IFSC 3360: System Analysis and Design

Final Project Report

Automated Baggage System Controller

Paul Larison

Introduction

The purpose of a baggage control system is to transport passenger bags from check-in to destination in a fast, reliable and secure manner. It must be fast because the volume of bags is high; even a large, well organized baggage crew would be overwhelmed if all bags needed to be transported manually. The system therefore uses automated sensors and conveyor belts to transport the bags from one station to the next. It must be reliable because lost baggage is a very big deal to customers. Misplaced or lost bags will influence the reputation of carriers, resulting in lost revenue. It must be secure, because security is a legal requirement for airports. Speeding up security processing is one reason checked bags are separated from passengers prior to screening.

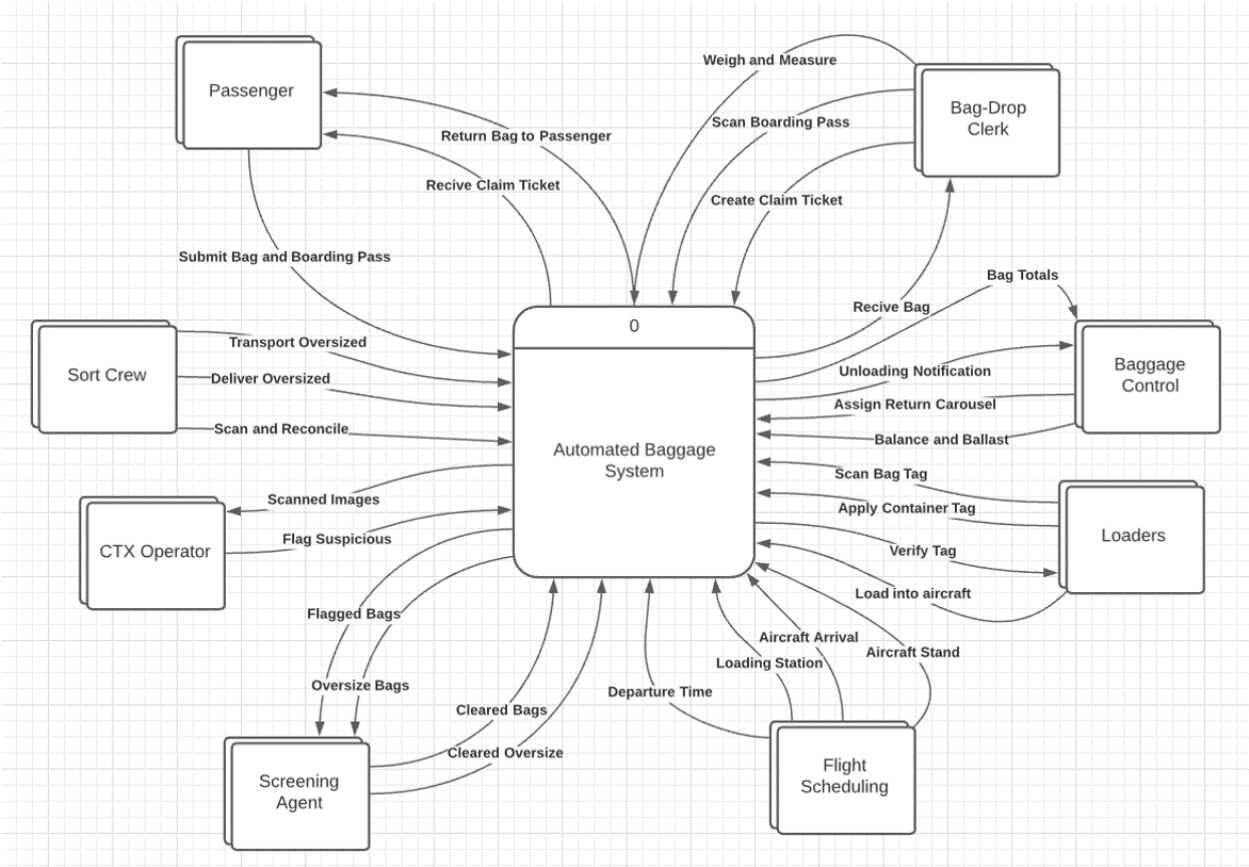
This is a new system. Each airport has its own layout and needs, so each baggage control system must be custom built. Updates and upgrades are possible, but major changes would require the system to be rebuilt. The system outlined by this assignment is intentionally generic. It is also very simplified; the design of an actual baggage handling system would be far in excess of the scope and complexity required by this assignment. The controls for individual subsystems, such as bagstore cranes and track switches, are also beyond the scope of this project.

An automated baggage handling system is a requirement for any major airport. This design is for the computerized control system that runs the physical system. While a simple conveyor system would indeed be an improvement over manual transport, a truly efficient system requires features such as intelligent routing, automated sensors, on demand status updates and opportunities for human judgement to correct problems.

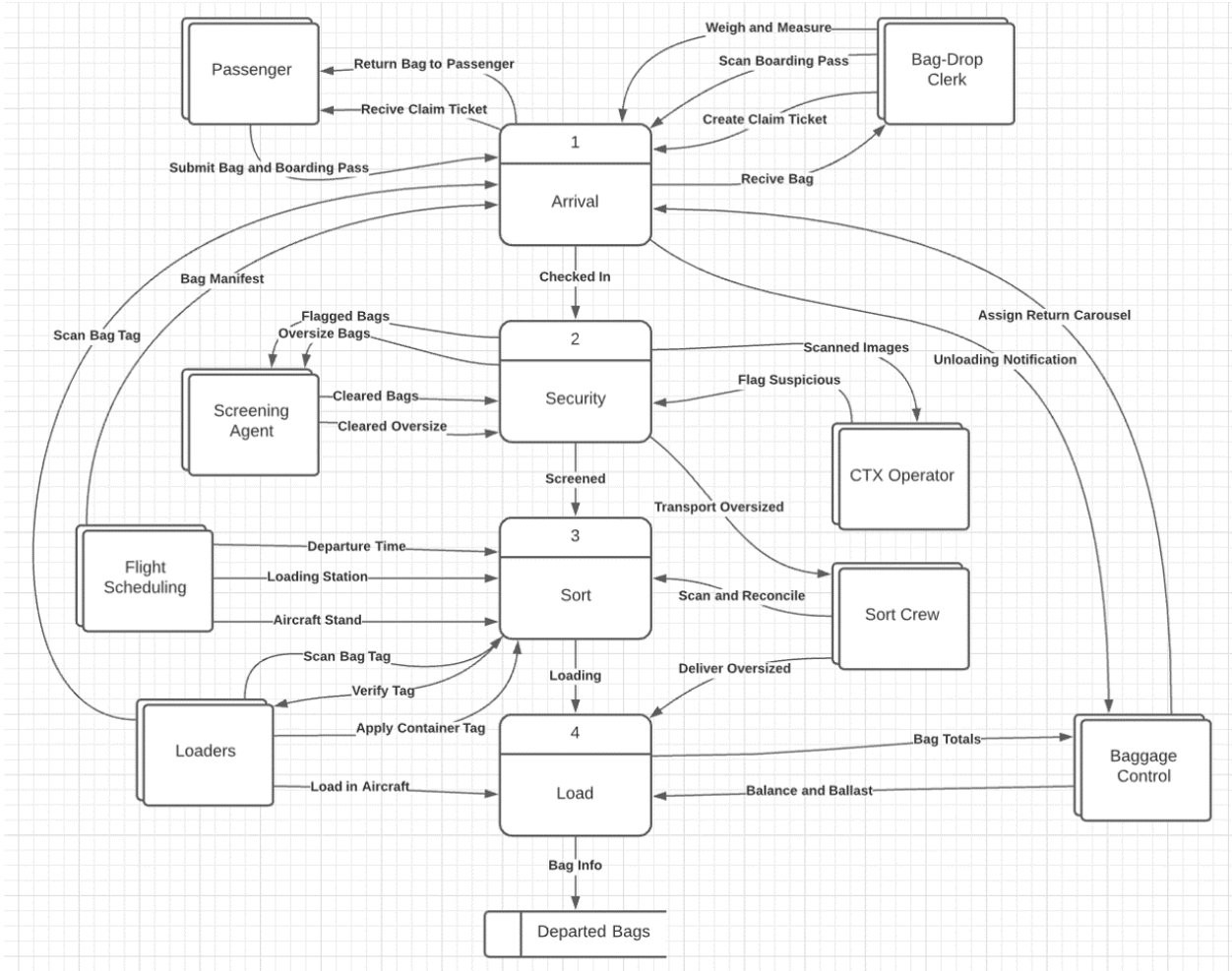
The design of the logical system mirrors the design of the physical system. A Bag object is created when the bag enters the system, either at check-in or when an aircraft is unloaded. The Bag object is then passed from subsystem to subsystem along the same logical route as the physical bag. At certain points in the flow the barcode on the physical bag is scanned and compared to the ID number of the Bag Object to ensure reliability. When the Bag object is no longer needed, its information is archived in case it needs to be referenced by Customer Service.

