Final Report

ICAM'Explorer

Groupe 06 - Cansat

SUMMARY

List of Figures

1/ Context

A. The team and the stakeholders

Stakeholders

Stakeholders of this project Icam'explorer are the customer Grégoire CHABROL, the supervisor: Mathias REHEISSER, Ahmad KAZEMI who is our technical electric expert, Yamen OTHMANI the mechanical expert and Guillaume COUDREUSE: CAD expert. General coordinator of all the projects is also Guillaume Coudreuse.

The team and our organization

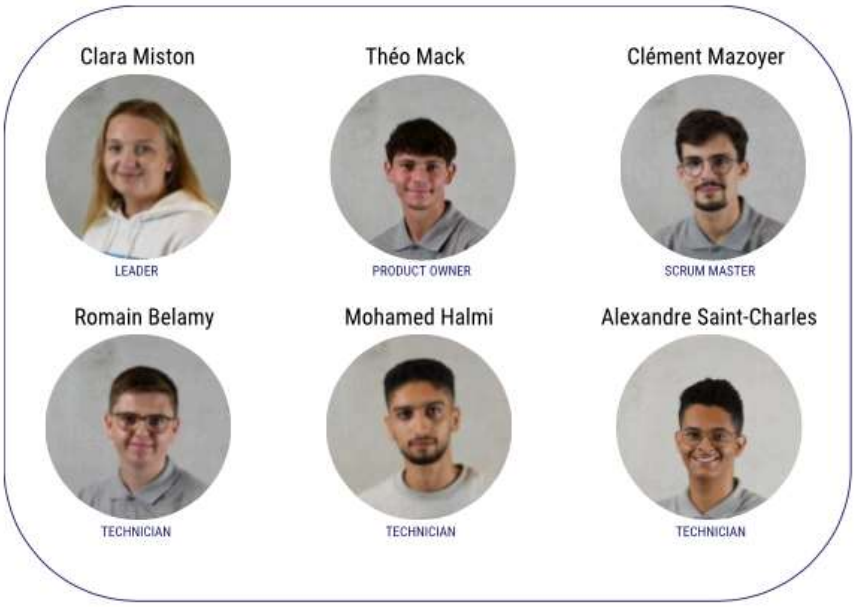


Fig 1: Team members

To carry out this project, the team is made up of 5 students.

Following the Agile method, there's a leader: Clara MISTON, who is responsible for the project's progress. She is also the privileged contact with the customer and the supervisor.

There's a Product owner: Théo MACK, who represents customers and users. He defines the order in which functionalities will be developed, and makes important decisions about the direction of the project.

There's also a scrum master : Clément MAZOYER, who is the guarantor of the Scrum methodology for the functional team. He is responsible for ensuring that the team achieves its objectives and that milestones are met.

At the end, there are technicians: Romain BELAMY, Mohamed HALMI and Alexandre SAINT-CHARLES, who complete the team. They are the driving force behind the project and are in charge of the technical aspects.

In terms of organization, the Agile method is used as mentioned above. Organizing a Cansat project using the Agile method involves a flexible, iterative approach, dividing the project into several sprints. Each sprint focuses on specific objectives, from design to validation, with daily meetings for follow-up and reviews at the end of each sprint to assess progress. This method encourages collaboration, adaptability to change and innovation.

To illustrate this, this table shows the part of the backlog concerning the organization aspect of the project :

6	6.1	define roles on the team	low
	6.2	build each sprint efficiently and then discuss possible improvements	medium
	6.3	organize regular meeting with the team to follow the advancement	low
7	7.1	deliver on time	high
8	8.1	respect the dimensions of the CanSat described on the regulations	high
	8.2	respect technical data for internal components described in the regulations	high

Fig 2: Organisational part - Backlog

B. Challenge

This project offers the team a chance to participate in the CanSat competition, an annual international event organized by Planète Sciences. Open to students, this competition is held in various countries including the United States, Japan, Argentina, France and others. The challenge is to build a miniature satellite, sized to fit within a standard 33cL or 1L can, capable of executing multiple scientific tasks assigned by the organizers, as well as additional missions chosen by the team. The project spans a year and culminates in a competitive gathering of all teams.

During the competition, the CanSats are released from approximately 150 meters above ground, and teams strive to successfully complete both the assigned and self-chosen missions.

For the notation of the drop, there is a point system within the target that the ICAM'Explorer have to mark. More the marking is precise and at the center of the target, more the ICAM'Explorer will get points.

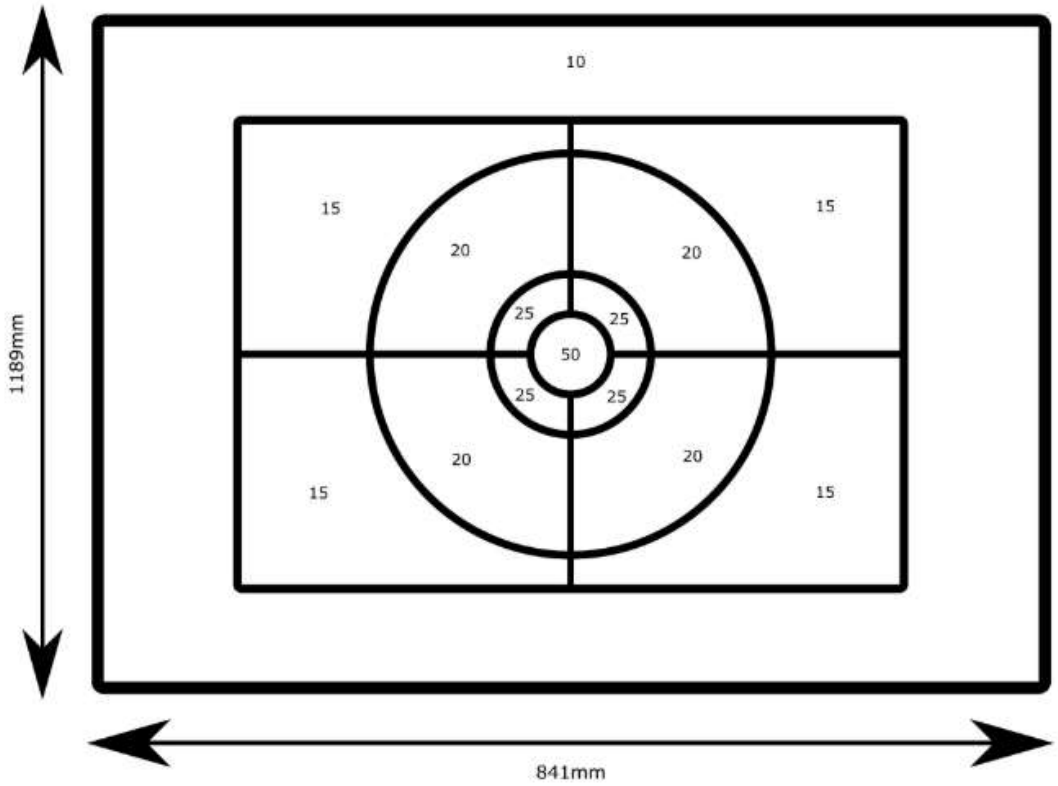


Fig 3: Target scheme

This project serves a dual purpose : it not only enhances scientific skills in programming, design, execution, and testing, but also fosters collaborative teamwork and project management.

2/ The project

A. What's a CanSat ?

Imagine a CanSat as a small, self-contained satellite that's about the size and shape of a soft drink can. It's not a full-fledged satellite like the ones launched into space but a scaled-down version.

CanSats are designed for educational and training purposes. They are used to teach students or aspiring engineers and scientists about the principles of space science, engineering, and data collection.

Within the confines of a CanSat, you'll find an array of miniature components carefully arranged to fulfill their unique tasks.

CanSats are not left on a shelf to gather dust; they are meant to go on a journey. The launch phase is where the CanSat is propelled into the sky, In our case thanks to a drone.

As the CanSat ascends into the sky and later descends back to Earth, it diligently collects data. This data could be about temperature changes, pressure variations, or even images of the Earth's terrain.

B. Technical Data

- Drop Height : 150 meters
- CanSat's maximum volume : 1 liter
- Maximum CanSat Weight : 1 kg
- CanSat minimal autonomy : 45 minutes

There are more constraints on the CanSat physical properties due to the drone's CanSat dropping box. Thus, the CanSat can not exceed the following dimensions: 80 * 200 mm.

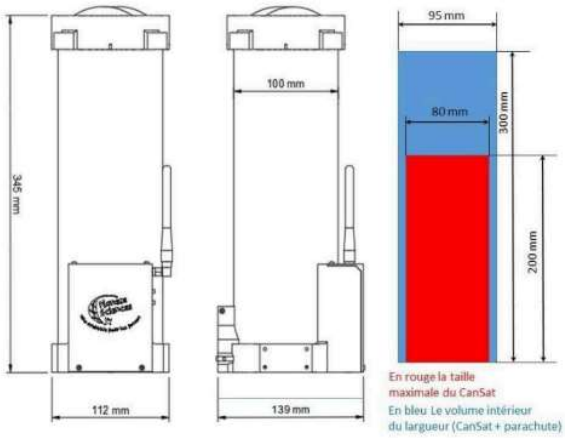


Fig 4: General CanSat Dimensions

C. Description of the objectives

Main Objective : Marking the position

The primary goal is to capture the most extensive ground area possible. This is achieved by marking a designated area. The most significant portion lies at the center of a specified target point, with the size of the areas diminishing as you move away from the center. The target GPS coordinates are set at 43° 13 '18.7"N 0° 03' 10.0"W, but note that these coordinates may be altered by the organizers and should be programmable into the CanSat on the day of the drop.

Environmental considerations are paramount; thus, only biodegradable paint (verified with a reference) or other eco-friendly marking techniques (also backed by references) are permitted. The target area is demarcated using a square of OSB wood, with specific regions marked for scoring. The organizers will measure the marked area within five minutes of the CanSat's landing.

Secondary Objective :

Only one secondary mission was chosen between the three given by Planet Science:

- **Stand upright:**

On the ground, the CanSat must be oriented vertically with the axis of the CanSat pointing skywards. The mission is validated if the CanSat is upright when recovered :

- 20 points if the CanSat is upright with an inclination between 70° and 90°.

- 10 points if the CanSat is tilted between 45° and 70°.

D. SMART objectives

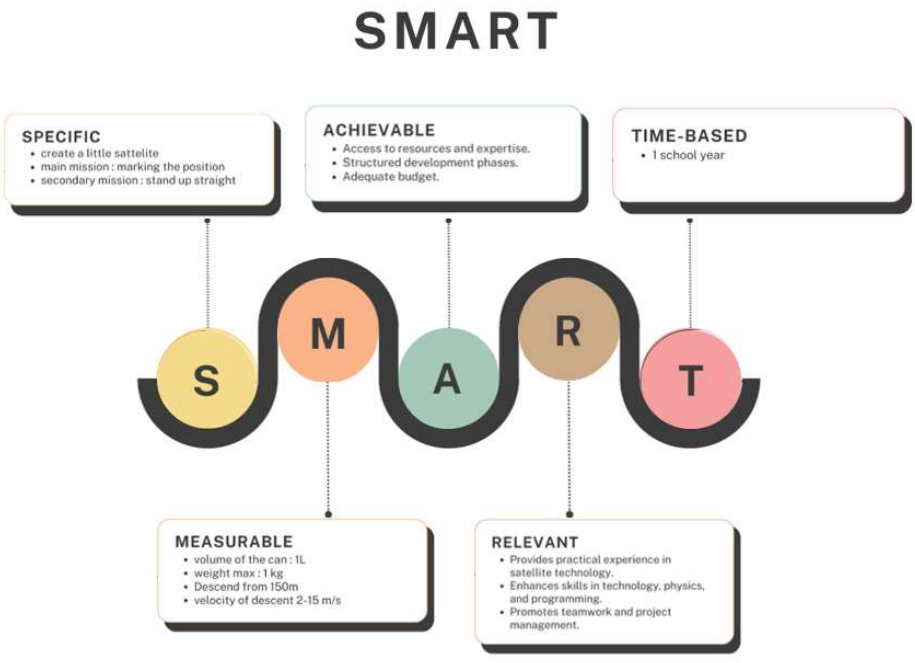


Fig 5: SMART Diagram

To get a feel for the project, a SMART analysis to visualize and better understand the project was made.

E. Functional analysis

To more precisely articulate the goals of the project, we conducted a functional analysis for Icam'Explorer. This includes an "octopus diagram" and an accompanying table that outlines the various functions identified in the diagram. This approach helps in breaking down and understanding the different components and roles within the project.

