# Mission Analysis and Feasibility Study for the Recovery System of a Suborbital Rocket

École Polytechnique Fédérale de Lausanne
Lausanne, Switzerland
June 2025

---

Ryan Svoboda 327 661

---

# Contents

# 1 Project & Document Overview

This report is the entry point to the documentation generated for the semester-long engineering study titled ***Mission Analysis and Feasibility Study for the Recovery System of a Suborbital Rocket***. The project was conducted as part of the *Space Technologies Minor* at EPFL in close collaboration with the **EPFL Rocket Team**, and spanned 14 weeks (12 ECTS credits).
The goal of the project was to evaluate the feasibility and define the preliminary architecture of a **recovery subsystem** for a student-designed suborbital rocket. This subsystem is responsible for the safe return of the vehicle or its components after flight. The study focuses on the earliest phases of system development and does not involve implementation or detailed design.

The scope is aligned with the early stages of the ECSS project lifecycle:

- **Phase 0 – Mission Analysis**: understanding stakeholder needs and system context.
- **Phase A – Feasibility Study**: evaluating solution concepts and defining the baseline architecture.

## 1.1 Purpose of This Document

This introductory section serves as a navigation aid for readers. It explains how the documentation set is organized, clarifies the logic behind document categorization, and outlines how different types of readers—such as academic supervisors, project reviewers, or future engineers—can engage with the content relevant to their role.
It does not include technical content itself. Instead, it introduces the structure and purpose of each document and provides a roadmap for locating key information.

## 1.2 Key Outputs

The study resulted in the following major deliverables:

- A **baseline system architecture** for the recovery subsystem.
- A **trade-off report** comparing and evaluating architectural concepts.
- A set of **project and technical requirements** for future development phases.
- A **development and verification roadmap** to support continued design activities.

## 1.3 Document Structure and Navigation

This report is structured as a collection of standalone yet interrelated documents, each addressing a specific aspect of the recovery system study. The organization reflects the logical flow of the project—from scoping and planning, through system design and trade studies, to tools and future development.
The documentation falls under two main categories:

- **Technical Documentation**: Describes the engineering and systems aspects of the project, including requirements, architecture, trade-offs, and tools.
- **Project Narration & Guidance (this section)**: Serves as a roadmap for navigating the above documents and understanding how they connect within the study's lifecycle.

Unlike a continuous narrative, the documentation is modular by design. Each part can be read independently but is traceable through cross-references and aligned with ECSS systems engineering practices.
Readers are encouraged to select entry points based on their focus:

- For a **systems-level view**, begin with the *System Engineering Plan* and *Trade-off Report*.

- For understanding **design constraints**, refer to the *Project Requirements Documentation* and *Preliminary Technical Requirements*.
- For implementation-oriented readers or future developers, the *Software Design Document* and *Software Development Plan* provide the most relevant insight.

## 1.4 List of Core Documents

Table 1 summarizes the key deliverables of the study. Each document plays a specific role in the development logic and supports the overall goal of defining a feasible and justified recovery system architecture.

**Table 1:** Core project documents and their role

| Document Name | Purpose |
|---|---|
| Project Requirements Documentation | Defines the scope, objectives, and constraints of the project from a mission and stakeholder perspective. |
| System Engineering Plan | Describes the project's systems engineering workflow, including methods, lifecycle stages, and integration with future work packages. |
| Preliminary Technical Requirements | Lists the technical requirements imposed on the recovery system and guides the trade-off and design processes. |
| Requirements Justification File | Explains the origin and rationale of each technical requirement, with traceability to sources and design logic. |
| System Concept Trade-off Report | Details the decision-making process and criteria used to select the baseline system architecture. |
| Pararec-Sim — Software Design & Development Document | Provides a high-level description of the custom simulator and optimiser used for architecture evaluation. |

# 2 Project Requirements Document

## 2.1 Introduction

This document defines the project-level requirements for the semester project titled *Mission Analysis and Feasibility Study for the Recovery System of a Suborbital Rocket*. It is part of the overall process documentation and aims to formally capture how project needs were identified, transformed into requirements, and managed throughout the study.

These requirements do not directly constrain the technical design of the recovery system. Instead, they define the boundaries, expectations, and evaluation criteria for the engineering feasibility study itself. In this sense, they guide how the study was conducted, not what the system must ultimately be.

This document supports traceability and transparency by providing:

- A structured overview of the project's scope and its boundaries;
- A step-by-step breakdown of the requirement definition methodology;
- Metadata and traceability between needs and formalized requirements;
- A clear distinction between project-level and system-level requirements.

The complete lists of needs and project requirements are included in the appendices.

### Title

**Mission Analysis and Feasibility Study for the Recovery System of a Suborbital Rocket**

### Description and Objectives

This project initiates the early-stage systems engineering work for the recovery subsystem of a student-built suborbital rocket. Its primary driver is the need to formally begin mission analysis, stakeholder need identification, and feasibility evaluation for this subsystem—corresponding to ECSS Phases 0 and A.

> *"Thus we would like to start the mission analysis, needs identification and feasibility study (ECSS Project Phase 0 and A) for the recovery subsystem."*

The project focuses on producing a first iteration of the systems engineer's responsibilities during these phases, including:

- Identifying and structuring stakeholder needs;
- Analyzing the feasibility of candidate recovery concepts;
- Defining an initial baseline system architecture.

In parallel with the technical content, the project applies systems engineering (SE) and project management (PM) practices relevant to this lifecycle stage. The outcome is both a technical and methodological baseline for future development.

### System Scope and Interfaces

This study focuses solely on the recovery subsystem—defined as the set of hardware and logic responsible for safely returning the rocket (or its components) to ground after flight. It explicitly excludes the avionics system, which is expected to host the sensors, actuators, and deployment logic. Interfaces with avionics are documented, but treated as external dependencies.

Due to the early development stage of the overall rocket, many high-level system requirements and upstream interfaces remain undefined. This uncertainty is acknowledged and addressed through flexible assumptions and scoping boundaries. The goal is to document and justify a feasible design space, not to finalize a system configuration.

#### Phases Covered

The project covers the first two lifecycle phases as defined in ECSS:

- **Phase 0 – Mission Analysis:** Identification of mission goals, stakeholder needs, and high-level constraints.
- **Phase A – Feasibility Study:** Exploration of candidate solutions, definition of technical requirements, and establishment of a baseline architecture.

Deliverables from this study are designed to support the transition to Phase B, where detailed design and implementation begin.

## 2.2 Requirements Definition Process

Requirements in this project were defined and maintained through a lightweight but traceable process appropriate for a Phase 0/A study. The goal was to produce a clear, justified set of project-level requirements that define the scope of work, align expectations, and guide all subsequent activities. The process follows five key steps:

#### Step 1: Stakeholder Identification

All stakeholders who influence or are affected by the project were identified. These include both technical actors (EPFL Rocket Team members) and academic supervisors. They provided the initial needs and validation inputs used throughout the study.

#### Step 2: Need Capture

Stakeholder inputs were translated into high-level needs. These needs are expressed in natural language, without technical prescriptions, and describe what the project must achieve—without defining how.

#### Step 3: Requirements Derivation

Each need was transformed into one or more project-level requirements. The resulting requirements follow ECSS guidance and aim to be specific, verifiable, and aligned with the scope of a feasibility study.

#### Step 4: Tagging and Metadata

Each requirement was annotated with a standardized set of metadata fields to support traceability and downstream usage. These fields include criticality, source, rationale, and more.

#### Step 5: Review and Update

The requirement set was maintained throughout the project. Formal review points (e.g., MDR, PRR) and informal design feedback loops were used to refine, prioritize, and verify the requirements progressively.

### Stakeholder Roles and Decision Points

- **Systems Engineer (Author):** Manages the requirement database and traceability model.
- **ERT Systems Engineer:** Provides needs and validates requirement scope.
- **Academic Supervisor:** Oversees alignment with project expectations.
- **ERT Team Leads:** Contribute technical inputs and participate in reviews.

Reviews were held at key milestones to validate the requirement set and approve updates based on evolving context.

## 2.3 Stakeholder Summary

For this study, three key stakeholder groups were identified. Their needs directly inform both project-level and system-level requirements.

### Stakeholder List

- **EPFL – eSpace (Academic Supervisor)**
- **EPFL Rocket Team – Spaceshot Project**
- **Launch Support – Esrange Launch Site**

### Roles and Responsibilities

**EPFL – eSpace and Rocket Team (Supervisory Stakeholders)**

- Define project quality expectations, review cadence, and documentation scope.
- Influence process methodology, deliverable format, and project boundaries.

**Rocket Team – Spaceshot (Technical Stakeholders)**

- Impose high-level technical constraints to ensure compatibility with the future rocket system.
- Provide inputs regarding interfaces, feasibility considerations, and system context.

**Launch Support – Esrange Site**

- Impose external operational constraints such as:
  - Maximum drift distance, landing accuracy
  - Submission of recovery risk assessments
  - Analysis of descent trajectory and impact profiles

All project needs and requirements are managed using a consistent set of metadata fields. These annotations support traceability, review, and filtering of the requirement database throughout the project lifecycle. The metadata structure is inspired by ECSS principles and adapted for use within the Obsidian-based project environment.

Each entry—whether a need or a requirement—is annotated with the following metadata:

**Table 2:** Metadata fields used for all needs and requirements

| Field | Description | Example / Options |
|---|---|---|
| `statement_type` | Distinguishes between needs and requirements | `Need`, `Requirement` |
| `domain` | Categorizes as process-related or technical | `Project`, `Technical` |
| `name` | Short label for identification | "Mission Definition Deadline" |

*(continued from previous page)*

| **Field** | **Description** | **Example / Options** |
|---|---|---|
| `text` | Full expression of the statement | (Free text) |
| `unique_identifier` | Traceable ID assigned to each entry | `NEED-001`, `REQ-005` |
| `source` | Origin of the statement | ECSS-E-ST-10C, Launch Site, Team Meeting |
| `original_author` | Author of the initial entry | "John Doe" |
| `date_entered` | First entry date | 2025-03-14 |
| `last_updated` | Date of most recent change | 2025-04-16 |
| `status` | Documentation and review status | Draft, In Review, Approved, Removed |
| `approval_date` | Date when entry was approved (if applicable) | 2025-04-18 |
| `verification_status` | Verification progress status | Not Started, Planned, Verified |
| `verification_type` | Method of verification | Analysis, Test, Inspection, Demonstration |
| `parent_ID` | Parent entry for traceability | `REQ-001` or `NEED-002` |
| `priority` | Priority level for implementation | High, Medium, Low |
| `criticality` | Whether non-satisfaction impacts project viability | Yes, No |
| `stability` | Expected stability over time | Stable, Likely to Change, Incomplete |
| `due_for` | Target milestone for completion | MDR, PRR |
| `rationale` | Justification for the statement | "Ensures compatibility with recovery timeline" |

## 2.4 Requirement Sources

All project-level requirements are derived from authoritative references or validated stakeholder inputs. The principal reference source is the ECSS standard set, selected due to its relevance to early-phase aerospace systems engineering and widespread use across institutional and academic contexts.

The ECSS standards were used for the following reasons:

- They offer a structured approach for requirement generation in feasibility phases;
- They define mandatory review stages and expected deliverables for each phase;
- They align well with academic engineering processes and enable traceability to future industrial contexts.

Other secondary sources—such as internal meetings, past work by the EPFL Rocket Team, and project supervisor feedback—were used to interpret, scope, or prioritize ECSS-derived content. However, they were not used to generate original requirements.

## 2.5 Needs-to-Requirements Traceability

Each project requirement is derived from at least one stakeholder need. This relationship is recorded through explicit traceability links to ensure that all requirements are grounded in justifiable expectations and that no stakeholder input is lost.

Traceability supports:

- **Completeness validation**: All expressed needs must result in corresponding requirements;
- **Justification tracking**: All requirements must trace back to a defined source;

- **Impact analysis**: Changes to needs can be propagated to linked requirements during updates.

A graphical traceability model is used to visualize relationships between needs and requirements. The full list of project-level requirements is managed dynamically using a markdown-based system in Obsidian. Each requirement is stored as a separate file and annotated with metadata to support traceability and status tracking.

The table below displays all requirements derived for this project. The table is generated using Obsidian's Dataview plugin and includes core metadata fields such as domain, source, and traceable parent needs.

**Example Dataview Query:**

**Listing 1:** Query to list all project-level requirements

```
TABLE
  text as "Text",
  domain as "Domain",
  source as "Source",
  due as "Due",
  parent_requirements as "Parent Statement"
FROM "Semester Project/Project Needs & Requirements"
WHERE contains(statement_type, "Requirement")
SORT length(parent_needs) ASC
```

The rendered result (not shown here) includes the current requirement set as maintained in the knowledge base. Each row links back to its original markdown file, enabling inline editing and review.

## 2.6 Requirement Management Approach

Requirements were managed using **Obsidian**, a markdown-based knowledge management platform. The environment enabled a flexible, version-controlled, and queryable system for authoring, organizing, and visualizing project-level needs and requirements.

### Requirement Authoring

- Each requirement or need is stored in its own `.md` file.
- A custom template ensures consistent structure, including:
  - Descriptive text and rationale;
  - Dropdowns for fields such as status, domain, criticality;
  - Links to parent/child relationships and verification types.
- Requirements derived from ECSS were imported, tagged, and adapted to this structure.

### Querying and Traceability

The **Dataview** plugin enabled all requirement pages to function as a dynamic database:

- Requirements were filtered and sorted by metadata (e.g., by milestone, priority, or stability).
- Tables were embedded into key documentation files and review dashboards.
- Pie charts and status summaries were created using JSON output from Dataview and rendered via the Charts plugin.

**Example Query – Requirements linked to the Trade-off Report:**

```
TABLE
  text as "Text",
  priority as "Priority",
  implementation_status as "Implementation Status"
FROM "Semester Project/Project Needs & Requirements"
WHERE
  contains(source, "ECSS-E-ST-10C Rev.1")
  AND contains(unique_identifier, "L.")  // Annex L: Trade-off report
  AND !contains(requirement_status, "Removed")
  AND !contains(priority, "Low")
SORT implementation_status ASC
```

This query identifies all high- and medium-priority requirements relevant to the trade-off study, supporting focused review and implementation.

### Version Control

All requirement files were version-controlled using Git. This provided:

- Fine-grained history of changes for each individual requirement;
- Rollback capability during major review cycles;
- Logs of changes to both text and metadata.

### Example Views

The system supported multiple embedded views:

- **Document Requirements View:** All requirements relevant to a specific document shown at the top of that file.
- **Review Task Pages:** Focused lists of entries needing review or update before specific milestones (e.g., PRR).
- **Project Status Overview:** Dashboards aggregating status, stability, and criticality ratings.
- **Full Index Page:** Dynamic list of all requirements with filters and live metadata summaries.

# 3 Systems Engineering Plan

## 3.1 Introduction

This Systems Engineering Plan (SEP) describes the structure, strategy, and outcomes of systems engineering activities conducted during the **Mission Analysis** and **Feasibility Study** phases (ECSS Phase 0/A) for the parachute recovery system.

The purpose of this document is twofold:

1. **To present the work that has been completed** since the beginning of the project—allowing the stakeholders to verify its completeness and alignment with the expected scope.
2. **To outline the activities planned until the Preliminary Requirements Review (PRR)**—enabling the stakeholders to validate and approve the direction and focus of the upcoming work.

The document captures both the methodology and outputs of each major engineering activity, including requirements definition, architecture exploration, interface identification, and trade-off studies.

## 3.2 Activities

### Phase 0

| Activity | Output |
|---|---|
| Identify Semester Project Needs | Project Needs and Requirements |
| Define Semester Project Requirements | Project Needs and Requirements |
| Identify Mission Needs | Mission Needs |
| Define Mission Requirements | Preliminary Technical Requirements Specifications, Requirement Justification File |
| Identify Key Technologies and Architectural Decisions | Trade Off Report |
| Define System Selection Criteria | Trade Off Report |
| Identify Interface with Past/Current/Future System Engineering Activities | Systems Engineering Plan |
| Propose Possible System Concepts | Trade Off Report, Mission Definition Document |

### Phase A

| Activity | Output |
|---|---|
| Propose System Solutions | Design Definition File, Design Justification File, Technology Matrix, Product Tree, Mission Operations Concept Document, Function Tree |
| Identify Risk Related to System Solution | Failure Modes and Effects Criticality Analysis, Risk Assessment Report |
| Identify Critical Items Related to System Solution | Critical Item List |
| Define Requirements for Next Lower Level | Preliminary Technical Requirements Specification for Next Lower Level |
| Propose Initial Verification Plan | Verification Plan |

| Activity | Output |
|---|---|
| Propose Initial Technology Plan | Technology Plan |
| Propose Initial Development Plan | Systems Engineering Plan, Work Breakdown Structure |
| Identify Interfaces | Interface Requirements Document |

## 3.3 Control Activities

### Process Compliance

To ensure engineering activities are aligned with the stakeholder's needs, A set of project needs and requirements were defined with the **EPFL Rocket Team** before the engineering activities commenced.

### Review Gates

Reviews with stakeholder within the **EPFL Rocket Team** were conducted throughout the project to validate progress and correct deviations early.

### Academic Supervision

Biweekly meetings with the academic supervisor served as recurring checkpoints for alignment, feedback, and planning. These meetings supported: - Early identification of technical or organizational issues - Progressive validation of deliverables - Iterative refinement of project scope and priorities

## 3.4 Interfaces with Other Work Packages

### Completed Work Packages

Past projects inform the current study through three main types of interfaces: 1. **References and Background Sources**
2. **Inherited Technical Requirements**
3. **Technology Feasibility Inputs**

### Work Packages Currently Being Worked On

The current study interfaces with two other projects that are also operating within ECSS Phase 0/A: - One project focuses on an **adjacent subsystem** (Payload). - The other works at the **system level**, developing architecture and mission-level requirements. The main form of interface is through requirements that constrain the recovery system's allowable architectures or design variable constraints. These constraints are still under definition and will evolve throughout the study.

