

Software Engineering: Specification, Implementation, Verification

Chapter 1: Analysis and Design

Suad Alagić

Springer 2017

Main topics

- Specification of use cases
- Structural modeling
- Behavioral specifications
- Use cases, entity diagrams, and sequence diagrams
- Aggregation
- Entity types as interfaces
- Entity types as classes

Modeling application environments

- Types of users
- Activities
- Actors
- Information requirements

Entities and use cases

- Types of entities
- Entity relationships
- Specification of activities
- Use cases

- Involved entities
- Actors
- Specifications of actions
- Constraints

Declarative specifications

- Preconditions
- Postconditions
- Frame constraints

Specification of use cases

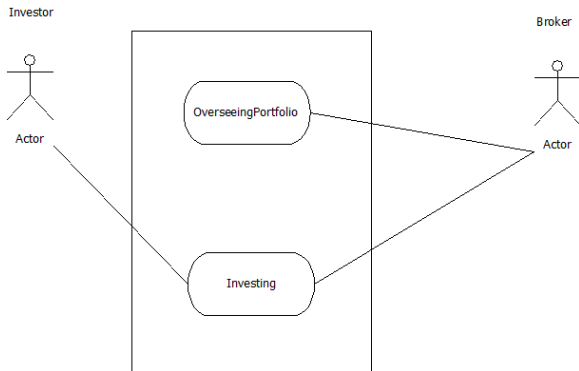


Figure: Investment management

Specification of use cases

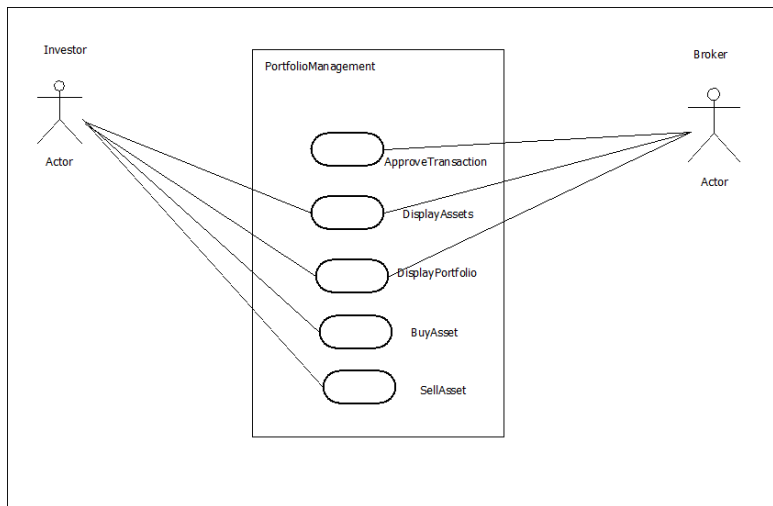


Figure: Portfolio management use cases

Specification of use cases

Use case: BuyAsset

Entities: Investor, Asset, Portfolio, Broker

Actors: Investor, Broker

Constraints:

Preconditions: assetPriceOK, brokerApproves

Postconditions: assetInPortfolio

Frame: All other assets in portfolio unaffected

Specification of use cases

Use case: SellAsset

Entities: Investor, Asset, Portfolio, Broker

Constraints:

Actors: Investor, Broker

Preconditions: assetInPortfolio, brokerApproves

Postconditions: not assetInPortfolio

Frame: All other assets in portfolio unaffected

Flight management

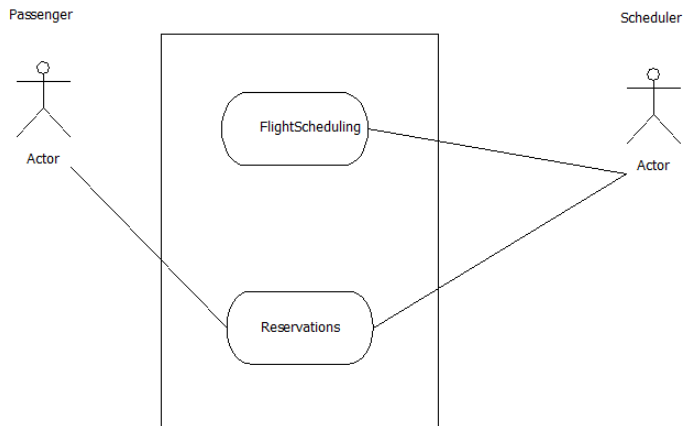


Figure: Flight management

Flight management use cases

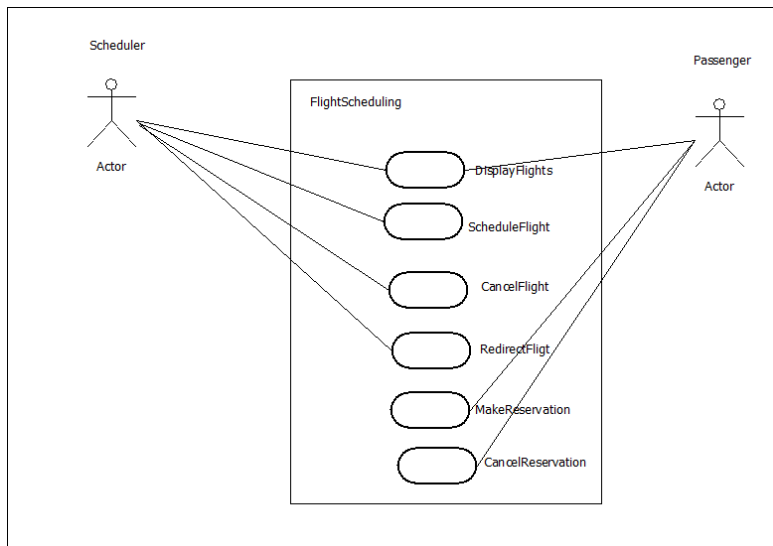


Figure: Flight management use cases

Schedule flight use case

Use case: ScheduleFlight

Entities: Scheduler, FlightSchedule, Flight, Aircraft, Airport

Actors: Scheduler

Constraints:

Preconditions: not flightScheduled, originAirportAvailable,
destinationAirportAvailable, aircraftAvailable

Postconditions: flightScheduled

Frame: All other scheduled flights unaffected

Cancel flight use case

Use case: CancelFlight

Entities: Scheduler, FlightSchedule, Flight

Actors: Scheduler

Constraints:

Preconditions: flightScheduled, not inFlight

Postconditions: not flightScheduled

Frame: Schedule of all other flights unaffected

Redirect flight use case

Use case: RedirectFlight

Entities: Scheduler, FlightSchedule, Flight, Airport

Actors: Scheduler

Constraints:

Preconditions: inFlight, newDestination notEqual destination,
newAirportAvailable

Postconditions: destination equal newDestination

Frame: flightScheduled, schedule of all other flights unaffected

Make reservation use case

Use case: MakeReservation

Entities: Passenger, FlightSchedule

Actors: Passenger

Constraints:

Preconditions: flightAvailable

Postconditions: reservationConfirmed

Frame: Other reservations not affected

Cancel reservation use case

Use case: CancelReservation

Entities: Passenger, FlightSchedule

Actors: Passenger

Constraints:

Preconditions: reservationConfirmed

Postconditions: not reservationConfirmed

Frame: Other reservations not affected

Structural modeling

- Entity types
- Relationships among entities
- Associations
- Inheritance

Structural modeling: Associations

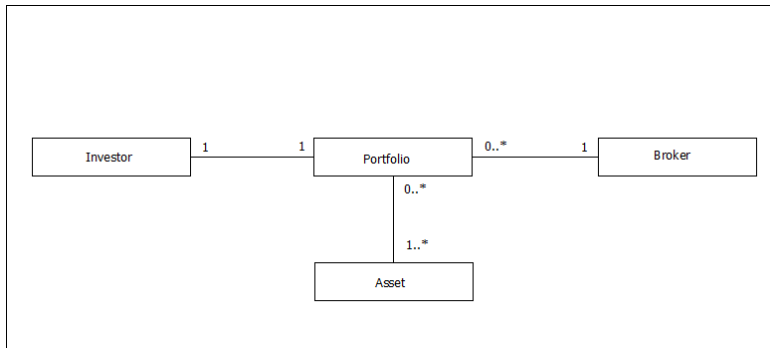


Figure: Investment management associations

Structural modeling: Inheritance

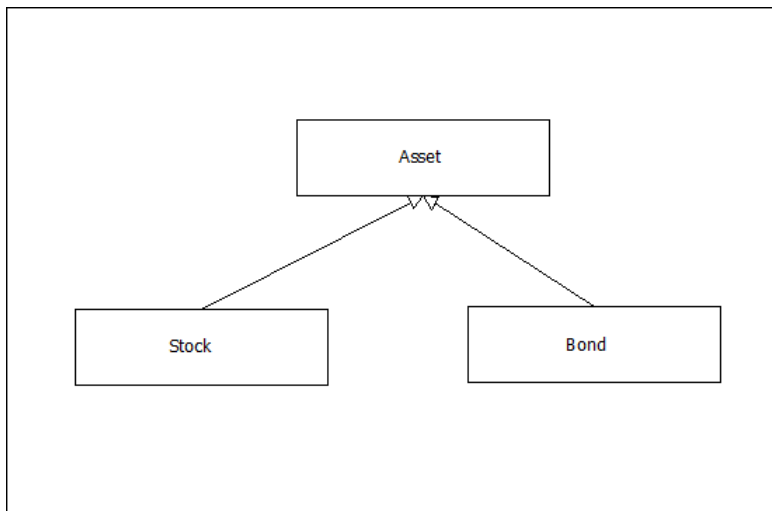


Figure: Asset inheritance hierarchy

Associations and inheritance

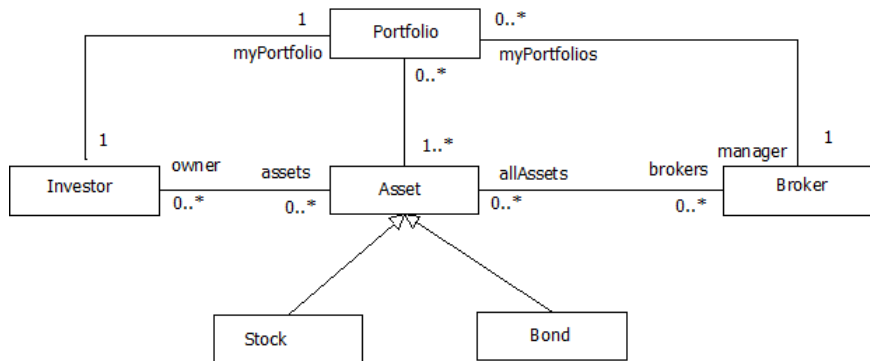


Figure: Portfolio management associations and inheritance

Inheritance

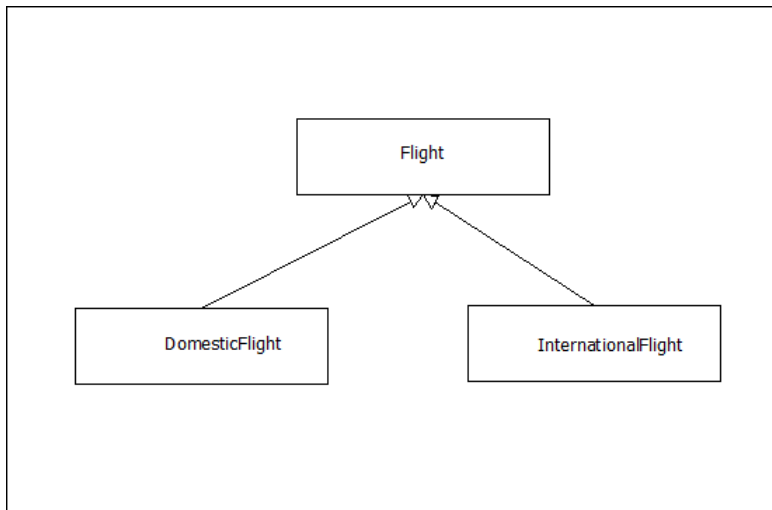


Figure: Inheritance in flight scheduling

Inheritance

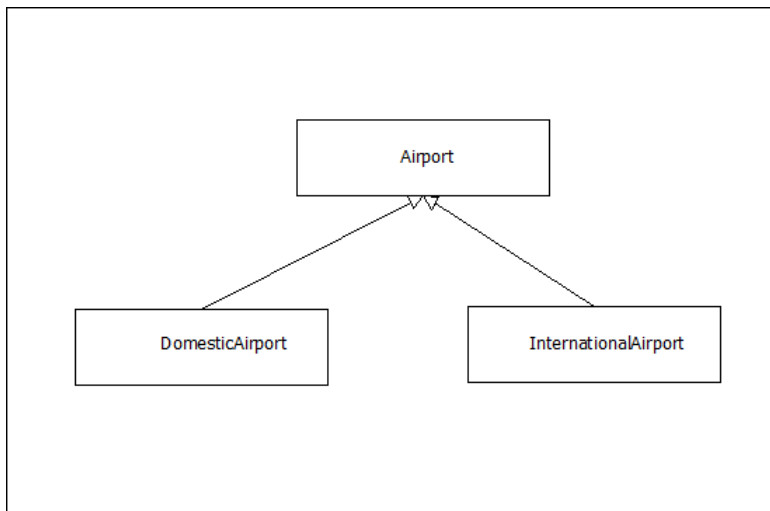


Figure: Airport inheritance hierarchy

Associations and inheritance

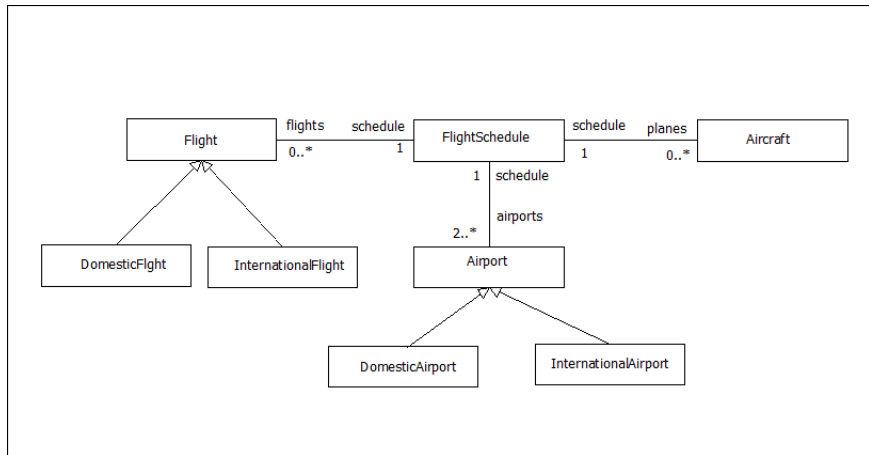


Figure: Flight scheduling associations and inheritance

Behavioral specifications

- Messages
- Sequence diagrams
- Procedural specifications
- Declarative specifications

