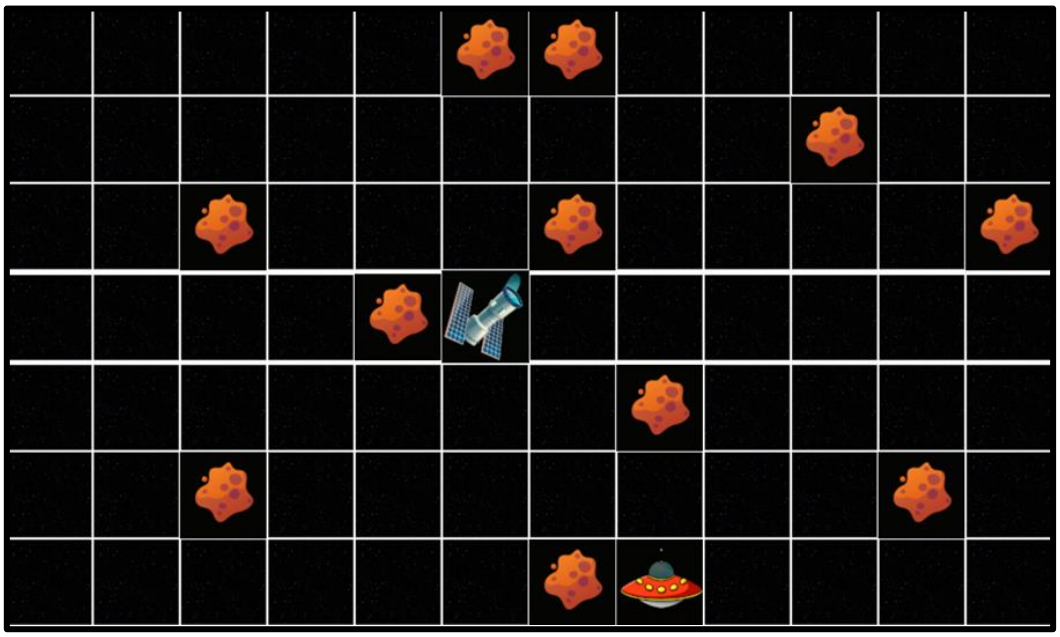
INFORMATICS INSTITUTE OF TECHNOLOGY

UNIVERSITY OF WESTMINSTER

MODULE: 6SENG005C.1 Formal Methods



Lecturer Name: Dr.Thilini Piyatilake

Student Name: H.K.J.N.Gunaweera

IIT ID: 20200003

UoW ID: w1810567

Degree Program: BEng. (Hons) in Software Engineering

Table of Contents

List of tables

List of Figures

1.0 Structure Diagram – B specification

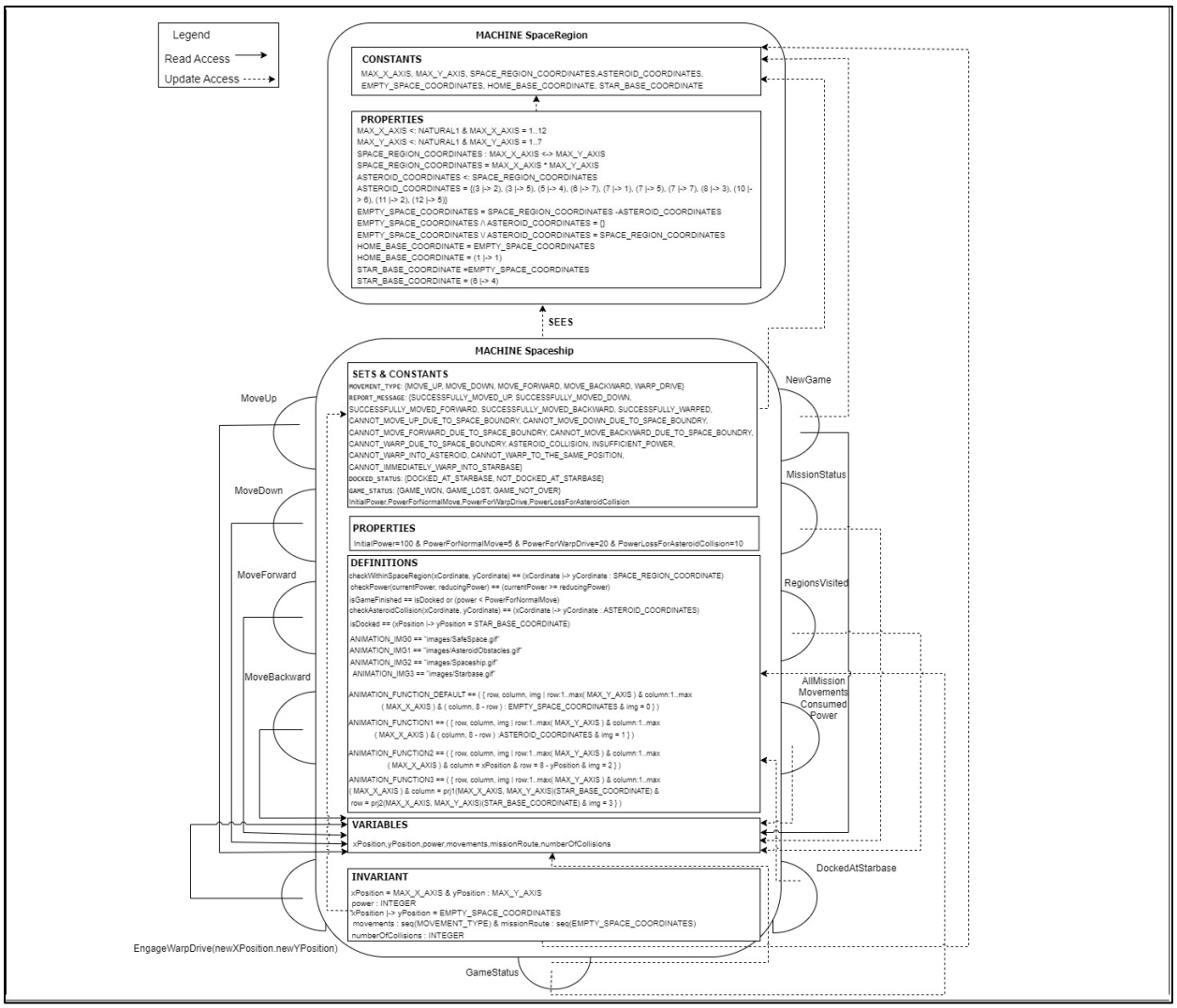


Figure 1 Structure Diagram

Visit https://drive.google.com/file/d/1HS33AT6JUt8qlj2LLbcvWzhdlK1eMQzE/view?usp=sharing to see the diagram more clearly

Model Type: Finite State Machine (FSM)

Model Overview:

- The model employs a finite state machine (FSM) to represent the spaceship's behavior within the game environment.
- The FSM is comprised of a finite set of states, interconnected by transitions triggered by specific events.
- The model's initial state is denoted as "Game_not_over."
- The model incorporates tables for sets, constants, properties, and variables, providing further context for its operation.

States:

- Game_not_over: The initial state, from which transitions to the following states can occur:
    - MoveUp
    - MoveDown
    - MoveForward
    - MoveReverse
    - EngageWarpDrive

- MoveUp: Represents the spaceship's upward movement. Transitions to:
    - Moved Successfully (upon successful movement)
    - Move Failed Unable to Visit Unknown Space (upon attempting to access an unknown area)

- MoveDown: Represents the spaceship's downward movement. Transitions to the same states as MoveUp.

- MoveForward: Represents the spaceship's forward movement. Transitions to:
    - Freespace (upon successful movement)
    - Asteroid Collision (upon collision with an asteroid)

- MoveReverse: Represents the spaceship's reverse movement. Transitions to the same states as MoveForward.

- EngageWarpDrive: Represents the spaceship's engagement of its warp drive. Transitions to:
    - Freespace (upon successful warp)
    - Warp Failed states (upon unsuccessful warp)

- Terminal States:
    - Moved Successfully
    - Freespace
    - DockedAtStarBase
    - These states signify the completion of a particular action or sequence, with no further transitions possible.

2.0 Rationale and Elaboration

This project entails the creation of a B specification for a simplified iteration of the Spaceship & Asteroids arcade game, utilizing B tools. A detailed exposition of the assigned and employed SETS, CONSTANTS, PROPERTIES, VARIABLES, and INVARIANTS is presented below.

2.1 SETS

The project involves two main machines: SpaceRegion and Spaceship. Each machine comprises several sets crucial for their functionality. The SpaceRegion machine sets define the spatial characteristics of the game, while the Spaceship machine sets are focused on the spaceship's movements, messages, and overall game status.

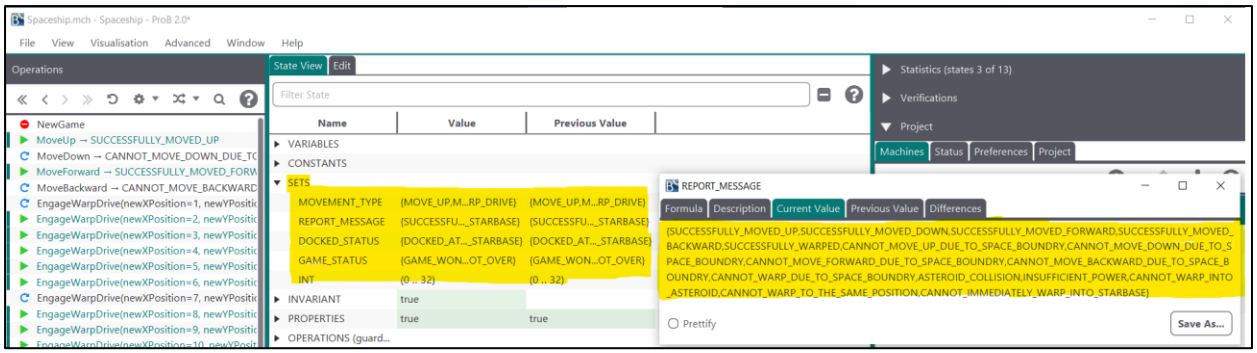


Figure 2 SETS

Machine	Set	Elements	Description
SpaceRegion	**MAX_X_AXIS**	1..12	Maximum value for the X-axis in the space region.
	MAX_Y_AXIS	1..7	Maximum value for the Y-axis in the space region.
	SPACE_REGION_COORDINATES	All pairs (x, y) where x is in MAX_X_AXIS and y is in MAX_Y_AXIS	Set of all coordinates in the space region.
	ASTEROID_COORDINATES	{(3,2), (3,5), (5,4), (6,7), (7,1), (7,5), (7,7), (8,3), (10,6), (11,2), (12,5)}	Set of coordinates occupied by asteroids.
	EMPTY_SPACE_COORDINATES	All coordinates in SPACE_REGION_COORDINATES not in ASTEROID_COORDINATES	Set of coordinates available for spaceship movement (excluding asteroid coordinates).

	HOME_BASE_COORDINATE	(1,1)	Coordinate of the home base.
	STAR_BASE_COORDINATE	(6,4)	Coordinate of the star base.
Spaceship	**MOVEMENT_TYPE**	{MOVE_UP, MOVE_DOWN, MOVE_FORWARD, MOVE_BACKWARD, WARP_DRIVE}	Movement types of spaceships
	REPORT_MESSAGE	{SUCCESSFULLY_MOVED_UP, SUCCESSFULLY_MOVED_DOWN, SUCCESSFULLY_MOVED_FORWARD, SUCCESSFULLY_MOVED_BACKWARD, SUCCESSFULLY_WARPED, CANNOT_MOVE_UP_DUE_TO_SPACE_BOUNDRY, CANNOT_MOVE_DOWN_DUE_TO_SPACE_BOUNDRY, CANNOT_MOVE_FORWARD_DUE_TO_SPACE_BOUNDRY, CANNOT_MOVE_BACKWARD_DUE_TO_SPACE_BOUNDRY, CANNOT_WARP_DUE_TO_SPACE_BOUNDRY, ASTEROID_COLLISION, INSUFFICIENT_POWER, CANNOT_WARP_INTO_ASTEROID, CANNOT_WARP_TO_THE_SAME_POSITION, CANNOT_IMMEDIATELY_WARP_INTO_STARBASE}	Appropriate report messages for various actions and conditions.
	DOCKED_STATUS	{DOCKED_AT_STARBASE, NOT_DOCKED_AT_STARBASE}	Spaceship docked status.
	GAME_STATUS	{GAME_WON, GAME_LOST, GAME_NOT_OVER}	Game status messages.
	InitialPower	100	Initial power for the spaceship.