### Future Work Packages

This feasibility study is expected to define the baseline for recovery system development in subsequent phases. The output of this work will serve as a formal input to future work packages through the following.

# 4 Preliminary Technical Requirements Specifications

## 4.1 User Needs Background

The recovery system's requirements originate from a combination of stakeholder expectations (EPFL Rocket Team, Esrange launch constraints), safety considerations, and mission performance objectives. The complete list of needs is included has been included as an appendix.

## 4.2 Product Perspective

The recovery system complements the rocket's structure, propulsion, avionics, and payload subsystems. It is constrained by the internal volume of the rocket airframe, mass budget limits, and deployment timing compatibility. Unlike orbital reentry systems, which require heat shields and complex guidance, this product is tailored for suborbital trajectories with a focus on simplicity, reliability, and compatibility with a 100 km apogee mission profile.

## 4.3 Product Independence Statement

The recovery system is not fully independent or self-contained. It is designed to rely on the sounding rocket's airframe for structural mounting and support during ascent. The system may also depend on the avionics subsystem for power and deployment initiation, though this interface has not been finalized and will depend on the selected system concept. While the recovery system performs its core function autonomously post-deployment, its integration and operational sequencing are contingent on the broader rocket system's design.

## 4.4 Higher-Tier System Context

The recovery system is a key component of the overall rocket system. It must interface effectively with structural and avionics components to ensure successful deployment and operation. Key interfaces include:

- **Airframe:**
    - Mechanical incorporation within the rocket structure;
    - Attachment to load-bearing elements for structural support.
- **Avionics:**
    - Provision of electrical power;
    - Transmission of deployment commands and operational signals.

## 4.5 Product Life Profile

The recovery system is expected to operate through the following lifecycle phases:

1. **Pre-Launch:** Integration, storage, transport, and final system checks at the launch site;
2. **Ascent — Thrust Phase:** From liftoff to motor burnout, with peak acceleration and vibration loads;
3. **Ascent — Ballistic Phase:** Post-burnout coasting to apogee, involving vacuum conditions and temperature extremes;
4. **High-Altitude Descent:** Initial descent through low-density atmosphere, including deployment events;
5. **Low-Altitude Descent:** Descent in denser atmosphere under parachute(s), subject to lateral winds;

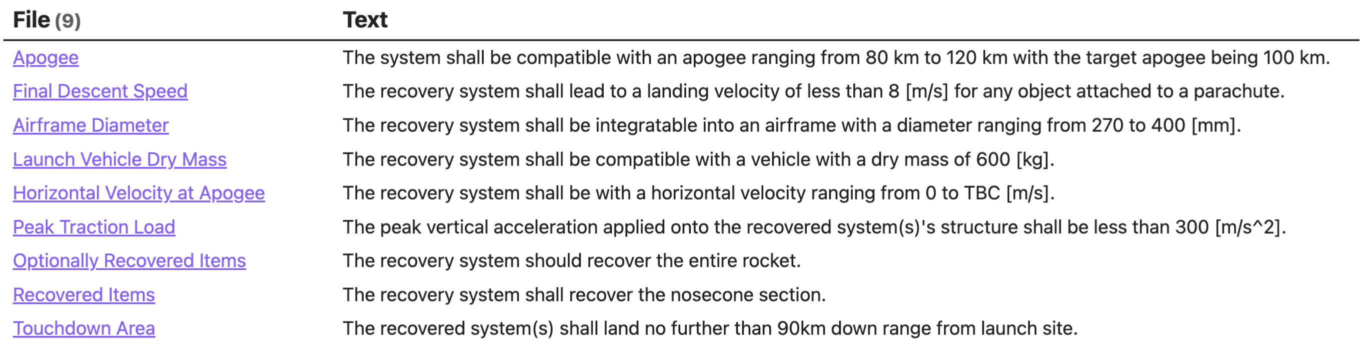
6. **Post-Landing:** Recovery, inspection, and transport of the landed system.

## 4.6 Environmental Constraints

The recovery system must withstand the following environmental conditions:

- **Pre-Launch:**
  - Exposure to ambient temperatures, humidity, and pressure variations;
  - Mechanical shocks during transport;
  - Static loads during handling and integration.
- **Thrust Phase of Ascent:**
  - High acceleration (TBC g-forces);
  - Strong vibration and acoustic loading (TBC frequency range);
  - Rapid altitude change and dynamic pressure increase.
- **Ballistic Ascent:**
  - Near-vacuum external pressure;
  - Low ambient temperatures (down to $-50°C$ or below);
  - High-altitude winds (e.g., jet streams).
- **High-Altitude Descent:**
  - Cold and low-pressure conditions;
  - Rapid dynamic pressure changes;
  - Aerodynamic instability and potential spin rates.
- **Low-Altitude Descent:**
  - Higher aerodynamic loads due to increased air density;
  - Wind-induced drift affecting landing location.
- **Landing and Post-Landing:**
  - Hard-surface impact loads (e.g., rocks, soil);
  - Risk of water landing (requires corrosion resistance and floatation if applicable);
  - Exposure to environmental contaminants (rain, dust, etc.).

## 4.7 Technical Requirements List



| File (9) | Text |
|---|---|
| Apogee | The system shall be compatible with an apogee ranging from 80 km to 120 km with the target apogee being 100 km. |
| Final Descent Speed | The recovery system shall lead to a landing velocity of less than 8 [m/s] for any object attached to a parachute. |
| Airframe Diameter | The recovery system shall be integratable into an airframe with a diameter ranging from 270 to 400 [mm]. |
| Launch Vehicle Dry Mass | The recovery system shall be compatible with a vehicle with a dry mass of 600 [kg]. |
| Horizontal Velocity at Apogee | The recovery system shall be with a horizontal velocity ranging from 0 to TBC [m/s]. |
| Peak Traction Load | The peak vertical acceleration applied onto the recovered system(s)'s structure shall be less than 300 [m/s^2]. |
| Optionally Recovered Items | The recovery system should recover the entire rocket. |
| Recovered Items | The recovery system shall recover the nosecone section. |
| Touchdown Area | The recovered system(s) shall land no further than 90km down range from launch site. |

# 5 Requirements Justification File

## 5.1 Stakeholder Summary

The complete list of stakeholders for the Spaceshot project is documented separately as part of the semester project covering phase 0 and phase A of the spaceshot project. For the scope of this semester project, three primary stakeholders have been identified:

**Stakeholders**

- **EPFL - eSpace (Academic Supervisor)**

- **EPFL Rocket Team – Spaceshot Project Team**

- **Launch Support – Launch Site (Esrange)**

**Roles, Responsibilities, and Influence**

**EPFL - eSpace (Academic Supervisor) and EPFL Rocket Team – Spaceshot**
- Define needs related to the scope, depth, and quality expected from the feasibility study.
- Define procedural needs regarding project reviews, including frequency, content expectations, and validation criteria.
- Influence the methodology, the documentation standard, and the timeline for deliverables.

**EPFL Rocket Team – Spaceshot (Technical Stakeholder)**
- Impose technical needs and high-level system requirements to ensure that the recovery system remains compatible with the evolving rocket system architecture.
- Act as the technical interface for system-level assumptions, constraints, and design feasibility feedback.

**Launch Support – Launch Site (Esrange)**
- Impose security and operational requirements applicable to the recovery system, including:
  - Performance requirements (e.g., maximum drift distance, landing zone constraints)
  - Documentation requirements (e.g., submission of risk assessments and recovery plans)
  - Analysis requirements (e.g., justification of descent trajectories and impact studies)

These stakeholders play a critical role in defining both the project-level and technical needs, and their inputs are directly reflected in the captured needs and derived project and technical requirements.

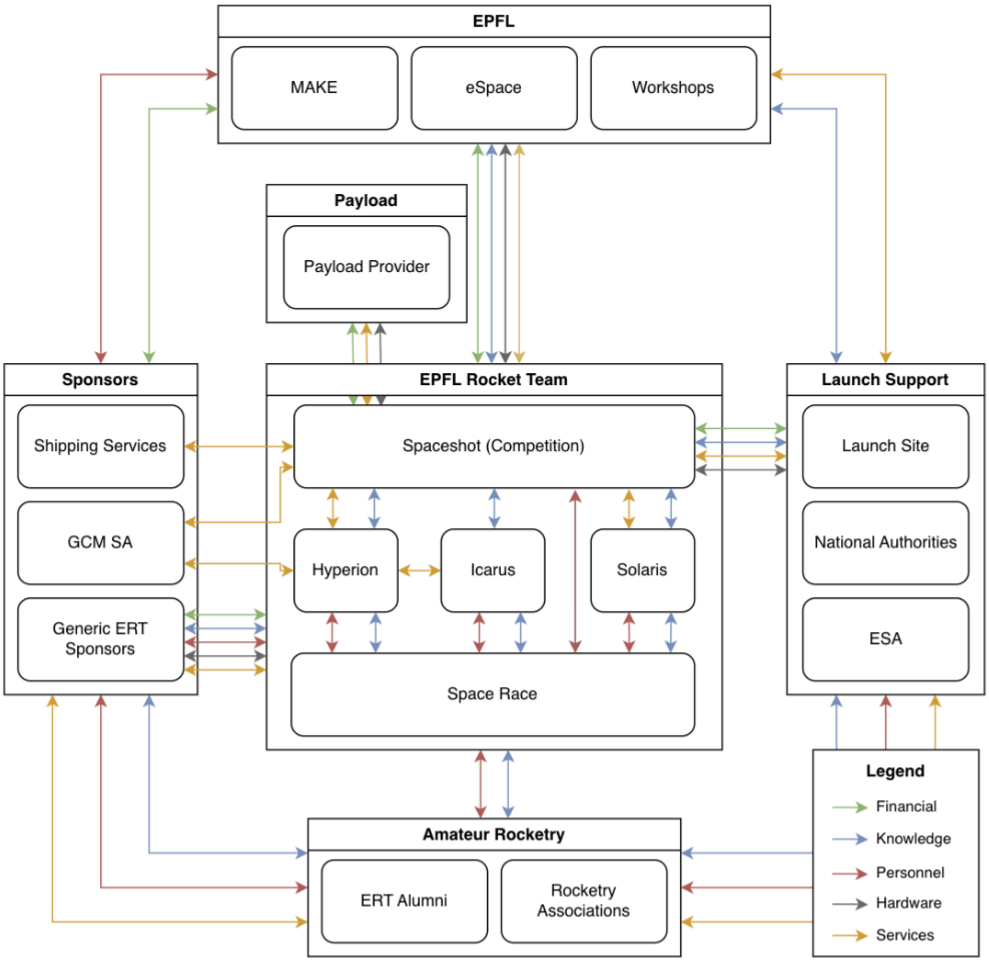


**Figure 1:** Stakeholder map (taken from: Truchot and Fuser [2025])

**Table 5:** Preliminary Technical Requirements

| ID | Requirement Text | Justification | Source | Verification |
|---|---|---|---|---|
| Apogee | The system shall be compatible with an apogee ranging from 80 km to 120 km, with a target of 100 km. | The target apogee of 100 km has been set as the mission objective by the rocket team, with a 20% tolerance margin. | EPFL Rocket Team (Stakeholder) | Analysis |
| Final Descent Speed | The recovery system shall lead to a landing velocity of less than 8 m/s for any object attached to a parachute. | Requirement taken from Esrange Safety Manual, Section 6.7.4. | Esrange Safety Manual | Analysis |
| Airframe Diameter | The recovery system shall be integratable into an airframe with a diameter ranging from 270 mm to 400 mm. | Inherited from prior studies; compatible with aerocover-equipped systems. | Truchot, Allenspach (2025 references) | Analysis |
| Launch Vehicle Dry Mass | The recovery system shall be compatible with a vehicle with a dry mass of 600 kg. | Pre-established constraint to avoid rework of launch transition mechanics. | Truchot (2025) | Analysis |
| Horizontal Velocity at Apogee | The recovery system shall be compatible with a horizontal velocity from 0 to TBC m/s. | TBD — value not yet defined. | — | Analysis |
| Peak Traction Load | The peak vertical acceleration on recovered components shall be less than 300 $m/s^2$. | Based on deceleration profile from rapid parachute deployment. | Allenspach (Static Load Study) | Analysis |
| Optionally Recovered Items | The recovery system should recover the entire rocket. | Stakeholder intention to support full-body recovery. | EPFL Rocket Team (Stakeholder) | Analysis |

# 6 Trade Off Report

## 6.1 Introduction

This document presents the Trade-Off Report for the recovery subsystem of a suborbital rocket, conducted as part of the ECSS Phase 0/A feasibility study. Its purpose is to document and justify the system-level decision-making process that led to the identification and recommendation of one or more viable recovery architectures. Because the final system will be implemented by future engineering teams, this study serves as a formal baseline that not only guides future development but also provides a transparent and defensible rationale for the decisions made.
The primary objective of this report is to make the decision logic both **transparent** and **traceable**. It does not prescribe a final implementation; rather, it evaluates a wide range of recovery strategies using a structured combination of qualitative and quantitative criteria. These evaluations are framed to match the maturity level expected of ECSS Phase 0/A, focusing on feasibility, architectural soundness, and early validation of assumptions.

> **Note:** As a Phase 0/A output, this document is intended to serve as a **decision-support reference**. It is not an implementation plan. The decisions and rationale captured here define a baseline to be carried forward. Unless critical flaws are identified in later phases, these trade-off activities are not expected to be repeated, though future development teams may adapt or extend them as new information becomes available.

### Scope

This report focuses on the **reasoning** and **evaluation logic** behind architectural decisions—not on fully specifying or detailing every option or subsystem.

### Document Structure

To maintain clarity and traceability, this document is structured as follows:

- **Trade Study Context**
  Describes the motivation for the study, outlines its role within the broader project, and identifies system boundaries and known biases.
- **Process Overview**
  Details how the trade-off process evolved, how criteria were selected, and how the tiered evaluation structure was applied.
- **Design Space Exploration**
  Outlines the recovery architecture design space and the taxonomy used to categorize candidate systems. Justifies why some branches are explored in depth while others are not.
- **Design Architecture Selection**
  Explains how Tier 1 and Tier 2 evaluations were conducted, presents the results, and defines the most promising system concepts.

## 6.2 Trade Study Context

### Motivation for the Trade Study

This trade study was initiated as a core activity of the project scope, recognizing that the selection of the recovery subsystem architecture is a foundational decision with long-term consequences. It will guide technical development, influence integration decisions, and shape the allocation of engineering and financial resources across future phases. Since the system will ultimately be implemented by student engineers not yet involved in the project, it is critical that the chosen architecture is not only **technically sound**, but also **rigorously justified and thoroughly documented** for continuity. Rather than focusing on detailed design selection, the objective of this trade study is to **establish a robust architectural baseline**. Its outcomes are expected to inform future definition work, investment planning, and more detailed subsystem design. If the trade-off is successful, its results should remain valid throughout downstream development and not need to be repeated—only refined as technical maturity increases.

In addition to supporting the current system-level decision, this trade study was also intended to establish a **structured and adaptable methodology**. The use of tiered evaluation, traceable selection criteria, and performance modeling was designed to allow: - **Adaptation to future changes** in constraints, requirements, or weighting of objectives, as new programmatic or technical information emerges. - **Reusability of the approach** for similar trade-offs in other subsystems or future student-led development phases.

The trade study is therefore positioned not only as a decision-support tool for this specific subsystem, but also as a prototype methodology for consistent, auditable architectural decisions within the broader systems engineering framework of the project.

### Evolution of the Trade-Off Approach

The original plan assumed a **uniform and exhaustive trade study**, in which the full design space would be explored and each option systematically evaluated against quantitative selection criteria before selecting the top-performing architecture. However, as the study progressed, the approach evolved into a **staged and adaptive process** out of both necessity and practicality.

- The **depth of exploration** varied depending on the architecture type: options with clear limitations or lower potential were explored more lightly, while promising branches received more detailed treatment.
- A **preliminary feasibility screening** (Tier 1) was introduced to eliminate clearly non-viable solutions early on. This freed up time and modeling resources for more promising candidates.
- A **refined evaluation phase** (Tier 2) focused on a narrower set of alternatives, applying detailed models, performance metrics, and optimization techniques to assess trade-offs more rigorously.
- Rather than forcing a single “optimal” solution, the final objective became to identify a **set of viable and defensible architectures**, each with distinct trade-offs, to serve as a robust foundation for future design work.

This shift in strategy reflected both the complexity of the design space and the practical constraints of the study. In particular, **tight scheduling and limited engineering bandwidth** were critical drivers. The majority of the trade-off study needed to be completed by the Mission Definition Review, held just five weeks after project kickoff. With much of the early phase consumed by requirements definition—and the project conducted alongside academic coursework—efficiency and prioritization became essential. The staged approach allowed the study to focus effort where it was most impactful, while still delivering strong justifications at each decision point.

### Scope and Boundaries of the Trade Study

This trade study focuses exclusively on the **recovery subsystem** of the Spaceshot rocket, considered as a **Level-1 system**. It does not address the overall rocket architecture (Level-0) or other major subsystems such as avionics or structure. The scope is limited to architectural and functional decisions — specifically, defining the **mission scope**, evaluating alternative **architecture-level**

**options** (e.g., full versus partial recovery, types of decelerators), and recommending a preferred system concept. It does **not** extend to detailed subsystem design, component sizing, or mechanism development.

