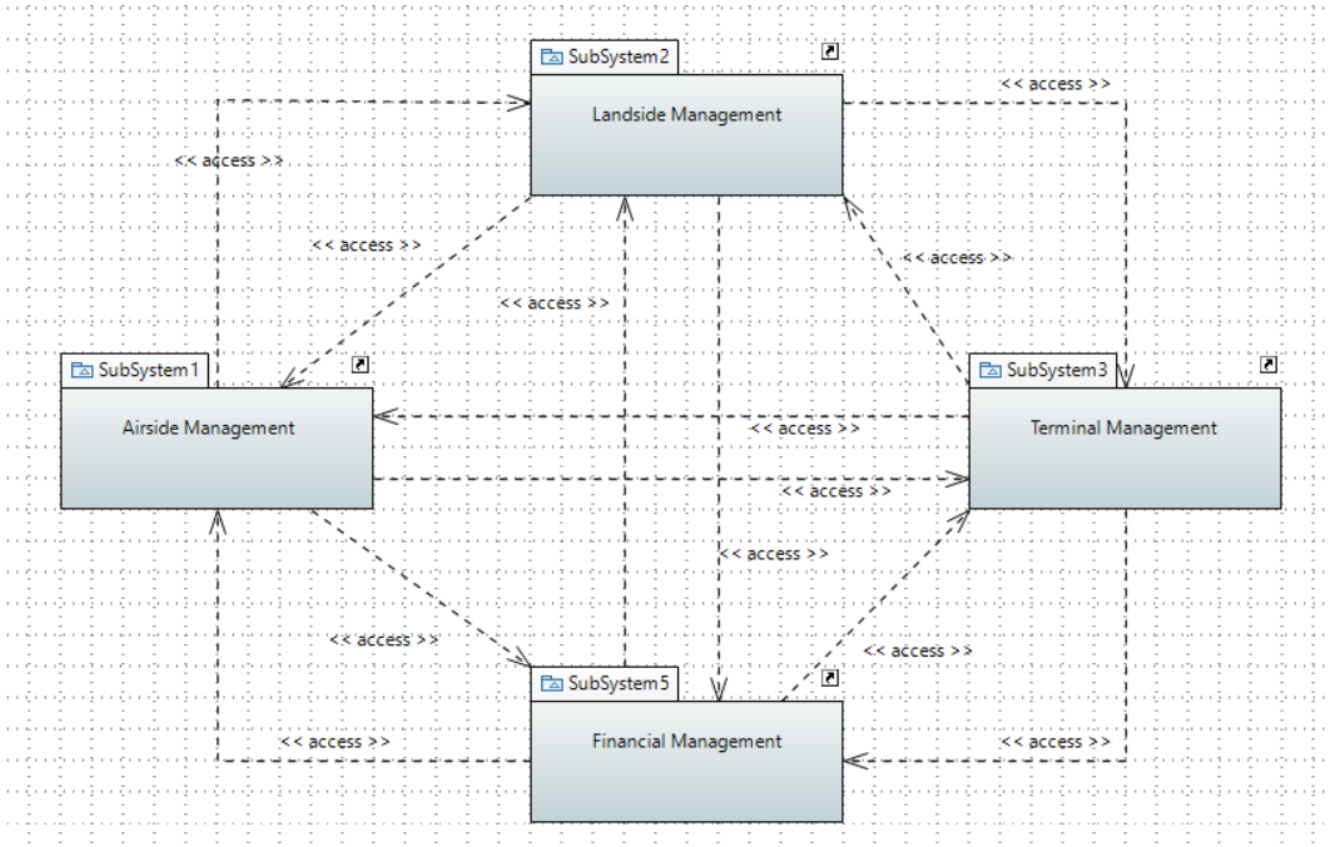
JKU - JOHANNES KEPLER UNIVERSITÄT LINZ | JKU LINZ
ISSE - Institute of Software Systems Engineering

Course: UE Software Engineering	**Course ID:** 343.309	**Semester:** 2021W
Simon Primetzhofer 11942035 simon.primetzhofer@live.at	Stefan Haslhofer 11908757 haslhofer.stefan@gmail.com	
Kaan Baylan 11910231 kaan.baylan28@gmail.com	Jonas Reichhardt 11908755 office@jonasreichhardt.at	

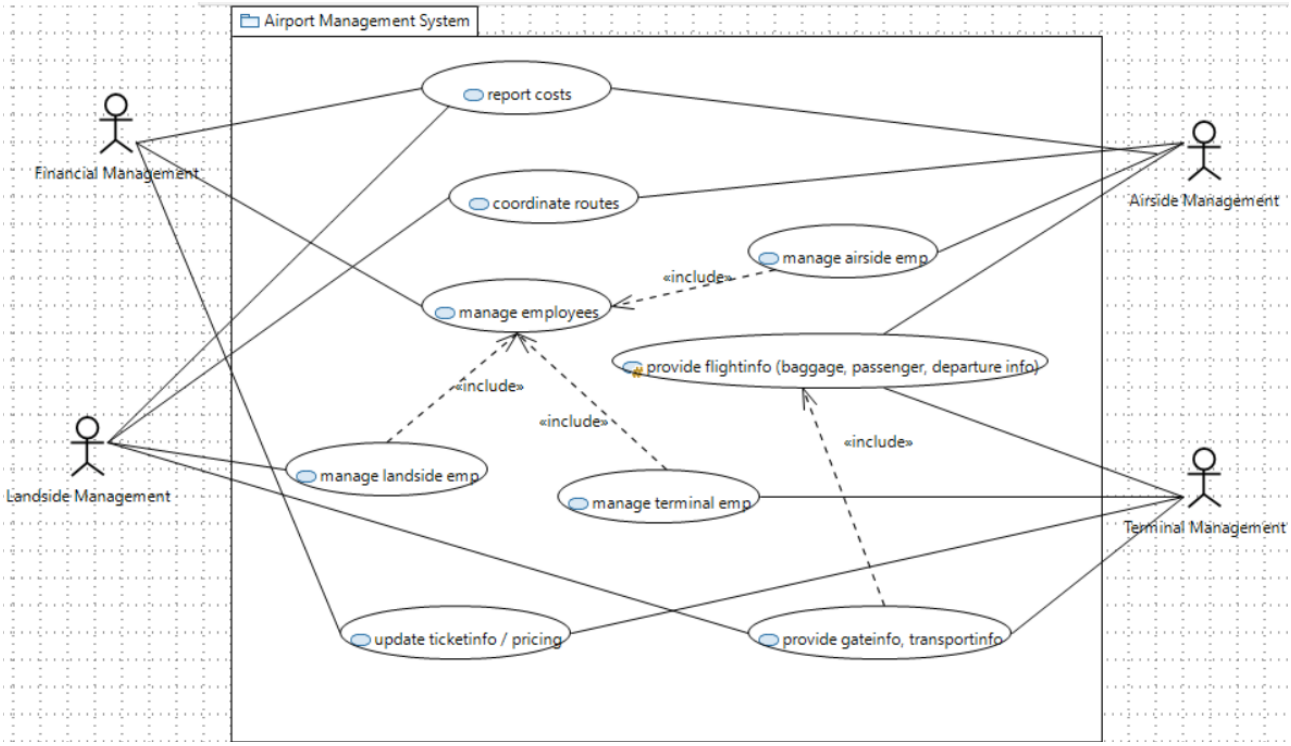
Milestone 2 Report – Team 2

1. Overall Architecture



The airport management system is based on the four subsystems like they are displayed in the above package diagram. Every subsystem directly communicates with all other subsystems since the control system management as a central part is not available in this case. This specifically means that Airside and Landside management provide their vehicles' route information and consume them from the respective other part instead of having the control system management between them. We can see that every subsystem provides and consumes information from every other subsystem which makes it quite interconnected.

2. System



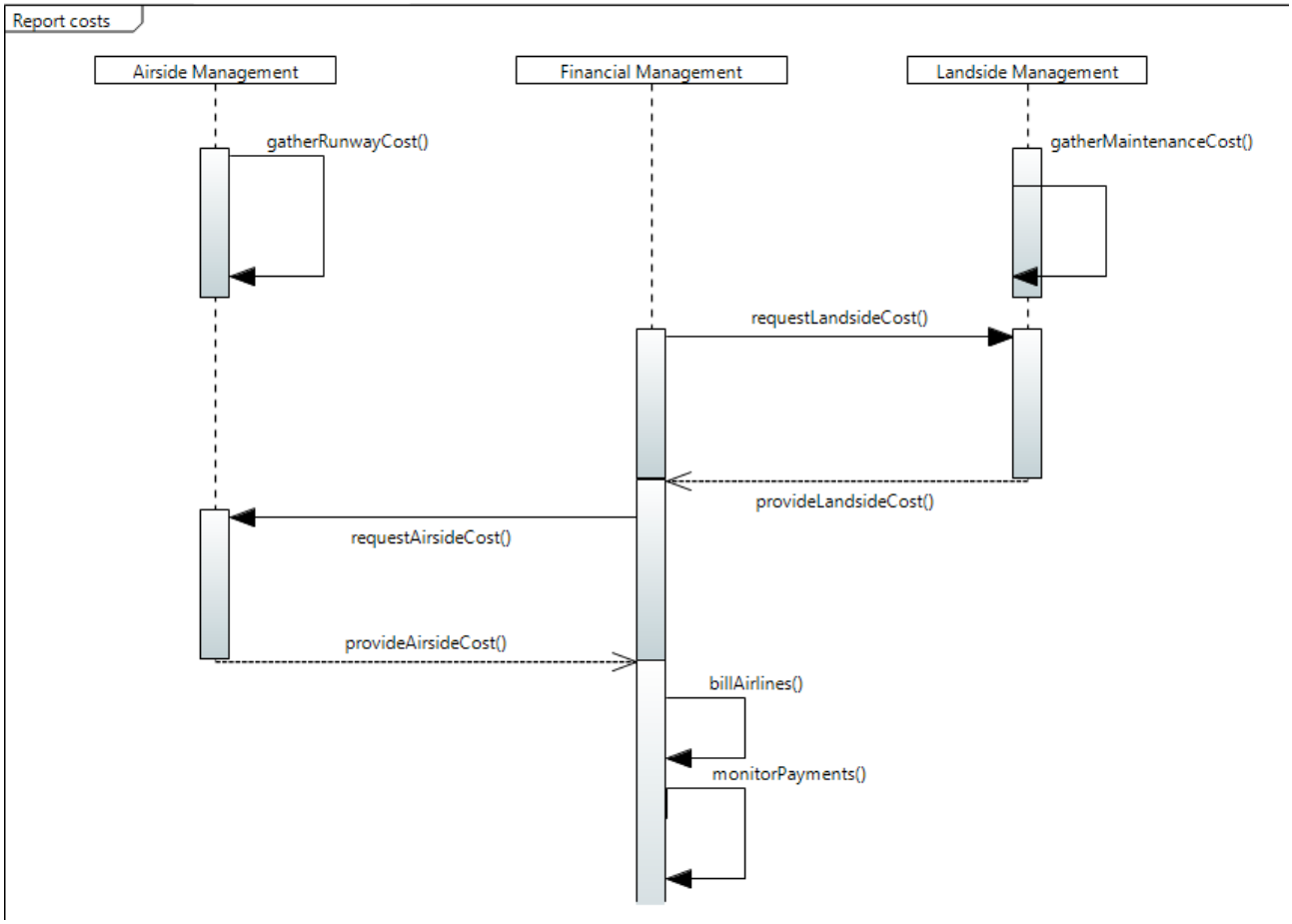
Financial Management is responsible for gathering information about the flights from the airlines, calculating the ticket prices, booking facilities for airlines, assigning employees to the respective management and managing maintenance of vehicles, fuel, material and equipment.

Airside Management is responsible for landings and takeoffs of aircrafts, monitoring flights and managing the traffic in the air as well as on the ground.

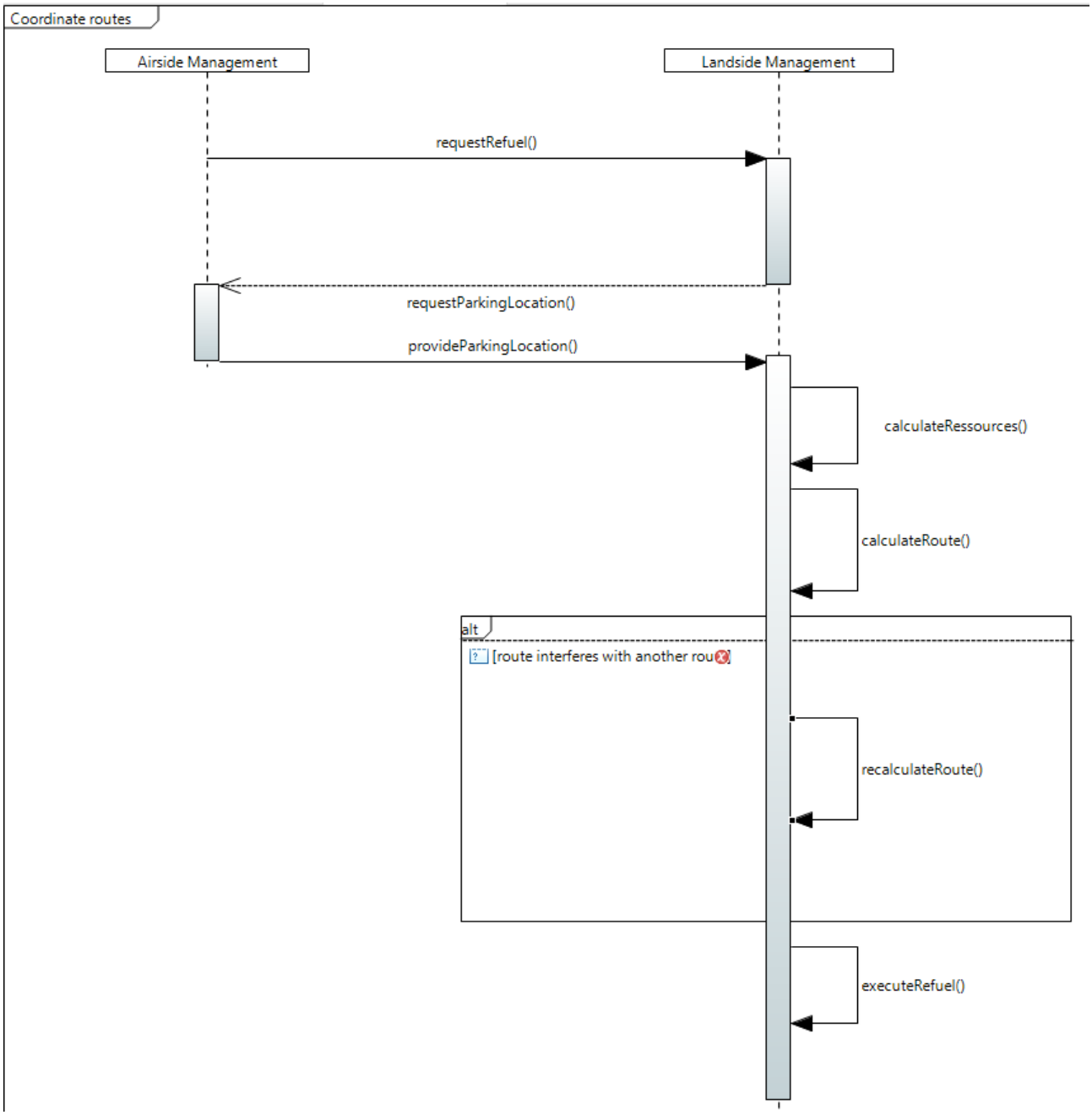
Landside Management is responsible for coordination of land-vehicles such as baggage carts and buses transporting luggage and persons from either gate to plane or vice versa. Furthermore, it is tasked to react to emergency calls as well as providing maintenance services to each plane.

Terminal Management is responsible for checking in passengers, handling all baggage related tasks, performing security checks and also displaying flight information publicly for all passengers.

Report costs
Step 1: Landside management gathers all costs from refueling, maintenance and facility usage
Step 2: Airside management gathers all costs for using the runway
Step 3: Financial management requests a cost report from Landside management
Step 4: Financial management requests a cost report from Airside management
Step 5: Airside and Landside management send cost report
Step 6: Financial Management sends out bills to airlines
Step 7: Financial Management monitors payment of the bills

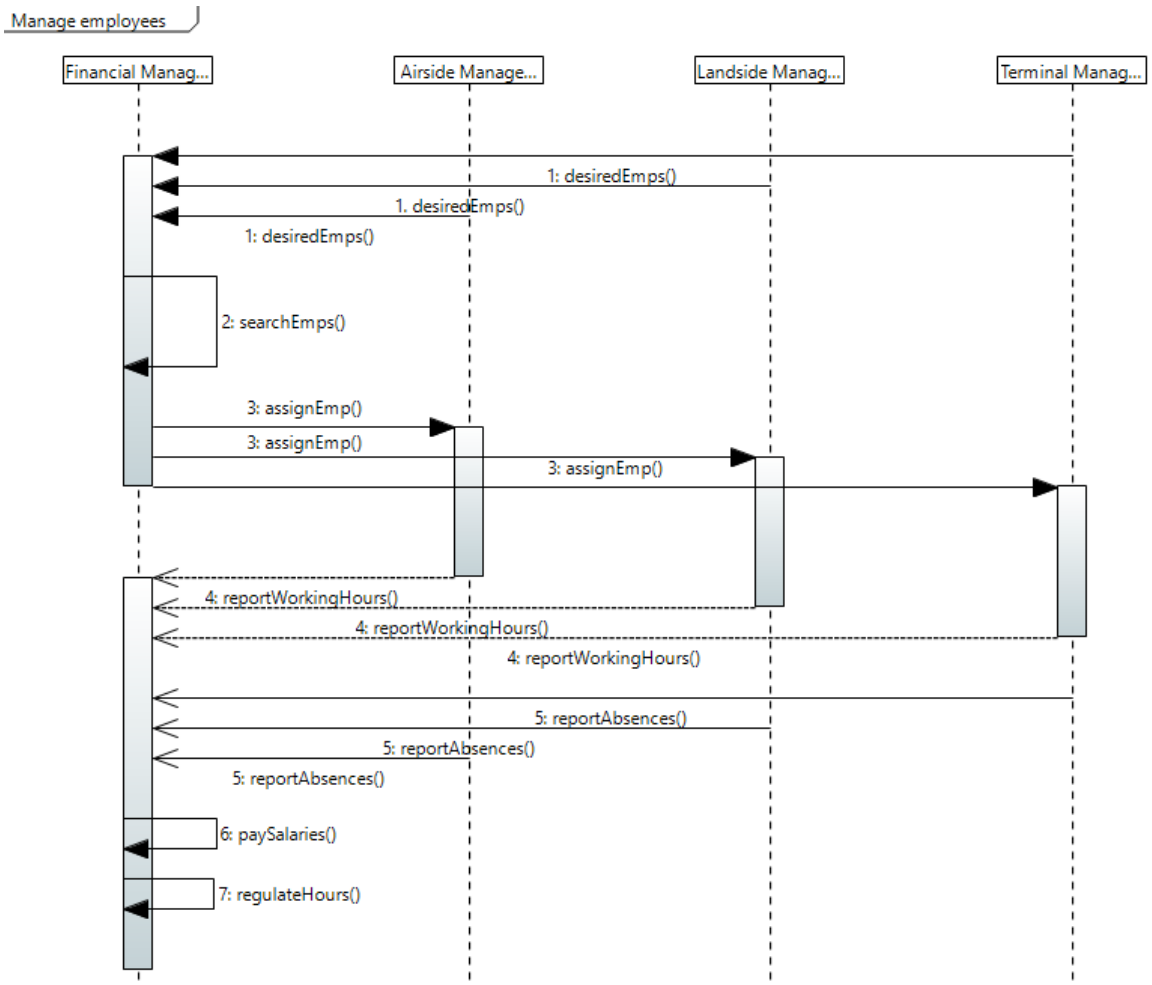

Report costs
Airside Management
Financial Management
Landside Management
gatherRunwayCost()
gatherMaintenanceCost()
requestLandsideCost()
provideLandsideCost()
requestAirsideCost()
provideAirsideCost()
billAirlines()
monitorPayments()

Coordinate routes
Step 1: Airside management requests fuel truck, stair truck, etc. from Landside Management
Step 2: Landside Management requests parking location of aircraft
Step 3: Airside Management provides parking location and arrival/departure time
Step 4: Landside Management calculates needed resources
Step 5: Landside Management plans best route for vehicles
Step 6: Landside Management assigns driver
Step 7: Landside Management executes requested service

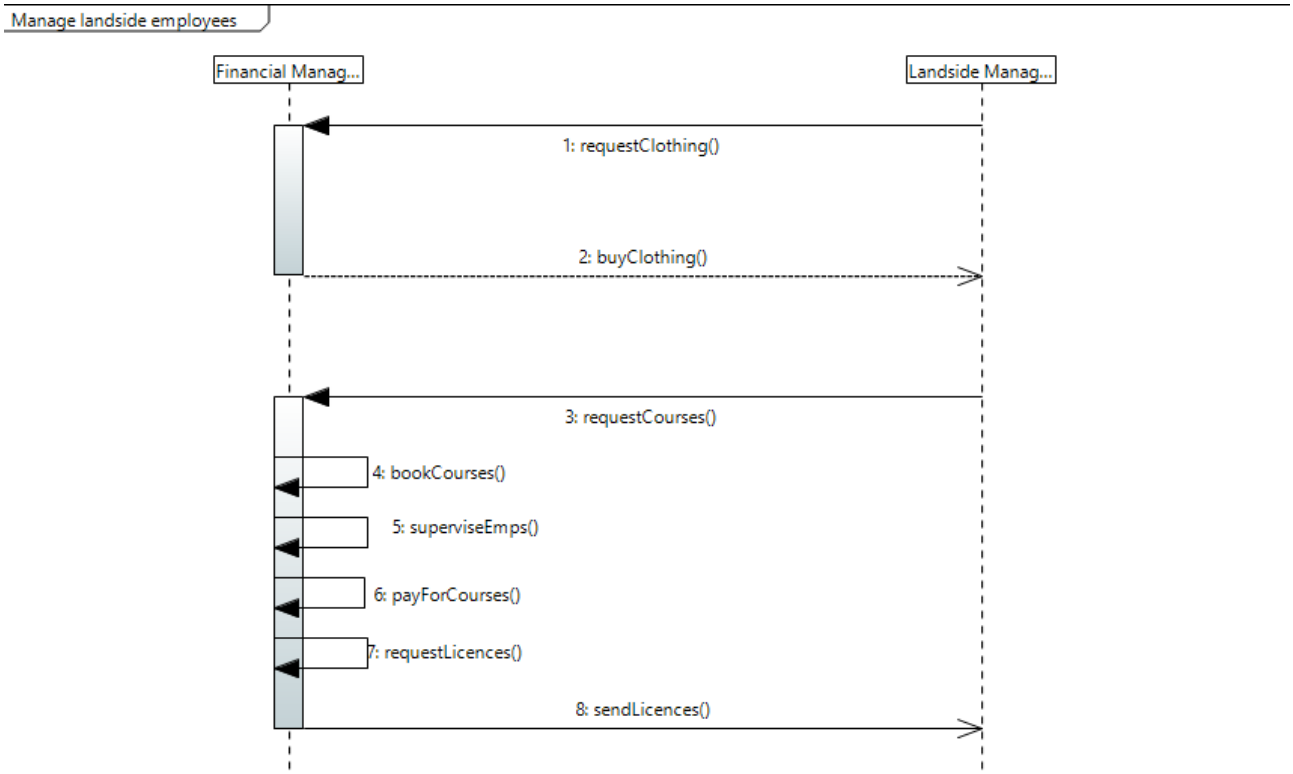


Manage employees
Step 1: Landside-, Airside and Terminal Management report desired amount of employees to Financial Management
Step 2: Financial Management searches for suitable employees according to knowledge and abilities
Step 3: Financial Management assigns employees to Landside, Airside and Terminal Management