	PowerForNormalMove	5	Power usage for a normal move.
	PowerForWarpDrive	20	Power usage for a warp drive.
	PowerLossForAsteroidCollision	10	Power loss for asteroid collision.

Table 1- SETS

2.2 CONSTANTS & PROPERTIES

CONSTANTS and PROPERTIES play a crucial role in defining the characteristics and behavior of the SpaceRegion and Spaceship machines.

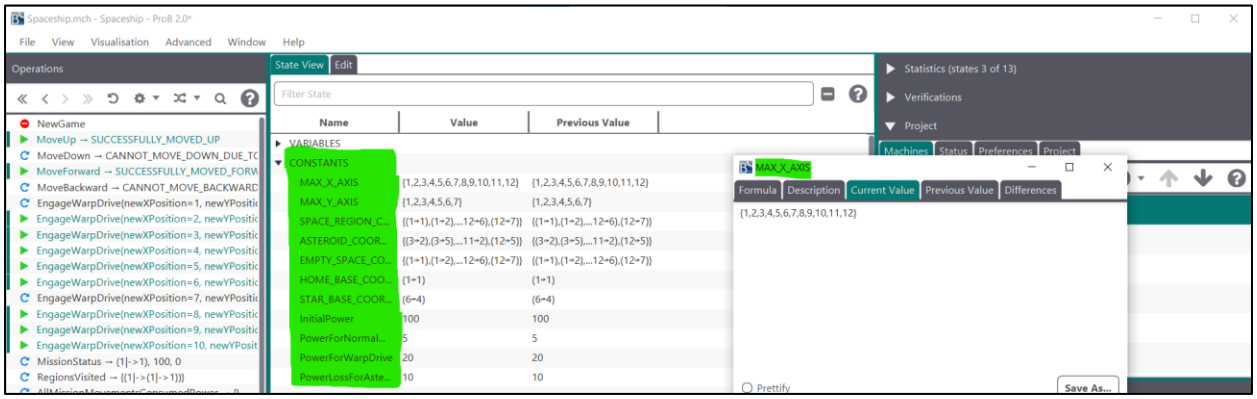


Figure3 CONSTANTS

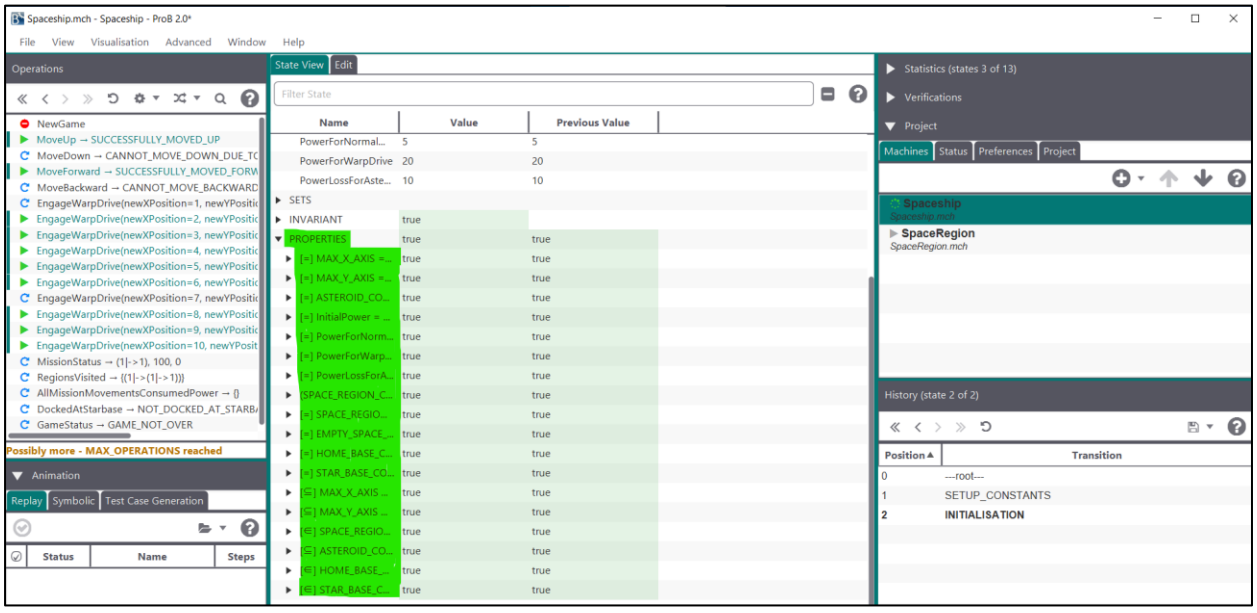


Figure4 PROPERTIES

Machine	CONSTANT	CONSTANT explanation	PROPERTY	PROPERTY explanation
SpaceRegion	MAX_X_AXIS, MAX_Y_AXIS	Maximum values for the X and Y axes in the space region.	MAX_X_AXIS <: NATURAL1 & MAX_X_AXIS = 1..12	Ensures that MAX_X_AXIS is a natural number between 1 and 12.
	SPACE_REGION_COORDINATES	Set of all coordinates in the space region based on MAX_X_AXIS and MAX_Y_AXIS.	MAX_Y_AXIS <: NATURAL1 & MAX_Y_AXIS = 1..7	Ensures that MAX_Y_AXIS is a natural number between 1 and 7.
	ASTEROID_COORDINATES	Set of coordinates occupied by asteroids in the space region.	SPACE_REGION_COORDINATES : MAX_X_AXIS <-> MAX_Y_AXIS	Establishes a relationship between SPACE_REGION_COORDINATES and the X and Y axes.
	EMPTY_SPACE_COORDINATES	Set of coordinates available for spaceship movement, excluding those occupied by asteroids.	ASTEROID_COORDINATES <: SPACE_REGION_COORDINATES	Specifies that ASTEROID_COORDINATES is a subset of SPACE_REGION_COORDINATES.
	HOME_BASE_COORDINATE, STAR_BASE_COORDINATE	Coordinates of the home base and star base within EMPTY_SPACE_COORDINATES.	EMPTY_SPACE_COORDINATES = SPACE_REGION_COORDINATES - ASTEROID_COORDINATES	Defines EMPTY_SPACE_COORDINATES as the set difference between SPACE_REGION_COORDINATES and ASTEROID_COORDINATES.
			HOME_BASE_COORDINATE : EMPTY_SPACE_COORDINATES & HOME_BASE_COORDINATE = (1	Specifies that HOME_BASE_COORDINATE is within EMPTY_SPACE_COORDINATES and defines its value.
			STAR_BASE_COORDINATE : EMPTY_SPACE_COORDINATES & STAR_BASE_COORDINATE = (6	Specifies that STAR_BASE_COORDINATE is within EMPTY_SPACE_COORDINATES and defines its value.
Spaceship	InitialPower, PowerForNormalMove, PowerForWarpDrive, PowerLossForAsteroidCollision	Constants for initial power, power usage for normal move, power usage for warp drive, and power loss for asteroid collision.	InitialPower = 100	Sets the initial power for the spaceship to 100.
			PowerForNormalMove = 5	Sets the power usage for a normal move to 5.
			PowerForWarpDrive = 20	Sets the power usage for a warp drive to 20.

			`PowerLossForAsteroidCollision = 10`	Sets the power loss for an asteroid collision to 10.

Table 2- CONSTANTS & PROPERTIES

2.3 VARIABLES & INVARIANTS

These VARIABLES and INVARIANTS are crucial for maintaining the consistency and correctness of the system's state and behavior during the execution of operations. They help define constraints on the allowed values and relationships between different components of the system.

Variables

Variables represent the dynamic state of a system. They are used to model values that can change during the execution of the system.

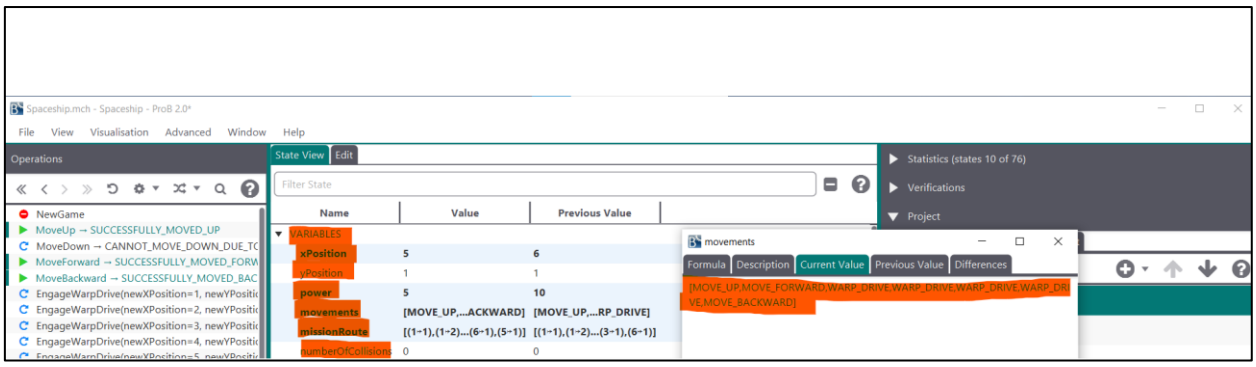


Figure4 VARIABLES

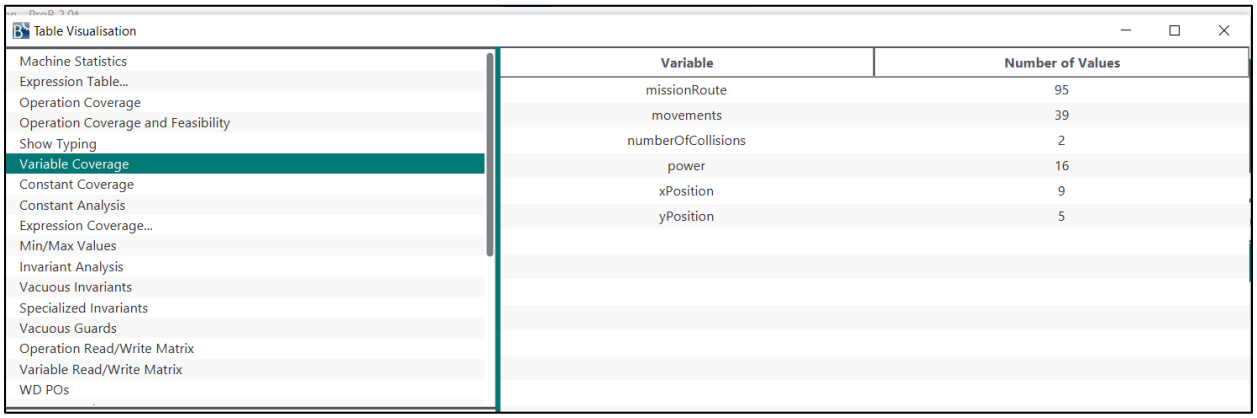


Figure VARIABLE Coverage

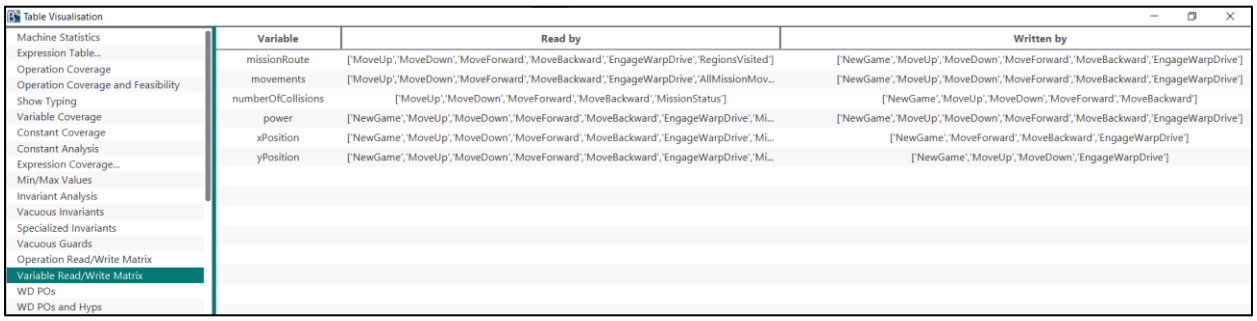


Figure VARIABLE Read/Write matrix

Purpose of variables

Variables help track the changing aspects of a system over time. In the context of the provided machines (SpaceRegion and Spaceship), variables include elements such as the current position of a spaceship (xPosition and yPosition), the remaining power of the spaceship (power), the movements made by the spaceship (movements), the route taken by the spaceship (missionRoute), and the number of collisions the spaceship has encountered (numberOfCollisions).

