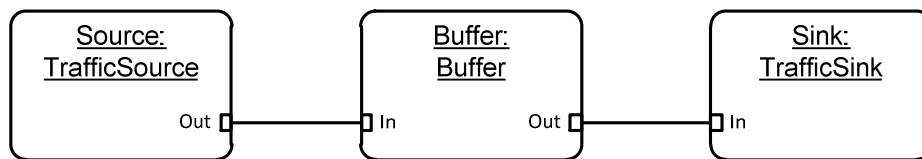


Performance Modelling and Analysis Workshop

Adequately Modeling a Store-and-Forward Buffer System

In this exercise, we will model and analyze the performance of a simple store-and-forward buffer system as illustrated below. The Source represents a (simplified) model of an Internet / Ethernet data source for TCP / IP packets modelled with data class Packet.



We apply the performance modelling patterns presented during the workshop to analyze:

- Maximum buffer occupation
- Time-weighted average buffer occupation
- End-to-end latency
- Throughput

We also investigate the effect of changing the performance estimation parameters.

Exercise 1 Simulate the given model to see that the Occupation variable of BufferContainer is updated at the moment of receiving the head of a Packet in the input handler, while it is only added to the Queue data structure at the moment that it has been received completely. This moment unblocks the guard in the output handler to send the Packet again as presented in the slides.

Exercise 2 Add a variable Maximum to the BufferContainer data class to evaluate the maximum occupation that occurs during a simulation run. Also, add a variable Average to enable evaluating the long-run time average buffer occupation. Initialize these variables in method `init`. Use a desired accuracy level of 0.95 and a desired confidence level of 0.95 as parameters for the Average monitor. Update methods `allocate` and `remove` to use method `rewardRC` for registering new occupation values as rewards to the Average monitor with the buffer being empty as recurrence condition. Recall that `currentTime` can only be used in process methods. Hence, you will need to add a parameter to methods `allocate` and

remove in order to forward the current model time to the Average monitor. Debug and inspect your model to find out what the performance results are.

Exercise 3 Add variables Latency and Throughput to the TrafficSink process class to evaluate the end-to-end latency (sample average) and packet throughput (rate average). Initialize them in a new method that is called initially. Use a desired accuracy level of 0.95 and a desired confidence level of 0.95 as parameters for the Latency and Throughput monitors and use method `rewardBM` to register rewards for newly received packets in method `ReceivePacket`. The end-to-end latency of individual packets can be determined by using method `getEntranceTime` of data class `Packet`. This method returns the time at which the Packet entered the Buffer. Debug and inspect your model to find out what the results are.

Exercise 4 The buffer occupation results are in packets instead of bytes. Since the packets send by the Source vary in size, the buffer occupancy results obtained so far are not realistic: the model is not adequate. We now improve the adequacy of the model to evaluate the maximum and average occupation in bytes. This requires changes to methods `allocate` and `remove` of data class `BufferContainer` as well as to the delay statements in the input and output handlers of the process class `Buffer`. Method `getSize` of data class `Packet` returns an Integer denoting its size in bytes. Debug and inspect your model to find out what the performance results are. Also investigate how much model time must be simulated until the results are accurate.

Exercise 5 Decrease the batch size of the latency and throughput metrics to 10 by using method `setBatchSize` on their monitors. What effect do you expect on the simulation time needed to obtain accurate results for all 3 average metrics? What happens to the analysis results for all metrics including the maximum occupation?

Exercise 6 Increase the desired accuracy and confidence levels for the average buffer occupancy, latency and throughput metrics to 0.99. Investigate the effect on the simulation time needed to obtain accurate results for all 3 averages. What happens to the analysis results for all metrics including the maximum occupation?

Exercise 7 In case you have time left, you may investigate what happens if the model is not conservative. Change the output handler of process class `Buffer` to remove the packet from the buffer at the moment of sending its head to the Sink. What happens to the results for all metrics including the maximum occupation?