Space Station

Stephen Messer

Edinburgh Napier University
Edinburgh EH10 5DT, United Kingdom
40314216@live.napier.ac.uk

Abstract. This paper discussed a proposed control system application for a space station using the Ada-SPARK language. This first section aims to provide a high-level overview of the problem and the solution implemented to solve it. The next section provides a high-level overview of the space-station control system, defining and justifying the chosen structure. The next section provides a lower-level view of the individual components of the system, including parameters, constants, post-conditions, and pre-conditions. The next sections described critical parts of the system which have been formally verified. Full proof of parts of the system has been provided using sequent calculus. The next section provides a safety plan for the developed system including a hazard and risk analysis. The next section provides a safety case using goal structuring notation.

Keywords: space, Ada

1 Introduction

The act of sending humans into space entails numerous potential risks and hazards, where even minor errors in any system utilized during the process can result in catastrophic consequences. Therefore, it is imperative to ensure that the life-critical systems utilized in the process are reliable and trustworthy and provably so.

The implemented system has been designed using the Ada-SPARK language to control some of the actions that may take place on a space station that could be unsafe if carried out incorrectly. Particularly the following actions have been covered:

- Opening of airlock doors.
- Adjusting the orbit height of the station.
- Adding and removing modules from the station.
- Determining if it is safe for a crew member to carry out a spacewalk.

Each component has been implemented with careful consideration of the preconditions and post-conditions that must hold for the system to function correctly and safely. These components of the system have been formally verified to prove the consistency of the system and their implementation has been justified.

A safety plan has also been created which includes a hazard and risk, a list of proposed mitigation measures and a failure analysis. Also, a safety case using goal structuring notation has been created to explicitly document the elements and structure of the arguments and their relationship to evidence. A safety manual has also been created to document any relevant information about the proposed system.

2 Controller Structure

Using Ada, a record has been created to model a space station. This record contains parameters relevant to carrying out and monitoring the defined actions. These parameters are laid out in Table 1. This record is available to the system via a global variable.

Table 1. Different parameters of the MonitoringStation record.

Parameter Name	Type	Derivation	Function
airLock1	Enumerated	Open/Closed	Represent the status of the airlock
airLock2	Enumerated	Open/Closed	Represent the status of the airlock
height	OrbitHeight	Range 0-2000(km)	Monitor the current orbiting height
modules	ModuleStack	Array of Modules	Keep track of the station's modules and their removal or addition
moduleCount	ModuleRange	Range 1-20	Keep track of the current number of modules.

A global variable of the 'MonitoringStation' type has been initialized in the Ada specification file. A global variable has been chosen so that there is only one point of data access for all subprograms and data can be shared between the subprograms without passing around data between them.

Where declared types only have a few valid values, enumerated types have been used which may help improve readability and maintainability of the code and help catch errors at compile time. The status of the stations two airlocks has been represented using the 'AirLockStatus' enumerated type which can have the value of 'Open' or 'Closed'. This has been chosen as it is the two logical states of the air locks.

The height of the station has been represented with a range of between 0-2000. This has been chosen to cover all situations the station may reasonably find it's self in. Constant variables MINIMUM_HEIGHT and MAXIMUM_HEIGHT define the range in which it is safe for the station to operate.

A stack data structure has been chosen to represent the collection of modules that the station consists of. This data structure has been used for its First In, Last Out nature which will be useful for fulfilling the systems requirements regarding the addition of modules. An array of 'Module' types is maintained through the 'modules' variable and 'moduleCount' acts as a pointer to the top of the stack allowing push and pop operations to take place. 'moduleCount' is a value within the range of 1-20. This is to cover the different number of modules that the station may consist of. The

‘Module’ type has been defined to represent a module on the station. This type has a value between 0 and 3 which represents the crew occupancy of the module.

Additionally, the ‘Move’ type has been defined. This type represents the actions crew members can take to move between the modules. It can have the value of ‘Left’ or ‘Right’ since the crew can only move to an adjacent module in the stack.