Usage of variables

They are manipulated by operations in the machine. Operations can modify the values of variables based on certain conditions and constraints.

Invariant

Invariants are logical conditions that must hold true throughout the execution of the system. They provide constraints on the possible values of variables.

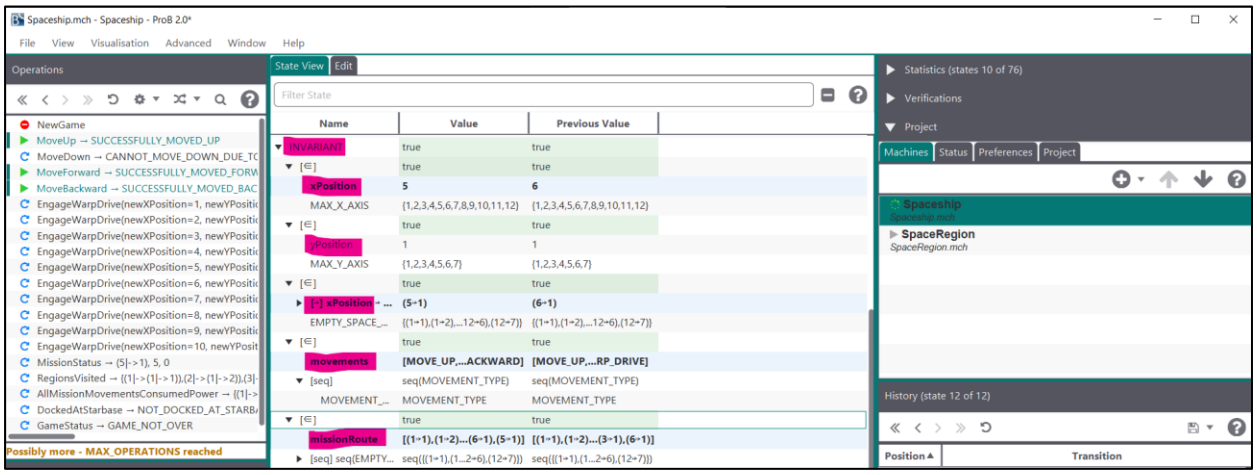


Figure3 INVARIANTS

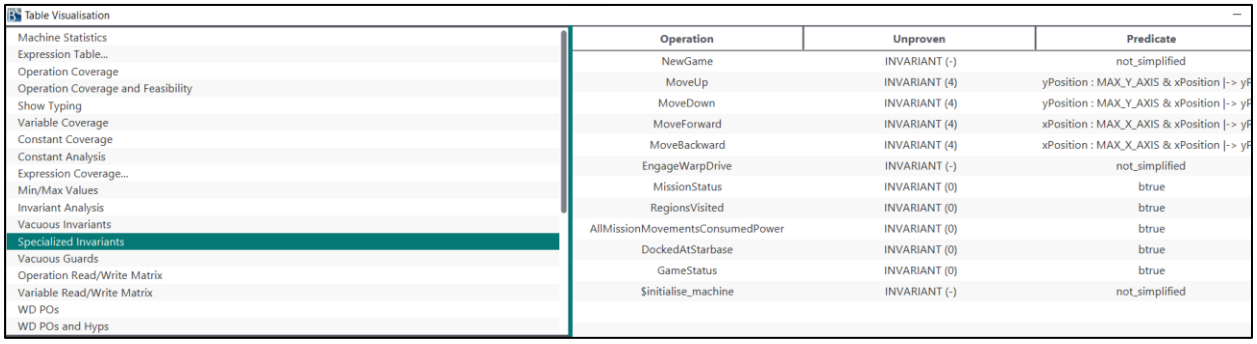


Figure Specialized INVARIANTS

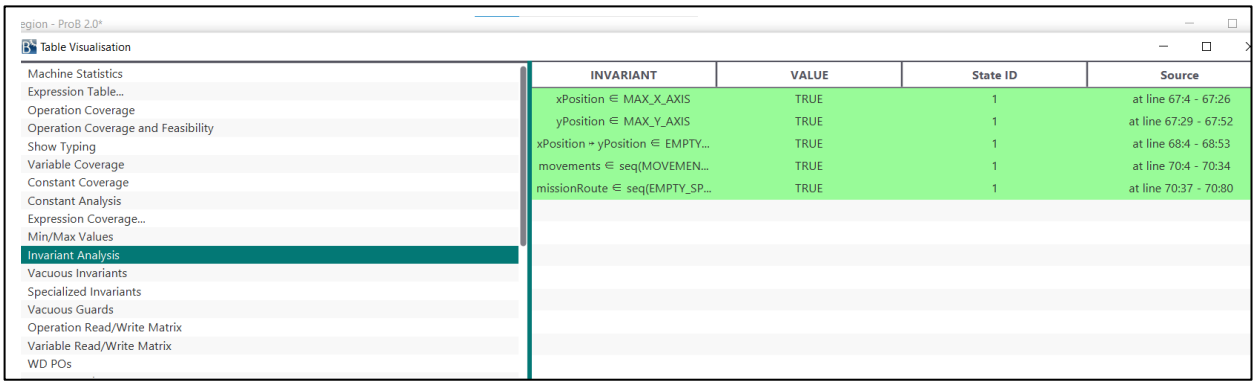


Figure INVARIANT Analysys

Purpose of invariants

Invariants ensure that certain properties remain unchanged during the execution of the system. They help in specifying correctness conditions that should be preserved at all times.

Usage of Invariants

They are conditions that must be satisfied before and after the execution of any operation. In the provided machines, invariants include conditions such as the current position being within the bounds of the space region, the power being an integer, the movements being a sequence of predefined types, the mission route consisting of coordinates in empty space, and the number of collisions being an integer.

Machine	VARIABLE	VARIABLE description	INVARIANT	INVARIANT description
	`(none)`	No specific variables defined in	`xPosition : MAX_X_AXIS`	The xPosition variable, representing the current x-

SpaceRegion		the SpaceRegion machine		coordinate of the spaceship, is within the bounds specified by the MAX_X_AXIS constant. This ensures that the x-coordinate of the spaceship is a valid value within the defined space region along the X-axis.
			yPosition : MAX_Y_AXIS	The yPosition variable, representing the current y-coordinate of the spaceship, is within the bounds specified by the MAX_Y_AXIS constant. Similar to the previous invariant, this ensures that the y-coordinate of the spaceship is a valid value within the defined space region along the Y-axis.
			xPosition |-> yPosition : EMPTY_SPACE_COORDINATES	The pair (xPosition, yPosition) represents a coordinate in the empty space as defined by EMPTY_SPACE_COORDINATES. This ensures that the current position of the spaceship is within the set of coordinates available for movement, excluding positions occupied by asteroids.
			power : INTEGER	The power variable, representing the current power level of the spaceship, is an integer. This ensures that the power level is a whole number, conforming to the mathematical definition of integers.
			movements : seq(MOVEMENT_TYPE)	The movements variable, representing the sequence of movements made by the

				spaceship, is a sequence of elements from the MOVEMENT_TYPE set. This guarantees that the recorded movements are a valid sequence of predefined types, ensuring consistency in the recorded actions.
			missionRoute : seq(EMPTY_SPACE_COORDINATES)	The missionRoute variable, representing the sequence of coordinates in the spaceship's route, is a sequence of coordinates in empty space. This ensures that the recorded route consists only of coordinates in the empty space, excluding positions occupied by asteroids.
			numberOfCollisions : INTEGER	The numberOfCollisions variable, representing the count of collisions encountered by the spaceship, is an integer. This ensures that the count of collisions is a whole number, conforming to the mathematical definition of integers.
Spaceship	**xPosition**	This variable represents the current x-coordinate of the spaceship in the space region.	**xPosition : MAX_X_AXIS**	This invariant ensures that the xPosition variable remains within the allowed bounds specified by MAX_X_AXIS.
	yPosition	This variable represents the current y-coordinate of the spaceship in the space region.	**yPosition : MAX_Y_AXIS**	This invariant ensures that the yPosition variable remains within the allowed bounds specified by MAX_Y_AXIS.

	power	This variable represents the current power level of the spaceship. It is used to model the available energy or resources that the spaceship has for its operations.	**xPosition |-> yPosition : EMPTY_SPACE_COORDINATES**	This invariant ensures that the current position of the spaceship is within the set of empty space coordinates, meaning it is a valid position for the spaceship to occupy.
	movements	This variable is a sequence that stores all the movements performed by the spaceship. Each element in the sequence corresponds to a specific type of movement (e.g., MOVE_UP, MOVE_DOWN).	**power : INTEGER**	This invariant ensures that the power variable takes on integer values.
	missionRoute	This variable is a sequence that stores the coordinates representing the route taken by the spaceship. The coordinates are expected to be in empty space, as specified by the invariant.	**movements : seq(MOVEMENT_TYPE)**	This invariant ensures that the movements variable is a sequence composed of elements from the MOVEMENT_TYPE set, representing valid spaceship movements.
	numberOfCollisions	This variable keeps track of the total number of collisions encountered by the spaceship.	**missionRoute : seq(EMPTY_SPACE_COORDINATES)**	This invariant ensures that the missionRoute variable is a sequence of coordinates that are all within the set of empty space coordinates.
			numberOfCollisions : INTEGER	This invariant ensures that the numberOfCollisions variable takes on integer values.

Table 3- VARIABLES & INVARIANTS

2.4 INITIALISATION

The initializations provided for both the SpaceRegion and Spaceship machines are a set of instructions that define the starting state of the machines when a new instance of the game is initialized. These initializations ensure that the game starts with the spaceship positioned at the **home base, full power, no previous movements or collisions**, and with the **necessary data structures initialized** for tracking the spaceship's journey.

