INSTITUTO POLITÉCNICO
DO CÁVADO E DO AVE
ESCOLA SUPERIOR
DE TECNOLOGIA

First Phase Report

Object-Oriented Programming

Rodrigo Ferreira Moura-27995

Introduction

Purpose of the Space System

The space system is designed as a comprehensive platform for exploring, simulating, and understanding a variety of space-related phenomena. Built in C#, it serves as an accessible tool for both educational and professional use in fields related to astronomy, astrophysics, and space exploration.

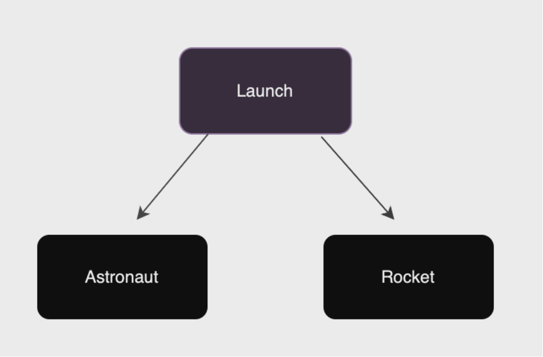
This project is a comprehensive simulation of a space agency's operations, structured in C# to model and manage various aspects of space missions. It includes classes and interfaces that represent astronauts, rockets, missions, and space stations, all organized in a modular format to allow for clear expansion and adaptability.

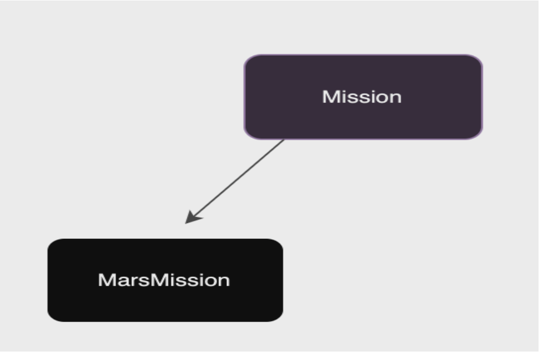
The key components of the system are:

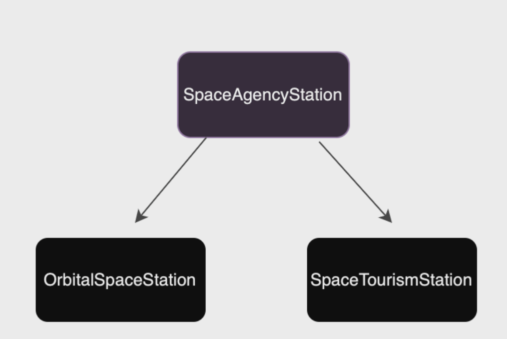
- **Launch:** Includes classes for astronauts and launch processes, as well as interfaces to handle launchable items and missions, setting a strong foundation for mission planning and execution.
- **Mission:** Contains classes representing different types of missions, such as general missions and specific ones like Mars missions.
- **SpaceAgencyStation**: Models different types of space stations, including orbital and tourism-focused stations, which adds realism to the simulation by showing the diverse facilities and purposes of a modern space agency.

With this structure, the project provides a scalable way to simulate real-world space agency activities, from astronaut training and launch operations to mission-specific planning and the management of space stations. It is designed to be easily extensible, allowing for future additions such as new mission types, additional launchable assets, and expanded space agency functionalities.

Diagrama de Estruturas de Classes







Implementations and Key Concepts Used in the Space Agency Project

This project uses key programming concepts to create a well-organized and expandable simulation of space agency operations. Here's an overview of the main ideas:

1. **Modular Design**:
   The project is divided into folders and classes like **`Launch`**, **`Mission`**, and

SpaceAgencyStation. Each section has a clear purpose, making the code easier to understand and expand.

2. **Interfaces for Flexibility**:
   Interfaces, like `ILaunchable` and `ILaunchableMission`, define shared behaviors that different classes can use. For example, both rockets and missions can be “launchable” because they follow the same rules set by these interfaces. This makes it easy to add new launchable items in the future.

3. **Encapsulation**:
   Each class (e.g., `Rocket`, `SpaceStation`) has its own data and functions. This keeps complex behaviors contained within each class, which makes it easier to troubleshoot and update.

4. **Inheritance**:
   Some classes are special versions of others. For instance, `MarsMission` inherits from `Mission`, adding Mars-specific details, and `OrbitalSpaceStation` is a type of `SpaceStation`. This allows us to create specific versions without rewriting everything.

5. **Organized File Structure**:
   Each part of the project has its own folder (e.g., `Launch`, `Mission`), which makes it easy to navigate and keep things organized.

6. **Clear Documentation**:
   The project includes a "first phase report.pdf" in the documents folder. This documentation tracks progress and provides useful information for anyone who works on or reviews the project.

7. **Easy to Expand**:
   The project's structure allows for new missions, types of stations, or other features to be added without major changes. For example, adding a new type of space station would be simple because of the existing `SpaceStation`class structure.

Future Projects

Looking ahead, we plan to make significant improvements in the next phase of our project. In Phase 2, we will introduce Windows Forms for a more intuitive user interface, along with the integration of a database to enhance data management and functionality. Additionally, we will refine the class structure and subclasses to improve the system's scalability and performance.

We will also focus on enhancing the project documentation, ensuring it is comprehensive and easier for future developers to understand. A universal programming language will be adopted to further streamline development and improve collaboration across teams.

These updates will ensure that the project not only meets current requirements but is also flexible and robust enough to support future growth.

Conclusion

This project demonstrated the power of object-oriented programming in designing and implementing a space system. By leveraging classes and modular design, we were able to create an organized framework for simulating and managing the various components of space missions.

The use of object-oriented principles allowed for a clear separation of responsibilities, making the system both scalable and easy to maintain. This approach also facilitated future enhancements and modifications, ensuring that the system remains flexible as new features or components are added.

In conclusion, the project showcased how a structured, class-based design can effectively model and manage complex systems, providing a simple yet powerful solution for simulating space mission dynamics.