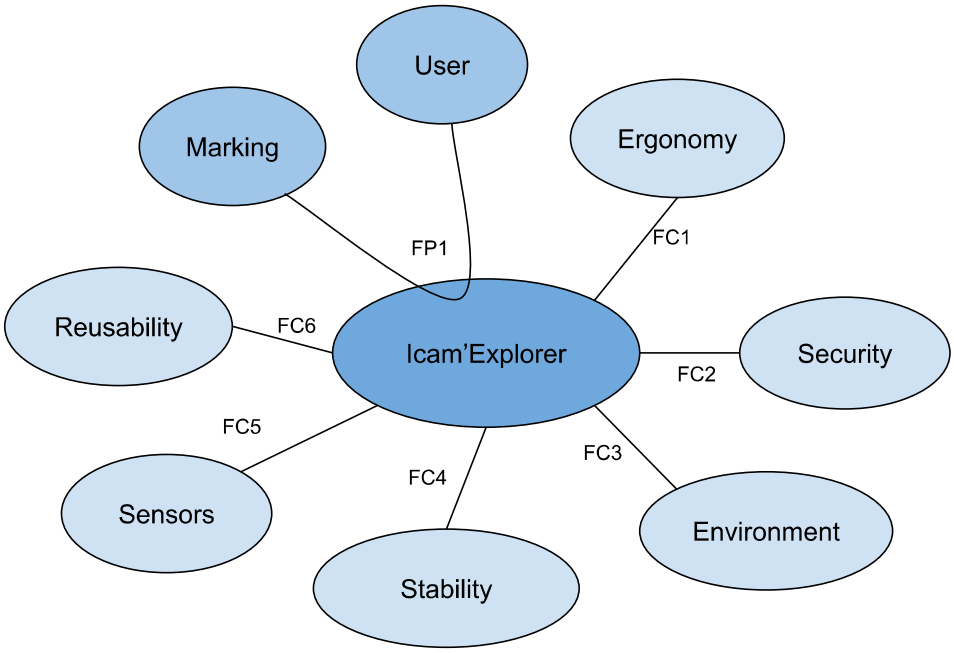


Fig 6: Octopus Diagram

F. Product Backlog

The following section presents the 'Product Backlog' from my report, which is a comprehensive list of tasks, requirements, and objectives that need to be addressed and completed throughout the project. This backlog serves as a central guide for the project's development process, outlining priorities and expectations in a structured manner.

topic	US ID	As a <type of customer>	I want <Perform some tasks>	So that I can <achieve some goal>
missions	1	customer	the cansat to land at a reasonable speed	lands safely without damaging the cansat
	2	customer	a parachute deployement	the CanSat can fall propely
	3	customer	a controlable CanSat system	control the CanSat trajectory
	4	customer	a system to mark the postion	demonstrate the position of the CanSat
	5	customer	a deployement system	the CanSat can stand up straight on the floor
organization	6	supervisor	the team apply the Agil method	follow and control regularly the evolution of the project
	7	customer	all the deliverables in time	follow the evolution of the project
	8	customer	to the team to follow the rules imposed by Planète Science	participate to the competition
conception	9	customer	to have an esthetic design for the CanSat	the project will stand out from others
	10	customer	to have an ergonomic design for the CanSat	fit in with the project's expectations
	11	customer	to have all components integrated into the CanSat	follow the rules of the competition
	12	customer	a CanSat which respect the dimensions imposed	participate to the competition
DD&RS	13	customer	a reusable CanSat	do several launches
	14	customer	CanSat to be made as much as possible from low-carbon materials	limit my impact on the environment

US ID	task ID	description task	priority
1	1.1	design a parachute capable of reducing the CanSat's fall speed	medium
	1.2	design a resistant and efficient parachute	medium
2	2.1	design a system which deploys the parachute on release	medium
	2.2	automate parachute deployment	low
3	3.1	design a remotely controllable on-board system	medium
	3.2	design a system with a minimum range of 150m	high
	3.3	design a system with an autonomy of at least 45min	medium
4	4.1	design a solution able to mark the position where the CanSat land	high
	4.2	design a solution which activates automatically on landing	high
5	5.1	design a solution which allows the CanSat to stand upright after landing	high
6	6.1	define roles on the team	low
	6.2	build each sprint efficiently and then discuss possible improvements	medium
	6.3	organize regular meeting with the team to follow the advancement	low
7	7.1	deliver on time	high
8	8.1	respect the dimensions of the CanSat described on the regulations	high
	8.2	respect technical data for internal components described in the regulations	high
9	9.1	create an original design to stand out from others groups	low
	9.2	print the first prototype	medium
	9.3	create an attractive design to draw attention to the project	low
10	10.1	create a CanSat easy to transport and store in a box	low
11	11.1	choose components to fit inside the CanSat	medium
	11.2	make decisional matrix to choose each components	medium
	11.3	research each of the selected components	low
	11.4	make a wiring plan with all the components	medium
	11.5	test the components	medium
	11.6	order individual components	medium
12	12.1	the CanSat must not exceed the dimensions authorized in the regulations	high
	12.2	make the dimensionning of the parachute	low
	12.3	elaborate a final design of the CanSat	medium
13	13.1	Design a CanSat with reusable mechanisms at least 50 times	medium
14	14.1	Use low carbon material for the prototype design	low

Fig 7: User Story & Product Backlog

3/ Mission 1 : Marking the position

A. Problematics

The main mission is to acquire the largest area on the ground. To acquire an area, simply mark something on it. The largest area is in the center of the target point, the further from the center, the smaller the areas. For that we have a GPS position who must be able to be transferred into the CanSat on the day of the drop.

The target is marked out by a square of OSB wood with the areas to be marked.
The organizers will survey the marked area at the end of the 5 minutes following the landing.

B. State of art

This mission is totally new to the CanSat project, so it's hard to know which methods work and which don't. But here's a list of the different methods we can use on our CanSat to answer the problem :

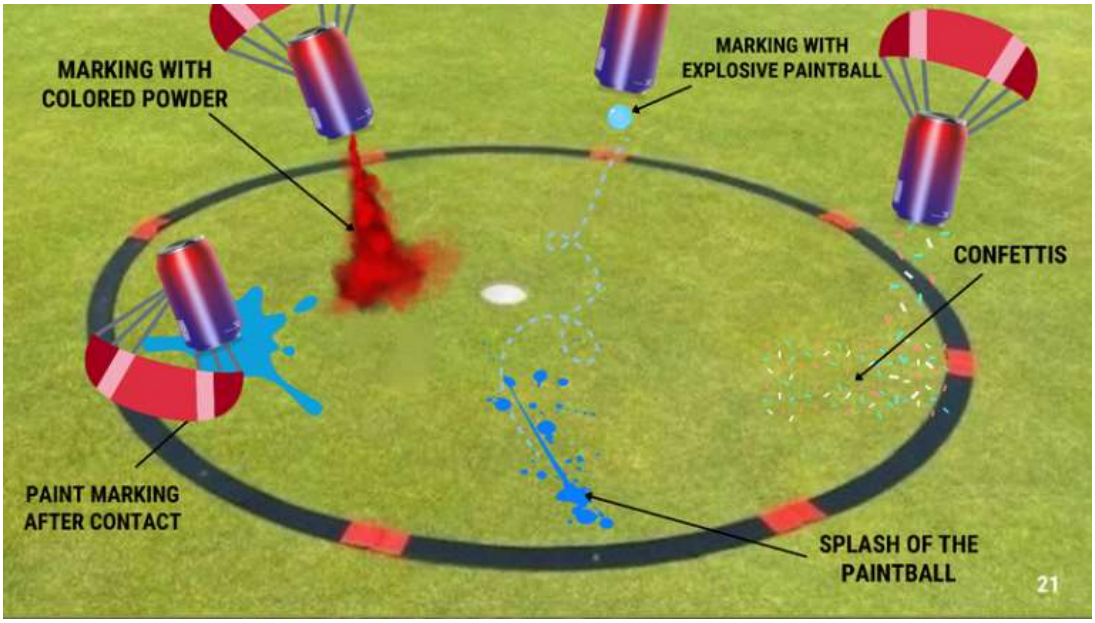


Fig 8: Illustration of marking techniques

This image depicts various methods of accomplishing the primary mission: marking a position. Notably, it includes the use of **colored powder** to mark the target, but this solution seems inappropriate due to the possible presence of wind and the volatility of the powder. Additionally, there is the action of **dropping confetti**, but this would encounter the same problem as with the colored powder because of the lightness of the confetti. Next, there is the

method of marking the position after the Cansat makes **contact with the ground.** Finally, there is the option of marking the position with **paintball-type** balls.

C. Chosen technical solutions

Clearly, using colored powder and confetti to mark the target is not suitable for this project. Furthermore, marking the target after the Cansat makes contact with the ground significantly reduces the marking scope of the Cansat as it will be confined to the landing area of the Cansat. This is why paintball-type balls were chosen to mark the position. Indeed, these balls will be released during flight and will fall onto the ground and the target. Thus, with this solution, chances of marking the target accurately increase.

Henry Barbenchon was also consulted (an expert from Planete Science who is working on the project) about the different ways of marking the target and opted for the method of spraying paint balls during the fall. In theory, this method not only provides the best possible hit on the target, but also effectively marks the wooden target.

The aim is to open a trapdoor under the CanSat, controlled by a servomotor. This trap holds a supply of paintballs which will open when the CanSat is above the target. This will cause a bunch of paintballs to fall, increasing the chances of hitting the target.

After testing various types of pellets (several kinds of homemade balls, paintballs used by professionals), water beads were selected to achieve this task.. These beads swell upon contact with water to measure 1,2 cm in diameter. They are placed at the trapdoor designed for this purpose underneath the Cansat.

Fig 9: Water beads

4/ Mission 2 : Stand up straight

*"**Stand straight** : The final state of the CanSat must be oriented vertically. The aim is to keep the CanSat upright for at least 3 minutes. Pay attention to the nature of the ground!"* extract from "Technical project : Note of context"

A. Problematic

The landing stability of a CanSat is a critical aspect of its design and functionality. Ensuring that the CanSat remains upright upon landing is essential to protect onboard instruments and facilitate subsequent operations. This stability is challenged by the unpredictable nature of landing terrains and the impact forces at touchdown, making it a significant engineering concern in CanSat development.

B. State of art

CanSat with Retractable Feet: This design incorporates extendable feet that deploy upon landing. The feet provide a stable base, reducing the risk of tipping over.

CanSat with Spring Suspension System: This system uses springs to absorb the impact of landing. It helps in distributing the landing force evenly, minimizing the risk of damage or instability.

CanSat with Tilting Weight: A mechanism where weights within the CanSat adjust its center of gravity. This reactive system helps in maintaining an upright position during and after landing.

C. Chosen technical solution

After extensive analysis using a decision matrix and other evaluative tools, the the **"CanSat with Retractable Feet"** design was selected. This decision was based on factors such as **reliability**, **simplicity**, and **effectiveness** in ensuring upright stability post-landing. The retractable feet mechanism provides a balance between technical complexity and functional robustness, making it the optimal solution for our project requirements.

5/ The parachute

A. Problematics

In order to design a parachute for a CanSat, several problems or challenges need to be addressed to ensure a successful mission. Here are some problem areas:

- Size and Weight
- Form of the sail
- Reusability
- Choice of material

B. State of art

Concerning the state of art of CanSat parachute, 3 design stands out:

- the rectangular parachute

+ maniability
- stability

Fig 10: Rectangular parachute

- the round parachute

+ stability
- magniability

Fig 11: Round parachute

- the cross parachute

+ stability and magniability
- fall speed too high

Fig 12: Cross parachute

C. Dimensioning

To calculate the dimensions of the sail, the following formula has been used:

$$A = \frac{m \times g}{cd \times \rho \times V}$$

There is a constraint in the CanSat falling speed: V: 2 m/s < V < 15 m/s

Calculation for minimal parachute size : m = 1kg, Cd= 2,5, ρ=1.225 kg/m3 and V = 15 m/s

$$A = \frac{1 * 9.81}{2.5 * 1.225 * 15} = 0.214$$

Calculation for maximal parachute size : m = 1kg, Cd= 2,5, ρ=1.225 kg/m3 and V = 2 m/s

$$A = \frac{1 * 9.81}{2.5 * 1.225 * 2} = 1.602$$

In the end, the minimal and maximal theoretical sail's sizes are about, respectively: 0.214 m^2 and 1.602 m^2.

D. Chosen solution

a. Choice of the parachute

Initially, a rectangular parachute was chosen, but after an appointment with an expert from Planete Science, the solution was changed to a cross parachute because it is a good compromise between stability and maneuverability. In the end, customization of the parachute was chosen. It combines different shapes to retain the advantages of each. It opted for a cross for its maneuverability, and the slowdown it brings thanks to good air intake.

What's more, following initial tests, the drop speed condition was not satisfied, it was too high. That's why canvas were added in the corners to increase the air intake.
It was also decided to make a hole in the center of the sail to increase the Cansat's stability so that when the paint balls were released, they would go straight down.
Finally the material of the sail that was chosen is nylon for its lightness, its resistance and its low cost.

