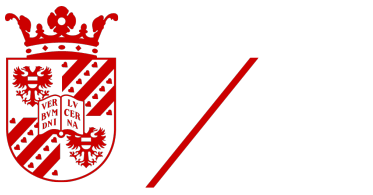
Autonomous Swarm Management for Bridge Inspection

Authors:
Eduard Sabo (S4121937)
Adrian Predescu (S4729307)
Minh Tam Le (S5723795)
Joep Scheltens (3487385)
Chan Chan Tran (S4116488)
Mojmír Majer (S5688760)

Lecturer:
Prof. Michael Stal
April 18, 2026



university of groningen

Contents

List of Figures

List of Tables

1 Introduction and Goals

Swarm Intelligence (SI) is a paradigm in distributed artificial intelligence that models decentralized, self-organized systems inspired by natural collectives such as insect colonies and bird flocks [3]. These systems rely on local interactions between agents to achieve global objectives, offering adaptability, scalability, and resilience. These traits are critical for autonomous drone swarms operating in dynamic, unstructured environments like bridge inspections [1].

Traditional bridge inspection methods, which depend on manual visual assessments or heavy machinery, are labour-intensive, costly, time consuming, and hazardous [2]. Recent advancements in Unmanned Aerial Vehicle (UAV) swarm robotics demonstrate their potential to address these inefficiencies. For instance, the European Drones4Safety (D4S) project has validated drone swarms in infrastructure inspections, achieving 40% cost reduction and eliminating human exposure to high-risk areas like bridge undersides [8]. Similarly, the University of Cambridge's aerial swarm research integrates machine learning for real-time crack detection, addressing limitations in conventional remote sensing techniques such as low spatial resolution and incomplete documentation [12].

1.1 Business Goals

Our business vision revolves around providing an Autonomous Swarm Management System for Bridge Inspection that follows a multi-tenant Software-as-a-Service (SaaS) model, allowing multiple customers to use the same platform while maintaining separate datasets and configurations. Each customer (expected to be an organization rather than an individual) will have dedicated access to features such as inspection planning, real-time monitoring, and AI-driven defect analysis, meaning that while the core functionalities remain the same across all clients, variability is only introduced through configurable mission planning, AI model customization, and compliance adjustments based on regional regulations.

The target customers of our system are primarily infrastructure owners, either government agencies or private ones, which also represent our main stakeholder (more information in subsection 1.4). Initially, the system will be deployed in the Netherlands, Germany, and France, where there is a pressing need for efficient bridge maintenance with a strong focus on innovation, with possible future expansions to North America and Asia.

The envisioned revenue model is based on a tiered subscription system, per-inspection fees, and optional drone leasing. Subscription plans offer, which offer unlimited number of missions per month but with only 3 active at once, range from €2,500/month for basic access to €5,000/month for enterprise features, including premium support and resource priority (drones and dates) when planning inspections. Customers who require inspections on a less frequent basis can opt for a pay-per-use model, with pricing between €1,500 and €4,000 per inspection, depending on mission complexity. Additionally, organizations can

also lease drones for €500 per drone/month, thus reducing the upfront investment but with the downside of having to manually pilot the drones.

1.2 Requirements Overview

In this section, we will go over the main use cases and functional requirements of our system. We define the main requirements for all stakeholders, such as general drone management and scheduling operations, performing the actual bridge inspection, drone safety, and non-drone related requirements such as outputting the final report and a description of the user interface. Afterwards, we investigate the most crucial 3 use cases, regarding the bridge inspection, scheduling and analyzing of the results.

Table 1: Functional Requirements Table

Id	**Requirement**	**Explanation**
F1	Manage swarm of drones	The drones will fly and perform tasks autonomously, but there should be a central unit where people oversee the swarm
F2	Drones inspect bridge	The drones will perform specific tasks related to the inspection of a bridge. Based on multiple environmental conditions, the swarm of drones can perform lesser capabilities, e.g. during heavy rain/wind/snow, the drones won't be able to perform their above ground, unless they check the structure of the bridge underneath the bridge, to avoid bad weather.
F3	Schedule drone operation	Schedule what each drone is doing within the swarm, with respect to the bridge inspection operation.
F4	Generate report	Based on collected data, generate a report following the bridge inspection, which states whether the bridge needs maintenance or not
F5	Implement user interface	Have a web user interface where users and administrators can securely log in and see data and what actions they are allowed to perform
F6	Drone safety	Maintain drone safety at all points during the operation, by making sure it avoids damage or hitting other objects/people.

1.2.1 UC-1 - Bi-yearly Bridge Inspection

Primary Actors	Drones, Bridge, Bridge Engineer, Drone Operator
Trigger	The time schedule indicates that a bridge inspection is due
Goal	Conduct a scheduled inspection of the bridge using autonomous drones

Preconditions	• Weather conditions must be optimal for drone operation • Drones must have legal flight clearance in the inspection area • The drone fleet must be fully operational with sufficient battery levels • The inspection system must have the bridge location and inspection plan configured
Main Success Scenario	1. The system detects that an inspection is due based on the schedule. 2. The system verifies weather conditions and regulatory permissions. 3. The drone swarm is assigned tasks and deployed from the designated launch point. 4. Drones autonomously navigate to the bridge using GPS and obstacle avoidance sensors. 5. Drones perform structural scanning using cameras, LiDAR, and other sensors. 6. Drones transmit real-time data to the ground station. 7. The system analyzes initial data to detect critical structural issues. 8. Drones return to the deployment location. 9. The collected data is stored in the system for expert analysis.
Extensions	• (4.a) If automated flight is restricted, the drone operator manually controls the flight. • (5.a) If a drone malfunctions mid-flight, an alert is triggered, and a backup drone is deployed. • (6.a) If real-time data transmission fails, data is stored locally on the drone and uploaded upon return. • (8.a) If a drone fails to return, the system notifies the operator and attempts to locate it.
Postconditions	The drone inspection data is successfully stored and ready for analysis
Related Requirements	F1, F2, F3, F6

1.2.2 UC-2 - Expert Analyzes Gathered Bridge Data

Primary Actors	Web Interface, System, Expert (Bridge Engineer)

Trigger	Time schedule of inspection; has the scanning by the drones been completed?
Goal	Generate an analysis report to determine whether the bridge requires maintenance.
Preconditions	• Data from the drone inspections is available. • The data has been cleaned, preprocessed, and feature engineered. • The expert has access to the necessary tools and systems for analysis.
Main Success Scenario	1. The expert logs into the system. 2. The expert navigates to the section for bridge data analysis. 3. The expert selects the bridge they are interested in from the available list. 4. The expert retrieves the relevant data collected from the drones. 5. The expert analyzes the data, looking for key indicators of maintenance needs. 6. The expert compiles the findings into a comprehensive report. 7. The expert submits the report to the relevant parties for review and action. 8. The expert logs out of the system.
Extensions	• (1.a) If the expert cannot access the data, an error message is displayed, guiding them to contact support. • (5.a) If the analysis reveals critical issues, the expert is prompted to escalate the findings to the maintenance team.
Postconditions	A detailed analysis report is generated, indicating whether the bridge needs maintenance, along with recommendations for any necessary actions.
Related Requirements	F4, F5

1.2.3 UC-3 - Schedule New Bridge Inspection Mission

Primary Actors	User, Web Interface, System
Trigger	The need for a new bridge inspection is identified, based on maintenance schedules or inspections that are due.

Goal	Successfully schedule a new inspection mission for the bridge using the web interface.
Preconditions	• The user is logged into the web interface securely. • The inspection requirements and parameters are defined (e.g., bridge, date, time, and specific tasks).
Main Success Scenario	1. The user navigates to the scheduling section of the web interface. 2. The user selects the desired bridge from the list of bridges in the system. 3. The user inputs the inspection date, time, and any specific tasks or requirements. 4. The system verifies the availability of drones and regulatory compliance for the scheduled time. 5. The user reviews the scheduling information and confirms the mission. 6. The system schedules the mission and sends a confirmation to the user. 7. The scheduled mission is updated in the system and made visible to other relevant parties.
Extensions	• (3.a) If the user fails to input required fields, an error message prompts them to complete all necessary information. • (4.a) If no drones are available for the scheduled time, the system suggests alternative times or additional resources. • (5.a) If the user cancels the scheduling process, the system safely discards the input without making any changes.
Postconditions	The new bridge inspection mission is successfully scheduled, with all relevant data stored and accessible for future reference.
Related Requirements	F1, F3, F5

1.3 Quality Goals

This project adheres to multiple quality goals for the swarm intelligence system used in bridge inspection. The system must meet high standards in both software architecture and operational procedures to guarantee safe and effective autonomous inspections.

To achieve this, we align our quality goals with:

- ISO 25010 [7] (Software Quality Model) – Defines key software characteris-

tics (see Figure 1).

Figure 1: ISO 25010 Quality Characteristics
[7]

- ISO 21384-3 [6] (Unmanned Aircraft Systems) – Ensures safe and efficient drone operations (see Figure 2).

Figure 2: ISO 21384-3 Operations Standards
[6]

The following quality goals have been identified, listed in order of importance:

1. **Reliability** - The system must ensure high reliability in both software and drone operations. Drones must function without mission-critical failures, and software should be fault-tolerant to handle communication loss, sensor errors, or drone malfunctions. Compliance with ISO 21384-3 [6] ensures redundancy measures like emergency landing protocols and self-healing swarm behavior.
2. **Performance Efficiency** - The system must optimize real-time data processing to support high-speed defect detection and seamless drone

coordination. Efficient battery usage, low-latency data transmission, and AI-driven analysis must ensure smooth inspections without delays.

3. **Safety & Security** - Drones must operate safely within restricted airspaces, avoiding collisions, unauthorized areas, and hazardous weather conditions. Security measures should comply with ISO 25010 [7] to protect data transmission and prevent drone hijacking. Following ISO 21384-3 [6], the system must implement geo-fencing, encrypted communication, and secure authentication to prevent unauthorized interference.
4. **Maintainability** - The system must be easily upgradable to adapt to new AI models and software updates. A modular architecture ensures smooth maintenance and minimizes downtime.
5. **Scalability & Portability** - The system should support multiple drone models and hardware configurations, ensuring flexibility for different inspection sites. Data should be exportable in standard formats for compatibility with structural analysis tools used in civil engineering.

By integrating ISO 25010 [7] for software quality and ISO 21384-3 [6] for drone safety, this system ensures high-performance, reliable, and secure bridge inspections while complying with industry standards.

1.4 Stakeholders

The success of the envisioned system depends on various stakeholders, each with different roles, responsibilities, and expectations. Understanding these stakeholders is critical to ensuring that the system meets their needs, aligns with regulatory requirements, and integrates seamlessly into existing workflows. Key stakeholders of the system include all persons, roles, or organizations that:

- should know the architecture,
- have to be convinced of the architecture,
- have to work with the architecture or with the code,
- need the documentation of the architecture for their work,
- have to make decisions about the system or its development.

Table 5 lists the key stakeholders, their role in the system, and their expectations regarding the system and its architecture. The stakeholders are split into primary and secondary, while also being ordered by their overall importance to the system's development. Primary stakeholders are those who are directly involved in the system's development and use, significantly impacting its requirements, design, and success [5]. Secondary stakeholders, on the other hand, have an indirect or supporting role, influencing the system through regulatory requirements, integration with other systems, or providing additional resources and expertise [5].

Role	Type	Description	Expectations
Infrastructure Owners	Primary	Government agencies or private companies responsible for bridges	A reliable and cost-effective system that ensures accurate inspections and timely maintenance recommendations.
Bridge Engineers	Primary	Experts responsible for evaluating bridge conditions	High-quality inspection data with minimal manual effort, easy access to reports, and compliance with industry standards.
Drone Operators	Primary	Individuals or teams managing drone operations	A user-friendly system for swarm coordination, scheduling, and monitoring autonomous drone missions.
Software Developers	Primary	Engineers maintaining and extending the system	Well-documented, modular, and scalable architecture with clear API specifications.
System Integrators	Secondary	Companies integrating the system with existing infrastructure	Standardized interfaces and seamless compatibility with other monitoring or maintenance platforms.
Regulatory Bodies	Secondary	Organizations ensuring compliance with aviation and inspection laws	Compliance with legal and safety regulations for autonomous drone inspections.
Research Institutions	Secondary	Universities or labs advancing drone swarm technology	Access to data, testing opportunities, and collaboration on new automation techniques.

Table 5: Key Stakeholders and their expectations

To effectively address the diverse needs of our stakeholders, it is essential to quantify their concerns regarding the various quality attributes of the swarm management system. In Table 6, each stakeholder is allocated 100 points to distribute among the chosen quality attributes, prioritizing them accordingly. This systematic evaluation will guide our decision-making process, ensuring that we meet the most important expectations and requirements of all involved parties.

Stakeholder	Reliability	Security & Safety	Scalability & Portability	Performance Efficiency	Maintainability
Infrastructure Owners	40	30	10	10	10
Bridge Engineers	35	30	10	15	10
Drone Operators	30	25	15	20	10
Software Developers	20	15	25	20	20
System Integrators	25	20	25	15	15
Regulatory Bodies	30	35	10	10	15
Research Institutions	15	10	25	25	25

Table 6: Stakeholder Concerns and Points Distribution Table

2 Architecture Constraints

This section outlines the constraints that influence architectural decisions in this project, which may arise from technical, organizational, or conventional sources and must be considered throughout the design and development phases. Each constraint is documented with a brief explanation to clarify its impact on architectural decisions.

