Dealing with uncertainties in drone-based missions

State University of Ceará and The Open University

Lucas Vieira Alves
Supervisor: Paulo Henrique Maia

GESAD
Adaptive and Distributed Software Engineering Group

The Open University

UnB

UAVs

Unmanned aerial vehicles (UAVs, also known as drones) have been widely used in the **military field**.

UAVs: From Military to Civil Applications

Due to technological **advances** drones have been used in **civil applications**.

Delivery

Delivery

Farming

Delivery

Farming

Inspection

Delivery

Farming

Inspection

Search and rescue

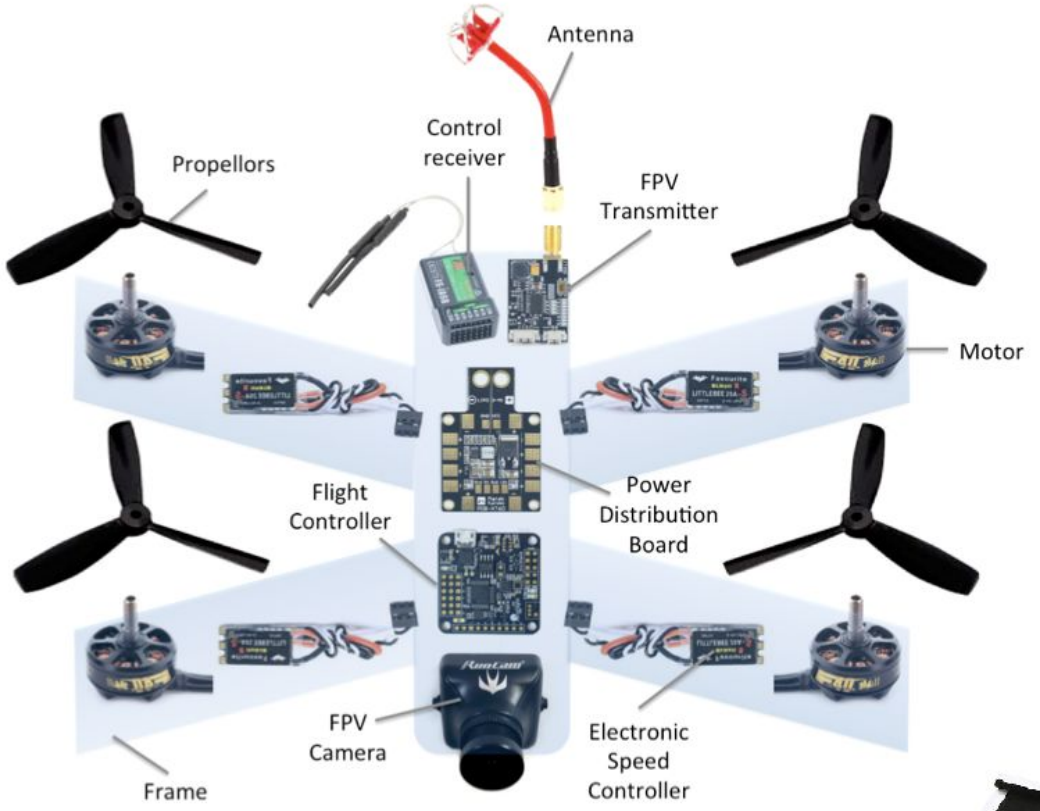
Software Evolution

Drone-based application may require constant **evolution**.

Software Evolution

Drone-based application may require constant **evolution**.

- Complex systems;



Software Evolution

Drone-based application may require constant **evolution**.

- Complex systems;
- Critical Systems;

Software Evolution

Drone-based application may require constant **evolution**.

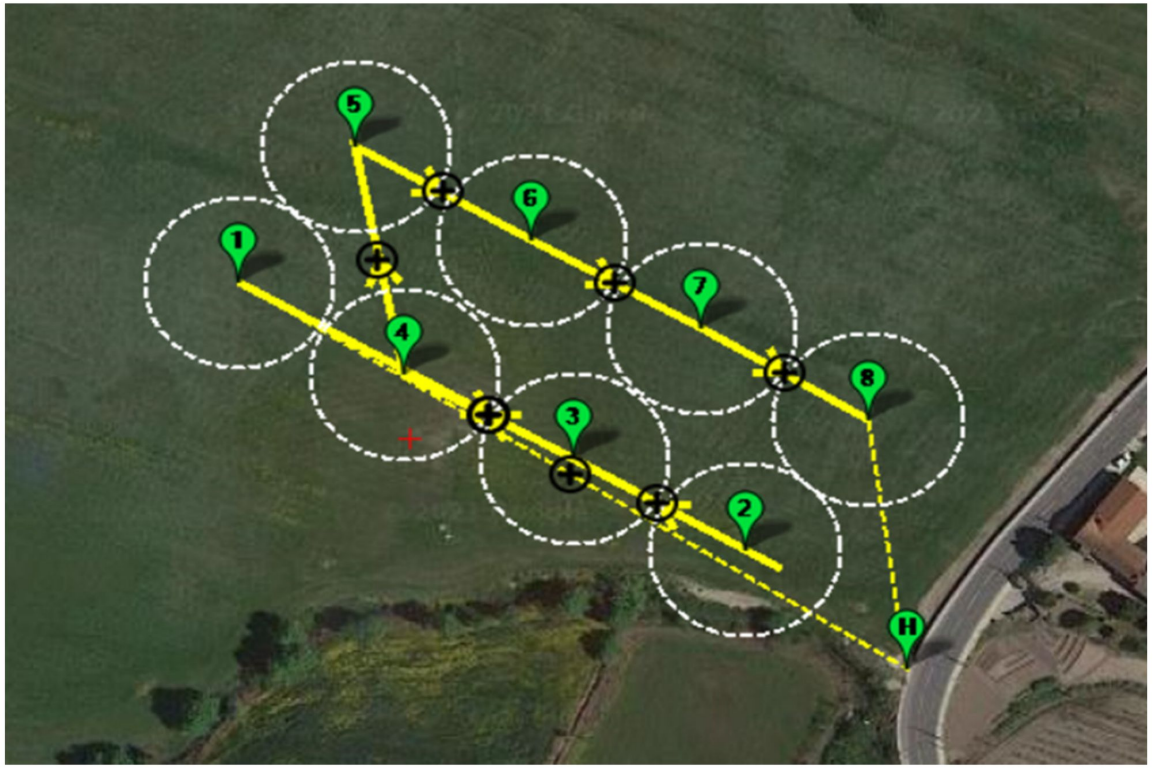
- Complex systems;
- Critical Systems;
- Dynamic and uncertain environment;

Self-adaptive system (SAS)

Increase the degree of autonomy with minimal interaction through **self-adaptation** techniques.

Self-adaptive drone

- Autonomous navigation;

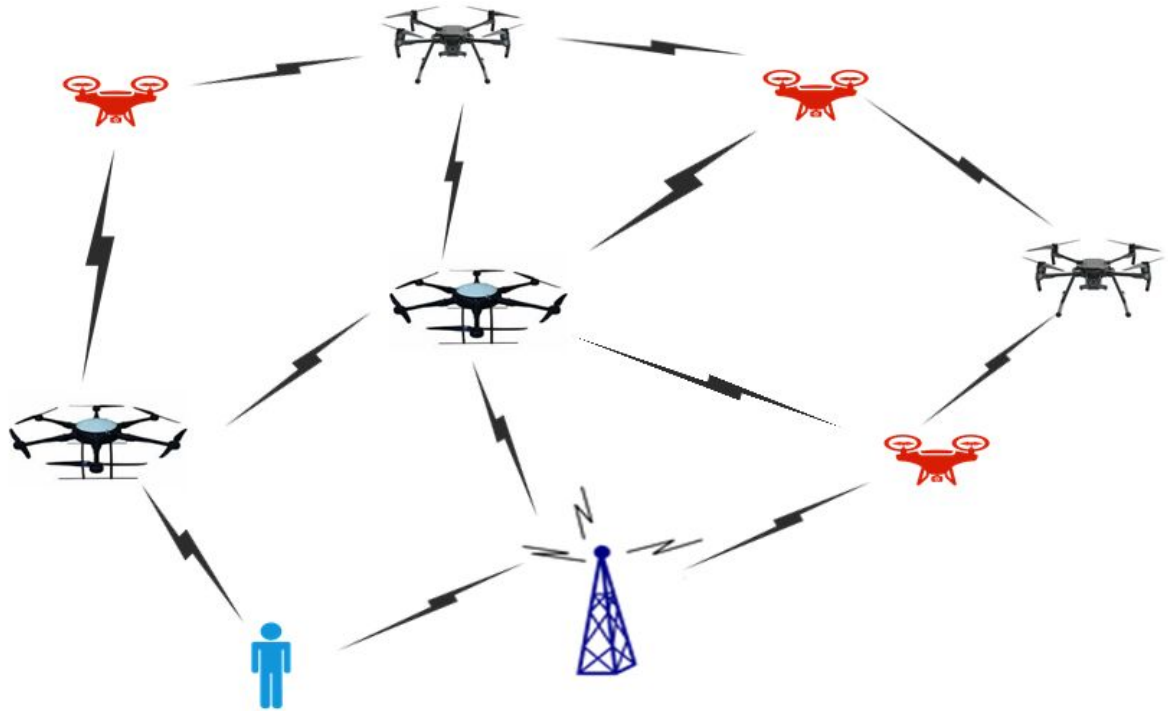


Self-adaptive drone

- Autonomous navigation;
- Real-time control;

Self-adaptive drone

- Autonomous navigation;
- Real-time control;
- Communication and connectivity;

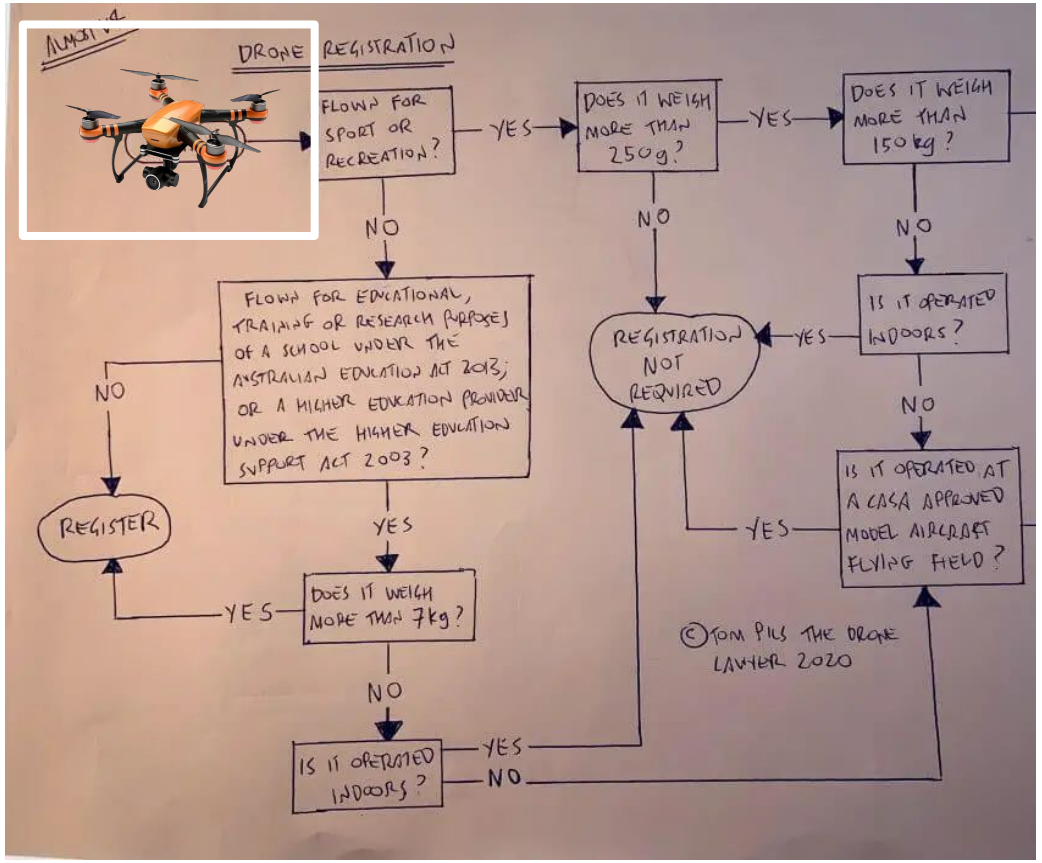


Self-adaptive drone

- Autonomous navigation;
- Real-time control;
- Communication and connectivity;
- Response to faults.