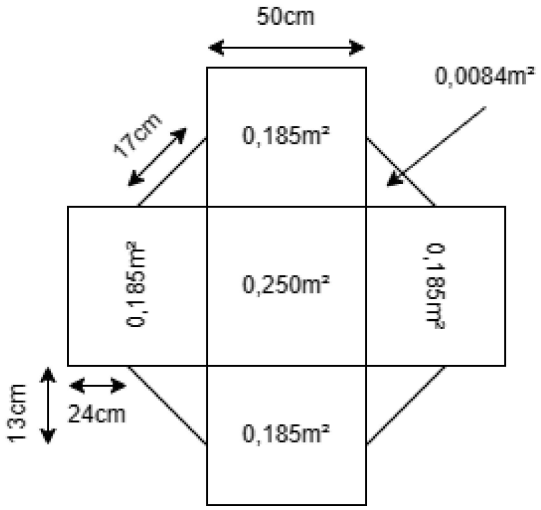


Fig 13: Parachute sizing plan

Dimensioning: Total area of the parachute = 0,250 + (4 x 0,185) + (4 x 0,0084) = 1,0336 ~ 1.034 m^2

In concrete terms, it was realized that compared with theoretical predictions, the area achieved was five times larger than the minimal size yet 35% smaller compared to the maximal sail's size. When it came to parachute sizing, the minimal theoretical size proved insufficient. It provided an initial concept, but it was only through subsequent testing that a result aligning with expectations was achieved. Following the cutting of the canvas and the attachment of the nylon threads, the total weight of the parachute component reached 100.00 grams.

b. Chosen solution to deploy the parachute

The parachute will be free, which means that upon release it will deploy with the speed of the fall.

c. Chosen solution to attach the parachute and to deprive of it after landing

The parachute will be connected to the CanSat by nylon wires and removable clips, allowing good resistance and easy maintenance. These wires are 1m long, which allows air to enter easily under the canvas and inflate it without being too long, which allows the CanSat to remain stable and steerable.

d. Launch data

Thanks to a homemade release system, a drone was firstly used to drop the CanSat from a height of 20 meters. A prototype was used to do this, in order to check certain functions of our product without running the risk of destroying final components. In particular, the right landing solution..

The launch conditions were as follows:

- Wind speed : 15 km/h
- Bottle weight : 450g
- Parachute weight : 100g

Recovered data :

- fall time : 6,67 seconds
- max speed : 3 meters/second

Fig 14: Final parachute

6/ Components

a. Main card choice

Four different cards came to the attention of the team, three different types of Arduino card and a Raspberry Pi. Thanks to this decision matrix the Arduino UNO was chosen because it offers the best compromise between weight and size, number of ports and ease of use.

CRITERIA	I/O		Size and Weigh		Energy consumption		Cost		Calculating Power		Simplicity		Storage Size		OS		TOTAL
FACTOR	4		5		2		3		3		5		3		4		
ARDUINO NANO	1	4	5	25	3	6	4	12	2	6	4	20	1	3	3	12	88
ARDUINO UNO	2	8	4	20	2	4	4	12	3	9	5	25	2	6	3	12	96
ARDUINO MEGA	3	12	2	10	1	2	3	9	3	9	5	25	3	9	3	12	88
RASPBERRY PI	4	16	2	10	1	2	2	6	4	12	2	10	4	12	1	4	72

Fig 15: Motherboard decision matrix

b. GPS choice

For the GPS, two different references that met our needs were found, but in the end the GPS Neo-6M was picked for its cheaper price.

CRITERIA	Size and Weight			Power Supply			Cost			Accuracy			TOTAL
FACTOR	2			1			2			4			
Grove 113020003	15x15x7mm / 1,8g	3	6	3-5V	2	2	28,80 €	2	4	9600 bauds	3	12	24
NEO-6M	16x16x7mm / 1,2g	3	6	3-5V	2	2	9,90 €	4	8	9600 bauds	3	12	28

Fig 16: GPS decision matrix

c. Servomotors choice

For the servo motors two different references were chosen among three: FT90M and FS90R, because a continuous motor is needed for the trapdoor and a step by step one for the parachute.

CRITERIA	Size and Weight			Power Supply			Cost			Torque			TOTAL
FACTOR	2			2			2			3			
D2S51	20x8,2x16,3mm / 2,5g	3	6	4-6V	1	2	11,20 €	1	4	0,8kg/cm	3	9	21
FS90R	23x12x28mm / 9g	2	4	4,8-6V	2	4	5,90 €	3	8	1,5kg/cm	3	9	25
FT90M	16x16x7mm / 13g	1	2	4,8-6V	2	4	8,80 €	3	8	0,7kg/cm	3	9	23

Fig 17: Servomotors decision matrix

d. SD Card reader

For the SD card reader, the only one available on the market that met our needs was chosen, so it is the SD GT126. It uses a full size SD card to record different values during the experiment.

7/ Electrical Assembly

a. Final Wiring plan

Below is the final wiring plan of the CanSat that contains all the components needed:

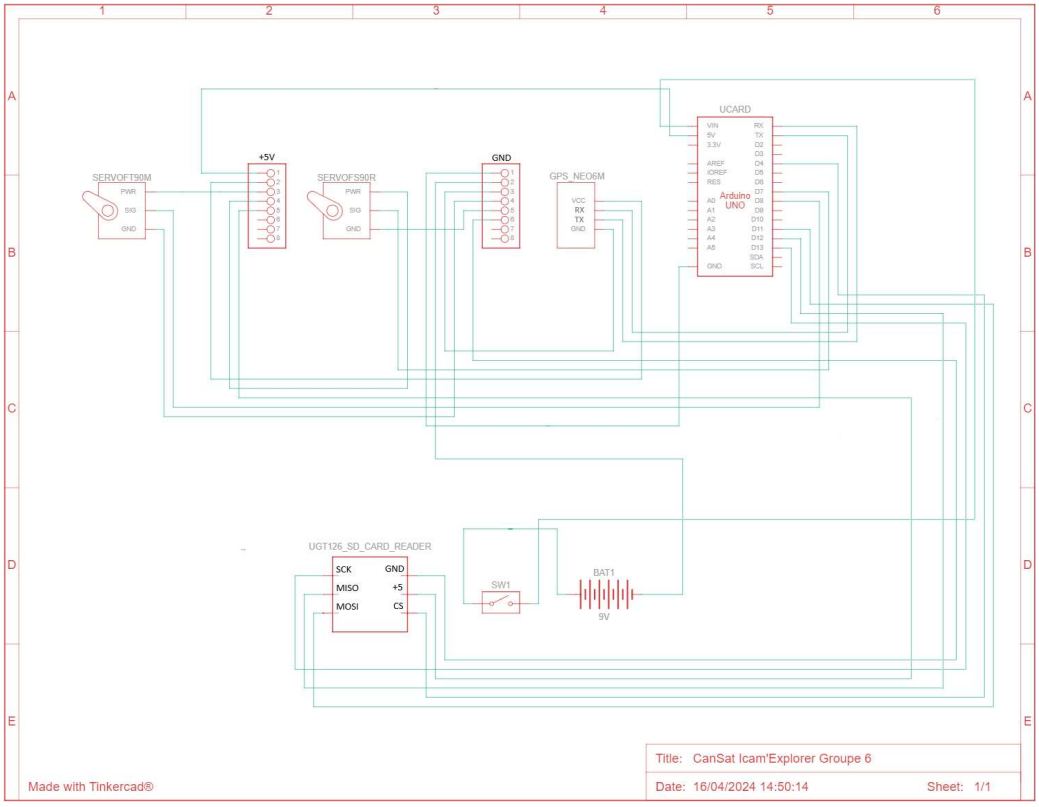


Fig 18: Final Wiring Plan

b. Power consumption

Arduino Uno:

Typical consumption: 100mA
Supply Voltage: 5V
Operation time for example: 60 minutes
Energy consumption: 0.1A*5V*1h = 0,5 Wh

GPS Neo-6M:

Typical consumption: 45mA
Supply Voltage: 5V
Operation time for example: 60 minutes
Energy consumption: 0.045A*5V*1h = 0,225 Wh

FS90R Servomotor:

Typical consumption: 5mA to 110mA (mean 57,5mA)

Supply Voltage: 5V
Operation time for example: 60 minutes
Energy consumption: 0.0575A*5V*1h = 0,2875 Wh

FT90M Servomotor:

Typical consumption: 5mA to 120mA (mean 62,5mA)
Supply Voltage: 5V
Operation time for example: 60 minutes
Energy consumption: 0.0625A*5V*1h = 0,3125 Wh

SD GT126:

Typical consumption: 5mA
Supply Voltage: 5V
Operation time for example: 60 minutes
Energy consumption: 0.005A*5V*1h = 0,025 Wh

So in total for 1 hour usage it needs 1,35Wh which is equal to **270mAh**.

The battery has 300mAh of capacity at 9V, so it can deliver up to 2,7Wh (0,3Ah*9V=2,7Wh) of energy. This allows with an efficiency of 90% a maximum of 1.8 hours of usage.

8/ Programme

a. General operating conditions

To steer the CanSat towards the target using the selected components (GPS, compass, accelerometer, etc.), it is needed to know the direction in which it needs to go, and the direction in which it's going..

For the targeted direction, calculation of a two-dimensional vector using the target coordinates (P1) and our live coordinates (P2) is used.

For the current direction, calculation of a second vector using the current and previous coordinates is needed.

These two vectors are used to calculate the angle of direction difference the CanSat currently has, which will be called α .

It's needed to steer the CanSat according to this angle, i.e. either to the left, straight ahead or to the right.

Once it's above the target (P1 = P2), it releases the marking solution.

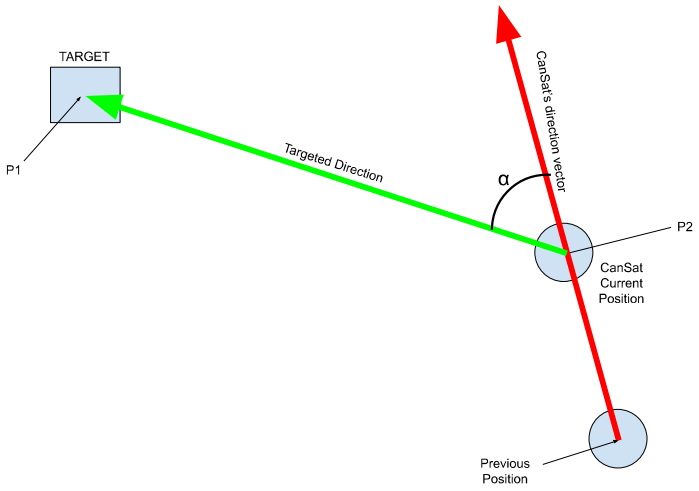


Fig 19: Algorithm illustration

b. Global architecture

Here's the overall program flow diagram. First, if the CanSat is not switched on, of course nothing happens. Once the CanSat is switched on, it will determine its direction of travel and its distance from the target. This determines whether it should go straight, left or right, and when it should mark the target's position.

This is just a simplified overview of how the CanSat works, and they'll go into more detail later on.

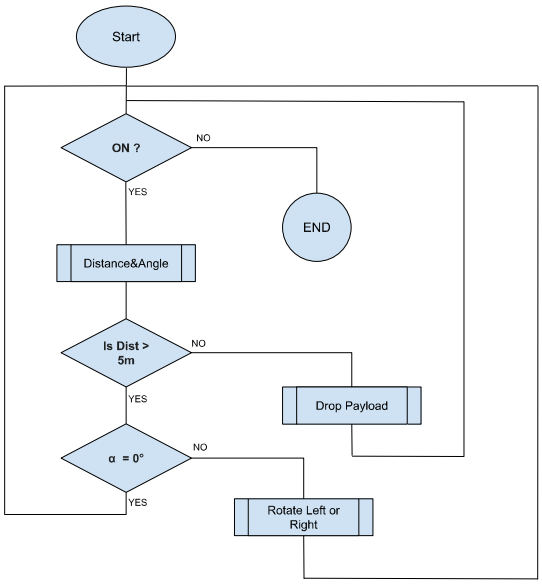


Fig 20 : Global Flowchart

c. Libraries used

To run this program, various libraries are needed:

TinyGPS++.h, which allows to manage the GPS in its entirety and provides functions for determining distances, directions...

Servo.h, which permits control of the servomotors, whether continuous or stepper.

Then the SD.h and SPI.h libraries, which are needed to use the SD card reader to open a file on the SD card and write on it.

d. Main function ReadGPS()

This main function, ReadGPS(), allows to record the GPS coordinates (current and previous) in order to determine the distance from the target, the direction towards the target, the direction and the difference (Heading-Bearing) of the two directions. It's also this function that writes to the SD card to record the movements during the experiment. This function calls two other void functions in orange, NeedToAdjust() and DropPayLoad() that will be looked at next.