2.1 Technical Constraints

- **Browser Compatibility:** The interface must support major web browsers, including Chrome, Firefox, Edge, and Safari.
- **Scalable Web Hosting:** The interface should be hosted on a platform that supports high availability, fast access, and scalability to handle live requests alongside other hosted systems.
- **Data Management and Performance:** The system must use a performant database to manage multiuser logins, account handling, and real-time data storage, prioritizing fast access, secure data handling, and synchronized data from multiple drones.
- **System Integration and Security:** The platform must integrate with external systems, including data analysis tools, regulatory APIs, and legacy systems, ensuring secure authentication, data encryption, and compliance with data privacy standards.

- **Drone Deployment Limitations:** Consider hardware-specific constraints such as drone battery life, GPS and sensor accuracy, signal range, and autonomous capabilities for launching, navigation, and landing.
- **Swarm Coordination:** The system must support efficient communication and coordination between multiple drones to ensure optimal coverage and collision avoidance.
- **User Interface and Experience:** Design an intuitive user interface that allows operators to easily monitor and control drone swarms, view inspection data, and generate reports.

2.2 Organizational, Political, and Regulatory Constraints

- **Compliance with Regulations:** Drones must operate within the legal frameworks of the inspection areas, including obtaining necessary flight clearances.
- **Data Management Policies:** Inspection data must be stored and managed according to organizational data governance policies.
- **Time Constraints:** The project must adhere to specified timelines for development, testing, deployment, and the regular inspection schedule.
- **Budget Constraints:** Financial limitations must guide technology choices, resource allocation, and project scope.
- **Resource Availability:** The project must account for the availability of skilled personnel, hardware, drones, and development tools.

2.3 Conventions and Guidelines

- **Coding Standards:** Follow organizational coding conventions, including naming conventions, documentation practices, and versioning guidelines.
- **Documentation Standards:** All architectural and design documentation must comply with internal documentation standards, ensuring clarity and consistency.
- **Design Principles:** Adhere to modular and scalable design practices to facilitate future updates and integrations.

3 Context and Scope

In this section we discuss the context and scope of the system. We will document what delimits ours from its communication partners from both a business perspective 3.1 and a technical perspective 3.2.

3.1 Business Context

The business context specifies all communication partners (users, IT-systems, ...) that interact with the Swarm Intelligence system. In Figure 3, a business-driven context diagram is given.

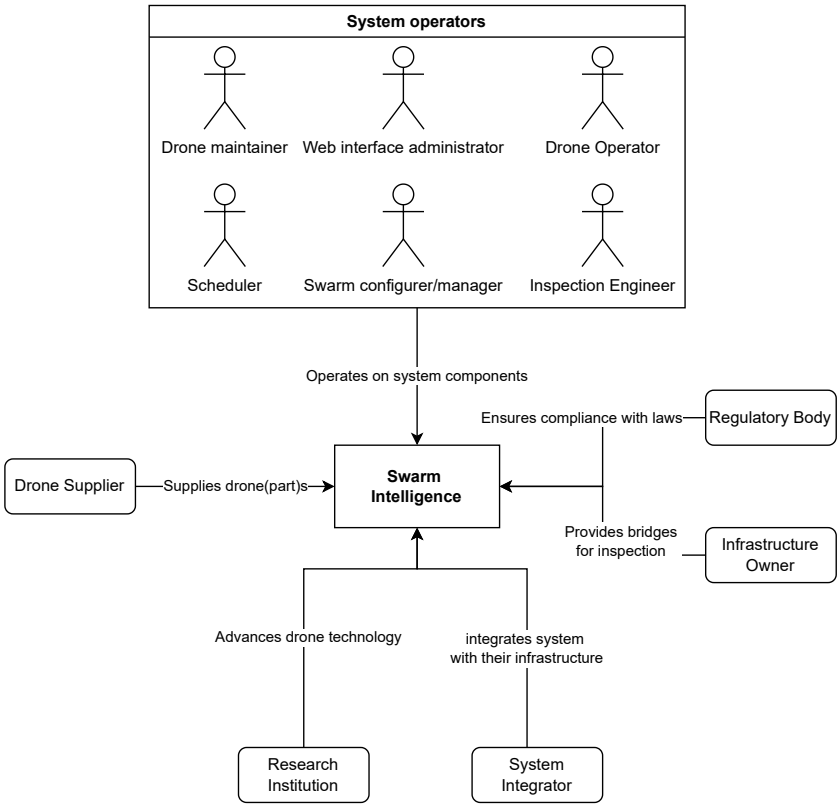


Figure 3: Business Context Diagram

Figure 3 is detailed in Table 7.

Communication Partner	**Description**
Drone Supplier	When drones or their components require replacement, or additional drones are needed, purchases need to be made. Maintaining direct communication with the drone supplier ensures quick resolutions for drone issues.
Infrastructure Owner	Infrastructure owners are in charge of the maintenance of bridges. This could be either the government or private bridge owners. Through the web interface, bridge management can request that a bridge needs to be analyzed before a specific deadline. The system provides infrastructure owners with detailed reports about the state of their bridges.
Regulatory Body	These are organizations or government agencies responsible for ensuring compliance with aviation and inspection laws. The system should provide regulatory bodies with information about the drones and their flight trajectories.
System Integrator	System Integrators are third party organizations that must integrate their existing infrastructure with the system. The system provides these system integrators with data about analyzed bridges through a central API they can access.
Research Institution	Research institutions are responsible for progressing drone swarm technology. They need access to data, testing opportunities, and the expect collaboration on new automation techniques.

Table 7: Business Context Element Description

Figure 3 also includes a list of new employee roles required for the system to operate. These roles are detailed in Table 8.

Employee type	Description
Drone maintainer	The drone maintainer works in warehouse management and is responsible for everything related to maintenance of the drones. For example, when a drone has a broken wing that needs replacement, the drone maintainer needs to provide the drone with a new wing.
Web interface administrator	The web interfaces administrator is responsible for managing access to the web interface. This includes providing roles to different users that require different tools of the web interface.
Drone operator	The drone operator are individuals or teams managing drone operations. When the drone analysis can't be fully automated, drone operators need to deploy the drones manually on site.
Inspection engineer	Inspection engineers are responsible for evaluating the conditions of bridges. They need to use the web interface to generate reports based on the information given by drones.
Scheduler	The scheduler schedules swarms for scanning missions.
Swarm configurer/ manager	The swarm configurer (or swarm manager) uses the web interface to configure drones into a swarm that schedulers can than select for scanning missions.

Table 8: System operators

3.2 Technical Context

The technical context specifies all technical interfaces (channels and transmission media) linking Swarm Intelligence to its environment. In Figure 4, a context diagram is given, including which transmission protocols are used between the communication partners and Swarm Intelligence.

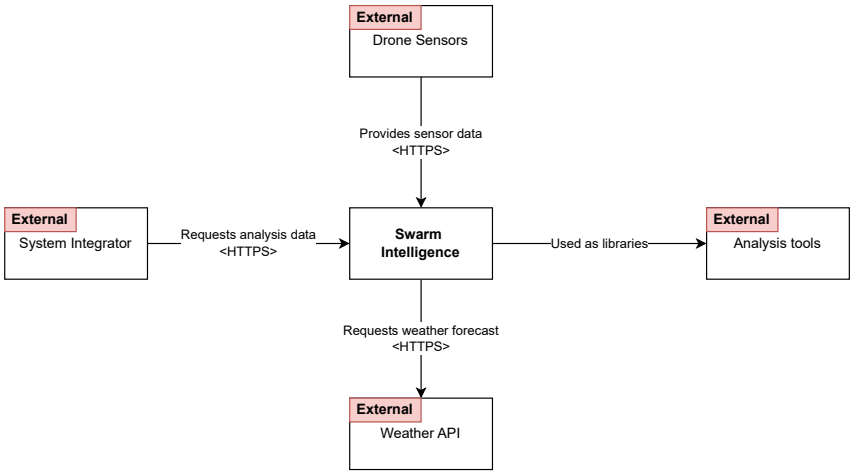


Figure 4: Technical Context Diagram

Figure 4 is detailed in Table 9.

Table 9: Technical Context Element Description

Communication Partner	**Description**
Drone Sensors	The drone sensors collect data about the state of bridges, which will eventually be transmitted to the central analysis database.
Weather API	Since weather forecasting is a very complex task, we will need a weather API to determine whether or not the weather on a certain date is acceptable for bridge analysis.
System Integrator	System Integrators are third party organizations that must integrate their existing infrastructure with the system. The system provides these system integrators with data about analyzed bridges through a central API they can access.
Analysis tools	The analysis tool uses third-party analysis libraries for generating reports.

4 Solution Strategy

In this section, we will go through our solution strategies to achieve our quality goals mentioned in Section 1.3. Table 10 presents various types of solutions, including technology choices, architectural and design patterns, and organizational decisions. Section 9 will provide further explanation of these decisions in relation to our business goals. Additionally, quality attributes will be discussed in detail in different documents.

Quality Goal	Scenario	Solution
Reliability	Mission-critical failure during drone operation.	Redundant hardware (dual IMUs, ESCs), fault-tolerant firmware, and real-time health monitoring via Prometheus/Grafana.
Reliability	Drone malfunctions mid-flight.	Self-healing swarm behavior (ROS 2) with task redistribution and failure detection via MAVLink heartbeats.
Reliability	Communication loss or sensor errors during data transmission.	Onboard SSD/microSD storage with MQTT/ROS 2 store-and-forward for delayed transmission.
Reliability	Emergency landing required.	Automated landing using GPS failsafe and LiDAR terrain awareness (ISO 21384-3 compliant).
Reliability	Drone fails to return post-mission.	GPS tracking with LTE/5G fallback, geofencing alerts, and "return-to-home" failsafe.
Performance Efficiency	Battery drain during long missions.	Reinforcement Learning (RL)-based power management and OpenCV-optimized flight paths.
Energy Efficiency	High power consumption during sensor operation.	Dynamic power management for sensors and communication modules using adaptive algorithms.
Safety	Drone detects obstacles during flight.	Obstacle avoidance using LiDAR and computer vision (OpenCV).
Security	Data transmission intercepted.	TLS 1.3 encryption over MQTT with HSM-backed key management.
Security	Unauthorized access to the web interface	Multi-factor authentication (MFA) has to be added to the web interface. Also, Roll Based Access Control (RBAC) should enforce users to only be able to access tools meant for them.
Security	Unauthorized access to drone controls.	Multi-factor authentication (NFC/RFID) and ML-driven intrusion detection (ISO 21384-3).
Maintainability	System downtime during maintenance.	Automated CI/CD pipelines (GitHub Actions, ArgoCD) with A/B-tested AI model updates.
Maintainability	Updating the system's software or AI models	Modular microservices (Docker/Kubernetes) enable easy AI model and firmware upgrades.
Usability	Operator control and monitoring complexity.	Responsive UI with WebRTC real-time video streaming (ISO 25010 compliant).
Testability	System validation under edge cases.	PyTest/Selenium unit/integration tests and Gazebo drone simulations.
Scalability	Integration of new drone models.	Hardware abstraction layer (HAL) via PX4/ArduPilot APIs.
Portability	Compatibility with civil engineering tools.	GeoTIFF/IFC/JSON exports for AutoCAD/QGIS integration.
Portability	Third-party system integration.	RESTful/gRPC APIs with OpenAPI specifications.

Table 10: Solution Strategies Overview

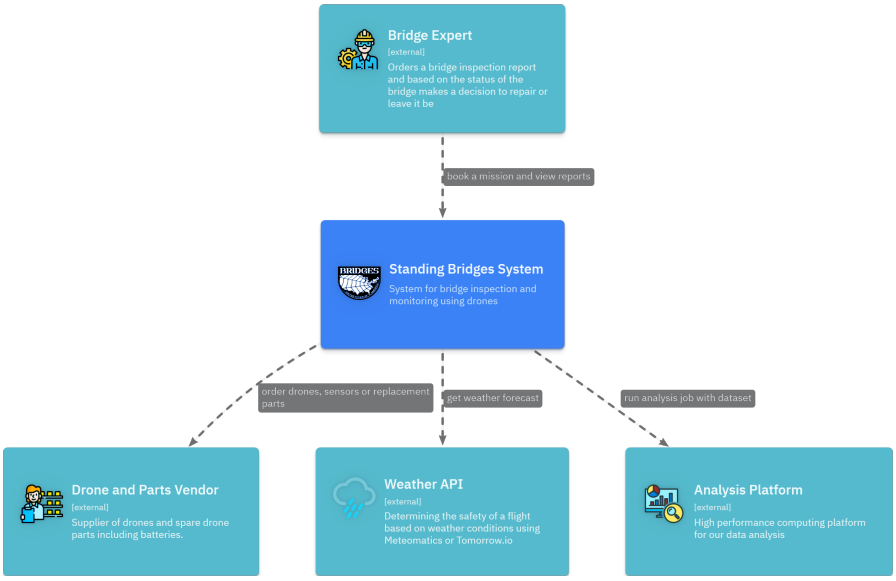


Figure 5: (Level 0) External System Interactions

5 Building Block View

In this section, we present a structured decomposition of the system architecture at different levels of granularity. A digital version of the C4 Model presented can be explored online (see [11] for more details).