Predefined situations

- The SAS may handle **predefined situations** designed at design time.



Uncertainties

- **Uncertainties** can happen at runtime causing **unexpected situations** and **risk to the mission**.

Sources of uncertainty

Unavailable internal/external resources

Sources of uncertainty

Unavailable internal/external resources

Changes in the environment

Sources of uncertainty

Unavailable internal/external resources

Changes in the environment

Sensor/actuator failures

Sources of uncertainty

Unavailable internal/external resources

Changes in the environment

Sensor/actuator failures

Lack of knowledge

Sources of uncertainty

Unavailable internal/external resources

Changes in the environment

Sensor/actuator failures

Lack of knowledge

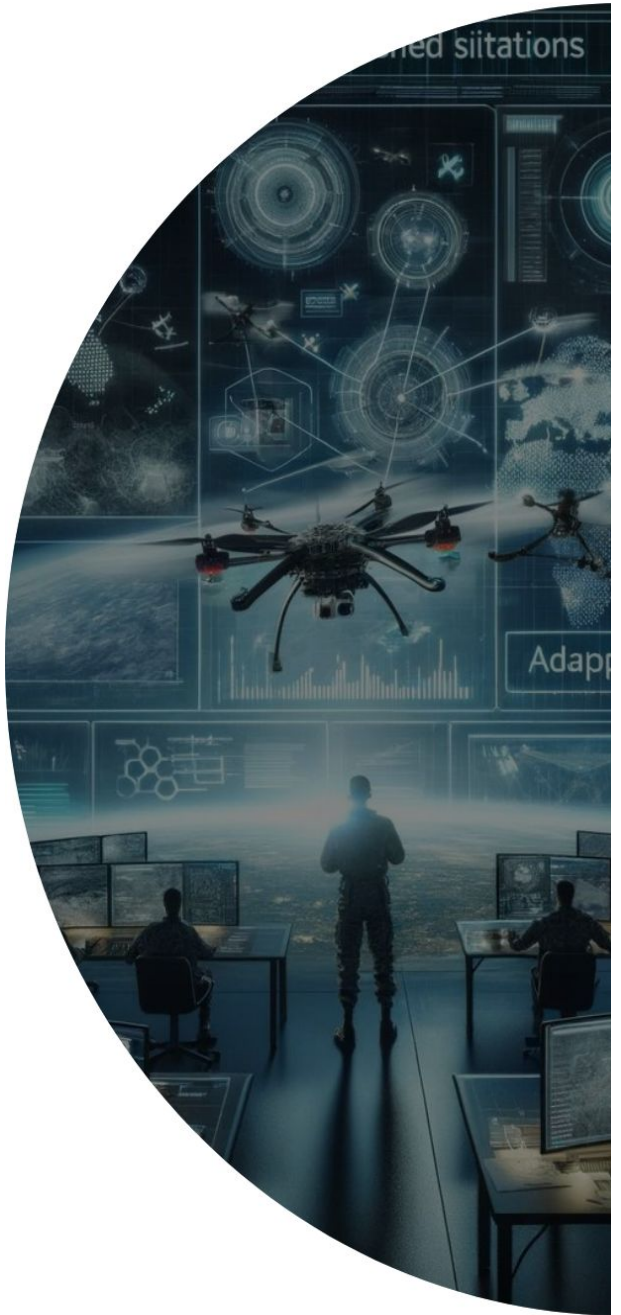
Interaction with humans

Opportunity

Maximizing the drone's chances of achieving its **mission goals** over **uncertainties** is essential.

Approach

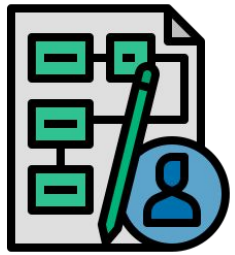
Handling unexpected situations by **monitoring** the system and environment and designing appropriate **on-the-fly adaptation strategies** to satisfy the **goals** of drone-based missions.



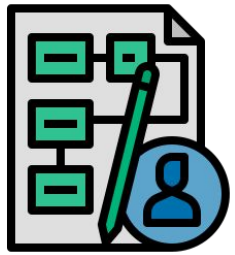
Overview

Design time	Runtime

Overview

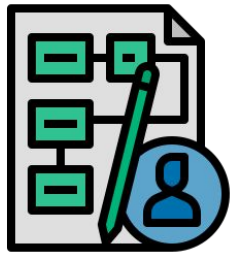
Design time	Runtime
 Expected Scenario specification	

Overview

Design time	Runtime
 Expected Scenario specification	Detection & Identification Unexpected situation

Overview

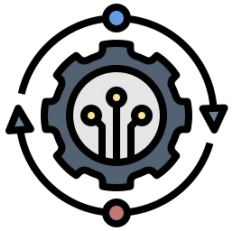
Design time



Expected Scenario specification

Runtime

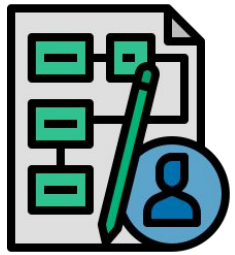
Detection & Identification Unexpected situation



Similarity-based adaptation

Overview

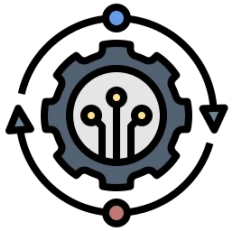
Design time



Expected Scenario specification

Runtime

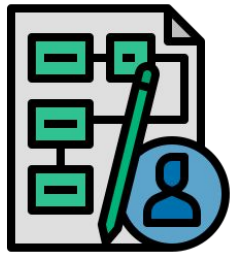
Detection & Identification Unexpected situation



Similarity-based adaptation

Overview

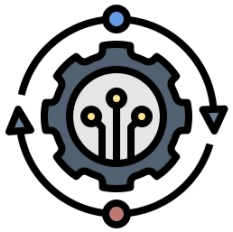
Design time



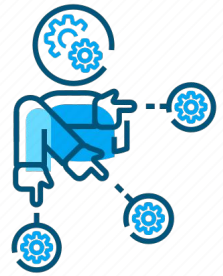
Expected Scenario specification

Runtime

Detection & Identification Unexpected situation

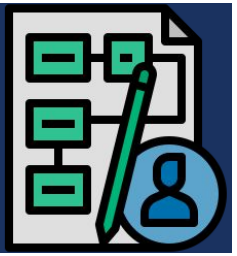


Similarity-based adaptation



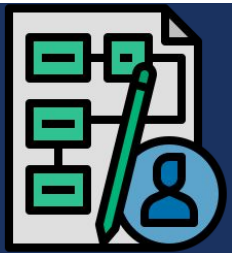
Delegation-based adaptation

Expected Scenario Specification



Behavior-driven development

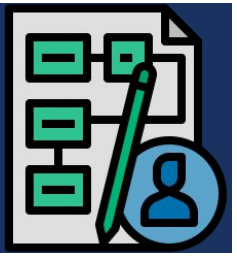
Expected Scenario Specification



Behavior-driven development

- Given (context);

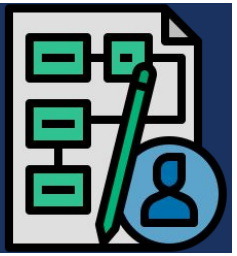
Expected Scenario Specification



Behavior-driven development

- Given (context);
- When (event/trigger);

