Development and Proof of a Submarine Copntrol System

Dylan Tyrie-Dron

SET10112

Abstract. This is where your abstract would go.

1 Introduction

A submarine system was implemented using Ada and was proven by SPARK. The submarine system was implemented based on the following rules:

- The submarine must have at least one airlock door closed at all times.
- The submarine can perform no operations unless both doors are closed and locked.
- If the oxygen runs out, the submarine has to surface.
- If the reactor overheats, the submarine has to surface.
- If the oxygen runs low, a warning must be shown.
- The submarine cannot dive beneath a certain depth.
- The submarine must be capable of storing, loading and firing torpedoes safely.

The system must carry out these requirements to a certain proven standard of SPARK. The system will aim to meet the platinum standard with all requirements covered.

2 Controller Structure

The submarine consisted of one air-locked door, an oxygen level indicator, a depth level indicator, reactor level indicator and a torpedo chamber. The separate components were put into different files to categorise the code.

The submarine file was the key file that brought all the other files together. It also contained code for the submarine's movement.

A file was created for the control of the doors of the submarine. This set up how the doors would behave and was included in the submarine file to be used by the submarine to prevent any action occurring when both doors were not closed.

Another file was created for the reactor which would control the temperature of the reactor. It was included in the submarine file to be later called when the submarine increased in depth.

A file was created for the torpedo. It controlled the firing of torpedoes by setting up a chamber with two hatches. The process of firing a torpedo involved the opening of the outer hatch the firing of the torpedo and then the closing of the outer hatch. The process of loading the torpedo involved opeing the inner hatch and then storing the torpedo.

A file was created for the control of the oxygen level. The oxygen level was decreased as the submarine increased in depth.

3 Desciptions of Procedures and Functions

The controller consists of many small parts. The first thing to do was to make a *submarineType* record. The record would be used to store an instance of a submarine and allow, for example, it's depth to be changed. The *submarineType* contained an array of type "Door" which contained two doors that formed an air-lock. The oxygen level was updated using an integer with initial value of 8. An integer was also set up to update the *depthLevel* and the *reactorLevel* both with initial values set to zero.

3.1 First Requirement

The first requirement was to make sure that at least one door was closed at all times. This was done by creating a Door record with a *doorStatus* variable attached to it to set the door to either closed or open. This door also had a lock attached to it, the lock had a *lockStatus* to set the door to be locked or unlocked. An array of Doors was set up with a range of 2.

A procedure was created to control the locking of doors: *closeDoors*, the procedure closes both doors of the submarine. The procedure took as input an array of doors. Every door of the array was closed and locked in the procedure's body, this was then made certain by checking that the status of every door and lock in the array was closed and locked using a post-conditon. An invariant was set up to use in pre-conditions for other functions in the submarine file that checked if both doors in the array were closed and locked. The invariant was placed in the submarine file so that it could be used by the functions of the submarine to check that the doors are locked before starting each function.

3.2 Second Requirement

The second requirement was to ensure that the submarine could not perform any operations unless two doors were locked and closed (airlocked). The second requirement was handled using the same solution as the first requirement (using the *closeDoors* procedure and invariant).

3.3 Third Requirement

For the third requirement, the submarine first had to have the ability to change depths. Secondly, it had to be ensured that a warning had to shown if the oxygen

levels of the submarine went below a certain threshold (3). The submarine also has to re-surface after the oxygen level is zero.

In the submarine file the movement was controlled. Three procedures were set up to handle the movement:

- *increaseDepth* - with parameters: depth level, oxygen level and an array of doors.
- *resurface* - with parameters: depth level, oxygen level, an array of doors and reactor temperature.
- *decreaseDepth* - with parameters: depth level, oxygen level and an array of doors.

The *increaseDepth* procedure contained a pre-condition that consisted of the oxygen level being higher than zero and lower than or equal to 8. It also made sure that the depth was higher than or equal to zero and lower than 6 and that both doors were closed. Through a post condition it also ensured that the depth would increase one higher than the old depth and that the depth would not go any higher than 6.