Flowchart for this function on the next page:

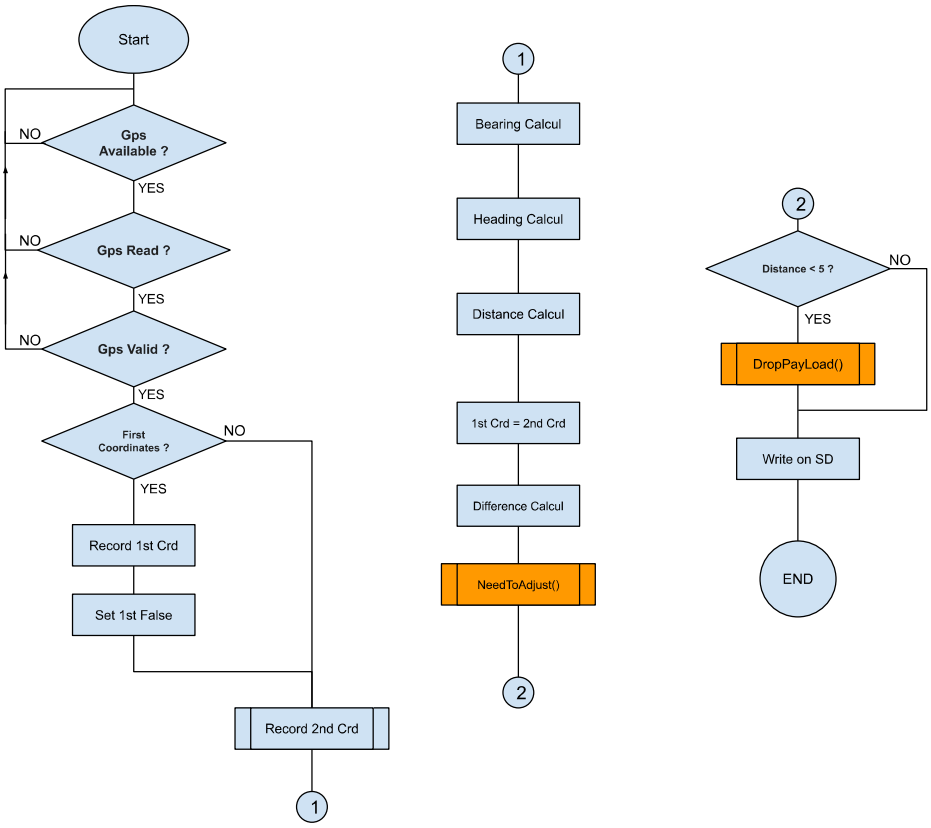


Fig 21 : ReadGPS() Flowchart

e. Function NeedToAdjust()

For the NeedToAdjust() function, it is needed to know the absolute value of the difference (Heading-Bearing). Then, if this absolute value is greater than 180, it needs to be normalized to be between 0 and 180(360 - abs(difference)). Then, if this absolute value is above the 5° angle limit, the SteerParachute() function (in orange) is called to turn the CanSat left or right. This function will be analyzed later on.
Flowchart of the NeedToAdjust() function::

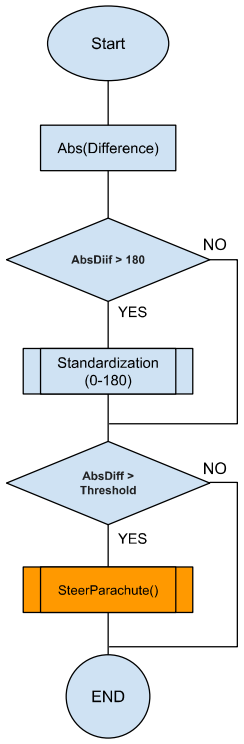


Fig 22 : NeedToAdjust() Flowchart

f. Function SteerParachute()

This function is very simple: if the Heading-Bearing difference is less than 0, you turn left; otherwise, you turn right. This function calls the two orange functions RotateServoLeft() and RotateServoRight(). They'll explain how these functions work in a moment.
Here's on the next page the flowchart for the SteerParachute() function:

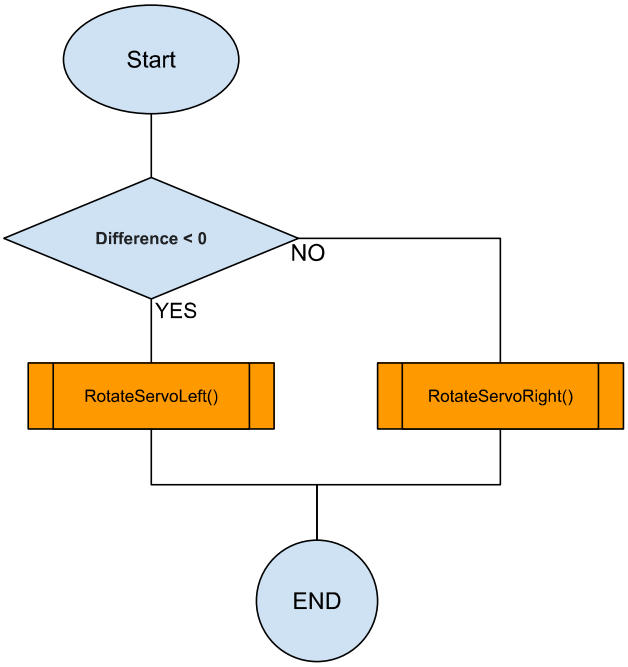


Fig 23 : SteerParachute() Flowchart

g. Function RotateServoLeft() and RotateServoRight()

In the functions RatateServoLeft() and RotateServoRight(), they firstly need to calculate the correspondence between the angle we need to turn the servomotor and the pulse wave size that they need to give to it.
Firstly they'll determine the angle needed to turn:
The servo motor can only turn by 45 degrees left or right in the configuration, so 180° in entry must be equal to 45° out. So the Angle in entry needs to be multiplied by 45/180 = 0.25 to have the correspondence.
Then it is needed to find the correspondence between degrees and milliseconds:
The neutral position for the servomotor is 140 degrees and this is equal to 1500 microseconds pulse, furthermore the minimum and maximum value is 1000 and 2000 microseconds, so 500 microseconds = 140 degrees. So they can determine the factor of multiplication with : 500/140 = 3.571.

So thanks to those calculations, the entry value of the function has to be multiply by 0.25*3.571= 0,89275 to have the corresponding pulse wave value that is needed to add or subtract from our neutral position to achieve the required angle.

This multiplication corresponds to the white box in the flowchart below.

Then the box in orange is the only one that is different between rotating left and rotating right:

-To turn left they subtract the value calculated before from the neutral position.

-To turn right they add the value calculated before to the neutral position.

Then they turn to this new servo position that has been calculated.
Finally they wait 2 seconds for the CanSat to turn and we go back to the neutral position.

Here the flowchart of the RotateServoLeft() and RotateServoRight() functions:

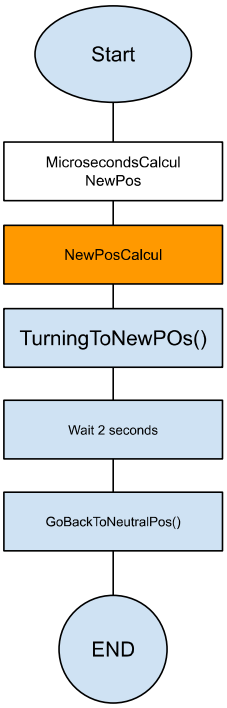


Fig 24 : RotateServoX() Flowchart

9/ 3D Modelling & First Design

A. First prototype

For the first prototype, many solutions were adopted to completely respond to the missions they had decided to do: Marking the target (main mission) and standing upright (secondary mission).

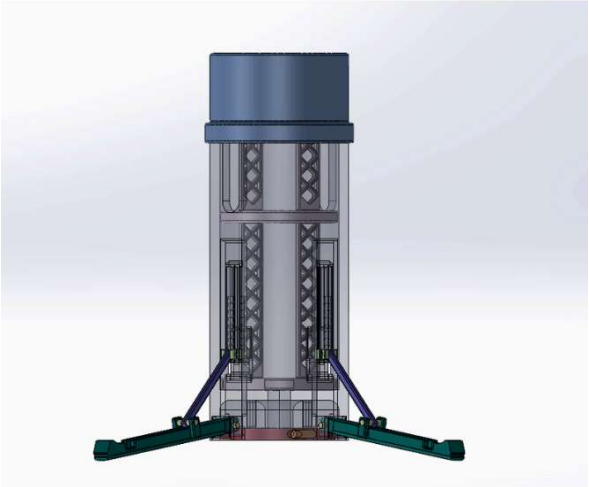


Fig 25: SolidWorks CAD CanSat

To fulfill the principal mission, a storage zone underneath the body is placed in order to keep our marking solution.
The storage is placed below to make the marking part and to not have to move the CanSat while it is in the air. The entire body stays straight up during the falling part of the air drop.

The locking system uses two dots put on the interior surface of the CanSat's head. For locking the head in the CanSat's body, we only need to slide the dots into the rails and follow the trajectory.

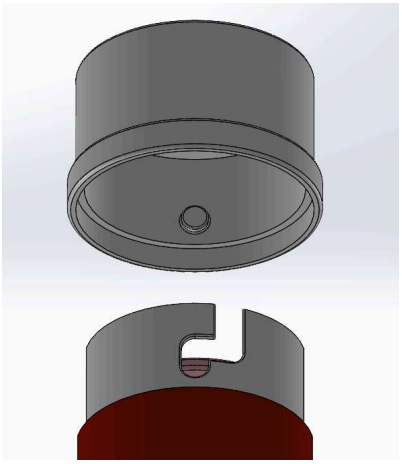
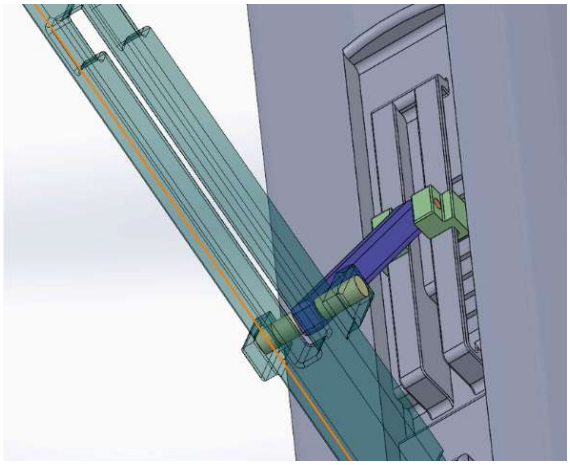


Fig 26: Locking system



For our secondary mission (stand up at the landing point), we designed feet that would use stopping racks. Connected to the main body, these feet were supposed to drop all the way down thanks to gravity and when the first contact with the ground was made, they would be going up and stopped by the rack.

Fig 27: Rack system

Since the third sprint (in December) was carried out with Mr.CHABROL, the first prototype was printed thanks to ICAM Strasbourg-Europe's Fablab. The main pieces, as the body and the head, were printed with little imperfections but functional. The head's locking system used was perfectly working and will be used for the next CanSat's iteration. It is, after the printing that they see that the rack solution (for the secondary mission) was not adapted to this project due to the dimensions of the CanSat. The racks were way too small to efficiently stop the feet.
The following pictures are some of the successfully printed component of the CanSat :

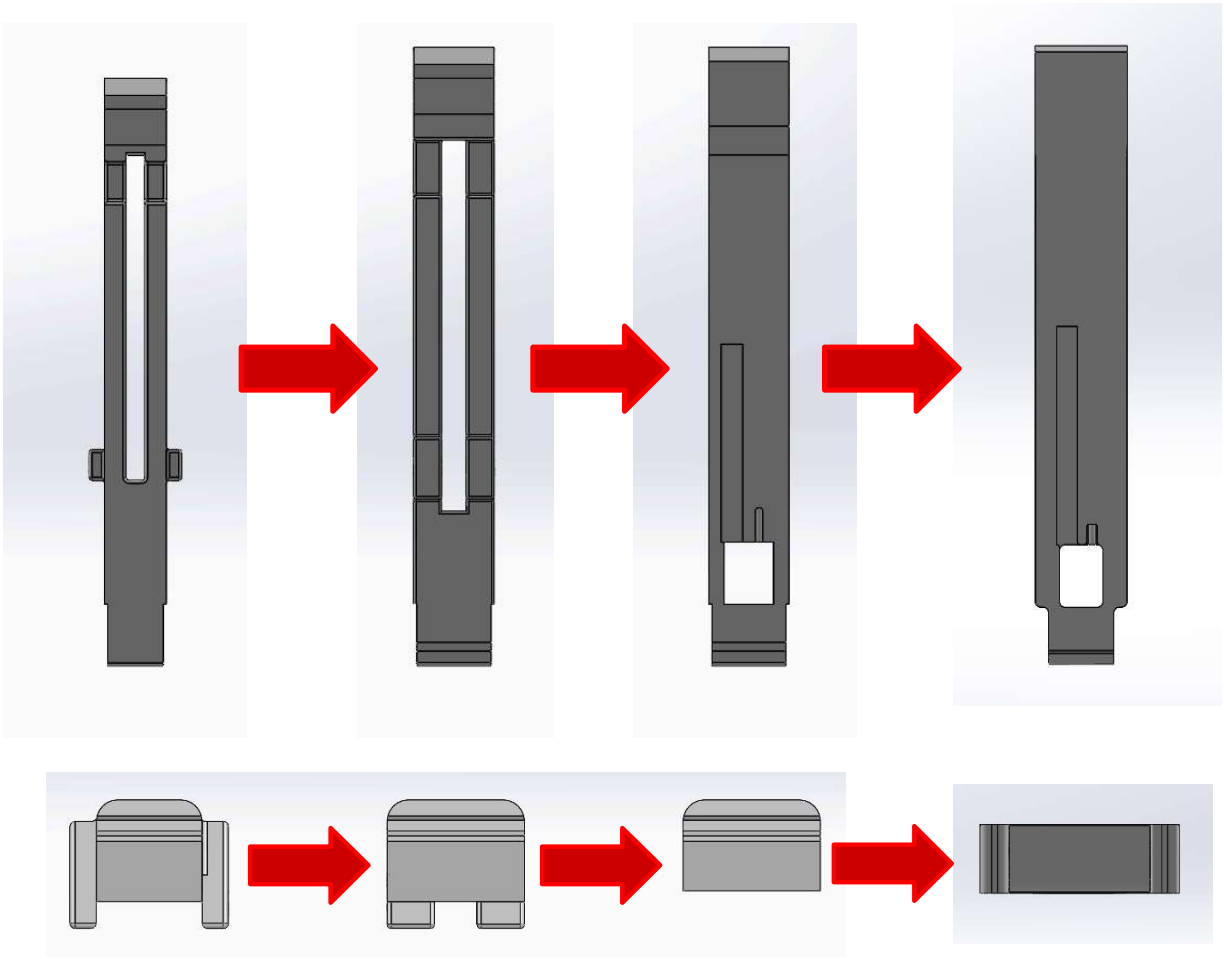
Fig 28: Printed prototype body, head and foot

However, during and after the impression, many issues were encountered. Three of four feet were not printed well and other pieces like the connector arm (used to link our feet to the rack) were not strong enough and broke too easily.

