Hochschule für
Wirtschaft und Recht Berlin
Berlin School of Economics and Law

Operations and Supply Chain Simulation with AnyLogic

Decision-oriented introductory notes for management students in bachelor and master programs

Prof. Dr. Dmitry Ivanov

Berlin School of Economics and Law

Professor of Supply Chain Management

To be cited as: Ivanov D. (2017). Operations and supply chain simulation with AnyLogic: Decision-oriented introductory notes for master students. 2nd Edition, E-Textbook, Berlin School of Economics and Law (preprint).

Content

Introduction **5**

Part I Process capacity analysis and workload balancing

1. Business Simulation Game „Process capacity analysis and workload balancing with AnyLogic“ **7**

1.1 Learning objectives *7*

1.2 Problem statement *7*

2. Model building **8**

2.1 Step 1. Create process model *8*

2.2 Step 2. Define rules for incoming orders *9*

2.3 Step 3. Define queue rules *10*

2.4 Step 4. Define processing rules *10*

2.5 Step 5. Create basic animation *11*

2.6 Step 6. Simulation experiment *12*

2.7 Step 7. Collecting statistics and KPI dashboard design *13*

2.7.1 Revenue, costs, profit *13*

2.7.2 Capacity utilization *15*

2.7.3 Lead time and flow time *17*

2.7.4 Backlog and queue analysis *18*

3. Experiments and managerial insights **21**

3.1 Experiment 1 ..... *21*

3.2 Experiment 2 ..... *22*

3.3 Experiment 3 ..... *25*

4. Extensions **28**

4.1 How to introduce the time limits for waiting in the queue *28*

4.2 Interface creation for playing the game: views, sliders, parameters, variables, functions, and events *29*

4.2.1 Views *29*

4.2.2 Parameters *30*

4.2.3 Variables *30*

4.2.4 Functions *30*

4.2.5 Events *31*

4.2.6 Sliders *32*

4.2.7 KPI dashboard creation *32*

4.2.8 Resources *33*

5. Playing the game **34**
5.1 Game rules *34*
5.2 Round #1 ..... *34*
5.3 Round #2 ..... *35*
5.4 Round #2: Optimization experiment *37*
5.5 Round #2: Optimization-based simulation experiment: multiple objective decision making *39*
5.6 Rounds #3-#10 and game evaluation *41*

6. Further possible extensions and other games **41**

Part II Capacity flexibility simulation

7. Business Simulation Game „Capacity flexibility simulation with AnyLogic“ **44**
7.1 Learning objectives *44*
7.2 Problem statement *44*
8. Model building **46**
8.1 Step 1. Create process model *46*
8.2 Step 2. Create custom agent and define rules for order arrival, waiting, and processing: usage of time functions and resource pools *47*
8.3 Step 3. Collecting statistics and KPI dashboard design *50*
8.3.1 Revenue, costs, profit *50*
8.3.2 Resource pool capacity utilization *50*
8.3.3 Lead time with the use of time function *51*
8.3.4 Total output, completed on time (OTD), delayed and lost orders: usage of Java code for conditions *51*
9. Playing the game **53**
9.1 Game rules *53*
9.2 Round #1 ..... *54*
9.3 Round #2 ..... *55*
9.4 Round #2: Optimization experiment *57*
9.5 Round #2: Optimization-based simulation experiment *57*
9.6 Rounds #3 – #10 and game evaluation *58*
9.7. Capacity flexibility analysis *60*

Part III Supply Chain Coordination

10. Supply Chain Coordination with AnyLogic 65
10.1 Learning objectives 65
10.2 Problem statement 65
11 Model building 66
11.1 Create process model 66
11.2 Assembler 66
11.3 Transport with the help of "Conveyor" and "MoveTo" 67
11.4 Batch 67
12. Experiments and managerial insights 68

Part IV Inventory Control

13. Modelling periodic and continuous review inventory control policies with AnyLogic ... 72
13.1 Learning objectives 72
13.2 Problem statement 72
14. Model building 73
14.1 EOQ model: event-based simulation 73
14.2 Modelling stochastic demand: periodic and continuous review policies 75
14.3 Modelling stochastic demand and lead time: re-order point using Java 77
14.4 Inventory holding, ordering and stockout costs 79
14.5 Re-order point and safety stock 81
15. Experiments and managerial insights 84
15.1 Preparing experiment: using Action Charts 84
15.2 Preparing experiment: dynamic target inventory 86
15.3 Experiment 1 for engine oil: impact of demand dynamics 89
15.4 Experiment 2 for nuts and bolts: impact of lead time dynamics 92
15.5 Experiment 3 for nuts and bolts: impact of order quantity 94
16. Extensions 96
16.1 Inventory management in supply chains with production and transportation considerations: system dynamics 96
16.2 Agent-based modelling the market demand 96
17. Literature 97
18. Discussion 97

Introduction

This introductory note was created in order to support undergraduate and master students with majors in supply chain and operations management as well as instructors giving classes in supply chain and operations simulation for such students. Without relying heavily on statistics and mathematical derivations, this guideline offers applied models and a simple, predictable format to make it easy to understand for management students without engineering background.

While teaching management students in simulation and optimization classes, it is a challenging task to combine modelling and management decision-making views. On one hand, application of optimization and simulation software implies some background in programming. On the other hand, technical issues in development of optimization and simulation models may distract the attention and time from the real objective, i.e., management decision analysis and decision-making support with the help of simulation and optimization software.

In this introductory note to operations and supply chain simulation using AnyLogic, the focus is to introduce into the basic principles of using simulation for decision-making. In reducing technical complexity to the necessary minimum, the main attention is paid to management decision analysis and using KPI for operational, customer and financial performance for decision-making.

The material is grouped into four parts that correspond to three basic process structures as well as basic inventory control policies:

- Linear manufacturing system analysis
- Single-stage manufacturing system analysis
- Supply chain network system analysis
- Periodic and continuous review inventory control system analysis

For these system structures, simulation games are presented. First, technical development of the models is described. Step-by-step, the process model building and KPI evaluation techniques are introduced and illustrated. Second, the developed models are used for playing the games and using simulation results for decision-making. Finally, we discuss possible extensions that can be used for student assignments or master thesis.

All the models and their source files can be downloaded at AnyLogic Cloud.

Instructors can also download the models from Companion Web Site to the book "Global Supply Chain and Operations Management" http://global-supply-chain-management.de
Being focused on the management issues, the model developments are described as easy as possible. At times, simulation professionals would suggest more elegant but also more complicated ways to model building. In any case, this textbook has to provide elementary basics for simulation model design and using the simulation results for decision-making. The AnyLogic is a tool for these objectives since it uses three simulation methods (discrete events, system dynamics, and agent-based modelling) along with optimization-based simulation experiments.

Since this guide is just at the beginning of its development, we excuse some possible errors in the texts and formatting. We consider this guide rather as a working material.

The author of this guide has also co-authored the textbook "Global Supply Chain and Operations Management" by Springer and its companion web site global-supply-chain-management.de where AnyLogic and anyLogistix models can be found.

NEW! anyLogistix textbook available at https://www.anylogistix.com/resources/books/alx-textbook/

The author deeply thanks The AnyLogic Company for valuable feedbacks and improvement suggestions.

Part I Process capacity analysis and workload balancing

1. Business Simulation Game „Process capacity analysis and workload balancing with AnyLogic“

1.1 Learning objectives

1) To develop analytical and management skills on demand, capacity and workload management decision-making
2) To develop soft skills on coordinated team-based decision making
3) To develop technical skills on creating discrete-event process simulation models and perform performance measurement analysis on example of AnyLogic multimethod simulation software

1.2 Problem statement

We consider a linear manufacturing process that contains four machines (Fig. 1):

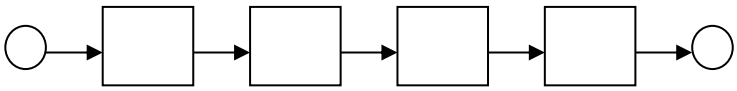


Fig. 1. Manufacturing line

The customer orders arrive at some partially uncertain rate at partially uncertain time, e.g., each 3-5 hours, the range of 20-30 orders is to expect. The orders are different regarding the processing times at the machines but they are processed in the same sequence (i.e., the flow shop) at four subsequent machines. Each order has partially variable processing times at the machines, i.e., machine 1 may need 1.5; 2; or 4 minutes to process the order. The order completion at the last machine brings us revenue X, but also creates the costs Y. If an incoming order cannot be processed because of insufficient line capacity, it needs to wait. If the waiting time of an order exceeds some time limit, the customer withdraws this order. This implies opportunity costs O. The waiting orders build a backlog. Each unprocessed backlog unit creates costs Z (this costs can be computed either at the end of planning period or for some time periods, e.g., for each weak of waiting). Each partially processed but not completed unit is called WIP (work-in-progress) and creates inventory costs W to be computed on the principles similar to the backlog costs Z.

The management objective is to balance incoming customer orders with capacity and the demand in order to achieve maximal possible profit under given constraints.

The students are divided into *teams* each of which comprises the following *players*:

- A division head who strives to maximize total profit of the division and unit profits for the orders
- A sales manager who strives to maximize revenue, to accept as many customer orders as possible and to complete the orders as quick as possible subject to lead-time
- A manufacturing manager who seeks for the lowest unit costs, high capacity utilization, and low flow time
- A workload balancer who needs to balance the objectives of the division head, the sales manager, and the manufacturing manager in order to achieve maximal possible profit under given constraints.

The teams get from the instructor input data set and compete against each other subject to total profitability. At the same time, the managers within each team compete against each other to become the best from all the teams in regard to their individual KPI.

2. Model building

With the help of AnyLogic multimethod simulation software, the management problem will be transferred to a simulation model that will allow performing experiments in order to understand basic trade-offs and relations in process flow analysis with consideration of financial and operational KPIs (Key Performance Indicators).

For building simulation models in AnyLogic, a free PLE (Personal Learning Edition) license can be downloaded from web site www.anylogic.com. Having installed the software, the user gets access to an introductory course (Free Book: AnyLogic 7 in Three Days) of model building in AnyLogic where technical basics are considered in detail. We refer to this book and additional extensive Helps in AnyLogic in order to learn how to build models technically. We highly recommend considering example models available in each PLE package in the Help option. First of all, the models "Activity Based Costing Analysis", "Call Center", "Job Shop" need to be studied to get feeling on building basic types of process flows, animation and statistical analysis. Subsequently, we recommend to consider models "AB Market and SD Supply Chain", "Adaptive Supply Chain", "Supply Chain", "Distribution Center", and Wholesale Warehouse" to understand agent-based simulation, system dynamics simulation, and logistics processes.

2.1 Step 1. Create process model

In this step, we create a process model for our problem statement (Fig. 2).

