Appendix G Applying Current SE Methods (HAZOP, STAMP, FRAM)

G.1 Applying HAZOP method

This process will attempt to apply HAZOP methods to solve two aspects. The first is articulating the problem, and the other is articulating the hazards associated with introducing the Eagle Drone to counter adversarial drone behaviour.

G.1.1 Problem articulation

In this section, we will be applying the HAZOP method for problem articulation prior to the introduction of the Eagle Drone. The HAZOP study requires an existing design and system schematics. In this instance, we will model the problem domain and use this as a schema to apply the HAZOP analysis. To initiate the articulation, we require a schematic that describes the system. The system itself is the problem. We will use a generic block diagram to model the interactions in the problem domain:

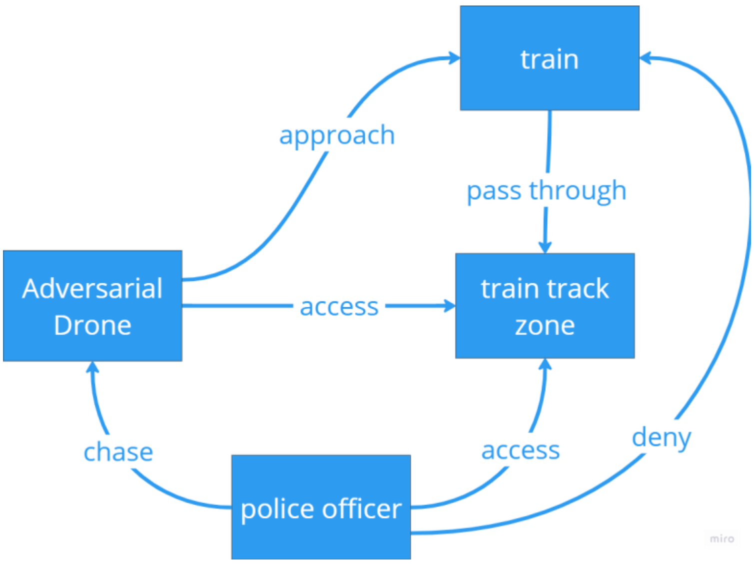

Figure G.1 control-based AIC-schema as assumed by HAZOP method

A system diagram illustrating interactions between different entities: a **train**, a **train track zone**, an **adversarial drone**, a **police officer**, and the relationships (or interactions) between them. Below is a list of all the possible interactions based on the diagram, which can be used for a HAZOP (hazard and operability) study.

1. Train – train track zone the train passes through the train track zone.
2. Adversarial drone – train track zone: the adversarial drone accesses the train track zone.

3. Police officer – adversarial drone: the police officer chases the adversarial drone.
4. Police officer – train track zone: the police officer gains access to the train track zone.
5. Adversarial drone – train: the adversarial drone approaches the train.
6. Police officer – train: the police officer denies train access.

To make HAZOP work as a problem articulation method, we will need to reinterpret the guidewords:

Table G.1 HAZOP guidewords meaning

Guideword	Adapted meaning
No	None of the individual systems achieve their desired intent.
More	Quantitative increase in a parameter
Less	Quantitative decrease in a parameter
As well as	An additional activity occurs.
Part of	Individual systems partially achieve the desired intention.
Reverse	The logical opposite of the individual systems design intention occurs.
Other than	Complete substitution occurs, or an unusual condition exists.
Where else	Applicable for flows, transfers, sources, and destinations
Before/after	Step (or part of it) is affected out of sequence.
Early/late	Timing is different from the intention.
Faster/slower	The step is done/not done with the right timing.

Next, we create the Table that captures the HAZOP process:

Table G.2 HAZOP process from problem articulation

Interaction	Guide word	Cause	Consequence	Safeguards	Actions
Train – train track zone	No	Obstruction on the track, equipment failure, signalling malfunction	Train delay, potential derailment if an object is hit	Surveillance cameras, track inspection protocols, remote monitoring	Ensure regular inspection and provide real-time obstruction alerts

Train – train track zone	Late	Delayed or advanced scheduling, signal failure	Increased risk of collision or hazards if other operations are unprepared	Automated scheduling and signalling systems, manual overrides	Additional scheduling checks, implement manual override protocols
Adversarial drone – train track zone	No	No deterrence by the track zone	Potential collision with a train, leading to damage or derailment	**Missing safeguards**	Increase enforcement measures and add automated drone interception mechanisms.
Adversarial drone – train track zone	More	Increased adversarial drone activity	Higher likelihood of a train-drone collision, train operations disruption	**Missing safeguards**	Implement stricter airspace controls and drone deterrence systems
Police officer – adversarial drone	Faster	Incorrect assessment of the drone's risk level	Officers may enter train zone without precautions, leading to safety risks.	**Missing safeguards**	Define threat levels and response protocols
Police officer – adversarial drone	Other than	Misidentification of the adversarial drone	Resources misallocated, delay in response to actual threat	**Missing safeguards**	Enhance identification accuracy with ai-assisted drone recognition
Police officer –	As well as	Miscommunication about train timing or lack of	High risk of collision,	Entry control systems, real-	Implement alert systems, define

train track zone		situational awareness	endangering officer	time tracking of officers	restricted access times
Police officer – train track zone	Early	Communication errors	Extended exposure to potential hazards in track zone	Access coordination, timing protocols	Review and improve timing protocols for entry
Adversarial drone – train	Reverse	Malicious intent, drone control malfunction	Potential collision, damage to train infrastructure, passenger risk	**Missing safeguards**	Enable police or automated systems to disable/intercept intruding drones
Adversarial drone – train	Faster	Adversarial tactics or uncontrollable drone behaviour	Increased likelihood of collision, reduced response time for mitigation	**Missing safeguards**	Implement rapid drone interception measures
Police officer – train	No	Restricted access, malfunction of access points	Inability to mitigate threat or access train for safety	Emergency access protocols, clear entry points	Install alternate entry systems, and enhance emergency accessibility
Police officer – train	Less	Inaccurate timing, restricted access	Insufficient time to respond to threats or ensure area safety	Communication systems, access timing controls	Improve timing and access protocols to ensure officer safety

G.1.2 Solution hazards identification

Applying the HAZOP method for analysing the hazards associated with the Eagle Drone system requirement. Applying the HAZOP method for analysing the hazards associated with the Eagle

Drone system requirement. In this section, we will re-examine HAZOP technique to analyse hazards associated with the introduction of the Eagle Drone with the mission to patrol the train track zone and inhibit adversarial drones:

Interactions:

1. Eagle Drone - adversarial drone: Eagle Drone physically inhibits the adversarial drone.
2. Eagle Drone-train track zone: Eagle Drone performs security patrol in the train tracks zone.
3. Eagle Drone-train: Eagle Drone avoids train.

Table G.3 HAZOP application to solution space

Interaction	Guide word	Cause	Consequence	Safeguards	Actions
HAZOP1: Eagle Drone - adversarial drone	**No:** the Eagle Drone does **none** of its expected detection work against an adversary drone.	Dual-sensor failure (EO camera and RF detector) due to power loss or severe EMI (>50 V/m). **EO camera** = Electro Optic, an onboard video camera that "sees" things. **RF detector** = a radio-frequency sensor that "hears" the enemy drone's radio signals.	An adversarial drone remains undetected, posing a risk of a proximity attack on the train.	• Independent EO and RF detection chains. The video camera and the RF detector run on entirely separate wiring and circuits. • Health-check watchdog separate from mission-check. A "watchdog" computer is always monitoring	• Hourly AI health-diagnostics and automatic reboot on failure • Hot-swap battery module with <5 s failover. If the main battery starts to fail or drop below a safe voltage, the drone doesn't lose power entirely, it seamlessly switches over to a backup.

				the health of the sensors themselves.	
HAZOP2: Eagle Drone - adversarial drone	**More:** When the Eagle Drone tries to fire its net to stop a bad drone, the net launcher pushed out with too much force, over 10 newtons (a unit of push/pull)	Engagement force >10 N (exceeds net-launcher limit) triggered by mis-calibrated actuator feedback.	Drone/net fragments damage infrastructure or fall onto tracks → derailment risk.	• Collision-avoidance LIDAR with 0.1 m resolution • Hard force-limit hardware (≤10 N). A mechanical or electrical “speed limiter” .	• Pre-flight environmental check (wind ≤ 20 km/h). Strong gusts can change how the net flies. • Automated force-calibration routine before each sortie. Right before takeoff, the drone runs a quick self-test to ensure its launcher is set to the exact right power level, neither more nor less. If it’s off by even a little, the drone won’t fly until it’s corrected.
HAZOP3: Eagle Drone -	**No** “No” here simply signals	GNSS multipath error (>2 m drift) causes	Blind-spot in surveillance: a little patch of ground the	• Multi-constellation GNSS + IMU dead-	• Fallback to “hover-and-scan” pattern on nav loss

train track zone	"complete absence" of the intended action. In this case, it flags the situation where the Eagle Drone does **not** patrol its assigned route at all.	navigation failure, skipping designated waypoints.	Eagle Drone never flies over. → unauthorised drone intrusion undetected. a bad-guy drone could slip in through that hole.	reckoning fusion. using all available satellite systems (GPS, Galileo, GLONASS, etc.) instead of just one. • Visual-landmark SLAM cross-check	• Real-time route re-uplink from ground station within 10 s.
HAZOP4: Eagle Drone – train track zone	**Other than:** instead of patrolling **inside** its security zone, the drone ends up **outside** it.	Operator waypoint mis-entry or GPS drift >2 m leads drone outside geo-fence.	Patrol gap in target zone → area vulnerable to adversarial ingress.	• Firmware-enforced geofence with boundary auto-correction. • Continuous geofence-breach sensor (≤1 s latency)	• Auto-return-to-last-valid-WP on breach. • Operator alert (≤2 s) with "confirm or cancel" override
HAZOP5: Eagle Drone – train track zone	Faster	Onboard speed command >5 m/s (due to tailwind or algorithm error).	Insufficient dwell time (<2 s per 10 m cell) → missed detection of small drones.	• Max-speed limiter firmware (5 m/s). • Coverage algorithm enforcing ≥2 s dwell per 10 m segment	• Dynamic speed adjust: speed = f(threat_score) × ≤5 m/s. • Tailwind compensation flag in autopilot

HAZOP6: Eagle Drone – train	No	Single-sensor (LiDAR) failure or GNSS spoofing prevents train-path detection.	Collision with train → major damage to both drone and train.	• Sensor fusion: LiDAR + radar altimeter + proximity ultrasonic. • Secure GNSS anti-spoof firmware	• Immediate hover-abort within 50 m of any track path upon mismatch. • Cross-check track-occupancy data from dispatch every 5 s
HAZOP7: Eagle Drone – train	Before	Local clock drift >5 s vs. train-schedule server (NTP lapse) → exit command delayed.	Drone crosses path during unexpected train arrival → collision risk.	• NTP-synced onboard clock (drift ≤5 s). • Real-time schedule-sync link (update ≤60 s)	• Automatic schedule pull every 60s with < 1s latency requirement. • Contingency exit-path pre-computed for ±10 s timing errors

G.1.3 Interpretation

HAZOP1	If both the ‘eye’ (video camera) and ‘ear’ (radio detector) of the Eagle Drone go dark—because of a jammed power line or heavy electronic noise, it won’t just sit there blind. It’s built with separate wiring, a watchdog computer watching its sensors, hourly self-tests with AI, and a backup battery that kicks in in under five seconds. All of that prevents it from going completely offline and allows it to continue spotting bad drones before they get too close to the train.
HAZOP2	More’ here means the net-gun pushed too hard, over its 10-Newton safe limit, because its power gauge was off. To stop that, the drone has a laser-based ‘radar’ to check it’s clear, a physical limiter so it literally can’t push harder, a wind-speed cut off so gusts don’t mess things up, and a pre-flight check that tunes the launcher exactly to 10 Newtons before it ever leaves the ground.

HAZOP3	If the Eagle Drone ever loses its spot on the map by more than a couple of metres, it won't just blindly keep flying. It has multiple backup methods, satellite mash-ups, motion sensors, and even camera-based landmark checks, to stay on track. If all of those falter, it will hover in place, look around, and ask the ground team for a fresh set of directions within ten seconds.
HAZOP4	Other than' here means the Eagle Drone has left its allowed zone entirely, either because someone typed in the wrong coordinates or the GPS puckered out by a couple of metres. To stop that, its built-in software won't let it cross the boundary and even nudges it back if it drifts, while a sensor checks every second for any slip. If it does sneak out, it auto-flies back to the last safe checkpoint, and the pilot on the ground is warned within two seconds so they can either OK its new course or pull it back
HAZOP5	Faster means the drone could zip along above its safe patrol speed of 5 m/s, either because of a gusty tailwind or a software glitch. To prevent missing small intruder drones, its firmware physically prevents it from exceeding 5 m/s, and its patrol plan ensures it hovers over every 10 m stretch for at least two seconds. Additionally, it dynamically slows down in suspicious areas based on a 'threat score' and automatically throttles back if the wind tries to push it too fast.
HAZOP6	No' here means the drone could completely lose track of where the rails are, either because its laser sensor died or someone spoofed its GPS. To prevent it from ploughing into a train, it uses three different 'eyes' (laser, radar, sound) fused together, plus GPS software that spots fake satellite signals. If any of those disagree when it's within 50 m of the tracks, it simply stops and hovers. On top of that, it asks the rail dispatch every five seconds which tracks are in use, so it always has a live, back-up safety check
HAZOP7	Before,' here means the drone could miss its window to clear the tracks because its clock or schedule info is off. To prevent this, it keeps its clock synced to network time (never off by more than 5 seconds) and refreshes the train timetable every minute (in under 1 second). On top of that, it pre-computes backup escape paths that work even if trains are up to ten seconds early or late, so it always gets out of the way in time.

G.1.4 Key definitions and requirements:

1. <2 s per cell: The drone must hover over each small square for at least 2 seconds to allow its cameras or sensors sufficient time to detect small, fast-moving targets.

2. >10 N: more than 10 Newtons of push, above the safety setting for the drone’s net gun.
3. 10 m cell: Imagine dividing the patrol area into squares 10 metres on a side.
4. Actuator: the little motor or piston that drives the net out.
5. Adversarial ingress: a bad-guy drone could slip in through that unmonitored hole.
6. Algorithm error: a bug or incorrect data in the drone’s speed-control software can inadvertently allow it to accelerate.
7. Automatic reboot on failure: If a fault is detected, the drone immediately reboots the computer associated with that sensor.
8. Automatic schedule pull every 60 s with <1 s latency: Once a minute, the drone downloads the latest timetable and must receive it within one second, so there’s no long delay updating.
9. Auto-return-to-last-valid-WP: If the drone ever actually steps outside the fence, it immediately flies back to the last waypoint it was known to be safely inside.
10. Boundary auto-correction: If the drone ever nudges toward the fence line, the firmware gently steers it back inside before it crosses.
11. Contingency exit-path pre-computed for ±10 s timing errors: Before flying, the drone calculates two extra “get-off-the-tracks” routes—one assuming it’s up to ten seconds early, another assuming it’s up to ten seconds late. If the train is off-schedule by up to ten seconds, the drone still has a safe path planned to clear the area in time.
12. Continuous geofence-breach sensor (≤1 s latency): A background check runs every second or faster, monitoring for any indication that the drone has actually left the safe zone. “≤1 s latency” means the system will notice any breach within one second.
13. Coverage algorithm: the higher-level routine that plans how the drone moves. It calculates the route in 10 m segments and makes sure the drone stays over each segment at least 2 seconds before moving on.
14. Drift (>2 m): the drone’s believed position can be off by more than two metres.
15. Dual-sensor failure: both of its key “eyes” go dark at once.
16. Dwell time: the duration the drone spends looking at a specific patch of ground.
17. Dynamic speed adjustment: The drone calculates a threat score, which indicates the perceived risk of the area (e.g., radar detects suspicious objects or cameras identify fast-moving targets).
18. EO camera = Electro-Optic camera: a daytime video camera that sees what’s in front of the drone.
19. Force-calibration routine: Right before takeoff, the drone runs a quick self-test, measuring the strength of its net launcher and adjusting until it reaches exactly 10 Newtons (no more, no less). If it can’t meet the tolerance, the mission is scrubbed until the launcher is repaired.