Key components of the Standing Bridges system, their responsibilities, and interactions are modeled according to the C4 Model. Following are the diagrams of the C4 Model, split into three levels. First, a broad overview of the system its context is shown—called level zero (5.1). Level one (5.2) shows a decomposition of the system into its primary subsystems and their interactions. In level two (5.3) we further break down the subsystems, and, finally, in level three (5.4) we analyze a few of the most important components.

5.1 System Context

At the highest level, Figure 5 illustrates the system's interactions with external actors and services. The Bridge Expert, our primary customer, books inspection missions to evaluate and manage bridges using our swarm intelligence solution. Once a mission is initiated, the system autonomously handles scheduling, flight operations, and data analysis. Replacement drones and parts are sourced from the Supplier. Weather forecasts from the Weather API ensure safe conditions, boosting cost efficiency and performance. Finally, the Analysis Platform handles large-scale data processing, offloading intensive computing tasks from our system.

We define the external actors as follows:

Bridge Expert Possesses domain knowledge and acquires our services to perform bridge inspections using our swarm intelligence solution.

Drone and Parts Vendor Provides the necessary hardware components and replacement parts to ensure the continuous operation of our system infrastructure.

Weather API Supplies weather forecasts, which are critical for planning and scheduling inspection missions. Accurate weather forecasts for the area surrounding the scheduled bridge inspections are essential, as our drones are highly sensitive to weather conditions. Operating under suitable conditions ensures cost efficiency and optimal performance.

Analysis Platform Offloads large-scale data processing, enabling efficient handling of high-volume inspection data while reducing our computational overhead.

5.2 Level 1

In level one we analyze and decompose the system into its primary subsystems which represent bounded logic of the system.

5.2.1 Decomposition

We decompose the system standing bridges into four distinct subsystems in figure 6.

We start with mission management which tracks all necessary information about missions. This is where the customer orders a new inspection. Mission management then handles all the necessary steps to fulfill a mission including reserving drones and storing the resulting report from the analysis. All drone operations are managed within the warehouse management system. this is where all of the drones are charged to maintain and managed. When drones are deployed on a mission they are managed by the deployment subsystem. Here we represent all the on-site operations that are necessary during mission or inspection execution. The results of any mission go into the data analysis subsystem where the final data is analyzed by by an analyst and process further into a final report for the customer.

The responsibilities of each subsystem are as follows:

Mission Management Handles the planning, execution, and monitoring of bridge inspection missions. It interacts with the Weather API to ensure safe flight conditions and provides mission templates for the Mission Planner. It also coordinates with the Drone Operator for specialized missions.

Data Analysis Processes and analyzes data collected during missions. It provides interfaces for data interaction and reporting, enabling the Data

Analyst to generate detailed reports for the Bridge Expert, who uses them to make informed decisions about bridge maintenance.

Deployment Manages on-site operations, including mission execution, environmental monitoring, and safety assurance. It interfaces directly with the Drones and provides real-time monitoring capabilities for the Drone Operator.

Warehouse Management Centralizes the management of drone inventory, charging, and maintenance. It coordinates with the Drone Technician for repairs and the Supplier for replacement parts, ensuring drones are always operational and ready for deployment.

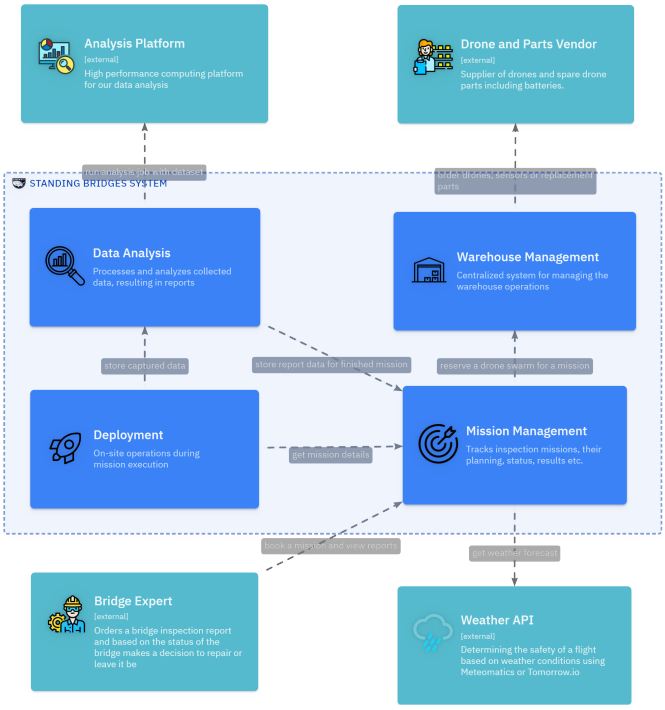


Figure 6: (Level 1) System Decomposition

5.2.2 Internal Actors

In this view we can see the internal the interactions between internal actors and the subsystems. For the mission management we have an important actor called Mission Planner who analyzes and creates templates for inspections. This person is knowledgeable in how to analyze bridges and can help create paths and tasks or objectives for the drones to fulfill.

The warehouse management system has interaction with three actors which have a role within the warehouse. First the warehouse manager is responsible for overseeing the warehouse as a whole and as such is gathering information about the status of the operations within the warehouse. Second the logistics coordinator is the person who oversees the transactions or the orders between the warehouse and the external vendors as well as managers internal tasks and distribution. Lastly the drone technician is an operator who operates on drones whatever that means note whatever that means but is a repair person is an expert is an engineer is a drone engineer who repairs maintains and overseas that the drones are in good working order for missions.

For the data analysis system we have a specialized data analyst whose job and responsibility is to analyze the collected information and images and all the data about the bridge and concatenate or analyze them in a way where they in the end output a report which is the final product of our system ready for the customer.

During a mission deployment we have the drone operator who is there to overlook the operations of the swarm and if necessary can take manual control or cancel the mission. Lastly we have a drone which is an unmanned autonomous flying vehicle which is used to gather and collect data about the bridge and perform the inspection itself.

5.3 Level 2

In this section we decompose the internal structure of each subsystem deeper, breaking them down into their key components.

5.3.1 Mission Management

The mission management subsystem has a narrow scope in terms of interactability with the rest of the system. We see, in figure 8, that the most important part is the ui, which serves as a connector between the customers, the mission planners and the rest of the system as a whole. This is a component accesses the information in the rest of the subsystem through a gateway. which serves as a. point for interaction between all the other sub systems. One of the key components is the mission processor, which handles all the state changes and interactions of the missions, making sure that each and every mission goes through the pipeline as necessary without getting into a bad state. the state of these missions is saved in the missions database.

5.3.2 Data Analysis

The data analysis subsystem, shown in figure 9, follows a similar structure to the mission management subsystem. The central component is the job processor, which, again, fulfills the job of processing all the data and handling all the state of the process pipeline. The results of a mission are stored in

Figure 7: (Level 1) Internal Actors interactions

inspection data database, which has processed data coming in from the data in just component. Then the data analysis uses the analysis interface to create a job which will be then handled by the job processor, which will send it to an external analysis platform for faster computation. Lastly, when the data analyst has the successfully found and processed all the data that they need, they create a report which is appended. into the mission through the mission management system. The data analysis system automatically pushes the report. data for the given mission into the mission management system.

5.3.3 Deployment

The deployment subsystem is shown in figure 2. also contains a UI component for the drone operator Here they can see and analyze information about the life status of the mission deployment and how the drones are doing how the swarm is performing. The mission handler acts as a gateway for all the other components

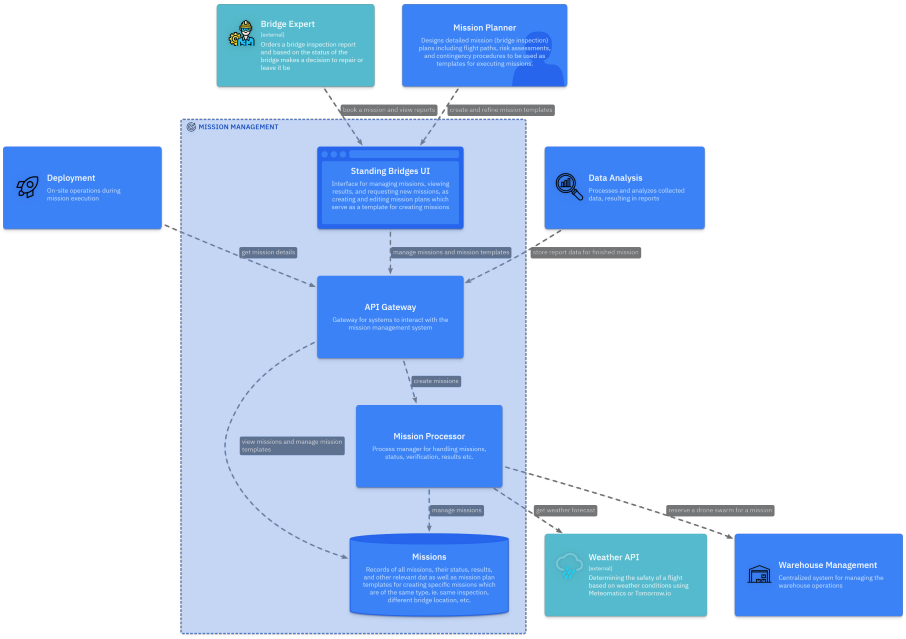


Figure 8: (Level 2) Mission Management

where the deployment is split into two similar components, subcomponents They are mission operations and bridge inspection. The next section will go into further detail of what these two do. However, we can say that both of these just encapsulate. the handling of data coming in from the drones where a mission operations deals with the operational data, which means logs, etc. And the bridge inspection operates or handles the data which is the business data, meaning the capture data about the bridges, etc. Before the mission is deployed, the mission handler gathers mission details where it gets all the tasks necessary for the drones to fulfill. And then. handles the events needed to be fulfilled during the operation itself. During the mission, the drone operator can easily check how the mission is performing or how the drones are performing on the live dashboard. Lastly, when the deployment is finished and the mission has concluded, the capture data is stored in the data analysis subsystem.

5.3.4 Warehouse Management

Managing warehouse proved to be one of the more difficult areas of this design we have the most actors here and also many different processes which are present which we need to take care of as such we split the warehouse operations into three different subsystems the inventory charging and drone maintenance Drone maintenance schedules logs and monitors maintenance activities charging is about maintaining the charging infrastructure for the drones while they are in

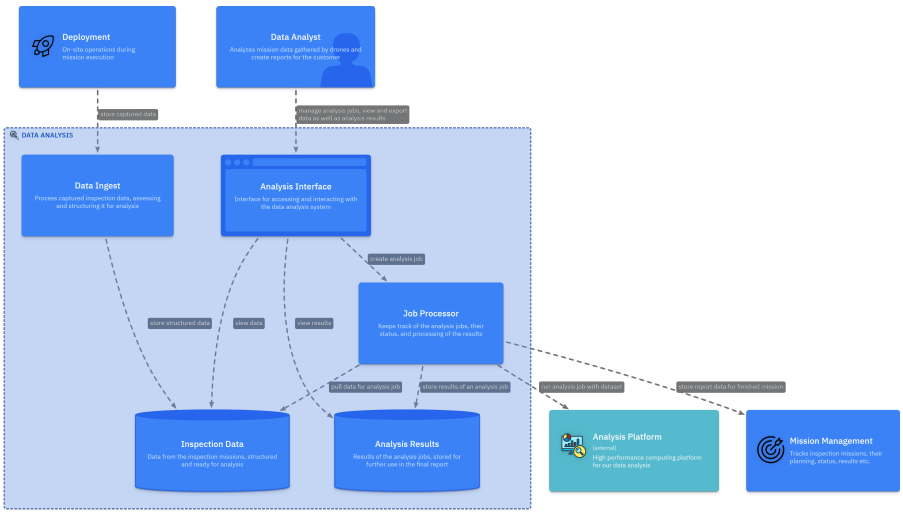


Figure 9: (Level 2) Data Analysis

between missions and inventory is about providing and managing the inventory in the warehouse itself which includes the spare parts batteries drones etc because of this we have two different user interfaces which fulfill different functions. The operations ui is concerned with managing the internal operations of the warehouse including charging schedules maintenance tasks and charging schedules as well as inventory status. While the management ui is an interface for managing the warehouse operation which is used by the warehouse manager and logistic coordinators to look at orders and energy information.

Communications between the components is modeled as communications between the processes in the warehouse itself and as such we decided to use an event bus which transmits and publishes all the events happening within the system so that the subsystems or sub components can interact with each other asynchronously and all the information is published to any component which may need it.

5.4 Level 3

In level three the most detailed level of them all we look into specific and interesting parts of the system which are necessary to design a little bit more deeply.