Within this framing, the trade study **includes**: - Definition of the recovery mission scope. - Architecture-level trade-offs between major solution families. - Initial feasibility assumptions based on existing vehicle geometry, mass properties, and lessons learned from previous projects.

The study **excludes**: - Avionics hardware and software development (e.g., triggering, actuation, telemetry). - Detailed structural integration work (e.g., interface hardware, anchoring mechanisms). - Selection of final hardware technologies, design variables, or sizing of subsystem elements.

In terms of **system assumptions and interfaces**: - Recovery is expected to interface with the **structure**, **payload**, and **avionics** subsystems. - No formal interface planes were fixed at the time of the study. Structural assumptions (e.g., maximum allowable loads, preliminary mass budgets) are based on early estimates extrapolated from the Firehorn/Nordend architecture, made prior to formal system (system level 0) Phase 0/A activities. These estimates are appropriate for feasibility studies but are expected to evolve during detailed development phases. - The avionics interface remains undefined. Consistent with historical practices from Nordend and Firehorn, recovery is assumed **not responsible** for avionics hardware or software development; recovery equipment is actuated via avionics systems but developed independently.

The architectural recommendations made in this study are designed to be **compatible with the assumed subsystem interfaces**, while recognizing that formal interface definition will be required in subsequent project phases.

Regarding the **baseline context**, no specific architecture was mandated. However, the recovery solutions implemented in previous rockets — particularly parachute-based recovery with reefing and full vehicle recovery — provided an **established reference baseline** that informed feasibility assumptions and option prioritization.

Finally, although subsystem requirements (values) and interfaces will likely evolve as the Spaceshot system definition matures, the architectural decisions proposed here aim to offer **robustness** against foreseeable changes. Minor adaptations and performance re-validations are anticipated but the core architectural direction is intended to remain valid.

### Role of the Trade-Off in the Overall Project

This trade-off study establishes the architectural baseline for the recovery subsystem of the Spaceshot rocket. It defines a critical decision point that will guide all future subsystem design activities, providing a structured and traceable foundation for subsequent project phases. Beyond selecting a preferred recovery concept, the study formalizes a system-level decision-making framework aligned with ECSS Phase 0/A objectives.

Conducted during Phase 0/A of the overall project and in parallel with feasibility studies for other subsystems, this work takes place approximately five years before the planned launch. Its outputs directly support the Mission Definition Review (MDR) and Preliminary Requirements Review (PRR) and the transition toward Phase A & B activities. The study also leverages lessons learned from the Firehorn I project, ensuring that past experiences inform future architectural decisions.

The study integrates into the broader development at multiple levels: its outputs define recovery system expectations, guide the definition of subsystem interfaces, and reduce architectural uncertainty ahead of detailed design phases. The selected architecture is intended to serve as a stable baseline, with only minor refinements expected unless future developments reveal significant flaws. By establishing early clarity, the study aims to mitigate risks of major architectural changes later in the project.

The primary users of the results will be future recovery subsystem engineering teams, systems engineers defining interfaces and verification strategies, and project management teams planning resources and schedules. The study also demonstrates the use of formal trade-off methods within a student-led project, offering a methodology that can be adapted for future architecture-level decisions and strengthening continuity across project generations.

### Acknowledgment of Potential Bias

In the spirit of transparency, it is important to acknowledge potential sources of cognitive bias that could have influenced the trade-off process and architectural recommendations made in this study.
The author previously worked on the **Nordend rocket** as an engineer focused on parachute systems with reefing, and later served as a **systems engineer for the Firehorn I rocket**. During the Firehorn I project, the architectural decisions for the recovery system were made based on more limited trade-off studies and less formal systems engineering processes compared to the approach followed in this study. A parachute-based recovery system incorporating reefing stages was ultimately selected for Firehorn I.
Given this background, there is a recognized risk of **confirmation bias** — a tendency to favor solutions that resemble previously implemented architectures, particularly those involving parachute systems with reefing. Familiarity with these technologies and positive past project experiences could unintentionally influence the assessment of competing options, especially under conditions of limited time or incomplete data.
Mitigation of this bias was attempted through: - Formalization of a structured, tiered evaluation process. - Documentation and transparency of each evaluation step. - Explicit definition and use of objective selection criteria wherever feasible.
Nevertheless, readers and future reviewers should remain aware of this background when interpreting the trade-off outcomes and recommendations.

## 6.3 Process Overview

The trade-off process for the recovery subsystem was structured with two primary objectives: to ensure **traceability and clarity** of the decision-making logic, and to develop a **repeatable, adaptable methodology** for future applications. Recognizing the complexity of early architectural choices and the limited availability of detailed technical data, the process intentionally balanced **qualitative screening** with **quantitative performance evaluation**. This balance allowed efficient resource use without sacrificing decision robustness, particularly under the tight scheduling constraints of the semester project.
The overall process was organized into four sequential phases: - **Design Space Exploration**: Identify the full range of potential architectural options and categorize decisions using a structured taxonomy. - **Selection Criteria Definition**: Define a multi-tiered evaluation framework, linking stakeholder values to screening and scoring criteria. - **Evaluation of Design Solutions**: Apply the criteria in a phased manner—first through qualitative feasibility screening (Tier 1), then through quantitative performance modeling and optimization (Tier 2). - **Recommendations and Conclusions**: Synthesize evaluation outcomes into a set of viable architecture recommendations, each representing a distinct balance of trade-offs.
A visual representation of the process is provided below, illustrating how the design space was progressively narrowed through successive evaluation gates.

### Phase 1: Design Space Exploration

The first phase focused on **mapping the decision landscape**. A design space taxonomy was developed to organize architecture decisions into logical categories (e.g., mission framing choices, decelerator types, deployment methods). Options were sourced through literature reviews, ECSS standards, lessons learned from prior projects (Firehorn, Nordend), and recovery system design manuals. Each option was documented with a description, and initial pros and cons to support downstream evaluation.

### Phase 2: Selection Criteria Definition

Given the diversity and number of possible options, a **two-tiered evaluation system** was defined: - **Tier 1 criteria** focused on fast, mainly qualitative feasibility screening, allowing early elimination of clearly unsuitable options. - **Tier 2 criteria** targeted more detailed, quantitative comparison between viable architectures. These criteria related to performance metrics (e.g., mass, structural load, drift distance) that could only be evaluated through modeling.

Selection criteria were linked to high-level stakeholder objectives, ensuring that the architectures retained through the process aligned with the overall mission priorities. Cutoff thresholds were set for certain Tier 1 criteria to allow efficient elimination without requiring exhaustive evaluation of all metrics for low-performing options.

### Phase 3: Evaluation of Design Solutions

In Tier 1, each design option was assessed against the defined feasibility criteria. Options failing to meet the minimum thresholds were documented and discarded, with justification based on engineering judgment and reference sources.

In Tier 2, a **custom analytical model** was developed to simulate the performance of full architecture candidates. This model enabled estimation of key metrics such as drift, system mass and opening loads. A **Genetic Algorithm (GA)** optimization approach was applied to optimize design variables (e.g., deployment altitudes, parachute sizing) for each architecture, allowing fair comparison of optimized candidates rather than arbitrary configurations.

Architectures were then compared using the Tier 2 criteria, focusing on identifying a **set of high-performing solutions** along the Pareto front, each representing different trade-offs between key objectives.

### Phase 4: Recommendations and Conclusions

The trade-off process culminated in the recommendation of **several architectures**, each offering distinct advantages depending on future mission priorities and system evolution. Rather than prescribing a single optimal solution, the study offers a **portfolio of viable choices** that can be refined as constraints, requirements, and available technologies mature.

The structured, tiered methodology is designed to be flexible: while the Tier 1 feasibility screenings are expected to remain stable, Tier 2 evaluations may need adjustment if significant changes occur in system-level requirements or available technologies. Nonetheless, the overall architecture recommendations are intended to provide a robust foundation for recovery system development in subsequent phases.

## 6.4 Design Space Exploration

This section presents the full range of architectural and technological options considered for the recovery subsystem. It defines the “design universe” from which all later trade-off decisions are drawn, serving as the foundation for the Tier 1 and Tier 2 evaluation phases.

The primary goal of this section is to ensure **transparency, traceability, and breadth** in the decision-making process. Each option is described and categorized to allow downstream reviewers, engineers, and decision-makers to understand what was considered, why it was included, and how it compares to other alternatives. The design space exploration also supports future re-evaluation by providing a well-documented set of baseline possibilities.

Only **design decisions** are included here. These define how a recovery architecture is conceptually framed. This section does **not** cover **design variables**, which are quantitative parameters (e.g., geometry or timing values) optimized later through simulation and analytical modeling.

The design space includes: - **Mission-level framing decisions** (e.g., full vs. partial recovery) - **Decelerator types** (e.g., parachutes, IADs, propulsive systems) - **Deployment and enabling mechanisms** (e.g., reefing, clustering, flat-spin deployment)

Excluded from this section are subsystem-level implementation details and numerical design parameters, which are evaluated separately during performance modeling.

The **depth of exploration** is not uniform across all categories. Certain branches—such as parachute technologies—were investigated more thoroughly based on: - Prior team experience (e.g., Nordend, Firehorn) - Preliminary feasibility and maturity - Relevance to likely candidate architectures

In contrast, less common or less feasible branches, such as helicopter-based recovery or propulsive landing, were included with minimal detail.

This design space forms the **core input** to the trade-off process. A total of **22 decision categories** and **137 decision options** are captured. These are progressively filtered through Tier 1 evaluation and used to construct candidate architectures for Tier 2 modeling and optimization.

### Design Space Taxonomy

To structure the wide variety of architectural decisions involved in recovery system design, a **design space taxonomy** was developed. This taxonomy serves multiple purposes: it helps **organize the decision space**, supports **modular documentation**, and clarifies how different design elements relate to one another. It also improves **traceability** across the evaluation and documentation process.

Each design decision category is tagged with three **attribute labels**: - `Level` — describes the abstraction tier of the decision. - `Role` — captures the logical function or purpose of the decision in the trade-off. - `Type` — indicates whether the decision introduces a new design path or modifies an existing one.

These attributes were **not used directly in the evaluation process**, but they improve documentation clarity and may support future decisions around **responsibility assignment** (e.g., which subsystem team owns the decision) or **sequencing** (e.g., which decisions must be made early in the design process).

### Attribute Definitions

| Attribute | Definition | Possible Values |
|---|---|---|
| **Level** | Describes the structural tier of the design decision in the system hierarchy, from broad mission choices to specific implementations. | Tier-0 Mission Framing, Tier-1 Architectural Structure, Tier-2 Technology Group, Tier-3 Implementation Detail |
| **Role** | Describes the logical function or purpose of the design decision (i.e., what kind of reasoning or input governs it). | Principle, Architecture Decision, Technology, Configuration, Constraint, Design Strategy |
| **Type** | Describes how the decision behaves in the design space—either by introducing a new path (Driver) or tuning an existing one (Modifier). | Set Driver, Set Modifier |

### Attribute Option Definitions

| Attribute | Option | Definition |
|---|---|---|
| **Level** | **Tier-0 Mission Framing** | High-level decisions about mission intent, operating environment, and core system behavior. |
| | **Tier-1 Architectural Structure** | Defines the major system architecture, overall recovery configuration, and structural breakdown. |
| | **Tier-2 Technology Group** | Represents families of candidate technologies within an architecture or subsystem. |
| | **Tier-3 Implementation Choice** | Focuses on detailed technology selection, parameter tuning, configuration logic, and subsystem integration. |
| **Role** | **Principle** | A guiding physical or operational concept that constrains or initiates downstream design logic. |
| | **Architecture Decision** | A major system design choice shaping overall structure and behavior. |
| | **Technology** | A selectable technology or method available within a given architecture. |
| | **Configuration** | A tunable parameter, quantity, or structural choice used to adjust a selected system or technology. |
| | **Constraint** | A condition or requirement derived from mission, safety, or system-level limitations. |
| | **Design Strategy** | A cross-cutting design logic or integration pattern that governs how elements interact or are grouped. |
| **Type** | **Set Driver** | Creates a new design branch with cascading influence on other decisions. |
| | **Set Modifier** | Refines or tunes an existing design branch without changing the overall design path. |

### Overview of Design Decision Categories

The recovery system design space was decomposed into **22 distinct decision categories**, each associated with a specific role and abstraction tier in the design hierarchy. These categories were classified using the attribute taxonomy described in the previous section: `Tier`, `Role`, and `Type`. The table below summarizes all identified decision categories:

| **Category Name** | **Tier** | **Role** | **Type** |
|---|---|---|---|
| Mission Scope | Tier-0 Mission Framing | Architecture Decision | Set Driver |
| Decelerator Type | Tier-0 Mission Framing | Principle | Set Driver |
| Landing Zone | Tier-0 Mission Framing | Architecture Decision | Set Driver |
| Aero Dynamic Decelerator Types | Tier-1 Architectural Structure | Architecture Decision | Set Driver |
| Partial Recovery Types | Tier-1 Architectural Structure | Architecture Decision | Set Driver |
| Full Recovery Types | Tier-1 Architectural Structure | Architecture Decision | Set Driver |
| Orientation at Touchdown | Tier-1 Architectural Structure | Architecture Decision | Set Driver |
| Recovery Module Storage Area(s) | Tier-1 Architectural Structure | Architecture Decision | Set Driver |
| Solid Textile Parachute Types | Tier-2 Technology Group | Technology | Set Driver |
| Slotted Parachute Types | Tier-2 Technology Group | Technology | Set Driver |
| Rotating Parachutes | Tier-2 Technology Group | Technology | Set Driver |
| Maneuverable (Gliding) Parachutes | Tier-2 Technology Group | Technology | Set Driver |
| Inflatable Aerodynamic Decelerators Types | Tier-2 Technology Group | Technology | Set Driver |
| Attached IAD | Tier-2 Technology Group | Technology | Set Driver |
| Trailing IAD | Tier-2 Technology Group | Technology | Set Driver |
| Impact Attenuation Systems | Tier-3 Implementation Choice | Technology | Set Modifier |
| Buoyance Device | Tier-3 Implementation Choice | Technology | Set Modifier |
| Reefing | Tier-3 Implementation Choice | Technology | Set Modifier |
| Deployment Mechanism | Tier-3 Implementation Choice | Technology | Set Modifier |
| Clustering | Tier-3 Implementation Choice | Design Strategy | Set Modifier |
| Supersonic Deployment Requirement | Tier-3 Implementation Choice | Constraint | Set Modifier |
| Flat-Spin | Tier-3 Implementation Choice | Constraint | Set Modifier |
| Number of Stages | Tier-3 Implementation Choice | Configuration | Set Modifier |

This classification enables consistent referencing throughout the evaluation process and supports **layered design analysis**.

### Sources and References

The recovery system design options were identified and documented using a broad range of sources. These sources include: - **Past EPFL Rocket Team projects**, such as *Firehorn* and *Nordend*, which provided firsthand engineering experience and served as contextual baselines. - **External literature and studies**, which expanded the range of candidate technologies and offered insights into modeling practices and historical implementations. - **Design handbooks** and technical manuals that formalize recovery system engineering practices and standard architectures.

No formal weighting or ranking was assigned to the sources. Internal knowledge (from Firehorn and Nordend) was treated with the same consideration as external sources.

> **Note:** The sources listed below were primarily used to define and describe the **design options** themselves. Other references—such as performance evaluations, modeling techniques, or historical applications—are cited in the individual design option pages or referenced during the evaluation phase.

| ID | Title | Type |
|---|---|---|
| 1 | Design of a Helicopter System Recovery for a Sounding Rocket, Caldeira et al. [2024] | Paper (Research) |
| 2 | Parachute Recovery Systems, Knacke [1992] | |
| 3 | Flat-Spin Recovery System, Fall [1964] | Paper (Application Technique) |
| 4 | Supersonic Inflatable Aerodynamic Decelerators for use on Sounding Rocket Payloads, Miller et al. [2023] | Paper (Technology Assessment) |
| 5 | Estimating Mass of Inflatable Aerodynamic Decelerators Using Dimensionless Parameters, 8th [2011] | Paper (Mass Estimation Technique) |
| 6 | A Historical Review of Inflatable Aerodynamic Decelerator Technology Development, Smith et al. [2010] | Paper (Historical Review of Technology Development) |
| 7 | STATE-OF- THE-ART STUDY FOR HIGH-SPEED DECELERATION AND STABILIZATION DEVICES, C and A [1966] | Study (State of The Art) |
| 8 | Comparison of Various Parachute Deployment Systems for Full Rocket Recovery of Sounding Rockets, EUC [2019] | Paper (Architecture Selection) |

### Design Decision Tables

This section presents the full set of design options identified during the **Design Space Exploration** phase. Each **decision category**—representing a specific architectural or technological choice—is documented using a standardized table format. These tables provide a **side-by-side overview** of the options available, along with their key attributes and trade-offs.

Each table includes: - A **short description** of the option. - The **primary sources** that informed its identification or rationale. - A summary of **pros, cons**, and **main risks** associated with the option. - Where applicable, a **diagram or sketch**.

The purpose of these tables is strictly **exploratory**: - No filtering, selection, or prioritization is applied at this stage. - All options are retained.

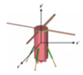
The following pages include one table per design decision category.