20. Fusion means the drone's computer combines the outputs of all three sensors. If one fails or lies (e.g. GPS spoofing), the others still "see" the tracks.
21. Geo-fence: a virtual fence in the drone's software that defines "inside the allowed area" vs "outside the area" by GPS coordinates.
22. Geo-fence: a virtual fence in the drone's software that defines "inside the allowed area" vs "outside the area" by GPS coordinates.
23. GNSS (Global Navigation Satellite System): the family of navigation satellites (like GPS, but also Europe's Galileo, Russia's GLONASS, etc.).
24. GNSS spoofing: Someone on the ground broadcasts fake satellite signals, tricking the drone's GPS into thinking it's somewhere else.
25. GPS drift >2 m: Natural errors in satellite positioning can cause the drone to think it's two metres away from its true location.
26. GPS drift >2 m: Natural errors in satellite positioning can make the drone think it's two metres away from its true spot.
27. Hard force-limit hardware: a mechanical or electrical limiter inside the launcher that physically can't exceed 10 Newtons.
28. Health-check watchdog vs. mission-check: There is a small, dedicated computer (the "watchdog") whose sole job is to monitor the health of the sensors (are they powered? are they responding?). Separately, another computer handles the mission tasks (navigation, engagement, etc.).
29. Hourly AI Health Diagnostics: Every hour, the drone runs a quick self-test (using onboard AI routines) to verify that the camera and RF detector are working correctly.
30. Hover-and-scan pattern: if the drone realises it has lost accurate position, it stops moving forward and hovers in place, slowly panning its sensors to look all around.
31. Immediate hover-abort within 50 m of track: If the fused sensor data and GPS disagree about the location of the tracks, and the drone is within 50 metres of the suspected track line, it stops forward flight and hovers in place.
32. IMU (Inertial Measurement Unit) dead-reckoning fusion: The IMU is the little package of accelerometers and gyros inside the drone that can track how it's tilted or turned. "Dead reckoning" refers to estimating position by tracking every movement from a known starting point. By blending IMU data with GNSS, the system can fill in the gaps when satellite signals become confusing.
33. Independent detection chains: The camera and the RF sensor each have their own power lines and circuit boards.
34. Firmware-enforced geofence: Firmware is the drone's low-level operating software (the code etched onto its control board). It constantly checks the drone's GPS position against the "inside" area, and will not let the drone cross the boundary in software.

35. LIDAR: like a laser-based radar. It sends out tiny laser pulses to measure distance.
36. Local clock drift >5 s: the drone's onboard clock has wandered by more than five seconds.
37. Max-speed limiter: This code prevents sending any "go faster" commands above 5 m/s, regardless of the autopilot or wind's attempts.
38. Mis-calibrated feedback: The system that measures how hard the actuator is firing is giving wrong readings, so it fires too strongly.
39. Multi-constellation GNSS: using all available satellite systems (GPS, Galileo, GLONASS, etc.) instead of just one. That makes you less likely to lose the lock or get misled.
40. Multipath error: when the drone's antenna picks up a satellite signal not just directly from space, but also after it's bounced off things on the ground (buildings, water, rock). These echoes confuse the receiver.
41. Newton (N): the basic unit of force, think of it as roughly the push needed to hold up a small paperback book against gravity.
42. NTP (Network Time Protocol): The Internet protocol that keeps computers' clocks in sync to within milliseconds.
43. NTP lapse: The drone hasn't synced its clock recently, so it's become inaccurate.
44. NTP-synced onboard clock: The drone automatically checks and corrects its own clock against an accurate time server often enough that it never drifts by more than five seconds.
45. Onboard speed command >5 m/s: the drone's own flight computer has set its maximum forward speed above that 5 m/s limit.
46. Operator alert (≤2 s): Within two seconds of that breach, a message pops up on the ground-station screen, "Drone just left its zone. Do you want to confirm its new path or cancel and have it return home?"
47. Operator waypoint mis-entry: someone typing the wrong GPS coordinates into the drone's mission plan (e.g. entering 51.500 N instead of 51.505 N).
48. Operator waypoint mis-entry: someone typing the wrong GPS coordinates into the drone's mission plan (e.g. entering 51.500 N instead of 51.505 N).
49. Patrol gap: a hole or uncovered patch in the area the drone is supposed to watch.
50. Power loss: the drone's battery or power line drops below the needed voltage.
51. Proximity ultrasonic: uses sound pulses (like a bat) to sense nearby objects.
52. Radar altimeter: bounces radio waves off the ground to measure height.
53. Real-time schedule-sync link: Every minute (at least), the drone requests updates from the train-schedule server, ensuring it always knows exactly when the next train is due.
54. RF detector = Radio-Frequency detector: a sensor that listens for the radio transmissions (control signals) of other drones.

55. Route re-uplink: Ground operators resend the correct flight path to the drone. "Within 10 s" means the drone won't sit clueless for more than ten seconds before it receives a fresh update.
56. Secure GNSS anti-spoof firmware: Special software that checks incoming satellite signals for tell-tale signs of fakery (wrong timing, odd signal strength, inconsistent satellite IDs) and rejects them if they look bogus.
57. Severe EMI (>50 V/m): Electromagnetic Interference stronger than 50 volts per meter—think a nearby radio jammer or lightning, which can fry or block electronics.
58. Tailwind compensation flag: The autopilot watches the wind speed and direction. If it detects a tailwind stronger than, say, 3 m/s, it reduces its throttle so the combined engine+wind speed never exceeds the 5 m/s cap.
59. Tailwind: wind blowing in the same direction as the drone is flying. A strong tailwind can propel the drone forward faster than its throttle setting allows.
60. Train-schedule server: the central computer that knows exactly when each train will pass each point.
61. Visual-landmark SLAM cross-check: SLAM stands for "Simultaneous Localisation And Mapping". In practice, the drone's camera recognises familiar features on the ground (trees, buildings, painted lines) and says, "Ah, this must be here," using that to correct its location estimate.
62. Waypoint (WP): a programmed "checkpoint" in the drone's flight plan, a GPS coordinate the drone is supposed to visit.
63. Waypoint (WP): a programmed "checkpoint" in the drone's flight plan, a GPS coordinate the drone is supposed to visit.
64. Waypoints: the "checkpoints" you program into the drone's flight plan.
65. Wind check: If gusts are stronger than 20 km/h, they don't launch the net, because hard winds can throw the net off course and make it hit harder than intended.

G.2 Applying STAMP/STPA method

In this session, we will attempt to apply the STAMP methodology to the same aspects as HAZOP and see whether methods help us to arrive to some black swan events. Black swan events are discovered from unexpected or unfamiliar interactions within the analysed complexity field.

G.2.1 Problem articulation

The assumption to be made here for STAMP to work as a problem articulation technique is that the problem controller is the system whose control actions are leading to the undesired complexity. In this case, we will imagine a virtual track zone safety controller (TZSC) a hypothetical control solution whose regulatory actions are inadequate to keep the track zone safe. As such, we can follow the STPA process:

G.2.1.1 Step 1: Define the purpose of the analysis

Purpose of the analysis: the objective is to assess and enhance the track zone safety controller (TZSC) in preventing unauthorised drone intrusion into a defined train track zone and ensuring the safety of people, infrastructure, and train operations. The TZSC aims to mitigate losses by controlling access to the track zone's ground and airspace.

Potential losses

1. **Loss of human life or injury:** unauthorised drones intruding into the track zone may collide with a train, staff, or bystanders, resulting in injury or fatality.
2. **Damage to property:** intruding drones might damage critical infrastructure, such as the train or track components, and surrounding structures.
3. **Disruption of train operations:** drone intrusion could force trains to halt, reroute, or delay, causing significant disruptions in railway service.
4. **Loss of security:** unauthorised drone presence increases the risk of security breaches, including unauthorised data access related to sensitive train operations.

System boundary

The boundary defines the entities included and excluded in the STPA analysis of the TZSC.

- **Included:**
    - **Train:** the vehicle operating within the track zone.
    - **Train track zone:** the designated ground and airspace surrounding the track area that the TZSC monitors.
    - **Police officers:** authorised personnel responsible for handling security breaches in the track zone.
    - **Railway traffic control system:** monitors and directs train movements, coordinating with the TZSC for safe train passage.
- **Excluded:**
    - **Passenger reactions** to delays or disruptions.
    - **Operations outside the immediate train track zone,** such as adjacent urban airspace.
    - **Unrelated drone activities** outside the scope of railway safety concerns.

Identifying system-level hazards

System hazards: the following hazards outline conditions under which the TZSC could fail to ensure track zone safety due to unauthorised drone intrusion:

1. **H1: open airspace for unauthorised aerial intrusion:** The TZSC architecture lacks airspace control capabilities by design, making drone intrusion inherently possible.
2. **H2: failure to trigger alerts or response actions to neutralise intrusion:** TZSC fails to communicate the drone threat to train control, risking unsafe train operations.
3. **H3: Communication failures between TZSC and railway traffic control system:** A failure in communication links between the TZSC and the railway control can result in trains moving through the track zone when a drone threat is present, risking collision.
4. **H4: Insufficient response authority for police officers:** police officers may lack the authority or technical capacity to neutralise unauthorised drones in the track zone. [This hazard refers to organisational constraints, not a technical failure.]

Identifying system-level constraints

We can define the constraints by understanding the hazards and their impacts. In the context of problem articulation, constraints mean solutions or mitigation of system-level hazards.

Table G.4 system-level constraints

Hazard	Description	Potential impact	System-level constraints
H1: open airspace for unauthorised aerial intrusion	Drone not detected due to design or system failure	Collision, security breach	Implement full-coverage detection within 200m, <2s latency, 95% confidence.
H2: failure to trigger alerts or responses	Alert not triggered after detection	Delayed police/train response	Alert must be issued in <1s and acknowledged by control staff
H3: communication failures with railway control	Communication with train control fails	Unsafe train movement	Comms must be redundant (N+1), with <200ms failover
H4: Insufficient police authority over drones	Police response was not effective due to external factors	Unresolved threat	(Contextual) Train operations depend on external law enforcement response capabilities

G.2.1.2 Step 2: Building the abstract functional control structure

In this model, we'll represent the hierarchical control structure for the track zone safety controller (TZSC) and its related components, breaking down control loops to identify where authority, feedback, control actions, and interactions occur to support the system's goals.

(a) Define the relevant controllers (entities)

The relevant controllers:

- **TZSC system controller:** manages and controls access to the track zone airspace and ground zone. It detects unauthorised drones, triggers alerts, and coordinates with police and railway traffic control.
- **Railway traffic control system:** this system controls train operations and coordinates with the TZSC to ensure that trains operate within a safe track zone.
- **Police response unit:** authorised personnel who act upon alerts from the TZSC to neutralise threats.
- **Surveillance system:** this system monitors and feeds real-time data about drone activity in the track zone to the TZSC.

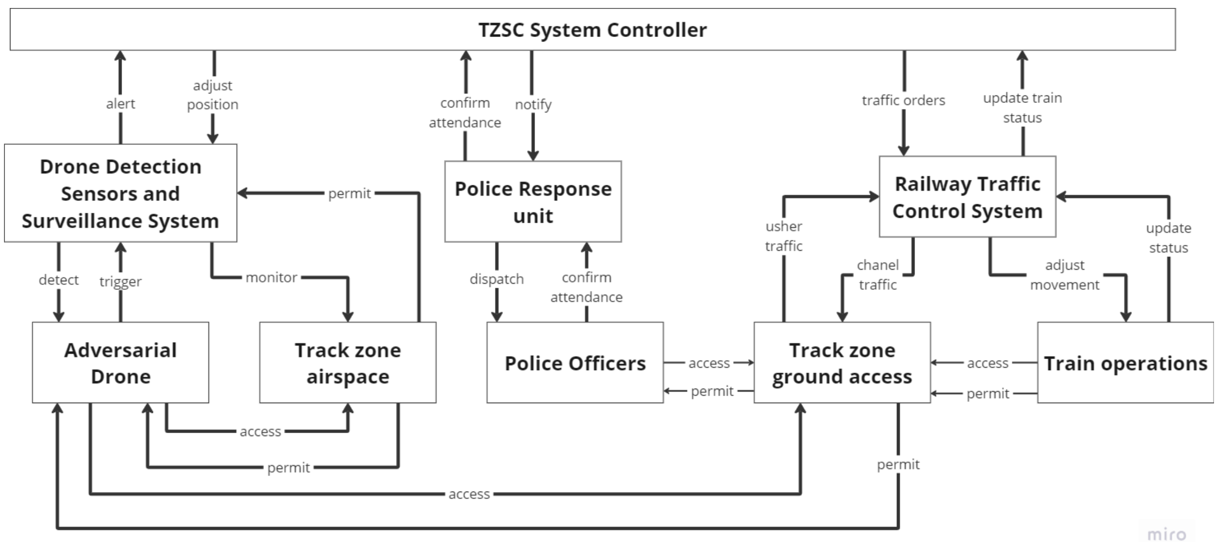

Figure G.2 track zone control structure

(b) Level 1: TZSC system controller (top-level controller)

The **TZSC system controller** is the primary authority responsible for overseeing and coordinating actions to maintain safety within the track zone. This controller manages:

- Detection and surveillance of unauthorised drones.
- Communication with the railway traffic control system and police response unit.

- Coordinate safety responses to ensure trains' safe operation and infrastructure protection.

(c) Level 2: supporting controllers

1. **Railway traffic control system**
    - Manages train movements within the track zone and works in coordination with the TZSC system controller to maintain operational safety by halting or rerouting trains if a drone threat is detected.
2. **Police response unit**
    - Responds to alerts from the TZSC to assess and neutralize drone threats in the track zone.
3. **Drone detection sensors and surveillance system**
    - Provides real-time data on drone activity in the track zone to the TZSC system controller, supporting early detection and monitoring.

(d) Level 3: controlled elements

1. **Adversarial drone**
    - The unauthorised drone risks the track zone by entering controlled airspace or accessing ground areas.
2. **Track zone airspace**
    - The designated airspace is monitored and controlled by the TZSC for intrusion prevention.
3. **Police officers**
    - Field personnel who respond to the scene upon alerts from the police response unit and take actions to mitigate drone threats.
4. **Track zone ground access**
    - The physical ground area of the track zone, controlled to prevent unauthorized access.
5. **Train operations**
    - The trains operating within the track zone, whose movements must be controlled to avoid collision risks in the presence of a detected drone threat.

(e) Control actions:

1. Drone detection sensors and surveillance systems detect adversarial drone.
2. Drone detection sensors and surveillance system alert TZSC system controller.

