Mission Control Starbase

COS214 Project 2021

Jonathan Enslin u19103345
Isak van der Walt u20468203
Adir Miller u20692286
Ryan Broemer u20519517
Benjamin Osmers u16068344

Link to Doc:
https://docs.google.com/document/d/1tnu9eCx_knm92U4yn_u4neimcGoxra36mVMdNcZCTNQ/edit?usp=sharing

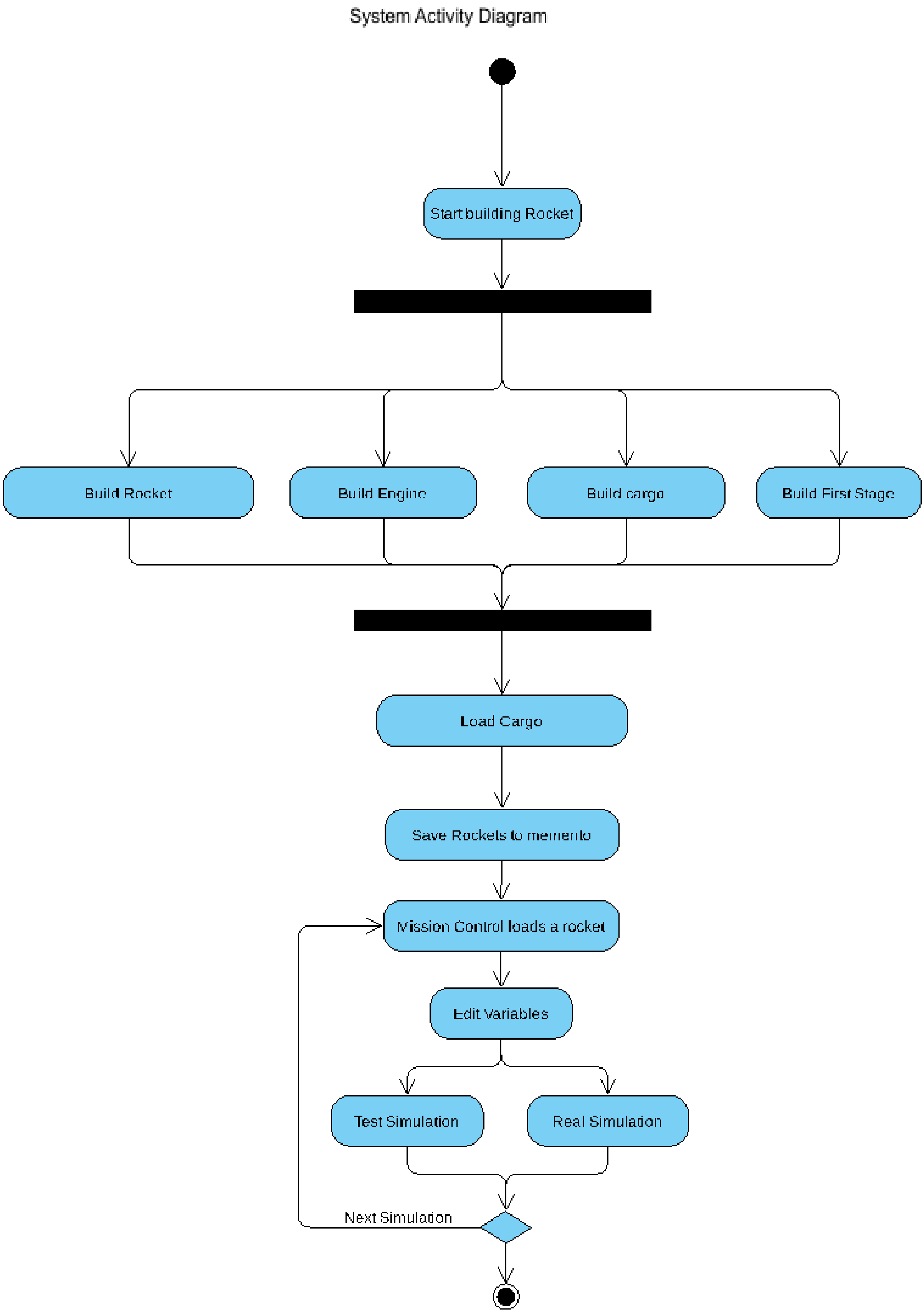
1. Design

1.1 Functional Requirements

- Building rocket, and rocket components
  - Build stages
    - First stage
    - Second stage
  - Build Dragon
    - Cargo Dragon
    - Crew Dragon
  - Build Merlin engines
  - Build Starlink satellites and payload
- Be able to launch the rocket and observe the components.
  - Launch rockets with different payloads.
    - Launch Starlink Satellites. (General Payload)
    - Launch Crew and Cargo to ISS (Crew Dragon)
    - Launch Cargo to ISS (Dragon)
  - Watch fuel capacity.
    - Make sure engines are never empty on launch.
    - Observer fuel capacity during flight.
      - Make sure fuel does not go below limit before destination is reached
    - Notify and warn once fuel is below allowed limit.
- Keep different states of the build, and launch process
  - Using a memento to store Rockets that mission control uses.
    - Store Rockets so that we can use them later or reset them.
    - Change values and states, then relaunch.
- Simulations of Rockets and their payloads.
  - Using Mission Control Starbase.
    - Mission Control can start Rockets and use observers to watch various parts of the Rocket.
    - Mission Control should also be able to start or stop simulation depending whether it is a test simulation or actual simulation.
    - You cannot stop and change actual simulations, but you can stop and change test simulations.
    - Mission Control should also be able to input certain parameters. For example: Fuel capacity.
- Be able to change the state of the rocket
  - Change between the following states:
    - Build
      - While the rocket is being built it will stay in this state
    - Fuel
      - The rocket will remain in this state until it is fuelled, and pre-launch checks are complete
    - Launch
      - In this state the rocket will have a simulated launch
    - Real Launch

            - In this state, the real launch will occur, the rocket will cycle through all the previous states in a continuous sequence.
- Be able to iterate through a collection of rockets stored in a caretaker, part of a memento pattern
    - Use the memento and iterator pattern to reinstate each of the saved rockets to be used later by Mission Control.
    - Memento will also be used to save a list of rockets.
- Users running Mission Control must be able to change variables.
    - Variables like:
        - Distance
        - Fuel
        - Weight
    - When running a test simulation. Users must be able to pause and test simulation with different variables.
