Communication in Drone Fleets

How a Simulator Can Help Resolve the Issue

Författare: **Nicolas Björkefors**
(980105)

HT21

Informatik med systemveteskaplig inriktning, kandidatkurs, 30hp

Ämne: *Informatik*

Handelshögskolan vid Örebro universitet

Handledare: Kai Wistrand

Examinator: Johan Pettersson

Abstract

Drone usage is becoming increasingly common in many industries and fields, and with a larger scale of adoption the next step forward is increasing capacity—going from single drones to fleets numbering in the hundreds or more. This step is a large one, fraught with issues that need resolving. One such issue is the matter of communication between the drones themselves and with a control centre; this research effort aims to contribute to the resolution of that issue by producing an artifact in the form of a drone simulator. The artifact is not a solution per se, but can be used to test theories and facilitate the testing and development of control systems which could more directly solve the issue. It is used here to demonstrate the feasibility of IoT connectivity and the MQTT protocol to address identified problems within the issue.

Table of Contents

Communication in Drone Fleets ....

Abstract ....

Background 1

Current State of Drone Usage 1

Expanding to Drone Fleets 2

Dewire 2

AWS and the Simulator 3

Purpose 3

Delimitations 4

Scope, Uniqueness, and Research Questions 4

Methods 5

Literature Review 5

Design Science Research 6

The DSR Process 7

DSR in Action 8

The first phase 9

The second phase 10

Results 14

Setup and User Input as a Tester 14

Web Console and Core Functionality 16

IoT Connectivity and MQTT messaging 19

Discussion and Conclusions 21

Discussing the results 21

Conclusions and Future Work 23

References 25

Background

Current State of Drone Usage

Technological advancements impact human society in many ways. One facet that has seen recent growth is the use of drones, or unmanned aerial vehicles (UAVs). Many industries were attracted by the possibility of employing drones, and indeed, many of those industries are already being transformed to some extent with nascent drone usage (Cohn et al., 2016). Examples include, but are not limited to: humanitarian efforts (Lichtman & Nair, 2015), search and rescue (Mishra et al., 2020), and agriculture (Ahirwar et al., 2019).

While drones are already being used to some extent in these fields, most applications only utilise a single drone, others are few in number, and in both cases are typically controlled individually by a pilot on the ground (Vergouw et al. 2016). Kinaneva et al. (2019) proposed a system using drones and artificial intelligence (AI) for the early detection of forest wildfires. The platform would consist of two drones flying at different altitudes, working in tandem to detect the outbreak of a fire. This pair of drones is supposed to cover the territory of the Rusenski Lom national park in Ruse, Bulgaria—an area of just over 34 square kilometres. For reference, Sweden has about 279,800 square kilometres of forest-covered land (Food and Agriculture Organization, 2020). While not every bit of that land might require surveillance, it is plain to see that to cover any significant portion of forested areas in a whole country, there is a need to dramatically increase the capacity of deployed drones.

According to Ahirwar et al. (2019), food production must see an increase of 70% to meet the consumption needs of the projected 2050 world population of 9 billion people. All the while, the agricultural field is fraught with problems like labour unavailability, extreme weather conditions, and inefficiency in fertiliser application. However, the application of drones is already proving a helpful addition in the field. Drones are used today in areas such as irrigation, crop and equipment monitoring, soil analysis, and bird control (Ahirwar et al., 2019; Veroustraete, 2015).

The extent to which the use of drones can be scaled, as evidenced by the examples provided, is practically limitless. Whether it be a network of drones patrolling hundreds of thousands of square kilometres of forested areas or monitoring farmland producing billions of tonnes of food annually, the advancement of drone technology will facilitate large-scale work in many different fields and industries.

Expanding to Drone Fleets

Expanding capacity to fleets of drones is the obvious next step, but it's easier said than done; Borzoo Bonakdarpour at Iowa State University (2019) explains that taking the step from a small number of individually-controlled drones to fleets of drones is not as simple as simply launching more of them. It would require an automated system to coordinate the drones and the tasks they are to perform, while still allowing drones to respond individually to events.

In their survey, Gupta et al. (2016) go into some detail about the important issues that must be resolved within UAV communication networks. The architectural design of the network would not be intuitive due to the fluid topology of drone networks. Their battery supply is limited, so drones might need to leave to recharge and thus be replaced by a new drone. Seamless transferring of user-session information between drones, or to a control station, is necessary, so that the replacement drone can continue the work without delay. In addition, in order for drones in a fleet to stay in formation, they must consistently exchange flight information (Yoo et al., 2015).

The nature of the fleet and the mobility of the drones depend on the application (Gupta et al., 2016). In providing communication, for example, over an area ravaged by an earthquake, the drones might hover in place. This in contrast to applications such as agriculture and forest surveillance, which require the drones to move quickly and span a large area.

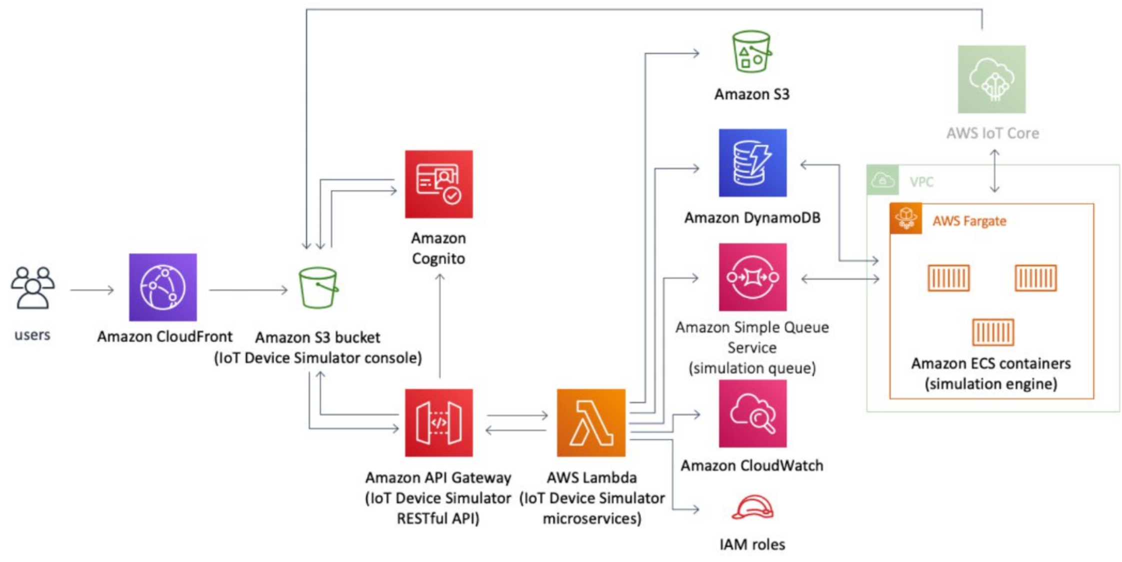
Dewire

The basis of the research effort was afforded by Dewire (part of Knightec), a consulting firm specialising in digital solutions using Amazon Web Services (AWS). They foresee clients contracting them for work on systems involving fleets of drones in the future, and are hence interested in working on an asset for building such systems. Dewire offers thesis work opportunities for students, and I was given the chance to work with them on a projected rooted in the aforementioned drone fleet system. At Dewire, I was paired with a supervisor, together with whom I have planned and structured the programming work to be done on a drone simulator, where the simulated drones would be used to test systems overseeing and controlling fleets of drones. The simulator is the artifact that will be evaluated in this paper, and any subsequent mentions of "the artifact" will refer to the simulator.

AWS and the Simulator

Figure 1

AWS IoT Device Simulator architecture overview



Because Dewire specialises in cloud solutions using AWS, their initial proposal was to repurpose or build upon Amazon's existing Internet of things (IoT) Device Simulator. It utilises several components of the AWS suite (see figure 1) and allows users to simulate up to 1,000 instances of user-defined widgets. However, the capability of the simulator in regards simulation itself is somewhat lacking, producing only random values within a range for a set number of attributes. In addition to the somewhat lacklustre simulation of user-created widgets, there exists a separate, ready-made section for simulation of self-driving cars, which is, relatively, more sophisticated. This automotive section of the simulator served as the substrate upon which the drone simulator was built, by making additions and modifications to the code.