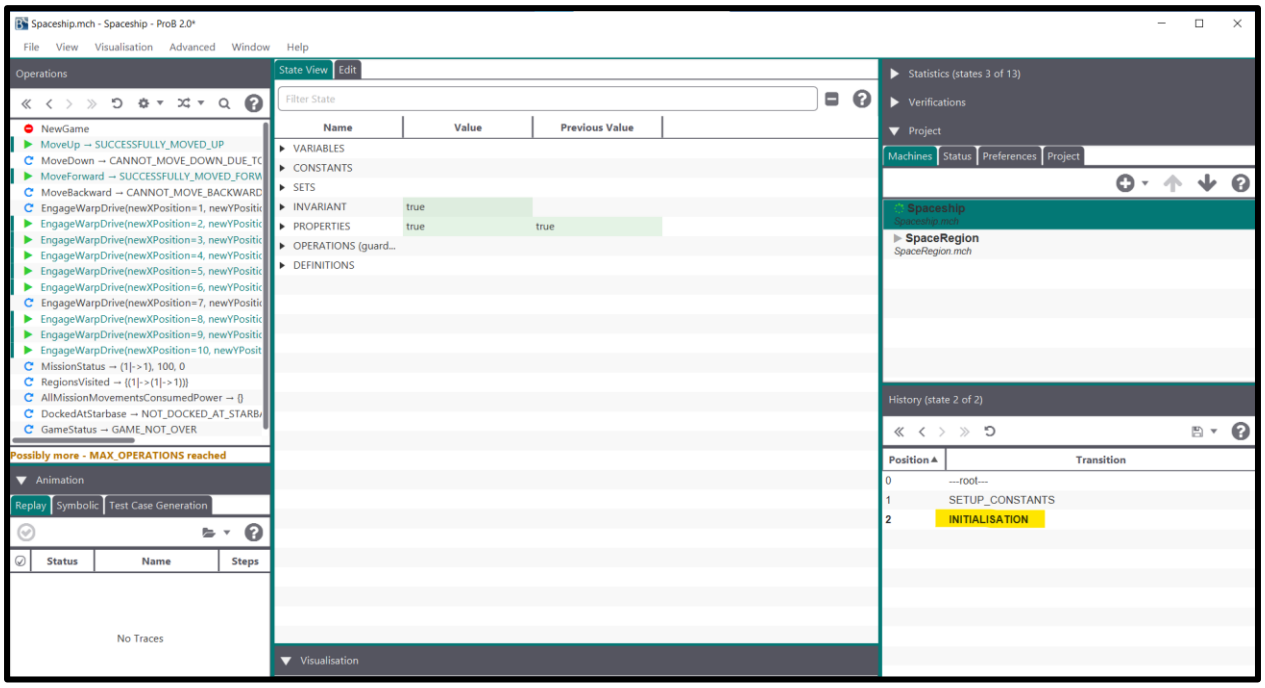
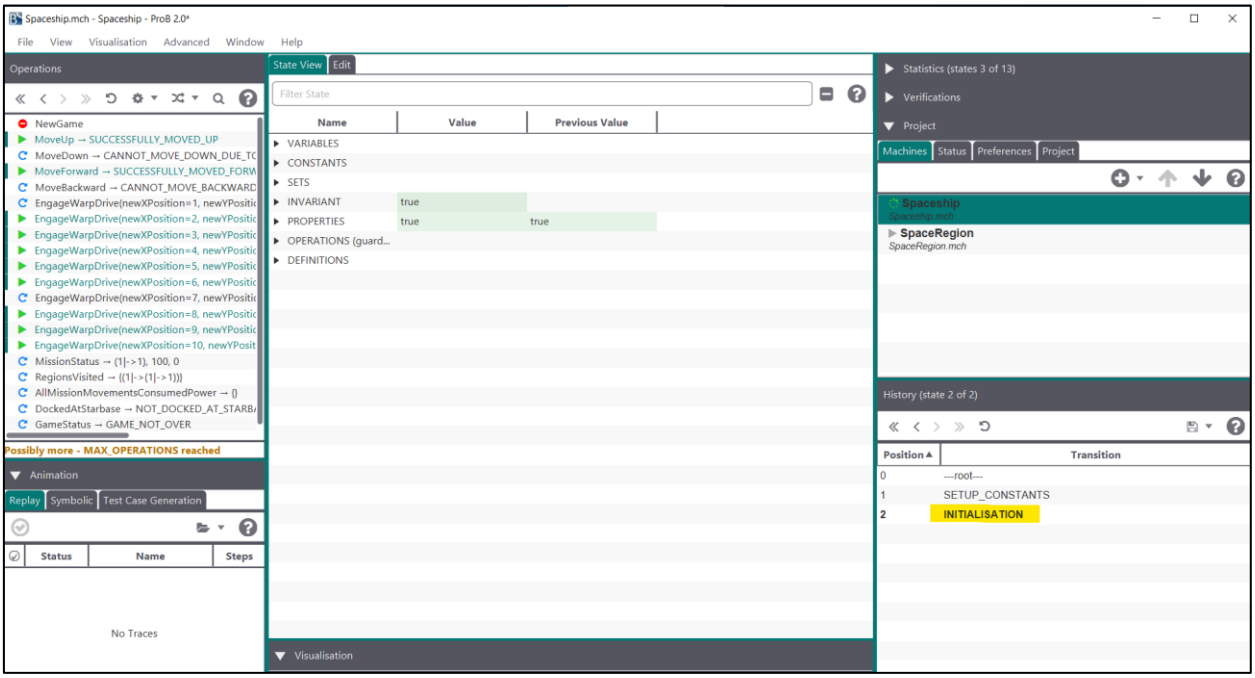


