SAFETY ASSESSMENT OF STATIC SYSTEMS

Date: 20/01/2016

Developed by:

Inaki AGUILAR HERNANDEZ

Prerana SHAMSUNDAR PUNJABI

Table of Contents

Executive Summary

Dependability is a measure of a system's availability, reliability, and its maintainability. This also encompasses mechanisms designed to increase and maintain the dependability of a system.

Safety Assessment Process: Comprises of 4 stages FHA, PSSA, SSA and CCA.

Preliminary System Safety Assessment (PSSA) process - is systematic examination of proposed system architecture to determine how failures can lead to the functional hazards identified by the FHA, and how the FHA requirements can be met. The PSSA should identify failures contributing to the failure conditions from the system FHA. In the preliminary stage, possible contributing factors are identified by qualitative and/or quantitative assessment methods.

System Safety Assessment (SSA) is a systematic, comprehensive evaluation of the implemented system to show that relevant safety requirements are met.Fault Tree Analysis is structured, top-down approach that is utilized for qualitative assessments in either preliminary design or detail design and verification stage.

The DAL is the level of rigor of development assurance tasks performed on functions and items (software, hardware).

This assignment outlines problem statements were Failure Modes and Effects Analysis for the safety assessment of a static system is to be developed. Failure Modes and Effects Analyses were completed for the system to ensure that safety goals for the system had been meet.

The Static System comprises mainly of the following:

- Three Applications
- Each application is implemented by two task
- Each task is comprises of a number of computers
- A task fails if all its computers fail
- An application fails if both its task fail
- The system fails if at least one of the Application fails
- All failure conditions are Major
- Qualitative requirement: No single failure shall lead to the failure condition
- Quantitative requirement: The probability of failure condition shall be smaller than 10^5 / flight hour
- Probability of the loss of a computer is 2 * 10^3

For the three computing platforms the Analysis is completed and DAL levels are assigned and compared successfully.

1 Computing Platform Design-1

Design Description: In Platform-1 each task comprises of one computer.

1.1 Fault Tree:

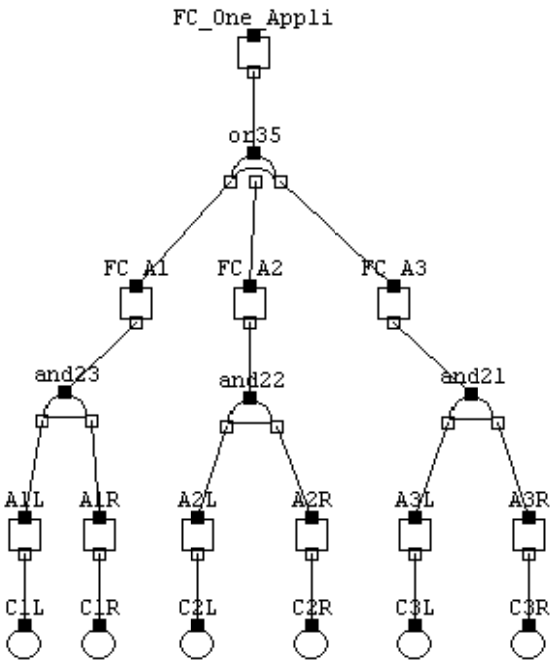

FIGURE 1: FAULT TREE – PLATFORM-1

1.2 Minimal Cut Set

The minimal cut set is defined as the minimal set of computers that will cause a failure on the desired application. The minimal cut sets of certain failure conditions are shown below:

1.2.1 FC_one_appli:

This is the minimal cut set of the failure condition FC_one_appli. This failure condition appears when at least one application is lost.

```
products(MSS('FC_One_Appli.O.true')) =
{'C1L.set1', 'C1R.set1'}
{'C2L.set1', 'C2R.set1'}
{'C3L.set1', 'C3R.set1'}
end
```

1.2.2 FC_Ai:

The minimal cut sets for each application FC_Ai (where i=1, 2, 3) is written down as follows:

```
products(MSS('FC_Ai.O.true')) =
{'CiL.set1', 'CiR.set1'}
end
```

1.3 Qualitative Requirement

It can be easily seen from the above results that all the failure conditions FC_Ai and FC_one_appli meet a qualitative requirement. This requirement implies that no single failure leads to a failure condition. Since each minimal cut set is composed of two elements we can be sure this condition is met.