5.4.1 Warehouse Management

First we dissect 3 the three subsystems subcomponents or whatever in the warehouse management subsystem. The subcomponents don't have a user interface since the warehouse management subsystem has to user interfaces on

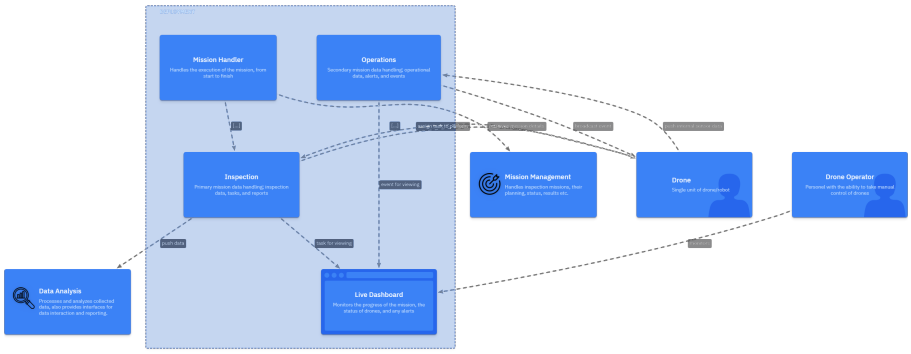


Figure 10: (Level 2) Deployment

the top level which gather data and operate on the subsystems themselves.

Warehouse Inventory The inventory component is a trivial example of management system which handles events and stores them in a database called inventory items. This component subscribes to the maintenance request drawn status charges and drawn allocation since it handles what in what status are the parts or any kind of inventory inside the system. Finally it also places orders for drones and replacement parts which are out of stock or go below a certain threshold. All this information is available in the management ui of the warehouse management system.

Maintenance In the maintenance component we have another database which stores maintenance records histories and task assignments and the maintenance scheduler which handles the logic between managing the status of the tasks and their logical division and allocation. The component subscribes to maintenance requests to create maintenance tasks and to update the drone status since we can update what happened with the drone and update the inventory with the parts we used or didn't or returned or they're broken. All of this is managed by the operations UI in the warehouse management system.

Charging Room charging in the warehouse management system is divided into charging schedule which determines optimal charging times and the charging monitor which monitors the charging stations and the drones themselves and triggers events based on what it analyzes occurs during charging. Drone charging is an integral part of the warehouse management system and because of that we keep battery records in the charging subsystem so that we can better schedule and understand how the drones operate because we manage their lifetime operation. Finally the charging monitor sends events to the rest of the warehouse management system by a maintenance request about swapping battery or it will update that battery status information in the inventory system so that it can

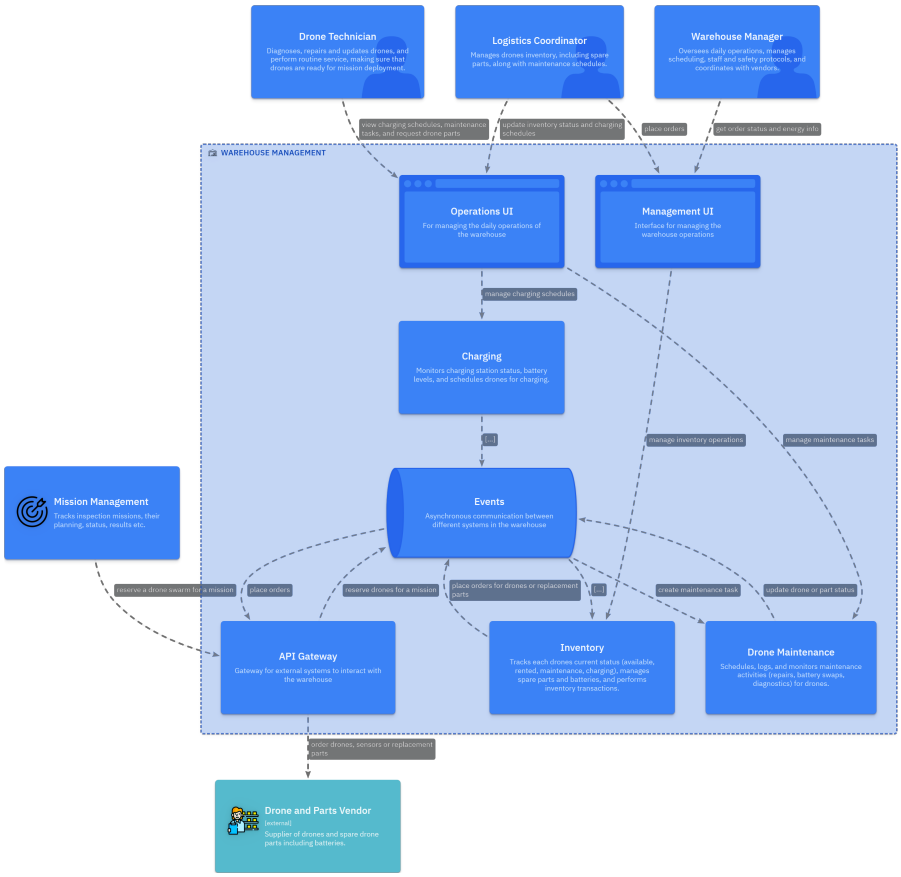


Figure 11: (Level 2) Warehouse Management

order a new set of batteries if these ones are dying.

5.4.2 Deployment

In this section we dissect the deployment subsystem into its bridge inspection and operations data components. Information from both of these components is available for the mission handler which then forwards it to the user interface for the drone operator to be able to see during the actual deployment.

Inspection The inspection component handles bridge inspection data collection and as such it has interacts with the drone from the drone it gathers the captured data into sensor data queue which is then ingested and finally analyzed in the ingest into additional tasks which can be added if a special crack or maintenance or something that we want to depress more deeply analyze is

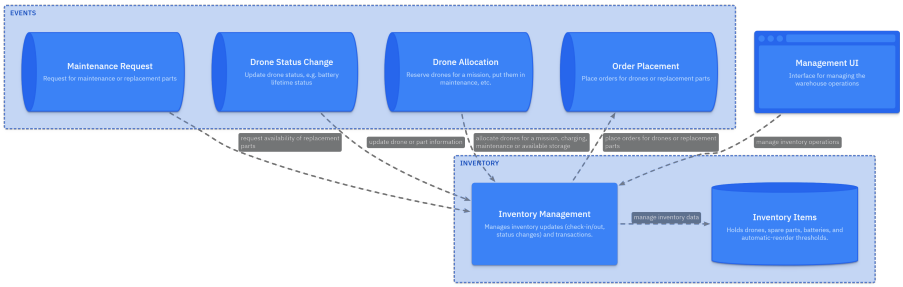


Figure 12: (Level 3) Inventory subsystem of the Warehouse Management system

present and alongside everything it will store this data into the bridge data database for further analysis in the data analysis subsystem. The mission handler manages the operations within this component fetching the capture data and pushing missing tasks which are grabbed from the initial mission.

Operations In contrast with the bridge inspection component the mission operations is very similar except for handling non mission critical information. That means that here the drones don't push capture data however they capture sensor information like temperatures or other logging information which is necessary to keep a good running drone swarm operating. This information is then analyzed in the log ingest and it can also push an event into the drone events to propagate to the drones in case some drum will fail or exhibit nonstandard operating variables.

6 Runtime View

The runtime view describesthe concrete behavior and interactions of Standing Bridges' building blocks in the form of scenarios. These scenarios are as follows:

- **The log-in process on the web interface.** The log-in process on the web interface is a crucial scenario to ensure the reliability quality attribute.
- **The mission planner planning a swarm for deployment.** This scenario kicks off the entire drone deployment process (see UC-3 in Section 1.2).
- **The Expert analyzing a bridge** This scenario is the final step before providing infrastructure owners with an analysis of their bridge (see UC-2 in Section 1.2).

In Figure 17, a detailed sequence diagram is given for the user logging in. The validation of user input is fairly standard in the industry. Our system uses user groups to define which roles a user can have. What is not specified in the sequence diagram is that an administrator can give out rights to any user

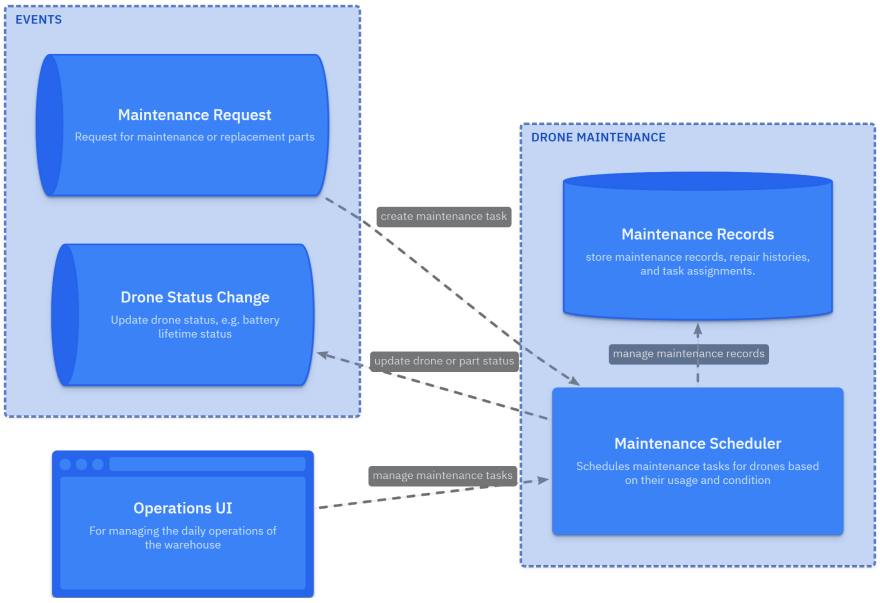


Figure 13: (Level 3) Maintenance subsystem of the Warehouse Management system

wanting to access the web interface. When a user has the **planner** role, he will have access to the mission planner tab in the web interface. But the user might also have the **analysis** role, giving him access to the analysis tool in the web interface.

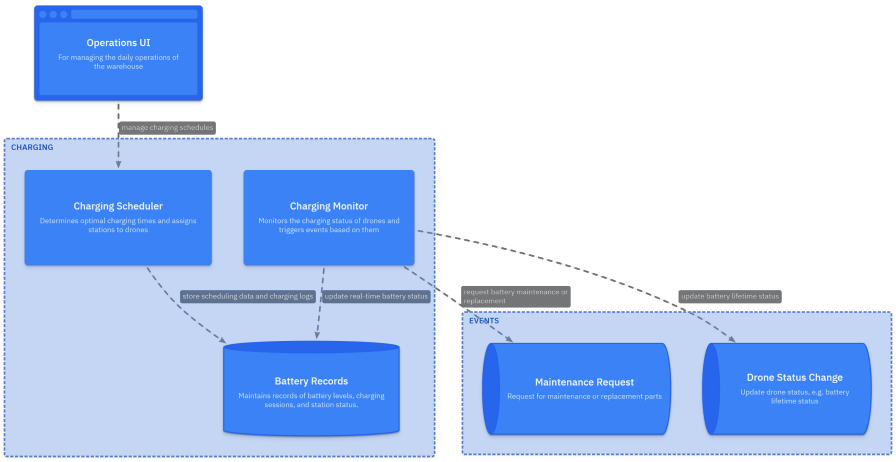


Figure 14: (Level 3) Charging subsystem of the Warehouse Management system

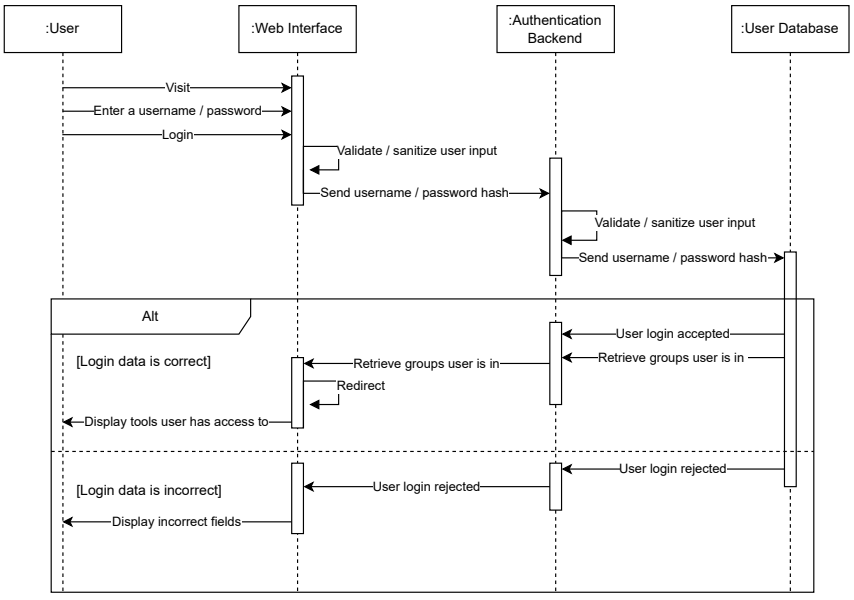


Figure 17: Sequence diagram of a user logging into the web interface

In Figure 18, a detailed sequence diagram is given for a Mission Planner (user) scheduling a drone for deployment. The preconditions for the diagram are that the user has the **planner** role assigned to them and that they are logged in