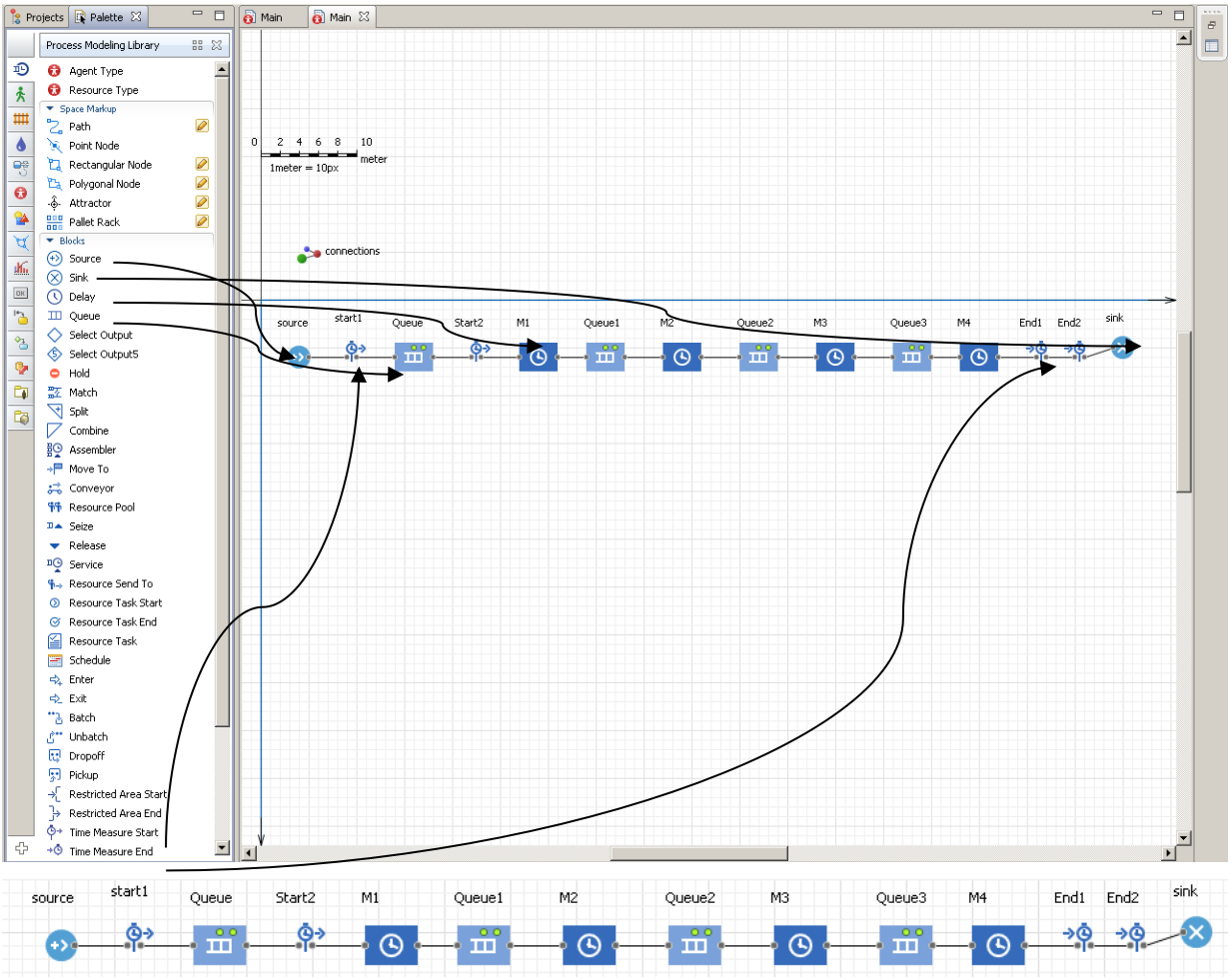


Fig. 2. Process model

The created model contains a source node where the customer orders arrive, four machines (M1-M4), the queues where the orders will wait if capacity of respective subsequent steps is insufficient, and a sink node as the process end. We also included four points to measure the times. Point "start1" measures the time of customer order arrival, point "Start2" measures the time of customer order dispatching at the manufacturing line, and points "End1" and "End2" measure the time of order completion. The time period "End1 – start1" is called *lead time* and time period "End2 – Start2" is called *flow time.*

2.2 Step 2. Define rules for incoming orders

In this step, we define the rules for the quantity and frequency of incoming orders (Fig. 3).

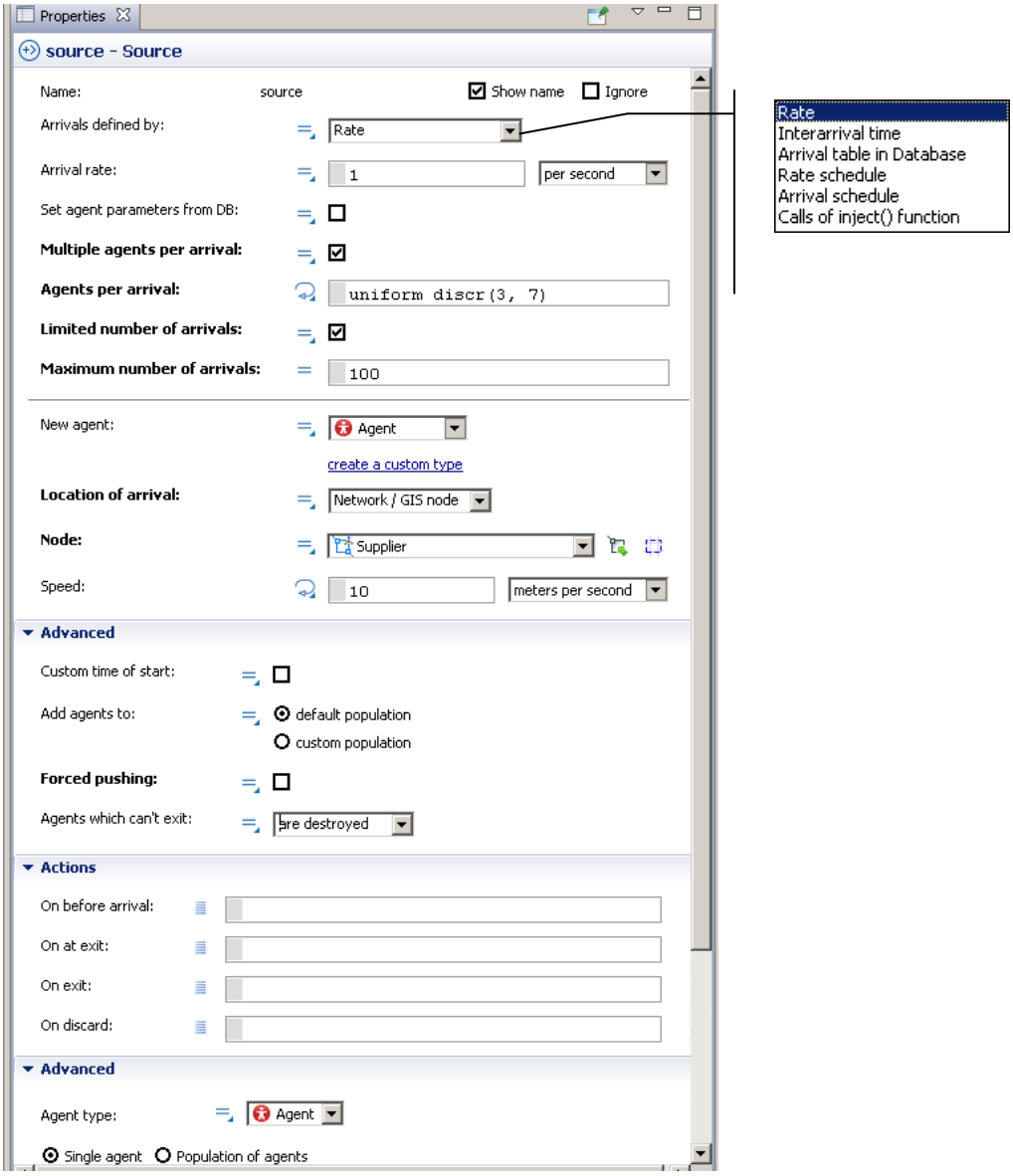


Fig. 3. Defining incoming flow (input)

In the source properties, arrival can be defined by rates, interarrival time, etc. In the example on Fig. 3, the range of 3-7 orders will arrive at an exponentially distributed arrival rate of 1 second. We can limit our experiments, e.g., to 20 arrivals.

Next, *push vs pull* policy for the process can be selected in the source properties. By default, the field "Forced pushing" is enabled and the source does not allow the orders to wait at its output until they can be consumed by the next object. You can change this setting by deselecting the

advanced option “Forced pushing” and choosing the required behaviour (destroy or wait in the queue) from the list “Agents which can't exit”. This will imply the policy where the orders will be not pushed but rather pulled in the manufacturing line in dependence of available capacity in correspondence to one-piece flow and CONWIP control techniques.

2.3 Step 3. Define queue rules

In this step, we define the rules for the transition of customer orders into manufacturing line as well as internal transitions in the line with the help of setting the queue properties (Fig. 4).