3. Drone detection sensors and surveillance system monitors track zone airspace.
4. Adversarial drone detection sensors and surveillance system .
5. Adversarial drone access, track zone ground access.
6. Adversarial drone access track zone airspace.
7. TZSC system controller traffic orders railway traffic control system.
8. TZSC system controller notify police response unit.
9. TZSC system controller adjust the position drone detection sensors and surveillance system.
10. Police response unit dispatch police officers.
11. Police response unit confirms attendance to the TZSC system controller.
12. Police officers confirm attendance at the police response unit.
13. Police officers access the track zone ground.
14. Train operations access track zone ground.
15. Train operations update status railway traffic control system.
16. Railway traffic control system update status TZSC system controller.
17. Railway traffic control system adjust movement train operations.
18. Railway traffic control system channel traffic track zone ground access.
19. Track zone ground access usher traffic railway traffic control system.
20. Track zone ground permit train operations.
21. Track zone ground permit police officers.
22. Track zone ground access permit adversarial drone.
23. Track zone air space permit adversarial drone .
24. Track zone air space permits drone detection sensors and surveillance systems.

(f) Control action analysis

Control action analysis and identification of unsafe actions. We consider scenarios where unsafe actions could lead to system failures or accidents for each control action. A Table of control action analysis and identification of unsafe actions. The Table evaluates each control action based on scenarios where unsafe actions could lead to failures or accidents. Each row describes a control action, and each column considers potential outcomes.

Table G.5 control action analysis

Control action	What can go wrong if issued?	What can go wrong if not issued?	Issued too soon/late?	Issued for too long/short?

1. Drone detection sensors and surveillance system detects adversarial drone	False positives could lead to unnecessary system responses.	Real drone threats remain undetected, risking security.	Late detection may lead to an accident.	Brief detection might fail to track drones accurately.
2. Drone detection sensors and surveillance system alerts TZSC system controller	An unnecessary alert overwhelms TZSC, leading to potential resource misallocation.	TZSC is unaware of the drone threat, so it is risking a delayed response.	Late alert delays response; early alert risks unnecessary resource allocation.	Too short alert may go unnoticed.
3. Drone detection sensors and surveillance system monitors track zone airspace	False detections can clutter the system with irrelevant data.	Unauthorised drones remain undetected in the airspace.	Late monitoring initiation misses early detections	Short monitoring intervals may miss drone re-entry
4. Adversarial drone trigger drone detection sensors and surveillance systems	N/a	The adversarial drone may exploit undetected access, remaining in the airspace.	Late detection allows for prolonged presence.	Short detection may lose track of evasive drones, and high-risk detection failures.
5. Adversarial drone access track zone ground-level (low-level flight up to a meter or so)	Drones pose a threat to ground infrastructure and personnel.	N/a	N/a	Short access may prevent counter-action

6. Adversarial drone accesses track zone airspace (above human reach)	Airspace is compromised, risking collisions with infrastructure or trains.	N/a	Early access enables extensive infiltration	Brief access might be missed; prolonged access increases response time.
7. TZSC system controller orders railway traffic control system	Unnecessary halts or reroutes could disrupt operations and lead to confusion.	Trains continue despite drone presence, risking collisions or other accidents.	Premature orders can cause train delays; late orders increase accident risk.	Short orders may result in trains resuming too soon and long-disrupted schedules.
8. TZSC system controller notifies police response unit	Unnecessary alerts could lead to response desensitization.	Delayed notification leaves the drone threat unaddressed, risking escalation.	Early notification risks resource allocation before confirmation.	Short alert prevents thorough response; prolonged alert desensitises responders.
9. TZSC system controller adjusts position of drone detection system	Position adjustment can reduce coverage of critical areas.	The drone may evade detection in an unmonitored area.	Early repositioning misses optimal coverage: delayed repositioning may miss detecting threats.	Short adjustments may not fully cover threats; long-term adjustments drain resources.
10. Police response unit dispatches police officers	Unnecessary dispatch strains police resources.	Delayed dispatch leads to a slower response, risking escalation.	Early dispatch may disrupt readiness for true threats; late delays response.	Brief deployment may fail to address threat; prolonged strains resources.

11. Police response unit confirms attendance to TZSC system controller	False confirmation could lead to safety complacency in the TZSC.	TZSC is unaware of the response status, risking coordination.	Early confirmation risks miscommunication: late confirmation increases uncertainty.	Short confirmation may not fully verify attendance; long confirmation may delay other actions.
12. Police officers confirm attendance to police response unit	False confirmation could disrupt response timing.	Police unit unaware of actual officer presence, risking coordination breakdown.	Early confirmation may lead to misalignment; late confirmation causes delays in response.	Short confirmation may cause insufficient information relay, and long delays.
13. Police officers access track zone ground access	Misinterpretation could lead to traffic interference with train operations.	Officers lack ground access, slowing down response to drones.	Early access disrupts train operations; late risks drone escape.	Brief access may hinder response; long access interferes with train movement.
14. Train operations access track zone ground access	Uncontrolled access risks collision with personnel or police in the zone.	Limited access hinders operational continuity.	Early access may interfere with police response; late affects train operations.	Short access delays train movement; prolonged access interferes with response.

15. Train operations update status to railway traffic control system	Excessive updates could lead to information overload and inefficiency.	Rtc lacks updated status, risking train safety and efficiency.	Early updates may lack accuracy; late updates reduce rtc's situational awareness.	Short updates lack details; long updates cause delays in real-time actions.
16. Railway traffic control system updates TZSC system controller	Excessive updates could overwhelm TZSC and reduce processing efficiency.	TZSC lacks updated status, risking ineffective decision-making.	Early updates may cause premature action; late updates reduce situational awareness.	Brief updates lack necessary data; prolonged updates delay response.
17. Railway traffic control system adjusts the movement of train operations	Excessive adjustments may lead to confusion and reduced operational reliability.	Delayed adjustments risk collision or accident.	Early adjustments may not be necessary; late could lead to accidents.	Short adjustment may be insufficient; prolonged adjustment disrupts train schedules.
18. Railway traffic control system channels traffic in track zone	N/a	N/a	Late control may lead to loss of income and penalties to the train operator.	N/a

19. Track zone ground access, usher traffic for the railway traffic control system	N/a	Lack of ushering (unclear or blocked signalling) risks collisions with unauthorised vehicles in the zone.	N/a	N/a
20. Track zone ground access permits train operations	Unnecessary permits increase risks of collision with ongoing responses.	Trains unable to access zones needed for operations.	Early permits lead to potential interference, and late permits disrupt train schedules.	Short permit may not allow necessary operations; long permit increases collision risk.
21. Track zone ground access permits police officers	Uncontrolled access risks interference with train schedules and security.	Officers are unable to access ground zones needed for response.	Early permits disrupt train operations; late permit risks losing drone.	A brief permit may not enable full response or prolonged interference with operations.
22. Track zone ground access permits adversarial drone	Allowing access would compromise safety and increase collision risk.	N/a	N/a	N/a
23. Track zone air space permits adversarial drone	Allowing access would increase collision risk with trains or police units.	N/a	N/a	N/a

24. Track zone air space permits drone detection sensors and surveillance system	N/a	Insufficient permits limit airspace monitoring and detection ability.	N/a	Short permits risk incomplete monitoring and prolonged waste of resources.

G.2.1.3 Step 3: causes of unsafe actions

Step 3: identify the causes of unsafe actions scenarios and safety constraints.

This section will consider only one action to examine the scenario that led to the unsafe control action:

3. Drone detection sensors and surveillance system monitors track zone airspace

Below is a Table that examines the causes and safety constraints:

Table G.6 unsafe actions and safety constraints

Unsafe action	Causes	Mitigation (safety constraints)
False detections clutter the system with irrelevant data	Sensor malfunctions, environmental interferences (e.g., birds, weather), and poor sensor calibration.	- implement advanced filtering algorithms to distinguish drones from false positives.
		- regularly calibrate and maintain sensors to minimise errors.
		- machine learning models are used to enhance differentiation between threats and non-threats.
Unauthorised drones remain undetected in the airspace	Insufficient sensor coverage, delayed response to detection events, and inadequate sensitivity of detection systems.	- expand sensor coverage to eliminate blind spots.
		- implement continuous, real-time detection protocols.
		- increase detection sensitivity while minimising false positives through adaptive sensor calibration.
Late monitoring initiation	Communication delays, manual setup required, or	- automate monitoring activation linked to proximity alerts.

misses early detections.	ineffective automated response protocols.	- establish protocols for immediate monitoring activation upon potential threat identification.
		- ensure real-time communication and minimise human intervention for quicker response.
Short monitoring intervals may miss drone re-entry; long intervals drain resources	Resource allocation constraints, insufficient monitoring capabilities, and incorrect interval configuration.	- optimize interval settings based on historical drone activity patterns.
		- balance intervals with resources, using event-based monitoring for more targeted detection.
		- implement predictive analytics to adjust intervals dynamically based on real-time threat analysis.

G.2.2 Solution hazards identification

Applying the STAMP method for analysing the hazards associated with the Eagle Drone system requirement. We will investigate how STAMP will perform by analysing the following requirements:

Eagle Drone -train track zone: Eagle Drone performs security patrol in the train tracks zone.

G.2.2.1 Step 1: purpose of the analysis:

This analysis aims to assess potential hazards in the Eagle Drone's operation, identifying constraints to mitigate these hazards and ensure safe, secure patrol in the train track zone.

Define potential losses

Potential losses involve scenarios where the Eagle Drone fails to secure the train track zone, leading to the following consequences:

- **Loss of security**: unauthorized drones breach the train track zone.
- **Loss of train safety**: drones are not detected, potentially impacting train safety.
- **Loss of operational integrity**: the Eagle Drone system malfunctions, impacting its autonomous patrolling ability.
- **Loss of public confidence**: increased risk due to perceived vulnerability of track security.

Define system of interest boundary

The boundary clarifies the entities within and outside the analysis for the Eagle Drone STPA (system-theoretic process analysis):

Included:

- **Eagle Drone system**: autonomous drone hardware, computer vision system, software algorithms, onboard sensors.
- **Communication systems**: drone's communication with base station, alert systems for notifying intrusions.
- **Adversarial drone detection**: mechanisms for identifying, tracking, and responding to adversarial drones.
- **Train track zone environment**: physical boundaries of the train tracks and environmental sensors along the track.

Excluded:

- **Train operations**: direct control over trains and track signal systems.

Identifying system-level hazards

System hazards outline conditions where the Eagle Drone might fail to maintain secure track zone surveillance, thereby allowing potential unauthorised intrusions:

- **H1**: Eagle Drone fails to detect an intruding adversarial drone, causing a train track zone security breach.
- **H2**: Eagle Drone misidentifies a non-hostile object as an adversarial drone, leading to false alarms and operational disruptions.
- **H3**: Eagle Drone loses communication with the base station, compromising its ability to receive updates or report incidents.
- **H4**: a malfunction in the Eagle Drone's computer vision system results in compromised detection accuracy, reducing the effectiveness of patrol operations.
- **H5**: Eagle Drone's navigation system malfunctions, causing it to exit the train track zone or collide with objects.

Identifying system-level constraints

The constraints are defined to mitigate identified hazards, ensuring that the Eagle Drone operates safely and reliably:

Table G.7 system-level constraint identification

Hazard	Description	Potential impact	System-level constraints
H1: failure to detect an intruding adversarial drone	The Eagle Drone fails to identify unauthorised drones within the track zone.	Unauthorised drone access could compromise track security, potentially leading to train collisions or disruptions.	**C1:** Eagle Drone must continuously monitor and detect any unauthorised drones within the defined train track zone to avoid security breaches.
H2: misidentification of non-hostile objects	The Eagle Drone incorrectly classifies a non-hostile object as a threat, triggering a false alarm.	False alarms cause unnecessary operational responses, potentially straining resources and reducing system reliability.	**C2:** the Eagle Drone's identification system must distinguish accurately between adversarial and non-hostile objects to prevent false alarms.
H3: communication loss with base station	Communication between the Eagle Drone and the base station is interrupted.	Loss of situational awareness, inability to report incidents, or delayed response times.	**C3:** the communication link between the Eagle Drone and the base station must be maintained to ensure situational awareness and data updates.
H4: malfunction in computer vision system	The computer vision system fails to function accurately, potentially due to environmental factors or system errors.	Reduced detection accuracy, which may result in undetected threats or increased false positives.	**C4:** the Eagle Drone's computer vision and detection algorithms must maintain high accuracy even in adverse environmental conditions.
H5: navigation system malfunction	The navigation system causes the Eagle Drone to leave the train track zone or	Increased risk of accidental damage, system loss, or intrusion	**C5:** the Eagle Drone's navigation system must be reliable and constrained to operate strictly within the train track zone to

	collide with objects.	into unauthorised zones.	avoid unintended exits or collisions.

G.2.2.2 Step 2: abstract functional control

Step 2) building the abstract functional control structure for the Eagle Drone

To build the abstract functional control structure for the Eagle Drone's patrolling of the train track zone, we'll identify relevant controllers across different hierarchical levels, defining the key entities and control actions involved. This hierarchical control structure models how the Eagle Drone's systems interact and control each function, focusing on its mission to detect and respond to potential adversarial drones.

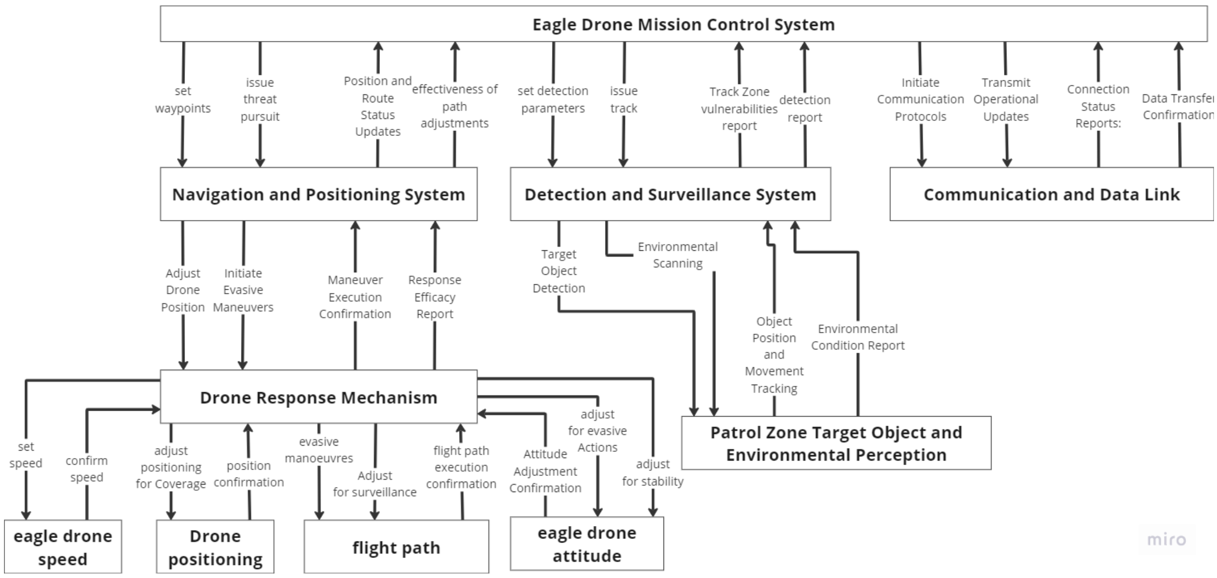

Figure G.3 Eagle Drone control structure

(a) relevant controllers (entities)

(b) Level 1: top-level controller

Eagle Drone mission control system: this high-level controller manages the overall objectives and rules for the Eagle Drone's operations. It determines patrol routes, sets detection and identification parameters, and provides directives for responding to adversarial drones. It also oversees the Eagle Drone's compliance with mission protocols and receives status updates on detected intrusions or anomalies.