Purpose

The purpose of the research effort was to contribute knowledge to problems that exist relating to deploying, overseeing, and controlling fleets of drones. The way this would be done was through developing a tool that could be used to test drone control systems, and which could also be used independently to demonstrate that these problems could be solved. This tool was formed to suit the organisational needs of Dewire, but can also be used to more generally test or prove solutions to drone-related problems. It would take the form of a low-fidelity (less life-like) simulator. The developed artifact will simulate one approach to solving the issue of communication, and will also be used test aspects of in-development control systems by feeding into the system the output of the simulator. The requirements of the simulator were conceived based on findings from the literature study as well as input from my supervisor at Dewire, with the perspective of the artifact as a tool for use in developing systems at Dewire.

Delimitations

While militaries have made extensive use of drones, for example in counterterrorism and counterinsurgency (Mahadevan, 2010), the work done in this project was done only with civil applications in mind. Certain aspects may very well be applicable to military usage as well, but that was never a consideration on our part. Ethical concerns of drone usage in these fields (such as privacy-related matters), practical but non-technical challenges such as the allocation of airspace, and legal aspects of drone use were not be considered either. These matters have been excluded to keep the scope of the research manageable.

Certain technical matters are infeasible to work on in our environment and situation, such as drone model compatibility within a given system, which Huang et al. (2019) address in their proposal, and also fall outside of the scope. We can only test functionality that can be abstracted. For instance, we could make sure different drone models are handled differently should that be desired, but the actual compatibility is something that would have to be solved by the user in a real-world application.

It quickly became apparent to our team of two that creating a high-fidelity simulator was all but impossible, due to lack of technical and mathematical know-how, resources, and time. The idea then was to develop a low-fidelity simulator, work at a higher level of abstraction, and test most functionality at a conceptual level with set, user-provided values in lieu of dynamic calculations. The simulation will be less lifelike, but the generated data can be just as valuable.

Scope, Uniqueness, and Research Questions

The growth potential of drone usage is great, and the amount of work that has to be done to realise this growth equally so. As such, in addition to ruling that certain aspects fall outside of the scope by nature, the scope had to be further narrowed by focusing on select issues within the domain. There are many aspects that must be considered and angles from which to approach; many proposed systems and solutions for various constellations of problems involving drones have already been proposed. Huang et al. (2019), for example, propose a platform that provides networking protocols and energy model incorporation. Coldova and

Olivares (2016) proposed a fleet-managing model for delivery, calculating fleet size based on load capacity, transport time, and production demand. They also incorporated alternation of drones to ensure efficient battery usage.

Going into the project, we knew that our particular configuration and situation was unique: trying to utilise the AWS simulator as a base for drone fleet simulations. In no prior research had this concept been attempted. In its design, the novelty lies within its instantiation using AWS, and in its application, it will demonstrate one possible approach to an as-of-yet unsolved issue in drone communication. The artifact in itself will also serve as an additional tool to help Dewire build and test drone fleet control systems in the future.

Hedström and Gudjonsson (2020) had worked with Dewire prior to this, exploring different cloud-based architecture alternatives for IoT-connected drones. They also wanted to investigate the feasibility of simulating drones for testing using AWS as part of their research, but concluded that it was too complex and time consuming for their project, instead suggesting future work in this area. Picking up the torch, this research effort started as a follow-up to their work. While keeping in mind my relative inexperience in the area, this being mostly a solo effort in execution, and the needs of Dewire in regards to the end product, we tried to single in on what issues we wanted to work on and formulated requirements and a work plan.

Ultimately, we decided that the core functionality we wanted to work on in our simulation was the communication between drones in the fleet and a system. Thus, the research questions this paper aims to answer are: “how can we use a simulator to test drone fleet control system functionality?” and “how can issues of communication requirements within drone fleets be solved?”

Methods

Literature Review

Before embarking on a new research endeavour, it is critical to see just how you can make a new application of a methodology or in another way contribute to the knowledge of a problem (Hart, 2018). This is difficult without systematic search, selection, and critical reading of a body of literature; only by knowing the intellectual context of a development can you know the implications of said development. Simply put, without a literature review, you will not acquire an understanding of a topic. Thus, a literature review was conducted to ascertain the current state of drone usage in various fields, and in what scenarios they could benefit from an increase in capacity. Similarly, a literature review was conducted in order to get a grasp on the obstacles that stand in the way of realising this increase.

I queried Google Scholar for articles related to drones and their usage. While non-exhaustive, following is a list of major keywords used in conjunction with each other, listed in their singular form: drone, fleet, use, control system, mock, simulation, AWS, Amazon Web Services, IoT, Internet of things, agriculture, forest fire, search and rescue, commercial, problem, challenge, military, legal, and design science (research).

After verifying that the results were published by a credible source, such as IEEE, which subjects every submitted article to peer-review, the articles' relevancy was judged first by reading the abstract, and then with more scrutiny in the full text in selected works.

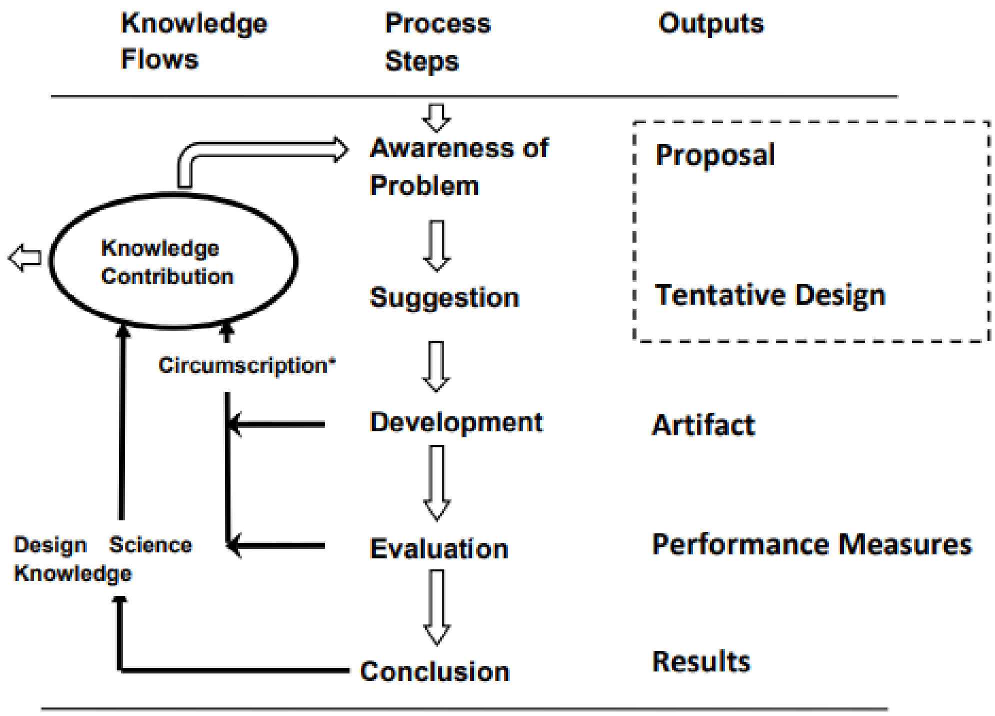
Design Science Research

The design science research (DSR) methodology was used as the framework for the results presented in this dissertation—the knowledge contributions of the developed artifact. Because the project that Dewire proposed involved the development and improvement of an artifact, the DSR methodology was an obvious fit; what I was undertaking is practically the definition of design science. According to Hevner et al. (2004), "design science … creates and evaluates IT artifacts intended to solve identified organizational problems" (p. 77), and Baskerville et al. (2018) posit that "in order to make a better world, the goal of DSR is to invent new artifacts where none exist and to improve existing artifacts to enhance organizational, group, and individual human productivities and effectiveness" (p. 362).

The DSR Process

Figure 2

Design Science Research Process Model (DSR Cycle)



* Circumscription is discovery of constraint knowledge about theories gained through detection and analysis of contradictions when things do not work according to theory (McCarthy, 1980)

The model shown in Figure 2 is developed by Vaishnavi et al. (2004), and describes the process followed by design science research. It is noted that the DSR process is fluid, and that in addition to the loop that is visualised in Figure 2, any of the phases may be spontaneously revisited—this is especially frequent in the early stages of a project.

Awareness of Problem. The awareness of an interesting (to a research community) problem, which may come from multiple sources. The researcher(s) considers criteria for evaluating the final product of the research effort, and the output of the phase is a proposal for a new research effort.

Suggestion. A creative step where the researcher(s) envision new functionality based on a novel configuration of either existing, or new and existing elements. A tentative design is drawn up, from which at least a prototype can begin to be constructed.

Development. The tentative design is further developed. The artifact can assume many different forms that range from design theories to concepts, models, processes, or instantiations (Gregor & Jones, 2007; March & Smith, 1995; Hevner et al., 2004, as cited in Vaishnavi et al., 2004). The technique for implementation will vary depending on the artifact, and can itself be very pedestrian; the novelty of the artifact lies in its design, not its construction.

