Drone Air Traffic Control System

Software to Coordinate Automated Drones, Safely

Functional Specification

INSTITUTE of TECHNOLOGY CARLOW

Institiúid Teicneolaíochta Cheatharlach

Ronan Donohue - c00208501

Introduction:

The proposed application is titled ‘Drone Air Traffic Control System’. It is intended primarily as a system that will enable drone hobbyists to automate their drone flight paths and ensure said drones arrive safely at their designated arrival points. The two main keywords for this application are **coordination** and **control**. Each drone connected to the application should be able to send data about itself and also receive commands issued by the application. Drone flight plans will need to be scheduled by the user and these proposed plans scrutinised by the system prior to take-off in order to minimize potential accidents. Once airborne, drones will need to be controlled and monitored. This will be handled by the software. If it looks like there could be a mid-air collision, drone control will be passed to an automated collision-avoidance handler to resolve the collision before it happens. Once resolved, the drones will resume their routes. The application should also visualise drone routes in real time on the users device, along with simulating drone routes using a virtual drone (also represented visually). Drone flight path information along with on-board drone recording data will also need to be stored to comply with Irish government legislation.

Core Functionality:

1. Drone Communication.

It is necessary that the application be capable of receiving drone data. Likewise, the application should be capable of sending data such as a flight plan to the drone. In order to facilitate this, each drone will need to be identifiable to the system. Once ID'd, the drone will be paired with a drone controller object. This object will issue commands to the drone and receive drone telemetry data once paired. In the eyes of the application, this object is an interface between the drone system and the application.

2. Air Traffic Control Algorithm.

It is important that the user be able to create flight plans that their drones will then fly. Upon entering a proposed route, the scheduler object within the software application will handle this new route information. The scheduler object will take into account the drone flights that will be running during this proposed route. It will be able to access this information from a list of accepted flight plans. The scheduler will ensure that each drone route keeps a given drone at a minimum distance from other airborne drones. Once a route has been established, each drone will need pre-flight checks performed via its drone controller object to make sure that the drone is in good condition for flying. After all that, the drone controller will receive the flight plan from the scheduler.

Once airborne, the drone will be executing its assigned flight plan. Monitoring and event handling will fall to the Drone Controller object.

3. Collision Avoidance Algorithm.

Drones will be visualised to the end user as marks on a 2 - dimensional plane. Each mark will have a green shell around it and a red shell within that shell. As drones move in real time on the visualisation window, should a mark's green shell overlap with anothers, a Collision Avoidance Handler will be invoked. The role of the Collision Avoidance Handler is to determine whether or not a crash between two airborne drones is imminent. It will analyze each drones speed and heading to ascertain this. If a crash looks likely, the Handler will pause the drones executing flight plan, store it temporarily, work out a strategy to avoid collision, send the newly devised coordinates to the drones for them to execute before allowing each drone to resume its original flight plan. The flight plan will have been altered to reflect the Handlers recommended solution and pick up as closely as possible to where it left off.

4.Scheduling Routes.

The user will be able to create custom routes for a drone to fly. Alternatively, they can select from a list of successfully completed routes and edit the route settings i.e last run job but alter the start time etc. Each new route that is proposed is passed to the scheduler. The scheduler will perform various checks and determine the best possible parameters for the proposed flight. For example, the user may want to schedule a similar flight to one that is about to run. The scheduler will look at both flight plans and determine there is a possibility of a collision should both flights be running simultaneously. The scheduler alters some aspects of the proposed route to ensure that the two drones are kept at sufficient distance from one another.

The scheduler is also responsible for passing route information or flight plans to each distinct drone controller prior to take off. Once handled by a drone controller, the scheduled flight plan is removed from the list of upcoming flights and will be represented on the active visualisation of drone routes.

