Final Project Group

CAR DOORS Production Process

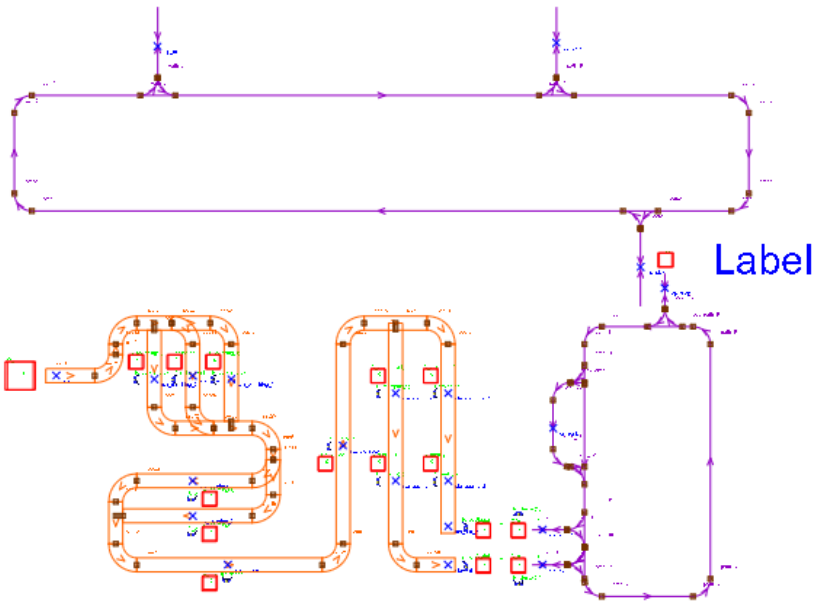
Group Members		
Dian Alberto	*Autonomous System - EIT Digital*	*247376*
Nascivera Davide	*Autonomous System - EIT Digital*	*248232*
Li Yunju	*Autonomous System - EIT Digital*	*247402*

1. Presentation

This report showcases a simulated representation of a car door manufacturer's production system. The company specializes in manufacturing two primary products: left doors and right doors. Both products flow within the system, where the initial tasks performed are the same for both until reaching the final section, where different processes are applied. Firstly, the raw materials go through the polishing process to remove roughness from the surface. Once the product is polished, it is ready to be painted and subsequently dried in the respective stations. To ensure a higher quality and resistance of the product, the products flow in the clearcoat station where an additional level of protective paint is applied. Once the product has reached this point in the conveyor system, it goes to one of the two stamping stations according to its type, since the operation performed differs from left to right door. Finally, the quality of the finished product is checked. Once the final product is obtained it is stored in pallets and sent to the shipping area using a forklift, where it will await the arrival of the truck to pick them up.

1.1 The layout of the simulation

The image below shows the layout of the factory.



1.1.1 Conveyor system

The products flow through the line using a conveyor system, where each workstation is placed sequentially. For each station one and only one human operator is assigned to perform the tasks assigned, and each station is preceded by a queue to store the loads during the whole production process.

At the beginning, the load is generated in the queue Q_in and then immediately placed in the station sta_in. As stated in the previous section, the raw material moves firstly to the polishing station. After this first process it goes to the painting station, followed by the drying station and then the clearcoat one. Then the load goes to one of the two stamping machines, based on the type of the product. Once the stamping process has finished, the quality of the product is checked and it will be put into the Q_out queue, waiting to be clustered into pallets by the human operator.

1.1.2 Pathmover (Forklift)

Once the pallets are prepared in the buffers denoted as queue Q_cluster, the material handler needs to move the products to the final queues Q_ship, ready to be loaded on arrival of the truck. Specifically, whenever the forklift detects that a new SKU has been completed, it picks the pallet from cp_pick1 and cp_pick2 and subsequently moves it to cp_drop. The difference between the two control points cp_pick1 and cp_pick2 is related to the type of product retrieved, where cp_pick1 refers to left doors while cp_pick2 refers to right doors. During idle times, the forklift remains parked at the control point cp_park.

1.1.3 Pathmover (Truck)

The third and final system proposed is related to the truck, used to retrieve and deliver the final products. The truck is designed to arrive at the warehouse two times during the 8-hour shift: the first time after 4 hours while the second at the end of the shift. In the designed system, the products are retrieved at the node cp_pick, placed in the warehouse right next to the Q_ship queue where the finished products are stored. Once the truck retrieves the loads, it will travel to the control point cp_drop, where the products are dropped, simulating the delivery phase. An additional node is called cp_park, where the truck waits before and after the delivery/retrieval phase.

1.2 Input file

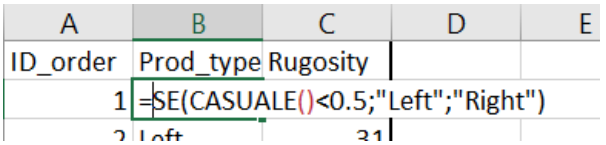
The Automod simulation runs by reading the order list in a given CSV file, containing all the information needed for the production of all the listed products.

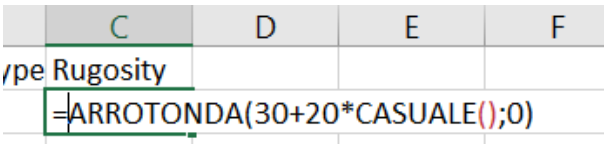
The data is generated using an external Excel file: Each order line is generated and characterized by “ID_order”, “Prod_type”, and “Rugosity”.

- “ID_order” is an attribute responsible for load identification. The loads are ordered in an ascending order.
- “Prod_type” is a string variable that states the product type. It can be “left” or “right” and it’s obtained by using the functions “IF” and “RAND” (as mentioned below). In particular it’s possible to select the probability of getting one product over the other.

- "Rugosity" is an attribute that represents the roughness of the initial product. This measure affects the "polishing processes" and the value is a random variable following a random distribution. In particular, in our case we have a mean of 40 and a standard deviation of 10.

	A	B	C
1	ID_order	Prod_type	Rugosity
2	1	Right	48
3	2	Left	31
4	3	Right	49
5	4	Left	32
6	5	Left	32
7	6	Left	44
8	7	Left	38
9	8	Left	33
10	9	Right	46
11	10	Left	44
12	11	Right	34
13	12	Right	33
14	13	Right	48
15	14	Right	40
16	15	Left	41
17	16	Right	41
18	17	Right	46
19	18	Right	43
20	19	Left	35
21	20	Right	47





1.3 Optimization goals

The main goals of our project are as follows:

(1) Minimize the delay of the truck.

(2) Maximize the utilization ratio of the resources.

(3) Minimize the waiting time of the loads.

Delay of trucks represents the time that the truck waits for the warehouse to be filled, or the time that the products in the warehouse wait to be picked. Both of these result in inefficiency. When it's positive, it means the truck comes but the warehouse is not full yet. When it's negative, it means that the warehouse is full but the truck hasn't come. Maximizing the utilization ratio of the resources means minimizing the idle time of the human operators. We defined different layouts to look for the best configuration. Waiting time of the loads means that the load has arrived at the station but the human operator in this station is still working on the previous product. In this case, the line is blocked, making the production process less efficient.

1.3 Different scenarios

To properly optimize the performance of the production process with respect to the goals we have defined, we designed different configurations starting from a simple one, gradually increasing the complexity to match the production system needs. The main differences between the layouts are the number of stations used for polishing and painting.

- One station for polishing and one station for painting
- Two stations for polishing and one for painting
- Two stations for polishing and two for painting
- Three stations for polishing and two for painting

2. Simulation Entities

2.1. Load

- *L_dummy(Generation Limit: 1, Distribution:Constant, Mean: 0): L_dummy is used to read the input data, including load type and roughness of the surface of the load. For each read row of the document the process associated with L_dummy creates a clone of the initial product and sends it to the next operation. This procedure is done with a cadence that is a normal distribution(150,5). At the end it's sent to die.*
- *L_prod: L_prod is used to distinguish two different types of products. After stamping, there are two types of products. Based on the type they will be sent to different quality-checking stations.*
- *L_prod_init: L_prod_init represents the raw load which has not been processed yet. We know its load type which is represented in the load_attribute V_type_ID.*
- *L_product_big: Before the products are shipped by the forklift to the warehouse, the loads are clustered into pallets. L_product_big is used to represent the pallets.*
- *L_truck(Generation Limit: 2, Distribution:Constant, Mean: 4 hours): L_truck is used to schedule the arrival time of the truck. Every 4 hours this load is generated and the truck will come to the warehouse to take the products and deliver them.*
- *L_stop_production(Generation limit: 2, Distribution: Constant, Mean: 8 hours): L_stop_production is used to stop the production process. Since the target of the simulation is to simulate the production of one day with one shift of 8 hours, this load is generated after 8 hours.*

2.2. Global Variables

- *V_delay (Type: Integer; Dim:1): V_delay is used to measure the delay time of the trucks. It can be positive or negative. When it's positive, it means the truck comes but the warehouse is not full yet. When it's negative, it means that the warehouse is full but the truck hasn't come.*
- *V_pointer (Type: FilePtr; Dim:1):* V_pointer is used to read the input file. It's a variable pointing to the file that is being read.
- *V_schedule_truck (Type: Integer; Dim:1):* V_schedule_truck is used to record the time that the truck arrives at the warehouse and the time that the truck leaves.