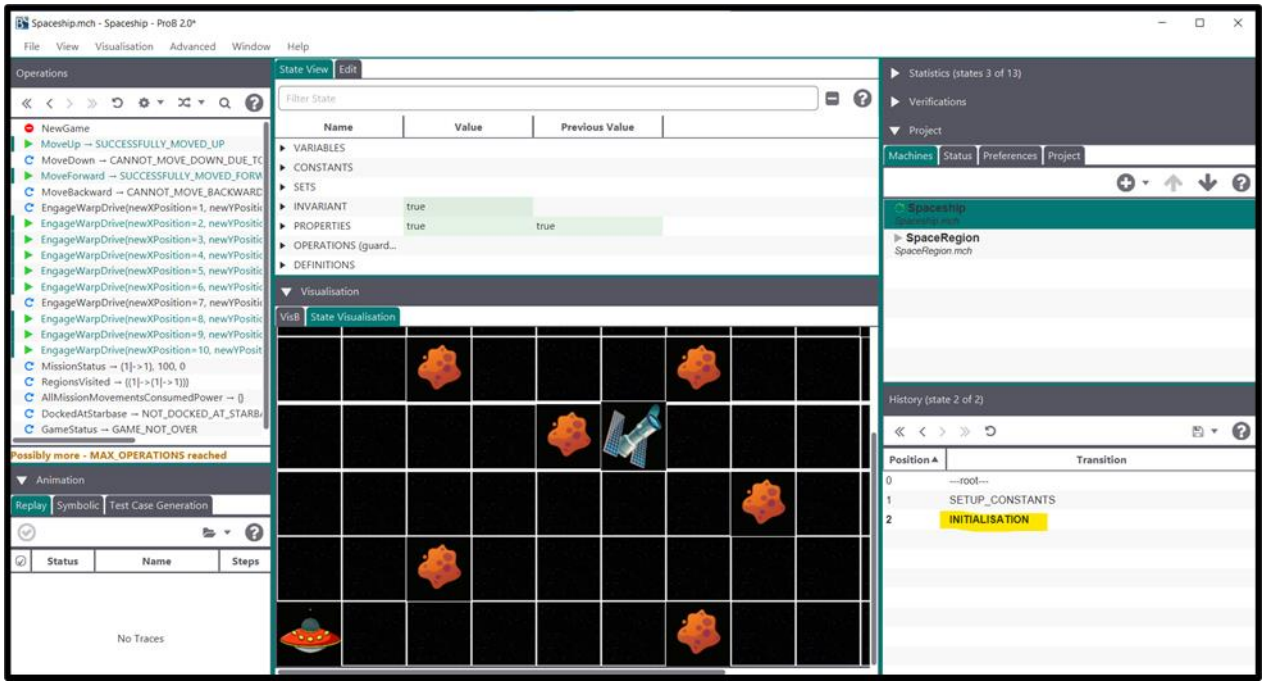
Figure 4 INITIALISATION

3.0 Implementation Screenshots

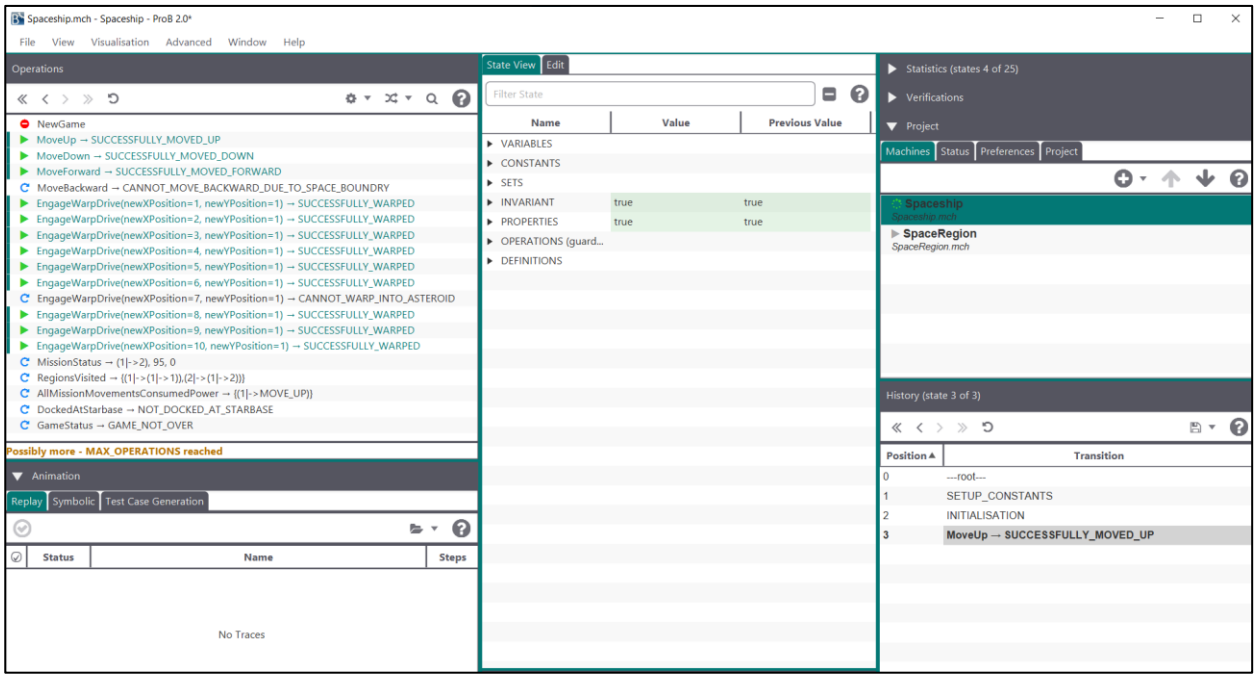
1. Initialising the Game

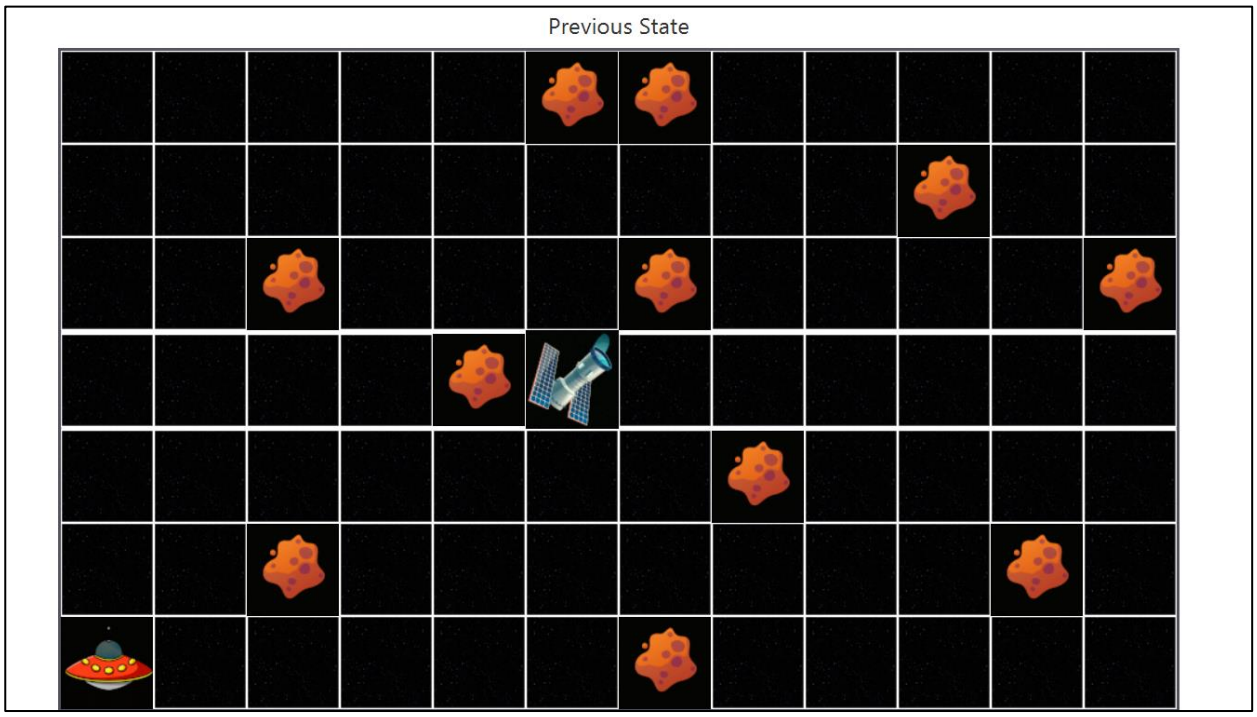


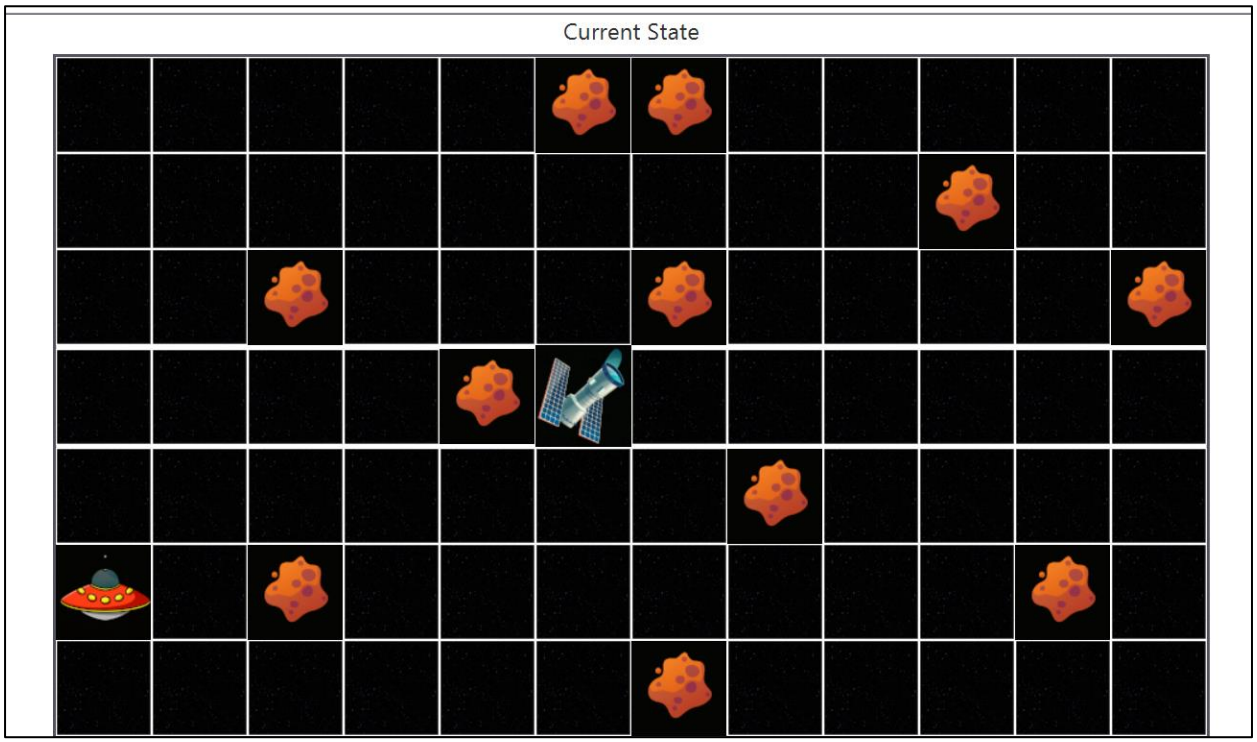
2. Start of the Game



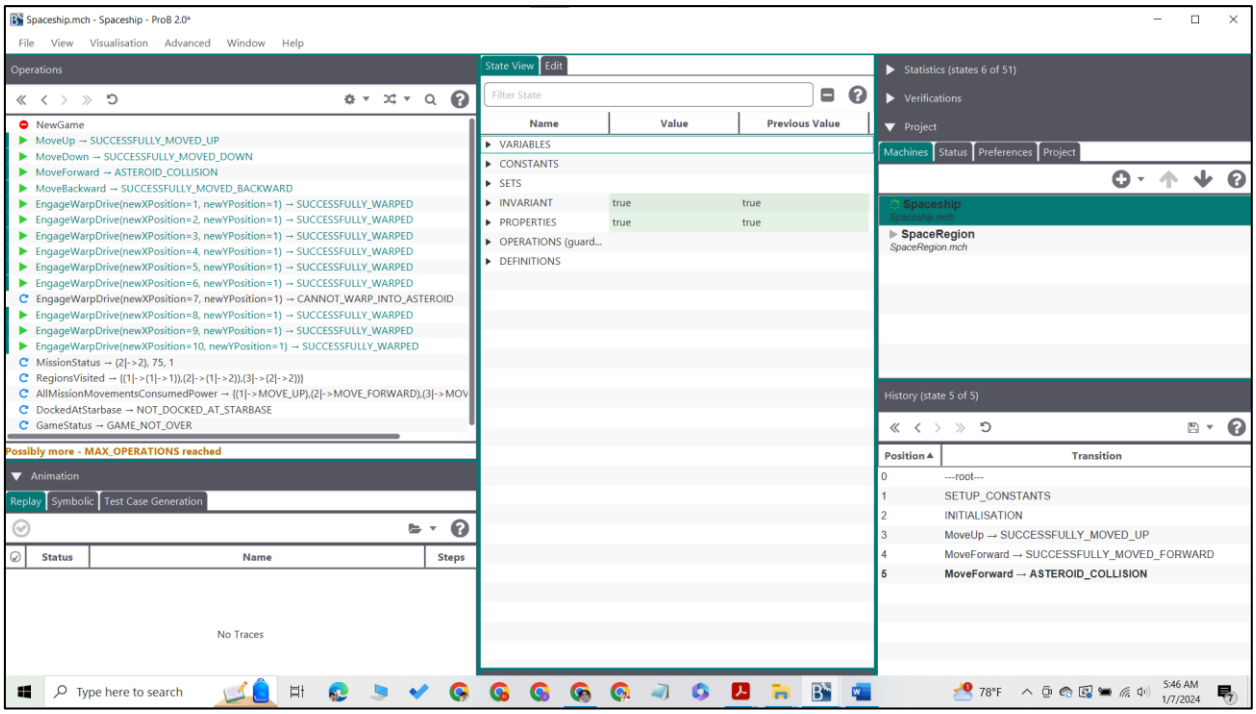
3. Successful Normal Movement -MOVE_UP

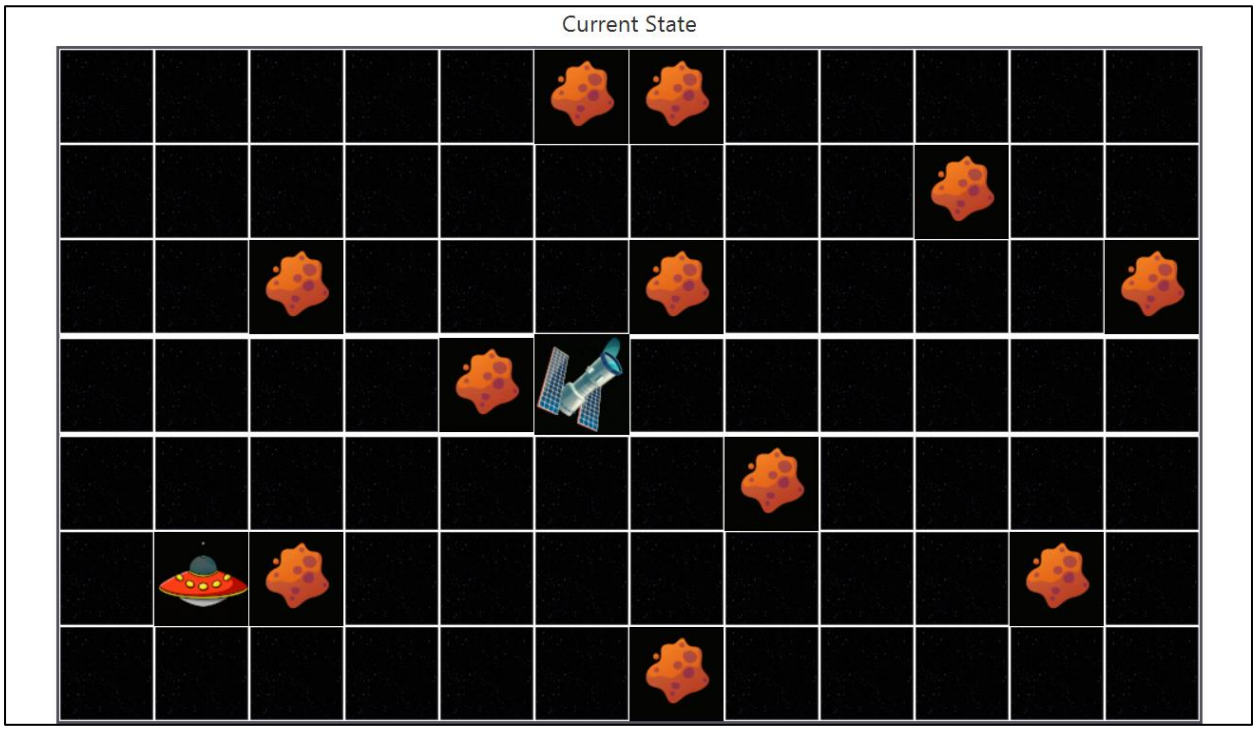




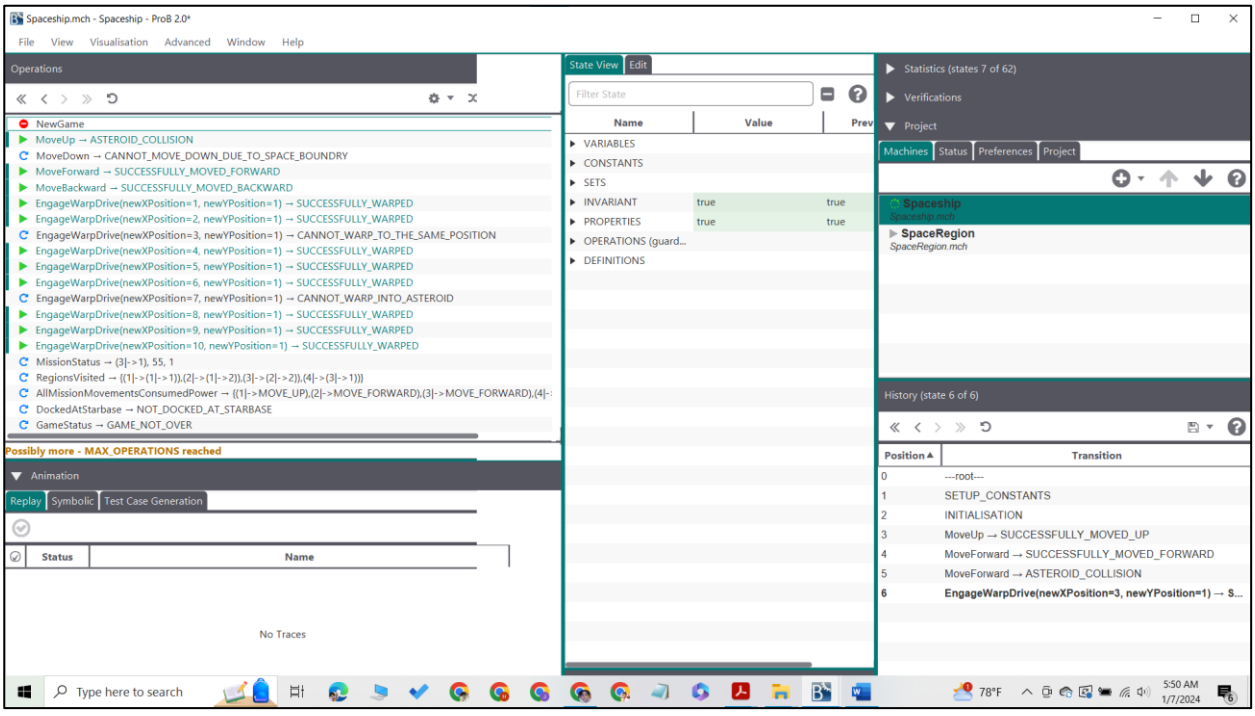


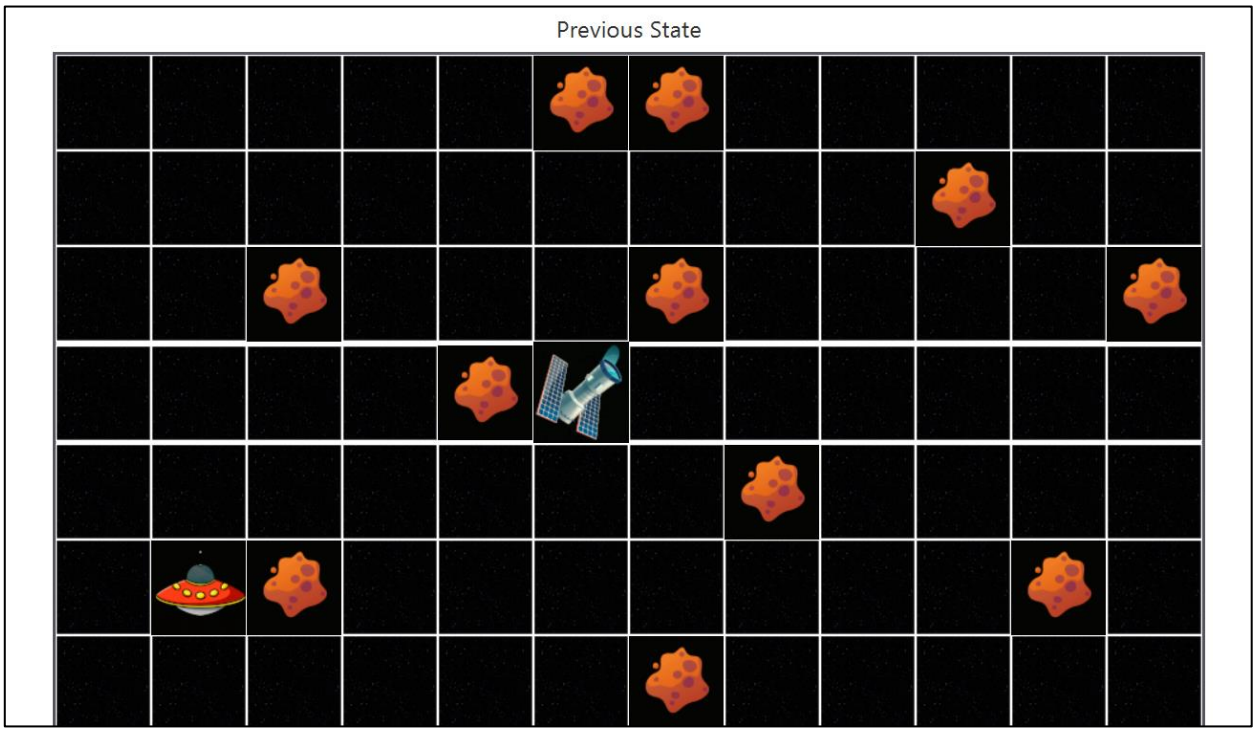
4. Failed movement due to unknown space and asteroid collision

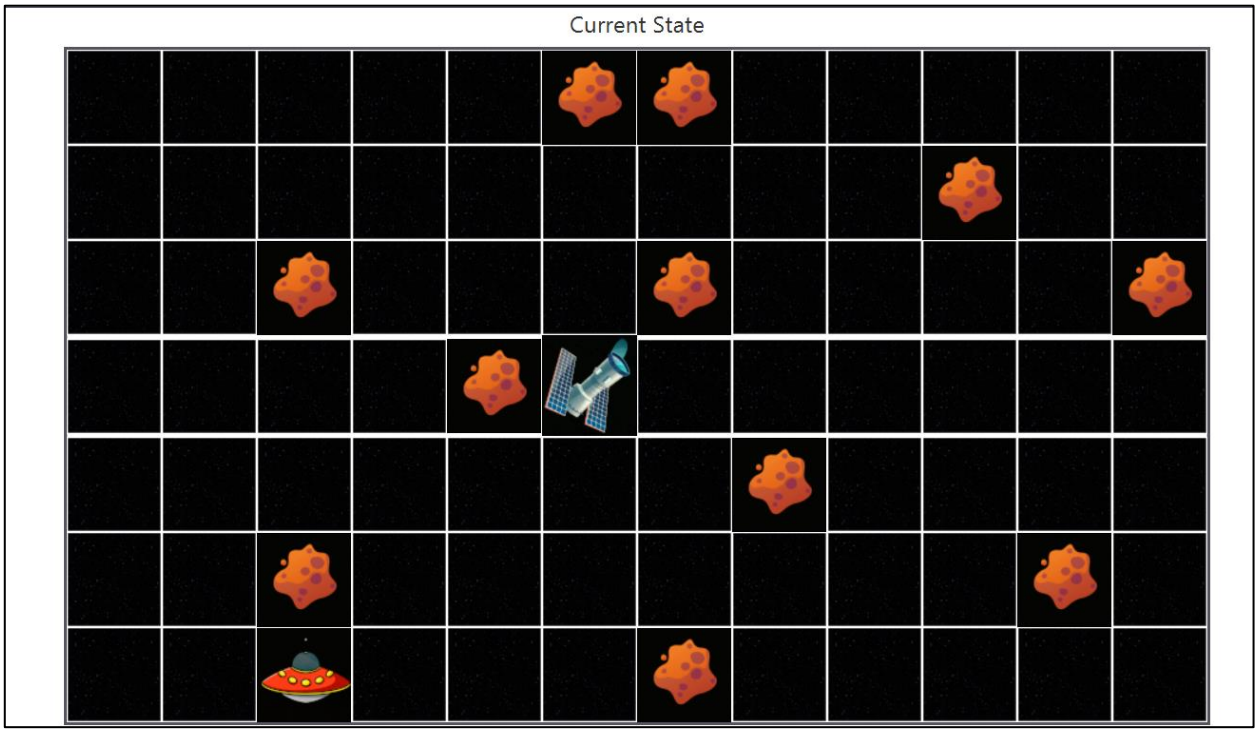