# Mission & Architectural Choices

## Mission Scope

| File (2) | Source | Description | Pros | Cons | Risks | Diagram / Image |
|---|---|---|---|---|---|---|
| Full Recovery | - | Recover the entire rocket, including propulsion and structural systems. | Enables full post-flight analysis; minimizes environmental pollution; enables public display or reuse. | Increased weight and system complexity; may reduce payload capacity. | System may exceed mass or complexity limits; difficult to qualify under tight development timelines. | |
| Partial Recovery | - | Recover only part of the rocket (e.g. payload or avionics). | Lighter system; fewer subsystems to develop; lower cost. | Loss of valuable hardware; less comprehensive post-flight data. | Local regulations may prohibit uncontrolled impact of unrecovered parts. | |

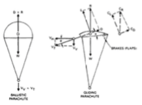
## Decelerator Type

| File (3) | Source | Description | Pros | Cons | Risks | Diagram / Image |
|---|---|---|---|---|---|---|
| Aero Dynamic Decelerator | - | Uses parachutes, ballutes, or inflatable drag surfaces to increase drag and slow descent through atmospheric braking. | Well-understood, flight-proven system; lightweight and passive; low power demand. | Performance strongly dependent on atmospheric density and dynamic pressure profiles. | Over-constrains flight profile (e.g., coast phase duration); risk of failed deployment. | |
| Helicopter System Recovery | - | Recovery using rotary wings via the autorotation phenomenon, enabling controlled, safe landings and potential system reuse. Offers flexibility in mission planning and can support applications like atmospheric research thanks to adjustable descent profile. | Flexible mission concepts; potential for reusability; controllable descent path. | No spaceflight heritage; may be aerodynamically unstable at high speed. | Too complex to develop within current team and time constraints. |  Fig. 4 Coordinate system of the first stage's body. Adapted from [3] |
| Propulsive Landing | - | Landing using engine reignition with thrust vector control (TVC) and aerodynamic control surfaces to decelerate and stabilize descent. | Enables highly precise landings; no need for a large landing zone. | Extremely complex and expensive; requires real-time engine control and redundancy. | Too complex to develop with available resources and time. | |

## Landing Zone

| File (2) | Source | Description | Pros | Cons | Risks | Diagram / Image |
|---|---|---|---|---|---|---|
| Impact on Land | - | Final touchdown occurs on solid terrain (e.g. desert or field). | Easier to locate and recover post-landing; vehicle remains stationary; accessible by ground vehicle. | May require an impact attenuation device to reduce damage on impact. | Possible structural damage on hard surface; may destroy part of the rocket. | |
| Impact on Water | 2 | Final touchdown occurs on water. | Allows for harder landings with reduced structural damage; potentially softer deceleration | May require a buoyancy device to remain afloat; adds complexity | If buoyancy system fails, rocket could sink and be unrecoverable. | |

## Aero Dynamic Decelerator Types

| File (5) | Source | Description | Pros | Cons | Risks | Diagram / Image |
|---|---|---|---|---|---|---|
| Inflatable Aerodynamic Decelerators | 2, 4 | Inflatable drag devices (e.g., ballutes, IADs) that deploy and inflate to increase drag during descent. Useful in thin atmospheres or high-altitude entry. | Low mass; compact storage; scalable; effective at high speeds and altitudes. | Complex inflation system; sensitive to aerodynamic heating and flutter. | Failure to fully inflate can result in insufficient deceleration or instability. |  |
| Maneuverable Parachute | 2 | Gliding parachutes capable of controlled turning and forward flight, enabling guided landings to target zones. Used in sport and precision recovery applications. | Can steer toward target; extended glide range; enables soft landings. | Larger footprint; reduced drag compared to traditional parachutes. | Requires active guidance or skilled control to ensure accuracy. |  |
| Rotating Parachute | 2 | Features angled vents or asymmetric shapes to generate rotation, improving stability and increasing projected area through centrifugal force. | High drag coefficient; stabilizing spin; good for compact deployment. | Difficult to scale up; complex suspension geometry; deployment failures at large size. | Wrap-up of canopy or failed spin initiation can cause total failure. | |
| Slotted Parachute | 2 | Uses slits or openings to control porosity and improve performance across various Mach regimes. Includes ribbon, ringslot, ringsail, and disk-gap-band variants. | Tunable performance; can operate at subsonic and supersonic speeds; reduces canopy stress. | Requires careful porosity selection; more complex manufacturing. | Improper porosity may result in excessive opening forces or low drag performance. | |
| Solid Textile Parachute | 2 | Classic non-porous fabric parachute type (e.g. flat circular, hemispherical), traditionally used for descent and recovery. Offers high drag performance and simple deployment. | Proven and widely used; reliable; simple construction. | Can have poor stability; limited supersonic capability. | Risk of canopy collapse or oscillation if not properly reefed or sized. | |

## Partial Recovery Types

| File (3) | Source | Description | Pros | Cons | Risks | Diagram / Image |
|---|---|---|---|---|---|---|
| Recover Nosecone and Avionics | - | The nosecone and avionics are recovered together as a single unit. | Balances mass and value retention; retains both data and payload. | Slightly heavier recovery system | - | |
| Recover only Avionics System | - | Only the avionics bay (flight computer, sensors, etc.) is recovered. | Preserves critical flight data; small recovery system; low mass. | Payload is lost | - | |
| Recover only Nosecone | - | Only the nosecone containing the payload, is recovered. | Minimizes recovery system mass; protects valuable payload; simple. | Avionics and propulsion systems are lost; limited data retrieval. | - | |

## Full Recovery Types

| File (3) | Source | Description | Pros | Cons | Risks | Diagram / Image |
|---|---|---|---|---|---|---|
| Multi-Bodied Recovery | - | The rocket separates into multiple modules, each with its own dedicated recovery system. | Modular; optimized recovery per module; scalable architecture; allows descoping recovery of propulsion stage later in development. | Increased complexity; Two recovery systems. | Timing issues or misdeployment could result in collision or incomplete recovery. | |
| No Rocket Separation | - | The entire rocket descends as a single rigid body, without separating into stages or modules. | - | High mass. Larger aerodynamic decelerator required. | - | |
| Separate rocket but keep both bodies attached to a single recovery system | - | The rocket separates (e.g., nosecone and body), but both parts remain connected via a common parachute system. | Single recovery system; reduced chute mass compared to multi-body recovery; | Complex rigging; | Failure of tether or imbalance could cause collision or spin during descent. | |

### Driving Technology Choices

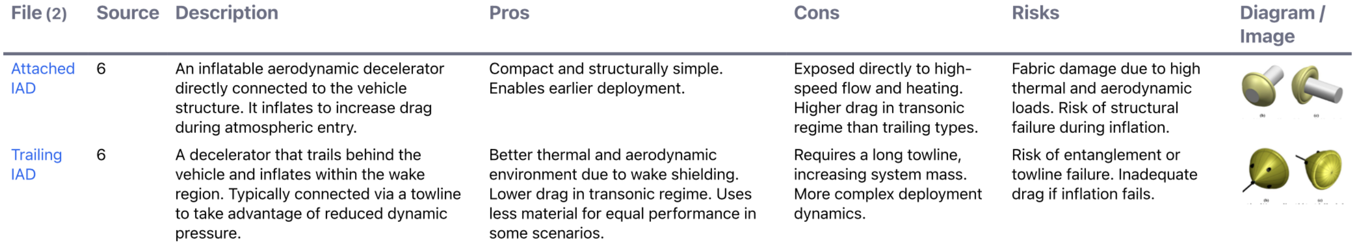
**Solid Textile Parachute Types** (Source ID: 2)

**Slotted Parachute Types** (Source ID: 2)

**Rotating Parachutes** (Source ID: 2)

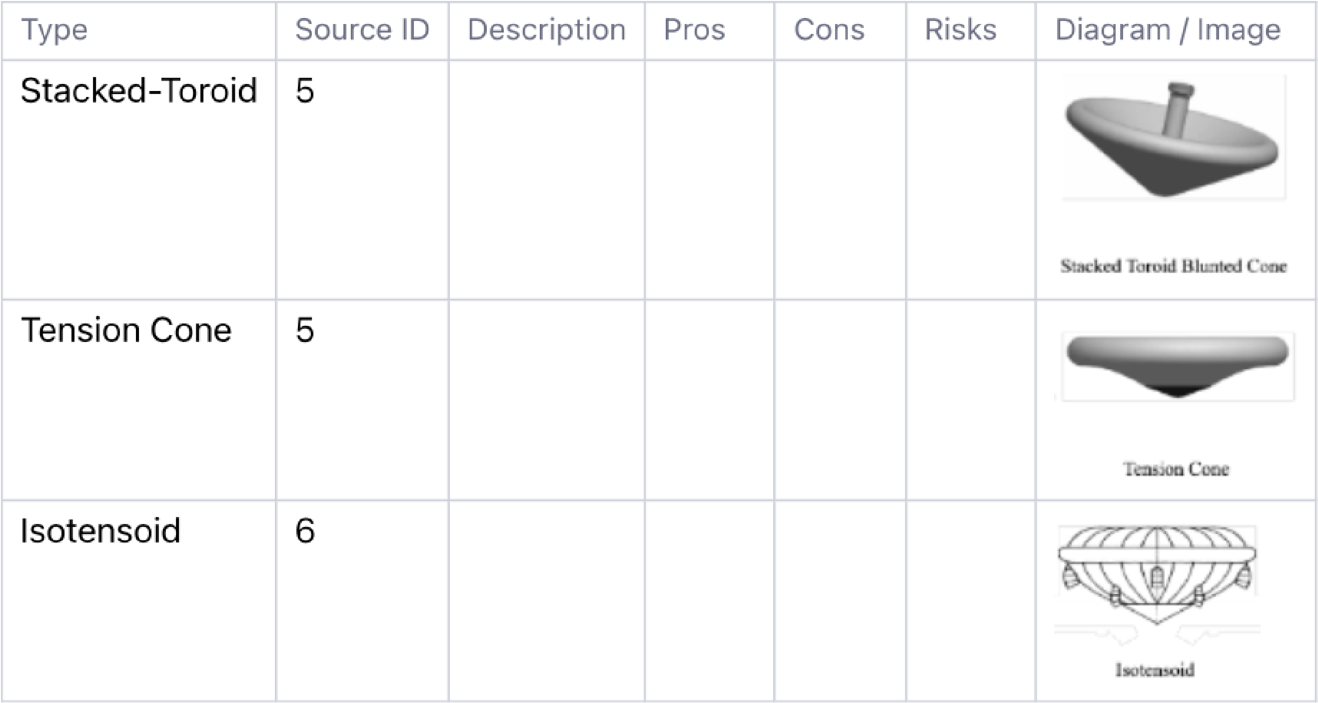
### Manoeuvrable (Gliding) Parachutes (Source ID: 2)

### Inflatable Aerodynamic Decelerators Types



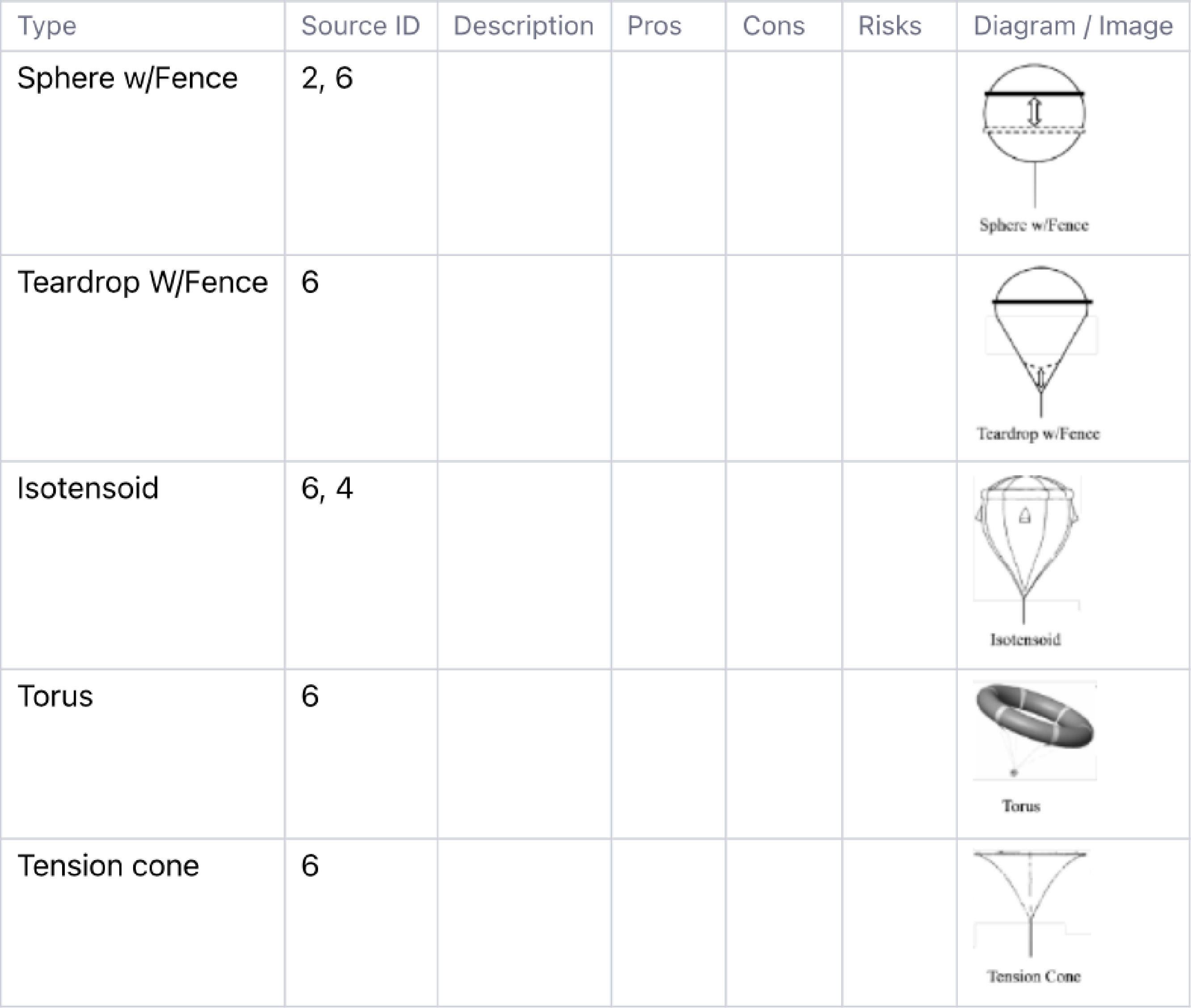
| File (2) | Source | Description | Pros | Cons | Risks | Diagram / Image |
|---|---|---|---|---|---|---|
| Attached IAD | 6 | An inflatable aerodynamic decelerator directly connected to the vehicle structure. It inflates to increase drag during atmospheric entry. | Compact and structurally simple. Enables earlier deployment. | Exposed directly to high-speed flow and heating. Higher drag in transonic regime than trailing types. | Fabric damage due to high thermal and aerodynamic loads. Risk of structural failure during inflation. | |
| Trailing IAD | 6 | A decelerator that trails behind the vehicle and inflates within the wake region. Typically connected via a towline to take advantage of reduced dynamic pressure. | Better thermal and aerodynamic environment due to wake shielding. Lower drag in transonic regime. Uses less material for equal performance in some scenarios. | Requires a long towline, increasing system mass. More complex deployment dynamics. | Risk of entanglement or towline failure. Inadequate drag if inflation fails. | |

### Attached IAD



| Type | Source ID | Description | Pros | Cons | Risks | Diagram / Image |
|---|---|---|---|---|---|---|
| Stacked-Toroid | 5 | | | | | Stacked Toroid Blunted Cone |
| Tension Cone | 5 | | | | | Tension Cone |
| Isotensoid | 6 | | | | | Isotensoid |

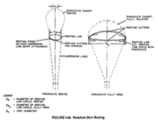
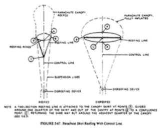
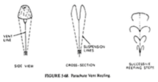
### Trailing IAD



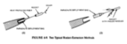
| Type | Source ID | Description | Pros | Cons | Risks | Diagram / Image |
|---|---|---|---|---|---|---|
| Sphere w/Fence | 2, 6 | | | | | Sphere w/Fence |
| Teardrop W/Fence | 6 | | | | | Teardrop w/Fence |
| Isotensoid | 6, 4 | | | | | Isotensoid |
| Torus | 6 | | | | | Torus |
| Tension cone | 6 | | | | | Tension Cone |

## Set Modifiers

### Reefing

| File (7) | Source | Description | Pros | Cons | Risks | Diagram / Image |
|---|---|---|---|---|---|---|
| Continuous Disreefing | 2 | Concept where reefing follows a force-time profile via automated line payout. Investigated but not operationalized. | Theoretically allows smooth, optimized inflation. | No practical implementation to date. | Development complexity; unproven reliability. | |
| Fixed Pocketband Reefing | 2 | Used in high-porosity ribbon/ringslot parachutes to stabilize inflation when porosity exceeds limits. | Improves stability for specific porosities. | Limited to niche cases; lacks inflation control flexibility. | May hinder proper deployment if misapplied. | |
| No reefing | 2 | Parachute deploys to full inflation without any staged delay. | Simplest deployment system; fewer failure points. | High opening shock; unsuitable for high-speed descents. | May exceed structural limits or cause failure at deployment. | |
| Skirt Reefing | 2 | The most common reefing technique where rings and a continuous reefing line restrict canopy opening. Reefing cutters release the line to allow gradual deployment. | Widely used; effective force control; compatible with many canopy types. | Requires pyrotechnic cutters and control systems. | Cutter failure may prevent full deployment or cause premature inflation. |  |
| Skirt Reefing with Control Line | 2 | Uses a two-section reefing line guided around the canopy and controlled by a single line for manual or sensor-based disreefing. | Allows continuous or dynamic reefing; | Complex routing and tension balancing. | May fail to disreef if control line does not payout properly. |  |
| Slider Reefing | 2 | A slider (fabric with grommets) passes over suspension lines to delay canopy inflation. Common in sport gliders. | Simple, reusable. | Not suitable for high-speed or large cargo applications. | - | |
| Vent Reefing | 2 | A line is attached to the center of the parachute vent; pulling it reduces drag area and increases drag coefficient temporarily. | Allows high drag configurations; suitable for annular/airfoil types. | Complex actuation; risks suspension line wrap. | - |  |

