

Technical Journal

Aurelia Jennifer Gunawan
13524089

Last Updated: 2026-02-03

Contents

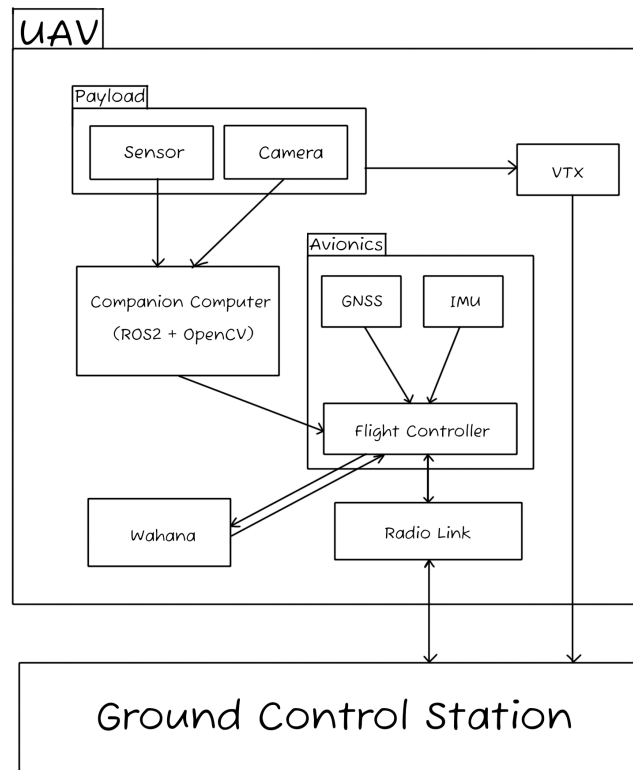
1	Catatan Pendidikan	3
1.1	Arsitektur Sistem UAV	3
1.1.1	Komponen” UAV	3
1.1.2	Komunikasi	5
1.1.3	Logika Operasi	7
1.2	Dasar Pemrograman	8
1.2.1	Konsep Dasar OOP	8
1.2.2	Design Pattern	8
1.2.3	Error Handling	9
1.2.4	Virtual Function	9
1.2.5	Pengenalan Pemrograman Konkuren	9
1.3	MAVLink	10
1.3.1	Pendahuluan	10
1.3.2	System ID, Component ID, Message ID	10
1.3.3	Pola Interaksi dengan MAVLink	10
1.3.4	Anatomi Paket MAVLink	11
1.3.5	MAVLink Interfaces	11
1.4	Software In The Loop (SITL)	11
1.4.1	Pendahuluan	11
1.4.2	Arsitektur sistem SITL	12
1.4.3	Tools	12
2	Hands-on 1: Daspro	12
2.1	Log Progress	12
2.2	Catatan	13
2.2.1	Brainstorming	13
2.2.2	RAII	13
2.2.3	Observer Pattern	13
2.2.4	Strategy Pattern	14
3	Hands-on 2: MAVLink dan SITL	14
3.1	Log Progress	14
3.2	Kendala dan Solusi	14
3.3	Catatan	15

4	THT 2	15
4.1	Log Progress	15
5	Referensi	16

1 Catatan Pendidikan

1.1 Arsitektur Sistem UAV

1.1.1 Komponen” UAV



Wahana terdiri atas airframe dan sistem propulsi. Airframe pada pesawat terdiri atas fuselage, sayap, dan ekor. Airframe pada drone terdiri atas frame atau center plate, dan arm. Sistem propulsi terdiri atas tiga komponen:

- **Propeller**
Menghasilkan gaya angkat/dorong melalui rotasi.
- **Motor**
Mengubah energi listrik menjadi energi kinetik rotasi untuk memutar *propeller*
- **Electronic Speed Controller (ESC)**
Mengatur kecepatan dan arah putaran motor sesuai perintah dari FC
- **Baterai Lithium Polymer (LiPo)**
Sumber energi utama

Flight controller berperan sebagai sistem saraf motorik dari UAV yang memproses dan menyatukan data dari sensor serta menjalankan algoritma kontrol untuk mengendalikan penerbangan, baik berdasarkan perintah pilot (dari GCS0 maupun secara otonom (dari companion computer). FC terdiri atas beberapa *chip* atau unit:

- Microcontroller Unit (MCU) Prosesor utama yang menjalankan *firmware* dan mengolah semua data dan logika kontrol penerbangan
- Barometer
Mengukur tekanan atmosfer untuk mengestimasi ketinggian wahana
- Inertial Measurement Unit (IMU) Mengintegrasikan *accelerometer* dan *gyroscope* yang mengukur percepatan, kecepatan sudut, dan orientasi UAV untuk mengetahui pergerakannya.
- Blackbox Mencatat log keseluruhan UAV sejak arm hingga disarm, untuk analisis pasca penerbangan.
- Modul GNSS Menerima sinyal dari sistem navigasi satelit global, seperti GPS, GLONASS, Galileo, dan BeiDou untuk menentukan posisi, kecepatan, dan waktu UAV.
- Airspeed sensor Mengukur kecepatan relatif wahana terhadap udara menggunakan tabung pitot
- Magnetometer (kompasnya UAV) Mendeteksi medan magnet bumi untuk menentukan arah mata angin yang dituju wahana. Biasanya terintegrasi dalam modul GNSS sebagai *chip* kompas.

Payload dapat berupa:

- Kamera FPV Menyediakan video real time ke pilot/GCS
- Kamera Depth Menghasilkan peta kedalaman/*depth map* dan titik 3D secara real-time
- Kamera Tracking Mengestimasi posisi dan orientasi UAV secara real time tanpa GPS
- LiDAR Sensor pemetaan 3D high precision untuk misi survei topografi dan kehutanan
- Sensor Partikel Sensor untuk memantau kualitas udara
- and more...

Ground control station (GCS) adalah sistem di darat untuk memantau, mengendalikan, dan mengelola misi uAV lewat gelombang radio. GCS terdiri atas remote controller, laptop dan aplikasi GCS, modul radio telemetry, dan antena yagi. UAV dan GCS melakukan komunikasi dengan **gelombang radio** pada spektrum frekuensi tertentu. Jalur komunikasi fungsional yang terbentuk dari pertukaran data lewat frekuensi ini disebut dengan **link**. Terdapat dua jenis link utama dalam sistem UAV:

1. C2/Data/Radio/Non-payload Link Saluran utama yang bertanggung jawab atas pengendalian UAV. Secara fisik hubungan ini dijalin antara *radio transceiver* di GCS dan di UAV. Terdiri atas:
 - Uplink (*telecommand*) mengirimkan perintah kendali dari GCS/pilot ke UAV

- Downlink (*telemetry*) mengirimkan data status UAV (attitude, baterai, GPS, ketinggian, dsb.) ke GCS
2. Payload Link Saluarn khusus untuk mengirimkan data dari payload ke GCS secara real-time. Untuk kamera, link ini dijalin secara fisik antara VTX (video transmitter) dan VRX (video receiver)

Companion computer adalah "otak" yang melakukan komputasi tingkat tinggi, yang melengkapi FC. CC disambungkan ke FC secara serial dan berkomunikasi dengan protokol MAVLINK. Payload dan sensor memberikan data ke CC untuk mengambil keputusan, lalu CC mengirimkan perintah ke FC untuk dieksekusi.

1.1.2 Komunikasi

Terdapat dua cara mengirim data:

1. Wired
 - Digunakan untuk komunikasi jarak pendek antarkomponen
 - Media fisik berupa kabel tembaga
 - Laju transfer sangat tinggi, latensi rendah, dan lebih baik dari segi keamanan, tapi jangkauannya terbatas
2. Wireless
 - Digunakan untuk komunikasi jarak jauh, misalnya antara GCS dan UAV
 - Media fisik berupa gelombang radio dari transceiver
 - Mobilitas lebih tinggi, tapi rentan interferensi dan bandwidth, serta latensi terbatas

Proses pengiriman data diatur oleh **protokol**. Protokol mengatur:

1. sintaks: cara menyusun struktur data
2. semantik: makna dari setiap bit data
3. sinkronisasi: keselarasan kecepatan pengiriman data