Fig 29: Unusable printed parts

B. Improvement made

The main issue with the previous version of the CanSat was the dimensions of the feet, leading to failed impressions. So, at first, they had to re-dimension the CanSat's feet to avoid this kind of problems on the next impressions.
Furthermore, thanks to Planet Science's representant advice following the meeting they booked in December, a lot of new leads were explored and will be in practice in the next printing iteration.

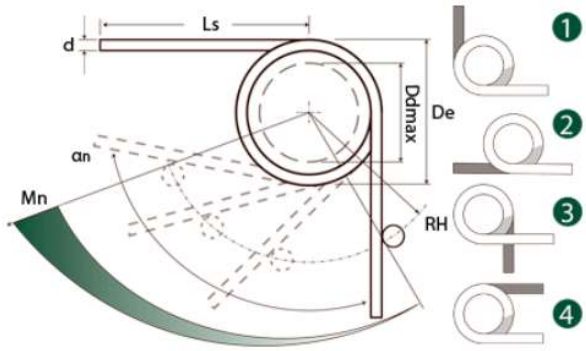


Prototype's version Sprint 2 version Sprint 4 version Latest version

Fig 30: Feet evolution

This latest version of our feet are 5.26% much longer than the Sprint 4 ones.
In addition, of these new feet, a completely different solution to make the CanSat stand up at the landing point was discussed with Planet science in comparison with the Prototype and Sprint 2 version. The CanSat's body had to be redesigned. The purpose of this change was to use torsion springs instead of a mechanism of racks.

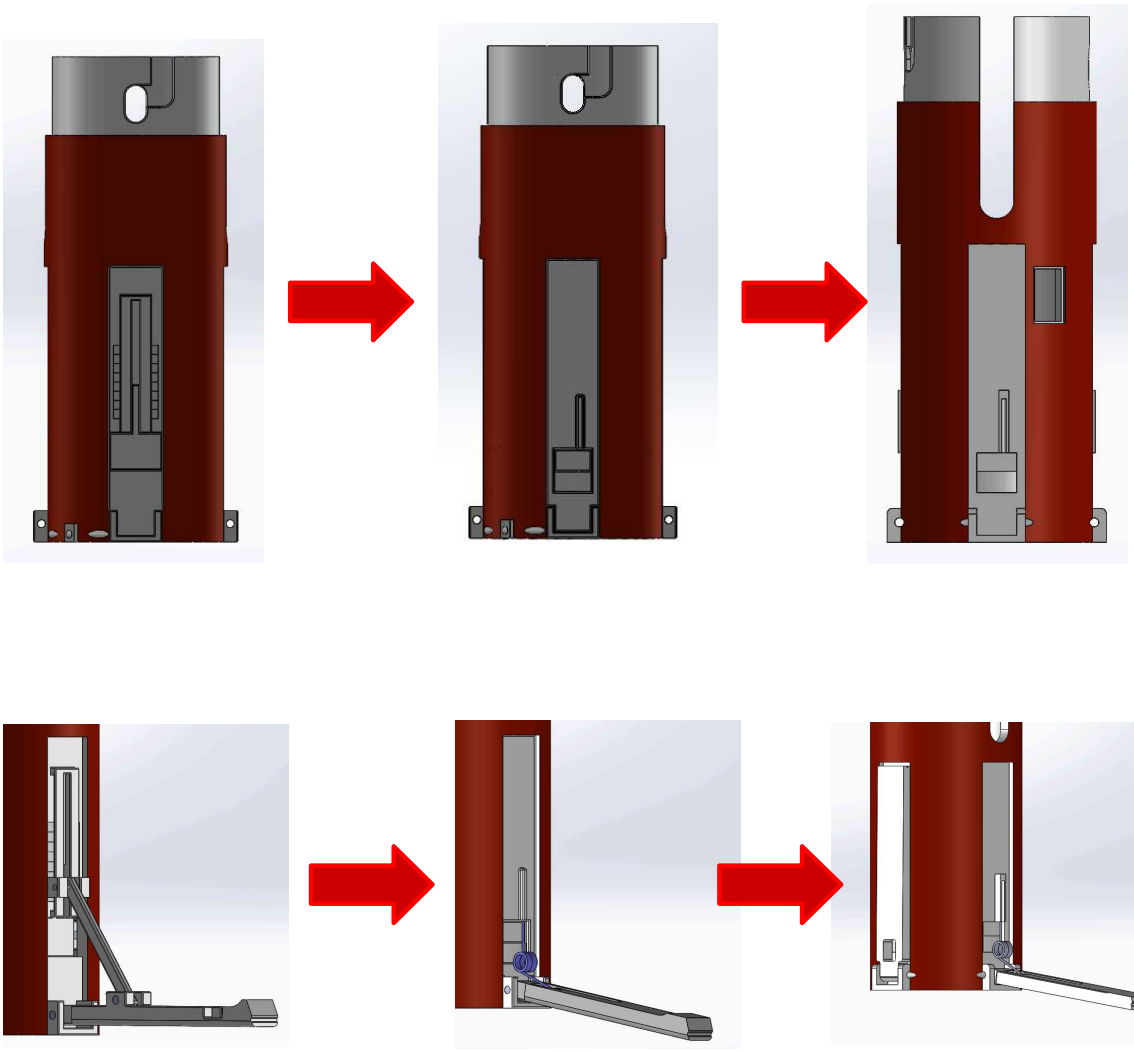
This new system of torsion springs prevents rack's failures and saves raw materials, because they don't need the racks and the connector arms anymore.
The torsion spring they want to use are T035-180-187R torsion spring, which have the following specifications :



Matériau	Acier inoxydable 302
d - Diamètre du fil (mm)	0.89
De - Diamètre extérieur (mm)	7.70
Di - Diamètre intérieur (mm)	5.92
Ddmax - Arbre max. (mm)	4.75
n - Spires élastiques (Quantité)	6.00
Ls1 - Longueur de pied (mm)	31.80
αn - Pivotement max. (degrés)	180.00
Position du pied	2.00
Mn - Couple max. (N mm)	113.00

Fig 31: Torsion spring characteristics

This is, in the following pictures, the solution agreed to go on with, thanks to all the Mr.CHABROL and Mr.LE BARBENCHON :



Prototype's version | Sprint 4 version | Final version

Fig 32: Standing up solution evolution

The changes made on the body aimed to: save raw materials by opening the spring's sliding parts. There is also a new hole on the side of the CanSat to be able to put a power switch facing the exterior environment.

The last improvement on the old version of the ICAM'Explorer CanSat was the bottom trap which is used to store the solution to mark the target. The previous trap used two separate doors to open and close the storage zone, thanks to a servo-motor, but, after the meeting with Planet Science they agreed to use only one door, that can be opened and closed thanks to a servo motor linked with a cable.

The "ancient" two doors system:

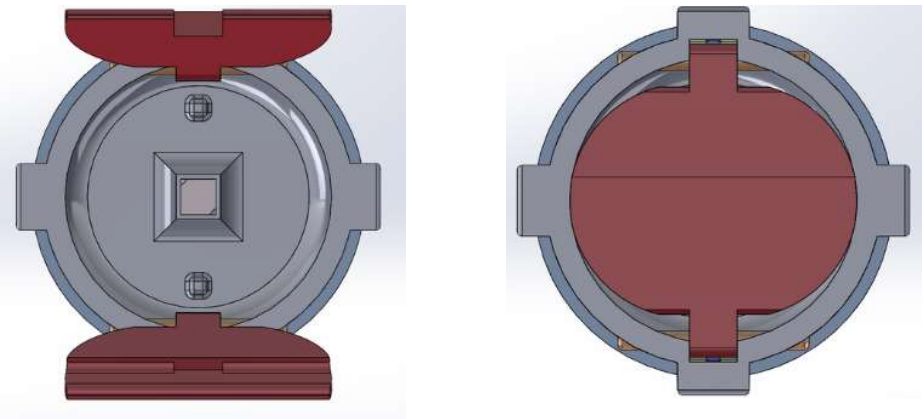


Fig 33: Two doors system

The "new" one door system:

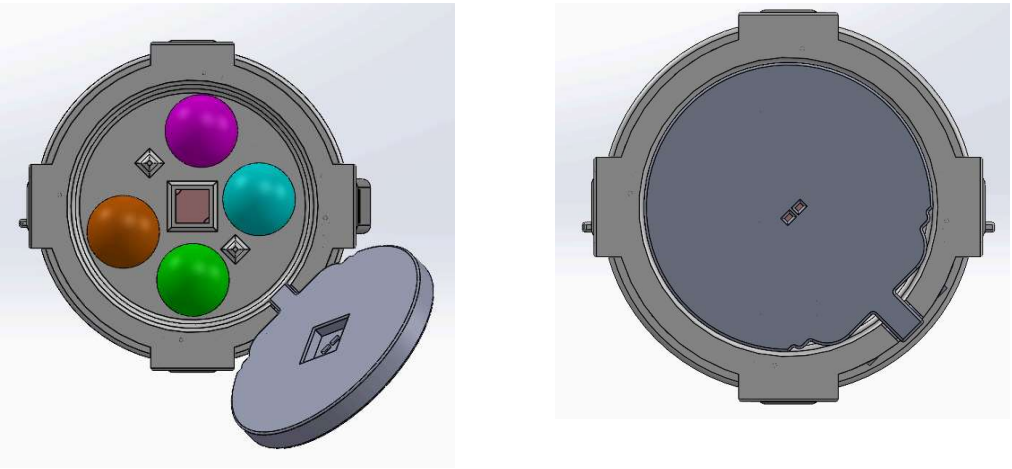


Fig 34: Sprint 4 one door system

The door is put diagonally so as not to disturb the feet in their movement like did the previous two door system.

During the Sprint 4, Mr.CHABROL commented that this one-door system will not be stable when the CanSat is on the point of landing. So, the new improvement made on this system was to add 2 more 'stabilizing' pillars to be sure that when the trap will be closed, the door does not move.