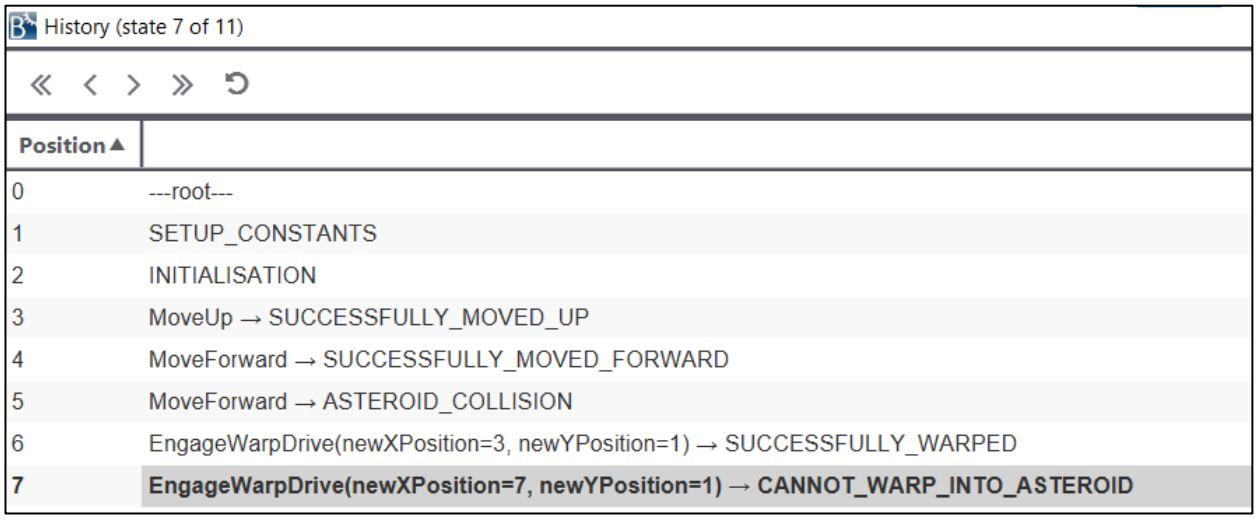
5. Successful warp jump




Previous State


Current State

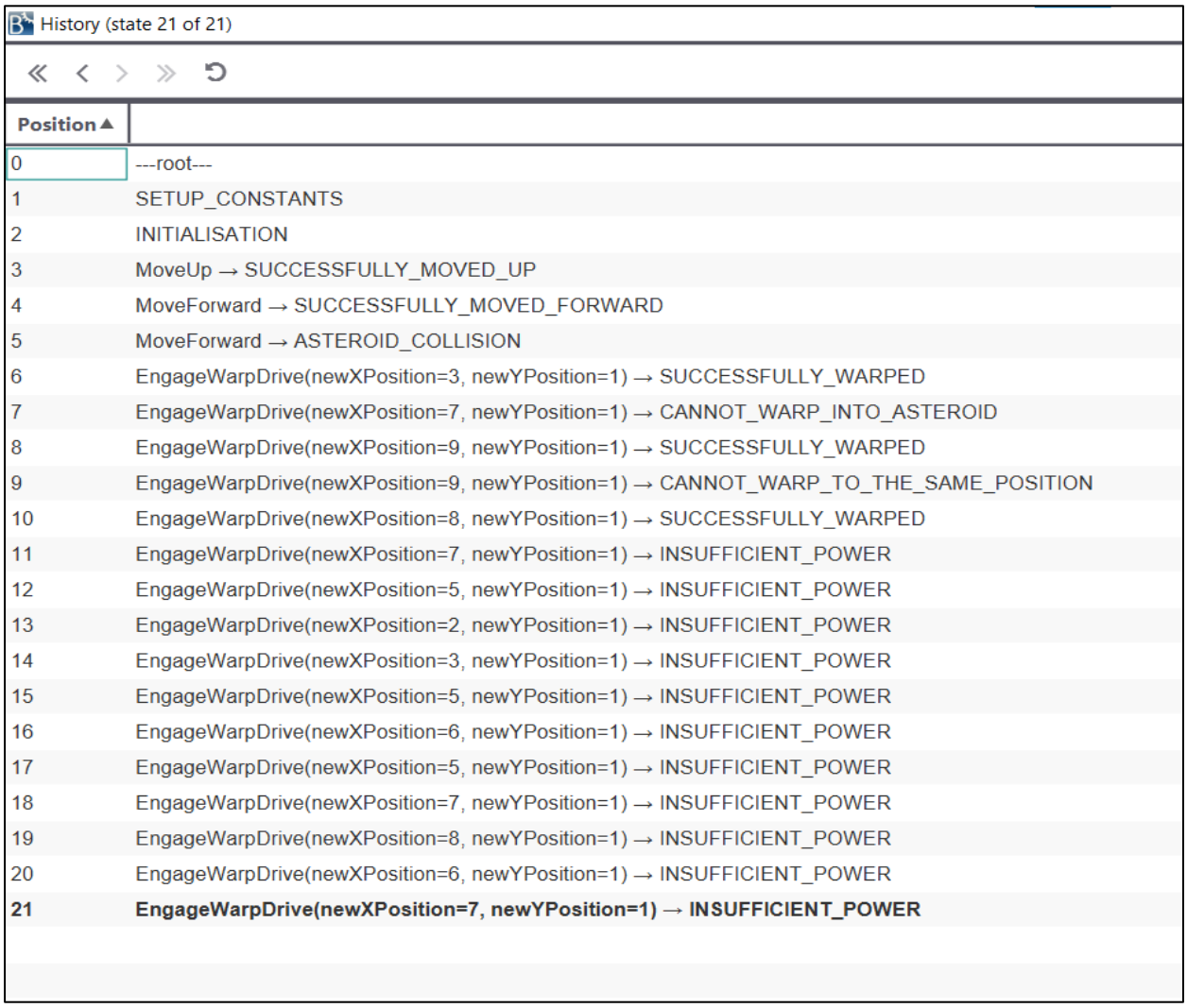
6. Failed warp jump due to same position, asteroid collision and unknown space errors



History (state 7 of 11)

Position ▲	
0	---root---
1	SETUP_CONSTANTS
2	INITIALISATION
3	MoveUp → SUCCESSFULLY_MOVED_UP
4	MoveForward → SUCCESSFULLY_MOVED_FORWARD
5	MoveForward → ASTEROID_COLLISION
6	EngageWarpDrive(newXPosition=3, newYPosition=1) → SUCCESSFULLY_WARPED
7	**EngageWarpDrive(newXPosition=7, newYPosition=1) → CANNOT_WARP_INTO_ASTEROID**