Wired:

1. UART (Universal Asynchronous Receiver-Transmitter)
 - Bersifat **asinkron**. Pengirim dan penerima menyepakati laju pengiriman data (baud rate, bit/s), panjang data, dan bit (0 atau 1) untuk menandakan awal dan akhir satu pesan.
 - Mempunyai dua pin, yaitu **RX** dan **TX**. RX untuk menerima data dari perangkat lain. TX untuk mengirim data ke perangkat lain.
 - Digunakan untuk menghubungkan FC ke modul telemetri, GPS, dan CC.
2. SPI (Serial Peripheral Interface)
 - Bersifat **sinkron**.

- Mempunyai empat pin, yaitu **SCLK**, **MOSI**, **MISO**, dan **SS**. SCLK mirip SCL di I2C. MOSI adalah jalur data dari master ke slave. MISO jalur data dari slave ke master. SS adalah pin untuk memilih slave mana yang diajak bicara.
- Digunakan untuk menghubungkan FC (master) dengan komponen kritikal (slave) dengan data padat seperti IMU dan blackbox

3. I2C (Inter-Integrated Circuit)

- Bersifat **sinkron**.
- Mempunyai dua pin, yaitu SDA dan SCL. SDA adalah jalur dua arah untuk mengirim dan menerima data. SCL membawa sinyal detak dari master untuk sinkronisasi.
- Digunakan untuk menghubungkan FC ke sensor berkecepatan rendah seperti airspeed sensor dan barometer

Wireless:

1. TCP (Transmission Control Protocol)

- Sebelum mengirim data, pengirim dan penerima melakukan **handshake** untuk memastikan kesiapan pengiriman data.
- Kalau ada data yang hilang/rusak di tengah jalan, pengirim berhenti dan mengirim ulang data tsb sampai diterima secara sempurna.
- Digunakan untuk **upload waypoints** atau **mengubah parameter**, karena merupakan informasi krusial yang tidak boleh salah satu bit pun (dapat berakibat fatal), dan reliability diprioritaskan.

2. UDP (User Datagram Protocol)

- Data dikirim terus menerus tanpa memedulikan apakah penerima sudah siap atau data sampai secara sempurna.
- Pengiriman data sangat cepat dengan latensi rendah, tapi tidak reliable
- Digunakan untuk **streaming video dan data telemetry secara real time**

Setiap data yang dikirim umumnya dikemas berdasarkan protokol perangkat lunak untuk UAV, seperti MAVLink (Micro Air Vehicle Link). MAVLink adalah protokol komunikasi pesan *open source* yang sangat ringan dan penting karena jaringan nirkabel memiliki *bandwidth* yang terbatas.

- Setiap wahana UAV memiliki system ID (1-255) dan setiap komponen dalam wahana memiliki component ID.
- UAV mengirimkan pesan HEARTBEAT secara rutin untuk memberi tahu GCS bahwa koneksi masih aktif dan sistem dalam keadaan normal.
- MAVLink lebih sering menggunakan UDP (kalau pake TCP akan ada delay yang berpotensi membuat wahana *crash*. MAVLink memiliki *checksum*, kode verifikasi yang dihasilkan secara matematis dari isi pesan itu sendiri sehingga ketidakcocokan *checksum* akan mengakibatkan paket data terbuang.)

1.1.3 Logika Operasi

System state dan flight mode menentukan seberapa besar kontrol pilot dibandingkan dengan kontrol otomatis oleh FC dan apa yang dilakukan wahana dalam suatu kondisi/mode.