1.4 Quantitative Requirement

1.4.1 FC_one_appli:

Probability [P] of the system = P ('C1L.set1') *P ('C1R.set1') +

P ('C2L.set1') *P ('C2R.set1') +

P ('C3L.set1') *P ('C3R.set1')

= 3 * (2 * 10^-3) ^ 2

= 12 * 10^-6

= 1.2 * 10^-5

As 1.2 * 10^-5 is greater than 10^-5 the system is **not** Quantitative

1.4.2 FC_Ai:

Probability [P] of the FC_Ai = P ('CiL.set1') *P ('CiR.set1')

= (2 * 10^-3) ^ 2

= 4 * 10^-6

Where i=1, 2, 3.

We find that the probability of losing each FC_Ai is less than the probability of losing the whole system.

We should redesign the system so as to avoid problems. Because the probability of losing the system should be less than 10^-5.

2 Computing Platform Design - 2

Design Description: In Platform-2 each task comprises of one computer and additionally there are backup computers for A1L and A3R

2.1 Fault Tree:

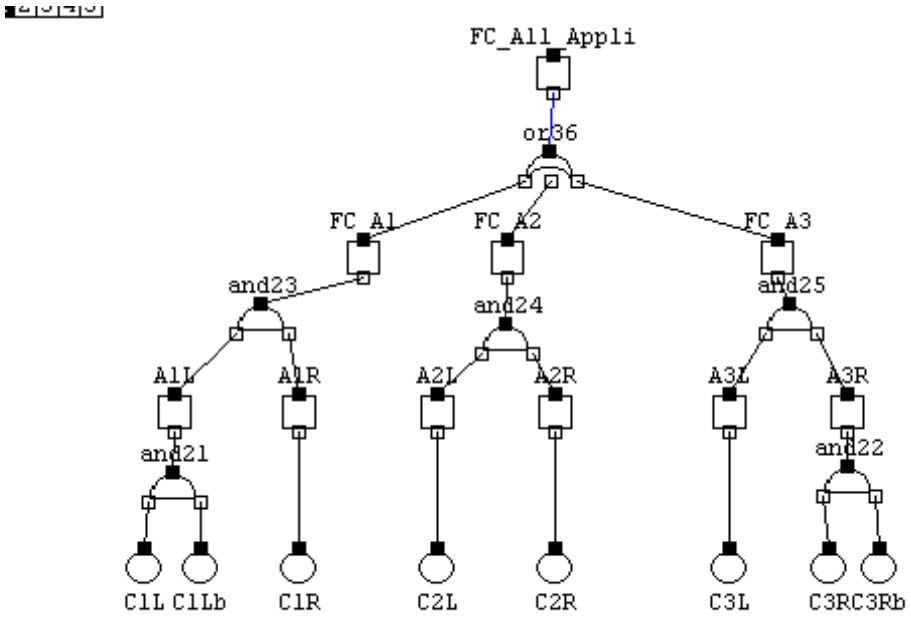

FIGURE 2: FAULT TREE – PLATFORM-2

2.2 Minimal Cut Set

The minimal cut set is defined as the minimal set of computers that will cause a failure on the desired application. The minimal cut sets of certain failure conditions are shown below:

2.2.1 FC_one_appli:

This is the minimal cut set of the failure condition FC_one_appli. This failure condition appears when at least one application is lost.

```
products(MSS('FC_All_Appli.O.true')) =
{'C2L.set1', 'C2R.set1'}
{'C1L.set1', 'C1Lb.set1', 'C1R.set1'}
{'C3L.set1', 'C3R.set1', 'C3Rb.set1'}
end
```

2.2.2 FC_Ai:

The minimal cut sets for each application FC_Ai (where i=1, 3) is written down as follows:

```
products(MSS('FC_Ai.O.true')) =
{'CiL.set1', 'CiLb.set1', 'CiR.set1'}
end
```

The minimal cut sets for the application FC_A2 is:

```
products(MSS('FC_A2.O.true')) =
{'C2L.set1', 'C2R.set1'}
end
```

2.3 Qualitative Requirement

It can be easily seen from the above results that all the failure conditions FC_Ai and FC_one_appli meet the qualitative requirement of no single failure leading to a failure condition.