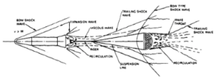
### Deployment Mechanism

| File (5) | Source | Description | Pros | Cons | Risks | Diagram / Image |
|---|---|---|---|---|---|---|
| Deployment by Rocket Extraction | 2 | A rocket pulls the parachute bag, enabling line-first deployment. | Clean, controlled deployment; minimizes forces on the vehicle. | Requires careful integration of rocket components; thermal protection for nearby elements; requires extra pyrotechnics; requires extra safety procedure | Rocket malfunction could result in no deployment or damage to the vehicle or parachute. |  |
| Drogue Gun | 2 | A drogue gun operates like a pistol, with a piston-fired slug that ejects from the vehicle upon command. This slug travels into clean airflow behind the vehicle and pulls a bridle that either deploys the pilot chute directly or extracts a deployment bag containing it. | Simple and compact | Limited energy compared to mortars; can only handle lighter parachutes. | Misfire or misalignment can prevent clean extraction. | |
| No Deployment Mechanism | 2 | Parachute is deployed by environmental forces alone (e.g., gravity or airflow after ejection). | Simplifies system; fewer failure modes. | Less predictable behavior; complex to design for consistent deployment; not adapted for deploying from spinning or tumbling systems or systems with large forebody | Risk of delayed or no deployment due to orientation or airflow conditions. | |
| Parachute Mortar | 2 | A cannon-like deployment that fires the parachute assembly away from the wake into clear airflow, using inertia rather than pull. | Very powerful; can deploy heavy parachutes reliably from unstable orientations. | High reaction force on vehicle; bulky and heavy system. | Damage from recoil, or incomplete bag strip if not enough velocity. | |
| Static-Line Deployment | 2 | A fixed-length cord connects the parachute to the rocket, automatically extracting it when the nosecone separates. | Very reliable for low-altitude or short-delay deployments. | No control over timing once released; line must be correctly sized and routed. | Entanglement if line tension isn't sufficient or geometry isn't controlled. | |

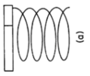
### Clustering

| File (2) | Source | Description | Pros | Cons | Risks | Diagram / Image |
|---|---|---|---|---|---|---|
| Clustering | 2 | A parachute cluster consists of two or more parachutes used to stabilize, decelerate, or lower a payload or air vehicle. It allows distribution of load across multiple canopies. | Easier fabrication, rigging, and handling; supports wide payload ranges; higher system reliability; improved descent stability; shorter filling time than single large parachutes | Increased complexity in deployment; synchronization issues; drag reduction due to interference; higher total weight and volume; variation in disreef times; lead/lag inflation risks | Asynchronous inflation can cause line entanglement, increased loads on early-opening parachutes, and potential canopy collisions. | |
| No Clustering | 2 | The system uses a single large parachute to handle the full recovery load instead of distributing it across several smaller chutes. | Simpler deployment system; avoids synchronization issues; | Harder to fabricate, rig, and handle; lower redundancy; greater consequences if the parachute fails | Higher risk of complete failure in case of malfunction or misdeployment of the single canopy. | |

## Supersonic Deployment Requirement

| File (2) | Source | Description | Pros | Cons | Risks | Diagram / Image |
|---|---|---|---|---|---|---|
| Subsonic Deployment | 2 | Systems that require deployment below Mach 1. Simplifies deployment environment. | Less demanding materials | May constrain trajectory profile | Limits coast or reentry time flexibility | |
| Supersonic Deployment | 2 | Systems that can deploy at Mach 1+ speeds. Enables high-speed missions and reduces coast time. | Higher entry speed possible | Requires high strength and thermal protection | Risk of deployment failure or structural failure |  FIGURE 5-89. Supersonic Flow Around a Vehicle-Parachute System. |

## Flat-Spin

| File (2) | Source | Description | Pros | Cons | Risks | Diagram / Image |
|---|---|---|---|---|---|---|
| Flat Spin | 3 | The recovered body is aerodynamically unstable, causing it to enter a controlled tumble or spin during descent. This increases drag and reduces speed. | Reduces ballistic coefficient; lowers parachute inflation loads; no need for active stabilization. | Difficult to control trajectory precisely; spin may damage sensitive internal components; Requires a deployment mechanism for the recovery system. | - |  |
| No Flat Spin | 2 | The recovered body is aerodynamically stabilized to prevent tumbling or spinning during descent, resulting in a more controlled reentry profile. | Predictable flight path; | Higher parachute inflation loads. | - | |

## Number of Stages

| File (2) | Source | Description | Pros | Cons | Risks | Diagram / Image |
|---|---|---|---|---|---|---|
| Multi-Stage | - | Uses at least two stages (e.g., drogue + main) to decelerate. | Optimized for speed control and safe landing. | More complex sequencing; more mass. | Deployment timing failure could lead to system loss. | |
| Single-Stage | - | A single recovery system handles deceleration and landing. | Simpler, fewer parts; less potential failure points. | May not handle high-speed or heavy payload well. | Too fast or too unstable descent; harder landing. | |

## Impact Attenuation Systems

| File (5) | Source | Description | Pros | Cons | Risks | Diagram / Image |
|---|---|---|---|---|---|---|
| Air Bags | 2 | Inflatable textile bags that absorb impact by controlled gas compression and venting. Stored deflated and inflated just before ground contact. | Reusable; compact storage; adjustable design; efficient energy absorption per unit weight. | Complex inflation and venting system; tuned to a specific energy level. | Over- or under-inflation can cause bounce, underperformance, or early termination of descent. | |
| Crushable Impact Attenuators | 2 | Use materials like aluminum honeycomb, foam, or paper that deform on impact to absorb energy. Often used for cargo drops and primarily absorb vertical energy. | Simple; no active systems; good for heavy payloads; lightweight for absorption effectiveness. | Not reusable; limited to one-time use; bulky when unpacked. | If material fails or is over-compressed, insufficient energy absorption may occur. | |
| No Impact Attenuation | - | No impact attenuation system is used; relies solely on parachute or aerodynamic braking for safe touchdown. | Simplifies recovery system; no extra mass or components. | High loads on structure; limits landing surface options. | High risk of structural damage or loss depending on descent speed and surface condition. | |
| Retrorocket | 2 | Use downward-facing rockets to decelerate the vehicle just before touchdown. Suited for high-mass landers requiring low impact acceleration. | High energy-to-weight ratio; allows very low impact velocity; deployable before ground contact. | Requires precise timing and ignition; adds mass and complexity. | Misfire or mistiming could lead to excessive impact velocity or hard landing. | |
| Skirt Jet Retrorocket | 2 | A retrorocket concept where slotted rocket tubes are arranged around the vehicle's skirt to direct thrust close to the ground, improving efficiency. Tested for the Apollo command module. | - | Experimental; less flight heritage; complex nozzle integration. | Poor thrust alignment could destabilize descent or induce tilt at touchdown. | |

## Buoyance Device

| File (2) | Source | Description | Pros | Cons | Risks | Diagram / Image |
|---|---|---|---|---|---|---|
| Buoyance Device | - | A dedicated system (e.g., inflatable bags or floatation structures) added to keep the rocket afloat after water landing. | Protects hardware from submersion. | Adds weight and complexity; requires deployment or inflation system. | Failure to deploy or puncture may result in sinking or total loss. | |
| No Buoyance Device | - | The rocket structure is designed to be naturally buoyant without additional flotation mechanisms. | Simpler and lighter; no separate system needed. | Limited buoyancy margin; may not perform well in rough sea states. | Risk of partial submersion or tipping, making recovery difficult. | |

## Summary and Transition

The design space established in this section includes **137 distinct options** distributed across **22 decision categories**, covering a wide range of architectural strategies, enabling technologies, and

mission framing choices. This extensive mapping reflects the diversity of potential recovery approaches and ensures that no viable concept is prematurely excluded from consideration.
While the **level of detail varies** between categories—depending on perceived feasibility, historical relevance, and available information—all options were retained at this stage to promote **completeness and traceability**. Some paths are more mature and better understood; others are exploratory or speculative, but included to preserve openness in the early phases of system definition.
In the sections that follow, this design universe will be **progressively refined** through feasibility screening (Tier 1), performance-based evaluation (Tier 2), and synthesis of recommended architectures. The goal is to transition from this broad exploratory baseline to a focused set of **justified and robust architecture options** aligned with project needs and constraints.

## 6.5 Selection Criteria Definition

### Purpose of Selection Criteria

Selection criteria provide a **structured and traceable basis** for comparing design options throughout the trade study. They enable consistent evaluation and justification by translating stakeholder needs and abstract project objectives into measurable metrics.
By defining criteria early, the study ensures that filtering and scoring are applied systematically, allowing for consistent comparisons and well-documented decisions—especially important in early-phase studies with limited time and modeling resources.

### Stakeholder Involvement and Derivation Process

The selection criteria were developed through early discussions with systems engineers from the EPFL Rocket Team, who served as the primary stakeholders for this study. Their goals and concerns—such as minimizing development risk— were formally captured in the Project Requirements Document and used as the foundation for defining evaluation metrics.
Qualitative objectives like **“reduce development complexity”** were transformed into more operational proxies such as **Technology Readiness Level (TRL)**, allowing for easier and more consistent evaluation. In parallel, performance-driven criteria such as **system mass**, **volume**, and **drift distance** were included to assess the functional viability and effectiveness of candidate architectures.
Some selection criteria exhibit clear dependencies or partial overlaps—for example, multiple metrics may reflect different aspects of system complexity. These interrelations were acknowledged during the definition process, but no attempt was made to de-correlate or consolidate them. The criteria were instead treated as complementary indicators, each offering a distinct perspective on stakeholder priorities or design implications. This approach favors transparency and traceability, even at the cost of introducing some redundancy.
The criteria were reviewed during academic supervision meetings and formal milestone reviews. While no formal validation mechanism was used, this feedback loop helped clarify definitions and ensured alignment with stakeholder expectations. Explicit weighting of criteria was not implemented in this iteration of the trade study but may be introduced in future phases.

### Rationale for a Multi-Tier System

A single-tier evaluation framework was initially considered but quickly dismissed due to its practical limitations. Evaluating every technology or architecture with full modeling would have required disproportionate effort, while relying solely on qualitative scoring would have reduced the precision and credibility of later-stage decisions.

Instead, a **two-tier evaluation system** was adopted to balance analytical rigor with time and resource constraints. **Tier 1** enables rapid screening of design options using qualitative or easily assessable criteria, allowing clearly unsuitable or infeasible solutions to be eliminated early. **Tier 2** focuses on a reduced set of viable architectures, applying detailed modeling, simulation, and optimization to compare their performance.
This approach avoids unnecessary modeling work on poor candidates while preserving in-depth evaluation for high-potential solutions, ensuring both efficiency and decision quality.

### Definition of Tier 1 Criteria

**Tier 1 criteria** serve as a first-level filter for individual technologies and design options. They are qualitative or low-effort metrics that assess fundamental feasibility, maturity, and resource requirements. These criteria were applied at the **individual technology level**, not at the level of full system architectures.
The primary goal of Tier 1 evaluation is to eliminate clearly unsuitable or low-potential options before committing effort to detailed modeling. This includes technologies that: - Violate project constraints (e.g., too immature or costly). - Offer significantly lower performance relative to alternatives. - Impose high development complexity or integration risk.

**Cutoff Logic** Not all Tier 1 criteria were applied uniformly across every option. Instead, a **cutoff-based logic** was used: - If a **single Tier 1 criterion** scored below an acceptability threshold, the technology was typically **removed from further study**. - These thresholds were set based on engineering judgment: - Some were **strict and binary** (e.g., TRL too low to ensure reliability). - Others were **relative or comparative** (e.g., significantly heavier than competing options).
This screening process ensured that resources were focused on options that were feasible, promising, and aligned with project needs.

## Tier 1 Selection Criteria

| Criteria | Value Type | Description | Justification | Applied To |
|---|---|---|---|---|
| TRL (Technology Readiness Level) Outside of The EPFL Rocket Team | Qualitative (e.g., TRL 1-9) | Maturity level of the technology based on external developments. | Ensures that the technology is sufficiently mature and proven outside of the team. Reduces risk of adopting unproven solutions. | Technology |
| TRL (Technology Readiness Level) at The EPFL Rocket Team | Qualitative (e.g., TRL 1-9) | Internal maturity level based on prior experience within the team. | Accounts for the team's experience and familiarity with the technology, impacting development effort and risk. | Technology |
| Development Complexity | Qualitative (High / Medium / Low) | Estimated complexity and effort required to develop the technology. | High development complexity increases schedule and resource risks. | Technology |
| COTS Solution Availability | Qualitative (Yes / Partial / No) | Availability of Commercial Off-The-Shelf solutions for the technology. | Presence of COTS solutions can provide a low-risk backup option or reduce development effort if needed. | Technology |
| Weight Efficiency (Relative) | Relative Ranking (1st, 2nd, . . . ) | Relative ranking of weight efficiency across all evaluated technologies. | Weight directly impacts rocket performance and recovery system effectiveness. Helps eliminate excessively heavy solutions. | Technology |
| Volume Efficiency (Relative) | Relative Ranking (1st, 2nd, . . . ) | Relative ranking of volume efficiency across all evaluated technologies. | Volume constraints are critical in small-scale rocketry. This criterion helps filter out bulky solutions early. | Technology |
| Development Cost Estimate | Qualitative (High / Medium / Low) | Estimated development cost of the technology. | Allows early filtering of technologies that would require unsustainable development budgets or effort. | Technology |

### Definition of Tier 2 Criteria

**Tier 2 criteria** are applied at the **system architecture level**, once candidate technologies have passed Tier 1 screening and have been assembled into full recovery concepts. These criteria evaluate the combined performance, feasibility, and complexity of integrated systems—each consisting of a set of interacting technologies, design choices, and operating assumptions.
Unlike Tier 1, which focuses on early-stage filtering, Tier 2 introduces **both quantitative and qualitative metrics** that require in-depth modeling, estimation, or system-level analysis. The evaluation of Tier 2 criteria is supported by a dedicated **performance modeling tool**, introduced in a later section, which allows simulation of recovery behavior and estimation of key performance metrics.
Tier 2 evaluation is used to: - Compare **complete system architectures** on the basis of operational feasibility and performance. - Prioritize systems that best balance stakeholder goals (e.g., safety, reliability, cost, scalability). - Justify the selection of a small set of preferred architecture(s) for recommendation.

### Tier 2 Selection Criteria

| Criteria | Value Type | Description | Justification | Applied To |
|---|---|---|---|---|
| System TRL Outside of The EPFL Rocket Team | Qualitative (e.g., TRL 1-9) | Maturity level of the overall system concept based on external developments. | Ensures that the system concept is based on technologies with demonstrated external maturity, reducing development risk. | System |
| System TRL at The EPFL Rocket Team | Qualitative (e.g., TRL 1-9) | Internal maturity level of the system concept based on the team's experience and past projects. | Accounts for the team's experience and familiarity with the technologies and system as a whole, impacting development effort and risk. | System |
| Development Complexity | Qualitative (e.g., High / Medium / Low) | Estimated technical and organizational complexity required to develop the full system. | High development complexity increases schedule and resource risks. | System |
| Count of Single Points of Failure | Quantitative (e.g., number of SPOFs) | Number of failure modes that would result in total mission failure without redundancy. | Helps identify systems that are inherently less reliable and require additional mitigation measures. | System |
| Mass Estimation | Quantitative (e.g., kg) | Estimated total mass of the recovery system integrated into the rocket. | Mass impacts overall rocket performance and payload capacity; lighter systems are preferred. | System |
| Volume Estimation | Quantitative (e.g., $m^3$) | Estimated total volume occupied by the recovery system in the rocket. | Volume constraints affect the integration and design flexibility of the rocket airframe. | System |
| Opening Forces / Deployment Forces | Quantitative (e.g., N) | Maximum force applied on the system and rocket structure during deployment. | Excessive forces can damage the rocket structure or recovery system, impacting safety and performance. | System |
| Touchdown Speed | Quantitative (e.g., m/s) | Estimated impact speed of the rocket at landing with the recovery system deployed. | Determines the risk of hardware damage at landing and the overall safety of recovery operations. | System |
| Environmental Adaptability | Qualitative (High / Medium / Low) or Quantitative (Safety Margins) | Capability of the system to perform reliably under varying environmental conditions (wind, temperature, etc.). | Ensures system performance is maintained in real flight conditions, reducing operational risk. | System |

| **Criteria** | **Value Type** | **Description** | **Justification** | **Applied To** |
|---|---|---|---|---|
| Growth Potential | Quantitative (e.g., System Mass & Volume as a function of Payload Mass) | Flexibility of the system to be adapted or scaled with different payload requirements. | Promotes long-term usability and scalability of the system across future missions. | System |
| Risk and Safety Assessment | Qualitative / Quantitative (Number & severity of risks) | Overall risk profile of the system, including operational and safety hazards. | Helps ensure that the selected system does not introduce unacceptable risks to mission success or safety. | System |
| Life Cycle Cost Estimate | Quantitative (e.g., CHF or $) | Total estimated cost of development, production, operation, and maintenance over the system's life cycle. | Enables cost-informed decision-making and helps avoid financially unsustainable solutions. | System |
| Development Duration Estimate | Quantitative (e.g., months) | Estimated time required to fully develop and qualify the system concept. | Ensures that the selected system can be developed within the project timeline constraints. | System |

## 6.6 Evaluation of Selection Criteria

This section presents the evaluation of design options based on the selection criteria defined earlier in the report. A two-tiered screening methodology was used:

- **Tier 1 (Basic Feasibility Screening)**
- **Tier 2 (Detailed Trade-Off Analysis)**

This progressive filtering approach helps streamline the decision process by focusing detailed effort on concepts that are already proven to be broadly feasible. For transparency, rejected options are listed with the criterion that caused rejection, along with a short rationale. Options that passed Tier 1 will be fully evaluated in Tier 2.

