CSC675-775, Section 3

Flight Monitor

Team 7

Piero Calenzani
Jimmy Kwan
Chinmay Purav
Jason Wei
Navneet Singh
Pratik Singh

Table of Contents

1. Executive Summary

The Airport Flight Monitor is an application that will serve airport organizations with keeping track of flights and passengers. The application helps management with airport employees, terminal gates, and baggage claim as well as keeping track of what any luggage the passengers check-in. This project will be designed with the primary user being airport managers and other employees. Additional implementation can include the concurrent status of given flights for the airport to display to passengers. The project's database will include only the relevant information to an airport with the primary focus being flight scheduling.

This application is designed with smaller airports in mind and will observe all gates from the given airport and provide the necessary information to keep track of incoming and departing flights. Since the airport itself is being kept in mind as the user, terminal gates and baggage claim are important information to display with each flight in the project's scope. Ticketing and pilots will be monitored through this application for each participating flight company in the airport.

In addition, summarized flight info can be readily displayed for passengers to view through any airport screens the airport wishes to hook up. This usability extends to individually checked luggage and its corresponding baggage claim carousel and allows potential passenger users as well as airport personnel to view baggage info for corresponding flights without having access to unnecessary information.

2. Use Cases

Use Case #1: Front desk employees

The visitor is picking up a friend from his flight but needs to know exactly where to go. So the visitor asked the front desk for the flight information of his friend. So the front desk employee uses the application to check the schedule and corresponding gate of the flight.

Use Case #2: Airport Luggage Employee

When a flight has arrived, luggage employees need to know which belt the bags have to be distributed to. The employee can open our application and know which baggage claim has been assigned for this particular flight. One passenger get off the flight and he is wondering where his luggage is. So he asked the employee for help. Since employee already checked the baggage claim, he points the baggage claim to the passenger.

Use Case #3: Flight Delay

A passenger's flight may get delayed and assigns to a different gate. So one employee updated flight departure times, assigned gate and flight duration by using the application. Therefore, the employee broadcasts the updated flight information.

Use Case #4: Flight Traffic Controller

A pilot needs to know which gate is assigned to his current flight and contacts the Flight Traffic Controller to know the designated gate. The traffic controller uses this application to know the available gates and coordinate with the pilot.

3. Entities Glossary

User Case: Visitor
Entities - flights (schedule), company, terminal gate, and visitor

Flights - **flights** are the main entity, which is the main transportation at the airport that take passengers to and from locations. Each flight is through a **company.**
They have a certain schedule that correspond to certain **terminal gates**. **Visitors** can check these data.

User Case: Airport Luggage Employee
Entities - baggage_employee,flight, baggage claim (baggage belt), baggage
baggage - is its own entity, baggage is product that is carried
Baggage claim - is its own entity, is a where visitors pick up their baggage

Baggage_employee - A **baggage employee**'s duty include : to send **baggage** to the right **flight** at the right time, to transfer luggage from one flight to the correct **baggage belt** that leads to customer **Baggage claim**.

User Case: Passenger (Arrivals)
Entities - flight, baggage_employee, passager, baggage claim

Passengers arrive from flights and proceed to baggage-employees to get info about the baggage claim corresponding to their flight.

User Case: Passenger (Layover)
Entities - passenger, flight(schedule , add flight ID), terminal gate

Passengers are assigned **flights** but they can be delayed or changed. The change will include schedule updates and **terminal gate** info for updated flight. Passenger will then be informed of the new update.

User Case: Flight Traffic Controller

Entities - Pilots, terminal gate, flight traffic control, flight

Pilot operates airplanes. Once they land, they contact the **flight traffic control** to know what **terminal gate** they need to park at.