3 Description of Procedures and Functions

Two invariants have been defined to ensure that certain properties remain true throughout the execution of the program to prove the correctness of the program and ensure it functions as expected. These appear as pre-conditions and post-conditions to all the systems’ functions and procedures on top of their individuals ones listed below.

- **AirLockInvariant** – this ensures that at least one of airlock1 or airlock2 is always closed.
- **OrbitHeightInvariant** – this ensures that the stations orbit height always remains between the defined MINIMUM_HEIGHT and MAXIMUM_HEIGHT constants.

openAirLock1 - Sets the status of airLock1 to Open.
- Pre-conditions:
  - airLock2 must be Closed.
- Post-conditions:
  - airLock1 must be Open.
  - airLock2 must be Closed.

openAirLock2 - Sets the status of airLock2 to Open.
- Pre-conditions:
  - airLock1 must be Closed.
- Post-conditions:
  - airLock2 must be Open.
  - airLock1 must be Closed.

closeAirLock1 - Sets the status of airLock1 to Closed.
- Post-conditions:
  - airLock1 must be Closed.

closeAirLock2 - Sets the status of airLock2 to Closed.
- Post-conditions:
  - airLock2 must be Closed.

increaseHeight – increases the stations height by 1.

- Pre-conditions:
  - The stations must not be at its maximum height.
- Post-conditions:
  - The stations height is one more than before.

decreaseHeight – decreases the stations height by 1.
- Pre-conditions:
  - The stations must not be at its minimum height.
- Post-conditions:
  - The stations height is one less than before.

LastModule – returns whether the number of modules the stations has is at its minimum.

Full – returns whether the station has reached its maximum number of modules.

pushModule – adds a new module to the space station.
- Pre-conditions:
  - The station is not at the maximum number or modules.
- Post-conditions:
  - The number of modules in the station has increased by one.

popModule – removes a module from the space station.
- Pre-conditions:
  - The station is not at the minimum number or modules.
  - The module to be removed is not occupied.
- Post-conditions:
  - The number of modules in the station has decreased by one.

safeSpaceWalk – returns whether it is safe to carry out a spacewalk i.e., if the other two members of the crew are at opposite ends of the station.

crewMove – moves a member of the crew from a given module to an adjacent module.
- Pre-conditions:
  - The given module is occupied.
  - It is not the case that the given module is the furthest left, and the occupying crew member is trying to move to the left.
  - It is not the case that the given module is the furthest right, and the occupying crew member is trying to move to the right.
- Post-conditions:
  - The number of occupants of the given module has decreased by one.
  - The adjacent module indicated by the given move has increased occupancy by one.

4 Proof of Consistency

All the procedures in the previous section have been proven to the Ada-SPARK gold level as demonstrated in the demo video.

The maintenance of a safe orbit height is an important part of the control system due to the consequences if it behaves unexpectedly. In particular, the increaseHeight procedure must reliably increase the station's height whilst ensuring it does not exceed the maximum height. The proof obligations for this procedure is laid out in Fig 1.

M - height < MAXIMUM_HEIGHT
H - height
A - AirlockInvariant
O - OrbitHeightInvariant

$$\dfrac{\dfrac{\dfrac{\dfrac{Ax}{M, H = H+1, A, O \Rightarrow H = H+1} \quad \dfrac{\dfrac{Ax}{M, H = H+1, A, O \Rightarrow A} \quad \dfrac{Ax}{M, H = H+1, A, O \Rightarrow O}}{M, H = H+1, A, O \Rightarrow A \wedge O}\, R\wedge}{M, H = H+1, A, O \Rightarrow H = H+1 \wedge A \wedge O}\, R\wedge}{M \wedge H = H+1 \wedge A \wedge O \Rightarrow H = H+1 \wedge A \wedge O}\, L\wedge *}{\Rightarrow M \wedge H = H+1 \wedge A \wedge O \supset H = H+1 \wedge A \wedge O}\, R\supset$$

Figure 1. proof of consistency for increaseHeight procedure.

