# Discussion about the model

SWEN90004: Modelling Complex System

Student: Bowen Bai (969899)

In the original model of the space system, different from implementing the simulator in Java, the FSP model uses share action to model the interactions between different processes. An input value with the process is used as a counter to record the number of available tugs or space in wait zones. In Java, we use the synchronized properties and monitors to prevent the system from mutual exclusion.

When I was observing the system to check if there are some mutual exclusions or deadlocks, the way used was to observe the output and to check if there were any errors. However, this way wasn't reliable enough and could only help me find potential problems by chance. The main problem of the original model is a deadlock will appear if there is a ship just finishes unloading and trying to acquire two tugs for undocking meanwhile there is another ship trying to go into the berth and the number of available tugs is less than two. Both those two ships are waiting for each other to release the resources which are the tugs here. What's more, none of them could force the other one to give up the resource. As a result, the decklock came out.

In the previous implementation in Java, there were no safety properties and liveless properties which are provided in FSP. If the random seed we used in the previous Java model didn't create that 'coincidence', it's also impossible for us to detect that problem. However, in the FSP model, safety properties and liveless properties will always reveal that problem if we change the parameters.

To remove this potential deadlock which will occur when the number of tugs changes, the best way is to remove those four essential conditions for deadlocks. For the space ship model, obviously, it's impossible to make the tugs become unreusable. The resources will be held for the whole time when the process is waiting for others to release resources. Thus, we should not let a new ship get tugs if there is a potential deadlock. Which means when a ship is trying to acquire tugs for docking, the rest tugs should be enough for the ship in the berth to undock. To be specific, I changed the guard of TUGS process, the condition now is that only when the value input greater than the sum of tugs required for docking and undocking can the ship acquire tugs for docking. This method successfully removed the deadlock even the parameters of pilots' number or tugs' number.