Evaluation. After being constructed, the artifact is evaluated according to criteria. These criteria are frequently made explicit in the proposal—the product of the Awareness of Problem-phase. Deviations from expectations are noted and tentatively explained. The evaluation phase contains an analytic sub-phase in which expected behaviour and impact of the artifact are hypothesised. In contrast to positivist research, where analysis either confirms or contradicts a hypothesis, this step in design science research consolidates the results of the evaluation and any additional information gained during the construction and implementation of the artifact, and feeds them back to another round of Suggestion, as per the circumscription arrow in Figure 2. Hypotheses are rarely discarded, and are instead modified according to new observations. The results of the Evaluation often suggest a new or modified design.

Conclusion. The end of a research cycle, or the research effort as a whole. If it is the latter, it is typically the result of satisficing—the behaviour of the artifact might not perfectly align with the hypothesis, but is judged as "good enough". The results of the effort are written up, and the knowledge gained is typically categorised as "firm", where it has been learned and can be applied or repeatedly invoked, or as having "loose ends", where the artifact exhibits anomalous behaviour that defies explanation and requires further research. The researcher reflects on what was learned, and what did or did not work, and in communicating the results, abstraction can enable the researcher to draw broad and generally applicable conclusions based on the knowledge gained.

DSR in Action

The research effort that took place started as an opportunity for thesis work provided by Dewire. They had identified a need within their organisation for a drone simulation tool in order to help build systems involving drones, preferably within the AWS ecosystem. Beyond that, very little was set in stone; we would make decisions about design and function as we discovered more about the possibilities and constraints of our circumstances. In this section, rather than presenting solely the final design decisions that were made, our journey through the DSR process model (figure 2) will be detailed, in order to showcase the evolution of the artifact's design and purpose, as well as the generated knowledge.

The first phase

Serving as a trial run of sorts, we approached the first iteration of the cycle with a slightly narrower perspective than might be prescribed. While DSR can and should solve organisational problems, the need for the artifact-to-be should really be of interest to a wider audience—a research community at least. However, at this initial stage we knew so few concrete things about what we wanted to create, and were more focused on learning what was possible with the instruments we set out to work with; we thus only considered Dewire's organisational need for a tool when it came to defining the problem and devising a solution.

The first step of the DSR process, Awareness of Problem, was more or less taken care of given the context with which the thesis work opportunity was provided. The problem had already been identified by Dewire, and a solution proposed alongside it: they want to build and test systems for controlling and overseeing fleets of drones, and the suggested solution was to build a drone simulator to facilitate this. This generated the expected output of a proposal for a research effort. The generic knowledge that would be generated as a result of the research effort was how a simulator could be used as a testing tool, more specifically for drone control systems. This served as a basis for the initial research question, "how can we use a simulator to test drone fleet control system functionality?".

Before starting the developmental phase of the research effort, my supervisor and I had a meeting to discuss what we hoped to achieve by the end, i.e., drawing up the tentative design of the suggested solution. However, the discussed functionality and design of the artifact were kept quite high-level and abstract due to our being unfamiliar with the tools and implementation of the artifact. I in particular had no prior experience working with AWS, let alone the simulator; we were both aware that only when we knew what we were working with could more concrete design decisions be made and goals set. Once we had gotten an understanding of the capabilities and limitations of the simulator as a base, only then could we begin to formulate reasonable goals. As such, the output of this phase, the tentative design, was chiefly based on pre-existing knowledge of other simulators and the properties of drones. We knew that we wanted the simulated drones to communicate and that they should follow a flight path, but exactly how this would be handled could not be known at the time.

Because the simulator's original purpose was to simulate self-driving cars, we thought it prudent to map the properties and metrics that were tracked to those that a drone might have. For instance, tracking battery life instead of fuel levels. We also realised that there would be properties unique to drones, and spent some time researching drone mechanics. How many propellers does the drone have? Is the motor brushed or brushless? What types of sensors is the drone equipped with? In addition to cartesian coordinates (X, Y, Z), we might also be interested in the pitch, roll, and yaw of the drone—clearly there was a lot to consider. In the beginning we set our sights high and were ambitious and optimistic about the sophistication level of the simulator we would develop, so while the tentative design was mostly high level due to unfamiliarity with the topic, we were expecting to account for many variables and create a somewhat high-level, or, in other words, realistic, simulator.

However, as we moved into the Development phase and delved into the code of the base simulator in order to understand its structure and workings, we quickly realised that most of these aspirations were out of reach. A lot of the additional features would prove too complex and time-consuming to program. We realised that there was a need for us to rethink what the final artifact would look like, and perhaps even redefine or reconsider the problem to be solved. Learning about the limits and capabilities of the simulator allowed us to form more concrete ideas concerning the design and function of the artifact. It also shaped the knowledge we hoped to generate; having a rough idea of what the simulator will be like, it was easier to conceive testing scenarios utilising it. These discoveries served to further the branch of the knowledge pertaining to Dewire's organisational needs.

Encountering this sort of roadblock is an expected occurrence in a DSR research effort, and the process model prescribes revisiting earlier steps and performing multiple cycles, partial or full. In line with this, we returned to the first stage with newly acquired knowledge.

The second phase

Having garnered more knowledge of the simulator, the limitations of which we became aware served as circumscription in forming our subsequent cycle through the DSR process. Dewire's need for a tool to help build and test drone fleet control systems still existed, but now somewhat understanding what was within our power to create, we had to survey the field of drone usage today in order to find a larger problem to which we could contribute.

Dewire was especially interested in fleets of drones, so this served as the starting point for the literature study. Learning about the fields in which drones are used today and how they could benefit from increasing capacity made it clear that this research was relevant to many communities and industries. The other part of the study focused on identifying what problems needed to be solved in order to achieve the goal of employing fleets of drones, rather than one or a few, and what other researchers have already contributed. While the output of the Awareness of Problem phase didn't change, we had identified another problem to which the development of the artifact could contribute.

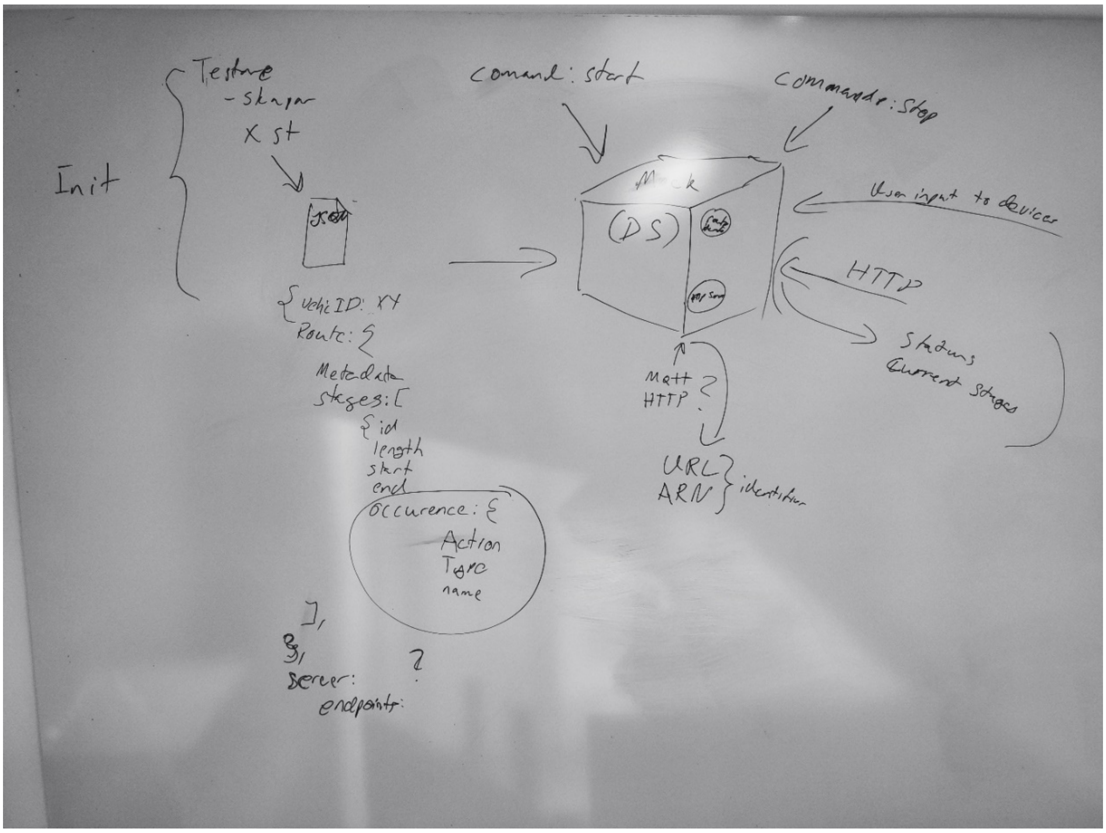
This new knowledge let us refine our design and choose specific issues to focus on within the greater context of enabling the deploying drone fleets; i.e., it let us envision a more grounded and concrete tentative design in terms of functionality. Something that stood out to us was the requirement of consistently exchanging flight information, as it aligned well with what the simulator was capable of and what Dewire hoped to use it for. Thus, we designated communication within drone fleets and to governing systems as the issue we would focus on. The tentative functionality-goal was to have the simulated drones regularly communicate their coordinates, as well as any events that occurred along the route, for instance, running out of battery. We also wanted to be able to get an overview of all deployed drones, while still being able to see or request specific information about any given one—traceability was an important aspect of the communication to Dewire. The addition of another problem to which we could contribute came with the expectation of more knowledge being generated. Having now chosen a more specific direction in our work, the additional knowledge we sought to produce concerned the communication within drone fleets, and served as the basis for the second research question, "how can issues of communication requirements within drone fleets be solved?".