- **System State** Sebelum terbang, wahana harus melewati tahapan status berikut:
 1. **Initialization:** FC melakukan kalibrasi sensor internal tepat setelah baterai dihubungkan.
 2. **Pre-arm Check:** Diagnosa otomatis yang dilakukan wahana untuk memastikan kondisinya layak terbang.
 3. **Disarm:** Kondisi standar saat di darat. Motor terkunci total dan tidak akan berputar.
 4. **Arm:** Kondisi aktif, motor mulai berputar pada kecepatan rendah dan wahana siap menerima input *throttle* untuk lepas landas.
- **Flight Mode** Berikut adalah beberapa contoh mode terbang untuk ArduPilot:
 1. **Acro:** Kendali penuh tanpa stabilisasi otomatis.
 2. **Stabilize:** Wahana otomatis kembali datar jika stick dilepas.
 3. **Alt-Hold:** Menjaga ketinggian otomatis, posisi horizontal tetap diatur manual.
 4. **Loiter:** Menjaga koordinat dan ketinggian secara statis di udara.
 5. **RTL/RTH (Return To Land/Home):** Kembali ke titik landas/home.
 6. **Auto:** Menjalankan misi berdasarkan waypoints yang telah ditentukan.
 7. **Guided:** Bergerak secara interaktif berdasarkan input pada peta di GCS.

Mekanisme Failsafe adalah prosedur otomatis yang dilakukan UAV ketika terjadi kegagalan sistem untuk mencegah *crash* atau kehilangan wahana. Beberapa pemicu failsafe meliputi:

- **RC Link Lost:** sinyal antara *remote control* dan *receiver* di UAV terputus
- **GCS Link Lost:** koneksi telemetri antara GCS dan UAV terputus
- **Low Battery**
- **EKF Check/GPS Glitch**
- **Fence Breach:** wahana keluar dari batas wilayah geografis yang sudah ditentukan (geofence).

Beberapa tindakan yang diambil FC dalam kondisi failsafe (di-configure sebelumnya) adalah:

- RTL
- Land
- Brake
- and more...

1.2 Dasar Pemrograman

1.2.1 Konsep Dasar OOP

Why OOP?

- Fokus pada objek yang memodelkan benda nyata (data + perilaku)
- Modularitas tinggi, kode bisa dipecah menjadi bagian” (class)
- Rapi, terstruktur, dan mudah di-maintenance Class = skematik/datasheet komponen (rancangan) Object = Komponen fisik yang sudah dirakit Sebuah class memiliki atribut (variabel yang menyimpan data), dan metode (fungsi/prosedur yang mendefinisikan tindakan atau perilaku yang bisa dilakukan oleh class tersebut).

Encapsulation :

1. Membungkus data sensitif agar tidak bisa diakses sembarangan dari luar. Akses untuk membaca dan mengubah data dilakukan dengan *getter* dan *setter*.
2. Menentukan siapa yang bisa mengakses suatu atribut/method dari suatu kelas dengan access modifier:

Modifier	Own Class	Derived Class	Main()
Public	Yes	Yes	Yes
Protected	Yes	Yes	No
Private	Yes	No	No

Abstraction bertujuan untuk menyembunyikan proses/cara kerja di belakang layar.

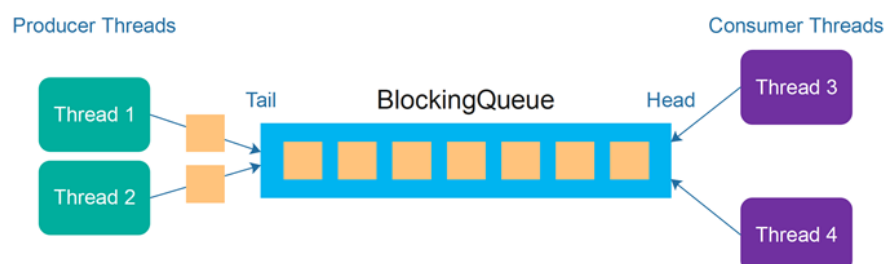
Inheritance (based on prinsip Don't Repeat Yourself/DRY). Parent class/base class adalah class yang mewariskan sifat. Child class/derived class adalah class yang mewarisi sifat dari parent.

Polymorphism adalah prinsip yang memungkinkan respon yang berbeda-beda berdasarkan perintah yang sama, tergantung object.

1.2.2 Design Pattern

Pola umum yang digunakan untuk mempermudah pengembangan aplikasi.

Producer-Consumer Pattern adalah pola desain konkuren, di aman satu atau lebih thread producer menghasilkan objek yang diantrekan, lalu dikonsumsi oleh satu atau lebih consumer. Objek yang diantrekan umumnya mewakili beberapa pekerjaan yang perlu dilakukan.



