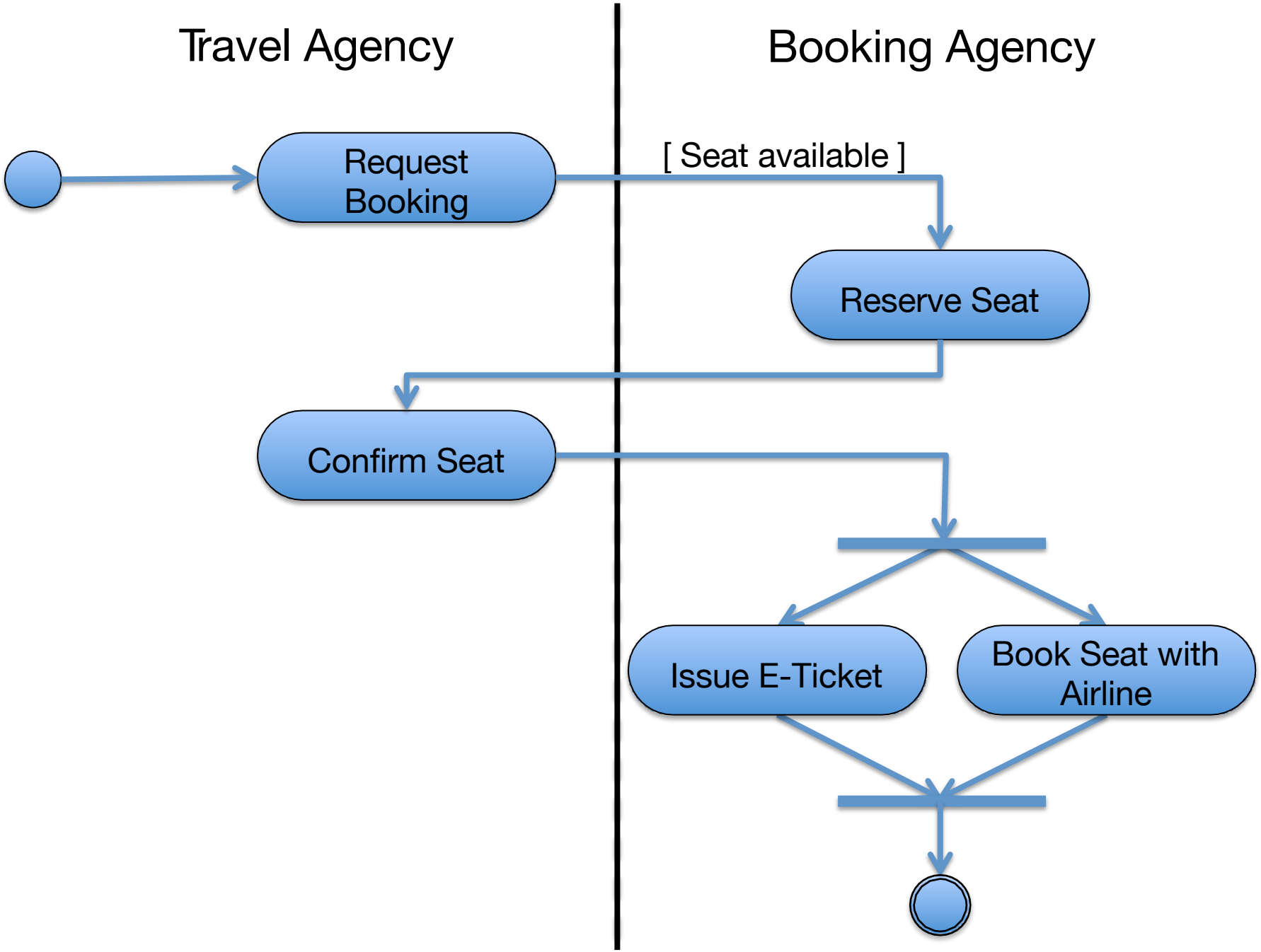
Diagram Types



Activity Diagrams

- Flow charts
  - *Business workflow, algorithms*
  - *Early planning, but applicable in all phases*
  - *Swimlanes show passing of control*
- Diagram Elements

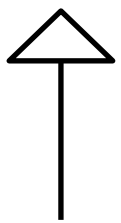




Activity Diagram:
Booking a Flight

Class Diagrams

- Classes with names, attributes, and methods
- Visualizes relationships among classes