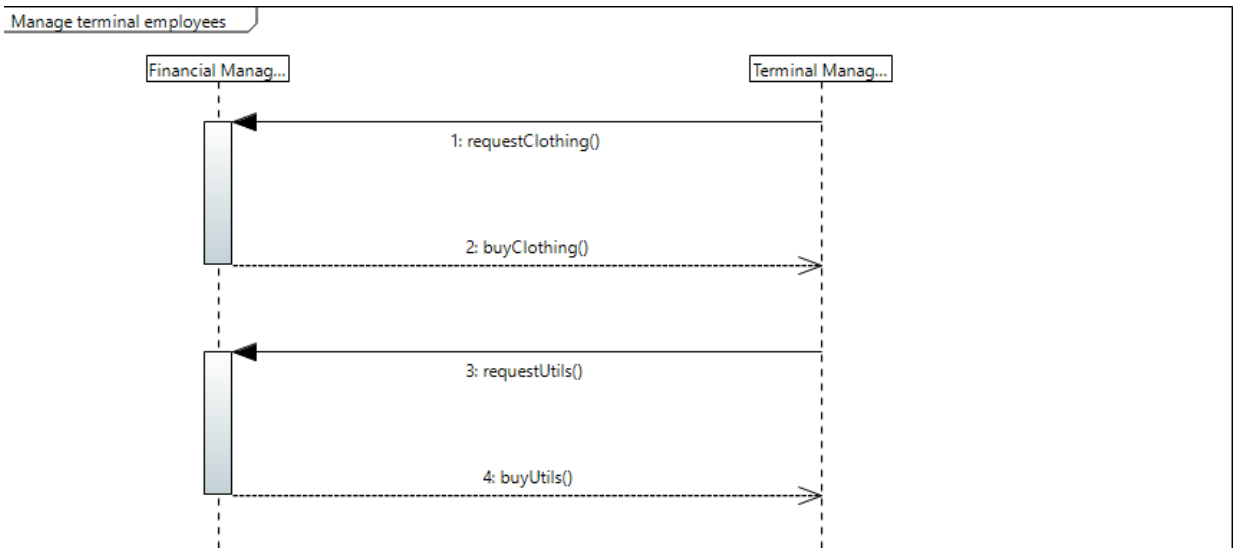
Step 4: Landside, Airside and Terminal Management report working hours of employees to Financial Management
Step 5: Landside, Airside and Terminal Management report absences to Financial Management
Step 6: Financial Management pays salaries to employees
Step 7: Financial Management regulates hours of labor according to working hours



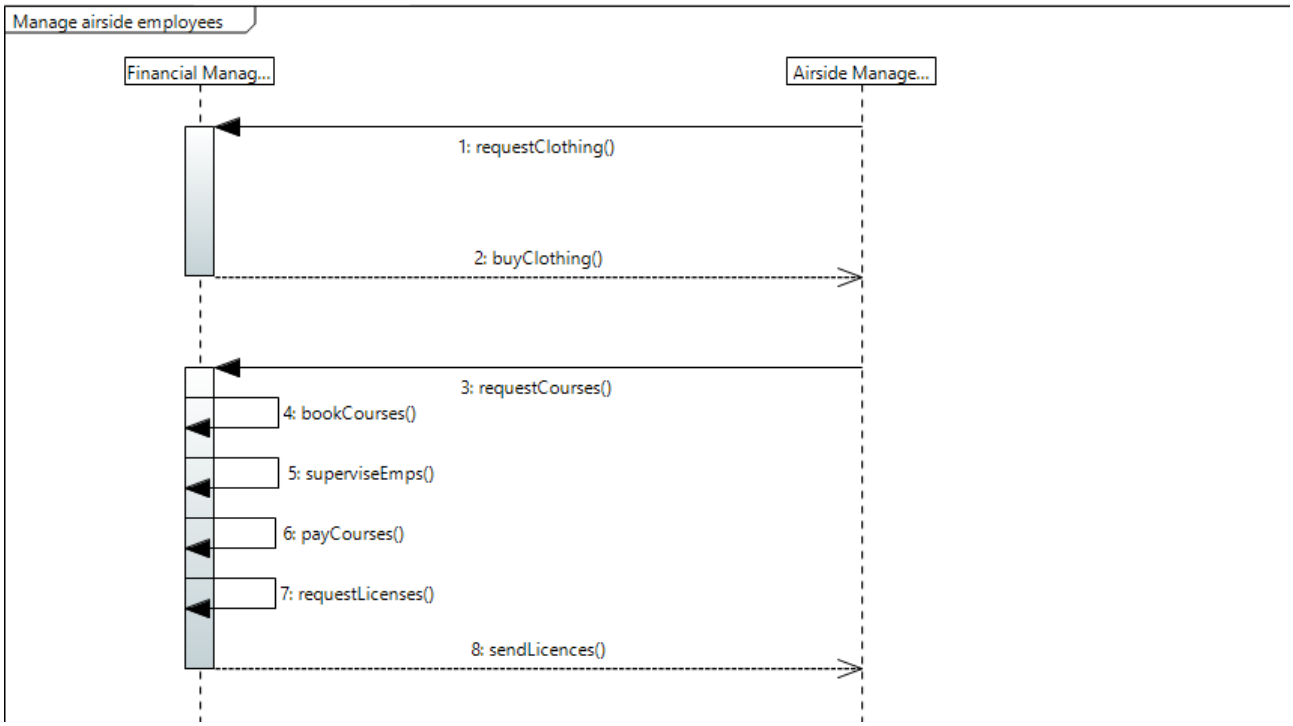
Manage landside employees
Step 1: Landside Management requests security clothing for its employees from Financial Management
Step 2: Financial Management buys requested clothing from an external vendor
Step 3: Landside Management requests driving licence courses for the drivers for being able to drive the vehicles
Step 4: Financial Management books driving instructor and coordinates appointments
Step 5: Financial Management supervises attendance of employees at appointments
Step 6: Financial Management pays for course and requests driving licence for Landside Management employees



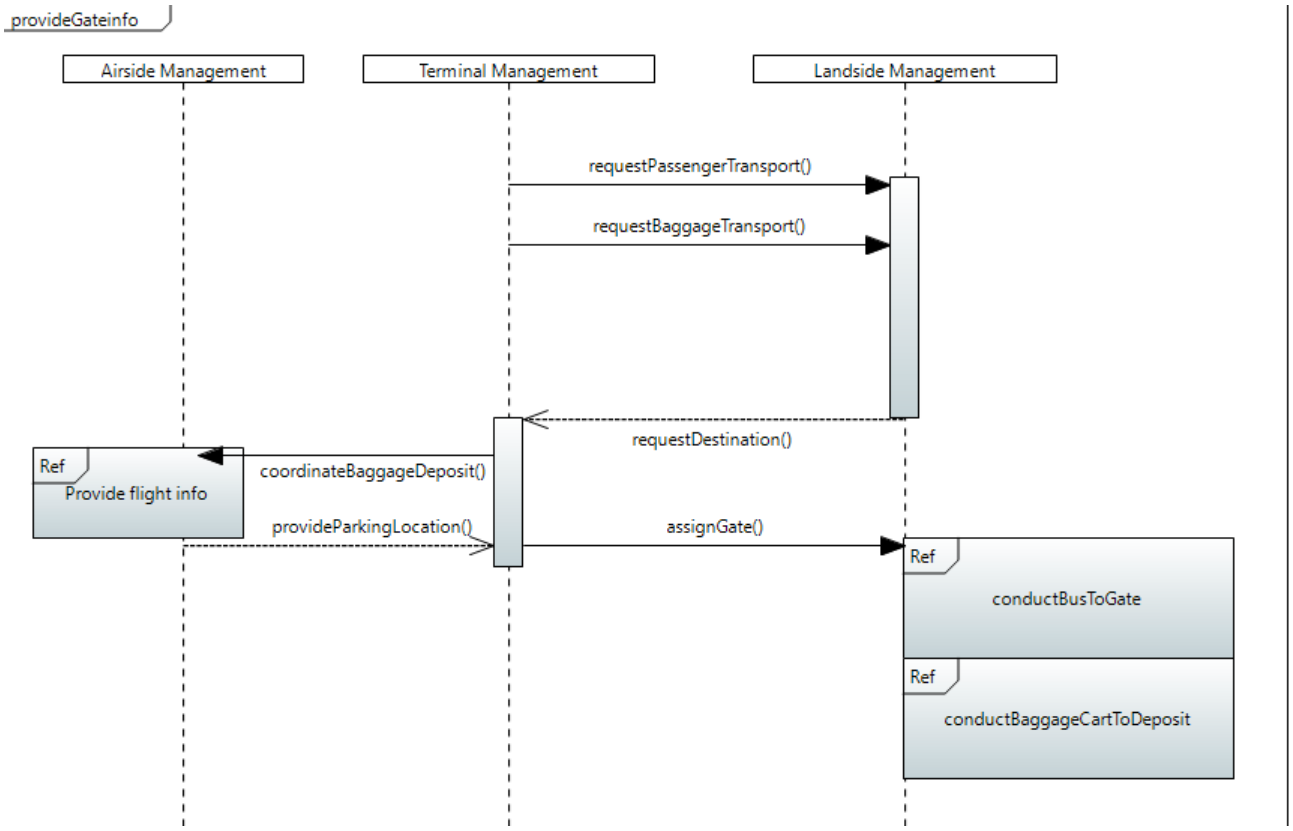
Manage terminal employees
Step 1: Terminal Management requests adequate clothing for employees from Financial Management
Step 2: Financial Management buys requested clothing from an external vendor
Step 3: Terminal Management requests computers with access to flight information and passenger data in order to do work
Step 4: Terminal Management requests security utils (metal detector, scanner, ...) from Financial Management
Step 5: Terminal Management requests big screens for display flight information from Financial Management
Step 6: Financial Management provides requested utils and buys the from an external vendor if needed



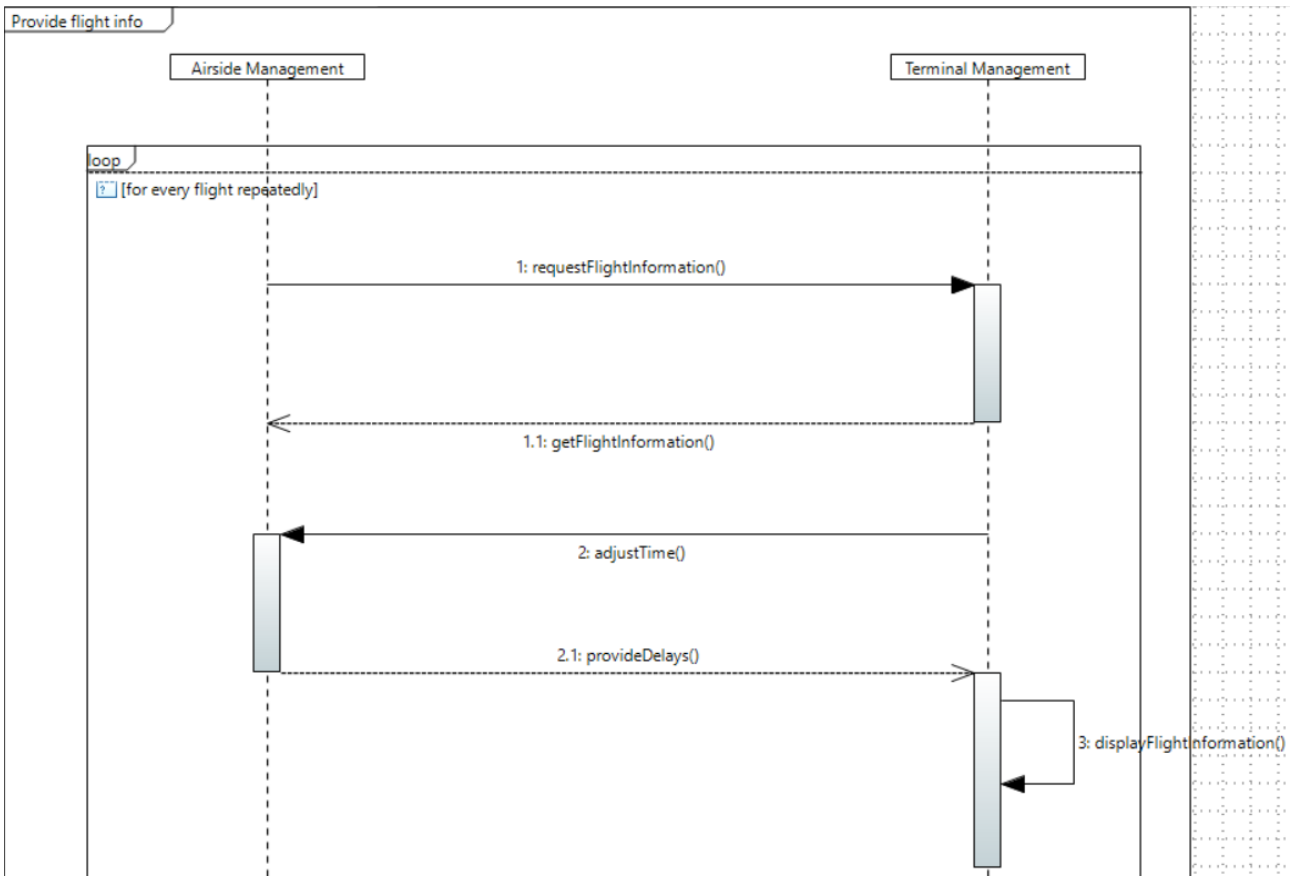
Manage airside employees
Step 1: Landside Management requests security clothing for its employees from Financial Management
Step 2: Financial Management buys requested clothing from an external vendor
Step 3: Airside Management requests courses for air traffic controllers, tower personnel, etc. from Financial Management in order to ensure operability
Step 4: Financial Management books instructors and coordinates appointments
Step 5: Financial Management supervises attendance of employees at appointments
Step 6: Financial Management pays for course and requests driving licence for Airside Management employees



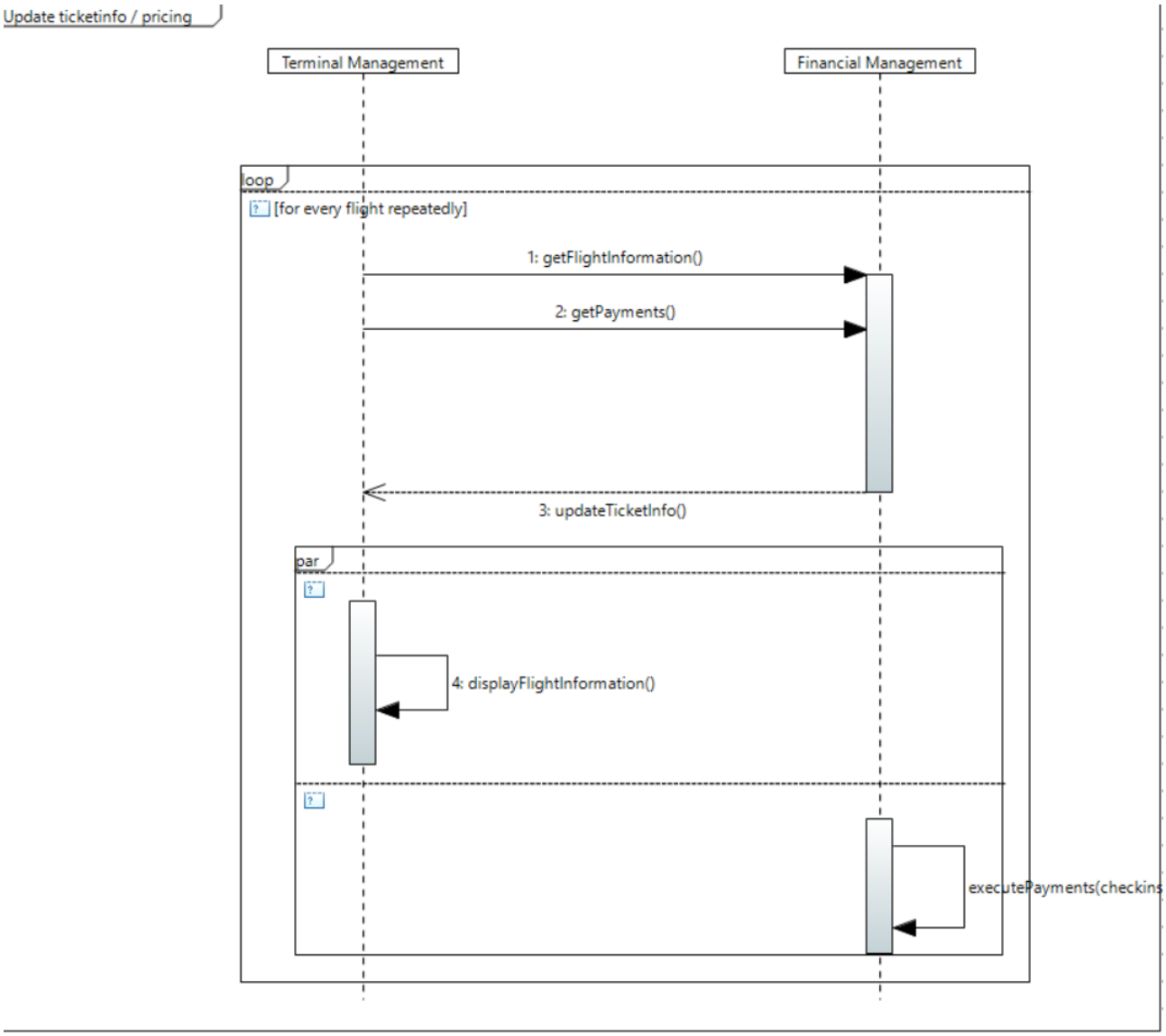
Provide gate and transport information
Step 1: Terminal Management requests passenger/cargo transport from Landside Management
Step 2: Landside Management requests gate and transport information from Terminal Management
Step 3: Terminal Management assigns gate
Step 4: Terminal Management forwards baggage to transportation units
Step 5: Terminal Management and Airside management coordinate flight details to ensure that baggage gets delivered correctly and on time
Step 6: Landside Management transports baggage and passengers to the airplane

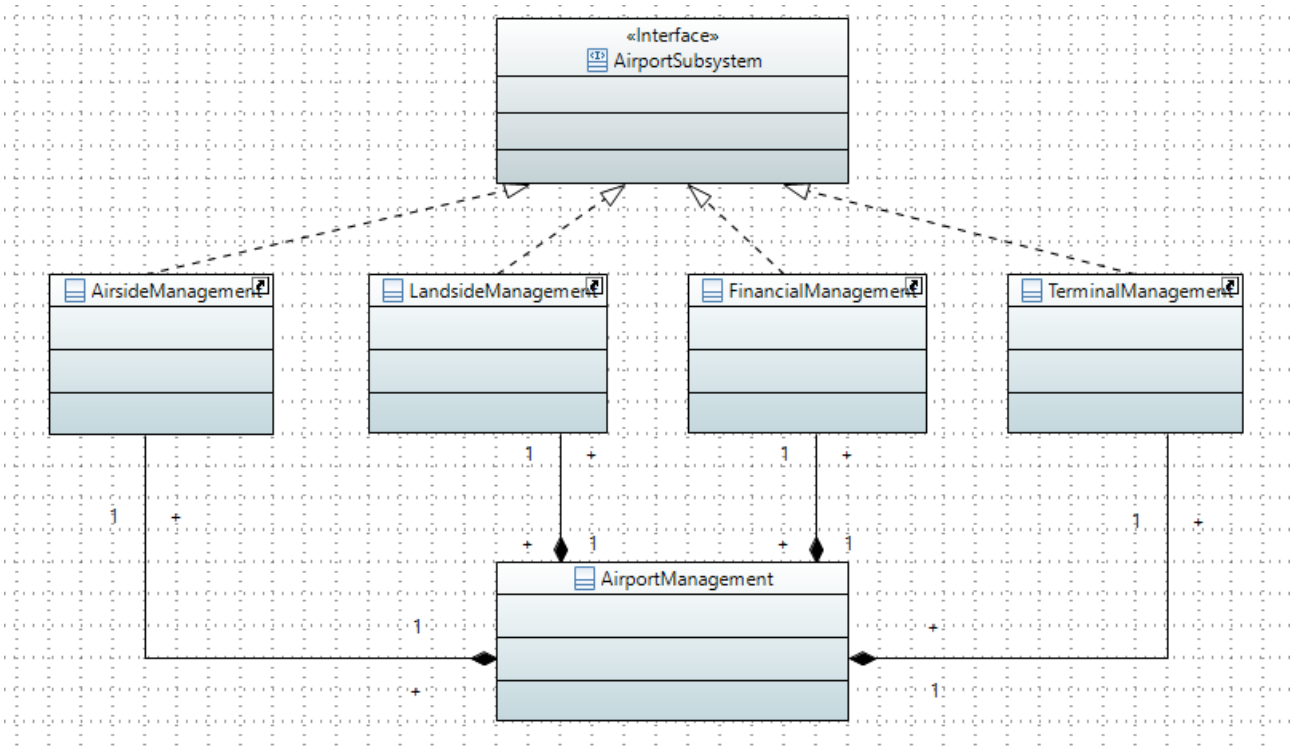


Provide flight information
Step 1: Airside Management requests flight information from Terminal management (scheduled departure/arrival)
Step 2: Terminal Management adjusts departure/landing time according to check-in status of passengers
Step 3: Airside Management provides actual departure/landing time (possible delays)
Step 4: Terminal Management displays possibly changed flight information to passengers
Step 5: Terminal Management coordinates baggage and passenger transportation according to aircraft status



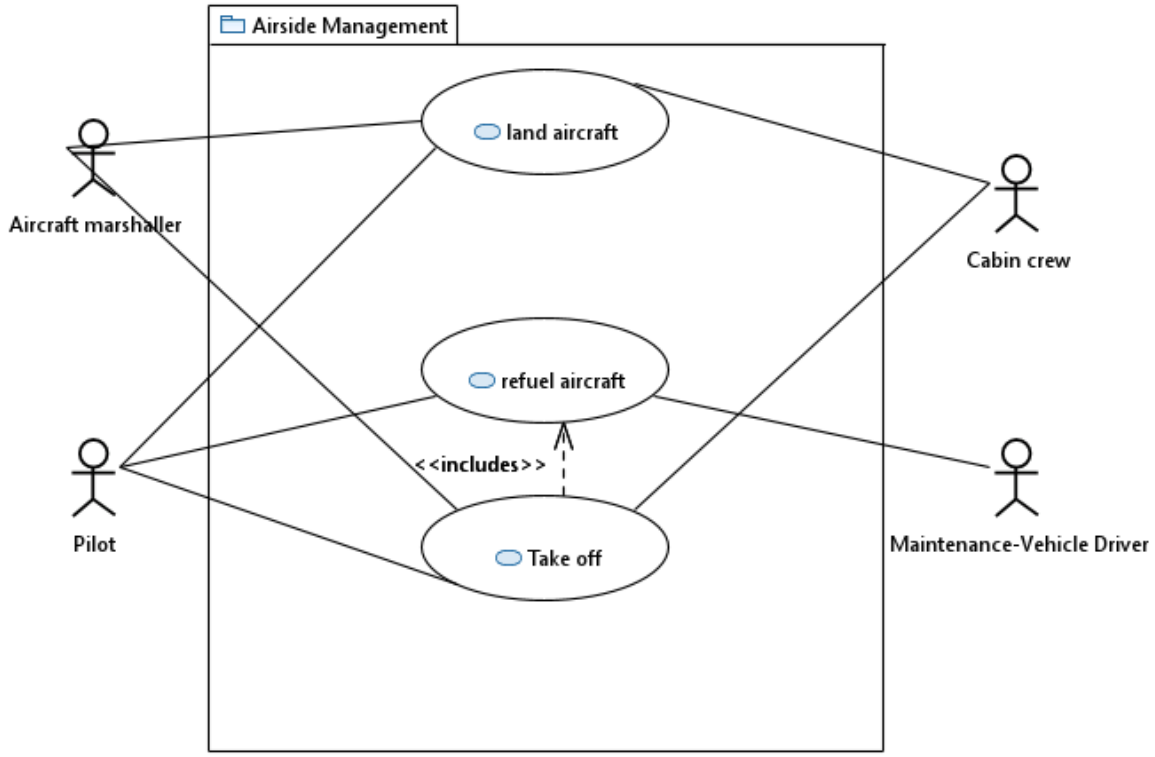
Update ticket information / pricing
Step 1: Terminal Management reports check-in status of flights to Financial Management
Step 2: Terminal Management reports payments (too heavy baggage, better seat category, ...) to Financial Management
Step 3: Financial Management updates ticket prices, departure times and flight states
Step 4: Terminal Management displays updated data on flight information system
Step 5: Financial Management executes payments depending on the payment method (excluding cash since that is paid immediately)




«Interface»
AirportSubsystem
AirsideManagement
LandsideManagement
FinancialManagement
TerminalManagement
AirportManagement
1
+

2.1 Sub-system 1 / Airside management - [Jonas **Reichhardt**]

The airside management handles aircraft related information and actions. For example, aircraft landing and takeoff.