1.2.3 Error Handling

Mekanisme **Try-Catch**:

- Blok try berisi kode yang dicoba untuk dijalankan
- Kalau ada error pindah ke block catch dengan Exception yang sesuai
- better than if-else karena dapat lebih mengcover error tak terduga

1.2.4 Virtual Function

Virtual function adalah fungsi parent class/base class yang dideklarasikan dengan kata kunci virtual, dan di-override dala child class/derived class. Fungsi virtual memungkinkan *runtime polymorphism*, dimana fungsi yang tepat dipanggil melalui base class pointer atau reference.

1.2.5 Pengenalan Pemrograman Konkuren

Multithreading

- Thread adalah unit terkecil dari proses
- Multithreading adalah teknik memecah proses menjadi task kecil yang dijalankan oleh thread di dalamnya sehingga bisa berjalan secara konkuren/bersama.
- Problems? Race condition, dan deadlock!

Race condition:

- thread saling mengakses resources yang sama, maka dapat terjadi rebutan resources antar thread.
- solusi: use **mutex/lock** sehingga hanya satu thread yang bisa mengakses suatu resource di suatu waktu.

Deadlock:

- dua atau lebih proses/pihak dapat saling menunggu resources yang dipegang oleh pihak lain
- solusi:
 1. deadlock prevention: menghilangkan salah satu syarat kondisi terjadinya deadlock, yaitu mutual exclusion, hold and wait, no preemption, dan circular wait.
 2. deadlock avoidance dengan banker's algorithm (memeriksa secara dinamis status alokasi resource untuk memastikan sistem tidak memasuki unsafe state)
 3. deadlock detection and recovery dengan timeout dan deteksi cycle.

1.3 MAVLink

1.3.1 Pendahuluan

MAVLink atau Micro Air Vehicle Link adalah protokol komunikasi pesan open source yang sangat ringan, yang penting karena jaringan nirkabel memiliki bandwidth terbatas. MAVLink mendukung komunikasi dua arah yang efisien dan memungkinkan integrasi wahana UAV dengan jaringan internet.

1.3.2 System ID, Component ID, Message ID

Setiap entitas yang bisa mengirim atau menerima pesan MAVLink harus punya identitas unik. Suatu sistem MAVLink (wahana/GCS) bisa memiliki banyak komponen.

1. System ID

- Identitas unik satu sistem UAV.
- Rentang nilai 1-255.
- System ID 255 dialokasikan untuk GCS.
- Semua komponen dalam wahana yang sama memiliki system ID yang sama.

2. Component ID

- Identitas spesifik untuk setiap komponen atau subsistem dalam satu wahana.
- Sudah ada penomoran bawaannya dari MAVLink, i.e. FC: 1, Kamera: 100-105, Gimbal: 154, CC: 191-194.

MAVLink punya berbagai jenis pesan untuk pertukaran data antarkomponen dalam sistem UAV. Setiap pesan memiliki message ID yang unik. Untuk MAVLink v1 ada 255, untuk MAVLink v2 hingga 16 juta. Dialect adalah kumpulan pesan yang dapat disesuaikan dengan kebutuhan sistem tertentu. Berdasarkan tujuannya, message dibagi menjadi dua macam:

1. State messages: dikirim dari wahana ke GCS untuk melaporkan status (posisi, orientasi, baterai, dsb.)
2. Command messages: dikirim dari GCS ke wahana untuk mengeksekusi aksi atau misi tertentu (event based)

1.3.3 Pola Interaksi dengan MAVLink

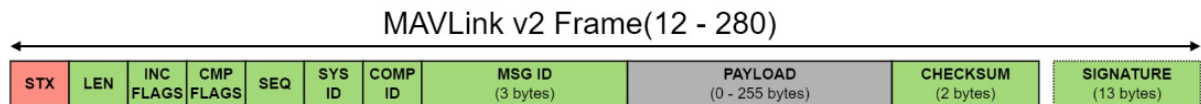
1. Streaming (Telemetry)