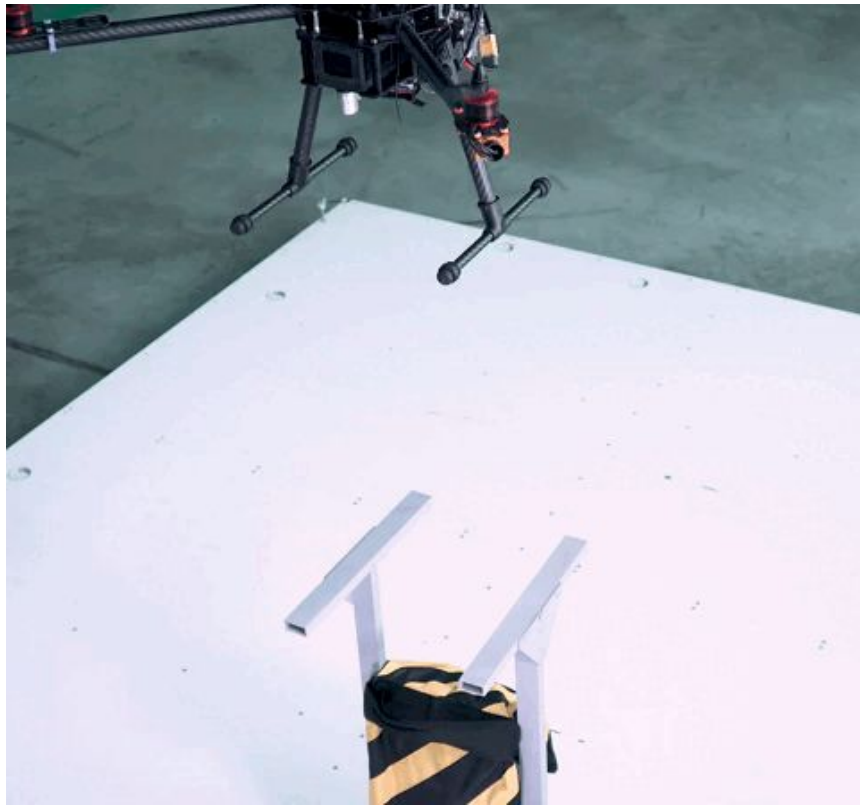
Expected Scenario Specification



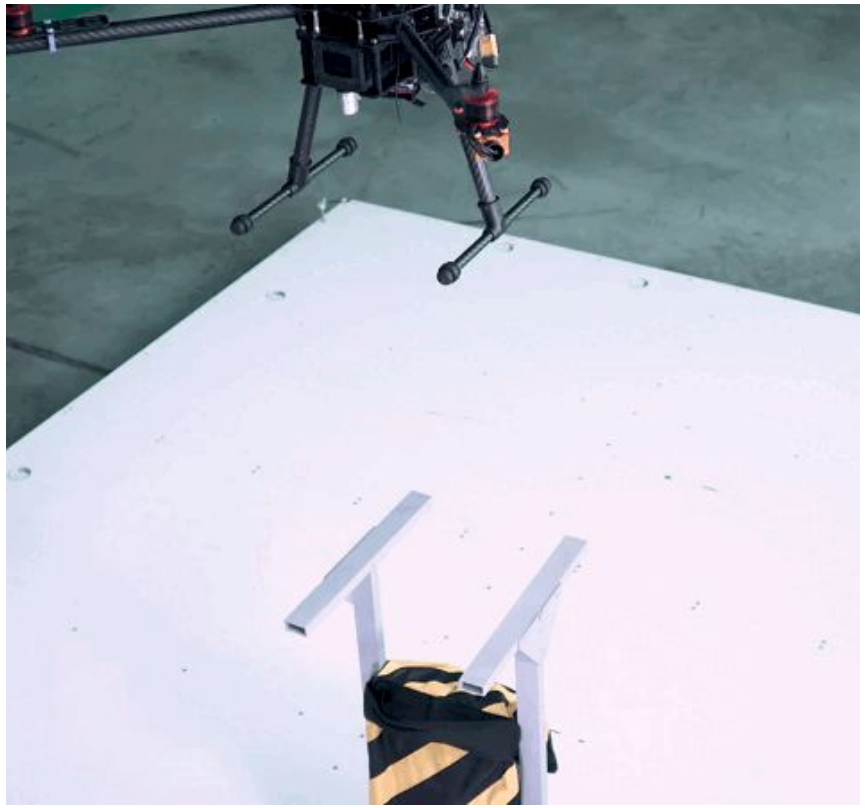
Behavior-driven development

- Given (context);
- When (event/trigger);
- Then (outcome).

Expected Scenario Specification



Expected Scenario Specification



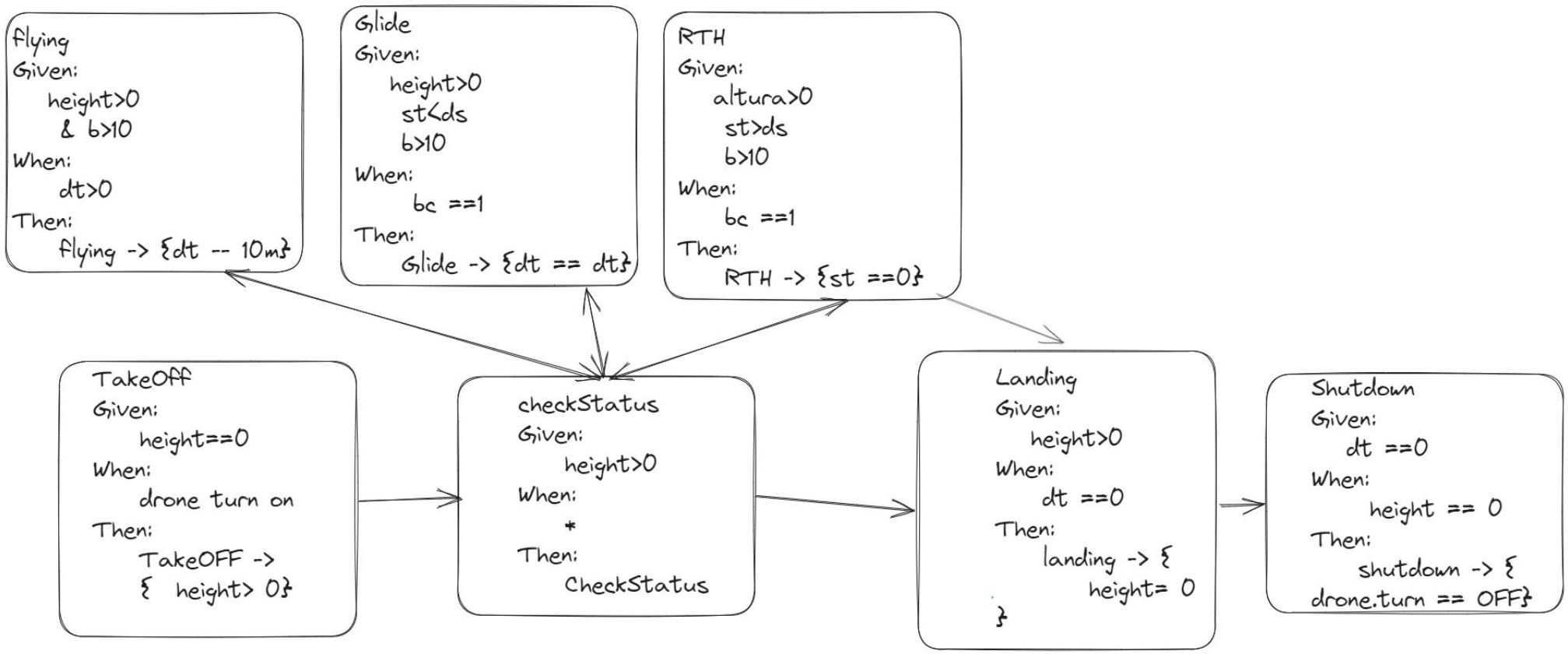
(Flying)

(Over destination)

(Landed at the destination)

```
Deliver
Given:
    drone.height > 0
When:
    drone.targetDist==0
Then:
    landing -> {
        drone.height = 0
        drone.targetDist=0
}
```

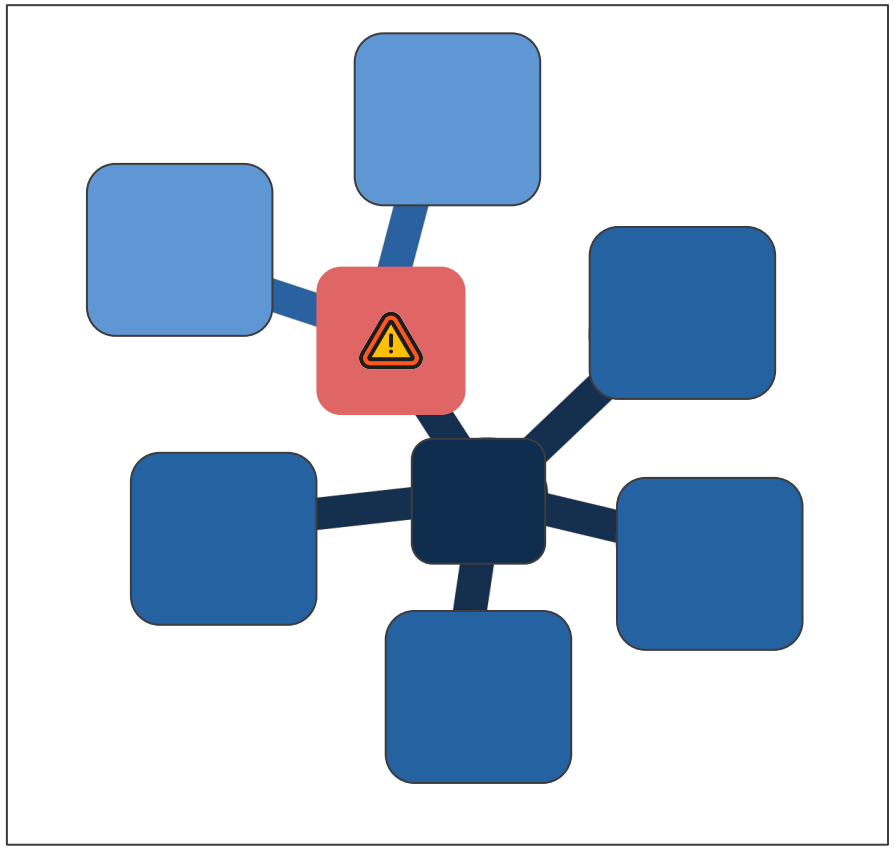
Behavioral model



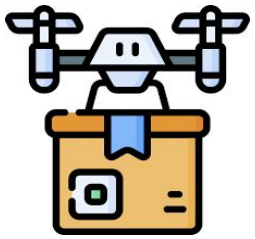
Detection & Identification Unexpected Situation

Detection & Identification of an US:

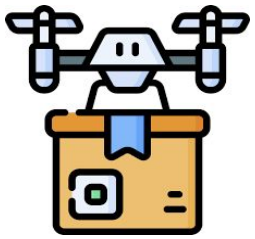
- When a **context** or **outcome** is not as expected.