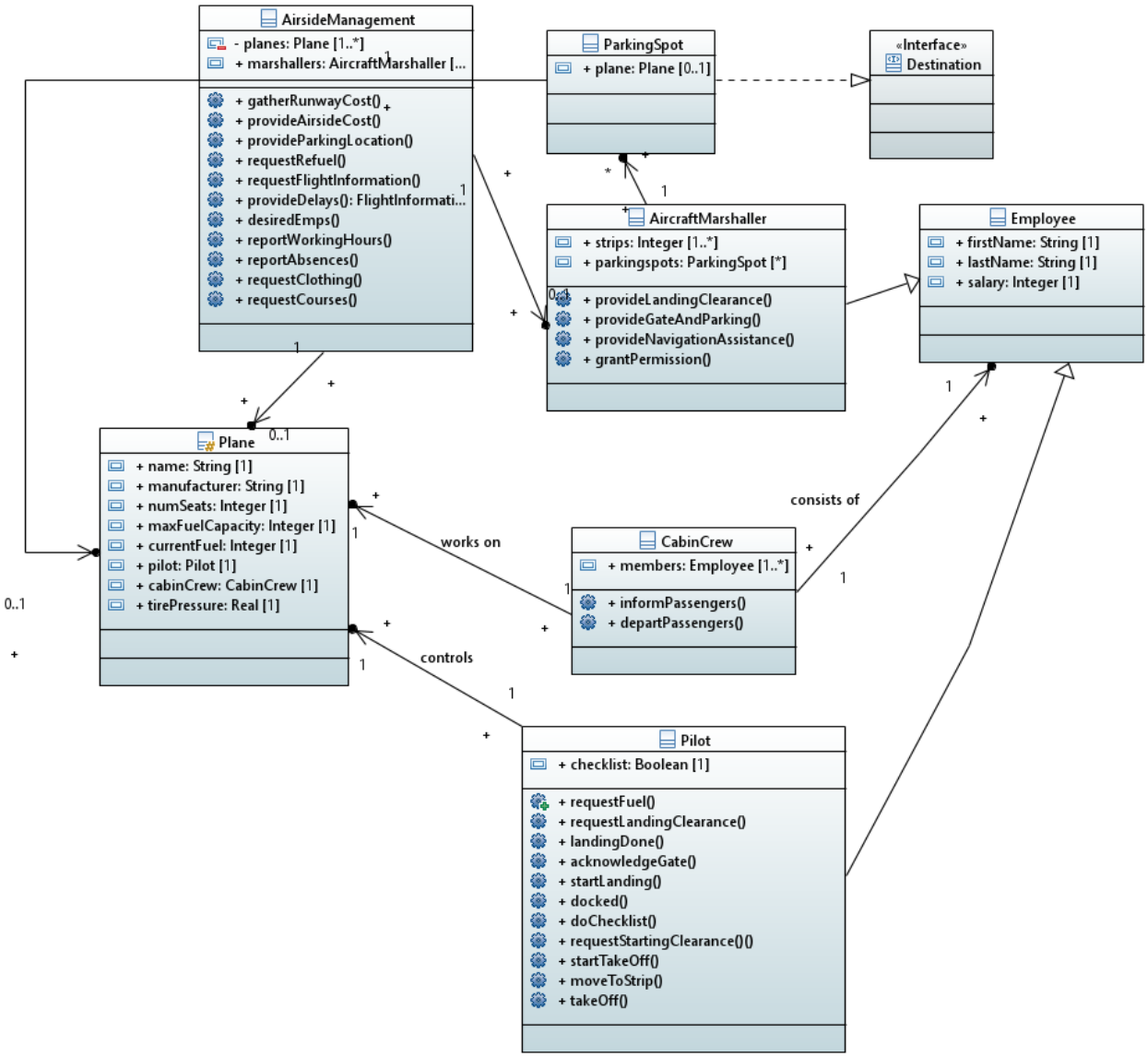


The aircraft marshaller is ground personnel which guides pilots from starting/landing strip to parking spot and vice versa, in our case he also gives landing/starting permission.

On the aircraft the cabin crew manages customer request, safety briefing and it acts as a communication relay between pilot and passengers.

Regardless of the situation pilots are involved in every action in the airside management. They land the aircraft, notify the landside management to get the aircraft refueled and takeoff when all passengers and cargo is loaded.

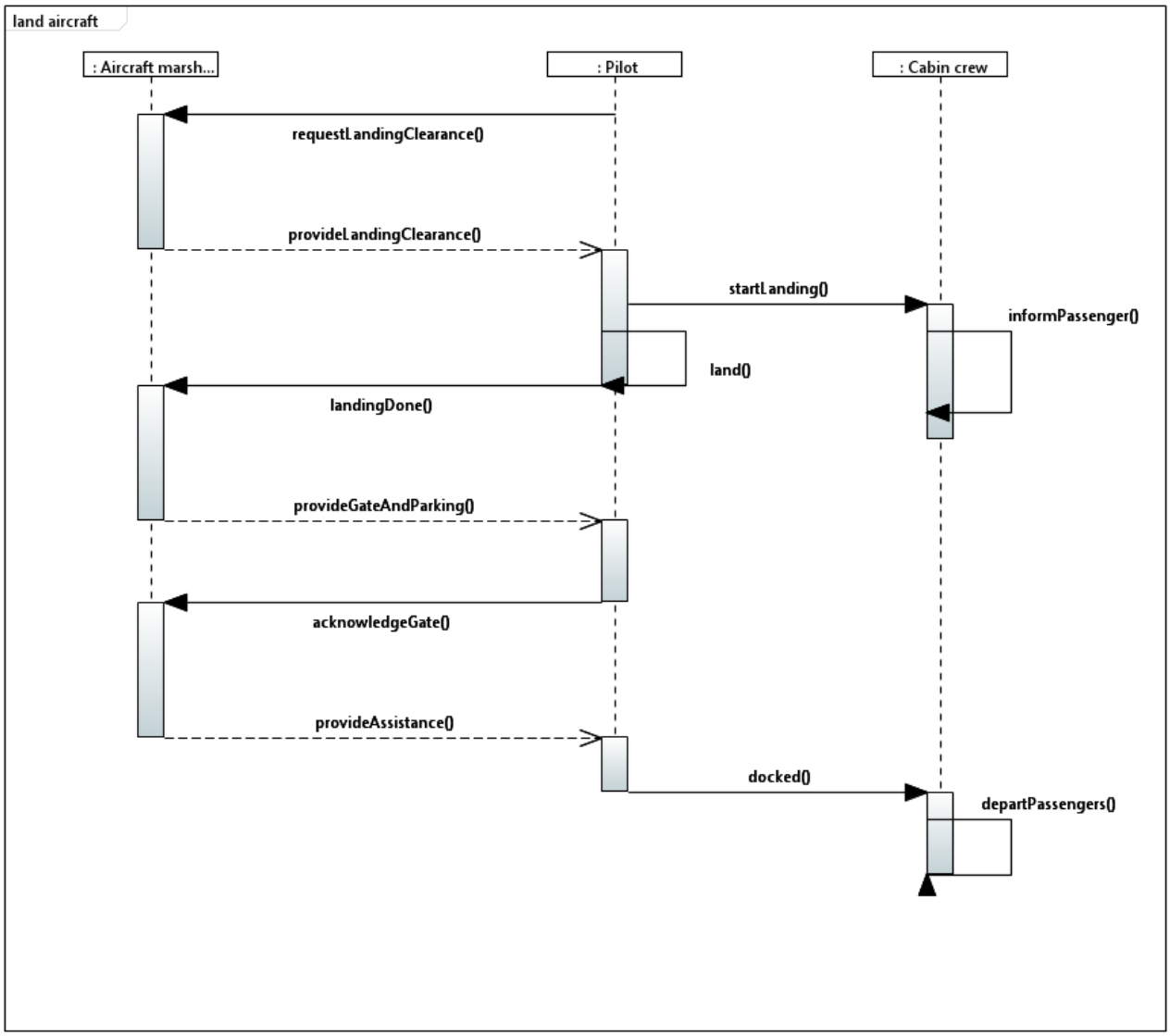
The fuel truck driver is part of the landside management but is necessary to get aircrafts refueled and ready to takeoff.



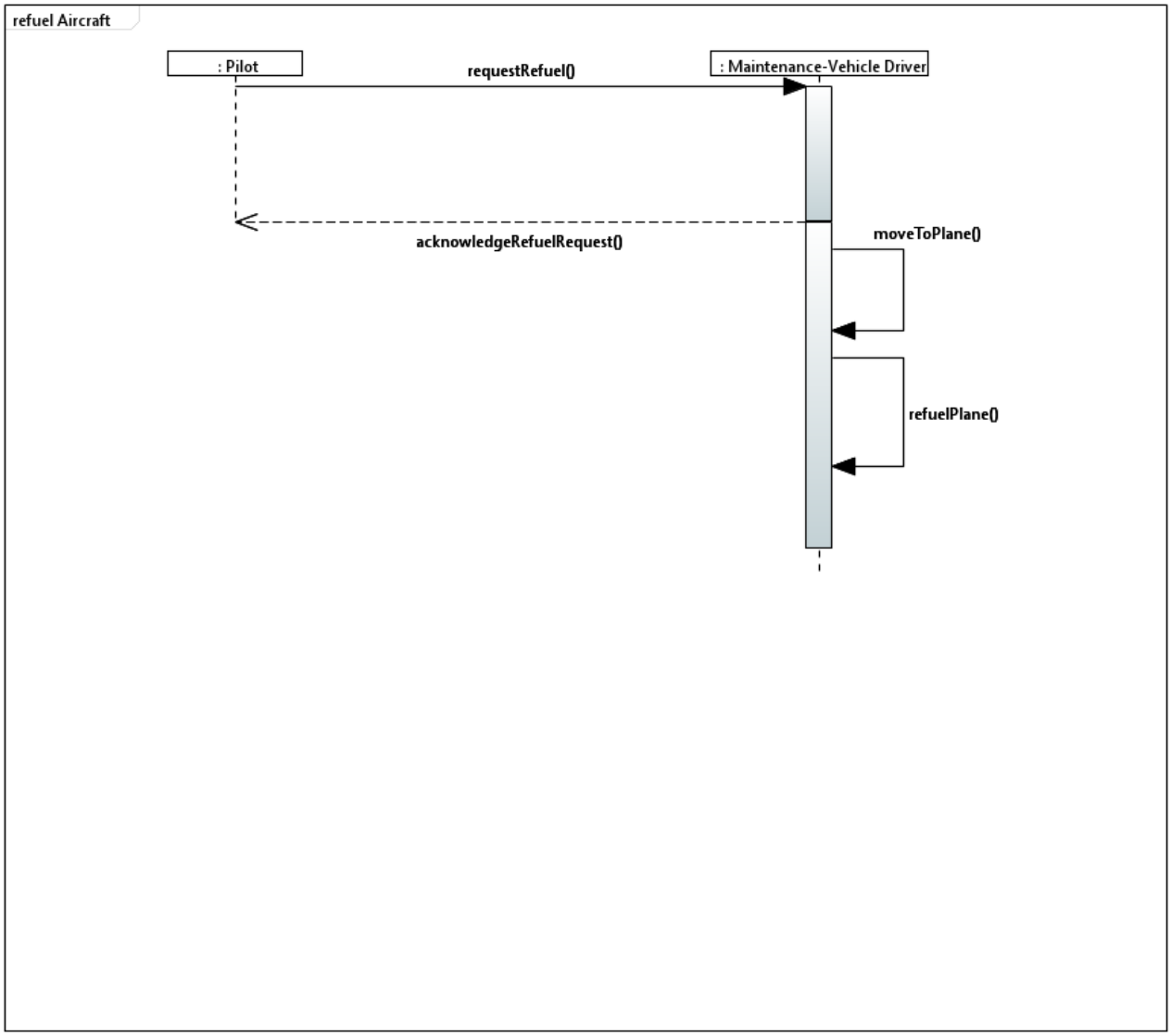
To support the described use-cases, this class diagram was created. Most classes represent an actor from the use-case diagram. Each class has it's set of functions which are defined by the various sequence diagram which are described later. The Interface "Destination" is imported from Subsystem2 to guarantee that both subsystems have the same understanding what a "Destination" is. The personnel classes inherit from the base class which represents an employee of the airport.

Note:
The sequence diagrams had to be remade after Papyrus locked them and they could not be edited after the feedback was received.

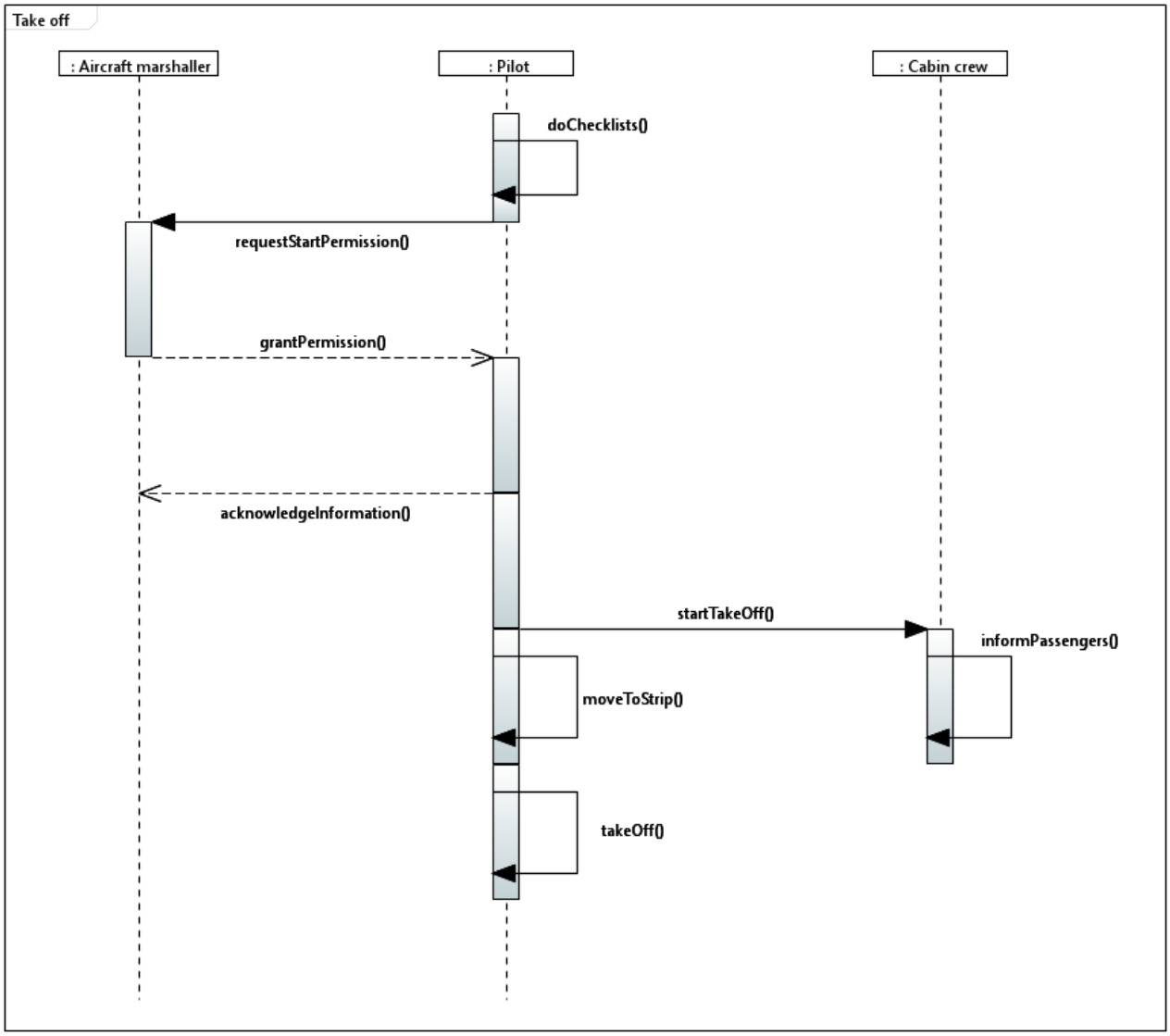
Land aircraft
Step 1: Pilot requests landing clearance from the aircraft marshaller
Step 2: Pilot notifies cabin crew that the aircraft will land soon
Step 3: Cabin crew informs the passengers that they should put on their seatbelts
Step 4: Aircraft marshaller gives clearance and provides the landing strip
Step 5: Pilot performs landing
Step 6: Aircraft marshaller provides a gate number and the parking spot after the passengers left.
Step 7: Aircraft marshaller signals instructions to the pilot to ease parking.
Step 8: Pilot acknowledges the information provided
Step 9: Pilot informs Cabin crew that gate docking is completed
Step 10: Cabin crew departs the passengers row-wise



Refuel aircraft
Step 1: Pilot requests fuel truck from landside management and provides location information
Step 2: Fuel truck driver gets information
Step 3: Fuel truck driver reports to pilot that the truck will refuel the aircraft
Step 4: Fuel truck moves to aircraft
Step 5: Fuel truck refills aircraft

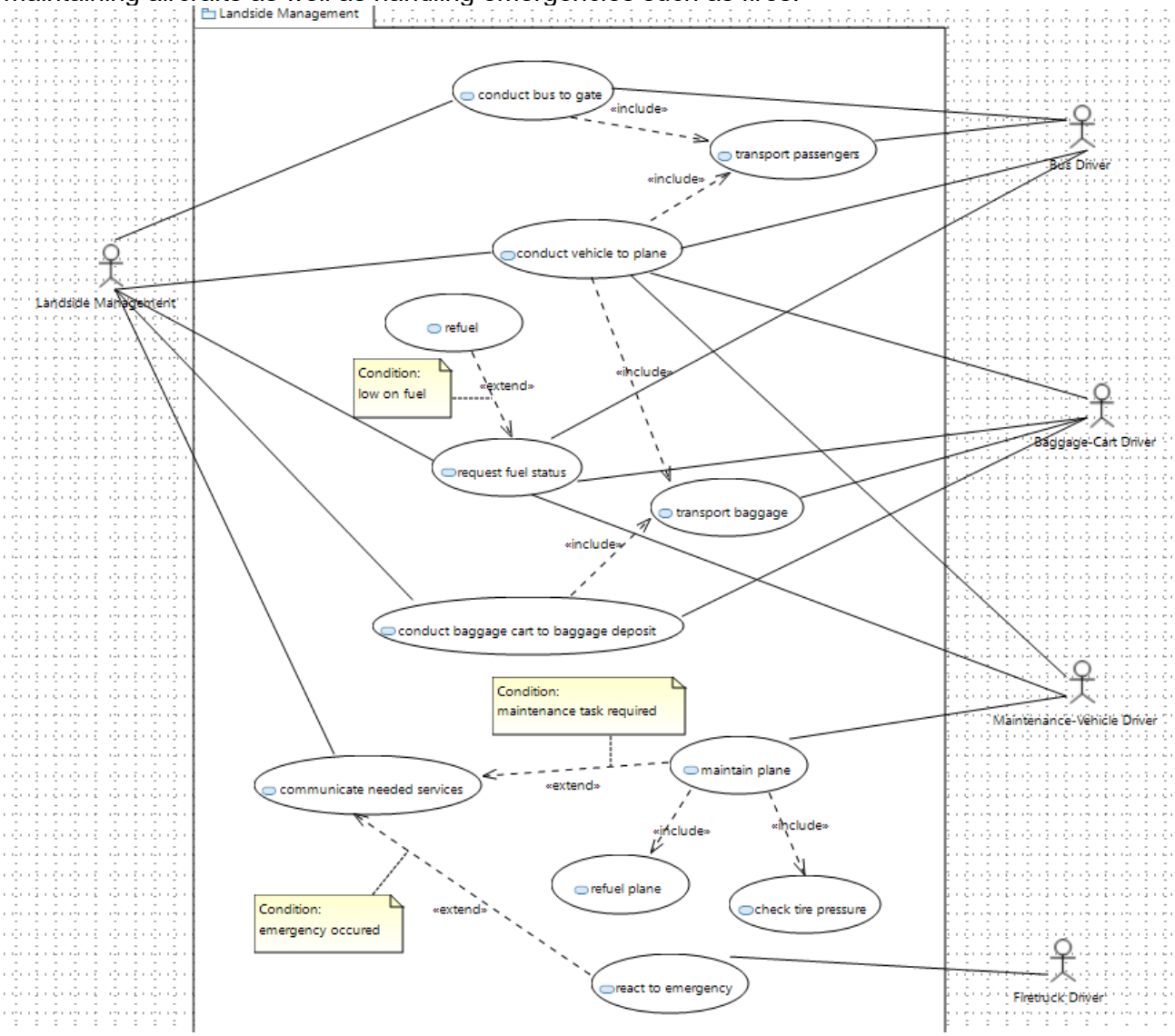


Take off
Step 1: Pilot goes through all checklists
Step 2: Pilot requests start permission from aircraft marshaller
Step 3: Aircraft marshaller gives permission together with starting strip
Step 4: Pilot acknowledges the information provided
Step 5: Pilot informs cabin crew that the aircraft has started the takeoff procedure
Step 6: Cabin crew informs passengers that they should put on their seatbelts
Step 7: Pilot moves the aircraft to the designated starting strip
Step 8: Pilot performs take-off



2.2 Sub-system 2/Sub-system name - [Stefan **Haslhofer**]

The landside management coordinates land vehicles picking up passengers and baggage, maintaining aircrafts as well as handling emergencies such as fires.



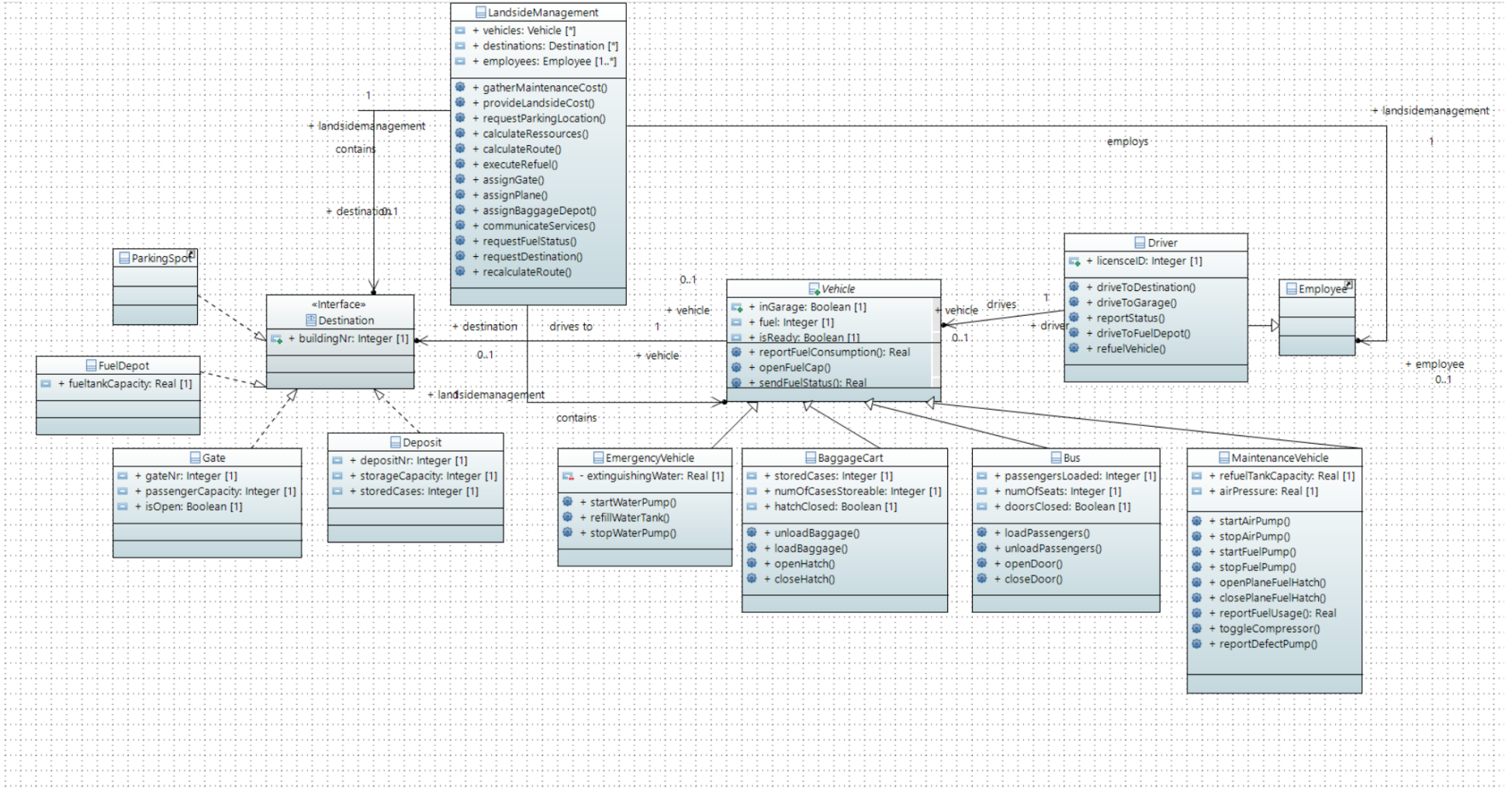
The landside management is responsible for the assignments of land vehicles to planes or gates. It provides the necessary information such that a land vehicle driver is able to head for the correct destination at the correct time.