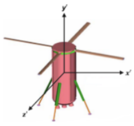
### Tier 1 Selection

### Mission & Architectural Choices

#### Mission Scope

| File (2) | Rejection Criteria | Rationale | Decision Status | Diagram / Image |
|---|---|---|---|---|
| Full Recovery | - | - | Selected | |
| Partial Recovery | - | - | Selected | |

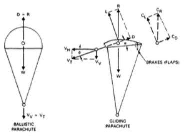
#### Decelerator Type

| File (3) | Rejection Criteria | Rationale | Decision Status | Diagram / Image |
|---|---|---|---|---|
| Aero Dynamic Decelerator | - | - | Selected | |
| Helicopter System Recovery | • TRL out<br>• Development Complexity | Concept is only documented in academic literature and lacks demonstrated implementation or existing prototypes | Rejected with Reason |  Fig. 4 Coordinate system of the first stage's body. Adapted from [3] |
| Propulsive Landing | • Development Complexity<br>• Development Cost | Requires complex control systems and is prohibitively expensive to qualify and acceptance test within the scope of this project. | Rejected with Reason | |

#### Landing Zone

| File (2) | Rejection Criteria | Rationale | Decision Status | Diagram / Image |
|---|---|---|---|---|
| Impact on Land | - | - | Selected | |
| Impact on Water | - | - | Selected | |

#### Aero Dynamic Decelerator Types

| File (5) | Rejection Criteria | Rationale | Decision Status | Diagram / Image |
|---|---|---|---|---|
| Inflatable Aerodynamic Decelerators | • Development Complexity<br>• Development Cost | Development of inflatable aerodynamic decelerators involves complex structural design, inflation control, and deployment sequencing. The associated cost and effort required to design, prototype, and test such systems exceed the scope and resources of this project. | Rejected with Reason |  |
| Maneuverable Parachute | • Weight Efficiency<br>• Development Complexity | The ability to guide the parachute is not required for this mission, and implementing maneuverability would add unnecessary weight and complexity due to additional control hardware and software. | Rejected with Reason |  |
| Rotating Parachute | • Requirement Compliance | Rotating parachutes are not compatible with large payloads and have limited reliability, making them unsuitable for meeting mission requirements. | Rejected with Reason | |
| Slotted Parachute | - | - | Selected | |
| Solid Textile Parachute | - | - | Selected | |

## Partial Recovery Types

| File (3) | Rejection Criteria | Rationale | Decision Status | Diagram / Image |
|---|---|---|---|---|
| Recover Nosecone and Avionics | - | - | Selected | |
| Recover only Avionics System | - Requirement Compliance | This option does not comply with the requirement to recover the nosecone, which contains the payload and must be retrieved for mission success. | Rejected with Reason | |
| Recover only Nosecone | - | - | Selected | |

## Full Recovery Types

| File (3) | Rejection Criteria | Rationale | Decision Status | Diagram / Image |
|---|---|---|---|---|
| Multi-Bodied Recovery | - | - | Selected | |
| No Rocket Separation | - | - | Selected | |
| Separate rocket but keep both bodies attached to a single recovery system | - Weight Efficiency<br>- Volume Efficiency | This configuration provides no clear advantage over unseparated recovery while introducing additional mass and volume penalties. | Rejected with Reason | |

## Driving Technology Choices

## Solid Textile Parachute Types (Source ID: 2)

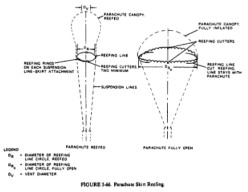
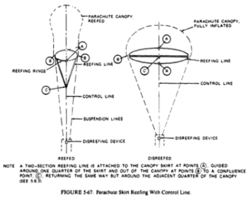
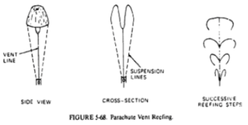
**Slotted Parachute Types** (Source ID: 2)

**Rotating Parachutes** (Source ID: 2)

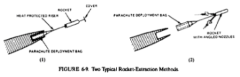
**Manoeuvrable (Gliding) Parachutes** (Source ID: 2)

## Set Modifiers

### Reefing

| File (7) | Rejection Criteria | Rationale | Decision Status | Diagram / Image |
|---|---|---|---|---|
| Continuous Disreefing | - TRL out<br>- Development Complexity | This concept has never been implemented in practice, indicating a low technology readiness level and high development uncertainty that exceed the project's scope. | Rejected with Reason | |
| Fixed Pocketband Reefing | - | - | Selected | |
| No reefing | - | - | Selected | |
| Skirt Reefing | - | - | Selected |  |
| Skirt Reefing with Control Line | - | - | Selected |  |
| Slider Reefing | - | - | Selected | |
| Vent Reefing | - | - | Selected |  |

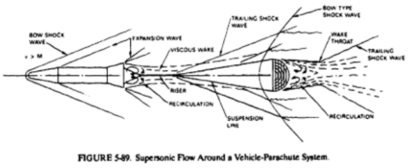
### Deployment Mechanism

| File (5) | Rejection Criteria | Rationale | Decision Status | Diagram / Image |
|---|---|---|---|---|
| Deployment by Rocket Extraction | - Development Cost | Deployment by rocket extraction requires precise timing and introduces additional risk. The use of small rocket engines adds development cost and complexity. | Rejected with Reason |  |
| Drogue Gun | - | - | Selected | |
| No Deployment Mechanism | - | - | Selected | |
| Parachute Mortar | - | - | Selected | |
| Static-Line Deployment | - TRL out | Static-line deployment has not been demonstrated in similar missions and lacks repeatability. | Rejected with Reason | |

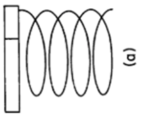
### Clustering

| File (2) | Rejection Criteria | Rationale | Decision Status | Diagram / Image |
|---|---|---|---|---|
| Clustering | - | - | Selected | |
| No Clustering | - | - | Selected | |

### Supersonic Deployment Requirement

| File (2) | Rejection Criteria | Rationale | Decision Status | Diagram / Image |
|---|---|---|---|---|
| Subsonic Deployment | - | - | Selected | |
| Supersonic Deployment | - | - | Selected |  |

### Flat-Spin

| File (2) | Rejection Criteria | Rationale | Decision Status | Diagram / Image |
|---|---|---|---|---|
| Flat Spin | - Development Complexity<br>- Requirement Compliance | - | Rejected with Reason |  |
| No Flat Spin | - | - | Selected | |

### Number of Stages

| File (2) | Rejection Criteria | Rationale | Decision Status | Diagram / Image |
|---|---|---|---|---|
| Multi-Stage | - | - | Selected | |
| Single-Stage | - | - | Selected | |

### Impact Attenuation Systems

| File (5) | Rejection Criteria | Rationale | Decision Status | Diagram / Image |
|---|---|---|---|---|
| Air Bags | - Weight Efficiency<br>- Volume Efficiency<br>- Development Complexity<br>- Development Cost | Airbags introduce significant mass and volume penalties and require complex inflation and sequencing systems. Their development and testing would also incur high costs, making them unsuitable given the project's constraints. | Rejected with Reason | |
| Crushable Impact Attenuators | - | - | Selected | |
| No Impact Attenuation | - | - | Selected | |
| Retrorocket | - Development Cost<br>- Development Complexity | Retrorockets require precise control, timing, and integration with the descent system. The associated development cost and complexity are excessive. | Rejected with Reason | |
| Skirt Jet Retrorocket | - Development Cost<br>- Development Complexity | Skirt jet retrorockets demand highly precise timing and pose additional technical risks. Their integration adds significant development complexity and cost, which are not justifiable. | Rejected with Reason | |

### Buoyance Device

| File (2) | Rejection Criteria | Rationale | Decision Status | Diagram / Image |
|---|---|---|---|---|
| Buoyance Device | - | - | Selected | |
| No Buoyance Device | - | - | Selected | |

### Tier 2 Selection

The Tier 2 selection process is focused on performing a fair and traceable comparison between fundamentally different recovery system architectures. Its purpose is to evaluate each architecture not based on arbitrary sizing, but on its **optimized configuration**, thereby enabling a higher-quality selection decision that reflects realistic performance trade-offs.

**Architecture Optimization** To support this process, a dedicated **system concept generator** has been developed. This software performs architecture-specific optimization using a genetic algorithm and a physics-based simulation. The tool automatically explores the design space for each candidate architecture, subject to constraints and objectives, to identify a **feasible and near-optimal configuration**.
The three architectures currently under evaluation are:

1. **Full recovery with no separation**
2. **Nosecone-only recovery**

3. **Full recovery using two distinct systems**

Each architecture is optimized independently. Once a best-performing configuration is identified for each, these optimized concepts will serve as the input to the Tier 2 selection.

**Selection Process** After optimization, the shortlisted architecture concepts will be evaluated using the Tier 2 selection criteria.

### Current Status

The architecture optimization runs are still in progress. The Tier 2 comparison will be completed once:

- Optimization has converged for all three architecture types
- Final scores and constraint compliance are validated
- System-level performance plots and metrics are compiled

# 7 Pararec-Sim — Software Design & Development Document

## 7.1 Introduction

This document records the **architecture, requirements, design choices and development strategy** for the minimum-viable product (MVP) of *Pararec-Sim*—a Python tool for sizing and analysing multi-stage parachute recovery systems. It supersedes separate "design" and "development" files: everything needed for the Preliminary Requirements Review (PRR) now lives here.

### Scope

The document covers the full MVP feature set. Interfaces intended for post-MVP extensions are noted, but detailed design of those future capabilities is out of scope.

## Needs

### Entire Software

| # | Need statement | Rationale |
|---|---|---|
| N-1 | Provide **physics-based simulation of parachute descent, drift and loads** for all planned flight regimes. | Engineers must verify performance & compliance at each design review. |
| N-2 | **Evolve with system maturity**: allow models (drag, mass, failure rates, CFD tables, etc.) to be swapped or refined without re-writing the whole program. | Keeps the tool relevant as the design is iterated and new test data appear. |
| N-3 | Support both **deterministic and probabilistic analyses** (Monte Carlo / reliability) once uncertainty data become available. | Range safety & certification require dispersion envelopes and confidence levels. |
| N-4 | Enable **batch studies & optimisation loops** driven by external scripts or built-in algorithms. | Weight minimisation, cut-timer tuning, reefing schedule optimisation. |
| N-5 | Offer **traceable input & output formats** with metadata (model version, commit ID, run settings) aligned to ECSS documentation practice. | Auditors need to reproduce analyses years later. |
| N-6 | Integrate with **external data sources and tools** (wind database, requirements database, flight-test telemetry, CFD results). | Eliminates manual data re-entry errors. |
| N-7 | Provide **programmable and human-friendly interfaces** (command-line, Python API, optional GUI) so that both automation and ad-hoc studies are easy. | Different user profiles: system engineers, research students, range analysts. |
| N-8 | Deliver **result visualisation & reporting hooks** (plots, CSV/JSON exports, dashboards) for rapid review and for inclusion in mission documents. | Speeds up decision cycles. |
| N-9 | Be **verifiable and validated**: include built-in test cases, unit tests and “golden flights” so that every code change can be confidence-checked. | Maintains credibility of results. |
| N-10 | Achieve **acceptable performance** (wall-clock time & memory) for hundreds to thousands of Monte Carlo cases on a laptop or small cluster. | Allows timely trade-offs during design reviews. |
| N-11 | Be **platform-agnostic and open-source-friendly** (compiles/runs on Linux, macOS, Windows; uses permissive dependencies) to ensure longevity and collaboration. | University teams and suppliers use mixed environments. |
| N-12 | Provide **extensive user and developer documentation** including an “interface control document” for every public API. | Lowers onboarding effort and enforces configuration control per ECSS-E-ST-40. |
| N-13 | Maintain a **documented Software Design Document** (SDD) and **Software Development Plan** (SDP) covering architecture, verification strategy, configuration management and release process. | Required by ECSS-Q-80 for software projects and critical for long-term maintainability. |

### First Implementation

| # | Need statement | Rationale for MVP |
|---|---|---|
| M-1 | Simulate **single-stage or multistage free-body and parachute descent** under vertical, axisymmetric motion with fixed drag coefficients. | Sufficient fidelity to discriminate major architecture options. |
| M-2 | Accept **simple, declarative input files** (e.g., YAML/JSON) describing vehicle mass, drag area, deployment altitude, wind-at-altitude table. | Quick to edit by non-software specialists. |
| M-3 | Compute and output **key performance metrics**: peak opening load, max dynamic pressure, landing speed, horizontal drift distance. | Directly support go/no-go trade criteria. |
| M-4 | Provide a **command-line interface** to run one case or a small parameter sweep from a shell script. | Allows batch studies without extra UI work. |
| M-5 | Produce **plain-text and CSV results** plus one summary plot (altitude vs. time) per run. | Easy to paste into trade-study spreadsheets and reports. |
| M-6 | Include **unit tests for fundamental equations** (drag force, atmosphere lookup) and one "golden case" regression test. | Ensures numerical stability while code is still small. |
| M-7 | Be **installable with one command** (e.g., `pip install pararec-sim-mvp` or a standalone ZIP) with minimal external dependencies. | Engineers can start using it within minutes. |
| M-8 | Expose **well-documented extension points** (e.g., abstract base class for a DragModel) even if only one concrete model exists today. | Sets the groundwork for later evolution without massive refactor. |
| M-9 | Ingest a **library of technology options** (parachute types, stage counts, reefing schemes, attachment hardware) including their design variables and any pre-defined performance correlations. | Lets the optimiser "pick & mix" building blocks when forming candidate architectures. |
| M-10 | Provide an **architecture-assembly engine** that can combine compatible technology options into complete system definitions according to simple compatibility rules supplied in the input file. | Automates creation of feasible design candidates; avoids hand-crafting every variant. |
| M-11 | Include a **generic optimisation driver** capable of exploring the discrete combinations and continuous design variables (e.g., genetic algorithm with user-settable population size & iterations). | Rapidly identifies promising architectures without exhaustive manual sweeps. |
| M-12 | Allow users to **declare objective functions and constraints** (e.g., minimise mass, constrain landing speed 6 m/s) in the same input file. | Keeps optimisation logic data-driven and avoids code edits when study criteria change. |
| M-13 | Output **ranked results and basic Pareto data** (objective values and constraint violations) in CSV/JSON plus an optional scatter-plot image. | Enables quick visual trade-off analyses and easy import into existing decision decks. |

| # | Need statement | Rationale for MVP |
|---|---|---|
| M-14 | Store **run metadata and random seeds** so that every optimisation can be reproduced exactly and resumed if interrupted. | Essential for traceability and incremental improvement during architecture down-selection. |
| M-15 | Permit **multiple independent optimisation cycles** (e.g., different random seeds or initial populations) to be executed automatically in one command. | Enables assessment of optimiser repeatability and robustness. |
| M-16 | After the cycles finish, **compute and report convergence / similarity metrics** (e.g., hyper-volume delta, best-objective spread, Pareto-set overlap) and flag significant discrepancies. | Gives users confidence that the identified optima or Pareto front are not artefacts of a single run. |
| M-17 | Perform **automatic input-file validation and compatibility checks** (e.g., parachute type must match reefing scheme, stage count 1). | Prevents wasted optimiser time on impossible or nonsensical designs. |
| M-18 | Offer a **one-at-a-time sensitivity-analysis mode** that perturbs each design variable ± and reports the resulting change in key metrics. | Quickly reveals which variables drive performance, guiding requirements refinement. |
| M-19 | Generate an **optional consolidated report** (Markdown or simple HTML) that bundles input summary, optimiser settings, top-N solutions, plots and convergence stats. | Saves manual work assembling review slides or documentation for down-selection meetings. |

**Requirements**

| Req ID | Requirement statement | Type | Derived from | Required for initial version? |
|---|---|---|---|---|
| **GS-01** | The software **shall simulate the vertical descent trajectory** of the recovery system from apogee to landing by numerically integrating the equations of motion. | F | N-1 | **Yes** |
| **GS-02** | The software **shall simulate horizontal drift** due to atmospheric winds and cross-range motion, providing time-history and final lateral displacement outputs. | F | N-1 | **Yes** |
| **GS-03** | The software **shall compute parachute loads** (opening force, steady-state drag force, line tension) throughout deployment(s) and descent. | F | N-1 | **Yes** |
| **GS-04** | The software **shall allow physics sub-models (e.g., drag, mass variation, failure-rate tables) to be added, replaced or removed at run-time** via a documented plug-in interface. | F | N-2 | **Yes** |
| **GS-05** | The software **shall execute Monte-Carlo analyses** in which any input parameter can be specified as a statistical distribution (normal, uniform or user-defined) and shall record the resulting distribution of key performance metrics. | F | N-3 | No |
| **GS-06** | The software **shall expose a public Python API that enables external scripts to launch batches of simulations and optimisation loops without spawning new processes.** | F | N-4 | No |
| **GS-07** | Every output file **shall embed traceability metadata** (software semantic version, Git commit hash, input-file checksum, model plug-in identifiers, date–time stamp). | NF | N-5 | **Yes** |
| **GS-08** | The software **shall import wind-profile data** from CSV files conforming to ICD-WIND-01. | F | N-6 | No |
| **GS-09** | The software **shall export simulation summaries** in JSON files conforming to ICD-RESULT-01. | F | N-6 | Yes |
| **GS-10** | The software **shall provide a command-line interface (CLI)** for running individual cases, sweeps and optimisations. | F | N-7 | Yes |
| **GS-11** | The software **shall provide a Python package API** that exposes the same capabilities as the CLI. | F | N-7 | No |
| **GS-12** | On user request the software **shall generate publication-quality plots** (PNG and SVG) of altitude-, velocity- and load-time histories. | F | N-8 | **Yes** |
| **GS-13** | The codebase **shall include an automated test suite (unit, integration, regression) that achieves 80 % statement coverage** when run with `pytest-cov`. | NF | N-9 | **Yes** |

| Req ID | Requirement statement | Type | Derived from | Required for initial version? |
|---|---|---|---|---|
| GS-14 | On a reference laptop (Intel i7-12700H, 16 GB RAM) the software shall complete 1 000 Monte-Carlo simulations of the baseline scenario in 10 minutes. | NF | N-10 | No |
| GS-15 | The software shall install and run on Windows 10+, Ubuntu 22.04 LTS and macOS 13+ using only OSI-approved-licence dependencies. | NF | N-11 | Yes |
| GS-16 | Each tagged release shall publish a user manual covering installation, basic usage and input-file schema. | NF | N-12 | Yes |
| GS-17 | The project shall maintain and version-control a Software Design Document (SDD) and a Software Development Plan (SDP) compliant with ECSS-Q-80. | NF | N-13 | Yes |