    - When running real simulations. Users can not change variables on pause, since you cannot do so in real life rocket tests.
    - Before rocket simulation starts the user must be able to change variables.

1.2 Activity Diagram



1.3 Selected Design Patterns

1. Chain of Responsibility

a. **Description:** Chain was chosen as the satellites follow along from one another. each one will deploy then signal the next to be deployed. This is perfect for a chain of responsibility.

b. **Members:**

   i. Starlink

   ii. SatelliteLauncher

   iii. Satellite

2. Factory Method

a. **Description:** Factory was chosen since we need to produce many satellites so having one object that can control the making of them is very important.

b. **Members:**

   i. Factory

   ii. SatelliteFactory

   iii. Satellite

3. Composite

a. **Description:** Composite was chosen because the rockets are made up of different parts that add on to each other. Composite allows for a more dynamic way of making the rockets where not all have the same amount of boosters for example allowing us to create the rocket at run time.

b. **Members:**

   i. Booster

   ii. Falcon

   iii. FalconHeavy

   iv. SecondStage

   v. Leaf: Engine

4. Observer

a. **Description:** Observer allows the system to monitor the fuel level which is a very important value for a successful launch. Having a system watch over the fuel level so the client does not need to keep checking allows for more separation of concerns.

b. **Members:**

   i. Observer

   ii. FuelObserver

   iii. Booster

5. State

- a. **Description:** State was chosen since we move from one state to another in a linear fashion, no matter the rocket we still need to follow the same set of steps, namely Build->Fuel->Launch->ActivateLaunch. By extracting out the state we allow for a more clear system so we can easily track where in the process we are currently.
- b. **Members:**
    - i. State
    - ii. BuildS
    - iii. FuelS
    - iv. LaunchS
    - v. ActiveLaunchS

6. Builder

- a. **Description:** The rockets are made up of different parts and depending on the rocket we can change out which parts we use. using a builder allows us to have a central object that allows us to build any given rocket if it has different parts to its counterpart.
- b. **Members:**
    - i. Builder
    - ii. ConcreteBuilder
    - iii. MissionControlStarbase

7. Iterator

- a. **Description:** The iterator pattern is used to iterate through a collection of memento objects, each storing a different state of the rocket.
- b. **Members:**
    - i. Iterator
    - ii. RocketIterator
    - iii. Aggregate
    - iv. RocketAggregate
    - v. Caretaker

8. Memento

- a. **Description:** The memento pattern is used to store the rocket state, propulsion and payload. These attributes can then be retrieved when required.
- b. **Members:**
    - i. Caretaker
    - ii. RocketMemento