5. Route Visualisation.

As each drone is flying, they will be represented to the user as marks on a 2 - dimensional plane. The x and y coordinates of the mark will correspond to the telemetry data being received from the drone. There will be a green shell around the mark, with a smaller red shell within the green shell. These are to aid the application in avoiding a possible crash once airborne. The user will be able to click on the mark and be presented with the option of viewing either the flight plan information or the drone camera feed. If the user opts for the flight plan information, the flight plan currently being executed by the drone will be displayed. If the user opts for the camera view, a new window displaying the drones camera feed in real time will be displayed. For simulated drones, this will be greyed out.

6. Data Storage.

The information from every successful drone flight will be recorded in a database. This will happen without fail after every flight, real or simulated. Previous flight information can be pulled from the database if requested, and flights that have run before can be rescheduled and used again.

Target Market:

The intended market for this software is primarily drone enthusiasts or companies who wish to exercise more control and coordination from their fleet of drones. For example;

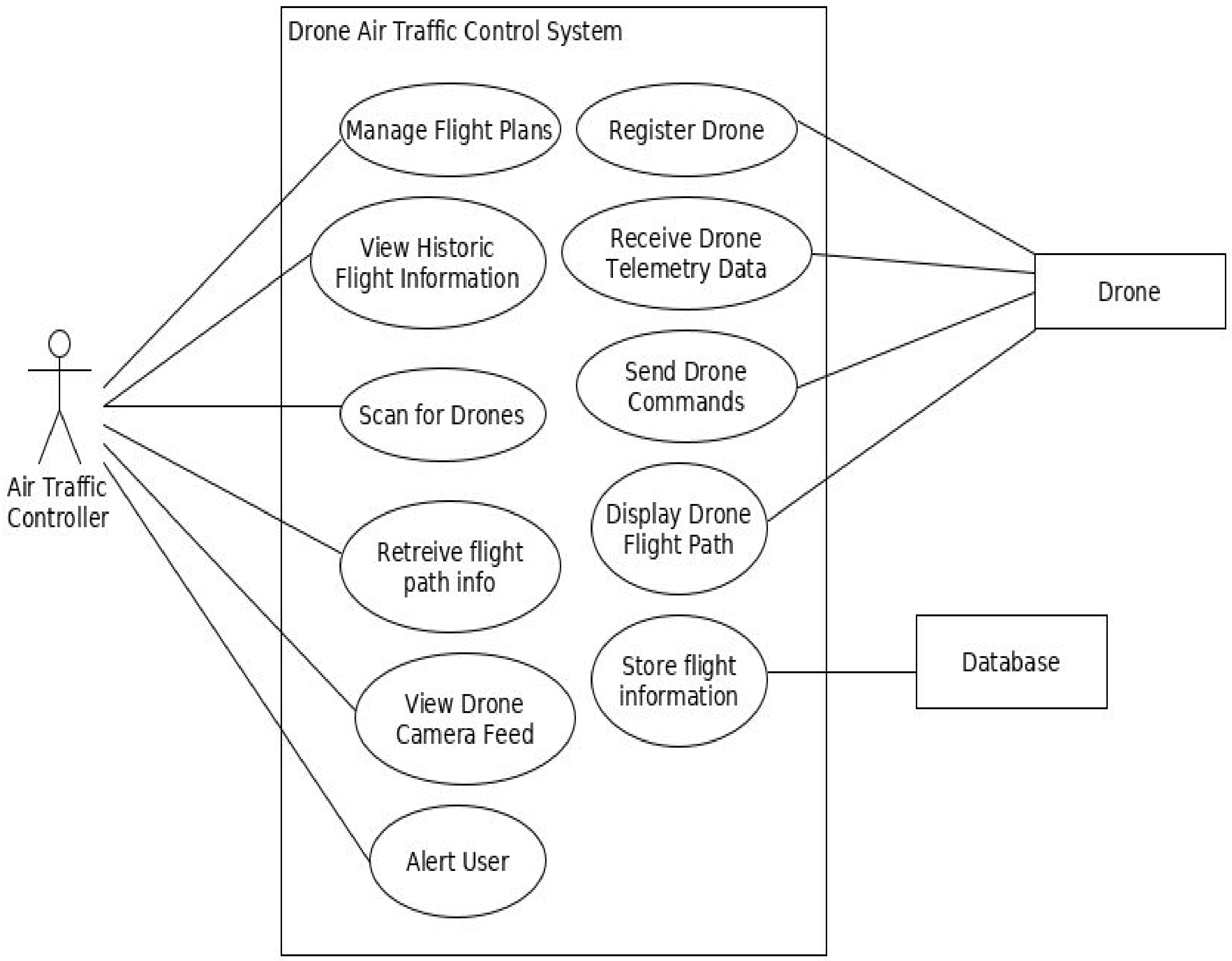
Businesses operating in the commercial sector can avail of drone automation to deliver food, post or other goods to remote or inaccessible locations, or to elderly patrons who cannot make the journey unassisted.