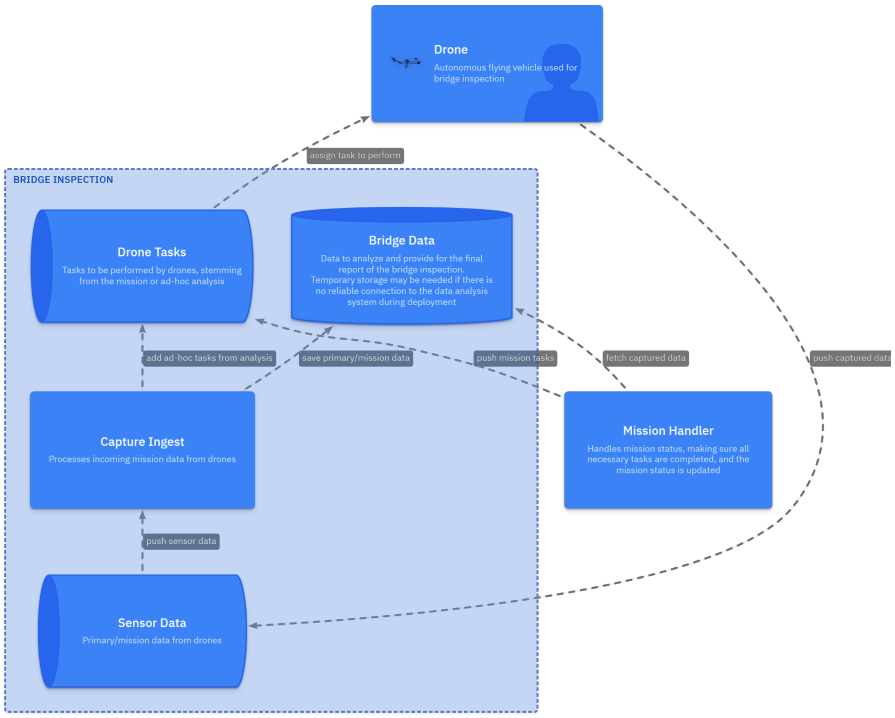


Figure 15: (Level 3) Inspection subsystem of the Deployment system

on the web interface (the sequence diagram for this is specified in 17.) The qualified swarms in the planTemplates database need to be updated periodically or whenever a swarm is configured, giving the user the most current state of the swarms. What is not shown in the diagram, but is important at every retrieval of items from the database, is that when an error occurs, the error is logged, and a clear message is given to the user. Figure 22 gives a mock-up of the scheduling tab.

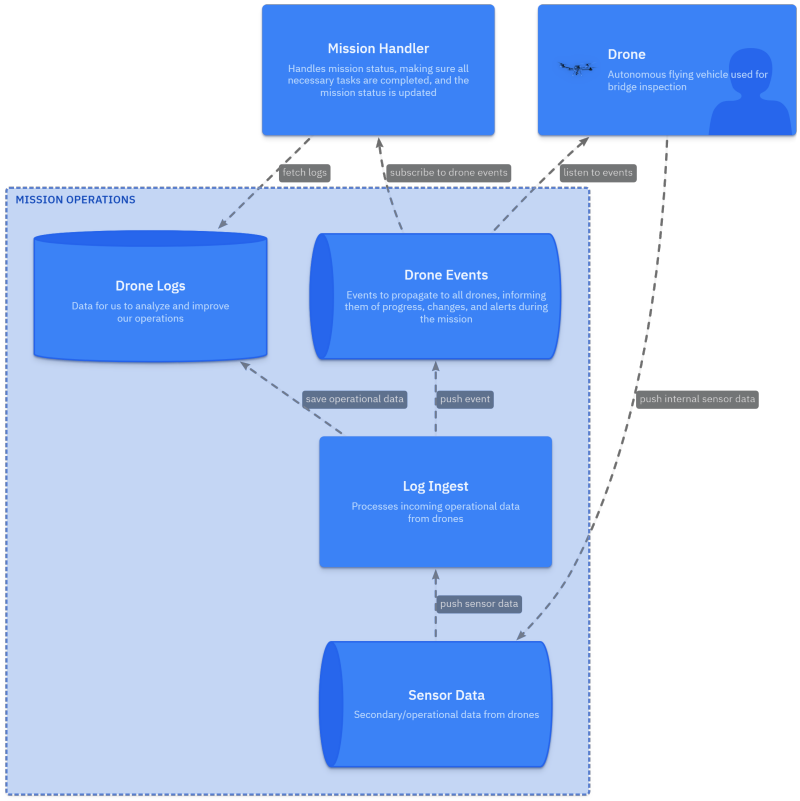


Figure 16: (Level 3) Operations subsystem of the Deployment system

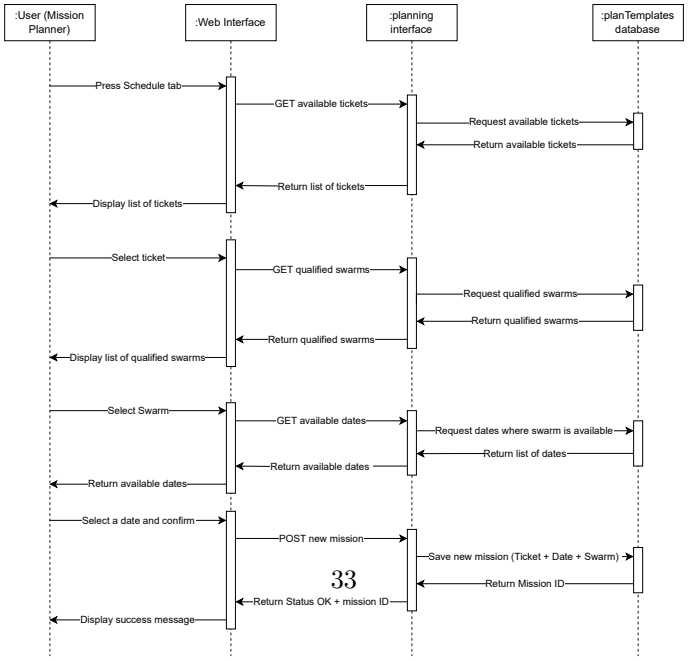


Figure 18: Sequence diagram of a scheduler scheduling a swarm using the web interface

In Figure 19, a sequence diagram is given describing an expert analyzing bridge scanning data. This diagram looks similar to the sequence diagram in Figure 19. Here, the user also opens a tab in the web interface, which displays a list of options to choose from. The data analyst does not have to do all analysis by themselves. A large part of the analysis can be done by third-party analysis libraries. Then, the scheduler backend generates a prototype report based on the raw data and the output of the analysis library, which will either be displayed to the data analyst or can be downloaded. This report can then be manually edited by the data analyst and, when finished, can be submitted to the relevant stakeholders.

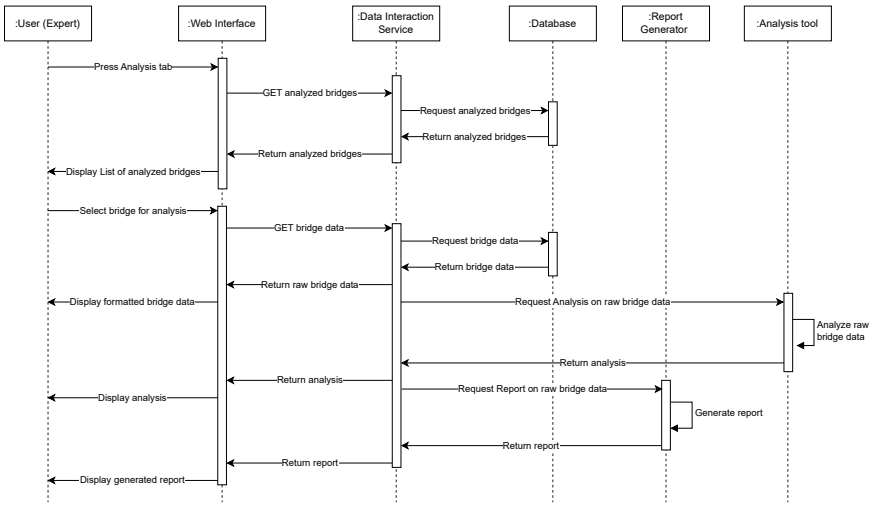


Figure 19: Sequence diagram of a data analyst analyzing the state of a bridge.

7 Deployment View

This section describes the technical infrastructure and deployment architecture of the autonomous swarm management system, detailing how software components map to hardware nodes, communication protocols, and environments. Figure 20 provides a visual overview of the deployment, illustrating the interaction between cloud services, edge devices, drones, and the warehouse system.

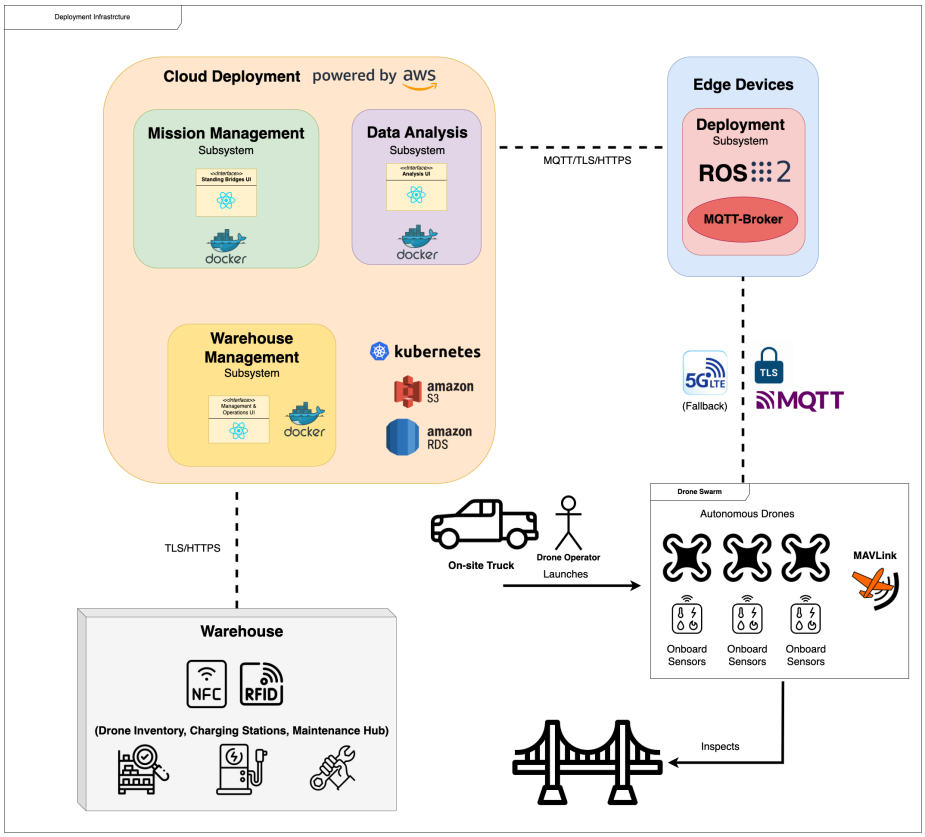


Figure 20: Deployment View

The system is deployed as a hybrid cloud-edge solution to ensure scalable data processing and real-time operations. Cloud-based services host the Mission Management, Data Analysis, and Warehouse Management subsystems using Kubernetes on AWS, with Docker containers for modular, scalable deployment. Amazon RDS and S3 provide scalable, secure storage for inspection data and logs.

On-site edge devices, housed on the on-site truck near inspection sites, execute the Deployment subsystem, interfacing directly with the drone fleet to guarantee low-latency mission execution, real-time monitoring (via ROS 2), and immediate safety responses. The on-site truck serves as the launch hub for the drone swarm, equipped with ruggedized computers to manage MAVLink for drone-to-drone coordination and MQTT over WiFi (with LTE/5G fallback) for telemetry and command execution.

Drones are equipped with onboard hardware and sensors, including:

- LiDAR for 3D mapping and obstacle detection
- High-resolution cameras for visual inspections
- GPS modules for precise navigation
- Inertial Measurement Units (IMUs) for stability
- Obstacle avoidance sensors for collision prevention
- Onboard SSD/microSD for local data storage
- Dual IMUs for stability and redundancy
- Redundant ESCs for motor control and fault tolerance

The Warehouse Management subsystem oversees drone inventory, charging, and maintenance in the warehouse facility. RFID/NFC technology is used within the warehouse to:

- Authenticate drones during docking (RFID), and personnel physical access (NFC)
- Track spare parts and battery inventory
- Update maintenance logs in real time

Security is maintained through centralized OAuth 2.0–based authentication and Role-Based Access Control (RBAC) for the web interface, ensuring that bridge engineers, drone operators, and administrators access only permitted functionalities. Dual communication channels (MQTT over WiFi with LTE/5G fallback)) and onboard fail-safes (e.g., emergency landing protocols) enhance fault tolerance and operational continuity.

This multi-tiered deployment architecture supports fault tolerance, scalability, and high availability while aligning with ISO 25010 (software quality) and ISO 21384-3 (drone operations) standards. It ensures seamless autonomous drone swarm operations for bridge inspections in diverse and challenging environments.

8 Cross-Cutting Concepts

In the process of designing complex system architectures, some principles and concerns extend across multiple components, influencing the overall functionality. These cross-cutting concepts are essential for maintaining consistency, coherence, and structural integrity throughout the entire system, and by consolidating them in one location, we can ensure they are easily referenced and maintained. This section highlights key guidelines and solution strategies that apply broadly across different system elements, such as (but not limited to):

- Architecture and design patterns
- Rules for utilizing certain technologies

- Principal technical decisions
- Implementation guidelines

Figure 21 shows the main groupings of the different cross-cuttings concepts that will be discussed in the remainder of this section.