Data Flow Diagrams

The Context Diagram, showing how the system as a whole interacts with various users. This system has many users; for the purpose of this project they have been simplified into groups. For example, “Sort Crew” would be comprised of dozens of individual users.

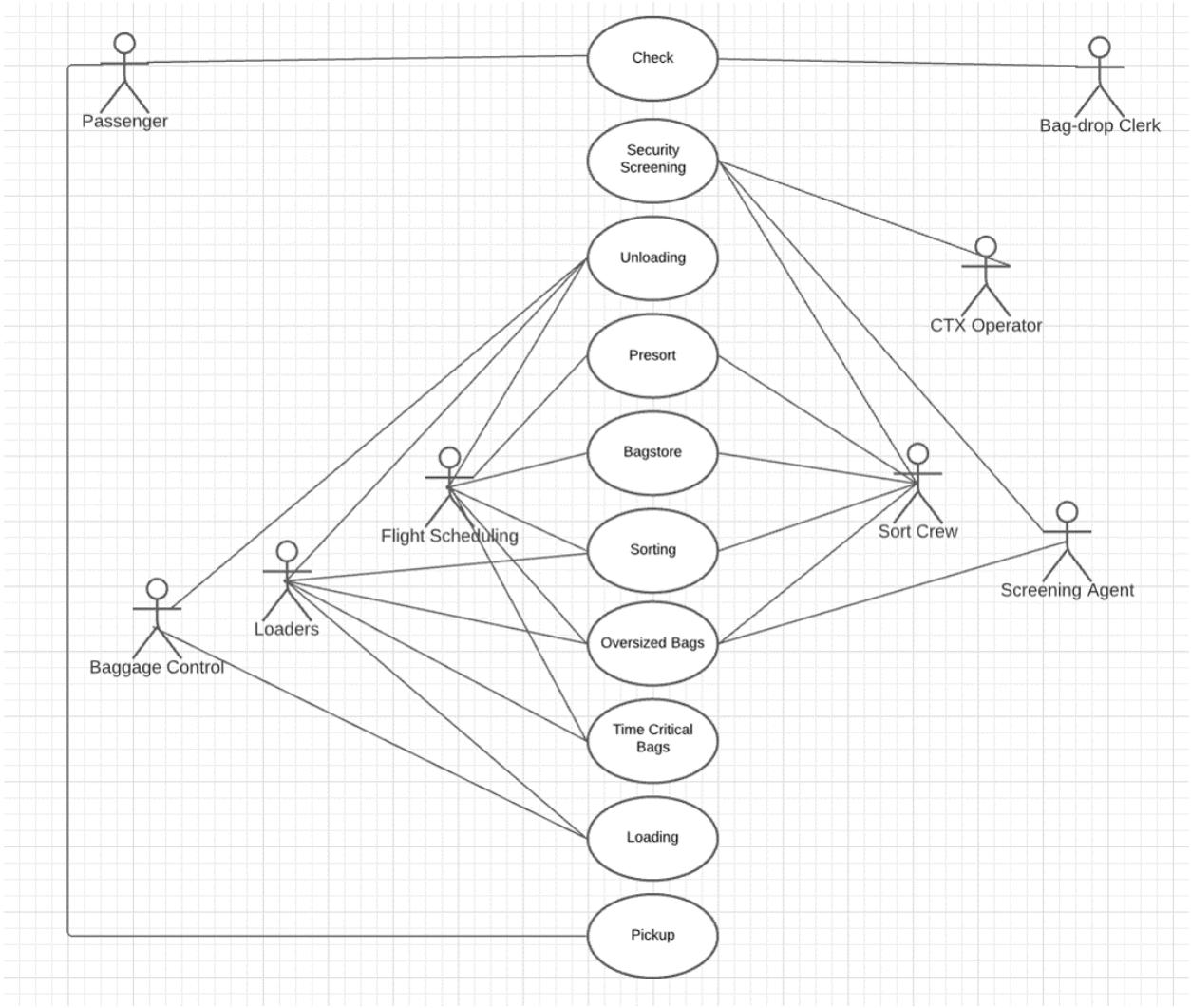


The DFD0 Diagram expands the system into its most basic components while leaving the users unchanged. “Arrival” is the arrival of the bag into the system from any source; because of this it is also where bags arriving on aircraft leave the system.



Use Case Diagram

The Bag object is created at Use Case 1 or 3. It then travels down the use cases until it is destroyed at Use Case 9 or 10.