Diving into the specifics of the envisioned functionality, positional data was something that was generated by the simulator in its base state, we just had to supplement, consolidate, and send it through a channel of our choice. As reasoned by Hedström and Gudjonsson (2020), MQTT is a good fit for the situation due to its lightweight format, allowing it to work well even with limited bandwidth or memory. The small footprint of MQTT is also of importance when dealing with fleets of drones. The messages may be small as individual packets, but when hundreds of drones are sending multiple packets every second, the volume of the data quickly adds up. When it came to events, as we'd call them—things that could occur during the flight on the specified route—we were more concerned with the communication of the event occurring, rather than *how* the specific event occurred. Returning to the example about battery levels, we were only interested in having drones send a message when they ran out of battery, and perhaps having them take an action in response, while not putting much focus on the mechanics of battery drainage or the taken action. In a high-fidelity simulation, one might have to consider motor and propeller models, environmental factors such as heat and wind, amongst other things. Not having to concern ourselves with such technical and mechanical aspects shrunk the scope of the work considerably and made it more manageable for the time frame and manpower (or lack thereof) we were working with.

With a clearer vision of what the design should look like, we moved onto the Development phase for the second time, this time having done away with our vision of a somewhat high-fidelity (realistic) drone simulator, as developing those features would require time and expertise we did not have at hand. As such, the tentative design from the previous phase had shifted from a high-level design to a more low-level design, as we could envision specific features and functions of the simulator. However, the scope of the project had shrunken, and the sophistication-level of the simulator had been reduced from high fidelity to low fidelity. We placed less emphasis on the simulation itself, and more emphasis on the output it would generate, keeping in line with our approach to event communication and battery drainage.

Figure 3

Design Brainstorming on Whiteboard



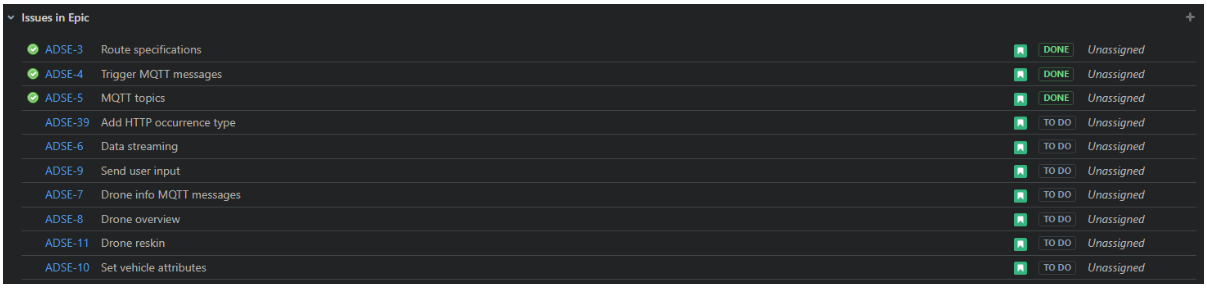
In order to keep the workload manageable, we needed to keep the way the simulator worked largely the same. What we wanted to do was modify functions and create new ones that fit within the framework and rules of the base simulator. Figure 3 showcases one of our meetings where we brainstormed for solutions and implementations of different aspects of the simulator. Decisions about functionality and user interaction were made from a tester's perspective using my supervisor's insight as a software consultant. This input and the simulator's initial construction served as constraints when it came to design decisions.

Discussions about practical design choices were had in person and we often employed whiteboards to write and draw diagrams to aid the process. Management and work-related issues were similarly had in person, but with the help of digital tools. We used Jira, an agile project management tool, to divide the work into epics, stories, and sprints.

One such design decision concerned the degree to which the user could control input and output. As the artifact would see use as a testing tool, it was important that the user could make specifications about the simulations. We went through a few iterations of feeding input to the simulator before settling on one that collated many options into one action, which also aligned with how the simulator handles data internally so that we could transmit the data generated by the user's input in MQTT messages.

Figure 4

Jira Epic at One Point in Development



As the core features started to take shape, like triggering MQTT messages and route specifications, we decided that getting the most important functions working well was more important than developing crude methods for less-prioritised ones. As a result of this, not everything pictured in Figure 4 was finished. Features that were deemed "nice to have" but ultimately of lesser priority include data streaming and allowing real-time user input. Part of the reason why these features were prioritised lower was also due to their complexity; it would use up too much of our allotted time.

As we worked using agile methods, the development phase contained its own sub-cycles, where individual features were worked on in iterations. As we conceived, developed, and evaluated each feature, we would learn more about the simulator itself and how we wanted to handle certain aspects. Because of the heavy learning aspect of this particular research effort, the insight gained during the development of each feature led us to being able to form new hypotheses, as well as confirm or having to modify existing ones. Thus, the development led to frequent returns to the earlier steps in the DSR process, yielding slight modifications to envisioned functionality or hypotheses.

Later, all features would be considered as one entity—the artifact—for a final evaluation. It was when this final evaluation took place that the generated knowledge really took shape. As we evaluated the artifact's behaviour against the specifications of our design in Jira, we could consolidate the results and form hypotheses about the impact it could have.

Piecing together functionality, like generating positional data at specified intervals and sending that data via MQTT messages, gave credence to hypotheses that spawned during development. In the given example, we learned that this could be a viable protocol for communication. We then attempted to incorporate the knowledge we had generated and the piecings-together we had made into existing research, which will be elaborated upon in the Discussion and Conclusions chapter.

Results

The research effort's resultant artifact and its functionality will be described below. It consists of a simulation engine, which is able to simulate drones and other user-defined widgets, and a web console user interface (UI) where one can oversee and configure the running simulations.

As shown in Figure 1, the simulator works through the combined effort of a host of AWS services. This architecture remains the same, as large-scale changes to the simulator and its structure has not been made.