Another system that must be trustworthy is the component that removes modules from the station – popModule. The proof of obligations for this procedure is laid out in Fig 2.

L - moduleCount = ModuleRange'First (minimum number of modules has been reached)
U - modules[moduleCount] = 0 (the popped module is unoccupied)
A - AirlockInvariant
O - OrbitHeightInvariant
C - moduleCount = moduleCount - 1

$$\dfrac{\dfrac{\dfrac{\dfrac{Ax}{\neg L, U, A, O, C \Rightarrow C} \quad \dfrac{\dfrac{Ax}{\neg L, U, A, O, C \Rightarrow A} \quad \dfrac{Ax}{\neg L, U, A, O, C \Rightarrow O}}{\neg L, U, A, O, C \Rightarrow A \wedge O}\, R\wedge}{\neg L, U, A, O, C \Rightarrow C \wedge A \wedge O}\, R\wedge}{\neg L \wedge U \wedge A \wedge O \wedge C \Rightarrow C \wedge A \wedge O}\, L\wedge *}{\Rightarrow \neg L \wedge U \wedge A \wedge O \wedge C \supset C \wedge A \wedge O}\, R\supset$$

Figure 2. proof of consistency for popModule procedure.

5 Safety Plan

5.1 Hazard and Risk Analysis

Hazard and Operability Analysis (HAZOP). HAZOP guide words have been used to explore potential hazards in the developed system that may arise from deviations from its intended use.

Table 2. HAZOP analysis

No.	Element	Deviation	Possible Cause	Consequences	Safeguards
1	Air Locks	Failure of airlocks to properly seal.	Wear and tear, improper maintenance.	Loss of atmosphere and harm to occupants.	Have emergency response plan outlined so crew can react quickly to the danger.
2	Air Locks	Both air locks open at the same time.	Improper use or software error.	Loss of atmosphere and harm to occupants.	Use interlock system to ensure doors cannot be open simultaneously.
3	Air Locks	Either air lock does not open	Wear and tear, improper maintenance or software error.	Occupants unable to leave station, or crew member locked out.	Have multiple airlocks that grant access to the station.
4	Orbit Maintenance	No sensors to determine height.	Sensor failure through wear and tear or damage from impact of debris.	Unable to calculate necessary altitude adjustments.	Build redundancy into the sensor system to ensure data is always available.
5	Orbit Maintenance	No method to correct altitude.	Failure of thrusters through wear and tear, power failure or improper use.	Uncontrolled decent of the station and ultimate crash landing.	Regular maintenance schedules to ensure all components are functioning properly.
6	Orbit Maintenance	Station goes outside safe height	Human error, equipment malfunction.	Uncontrolled decent of station or station enters	Have automated safeguards that take over when unsafe

		range.		unstable orbit.	course identified.

Risk Evaluation. The hazards have been analyzed to determine their probability and severity before any safeguards have been implemented. A 5x5 risk matrix has been used to determine the ultimate risk category and whether they are acceptable.

Table 3. Risk evaluation for hazards identified in HAZOP.

Hazard	Severity and Probability	Risk Category
1	**Very High** severity due to the immediate risk to life. **Medium** probability.	**Very High**
2	**Very High** severity due to immediate risk to life. **Very High** probability due to potential for human error.	**Very High**
3	**High** severity due to interruption of operations and risk to life. **Medium** probability.	**High**
4	**Medium** severity due to the reduced confidence in decisions. **High** probability due to wear on equipment.	**High**
5	**Very High** severity due to loss of station and crew. **Medium** probability.	**Very High**
6	**Very High** severity due to loss of station and crew. **Very High** probability due to human error.	**Very High**

Due to the presence of many high and very high risks, the highest level of safety integrity will be required – SIL 4.

5.2 Mitigations

The following mitigations could be used to reduce the chances of the identified hazards and lessen their damage if they do happen.