9. Prototype

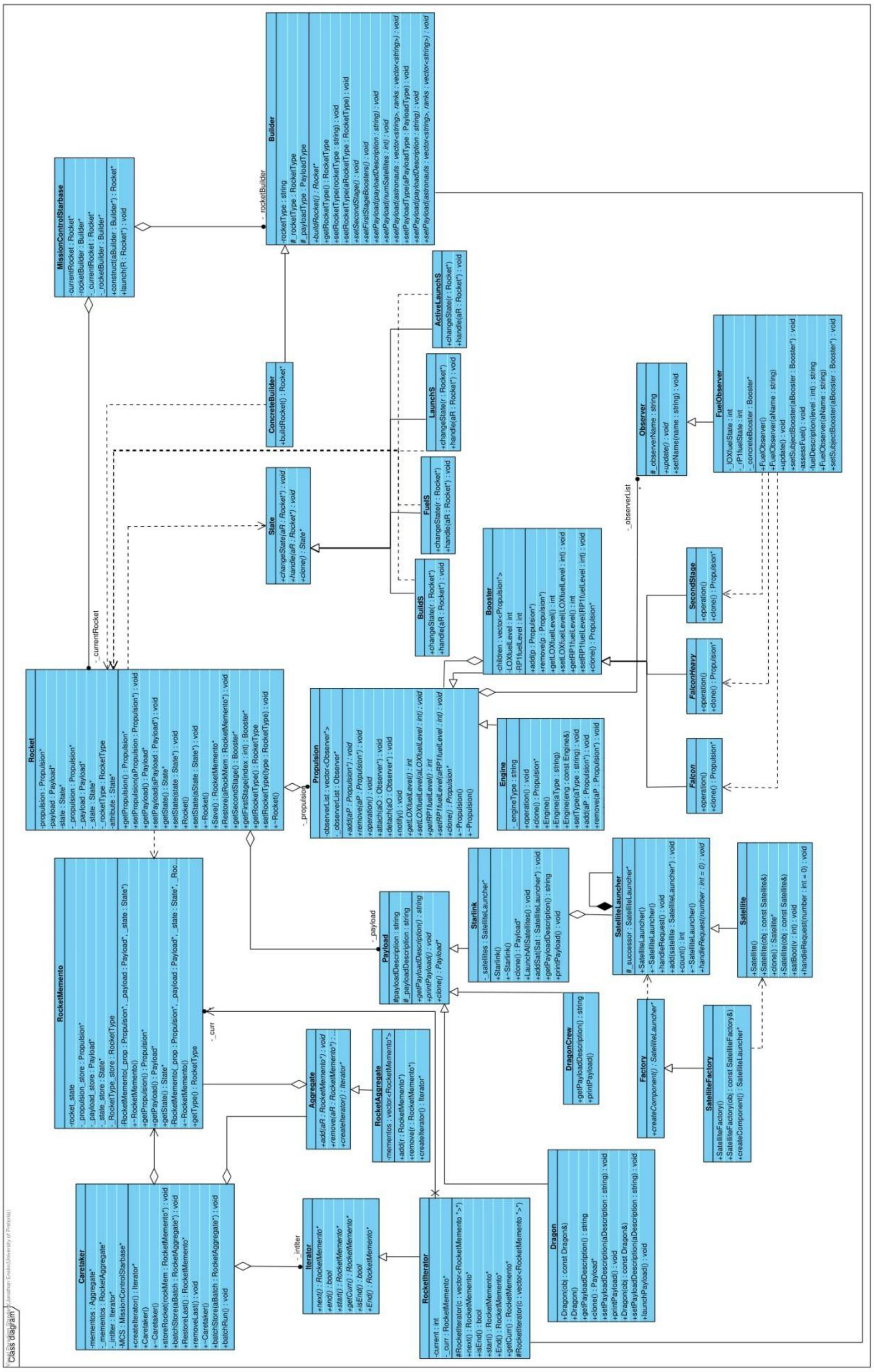
a. **Description:** Most satellites and engines are the same therefore being able to clone them with the prototype can help simplify the creation of groups and large numbers of objects. We can also make “groups” of each object but cloning different prototypes allowing for a dynamic way to create those objects instead of having to manually make each one.
b. **Members:**
    i. Satellite
    ii. Engine

10. Template Method

a. **Description:** Templates allow us to reduce the need for repeated code. It is used to create the booster hierarchy.
b. **Members:**
    i. Booster