Nature photographers wishing to take full advantage of the route scheduling feature of the application in order to photograph or video more scenery from new angles.

Automated drone security. A set route for drones to fly and monitor is a very useful feature for owners of larger-than-normal sites. The drones can move faster over terrain than a person and can cover more ground.

Smaller construction firms can also avail of automated drone routes in surveying the land. It would also be useful in assessing damage after natural events such as flooding or inspecting general wear-and-tear of hard-to-reach areas.

Context Diagram



Use Cases

Use Case UC1: Register Drone

Primary Actor: Drone.
Stakeholders & Interests:
Air Traffic Controller - wants their drone to be connected to the application.
Preconditions:
Drone must be powered on and emitting a signal.
Success Guarantee:
Drone is registered with the system, Drone Controller object instantiated.
Main Success Scenario:

1. The hobbyist powers on their drone.
2. The drone emits a signal.
3. The user scans for any active drones (see UC2).
4. The system picks up the drone signal.
5. A connection is established.
6. The drone is registered with the system.

Alternate Flows:

4a. The system does not pick up the drone signal.
 4b. The system or the drone are reset.
 4c. Loop to step 1.
6a. The drone is already registered with the system.

Use Case UC2: Scan for Drones

Primary Actor: Air Traffic Controller.
Stakeholders & Interests:
Air Traffic Controller - wishes to pair their drone to the application.
Preconditions:
Drone must be powered on and emitting a signal.
Success Guarantee:
The scan finds the drone signal.
Main Success Scenario:

1. The user wishes to pair an active drone with the system.
2. The user selects ‘Scan For Drones’.
3. The system scans for any active drone signals, including previously paired and unpaired drones.
4. The system presents the user with a list of drones discovered.
5. The user can select whichever drone they wish to pair.

Alternate Flows:

3a. The system cannot find any unpaired drones.
   3b. The drone may need to be reset or the scan run again.

Use Case UC3: Manage Flight Plans

Primary Actor: Air Traffic Controller.
Stakeholders & Interests:
Air Traffic Controller - wishes to create, view, edit or remove flight plans from the system.
Preconditions:
At least one active drone that is currently unassigned a flight plan must be available.
Success Guarantee:
A flight plan has been generated, modified, viewed or removed.
Main Success Scenario:

1. The user wishes to examine the existing flight plans scheduled to run.
2. The user selects 'Manage Flight Plans'
3. The system presents the user with a list of currently scheduled flight plans with options to create a new flight plan, edit an existing flight plan or delete a flight plan.
4. The user selects a flight plan for viewing.