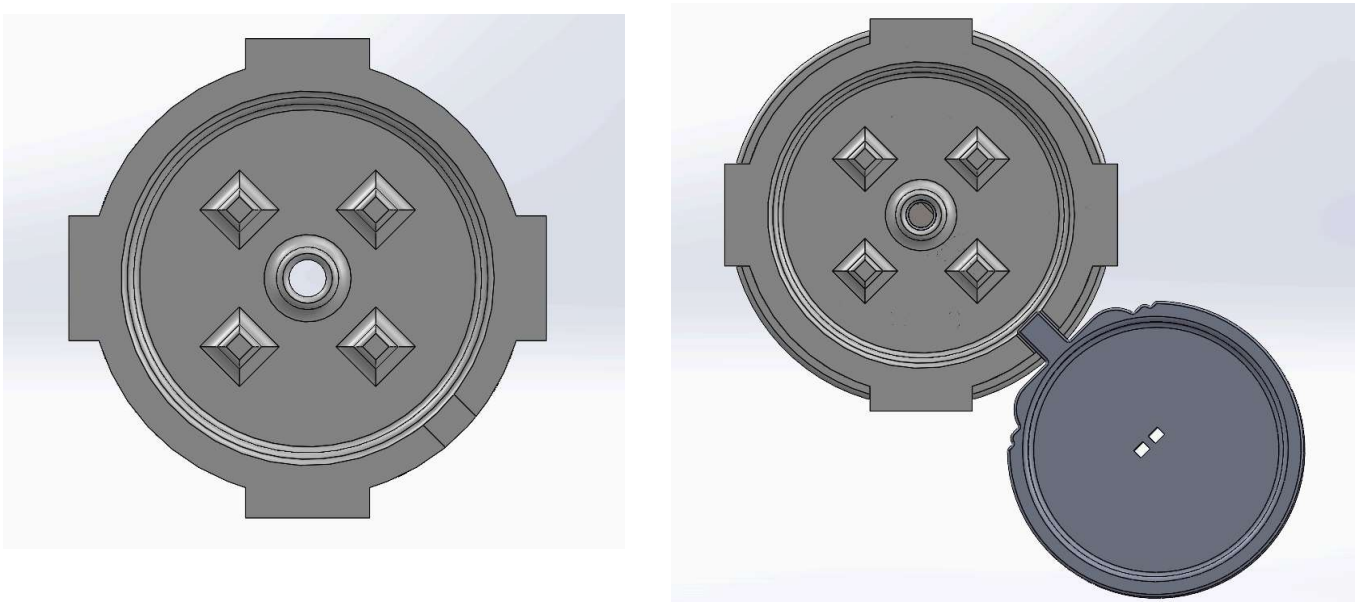


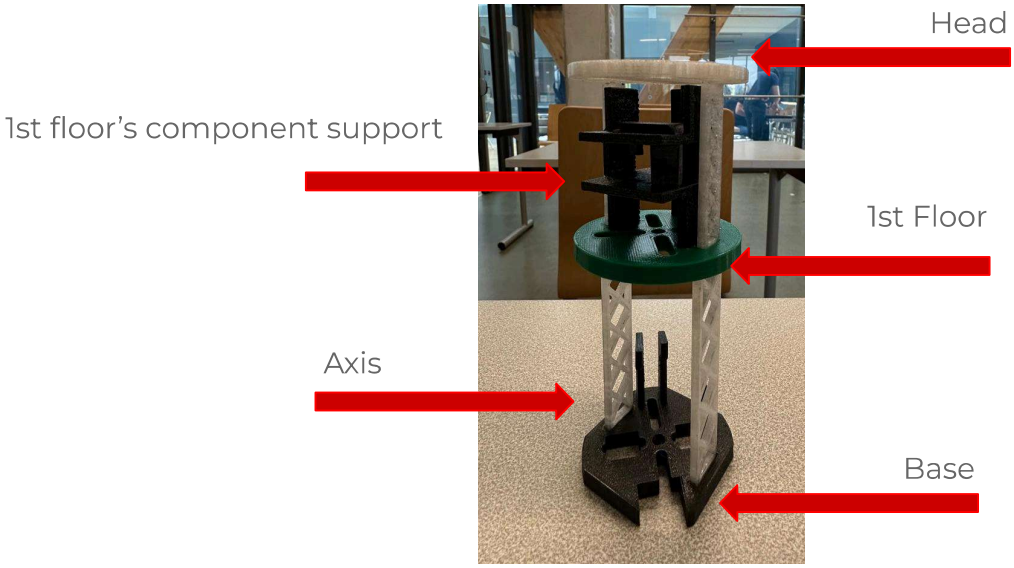
Fig 35: One door system final version

C. Final design

With all the previous presented improvements, there is the final CanSat design. The secondary mission (standing upright) is still relevant today. There are all the PETG plastic , used pieces without all electrical components.

Fig 36: CanSat's PETG printed parts

The following photos, provide a look at the CanSat when his fully assembled



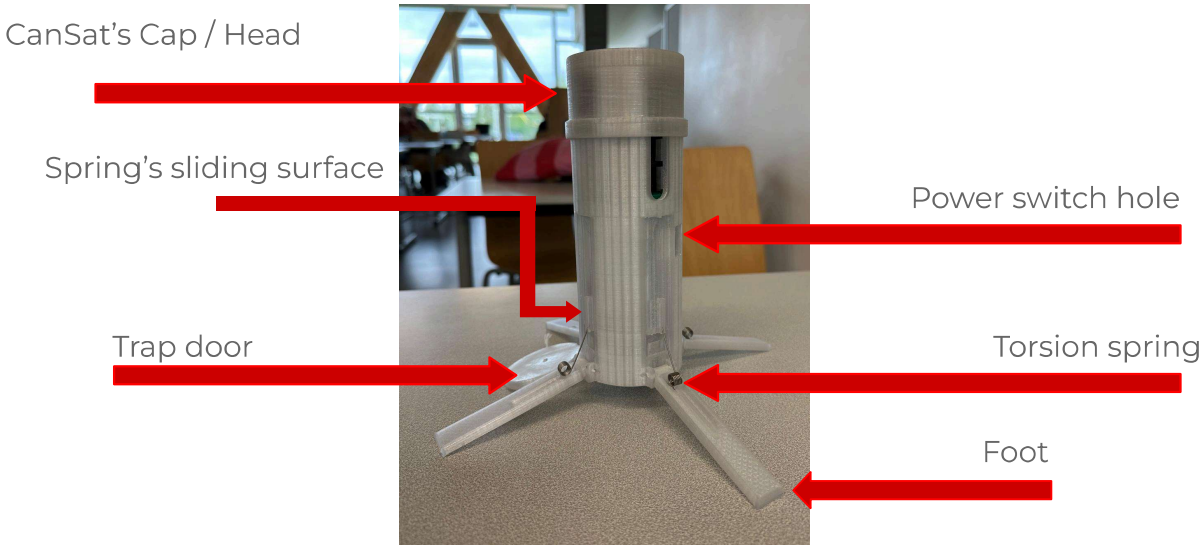


Fig 37 : Support axis and CanSat composition

The following photo shows how all the electrical components are arranged, within the component support designed to easily fit in the CanSat body. The main goal of these pieces was to have a precisely defined and separated slots for each electric component within the CanSat.

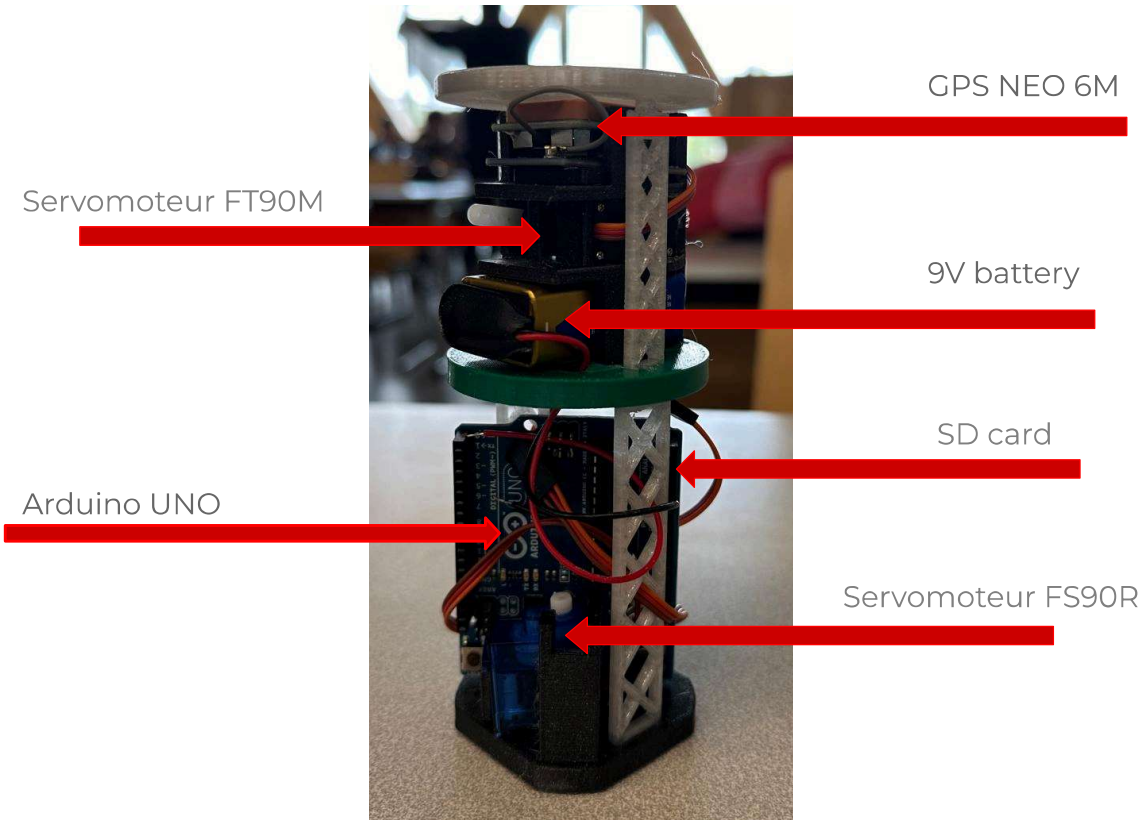


Fig 38: Final design internal composition

10/ Environmental aspect

The Icam'Explorer team has placed particular importance in considering the environmental aspect throughout the realization of the project. Indeed, in the current context, it seemed essential to them to adjust their actions accordingly.

Regarding the design of the CanSat, the team used PETG for the 3D printing, and here are the two main reasons for this choice:

- **Recyclability:** PETG is a recyclable material that can be processed through conventional recycling streams. This means that once the life cycle of the printed object is over, the material can be recycled to produce new plastic products, reducing waste and the need for virgin resources.
- **Durability and Longevity:** PETG is known for its strength and durability, which means products made from PETG are likely to last longer and not require frequent replacement. By creating long-lasting components, the environmental impact is reduced as the rate of consumption and waste generation decreases. The extended lifespan of PETG parts ensures resources are utilized more efficiently, contributing to sustainability goals.

Regarding the components of the Cansat, the team first asked the school's teachers if they had the desired components available so they could be reused and avoid purchasing new ones. This not only helps to reduce the costs of our project but also to limit the environmental impact.

Regarding the balls that we use to mark the position, we selected biodegradable pellets whose components are environmentally friendly.

11/ Budget

There is a table summing up all the components that were used for the project.

Components	Cost (unit)
Arduino Uno	23.9€
Module GPS	6,50€
Servomotor FS90R	5.90€
MPU 5060	3.90€
Battery 9V	3.90€
Parachute sail (x2)	20€
nylon threads (5m)	5,99€
Torsion spring T035-180-187R	4 * 5.48€
Split pin 3.2mm x 32mm (DIN 94)	5 * 0.92€
Spray glue	11,90€
Spray paint	3,90€
Beads balls	7,99€
FT90R	8,90€
SD GT126	2,10€
TOTAL	**131,40€**

Fig 39: Table of purchase

12/ Conclusion

Since the last report, concrete progress has been made. Despite falling behind in certain areas halfway through the project, ICAM'Explorer have been able to rectify the situation and catch up, particularly on electronic components implementation. To do this, a focus has been placed on investment in the technical and material aspects. These advances are due to the Sprints carried out with Mr. CHABROL and his advice, but also to the meetings with PlanetScience experts like Mr.LE BARBENCHON. In concrete terms, once the decision had been taken on the modifications to be made, they were quickly implemented.

Progress was continuous, with each area making progress on their final solutions as the weeks went by. As everyone had a predefined role and mission, close communication with each member of the team was kept, as some improvements had an impact on another area. This meant that before each Sprint, the entire team would hold a meeting to bring together any progress made and show where it was going.

The parachute has been modified several times in response to requests. Whether in terms of stability, to mark the target correctly, or in terms of falling speed, which had to be reduced to avoid crashing and landing straight. Many dropping tests were made but they were irrelevant concerning the entire project because their only purposes were to test out the capabilities of the parachute.

The Arduino's code development for the CanSat project was an enlightening experience, particularly in understanding and integrating diverse technologies. While the fundamental concepts behind the functionalities—like GPS integration and data logging to an SD card—were straightforward and quickly grasped, the real challenge lay in harmoniously combining these elements into a single, cohesive codebase. This task was further complicated by unexpected bugs and interferences arising from conflicts between different libraries, which required meticulous debugging and adaptation.

As far as the CAD solution is concerned, further improvements have been made compared with previous prototypes, particularly with regard to the standing upright mission, component implementation and final design. Thanks to all the advice of the different stakeholders, any ICAM'Explorer defective CanSat parts can be easily replaced, by changing the defective piece itself (a foot or the door trap) or by changing the torsion spring if necessary.

The objectives for the rest of the remaining time that ICAM'Explorer's team have are:

- Test in real conditions, the entire and final CanSat (parachute + CanSat + electronic components)
- Complete the User Guide due for the meeting with the client, Mr.CHABROL.
- Create the project's poster for the stand that ICAM'Explorer CanSat will hold next week.

12/ Result of the flight

Unfortunately, due to an electronic problem, we didn't obtain data from the GPS but the flight was almost a success. We had a nominal flight and the Cansat landed straight. There was also too much wind during the official flight and our CanSat wasn't able to reach the target. The system to mark the target with paint was efficient but due to the wind and the weight of the ball soaked by ink, when the system opened the ball wasn't able to reach the target as expected.

If we had to fly our CanSat again, there would be some parameters to modify. We should improve the marking system so that it's ready for action whatever the weather is. We should also do a chimney on the parachute to stabilize the descent. Finally, we should also find a way to prevent our electrical problem.

Overall, the flight was a success, despite a few minor problems. This flight was a reward for a year's work, and we've grown from it by understanding our mistakes so that we don't make them again on our next projects.