- Dipakai untuk **data yang harus dipantau terus-menerus** secara real-time, yaitu pesan bertipe **state messages**. Contohnya heartbeat, attitude, global_position_int
- Bersifat **broadcast** atau **satu arah** secara periodik dari wahana ke GCS
- Mengandalkan message ID untuk menentukan jenis data tanpa alamat tujuan spesifik

2. Microservices

- Dipakai untuk **perintah kritis yang butuh konfirmasi**, yaitu pesan bertipe **command messages**. Contohnya upload/download waypoint, instruksi arm/disarm/takeoff/land, dan membaca/mengubah pengaturan internal (parameter) pada FC.
- Bersifat **point-to-point** atau **dua arah**, seperti client-service
- Menggunakan system ID dan component ID agar perintah tidak salah sasaran.

1.3.4 Anatomi Paket MAVLink



- Setiap paket dimulai dengan STX (0XFD) sebagai penanda bahwa paket tersebut menggunakan protokol MAVLink
- SYS_ID (System ID) dan COMP_ID (Component ID) untuk identitas
- MSG_ID (Message ID) untuk jenis message yang dikirim
- PAYLOAD untuk membungkus konten data utama
- Checksum memastikan paket tidak corrupt saat diterima, menggunakan algoritma CRC-16 (ITU X.25) untuk menghasilkan kode verifikasi 2-byte (CKA dan CKB) dengan data dari LEN hingga PAYLOAD
- SIGNATURE adalah tanda tangan digital (SHA-256) untuk memastikan bahwa paket MAVLink yang diterima berasal dari pihak terpercaya.

1.3.5 MAVLink Interfaces

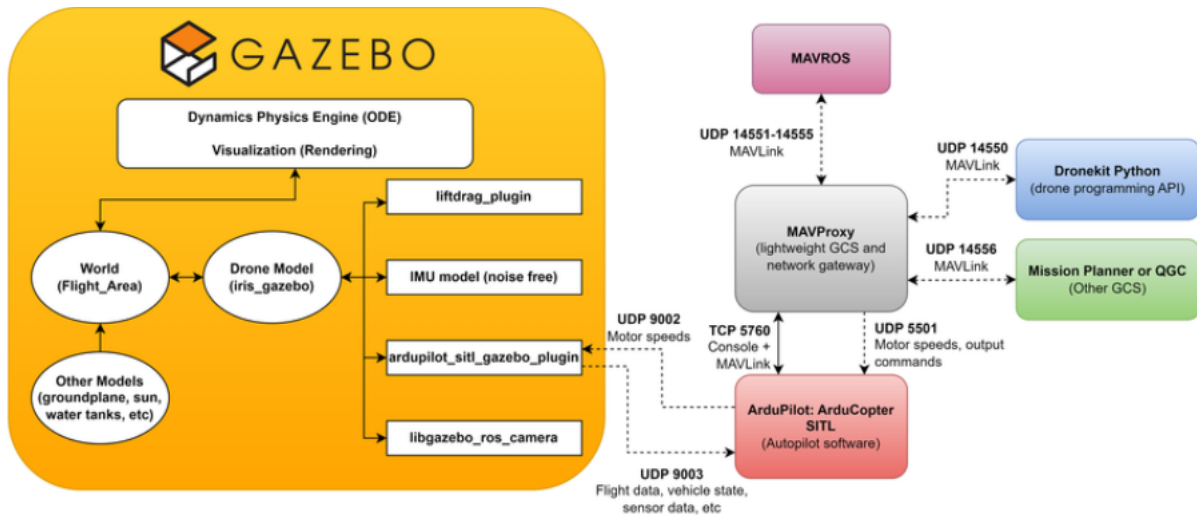
1. MAVProxy: CLI-based GCS, yang sangat ringan dan stabil
2. MAVSDK: Software development kit untuk membangun aplikasi berbasis MAVLink dengan bahasa tingkat tinggi (C++, Python, dsb.)
3. MAVROS (MAVLink + ROS): Penerjemah antara MAVLink dan ROS pada drone berbasis ROS