The bus driver brings persons either from a plane to a gate or vice versa. He receives his destinations from the landside management.

The baggage-cart driver delivers all the baggage from a plane to a baggage deposit.

The maintenance vehicle driver checks tire pressure off plane tires as well as refueling the plane itself. The amount of fuel needed is communicated by the management.

The firetruck driver responds to emergency calls in case of a fire or other malfunctions that need specialists on the runway.

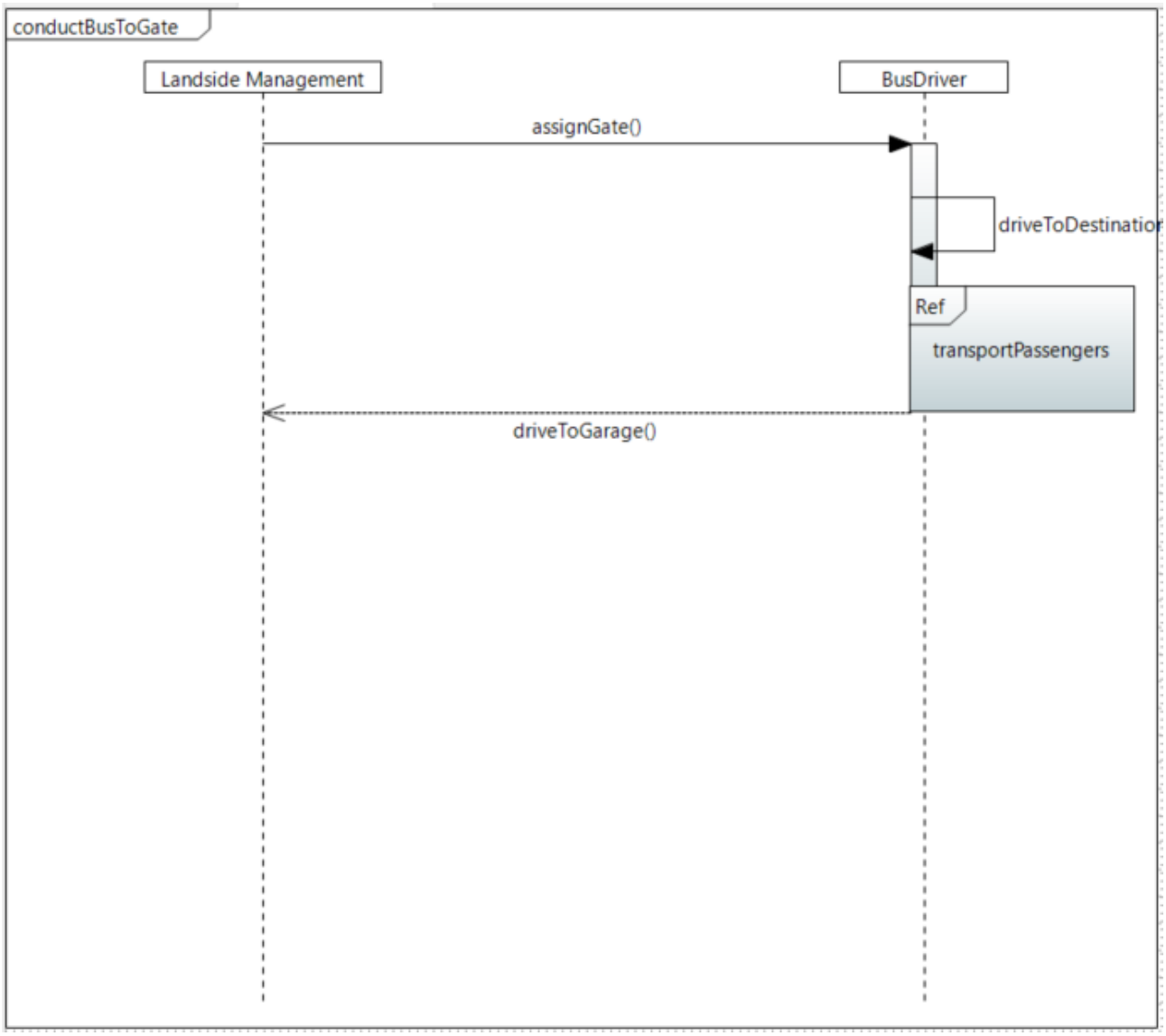

LandsideManagement
+ vehicles: Vehicle [*]
+ destinations: Destination [*]
+ employees: Employee [1..*]
+ gatherMaintenanceCost()
+ provideLandsideCost()
+ requestParkingLocation()
+ calculateRessources()
+ calculateRoute()
+ executeRefuel()
+ assignGate()
+ assignPlane()
+ assignBaggageDepot()
+ communicateServices()
+ requestFuelStatus()
+ requestDestination()
+ recalculateRoute()
1
+ landsidemanagement
contains
+ destination
0..1
+ landsidemanagement
employs
1
+ employee
0..1
ParkingSpot
«Interface»
Destination
+ buildingNr: Integer [1]
+ destination
0..1
drives to
1
+ vehicle
0..1
+ vehicle
+ landsidemanagement
contains
Vehicle
+ inGarage: Boolean [1]
+ fuel: Integer [1]
+ isReady: Boolean [1]
+ reportFuelConsumption(): Real
+ openFuelCap()
+ sendFuelStatus(): Real
+ vehicle
drives
1
+ driver
0..1
Driver
+ licensceID: Integer [1]
+ driveToDestination()
+ driveToGarage()
+ reportStatus()
+ driveToFuelDepot()
+ refuelVehicle()
Employee
FuelDepot
+ fueltankCapacity: Real [1]
Gate
+ gateNr: Integer [1]
+ passengerCapacity: Integer [1]
+ isOpen: Boolean [1]
Deposit
+ depositNr: Integer [1]
+ storageCapacity: Integer [1]
+ storedCases: Integer [1]
EmergencyVehicle
- extinguishingWater: Real [1]
+ startWaterPump()
+ refillWaterTank()
+ stopWaterPump()
BaggageCart
+ storedCases: Integer [1]
+ numOfCasesStoreable: Integer [1]
+ hatchClosed: Boolean [1]
+ unloadBaggage()
+ loadBaggage()
+ openHatch()
+ closeHatch()
Bus
+ passengersLoaded: Integer [1]
+ numOfSeats: Integer [1]
+ doorsClosed: Boolean [1]
+ loadPassengers()
+ unloadPassengers()
+ openDoor()
+ closeDoor()
MaintenanceVehicle
+ refuelTankCapacity: Real [1]
+ airPressure: Real [1]
+ startAirPump()
+ stopAirPump()
+ startFuelPump()
+ stopFuelPump()
+ openPlaneFuelHatch()
+ closePlaneFuelHatch()
+ reportFuelUsage(): Real
+ toggleCompressor()
+ reportDefectPump()

In order to match the roles, there are four types of vehicles, each of them resembling one type of driver. Although there is only one driver, he is distinguishable by the vehicle he commanders. Every driver is also an employee and each car/bus/cart is a vehicle. Employee is imported from system-level. A vehicle retrieves a destination from the management which it has to be driven to by the driver to do certain tasks. A destination can be a fuel depot, a (plane) parking spot, the vehicle´s garage or a gate.

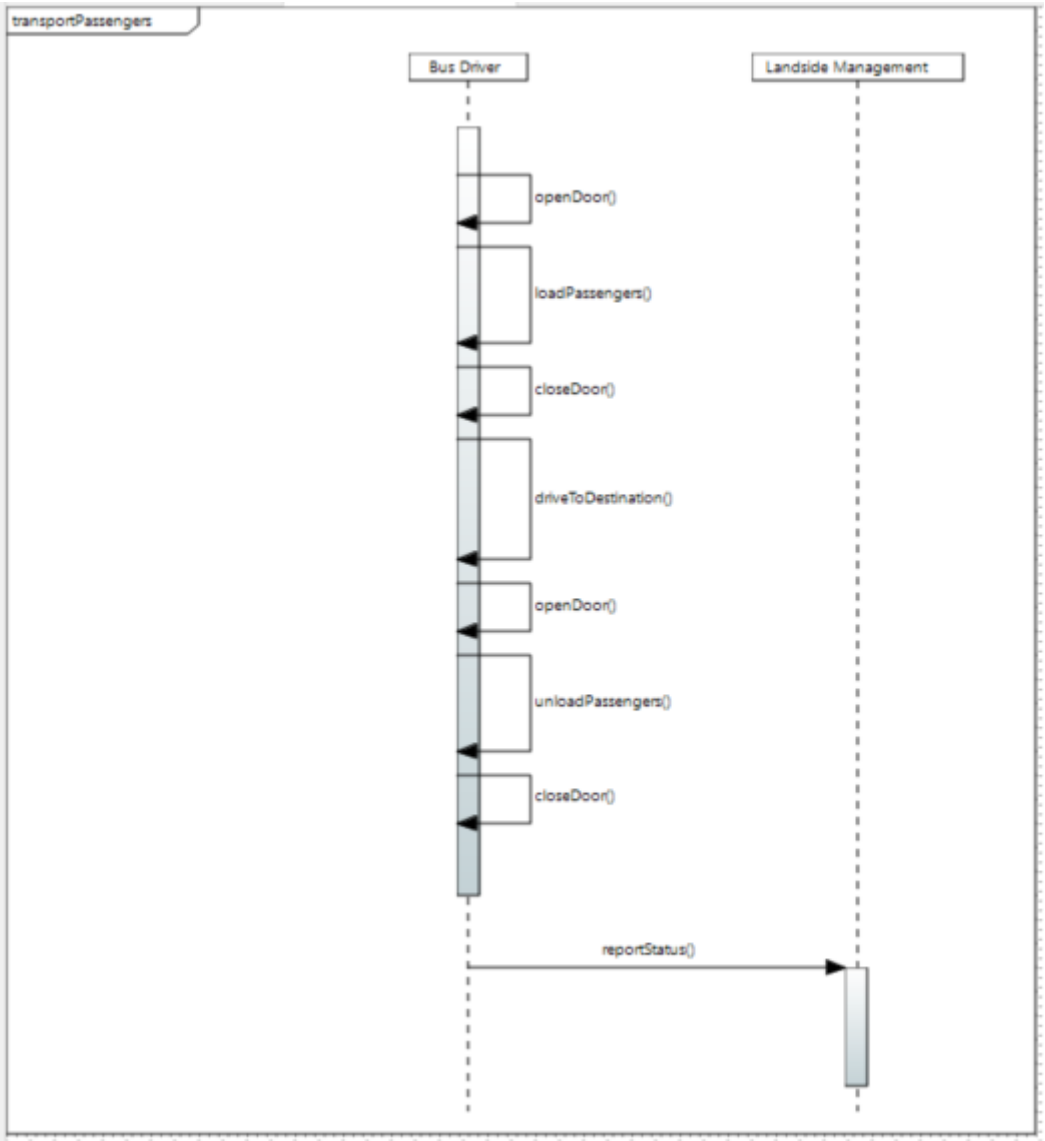
Sequence diagrams:

Conduct bus to gate

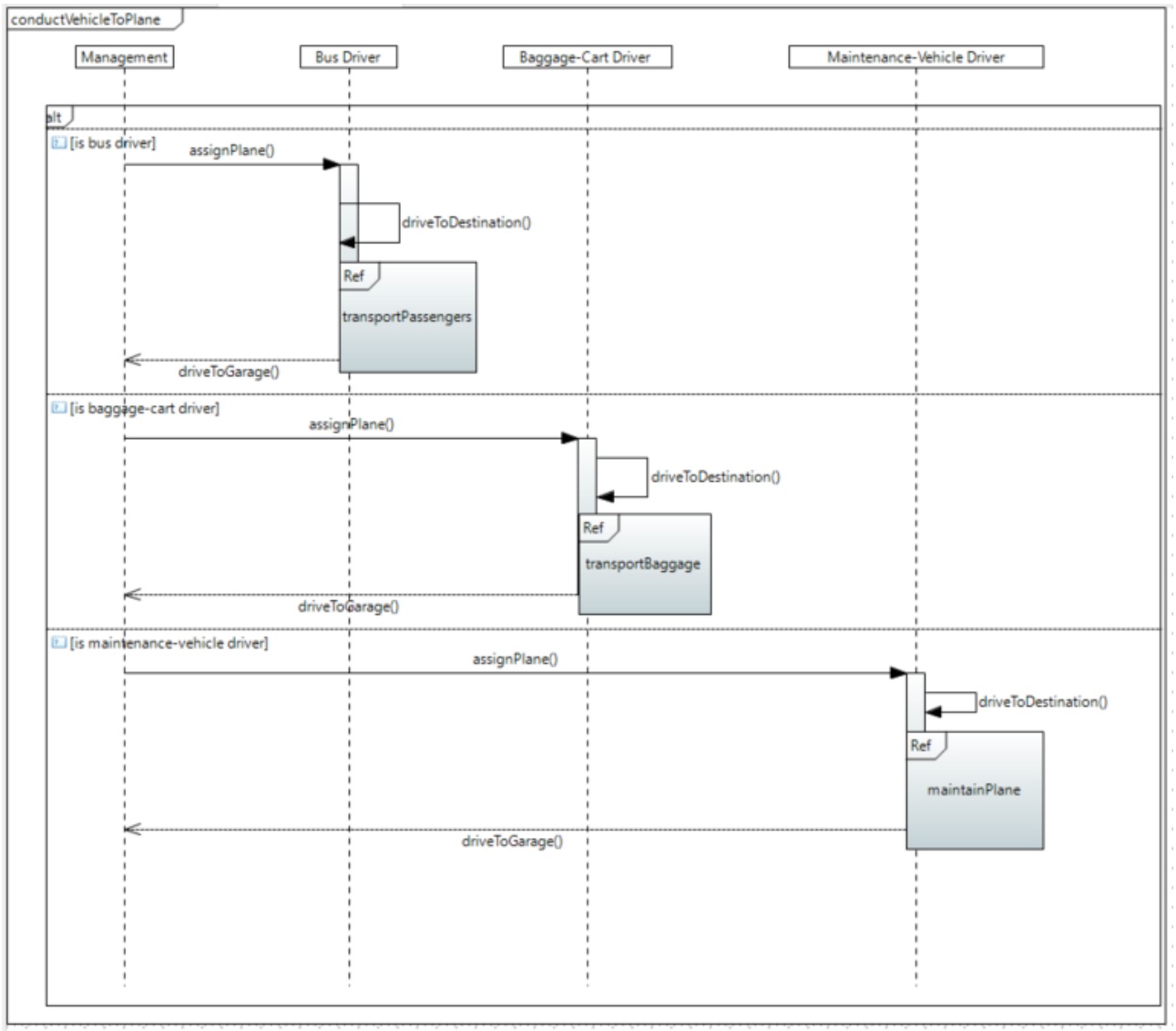
Step 1: management assigns a gate to a bus driver
Step 2: bus driver drives to the gate
Step 3: bus driver arrives at gate
Step 4: the bus driver **transports the passengers** to a plane
Step 5: bus driver drives back to garage
Step 6: bus driver awaits new orders



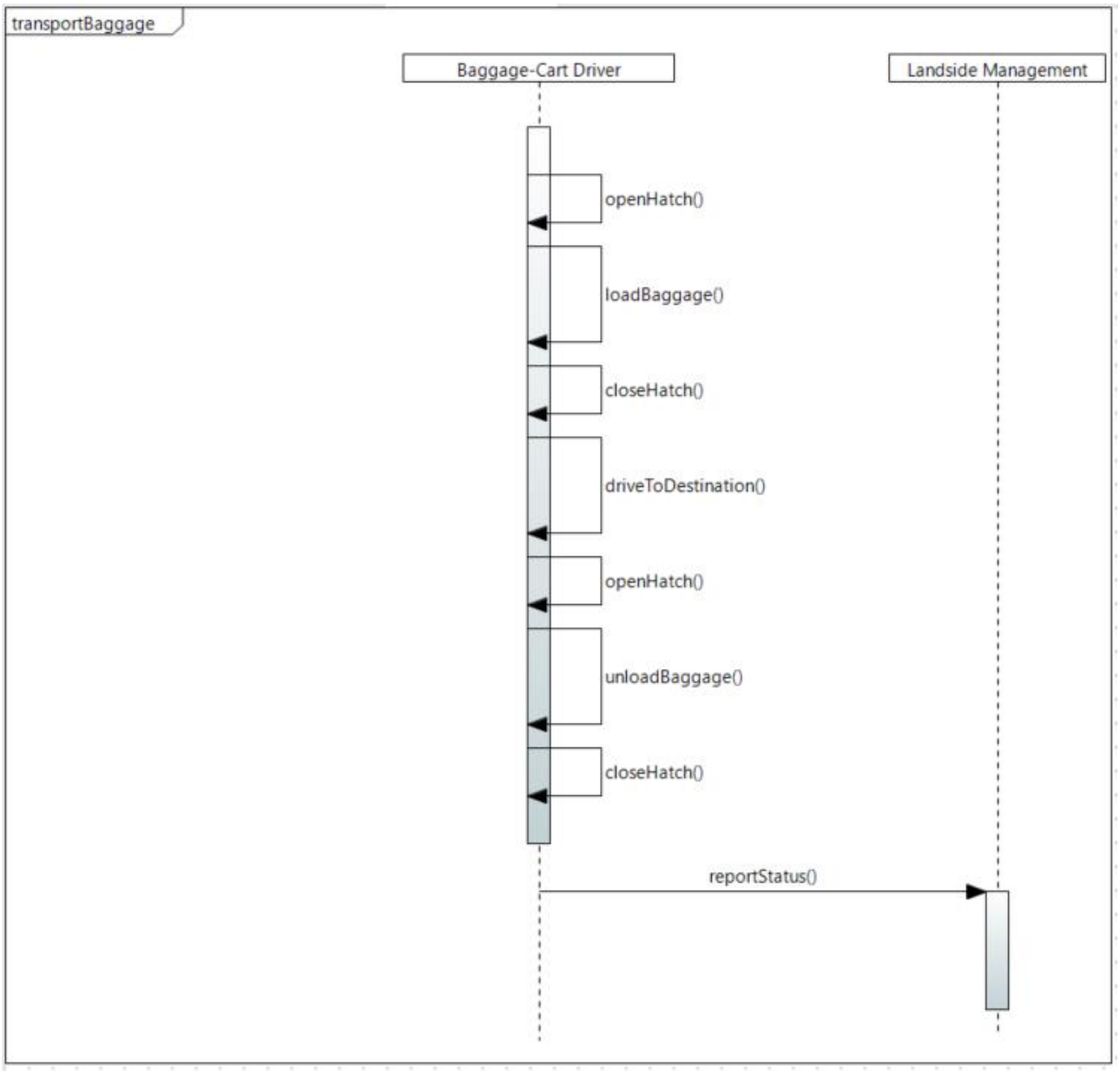
Transport passengers
Step 1: bus driver opens bus doors
Step 2: bus driver waits until bus is full or no more people are left
Step 3: bus driver closes doors
Step 4: bus driver drives to destination
Step 5: bus driver arrives at destination
Step 6: bus driver opens doors
Step 7: bus driver waits until bus is empty
Step 8: bus driver reports status back to management
Step 9: bus driver closes doors



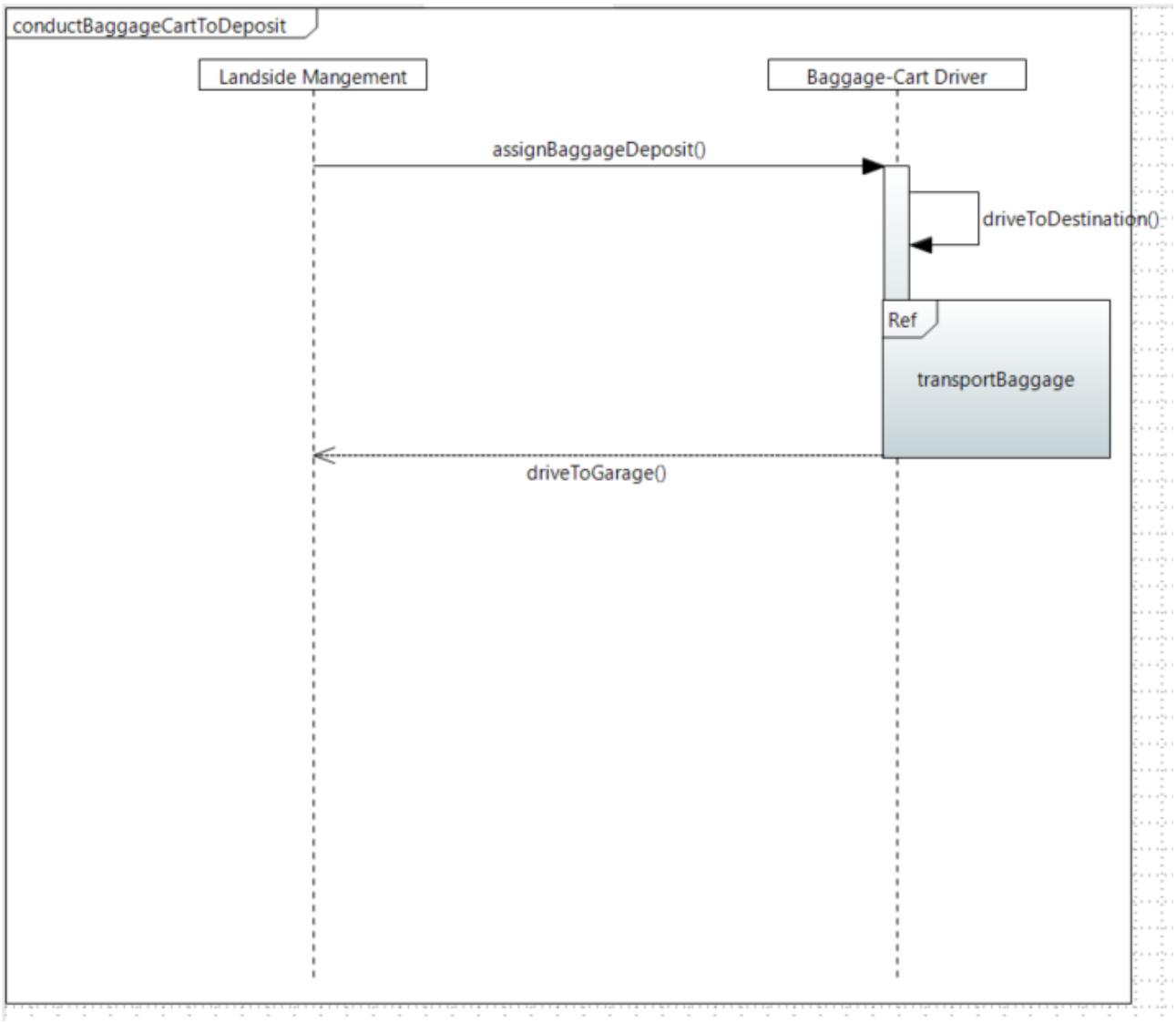
Conduct vehicle to plane
Step 1: management assigns a plane to a driver
Step 2: driver drives to the plane
Step 3: driver arrives at plane
Step 4: the bus driver **transports the passengers** to the gate
Step 5: the baggage vehicle driver **transports baggage** to a baggage deposit
Step 5: driver drives back to garage
Step 6: driver awaits new orders