Sequence diagrams

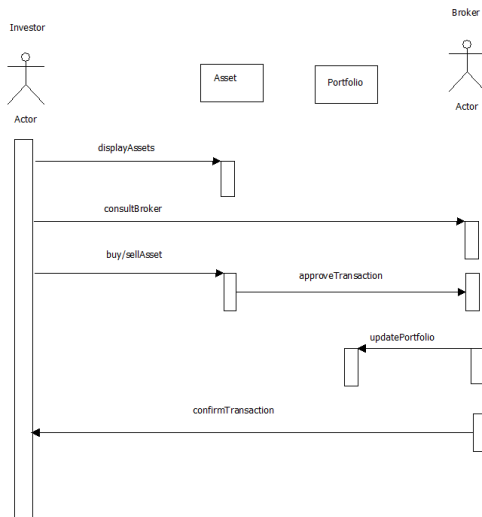


Figure: Investment sequence diagram

Sequence diagrams

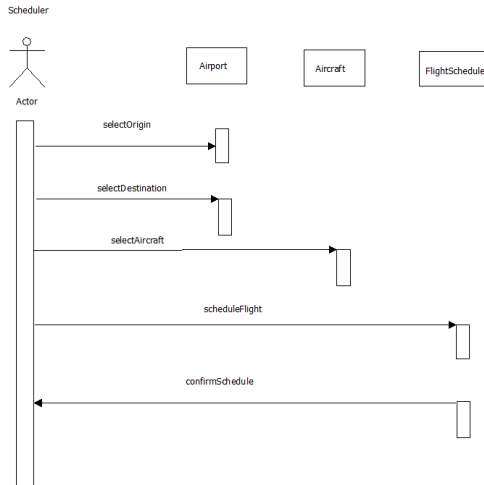


Figure: Flight scheduling sequence diagram

Course management application

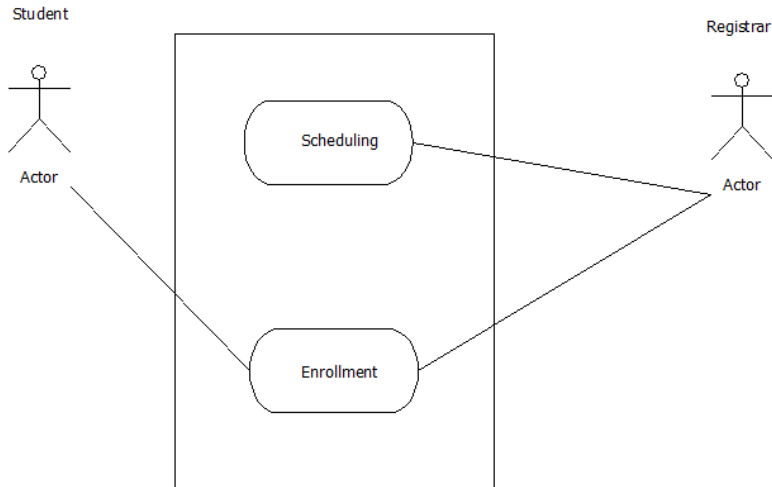


Figure: Course management application

Schedule course use case

Use case: ScheduleCourse

Entities: Course, Registrar

Actors: Registrar

Constraints:

Preconditions: not courseScheduled, instructorAvailable,
classroomAvailable

Postconditions: courseScheduled

Frame: Schedule of all other courses unaffected

Delete course use case

Use case: DeleteCourse

Entities: Course, Registrar

Actors: Registrar

Constraints:

Preconditions: courseScheduled

Postconditions: not courseScheduled

Frame: All other scheduled courses unaffected

Associations

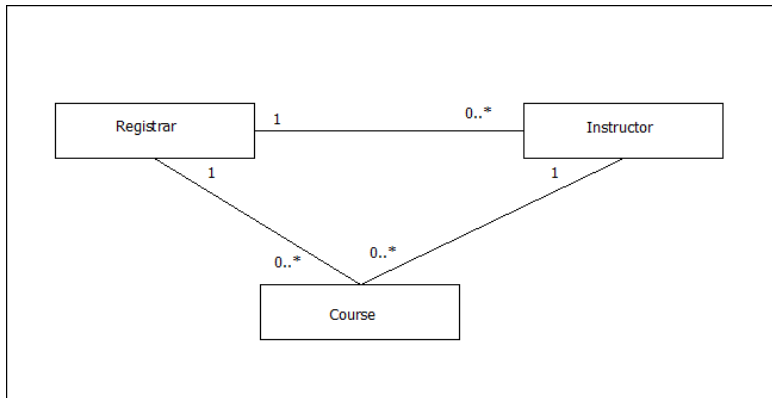


Figure: Associations for ScheduleCourse use case

Course management use cases

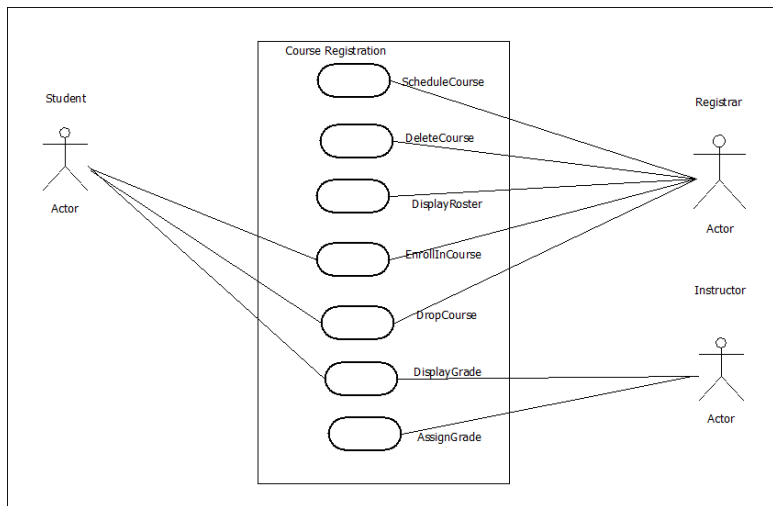


Figure: Course registration use cases

Enroll in course use case

Use case: EnrollInCourse

Entities: Student, Course, Registrar

Actors: Student, Registrar

Constraints:

Preconditions: courseOpen, prerequisitesSatisfied

Postconditions: enrolledInCourse

Frame: All other enrollments unaffected

Drop course use case

Use case: DropCourse

Entities: Student, Course, Registrar

Actors: Student, Registrar

Constraints:

Preconditions: enrolledInCourse, withinDropPeriod

Postconditions: not enrolledInCourse

Frame: All other enrollments unaffected

Enroll/drop associations

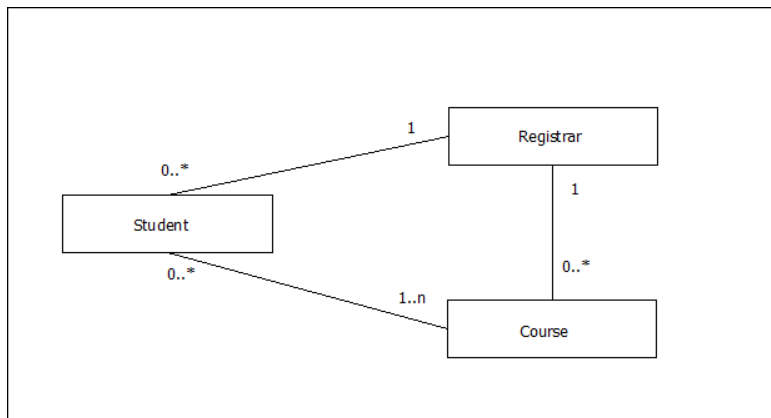


Figure: Enroll in course association diagram

Sequence diagram

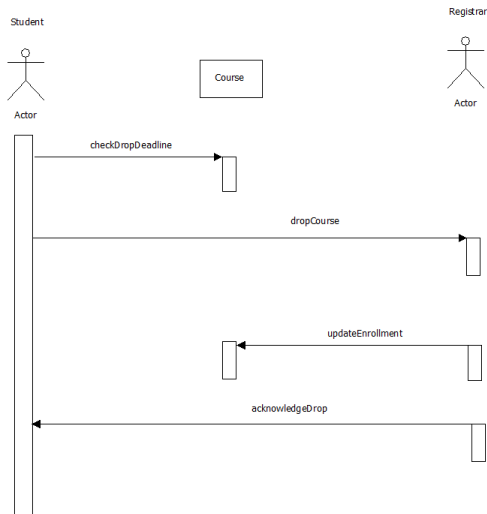


Figure: Drop course sequence diagram

Inheritance

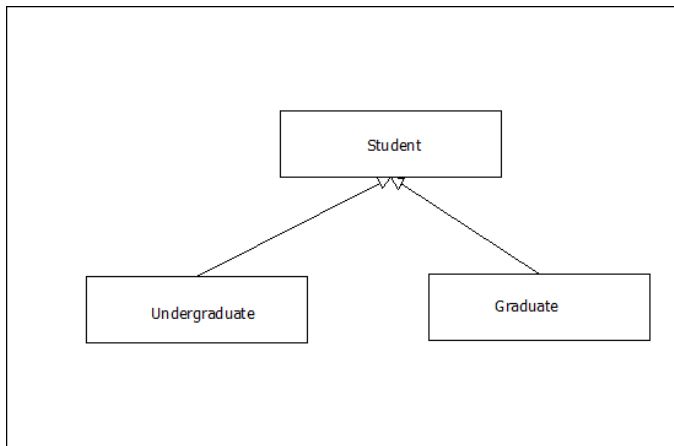


Figure: Inheritance

Inheritance

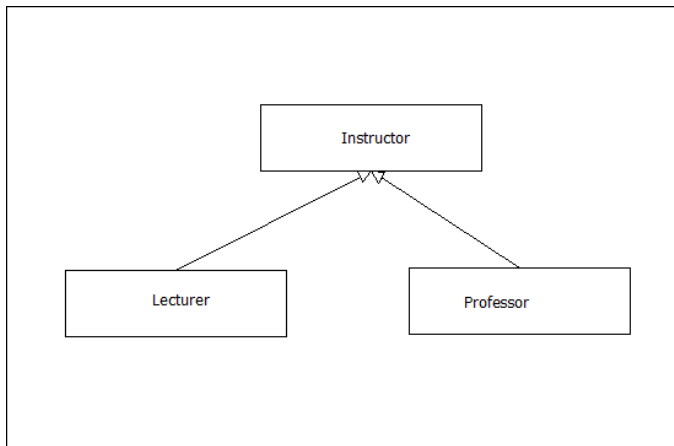


Figure: Inheritance

Associations and inheritance

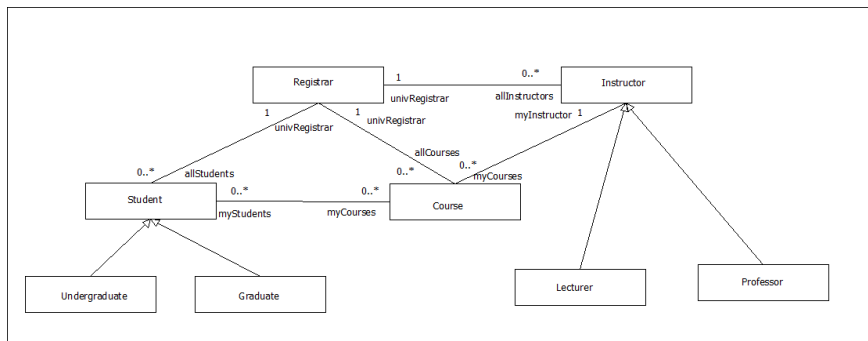


Figure: Course management associations and inheritance