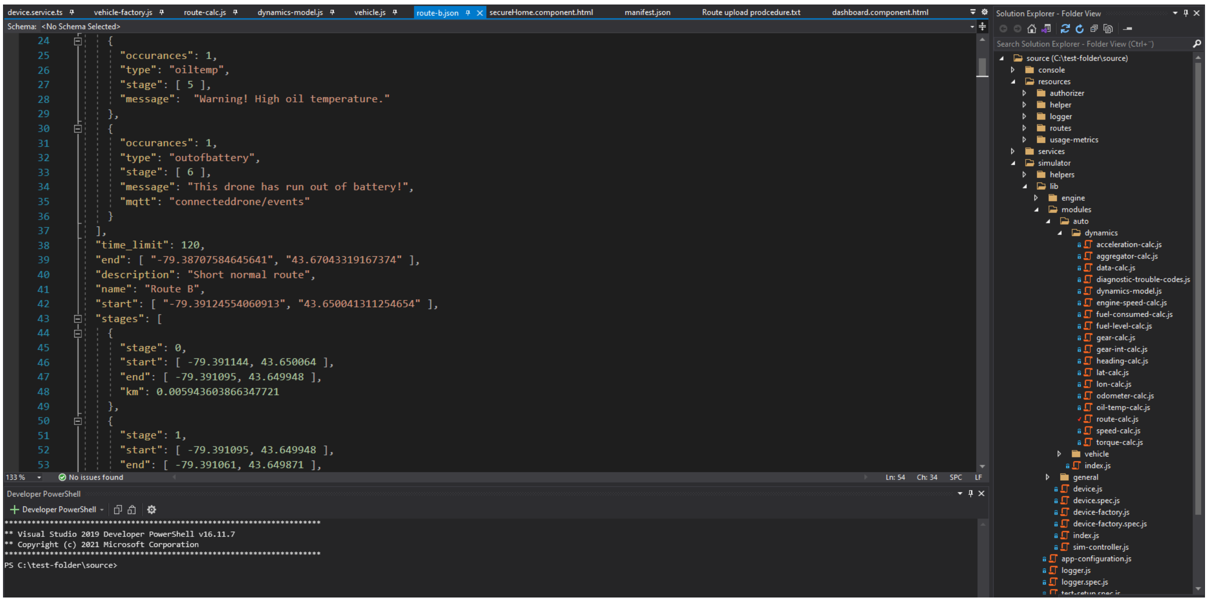
Setup and User Input as a Tester

Before launching the simulator, context-specific changes can be made to JSON files which provide the simulation engine with instructions, route coordinates, and event triggers. These files, and all other files related to the console and routes, are stored in AWS S3 Buckets, as per the diagram in Figure 1. Changes can be made to, for instance, have the simulated route always produce a given event at a specific point, or let the simulation engine generate random events from a pool at a random stage of the route. This way, the tester can use the simulator more effectively as a tool to specifically test variations of certain scenarios multiple times while having fine control over the parameters, or simulate a more organic iteration of one or more routes. A snippet of the JSON file is shown in Figure 4, and a screenshot of the S3 Bucket content in Figure 5.

The simulation engine is a Docker image hosted in AWS Elastic Container Registry (ECR). Changes to how the simulator performs calculations or how it interprets data stored in the S3 Buckets must be pushed to ECR as a new Docker image; this compilation of the code is separate from the console's code. Changes made to the engine can affect the core functionality of the simulator, and is thus not something that should be a regular occurrence. Still, the tester does have the ability to do so should the situation require it—the artifact is quite adaptable in suiting the needs of the tester.

Figure 5

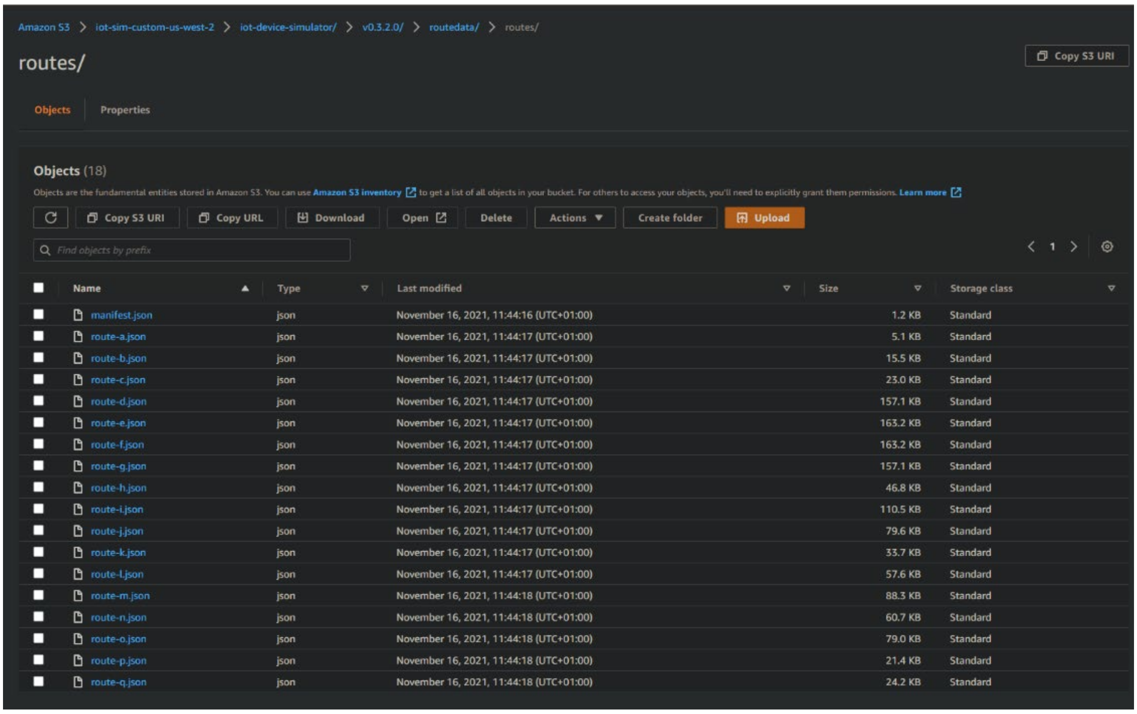
Structure and Content of the JSON file



Note. A snippet of a route JSON file, displaying settings such as events that should occur along the route, the coordinates to which the drone should travel, and route metadata like name and description. On the right can be seen the file structure of the artifact, in focus on the screen being calculation files for simulation behaviour.

Figure 6

Amazon S3 Bucket Content



Web Console and Core Functionality

Figure 7

Drone Simulator Web Console UI Overview

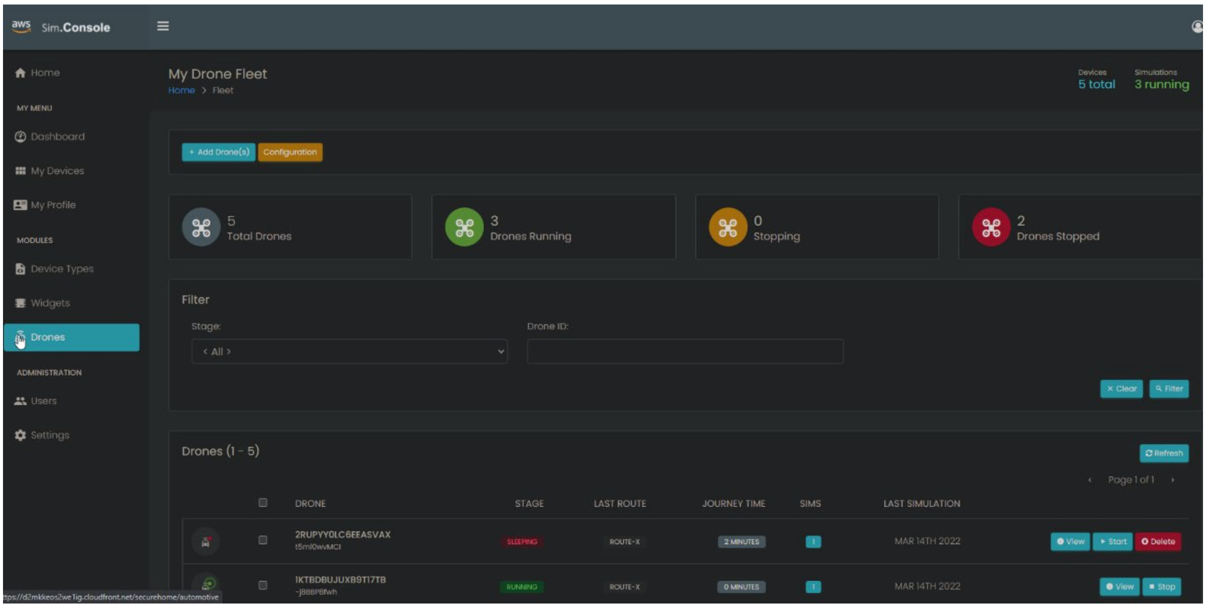
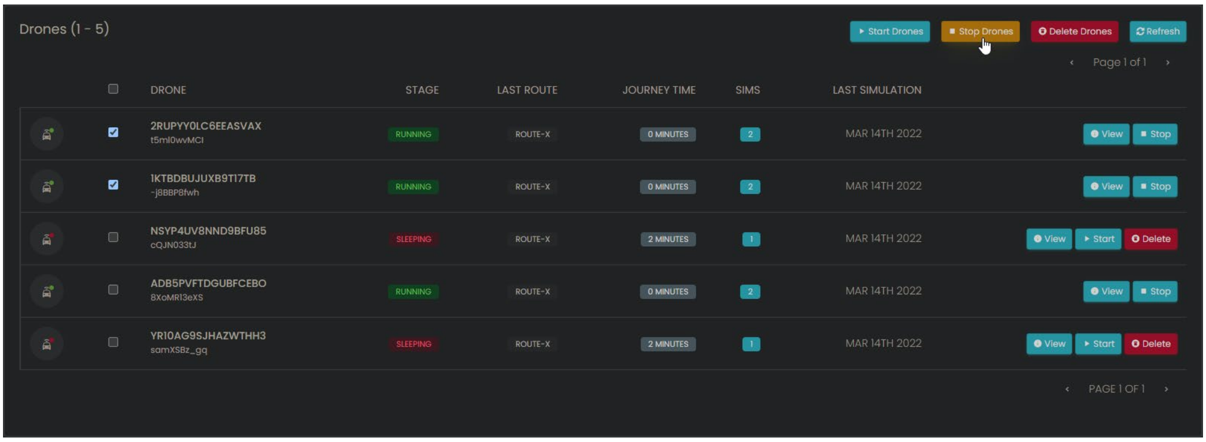


Figure 6 shows the overall structure of the web console UI, with focus on the Drones tab. Here one can create and manage drone simulations. Up to 1,000 simultaneous simulations can be accommodated. An overview of all drones and their statuses is presented to the tester, and they can navigate the console to access various functions, showcased in the figures below.