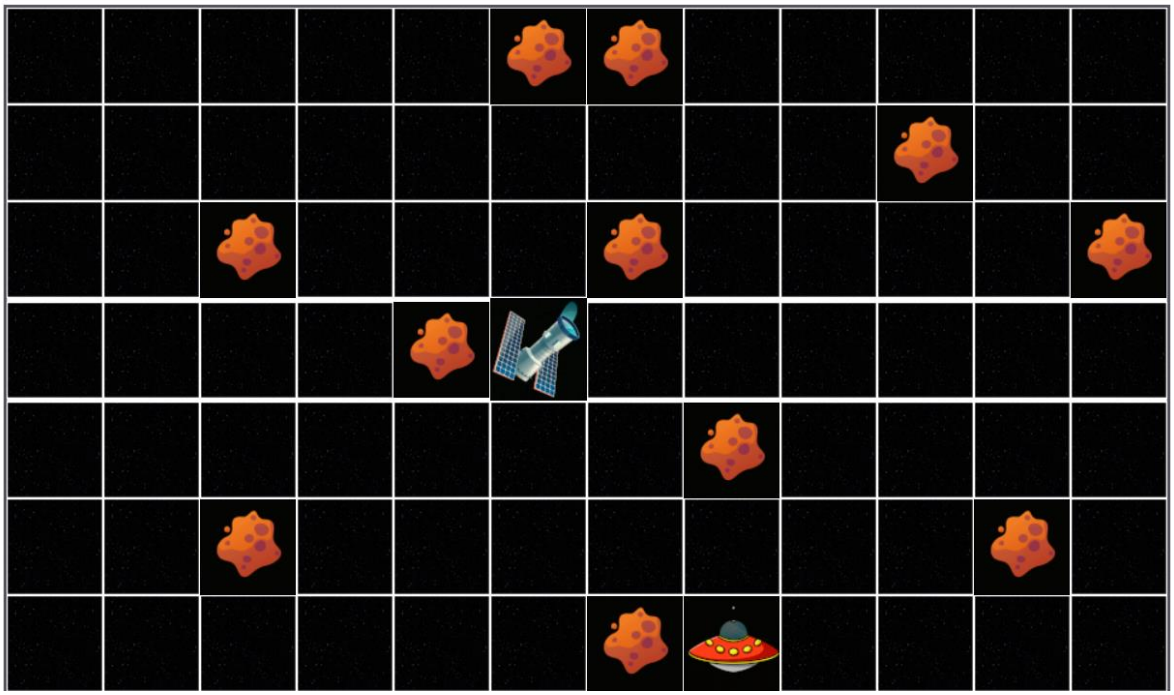
7. Failed warp jump due to insufficient power



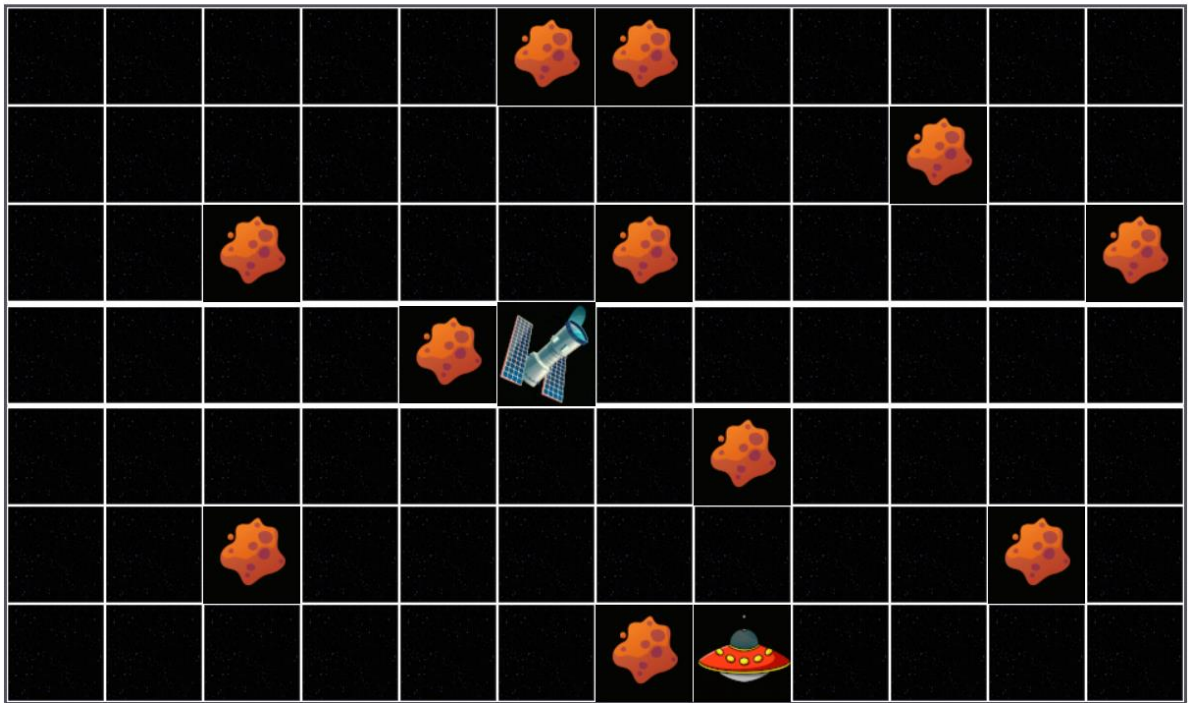
History (state 21 of 21)

Position ▲	
0	---root---
1	SETUP_CONSTANTS
2	INITIALISATION
3	MoveUp → SUCCESSFULLY_MOVED_UP
4	MoveForward → SUCCESSFULLY_MOVED_FORWARD
5	MoveForward → ASTEROID_COLLISION
6	EngageWarpDrive(newXPosition=3, newYPosition=1) → SUCCESSFULLY_WARPED
7	EngageWarpDrive(newXPosition=7, newYPosition=1) → CANNOT_WARP_INTO_ASTEROID
8	EngageWarpDrive(newXPosition=9, newYPosition=1) → SUCCESSFULLY_WARPED
9	EngageWarpDrive(newXPosition=9, newYPosition=1) → CANNOT_WARP_TO_THE_SAME_POSITION
10	EngageWarpDrive(newXPosition=8, newYPosition=1) → SUCCESSFULLY_WARPED
11	EngageWarpDrive(newXPosition=7, newYPosition=1) → INSUFFICIENT_POWER
12	EngageWarpDrive(newXPosition=5, newYPosition=1) → INSUFFICIENT_POWER
13	EngageWarpDrive(newXPosition=2, newYPosition=1) → INSUFFICIENT_POWER
14	EngageWarpDrive(newXPosition=3, newYPosition=1) → INSUFFICIENT_POWER
15	EngageWarpDrive(newXPosition=5, newYPosition=1) → INSUFFICIENT_POWER
16	EngageWarpDrive(newXPosition=6, newYPosition=1) → INSUFFICIENT_POWER
17	EngageWarpDrive(newXPosition=5, newYPosition=1) → INSUFFICIENT_POWER
18	EngageWarpDrive(newXPosition=7, newYPosition=1) → INSUFFICIENT_POWER
19	EngageWarpDrive(newXPosition=8, newYPosition=1) → INSUFFICIENT_POWER
20	EngageWarpDrive(newXPosition=6, newYPosition=1) → INSUFFICIENT_POWER
21	**EngageWarpDrive(newXPosition=7, newYPosition=1) → INSUFFICIENT_POWER**