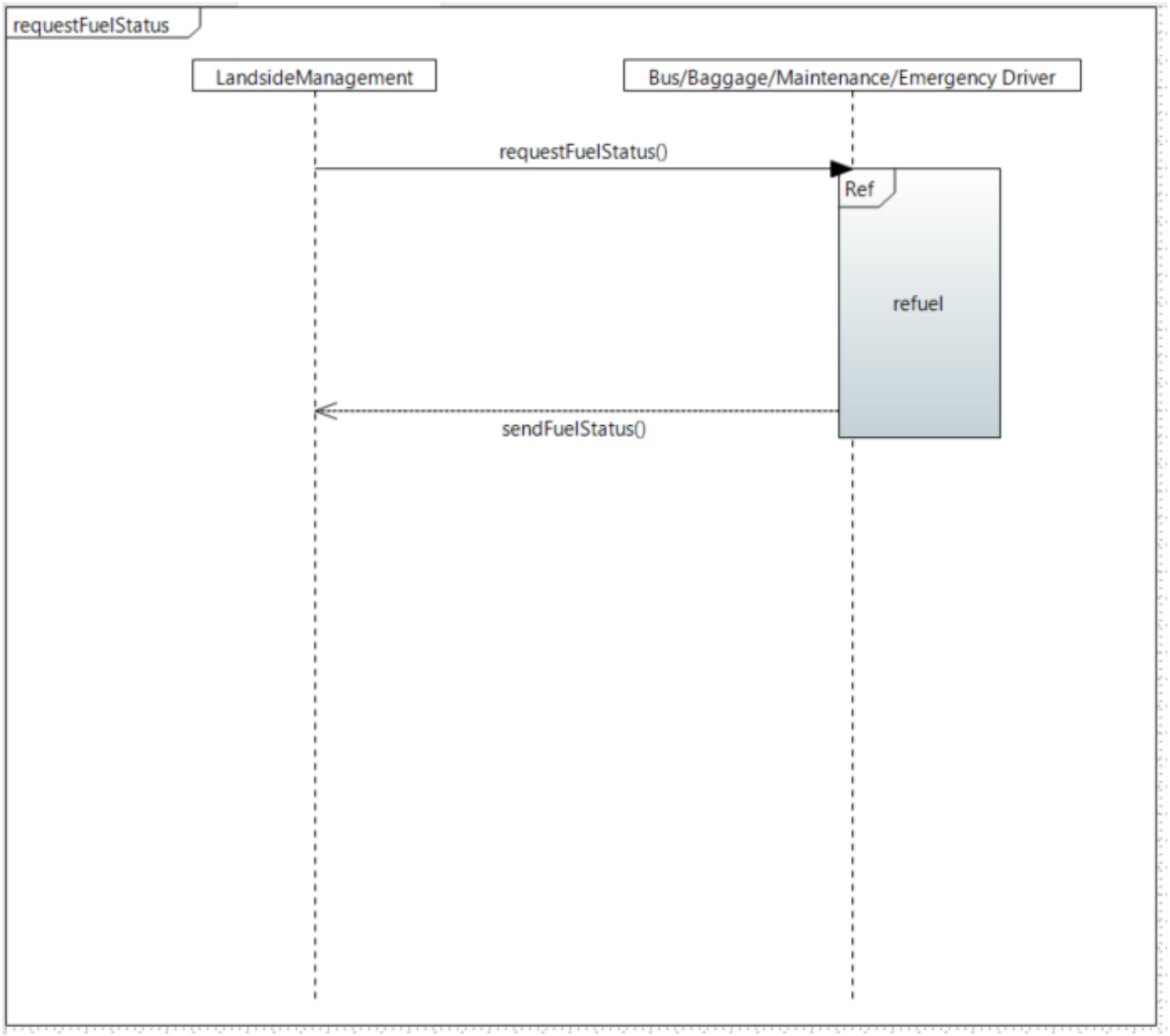
Transport baggage
Step 1: baggage-cart driver opens baggage hatch
Step 2: baggage-cart driver waits until baggage is fully deposited inside the vehicle
Step 3: baggage-cart driver closes and locks baggage hatch
Step 4: baggage-cart driver drives to destination
Step 5: baggage-cart driver arrives at destination
Step 6: baggage-cart driver opens hatch
Step 7: baggage-cart driver waits until baggage is unloaded
Step 8: baggage-cart driver reports status back to management
Step 9: baggage-cart driver closes hatch



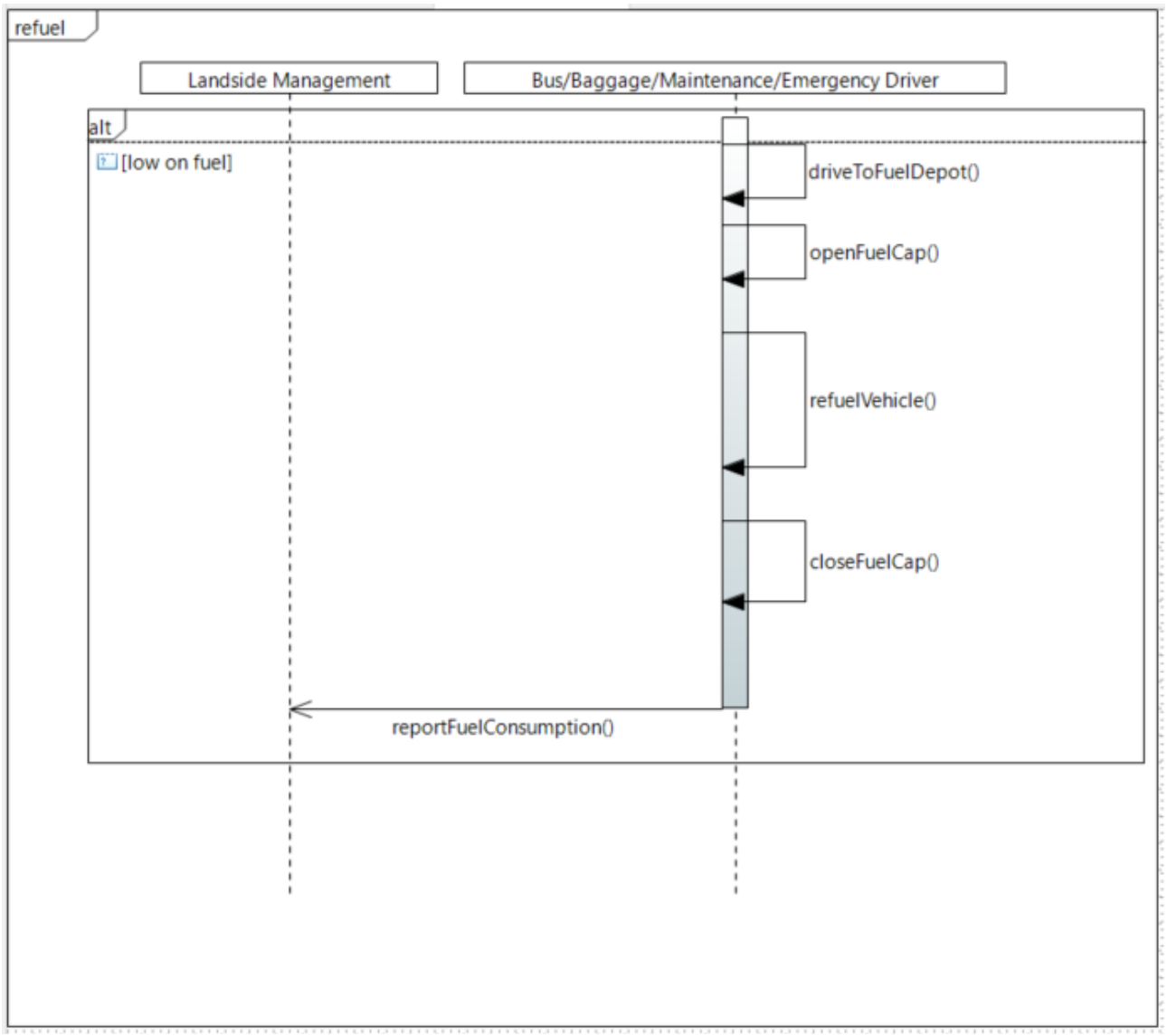
Conduct baggage cart to baggage deposit
Step 1: management assigns a baggage deposit to a baggage-cart driver
Step 2: baggage-cart driver drives to the baggage deposit
Step 3: baggage-cart driver arrives at baggage deposit
Step 5: the baggage-cart driver **transports the baggage**
Step 5: baggage-cart driver drives back to garage
Step 6: baggage-cart driver awaits new orders



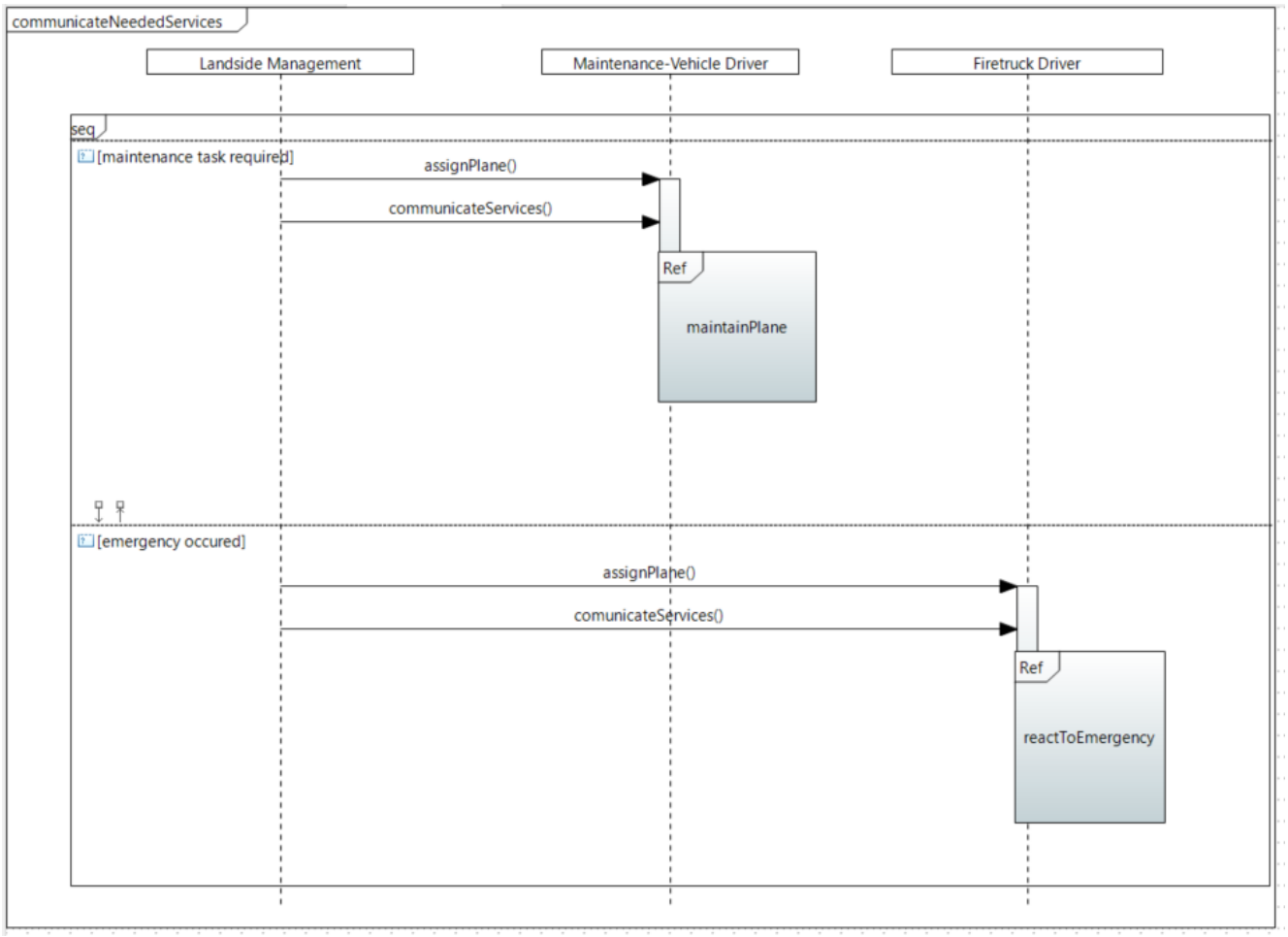
Request refuel status
Step 1: the management asks a driver for his fuel status
Step 2: the management waits until the driver returns from his assignment
Step 3: if the vehicle is low on fuel the driver needs to **refuel**



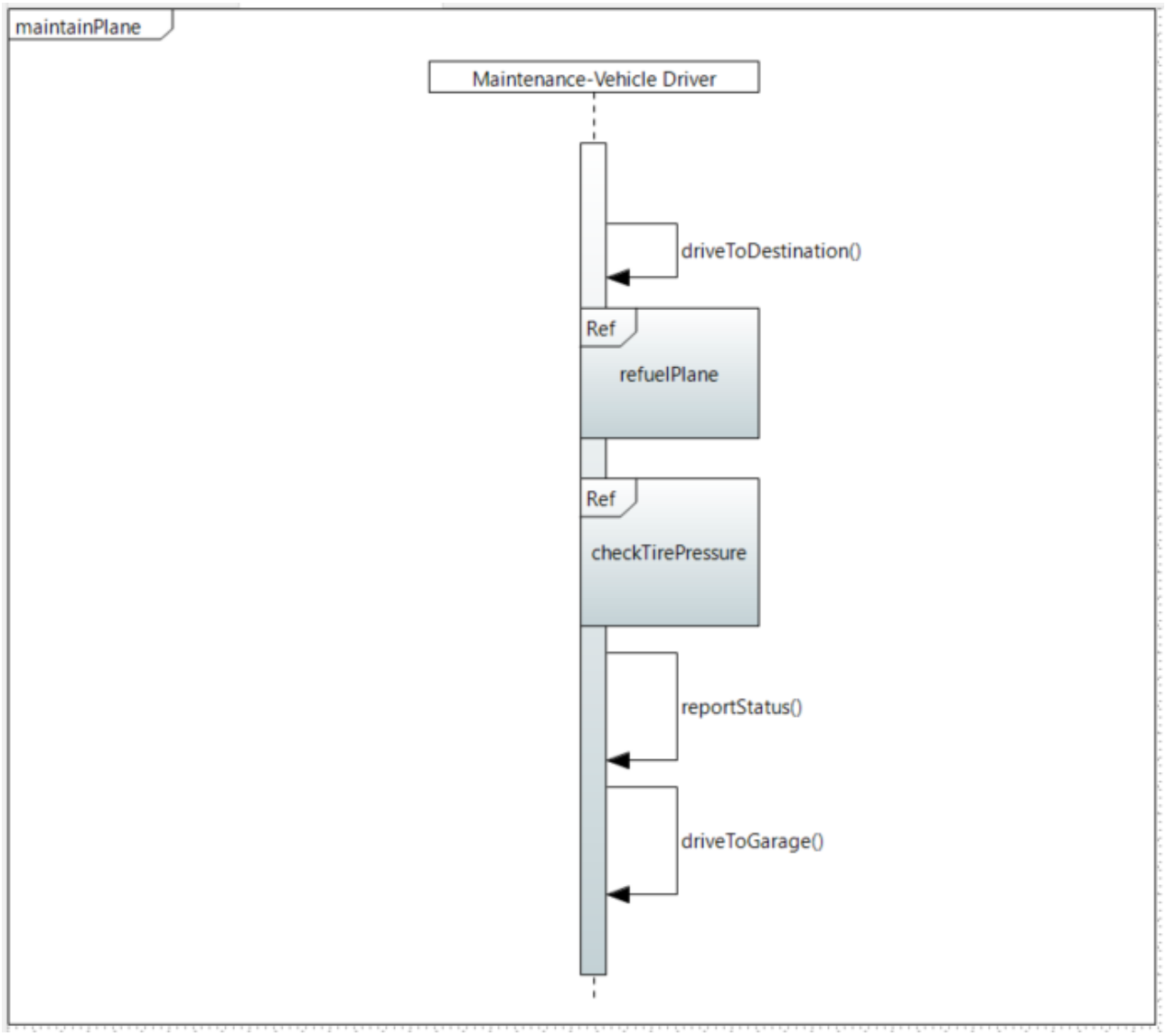
Refuel
Step 1: the management assigns the driver to the nearest fuel depot
Step 2: driver drives to the depot
Step 3: driver arrives at the depot
Step 4: driver opens the fuel cap
Step 5: driver puts the gasoline hose into the open tank
Step 6: driver starts pumping gasoline
Step 7: driver waits until the tank is full
Step 8: driver stops pumping gasoline
Step 9: driver puts gasoline hose back
Step 10: driver closes the fuel cap
Step 11: driver reports consumed liters back to the management



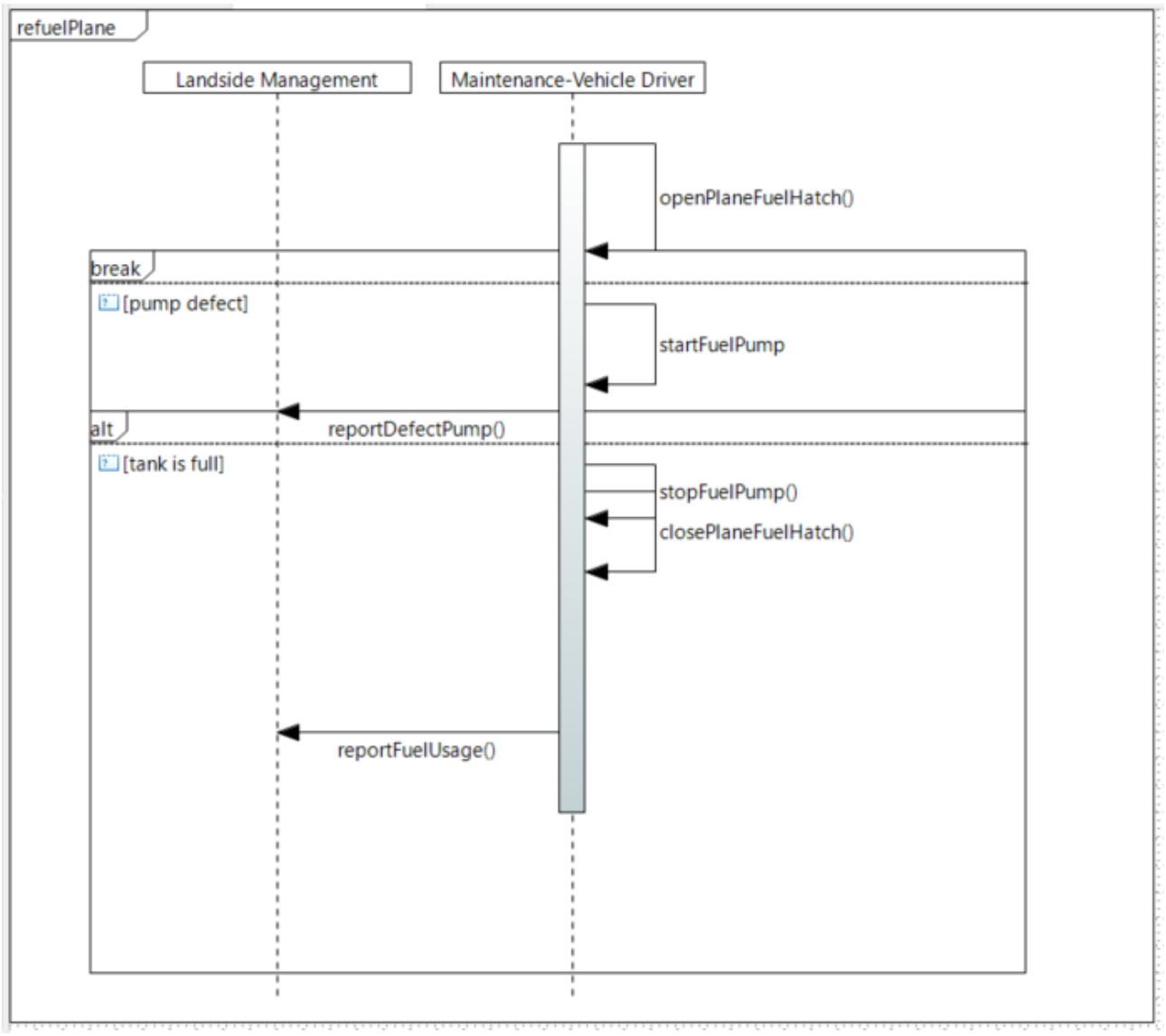
Communicate needed services
Step 1: management assigns vehicles/driver to plane
Step 2: management tells the driver what to do



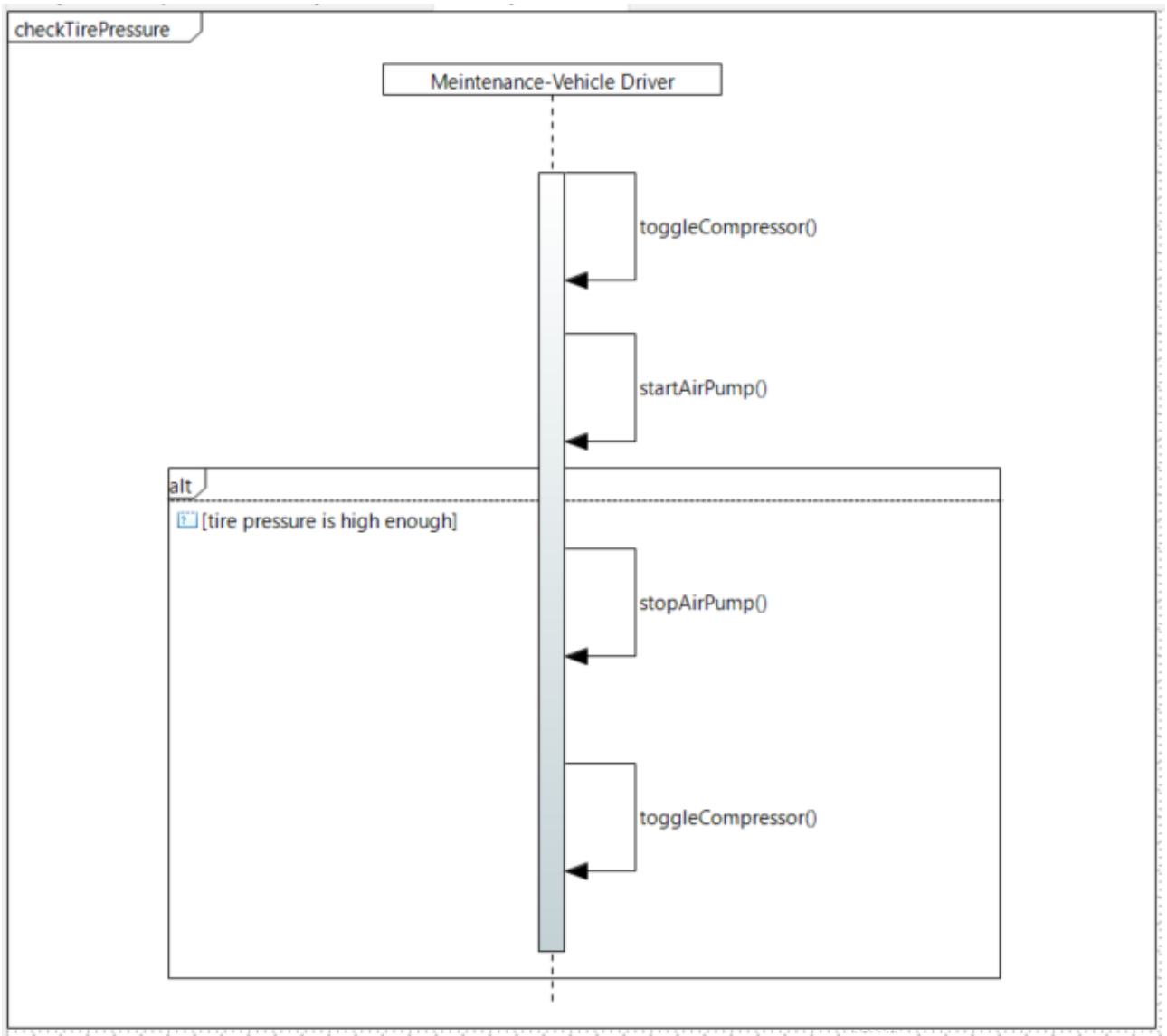
Maintain plane
Step 1: maintenance vehicle driver drives to plane
Step 2: maintenance vehicle driver arrives
Step 3: maintenance vehicle driver refuels plane
Step 4: maintenance vehicle driver check tire pressure
Step 5: maintenance vehicle driver reports status back to management
Step 6: maintenance vehicle driver drives back to garage
Step 7: maintenance vehicle driver awaits new orders