Fully Dressed Use Cases

UC1: Check In
Actors: Passenger, Bag-drop clerk

Description: The use case submits a bag into the system

Main success scenario:

1. Clerk greets passenger
2. Passenger submits boarding pass
3. Clerk scans boarding pass
4. Passenger submits bag
5. Clerk weighs and measures bag
6. System prints bag ticket and claim ticket
7. Bag status initialized as “checked-in”
8. Clerk applies bag ticket to bag
9. Clerk submits bags into automated system
10. Clerk gives claim ticket to passenger

Alternate Scenarios:

3a. Boarding pass will not scan
    1. Hand key pass number
3b. Pass number not recognized
    1. Refer passenger to customer service
5a. Bag overweight
    1. Additional charge to passenger
5b. Bag oversize
    1. Additional charge to passenger
    2. Route bag directly to aircraft stand
5c. Customer refuses additional charges
    1. Bag cannot be checked
    2. Call supervisor for support if needed
6a. Tickets fail to print
    1. Refill printer
    2. Call help desk

Preconditions

1. Passenger has valid boarding pass

Postconditions

1. Passenger receives claim ticket from clerk
2. Passenger continues on to flight
3. Bags proceed to UC2 security screening

UC2: Security Screening
Actor: CTX Operator, Screening Agent, Sort Crew

Description: The use case checks the baggage for bombs or other safety hazards

Main success scenario:

1. Tip and align bags so they lie flat
2. CTX scan
3. CTX operator allows bag to pass
4. Bag status updated to “screened”
5. Standard bags continue on automated system
6. Oversized bags transported by Sort crew

Alternate scenarios:

1a. Oversize bags

1. Manually checked by screening agent

3a. CTX operator flags bag for inspection

1. Manually checked by screening agent

3b. Contraband found

1. Remove bag from luggage flow by screening agent
2. Screening agent notifies supervisor

Preconditions

1. Bag arrives from UC1 Check-in

Postconditions

1. Standard bags proceed to UC4 Timeliness Sort
2. Oversize bags proceed to UC7 Oversized Bags

UC3: Unloading
Actors: Loaders, Baggage control, Flight Scheduling

Description: Inputs bags from incoming aircraft into the system

Main success scenario:

1. Baggage control notified of arriving aircraft by Flight Scheduling
2. Baggage control assigns reclaim carousel to flight
3. Loaders transfer bags from aircraft to automated system
4. Loaders scan bag tags
5. Bag status initialized as “transfer” or “arrival”
6. Route bags to sorter or carousel

Alternate Scenarios:
4a. Tag scan error

1. Manual scan and resolution by Baggage crew

Postconditions:

1. Transfer bags proceed to UC4 presort
2. Arrival bags proceed to UC10 pickup

UC4: Presort
Actor: Flight Scheduling, Sort Crew

Description: The use case prevents congestion by delivering bags at the correct time

Main success scenario:

1. Automated bag ticket scan
2. Get departure time from Flight Scheduling
3. Early bags a routed to bag store
    a. Bag status updated to “storage”
4. On time bags go to sorter
    a. Bag status updated to “sorting”
5. Time-critical bags go to aircraft stand
    a. Bag status updated to “rush”

Alternate Scenarios:

1a. Tag scan error

1. Manual scan and resolution by Sort crew

Preconditions

1. Standard sized bags

Postconditions

1. Early bags proceed to UC5 Bagstore
2. On time bags proceed to UC6 Sorting
3. Time-critical bags proceed to UC8 Time-critical

UC5: Bagstore
Actor: Sort crew

Description: Early bags are stored here to prevent congestion at loading terminals

Main success scenario:

1. Deposit bag in numbered tray
2. Crane places tray in storage receptacle
3. When bag status changes to on time, crane retrieves tray from receptacle
4. Bag returned to conveyor system

Alternate scenarios:

*a. Bag jammed or not found

1. Sort crew investigates problem

Precondition:

1. At least 1 hour until departure time

Postcondition

1. Bag proceeds to UC6 sorting

UC6: Sorting
Actor: Sort crew, Flight Scheduling, Loaders

Description: Bags are delivered to the appropriate loading bays

Main success scenario:

1. Get loading station from Flight Scheduling
2. Print container tag
3. Container tagged by Loaders
4. Bags dropped down chute to loading station
5. Bag tag scanned and verified by Loaders
6. Update bag status in system to “loading”
7. Bags loaded into Container by Loaders

Alternate scenarios:

2a. Container tag won’t print
    1. Refill printer
    2. Call help desk
4a. Bag jammed
    1. Sort crew investigates problem
5a. Tag won’t scan
    1. Loaders reconcile and retag bag
5b. Bag at wrong station
    1. Manually relocated by Loaders

Preconditions:

1. Flight Scheduling has assigned a loading station

Postconditions:

1. Bags continue to UC9 Loading

UC7: Oversized bags
Actors: Sort crew, Screening agent, Flight scheduling, Loaders

Description: Oversized bags could jam the system and must be transported manually

Main success scenario:

1. Sort crew retrieves bag from screening agent
2. Sort crew retrieves loading station from Flight Scheduling
3. Sort crew delivers bag to loading station
4. Bag tag scanned and verified by Loaders
5. Update bag status in system to “loading”
6. Bags loaded into cart by Loaders

Alternate Scenarios

4a. Tag won’t scan

1. Loaders reconcile and retag bag

4b. Bag at wrong station

1. Manually relocated by Sort crew

Preconditions:

1. Oversized Bag

Postconditions:

1. Bags continue to UC9 Loading

UC8: Time-Critical bags
Actors: Flight Scheduling, Loaders

Description: Time-critical bags must be rushed directly to the aircraft

Main success scenario:

1. Retrieve aircraft stand from Flight Scheduling
2. Bag travels from security screening on high-speed rail
3. Bag arrives at aircraft stand
4. Bag tag is scanned by Loaders
5. Bag status updated to “loading”
6. Bag placed in cart by Loaders

Alternate Scenarios

4a. Tag won’t scan

1. Baggage crew reconciles and retags bag

4b. Bag at wrong station

1. Manually relocated by baggage crew

Preconditions:

1. Cart has departed from loading station

Postconditions:

1. Bags continue to UC9 Loading

UC9: Loading
Actors: Loaders, Baggage control

Description: Bags must be physically loaded onto the aircraft

Main success scenario:

1. Cart driven to aircraft by Loaders
2. Distribution and ballast assigned by Baggage control
3. Baggage crew stacks bags in aircraft
4. When container is empty, update bag status to “loaded”

Alternate scenarios:
None

Preconditions:

1. All on time bags have arrived at loading station

Postconditions:

1. Flight departs with all bags aboard.

UC10: Pickup
Actors: Passenger

Description: Arriving passengers are reunited with their checked luggage

Main success scenario:

1. Routing system updates bag status to “arrived”
2. Bag arrives in reclaim carousel
3. Passenger retrieves bag

Alternate scenarios:

3a. Passenger cannot find bag

1. Refer passenger to customer service

Precondition:

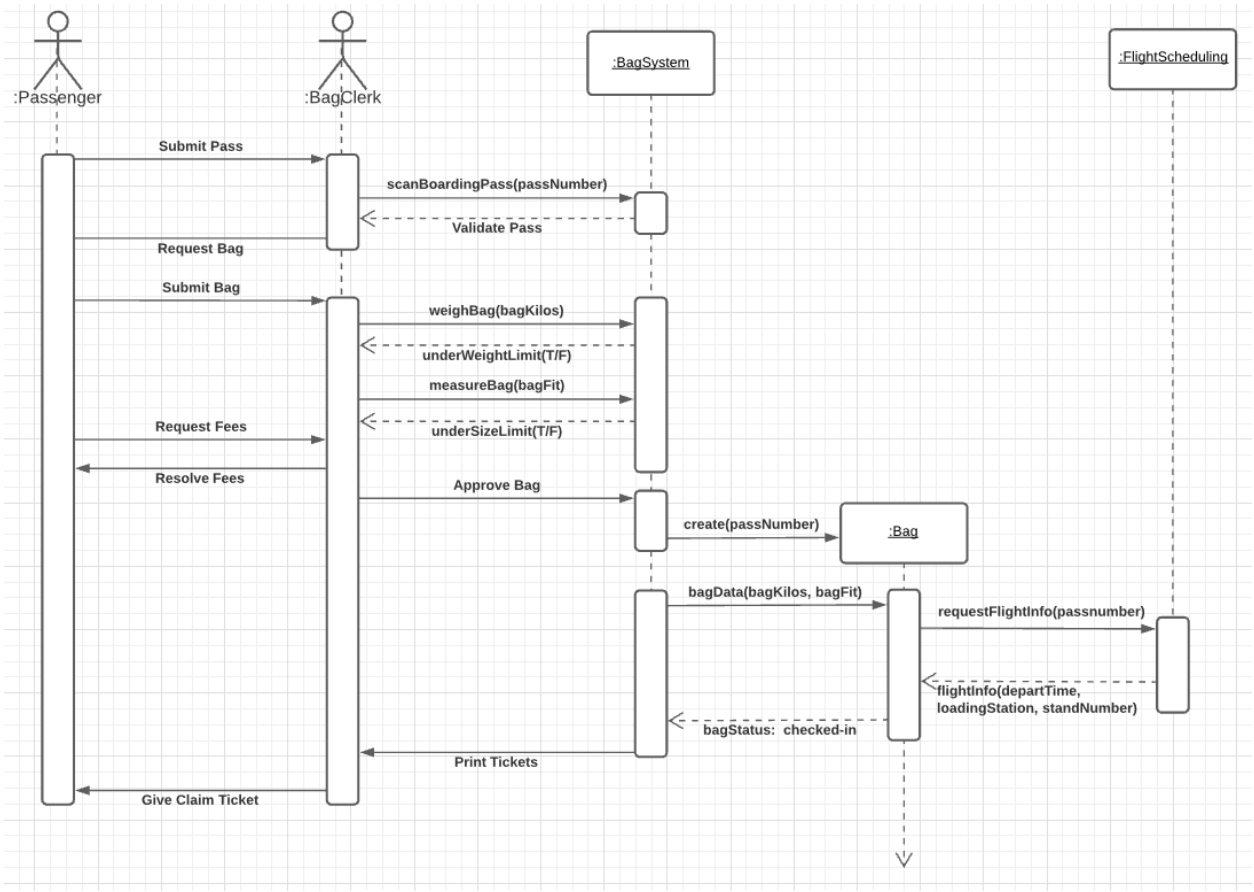
1. Bag at final destination

Postconditions:

1. Passengers depart the airport with all their bags.

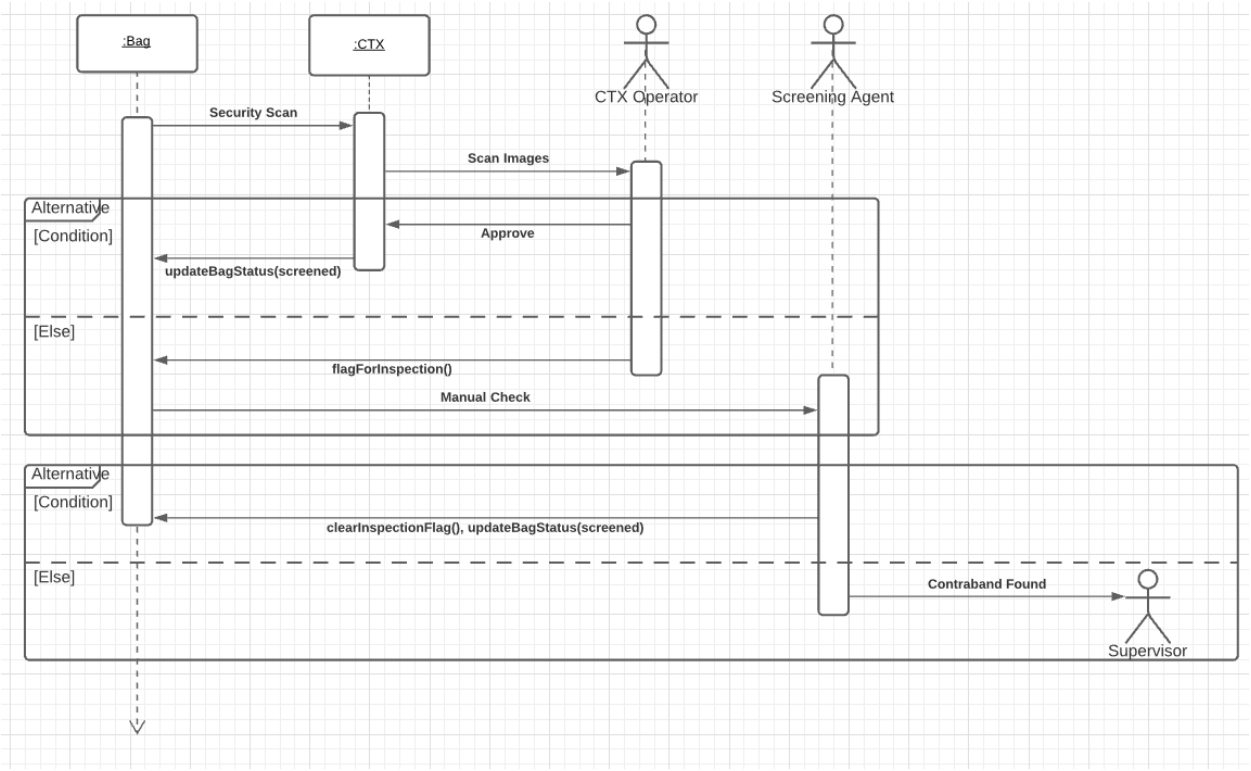
Use Case 1: Check In

The primary user is the bag clerk, who assists the passenger with the check-in process. The sub-system at the checkout terminal gathers the data needed to create a Bag object, which is then sent to the next use case.