Key control actions:

Eagle Drone mission control system → detection and surveillance system

Control actions:

- **Set detection parameters**: mission control sets the detection parameters, such as sensitivity levels, target classifications, and response protocols.
- **Issue threat response commands**: when a threat is detected, mission control may issue commands for the detection and surveillance system to track or monitor the identified object more closely.

Feedback actions:

- **Detection reports**: the detection and surveillance system sends regular updates or alerts about objects detected in the area, including threat level, location, and type.
- **Status of surveillance coverage**: feedback on current surveillance coverage and any detected anomalies, blind spots, or system limitations.

This feedback allows mission control to adjust detection parameters or change the drone's position for optimal coverage.

Eagle Drone mission control system → communication and data link

Control actions:

- **Initiate communication protocols**: mission control sets up and maintains communication protocols with the drone, ensuring reliable data transfer rates and encryption settings.
- **Transmit operational updates**: mission control sends updates on mission parameters, changes in detection thresholds, or new response protocols to the communication and data link.

Feedback actions:

- **Connection status reports**: the communication and data link system provides updates on connection quality, latency, and any communication issues that could impact real-time data exchange.
- **Data transfer confirmation**: confirmation that data, such as detection alerts or surveillance updates, has been successfully transmitted to mission control.

This loop enables mission control to monitor and maintain secure, high-quality communication with the drone for real-time situational awareness and adaptive control.

Eagle Drone mission control system → navigation and positioning system

Control actions:

- **Set patrol routes and waypoints**: mission control assigns specific routes, waypoints, or patrol zones for the drone's navigation.
- **Adjust speed and altitude parameters**: mission control can modify the drone's speed or altitude to adapt to mission needs, obstacles, or potential threats.
- **Issue positioning commands in response to threats**: mission control may command the drone to adjust its position or move closer to investigate based on detected threats or anomalies.

Feedback actions:

- **Position and route status updates**: the navigation and positioning system provides feedback on current location, speed, altitude, and any deviations from the assigned patrol path.
- **Environmental and obstacle feedback**: reports on detected obstacles, environmental factors (e.g., wind or weather conditions), and the effectiveness of path adjustments in maintaining surveillance coverage.

This loop allows mission control to adjust the drone's positioning dynamically, ensuring it remains within the patrol area and is optimally positioned to detect unauthorised objects.

(c) Level 2: supporting controllers

Navigation and positioning system → drone response mechanism

Control actions:

- **Adjust drone position**: the navigation and positioning system provides specific positioning adjustments to the drone response mechanism, such as new coordinates, altitude adjustments, or re-routing to maintain coverage or avoid obstacles.
- **Initiate evasive manoeuvres**: if an object or threat is detected near the drone's patrol path, the navigation and positioning system might direct the drone response mechanism to initiate evasive manoeuvres to prevent a collision or avoid detection.

Drone response mechanism → navigation and positioning system

Feedback actions:

- **Manoeuvre execution confirmation**: the drone response mechanism sends feedback confirming the successful execution of evasive or defensive manoeuvres. This ensures the navigation and positioning system knows the drone has responded effectively.
- **Response efficacy report**: feedback is provided on the effectiveness of the response, such as whether the manoeuvre successfully avoided an obstacle or maintained surveillance coverage.

Detection and surveillance system → patrol zone target object and environmental perception

Control actions:

- **Target object detection**: using onboard sensors and computer vision, identify and isolate potential adversarial drones or unauthorised objects within the patrol zone.
- **Environmental scanning**: monitor environmental elements (e.g., weather conditions, lighting changes, and presence of non-hostile objects) that may affect the accuracy and reliability of surveillance.

Patrol zone target object and environmental perception → detection and surveillance system

Feedback actions:

- **Environmental condition report**: provide real-time updates on environmental conditions (e.g., sudden weather shifts, changes in lighting) that could impact detection capability or sensor accuracy.
- **Object position and movement tracking**: provide continuous updates on the positioning and movement of detected objects within the patrol zone to the detection and surveillance system, enabling ongoing tracking and reassessment.

(d) Level 3: controlled subsystems

Drone response mechanism → Eagle Drone speed

Control actions:

- **Adjust speed in response to threats**: increase or decrease speed based on threat proximity. For example, if an intruding drone is detected nearby, speed may be increased to either evade or pursue the intruder.

Feedback actions:

- **Confirm speed adjustment**: the system confirms whether the speed adjustment was successfully executed and if it achieved the desired operational effect (e.g., quicker response or improved stability).

Drone response mechanism → drone positioning

Control actions:

- **Position adjustment for coverage**: adjust positioning to keep the drone within the designated patrol zone or move to areas requiring more focused surveillance.

Feedback actions:

- **Position confirmation**: this confirms that the drone has moved to the instructed position, enabling accurate tracking of its location.

Drone response mechanism → flight path

Control actions:

- **Adjust flight path for evasive manoeuvres**: initiate evasive manoeuvres by altering the current flight path to avoid detected intrusions or obstacles.
- **Flight path optimization for surveillance**: modify the flight path to optimise surveillance coverage, ensuring critical areas are regularly patrolled.

Feedback actions:

- **Flight path execution confirmation**: confirmation that the modified path was successfully adopted or, if not, provide details on the issue encountered (e.g., obstacle interference).

Drone response mechanism → Eagle Drone attitude (orientation and stability)

Control actions:

- **Adjust attitude for stability in adverse conditions**: adjust orientation (pitch, roll, and yaw) to maintain sTable flight, especially in turbulent conditions or during complex manoeuvres.
- **Orientation adjustment for surveillance focus**: change the drone’s attitude to better angle sensors or cameras toward areas of interest, improving detection accuracy.

- **Manoeuvre attitude adjustments for evasive actions**: quickly adjust orientation when performing rapid evasive or defensive manoeuvres to maintain balance and avoid destabilization.

Feedback actions:

- **Attitude adjustment confirmation**: confirm that the attitude change was successfully implemented, supporting a sTable and responsive flight.

Control actions

1. Eagle Drone mission control system set detection parameters detection and surveillance system
2. Eagle Drone mission control system issue threat response commands detection and surveillance system
3. Detection and surveillance system detection reports Eagle Drone mission control system
4. Detection and surveillance system status of surveillance coverage Eagle Drone mission control system
5. Eagle Drone mission control system initiate communication protocols communication and data link
6. Eagle Drone mission control system transmit operational updates communication and data link
7. Communication and data link connection status reports Eagle Drone mission control system
8. Communication and data link data transfer confirmation Eagle Drone mission control system
9. Eagle Drone mission control system set patrol routes and waypoints navigation and positioning system
10. Eagle Drone mission control system adjust speed and altitude parameters navigation and positioning system
11. Eagle Drone mission control system issue positioning commands in response to threats navigation and positioning system
12. Navigation and positioning system position and route status updates Eagle Drone mission control system
13. Navigation and positioning system environmental and obstacle feedback Eagle Drone mission control system
14. Navigation and positioning system adjust drone position drone response mechanism
15. Navigation and positioning system initiate evasive manoeuvres drone response mechanism

16. Drone response mechanism manoeuvre execution confirmation navigation and positioning system
17. Drone response mechanism response efficacy report navigation and positioning system
18. Detection and surveillance system target object detection patrol zone target object and environmental perception
19. Detection and surveillance system environmental scanning patrol zone target object and environmental perception
20. Patrol zone target object and environmental perception environmental condition report detection and surveillance system
21. Patrol zone target object and environmental perception object position and movement tracking detection and surveillance system
22. Drone response mechanism adjust speed in response to threats Eagle Drone speed
23. Eagle Drone speed confirm speed adjustment drone response mechanism
24. Drone response mechanism position adjustment for coverage drone positioning
25. Drone positioning position confirmation drone response mechanism
26. Drone response mechanism adjust flight path for evasive manoeuvres flight path
27. Drone response mechanism flight path optimization for surveillance flight path
28. Flight path flight path execution confirmation drone response mechanism
29. Drone response mechanism adjust attitude for stability in adverse conditions Eagle Drone attitude (orientation and stability)
30. Drone response mechanism orientation adjustment for surveillance focus Eagle Drone attitude (orientation and stability)
31. Drone response mechanism manoeuvre attitude adjustments for evasive actions Eagle Drone attitude (orientation and stability)
32. Eagle Drone attitude (orientation and stability) attitude adjustment confirmation drone response mechanism

(e) control action analysis and identification of unsafe actions

Table G.8 control action analysis

Control action	What can go wrong if issued?	What can go wrong if not issued?	Issued too soon/late?	Issued for too long/short?
1. Set detection parameters	Improper parameters could lead to false alarms or missed threats.	The drone may operate with default or outdated settings, reducing detection accuracy.	If issued too late, initial threats may go undetected; too soon may cause unprepared detection criteria.	N/a
2. Issue threat response commands	Unnecessary commands could lead to resource drain or overreaction to non-threats.	No response to threats, leaving the patrol zone vulnerable to intrusions.	Late response may allow the threat to be evaded; early response may cause unnecessary actions.	Too long may cause continuous threat response, draining resources; too short could be insufficient to mitigate threats.
3. Detection reports	Excessive reports may overload mission control with data.	Mission control lacks real-time threat data, increasing vulnerability.	Delayed reports could hinder timely threat response; early reports may be inaccurate.	N/a

4. Status of surveillance coverage	Inaccurate coverage reports may lead to undetected gaps in surveillance.	Mission control may remain unaware of surveillance gaps and missing intrusions.	Late coverage status might mean undetected threats; too soon may provide misleading data.	N/a
5. Initiate communication protocols	Incorrect protocol could risk data integrity or security breaches.	No communication means the drone is isolated and unable to receive mission updates.	Late initiation can delay mission start; early may cause premature data transmission issues.	N/a
6. Transmit operational updates	Incorrect updates can cause mission deviation or operational error.	Drone operates on outdated information, risking mission failure.	Late updates risk obsolete data transmission; early may disrupt current operations.	N/a
7. Connection status reports	Incorrect status may give a false sense of secure communication.	Mission control remains unaware of connection issues, risking data gaps.	Delayed status could mask critical lapses; early reports of issues may be premature.	Status held too long can mask real-time problems; too short, limits connection awareness.

8. Data transfer confirmation	Over-reporting may congest communication lines.	Data is not verified as received, risking mission-critical gaps.	Delayed confirmation could create mission data uncertainty; early may not confirm actual completion.	Too long can delay the next transmission; too short may miss data verification.
9. Set patrol routes and waypoints	Inaccurate routes may lead the drone out of the patrol zone.	Drone operates without assigned route, risking coverage gaps.	Late issuance may delay patrol start; early may disrupt other setup tasks.	Routes held too long may ignore dynamic threats; too short routes risk incomplete patrol.
10. Adjust speed and altitude parameters	Incorrect adjustments may destabilise the drone.	Drone may not respond effectively to threats or environmental challenges.	Late adjustments may cause the drone to miss critical positioning;	Holding too long risks stability; being too short may not allow for adaptation to needed altitudes.
11. Positioning commands in response to threats	Premature or excessive positioning changes may compromise other operations.	No adjustment leaves drones vulnerable to threats.	Late positioning risks delayed response; too early could trigger a response unnecessarily.	N/a

12. Position and route status updates	Excessive updates may clutter mission control.	Control lacks real-time navigation status, risking position error.	Late updates risk delayed response; too early may provide premature information.	N/a
13. Environmental and obstacle feedback	Misreported obstacles may lead to unnecessary manoeuvres.	Drone lacks situational awareness, increasing collision risk.	Late feedback may miss real-time obstacles; too early may not reflect current path.	N/a
14. Adjust drone position	Premature adjustment may impact coverage.	Drones stay in ineffective positions, compromising area surveillance.	Delayed position change may leave the threat unchecked; too early may miss optimal positioning.	N/a
15. Initiate evasive manoeuvres	Unnecessary evasion may waste resources and destabilise the drone.	No evasion leaves the drone exposed to threats or collision.	Late evasion increases collision risk; too early might avoid non-threatening objects.	N/a
16. Manoeuvre execution confirmation	Incorrect confirmation may falsely suggest a successful action.	Control assumes manoeuvre failed, risking additional or unnecessary commands.	Delayed confirmation could cause redundant commands; too early may	N/a

			give a false signal.	
17. Response efficacy report	Misreporting may mislead on manoeuvre success.	No feedback limits the assessment of manoeuvre effectiveness.	Late reports may lead to missed follow-up; early may reflect inaccurate responses.	N/a
18. Target object detection	Misidentification could result in false alarms or missed threats. [false positive]	No detection leaves unauthorised objects unmonitored in the zone. [false negative]	Delayed detection can allow intrusion; too early may trigger false positives.	N/a
19. Environmental scanning	Excessive scanning can cause false environmental alerts.	Missed environmental cues can lead to detection errors.	Delayed scanning may miss changes; too early can lead to unnecessary responses.	Short scans may miss critical changes.
20. Environmental condition report	Misreporting may mislead the detection system on real conditions.	Unawareness of changes risks detection errors and response delay.	Late reports may not account for real-time changes; early may pre-empt environmental shifts.	N/a

21. Object position and movement tracking	Incorrect tracking may mislead system response.	No tracking limits threat anticipation and response timing.	Late tracking risks losing object data; early may initiate too soon.	N/a
22. Adjust speed in response to threats	Speed misadjustment risks collision or resource waste.	Drone remains at ineffective speed, risking failure to intercept or evade threats.	Delayed adjustment limits response effectiveness	Speed held too long may destabilise; too short may fail to achieve response objective.
23. Confirm speed adjustment	Incorrect confirmation may mislead on speed status.	Unconfirmed speed leaves mission control unaware of drone readiness.	Late confirmation can delay response timing; too early may mislead on final speed.	N/a
24. Position adjustment for coverage	Premature adjustment risks missing critical zones.	No adjustment leaves blind spots, risking intrusion.	Late adjustment misses timely response; early may move out of position.	N/a
25. Position confirmation	Incorrect confirmation may mislead positioning status.	Control assumes position adjustment failed, risking redundant commands.	Late confirmation delays next actions; early may mislead on exact location.	N/a

26. Adjust flight path for evasive manoeuvres	N/a	Failure to adjust risks collision or exposure.	Late adjustment limits response	N/a
27. Flight path optimization for surveillance	N/a	Poor pathing reduces patrol effectiveness and coverage.	Late path change risks surveillance gap; too early may change from the optimal zone.	Paths that are too long cause resource strain; paths that are too short may skip essential zones.
28. Flight path execution confirmation	Incorrect confirmation may suggest a faulty path is in place.	No confirmation causes mission control to question path integrity.	Late confirmation can cause redundant commands; too early may mislead.	N/a
29. Adjust attitude for stability	Misadjustment risks destabilization or mission drift.	No attitude adjustment leaves the drone unsTable, risking the mission.	Late attitude may cause a loss of balance; too early may be unnecessary.	Holding too long wastes energy; being too short may not stabilise completely.
30. Orientation adjustment for surveillance focus	Incorrect adjustment may divert from target areas.	No orientation adjustment limits surveillance accuracy.	Late orientation adjustment risks losing focus; too early may reduce flexibility.	Too long can waste resources; too short may not acquire the full target.