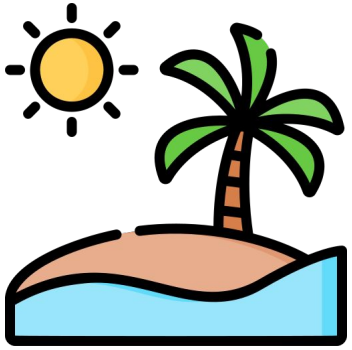


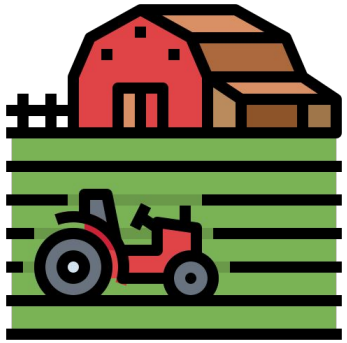
Example



Example

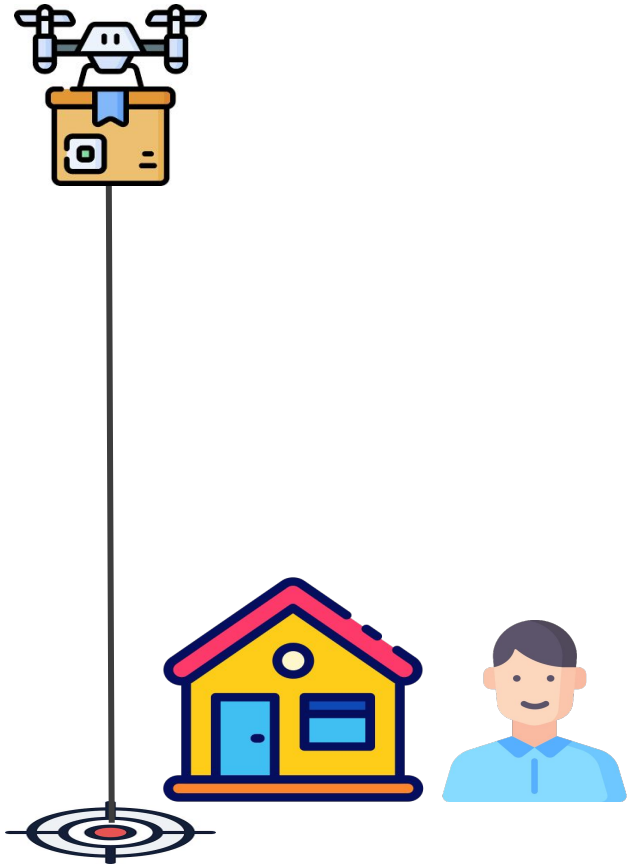






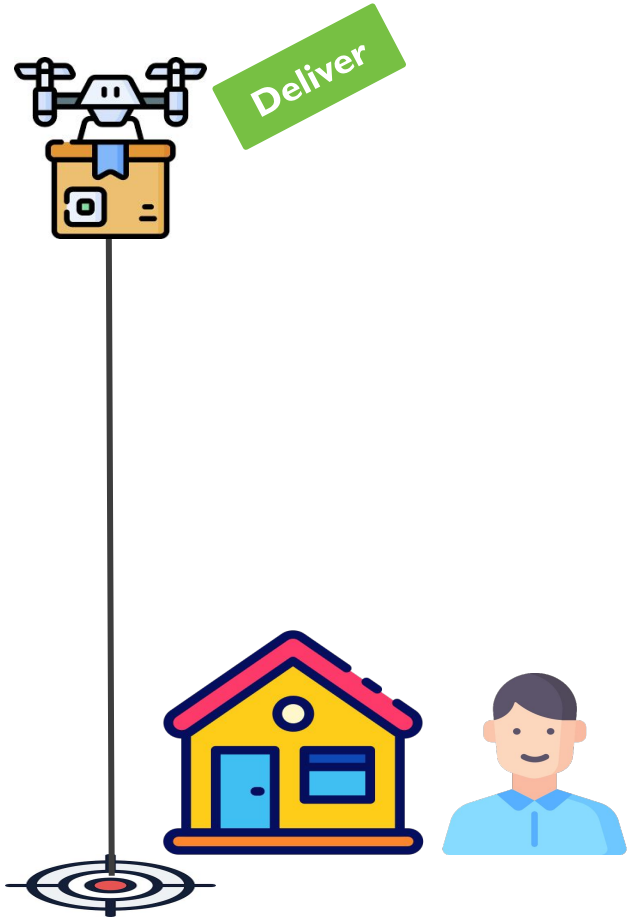
Example

Example



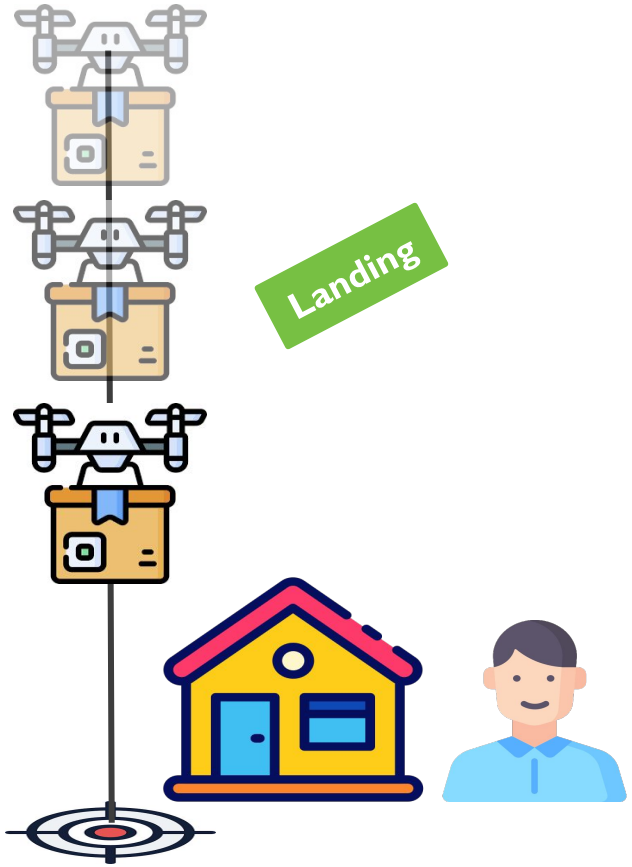
Deliver
Given:
h > 0
When:
dt ==0
Then:
landing -> {
h = 0
dt=0 }

Example



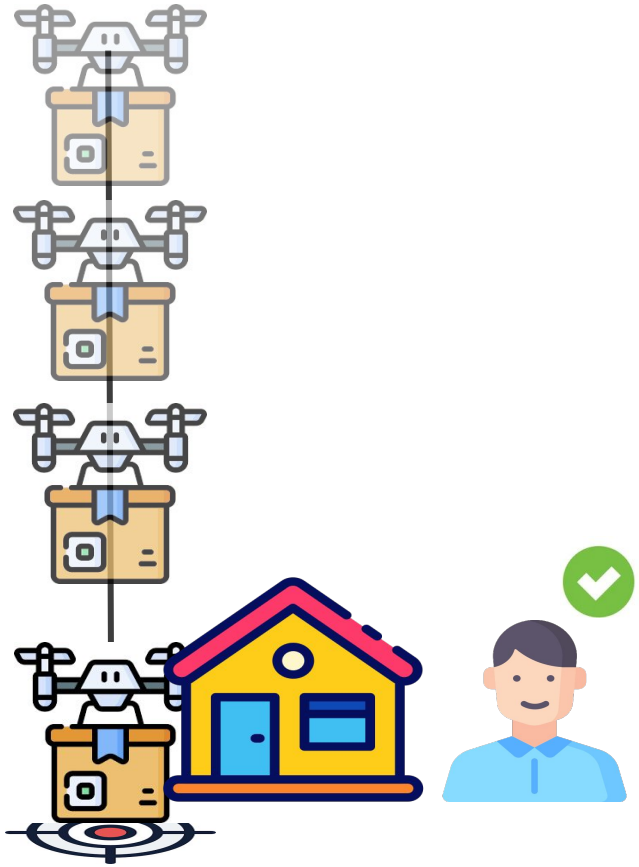
Deliver
Given:
h > 0
When:
dt ==0
Then:
landing -> {
h = 0
dt=0 }

Example




```
Deliver
Given:
    h > 0
When:
    dt ==0
Then:
    landing -> {
        h = 0
        dt=0 }
```

Example



Deliver

Given:

h > 0

When:

dt ==0

Then:

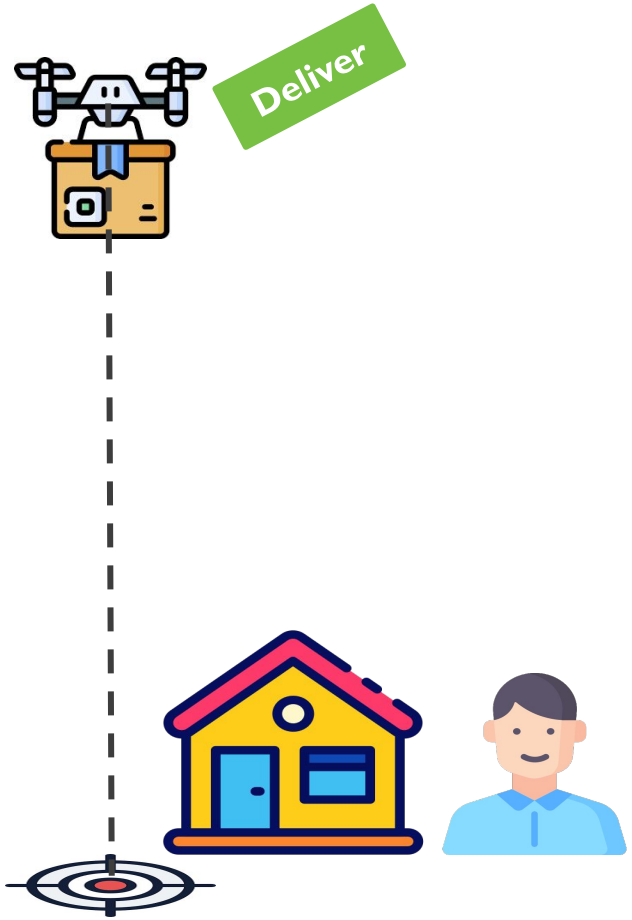
landing -> {

h = 0

dt=0 }

Expected scenario

Example



Deliver

Given:

h > 0

When:

dt ==0

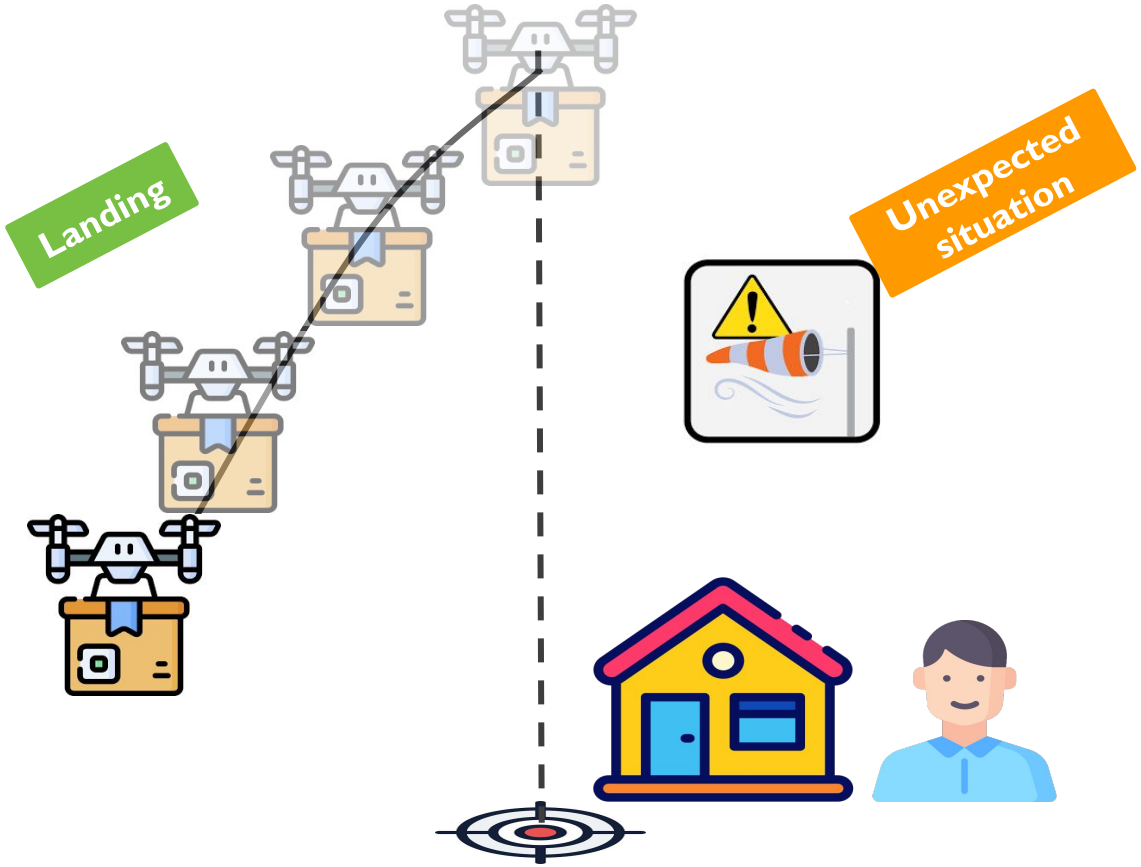
Then:

landing -> {

h = 0

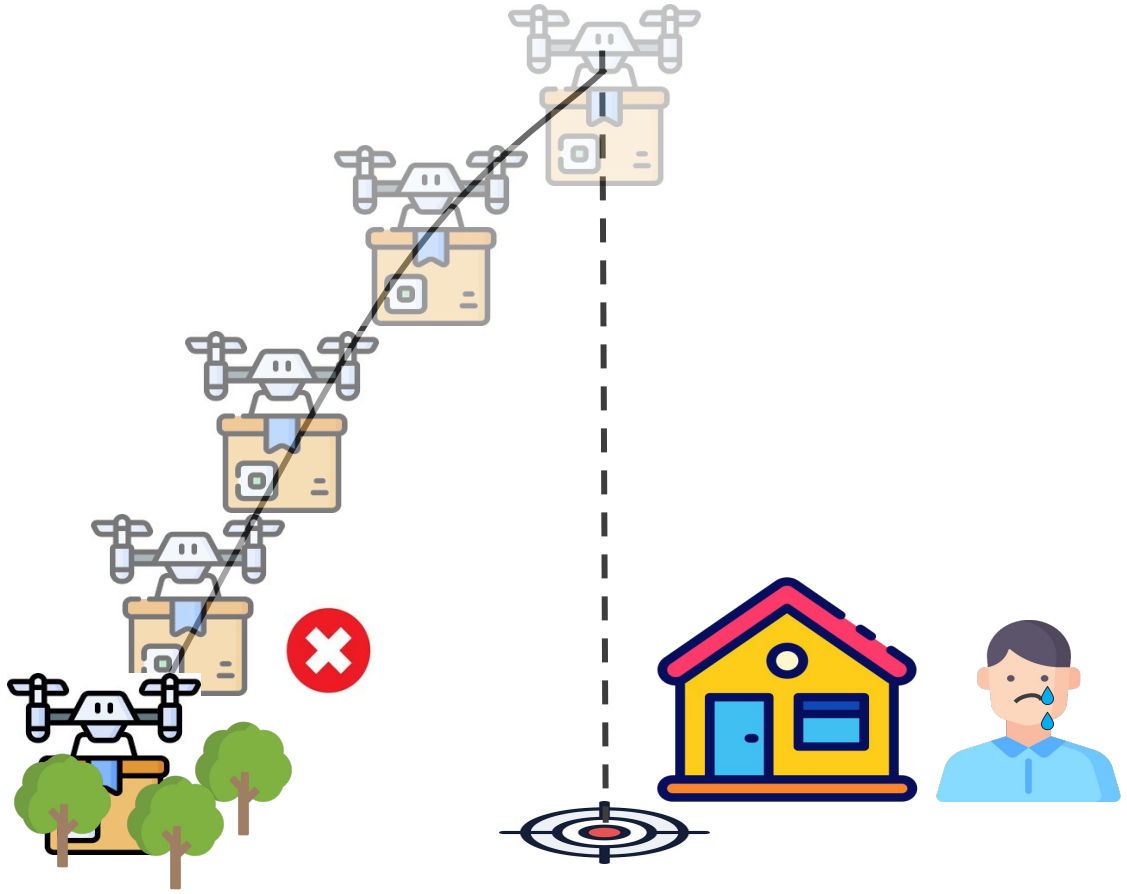
dt=0 }

Example




```
Deliver
Given:
    h > 0
When:
    dt ==0
Then:
    landing -> {
        h = 0
        dt=0 }
```

Example



Deliver

Given:

h > 0

When:

dt ==0

Then:

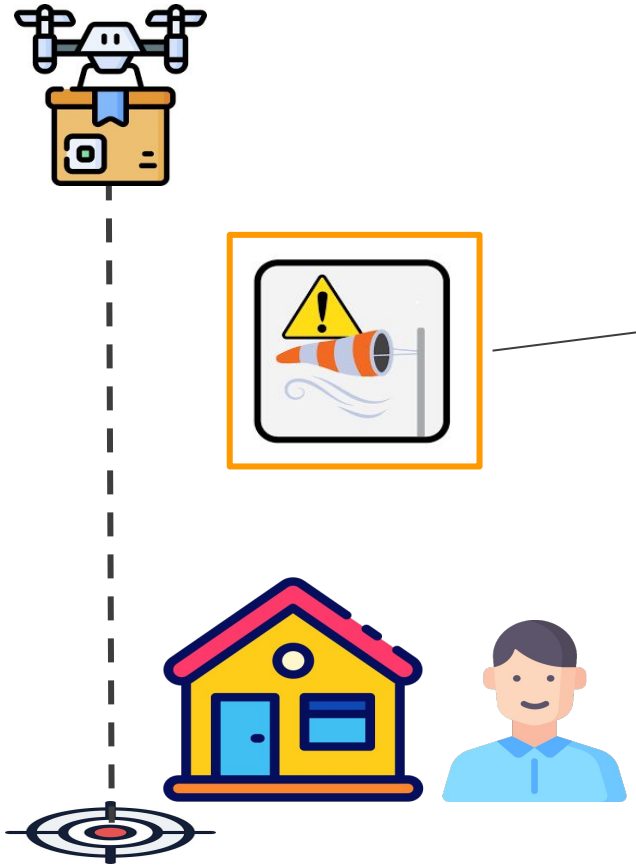
landing -> {

h = 0

dt=0 }

Unexpected scenario

Example

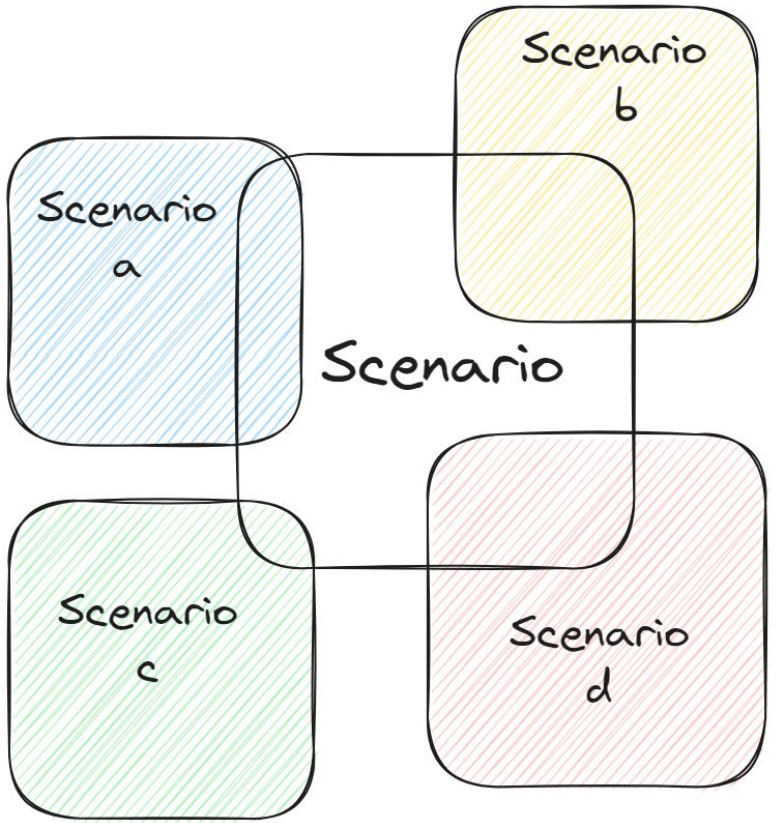


Deliver
Given:
h > 0 AND STRONG_WIND
When:
dt ==0
Then:
??? -> {
h = 0
dt=0 }

Approach: Similarity-based adaptation

Similarity between scenarios

(Work in progress)

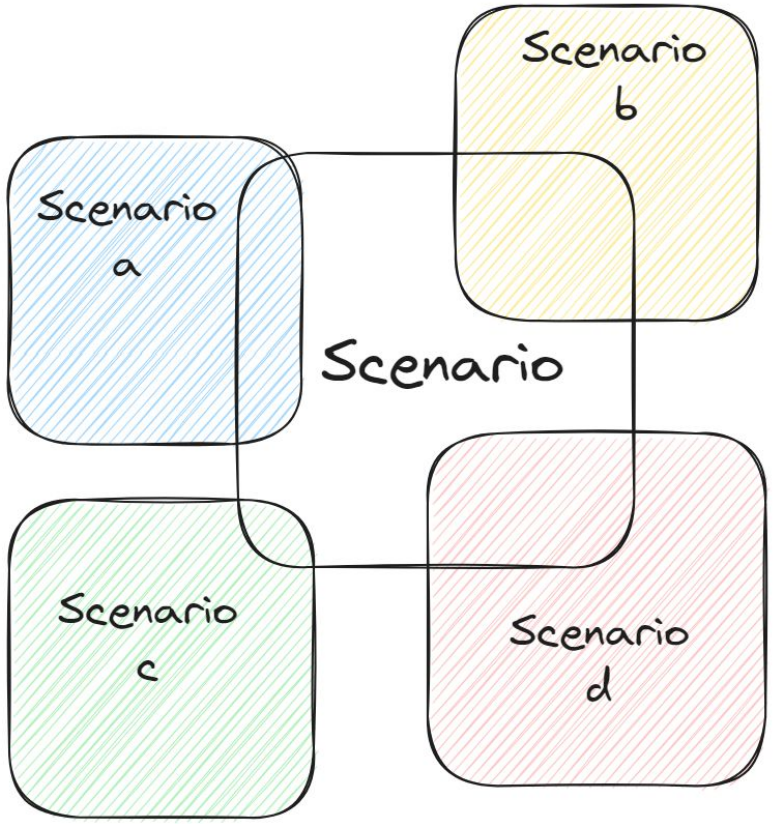


Approach: Similarity-based adaptation

Similarity between scenarios

(Work in progress)

- Method that quantifies the **similarity** between **scenarios**.