A clearer view of the list of simulated drones is shown in Figure 7 below. The tester can start and stop simulations, as well as delete drones. Information is presented to the tester so that they can tell at a glance the status of any given drone. From this view, the tester can navigate to a detailed view of a specific drone by pressing the View button on the right. This detailed view is shown in figure 9.

Figure 8

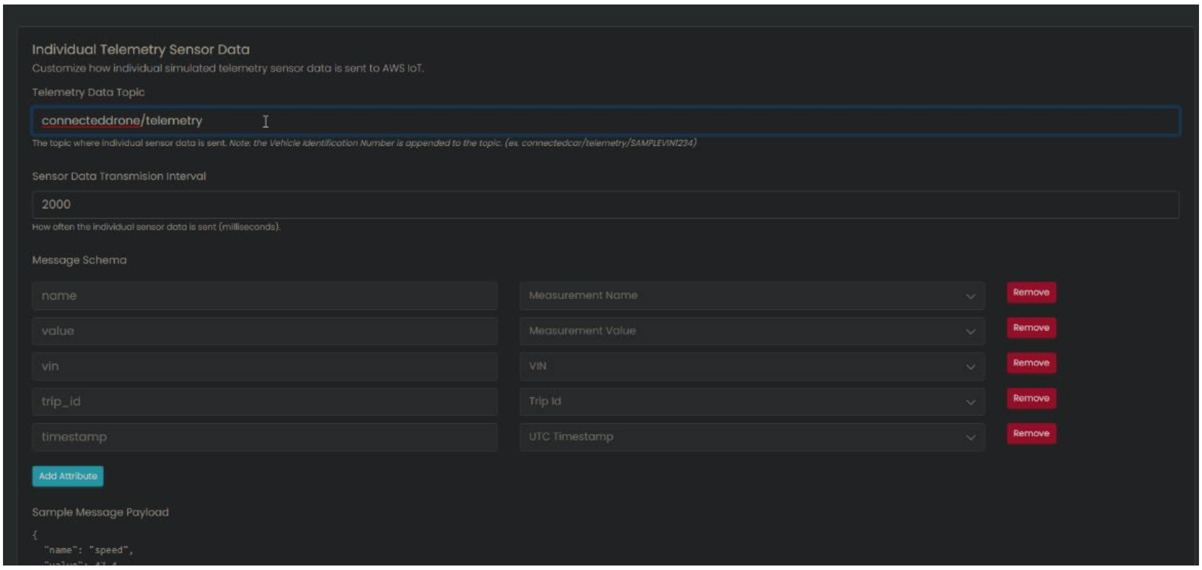
List of Simulated Drones



At the top of the interface in Figure 6, the tester can access a configuration page, displayed below in Figure 8. Here, changes can be made to the communication aspects of the drone. The MQTT topic to which messages are published can be edited, overriding the default or the topic specified in the JSON file pre-launch. The interval at which information is sent and the contents of the message can be edited as well.

Figure 9

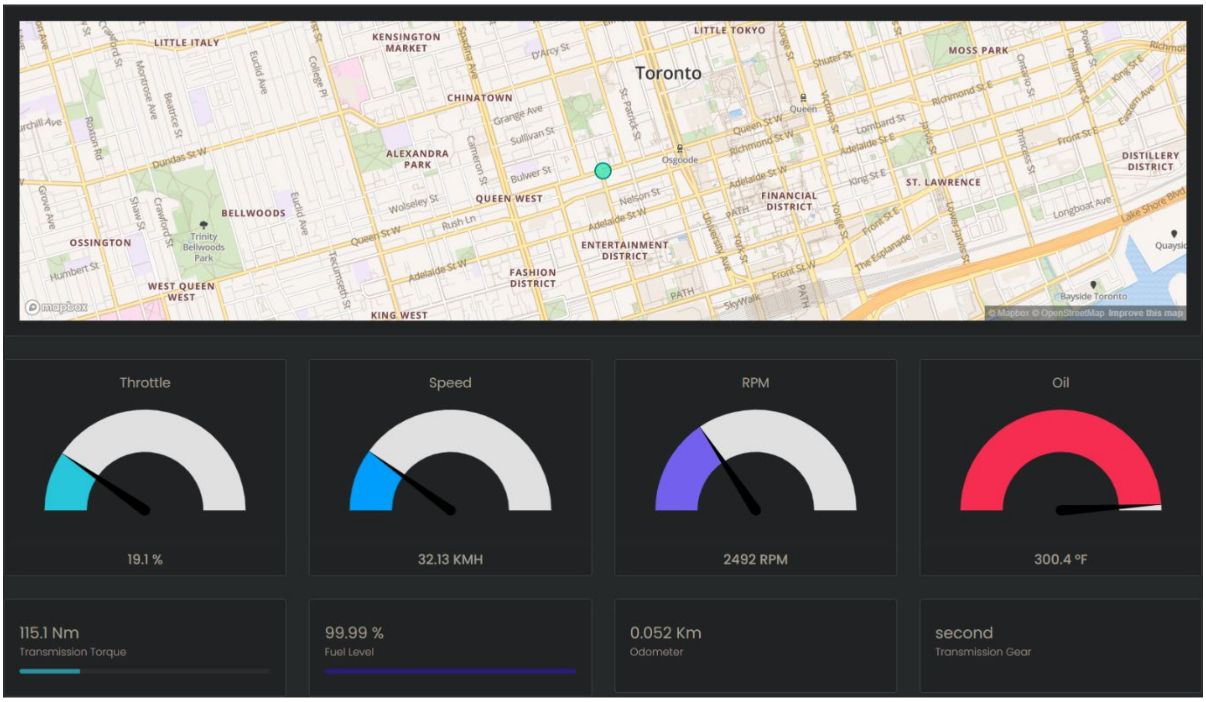
Drone Telemetry Configuration Interface



When navigating from the overview to the detailed view, the interface in Figure 9 is presented. The location of the drone, based on the route coordinates in the JSON file of the selected route, is drawn on a map and update in real-time.

Figure 10

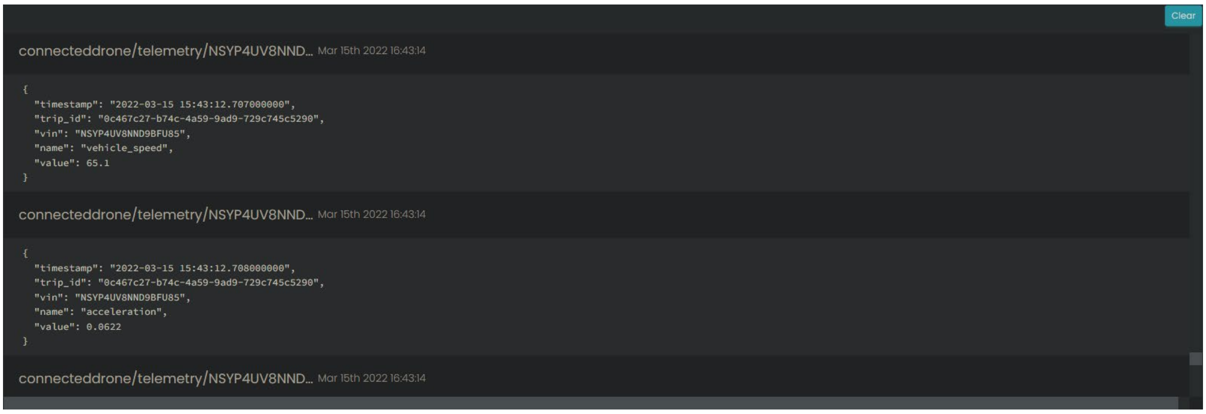
Detailed View of Simulated Drone



Also on the detailed view is all the telemetry data generated by the drone. These packets are sent as MQTT messages to the topic shown at the top of each. The random string at the end of the topic is the drones unique ID, which allows one to isolate messages sent by a specific drone should that be desired.