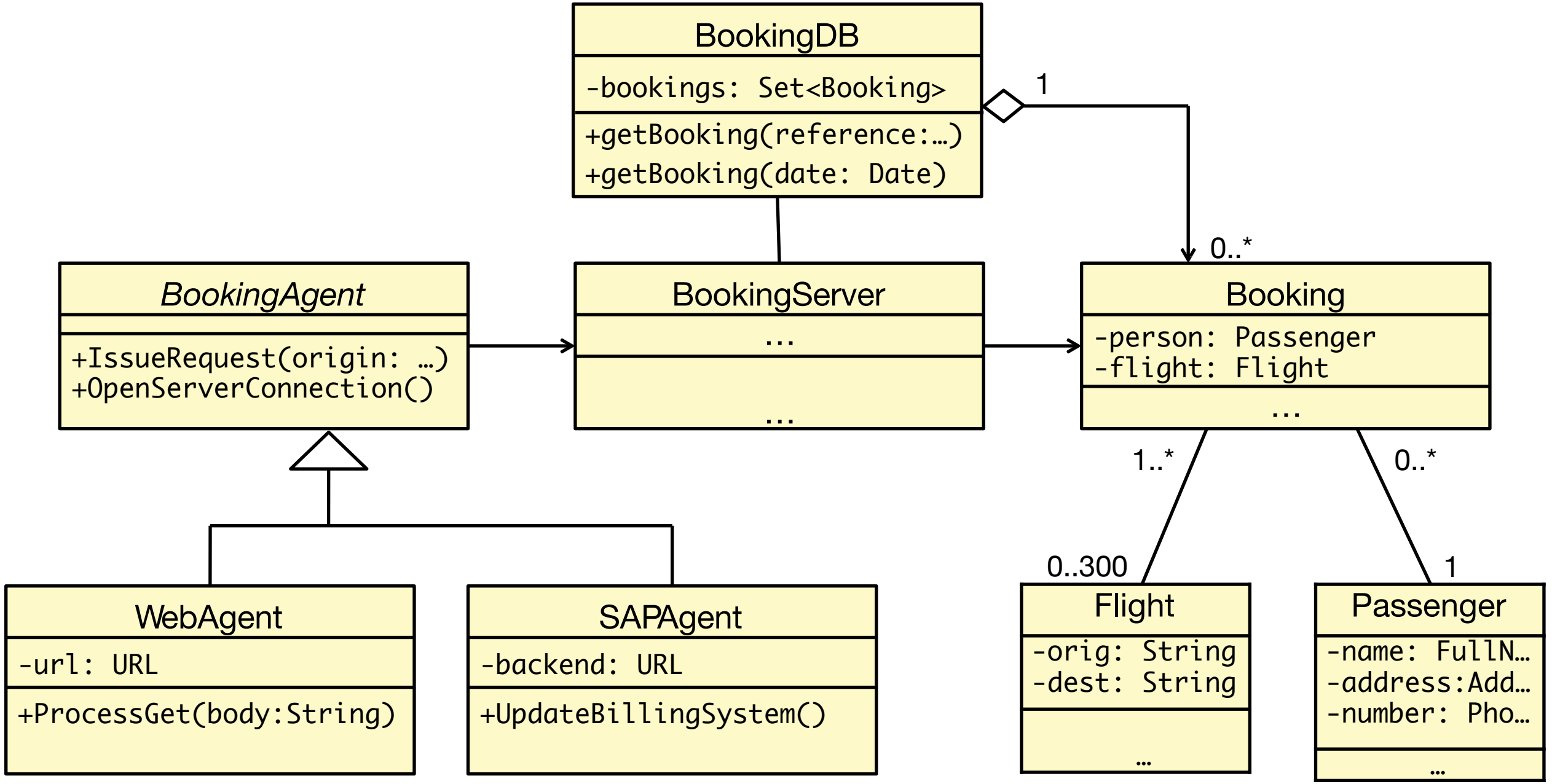
Inheritance

Aggregation

Association

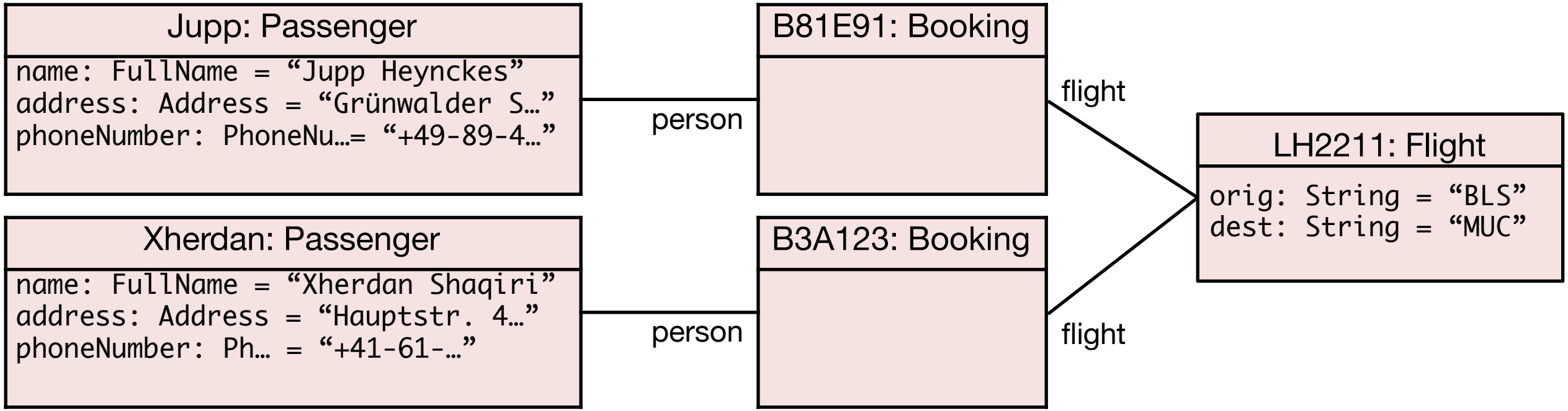
Dependency

- Properties of association and aggregation
  - *Multiplicities* *0..*, 1, 5..7*
  - *Direction:*


BookingDB
-bookings: Set<Booking>
+getBooking(reference:…)
+getBooking(date: Date)
1
0..*
BookingAgent
+IssueRequest(origin: …)
+OpenServerConnection()
BookingServer
…
…
Booking
-person: Passenger
-flight: Flight
…
1..*
0..*
0..300
1
WebAgent
-url: URL
+ProcessGet(body:String)
SAPAgent
-backend: URL
+UpdateBillingSystem()
Flight
-orig: String
-dest: String
…
Passenger
-name: FullN…