2.4 Quantitative Requirement

2.4.1 FC_one_appli:

Probability [P] of the system = P ('C2L.set1') *P ('C2R.set1') +

P ('C1L.set1') *P ('C1Lb.set1')*P ('C1R.set1') +

P ('C3L.set1') *P ('C3R.set1')*P ('C3Rb.set1')

= ((2 * 10^-3) ^ 2) + (2 * (2 * 10^-3) ^ 3)

= 0.4 * 10^-5 + 16 * 10^-9

The above is less than 10^-5, hence the system is Quantitative

2.4.2 FC_Ai:

Where i=1, 3.

Probability [P] of the FC_Ai = P ('CiL.set1') *P ('CiLb.set1')*P ('CiR.set1')

= (2 * 10^-3) ^ 3

= 8 * 10^-9

The above is much less than 10^-5, hence the FC_A1 and FC_A3 is Quantitative.

Probability [P] of the FC_A2 = P (`'C2L.set1'`) *P (`'C2R.set1'`)

= (2 * 10^-3) ^ 2

= 0.4 * 10^-5

The above is less than 10^-5, hence the FC_A2 is Quantitative.We find that the probability of losing each FC_Ai and FC_one_appli is smaller than 10^-5 / flight hour. The system is reliable.

3 Computing Platform Design - 3

Design Description: In Platform-3 each task comprises of one computer and a spare computer per side (left and right)