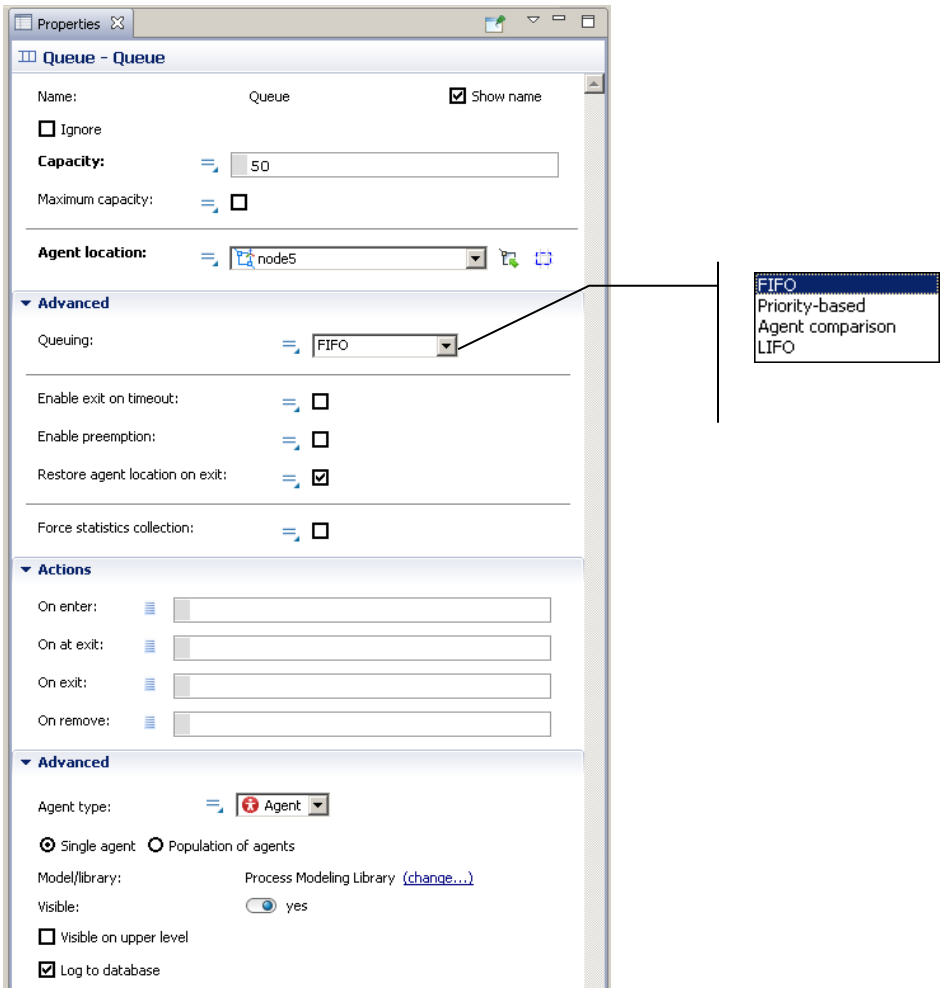


Fig. 4. Defining queue rules

First, queue capacity can be limited or unlimited (maximum capacity). Second, dispatching rules can be set up as FIFO, priority-based (i.e., orders from the most important customers are dispatched first; this needs to be activated by “enable pre-emption”), agent comparison (i.e., preferences for certain orders), and LIFO. By activating “enable exit on timeout”, it becomes possible to describe the situation where the waiting time in the queue is limited and if an order is not dispatched during this time, it disappears. For example, if a customer order cannot be dispatch within two weeks, the customer will withdraw it and place at our competitor. Such orders create opportunity costs that can be computed based on comparison of Output at the sink model, Input at the source node, and the number of the dispatched orders at the M1 (or at exit in the queue prior to M1).

2.4 Step 4. Define processing rules

In this step, we define the rules for order processing at the machines in manufacturing line with the help of setting the delay properties (Fig. 5).

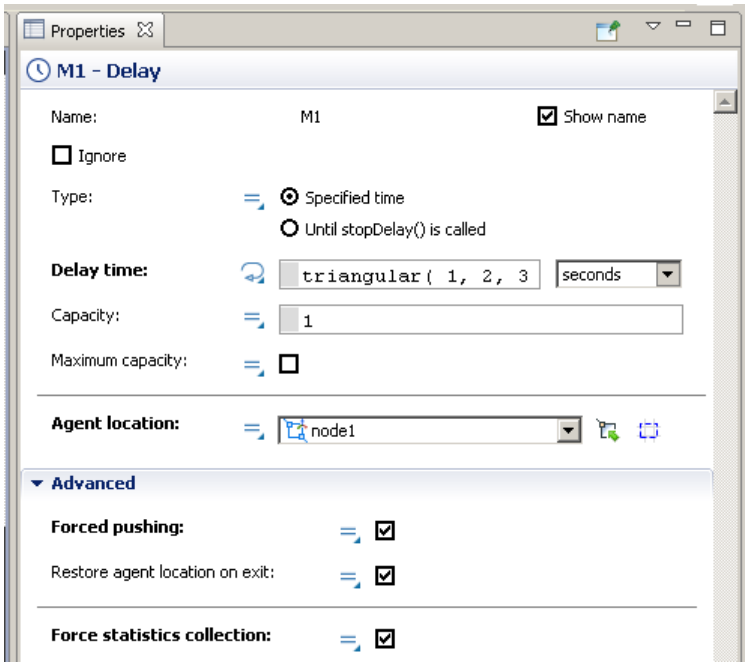


Fig. 5. Defining processing rules

The processing time can be fixed or partially uncertain. Capacity of the processing machine defines maximum number of orders that can be processed at the machine simultaneously. In the advanced models, the processing stage can comprise not only machines but also different resources (e.g., a forklift). Such advanced description can be implemented with the help of functions "Resource", "Seize", Release" or "Service".

2.5 Step 5. Create basic animation

One of the key simulation advantages in the process visualization. The visualization helps the managers to understand the simulation results better and makes the experiments to be an exciting task. The range of different visualization levels is very broad, from basic blocks to 3D animation. We consider here a very simple procedure (Fig. 6).

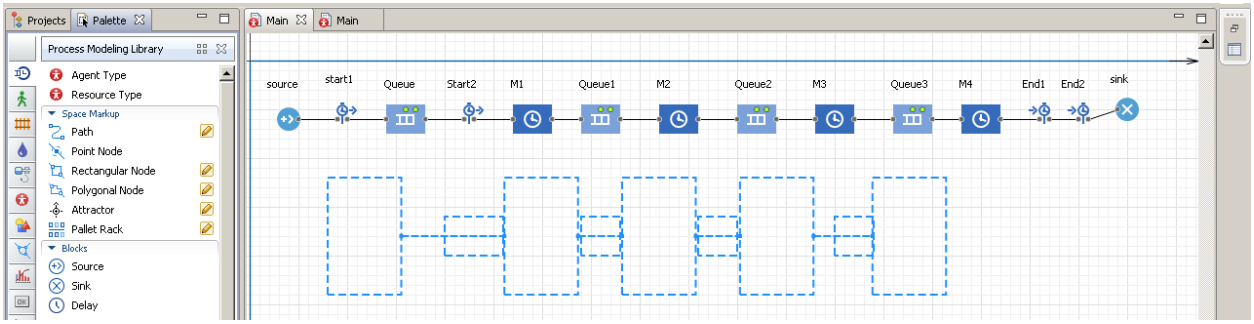


Fig. 6. Process visualization

Using space markup function, we draw nine rectangular nodes (going to be the source node, four waiting areas and four machines), name them and connect them with the help of paths. Subsequently, in the properties of each process diagram element (i.e., source, queues and delays), we assign the drawn blocks (Fig. 7).

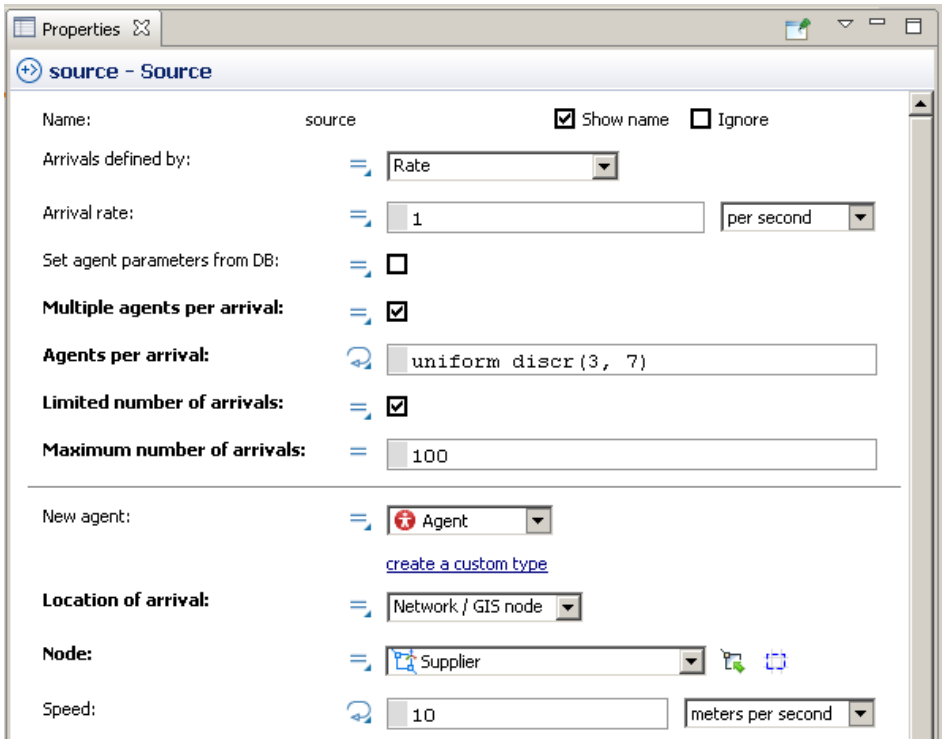


Fig. 7. Matching process diagram and visualization

First, the location of arrival is selected as "Network / GIS node". Second, the corresponding node from the markup diagram needs to be selected.

2.6 Step 6. Simulation experiment

Now we are ready to run the experiment by going to "Run Model" (Fig. 8).

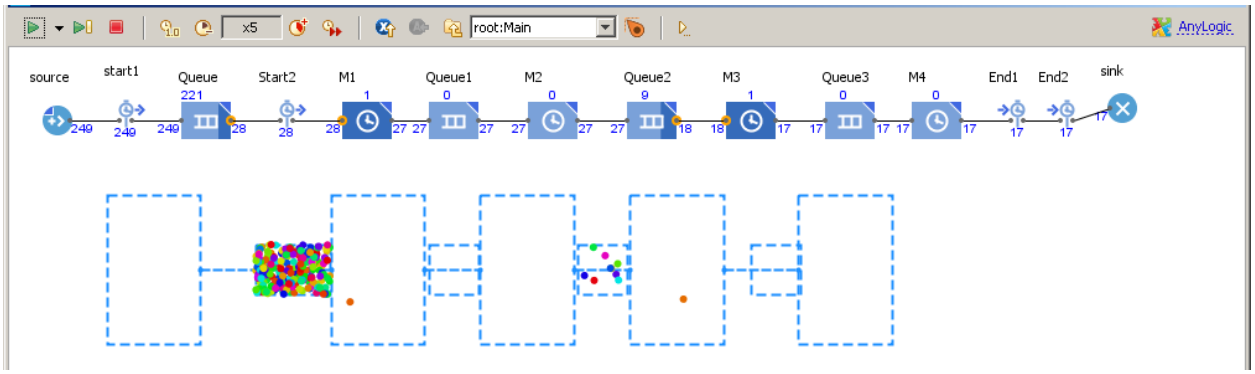


Fig. 8. Experiment window

In the process diagram and visualization field, the operational process dynamics can be observed. In the "Projects – Simulation", experiment parameters can be set up, e.g., maximal experiment time 100 seconds that corresponds to 100 days or hours of real life (Fig. 9).