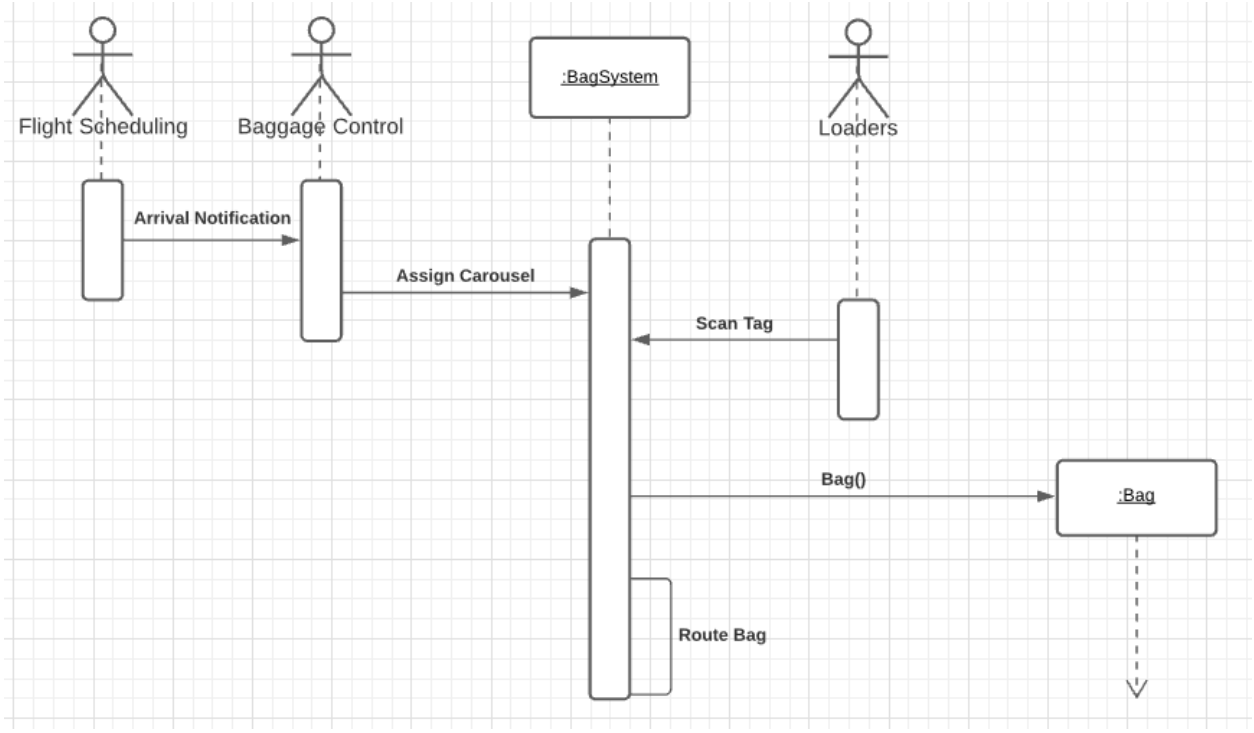
Use Case 2: Security Screening

All bags must pass through a security checkpoint. The properties of the Bag object are updated with the results of the screening.



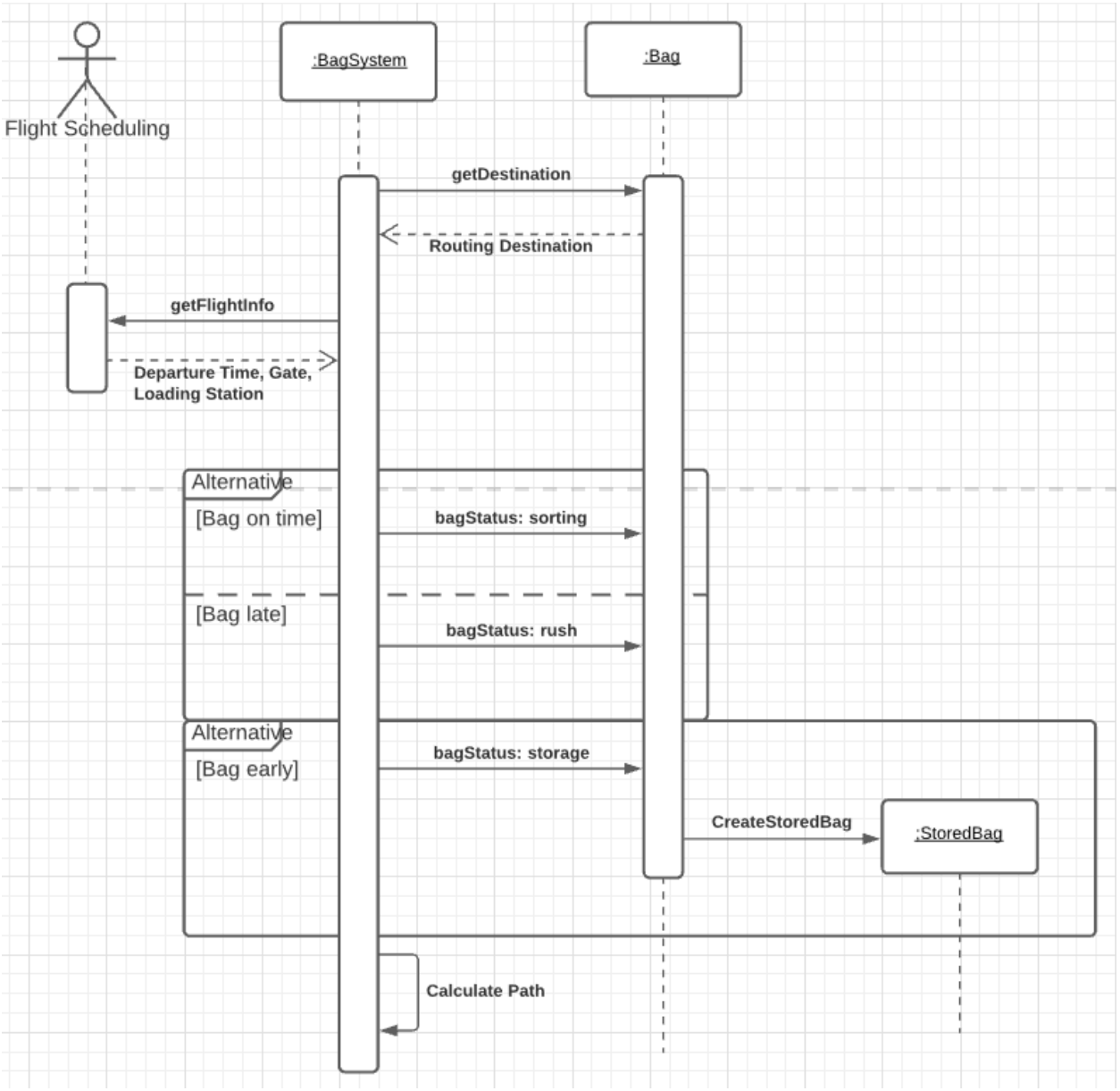
Use Case 3: Unloading

This is the other process that creates Bag objects. Bags are entered into the system during unloading, then routed for either transfer or pickup.