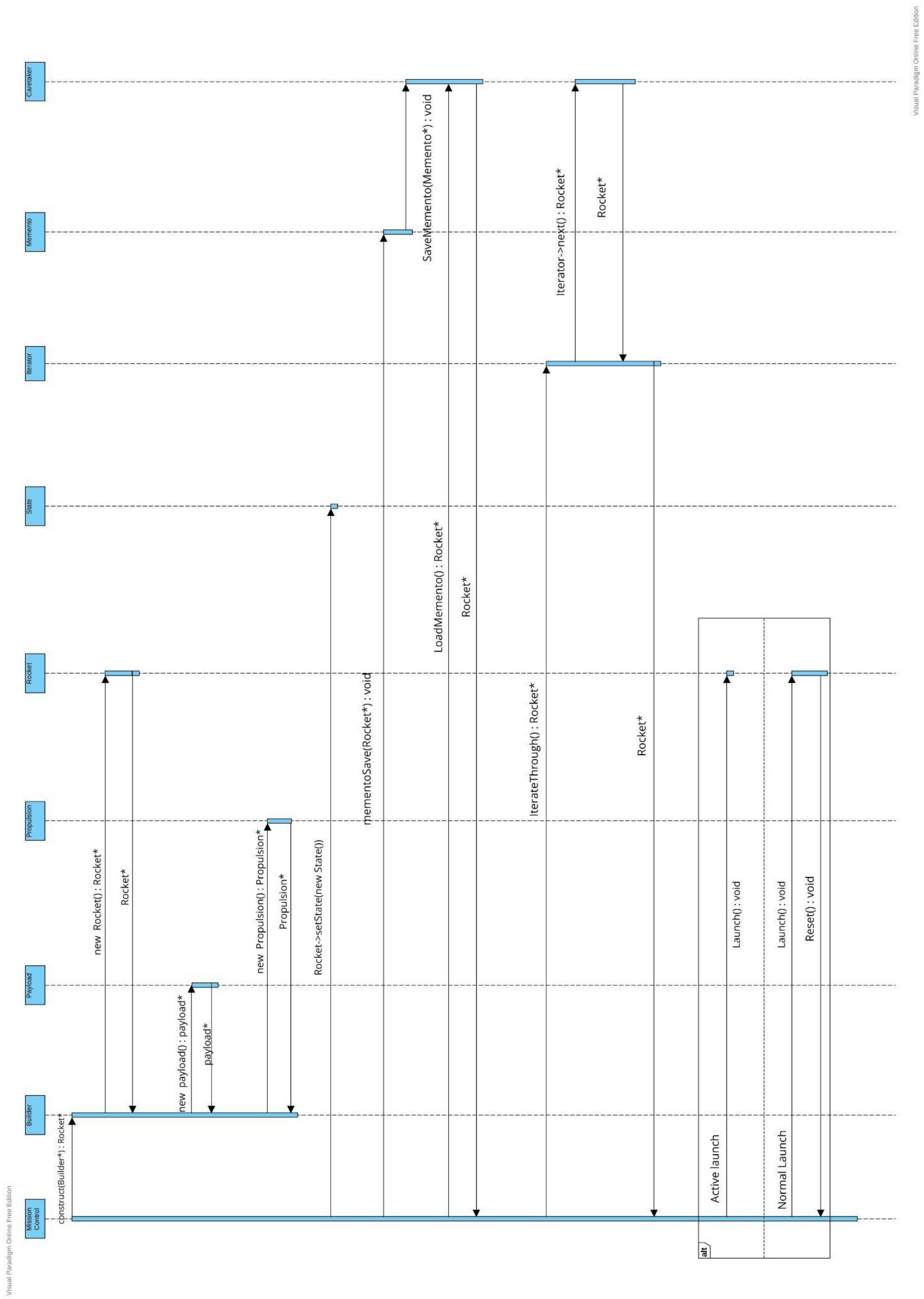
1.4 Class Diagram

Overall Class Diagram

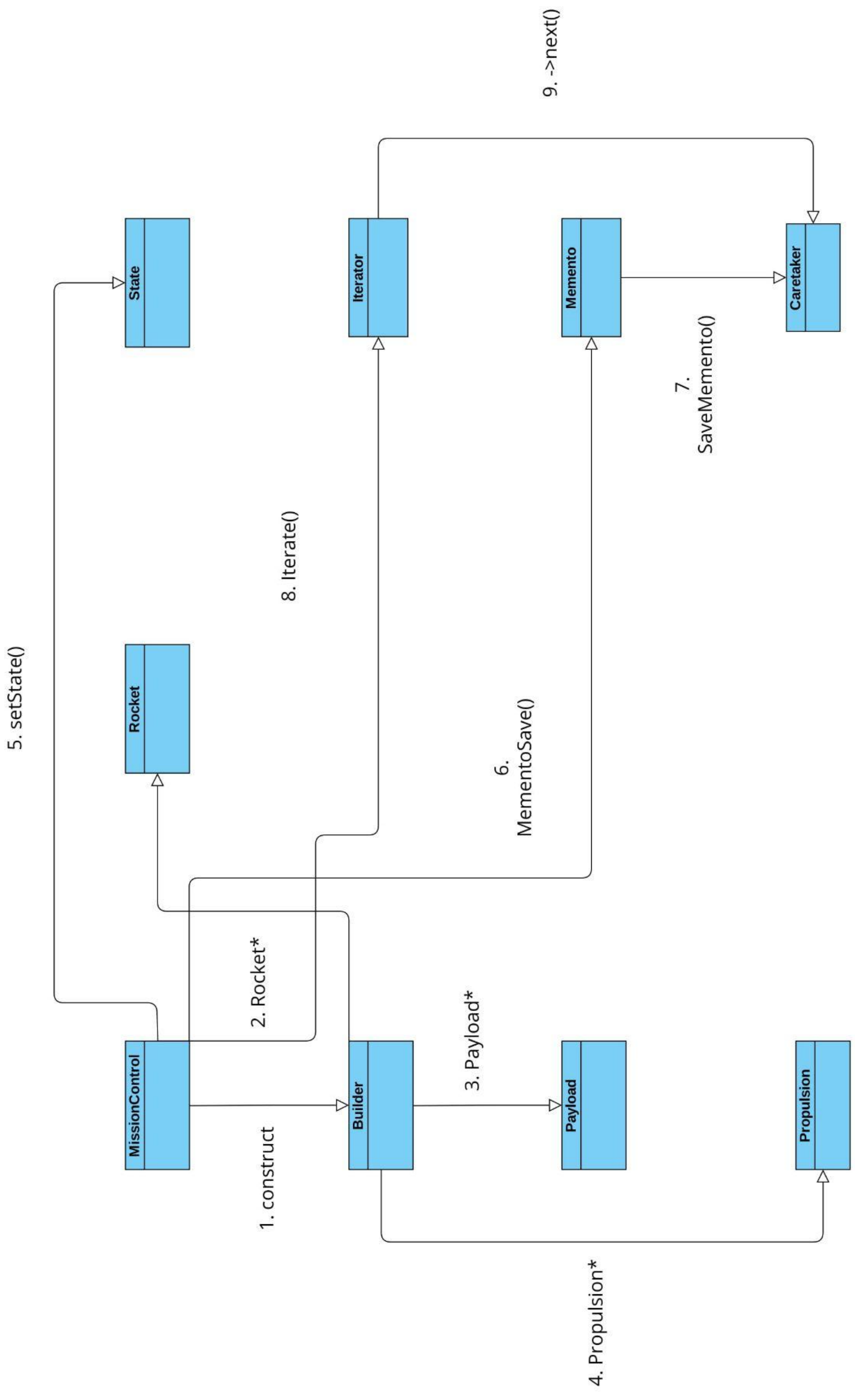