- Build redundancy into the system for critical components to ensure the system is always operational and reliable (Hazards: 3,4,5)
- Comprehensive training can be provided to the crew, including the proper use of the system. This can help minimize the risk of incidents due to human error. (Hazard: 1,3)
- Regular maintenance of and inspection to ensure all components are functioning properly and reliable. (Hazards: 1,2,4,5,6)
- Develop automated fail-safe mechanisms to ensure that the system can automatically take corrective action if it detects any unexpected changes. (Hazards: 1,2,6)
- Develop rigorous software tests that fully cover the subprograms. (Hazards: 2,5)

5.3 Failure Analysis

Fault tree analysis has been carried out on some of the identified hazards in order to better understand how the system might fail and identify any residual risk after mitigation.

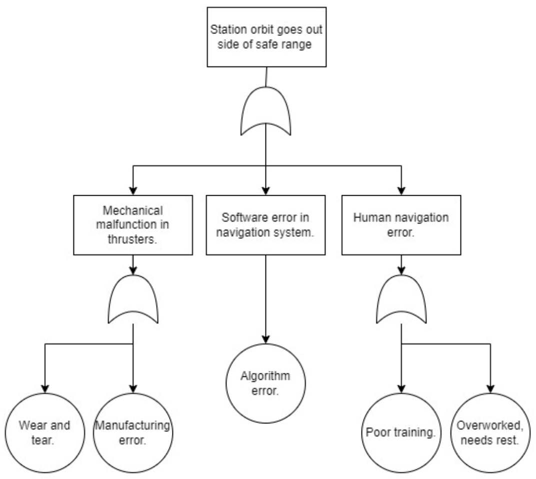
Station orbit goes out side of safe range
Mechanical malfunction in thrusters.
Software error in navigation system.
Human navigation error.
Wear and tear.
Manufacturing error.
Algorithm error.
Poor training.
Overworked, needs rest.

Figure 3. Fault tree analysis for hazard 6.

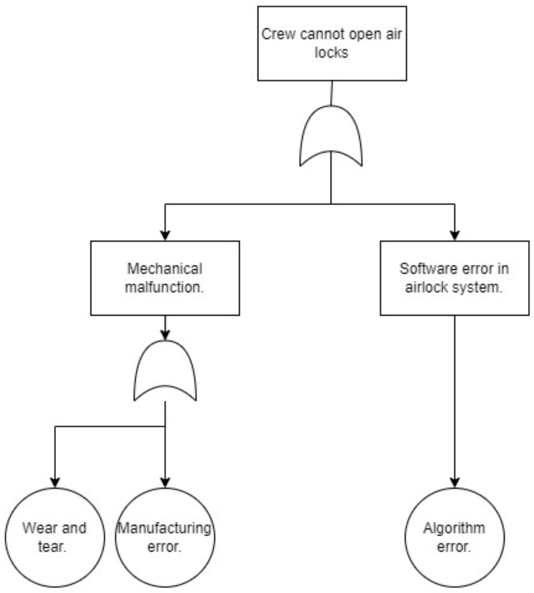

Figure 4. Fault tree analysis for hazard 3.

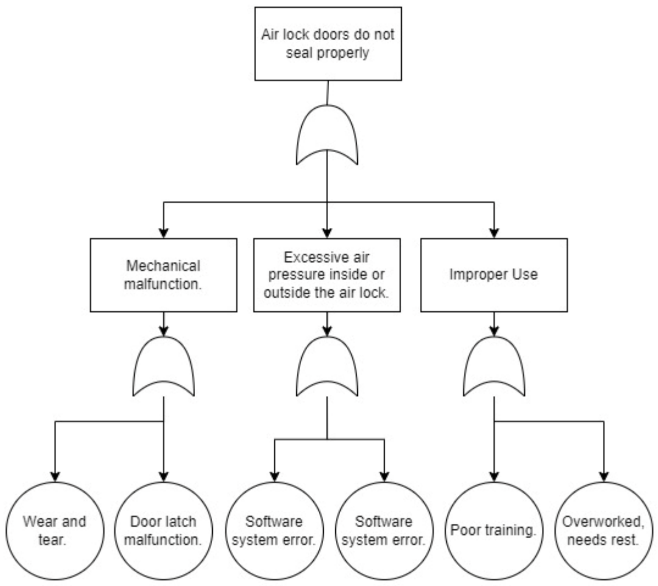

Figure 5. Fault tree analysis for hazard 1.