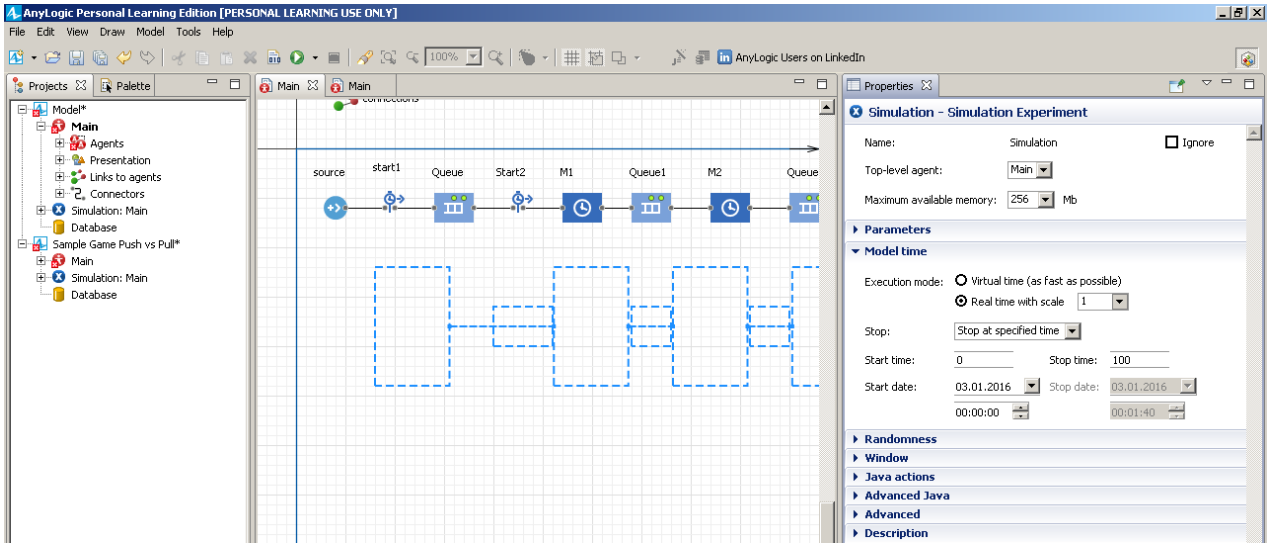


Fig. 9. Experiment settings

The process dynamics observation is very useful but for managers it is insufficient. A manager needs an evaluation of both financial (i.e., revenue, costs, profits) and operational performance (i.e., lead time, flow time, capacity utilization, and backlog). In other words, in the last step prior to starting the experimental part, *a KPI cockpit or dashboard* needs to be created.

2.7 Step 7. Collecting statistics and KPI dashboard design

In this step, we are going to create diagrams to analyse the following KPIs:

- Revenue, costs, profit
- Capacity utilization
- Lead time and flow time
- Backlog

The statistical analysis is designed in "Palette → Analysis" (Fig. 10)

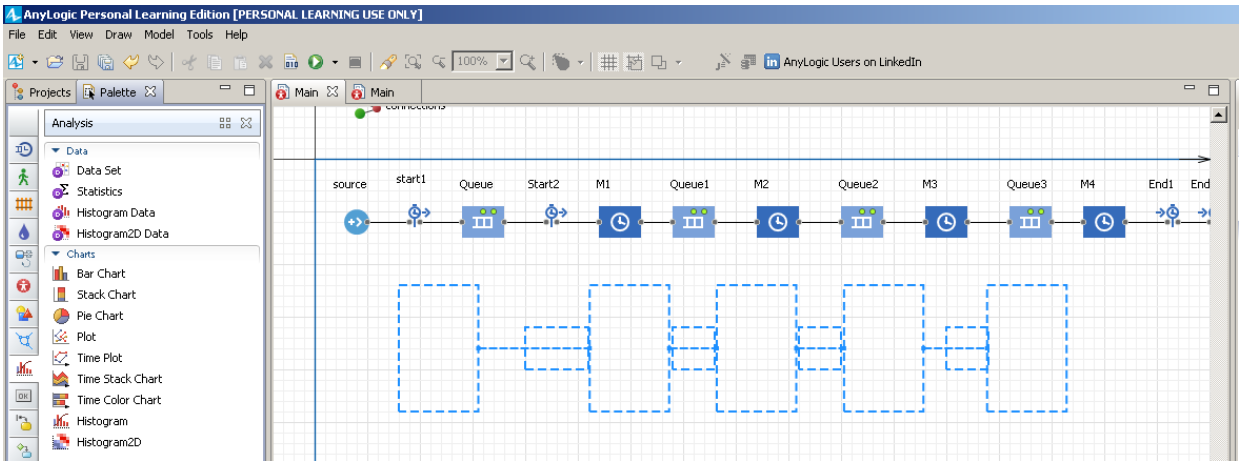


Fig. 10. Statistical analysis design

Depending on the KPI, different kinds of statistics collection and representation as bar chart, stack chart, plot, time plot or histogram can be used.

2.7.1 Revenue, costs, profit

To compute financial performance as revenue, costs and profit, we need to define some parameters and variables in "Palette → Agent" as shown in Fig. 11:

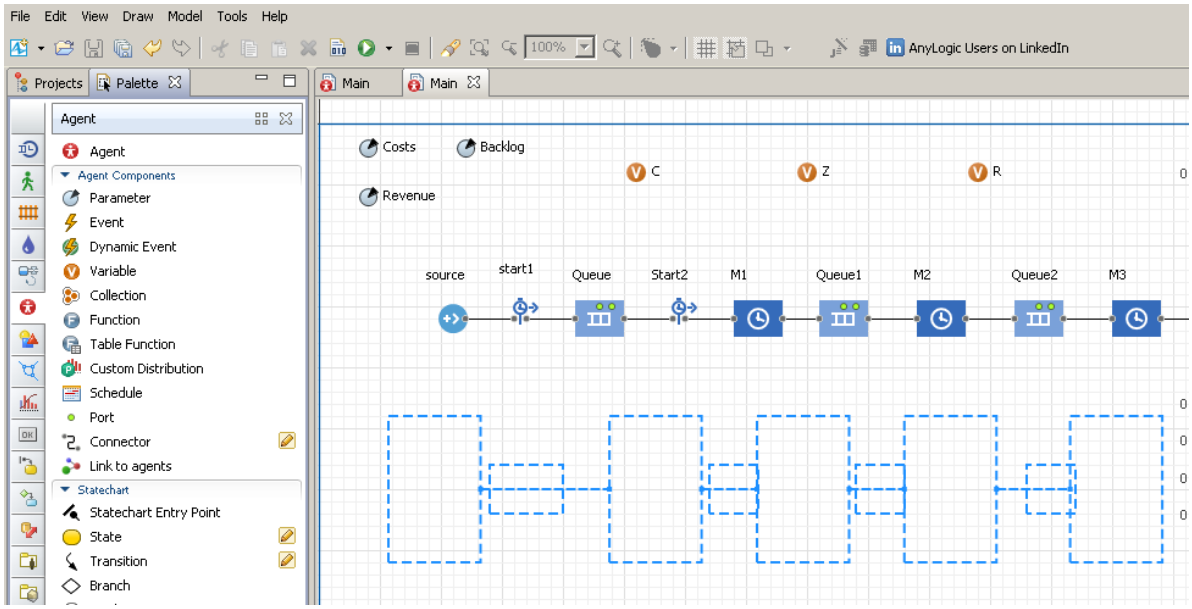


Fig. 11. Parameter and variable definition

In the parameters “Costs”, “Revenue”, and “Backlog”, we define the monetary estimations of manufacturing costs, revenue, and backlog costs for each order respectively. Then we select a bar chart and define the data using the variables C (costs), R (revenue) and Z (backlog) as shown in Fig. 12.

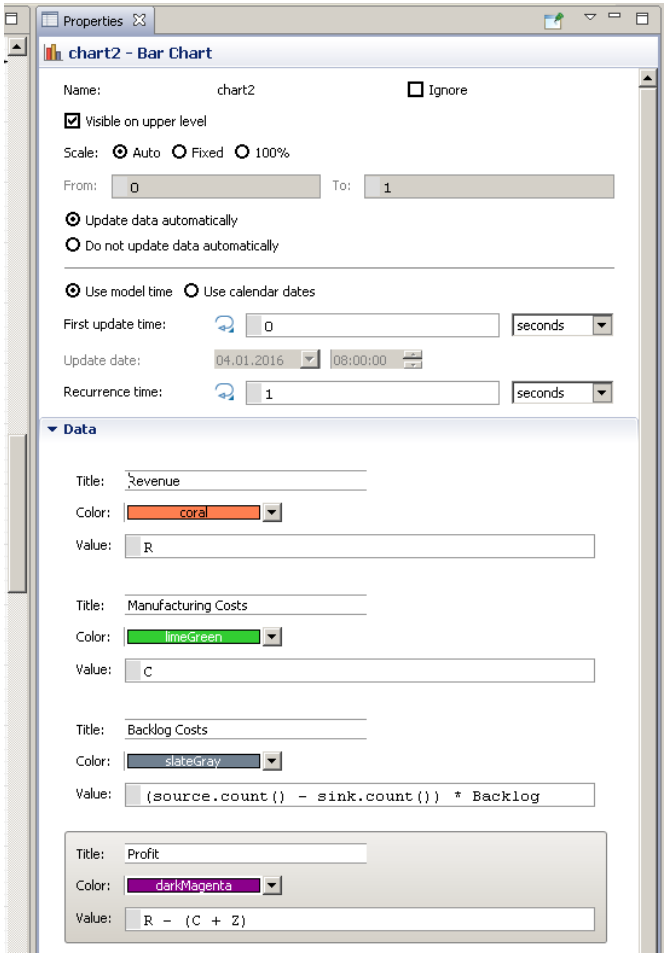


Fig. 12. Data definition for financial performance analysis

In order to update the variables R, C and Z, we need to define some time points in the process flow diagram for this updating. For example, we can do it in the way shown in Fig. 13.