Initial Implementation

| Req ID | Requirement statement | Type | Derived from |
|---|---|---|---|
| MVP-01 | The MVP shall simulate vertical, axisymmetric descent of the recovery system. | F | M-1 |
| MVP-02 | The MVP shall support up to ten parachute stages in the descent simulation. (be more precise on differenec in between recovery stafe and unreeefing) | F | M-1 |
| MVP-03 | The MVP shall use fixed drag-coefficient values for every parachute stage. (maybe modify for super and subsonic?) | F | M-1 |
| MVP-04 | The MVP shall read one JSON scenario file per run. | F | M-2 |
| MVP-05 | The MVP shall validate the scenario file against schema input-v0.1 . | F | M-2 |
| MVP-06 | The MVP shall compute and record the peak opening load for each run. | F | M-3 |
| MVP-07 | The MVP shall compute and record the maximum dynamic pressure for each run. | F | M-3 |
| MVP-08 | The MVP shall compute and record the landing speed for each run. | F | M-3 |
| MVP-09 | The MVP shall compute and record the horizontal drift distance for each run. | F | M-3 |
| MVP-10 | The MVP shall provide the CLI command pararec-sim . | F | M-4 |
| MVP-11 | The CLI shall offer a run sub-command to execute a single scenario. | F | M-4 |
| MVP-12 | The CLI shall offer a sweep sub-command to execute a parameter sweep. | F | M-4 |
| MVP-13 | Each run shall output a plain-text summary file. | F | M-5 |
| MVP-14 | Each run shall output a CSV time-series file. | F | M-5 |

| Req ID | Requirement statement | Type | Derived from |
|---|---|---|---|
| MVP-15 | Each run shall output a PNG plot of altitude versus time. (and a drift over time, a dynamic pressure over and a vertical velocity over time, ) | F | M-5 |
| MVP-16 | Each run shall output a PDF document summarising the results. | F | M-5 |
| MVP-17 | The test suite shall include unit tests for drag-force computation. | NF | M-6 |
| MVP-18 | The test suite shall include unit tests for ISA atmosphere lookup. | NF | M-6 |
| MVP-19 | The test suite shall include a regression test verifying the golden-case altitude-time curve within ± 1 % RMS error. | NF | M-6 |
| MVP-20 | The MVP shall be installable via `pip install pararec-sim` with no compiled extensions. | NF | M-7 |
| MVP-21 | The MVP shall expose an abstract base class `DragModel` in the public namespace. | F | M-8 |
| MVP-22 | The MVP shall discover concrete `DragModel` implementations via Python entry-points. | F | M-8 |
| MVP-23 | The MVP shall read technology-option files in JSON format from a user-specified folder. | F | M-9 |
| MVP-24 | The MVP shall generate candidate system architectures by applying compatibility rules contained in a rule file. | F | M-10 |
| MVP-25 | The MVP shall include a built-in genetic-algorithm optimiser. | F | M-11 |
| MVP-26 | The built-in optimiser shall handle 50 discrete and 20 continuous design variables. | F | M-11 |
| MVP-27 | Users shall define optimisation objectives in a JSON file. | F | M-12 |
| MVP-28 | Users shall define optimisation constraints in a JSON file. | F | M-12 |
| MVP-29 | After optimisation, the MVP shall write `pareto.csv` containing the non-dominated solutions. | F | M-13 |
| MVP-30 | After optimisation, the MVP shall generate a PNG scatter plot of the first two objectives. | F | M-13 |
| MVP-31 | Each run shall write `run_meta.json` recording start/finish timestamps, random seed, git hash, input checksum and optimiser settings. | NF | M-14 |
| MVP-32 | When invoked with `--rep N`, the MVP shall execute N independent optimisation runs using unique random seeds. | F | M-15 |
| MVP-33 | The MVP shall compute the hyper-volume indicator for every Pareto set produced. | F | M-16 |
| MVP-34 | The MVP shall flag the optimisation as Not Converged if the maximum pairwise hyper-volume difference among runs exceeds 5 %. | F | M-16 |
| MVP-35 | The MVP shall abort with an explanatory message if any input file fails schema validation. | F | M-17 |
| MVP-36 | When invoked with `--sensitivity`, the MVP shall perturb each design variable by ± 5 % (or ± 1 for discrete). | F | M-18 |
| MVP-37 | The sensitivity run shall output a CSV table of resulting metric changes. | F | M-18 |
| MVP-38 | When invoked with `--report`, the MVP shall generate a Markdown file `report.md` that includes the input summary. | F | M-19 |
| MVP-39 | The generated report shall include optimiser settings and the top-5 ranked solutions. | F | M-19 |

| Req ID | Requirement statement | Type | Derived from |
|---|---|---|---|
| MVP-40 | The generated report shall include convergence metrics and embedded plots. | F | M-19 |
| MVP-41 | The MVP shall compute and record the total canopy + rigging + hardware mass for each candidate architecture, using the values defined in the technology-option library. | F | - |
| MVP-42 | The software shall estimate the packed volume of the recovery system for every candidate architecture. | F | - |
| MVP-43 | The software shall account for wake-interference drag reduction when multiple bodies are aligned in the flow. | F | - |
| MVP-44 | The software shall model the reduction of drag coefficient in the supersonic regime using user-selectable correlations. | F | - |
| MVP-45 | The software shall allow suspension-line length to modify the effective drag area through a configurable law. | F | - |
| MVP-46 | The software shall compute canopy opening time based on user-selectable inflation models and record the value in the results file. | F | - |

## 7.2 System Overview & Context

### Product Definition

Pararec-Sim is a command-line Python application that combines physics-based trajectory simulation with a genetic-algorithm optimiser. Its primary purpose is to size and evaluate multi-stage parachute recovery systems for a sounding rocket, reporting metrics such as opening load, landing speed and drift.

### External Context

| Actor / Data | Role in MVP |
|---|---|
| Systems engineer (end-user) | Launches the CLI, inspects results. |
| Wind-profile CSV | Supplies altitude-dependent wind data. |
| Super-system constraints JSON | Provides rocket-level properties. |
| Mission constraints JSON | Defines acceptance criteria. |
| Technology-option library JSON | Catalogue of parachute and reefing options. |
| Matplotlib | External library for plotting time histories. |

### Top-level Data Flow

1. Simulation path: Load scenario → validate → assemble architecture → simulate → compute metrics → write CSV/JSON, plots, report.
2. Optimisation path: Load optimisation config → iterative loop (assemble → simulate → evaluate → log) → final Pareto set and visuals.

### MVP Use-Cases

- Deterministic run: pararec-sim run <scenario.json>
- Parameter sweep: pararec-sim sweep <grid.json>
- Architecture optimisation: pararec-sim sweep optimise <study.json>
- Sensitivity analysis: pararec-sim sensitivity <scenario.json>

## 7.3 Static Architecture Overview

The Recovery Simulator repository is split into five logical layers, each residing in its own top-level package. Dependencies flow strictly downward (Figure 2), keeping the import graph acyclic and test-friendly.

1. Domain Model models/ Pure Python data classes that capture the physical state of the rocket and its environment: Environment (atmospheric profiles and look-ups), Section (an air-frame segment), SectionState (per-timestep kinematics) and RecoveryStage (parachute or ballistic phase). These files contain no I/O or plotting code.
2. Simulation Core sim/ The engine that advances the state in discrete time simulation.py drives the update loop and delegates bookkeeping to results.py, which can also render diagnostic plots. The simulator is deliberately stateless outside its parameters, enabling repeated calls from optimisation routines.
3. Optimisation Layer opti/ Search heuristics that wrap the simulator. A common evaluator.py scores each candidate design, while concrete strategies live in separate modules (genetic_optimizer.py

random_optimizer.py, etc.). config_builder.py translates JSON/YAML design files into the simulator's expected input schema.

4. Configuration config/ Declarative scenario data (design_variables.json, constraints.json, parachute database) plus the ConfigLoader utility that performs validation and casting. Keeping numbers out of code promotes reproducibility.
5. Presentation outputs/ Auto-generated artefacts: plots, logs and optimiser run folders. Nothing in the source tree imports from this directory.

The repository root contains an orchestration script, main.py, which performs the following pipeline:

1. Load scenario via ConfigLoader.
2. Instantiate an optimiser of choice, which repeatedly:
   - (a) converts the candidate vector with config_builder.py,
   - (b) calls sim.run_simulation(),
   - (c) receives a scalar fitness from evaluator.py.
3. Persist the best design and forward the run summary to sim.results for plotting.

main.py
↓
opti/
↓
sim/
↓
models/

Figure 2: Layered dependence only downward arrows exist.

## 7.4 Life-cycle Model & Development Plan

### MVP Phase (Spring 2025)

Development follows a compact, six-week V-model:

1. Requirements → Design: requirements baselined; architecture drafted.
2. Implementation: focussed weekly coding sprints.
3. Verification: unit and integration tests run continuously.
4. Release: MVP was supposed to be demonstrated at the PRR (week of 26 May 2025), however the developer ran out of time.

One primary developer drives the effort; weekly informal check-ins replace formal stage reviews, and this document is updated incrementally.

### Post-MVP Evolution (subject to continuation)

If the EPFL Rocket Team continues development, Pararec-Sim will adopt an evolutionary life-cycle aligned with rocket programme phases. Each semester delivers a minor release (v2.x, v3.x, . . . ), and feature branches merge only after an internal design review and automated quality checks.

### Configuration & Change Management

- Branch model: main (stable), dev (integration), feature branches per capability.
- Semantic versioning: major = rocket phase, minor = semester, patch = hot-fix.

- Every change is tracked by a GitHub issue linked to at least one requirement ID. Breaking interface changes require a formal change-request and may only ship in a minor release.

Future Work Disclaimer

Sections describing capabilities beyond the MVP serve only as planning place-holders; continuation depends on EPFL Rocket Team approval.

# 8 Conclusion

## 8.1 Summary of Results

The semester project resulted in the development of foundational tools and documentation to support future recovery system design activities for the EPFL Rocket Team. Key outcomes include:

- A first version of a custom simulator and architecture optimizer tailored to the evaluation of recovery system configurations. This tool provides a reusable framework that can be refined in later phases.
- An initial set of project and technical requirements for the recovery system, providing structure and traceability for future work packages.
- A documented methodology for system architecture trade-off analysis, aligned with ECSS practices, which can be applied or adapted for future subsystems and missions.

## 8.2 Outlook and Upcoming Work

Several critical tasks remain before a recovery system baseline can be formally proposed:

- Further debugging and validation of the simulator and optimizer are required to enable consistent architecture evaluation and convergence toward optimal solutions.
- A structured risk assessment and FMECA (Failure Modes, Effects, and Criticality Analysis) must be conducted to ensure the feasibility and safety of the candidate architectures.

These activities will form the bridge between this feasibility-phase work and the future preliminary design phase.

## 8.3 Limitations and Recommendations

The current phase did not culminate in the selection of a final baseline architecture. This limits the immediate usability of the study's outputs for the EPFL Rocket Team. Moving forward, two main pathways are available:

- Restart the trade-off process: using the established methodology and requirements but beginning anew with improved tools and clearer evaluation criteria.
- Iteratively complete the existing process: by refining the simulator and continuing from the current state to derive a decision.

Both paths are viable, but care should be taken to reduce complexity and ensure traceability of assumptions moving forward.

## 8.4 Self-Assessment and Autocritique

From a personal development perspective, the semester project was a success. It enabled learning in systems engineering practices, rocket recovery systems, and structured project documentation under ECSS guidance. The process of building and attempting to apply a custom simulator provided valuable practical experience.

However, from the standpoint of delivering a usable system architecture for the EPFL Rocket Team, the project fell short of expectations. The root cause of this outcome lies in scope mismanagement: the project attempted to simultaneously achieve the following:

1. Define and formalize system-level and project-level requirements.
2. Establish a trade-off methodology from scratch.
3. Explore and document a large recovery system design space.

4. Implement a custom simulation and optimization tool.

This ambitious scope resulted in rapid progression across multiple fronts, but incurred significant technical debt particularly in the traceability and maturity of architectural decisions. In hindsight, the project would have benefited from a stronger focus on one single objective, particularly the simulator development. Topics 1 3 could have been treated more lightly or deferred to parallel efforts.

Nevertheless, the groundwork laid by this project can serve as a valuable stepping stone if its outputs are carefully consolidated and extended.

# References

Antoine Truchot and Michael Fuser. 2025. $h_s e_m$ bse$_s$vn; 2025. URL.

Pedro Caldeira, Alain de Souza, and Filipe Szolnoky Cunha. Design of a helicopter system recovery for a sounding rocket. International Journal of Aeronautical and Space Sciences, 25:662 675, 01 2024. 10.1007/s42405-023-00696-z. URL https://www.researchgate.net/publication/377669310_Design_of_a_Helicopter_System_Recovery_for_a_Sounding_Rocket.

T W Knacke. Parachute Recovery Systems. Para Pub, 1992.

Mc Fall. Flat-spin recovery system, 1964. URL https://ntrs.nasa.gov/citations/19650025606.

Matthew J. Miller, Bradley A. Steinfeldt, and Robert D. Braun. Supersonic inflatable aerodynamic decelerators for use on sounding rocket payloads | aiaa scitech forum. AIAA SciTech Forum, 2023. 10.2514/mafm14;page:string:Article. URL https://arc.aiaa.org/doi/10.2514/6.2014-1092.

Estimating Mass of Inflatable Aerodynamic Decelerators Using Dimensionless Parameters, 2011. 8th International Planetary Probe Workshop 2011 (IPPW-8).

Brandon P. Smith, Christopher L. Tanner, Milad Mahzari, Ian G. Clark, Robert D. Braun, and F. McNeil Cheatwood. A historical review of inflatable aerodynamic decelerator technology development, 03 2010. URL https://ieeexplore.ieee.org/abstract/document/5447013.

Alexander W C and Lau R A. State-of-the-art study for high-speed deceleration and stabilization devices, 09 1966. URL https://ntrs.nasa.gov/citations/19660027566.

Comparison of Various Parachute Deployment Systems for Full Rocket Recovery of Sounding Rockets, 2019. EUCASS.

Ralph Rowland Young. The requirements engineering handbook. Boston, Mass. Artech House, 2004.

ECSS. Ecss-e-st-10c rev.1 system engineering general requirements, 2017.

ECSS. Technical requirements specification - ecss-e-st-10-06c, 2009.

Louis S Wheatcraft, Michael J Ryan, and Tami Edner Katz. INCOSE Needs and Requirements Manual. John Wiley Sons, 10 2024.

INCOSE. INCOSE Systems Engineering Handbook. Wiley, 07 2023.

Sebok. URL https://sebokwiki.org/wiki/Guide_to_the_Systems_Engineering_Body_of_Knowledge_(SEBoK).

Barraza R M. Parachute recovery systems design and development efforts expended on mercury-redstone booster and saturn s-1 stage, 07 1962. URL https://ntrs.nasa.gov/citations/19730061696.

Edward Fallon II, Anthony Taylor, and Robert Meyerson. Landing system design summary of the k-1 reusable launch vehicle | aerodynamic decelerator systems technology conferences. Aerodynamic Decelerator Systems Technology Conferences, 1999. 10.2514/MADS99;article:article:10.2514/6.1999-1720;subPage:string:Abstract;csubtype:string:Conference. URL https://arc.aiaa.org/doi/abs/10.2514/6.1999-1720.

Spears L T. A review of launch vehicle recovery studies, 07 1962. URL https://ntrs.nasa.gov/citations/19730061699.

Morgan Brookfield, Deron Decker, Harold Gilbert, David Moore, and Mark Rist. Preliminary design of the shuttle-c avionics recovery system, 06 1989. URL https://ntrs.nasa.gov/citations/19900004942

Clara O'Farrell, Chris Karlgaard, Jake A. Tynis, and Ian G. Clark. Overview and reconstruction of the aspire project's sr01 supersonic parachute test, 03 2018. URL https://ieeexplore.ieee.org/document/8396502

Lars Pepermans, Mark Rozemeijer, Esmée Menting, Noah Suard, and Sayyam Khurana. Systematic design of a parachute recovery system for the stratos iii student built sounding rocket. 2018 Atmospheric Flight Mechanics Conference, 06 2018. 10.2514/6.2018-3626.

J. Huh and S. Kwon. A practical design approach for a single-stage sounding rocket to reach a target altitude. *The Aeronautical Journal*, pages 1 17, 03 2022. 10.1017/aer.2022.18.