4. Business Rules

1. Flight must be flown by **at least one** pilot, on a **single** airplane.
2. Flights must carry **at least one** passenger.
3. Airports must have **multiple** gates which can have **at most one** airplane.
4. Airports must have **at least one** employee.
5. Passengers can check-in **at most one** luggage per flight.
6. Passengers must order **at least one** ticket.
7. Tickets are sold by **one** company.
8. Pilots are employed by **at least one** company.
9. Airplanes are owned by **one** company.
10. Luggage can be picked up from **one** baggage claim, within the airport.

5. Initial List of Entities and Relationships

Entity: Airport (strong)
Relations: supervises, employs, contain
Attributes:

- airport_id (key)
- supervises (aggregation)
- location
- name

Entity: Airplane (strong)
Relations: flown_by, owned_by, used, docked_at
Attributes:

- aid (key)
- name
- cid (FK)
- tid (FK)

Entity: Flight (strong)
Relations: uses, piloted_by, requires, travels, transports
Attributes:

- fid (key)
- aid (FK)
- schedule
- tid (FK)

Entity: Passenger (strong)
Relations: buys, travels_on, carries
Attributes:

- passenger_id (key)
- name
- fid (FK)

Entity: Ticket (weak)
Relations: required_for, bought_by, sold_by
Attributes:

- tid (key)
- schedule (FK)
- fid (FK)
- passenger_id (FK)

Entity: Company (strong)
Relations: owns, sells, hires
Attributes:

- cid (key)
- name

Entity: TerminalGate (strong)
Relations: contained_by, docks
Attributes:

- tgid (key)
- name
- airport_id (FK)

Entity: Luggage (weak)
Relations: carried_by, held_at, transported_by
Attributes:

- passenger_id (FK)
- bcid (FK)
- fid (FK)

Entity: BaggageClaim (weak)
Relations: contained_by, holds
Attributes:

- bcid (key)
- name
- airport_id (FK)

Entity: Pilot (strong)
Relations: hired_by, flies, pilots
Attributes:

- pilot_id (key)
- fid (FK)
- name

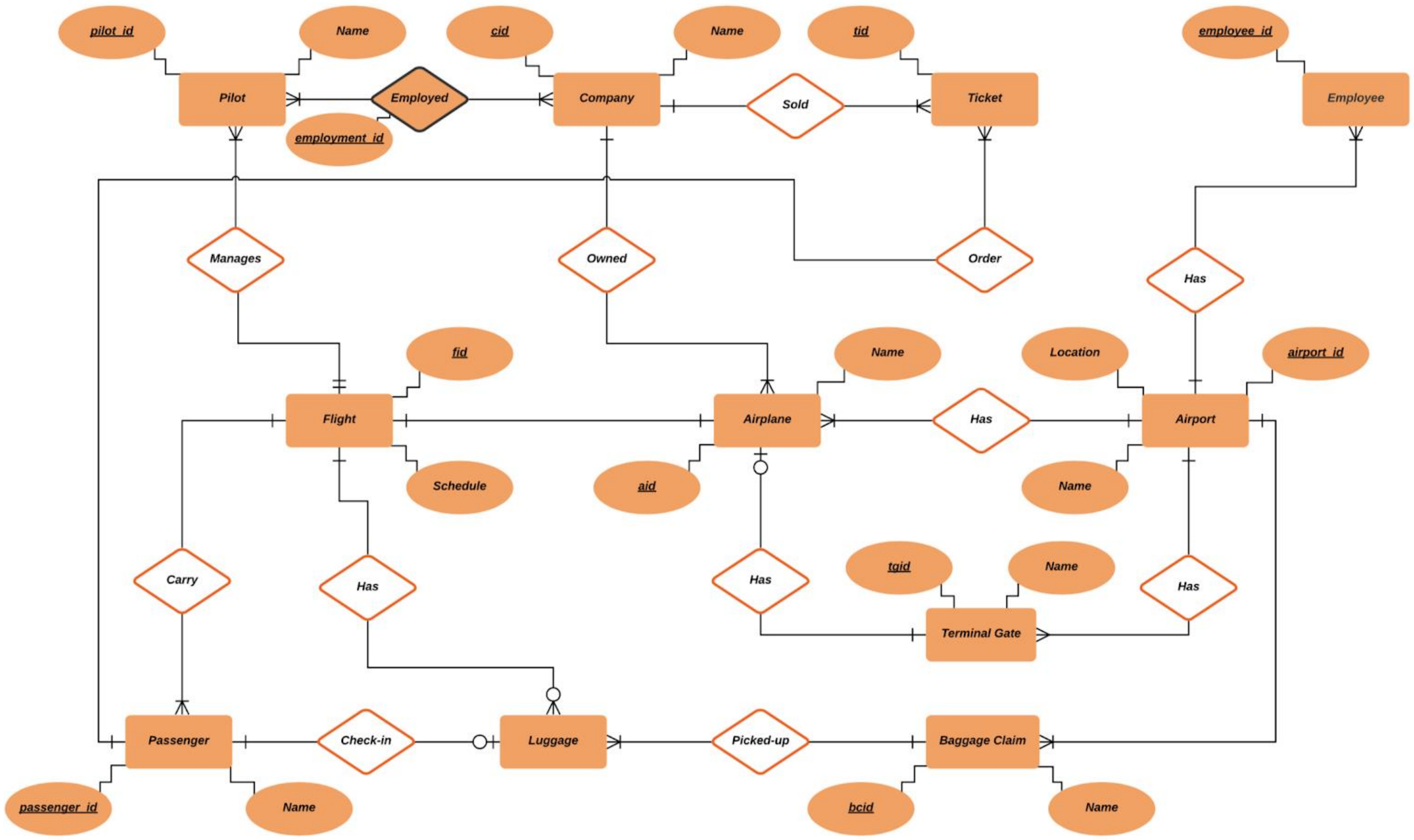
Entity: Employed (weak)
Relations: company_employs (pilot), employed_by (the company)
Attributes:

- employment_id (key)
- pilot_id (FK)
- cid (FK)

Entity: Employee (strong)
Relations: employed_by
Attributes:

- employee_id (key)
- airport_id (FK)

6. Entity Relationship Diagram



7. ERDs Test

Note: Missing tests as of 10/7: Diagram was completed late

Business Rule #	Entity 1	Relationship	Entity 2	Relationship type	Pass/Fail	Modify
1	Example	is_a	Template	1 to 1	Fail	Reason for failed.

8. Initial List of Non-functional Requirements

1. Application shall be optimized for desktop/phone displays.
2. Application should be able to handle 50+ users at any time.
3. Non-relevant information should be safely secured from public view.
4. Application shall be reliable after release.
5. Application needs to have quick performance time.
6. Application must have good operability for average adult worker.
7. Scalability must be able to account for any given airport size.
8. Capacity must be able to handle the amount of passenger and flight traffic on any day.
9. Application must have good maintainability post-release.
10. Data shall be stored on MySQL database on deployment server.

9. Team Member Contribution

Work done by each team member. Enumerated on a scale from 1-10.

1. Piero (Team Lead) - 9
    - Coordinated work done by each member
    - Wrote executive summary
    - Wrote nonfunctional requirements
    - Helped write each use case
    - Helped write and clarify business rules
2. Jason - 9
    - Finished parts quickly
    - Helped write and edit use cases
    - Wrote initial business rules and helped edit them
3. Jimmy - 6
    - Helped write use cases
    - Wrote and revised list of entities and attributes
4. Chinmay - 7
    - Late on first ERD iteration
    - Helped write use cases
    - Created and designed ERD
5. Navneet - 3
    - Late on all contribution
    - Helped with entities glossary
    - Absent in slack channel
    - Absent from class and meeting
6. Pratik - 5
    - Provided team with sample project features
    - Helped revise executive summary
    - Completed ERD tests (missing)