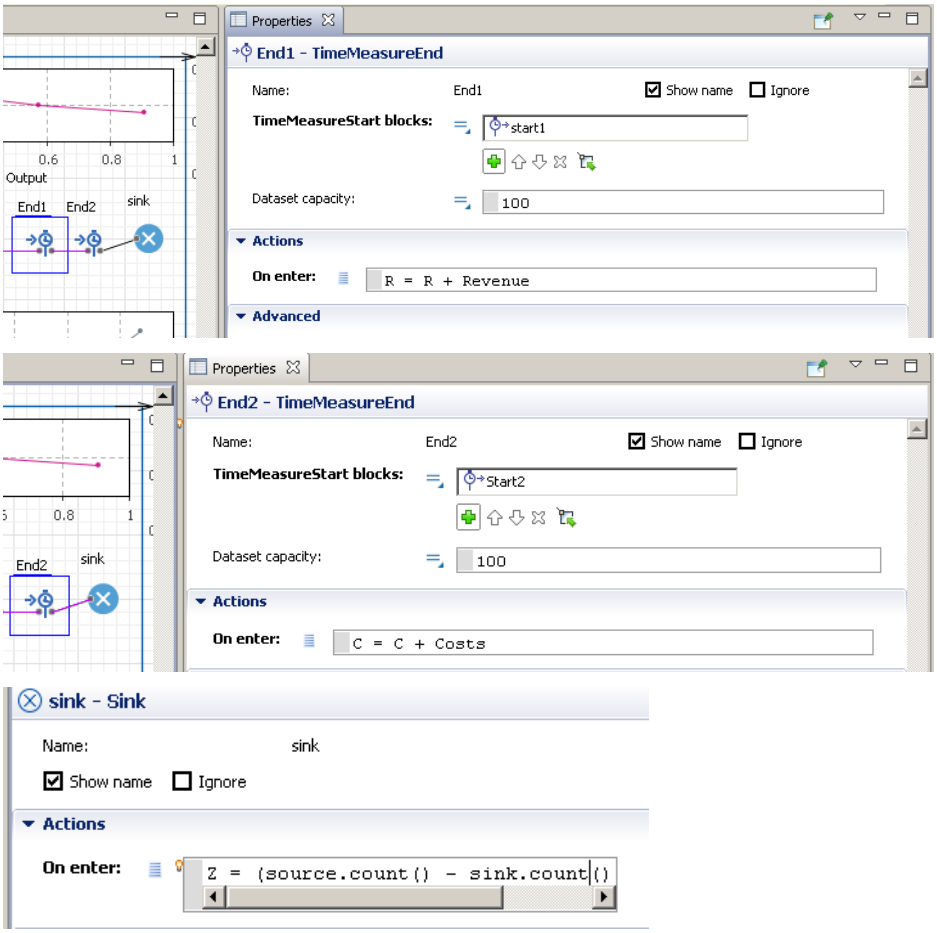


Fig. 13. Variable update for financial performance analysis

Probably this way of update is not the most elegant but it is quite easy to understand. The AnyLogic functionality offers significantly broader options for defining and returning values, events, functions, and states using *discrete event charts, agents, and system dynamics*.

2.7.2 Capacity utilization

In the standard delay function, the statistics of average capacity utilization can be retrieved using the function *delay.statsUtilization.mean()*. We select the bar chart and create a diagram for capacity utilization analysis (Fig. 14).

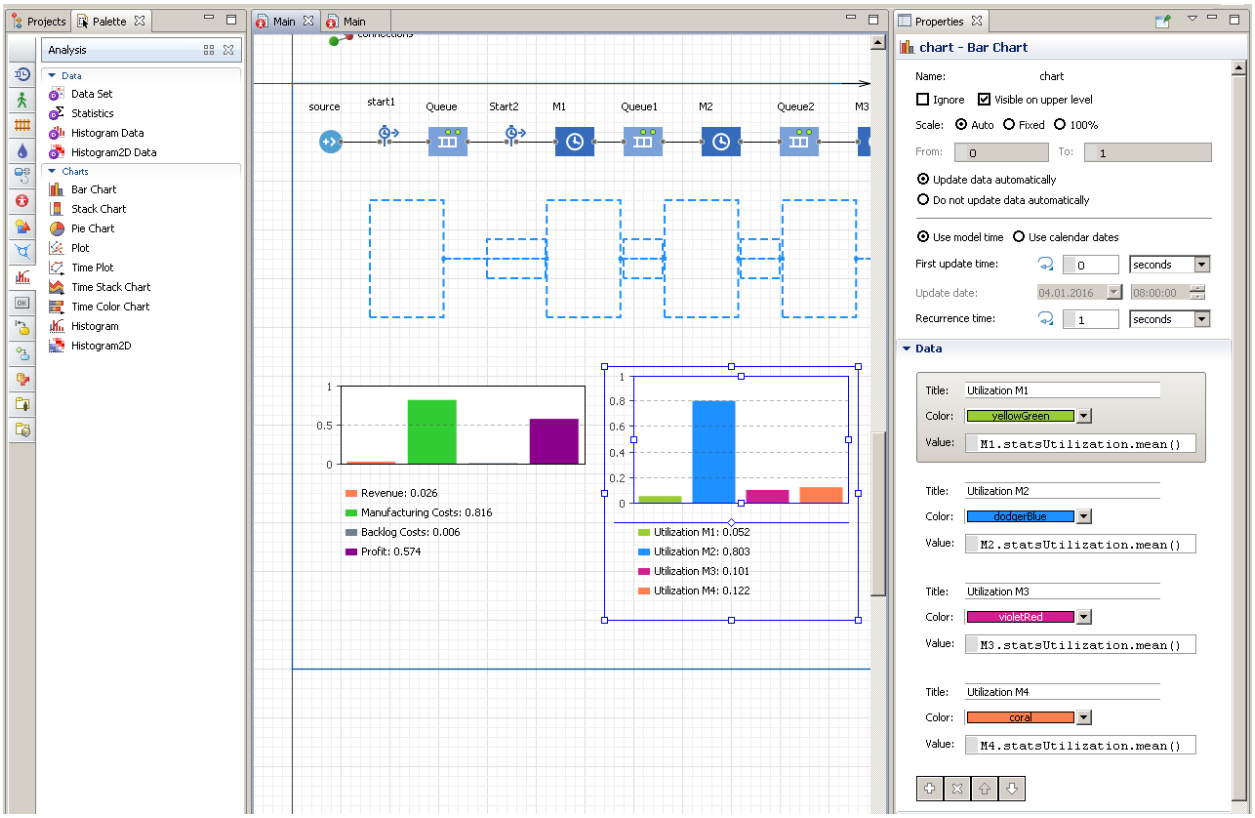


Fig. 14. Data definition for capacity utilization analysis

In order to analyse the capacity utilization in dynamics, we create a time plot as shown in Fig. 15.

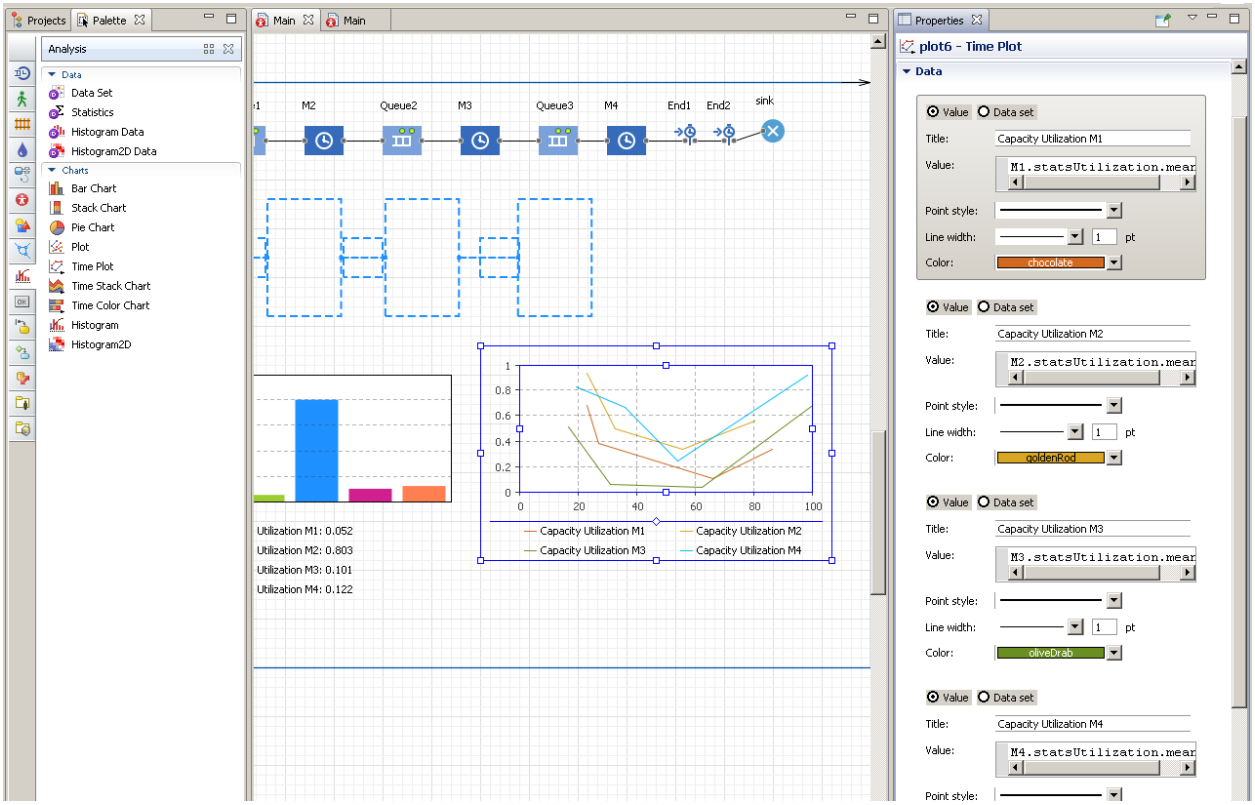


Fig. 15. Value definition for capacity utilization analysis in time

In order to estimate the dependence of the capacity utilization on the number of the waiting orders, we create a plot as shown in Fig. 16. In the standard queue function, the statistics of average queue length can be retrieved using the function *queue.statsSize.mean()*.

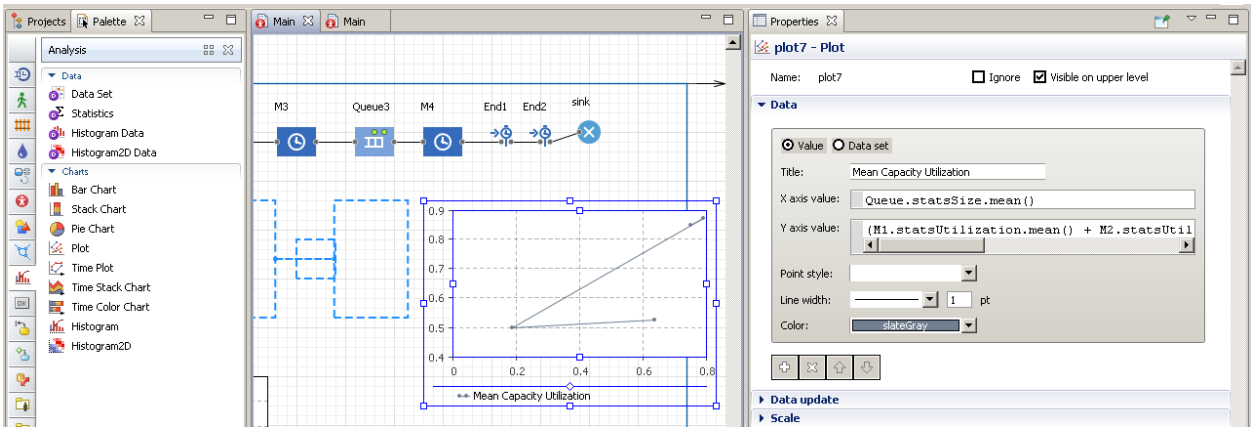


Fig. 16. Value definition for mean capacity utilization analysis in dependence on average queue length in time

2.7.3 Lead time and flow time

In the process flow diagram, we included four points to measure the times. Recall that the point "start1" measures the time of customer order arrival, point "Start2" measures the time of customer order dispatching at the manufacturing line, and points "End1" and "End2" measure the time of order completion. The time period "End1 – start1" is called *lead time* and time period "End2 – Start2" is called *flow time*. *Lead time* roughly equals *flow time* + *waiting time*. If an order passes the End1 point, its lead time is written in the End1 dataset. Similar, if an order passes the End2 point, its flow time is written in the End2 dataset. So the task of collecting statistics on lead and flow times is now reduced to representing End1 and End 2 datasets in the diagrams.