13/ Appendix

Planete Science general rule validation

Generale rule validation		
Rule	Description	Validation
Rule 01	Each team is made up of a minimum of two people. There is no limit to the number of participants. A person cannot be in two different teams at the same time."	
Rule 02	At least three-quarters of the team members must be students at the time of registration. Teams that do not meet this criterion may submit an application, which will be examined by the organization.	
Rule 03	If a document is returned late, a penalty of 1 point per day is applied.	
Rule 04	The CanSat has five minutes after opening the release hatch to complete the missions.	
Rule 05	The maximum weight of the CanSat is 1kg.	
Rule 06	In the release, all CanSat components, with the exception of the parachute must fit within a basic volume 80mm in diameter and 200mm high.	
Rule 07	Deployable elements must be deployed beyond the basic volume at the release end only	
Rule 08	Rotating or sharp components must be protected by fairings.	
Rule 09	Hazardous materials and pyrotechnics are prohibited.	
Rule 10	Pneumatic systems are limited to 10 bar.	
Rule 11	The maximum electrical voltage difference (Vmax-Vmin) is limited to 30 V.	
Rule 12	The CanSat must have an autonomy of 45 min.	
Rule 13	Lithium batteries are permitted if they are not modified, if they are charged are charged, transported and stored in fireproof bags, and if their use complies with the rules set out in the following document: "Case of lithium-based batteries".	
Rule 14	The use of resistive wire is authorized if its heating capacity is not sufficient to ignite a sheet of paper. It will be tested during the checks to guarantee safe conditions during release.	
Rule 15	The power switch must be accessible from outside the CanSat.	
Rule 16	The CanSat and all components ejected from it must descend to the ground at a speed of between 2m/s and 15m/s.	
Rule 17	The recovery chain must be capable of withstanding a force of 50 N.	
Rule 18	Telemetry systems must comply with French regulations.	
Rule 19	The telemetry system switch must be accessible from outside the CanSat.	
Rule 20	Each team can attach a module to the drone. The weight of this module must be less than 380g. Specifications on the volume and shape of the drone's carrying case will be provided on request.	
Rule 21	The release altitude is between 80 and 120m above ground level.	
Rule 22	Speed is almost zero at the moment of release.	
Rule 23	The drops take place in wind conditions of less than 5m/s.	

Validation		
OK	NOT OK	ONGOING OR NOT DONE

Fig 40: Rules validation

Organizational part:

TASK ID	TASK	MISTON Clara	MACK Théo	MAZOYER Clément	BELAMY Romain	HALMI Mohamed	SAINT-CHARLES Alexandre
		LEADER	PRODUCT OWNER	SCRUM MASTER	TECHNICIAN		
1.1	Design a parachute capable of reducing the CanSat's fall speed	I	R	A	R	C	C
1.2	Design a resistant and efficient parachute	I	R	A	R	C	C
2.1	Design a system which deploys the parachute on release	I	R	A	R	C	A
2.2	Automate parachute deployment	C	R	A	R	C	A
3.1	Coding a remotely controllable on-board system	C	C	R	A	C	C
3.2	The information system must have a minimum range of 150m	C	C	R	A	A	C
3.3	The CanSat must have an autonomy of at least 45min	C	R	C	C	R	A
4.1	Make a system able to mark the position where the CanSat land	C	C	C	R	R	A
4.2	Design a solution which activates automatically on landing	C	A	R	A	C	A
5.1	Design a solution which allows the CanSat to stand upright after landing	C	C	A	C	C	R
6.1	Define roles on the team	R	I	A	C	C	C
6.2	Build each sprint efficiently and then discuss possible improvements	R	A	R	C	C	C
6.3	Organize regular meeting With the team to follow the advancement	R	A	A	C	C	C
7.1	Return all delivrables on time	R	I	A	A	A	A
8.1	Respect the dimensions of the CanSat described on the regulations	A	A	C	C	C	R
8.2	Respect technical data for internal components described in the regulations	C	R	A	R	C	C
9.1	Create an original design to stand out from others groups	I	C	C	C	C	R
9.2	Create an attractive design to draw attention to the project	I	C	C	C	C	R
10.1	Create a CanSat easy to transport and store in a box	I	A	C	A	C	R
11.1	Design the CanSat to fit all components inside	C	A	C	A	C	R
12.1	The CanSat must not exceed the dimensions authorized in the regulations	A	A	C	C	C	R
13.1	Design a CanSat With reusable mechanisms at least 50 times	R	R	C	C	C	A
14.1	Use low carbon material for the prototype design	R	C	C	C	R	A

Fig 41: RACI matrix

Inrteractors description for the Octopus diagram:

FUNCTIONS	TITLE	CARACTERISTICS	CRITERIA	LEVELS	FLEXIBILITY	TASK ID
FP1	Mark a targert within a disgnated perimeter.	Efficiency	Size of the target.	200*200 mm	F1	1.1
			Time to mark the target.	5 min	F1	1.2
		Localisation	Get the CanSat localisation in real time	GPS NEO 6M	F0	1.3
		Autonomy	Sefi-sufficiency.	45 minutes minimum	F1	1.4
		Acceleration	Get the speed of the CanSat in real time .	in m.s^-2 by the accelerometre MPU-6050	F3	1.5
FC1	Be compatible with the transportation device carried by the drone.	Ergonomy	Weight.	1 kg maximum	F0	2.1
			Volume	1 litre	F0	2.2
			Size	200*80 mm	F0	2.3
FC2	Land softelly thanks to a parachute.	Security	Vertical Landing with a speed that must speed must be controlled	>= 10km/h	F1	3.1
FC3	Respect for the environment.	Global	Pollution Lifetime Recycling Energy consumption	No pollutant emissions 100% dismountable	F1	4.1
FC4	Adapt to different ground types.	Stability	Stand up straight while landing	90° angle	F3	5.1
FC5	On board sensors	Temperature	Get the outside temperature in real time	in °C by the accelerometre MPU-6050	F3	6.1
FC6	A CanSat reusable	Landing Speed	At the landing point, the CanSat must have a maximum speed	<= 5 m,s^-2	F1	7.1
		Dismantable	Can be dismantled at anytime		F1	7.2
		Water and Dust resistant	Water and dust must have minimal impact on the external and internal structure of the CanSat	Certification IP 31	F2	7.3

Fig 42: Interactors description

Theoretical formula for the parachute size calculation:

m < 1 kg Acceleration due to the gravity g = 9.81 m/s2

$$A = \frac{m \times g}{cd \times \rho \times V}$$

Drag Ratio 1,5 < Cd < 2,5 Air Density ρ = 1.225 kg/m3 Falling Speed 2 m/s < V < 15 m/s

Risk analysis diagram:

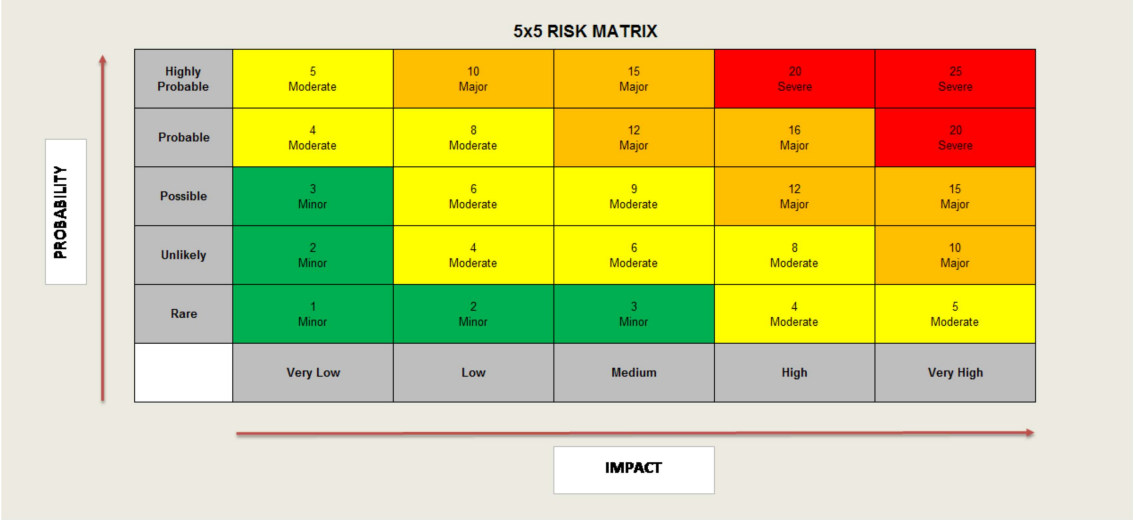


Fig 43: 5*5 risks analysis diagram

Design Risks:

- Component Failure: (Probable, Major) - Cell 12
- Power Supply Failure: Inadequate power management could result in mission failure: : (Possible, Major) - Cell 9
- Communication System Failure: Loss of signal or interference can prevent data transmission: (Probable, Severe) - Cell 16

Launch Risks:

- Deployment Failure: The CanSat could fail to deploy from the launch vehicle: (Possible, Severe) - Cell 15
- Collision: The CanSat might collide with other payloads or launch vehicle parts: (Unlikely, Severe) - Cell 14

Operational Risks:

- Data Loss: Data collected might be lost due to transmission errors or onboard storage failure: (Probable, Major) - Cell 12
- Battery Depletion: The battery could run out before mission completion: (Possible, Major) - Cell 9
- Overheating: Components may overheat during operation, leading to system failure: (Possible, Major) - Cell 9
- Recovery Issues: The CanSat may not be recoverable after its mission due to landing in inaccessible terrain or malfunction of the recovery system: : (Possible, Moderate) - Cell 6

Environmental Risks:

- Weather Conditions: Poor weather could delay the launch or affect the CanSat's operation: (Probable, Moderate) - Cell 8

Final Program within the Arduino UNO:

```
#include <TinyGPS++.h>
#include <Servo.h>
#include <SPI.h>
#include <SD.h>

static const uint32_t GPSBaud = 9600;

//SD Card variables:
File myFile;
const int chipSelect = 4;

// The TinyGPS++ object
TinyGPSPlus gps;

//Servo Objects
Servo ServoTrap;
Servo ServoParachute;

// The serial connection to the GPS device
//SoftwareSerial ss(4, 3);


//-----------------------------------------------------------------!!!!
//Enter tha TARGET Coordinates here:
float TgtLat = 0.0;
float TgtLng = 0.0;
//-----------------------------------------------------------------!!!!


//Bearing Coordinates and variables:
```

```
48
49  //Time before trap closure:
50  unsigned long Timer = 30000;
51
52  //Payload:
53  bool Oppened = false;
54  bool Set = false;
55
56  void setup() {
57    //Serial1.begin(9600);
58    Serial.begin(9600);
59    while (!Serial) {
60      ; // wait for serial port to connect. Needed for native USB port only
61    }
62
63    Serial.print("Initializing SD card...");
64    if (!SD.begin(4)) {
65      Serial.println("initialization failed!");
66      while (1);
67    }
68    Serial.println("initialization done.");
69
70    //myFile = SD.open("TEST.TXT", FILE_WRITE);
71
72    pinMode(ServoParachutePin, OUTPUT);
73    pinMode(ServoTrapPin, OUTPUT);
74    pinMode(11, OUTPUT);
75
76    //Initialisation Parachute:
77    ServoParachute.attach(ServoParachutePin);
78    ServoTrap.attach(ServoTrapPin);
79    ServoTrap.write(90);
80    ServoParachute.writeMicroseconds(pos);
81
82    digitalWrite(11, HIGH);
83
84  }
85
```

```
86  // fonction that reads the GPS data:
87  void ReadGPS() {
88    if (!Serial.available()) {
89      Serial.println("GPS not Available");
90    }
91    while (Serial.available() > 0){
92      if (gps.encode(Serial.read())){
93        if (gps.location.isValid()){
94          if(!FirstCoord){
95            Lat1 = gps.location.lat();
96            Lng1 = gps.location.lng();
97            Serial.println("Latitude 1: " + String(Lat1));
98            Serial.println("Longitude 1: " + String(Lng1));
99            FirstCoord = true;
100         }
101         //Here we wait 10 seconds before taking the 2nd coordinates:
102         delay(3000);
103
104         //Then we take the 2nd Coordinates:
105         Lat2 = gps.location.lat();
106         Lng2 = gps.location.lng();
107         Serial.println("Latitude 2: " + String(Lat2));
108         Serial.println("Longitude 2: " + String(Lng2));
109         //Distance, Heading, Bearing calculations:
110         //Bearing:
111         Bearing = gps.courseTo(Lat2, Lng2, TgtLat, TgtLng);
112         Serial.println("Target's bearing: " + String(Bearing));
113
114         //Heading:
115         Heading = gps.courseTo(Lat1, Lng1, Lat2, Lng2);
116         Serial.print("CanSat's Heading: " + String(Heading));
117         Serial.println(" °");
118
119         //Distance:
120         Distance = gps.distanceBetween(Lat1, Lng1, TgtLat, TgtLng);
121         Serial.print("Distance from target: " + String(Distance));
122         Serial.println(" m");
123
124         //New 1st coordinates
125         Lat1 = Lat2;
126         Lng1 = Lng2;
127
128         Difference = (Heading - Bearing);
129         NeedToAdjust(Difference);
130         if ((Distance < 5) && (Oppened == false)) {
131           Serial.println("Dropping Payload !!");
132           DropPayload();
133           Oppened = true;
134         }
```