Aggregations

- Complex objects
- Weak aggregation
- Strong aggregation

Aggregation

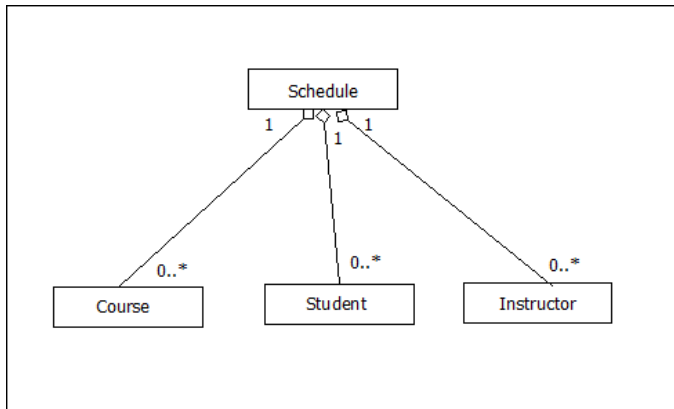


Figure: Course schedule as an aggregate entity type

Aggregation

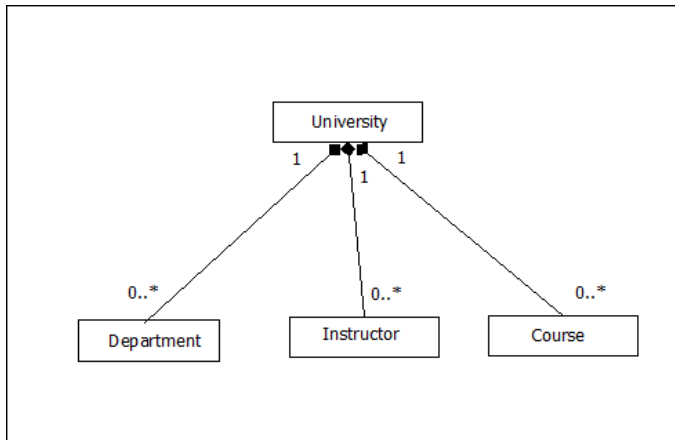


Figure: University as an aggregate entity type

Aggregation

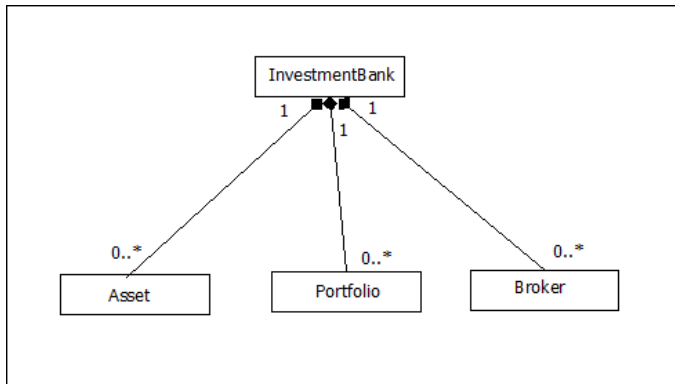


Figure: Investment bank as an aggregate entity type

Aggregation

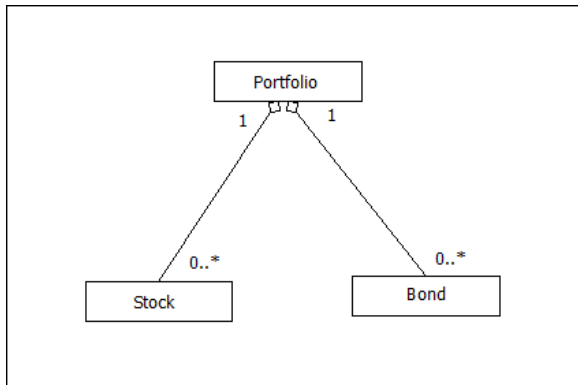


Figure: Portfolio as an aggregate entity type

Entity types as interfaces

- UML interfaces
- Signatures of operations
- Representing associations
- Inheritance