Note: in order to use dataset statistics, you need to create a dataset. Just drag the "Data Set" (in the "Analysis" – "Data" item to the working area.

We select a time plot and create the flow time diagram as shown in Fig. 17.

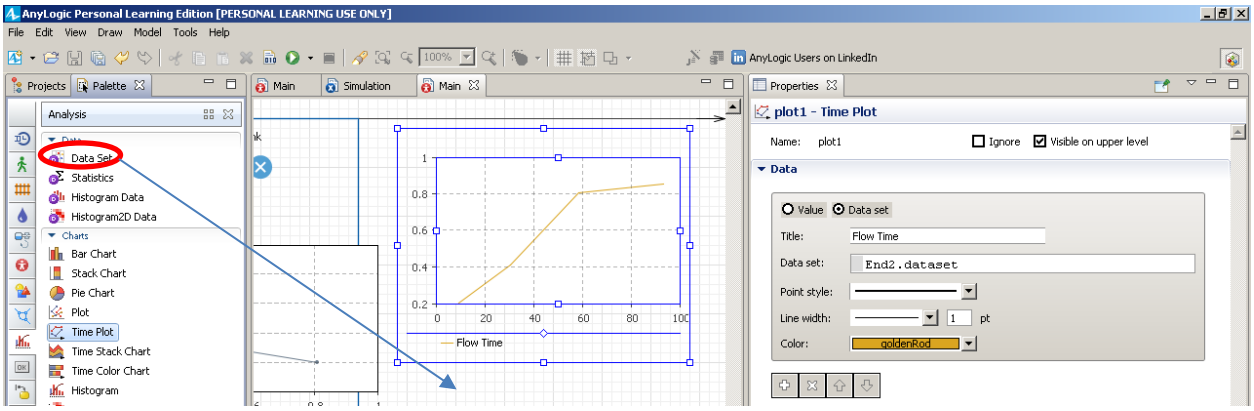


Fig. 17. Flow time analysis in time

Similar, lead time analysis diagram is created for End1 dataset. To analyse the flow time distribution, a histogram can be created as shown in Fig. 18.

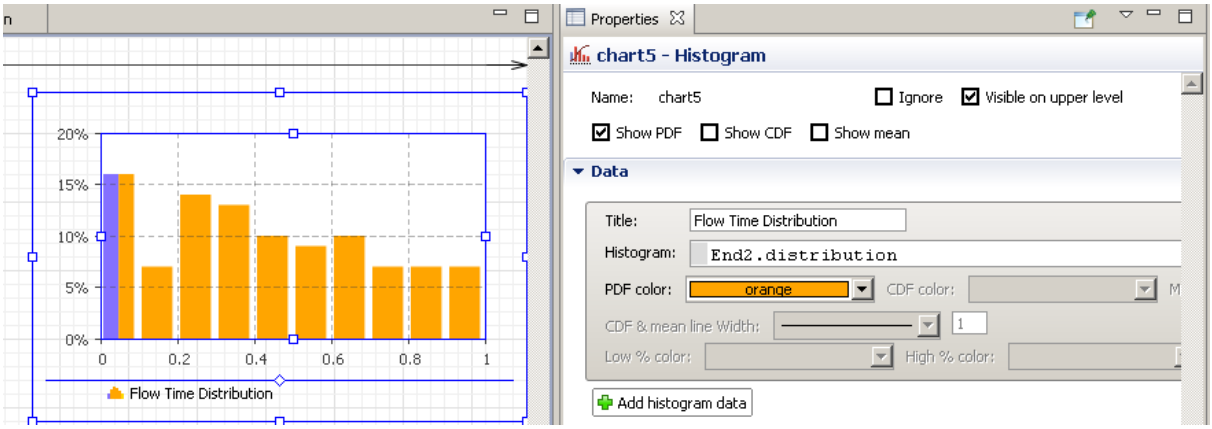


Fig. 18. Flow time distribution analysis in time

Similar, lead time distribution analysis diagram can be created for End1 dataset.

2.7.4 Backlog and queue analysis

First, we can analyse waiting orders and times. For analysis of waiting orders, we create a bar chart (Fig. 19).

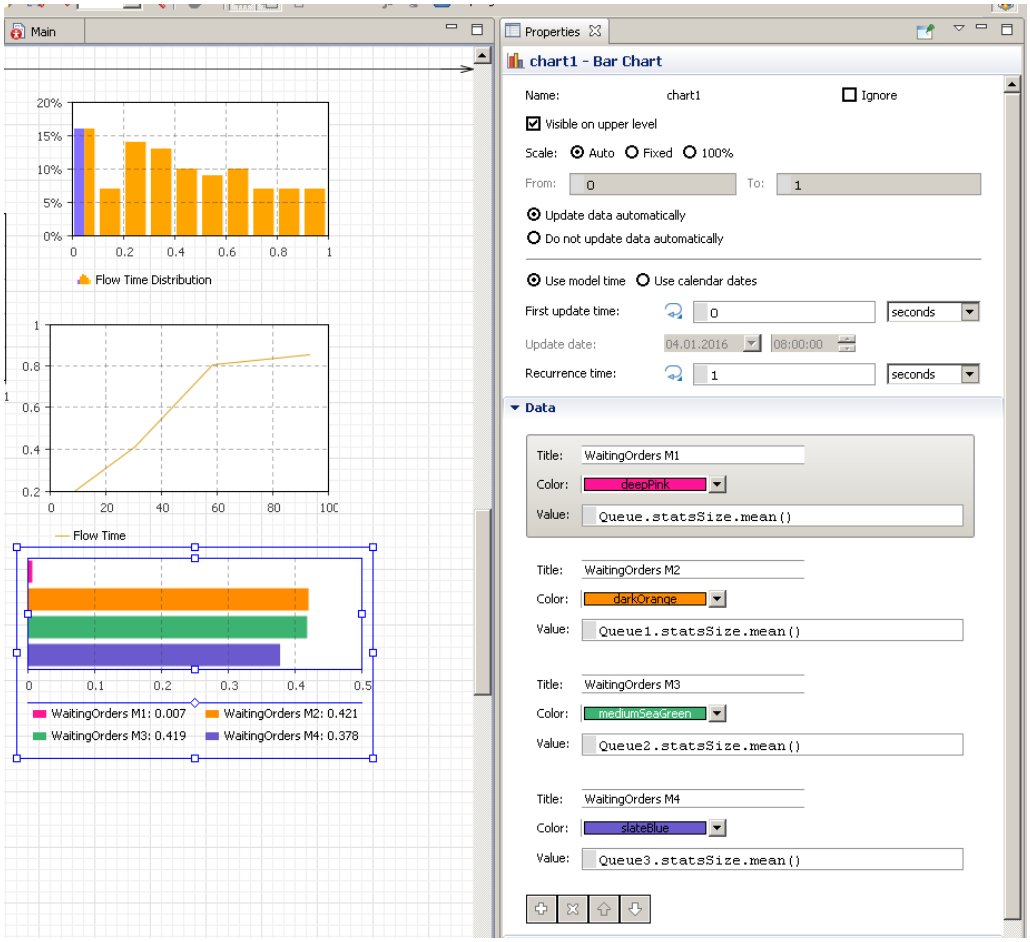


Fig. 19. Waiting orders analysis

Of course, we can also create a time plot (Fig. 20).

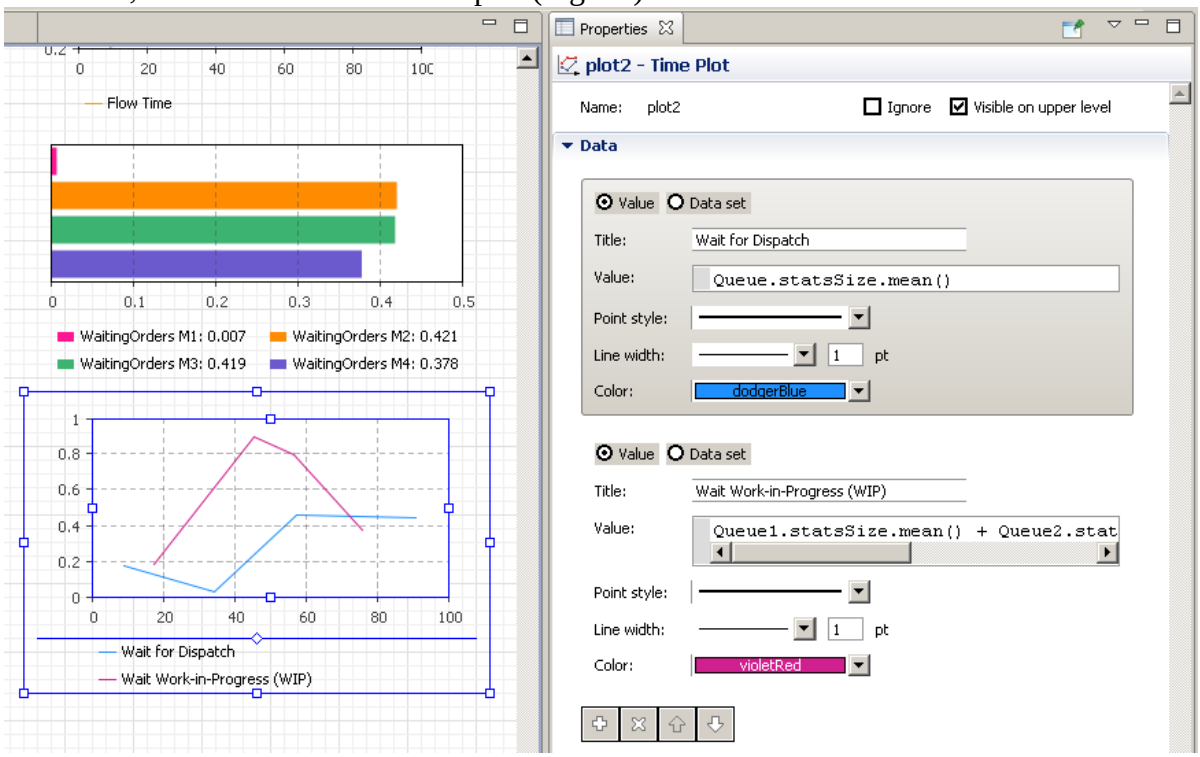


Fig. 20. Waiting orders analysis in time

To compute the backlog, we use the function (*source.count() - sink.count()*) (Fig. 21).