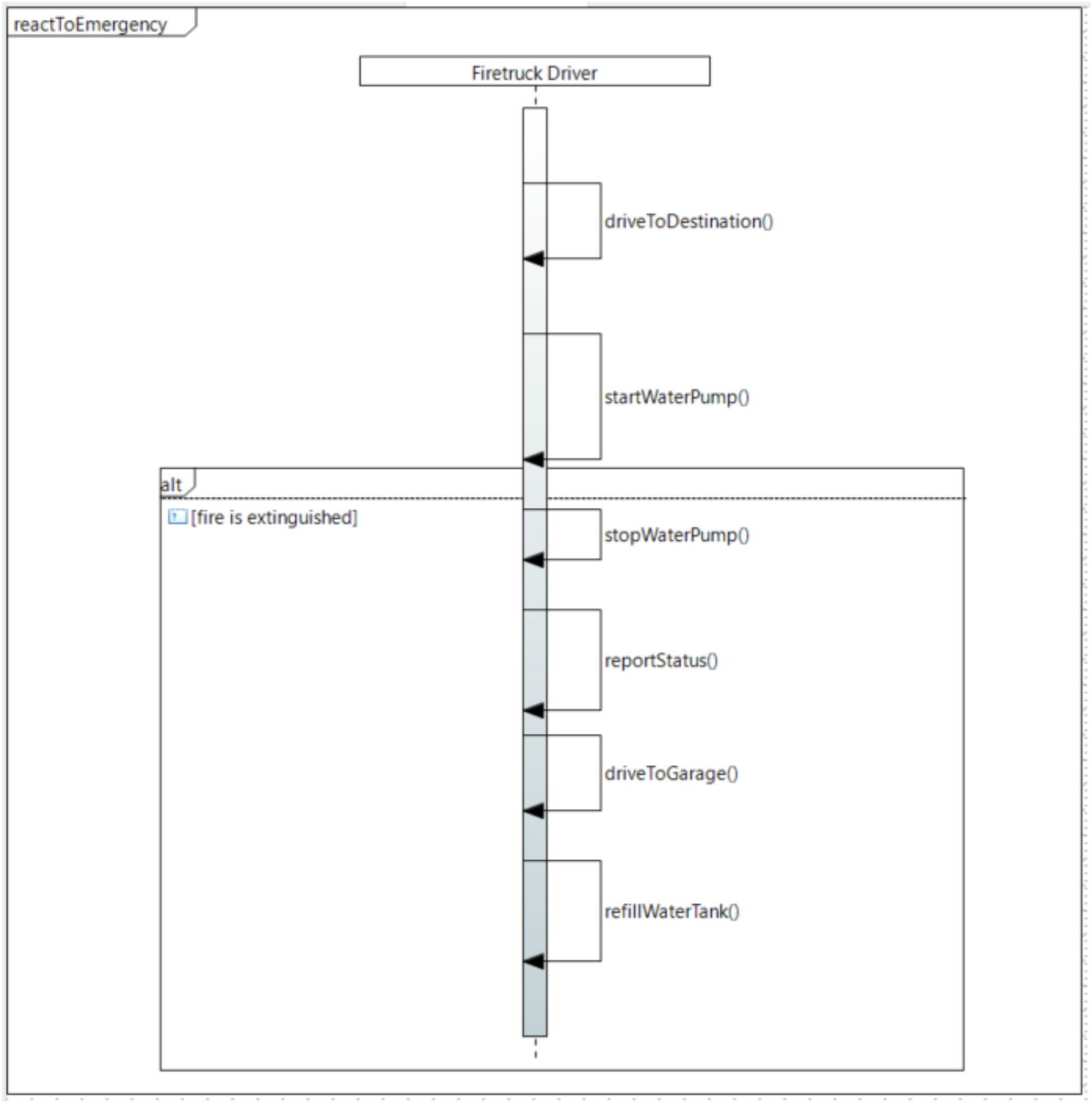
Refuel plane
Step 1: maintenance vehicle driver opens fuel hatch of plane
Step 2: maintenance vehicle driver attaches gasoline hose to hatch
Step 3: maintenance vehicle driver starts pumping kerosine
Step 4: maintenance vehicle driver stops pumping
Step 5: maintenance vehicle driver disconnects gasoline hose
Step 6: maintenance vehicle driver closes hatch
Step 7: maintenance vehicle driver reports amount of gasoline used to management
Step 8: maintenance vehicle driver double checks if planes fuel hatch is closed



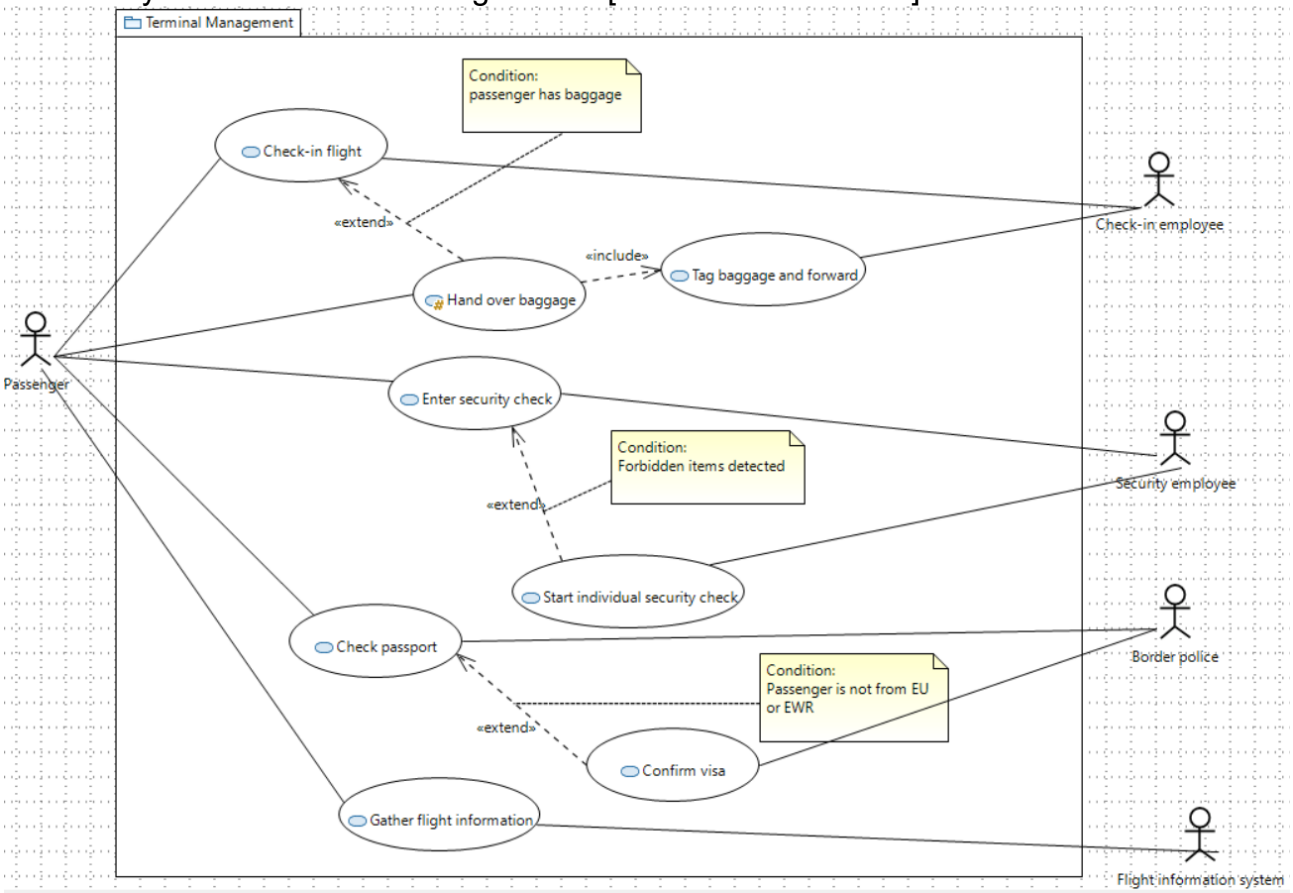
Check tire pressure
Step 1: maintenance vehicle driver starts compressor to fill the land vehicles high pressure air tank
Step 2: maintenance vehicle driver retrieves tire pressure gauge from vehicle
Step 3: maintenance vehicle driver connects tire pressure gauge to the vehicles high pressure air tank
Step 4: maintenance vehicle driver connects tire pressure gauge to the plane´s tire
Step 5: maintenance vehicle driver uses the tire pressure gauge to measures tire pressure
Step 6: maintenance vehicle driver pumps air into the tire until pressure is high enough
Step 7: maintenance vehicle driver stops the pump
Step 8: maintenance vehicle driver stops the compressor
Step 9: maintenance vehicle driver disconnects the tire pressure gauge from the tire
Step 10: maintenance vehicle driver puts the tire pressure gauge back into the vehicle



React to emergency
Step 1: firetruck driver drives to plane
Step 2: firetruck driver arrives
Step 3: firetruck driver starts water pump
Step 4: firetruck driver extinguishes fire
Step 5: firetruck driver stays until all passengers are brought to safety
Step 6: firetruck driver stops water pump
Step 7: firetruck driver reports status back to management
Step 8: firetruck driver drives back to garage
Step 9: firetruck driver refills water tank
Step 10: firetruck driver awaits new orders



2.3 Sub-system 3/Terminal management - [Simon **Primetzhofer**]



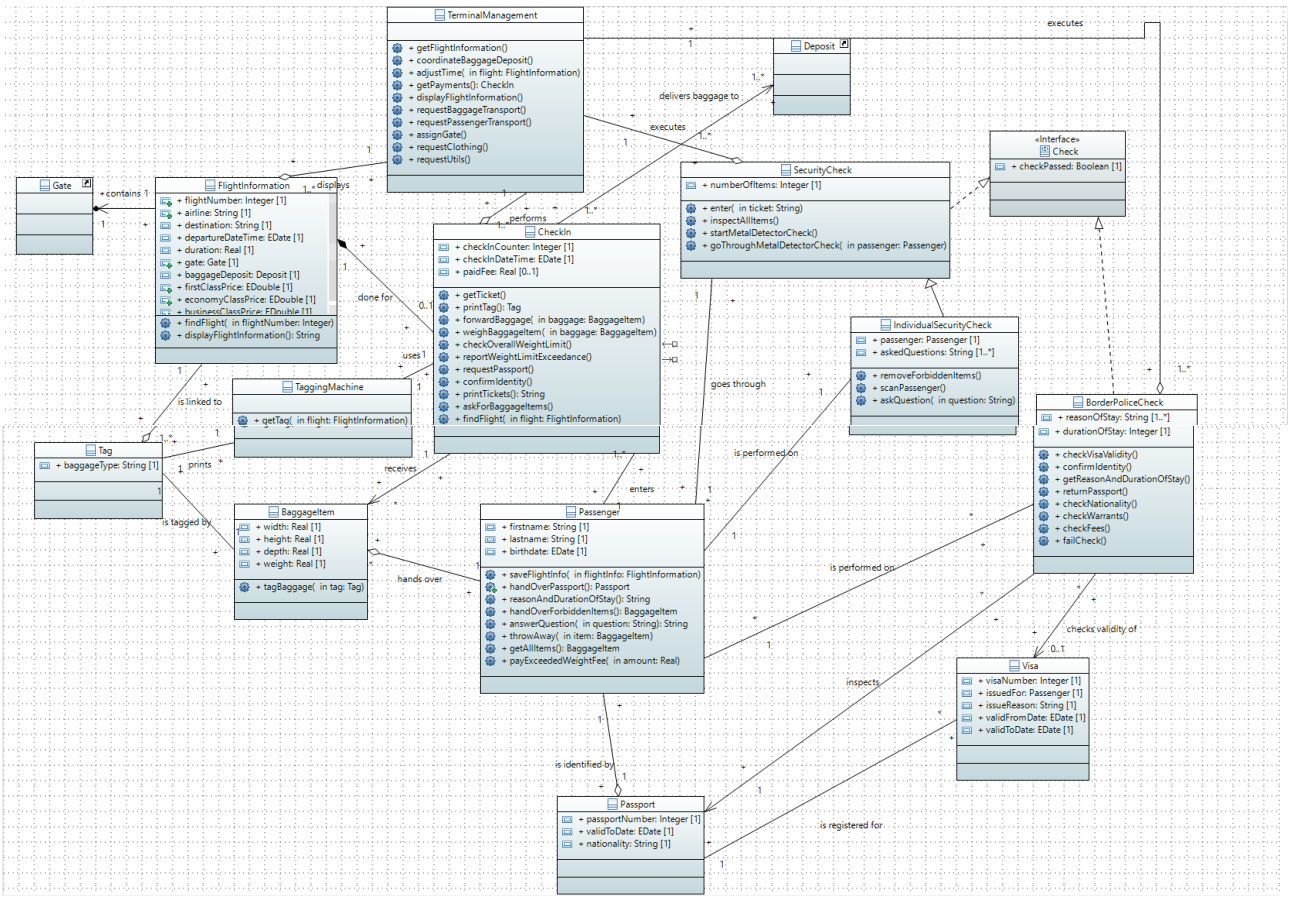
A passenger is a natural person who enters the airport in order to travel to another destination. Passengers may have some baggage with them – depending on the size, it must be handed over at the check-in. Furthermore, a passenger must enter the security check in any case and when arriving, already have a valid visa to enter the respective country.

Check-in employees are sitting at the check-in counter and have two tasks: Checking in a passenger for the booked flight and if a passenger has baggage, they must tag and forward it to the landside management.

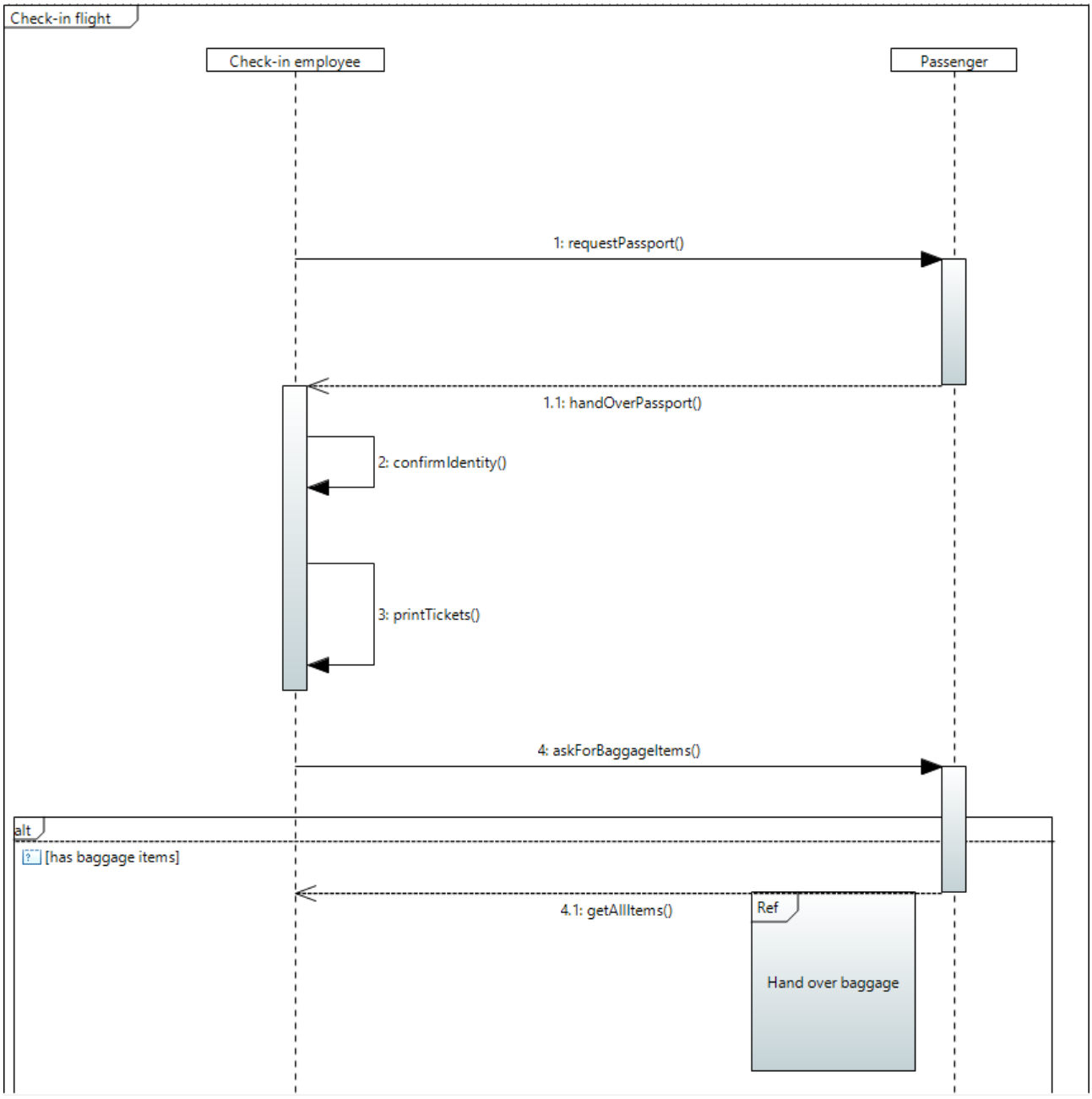
Security employees are responsible for inspecting the hand baggage and the passengers themselves. They remove forbidden items and are also allowed to examine a passenger in detail. They have the permission to deny entrance to the gate area when security concerns are present.

Border police officers are monitoring arriving passengers and confirm their admission. They must have knowledge about which nationalities need a visa and which can enter without further checks.

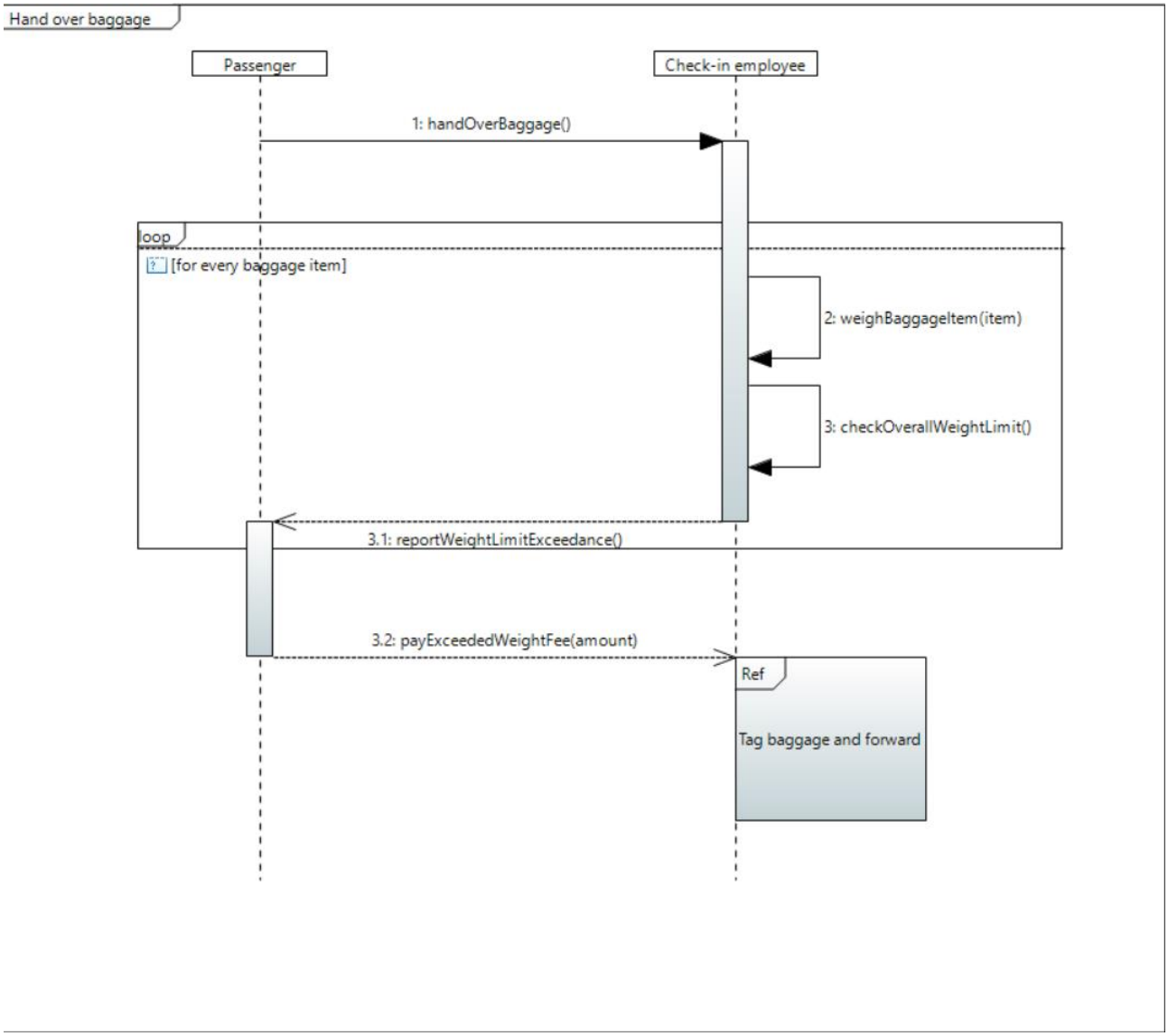
The flight information system is a digital system which allows passenger to gather all necessary flight information at any time. There are big screens all over the airport and also small screens directly at the gates which provide more detailed information.