3.1 Fault Tree:

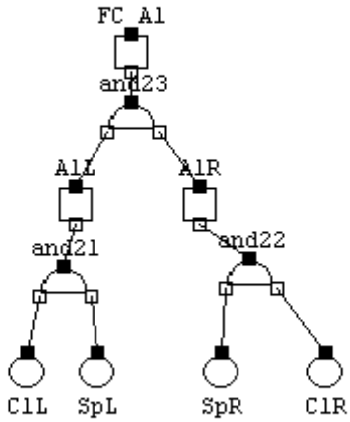

FIGURE 3: FAULT TREE – PLATFORM-3, FC_A1

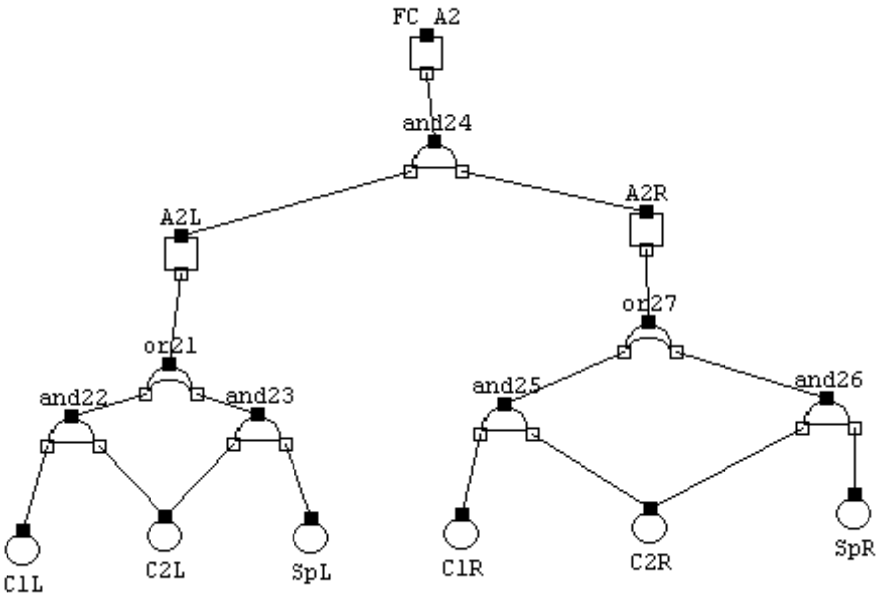

FIGURE 4: FAULT TREE – PLATFORM-3, FC-A2

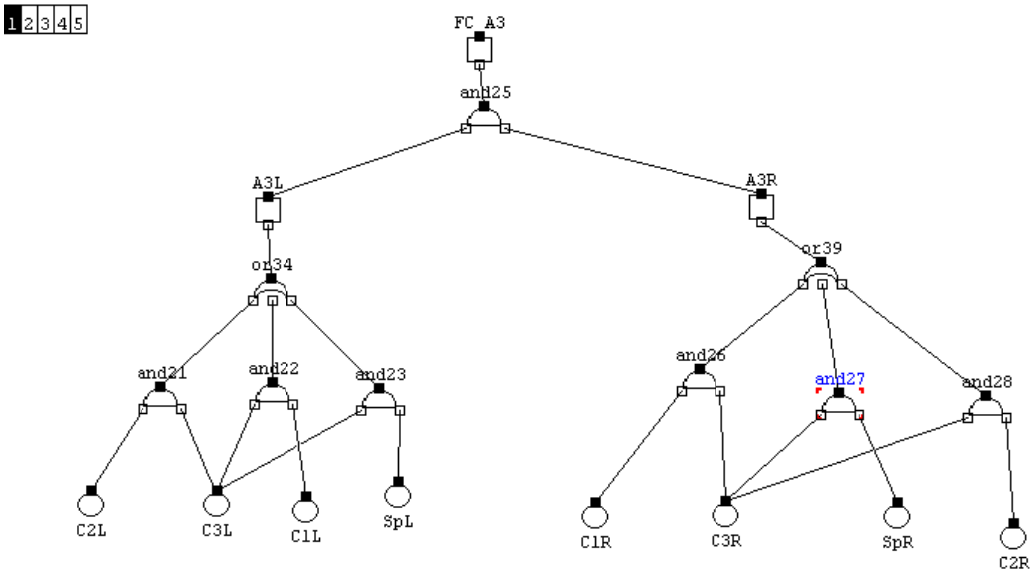

FIGURE 4: FAULT TREE – PLATFORM-3, FC-A3

3.2 Minimal Cut Set

3.2.1 FC_A1:

```
products(MSS('FC_A1.O.true')) =
{'C1L.set1', 'C1R.set1', 'SpL.set1', 'SpR.set1'}
end
```

3.2.2 FC_A2:

```
products(MSS('FC_A2.O.true')) =
{'C1L.set1', 'C1R.set1', 'C2L.set1', 'C2R.set1'}
{'C1L.set1', 'C2L.set1', 'C2R.set1', 'SpR.set1'}
{'C1R.set1', 'C2L.set1', 'C2R.set1', 'SpL.set1'}
{'C2L.set1', 'C2R.set1', 'SpL.set1', 'SpR.set1'}
end
```

3.2.3 FC_A3:

```
products(MSS('FC_A3.O.true')) =
{'C1L.set1', 'C1R.set1', 'C3L.set1', 'C3R.set1'}
{'C1L.set1', 'C2R.set1', 'C3L.set1', 'C3R.set1'}
{'C1L.set1', 'C3L.set1', 'C3R.set1', 'SpR.set1'}
{'C1R.set1', 'C2L.set1', 'C3L.set1', 'C3R.set1'}
```

```
{'C1R.set1', 'C3L.set1', 'C3R.set1', 'SpL.set1'}
{'C2L.set1', 'C2R.set1', 'C3L.set1', 'C3R.set1'}
{'C2L.set1', 'C3L.set1', 'C3R.set1', 'SpR.set1'}
{'C2R.set1', 'C3L.set1', 'C3R.set1', 'SpL.set1'}
{'C3L.set1', 'C3R.set1', 'SpL.set1', 'SpR.set1'}
End
```

3.3 Qualitative Requirement

As in the former sections the system does not fail if a single failure occurs. Hence the Qualitative requirement is met for each of the applications (i.e. FC_A1, FC_A2 and FC_A3) respectively.