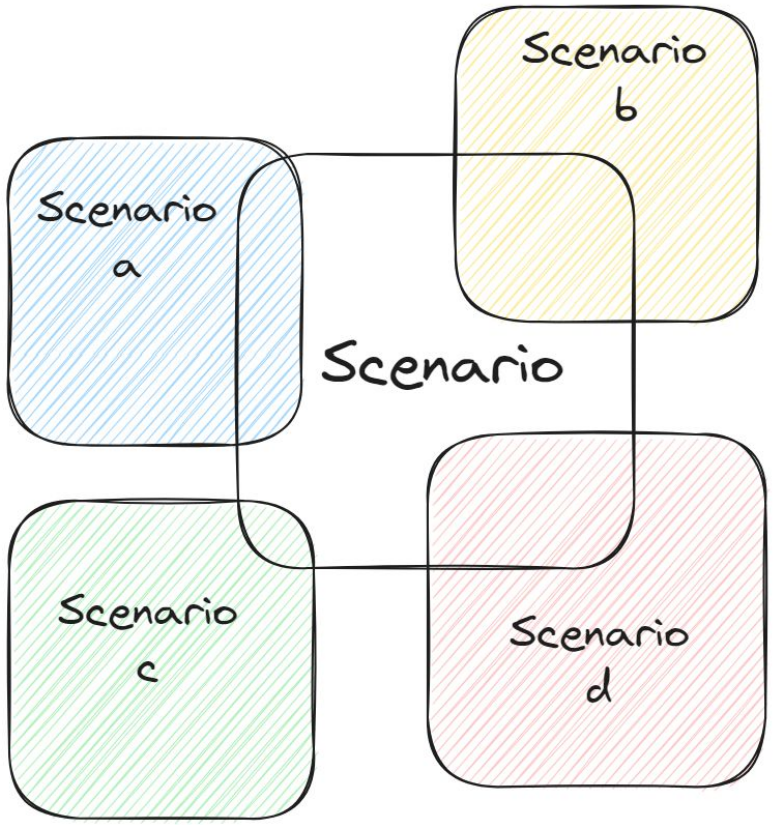
Approach: Similarity-based adaptation

Similarity between scenarios

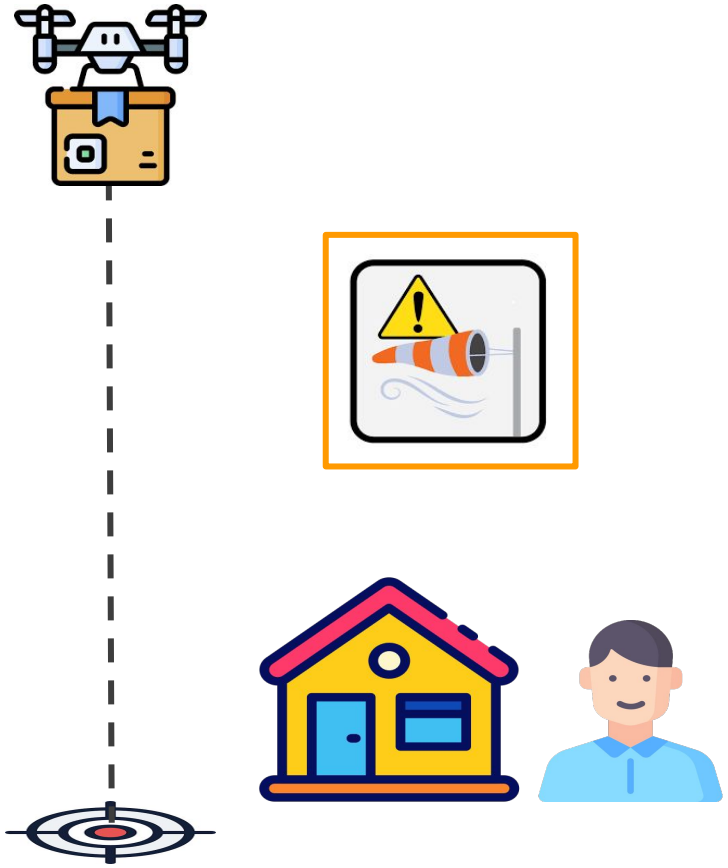
(Work in progress)

- Method that quantifies the **similarity** between **scenarios**.

- Search for a scenarios that is **closer** to the **unexpected scenario** and that will enable the **mission to continue**.

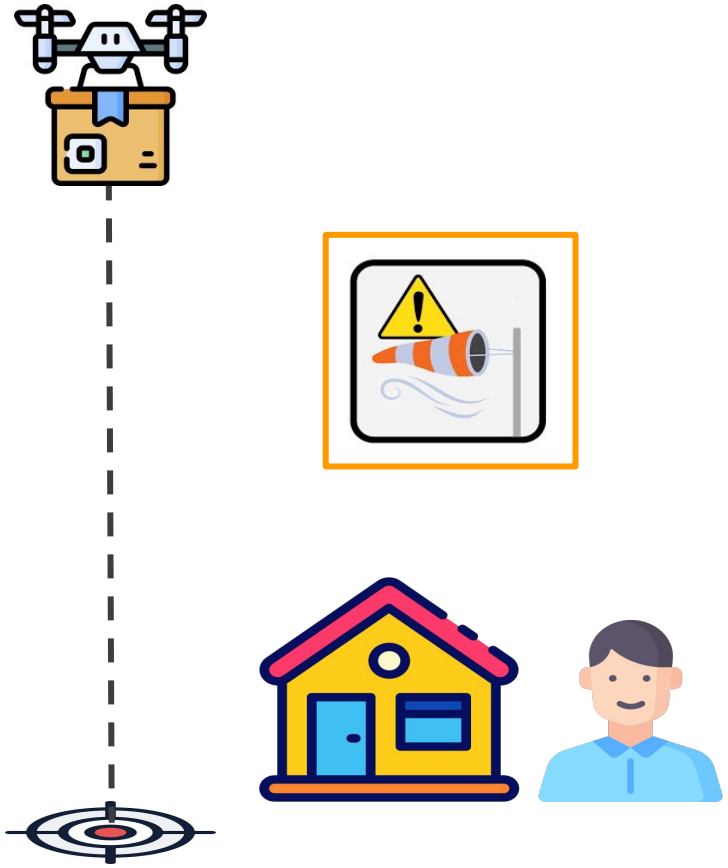


Approach: Similarity-based adaptation



Deliver
Given:
h > 0 AND STRONG_WIND
When:
dt ==0
Then:
??? -> {
h = 0
dt=0 }

Approach: Similarity-based adaptation

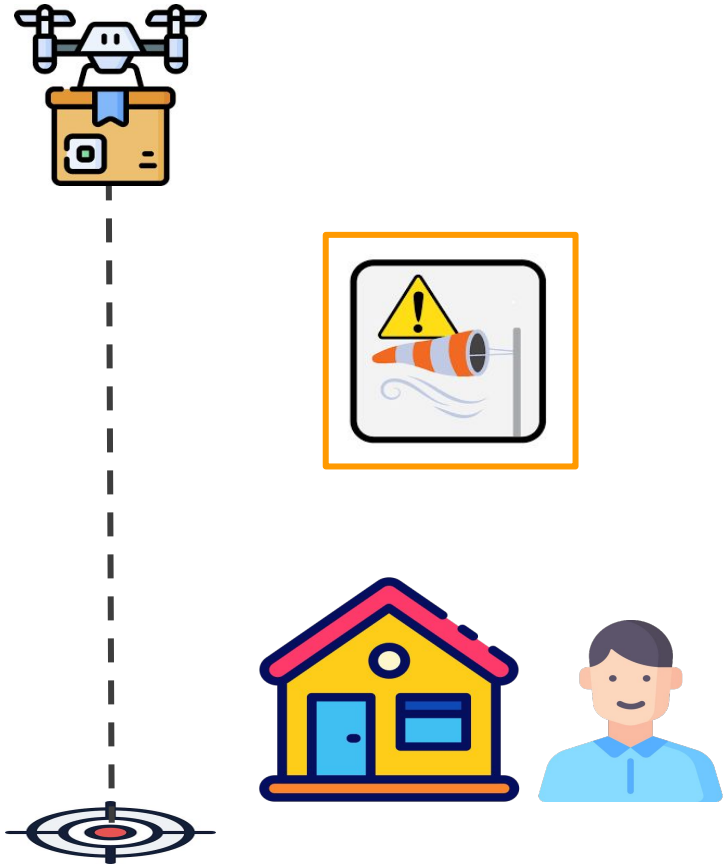


RTH

Given: var = 0
When var2>- 10
Then: xpto -> {
var 3 =0
}

Deliver
Given:
h > 0 AND STRONG_WIND
When:
dt ==0
Then:
??? -> {
h = 0
dt=0 }

Approach: Similarity-based adaptation



RTH

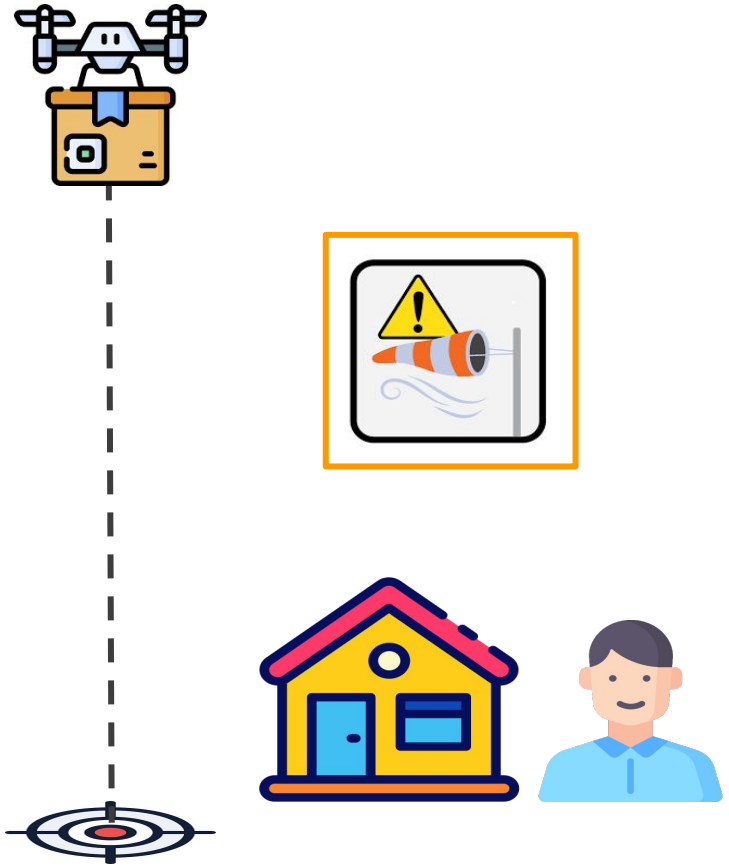
Given: var = 0
When var2>- 10
Then: xpto -> {
var 3 =0
}