- *V_stop_production (Type: Integer; Dim:1):* V_stop_production is used to stop the production. Once the simulation runs for 8 hours, a load which will set this variable to 1 is generated. Then we will stop reading the file and stop introducing new products in the conveyor system.
- *V_truck_isArrived (Type: Integer; Dim:1): V_truck_isArrived* is used to control the state of the truck. It can be 0, 1 or 2 . If it's 0, it means that the truck hasn't come yet. If it's 1, it means that the truck is currently in the warehouse ready to pick the products. If it is 2, it means that the production has finished before the arrival of the truck. This variable enables the possibility to calculate the delay in both scenarios: when the production finished earlier than the arrival time of the truck and when the truck has to wait for the production to be finished.
- *V_useless (Type: String; Dim:4):* V_useless in a 4-dimensional variable. It is used to eliminate the first line of the .csv when we start reading the input data. In particular it's the line with the names of different columns listed.

2.3. Load Attribute

- V_id *(Type: Integer; Dim:1):* V_id is an attribute responsible for load identification. It's given in the input file.
- V_type *(Type: String; Dim:1):* V_type represents *the type of the load. It can be left or right.*
- V_rugos *(Type: String; Dim:1):* V_rugos is used to measure *the roughness of the surface of the load. It's given in the input file. The roughness will influence the duration of the polishing process.*
- V_start_time *(Type: Integer; Dim:1): V_start_time represents the time that the load enters the production process [seconds].*
- V_type_ID *(Type: Integer; Dim:1): V_type_ID is used to represent the type of load. The difference with V_type is that V_type_ID is that V_type_ID is integer while another one is string. We transfer the way to represent types for later use.*
- V_Random *(Type: Integer; Dim:1): V_random is used to simulate quality check. It's a variable computed in code with a 90% probability of being 0 and 10% of being 10. If V_Random < 1, the product is qualified, otherwise it's not then it wouldn't be sent to the warehouse.*

2.4. Order Lists

- *OL_prod(Dim:2): OL_prod is used to realize the process of clustering the products into pallets. The final products wait to be ordered in this order list until there are enough products in queue Q_finished_prod to be clustered.*
- *OL_prod_ship(Dim:1):OL_prod_ship is used for the delivery of the products. The products will wait to be ordered in the order list until the warehouse is full and the truck is in the warehouse.*

2.5. Resource

- *R_clearcoat (Dim:1):* Human operator, who performs clearcoat in the station sta_clearcoat. It's used in the process P_clearcoat. The task duration is 2 min.
- *R_cluster (Dim:2):* Human operator who performs clustering between Q_finished_prod and Q_clusters. They collect products from Q_finished_prod and cluster them, then store the obtained pallets into Q_clusters. The task duration is 30 sec.

- *R_drying (Dim:1):* Human operator, who performs drying in the station sta_drying. It's used in the process P_drying. The task duration is 2 min.
- *R_out (Dim:2):* Human operator working between the station sta_out and queue Q_finished_prod. The task of the operator is to pick the final product in the conveyor and store it in the queue Q_finished_prod. The task duration is 30 sec.
- *R_painting (Dim:2):* Human operator, who performs clearcoat in the station sta_painting. It's used in the process P_painting. The task duration is 4 min.
- *R_polishing (Dim:3):* Human operator, who performs polishing in the station sta_polish. It's used in the process P_polish. The task duration is distinguished by a normal distribution with mean of 5.5*(1+roughness/100) min and variance to be 0.5 min.
- *R_quality (Dim:2):* Human operator, who performs quality-checking in the station sta_quality. It's used in the process P_quality. The task duration is distinguished by a normal distribution with mean 2 min and variance to be 0.4 min.
- *R_stamping (Dim:2):* Human operator, who performs stamping in the station sta_stamping. It's used in the process P_stamping. The task duration is 3 min.

2.6. Queue

- *Q_clearcoat (Dim:1 ; Capacity: 10):* Queue where the products are waiting for the clearcoat. It is used to measure the amount of time the load has to wait to be processed.
- *Q_clusters(Dim:2 ; Capacity: 1):* Queues to store the clustered products. Pallets inside will be picked by forklift and sent to the warehouse.
- *Q_drying(Dim:1 ; Capacity: 10):* Queue where the products are waiting to be dried. It is used to measure the amount of time the load has to wait to be dried.
- *Q_finished_prod(Dim:2; Capacity: 10): Queue used to store the final products.* After quality-checking, qualified products will be stored in this queue waiting to be clustered.
- *Q_in(Dim:1; Capacity: Infinite):* All generated loads are placed here at the beginning of the simulation.
- *Q_painting(Dim:2 ; Capacity: 10):* Queue where the products are waiting to be painted. It is used to measure the amount of time the load has to wait to be painted.
- *Q_polishing(Dim:3 ; Capacity: 10):* Queue where the products are waiting to be polished. It is used to measure the amount of time the load has to wait to be polished.
- *Q_quality(Dim:2 ; Capacity: 10):* Queue where the products are waiting for quality check. It is used to measure the amount of time the load has to wait to be checked.

- *Q_ship(Dim:1 ; Capacity: 15):* Queue that represents the warehouse. The forklift stores the clustered products in this queue.
- *Q_stamping(Dim:2 ; Capacity: 10):* Queue where the products are waiting to be stamped. It is used to measure the amount of time the load has to wait to be stamped.

2.7 Labels

- L_ship(Dim:1 ; Number of labels: 1): Used to print the capacity of Q_ship.
- L_delay(Dim:1 ; Number of labels: 1): Used to print the delay between the production and the truck.

2.7 Processes

The project was developed with the goal of keeping the simulation as flexible as possible, minimizing potential problems in future implementations. For this reason the processes are implemented as array entities, making use of the "procindex" and the "nexof()" functions.

At the beginning all the loads are initialized by the "P_read" and they flow in the assembly line with the usage of the functions mentioned above. The loads are treated the same until the process P_clearcoat then they are processed according to their load attribute "Prod_type" in the CSV.

P_read (Dim:1; Default Traffic Limit: Infinite)

```
begin P_read arriving
    open "dir/input.csv" for reading save result as V_pointer //Automod  v14  or greater
    read V_useless(1),V_useless(2),V_useless(3) from V_pointer with delimiter ";"
    while V_stop_production<1  do
        begin
        read V_id, V_type, V_rugos from V_pointer with delimiter ";" at end
            begin
                print " My input file is fully read" to message
                send to die
            end
        wait for normal 150,5 sec
        set V_start_time to ac
        clone 1 to P_init
```

```
            end
end
```

P_init (Dim:1; Default Traffic Limit: Infinite)

```
begin P_init arriving
        set this load type = L_prod_init
        if V_type= "Left" then set V_type_ID to 1
        if V_type= "Right" then set V_type_ID to 2
        move into Q_in
        move into conv.sta_in
        set V_Random to oneof(10:10, 90:0)
        send to nextof(P_polish(1), P_polish(2), P_polish(3))
end
```

P_polish (Dim:1; Default Traffic Limit: Infinite)

```
begin P_polish arriving
        travel to conv.sta_polishing_(procindex)
        move into Q_polishing(procindex)
        use R_polishing(procindex) for normal 5.5 *(1+(V_rugos/100)),0.5 min
        move into conv.sta_polishing_(procindex)
        send to nextof(P_painting(1), P_painting(2))
end
```

P_painting (Dim:1; Default Traffic Limit: Infinite)

```
begin P_painting arriving
        travel to conv.sta_painting_(procindex)
        move into Q_painting(procindex)
        use R_painting(procindex) for 4 min
        move into conv.sta_painting_(procindex)
        send to P_drying
end
```

P_drying (Dim:1; Default Traffic Limit: Infinite)

begin P_drying arriving

travel to conv.sta_drying

move into Q_drying

use R_drying for 2 min

move into conv.sta_drying

send to P_clearcoat

end

P_clearcoat (Dim:1; Default Traffic Limit: Infinite)

begin P_clearcoat arriving

travel to conv.sta_clearcoat

move into Q_clearcoat

use R_clearcoat for normal 2,0.3 min

move into conv.sta_clearcoat

travel to conv.sta_stamping_(V_type_ID)

send to P_stamping(V_type_ID)

end

P_stamping(Dim:1; Default Traffic Limit: Infinite)

begin P_stamping arriving

move into Q_stamping(procindex)

use R_stamping(procindex) for 3 min

move into conv.sta_stamping_(procindex)

set this load type = L_prod(procindex)

travel to conv.sta_quality_(procindex)

send to P_quality(procindex)

end

P_quality(Dim:1; Default Traffic Limit: Infinite)

begin P_quality arriving

move into Q_quality(procindex)

use R_quality(procindex) for normal 2,0.4 min

move into conv.sta_quality_(procindex)

if V_Random < 1

```
            begin
                travel to conv.sta_out_(procindex)
                send to P_out(procindex)
            end
        else
            begin
                send to die
            end
end
```