6 Safety Case and Safety Manual

6.1 Safety Case (GSN)

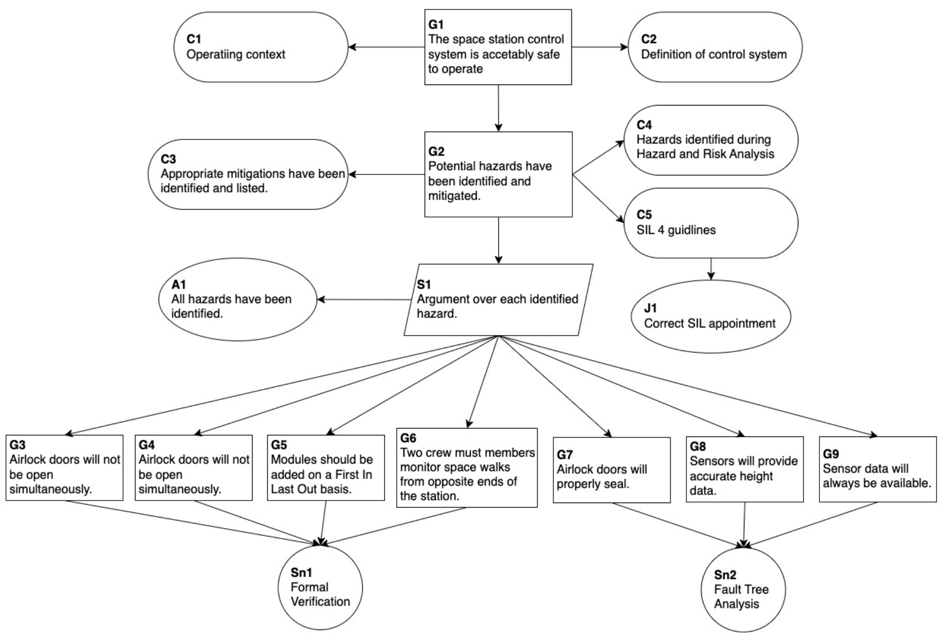
G1
The space station control system is accetably safe to operate
C1
Operatiing context
C2
Definition of control system
G2
Potential hazards have been identified and mitigated.
C3
Appropriate mitigations have been identified and listed.
C4
Hazards identified during Hazard and Risk Analysis
C5
SIL 4 guidlines
S1
Argument over each identified hazard.
A1
All hazards have been identified.
J1
Correct SIL appointment
G3
Airlock doors will not be open simultaneously.
G4
Airlock doors will not be open simultaneously.
G5
Modules should be added on a First In Last Out basis.
G6
Two crew must members monitor space walks from opposite ends of the station.
G7
Airlock doors will properly seal.
G8
Sensors will provide accurate height data.
G9
Sensor data will always be available.
Sn1
Formal Verification
Sn2
Fault Tree Analysis

Figure 6. Safety case argument for the system in goal structuring notation.

6.2 Safety Manual

Scope: The system is intended to be used on a fully functioning and operating space station.

7 Conclusion

To conclude, a system has been developed where all implemented procedures have been formally proved. One area that could be improved on further work is declaring pre-conditions for the functions in the system. Additionally, the functionality of the system does not cover the many more operations that may be carried out on an operational space station and more functions and procedures would need to be developed to cover these. Another way this system could be improved is by designing a safe state.