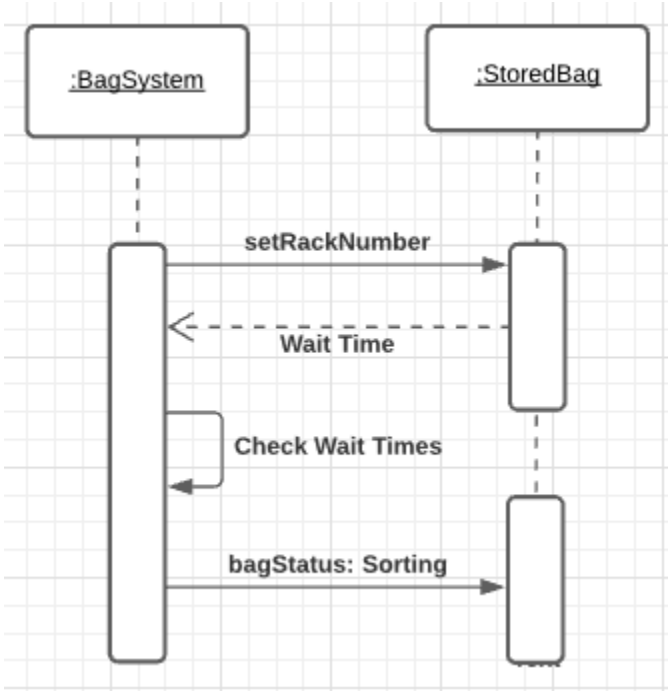
Use Case 4: Presort

Presort routes bags based on the time until departure. Late bags are “rush”, and go direct to the aircraft. Early bags go into storage to prevent congestion. On time bags are sorted and sent to a loading station.



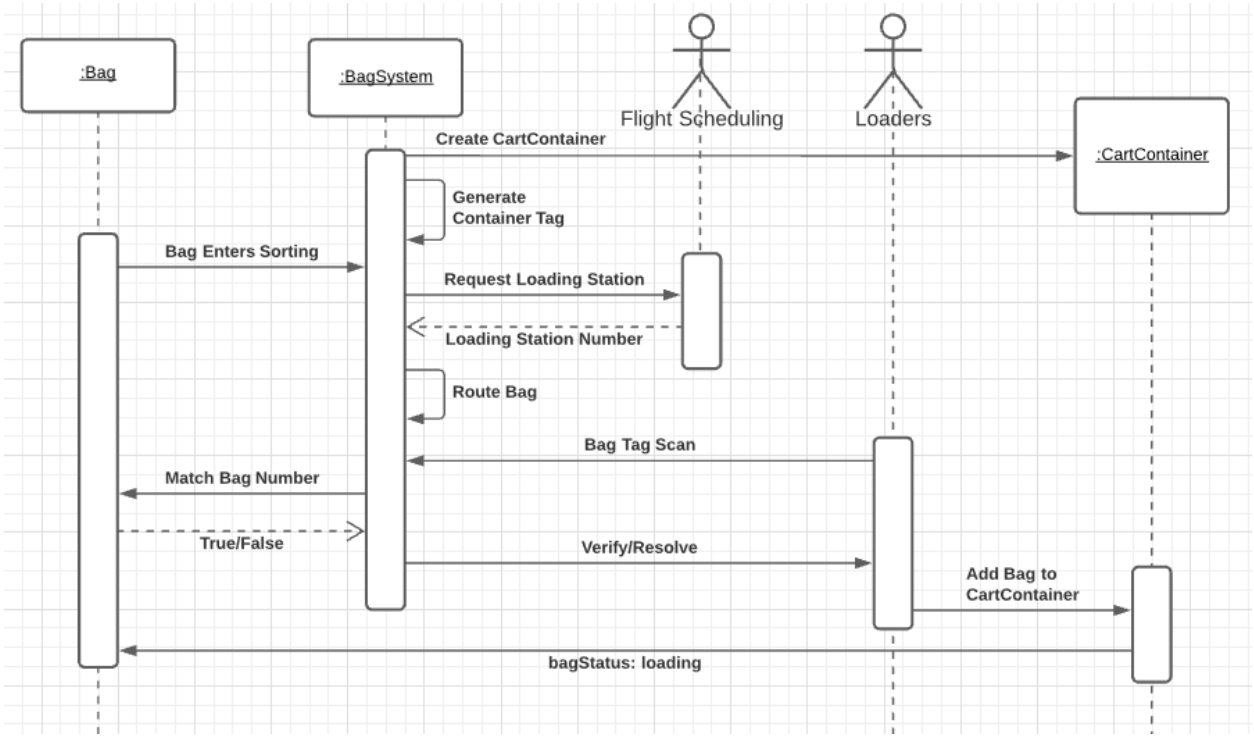
Use Case 5: Bagstore

Early bags are stored in cubicles by a robotic crane. The controller for this is basically a loop that checks to see which bag waits have expired at a preset interval.



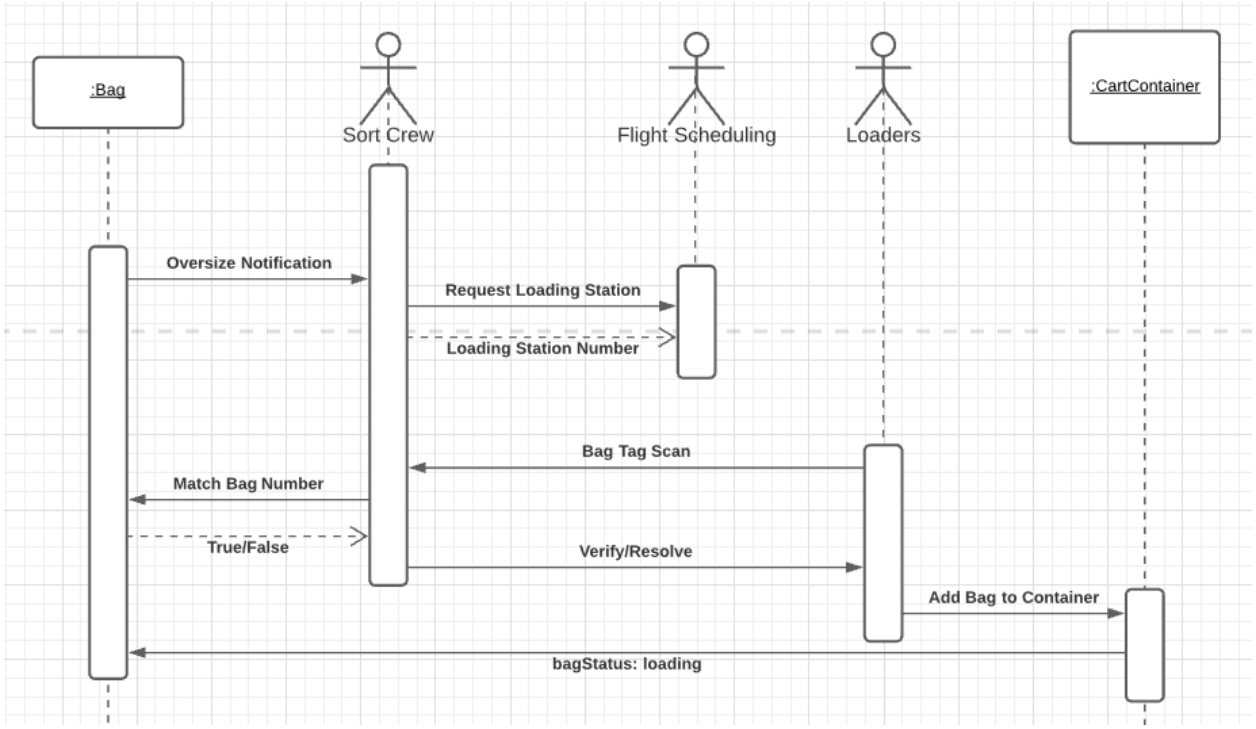
Use Case 6: Sorting

This subsystem routes bags to loading stations that are associated with cart containers. Loaders verify that the physical bag matches the Bag object, then the Bag object is added to a list in the CartContainer object.