31. Manoeuvre attitude adjustments for evasive actions	Incorrect adjustment risks destabilising flight path.	No attitude change limits evasive manoeuvre effectiveness.	Late adjustment risks collision; too early may trigger without threat.	N/a
32. Attitude adjustment confirmation	Misreporting may suggest false stability.	Control unaware of an attitude adjustment, risking stability concerns.	Delayed confirmation limits real-time response; too early may mislead on attitude status.	N/a

G.2.2.3 Step 3: causes of unsafe actions

Step 3: identify the causes of unsafe actions scenarios and safety constraints

The Table below for step 3, examines the causes and mitigation (safety constraints) for the unsafe action related to the **object position and movement tracking** when detecting an adversarial drone:

Table G.9 Causes of unsafe actions scenarios

Unsafe action	**Causes**	**Mitigation (safety constraints)**
Incorrect object position and movement tracking	- sensor calibration errors	- implement regular sensor calibration protocols
	- algorithmic misinterpretation of data	- use advanced algorithms with machine learning to improve accuracy
	- environmental interference (e.g., weather conditions, obstacles)	- develop redundancy in sensor types to minimise environmental impact
No object position and movement tracking	- sensor malfunction or failure	- establish robust system checks to ensure sensors are operational
	- system not activated or improperly conFigured	- implement fail-safes to activate tracking systems automatically

	- communication failures with the detection system	- regularly test communication links between systems
Late object position and movement tracking	- processing delays in data analysis	- optimize data processing algorithms for faster analysis
	- communication latency between systems	- prioritize critical alerts and streamline data communication protocols
	- overloading of the system with data inputs	- set limits on incoming data to reduce processing delays
Early object position and movement tracking	- premature activation of tracking algorithms	- define clear thresholds for activation of tracking systems
	- misinterpretation of sensor data as an immediate threat	- implement a confirmation process to validate detected threats before tracking
	- lack of a threshold for detection	- use historical data to refine tracking algorithms for better accuracy

G.3 Applying the FRAM method

In this section, we will reuse the problem brief model as we discussed in HAZOP section with the following interactions:

1. Train – train track zone the train passes through the train track zone.
2. Adversarial drone – train track zone: the adversarial drone accesses the train track zone.
3. Police officer – adversarial drone: the police officer chases the adversarial drone.
4. Police officer – train track zone: the police officer gains access **t**o the train track zone.
5. Adversarial drone – train: the adversarial drone approaches the train.
6. Police officer – train: the police officer denies train access.

The aim is to discover any further complications due to functional resonance that we may not have thought about otherwise in the list of interactions.

G.3.1 Step 1: identify and describe functions

Here's how we can identify and describe the critical functions involved in the given interactions:

1) **Function 1: train transit people and goods:** the train operates along its designated track, following set schedules and safety protocols. Its primary function is safely transporting passengers or goods through the train track zone.

2) **Function 2: train track zone facilitating safe passage for trains:** ensures trains can operate smoothly and safely, minimising the risk of obstacles or intrusions that could lead to delays or accidents.
3) **Function 3: police officer chase adversarial drone:** the police officer actively pursues the adversarial drone, using equipment or tactical manoeuvres to intercept and neutralise the threat posed by the drone.
4) **Function 4: adversarial drone approach train:** the adversarial drone manoeuvres towards the train, possibly to disrupt operations, deliver a payload, or capture sensitive information.

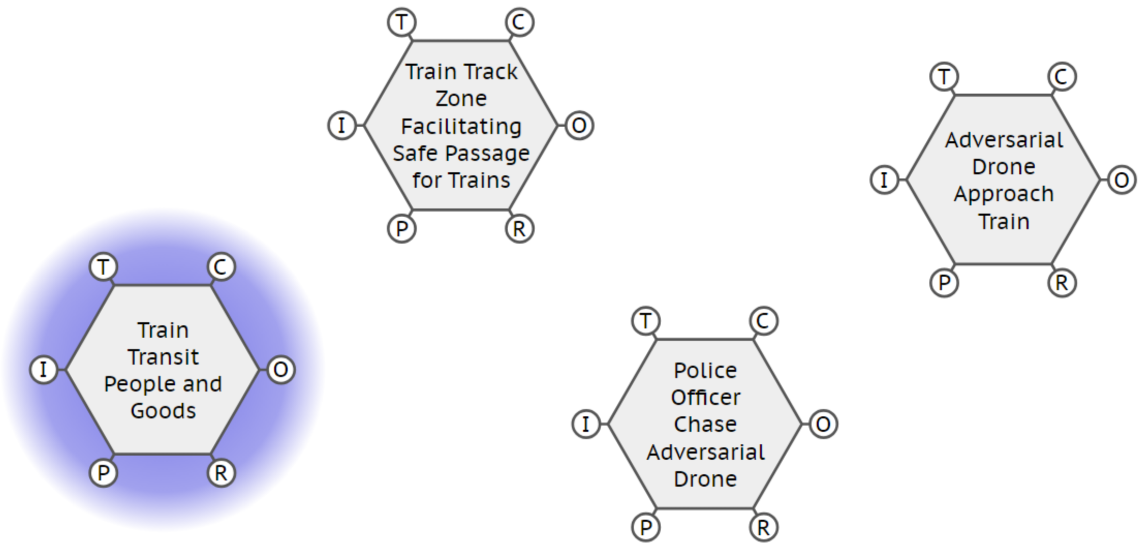

Figure G.4 main functions involved in the problem domain

Then describe each function using six aspects:

Using the six aspects to describe each function in the problem interactions further, here's an in-depth breakdown:

Function 1: train transit people and goods

- **Input (i)**: scheduled train passage time and route plan.
- **Output (o)**: safe arrival at the destination with passengers and goods.
- **Precondition (p)**: clear and secure track for train transit, the train is operational, and safety systems are active.
- **Resource (r)**: electric powerlines.
- **Control (c)**: train driver, railways traffic control, speed limits, and signal systems.
- **Time (t)**: scheduled time of passage through the train track zone.

Function 2: train track zone facilitating safe passage for trains

- **Input (i)**: presence of a train, scheduled train passage.
- **Output (o)**: clear and secure track for train transit.
- **Precondition (p)**: track inspection is complete, and no unauthorised objects or people are within the zone.
- **Resource (r)**: surveillance systems, fencing, track zone infrastructure.
- **Control (c)**: railway safety regulations.
- **Time (t)**: n/a.

Function 3: police officer chase adversarial drone

- **Input (i)**: alert of an adversarial drone in the train track zone.
- **Output (o)**: neutralisation or removal of the drone threat.
- **Precondition (p)**: the drone's location is detected, and police access to the train track zone is granted.
- **Resource (r)**: police officer, communication systems.
- **Control (c)**: law enforcement protocols, safety procedures for high-risk zones.
- **Time (t)**: 12 minutes to respond.

Function 4: adversarial drone approach train

- **Input (i)**: train schedule, route to target track zone plan.
- **Output (o)**: collision with train.
- **Precondition (p)**: clear train track zone, undetected or unimpeded approach.
- **Resource (r)**: GPS, well-trained adversarial drone perception, functional wireless communication network, full battery life.
- **Control (c)**: adversarial drone pilot.
- **Time (t)**: 30 minutes flight.

Defining background functions:

One helpful aspect of FRAM is the definition of background and foreground functions. Usually, the background functions are resources or conditional inputs (givens) or expected outcomes where the assumption is not to define p, r, c, or t. To do so, lets reiterate the problem brief:

> An anonymous drone intruded into a bounded train track zone while the train passed by. Police officers were called to the scene and tried to capture the drone but were unsuccessful.

We could identify the following background functions:

- **Open airspace:** this provides input access to adversarial drones and to passing trains and police officers.
- **Passenger safe arrival to destination** is the output of police, train track zone, and train.
- **Congested road traffic:** impacts the arrival time for police to attend the scene.
- **Environmental conditions:** impact adversarial drone success.

We simulated the model after defining background functions and we got the following results:

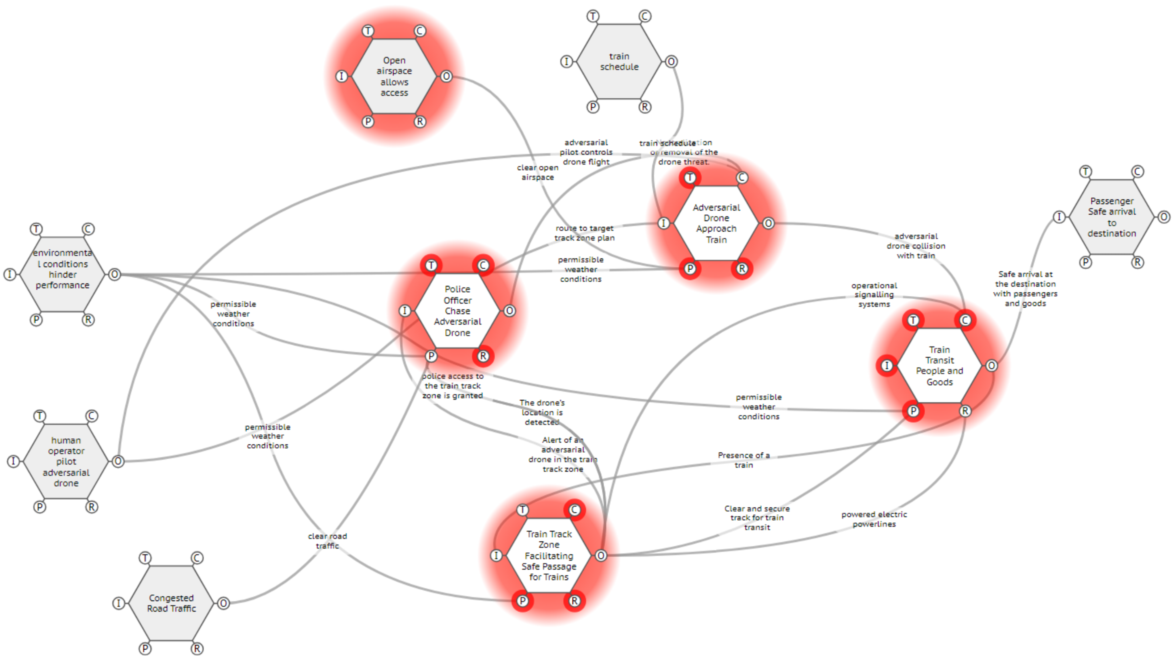

Figure G.5 background functions definition

The initial model of the problem has produced the following errors: Warning: <adversarial drone approach train> has an orphan Warning: <train transit people and goods> has an orphan Warning: <police officer chase adversarial drone> has an orphan Warning: <train track zone facilitating safe passage for trains> has an orphan

In the FRAM model interpreter context, the warning "has an orphan" typically means that the function listed lacks a connection to another function or aspect within the FRAM model. In a FRAM model, each function should have defined inputs, outputs, or dependencies that link it to other functions. When a function is identified as an "orphan," it indicates that:

1. **No downstream dependencies:** the function produces an output, but no downstream functions are using this output.

2. **No upstream inputs:** the function requires inputs, controls, or resources that aren't connected to any other function, suggesting that it might not receive the necessary context or triggering conditions.

Possible implications of orphan functions:

- **Model completeness issue:** the model may be incomplete if the orphan function lacks sufficient connections, as it leaves functional dependencies unaddressed.
- **Impact on functional resonance analysis:** orphan functions can affect the accuracy of the FRAM model's analysis because unconnected functions do not interact with the rest of the system, potentially overlooking critical variability interactions.

To complete the model, we discovered that all aspects (CRIPTO) of foreground functions must be associated with an output of some influencing function. This helped us identify more interactions we didn't consider initially. By making the adjustments we managed to discharge all errors:

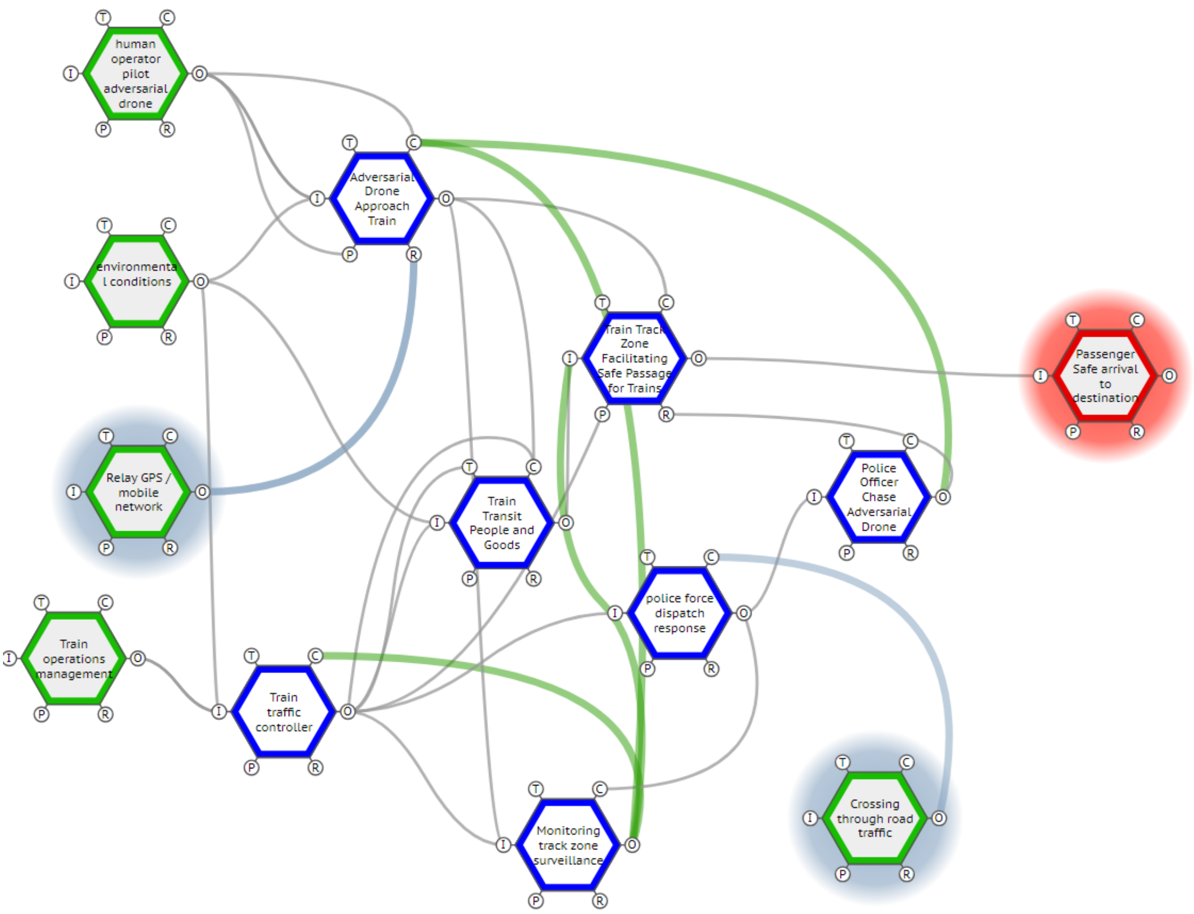

Figure G.6 full problem articulation

The following is the output interpretation of the model which also constitute a FRAM based problem articulation:

FRAM model interpreter - fmi basic ----- (c) erik hollnagel, 2020

Fmi session log: 10/27/2024 11:03:50 pm

Entry function <environmental conditions>

Exit function <passenger safe arrival to destination >

Entry function <human operator pilot adversarial drone>

Entry function <train operations management>

Interpretation profile

Function <train operations management>:

output: 4

Function <human operator pilot adversarial drone>:

output: 4

Function <environmental conditions>:

output: 4

Function <adversarial drone approach train>:

input: all precondition: all resource: all control: none time: all

Function <train transit people and goods>:

input: all precondition: all resource: all control: all time: all

Function <police officer chase adversarial drone>:

input: all precondition: all resource: all control: all time: all

Function <train track zone facilitating safe passage for trains>:

input: any precondition: all resource: all control: all time: all

Function <monitoring track zone surveillance>:

input: all precondition: all resource: all control: any time: all

Function <police force dispatch response>:

input: all precondition: all resource: all control: all time: all

Function <train traffic controller>:

input: all precondition: all resource: all control: none time: all

Function <passenger safe arrival to destination >:

input: all

Summary of fmilog

Begin initialisation

```
---  model initialisation completed.
Begin cycle 0
Background function <relay GPS / mobile network> has been activated
Entry function <environmental conditions> has been activated
Background function <crossing through road traffic> has been activated
Entry function <human operator pilot adversarial drone> has been activated
Entry function <train operations management> has been activated
Begin cycle 1
Function <adversarial drone approach train> has been activated
Function <train traffic controller> has been activated
Begin cycle 2
Function <train transit people and goods> has been activated
Function <police force dispatch response> has been activated
Begin cycle 3
Function <police officer chase adversarial drone> has been activated
Function <monitoring track zone surveillance> has been activated
Begin cycle 4
Entry function <environmental conditions> has been activated
Function <train track zone facilitating safe passage for trains> has been activated
Entry function <human operator pilot adversarial drone> has been activated
Entry function <train operations management> has been activated
Begin cycle 5
Function <adversarial drone approach train> has been activated
Exit function <passenger safe arrival to destination > has been activated
Function <train traffic controller> has been activated
```

Below is the interpretation of the problem articulation output:

1. **core functional interactions and dependencies:**

- **Train operations management**: this function orchestrates the train schedule and manages communications with train operators, fundamentally influencing the entire train traffic system. It outputs critical scheduling and coordination data to the train traffic controller, setting the operational context.