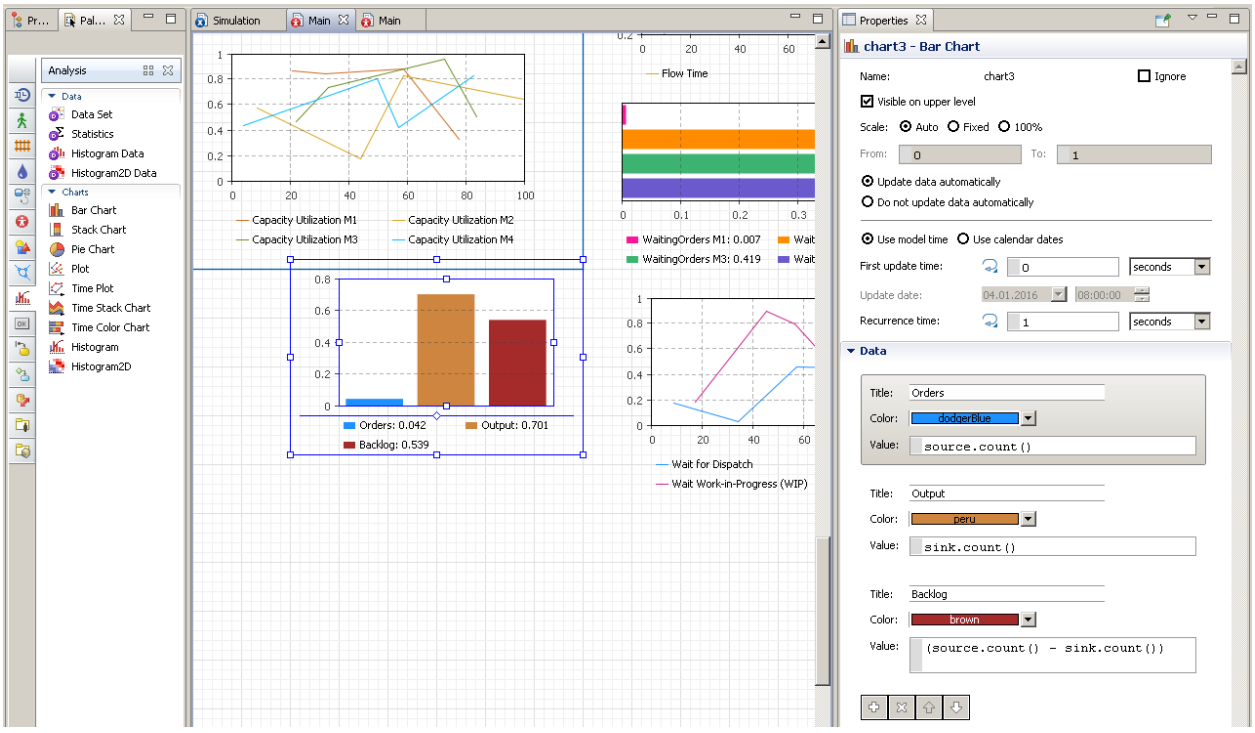


Fig. 21. Backlog analysis

Now we created the KPI cockpit (Fig. 22) and can start experimental part.

Fig. 22. KPI dashboard for process dynamics analysis and decision-making

Note: to enable statistics representation in the experiment view, the option "Force statistics collection" in each respective block of the process flow diagram needs to be enabled (Fig. 23).

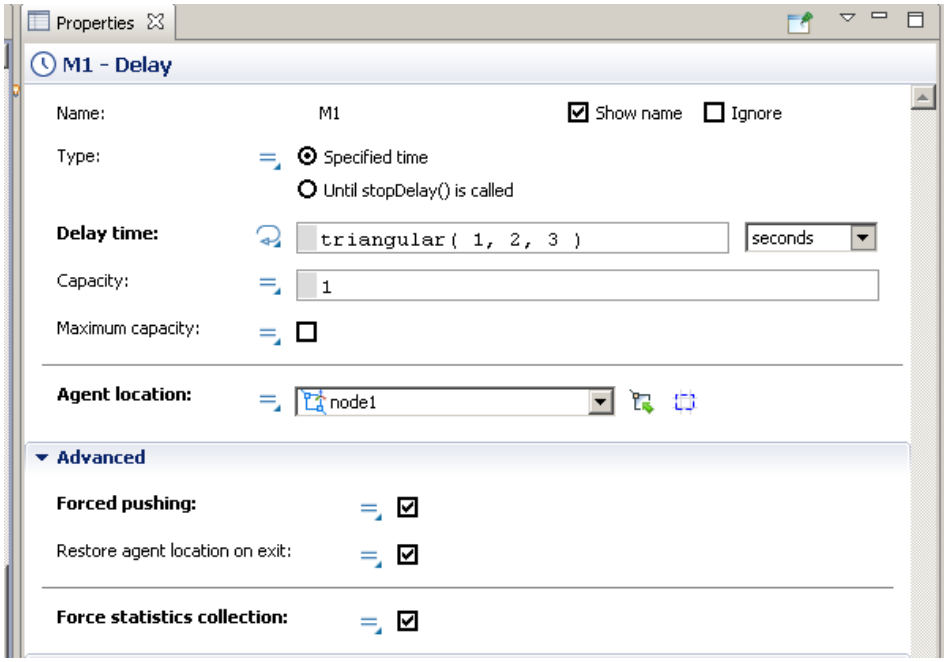


Fig. 23. Enabling statistics representation in the experiment view

3. Experiments and managerial insights

The following setups hold true for all experiments:

- Each experiment will contain 100 seconds as 100 days of real life
- Queuing rule is FIFO
- Queue capacities are maximum

3.1 Experiment 1

For the first experiment, we setup the following data:

Arrivals defined by rate: `1 per second`

Agents per arrival: `uniform_discr(3, 7)`

Force pushing - `TRUE`

Maximum number of arrivals: `100`

Delay times:

M1	M2	M3	M4
`triangular(1,2,3)`	`triangular(0.5,1,1.5)`	`triangular(2,3,4)`	`triangular(0.5,1,1.5)`

The simulation results are depicted in Fig. 24.

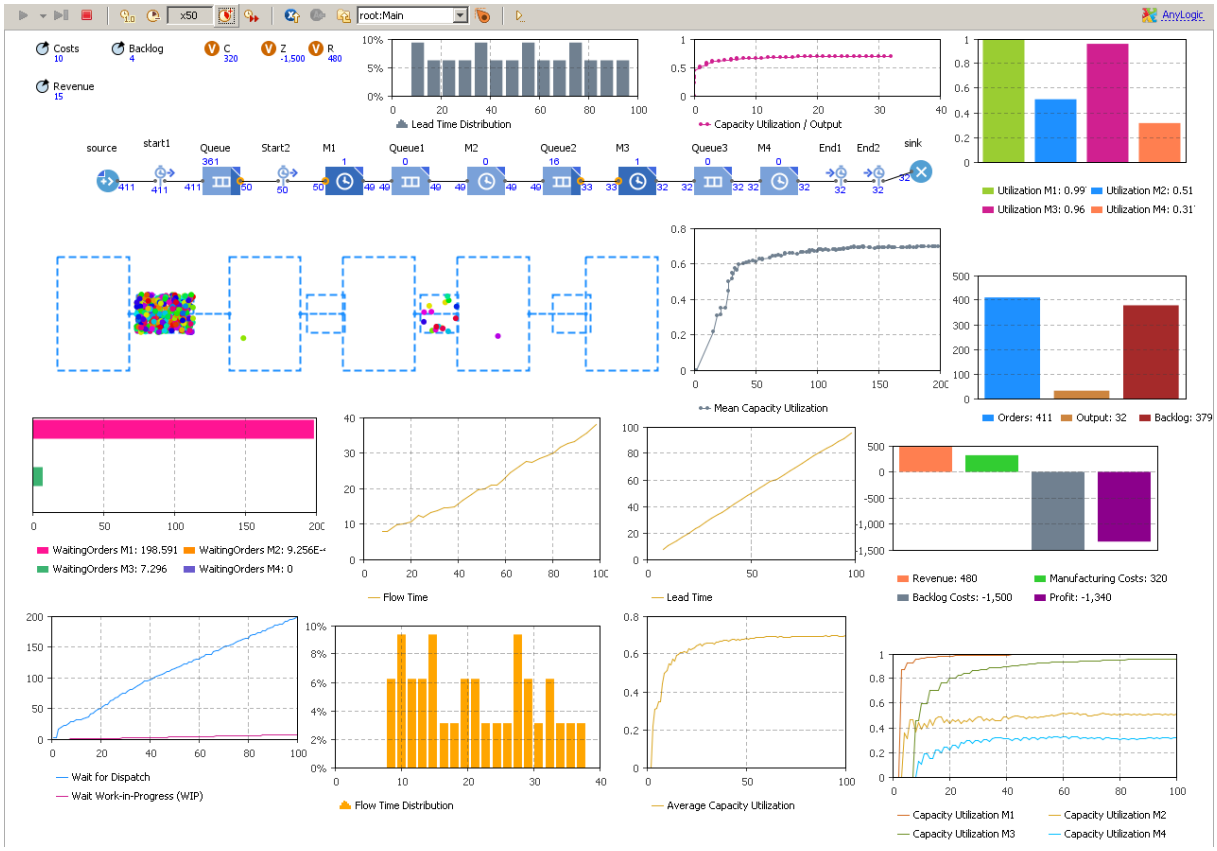
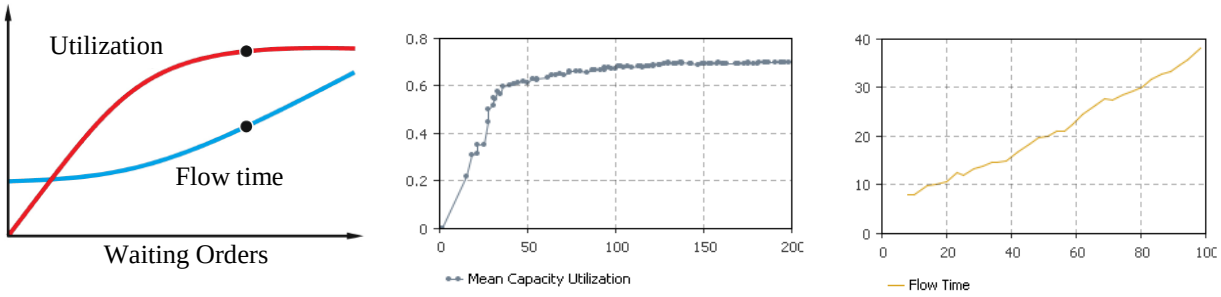


Fig. 24. Simulation results *Experiment 1*

We can observe that 411 orders came from customers while we could fulfill 32 orders only. Our backlog at the end of planning period (100 days) is 379 orders. This result provides us with a revenue of $480 and losses of $-1,340. We are running in negative result subject to high penalties

for non-fulfilled customer orders. Our service level (measured as Output/Input) is 32 / 411 = 7.8% that is very low.