The *resurface* procedure contained a pre-condition that consisted of the oxygen level either equalling zero or was higher than zero and the reactor temperature was 12. The depth had to be higher than zero and both doors were closed. Through a post-condition of the depth level being zero a while loop ensured that the submarine would go straight back up to the surface.

The *decreaseDepth* procedure contained a pre-condition that consisted of the oxygen level being higher than or equal to zero and the depth level was higher the first instance of the depth level and that both doors of the submarine were closed. A post-condition consisting of the depth level being 1 less than the old depth level ensured that the submarine decreased in depth.

The submarine was also set up to manage oxygen levels. The oxygen level was decreased using a procedure called *decrementOxgen* it took in an oxygen level and decremented it. If the oxygen was low enough (3), it sends a warning to the submarine. A pre-condition was set up to check if the oxygen level was lower than or equal to 8. A post-condition ensured that the oxygen level would decrease which consisted of the oxygen level being 1 less than the old oxygen level and the oxygen level was higher than or equal to zero.

3.4 Fourth Requirement

The fourth requirement is that the submarine must surface if the nuclear reactor overheats. Another file was created to keep the reactor code separate. A procedure was set up to increase the heat of the reactor by incrementing the inputted variable *reactorLevel*. A pre-condition was set up to check that the reactor temperature was less than 12. The post-condition contained the reactor temperature being 1 higher than the old reactor temperature and that the reactor temperature was less than or equal to 12.

3.5 Fifth Requirement

Another requirement is that the submarine cannot dive beneath a certain depth. This was handled from within the *increaseDepth* procedure in the submarine file.

3.6 Sixth Requirement

The last requirement was to make sure the submarine could load, store and fire torpedoes safely. Another file to separate the torpedo code from the main submarine code was created. In the torpedo file the following was set up:

- *torpedoType* that has a *torpedoStatus*.
- *torpedoChamber* that contains a torpedo and 2 hatches.
- *torpedoes* an array of torpedoes with size of 2.
- *hatch* that has a *hatchStatus*.
- *hatches* an array of hatches with size of 2.
- *torpedoStatus* which has 3 states fired, stored, loaded.
- *hatchStatus* which is either open or closed.

4 Proof of Consistency

Proof of consistency was used to test and prove that the code used to make sure the requirements were met.

4.1 First Requirement

The first requirement was to make sure that at least one door is closed at all times. To check if the requirement was met, a test was set up. The test involved looping through all the doors and setting their doorStatuses to open. The result was that

4.2 Second Requirement

The second requirement was to ensure that the submarine could not perform any operations unless two doors were locked and closed.

4.3 Third Requirement

For the third requirement the submarine first had to have the ability to change depths. Secondly, the submarine had to ensure that when oxygen levels went below a certain threshold a warning will show. After this, the submarine has to re-surface after the oxygen is empty.

4.4 Fourth Requirement

The fourth requirement instructed is that the submarine must surface if the nuclear reactor overheats.

4.5 Fifth Requirement

Another requirement is that the submarine cannot dive beneath a certain depth.

4.6 Sixth Requirement

The last requirement was to make sure the submarine could load, store and fire torpedos safely.

5 Conclusion

To sum up the submarine system was fully implemented with some additional functionality. It was proved by SPARK to be of a platinum standard of code. However, the way the system outputted to a console was not very user engaging. This could be improved by printing the system output in a while loop every few seconds to give a more visual representation to the user that the system works and what the system actually does. This could be further enhanced by allowing the user to control the input of the submarine using keyboard controls within this while loop. The system could also be enhanced by improving certain post-conditions. The problem with the system's torpedo system is that the amount of torpedoes are not tracked. This could be enhanced by creating an integer to count the number of torpedoes that have been fired and making sure that integer is conditioned accordingly to make sure that the count is at the right number.