1.4 Software In The Loop (SITL)

1.4.1 Pendahuluan

Software in the loop (SITL) adalah metode untuk menjalankan dan menguji sistem autopilot wahana UAV tanpa menggunakan perangkat keras fisik. Dalam SITL, FC dan lingkungan (hukum fisika) disimulasikan oleh perangkat lunak. Selain itu, flight stack, seperti ardupilot/PX4, dijalankan seperti program biasa di komputer. Pengujian melalui SITL bertujuan untuk memeriksa ketepatan implementasi logika navigasi karena tidak ada risiko kerusakan fisik akibat crash

1.4.2 Arsitektur sistem SITL



Arsitektur SITL memisahkan simulasi fisik dan logika kontrol drone. Gazebo untuk simulasi fisik, yang menghasilkan data sensor virtual (flight data dan vehicle state, sensor data, dsb.). Data sensor virtual dikirim ke Ardupilot SITL melalui protokol UDP. Ardupilot mensimulasikan FC, yang memproses data sensor dengan algoritma navigasi, EKF, dan kontrol untuk menghitung perintah motor. Ardupilot akan mengirim kembali output motor ke Gazebo agar pergerakan drone di-update. Di sisi lain, Ardupilot berkomunikasi dengan MAVProxy melalui protokol TCP untuk menerima perintah kontrol, mengirim waypoint dan mengendalikan misi. Selain itu, MAVProxy juga mengirimkan data motor speeds dan output commands ke Ardupilot melalui protokol UDP.

1.4.3 Tools

1. Ardupilot: flight stack untuk mensimulasikan FC
2. Gazebo harmonic: simulasi fisika dan lingkungan serta dinamika wahana yang realistis
3. MAVProxy: jembatan komunikasi antara pengguna dan flight stack

2 Hands-on 1: Daspro

2.1 Log Progress

- 30/01/2026 8 pm - mulai pengerjaan (brainstorming dan baca spek)
- 31/01/2026 1 am - selesai implementasi topic broker, node, dan SimTelemetryNode
- 01/02/2026 10 pm - selesai wiring: start 3 SimTelemetryNode dan subscribe dashboard ke 3 topic
- 01/02/2026 11 pm - selesai implementasi untuk ambil data terbaru dengan wait_for()
- 02/02/2026 5 pm - selesai implementasi policy

- 02/02/2026 6 pm - selesai menambahkan parsing cli args sekalian sambil finishing
- 02/02/2026 8 pm - selesai finishing & testing, dan bikin README
- 02/02/2026 8 pm - release v1.0

2.2 Catatan

2.2.1 Brainstorming

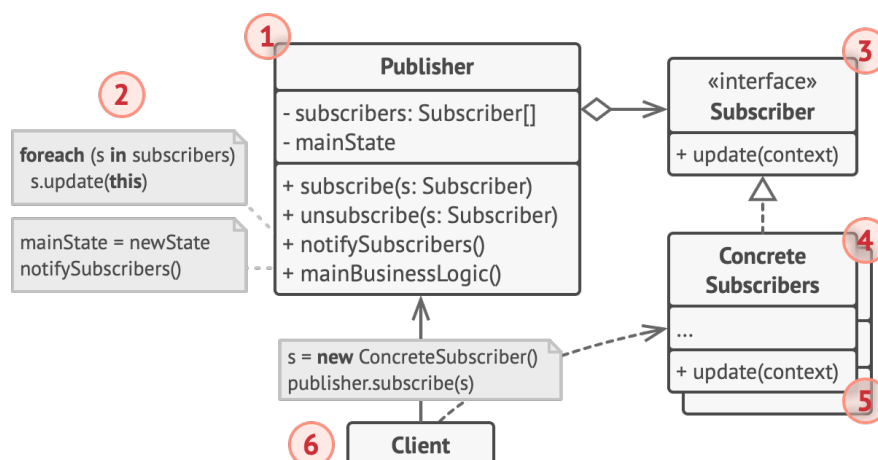
- Ada 3 drone simulator berjalan secara paralel (multithreading)
- Drone akan mengirim telemetry ke topic broker (publish-subscribe, drone publish data ke broker, dashboard subscribe data dari broker)
- Dashboard menampilkan telemetry real time, deteksi low battery dan telemetry stale.
- Pecah task dashboard. Ada yang handle data telemetry (dengan observer pattern), ada yang handle deteksi/alert (dengan strategy pattern)

2.2.2 RAII

RAII atau Resource Acquisition Is Initialization adalah konsep dalam pemrograman berorientasi objek yang menyatakan bahwa siklus hidup resource terikat secara langsung dengan masa hidup objek. Resource didapatkan (akuisisi) saat konstruksi objek dan dilepaskan secara otomatis saat destruksi objek.