- **Train traffic controller**: As the primary system overseer, this function guides the train's safe transit through scheduling, monitoring, and controlling passage zones. Dependencies on inputs like surveillance alerts, road traffic data, and fencing ensure secure passage, but make it vulnerable if any input fails or is delayed.
- **Monitoring track zone surveillance**: this function actively tracks the train's transit zone for unauthorised access, triggering alerts that guide responses from both the train traffic controller and police dispatch. Surveillance effectiveness is critical for detecting and mitigating intrusions, especially with an adversarial drone threat.

2. **introduction of the adversarial drone:**

- The function of the **adversarial drone approach train** introduces disruptive dynamics, relying on inputs like GPS signals, weather conditions, and control from a human operator (adversarial pilot) with knowledge of train tracks. The drone's capabilities for perception and targeting set the stage for adversarial interference, potentially bypassing or stressing surveillance and response protocols.

3. **Police and response coordination:**

- **Police force dispatch response** is activated upon monitoring track zone surveillance alerts, initiating a chase or defensive action against the drone. This function works alongside **police officer chase adversarial drone**, who captures the drone and secures the track zone. This layer adds resilience by enabling law enforcement to mitigate drone threats physically.
- **Relay GPS / mobile network** and **crossing through road traffic**: both background functions provide essential environmental data to various primary functions. Any GPS failure or unexpected traffic changes could disrupt not only police response times but also the drone's navigational controls, leading to unpredictability's in the operation.

4. **Interpretation of the cycles:**

This breakdown of cycles demonstrates the activation and interaction of key functions within the train transit system, particularly in the presence of an adversarial drone threat. Each cycle reveals how different functions respond and interconnect, contributing to the system's resilience—or points of potential vulnerability.

Cycle 0: initial setup and background conditions

- **Activated functions**: relay GPS / mobile network, environmental conditions, crossing through road traffic, human operator pilot adversarial drone, and train operations management.
- **Interpretation**: this initial cycle establishes essential background and entry functions, setting up the operational environment. Relay GPS / mobile network and crossing through road traffic indicate infrastructure support, while environmental conditions prepare the system to consider weather or external physical factors. Human operator pilot adversarial drone and train operations management entry functions introduce both the drone operator and the train schedule, representing the beginning of potential adversarial influence within the setup.

Cycle 1: drone engagement and train control activation

- **Activated functions**: adversarial drone approach, train and train traffic controller.
- **Interpretation**: The adversarial drone approach train function is activated, showing the adversarial drone entering the system and initiating an approach toward the train track zone. This triggers the train traffic controller, responsible for safely guiding train traffic. The system now must contend with managing regular train passage while accounting for the potential risk posed by the drone.

Cycle 2: transit and response readiness

- **Activated functions**: train transit people and goods and police force dispatch response.
- **Interpretation**: this cycle shows active train movement with train transit people and goods, indicating a scheduled or in-progress transit. Police force dispatch response is activated in response to the drone threat, setting up law enforcement to counter the adversarial presence. This cycle highlights how the system escalates to include law enforcement as soon as the drone activity is confirmed, showcasing a direct reaction to the security risk.

Cycle 3: drone pursuit and surveillance

- **Activated functions**: police officer chases adversarial drone and monitoring track zone surveillance.
- **Interpretation**: Here, a police officer chasing an adversarial drone reflects active measures to pursue or intercept the drone, indicating an attempt to mitigate the drone threat. Monitoring track zone surveillance ensures continuous oversight of the area, particularly important as law enforcement interacts with the drone. Surveillance acts as

a control mechanism, tracking both the train's safe transit and the evolving drone situation.

Cycle 4: reassessment and re-initialization of entry functions

- **Activated functions**: environmental conditions, train track zone facilitating safe passage for trains, human operator pilot adversarial drone, and train operations management.
- **Interpretation**: this cycle involves a refresh of environmental conditions, human operator pilot adversarial drone, and train operations management entry functions, possibly due to shifts in the situation or environmental factors. Train track zone facilitating safe passage for trains is also activated, maintaining safe transit by accounting for both the drone threat and real-time environmental conditions. Re-initializing entry functions indicates an adaptive response within the system, aiming to maintain a sTable operational environment.

Cycle 5: final threat management and system continuation

- **Activated functions**: adversarial drone approach train, passenger safe arrival to destination, and train traffic controller.
- **Interpretation**: the adversarial drone approach train function is triggered again, suggesting an ongoing or reoccurring drone threat. Despite this, the function passenger safe arrival to destination is activated, signifying that the train has successfully managed to pass through the affected zone safely. The train traffic controller remains active, ensuring continued oversight. This final cycle indicates that despite adversarial interference, the system achieves its goal of safe passenger arrival.

G.3.2 Step 2: Characterise variability in outputs

The following is the output list of coupled functions:

1. <train traffic controller>[scheduled train passage time] into {time}<train transit people and goods>

2. <train traffic controller>[guide train driver] into {control}<train transit people and goods>

3. <train traffic controller>[adequate fencing] into {precondition}<train track zone facilitating safe passage for trains>

4. <train traffic controller>[monitor track zone surveillance] into {input}<monitoring track zone surveillance>

5. <train traffic controller>[dispatch train] into {input}<train transit people and goods>

6. <monitoring track zone surveillance>[alert unauthorised access] into {control}<train traffic controller>

7. <human operator pilot adversarial drone>[pilot adversarial drone] into {control}<adversarial drone approach train>

8. <human operator pilot adversarial drone>[capable adversarial perception model] into {precondition}<adversarial drone approach train>

9. <human operator pilot adversarial drone>[train track zone location] into {input}<adversarial drone approach train>

10. <human operator pilot adversarial drone>[adversarial mission target] into {input}<adversarial drone approach train>

11. <environmental conditions>[permissible weather conditions] into {input}<adversarial drone approach train>

12. <environmental conditions>[permissible weather conditions] into {input}<train transit people and goods>

13. <environmental conditions>[permissible weather conditions] into {input}<train traffic controller>

14. <relay GPS / mobile network>[GPS signal] into {resource}<adversarial drone approach train>

15. <relay GPS / mobile network>[provide mobile network] into {resource}<adversarial drone approach train>

16. <adversarial drone approach train>[roam train track zone] into {control}<train track zone facilitating safe passage for trains>

17. <train transit people and goods>[passing through track zone] into {input}<train track zone facilitating safe passage for trains>

18. <monitoring track zone surveillance>[passing through track zone] into {input}<train track zone facilitating safe passage for trains>

19. <train traffic controller>[alert police force] into {input}<police force dispatch response>

20. <police force dispatch response>[dispatch police officers] into {input}<police officer chase adversarial drone>

21. <police officer chase adversarial drone>[capture adversarial drone] into {control}<adversarial drone approach train>
22. <crossing through road traffic>[permissible road traffic] into {control}<police force dispatch response>
23. <train track zone facilitating safe passage for trains>[safe transport of people] into {input}<passenger safe arrival to destination >
24. <police officer chase adversarial drone>[secure train track zone] into {resource}<train track zone facilitating safe passage for trains>
25. <police force dispatch response>[police monitor track zone surveillance] into {control}<monitoring track zone surveillance>
26. <train operations management>[train schedule management] into {input}<train traffic controller>
27. <train operations management>[communication with train operators] into {input}<train traffic controller>
28. <adversarial drone approach train>[miss train collision] into {control}<train transit people and goods>
29. <adversarial drone approach train>[trigger surveillance alert] into {input}<monitoring track zone surveillance>
30. <monitoring track zone surveillance>[detect adversarial drone] into {control}<adversarial drone approach train>

Then we need to define deviations for each coupled function:

Table G.10 FRAM deviation of the problem domain

Coupled functions scenario	Possible variability scenarios			
	Temporal	**Object**	**Direction**	**Distance**
<upstream> output [output] into aspect of <downstream>	Consider internal or external variability to cause the impact of time variability.	Consider internal or external variability to cause the	Consider internal or external variability to cause the impact of	Consider internal or external variability to cause the impact of

		impact of object variability.	direction variability.	distance variability.
<train traffic controller> output [scheduled train passage time] into time factor of <train transit people and goods>	**Too late**: the schedule is delayed, impacting the coordination with other scheduled train passages.	**Wrong train:** incorrect train information is scheduled, impacting other transit schedules.	**Wrong route:** the scheduled time reflects an incorrect route for the train, misaligning with traffic flow.	N/a
<train traffic controller> output [guide train driver] into control factor of <train transit people and goods>	**Too early**: guidance is given too early, confusing the driver about timing. **Too late**: delayed guidance disrupts coordinated train operations.	**Wrong train driver**: the guidance is mistakenly sent to a different train driver, impacting both operations.	**Wrong direction:** instructions are provided for an incorrect track, potentially leading the train off-course.	N/a
<train traffic controller> output [adequate fencing] into precondition factor of <train track zone facilitating safe passage for trains>	n/a	**Inadequate fencing:** incorrect type or insufficient fencing materials reduce safety in the transit zone.	**Fencing in wrong location:** fencing is reinforced in a non-critical area, leaving track zones vulnerable.	**Too short:** adequate fencing only covers part of the zone, failing to protect the entire transit area.
<train traffic controller> output [monitor	**Too late:** surveillance begins after the	**Incorrect zone:** surveillance is directed to a	**Wrong orientation:** surveillance	**Too short:** surveillance time is brief,

track zone surveillance] into input factor of <monitoring track zone surveillance>	train is already in the zone, potentially missing threats.	different zone, leaving the target area unmonitored.	cameras are oriented in the wrong direction, missing key areas.	failing to monitor the train's entire passage.
<train traffic controller> output [dispatch train] into input factor of <train transit people and goods>	**Too early**: train dispatch occurs before passengers have boarded, causing inconvenience and delays. **Too late:** train is dispatched late, delaying all subsequent schedules.	**Wrong train dispatched:** an incorrect train is sent, disrupting the train system schedule.	**Wrong direction:** the train is dispatched onto an incorrect track, leading to misdirection.	**Too long:** dispatch includes extended segments beyond its intended destination, causing inefficiency.
<monitoring track zone surveillance> output [alert unauthorised access] into control factor of <train traffic controller>	**Too early**: alert is triggered before unauthorized access occurs, leading to unnecessary resource allocation. Too late: alert comes after unauthorized access has already affected train operations.	**False alert:** the system misidentifies authorised personnel as unauthorised, causing operational delays.	**Incorrect alert zone**: alert is sent for an unrelated track zone, missing the target area of unauthorized access.	**Alert range too short**: alert fails to cover the entire track zone, missing parts of unauthorized access areas.
<human operator pilot adversarial	**Too early**: the adversarial drone is piloted toward	**Incorrect drone:** a different, possibly more	**Wrong path:** the drone is piloted along an	**Too close:** the drone is piloted dangerously

drone> output [pilot adversarial drone] into control factor of <adversarial drone approach train>	the train track zone too soon, allowing more time for strategic positioning and risk escalation. **Too late:** the drone is piloted too late, missing the intended train but potentially disrupting the following train instead.	capable adversarial drone is piloted, increasing risks to the train track zone.	unexpected path, making detection and interception by surveillance systems more difficult.	close to the track zone, increasing the likelihood of collision or interference with train sensors.
<human operator pilot adversarial drone> output [capable adversarial perception model] into precondition factor of <adversarial drone approach train>	**Too early**: perception model activates prematurely, gathering information on potential targets in advance, which could allow for enhanced tactics. **Too late:** perception model initializes after passing the track zone, leading to suboptimal targeting but still potentially distracting security.	**Incorrect model:** a more sophisticated perception model is deployed, enabling higher accuracy in detecting and tracking the train.	**Wrong focus:** the model focuses on unrelated zones, creating an unpredicTable threat profile and complicating security responses.	**Too broad:** the perception model captures data from an excessively wide area, increasing its potential to detect unprotected vulnerabilities within the rail network.
<human operator pilot	**Too early**: location data is	**Wrong zone:** the adversarial	**Wrong orientation:**	**Too far:** location data is

adversarial drone> output [train track zone location] into input factor of <adversarial drone approach train>	obtained prematurely, enabling adversarial drone operators to study the train track zone and potentially identify weak points. Too late: location information arrives too late, but the drone is still able to monitor other key areas nearby, increasing overall surveillance pressure.	drone is directed to an incorrect but nearby zone, where it can observe and adapt to regional transit behaviours.	drone surveillance focuses on the track zone from an unusual angle, complicating detection efforts from ground-based security.	gathered beyond the immediate area, but still within reach, potentially threatening additional track zones not covered by surveillance.
<human operator pilot adversarial drone> output [adversarial mission target] into input factor of <adversarial drone approach train>	**Too early**: mission targeting initiates before the train arrives, allowing the drone to establish an advanced position for disruptive actions. Too late: targeting occurs after the train passes, but it can still pursue or monitor subsequent	Wrong target: a different train or infrastructure target is selected, impacting a less protected segment of the rail network.	Wrong approach path: the drone's targeting path deviates from standard routes, making interception more challenging for defenders.	Too long: targeting area is extended along the track zone, allowing for prolonged observation or interaction with the train and its surroundings.