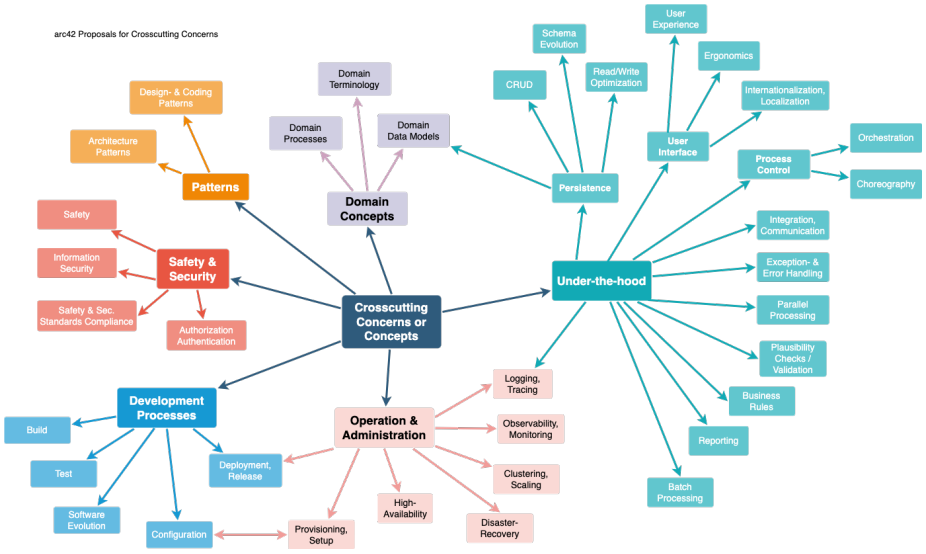


Figure 21: The structure of cross-cutting concepts [4]

8.1 User Experience Concepts

Providing an exceptional user experience is paramount for keeping user satisfaction high. We aim to make the interaction with the system fast, intuitive, and user-friendly for all stakeholders, including bridge engineers and maintenance personnel. To achieve this goal, we have designed the system with a strong focus on ergonomics and ease of use.

One of the key features that enhance user experience is our web-based application for scheduling inspection missions. The application provides an interface that allows users to plan inspection missions efficiently. Figure 22 displays the initial interface design of the scheduling web application.

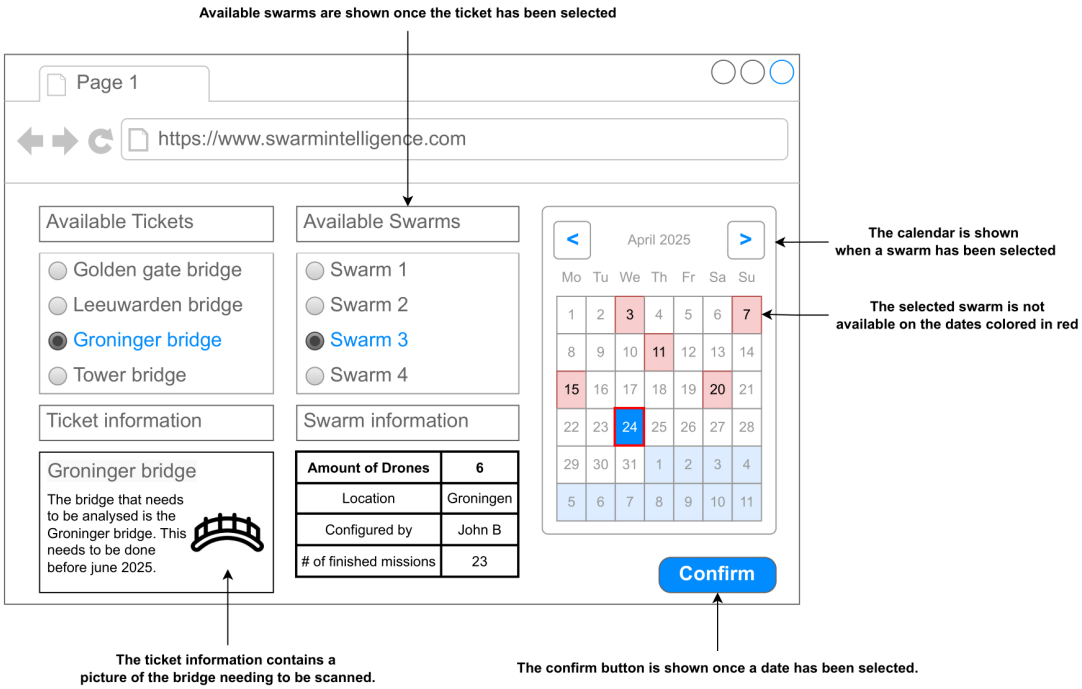


Figure 22: The UI of the Scheduler Page of the Web Interface

As mentioned before, the scheduling page is designed to be user-friendly and straightforward. Users can select a bridge to be inspected from the "Available Tickets" section, which displays relevant information, including a picture of the bridge and the analysis deadline. Once a ticket is selected, users can choose an available swarm from the "Available Swarms" section. This section provides all the relevant information regarding a swarm, such as the number of drones, location, configuration details, and the number of completed missions. The availability calendar, shown when a swarm has been selected, ensures that users can easily find suitable dates for the inspection, with unavailable dates highlighted in red. Once the user has decided on a date as well, the "Confirm" button will create and register the inspection mission.

8.2 Safety & Security Concepts

The autonomous drone swarm system should implement robust safety and security measures, which are important for maintaining data integrity, protecting sensitive information, and preventing unauthorized access. As a consequence, the system incorporates multiple security mechanisms to safeguard authentication, access control, data confidentiality, and operational safety, which are described in the list below.

- **Authentication and Access Control**
  - The system enforces authentication for all users interacting with the platform, including the data analysis and mission planning ones.
  - Only administrators and system operators can modify core system configurations, including security policies, swarm coordination settings, and emergency response protocols.
  - Drones participating in a bridge inspection must be verified and authorized by the system to prevent rogue or compromised units from joining the swarm.
- **Accountability and Auditability**
  - The system logs all access and control activities to ensure accountability. These records include user logins, mission changes, manual drone overrides, and database connections.
  - Each drone within the swarm is uniquely identifiable, allowing the system to track and monitor their activities.
- **Confidentiality and Data Protection**
  - To comply with GDPR and other data protection regulations, the system minimizes the collection of personal data, retaining only mission-critical information.
  - All communications between drones, the control platform, and cloud storage are encrypted using secure communication protocols to prevent unauthorized interception.
  - All inspection data, including high-resolution images and structural analysis reports, are stored securely and are only accessible by the intended recipients or administrators in case of need (for example, in the case of security investigations).
- **Operational Security Measures**
  - The system continuously monitors for potential security threats, including unauthorized drone activity, signal jamming attempts, or anomalies in swarm behavior.
  - In the event of a security breach, predefined countermeasures—such as emergency shutdowns or re-routing of drones—can be activated to mitigate risks.

8.3 Architecture & Design Patterns

For our autonomous drone swarm for bridge inspection system, we have adopted the Event-Driven Service-Oriented Architecture (ED-SOA) design pattern. This pattern is particularly well-suited for systems that need to respond to real-time

data and events, while also maintaining modularity and reusability through service orientation. For example, in an ED-SOA, the drones can act as event publishers, generating events related to their operations such as flight status, anomaly detection, and data collection. These events are then processed in real-time by various system components through an event bus, enabling immediate reactions and updates, as seen in Figure 23.

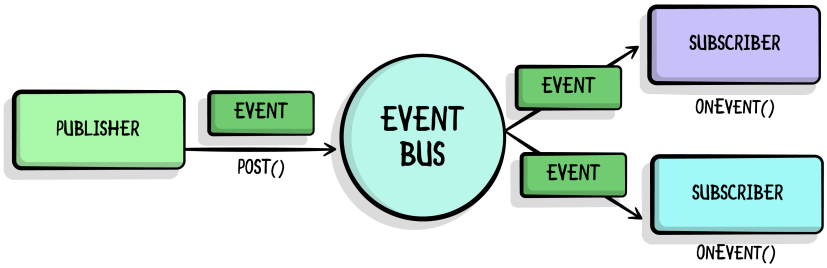


Figure 23: Event Driven Architecture Diagram [9]

The ED-SOA design pattern enables loose coupling between services, enhancing the system's flexibility and scalability. By communicating through events rather than direct calls, the architecture allows individual services and components to be updated or replaced without affecting the entire system [10]. This modular approach supports the high level of responsiveness required for drone coordination and data processing. Key components such as the central control platform, data processing units, and monitoring dashboards can act as event consumers (subscribers), reacting to the events generated by the drones and providing real-time insights and actions. Moreover, the fault tolerance and resilience of our system are also improved, since the modular architecture prevents issues from cascading and impacting the entire system.

8.4 "Under-the-hood" Concepts

At a high level, our system is designed around a set of core domain components, each built using modern and popular technologies. This subsection provides an overview of the foundational technologies that power our system, focusing more on the key tool choices rather than specific implementation details or optimization strategies, which are described in more detail in section 4. The list below highlights the major aspects of our system and the industry-standard technologies used to implement them:

- **Frontend**: The mission planning and data visualization interface is developed as a Single Page Application (SPA) using React, ensuring a responsive and interactive user experience.

- **Backend**: A Node.js-based API serves as the core backend of our Web Application, handling mission requests, data viewing requests, and authentication.
- **Database**: We utilize PostgreSQL with PostGIS extensions for geospatial data processing, enabling accurate mapping of bridge structures and drone flight paths.
- **Cloud Deployment**: The system is deployed using Kubernetes (K8s) for container orchestration, which ensures scalability and fault tolerance.
- **Authentication & Security**: User authentication is managed via OAuth 2.0, with role-based access control (RBAC) to ensure that only authorized users can modify mission plans or access sensitive data.
- **Drone Communication**: The drones communicate with the system using Message Queuing Telemetry Transport (MQTT) for real-time telemetry streaming and command execution.
- **AI & Data Processing**: Image and sensor data collected by the drones are processed using TensorFlow and OpenCV, enabling automated defect detection and bridge condition assessment.
- **Infrastructure Management**: The system is hosted on AWS, utilizing Amazon S3 for storing inspection images, AWS Lambda for serverless processing, and Amazon RDS for database management.
- **Monitoring & Logging**: We implement Prometheus for monitoring drone performance metrics and ELK Stack (Elasticsearch, Logstash, Kibana) for logging and analytics.

8.5 Development Concepts

Coordinating many people on large and complex projects is never a trivial task. Therefore, to make sure all development goals are achieved on time and to guarantee the robustness and maintainability of our system, we adhere to several best practices throughout the development lifecycle. These practices include, among many, the following:

- **Agile Methodology**: We follow the Agile principles to allow for iterative development, continuous feedback, and flexible adaptation to changing requirements.
- **Version Control**: Utilizing Git for version control enables efficient collaboration among team members and maintains a reliable history of code changes.
- **Code Reviews**: Regular code reviews ensure code quality, consistency, and adherence to coding standards.

- **Continuous Integration and Continuous Deployment (CI/CD)**: Implementing CI/CD pipelines facilitates automated testing, integration, and deployment, ensuring rapid and reliable delivery of software updates.
- **Automated Testing**: A comprehensive suite of automated tests, including unit, integration, and end-to-end tests, helps to identify and address issues early in the development process.
- **Documentation**: Maintaining clear and up-to-date documentation for both code and system architecture ensures that all team members, as well as any other teams involved, have a shared understanding of the system and its components.

8.5.1 Deployment Roadmap

Another important aspect of the development concepts is the deployment roadmap, which contains the timeline our system's deployment will follow, from its initial prototyping phase all the way to the fully-fledged product release. Figure 24 contains the details of the envisioned deployment timeline.