Why use RAII? C++ tidak memiliki garbage collection otomatis. Garbage collection adalah proses internal yang melepaskan heap memory dan resource lain selama berjalannya program. Kegagalan melepaskan resource yang tidak dipakai disebut *leak* atau kebocoran memori. Resource yang bocor tidak tersedia untuk program lain hingga prosesnya berakhir.

2.2.3 Observer Pattern



Observer adalah pola desain yang menciptakan relasi one-to-many antar objek. Suatu objek utama dengan status yang menarik disebut dengan subject atau publisher. Objek-objek lain yang ingin melacak perubahan pada status objek utama disebut observers atau subscribers. Kelas publisher memiliki mekanisme subscription, yang terdiri atas sebuah array yang menyimpan referensi ke object-object subscriber dan beberapa metode public untuk operasi pada array subscribarnya. Setiap kali terjadi peristiwa/perubahan penting pada publisher, ia akan menghubungi subscribarnya dan memanggil metode notification khusus.

2.2.4 Strategy Pattern

Kita bisa bikin policy atau aturan, yang kalau dilanggar akan memicu alert. Pattern ini lebih baik daripada menulis kode kondisional di dashboard supaya dashboard ga diotak-atik dan mudah untuk kita menambahkan aturan baru. Strategy pattern akan menyepakati bentuk kontrak untuk mengecek alert (dengan method evaluate), tapi caranya bisa berbeda-beda (polymorphism).

3 Hands-on 2: MAVLink dan SITL

3.1 Log Progress

- 30/01/2026 4 pm - selesai install + setup, lanjut ngerjain metode manual
- 30/01/2026 6 pm - selesai rekam metode manual
- 31/01/2026 1 pm - lanjut ngerjain metode script/mavsdk
- 31/01/2026 11 pm - selesai rekam metode script
- 31/01/2026 11 pm - selesai upload video ke youtube dan upload script ke gist

3.2 Kendala dan Solusi

Kendala:

- awal” aku coba run kode boilerplate just to see berhasil connect atau ga, hasilnya error message ”Connection using udp:// is deprecated, please use udpin:// or udpout:// (cli_arg.cpp:28)”
- lalu aku coba udpin dan udpout tapi dua”nya masih ga connect
- aku coba cari tau port pake output list, kemudian aku copas untuk gantiin yang ada di kode. tapi kendalanya masih sama.

Solusi:

- Jujur ini bukan solusi sebenarnya karena aku masih gabisa connect. Lebih kayak solusi alternatif, karena deadline malam ini dan aku stuck di koneksi sekitar 3 jam.
- Solusi alternatifnya adalah dengan menggunakan pymavlink.
- Bagian ini akan diupdate jika solusi utama sudah ditemukan.

3.3 Catatan

Pymavlink adalah pustaka pemrosesan message MAVLink tingkat rendah, yang ditulis dalam Python. Pymavlink biasanya dipakai untuk mengimplementasikan komunikasi MAVLink di berbagai jenis sistem MAVLink, seperti GCS (MAVPROXY), Developer APIs (DroneKit), dan aplikasi MAVLink companion computer. Umumnya, MAVSDK-Python dan DroneKit-Python digunakan dalam aplikasi MAVLink untuk komunikasi dengan autopilot.

4 THT 2

4.1 Log Progress

- 03/02/2026 10 pm - start pengerjaan THT 2
- 03/02/2026 11 pm - selesai mengerjakan soal no 2 bagian a dan b

5 Referensi

- Design Patterns
- The Producer Consumer Pattern
- Using Pymavlink Libraries (mavgen)
- Object lifetime and resource management (RAII)
- PPT materi saat day pendidikan