1.5 Sequence and Communication Diagrams

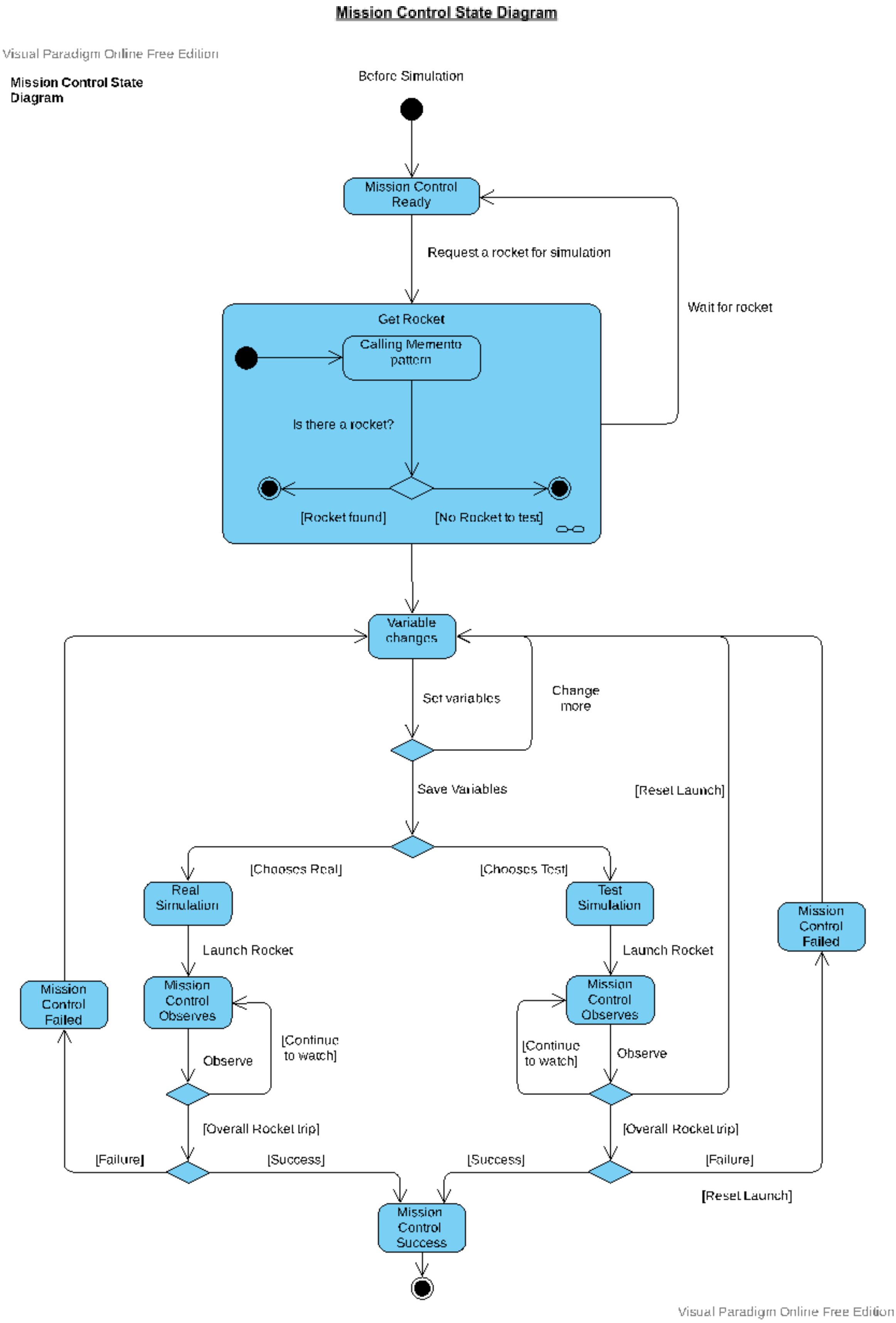
Sequence Diagram



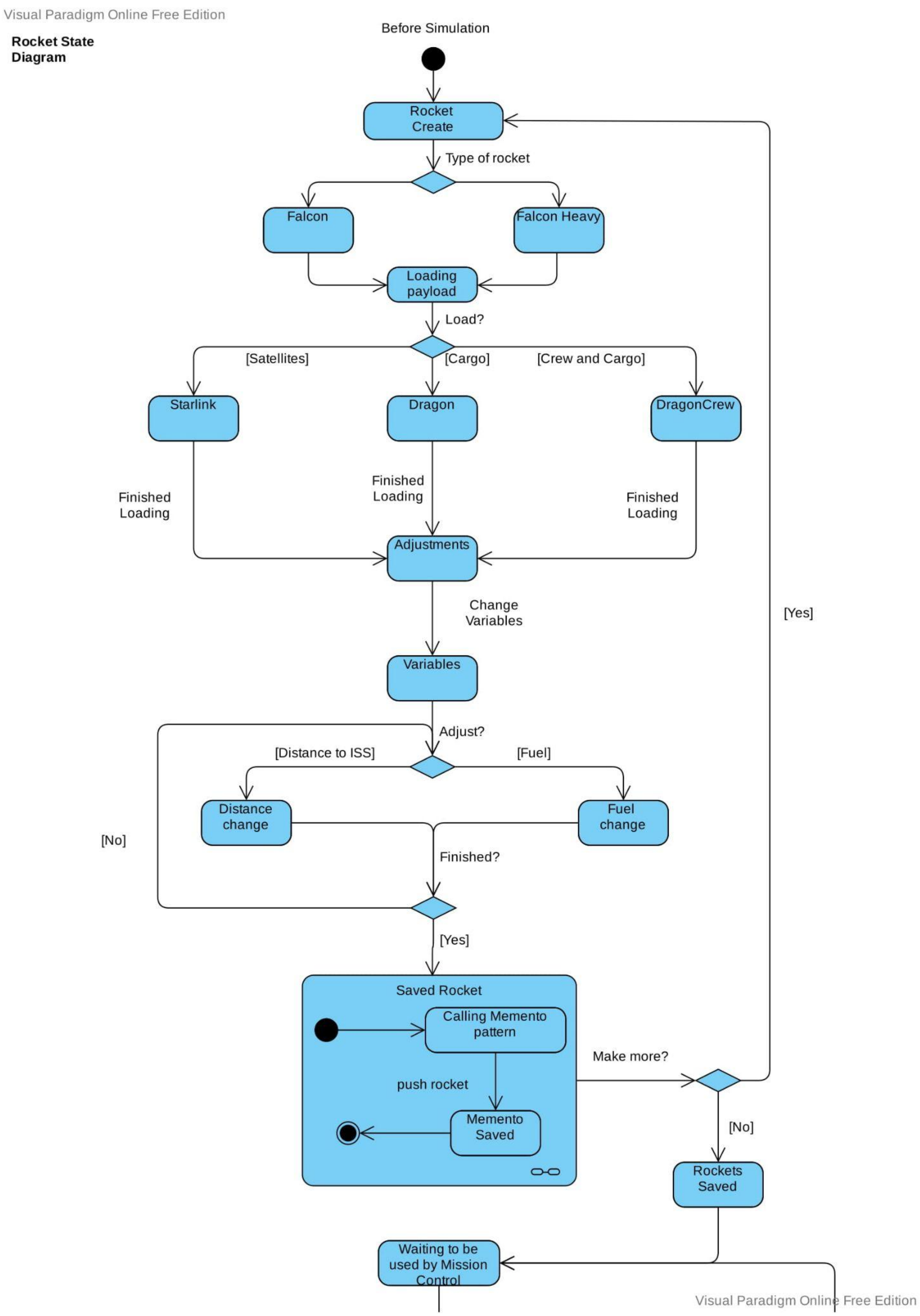