RechargeLand

Given: var = 0
When var2>- 10
Then: xpto -> {
var 3 =0
}

Deliver
Given:
h > 0 AND STRONG_WIND
When:
dt ==0
Then:
??? -> {
h = 0
dt=0 }

Approach: Similarity-based adaptation



RTH

Given: var = 0
When var2>- 10
Then: xpto -> {
var 3 =0
}

RechargeLand

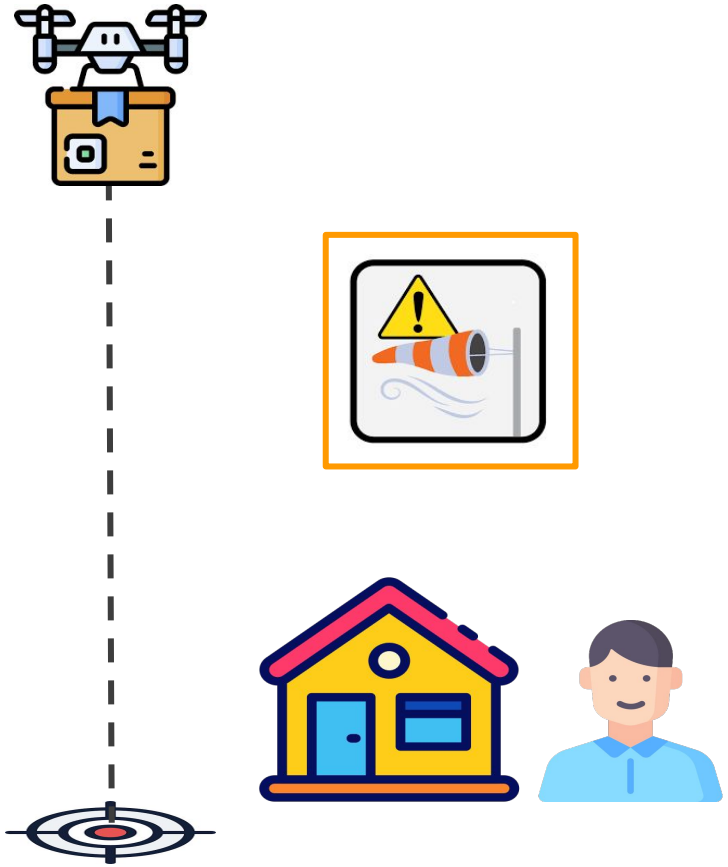
Given: var = 0
When var2>- 10
Then: xpto -> {
var 3 =0
}

SafeLand

Given: var = 0
When var2>- 10
Then: xpto -> {
var 3 =0
}

Deliver
Given:
h > 0 AND STRONG_WIND
When:
dt ==0
Then:
??? -> {
h = 0
dt=0 }

Approach: Similarity-based adaptation



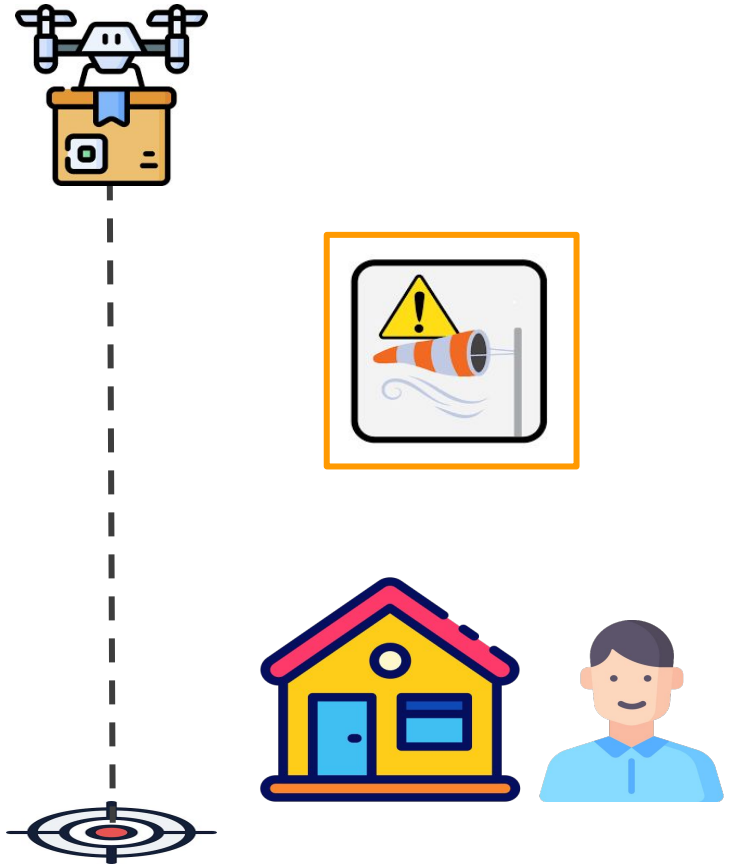
Deliver
Given:
h > 0 AND STRONG_WIND
When:
dt ==0
Then:
??? -> {
h = 0
dt=0 }

Command

SafeLand

Given: var = 0
When var2>- 10
Then: xpto -> {
var 3 =0
}

Approach: Similarity-based adaptation



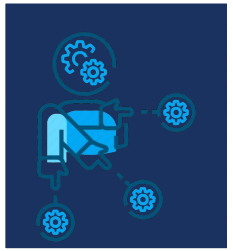
ADAPTED_Deliver

Deliver
Given:
h > 0 AND STRONG_WIND
When:
dt ==0
Then:
??? -> {
h = 0
dt=0 }

SafeLand

Given: var = 0
When var2>- 10
Then: xpto -> {
var 3 =0
}

Second Approach: Delegation-based adaptation



Delegation Strategy
(Work in progress)

Second Approach: Delegation-based adaptation

Delegation Strategy
(Work in progress)

- When there is a **support network**.

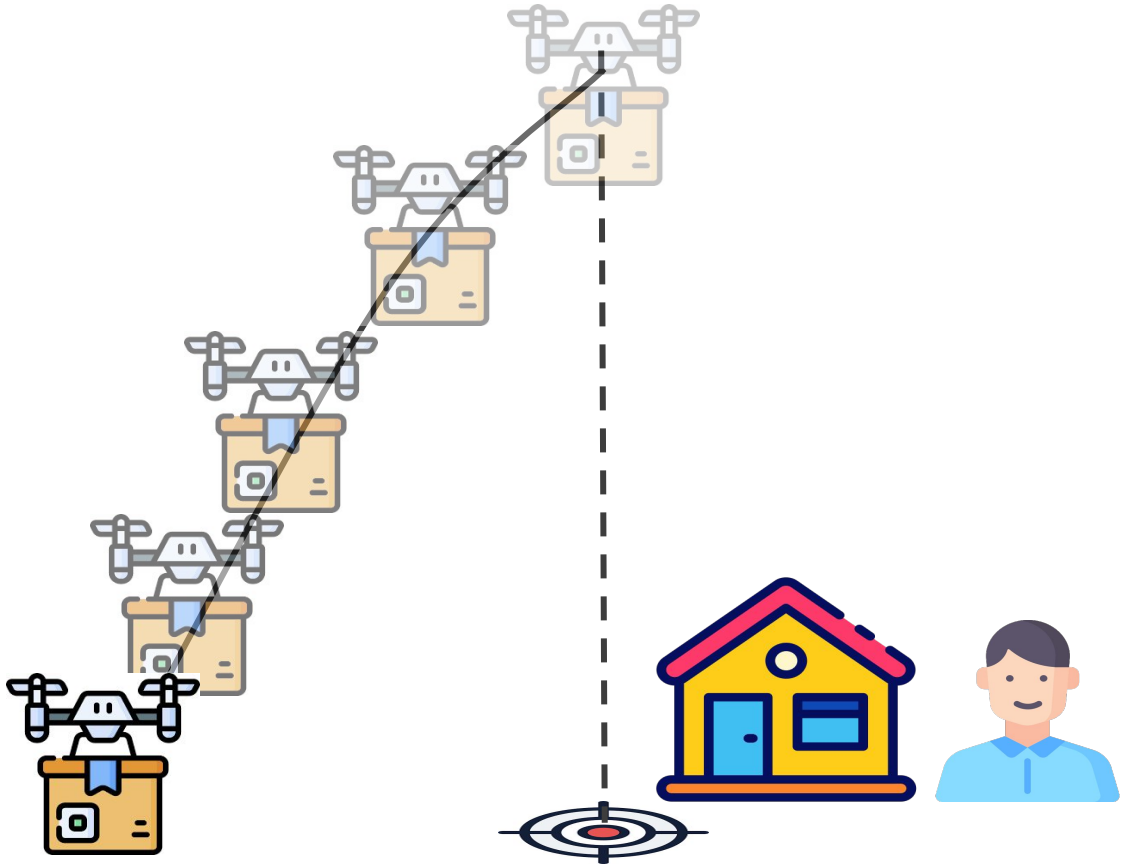
Second Approach: Delegation-based adaptation

Delegation Strategy
(Work in progress)

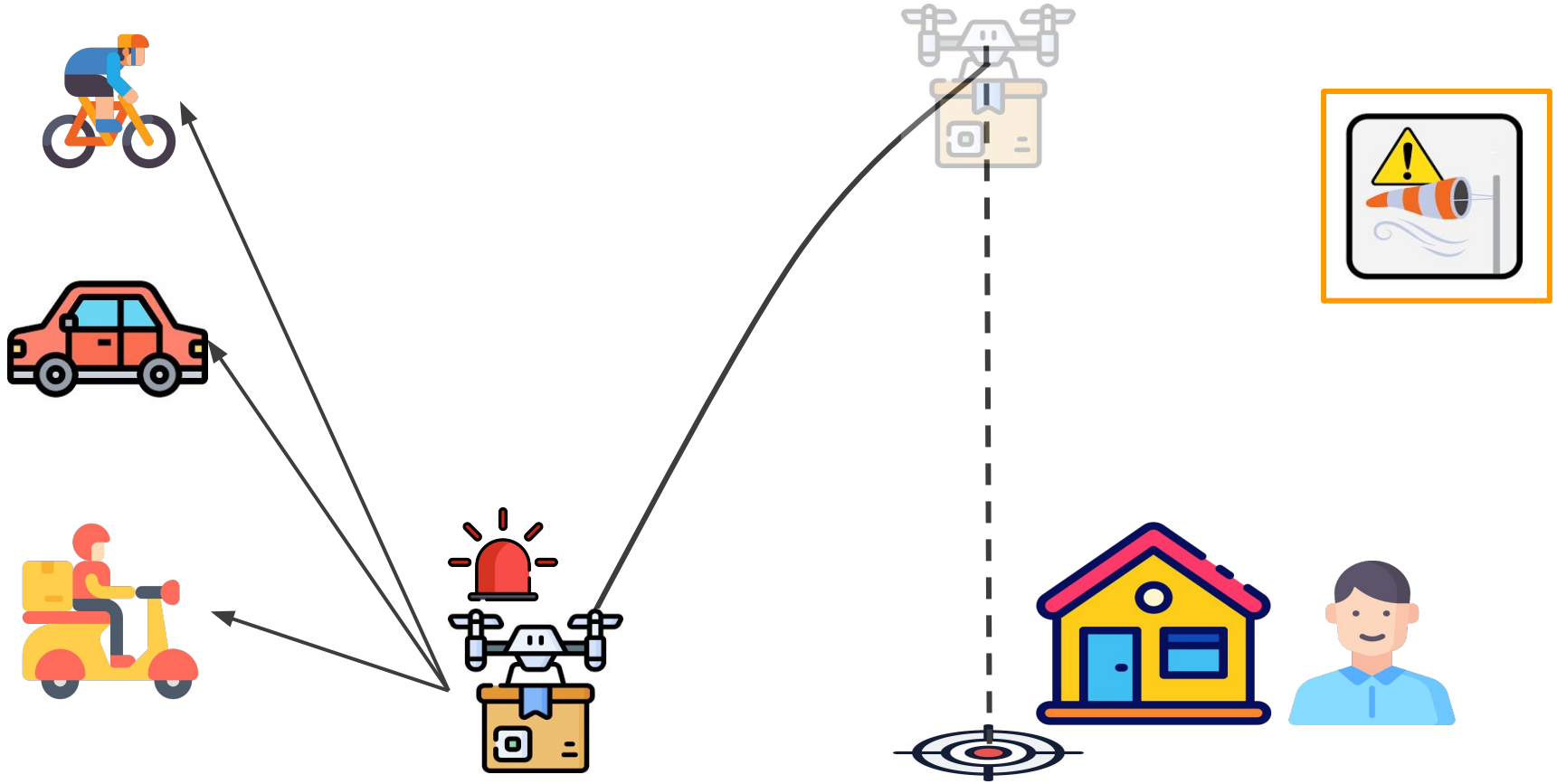
- When there is a **support network**.

