**Slide 1 -**

Our project consisted in the implementation of a deficit scheduler. /

**Slide 2**

There is a source that sends job to a receiver. The receiver serves jobs within one turn and atomically, so if he can’t serve them entirely before the end of the turn, he goes on vacation and uses the remaining time to enlarge the next turn duration.

We implemented such model in Omnet++ using two simple modules EMITTER and SERVER. The server starts on vacation and at the end of every vacation check whether there are job in the queue or not, when a job is received it’s always sent into the queue. The end of a vacation and of a service time are modelled using self messages after a specific period of time, that can be constant or taken from an exponential distribution.

For what concerns the messages, we decided to extend the normal message in Omnet++ by adding a field that contains the requested service time for that specific job, which is a value that is obtained by the EMITTER and it’s used by the RECEIVER.

**Slide 3 – Queueing Theory**

To validate our model, we made some simplifications so that we could study it using queuing theory. In particular we observed that, if the vacation time is null, then the system behaves like an M/M/1. This make sense since if we don’t have vacations, the time required for the system to build a turn big enough to serve a job is equal to 0, meaning that the turn time doesn’t affect the system anymore. Using this result, we slightly modified our OMNeT++ code to test it under this assumptions, and we obtained the expected value for the mean response time.

**Slide 4 – Warmup period**

In order to perform the following steps for studying our system we determined a warmup period.

This slide shows the sliding window average computed on the response time, as we can see the system becomes stable after around 3 seconds, therefore, after some trials for different scenarios, we decided to set the warmup period to 3seconds.

-------------------------------------------------------------------------------------------------------------------------------------

**Slide 5 - Constant Case**

Firstly, we took into consideration the case in which inter-arrival times, service times and the duration of vacations had a constant value. We found out that the stability condition was the one shown in the slide.

The formula can be derived from the fact that in order to build a turn large enough to serve a job with a service time we need vacations. We can conclude that each job will ask for a period equal to in order to be served. If the interarrival time is smaller than this value, jobs will queue up.

If the stability condition is verified, we can distinguish between two cases. If the interarrival-time isn’t a multiple of the vacation then the response time will swing around a certain value, as shown in the plot on the left. This is due to the fact that the server starts on vacation so some jobs will queue up.

If instead the inter-arrival time is a multiple of the vacation the response time will be constant, as we can see in the plot on the right.

**Slide 6 – Constant Case plot**

To check that our thesis was correct we tested the system with different scenarios, one such that our stability condition was met, one where it wasn’t. As we can see, the blue graphs show that even if the inter-arrival time is slightly smaller than the one required for the condition the system will immediately start queuing up jobs.

**Slide 7 – Exponential Case – 2kr**

To carry out the 2kr factorial analysis we performed 5 repetitions for 5 different values of Q (0.1 St, 0.5 St, 1 St, 5 St, 10 St). We discovered that the factors that impact the most the response time are the ones circled in red. For each of the circled factors we performed the data analysis.

As we can see, the impact of I\_t increases as Q increases and the same goes for S\_t and the interplay of I\_t ad S\_t. Instead as Q increases, the impact of V on the response time decreases.

-------------------------------------------------------------------------------------------------------------------------------------

**Slide 8 – Exponential Case – plots**

Performing the data analysis, we discovered that almost every case was well fitting an exponential.

* For every case we collected data from 15 replicas of each scenario in which we varied the most-impactful factor and kept fixed the others and we took the average response time.
* For each scenario we computed then the sample mean, variance and Cov
* Then we tested the assumption of normality with a QQ plot
* and we computed the CI for each sample mean.
* We tried to fit the sample mean of each scenario to a regression model that appeared to be exponential in many cases, so we applied a transformation to make it linear.

**Slide 9 – Exponential Case – fitting**

Here we reported for each case the model that we obtained and its linearization. We also reported the value of . As we can see from the table, in every case it is close to 1. Since we verified the hypothesis we can say that in each case an high percentage of the variability can be explained by the model.

**Slide 10 – Conclusions**

In conclusion we can say that the system’s dependencies change with the ratio between the mean service time and the turn time, if the turn time becomes bigger the system depends more and more on It and St, this make sense because it will behave again like an M/M/1, this because if the turn tends to infinite, except for the case in which the queue is empty and the system goes on vacation, it’s exactly like an M/M/1 and will almost never go on vacation. When the ratio is close to 1 the mean response time will mostly depend on the service time and vacation period because for each job on average the required time to fully complete a request is the sum of those two quantities. In the end, if the turn is smaller than the service time, since the server will have to do many vacations before having a turn big enough to serve that job, the vacation becomes the most significant factor.