Communication Diagram



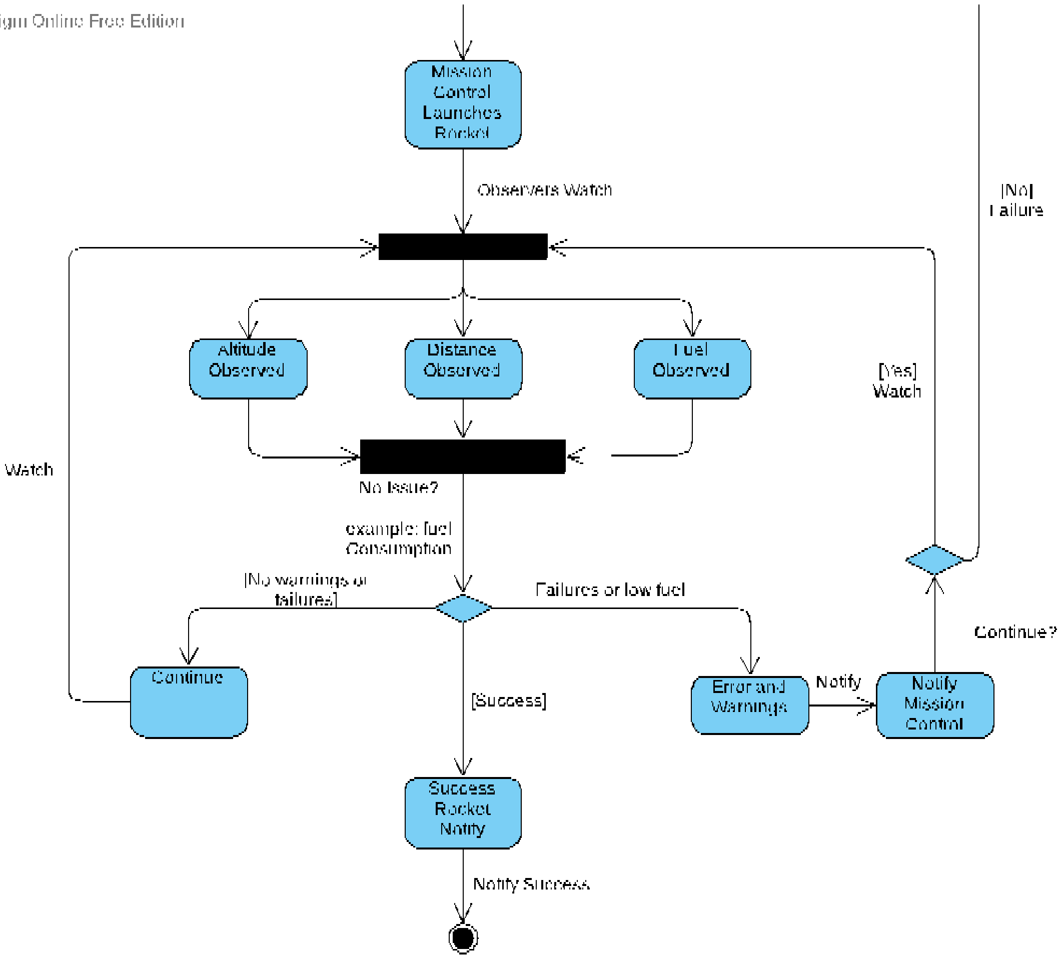
1.6 State Diagrams



Rocket State Diagram

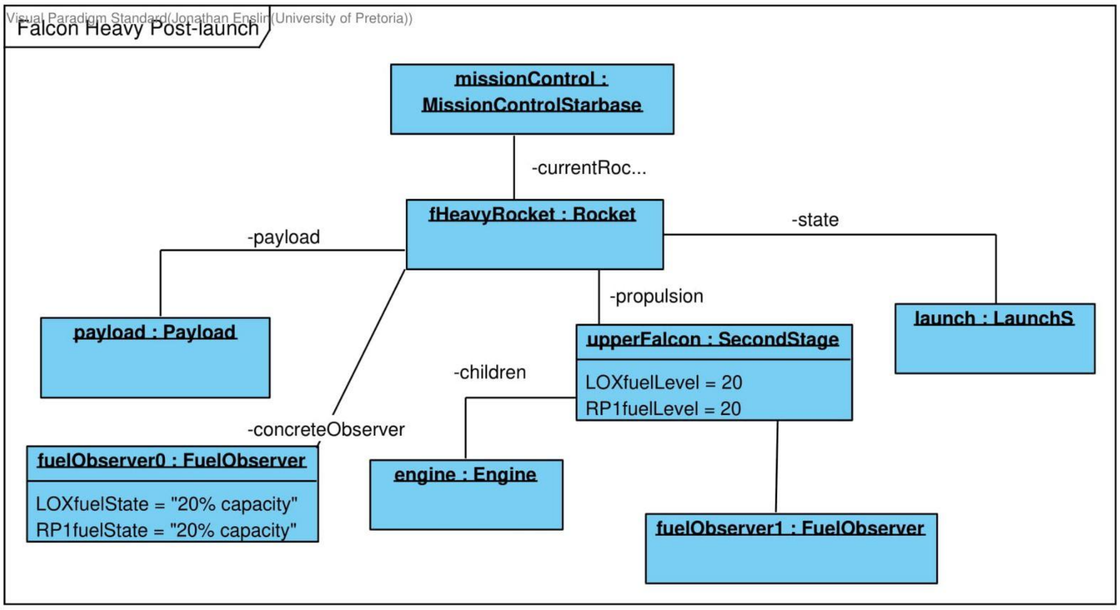