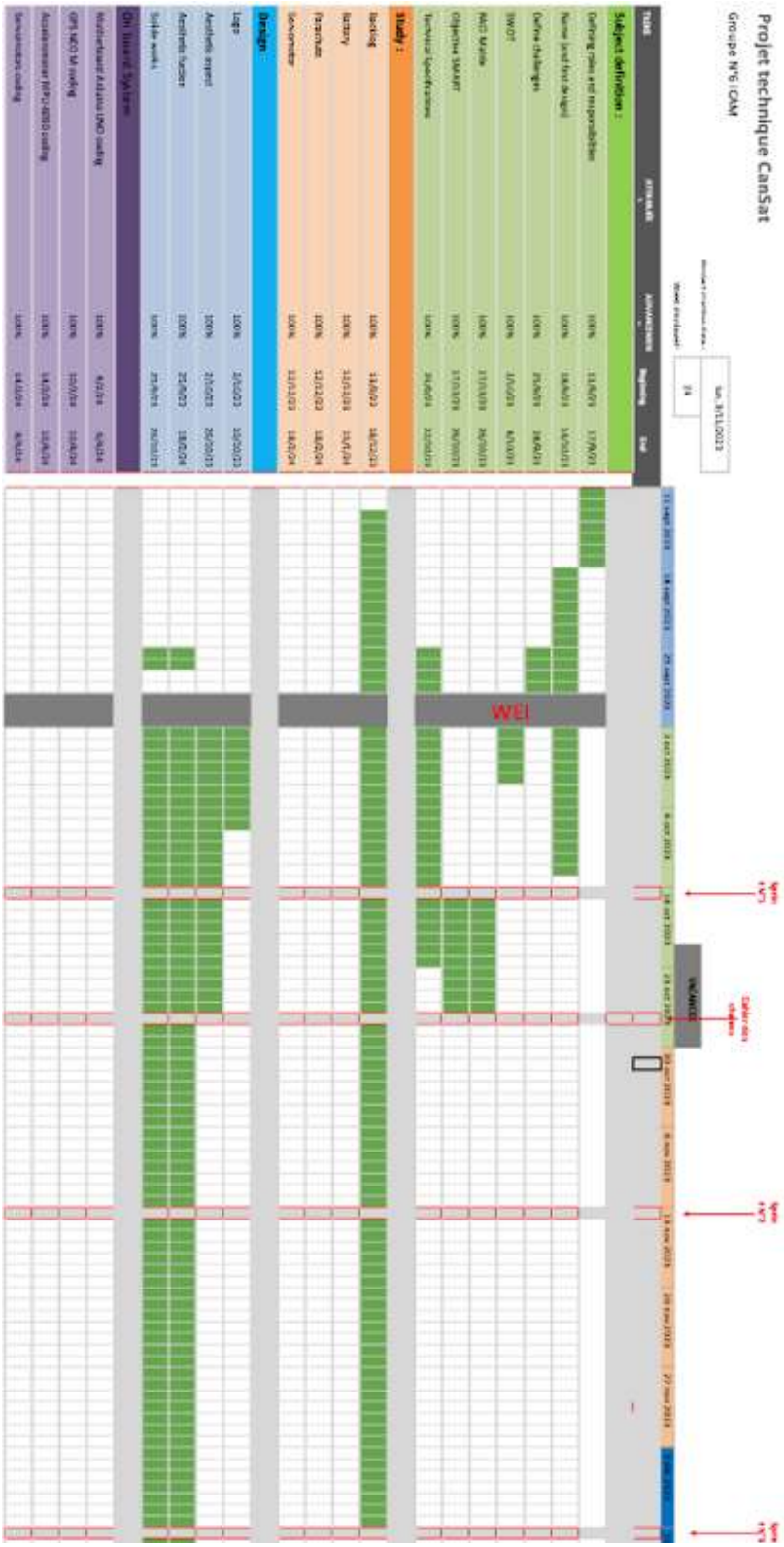
```
135           myFile = SD.open("TEST.TXT", FILE_WRITE);
136           delay(1000);
137           if (myFile) {
138             Serial.print("Writing to test.txt...");
139             myFile.print("Current Coordinates:" + String(Lat2));
140             myFile.print(" N");
141             myFile.print(" | " + String(Lng2));
142             myFile.print(" E");
143             myFile.print("Heading: " + String(Heading));
144             myFile.println(" °");
145             myFile.print("Bearing: " + String(Bearing));
146             myFile.println(" °");
147             myFile.print("Distance: " + String(Distance));
148             myFile.println(" m");
149             myFile.print("Difference" + String(Difference));
150             myFile.println(" °");
151             myFile.print("Turning Angle: " + String(Angle));
152             myFile.println(" °");
153             myFile.close();
154             Serial.println("SD Done.");
155           } else {
156             // if the file didn't open, print an error:
157             Serial.println("error opening test.txt");
158           }
159         }
160       }
161     }
162   }
163
164   void NeedToAdjust(float Difference){
165     float ADifference = abs(Difference);
166     if (ADifference > 180){
167       ADifference = 360 - ADifference;
168     }
169     if (ADifference > AdjustmentThreshold) {
170       Serial.println("Need Adjustments...");
171       SteerParachute(Difference, ADifference);
172     }
173     else {
174       NeedAdjust = false;
175       Serial.println("Heading's Good");
176     }
177   }
```

```
178
179 void SteerParachute(float Difference, float ADifference){
180   Serial.println("TurnAngle:" + String(CoefSteer*Difference));
181     // Truning Left or Right:
182     if (Difference < 0) {
183       RotateServoRight(ADifference);  // Faire tourner à gauche
184       delay(2000);            // Attendre 2 secondes
185     } else {
186       RotateServoLeft(ADifference);  // Faire tourner à gauche
187       delay(2000);            // Attendre 2 secondes
188     }
189 }
190
191 void RotateServoLeft(int Angle) {
192   Serial.println("Turning Left");
193   Serial.println("PosServo:" + String(Angle));
194   Angle = 0.89275*Angle;
195   int NewPos = pos - int(round(Angle));
196   Serial.println(NewPos);
197   ServoParachute.writeMicroseconds(NewPos);
198   delay(2000);
199   ServoParachute.writeMicroseconds(pos);
200   delay(500);
201   NeedAdjust = false;
202 }
203
204 void RotateServoRight(int Angle) {
205   Serial.println("Turning Right");
206   Serial.println("PosServo:" + String(Angle));
207   Angle = 3.57142857*Angle;
208   int NewPos = pos + int(round(Angle));
209   Serial.println(NewPos);
210   ServoParachute.writeMicroseconds(NewPos);
211   delay(2000);
212   ServoParachute.writeMicroseconds(pos);
213   delay(500);
214   NeedAdjust = false;
215 }
216
217 void DropPayload(){
218   ServoTrap.attach(ServoTrapPin);
219   ServoTrap.write(180);
220   delay(1000);
221   ServoTrap.write(90);
222   delay(5000);
223   ServoTrap.attach(ServoTrapPin);
224   ServoTrap.write(0);
225   delay(1000);
226   ServoTrap.detach();
227 }
228
```

```
229  void loop() {
230    if ((millis() > Timer) && Set == false) {
231      ServoTrap.write(0);
232      delay(1500);
233      ServoTrap.write(90);
234      Set = true;
235    }
236    //Serial.println("test");
237    ReadGPS();
238    //NeedToAdjust(Bearing, Heading);
239    // if (NeedAdjust == true) {
240    //   SteerParachute(Difference);
241    // }
242    // if ((Distance < 10) && (Oppened == false)) {
243    //   Serial.println("Dropping Payload !!");
244    //   DropPayload();
245    //   Oppened = true;
246    // }
247    // delay(2000);
248    // RotateServoLeft(30);
249    //delay(5000);
250  }
```

Fig 44: Full Program

Gantt diagram :



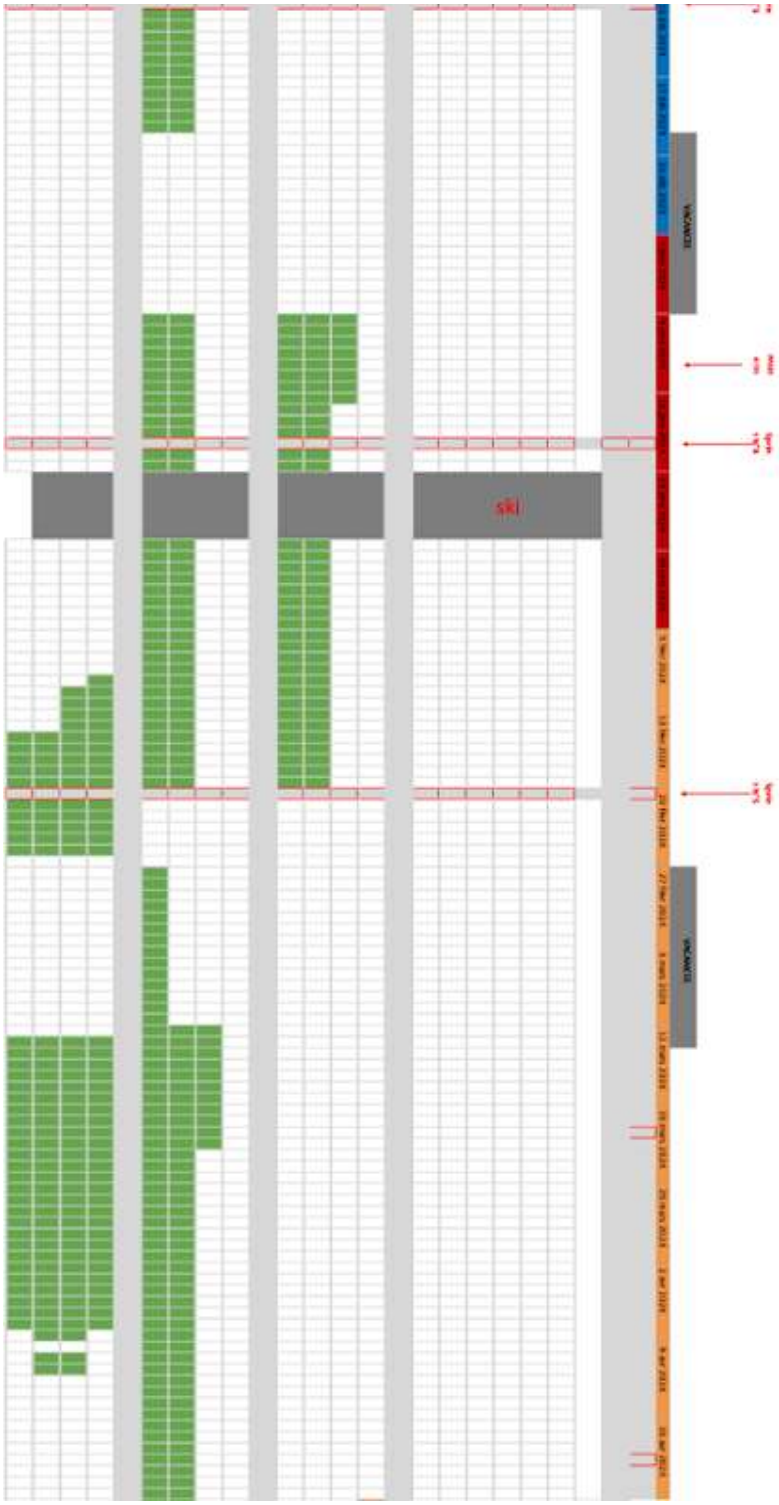


Fig 45: Gant diagram

Backlog :

US ID	task ID	description task	priority	status	Sprint n°
1	1.1	design a parachute capable of reducing the CanSat's fall speed	medium	finished	4
	1.2	design a resistant and efficient parachute	medium	finished	4
2	2.1	design a system which deploys the parachute on release	medium	finished	4
	2.2	automate parachute deployment	low	finished	4
3	3.1	design a remotely controllable on-board system	medium	finished	3
	3.2	design a system with a minimum range of 150m	high	finished	4
	3.3	design a system with an autonomy of at least 45min	medium	finished	3
4	4.1	design a solution able to mark the position where the CanSat land	high	finished	3
	4.2	design a solution which activates automatically on landing	high	finished	3
5	5.1	design a solution which allows the CanSat to stand upright after landing	high	finished	2 and 3
6	6.1	define roles on the team	low	finished	all
	6.2	build each sprint efficiently and then discuss possible improvements	medium	finished	all
	6.3	organize regular meeting with the team to follow the advancement	low	finished	all
7	7.1	deliver on time	high	finished	all
8	8.1	respect the dimensions of the CanSat described on the regulations	high	finished	all
	8.2	respect technical data for internal components described in the regulations	high	finished	all
9	9.1	create an original design to stand out from others groups	low	on going	5
	9.2	print the first prototype	medium	finished	3
	9.3	create an attractive design to draw attention to the project	low	finished	5
10	10.1	create a CanSat easy to transport and store in a box	low	finished	all
11	11.1	choose components to fit inside the CanSat	medium	finished	2
	11.2	integrate all components into the cansat body	medium	finished	5
	11.3	make decisional matrix to choose each components	medium	finished	1 and 2
	11.4	research each of the selected components	low	finished	2
	11.5	make a wiring plan with all the components	medium	finished	2
	11.6	test the components	medium	finished	3
	11.7	order individual components	medium	finished	2 and 3
	11.8	programming the components	medium	finished	4
12	12.1	the CanSat must not exceed the dimensions authorized in the regulations	high	finished	all
	12.2	make the dimensionning of the parachute	low	finished	2
	12.3	elaborate a final design of the CanSat	medium	finished	2
13	13.1	Design a CanSat with reusable mechanisms at least 50 times	medium	finished	all
14	14.1	Use low carbon material for the prototype design	low	finished	3

Fig 46: Backlog