Figure 11

Integrated List of Generated Drone Telemetry



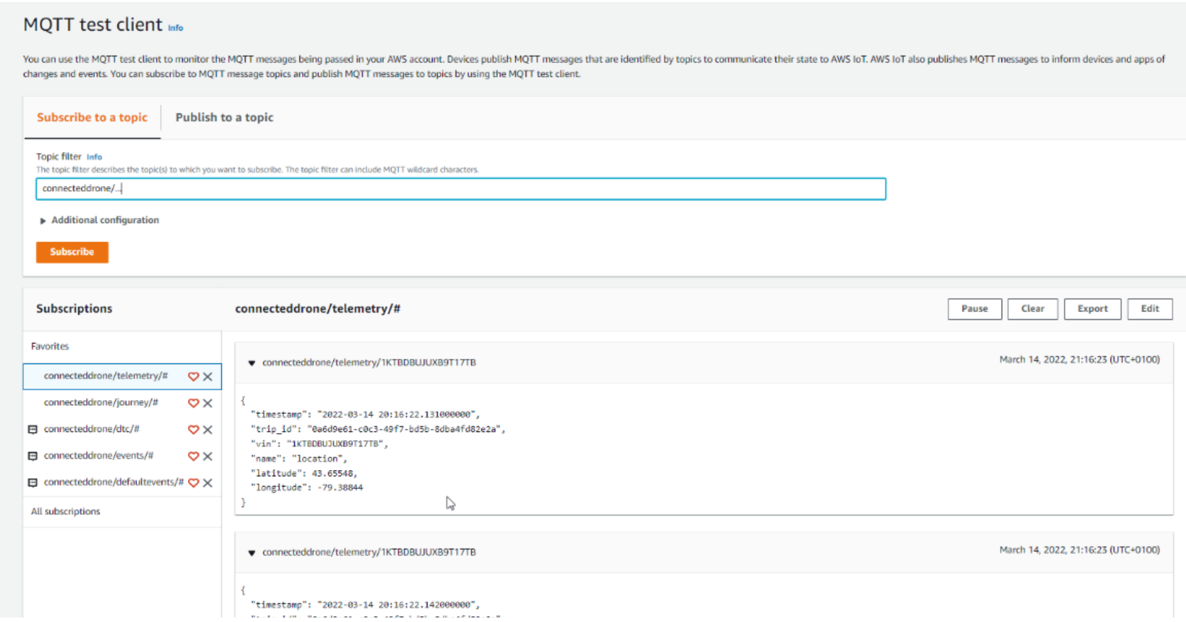
IoT Connectivity and MQTT messaging

The generated telemetry data of any given drone is received and displayed under their detailed view, but there is also the ability to receive the data via an external client. Via AWS's MQTT test client (or indeed any other MQTT client) we can subscribe to the topics specified in the web console configuration to receive the telemetry data of all simulated drones in one place.

Messages can be filtered by further specifying a topic, and further still by drone IDs. The user can thus set up multiple subscriptions in order to have separate feeds of messages that suit their needs.

Figure 12

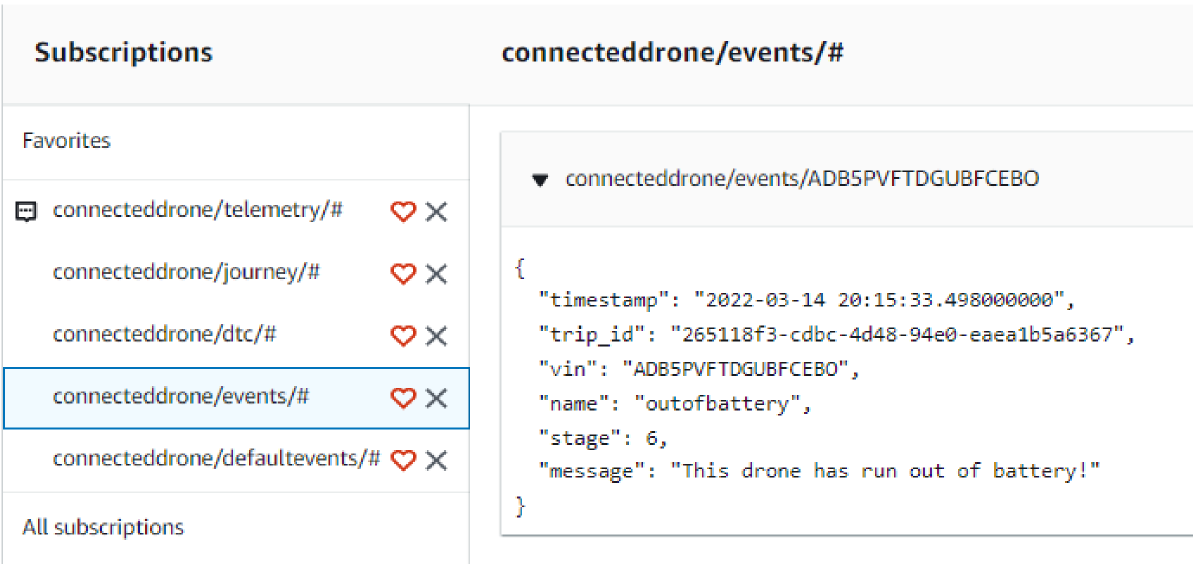
Overview of External MQTT Client



Note. Subscriptions to topics use the pound sign (#) as a wildcard, and is here used to receive messages published by all drones.

Figure 13

Detailed View of the Event MQTT Topic



When the drones encounter an event, like running out of battery, or anything specified by the tester in the JSON file, messages will be published to the events topic, with vital information like when it happened, to which drone it occurred, where on the route, and a generated message.

Discussion and Conclusions

In this chapter, the presented results will be discussed and related to the problems to which we intended to contribute knowledge—the issue of communication within drone fleets and to control systems. This discussion will enable us to draw conclusions about the impact of the research and the developed artifact.

Discussing the results

As stated in the Background and DSR in Action chapters, our chief focus was on developing communication capabilities for the simulated drones. In accordance with the issues outlined by Gupta et al. (2016) and Yoo et al. (2015) regarding the exchange of information between drones and communication with a control centre, we strove to build on the inherent MQTT messaging capabilities afforded by the simulator, due to its IoT connectivity within AWS, to address those requirements.

In spite of the many challenges surrounding the work leading to the simulator not being very feature-rich, the features that have been developed do live up to our expectations. The control the user has over the simulation via the JSON file unlocks its potential as a tool, and the simulated drones in our artifact consistently transmit the telemetry data they generate at a user-defined rate, which can be as often as more than once a second. This data is transmitted in an easily-accessible and lightweight format, MQTT. We can receive the MQTT messages via the web console in our artifact, as well as via an independent MQTT client. The simulation satisfies the requirement of communicating to a control station—in this case, the independent MQTT client overseen by a user—where operators can take action depending on the information they receive, or feed the data into analytics tools.

When considering this functionality apropos our research question “how can issues of communication requirements within drone fleets be solved?”, the fact that the simulated drones make use of the same line of communication as a real drone might lends credibility to the knowledge generated by the results of the simulation—the behaviour and limits of the simulation should map well to a real-life implementation. This is also where the other research question, “how can we use a simulator to test drone fleet control system functionality?”, becomes relevant. My supervisor stressed the importance of control on the part of the tester when utilising the simulator as a testing tool. By fine-tuning the parameters of the simulation, one can more exhaustively test a system’s limits. How are the messages best categorised as MQTT topics? How does the simulation react to throttling of the data via manipulation of the frequency or allotted bandwidth of messages? These things can be thoroughly ascertained by repeated testing using the simulator. And the knowledge derived from these tests feed back into the former research question; because the results and information the simulator generates are mappable or applicable to real-life scenarios, the continued use of the simulator as a tool will further contribute to the communication issue as a whole.

In the case of autonomous drone fleets, they must communicate with each other. While this was not something we achieved in our simulator during the allotted time, Devos et al. (2018) note that drones are often equipped with various types of equipment such as telemetry sensors, and microcomputers such as a Raspberry Pi to facilitate communication. Since drones are already equipped with these tools in order to send messages, the idea then is to use them to also receive the MQTT message and have the drone interpret it in real-time to make necessary adjustments in its operation.