Visual Paradigm Online Free Edition

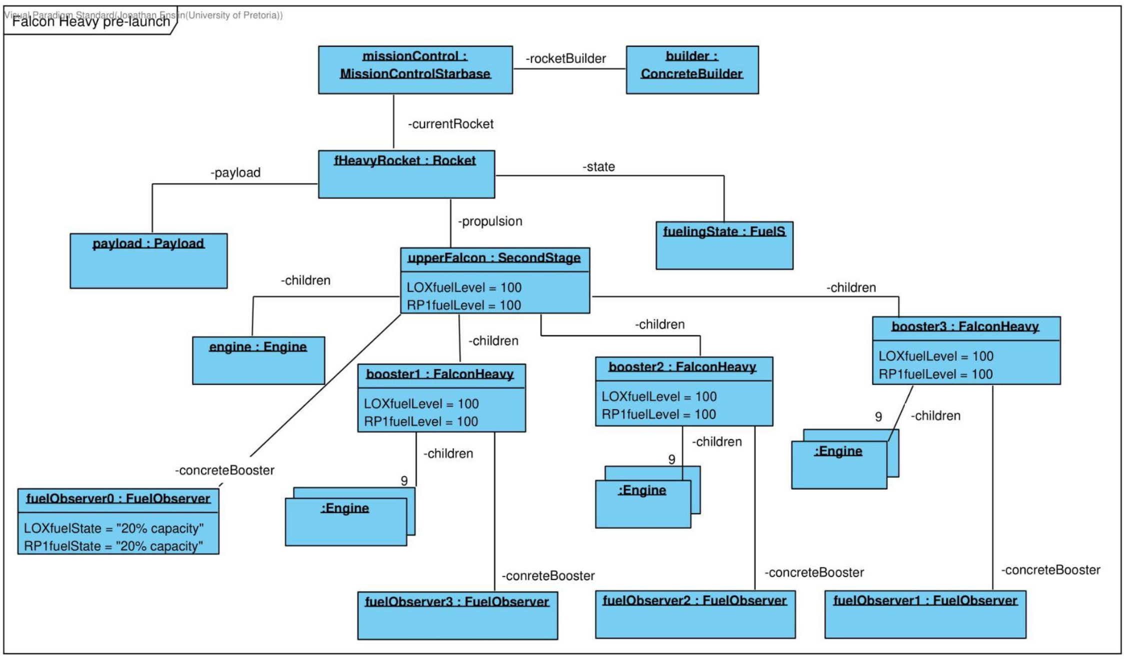


1.7 Object Diagrams