3.4 Quantitative Requirement

3.4.1 FC_one_appli:

Probability [P] of the system = P('C1L.set1') *P('C1R.set1') *P('SpL.set1')*P('SpR.set1') +
P('C1L.set1') *P('C1R.set1') *P('C2L.set1')*P('C2R.set1') +
P('C1L.set1') *P('C2L.set1') *P('C2R.set1')*P('SpR.set1') +
P('C1R.set1') *P('C2L.set1') *P('C2R.set1')*P('SpL.set1') +
P('C2L.set1') *P('C2R.set1') *P('SpL.set1')*P('SpR.set1') +
P('C1L.set1') *P('C1R.set1') *P('C3L.set1')*P('C3R.set1') +
P('C1L.set1') *P('C2R.set1') *P('C3L.set1')*P('C3R.set1') +
P('C1L.set1') *P('C3L.set1') *P('C3R.set1')*P('SpR.set1') +
P('C1R.set1') *P('C2L.set1') *P('C3L.set1')*P('C3R.set1') +
P('C1R.set1') *P('C3L.set1') *P('C3R.set1')*P('SpL.set1') +
P('C2L.set1') *P('C2R.set1') *P('C3L.set1')*P('C3R.set1') +
P('C2L.set1') *P('C3L.set1') *P('C3R.set1')*P('SpR.set1') +
P('C2R.set1') *P('C3L.set1') *P('C3R.set1')*P('SpL.set1') +
P('C3L.set1') *P('C3R.set1') *P('SpL.set1')*P('SpR.set1')
= (14 * (2 * 10^-3) ^ 4)
= (14 * (16 * 10^-12))
= 224 * 10^-12

The above is much less than 10^-5, hence the system is Quantitative

3.4.2 FC_Ai:

Probability [P] of FC_A1 = P ('C1L.set1') *P ('C1R.set1') *P ('SpL.set1')*P ('SpR.set1')

= (2 * 10^-3) ^ 4

= 16*10^-12

Probability [P] of FC_A2 = P ('C1L.set1') *P ('C1R.set1') *P ('C2L.set1')*P ('C2R.set1') +
P ('C1L.set1') *P ('C2L.set1') *P ('C2R.set1')*P ('SpR.set1') +
P ('C1R.set1') *P ('C2L.set1') *P ('C2R.set1')*P ('SpL.set1') +
P ('C2L.set1') *P ('C2R.set1') *P ('SpL.set1')*P ('SpR.set1')

= (4 * (2 * 10^-3) ^ 4)

= 64 *10^-12

Probability [P] of FC_A3 = P ('C1L.set1') *P ('C1R.set1') *P ('C3L.set1')*P ('C3R.set1') +
P ('C1L.set1') *P ('C2R.set1') *P ('C3L.set1')*P ('C3R.set1') +
P ('C1L.set1') *P ('C3L.set1') *P ('C3R.set1')*P ('SpR.set1') +
P ('C1R.set1') *P ('C2L.set1') *P ('C3L.set1')*P ('C3R.set1') +
P ('C1R.set1') *P ('C3L.set1') *P ('C3R.set1')*P ('SpL.set1') +
P ('C2L.set1') *P ('C2R.set1') *P ('C3L.set1')*P ('C3R.set1') +
P ('C2L.set1') *P ('C3L.set1') *P ('C3R.set1')*P ('SpR.set1') +
P ('C2R.set1') *P ('C3L.set1') *P ('C3R.set1')*P ('SpL.set1') +
P ('C3L.set1') *P ('C3R.set1') *P ('SpL.set1')*P ('SpR.set1')

= (9 * (2 * 10^-3) ^ 4)

= 144 *10^-12

Each application does respect the probability threshold

4 DAL Allocation

4.1 Option 1

This option allows to assign a lower level of DAL inside a minimal cut set in which you have two sets that are independent. This lower level of DAL will be assigned to one of the sets. The other set will maintain its DAL. The choice of whether to assign the lower level of DAL to one or to the other set is arbitrary. The lower level of DAL is the one you obtain by decreasing the former value of 2.