P_out(Dim:1; Default Traffic Limit: Infinite)

```
begin P_out arriving
    use R_out(procindex) for normal 30,5 sec
    send to P_cluster(procindex)
End
```

P_cluster(Dim:1; Default Traffic Limit: Infinite)

```
begin P_cluster arriving
    move into Q_finished_prod(procindex)
    if Q_finished_prod(procindex) current loads < 5 then
        wait to be ordered on OL_prod(procindex)
    else
        begin
            order 4 loads from OL_prod(procindex) to die
            use R_cluster(procindex) for 30 sec
            set load type to L_product_big(procindex)
            move into Q_clusters(procindex)
            send to P_lift(procindex)
        end
end
```

P_lift(Dim:1; Default Traffic Limit: Infinite)

```
begin P_lift arriving
    if Q_ship current loads < 15 then
```

```
                begin
                        move into forklift.cp_pick_(procindex)
                        travel to forklift.cp_drop
                        send to P_ship
                end
        else
                begin
                        print "Shipping congestion" to message
                        wait until Q_ship current loads is 0
                        send to P_lift(procindex)
                end
end
```

P_ship(Dim:1; Default Traffic Limit: Infinite)

```
begin P_ship arriving
        move into Q_ship
        print "Current load: " Q_ship current load to L_ship
        if Q_ship current loads < 15 then
                wait to be ordered on OL_prod_ship
        else
                begin
                        if V_truck_isArrived = 1 then
                                begin
                                        order 14 loads from OL_prod_ship to die
                                        move into trucks.cp_pick
                                        travel to trucks.cp_drop
                                        set V_truck_isArrived to 0
                                        print "Delay time:   " (ac-V_schedule_truck)/60 as .2 "min" to message
                                        print "Delay time:   " (ac-V_schedule_truck)/60 as .2 "min" to L_delay
                                        set V_delay to (ac-V_schedule_truck)/60
                                end
                        else
                                begin
                                        set V_truck_isArrived to 2
```

```
                                        print "The production finished earlier" to message
                                        set V_schedule_truck to ac
                                        wait to be ordered on OL_prod_ship
                                    end
                end
        send to die
End
```

P_truck_schedule(Dim:1; Default Traffic Limit: Infinite)

```
begin P_truck_schedule arriving
        print "Truck sent" to message
        move into trucks.cp_pick
        if V_truck_isArrived = 0 then
                begin
                        set V_truck_isArrived to 1
                        print "Truck has arrived" to message
                        set V_schedule_truck to ac
                end
        else
                begin
                        order 15 loads from OL_prod_ship to die
                        print "Anticipation time:   " (ac-V_schedule_truck)/60 as .2 "min" to message
                        print "Anticipation time:   " (ac-V_schedule_truck)/60 as .2 "min" to L_delay
                        set V_delay to (V_schedule_truck-ac)/60
                        move into trucks.cp_pick
                        travel to trucks.cp_drop
                        set V_truck_isArrived to 0
                end
end
```

P_stop_production(Dim:1; Default Traffic Limit: Infinite)

```
begin P_stop_production arriving
        increment V_stop_production by 1
```

end

2.8 Vehicles

- *Forklift(Number of vehicles: 1, Vehicle Start List:):*
- *Truck(Number of vehicles: 1, Vehicle Start List:):*

2.9 Lists

- *Named List which contains x list(s):*
  - ***park_list****(List items: cp_park)*
  - ***pick_list****(List items: cp_pick_1, cp_pick_2)*
- *Work List which contains:*
  - ***cp_drop****(List items: pick_list)*
  - ***cp_park****(List items: pick_list)*
- *Park List which contains:*
  - ***cp_drop(****List items: park_list)*

3. Simulation Results

The simulations aim to analyze and evaluate the following key performance indicators:

1. **Resources' statistics**

   The goal is to have an understanding of the utilization ratio of the human operators to evaluate the effectiveness of the conveyor system and the whole production process.

   We presented a TimeLine graph for each resource to visualize the utilization ratio during the shift. On The X-axis there is the time flow from the beginning of the simulation up to 8 hours, while on the Y-axis there is the percentage of resource utilization. To compare the performances of resources we used a BarChart graph.

2. **Loads' statistics**

   Evaluation of the time spent by the load in the system, mainly focusing on the average time spent in the queues between stations. The results are presented using BarChart graphs to compare the different queues.

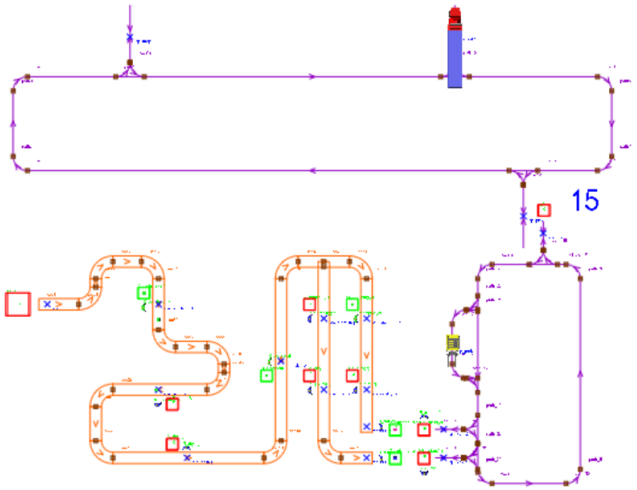
3. **Truck's statistics**

   We want to assess the delay between the production and the arrival of the truck, in order to optimize the assembly line to reduce retards.

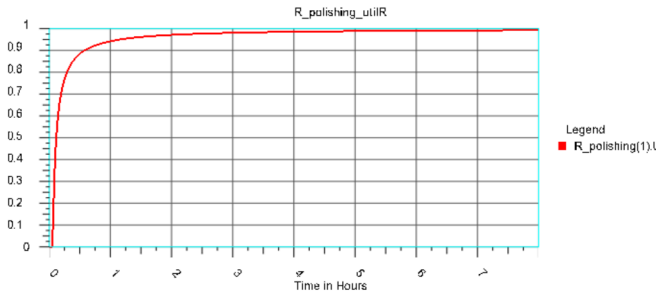
Simulation 1:

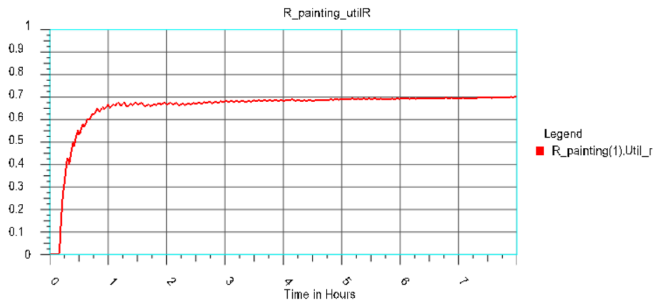
Settings:

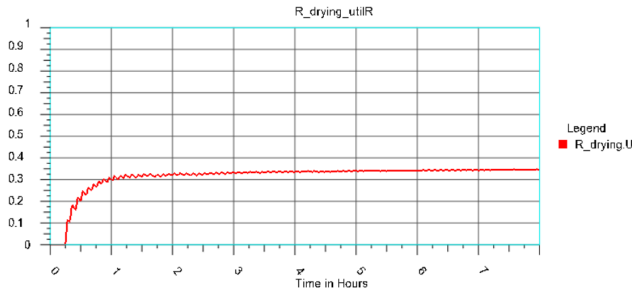
- One workstation for the polishing process
- One workstation for the painting process

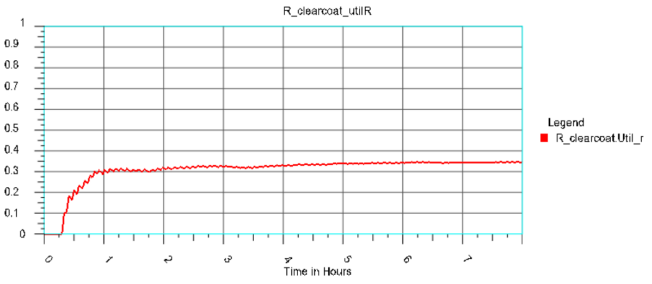


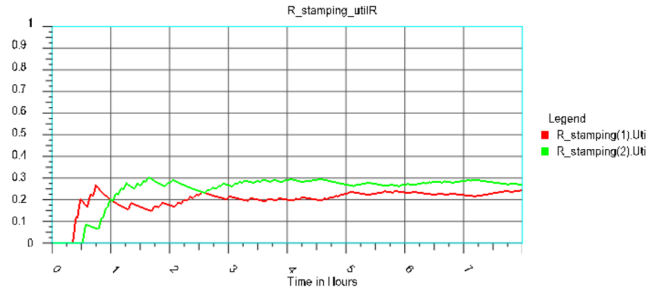
Results:

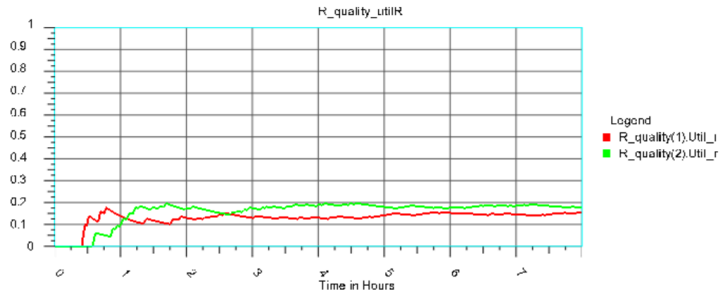


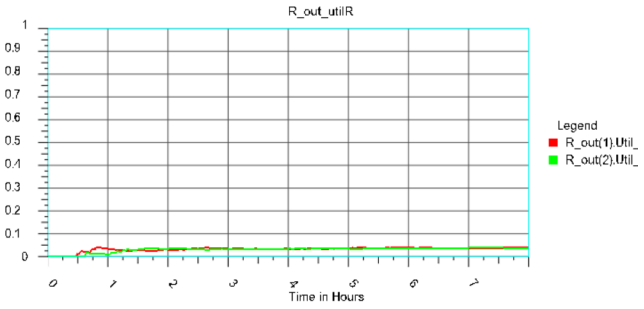


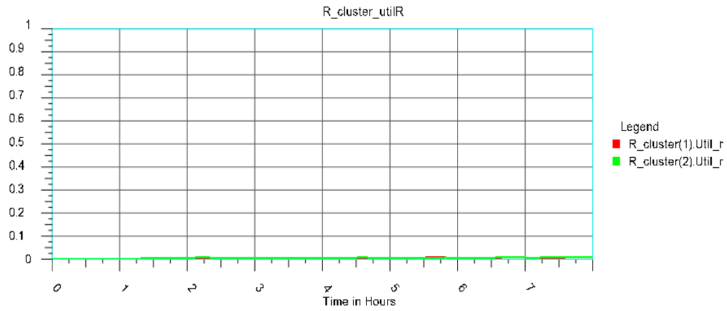


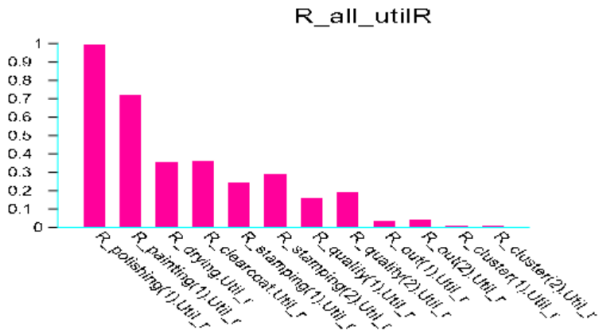


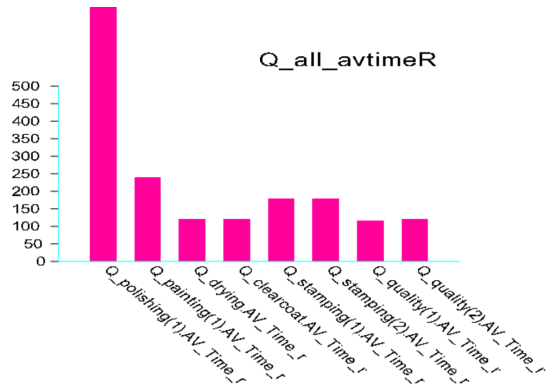


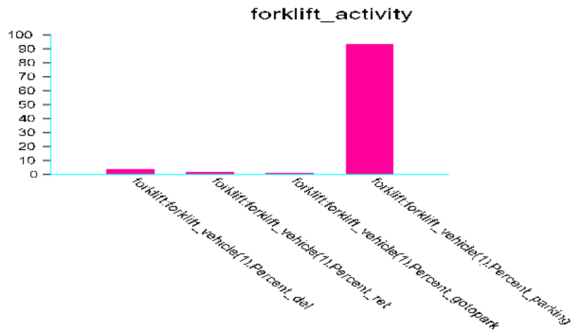












Time interval	Amount of delay
4hrs	+239 minutes
8hrs	+668 minutes

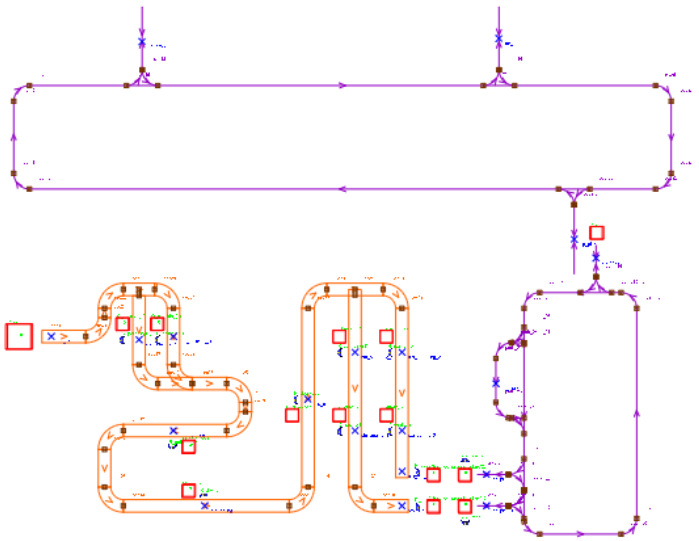
Observations:

The end result shows how this layout isn't capable of keeping up with the input frequency, having a delay in the production of 239 and 668 minutes. The main problem is related to the first process (Polishing), which is slower than the input frequency and gets quickly filled with products. This is observable by looking at the Q_all_avtimeR graph, where the average waiting time for each load is 14699 seconds (about 244 minutes), clearly a non feasible solution. The utilization ratio of the resource related to the polishing operation is nearly 100% because of the overload on that station, while the percentage of utilization for the other resources is less, suggesting that there is the need to balance the workload among the stations, in particular increasing the production rate of the first process.

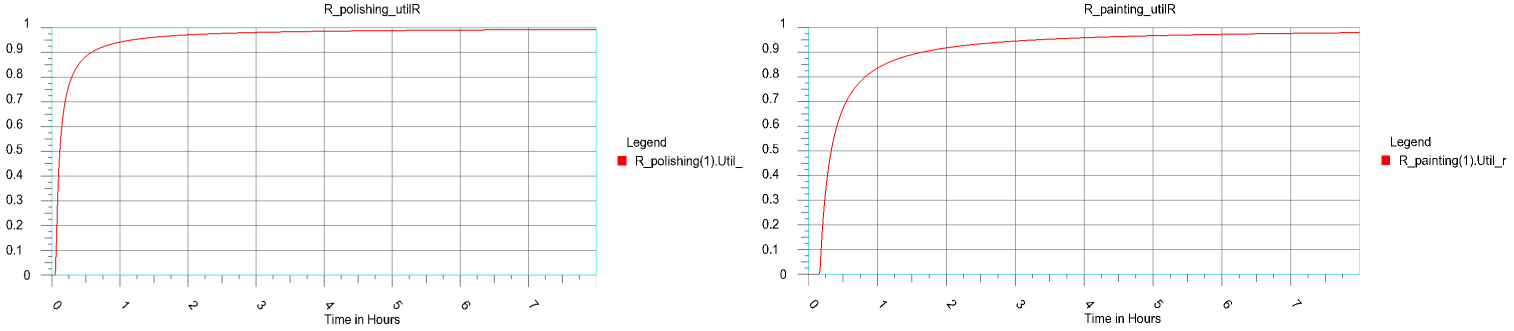
Simulation 2:

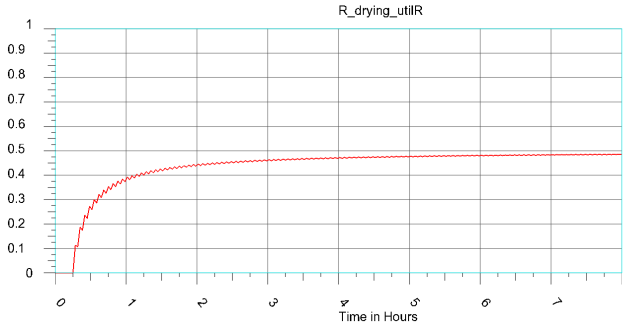
Settings:

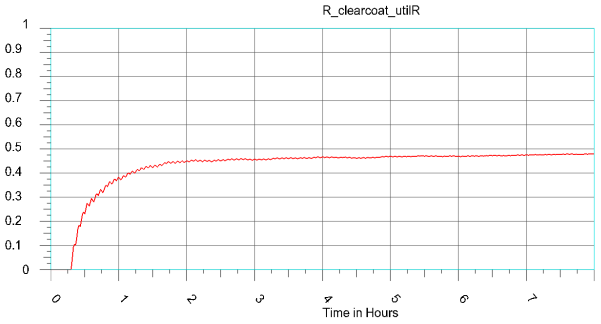
- Two workstations for the polishing process
- One workstation for the painting process

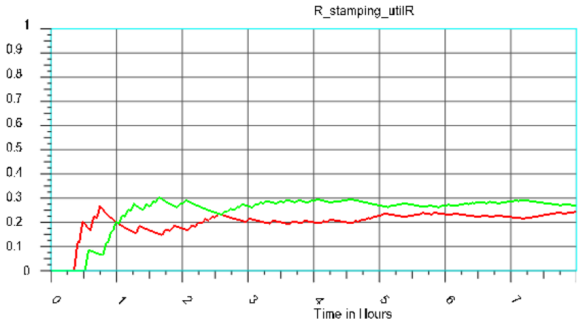


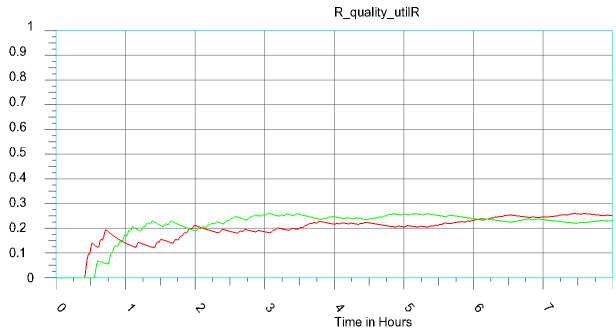
Results:

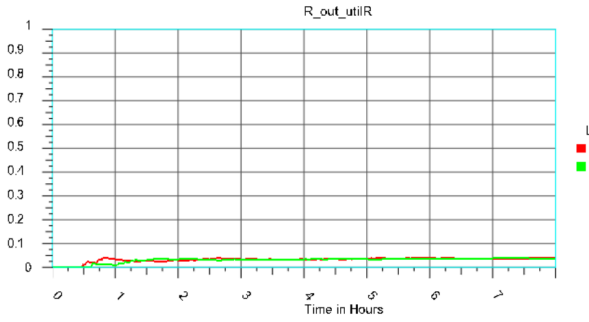


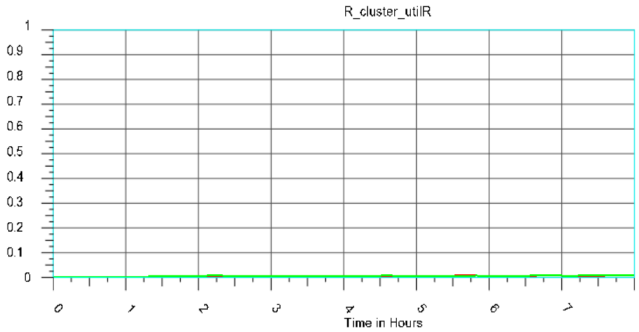

R_drying_utilR
Time in Hours
Legend
R_drying.Util_r

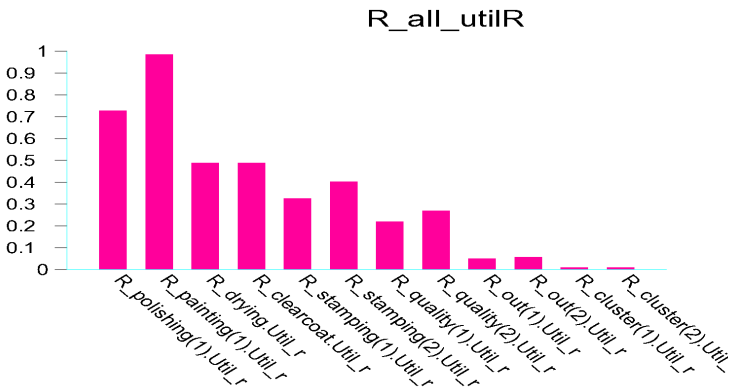

R_clearcoat_utilR
Time in Hours
Legend
R_clearcoat.Util_r

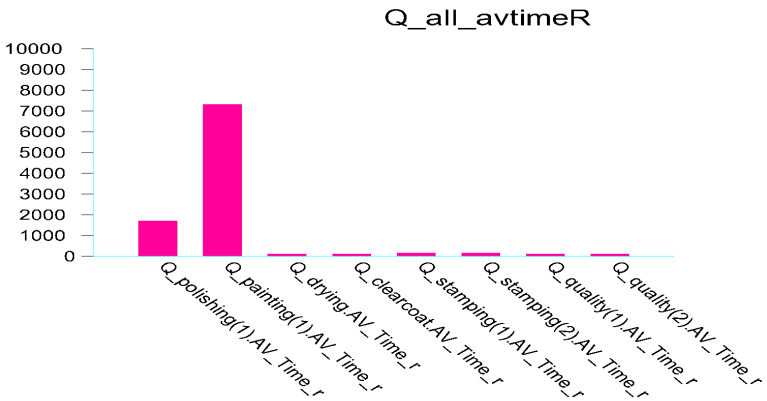

R_stamping_utilR
Time in Hours
Legend
R_stamping(1).Util_r
R_stamping(2).Util_r

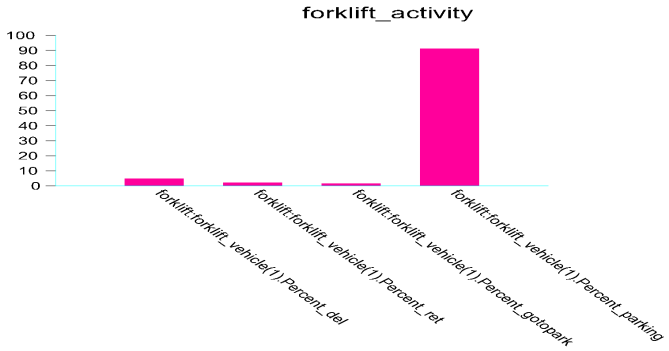

R_quality_utilR
Time in Hours
Legend
R_quality(1).Util_r
R_quality(2).Util_r


R_out_utilR
Time in Hours
Legend
R_out(1).Uti
R_out(2).Uti


R_cluster_utilR
Time in Hours
Legend
R_cluster(1).Util_r
R_cluster(2).Util_r


R_all_utilR
R_polishing(1).Util_r
R_painting(1).Util_r
R_drying.Util_r
R_clearcoat.Util_r
R_stamping(1).Util_r
R_stamping(2).Util_r
R_quality(1).Util_r
R_quality(2).Util_r
R_out(1).Util_r
R_out(2).Util_r
R_cluster(1).Util_r
R_cluster(2).Util_


Q_all_avtimeR
Q_polishing(1).AV_Time_r
Q_painting(1).AV_Time_r
Q_drying.AV_Time_r
Q_clearcoat.AV_Time_r
Q_stamping(1).AV_Time_r
Q_stamping(2).AV_Time_r
Q_quality(1).AV_Time_r
Q_quality(2).AV_Time_r



Time interval	Amount of delay
4hrs	+146 minutes
8hrs	+238 minutes

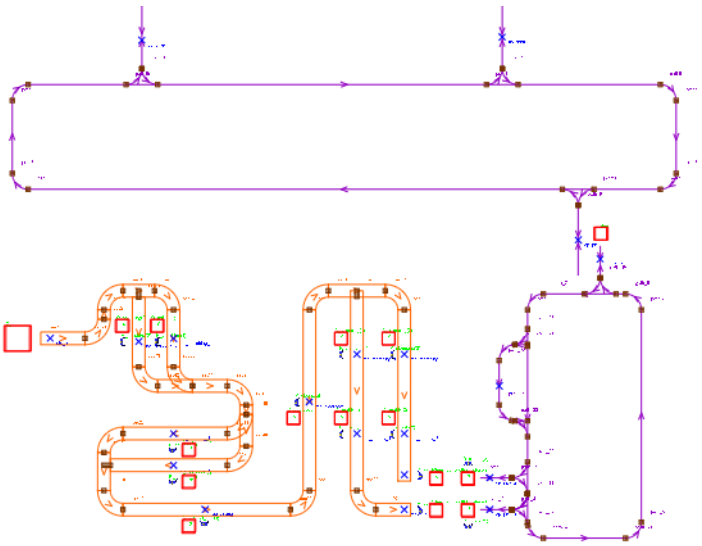
Observations:

By adding another workstation to the polishing process, we were able to reduce the waiting time for the first operation. On the other hand, the problem shifted to the second station. In fact, one single painting station isn't able to deal with the new throughput of the first process, resulting in an average waiting time per load in the queue Q_painting equal to more than 7000 seconds (about 115 minutes). In this scenario, both polishing and painting operators are saturated, having almost 100% of utilization ratio. The end result is still far from optimum, having a delay of 146 and 238 minutes, still not feasible.