-address:Add…
-number: Pho…
…

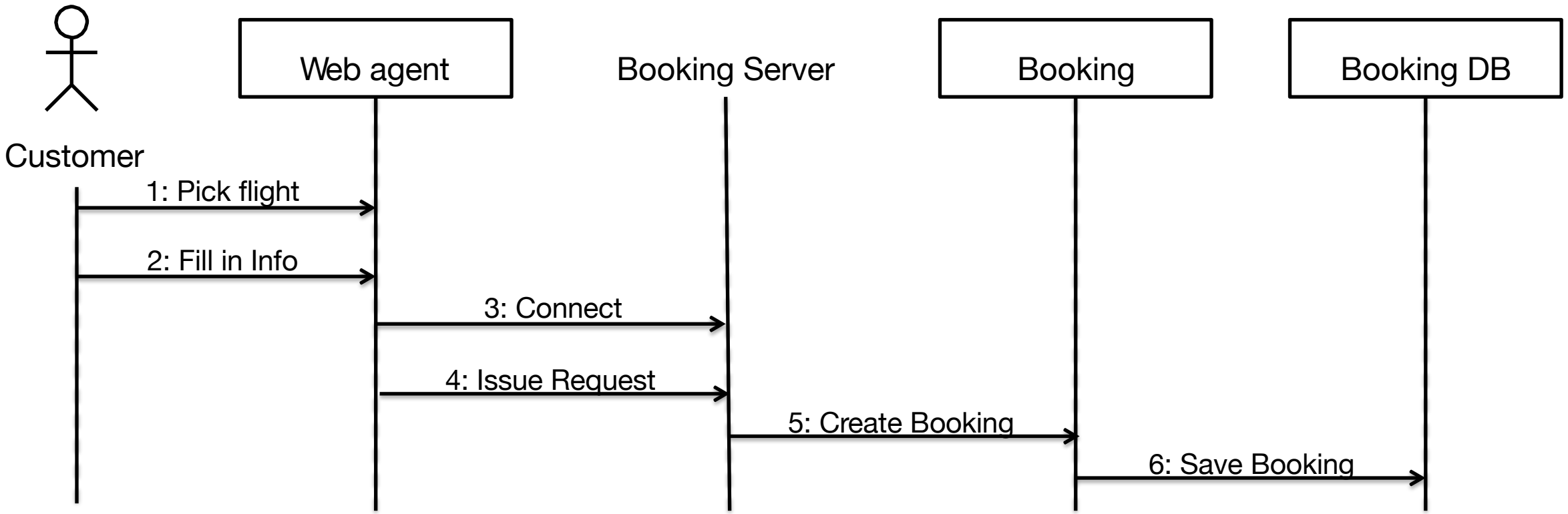
Object Diagrams

- Structure of the system at a particular time
- Class diagram with concrete instances and values



Sequence Diagrams

- Use cases as object interactions arranged in a time sequence



Summary

- Full use of UML mostly in highly formalized development
  - *CASE tools*
  - *Code synthesis from UML*
- Small teams rarely use anything beyond class diagrams
- Pick what you like
  - *UML helps standardizing ad-hoc diagrams*
  - *Requirements and high-level concepts can be formalized*
  - *Gives you an idea what aspects can be documented*