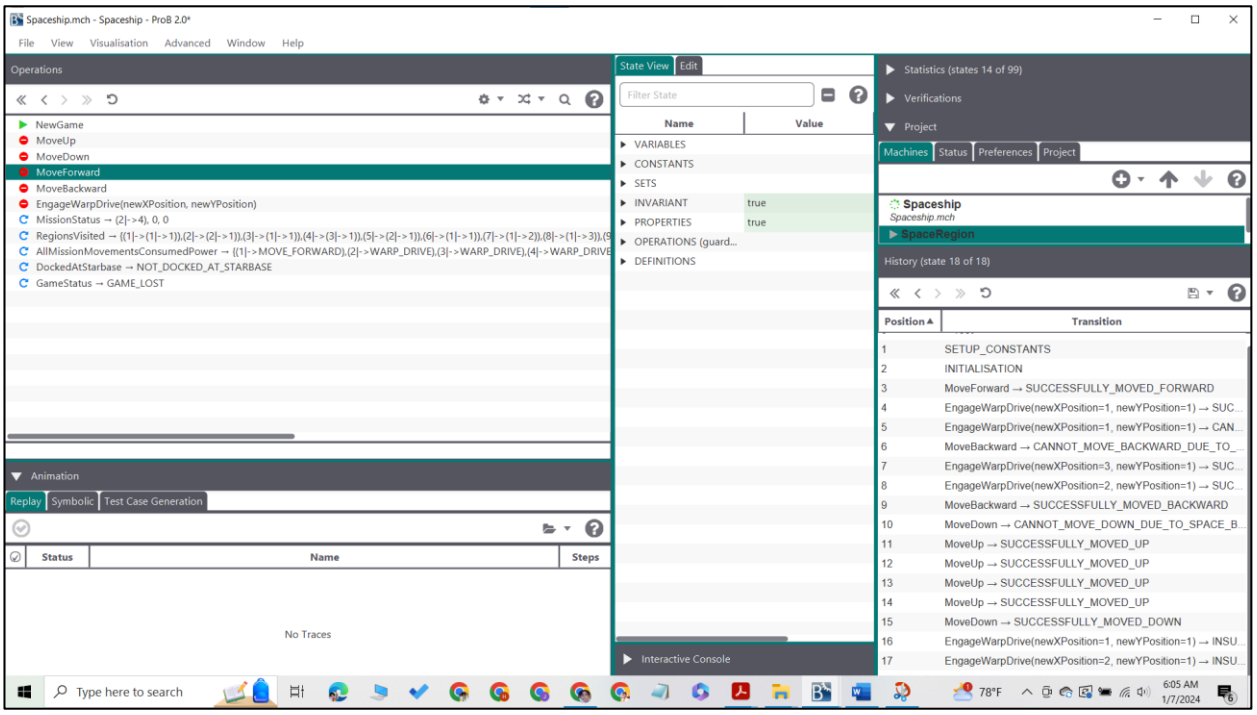
Previous State



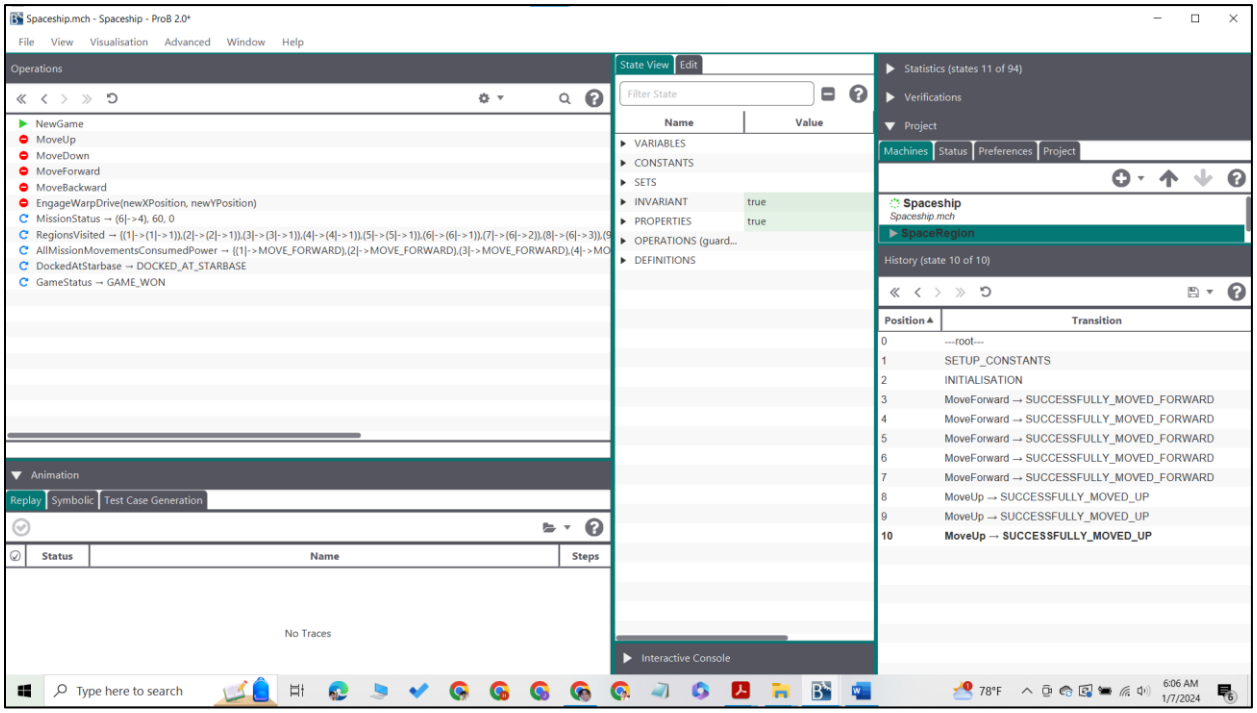
Current State



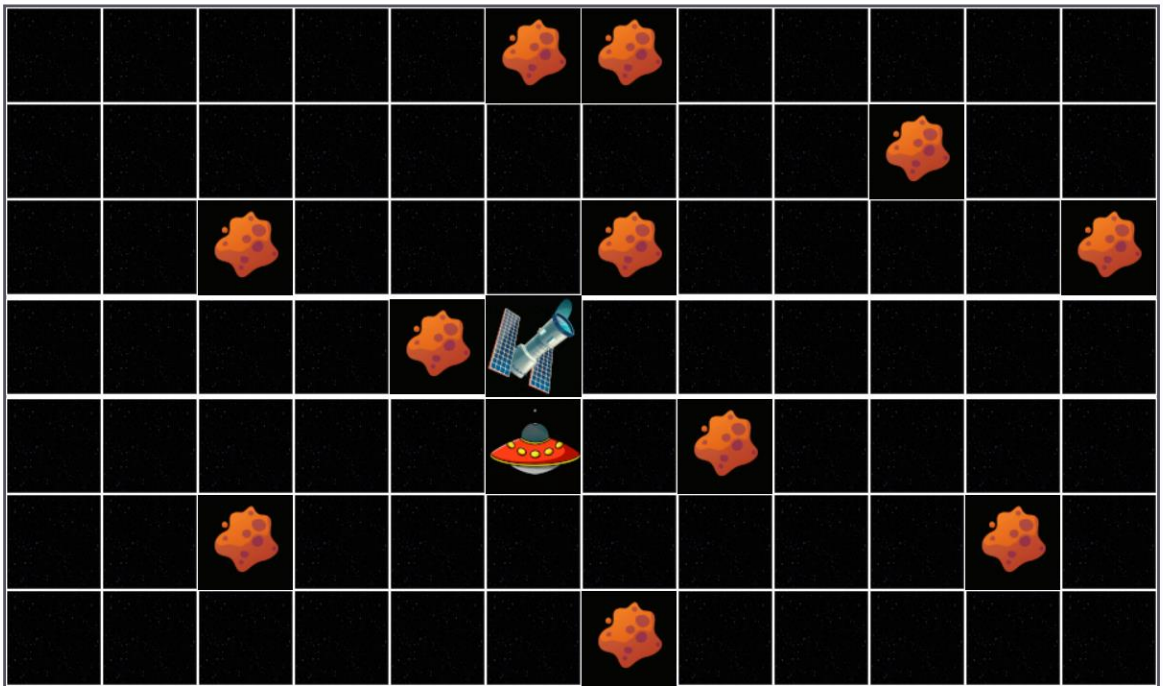
8. Game Lost



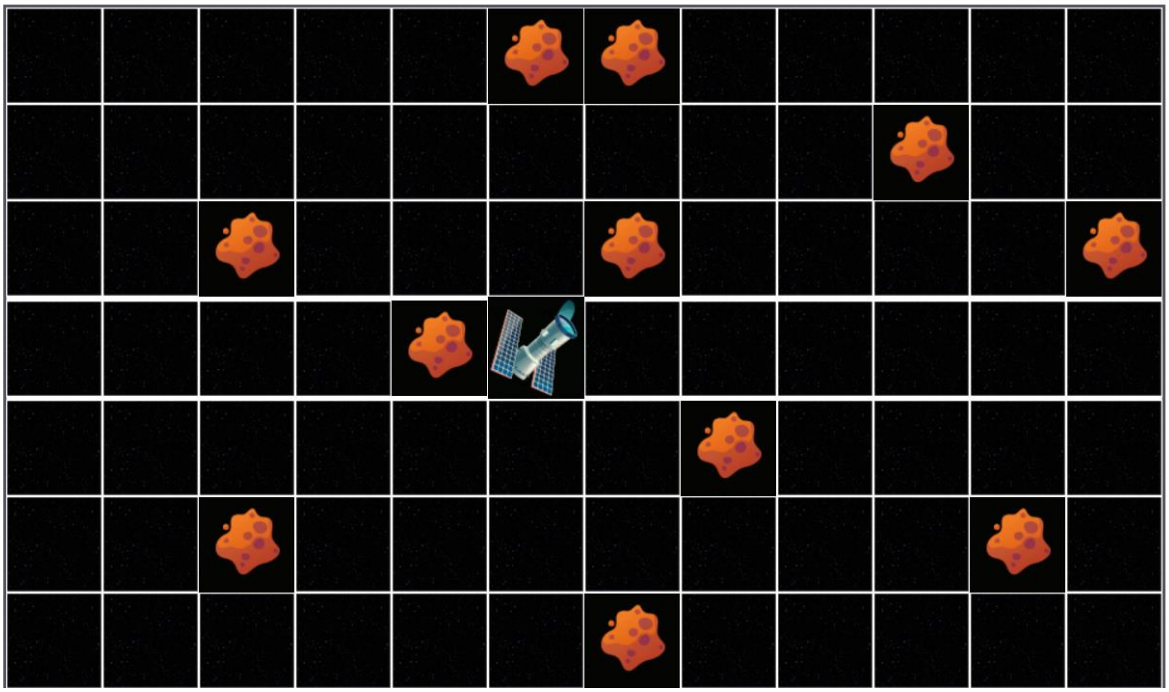
9. Game Won



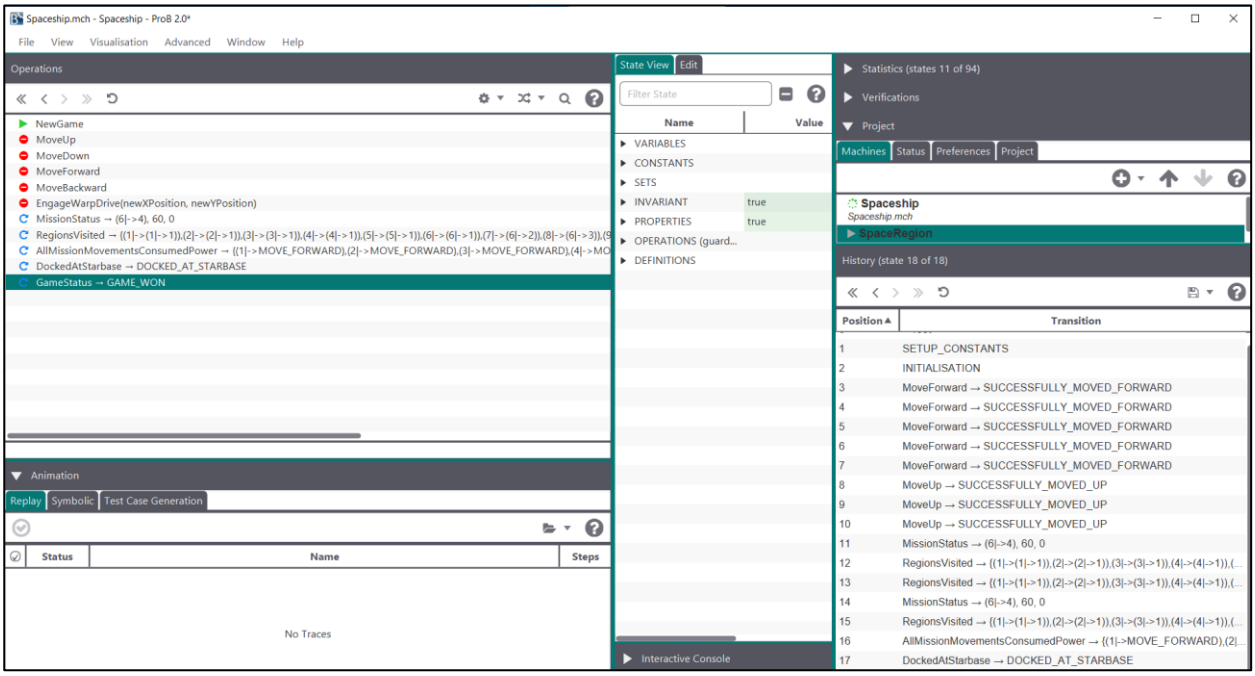
Previous State



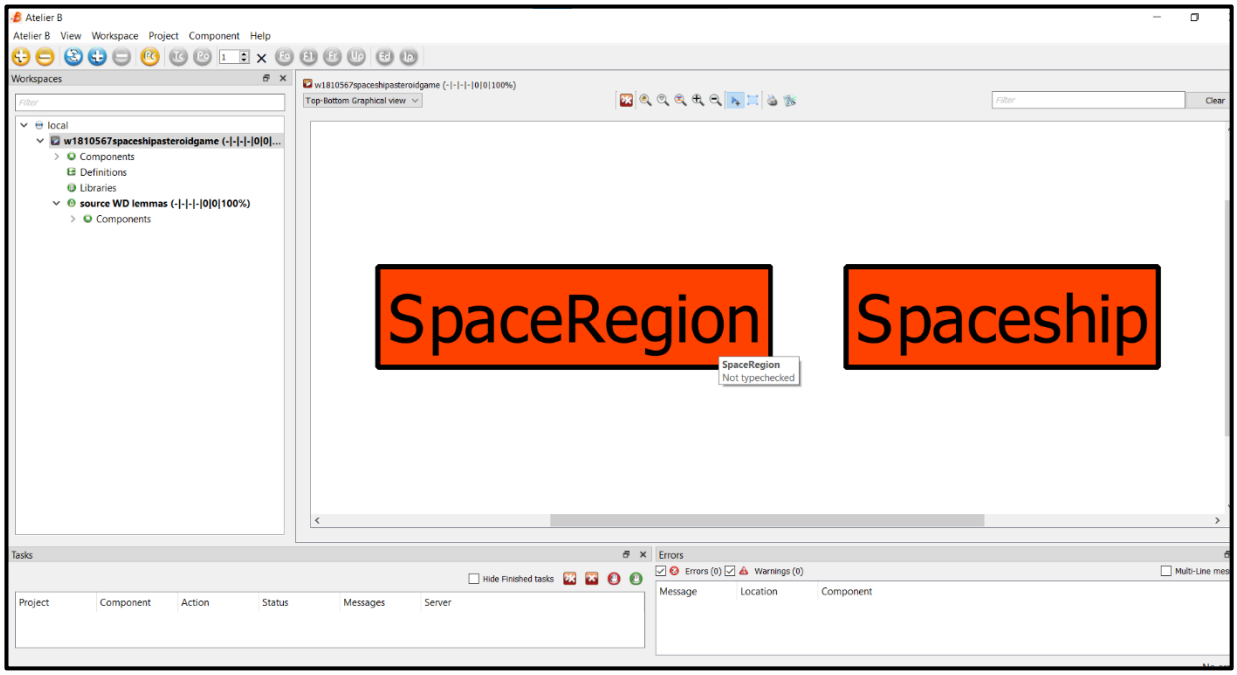
Current State



10. Status of the mission, Spaceship visited regions, Spaceship docked at starbase or not



11. Screenshot of the Atelier B before Type Check



12.Screenshot of the Atelier B after Type Check