	trains, creating a persistent threat.			
<environmental conditions> output [permissible weather conditions] into input factor of <adversarial drone approach train>	**Too early**: weather permits drone operation earlier than expected, allowing the drone to advance into the train zone before adequate countermeasures are in place.	N/a	N/a	N/a
<environmental conditions> output [permissible weather conditions] into input factor of <train transit people and goods>	**Too late:** updated weather conditions are reported after the train has already started transit, increasing the chance of adverse effects en route.	**Incorrect forecast:** incorrectly forecasts poor conditions, leading to unnecessary delays, while the real threat (adversarial drone) remains undetected.	**Wrong focus area:** weather reports focus on areas adjacent to the train’s path, missing critical conditions in the train's route.	N/a
<environmental conditions> output [permissible weather conditions] into input factor of <train traffic controller>	N/a	**Wrong weather conditions:** misreported conditions prompt unnecessary delays or actions, leaving potential threats unmonitored.	**Wrong weather zone:** the controller receives weather information for the wrong zone, misaligning control actions for the actual	N/a

			conditions faced by the train.	
<relay GPS / mobile network> output [GPS signal] into resource factor of <adversarial drone approach train>	**Too late:** delayed GPS signal disrupts the drone's timing, potentially causing it to miss or inaccurately target the train, risking broader security concerns.	**Incorrect signal source:** signal data from an incorrect source or spoofed signal leads the drone to the wrong location but still within reach of the train route.	**Wrong GPS path:** the drone follows an incorrect GPS path, leading it through unanticipated zones and complicating efforts to track or intercept it.	N/a
<relay GPS / mobile network> output [provide mobile network] into resource factor of <adversarial drone approach train>	N/a	N/a	N/a	N/a
<adversarial drone approach train> output [roam train track zone] into control factor of <train track zone facilitating safe passage for trains>	**Too early**: drone begins roaming before a train is present, enabling it to scout for weak points and increasing collision potential. **Too late:** drone enters the track zone after the train passes, causing	N/a	N/a	**Too wide:** drone roams an extended area beyond the immediate track zone, escalating risks to adjacent train operations.

	possible interference with other train schedules or future trains.			
<train transit people and goods> output [passing through track zone] into input factor of <train track zone facilitating safe passage for trains>	**Too late:** train arrival delays complicate coordination and increase the chance of collision with an adversarial drone lingering in the zone.	**Incorrect train:** another unscheduled train occupies the track zone, confusing surveillance and safety systems.	N/a	**Too short:** train doesn't fully clear the track zone, blocking subsequent trains' paths and causing logistical delays.
<train traffic controller> output [alert police force] into input factor of <police force dispatch response>	**Too early**: police are alerted before any threat fully materialises, risking resource misallocation and response fatigue. **Too late:** the alert comes after an incident has escalated, delaying police response and increasing risk to train and track safety.	**Incorrect alert target:** police are dispatched to the wrong area, leaving the true threat zone unprotected.	**Misleading alert:** the alert's focus deviates from the actual threat zone, creating confusion and response delays.	N/a
<police force dispatch response> output [dispatch	N/a	**Wrong unit:** a less equipped police unit is dispatched,	**Wrong direction:** officers approach from	**Too far:** officers are dispatched from too far station from the

police officers] into input factor of <police officer chase adversarial drone>		limiting the response capacity against the adversarial drone.	an unexpected route, reducing their efficiency in intercepting the drone's path.	track zone area, wasting resources and response time.
<police officer chase adversarial drone> output [capture adversarial drone] into control factor of <adversarial drone approach train>	**Too late:** capture attempt happens after the drone completes its mission, failing to prevent adversarial actions.	N/a	N/a	**Too far:** officers pursue the drone too far from the track zone, losing coverage of other vulnerable areas.
<crossing through road traffic> output [permissible road traffic] into control factor of <police force dispatch response>	**Too late:** traffic control updates are delayed, causing police to be stuck in traffic when urgent response is needed.	N/a	N/a.	N/a
<train track zone facilitating safe passage for trains> output [safe transport of people] into input factor of <passenger safe	N/a	N/a	N/a	N/a

arrival to destination >				
<police officer chase adversarial drone> output [secure train track zone] into resource factor of <train track zone facilitating safe passage for trains>	N/a	N/a	N/a	N/a
<police force dispatch response> output [police monitor track zone surveillance] into control factor of <monitoring track zone surveillance>	**Too late:** monitoring starts after an incident has already escalated, reducing the effectiveness of the response.	**Incorrect surveillance focus:** monitoring is directed toward a less vulnerable zone, leaving critical areas exposed.	**Wrong orientation:** surveillance is focused in the wrong direction, allowing threats to go undetected or poorly addressed.	**Too short:** surveillance monitoring covers only a portion of the track zone, potentially leaving other areas unmonitored.
<train operations management> output [train schedule management] into input factor of <train traffic controller>	**Too late:** schedule updates are delayed, leading to unsynchronised train dispatches, which may cause delays or missed connections.	**Incorrect schedule:** an outdated or incorrect schedule is provided, causing cascading delays and mismanagement	**Wrong timing focus:** scheduling fails to account for rush hours or peak times, leading to congestion and operational inefficiencies.	N/a

		across the transit system.		
<train operations management> output [communication with train operators] into input factor of <train traffic controller>	**Too late:** communication delays cause misalignment in coordination, with potential safety risks if operators aren't fully informed.	**Incorrect communication content:** misleading or incorrect information is conveyed to the train operators, risking operational or safety-related errors.	**Wrong communication direction:** information is relayed to the wrong team or department, causing unnecessary action or inaction in key areas.	**Too limited:** communication fails to reach all necessary operators, resulting in fragmented understanding or inconsistent operational execution.
<adversarial drone approach train> output [miss train collision] into control factor of <train transit people and goods>	N/a	**Wrong train target:** the drone's target is a different, unintended train, potentially impacting a non-critical route while still posing safety risks.	N/a	**Too far:** the drone flies beyond the immediate track zone, potentially endangering other critical infrastructure along the route.
<adversarial drone approach train> output [trigger surveillance alert] into input factor of <monitoring track zone surveillance>	**Too late:** the alert is triggered after the drone has entered the track zone, reducing response time and increasing safety risks.	**Wrong target alert:** the alert is mistakenly triggered for a different environmental object, causing distraction and possible resource misallocation.	**Wrong orientation:** the alert directs security focus to the wrong area, such as an unrelated section of the track zone, delaying interception.	**Too short:** the alert coverage is limited, leaving critical areas of the track zone unmonitored, increasing the risk of undetected unauthorised access.

<monitoring track zone surveillance> output [detect adversarial drone] into control factor of <adversarial drone approach train>	N/a	N/a	N/a	N/a

G.3.3 Step 3: identify functional resonance

Step 3: identify functional resonance across non-direct coupled functions

Table G.11 definition of functional resonance

Resonating functions variabilities	Emergent resonance scenario
Negative variability of <train traffic controller>[scheduled train passage time] into {time}<train transit people and goods> & <train traffic controller>[dispatch train] into {input}<train transit people and goods> **<u>Resonating with:</u>** **positive variability of** <adversarial drone approach train>[miss train collision] into {control}<train transit people and goods> **Resulting in:** Increased the likelihood of an adversarial drone missing a train collision.	**Given that:** **Too late:** the schedule is delayed, impacting the coordination with other scheduled train passages. **Wrong train:** incorrect train information is scheduled, impacting other transit schedules. **Wrong route:** the scheduled time reflects an incorrect route for the train, misaligning with traffic flow. **Then:** Leading to a change in train passage through the train track zone, thus reducing the impact on the adversarial pilot's expected timing.

Negative variability of <environmental conditions> output [permissible weather conditions] into input factor of <train transit people and goods> **<u>Resonating with:</u>** **Negative variability of** <adversarial drone approach train>[miss train collision] into {control}<train transit people and goods> **Resulting in:** Increased the likelihood of adversarial drone train collision.	**Given that:** **Incorrect forecast**: suppose the forecast anticipated adverse conditions; however, good conditions emerged earlier than expected. **Then:** This shift may influence the likelihood of an adversarial attack, as the sudden change could suggest an increased probability of such an event occurring.
Negative variability of <environmental conditions> output [permissible weather conditions] into input factor of <train transit people and goods> **<u>Resonating with:</u>** **Negative variability of** <crossing through road traffic> output [permissible road traffic] into control factor of <police force dispatch response> **<u>Resonating with:</u>** **Negative variability of** <adversarial drone approach train>[miss train collision] into {control}<train transit people and goods> **Resulting in:** Increased the likelihood of adversarial drone train collision.	**Given that:** **Incorrect forecast**: suppose the forecast anticipated good conditions, And police ignore road traffic condition since there is no bad weather to cause bad traffic (assuming police tacitly associate good weather with easy traffic), forecast, But the weather changes dramatically, leading to unexpected traffic congestion, **Too late:** traffic control updates are delayed, causing police to be dispatched when urgent response is needed during bad traffic. **Then:** This will lead to an increase in the likelihood of adversarial drone train collisions.

Negative variability of <train traffic controller> output [monitor track zone surveillance] into input factor of <monitoring track zone surveillance> **<u>Resonating with:</u>** **Negative variability of** <train traffic controller> output [adequate fencing] into precondition factor of <train track zone facilitating safe passage for trains> **<u>Resonating with:</u>** **Negative variability of** <adversarial drone approach train> output [trigger surveillance alert] into input factor of <monitoring track zone surveillance>	**Given that:** **Wrong orientation:** surveillance cameras are oriented in the wrong direction, missing key areas. **Inadequate fencing:** incorrect type or insufficient fencing materials reduce safety in the transit zone. **Then:** The fence's inadequacy could impact the surveillance system, which is untrained to consider the fence shape. An adversarial drone approach may lead to a false positive or negatives.

G.4 Gap analysis of FRAM using AIC

One helpful aspect of AIC, as it is a predictive thinking framework, is that it enables us to evaluate the comprehensiveness of other thinking tools and investigate the useful semantic rules they could employ to enhance the expressiveness of their models as predictive thinking tools. AIC is a systems thinking approach that aims to balance computational and lateral thinking; hence, it can potentially be used to evaluate the general coverage of thought processes like FRAM. In this section, we also use it as an opportunity to assess how AIC can enhance the solution capability of FRAM, which we consider as part of our future research direction (see Section 9.16.2 related to the integration of AIC and FRAM).

Figure G.7 demonstrates how AIC can be semantically integrated into the FRAM ontology. In the context of AIC, Figure G.7 can be interpreted as the following:

There exists some complex A influencing complex C , in order for complex A to influence C, it must control complex B, complex B controls complex C , in order for complex correctly and reliably control B, it must complex situation D. complex A cannot influence of control complex D, complex can influence complex A ability to control B to influence C. Complex C cannot control complex A, but may be influence A through B.

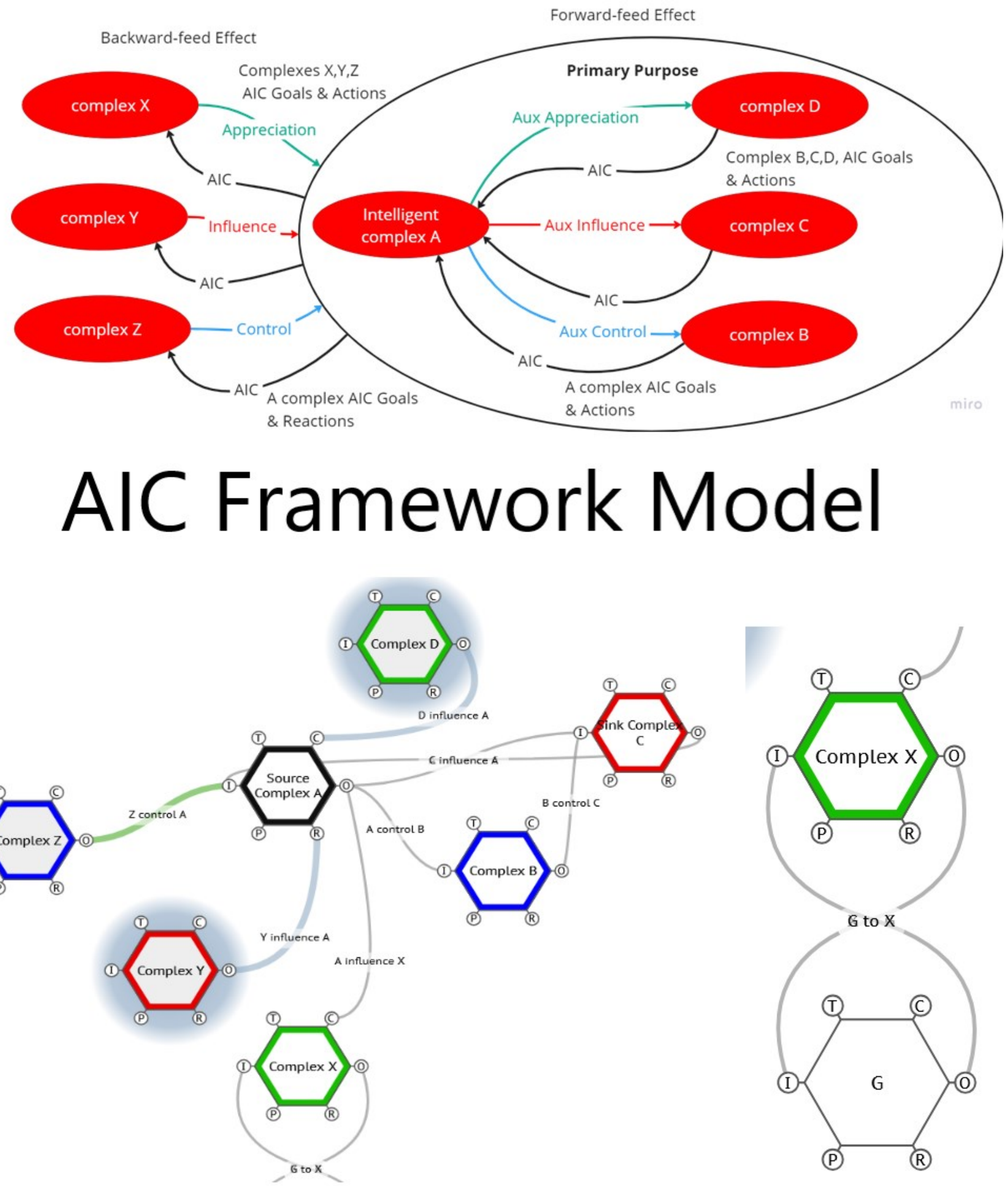

Figure G.7 modelling AIC framework using FRAM ontology. Green entity: appreciation, Red: influence, and Blue: control.

Simultaneously, there exists a complex X that appreciates A, and A influence X. Also, complex Z that controls A and Y that influence A behaviour.

When we run a model simulation cycle to visualise how AIC behaviour would emerge, we get the following behaviour:

FMI session log: 04/06/2025 15:16:18

Exit function <Sink Complex C>

Exit function <Complex X>

Entry function <Complex Z>

<u>Interpretation Profile</u>

Function <Complex Z>:

Output: 0

F**unction** <Source Complex A>:

Input: All   Precondition: All   Resource: All   Control: All   Time: All

Function <Complex B>:

Input: All   Precondition: All   Resource: All   Control: All   Time: All

Function <Sink Complex C>:

Input: All

Function <Complex X>:

Input: All

Summary of FMIlog

Begin initialisation

--- MODEL INITIALISATION COMPLETED.

BEGIN CYCLE 0

Background function <Complex D> has been activated

Entry function <Complex Z> has been activated

Background function <Complex Y> has been activated

BEGIN CYCLE 1

Function <Source Complex A> has been activated

BEGIN CYCLE 2

Function <Complex B> has been activated

Exit function <Complex X> has been activated

BEGIN CYCLE 3

Exit function <Sink Complex C> has been activated

The interpretation profile is set so that each function's aspects are enabled (e.g. "Input: All" means the function will activate when all its inputs are present). When you run a FRAM Model Interpreter (FMI) simulation, the tool advances in discrete "cycles". In each cycle, it looks for any function whose prerequisites (Inputs, Preconditions, Resources, Controls, Time) are now satisfied, and then "activates" that function, meaning it fires and produces its Output. The cycles proceed in order (0, 1, 2, ...) until every function that can run has run.