Alternate Flows:

4a. The user wishes to create a flight plan
  4b. The user is presented with the option to fill in various details of a proposed flight plan, incl. which drone the plan is for etc
  4c. The flight plan is passed to the scheduler.
  4d. The scheduler determines that the flight plan is OK.
  4e. The flight plan is appended to the list of upcoming flights.
5a. The user wishes to edit the flight plan they are viewing.
  5b. The user is presented with a screen displaying the the information about the flight plan.
  5c. The user decides to update a certain aspect of the flight plan eg scheduled start-time.
  5d. The user selects 'Save'.
  5e. The new flight plans are stored to the database and the scheduled flight plans list is updated.
5a1. The user wishes to delete the flight they are viewing.
  5b1. The user selects the check box beside the flight.
  5c1. The user clicks the 'Delete Flight Plan' button.
  5d1. The flight plan is removed from the list of upcoming flights but a record is kept in the database.
  5e1. The drone associated with the now deleted flight plan is free for reassignment.
3a. The system fails to present the flight plans.
  3b. An error message is displayed to that effect.

Use Case UC4: Receive Drone Telemetry Data

Primary Actors: Drone.
Stakeholders & Interests:
Air Traffic Controller - needs a working connection between drone and application.
Preconditions:
Drone must be emitting a signal and registered with the system.
Success Guarantee:
The drone will respond to a simple systems check command from the application.
Main Success Scenario:

1. The drone is registered with the system.
2. A Drone Controller object has been paired to the drone.
3. The Drone Controller object issues a system check to verify the drone is capable of receiving commands.
4. The drone responds with all the data necessary for the system check to pass.
5. The output of the system check is written to a log file.
6. Drone Controller requests Drone telemetry data.
7. Drone sends telemetry data.

Alternate Flows:

4a. The drone does not pass the system check.
  4b. The user is notified of this.
  4c. The results of the system check are displayed to user.
  4d. The results are written to a log file.
7a. Drone does not send telemetry data.
  7b. Drone connection lost, drone self-instructs to return home.

Use Case UC5: Retrieve Flight Path Information

Primary Actors: Air Traffic Controller.
Stakeholders & Interests:
Air Traffic Controller - wishes to view the attached flight information for an active drone.
Preconditions:
Drone has been registered, Drone has passed System Check, Drone has been assigned a valid flight path, Drone is currently active and flying.
Success Guarantee:
The user can see the flight path currently assigned to an active drone.
Main Success Scenario:

1. The user selects a drone from the visualisation of active drone routes.
2. The user selects 'View Flight Path Information'
3. The flight path associated with the drone is displayed alongside the visualisation.
4. The user clicks 'Close' button

Alternate Flows:

4a. The user does not click ‘Close’
    4b. The flight path info remains open until the drone has completed its run.

Use Case UC6: View Drone Camera Feed

Primary Actors: Air Traffic Controller
Stakeholders & Interests:
Air Traffic Controller - wants to view their drones camera feed in real time.
Preconditions:
Drone has been registered, Drone has passed System Check, Drone has been assigned a valid flight path, Drone is currently active and flying, Drone has a camera.
Success Guarantee:
User can see the drones camera feed in real time on the screen of their device.
Main Success Scenario:

1. The user selects a drone from the visualisation of active drone routes.
2. The user clicks the 'View Drone Camera Feed' button.
3. The camera feed is rendered in a new window.
4. The user clicks the Close icon in the top right corner of the camera feed window.

Alternate Flows:

1a. The user selects a simulated drone from the visualisation of active drones.
  1b. The 'View Drone Camera Feed' button is greyed out and will not respond to clicks.
3a. The camera feed is not rendered.
  3b. An error message is displayed to the user.

Use Case UC7: Store Flight Information

Primary Actors: Air Traffic Controller, Database
Stakeholders & Interests:
Air Traffic Controller - must keep a record of completed flights.
Preconditions:
Drone has been registered, Drone has passed System Check, Drone has been assigned a valid flight path, Drone has completed the assigned flight.
Success Guarantee:
Successful flight information has been written to the database.
Main Success Scenario:

1. The drone has completed its assigned flight path.
2. The system writes the details of the flight path, including associated drone data into a database.