Entity types as interfaces

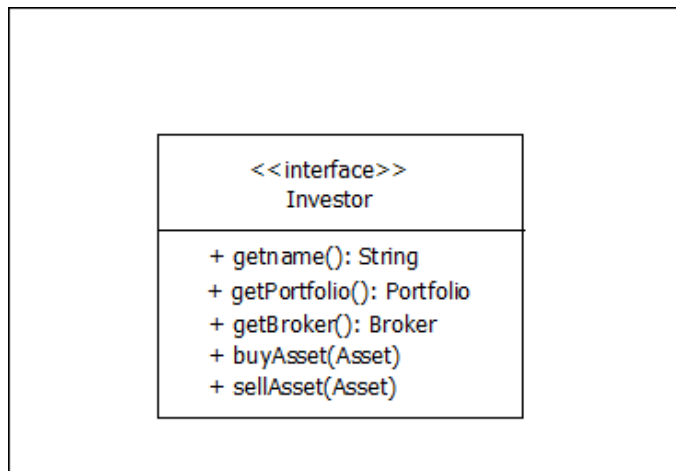


Figure: Investor interface

Entity types as interfaces

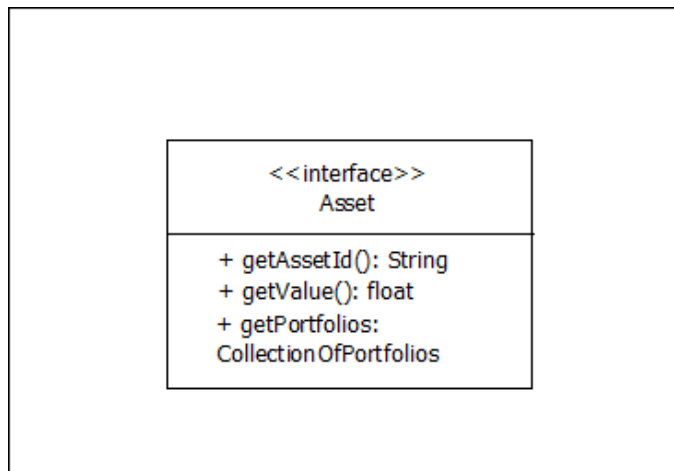


Figure: Asset interface

Entity types as interfaces

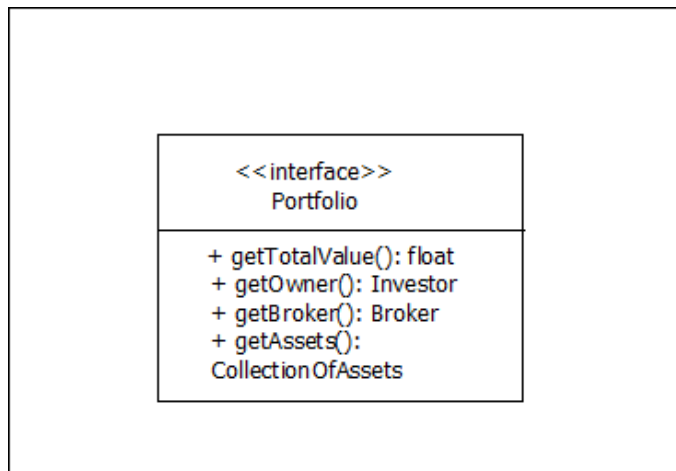


Figure: Portfolio interface

Entity types as interfaces

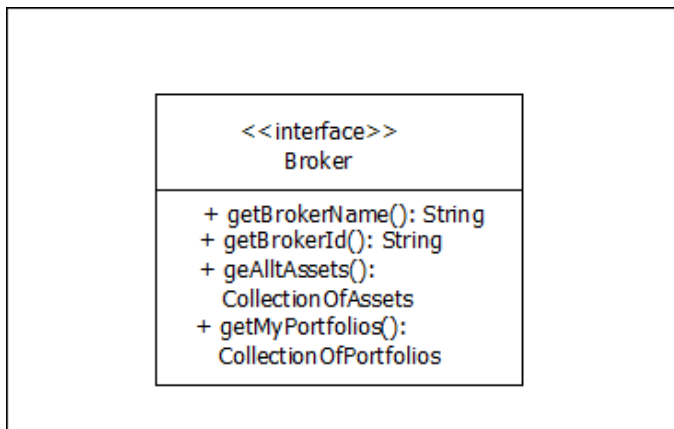


Figure: Broker interface

Entity types as interfaces

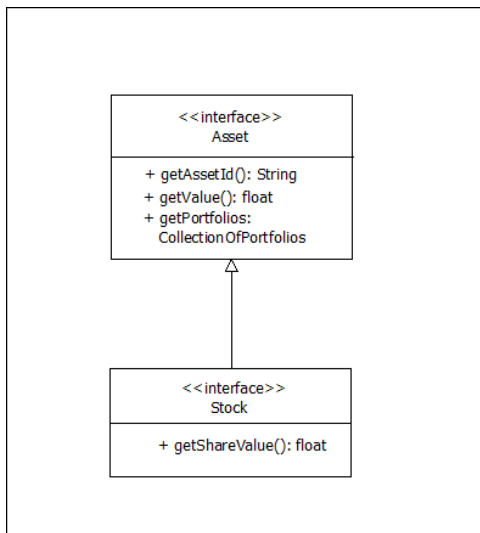


Figure: Stock interface

Entity types as interfaces

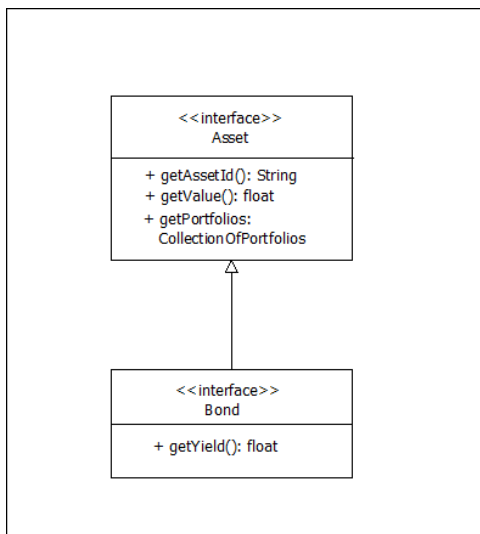


Figure: Bond interface

Entity types as classes

- UML classes
- Attributes or fields
- Method signatures
- Implementing interfaces
- Single and multiple inheritance

Entity types as classes

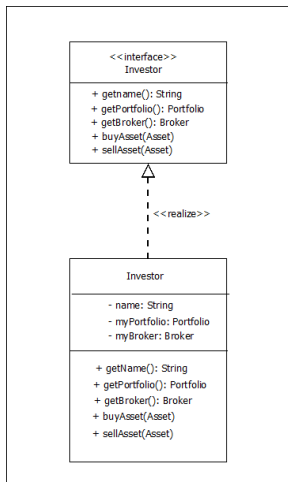


Figure: Investor class

Entity types as classes

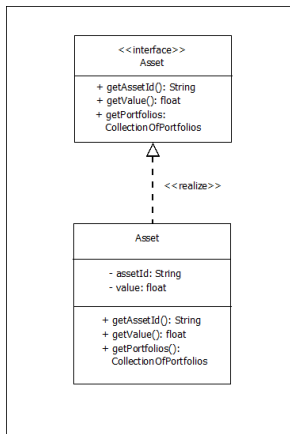


Figure: Asset class

Entity types as classes

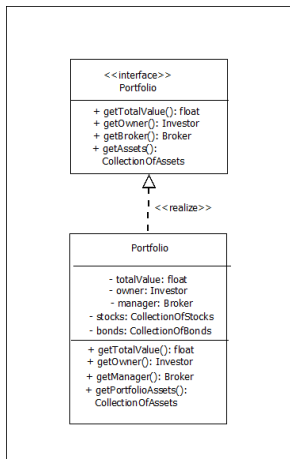


Figure: PortfolioClass

Entity types as classes

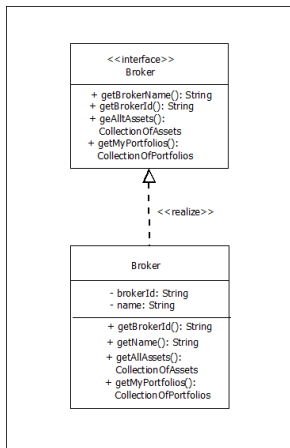


Figure: Broker class

Entity types as classes

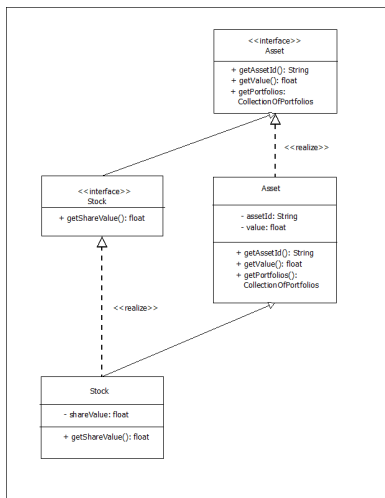


Figure: Classes and interfaces