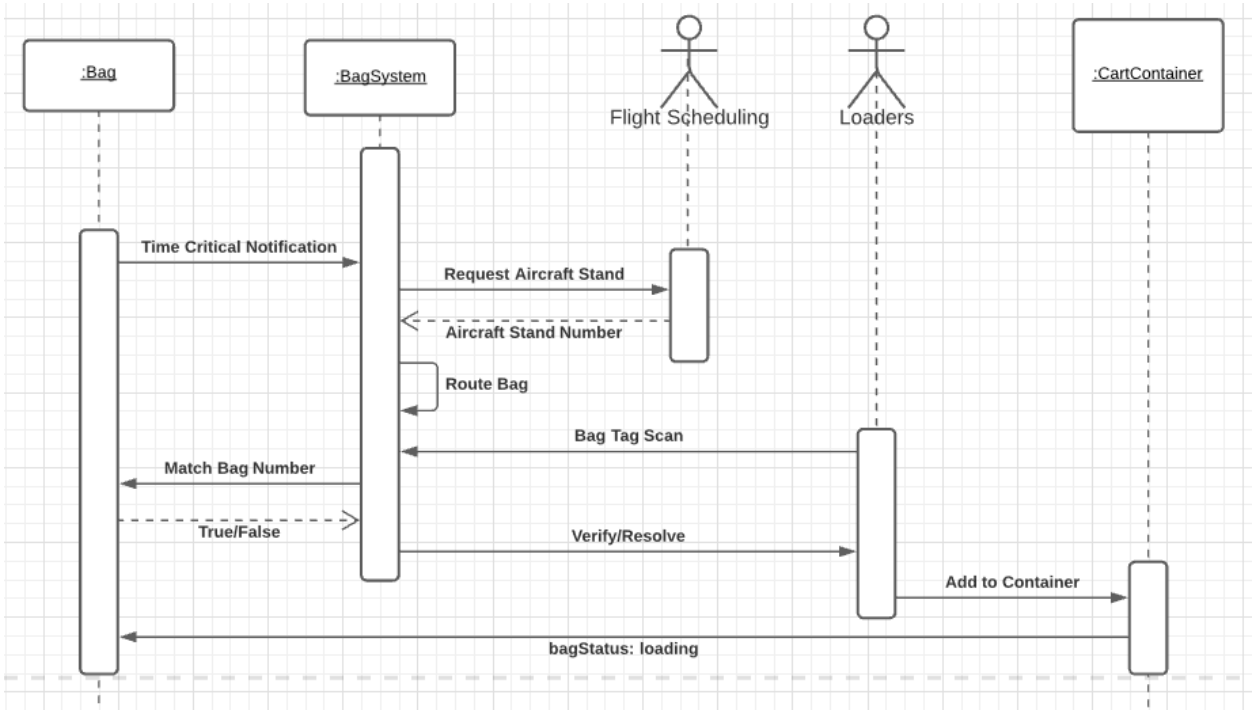
Use Case 7: Oversized Bag

This is much like UC6, but the CartContainer object is already created and the bag and the Bag object arrive from a different route.



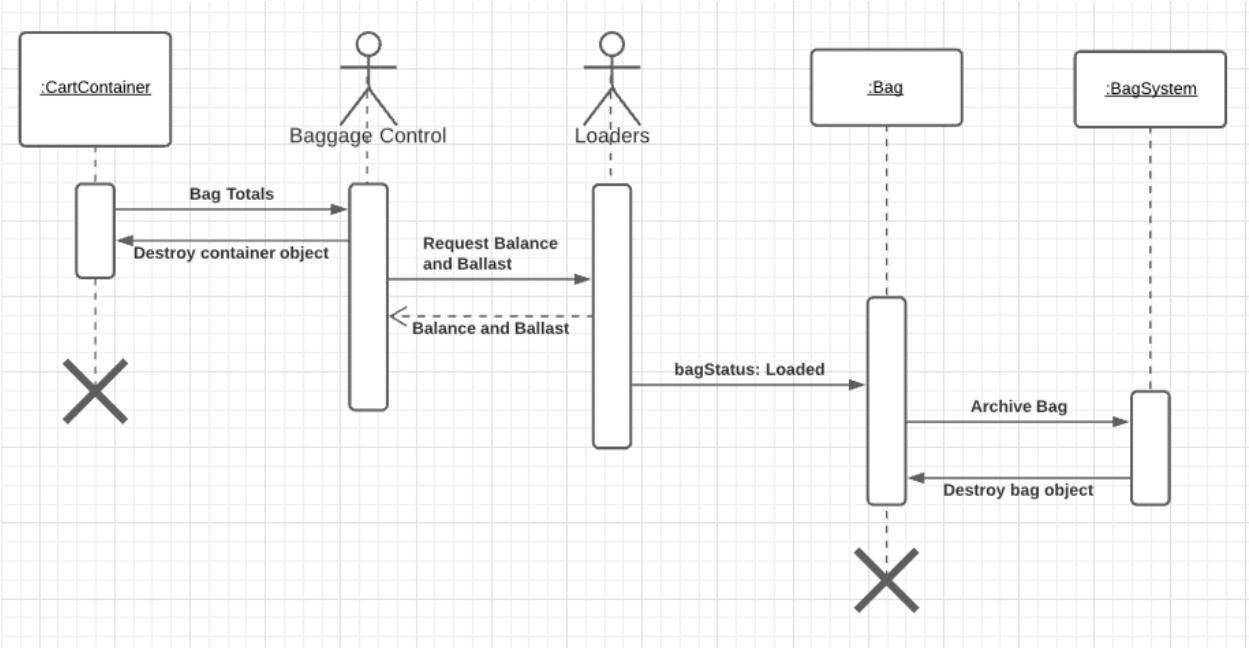
Use Case 8: Rush

The physical bag is transported directly to the gate by high-speed rail. The Bag object is updated and added to the CartContainer object.