E Pounder, E Framan, and J Brayshaw. Jpl requirements for spacecraft landing and recovery, 07 1962. URL https://ntrs.nasa.gov/citations/19730061705.

Gillis C L. Deployable aerodynamic decelerators for space missions., 08 1969. URL https://ntrs.nasa.gov/citations/19690062553.

# ë]1æ¥' €º ¥¥› o

| dä¥ h! ' i | ¥»€ | ?1 M$ | ëM]¥$€o€M€¥ ¥$€ |
|---|---|---|---|
| X1È¥¥ | Ó¥ o¾e¥ oÓM Œ¥ ' 1 XMæŒ¥ ² æÓM€M]È¥€MX1È¥¥ 1Å ý § | ¥' Ó$ä M | • ï ¥' 1¬¥]¾ù¾e¥ Å] ù1Œ$› æ$È ï 1' ý¥€ |
| X1È¥¥ ¶ M]Èæ$ | Ó¥ M ' ¥X€MŒ¥ MX1È¥¥ 1¬¥]oŒ$›¥]\|oÓ11€ M]Èæ$ ² ä Œ¥›¥e¥] æ$¥› ›Œ]æ$È €Ó¥ X]1æ¥' €, €Mýæ$È æ$€1 M ' 1Œ$€X¥]Å1] M$' ¥ ¬M]æM€æ1$o M$› e¥' Ó$ä M ' 1$o€]M$€o§ | ¥' Ó$ä M | • X1È¥¥ |
| ?¥ ä¬¥]MŒ¥o X¥] ï ¥¬æ¥² | ' 1 X]¥Ó¥$oä¥ æ€1Å›¥ä¬¥]MŒ¥o ]¥[Œ]¥› Å1] ¥M Ó]¥¬æ¥² oÓM Œ¥ › 1' Œ ¥$€e¥› § | ë]1æ¥' € | • ï ¥¬æ¥² ë M$<br>• H7ùù 71 X äM$' ¥ |
| ?¥X¥$› MŒæ€¾¶ ¥€]ä | ?¥X¥$› MŒæ€¾ ¥€]ä o oÓM Œ¥ ¥o€MŒæÓ¥› › Œ]æ$È €Ó¥ X]1æ¥' €§ | ë]1æ¥' € | • T]M ¥ ÅÅÅ |
| ?¥o' 1Xæ$È ÅX€1$o | ë1€¥$€æM›¥o' 1Xæ$È 1X€1$o oÓM Œ¥ ä¥$€æÅ¥› M$› › 1' Œ ¥$€e¥› § | ë]1æ¥' € | • T¥' Ó$ä M M$› ë]1È]M ï ¥' 1 ¥$› M€1$o |
| ?¥¬¥ 1X ¥$€ 71 X¥»æ€¾ | Ó¥ ¥»X¥' €e¥› ›¥¬¥ 1X ¥$€ ' 1 X¥»æ€¾oÓM Œ¥ €Mý¥$ æ$€1 M ' 1Œ$€›Œ]æ$È €Ó¥ o¾e¥ o¥ ¥' €1$§ | ë]1æ¥' € | • T]M ¥ ÅÅÅ<br>• ù€Œ ¥$€Å]ÈM$æM€1$ 71$o€]M$€o |
| H7ùù 71 X äM$' ¥ | 1Å€Ó¥ › 1' Œ ¥$€M€1$ X]¥XM]¥› oÓ1Œ› Mä €1 Œ¥ H7ùù ' 1 X äM$€§ | ë]1æ¥' € | • «MŒ$' Óù æ¥ ?1' Œ ¥$€M€1$<br>• ë]¥ ä æ$M]¾ù€Œ ¾ |
| Ho€ä M¥› ¶ 1$¥€M]¾ï M$È¥ | $ ¥o€ä M¥› ŒŒ È¥€]M$È¥ Å] €Ó¥ ' Ó1o¥$ o¾e¥ ' 1$' ¥X€oÓM Œ¥ ¥o€MŒæÓ¥› › Œ]æ$È €Ó¥ X]1æ¥' € €1 ÈŒ¥ ¥Å$M$' æM X M$$æ$È M$› › ¥' æo1$| Mýæ$È§ | ë]1æ¥' € | • ù¥ ¥' €e¥› ù¾e¥ ?¥o' ]æX€1$<br>• ¶ 1$¥€M]¾2Œ È¥€ |
| H»äæ$È T¥' Ó$1 1È¾o | Ó¥ ¥' Ó$1 1È¾ï ¥M æ$¥oo «¥¬¥ h ï «i 1Å]¥ ¥¬M$€¥»ææ$È e¥' Ó$1 1È¾o oÓM Œ¥ ¥¬MŒM¥› › Œ]æ$È o¾e¥ ' 1$' ¥X€o¥ ¥' €1$§ | ë]1æ¥' € | • T]M ¥ ÅÅÅ |
| H»X¥' €e¥› ë¥]Å1] M$' ¥ | Ó¥ ¥»X¥' €e¥› X¥]Å1] M$' ¥ 1Å€Ó¥ ' Ó1o¥$ o¾e¥ oÓM Œ¥ ¥o€MŒæÓ¥› › Œ]æ$È €Ó¥ X]1æ¥' €§ | ë]1æ¥' € | • ù¥ ¥' €e¥› ù¾e¥ ?¥o' ]æX€1$ |
| dM ææ€¾7 1$o€]M$€ | Ó¥ X]1æ¥' € oÓM €Mý¥ æ$€1 M ' 1Œ$€ ' 1$o€]M$€o æ$ý¥› €1 M¬MæMŒ¥ ÅM ææ€¾o§ | ë]1æ¥' € | • ù€Œ ¥$€Å]ÈM$æM€1$ 71$o€]M$€o |
| ý$€¥]ÅM ¥ ý ¥$€Åå M€1$ | ý$€¥]ÅM ¥o ² æÓ1€Ó¥] oŒæŒ¾o€¥ o oÓM Œ¥ ä¥$€æÅ¥› Å1] €Ó¥ o¥ ¥' €e¥› o¾e¥ ' 1$' ¥X€M$› › 1' Œ ¥$€e¥› § | ë]1æ¥' € | • ù¥ ¥' €e¥› ù¾e¥ ?¥o' ]æX€1$ |
| «MŒ$' Óù æ¥ 71 XMæŒæ€¾ | Ó¥ ' Ó1o¥$ o¾e¥ oÓM Œ¥ ' 1 XMæŒ¥ ² æÓ €Ó¥ X1€¥$€æM MŒ$' Óo¥o ² Óä ÓÓM¬¥ M]¥M ¾Œ¥¥$ ä ¥$€æÅ¥› § | ¥' Ó$ä M | • ï ¥' 1¬¥]¾ù¾e¥ Å] ù1Œ$› æ$È ï 1' ý¥€ |
| «MŒ$' Óù æ¥ ?1' Œ ¥$€M€1$ | Å$¥ 1Å€Ó¥ æ¥$¥$› ¥› MŒ ä$' ¥o Å1] €Ó¥ › 1' Œ ¥$€M€1$ ' 1Œ› Œ¥ MŒ$' Óo¥ 1ÅÅä äMo ]¥¬æ¥² æ$È €Ó¥ o¾e¥ ›¥oæÈ$§ | ë]1æ¥' € | • «MŒ$' Óù æ¥ 71 XMæŒæ€¾ |
| ¶ 1$¥€M]¾2Œ È¥€ | Mä Œ 1$¥€M]¾ŒŒ È¥€' 1$o€]M$€oÓM Œ¥ ä¥$€æÅ¥› M$› › 1' Œ ¥$€e¥› § | ë]1æ¥' € | • ù€Œ ¥$€Å]ÈM$æM€1$ 71$o€]M$€o |
| ë¥]Å1] M$' ¥ ¶ ¥€]ä o | ë¥]Å1] M$' ¥ ¥€]ä o Å1] €Ó¥ o¾e¥ oÓM Œ¥ ›¥Å$¥› M$› › 1' Œ ¥$€e¥› › Œ]æ$È €Ó¥ X]1æ¥' €§ | ë]1æ¥' € | • T]M ¥ ÅÅÅ<br>• H»X¥' €e¥› ë¥]Å1] M$' ¥<br>• ù¥ ¥' €e¥› ù¾e¥ ?¥o' ]æX€1$ |
| ë]¥ ä æ$M]¾ù€Œ ¾ | Ó¥ X]1æ¥' € oÓM Å' Œ ¥»' Œä ¥¾1$ €Ó¥ X]¥ ä æ$M]¾o€Œ ¾1Å€Ó¥ o¾e¥ § | ë]1æ¥' € | • ï ¥' 1¬¥]¾ù¾e¥ Å] ù1Œ$› æ$È ï 1' ý¥€ |
| ï ¥' 1 ¥$› M€1$ Å] dŒŒ]¥ W1]ý | ï ¥' 1 ¥$› M€1$o Å] ÅŒŒŒ]¥ ² 1]ý XM ýMÈ¥o oÓM Œ¥ X]1¬ä ¥› § | ë]1æ¥' € | • ë]¥ ä æ$M]¾ù€Œ ¾ |
| ï ¥' 1¬¥]¾ù¾e¥ Å] ù1Œ$› æ$È ï 1' ý¥€ | Ó¥ o€Œ ¾oÓM Å' Œ1$ ä ¥$€Å¾æ$È X1€¥$€æM o¾e¥ \| ¥¬¥ o1 ŒæM1$o Å] €Ó¥ ]¥' 1¬¥]¾o¾e¥ 1ÅM o1Œ$› æ$È ]1' ý¥€§ | ë]1æ¥' € | \| |
| ï ¥ŒæMŒ¥ ù1Å€² M]¥ | ýÅM$¾o1Å€² M]¥ æ›¥¬¥ 1X¥›¦ æ€Ó1Œ› Mä €1 Œ¥ ]¥ŒæMŒ¥ h¥¥ H7ùù\|î \|ù \|( 7 i§ | ë]1æ¥' € | • ë]¥ ä æ$M]¾ù€Œ ¾ |
| ï ¥¬æ¥² ë M$ | ]¥¬æ¥² X M$ oÓM Œ¥ ¥o€MŒæÓ¥› Å1] €Ó¥ X]1æ¥' €§ | ë]1æ¥' € | \| |
| ï æý TÓ]¥oÓ1 › | o XM]€1Å€Ó¥ o¾e¥ ' 1$' ¥X€o¥ ¥' €1$¦ €Ó¥ X]1æ¥' € €oÓM ›¥Å$¥ M ' ¥X€MŒ¥ ]æý €Ó]¥oÓ1 › o M$› › 1' Œ ¥$€€Ó¥ ² æÓ1$ €Ó¥ ]æý M$M¾oæo§ | ë]1æ¥' € | • ùMÅ¥€¾<br>• T]M ¥ ÅÅÅ |
| ùMÅ¥€¾ | ùMÅ¥€¾oÓM Œ¥ ' 1$oä ¥]¥› M€M ¥¬¥ o 1Å€Ó¥ X]1æ¥' €§ | ë]1æ¥' € | • ë]¥ ä æ$M]¾ù€Œ ¾ |
| ùMÅ¥€¾M$› p MËM]› $M¾oæo | oMÅ¥€¾M$› ÓMËM]› M$M¾oæo oÓM Œ¥ ' 1$› Œ e¥› Å1] €Ó¥ o¥ ¥' €e¥› o¾e¥ ' 1$' ¥X€§ | ë]1æ¥' € | • ùMÅ¥€¾ |
| ùMÅ¥€¾7 ]æ€¥]æM | ùMÅ¥€¾oÓM Œ¥ æ$' ŒŒ ¥› M$› M ' ]æ€¥]æM æ$ €Ó¥ €]M ¥\| 1ÅÅM$M¾oæo Å1] o¾e¥ ' 1$' ¥X€o¥ ¥' €1$§ | ë]1æ¥' € | • ùMÅ¥€¾<br>• T]M ¥ ÅÅÅ |
| ùMÅ¥€¾¶ ¥€]ä | ùMÅ¥€¾ ¥€]ä hoi oÓM Œ¥ ¥o€MŒæÓ¥› › Œ]æ$È €Ó¥ X]1æ¥' €§ | ë]1æ¥' € | • T]M ¥ ÅÅÅ |
| ù¥ ¥' €e¥› ù¾e¥ ?¥o' ]æX€1$ | Å$' ¥ Mo¾e¥ ' 1$' ¥X€Óo Œ¥¥$ o¥ ¥' €e¥› ¦ æ€o ›¥oæÈ$ oÓM Œ¥ Å]Œ€Ó¥] ›¥¬¥ 1X¥› § | ë]1æ¥' € | • ë]¥ ä æ$M]¾ù€Œ ¾ |
| ù€M¥ 1ÅTÓ¥ ]€ | o€M¥ 1Å€Ó¥ M]€M$M¾oæo oÓM Œ¥ X¥]Å1] ¥› €1 €È¥$¥]M¥ X1€¥$€æM o¾e¥ ' 1$' ¥X€o§ | ë]1æ¥' € | • ù¾e¥ 71$' ¥X€o 71$oä ¥]¥› |
| ù€Œ ¥$€Å]ÈM$æM€1$ 71$o€]M$€o | 71$o€]M$€o ]¥ M¥› €1 €Ó¥ o€Œ ¥$€1]ÈM$æM€1$ oÓM Œ¥ ä¥$€æÅ¥› M$› › 1' Œ ¥$€e¥› § | ë]1æ¥' € | • T]M ¥ ÅÅÅ |
| ù¾e¥ 71$' ¥X€o 71$oä ¥]¥› | Ó¥ o€Œ ¾oÓM Œ¥ Èæ$ ² æÓ M €Œ1 o' 1X¥ €1 ' 1$oä ¥] M² ä ¥]MÈ¥ 1Å¾e¥ ' 1$' ¥X€o§ | ë]1æ¥' € | • ë]¥ ä æ$M]¾ù€Œ ¾ |
| ù¾e¥ ¶ 1›¥ æ$È | X]¥ ä æ$M]¾o€æ$È M$› X¥]Å1] M$' ¥ ¥M›Œ]¥ ¥$€ 1›¥ oÓM Œ¥ X]¥XM]¥› Å1] €Ó¥ ' Ó1o¥$ o¾e¥ ' 1$' ¥X€§ | ë]1æ¥' € | • H»X¥' €e¥› ë¥]Å1] M$' ¥<br>• T]M ¥ ÅÅÅ |
| T¥' Ó$ä M M$› ë]1È]M ï ¥' 1 ¥$› M€1$o | ¥' Ó$ä M M$› ë]1È]M ]¥' 1 ¥$› M€1$o Å] ÅŒŒŒ]¥ ² 1]ý oÓM Œ¥ X]1¬ä ¥› § | ë]1æ¥' € | • ï ¥' 1 ¥$› M€1$ Å] dŒŒ]¥ W1]ý<br>• W1]ý ëM ýMÈ¥<br>• W1]ý 2]¥Mý› 1² $ ù€]Œ€Œ]¥ |
| T¥' Ó$ä M 71$o€]M$€o | Ó¥ o¾e¥ o e¥' Ó$ä M ' 1$o€]M$€o oÓM Œ¥ ' 1 X äM$€² æÓ €Ó1o¥ ä X1o¥› 1$ €Ó¥ X]¥ ä¥Ó1]¦ M]¥M ¾» ¥¬¥ 1X¥›¦ oXM ¥\| oÓ1€o¾e¥ o§ | ¥' Ó$ä M | • ï ¥' 1¬¥]¾ù¾e¥ Å] ù1Œ$› æ$È ï 1' ý¥€ |
| T¥' Ó$ä M º ¥¥› o | 1Å€Ó¥ o¾e¥ o e¥' Ó$ä M $¥¥› o oÓM Œ¥ ¥o€MŒæÓ¥› › Œ]æ$È €Ó¥ X]1æ¥' €§ | ë]1æ¥' € | • ë]¥ ä æ$M]¾ù€Œ ¾ |
| T¥o€«MŒ$' ÓÅXX1]€Œ$æ€¾ | ý€æ X M$$¥› €1 ÓM¬¥ M MŒ$' Ó 1XX1]€Œ$æ€¾² æÓ MX1È¥¥ 1Å! ý æ$ %§ Ó¥ MŒ$' Ó1XX1]€Œ$æ€¾ oÓ1Œ› Œ¥ ' 1$oä ¥]¥› Å1] o¾e¥ ¬¥]æÅ' M€1$§ | ë]1æ¥' € | • ù€Œ ¥$€Å]ÈM$æM€1$ 71$o€]M$€o |
| Tä ¥ 71$o€]M$€ | Ó¥ o¾e¥ ' 1$' ¥X€o¥ ¥' €1$ oÓM €Mý¥ æ$€1 M ' 1Œ$€€ä ¥ ' 1$o€]M$€o¦ ¥$oŒ]æ$È €ÓM€€Ó¥ ¥»X¥' €e¥› ›¥¬¥ 1X ¥$€€ä ¥ ¥$› æ$ ' 1 XMæŒ¥ ² æÓ €Ó¥ 1¬¥]M ]1' ý¥€€M o' Ó¥› Œ¥§ | ë]1æ¥' € | • ù€Œ ¥$€Å]ÈM$æM€1$ 71$o€]M$€o<br>• T]M ¥ ÅÅÅ |
| T]M ¥ ÅÅÅ | Ó¥ Œ¥o€o¾e¥ ' 1$' ¥X€oÓM Œ¥ o¥ ¥' €e¥› €Ó]1ŒÈÓ M€]M ¥\| 1ÅÅM$M¾oæo§ | ë]1æ¥' € | • ù¾e¥ 71$' ¥X€o 71$oä ¥]¥› |

## Project Requirements

https://sscspace.com/wp-content/uploads/Esrange-Safety-Manual.pdf