TerminalManagement
+ getFlightInformation()
+ coordinateBaggageDeposit()
+ adjustTime(in flight: FlightInformation)
+ getPayments(): CheckIn
+ displayFlightInformation()
+ requestBaggageTransport()
+ requestPassengerTransport()
+ assignGate()
+ requestClothing()
+ requestUtils()
Deposit
executes
delivers baggage to
executes
«Interface»
Check
+ checkPassed: Boolean [1]
Gate
+ contains 1
FlightInformation
displays
+ flightNumber: Integer [1]
+ airline: String [1]
+ destination: String [1]
+ departureDateTime: EDate [1]
+ duration: Real [1]
+ gate: Gate [1]
+ baggageDeposit: Deposit [1]
+ firstClassPrice: EDouble [1]
+ economyClassPrice: EDouble [1]
+ businessClassPrice: EDouble [1]
+ findFlight(in flightNumber: Integer)
+ displayFlightInformation(): String
SecurityCheck
+ numberOfItems: Integer [1]
+ enter(in ticket: String)
+ inspectAllItems()
+ startMetalDetectorCheck()
+ goThroughMetalDetectorCheck(in passenger: Passenger)
performs
CheckIn
+ checkInCounter: Integer [1]
+ checkInDateTime: EDate [1]
+ paidFee: Real [0..1]
+ getTicket()
+ printTag(): Tag
+ forwardBaggage(in baggage: BaggageItem)
+ weighBaggageItem(in baggage: BaggageItem)
+ checkOverallWeightLimit()
+ reportWeightLimitExceedance()
+ requestPassport()
+ confirmIdentity()
+ printTickets(): String
+ askForBaggageItems()
+ findFlight(in flight: FlightInformation)
done for
uses 1
IndividualSecurityCheck
+ passenger: Passenger [1]
+ askedQuestions: String [1..*]
+ removeForbiddenItems()
+ scanPassenger()
+ askQuestion(in question: String)
goes through
BorderPoliceCheck
+ reasonOfStay: String [1..*]
+ durationOfStay: Integer [1]
+ checkVisaValidity()
+ confirmIdentity()
+ getReasonAndDurationOfStay()
+ returnPassport()
+ checkNationality()
+ checkWarrants()
+ checkFees()
+ failCheck()
is linked to
TaggingMachine
+ getTag(in flight: FlightInformation)
Tag
+ baggageType: String [1]
prints
is tagged by
receives
BaggageItem
+ width: Real [1]
+ height: Real [1]
+ depth: Real [1]
+ weight: Real [1]
+ tagBaggage(in tag: Tag)
hands over
Passenger
+ firstname: String [1]
+ lastname: String [1]
+ birthdate: EDate [1]
+ saveFlightInfo(in flightInfo: FlightInformation)
+ handOverPassport(): Passport
+ reasonAndDurationOfStay(): String
+ handOverForbiddenItems(): BaggageItem
+ answerQuestion(in question: String): String
+ throwAway(in item: BaggageItem)
+ getAllItems(): BaggageItem
+ payExceededWeightFee(in amount: Real)
enters
is performed on
is performed on
checks validity of
Visa
+ visaNumber: Integer [1]
+ issuedFor: Passenger [1]
+ issueReason: String [1]
+ validFromDate: EDate [1]
+ validToDate: EDate [1]
inspects
is identified by
Passport
+ passportNumber: Integer [1]
+ validToDate: EDate [1]
+ nationality: String [1]
is registered for

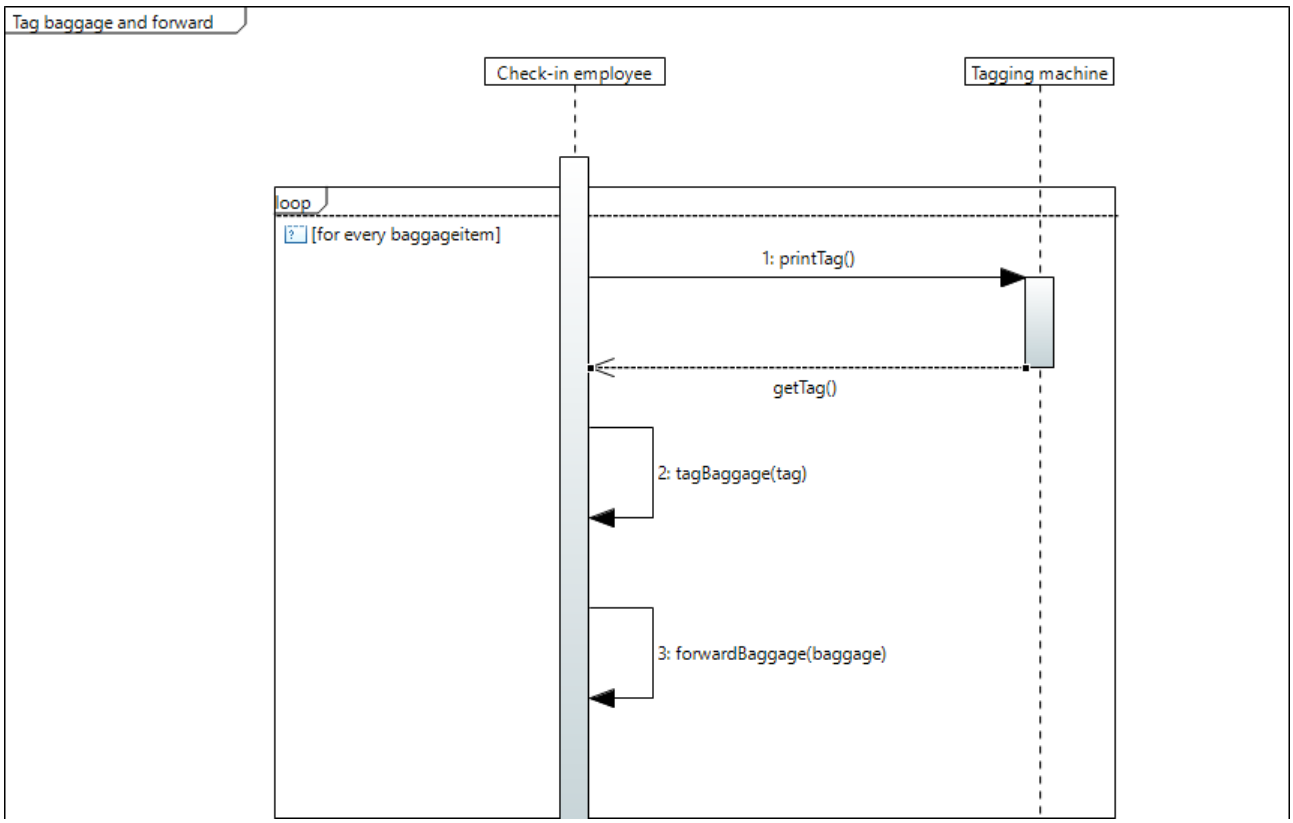
Check-in flight
Step 1: Passenger approaches check-in counter
Step 2: Check-in employee asks for passport
Step 3: Passenger hands over passport
Step 4: Check-in employee checks identity
Step 5: Check-in employee prints out flight tickets
Step 6: If the passenger has some baggage, he/she hands it over to the check-in employee
Step 7: Check-in employee gives back the passport including the flight tickets
Step 8: Passenger leaves counter



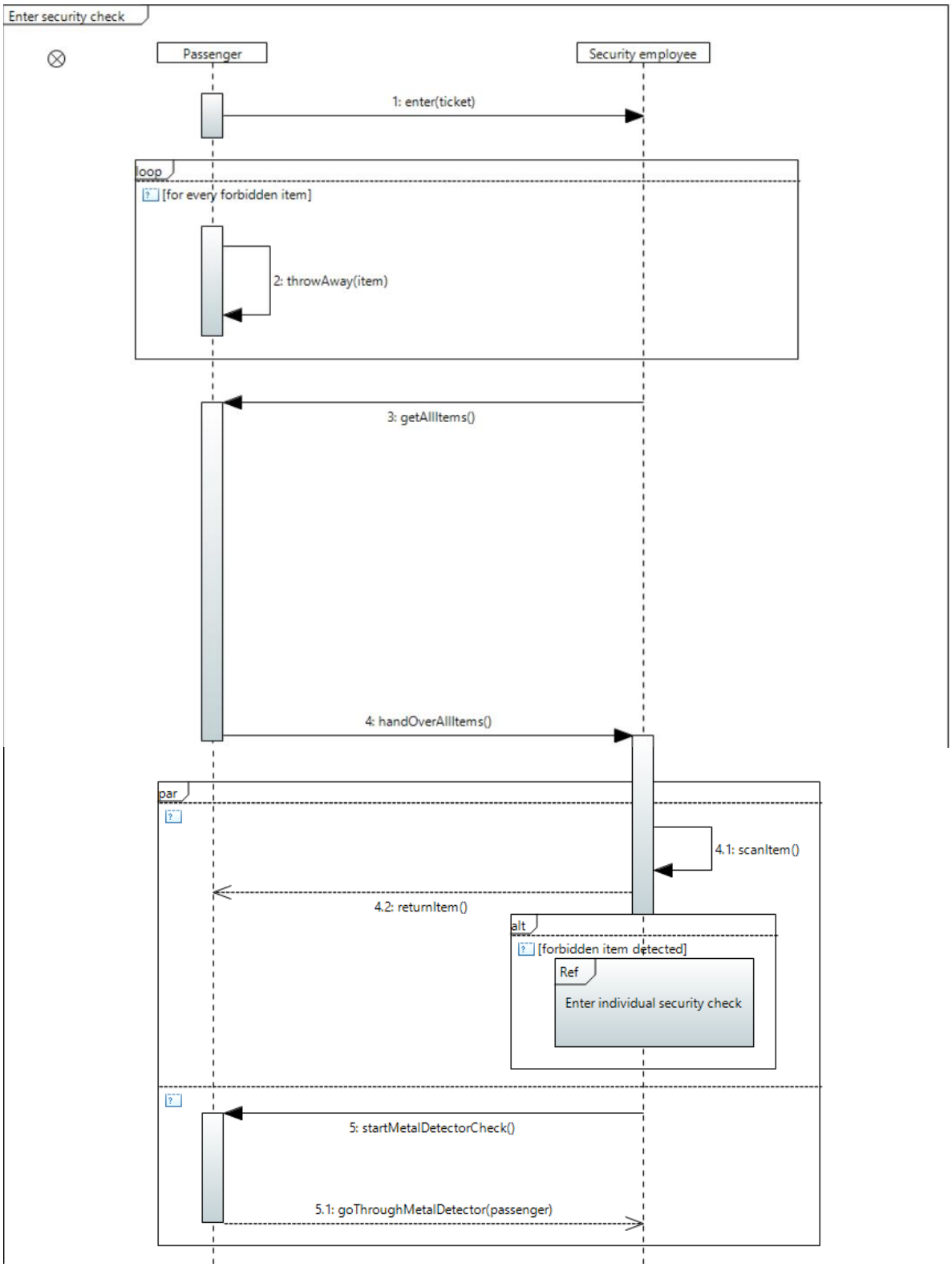
Hand over baggage
Step 1: Passenger hands over baggage
Step 2: Check-in employee weighs every baggage item
Step 3: Check-in employee checks weight limit of all baggage items
Step 4: Passenger pays additional fee for every kilogram which exceeds the weight limit



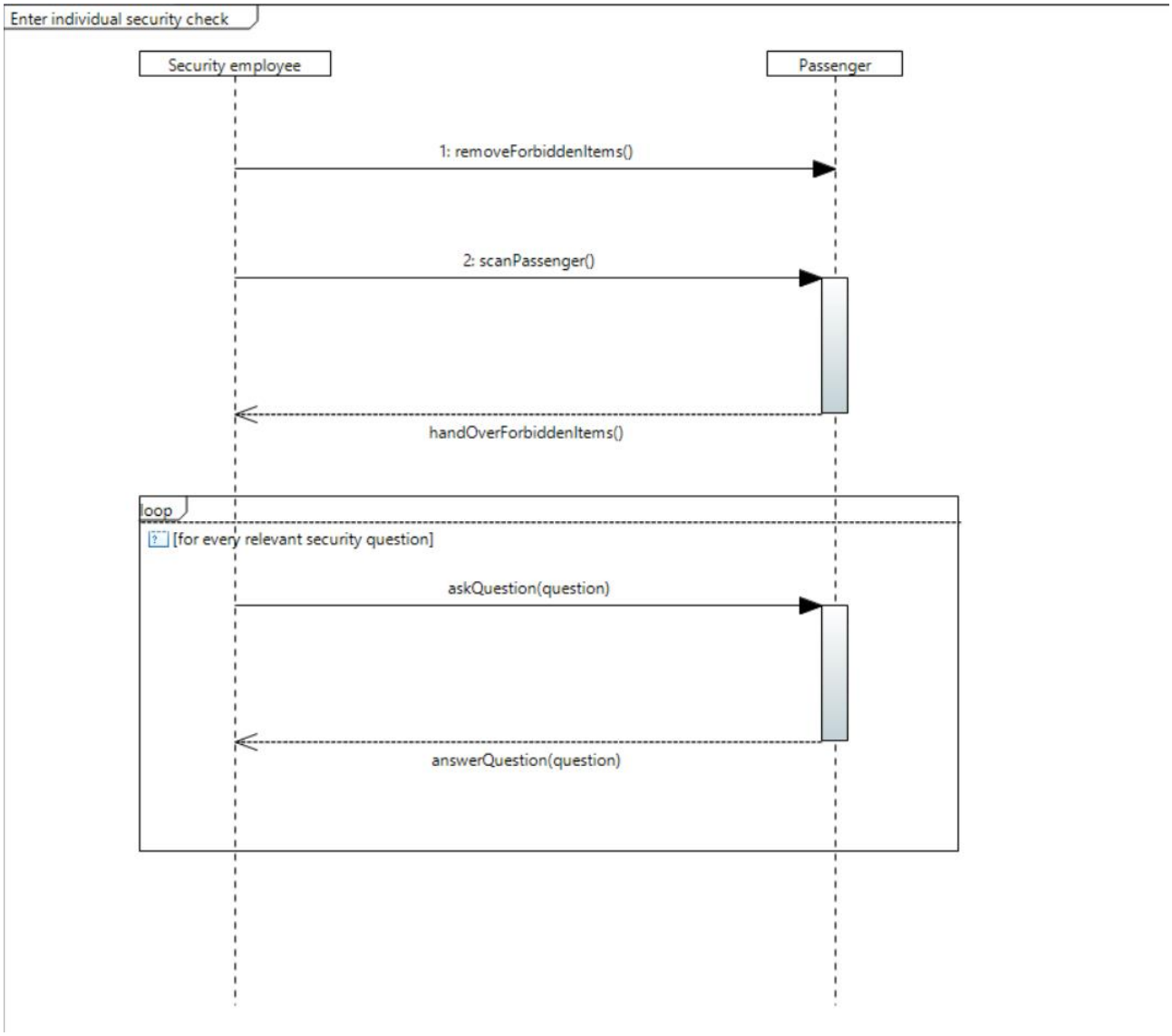
Tag baggage and forward
Step 1: Check-in employee prints out one tag per baggage item
Step 2: Check-in employee puts tag on every baggage item
Step 3: Check-in employee forwards all baggage items via the conveyor to the transportation location



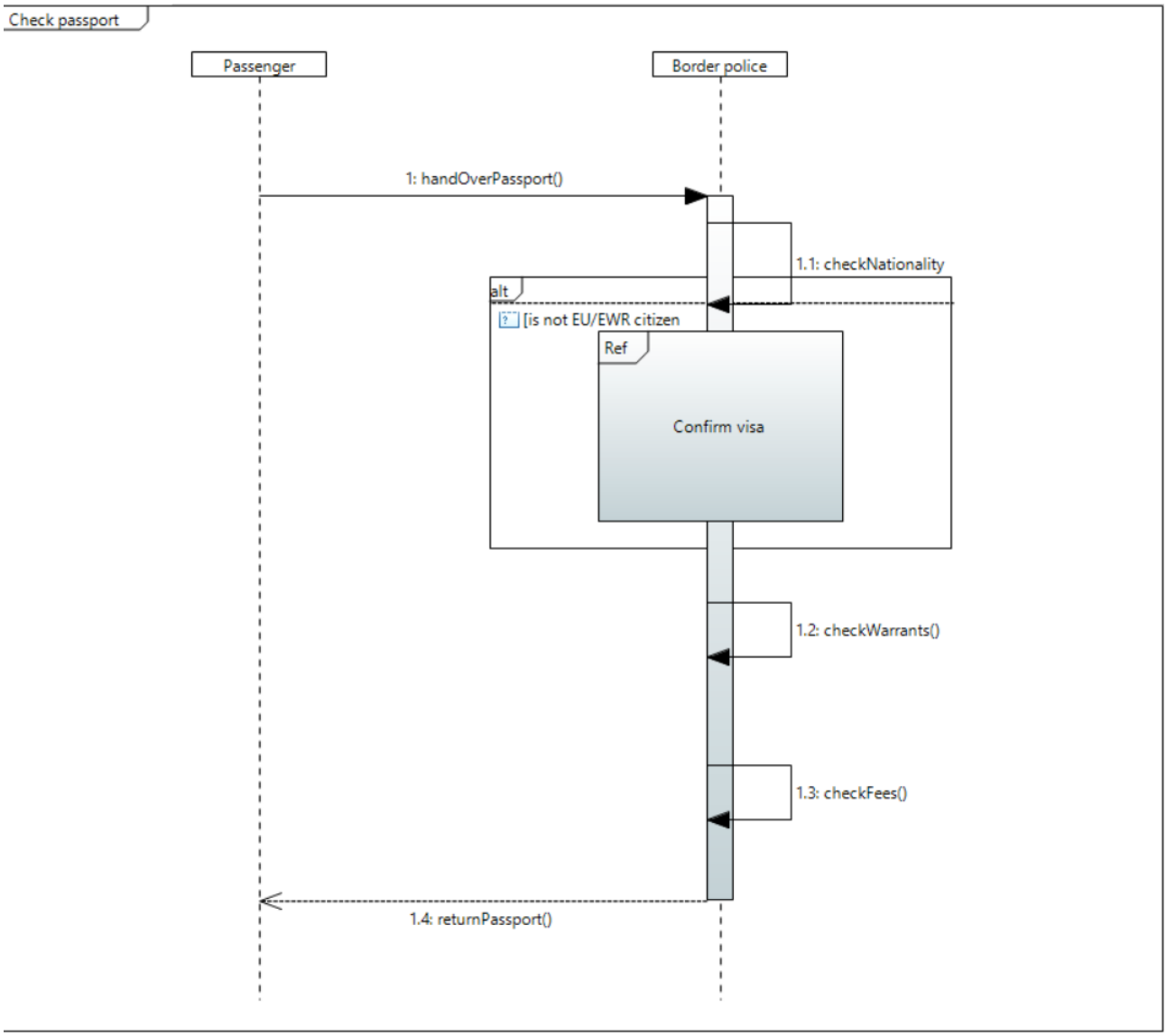
Enter security check
Step 1: Passenger enters security area with flight ticket
Step 2: Passenger throws all forbidden items into a bin
Step 3: Passenger puts all remaining personal items into a box
Step 4: Security employee scans personal items
Step 5: Passenger goes through metal detector
Step 6: Passenger takes back personal items
Step 7: Passenger leaves security area



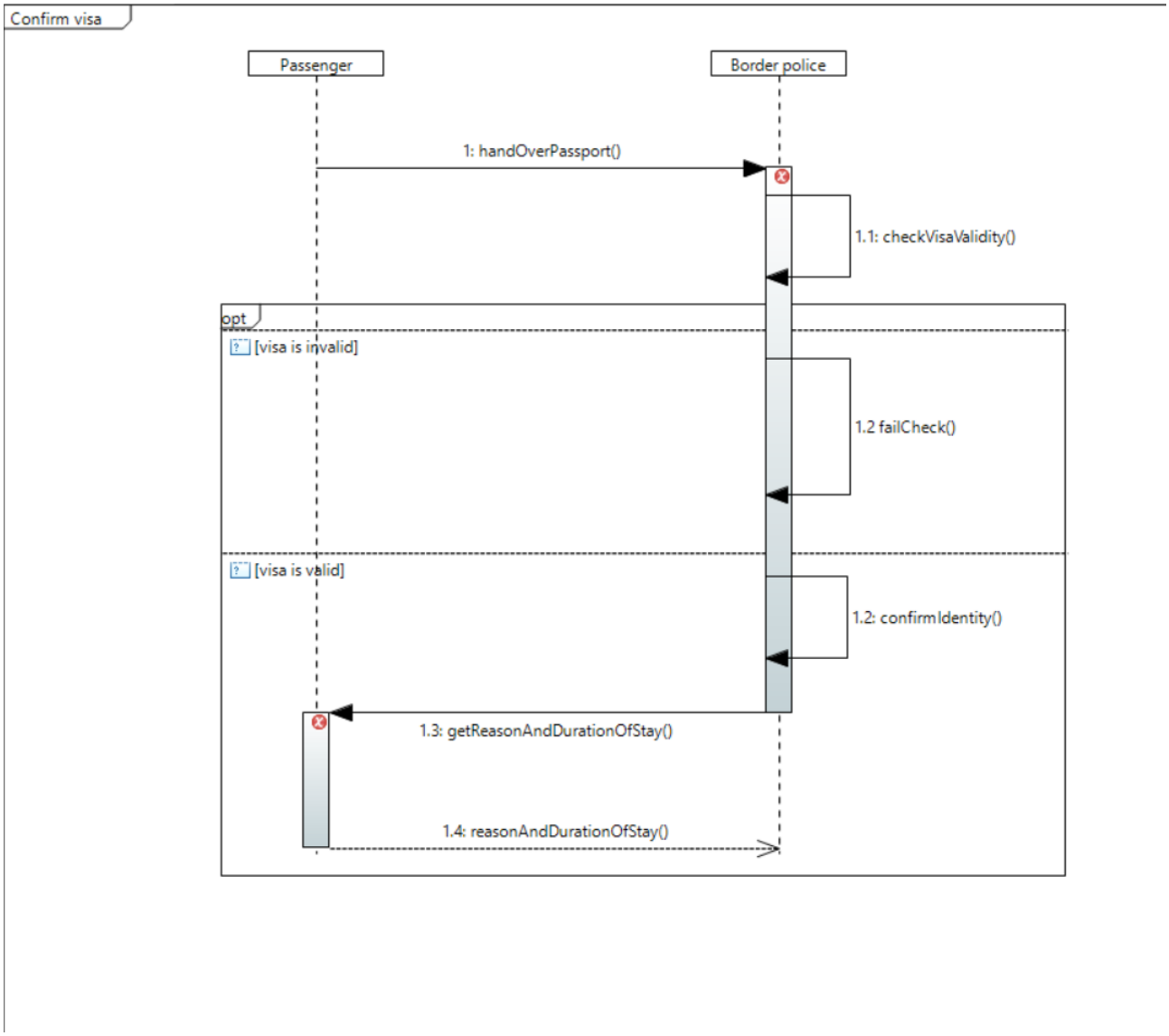
Start individual security check
Step 1: Security employee removes forbidden items from the passenger's personal belongings if present
Step 2: Security employee performs an individual scan on the passengers' body
Step 3: Security employee asks relevant questions
Step 4: Security employee resolves the situation
Step 5: Passenger leaves individual security check



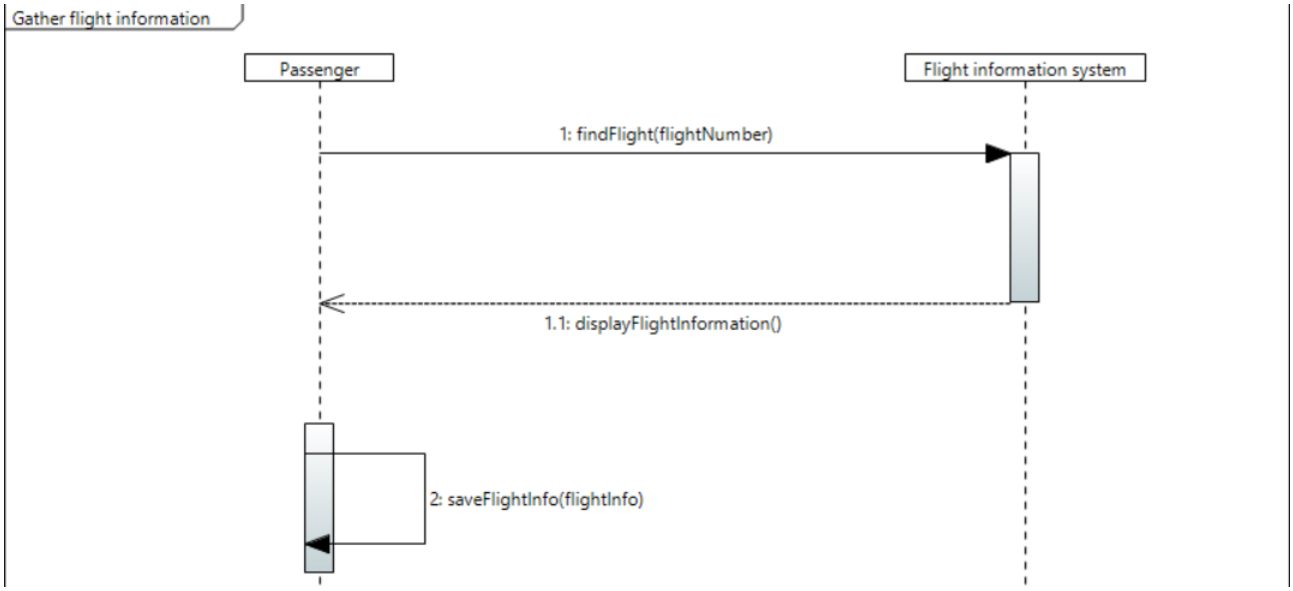
Check passport
Step 1: Passenger approaches border police
Step 2: Passenger hands over passport
Step 3: Border police checks nationality and possible warrants, fees, etc.
Step 4: Border police gives back the passport
Step 5: Passenger leaves border police check