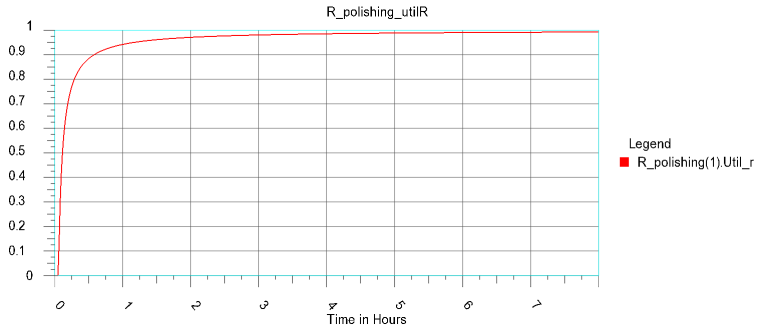
Simulation 3:

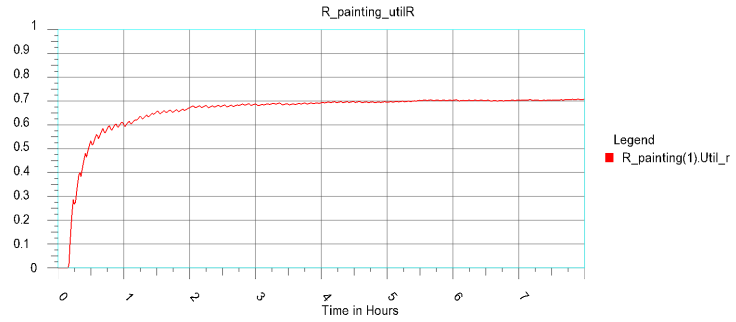
Settings:

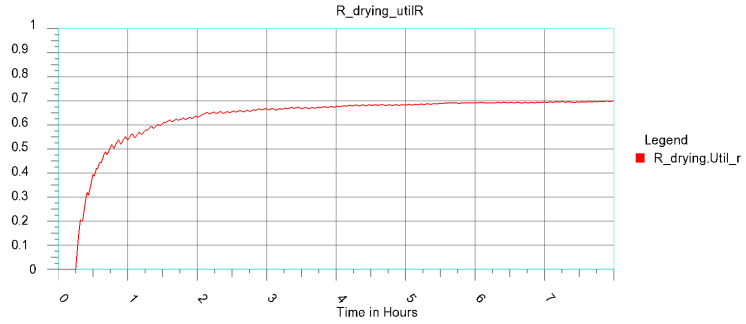
- Two workstations for the polishing process
- Two workstations for the painting process

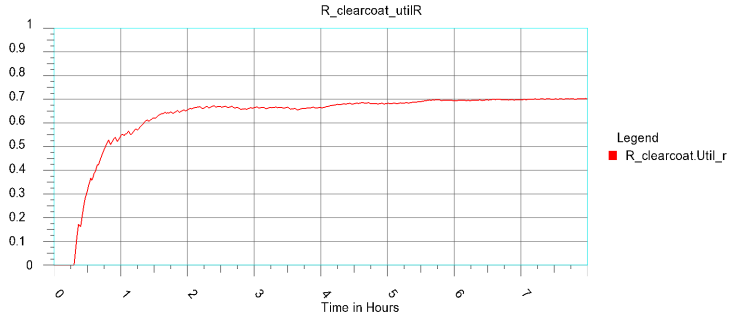


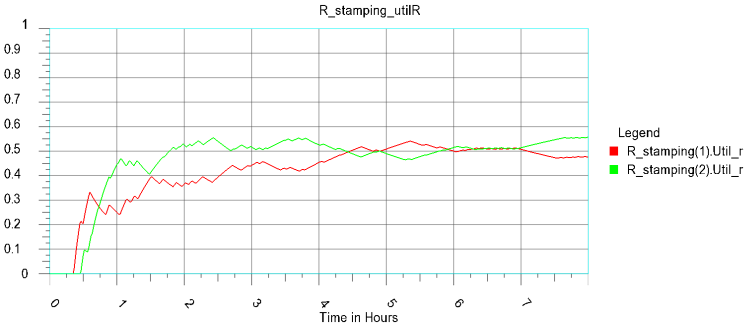
Results:

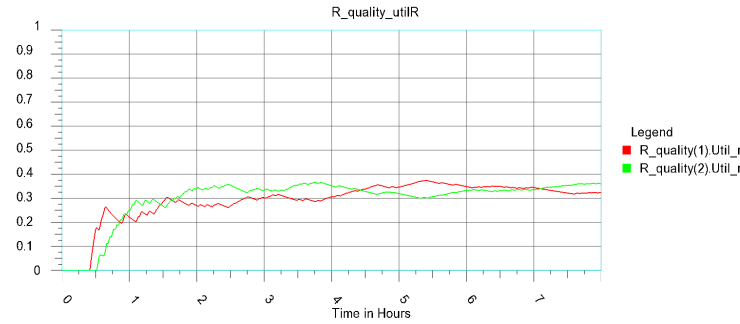

R_polishing_utilR
Time in Hours
Legend
R_polishing(1).Util_r

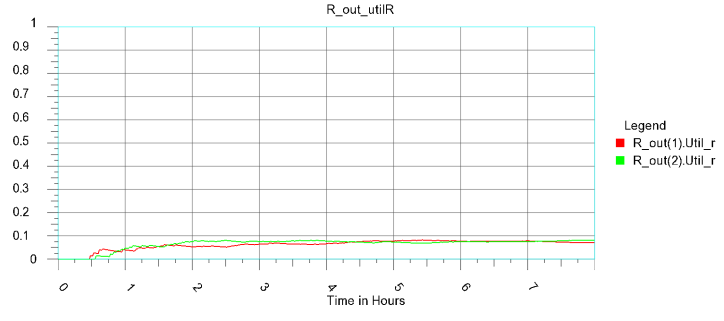

R_painting_utilR
Time in Hours
Legend
R_painting(1).Util_r

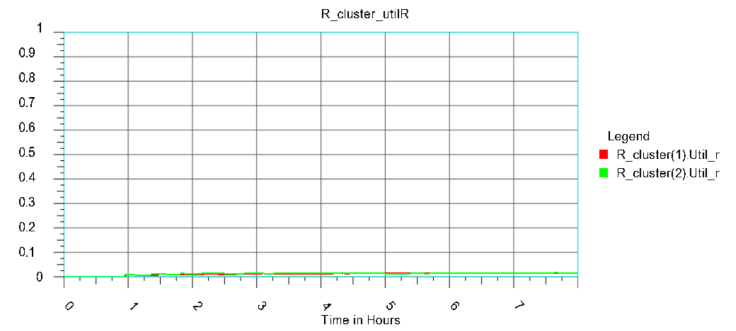

R_drying_utilR
Time in Hours
Legend
R_drying.Util_r

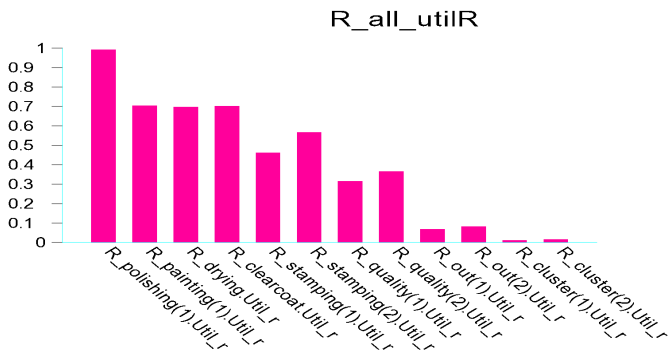

R_clearcoat_utilR
Time in Hours
Legend
R_clearcoat.Util_r

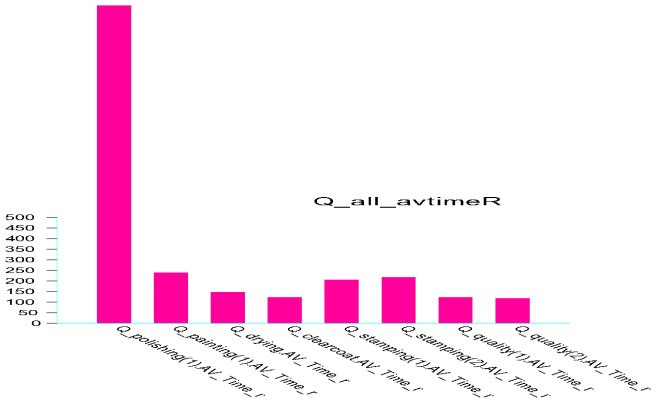

R_stamping_utilR
Time in Hours
Legend
R_stamping(1).Util_r
R_stamping(2).Util_r

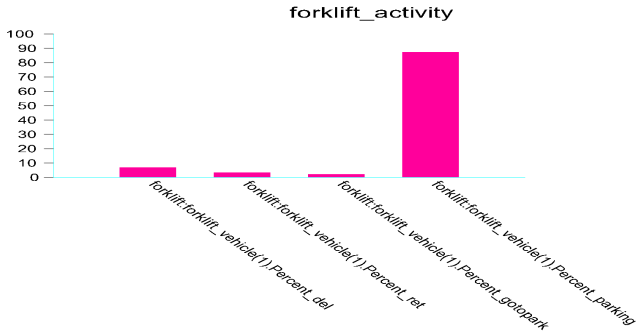

R_quality_utilR
Time in Hours
Legend
R_quality(1).Util_r
R_quality(2).Util_r


R_out_utilR
Time in Hours
Legend
R_out(1).Util_r
R_out(2).Util_r


R_cluster_utilR
Time in Hours
Legend
R_cluster(1).Util_r
R_cluster(2).Util_r







Time interval	**Amount of delay**
4hrs	+23 minutes
8hrs	+20 minutes

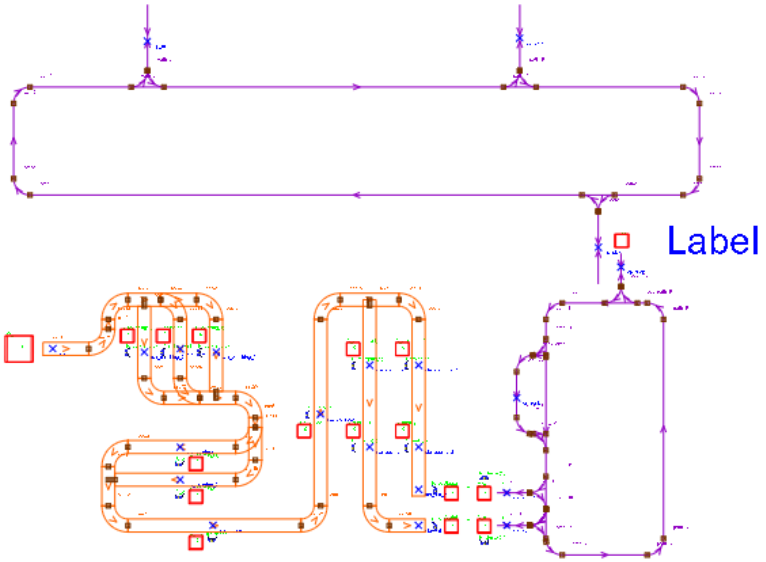
Observations:

In this scenario the end result is much better than the previous results. We have a delay of 23 and 20 minutes for the truck, which could be a feasible solution. Still, the production system is unbalanced. As it is possible to understand looking at the Q_all_avtimeR graph, the first station is again overloaded with work, having an average waiting time in the queue of about 2000 seconds (about 33 minutes) and a utilization ratio close to 100%. By adding a second station for the painting process we were able to optimize the process, but a bottleneck at the beginning of the whole production process remains.

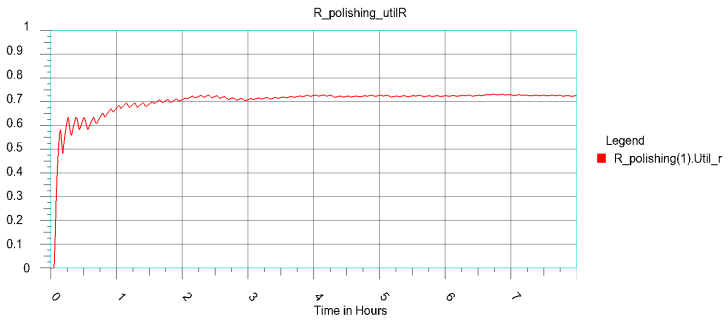
Simulation 4:

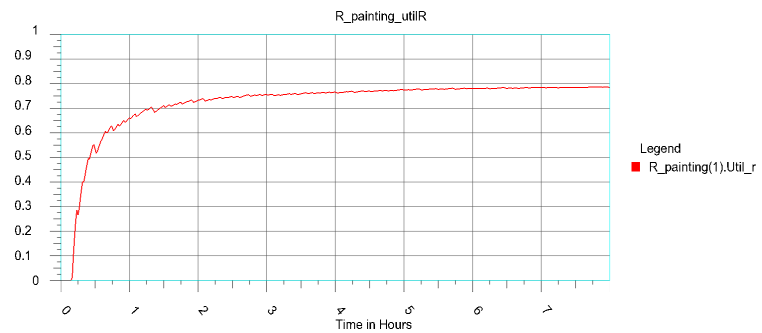
Settings:

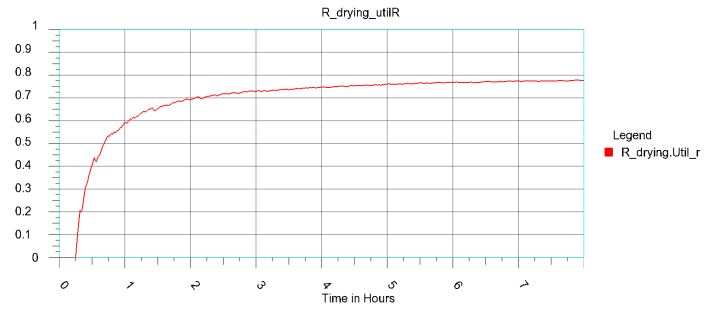
- Three workstations for the polishing process
- Two workstations for the painting process

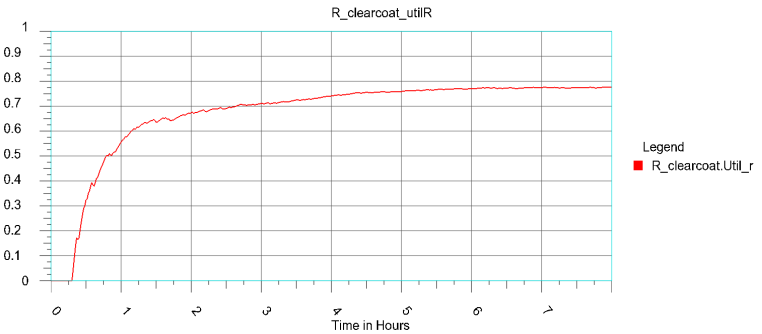


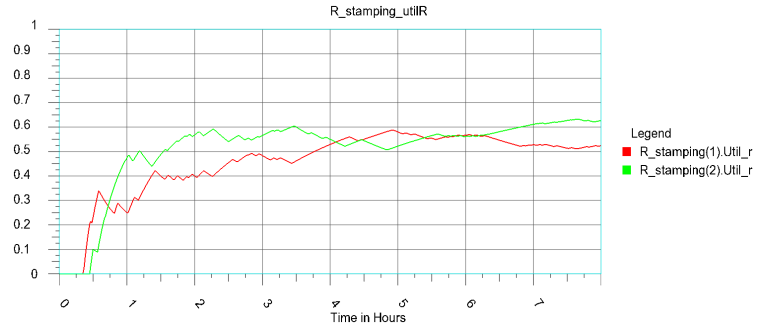
Results:

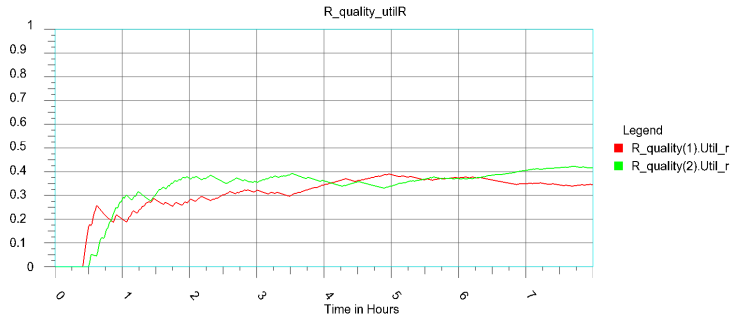


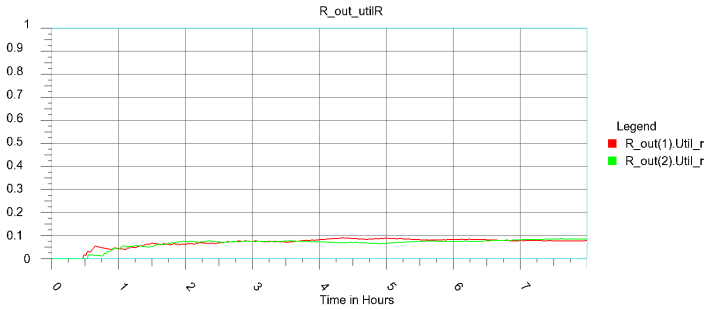


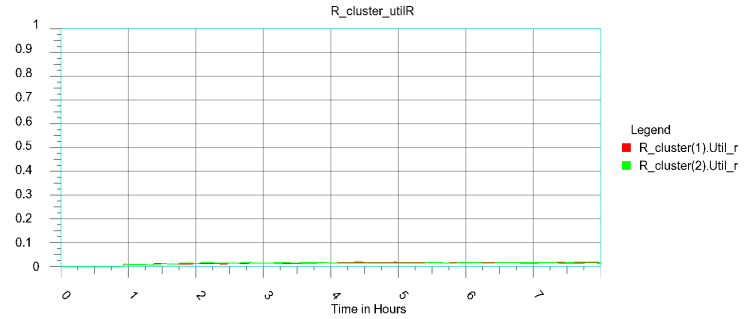


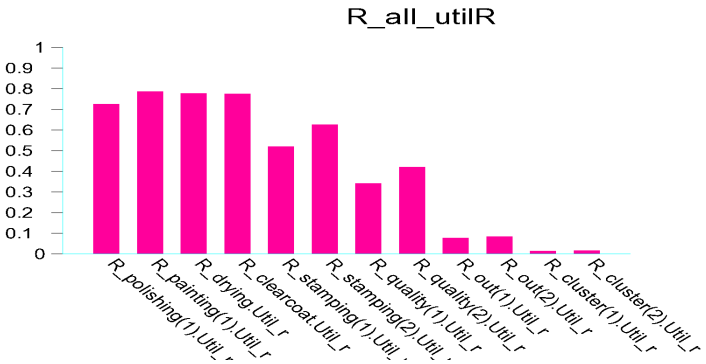


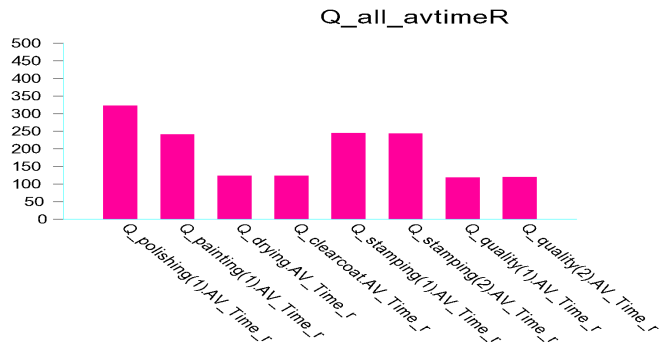

R_stamping_utilR
Time in Hours
Legend
R_stamping(1).Util_r
R_stamping(2).Util_r

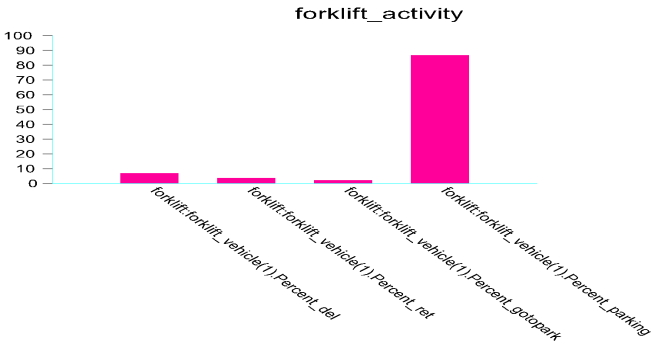

R_quality_utilR
Time in Hours
Legend
R_quality(1).Util_r
R_quality(2).Util_r


R_out_utilR
Time in Hours
Legend
R_out(1).Util_r
R_out(2).Util_r


R_cluster_utilR
Time in Hours
Legend
R_cluster(1).Util_r
R_cluster(2).Util_r


R_all_utilR
R_polishing(1).Util_r
R_painting(1).Util_r
R_drying.Util_r
R_clearcoat.Util_r
R_stamping(1).Util_r
R_stamping(2).Util_r
R_quality(1).Util_r
R_quality(2).Util_r
R_out(1).Util_r
R_out(2).Util_r
R_cluster(1).Util_r
R_cluster(2).Util_r


Q_all_avtimeR
Q_polishing(1).AV_Time_r
Q_painting(1).AV_Time_r
Q_drying.AV_Time_r
Q_clearcoat.AV_Time_r
Q_stamping(1).AV_Time_r
Q_stamping(2).AV_Time_r
Q_quality(1).AV_Time_r
Q_quality(2).AV_Time_r


forklift_activity
forklift:forklift_vehicle(1).Percent_del
forklift:forklift_vehicle(1).Percent_ret
forklift:forklift_vehicle(1).Percent_gotopark
forklift:forklift_vehicle(1).Percent_parking

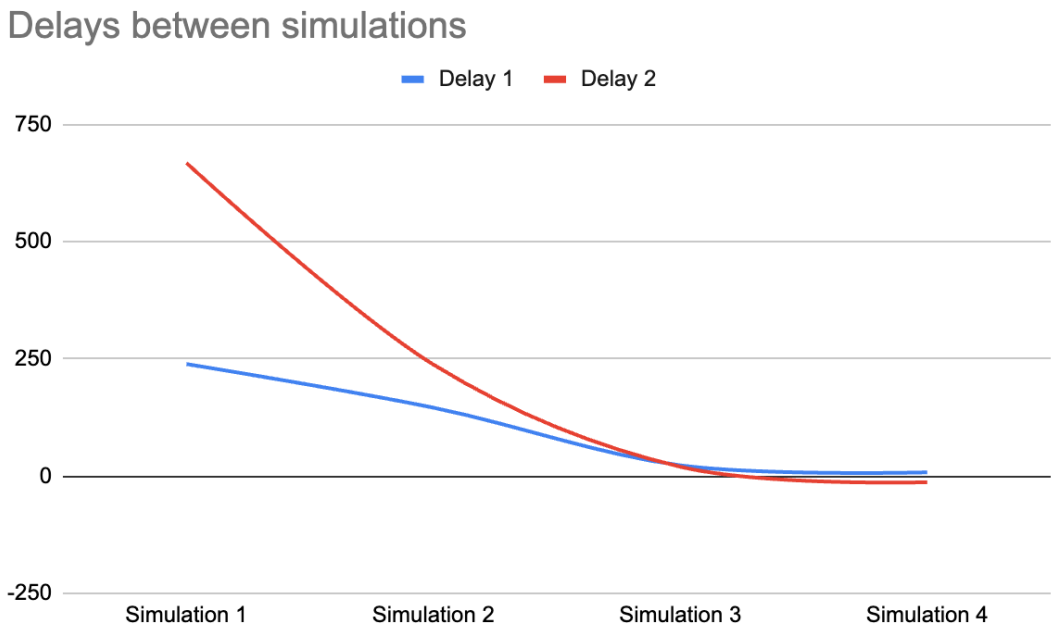
Time interval	Amount of delay
4hrs	+8 minutes
8hrs	-13 minutes

Observations:

The results given by this simulation represent the optimal solution. The production system is now balanced and the waiting time for the loads in the queue is related to the duration of the task itself, meaning that the processes are optimized to reduce the idle time as much as possible. The utilization of the resources is around 70-80% since there is no bottleneck in any of the processes. The final result shows a delay of 8 minutes for the first arrival of the truck, while for the second the production process finished 13 minutes earlier than the arrival time of the truck.

4. Conclusions

The work proposed focused on designing a production system able to minimize the delay of production compared to the arrival of the truck. We were able to obtain the optimal results by parallelizing the work for the polishing and the painting processes. More particularly, we added two additional stations for polishing and one for painting, increasing the number of resources. These changes made the whole production more smooth and efficient, reducing the idle time of the stations trying to maximize the utilization of the human operators. Here below is reported a graphical representation of the evolution of the delay considering all simulations:



Simulation 1	***Simulation 2***	***Simulation 3***	***Simulation 4***
Polishing stat. = 1 Painting stat. = 1	Polishing stat. = 2 Painting stat. = 1	Polishing stat. = 2 Painting stat. = 2	Polishing stat. = 3 Painting stat. = 2

By looking at the graph it is clear that the final delay is nearly optimal. We still have to consider the stochasticity introduced by the quality check process, which removes defective products randomly. Generally speaking, the last simulation shows how the optimal utilization ratio for resources is around 70-80%. Having a higher percentage of usage will create a worse scenario, where the production system produces products faster than needed. Consequently, the production will finish earlier than the truck arrival, creating a congestion in the last part of the forklift and conveyor system.