Use Case UC8: View Historic Flight Information

Primary Actors:
Air Traffic Controller, Database
Stakeholders & Interests:
Air Traffic Controller - Wishes to view previously run flights.
Preconditions:
At least one flight has been completed by a drone or simulated drone.
Success Guarantee:
The user is presented with a list of completed flights.
Main Success Guarantee:

1. The user wishes to view completed drone flights.
2. The user selects the 'View Completed Flights' option.
3. A query is sent to the database to fetch the most recent completed flight information.
4. The results of the query are displayed to the user in a list.

Use Case UC9: Send Drone Commands

Primary Actors:
Drone
Stakeholders & Interests:
Air Traffic Controller - needs the drone to respond to timely event handlers such as collision avoidance and obey proposed flight plans.
Preconditions:
Drone has been registered, Drone has passed System Check, Drone has been assigned a valid flight path, Drone is about to undergo a scheduled flight.
Success Guarantee:
The drone responds to commands issued by the application.
Main Success Guarantee:

1. The drone is cleared for flying its assigned route.
2. The route information is passed to the Drone Controller object.
3. The route is loaded into the drones memory.
4. The drone begins executing the pre-programmed route.

Alternate Flows:

3a. The route is not loaded into the drones memory.
    3b. The drone does nothing.
4a. The drone does not execute the pre-programmed route.
    4b. The drone is reset and the route is reuploaded.

Use Case UC10: Display Drone Flight Path

Primary Actors:
Drone
Stakeholders & Interests:
Air Traffic Controller - needs to see visual layout of drone route mapped to the screen.
Preconditions:
Drone has been registered, Drone has passed System Check, Drone has been assigned a valid flight path, Drone is currently airborne.
Success Guarantee:
Drone movements in real time correspond visually with what is displayed to the user.
Main Success Scenario:

1. The drone has begun its route.
2. The drone controller object receives drone telemetry data.
3. The drone controller object is mapped to a mark.
4. This mark is displayed to the user on the 2-dimensional visualisation of active drone routes.
5. The drone moves in x, y, or z space.
6. The drone controller object receives new telemetry data.
7. The visualised mark's coordinates in the 2-dimensional display are updated accordingly.
8. Loop 5 - 7 until drone's route is complete.

Use Case UC11: Alert User

Primary Actors:
Air Traffic Controller, Drone
Stakeholders & Interests:
Air Traffic Controller - needs to be updated and kept aware of issues if and when they arise.
Preconditions:
Drone has been registered, Drone has passed System Check, Drone has been assigned a valid flight path, Drone is currently airborne and a crash is imminent.
Success Guarantee:
A "Potential Collision Alert" is displayed to the user.
Main Success Scenario:

1. Drone is airborne and is within another drones green shell, visually.
2. The system determines that a crash is likely.
3. An alert is displayed to the user.
4. The system takes steps to correct the drones behaviour and avert a crash.
5. The resulting changes are deployed.
6. The results of the correction are displayed to the user.

Metrics:

- The application should run on the chosen device (laptop, desktop, phone etc).
- The travel routes for drones are well defined.
- The application can receive drone data and ensure that the drone can be issued commands.
- An air traffic control algorithm is in place that can effectively schedule multiple flights. Drones will need to be kept out of each other's way. The algorithm should also ensure that hardware checks are performed on the drone.
- Active drone routes are visualised to the user in real time.
- The user can view either the drone flight plan, the drones camera feed, or both from the visualisation display for all active drones.
- The user can CRUD flight plans, and is given the option to rerun old flights and assign them to a new drone if desired.
- Simulated drones should be displayed alongside real drones. Simulated drones should be treated as real drones by the system with the only differences being that simulated drones will not be capable of displaying any camera feed footage and that in the eyes of the Collision Avoidance Handler, the simulated drone has the lowest priority in rerouting.
- The system should work with more than one drone.

References:

Drone Clipart #5. (2018). [image] Available at: https://melbournechapter.net/explore/drone-clipart-transparent/#gal_post_1130_drone-clipart-3.png [Accessed 20 Nov. 2018].