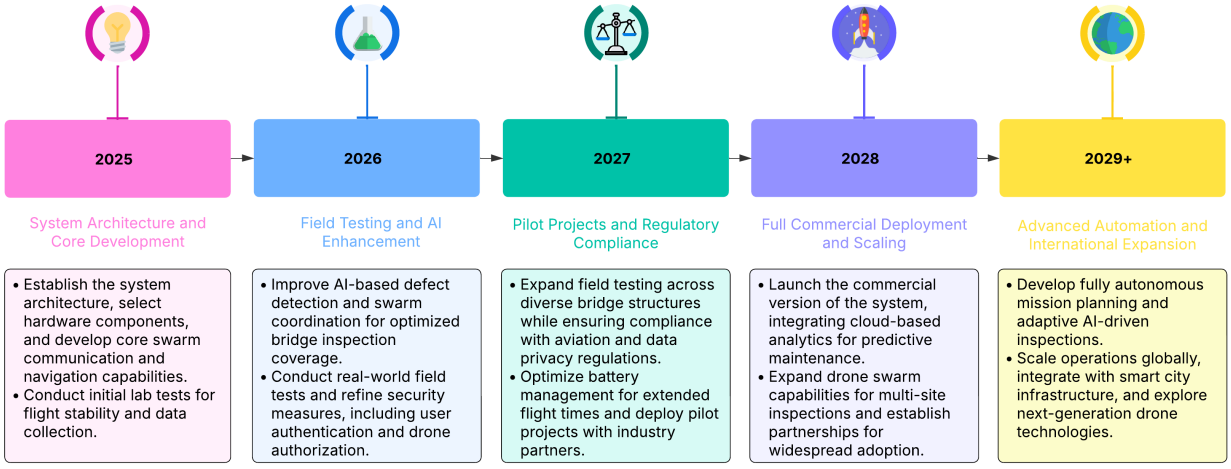


Figure 24: System Deployment Roadmap

The deployment roadmap for our system spans from 2025 to 2029 and beyond, outlining the key phases that assure an efficient deployment. In 2025, the focus is on system architecture and core development, establishing the foundational components and conducting initial lab tests. By 2026, efforts shift to field testing and AI enhancement, improving defect detection and refining swarm coordination. In 2027, the emphasis is on regulatory compliance and pilot projects, expanding field testing and optimizing battery management. The year 2028 marks the full commercial deployment and scaling phase, launching the

commercial version of the system and integrating cloud-based analytics. Finally, from 2029 onwards, the focus will be on advanced automation and international expansion, developing fully autonomous mission planning and scaling operations globally.

8.6 Operational Concepts

Our autonomous drone swarm for bridge inspection system is designed to offer efficient and thorough inspections of bridge structures, leveraging the latest advancements in drone technology. To achieve seamless operation, the system consists of an online platform through a Web Application that enables thorough mission planning, real-time monitoring, and post-flight data analysis. Different user roles, including structural engineers, inspection operators, and maintenance personnel, can interact with the platform based on their specific needs. Therefore, for operational efficiency, the system incorporates the following:

- **Autonomous Mission Execution:** The drone swarm follows predefined inspection routes or dynamically adjusts based on sensor-driven anomaly detection.
- **Real-Time Data Processing:** High-resolution images, LiDAR scans, and thermal imaging data are collected and processed in real-time to identify potential structural issues.
- **Centralized Monitoring and Control:** Operators can oversee swarm movements, adjust flight parameters, and receive instant alerts through a web-based dashboard.
- **Redundancy and Fail-Safe Mechanisms:** In case of drone failure or signal loss, fallback strategies ensure mission continuity without compromising safety.

9 Architecture Decisions

This section documents the most important architecture decisions made during the design of our autonomous drone swarm for bridge inspection system. These decisions are vital for ensuring the system's functionality, scalability, security, and overall performance, while also enabling stakeholders to understand the reasoning, context, and implications of these choices. Furthermore, this transparency helps maintain a coherent and well-structured system architecture, in which all design choices align with the project's main quality goals and stakeholder expectations.

For clarity, the decisions are organized in a list of Architecture Decision Records (ADRs), each following a uniform template that contains the following elements:

- **ID:** A unique identifier for the decision.
- **Title:** A descriptive title for the decision.

- **Context:** The background and circumstances that led to the decision.
- **Decision:** A clear statement of the decision made.
- **Alternatives:** Potential alternatives considered and evaluated.
- **Status:** The current status of the decision (e.g., pending, approved, implemented).
- **Consequences:** The positive and negative outcomes of the decision.

AD-1	One General Authentication Backend and Database
Context	Right now, the web interface requires authentication, and managing authentication across multiple services is a challenge. This problem is relevant for one of our main quality attributes: **security**.
Decision	We have decided to implement a single general authentication API or backend in the system that handles all authentication requests from the web interface.
Alternatives	• **Multiple Independent Authentication Modules** - The scheduler backend and the analysis tool use the same web interface, but requests are handled by different backends. Each of these backends could have its own authentication module. • **Shared Database** - We could have one general database for both drone inventory and user data.
Status	Pending
Consequences	1. **Separation of Concerns** - Having a dedicated component solely focused on authentication ensures a clear distinction of responsibilities, making the system more modular, maintainable, and secure. 2. **Improved Security** - Centralizing authentication reduces the risk of inconsistent security policies across different modules and ensures uniform enforcement of authentication mechanisms. 3. **Single Point of Failure (SPOF)** - If the authentication backend goes down, all authentication requests fail, potentially blocking access to the entire system. 4. **Scalability Considerations** - A single authentication backend might need to handle many authentication requests, requiring proper load balancing and scalability planning. 5. **Easier Maintenance and Updates** - Security patches, updates, and improvements can be applied centrally rather than across multiple independent modules.

AD-2	Use of Web App Instead of a Dedicated Application

Context	The system needs to provide an accessible, easy-to-use interface for scheduling and monitoring bridge inspection missions using drones. Users include engineers, operators, and maintenance personnel who may access the system from various devices and locations.
Decision	We have decided to implement a web-based user interface (Web App) instead of a dedicated desktop or mobile application.
Alternatives	• **Dedicated Desktop Application** - Develop a platform-specific application for desktop use, providing rich functionality and performance. • **Dedicated Mobile Application** - Develop a mobile-specific application for Android and iOS devices, offering mobile-friendly functionality.
Status	Approved
Consequences	1. **Accessibility** - A web app can be accessed from any device with a web browser, providing flexibility and convenience for users who may need to access the system from different locations or devices. 2. **Cross-Platform Compatibility** - The web app ensures compatibility across various operating systems and devices without the need for multiple platform-specific applications. 3. **Ease of Deployment and Updates** - Updates and maintenance are simplified, as changes to the web app can be deployed centrally without requiring users to download and install updates. 4. **Lower Development and Maintenance Costs** - Developing a single web app is generally more cost-effective than creating and maintaining separate desktop and mobile applications. 5. **Performance Considerations** - While web apps may not offer the same level of performance as native applications, modern web technologies (e.g., HTML5, CSS3, JavaScript) allow for highly responsive and interactive user experiences.

AD-3	User Roles and Permissions
Context	Different users, such as engineers, operators, and maintenance personnel, require specific access and permissions to interact with the system effectively. This means that users can perform their roles efficiently while maintaining system security.
Decision	We have decided to implement a Role-Based Access Control (RBAC) system to manage user roles and permissions. This system will define roles for different user types and assign appropriate permissions based on their responsibilities.

Alternatives	• **Single Role for All Users** - All users have the same access and permissions, which simplifies management but does not cater to specific needs. • **Custom Access Levels** - Create custom access levels for individual users based on their specific needs, which allows for detailed control but adds complexity.
Status	Approved
Consequences	1. **Enhanced Security** - Users have appropriate access to system features based on their roles, reducing the risk of unauthorized access to sensitive data. 2. **Operational Efficiency** - Users can perform their tasks more efficiently with access to the tools and data they need, improving overall system usability. 3. **Scalability** - The RBAC system can easily accommodate new roles and users as the system grows, providing flexibility for future expansion. 4. **Complexity in Management** - Requires ongoing management and updates to user roles and permissions, which may add administrative overhead. 5. **Auditing and Accountability** - Facilitates tracking and auditing of user actions, enhancing accountability and compliance with security policies.

AD-4	Cloud-Based Storage for Mission-Related Data
Context	The system collects large amounts of inspection and mission-related information (e.g. drone metrics/logs). Therefore, a reliable, secure, and accessible solution is required to ensure efficient data management. Cloud storage solutions offer redundancy, accessibility, and scalability, making them a strong candidate for our system.
Decision	We have decided to use a cloud-based storage solution (e.g. from the AWS suite of options) to store the mission-related data. That way, the data is always accessible and the storage capacity can easily scale with increasing demand.
Alternatives	• **Local Storage** - Storing data on on-premise servers was considered but was deemed infeasible due to scalability limitations and maintenance overhead. • **Hybrid Storage** - A combination of local and cloud storage was considered but adds complexity in managing synchronization and data consistency.
Status	Approved

Consequences	1. **High Availability and Reliability** - Cloud-based redundancy ensures data availability and protection against data loss. 2. **Scalability** - The storage solution can easily scale to accommodate increasing amounts of data without requiring infrastructure modifications. 3. **Security** - Cloud providers offer encryption and access control mechanisms to safeguard sensitive data. 4. **Dependency on internet connectivity** - Accessing the stored data requires a stable internet connection. 5. **Potential Costs** - Cloud storage incurs operational expenses based on storage volume and access frequency. These costs might change at any time based on the provider's pricing model and business decisions.

AD-5	Containerization with Docker and Orchestration with Kubernetes
Context	The system requires modular, scalable deployment of cloud-based subsystems (Mission Management, Data Analysis, Warehouse Management) to ensure high availability and seamless updates.
Decision	Use Docker containers for modular application packaging and Kubernetes for orchestration on AWS.
Alternatives	• **Monolithic Deployment** - Deploy subsystems as a single unit on virtual machines. • **Alternative Orchestrators** - Use AWS ECS instead of Kubernetes.
Status	Approved
Consequences	1. **Scalability** - Kubernetes auto-scales containers based on load. 2. **Fault Tolerance** - Self-healing pods ensure subsystem availability. 3. **Complexity** - Requires expertise in Docker/Kubernetes. 4. **Cost** - Kubernetes clusters incur higher AWS costs than simpler solutions.

AD-6	MAVLink for Drone Swarm Coordination
Context	The drone swarm requires real-time coordination and fault tolerance during missions.
Decision	Use MAVLink for drone-to-drone communication and ROS 2 for swarm coordination.
Alternatives	• **Custom Protocol** - Develop a proprietary communication protocol. • **MQTT for All Communication** - Rely solely on MQTT for drone coordination.
Status	Approved

Consequences	1. **Interoperability** - MAVLink is widely adopted in UAV systems. 2. **Latency** - Optimized for real-time drone communication. 3. **Complexity** - Requires integration with ROS 2 for swarm logic.

10 Quality Requirements

10.1 Quality Tree

Table 17: Quality Assessment Table

Quality Category	**Quality**	**Description**	**Scenario**
Reliability	Faultlessness	Atempt pre-checks before drone/swarm can perform any activity that could lead to faults	Faultlessness scenarios from figure 25
	Availability	The scheduler, web interface and drone monitoring have a high uptime	Availability scenarios from figure 25
	Fault Tolerance	The swarm of drones remain operational in case of a faulty drone. When a software component is failing, a backup is provided	Fault Tolerance scenarios from figure 25
	Recoverability	System data or component remains unaffected in case of a point of failure	Recoverability scenarios from figure 25
Performance Efficiency	Resource Utilization	The system is optimized for energy efficiency	Resource Utilization scenarios from figure 25
	Time behaviour	Throughput of data should handle large volumes of asynchronous data	Time behavior scenarios from figure 25
	Capacity	Swarm storage has a limited capacity for data stored per unit	Capacity scenario from figure 25
Security	Authenticity	System tracks authenticity of drones and users within its system	Authenticity scenarios from figure 26
	Accountability	System keeps track of all entities that are connected or try to connect to it	Accountability scenarios from figure 26
	Confidentiality	Data accessibility is compliant with regulations that are in place	Confidentiality scenarios from figure 26
Safety	Hazard Warning	System issues alerts when events that constitute unacceptable risks to the operation are about to take place	Hazard Warning scenarios from figure 26
	Risk Identification	System carefully monitors the area for various events that coud pose significant risks to it's functionability	Risk Identification scenarios from figure 26
	Operational Constraint	System has constraints put in place to ensure execution of tasks under safe parameters	Operational Constraint scenarios from figure 26