- **Cycle 0** is special: it automatically activates any **background** functions (those with no upstream dependencies) and any **entry** functions (designated "entry" by the interpretation profile).
- In **Cycle 1** and onward, the interpreter checks which remaining functions now have all their inputs or controls available (from outputs produced in previous cycles) and activates them.
- Functions labelled **"Exit"** are typically the sinks of the model; they only activate once their single (or final) Input has arrived.

Each "has been activated" line in the log simply means: "At this cycle, all the inputs/preconditions/resources/controls needed by that function were present, so the function ran and produced its output". The main properties of FRAM cyclic simulations:

- **Cycle 0** always runs any background or entry functions (they need no upstream inputs).
- In **Cycle n (n ≥ 1)**, any function whose inputs, preconditions, resources, controls, and time constraints are now fully met will run.
- If two or more functions become ready in the same cycle, they activate in parallel in that cycle.
- **Entry** = starts the flow (no inputs needed).
- **Background** = provides context or data (no inputs needed).
- **Exit** = final steps or sinks (only run once their one required input arrives).

Interpretation:

Cycle 0: Background + Entry functions

"Background function <Complex D> has been activated"

"Entry function <Complex Z> has been activated"

"Background function <Complex Y> has been activated"

- **Background functions** (D and Y) have no upstream dependencies in the model—they don't need any other function's output to run. By definition, they always fire in Cycle 0.
- **Entry function <Complex Z>** was explicitly designated in the interpretation profile with "Output: 0". That means its output is considered available right away (no inputs needed), so the simulator fires Z immediately in Cycle 0 as well.

At the end of Cycle 0, the outputs of D, Y, and Z now exist in the system. Those outputs will serve as the “incoming conditions” that let other functions run in later cycles.

Cycle 1: Source function runs

“Function <Source Complex A> has been activated”

- In Cycle 1, the simulator checks every function that hasn’t yet run. It finds that **Source Complex A** now has all its prerequisites:
    - It needs “Input = All,” “Precondition = All,” “Resource = All,” “Control = All,” “Time = All”.
    - Because Z’s Output and the background functions’ outputs are already present (from Cycle 0), all of A’s required inputs/controls/resources are satisfied.
- Therefore, the interpreter “activates” A in Cycle 1. A’s Output is produced at the end of this cycle.

Cycle 2: Mid-chain functions and intermediate exit

“Function <Complex B> has been activated”

“Exit function <Complex X> has been activated”

- Now that A has produced its output (in Cycle 1) and the background outputs still exist, **Complex B** sees that all its required aspects are available (Input/Precondition/Resource/Control/Time = All).
    - Specifically, B’s “Input” slot is likely satisfied by A’s Output, and any other controls/resources come from Z, D, or Y.
    - So in Cycle 2, B fires and produces B’s Output.
- Also in Cycle 2, the **Exit function <Complex X>** activates.
    - An “Exit” function has no output of its own—rather, it is a designated sink that only runs once its Input is present.
    - By the time B fired, whatever B or the background functions feed into X must now be available. As soon as X sees its Input, it is eligible to run. Hence in the same cycle (Cycle 2) that B ran, X can also run if X’s single Input comes from B’s Output (or from some background).

At the end of Cycle 2, B’s Output and X’s Output (if X has any side-effect) are both in the model.

Cycle 3: Final sink completes

“Exit function <Sink Complex C> has been activated”

- In Cycle 3, the simulator looks at the remaining functions.

- **Sink Complex C** is the last function left. It only has “Input = All” listed, so it wakes up when it sees its needed Input.
    - Earlier functions must have produced the only inputs that C requires, most likely B or X (depending on how the model was coupled).
    - Since B and X both ran in Cycle 2, C now has its required Input at the start of Cycle 3.
- Therefore, in Cycle 3, C runs and finishes the mission chain. Because it’s an “Exit” function, running C means the scenario has reached its final “sink”.

Putting it all together

Note that the model did not capture the feedback loop between X and G.

1. **Initialisation** sets up the FRAM model, identifies background and entry functions, and clears any “already-run” flags.
2. **Cycle 0**:
    - Background functions (D, Y) run immediately because they need nothing else.
    - Entry function (Z) runs immediately because its output was set to “0” (meaning “produce output at time 0”).
3. **Cycle 1**:
    - Now that the outputs from Z, D, and Y exist, the simulator checks **Source Complex A**. All of A’s inputs/controls/resources are present, so A runs.
4. **Cycle 2**:
    - With A’s output in place, **Complex B** fires next (it gets its input from A and other controls/resources from Z/D/Y).
    - Also in Cycle 2, **Exit Complex X** sees that its single input is now ready (most likely from B or a background), so it runs concurrently with B.
5. **Cycle 3**:
    - Finally, **Sink Complex C** finds that its required input—fed by B or X—is available, so it runs and completes the simulation chain.

At each cycle, the interpreter simply “wakes up” any function whose upstream dependencies were satisfied in earlier cycles. The labels **Entry** (for Z) and **Exit** (for X and C) help you see which functions kick off the process (entry) and which are the designated endpoints (exit).

Some implications can lead to richer meanings and thus more room to capture complicated interactions and predict Black Swan scenarios:

- **Appreciation and Influence constraints:** AIC provide an imposed constraint on control interactions. If two complexes are in influence, then there is no direct control between them.

For example, "A influence C" interaction means the control variable is disabled between A and C. As for appreciation, there is no clear aspect that can be used to link A to D in an appreciative function. Stricter rules are needed to ensure that the control variable on B is disabled, so we avoid the situation where the Architect makes a semantic error.

The semantic meaning of AIC indicates that the designer declares the constraints are predicted to be preserved. So, when we say A influences C, it means neither A nor C shall ever be in control of the other. This means that a scenario where one complex controls the other would constitute a Black Swan scenario. Because the basic assumption made is "never", so we expect control to be a very rare event as part of the design. Currently, it is not clear how FRAM capture such restrictions in behaviour.

- **No output appreciation (appreciative interaction):** The FRAM model does not allow for outputting an appreciative action between A and D (the appreciated environmental complex), the same goes between X to A.

The definition of a Background function could potentially refer to an appreciated entity; however, it is restricted to being a function. This means it is not easy to model non-functional situations, such as wind, rain, or the visual appearance of an adversarial drone. When we modelled the interaction between A and X, where X appreciates A but does not influence or control A, we ended up with an error in the model because it requires an input from X. So needed a background function that outputs a control or influence impact upon X. We required to have an output somewhere coming out from X, but we can not direct it to A as it is not allowed. Therefore, we needed to add a redundant entity, G, to prevent the error. However, now we have G with an output and without input, so we needed to impose a Feedback loop between X and G.

- **FRAM could not capture the nuance of unintended impact:** there is a distinct difference between A influence X and A influence C. In that C unintentionally influences X, while it is intended to influence C.
- **Delay of influence in cycles:** If we adopt a non-harmonic order of AIC, then control proceeds influence, which means that when we run the cycle, the model will activate appreciation and control interactions first, followed by influence. when we ran the model, complex in cycle 2, the interaction "A influence C" was activated before "B control C". Influence requires "A control B" and "B control C" interactions to be activated before "A influence C". FRAM cyles shows that:
    - Upon activation of A, B and C are controlled.
    - Then, when B controls C.

We have no means to control the sequence of activation, which means that the designer may not be able to model constrained activation delay without including two controlled functions that delay the impact of A upon C to simulate influence.

- **The simulation breaks when feedback loops are introduced:** Feedback loops are important in AIC model to capture complicatedness of systems.

When we modelled a feedback loop between G and X and then ran the simulation, we realised that the model's cyclic simulation stopped at the loop part of the model. We then tested the situation again by modelling a feedback loop between A and C, the same thing occurred. FRAM handbook mentions that hat it is **not** a flow or network model, so you do **not** draw loops as special constructs. Instead, you represent feedback by listing each function's aspects and coupling outputs back to inputs. Any circular dependency you create is automatically treated as a loop, with no special notation beyond the standard coupling arrows. This means that the architect may miss feedback loops during modelling, thus overlooking useful requirements or solutions.

- **Risks associated with Background functions (BG):** FRAM defines a useful tool, which is BG. It defines it as "A background (BG) function is similarly a function which is assumed not to vary or to be stable during the duration of the process or activity being analysed".

While making assumptions is useful for reducing the complexity of tasks, in compounded uncertainty problems, we need to take extra care to consider the risks we are exposing ourselves to when making them. The main risks of missing potential critical requirements are:

    - BG functions are treated in isolation. Unexpected interactions between BG functions.
    - The architect may forget BG functions when models scale in size.
    - BG functions are assumed to be predictable alone but may exhibit unpredictable behaviour when combined.

G.4.1 Potential missed requirements when using FRAM

Below is a categorised list of general interaction types that FRAM users may overlook in analysing complicated problems, along with a brief note on how thinking in terms of AIC (Appreciation, Influence, Control) can help catch them:

1. **Missing Appreciative Interactions**: Designers may omit functions or links that merely "observe" or "sense" another entity without actively controlling it. In FRAM, it is unclear how to model a non-functional entity that impacts another entity, while the latter merely appreciates the impact of the former.

For example, function G, we needed Background functions, but the model's restrictions make it easy to forget them altogether.

- **Why it matters:** Without an explicit "appreciation" link, you fail to record how one component simply perceives or monitors another (e.g., a drone's computer "appreciates" wind data, but doesn't directly control the wind).
- **How AIC can help FRAM:**
  - **Appreciation** reminds you to ask: "Which components simply need to know something (even if they can't change it)?"
  - Forcing yourself to declare an appreciative interaction (e.g., "Sensor B → D appreciates environmental data") prevents accidental omission of purely observational influences.

2. **Overlooked Influence Constraints (Indirect Dependencies)**
   - **What may be missed:** In FRAM, "A influences C" means neither A nor C can directly control one another, yet many designers may assume that any two linked functions can also "control" each other.
   - **Why it matters:** If you never explicitly prohibit "A controls C" the model may allow an unrealistic coupling where A bypasses B. Omitting that constraint hides the fact that A's effect on C must always pass-through B.
   - **How AIC helps:**
     - **Influence** forces you to distinguish between a "soft" impact (A → C) versus a "hard" control (A ⟶ C).
     - By declaring "A influences C" (and therefore disables any direct control link), you prevent semantic mistakes, e.g., assuming A can directly manage C's behaviour.
     - You think ahead: "If I ever see A directly toggling C, that's a Black Swan (because it violates my declared influence constraint)".
3. **Missing Unintended Interactions (Side-Effects)**
   - **What may be missed:** using FRAM, it's easy to model only the **intended** chain (A → B → C) and forget that A may inadvertently influence X, or that A's action might trigger side effects nobody planned for.

FRAM relies on the concept of functional resonance for practitioners to discover further impacts that may have been missed during deviation analysis. However, it relies on the architect to think of that sequence of events to realise it. It may be more efficient if practitioners are explicitly prompted to think about unintended impacts while modelling.

- **Why it matters:** Some consequences arise not from a direct control or even planned influence, but from unanticipated "ripple" effects. If you never look for them, you won't spot potential failure modes or emergent behaviours.
- **How AIC helps:**

        - **Influence** again nudges you to ask: "Which functions might be loosely affected by my action, even if I didn't intend to affect them?"
        - By labelling "C influences X (unintended)," you document that side-effect, instead of sweeping it under the rug.
        - AIC reminds you to write down both "intended" control/influence links and "unintended" actions, using a neutral sign to indicate actions that may only surface under certain conditions.

4. **Practitioners may miss hidden Feedback Loops, which are sources of unpredictability**
    - **What's my be missed:** FRAM's guideline ("no special loop notation") means you simply draw couplings; designers sometimes forget to trace a function's Output back into the system as that function's own Input (or another upstream Input).
    - **Why it matters:** A feedback loop can dramatically change system behaviour (oscillations, homeostasis). If you never include the circular coupling (e.g., X → G → X), the model will silently break or never show the loop's impact.
    - **How AIC helps:**
        - **AIC** distinctions prompt you to track: "Does any function's result eventually come back and control or influence itself (or something upstream)?"
        - By forcing yourself to document "X influences G" and "G controls X," for example, you make that loop explicit, ensuring that the tool can't silently discard or block it.
        - Even if FRAM doesn't have a "feedback" graphic, AIC thinking makes you systematically check for any chain of dependencies that circles back.
5. **Temporal Delays and Activation Order Issues**
    - **What may be missed:** FRAM cycles activate functions as soon as their inputs are present, but if you need a guaranteed delay (e.g., "A must wait two cycles before B"), you may forget to insert a placeholder.
    - **Why it matters:** If "A influences C" should only take effect after "A controls B" and "B controls C" have both fired, but the interpreter steps them in the wrong order, you end up with an unrealistic "influence" that appears too soon.
    - **How AIC helps:**
        - **Influence** encourages you to check: "Does an influence link really depend on prior control chains?" If so, you enforce a temporal ordering.
        - **Control** reminds you that control interactions typically happen in a strict sequence. Suppose you need to hold off on an influence until after you have control. In that case, AIC thinking pushes you to add two chained

control functions (or explicit ‘delay’ functions) so that the interpreter honours the intended order.

6. **Non-Functional Environmental Factors (Invisible Inputs)**

    - **What’s often missed:** “Wind direction”, “visual appearance of an adversarial drone”, or “cultural norms” are not ‘active steps’, they’re context. Designers may fail to model them because FRAM expects “functions” with six aspects.
    - **Why it matters:** Real-world variability often comes from these non-functional factors. If you ignore them, you’ll never capture how, for instance, a sudden change in clouds leads to unexpected sunlight reflection on object surfaces, which in turn impacts a perception model.
    - **How AIC helps:**

        - **Appreciation** guides you to ask: “Which components need to be aware of these context factors?”

7. **Neglecting Interactions Among Background Functions**

    - **What’s often missed:** FRAM prompt the designer when specifying Background functions (BG) to assume they are stable and no need to specify CRIPTO.

In a sense, BG can be understood as a form of defining the system boundary. While they are helpful for constraining model size and helping to determine when to stop, there is a tacit risk associated with them. That is, they are assumed to be predictable and treated in isolation from one another. You might miss that one context factor actually shapes another (e.g., heavy rain degrades, and sun illumination may degrade the way objects appear in ways that were not accounted for during the training of ML models).

- **Why it matters:** If “Rain” and “Sunlight” both act as separate inputs to D (environmental data) but you never connect, Sun → influences Rain → (influences) visual quality, then your training process may underestimate how the combo of those factors can indirectly degrade ML detection.
- **How AIC helps:**
    - **Appreciation** forces you to ask: “Which background entity appreciates another background entity?”
    - **Influence** nudges you to capture that “sunlight influences how rain influences Visual Sensor Quality”, even though neither is a “primary function”.

    - Once you draw that coupling, you see that B (navigation or ML) must account not only for raw data from D, but also for second-order variability (sunlight making the impact of rain more unpredictable).

When designers rely solely on a basic FRAM diagram, they may focus on the **straightforward control chains** (e.g., A → B → C) and may forget more subtle interactions. The AIC mindset helps reveal:

- Which entities only observe but don't control (Appreciation),
- Which entities merely exert a soft dependency but not direct control (Influence),
- Which functions have true gating or command power (Control).

By systematically asking, "Are there any influences, where no control is assumed, or appreciations I haven't drawn?" and "Am I missing a control constraint?" the architect catches nuanced dependencies, like feedback loops, unintended side-effects, shared resources, environmental factors, and rare Black Swan scenarios, before they derail the design.