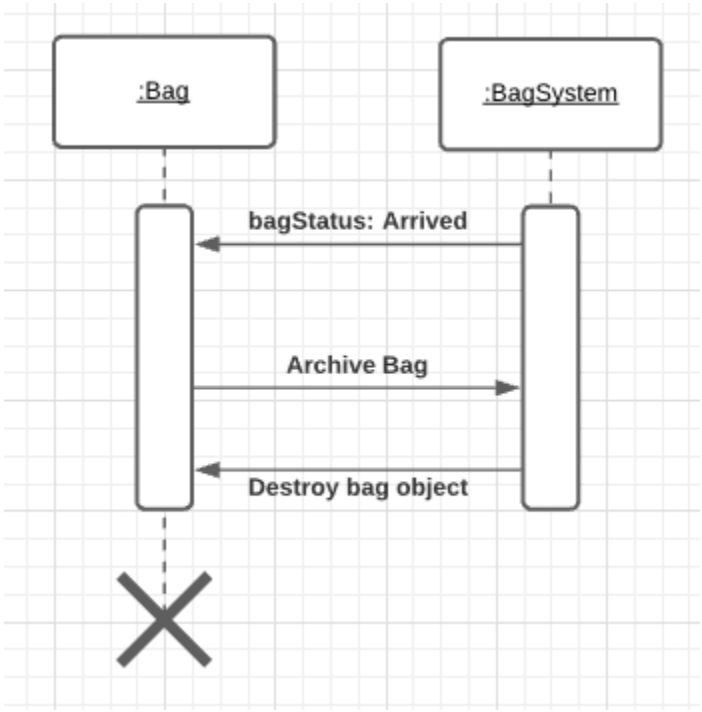
Use Case 9: Loading

Baggage control uses the Bag list in each of the aircraft's CartContainer objects to calculate how to arrange the bags, and how many sandbags are needed, in order to properly balance the aircraft. The data from the loaded Bag objects is archived for future reference. The Bag and CartContainer objects have completed their journeys and are discarded.



Use Case 10: Pickup

Some Bag objects created by the Unloading process have reached their final destination. These bags are routed to a pickup carousel, at which point their data is archived and the object is discarded.

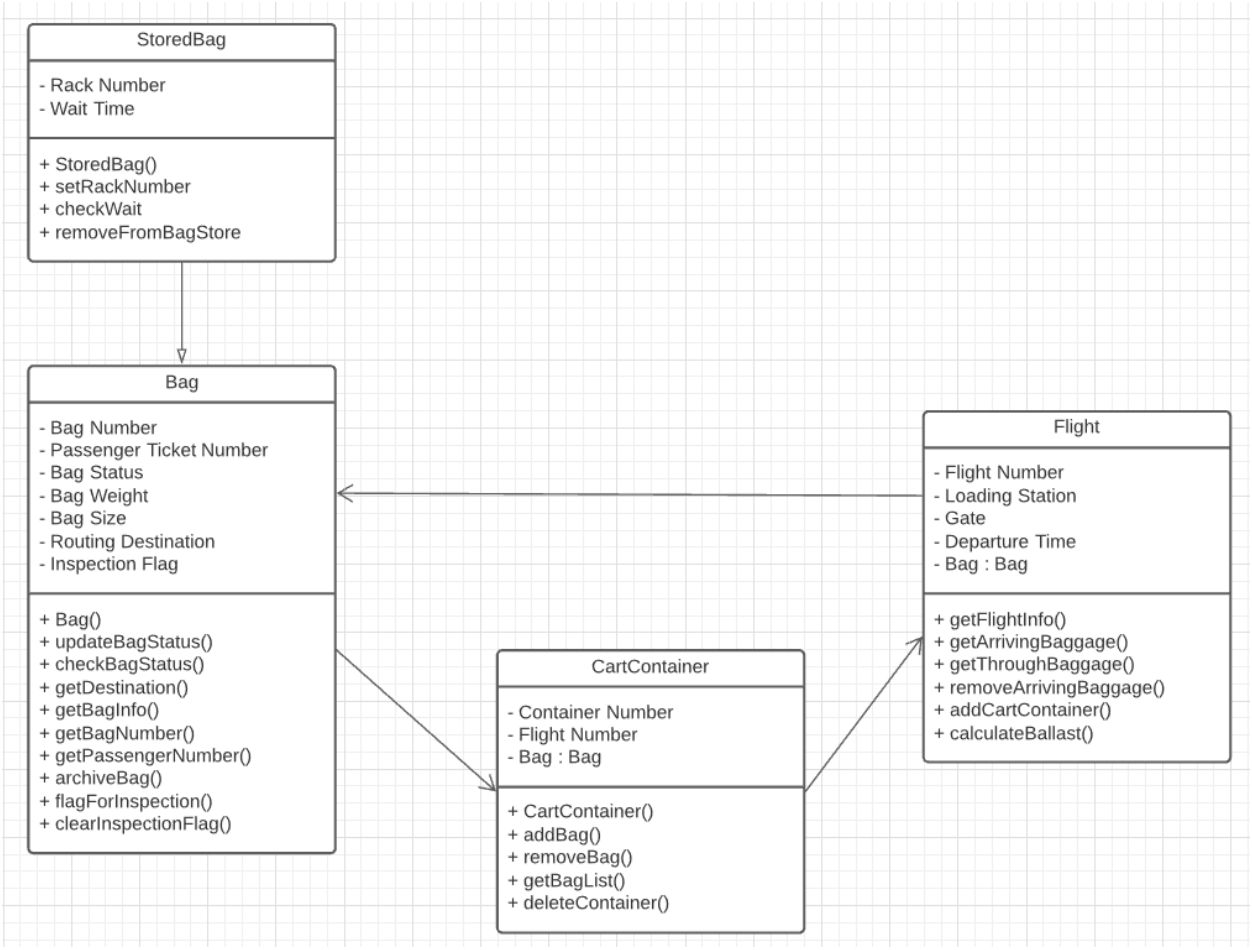


Class Diagram

This system is built around the Bag object. It has an ID number, the ID number of its associated passenger, the bag's intended destination (for troubleshooting purposes), a security flag and physical characteristics used for baggage arrangement. It has a constructor and methods for viewing and changing properties. StoredBag is an inherited class that contains properties and methods that are only useful to bags in the Bagstore subsystem.

CartContainer is mostly a list of Bag objects. CartContainers are associated with a Flight that will receive the Bag lists of many CartContainers. Once this list is complete, the Bag Size and Bag Weight properties of each Bag object will be used to calculate cargo distribution and any ballast.

The associations for a circle because Flight objects are used to create some Bag objects, while Bag objects from elsewhere are routed to Flights via CartContainer objects.



Lessons Learned

I learned how to create system diagrams.. This has already been useful for my capstone project, and will probably come in handy again at some point.

If returned to the beginning of the semester, I would probably choose to create one of the suggested systems instead. This was interesting, but over-ambitious. There are quite a few complex processes that have been glossed over as “bagsystem” or the like. Designing a real baggage system controller is definitely a team project.

If this project wasn’t schoolwork, I would treat the diagrams as a circular process. Each successive diagram presents new considerations that frequently require a rethink of previous diagrams.