10.2 Utility Tree

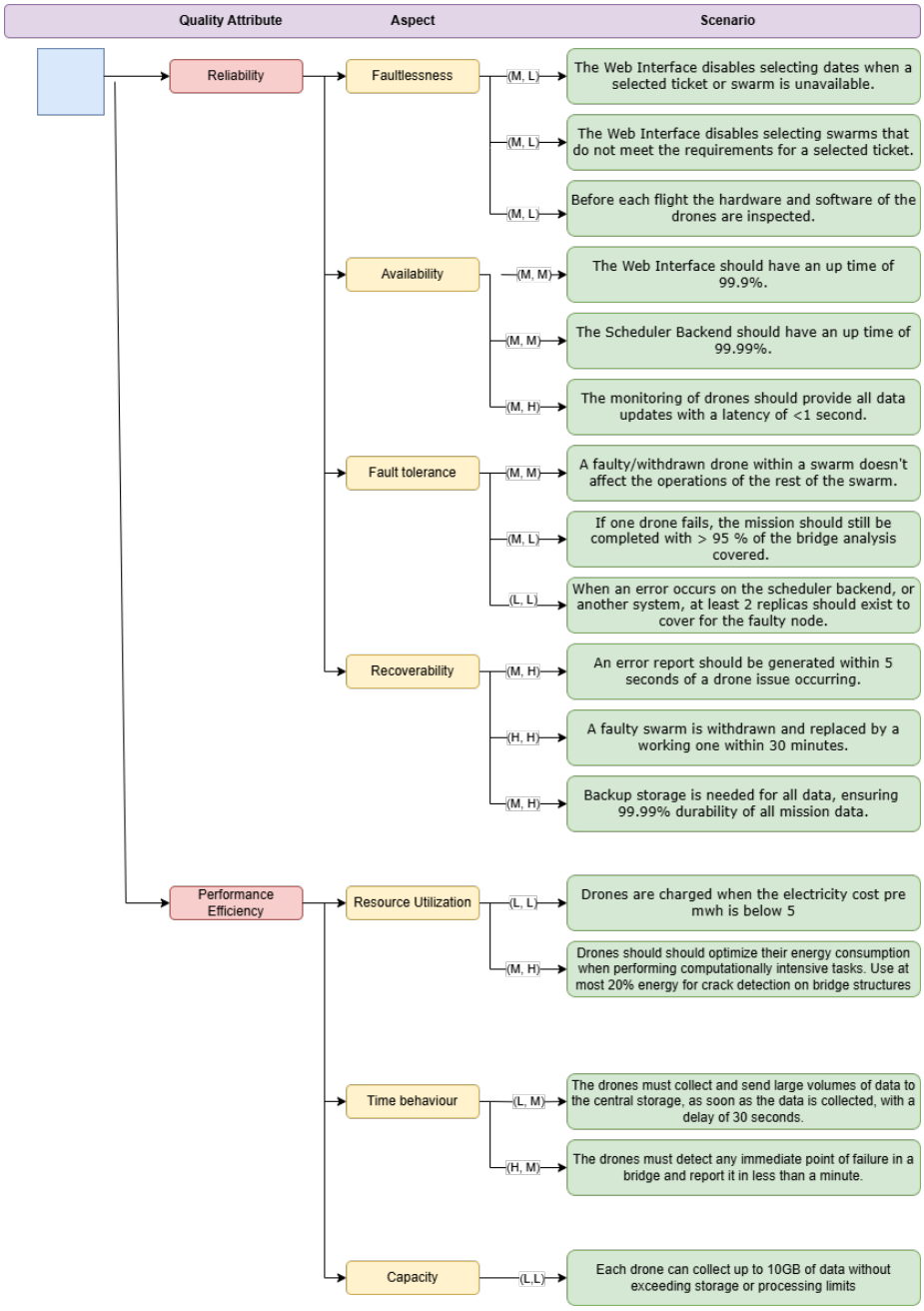


Figure 25: Swarm of Drones for Bridge Inspection Utility Tree

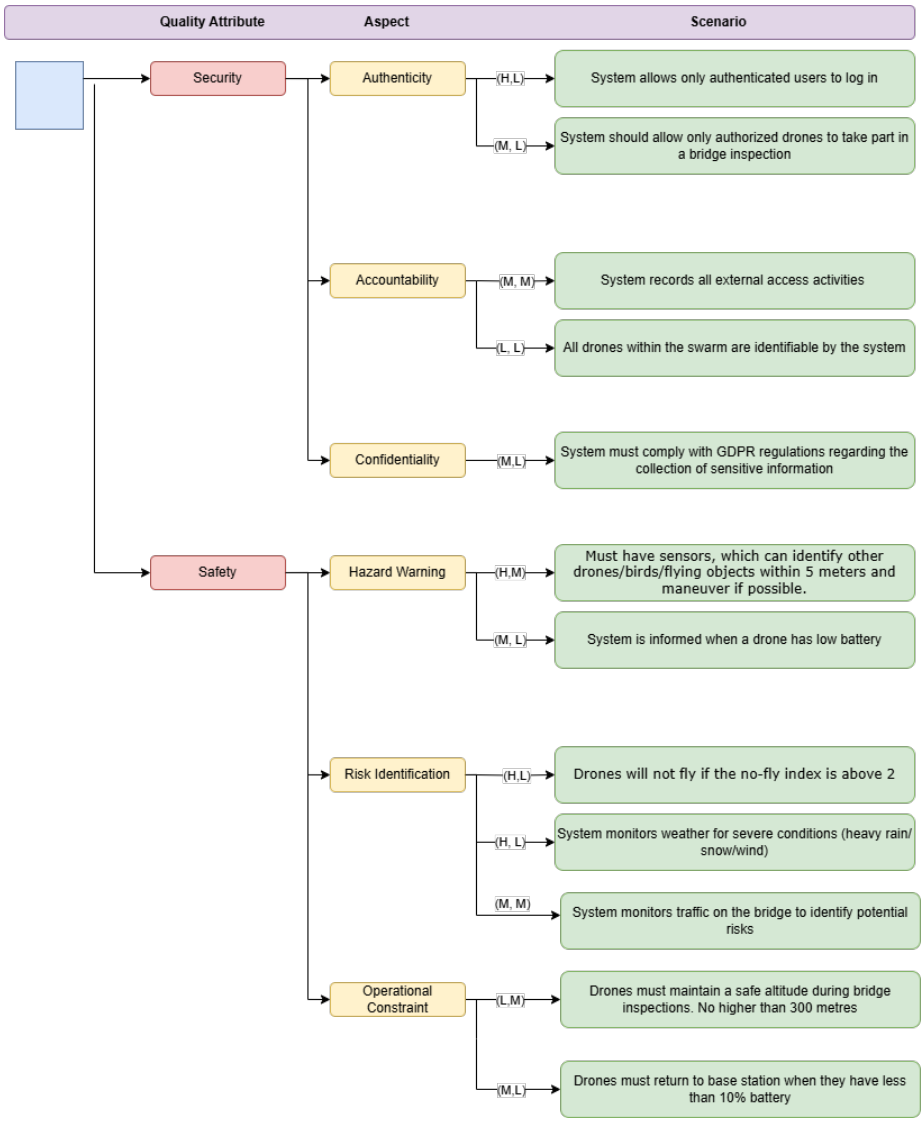


Figure 26: Swarm of Drones for Bridge Inspection Utility Tree (continued)

11 Risks and Technical Debts

While the Autonomous Swarm Management for Bridge Inspection system is designed with scalability and reliability in mind, risks and technical debts are not always avoidable. However, we need to think ahead and be aware of these potential challenges. To address this, we have detailed the top identified risks in Table 18.

Furthermore, certain trade-offs and constraints can lead to the accumulation of technical debts, which are showcased in Table 19. By proactively acknowledging these areas, we can implement strategies to mitigate risks and gradually reduce technical debt over time.

11.1 Risks

Risk	Impact	Mitigation Strategy
Communication Failure Between Drones	Loss of coordination in swarm, leading to incomplete inspections.	Implement redundant communication channels and fallback coordination algorithms.
Drone Malfunction Mid-Flight	Mission failure and potential loss of data if the drone is not recovered.	Deploy backup drones and implement real-time fault detection with automated recovery procedures.
Large Boats Passing Under the Bridge	Drones may be forced to abort inspection mid-mission to avoid collision.	Implement dynamic mission adaptation and enforce no-fly zones for large vessel crossings.
Cybersecurity Threats (Hacking, Data Breaches)	Unauthorized access to drone controls or sensitive inspection data.	Implement end-to-end encryption, role-based access control, and regular security audits.
Regulatory Compliance Issues	Different regions have varying regulatory requirements for drone operations.	Consult legal and regulatory experts in advance to ensure compliance with the appropriate standards for each geographical area.
Public Discontent Over Drone Usage	Complaints from residents or public strikes may restrict operations.	Engage with local authorities and the public for awareness and management.
Timing Issues on the Day of Deployment	Delays due to traffic jams or late arrivals may push mission outside the operational window.	Allocate buffer time for personnel movement and schedule flexible deployment slots.
Intentional Interference (Pranks or Malicious Actors)	Unauthorized access, jamming, or sabotage may disrupt missions.	Implement geo-fencing, and intrusion detection alerts. Ensure quick deployment of back-up drones and engage local authorities as needed.
Unexpected Weather Changes	Sudden strong winds or storms can cause mission delays or drone crashes.	Implement real-time weather monitoring and emergency return-to-base protocols.
Sick Personnel	Lack of key personnel may cause operational delays or cancellations.	Maintain a backup crew or on-call personnel to cover absences.

Table 18: Identified Risks and Mitigation Strategies

11.2 Technical Debts

Issue	Impact	Proposed Resolution
Limited Modularity in the Data Analysis Pipeline	Coupling between data processing and system logic makes updates difficult. Future AI model integrations require extensive code modifications.	Refactor into a microservices architecture to allow independent scaling and updates.
Hardcoded Drone Swarm Coordination Parameters	Adjusting swarm behavior requires code changes and redeployment instead of real-time updates.	Implement a configuration management system to allow dynamic tuning via operator interface.
Lack of Automated Testing for Swarm Behavior Simulations	Swarm coordination behaviors require extensive manual testing, slowing development.	Develop a simulation-based testing framework using tools like Gazebo or AirSim.
Inefficient Data Storage for Inspection Reports	SQL database storage leads to inefficiencies when handling large-scale historical queries.	Introduce hybrid storage with cloud-based object storage for images and NoSQL for fast metadata retrieval.
Manual Deployment and Infrastructure Management	Manual system deployment increases risks of errors and slows updates.	Implement CI/CD pipelines with automated testing and rollback features.

Table 19: Identified Technical Debts and Proposed Resolutions

12 Glossary

The glossary containing the most important technical and domain terms used when discussing the system.

Term	Definition
SPA (Single Page Application)	A web application or website that dynamically updates content without requiring full page reloads.
Node.js	A JavaScript runtime built on Chrome's V8 engine, commonly used for building scalable backend services.
OAuth 2.0	An open standard for authorization, which allows users to grant third-party applications access to their resources without exposing credentials.

Term	Definition
MQTT	Stands for Message Queuing Telemetry Transport which is a lightweight messaging protocol designed for low-bandwidth, high-latency networks, often used in IoT (Internet of Things) and real-time telemetry applications.
A/B Testing	A method of comparing two versions of a system or component to determine which performs better.
ArgoCD	A declarative, GitOps continuous delivery tool for Kubernetes.
ESC	An Electronic Speed Controller regulates motor speed and ensures stable, precise flight in drones.
gRPC	Remote Procedure Call is an open-source framework for efficient communication between services using protocol buffers for data serialization.
HAL	Hardware Abstraction Layer which makes software work with different hardware easily.
HSM	Hardware Security Module is physical device that protects encryption keys.
IMU	Inertial Measurement Unit is a sensor that measures movement and orientation.
LiDAR	Stands for Light Detection and Ranging which uses lasers to measure distances and map surroundings.
MAVLink	Stands for Micro Air Vehicle Link which is a protocol for drone communication.
PX4	Open-Source Flight Control Software that controls drones.
QGIS	Quantum GIS is an open-source software for working with maps and spatial data.
ROS 2	Robot Operating System 2 is a middleware that helps robots (and drones) work and communicate.
OpenCV	An open-source computer vision library used for image processing, object detection, and machine learning applications.
AWS	Amazon Web Services is a cloud computing platform offering a range of infrastructure services such as bulk storage, computing, and relational databases.

References

[1] Wilfried Yves Hamilton Adoni et al. “Intelligent swarm: Concept, design and validation of self-organized UAVs based on Leader–followers paradigm for autonomous mission planning”. en. In: *Drones* 8.10 (Oct. 2024), p. 575.

[2] Mostafa Aliyari, Behrooz Ashrafi, and Yonas Zewdu Ayele. “Drone-based bridge inspection in harsh operating environment: Risks and safeguards”. en. In: *Int. J. Transp. Dev. Integr.* 5.2 (June 2021), pp. 118–135.

[3] Yunes Alqudsi and Murat Makaraci. “UAV swarms: research, challenges, and future directions”. en. In: *J. Eng. Appl. Sci.* 72.1 (Dec. 2025).

[4] arc42 Documentation. *Section 8: Cross-cutting Concepts*. https://docs.arc42.org/section-8/. Accessed March 2025.

[5] PM Column. *Primary vs Secondary Stakeholders: How to Identify and Manage Them*. Accessed March 2025. URL: https://www.pmcolumn.com/primary-vs-secondary-stakeholders/.

[6] International Organization for Standardization. *ISO 21384-3:2019. Unmanned aircraft systems — Part 3: Operational procedures*. 2019.

[7] International Organization for Standardization. *ISO/IEC 25010:2011. Systems and software engineering — Systems and software quality requirements and evaluation (SQuaRE) — System and software quality models*. 2011.

[8] Rune Hylsberg Jacobsen et al. “Design of an autonomous cooperative drone swarm for inspections of safety critical infrastructure”. en. In: *Appl. Sci. (Basel)* 13.3 (Jan. 2023), p. 1256.

[9] Kodeco Team. *Common Design Patterns and App Architectures for Android*. https://www.kodeco.com/18409174-common-design-patterns-and-app-architectures-for-android/page/3. Accessed March 2025.

[10] Olga Levina and Vladimir Stantchev. “Realizing Event-Driven SOA”. In: *2009 Fourth International Conference on Internet and Web Applications and Services*. 2009, pp. 37–42. DOI: 10.1109/ICIW.2009.14.

[11] Authors of the Paper. *Standing Bridges C4 Model*. https://minhtam-le.github.io/advanced_software_architecture/view/index. [Accessed 14-03-2025]. 2025.

[12] Lauren Parker Shan Luo. *Aerial Swarm Robotics for Active Inspection of Bridges | Centre for Digital Built Britain completed its five-year mission and closed its doors at the end of September 2022 — cdbb.cam.ac.uk*. https://www.cdbb.cam.ac.uk/research/data-science-artificial-intelligence-machine-learning/aerial-swarm-robotics-active. [Accessed 19-02-2025]. 2019.