In this case however the choice mentioned previously is not arbitrary. The reason for this is that the DAL of a computer dependent on another computer is always equal. Therefore, if we have a minimal cut set without

any independent computer then all those dependent computers will have the same DAL. We find that situation on FC_A3:

```
{'C1L.set1', 'C1R.set1', 'C3L.set1', 'C3R.set1'}
```

In which all computers are dependent. Since the DAL of each one is C, then all the Basic computers will have to have DAL C. Hence the DAL of the Spare Computers will be E. To summarize it all:

DAL(Basic Computers) = C

DAL(Spare Computers) = E

4.2 Option 2

This option simply assigns a lower value of the DAL to all components. Hence we can write the following:

DAL(Basic Computers) = DAL(Spare Computers)= D

5 Quantitative Assessment

For instance, let us remove the computer C1L. We will see for each design the consequences:

5.1 Platform Design – 1

This is the result for the minimal cut sets if we remove one computer. That is to say, if one computer is permanently out of order.

```
products(MSS('FC_One_Appli.O.true')) =
{'C1R.set1'}
{'C2L.set1', 'C2R.set1'}
{'C3L.set1', 'C3R.set1'}
End
```

Qualitatively speaking the system does not meet the requirement because clearly one single failure leads to a failure condition.

Let us calculate the probability of failure to see the if the probability requirement is not satisfied:

Probability [P] of the system = P ('C1R.set1') +

P ('C2L.set1') *P ('C2R.set1') +

P ('C3L.set1') *P ('C3R.set1')

= (2 * 10^-3) + 2 * (2 * 10^-3) ^ 2

= 2 * 10^-3 + 8 * 10^-6

The probability has increased a lot. It is not a safe system since the probability is greater than the threshold designed for MAJOR of critical systems.

5.2 Platform Design –2

In this case the minimal cut set is:

```
products(MSS('FC_All_Appli.O.true')) =
{'C2L.set1', 'C2R.set1'}
{'C1Lb.set1', 'C1R.set1'}
{'C3L.set1', 'C3R.set1', 'C3Rb.set1'}
end
```

It can be easily seen that this system is protected against one single failure. The probability is the following:

Probability [P] of the system = P ('C2L.set1') *P ('C2R.set1') +

P ('C1Lb.set1')*P ('C1R.set1') +

P ('C3L.set1') *P ('C3R.set1')*P ('C3Rb.set1')

= (2*(2 * 10^-3) ^ 2) + ((2 * 10^-3) ^ 3)

= 0.8* 10^-5 + 8* 10^-9

Therefore this does comply with the requirement and we can consider the system as safe.

5.3 Platform Design – 3

If we remove one computer C1L the minimal cut set is modified. For simplicity reasons the minimal cut sets are going to be shown again. The result of removing one computer does not affect the capability of the system to cope with one single failure condition.

Let us move on to the computation of the probability:

Probability [P] of the system = P ('C1R.set1') *P ('SpL.set1') *P ('SpR.set1') +

P ('C1R.set1') *P ('C2L.set1') *P ('C2R.set1') +

P ('C2L.set1') *P ('C2R.set1') *P ('SpR.set1') +

P ('C1R.set1') *P ('C2L.set1') *P ('C2R.set1')*P ('SpL.set1') +

P ('C2L.set1') *P ('C2R.set1') *P ('SpL.set1')*P ('SpR.set1') +

P ('C1R.set1') *P ('C3L.set1') *P ('C3R.set1') +

P ('C2R.set1') *P ('C3L.set1') *P ('C3R.set1') +

P ('C3L.set1') *P ('C3R.set1') *P ('SpR.set1') +

P ('C1R.set1') *P ('C2L.set1') *P ('C3L.set1')*P ('C3R.set1') +

P ('C1R.set1') *P ('C3L.set1') *P ('C3R.set1')*P ('SpL.set1') +

P ('C2L.set1') *P ('C2R.set1') *P ('C3L.set1')*P ('C3R.set1') +

P ('C2L.set1') *P ('C3L.set1') *P ('C3R.set1')*P ('SpR.set1') +

P ('C2R.set1') *P ('C3L.set1') *P ('C3R.set1')*P ('SpL.set1') +

P ('C3L.set1') *P ('C3R.set1') *P ('SpL.set1')*P ('SpR.set1')

= (6 * (2 * 10^-3) ^ 3) + (8* (2 * 10^-3) ^ 4)

=48 * 10^-9 + 128 * 10^-12

Happily enough the probability does respect the condition.

6 Comparison

Cost safety and availability are a function of the number of computers for each design, redundancy and the probability of loss with one computer down respectively. These parameters are shown in the following table:

	Platform Design-1	Platform Design-2	Platform Design-3
# of Basic Computer	6	6	12
# of Spare Computer	0	2	6
Availability=Probability with a faulty computer	2 * 10^-3	0.8* 10^-5	48 * 10^-9

The first solution has 6 computers in total so, is the cheapest as the cost is directly proportional to the number of components. Nevertheless it is also the less safe as it has no spare computers. In addition it has the highest probability of failure with one computer down so is not reliable from that point of view.

The second solution has 8 computers in total so, is the second cheapest. It is sufficiently safe as it has two spare computers. Moreover,it has an acceptable value of availability.

Similarly, the third solution has 18 computers. That will make it a highly expensive design. It is greatly safe as it has 6 spare computers. Finally, it is highly reliable in case of having one failure in one computer.

It is important to underline that the cost and safety are related to the DAL of each computer. The previous conclusions did not consider this fact. As we know there are two options of assignment. Let us assign the DAL in those two options:

	Platform Design-1	Platform Design-2	Platform Design-3
# of Basic Computer	6	6	12
# of Spare Computer	0	2	6
option 1	6C	6C+2E	12C+6E
option 2	6C	8D	18D

With reference to this table we can identify the two possibilities. If **option 1** is taken then the previous comments related to cost and safety at the beginning of the section are true.

Nonetheless, with **option 2** this is not the case. In that situation we find that the most expensive solution could be design number 1. While the design 2 would be the cheapest. But the safety might be compromised as all the components are level D and this jeopardizes the quality of the system.

Design 2 will be preferred in any case as it is neither expensive nor unsafe. This assumption is valid for both options.

Imagining the best solution we can make a change in our model so as to reduce the probability of failure with one computer down. Let us compute the probability of our new model. It is important to be aware of the fact that in our model we have added one spare computer to the FC_A2 of the design number 2.

Probability [P] of the system = P (`'C2L.set1'`) *P (`'C2R.set1'`) * P (`'C2Rb.set1'`) +

P (`'C1L.set1'`) *P (`'C1Lb.set1'`)*P (`'C1R.set1'`) +

P (`'C3L.set1'`) *P (`'C3R.set1'`)*P (`'C3Rb.set1'`)

= ((3 * (2 * 10^-3) ^ 3)

= 24 * 10^-9

As we can see the probability is astonishingly low. We have reached a very good probability that is very close to the one coming from design three. Only that in our model we use a total of number of computers of 9 instead of the 18 used by the model of design platform number 3. The model is therefore very cheap and with a very high value of availability. It also has more reliability than the previous model as we add one more spare computer.

7 Conclusion

This project is a good opportunity to apply the advices and the lessons taught. The session gave us a unique exposure to realize a static model and provided us with an insight on safety assessments of static systems.

Nevertheless, in the end we were able to model a platform that is reliable, cost effective and has marginal availability.