Gupta et al. (2016) bring up certain difficulties with UAV networks, highlighting the ephemerality of drones as network nodes as a crucial aspect in choosing routing protocols for seamless information handovers. By utilising the IoT, AWS, and cloud computing, we can attempt to solve this aspect of the issue of communication. The messaging protocol of choice is, again, MQTT, which as stated before is very suitable for networks with less-than-ideal conditions. Hedström and Gudjonsson (2020) compare different IoT architectures for drone fleet operations, eventually concluding that a hybrid architecture proves most optimal. This hybrid architecture would comprise edge computing, where computations are performed by the drone itself, and fog computing, where a local node does the brunt of the computation.

A drone network adopting this hybrid architecture would see non-time-critical information, i.e., information that does not require the drone to take any immediate action, being sent to the cloud, while more time-critical computations would be made either by the drone itself (edge), or in a fog node (fog). This fog node would, in our case, be a part of AWS or another drone, and allows us to keep the drones lightweight, while also filtering data sent to the cloud to avoid overloading while simultaneously reducing latency for time-critical computations. It could also be possible to vary the intervals between transmission between data labelled time-critical and that which isn’t. By sending time-critical information are more frequent intervals one can assure that it is acted upon as soon as possible, while sending non-time-critical information less frequents—and some information only at an end-of-journey summary—one can reduce stress on the system.

This topology—the arrangement of the network—aligns well with the issues Gupta et al. identified; in a situation where many drones are deployed, there would be no shortage of nodes in the network, and when there are few, they can rely on the cloud to avoid network partitioning—that is, failure to communicate between nodes.

While we did not achieve something that could truly represent this constellation of components, we have learned through our developing the artifact that utilising the Internet of Things unlocks a suitable messaging protocol in MQTT. The results produced by the simulator serves as a base upon which we can build theoretical topologies, such as the one described above. While questions as open as “how can issues of communication requirements within drone fleets be solved?” rarely have a definitive answer, it can be said that one way it can potentially be solved is through having drones exist as IoT devices.

As for addressing the organisational needs of Dewire, my supervisor predicts that the controlled input and output will be of great importance when using the artifact as a testing tool. When considering actual applications of using the simulator as a tool, it is important that it can deliver consistent results, and that the tester has control over the output. Different outputs can in turn be used as inputs to test functions in a separate control system. However, no system that can utilise the artifact as such is currently in development, so its use can only be hypothesised for now. During development, the web console served as a sort of mock, or pseudo-control system, where one could keep track of and oversee all simulated drones.

Conclusions and Future Work

The employing of drones is in many fields of work a relatively novel phenomenon, with great potential for growth. The artifact produced through this research effort demonstrates one possible answer to the question: “how can issues of communication requirements within drone fleets be solved?”. That is, that by utilising IoT and MQTT it is possible to efficiently transfer large volumes—divided over many packets—of information, as is necessary in drone networks. It also opens the door to different architectures, and by utilizing edge and fog architectures it should theoretically be possible to make up for shortcomings of drone networks, such as node availability.

While it is currently not possible to evaluate the artifact’s actual efficacy as a testing tool, as no system to utilise it exists as of yet, we have use what we have learned to answer the question: “how can we use a simulator to test drone fleet control system functionality?”. The tester has fine control of parameters and is able to produce consistent, reliable outputs of desired nature. This output can be fed into a separate control system to test different functions. It is also possible to test the inherent properties of the systems that make up the artifact. By fine-tuning parameters, the user can attempt to find balances, breakpoints, or thresholds regarding things such transmission frequency and allocated bandwidth.

There is however still merit to continuing work on the simulator, as developing more functions would allow it to be used to test more aspects of a system. By using the simulator as described latterly above, one also makes contributions to the first research questions with the findings, as learning about these aspects can provide further insight into the issue.

The scope of this research effort was limited due to a number of factors such as lack of manpower, expertise, funding, and time. If working on the simulator was the next step following Hedström and Gudjonsson’s work, the next step from here would be to make practical applications of what we’ve learned, using actual drones. It is one thing to test theories in a simulator using mocks, and another to apply those theories in situ.

References

Ahirwar, S., Swarnkar, R., Bhukya, S., & Namwade, G. (2020). Application of drone in agriculture. *International Journal of Current Microbiology and Applied Sciences, 8*(1), 2500-2505. https://doi.org/10.20546/ijcmas.2019.801.264

Baskerville, R., Baiyere, A., Gregor, S., Hevner, A., & Rossi, M. (2018) Design science research contributions: Finding a balance between artifact and theory. *Journal of the Association for Information Systems*, 19(5), 358-376. http://doi.org/10.17705/1jais.00495

Cohn, P., Green, A., Langstaff, M., & Roller, M. (2017). *Commercial drones are here: The future of unmanned aerial systems*. McKinsey. https://www.mckinsey.com/industries/travel-logistics-and-infrastructure/our-insights/commercial-drones-are-here-the-future-of-unmanned-aerial-systems

Cordova, F., & Olivares, V. (2016). Design of drone fleet management model in a production system of customized products. *6th International Conference on Computers Communications and Control (ICCCC)*, 165-172. https://doi.org/10.1109/ICCCC.2016.7496756

Devos, A., Ebeid, E., & Manoonpong, P. (2018). Development of autonomous drones for adaptive obstacle avoidance in real world environments. *21st Euromicro Conference on Digital System Design (DSD),* 707-710. https://doi.org/10.1109/DSD.2018.00009.

Food and Agriculture Organization. (2020). *Global Forest Resources Assessment (FRA) 2020 Sweden – Report*. https://www.fao.org/3/cb0063en/cb0063en.pdf

Gupta, L., Jain, R., & Vaszkun, G. (2016). Survey of important issues in UAV communication networks. *IEEE Communications Surveys & Tutorials,* 18(2), 1123-1152. https://doi.org/10.1109/COMST.2015.2495297

Hart, C. (2018). *Doing a literature research: Releasing the research imagination*. SAGE publications Ltd.

Hedström, F. S., & Gudjonsson, A. (2020). *Drones in the cloud: A study of IoT architectures and simulation in AWS.* [Master's thesis, Lund University] Department of Computer Science.

https://lup.lub.lu.se/luur/download?func=downloadFile&recordOId=9024913&fileOId=9024914

Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design science in information systems research. *MIS Quarterly*, 28(1), 75–105. https://doi.org/10.2307/25148625

Iowa State University. (2019, May 2). Obstacles to overcome before operating fleets of drones becomes reality. *ScienceDaily*. Retrieved November 18, 2021 from www.sciencedaily.com/releases/2019/05/190502161644.htm

Kinaneva, D., Hristov, G., Raychev, J., & Zahariev, P. (2019). Early forest fire detection using drones and artificial intelligence. *42nd International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, 1060-1065. https://doi.org/10.23919/MIPRO.2019.8756696

Lichtman, A. & Nair, M. (2015). Humanitarian uses of drones and satellite imagery analysis: The promises and perils. *AMA J Ethics, 17*(10), 931-937. https://doi.org/10.1001/journalofethics.2015.17.10.stas1-1510

Mahadevan, P. (2010). The military utility of drones. *CSS Analysis in Security Policy, 78.* https://doi.org/10.3929/ethz-a-006253833

Mishra, B., Garg, D., Narang, P., & Mishra, V. (2019). Drone-surveillance for search and rescue in natural disaster. *Computer Communications*, *156*, 1-10. https://doi.org/10.1016/j.comcom.2020.03.012

Mock object. (2022, January 11). In *Wikipedia*. https://en.wikipedia.org/w/index.php?title=Mock_object&oldid=1049556133

Vaishnavi, V., Kuechler, W., & Petter, S. (2004). Design science research in information systems. *Association for Information Systems*. Last updated June 30, 2019. http://www.desrist.org/design-research-in-information-systems/

Vergouw, B., Nagel, H., Bondt, G., & Custers, B. (2016). Drone technology: Types, payloads, applications, frequency spectrum issues and future developments. *The Future of Drone Use. Information Technology and Law Series*, 27, 21-45. https://doi.org/10.1007/978-94-6265-132-6_2

Veroustraete, F. (2015). The rise of the drones in agriculture. *EC Agriculture, 2.* https://www.researchgate.net/publication/282093589_The_Rise_of_the_Drones_in_Agriculture

Yoo, S., Kim, K., Jung, J., Chung, A. Y., Lee, J., Lee, S. K., Lee, H. K., & Kim, H. (2015). Poster: A multi-drone platform for empowering drones' teamwork. *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking (MobiCom '15). Association for Computing Machinery*, 275–277. https://doi.org/10.1145/2789168.2795180

Huang, Z., Wu, W., Shan, F., Bian, Y., Li, Z., Wang, J., & Wang, J. (2019). CoUAS: A cooperative UAV fleet control and monitoring platform. *IEEE Globecom Workshops (GC Wkshps),* 1-6. https://doi.org/10.1109/GCWkshps45667.2019.9024481