- **Benefit-Cost Decision Making.**

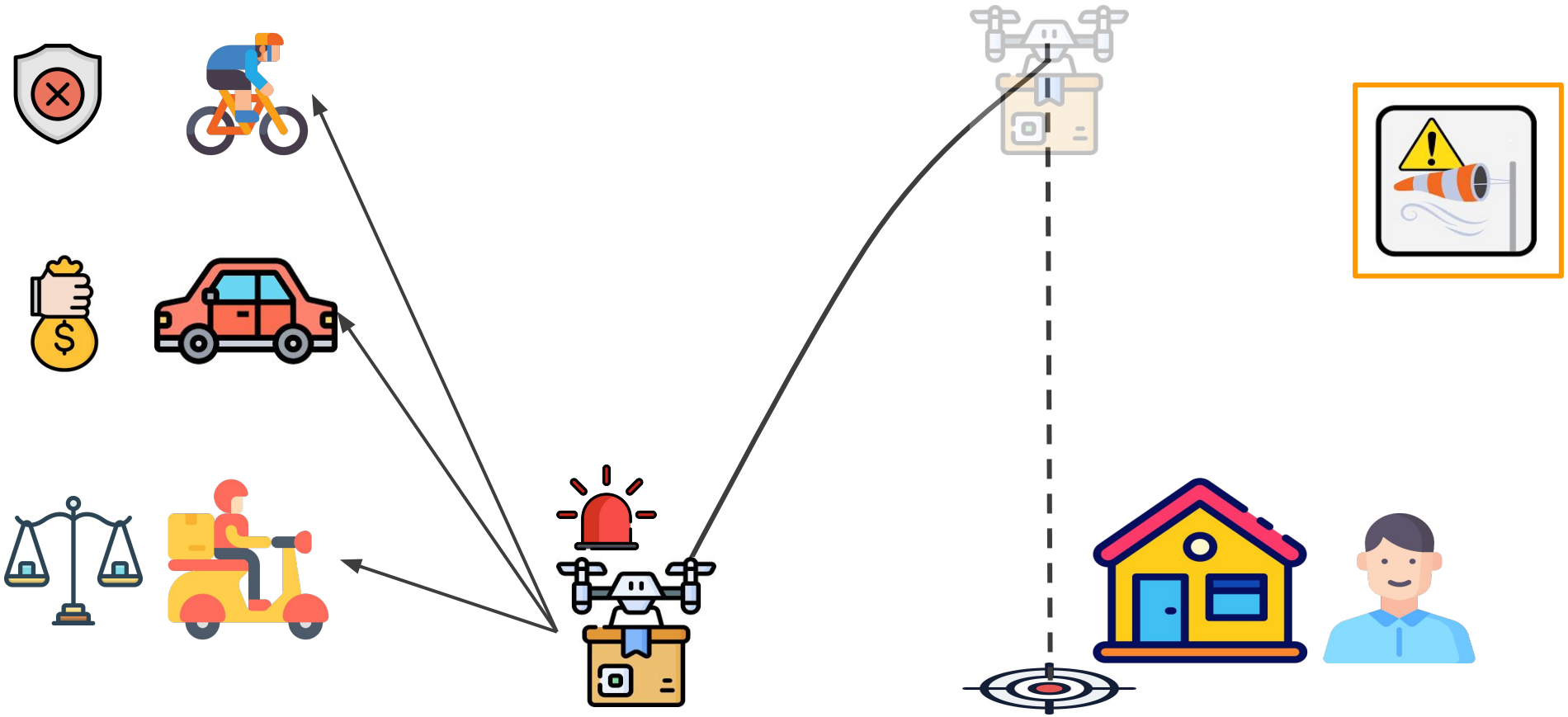
Example



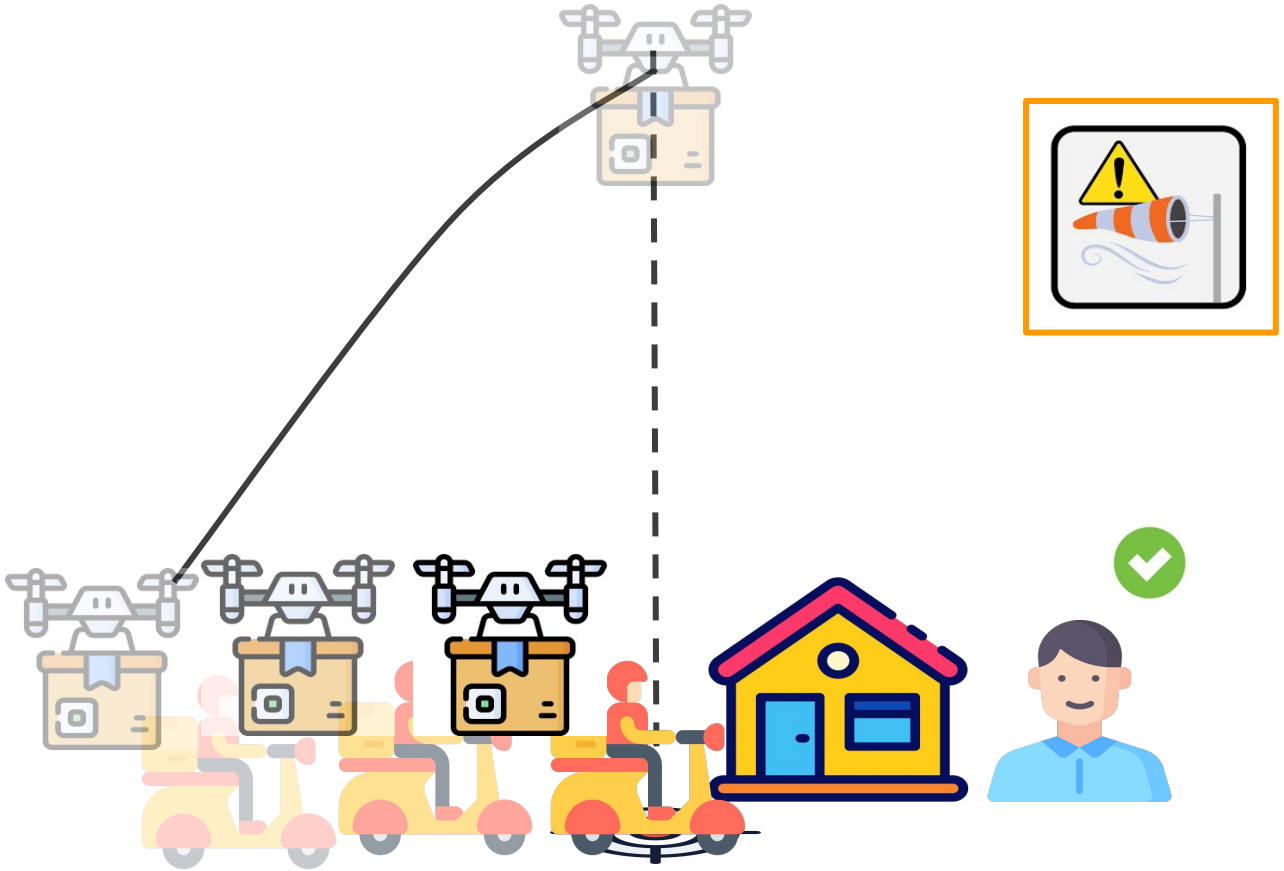
Example



Example



Example



Published Articles

DRESS-ML: A Domain-specific Language for Modelling Exceptional Scenarios and Self-adaptive Behaviours for Drone-based Applications (ICSE-SEIS, 2022)

Cautious Adaptation of Defiant Components (ASE, 2019)

01

02

Dragonfly: a Tool for Simulating Self-Adaptive Drone Behaviours (SEAMS, 2019)

03

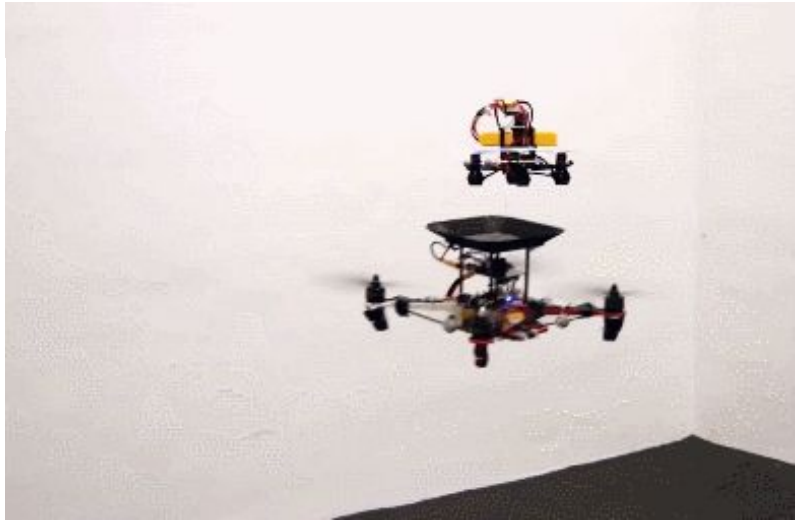
Thank you!

Lucas Vieira Alves
lucas.vieira@aluno.uece.br

GESAD
Adaptive and Distributed Software Engineering Group

The Open University

UnB



```
flying
Given:
    height>0
     & b>10
When:
     dt>0
Then:
     flying -> {dt -- 10m}
```

```
Glide
Given:
    height>0
     st<ds
     b>10
When:
      bc ==1
Then:
     Glide -> {dt == dt}
```

```
RTH
Given:
    altura>0
     st>ds
     b>10
When:
     bc ==1
Then:
     RTH -> {st ==0}
```

```
TakeOff
Given:
    height==0
When:
    drone turn on
Then:
    TakeOFF ->
    {  height> 0}
```

```
checkStatus
Given:
    height>0
When:
    *
Then:
    CheckStatus
```

```
Landing
Given:
    height>0
When:
    dt ==0
Then:
    landing -> {
        height= 0
}
```

```
Shutdown
Given:
    dt ==0
When:
      height == 0
Then:
     shutdown -> {
drone.turn == OFF}
```