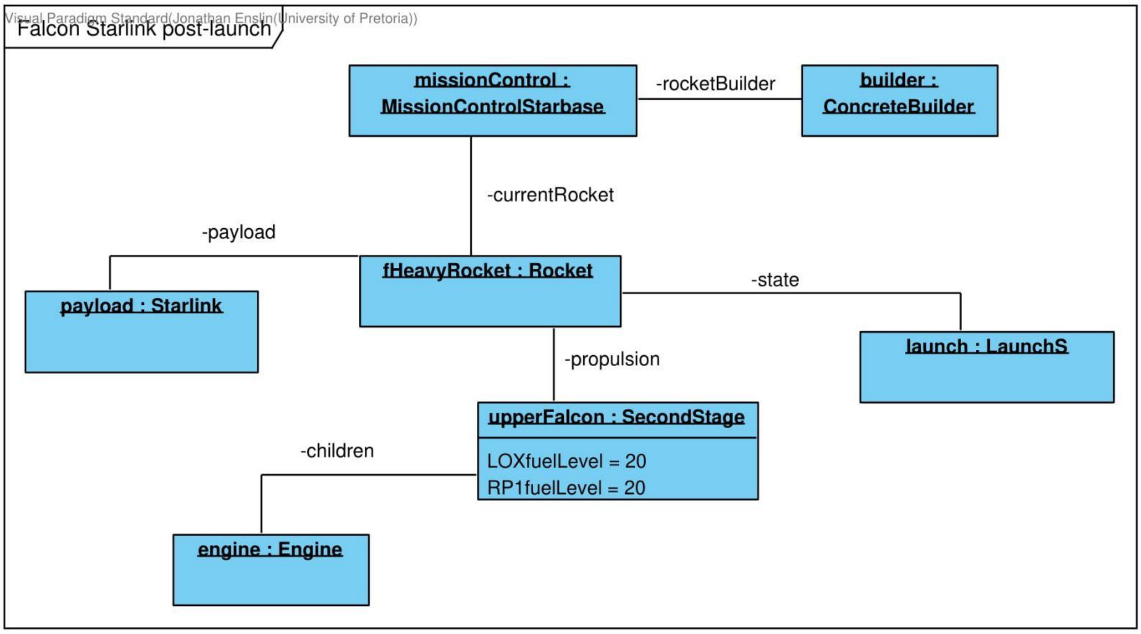
Falcon Heavy Post Launch



Falcon Heavy Pre-Launch



Falcon Starlink Post Launch



Falcon Starlink Pre-Launch