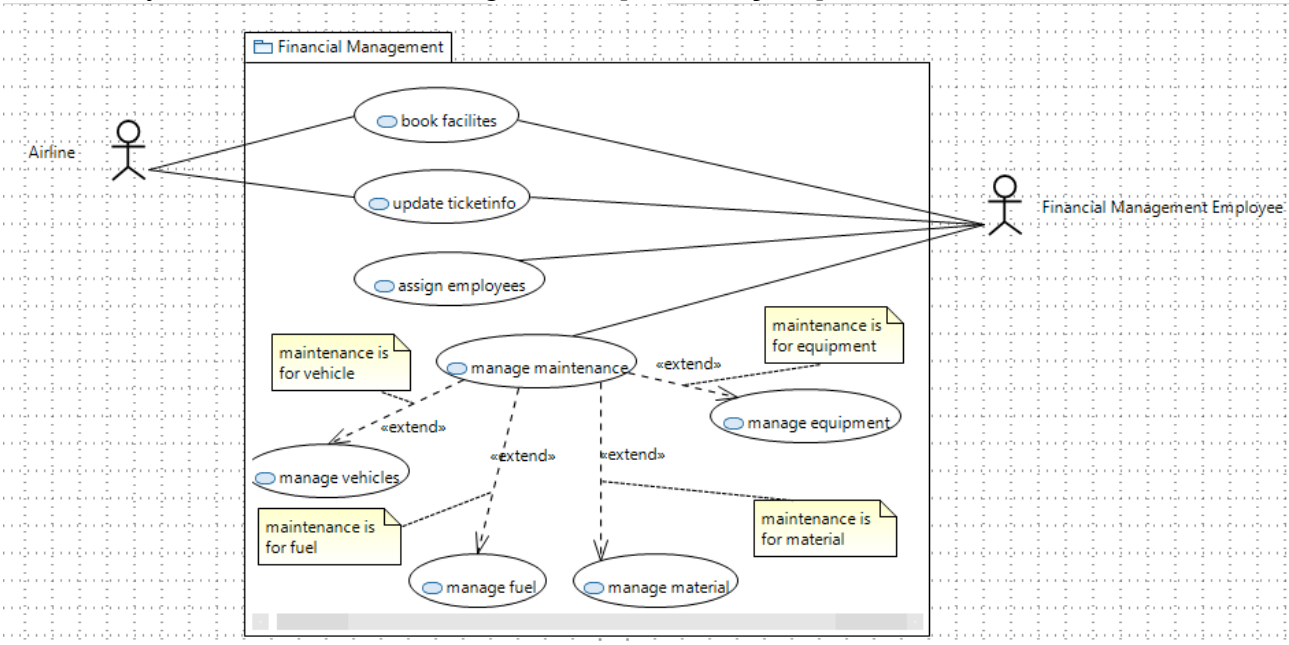
Confirm visa
Step 1: Border police inspects passport for visa
Step 2: Border police checks if visa is valid
Step 3: Border police confirms identity of passenger
Step 4: Border police gathers reason and duration of stay
Step 5: Passenger leaves border police office



Gather flight information
Step 1: Passenger spots flight information display
Step 2: Passenger looks for his/her own flight in the list of all upcoming flights
Step 3: Flight information provides flight number, departure/landing gate and time
Step 4: Passenger notes down important information
Step 5: Passenger leaves information area

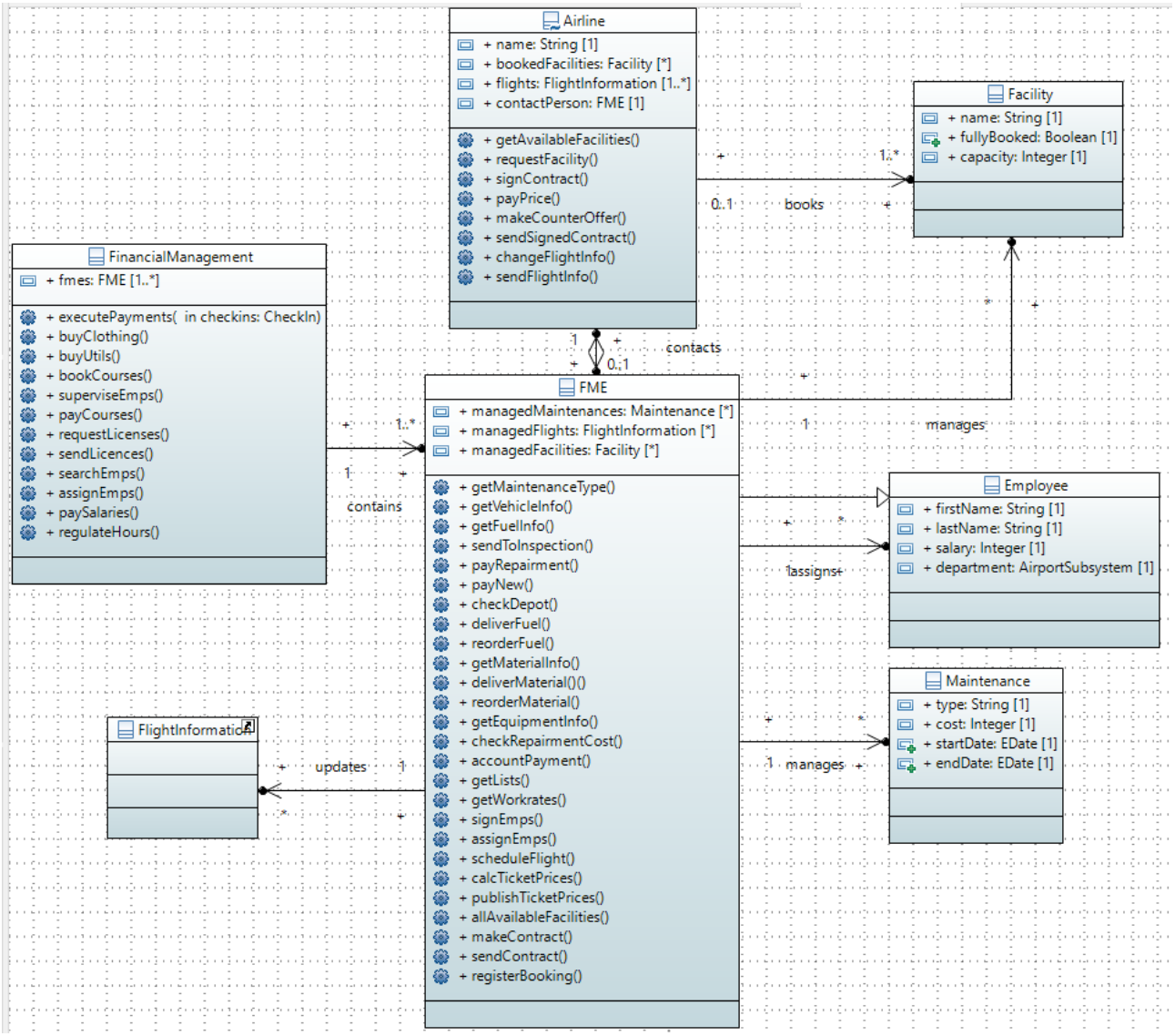


2.5 Sub-system 5/Financial Management - [Kaan **Baylan**]



The Airline is a business which operates regular services for carrying passengers or cargo by plane. This business will book a facility of the airside to park their aircrafts and will give the financial management info about their designated flight.

The FME is a person who works for the Financial Management department in our Airside. These employees will be receiving the desired bookings of the airside and reserve the facilities. Moreover, this department has got the task to assign employees to the other managements.
Furthermore, the employees are to be expected to manage the information of the tickets for the flight when the flight information gets changed. (Date, Time, and Gate)
Another important task is to manage the different maintenances.



This is the class diagram of the Financial Management. It contains the Actors of the Use-Case-Diagramm: Airline and FME. The Financial Management class is for the communication between the Subsystems. This class contains every FM Employee and through that it, contains everything, what is happening in this management.

The FME is the main class of this management because it contains all the functionalities. The FME contains the flights from the airlines, the different maintenances, and the different facilities.

The Airline contains the FME with, which he has contact, the different flights for which they are responsible and the facilities they have booked.

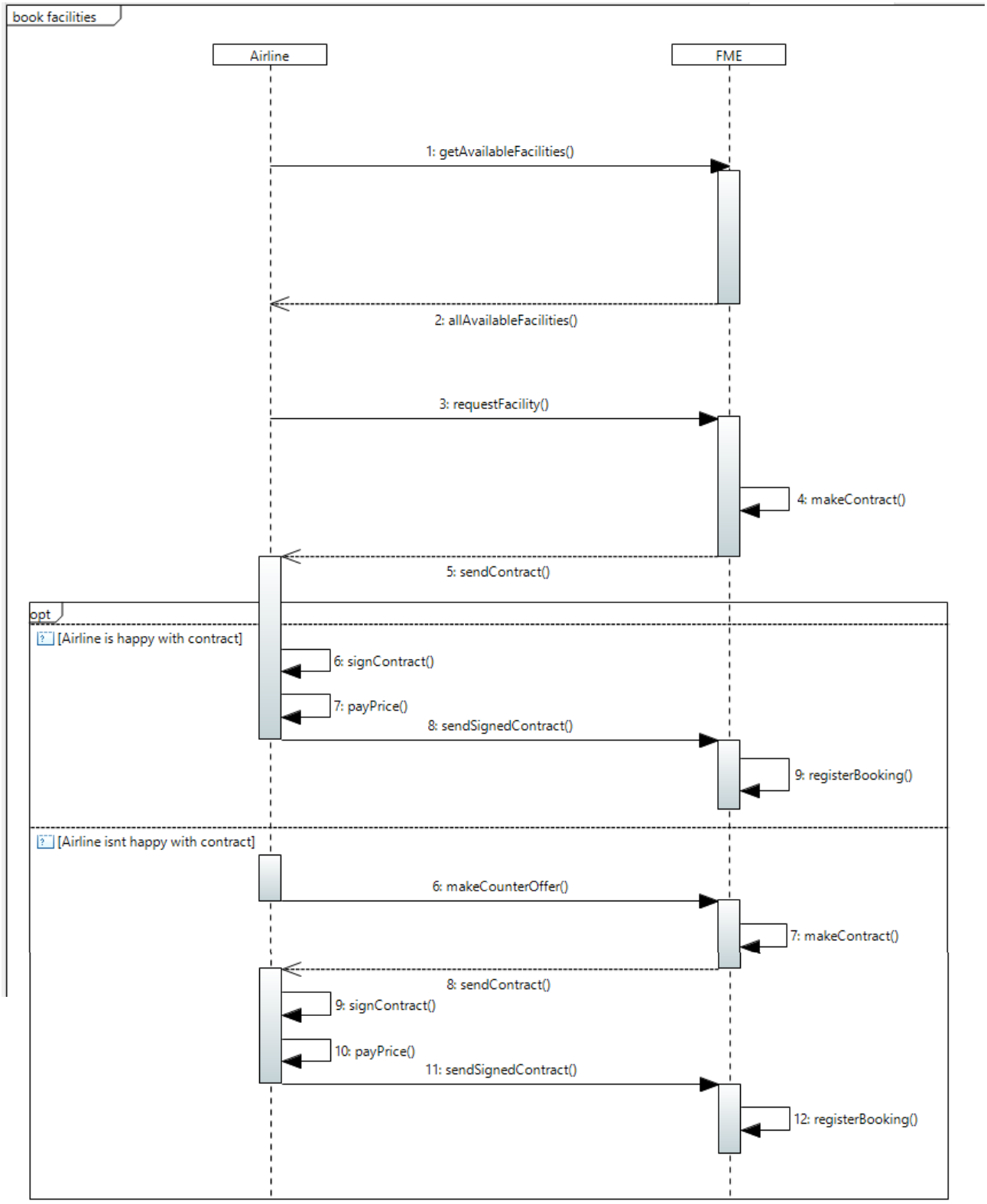
The Employee class is the super-class of every employee in our airport.

The Maintenance class contains the type, the cost, and the date when this maintenance started and when it ends.

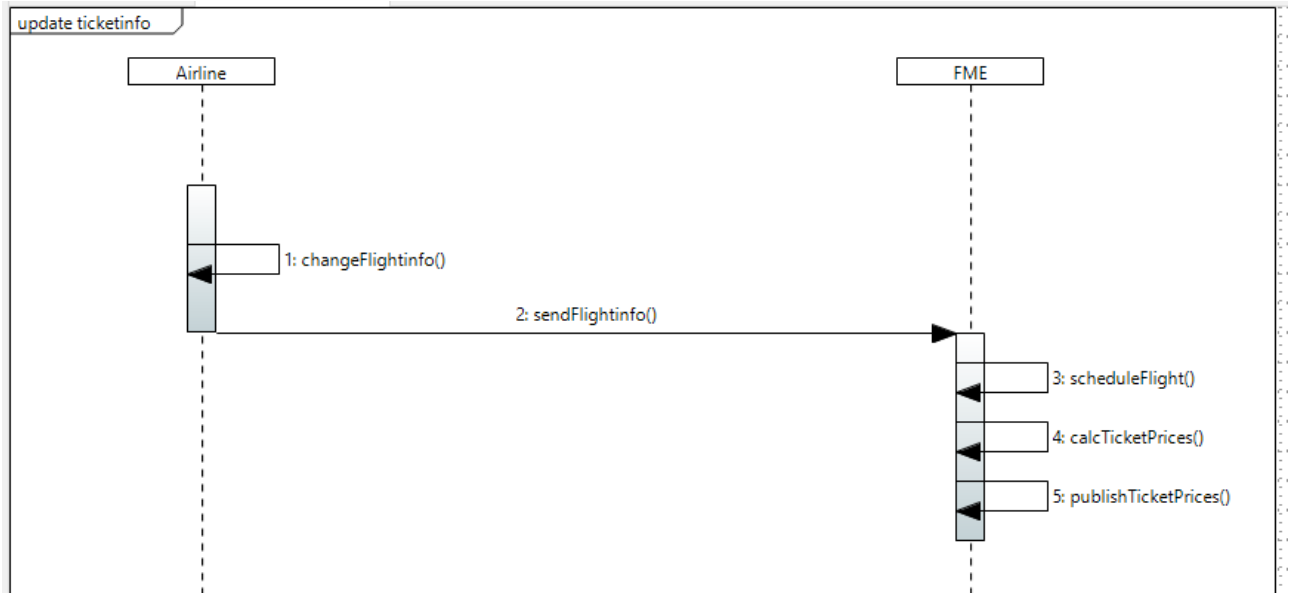
The Facility class has the name of the facility, its capacity and if it is fully booked.

The FlightInformation class is imported from Subsystem3.

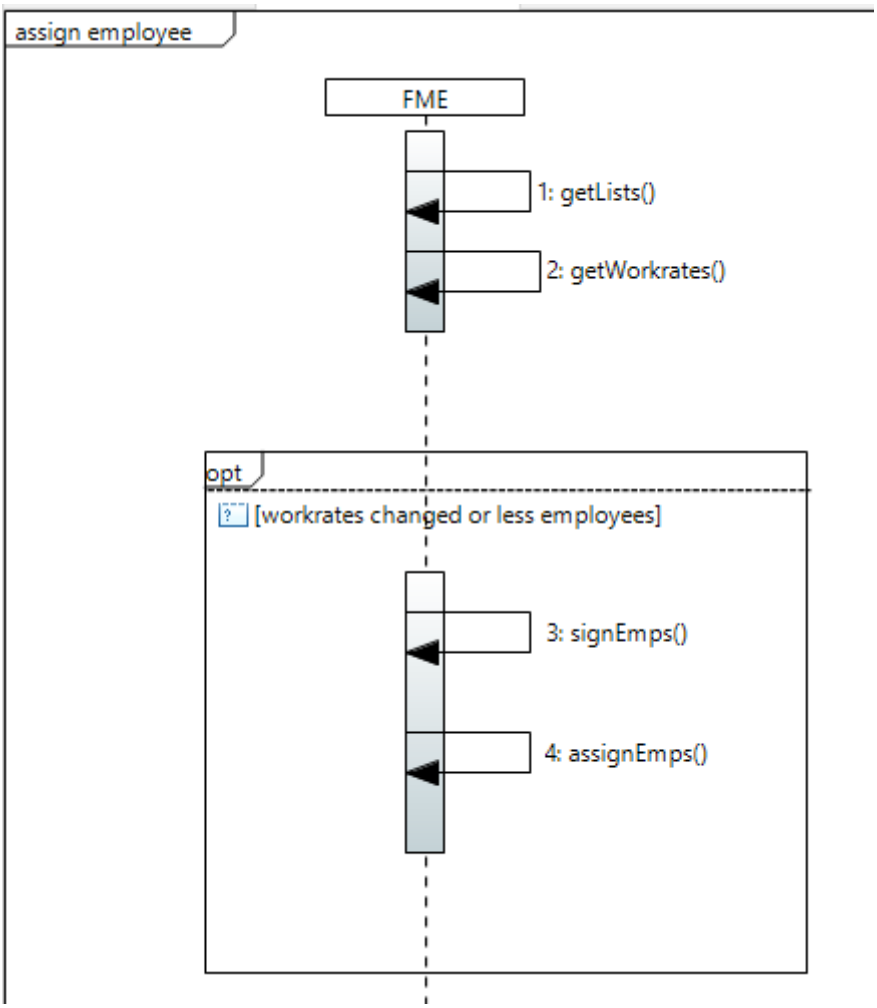
Book facilites:
Step 1: Airline gets information about which facilities can be booked through the system
Step 2: FME gives Airline information about which facilities are free
Step 3: Airline gives info about which facilities they would like to book
Step 4: FME gives Airline a contract about the booking
Step 5: Airline accepts and signs the contract if they are happy with it
Step 6: Airline pays the discussed amount
Step 7: FME registers the facilities as booked



Update Ticketinfo:
Step 1: Airline prepares a flight or changes a scheduled flight
Step 2: Airline gives the information to the FME
Step 3: FME checks info and (re-)schedules the flight
Step 4: FME calculates new ticket prices
Step 5: FME publishes new ticket information



Assign Employees:
Step 1: FME checks the employee lists of the different managements
Step 2: FME gathers info about the work rate of the other managements
Step 3: FME checks if any employee left the airside or one of the managements work rates is dropping or has dropped
Step 4: FME signs new employees
Step 5: FME assigns these employees to the management



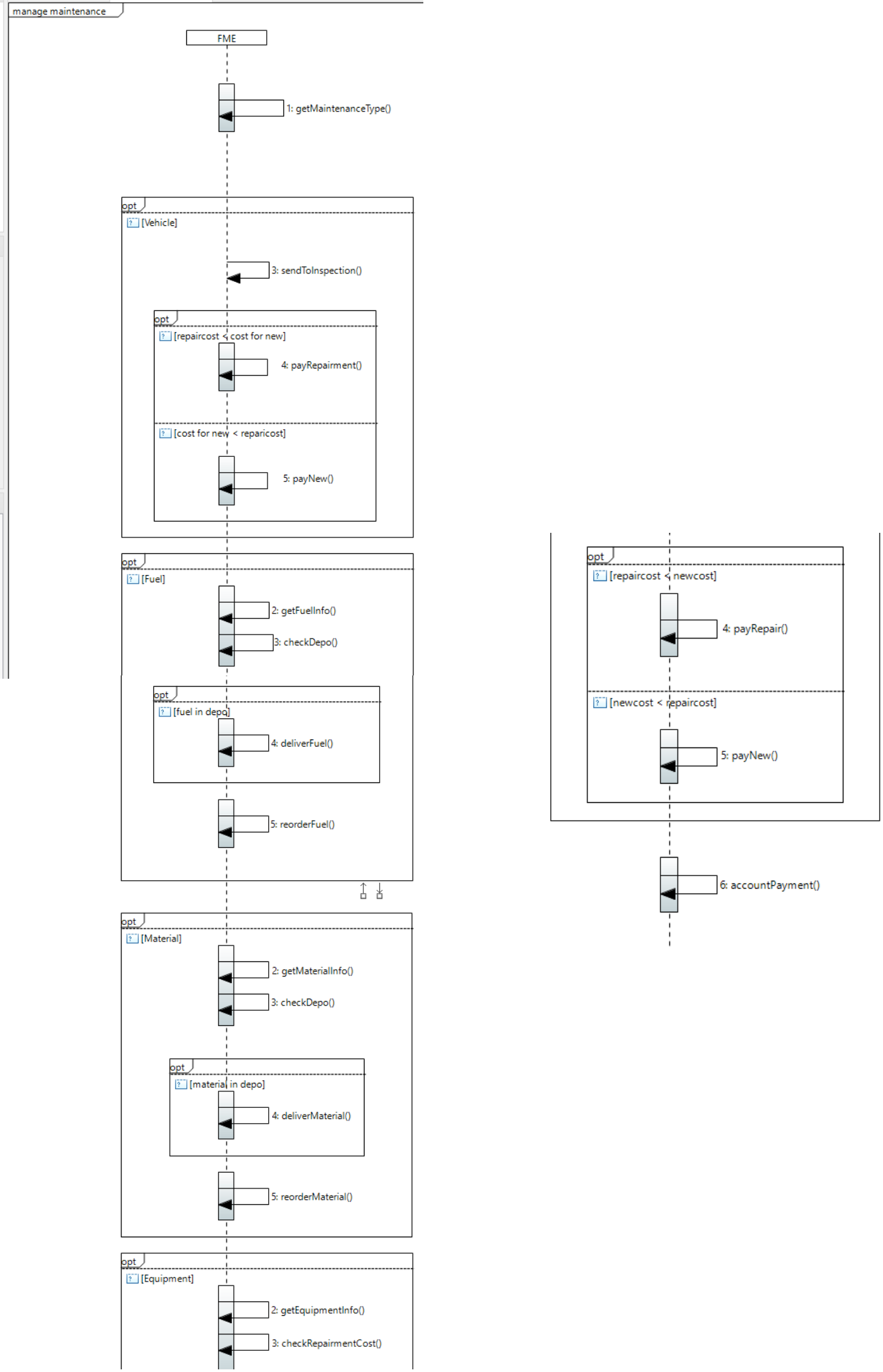
Manage Maintenances:
Step 1: Another management informs FME about needed maintenance
Step 2: FME checks which kind of maintenance it is
Step 3: Use case X is executed

Manage Vehicles:
Step 1: FME gets info about vehicle
Step 2: FME sends vehicle to inspection
Step 3: FME get info about the cost of repairment
Step 4: If repairment cost is too much FME pays for new vehicle
Step 5: FME accounts the cost of repairing / new vehicle

Manage Fuel:
Step 1: FME checks which fuel is needed
Step 2: FME checks if the needed fuel is stored in a depo of ours
Step 3: If fuel is in our depo, it gets delivered
Step 4: FME calculates how much fuel is left
Step 5: if fuel is low or no fuel was found in depo FME orders new
Step 6: FME accounts the cost of the new fuel

Manage Material:
Step 1: FME checks which material is needed
Step 2: FME checks if the needed material is stored in a depo of ours
Step 3: If material is in our depo, it gets delivered
Step 4: FME calculates how much material is left
Step 5: if material is low or no material was found in depo FME orders new
Step 6: FME accounts the cost of the new material

Manage Equipment:
Step 1: FME gets info about equipment
Step 2: FME checks if equipment can be repaired
Step 3: FME get info about the cost of repairment
Step 4: If repairment cost is too much FME pays for new equipment
Step 5: FME accounts the cost of repairment / new equipment


manage maintenance
FME
1: getMaintenanceType()
opt
[Vehicle]
3: sendToInspection()
opt
[repaircost < cost for new]
4: payRepairment()
[cost for new < repaircost]
5: payNew()
opt
[Fuel]
2: getFuelInfo()
3: checkDepo()
opt
[fuel in depo]
4: deliverFuel()
5: reorderFuel()
opt
[Material]
2: getMaterialInfo()
3: checkDepo()
opt
[material in depo]
4: deliverMaterial()
5: reorderMaterial()
opt
[Equipment]
2: getEquipmentInfo()
3: checkRepairmentCost()
opt
[repaircost < newcost]
4: payRepair()
[newcost < repaircost]
5: payNew()
6: accountPayment()

3. Design Decisions

Overall architecture package diagram:

We decided to connect every subsystem with the other ones since we have no central unit like the control system management. Therefore, Airside and Landside Management have to communicate directly with each other.

System level use case diagram:
Managing employees is split into one overall use-case which contains generic tasks of employee management and the subsystem specific use-cases (manage landside/terminal/airside employee) contain steps which are only needed for the respective subsystem.

General style of use-cases:

We decided to structure conditions and exceptions with <<extend>> since this was the best way to display such flows. Note that the condition which is formulated in the yellow box must be satisfied in order to enter the extending use-case.

System level class diagram:

We decided to use the composition relation between the different subsystems and the overall system since every subsystem is specifically modelled for an airport management system. It would be possible to extend this for other management systems as well and if that would be the case, we could exchange the composition for an aggregation in order to share these subsystems with other systems as well. But that would need some adaptions and this is not in the scope of this project.

It was not possible to create associations between the referenced subsystem classes in the system-level class diagram since papyrus throws an error in that case. But every subsystem directly communicates with all other subsystems and therefore, an association between all subsystems is present.

4. Change Log

Deliverable	**Changes**
D1	Formed team and choose subsystems
D2	Added overall architecture (one version per team member)
D3	Combined and refined overall architecture using a package diagram Added use-case diagrams for overall system and subsystems
D4	Added detailed use-case specification for every use-case

Report and Presentation	Refined use-case specifications to be more detailed
Feedback on Report	Discussed and updated use-case specifications according to feedback
D5	Added domain model
D6	Defined actions in sequence diagram per use-case Converted domain model to class diagram Added sequence diagram actions as methods in class diagram
Report and Presentation 2	Integrated changes from M2 Correction Guideline (especially to show one exception case in the sequence diagram)
Feedback on Report 2	Adapted changes from feedback Refinement of class relations Fixing sequence diagram problems