Average capacity utilization is unequally distributed among the machines and it is 99% at M1, 51% at M2, 96% at M3 and 31% at M4. The average capacity utilization increases rapidly at the beginning and remains steadily at about 75% in the second half of the planning period. It can also be observed that mean capacity utilization depends on the number of waiting orders (see diagram "Mean Capacity Utilization"). The same holds true for the flow and lead times. These results are in line with theoretical concepts of waiting line theory and Little's Law (see also Chapter 9 "Factory Planning and Process Design" of the textbook). In Fig. 25, these dependencies are depicted.



(a) Theory (b) Experimental results

Fig. 25. Theoretical and simulation results *Experiment 1*

From Fig. 24, it can be further observed that the incoming flow in the form of customer orders is not balanced with the line capacity. This is clearly indicated by very long queue at M1. In the process, we also can observe unbalanced capacities among different machines. This is indicated by different capacity utilization at the machines as well as by the queue of waiting WIP (work-in-progress) orders at M3.

How can we improve?

Based on the analysis of the first experiment, we need to strive balancing incoming flow and the capacity to increase output in the most profitable manner subject to the interests of all the players. At this stage, a meeting with participation of sales and manufacturing managers is needed. The sales manager needs to consider capacity availability prior to accepting the orders. Manufacturing needs to improve the output and the workload in the line. So we obviously also need a work balancer who will balance the incoming flow and the capacity.

3.2 Experiment 2

Based on the analysis of the first experiment, manufacturing manager increased capacity at machine M1 by adding an additional worker (so now this machine can process two orders at a time). At M3, a worker with a higher qualification replaced the previous worker who changed as second worker to M1. This allows reducing processing time at M3 to [1; 1.5; 2] seconds. However such actions increase the manufacturing costs for one order from $10 to $11.

The sales manager used the theory on assembly line balancing (§9.6 of the textbook) and estimated possible output based on average takt time at machines of 1.3 days. For the period of 100 days, this corresponds to 77 orders. So the decision was to confirm 1-2 orders each two days.

For the second experiment, we setup the following data:

- Arrivals defined by interarrival time: `each 2 seconds`
- Agents per arrival: `uniform_discr(1, 2)`
- Force pushing - `TRUE`

- Maximum number of arrivals: `50`
- Delay times:

M1	M2	M3	M4
`triangular(1,2,3)` `capacity 2`	`triangular(0.5,1,1.5)` `capacity 1`	`triangu-lar(1,1.5,2)` `capacity 1`	`triangular(0.5,1,1.5)` `capacity 1`

The simulation results are depicted in Fig. 26.

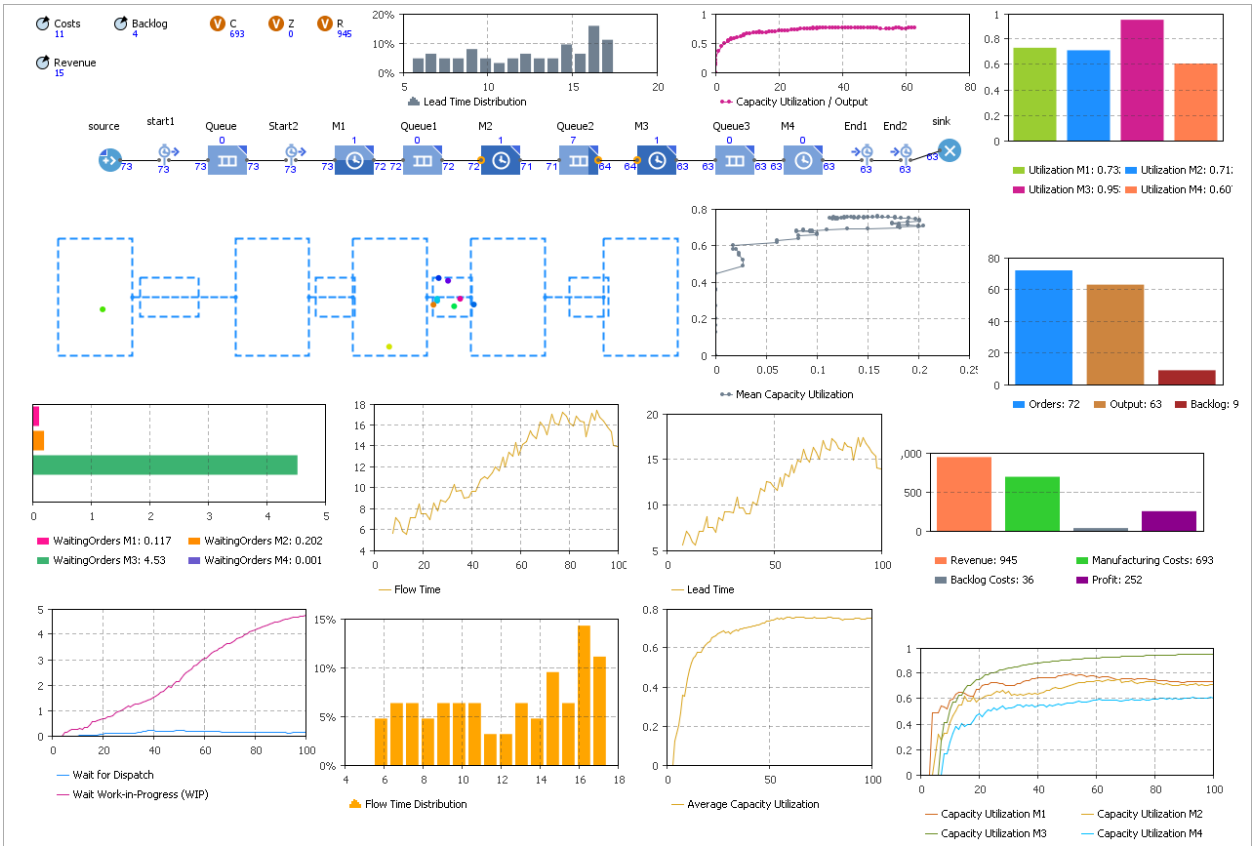


Fig. 26. Simulation results *Experiment 2*

In Table 1, the KPIs for financial and operational performance are compared.

Table 1 KPIs for financial and operational performance

KPI	Experiment 1	Experiment 2	Change
Revenue, $	**480**	**945**	**+96%**
Manufacturing Costs, $	320	693	+116%
Backlog Costs, $	1,500	36	-97.6%
Total Costs, $	**1,820**	**729**	**-149%**
Profit, $	**-1,340**	**252**	+$1,592
Customer Orders	411	72	-82%
Output (orders)	32	63	+99%
Backlog (orders)	369	9	-97.5%
Service Level, %	**7.1**	**87.5**	**+77.4%**
Unit Costs, $	**59.9**	**11.6**	**-80.6%**
Unit Profit, $	**-41.9**	**4.0**	**+$45.9**
Mean Capacity Utilization,% : **Average**	**69.25**	**61.25**	**-8%**
M1	0.99	0.73	
M2	0.51	0.71	
M3	0.96	0.95	
M4	0.31	0.6	
Average Waiting Customer Orders	198.6	0.1	
Average WIP **Total**	**7.4**	**4.7**	**-36.5%**
M2	0	0.2	
M3	7.3	4.5	
M4	0	0	
Lead Time, days			
Average	**51.5**	**12.5**	**-39 days**
Min	7.7	5.5	
Max	95.1	17.4	
Deviation	26.1	3.7	
Flow Time, days			
Average	**21.2**	**12.3**	**-8.9 days**
Min	7.7	5.5	
Max	37.9	17.4	
Deviation	9.3	3.7	

Hint: to see the statistical values (average, min, max, etc.) just click on the process flow diagram element in the experiment view (Fig. 27).

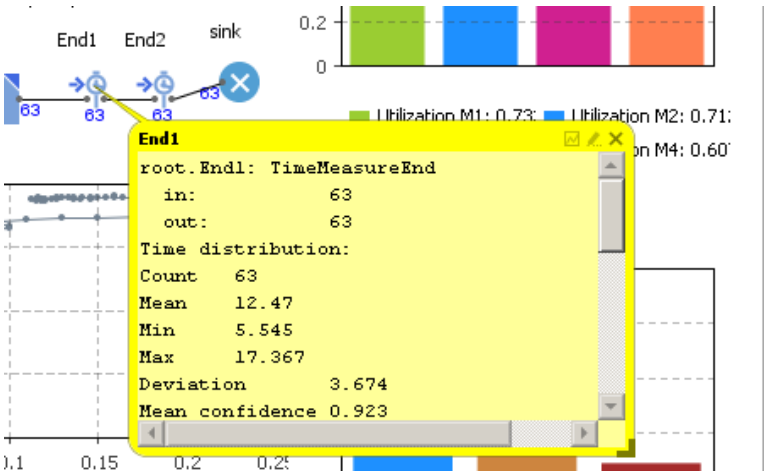


Fig. 27. Statistical results for *Experiment 2*

Let us compare the results and analyse whether our improvement suggestions would allow improving the workload balancing. First, improvement in all financial KPIs can be observed. We achieved a cost reduction by $991 or 149%, profit growth from -$1.340 to +$252, revenue increase by 96%, 99% increase in the output, backlog reduction by 97.5% and service level increase from 7.1% to 87.5%. Unit costs was reduced by 80.6% and unit profit increased by $45.9.
On the operational site, it became possible to reduce average waiting customer orders from 198 orders to 0.1 orders as well as average WIP was reduced of 36.5%. We could reduce average lead time from 51.5 days to 12.5 days as well as average flow time from 21.2 days to 12.3 days. Average capacity utilization was reduced from 69.25% to 61.25% that confirms the well-known dilemma of lead time and capacity utilization.

In general, it can be concluded that we improved significantly in almost all KPIs both on the customer and manufacturing sites. However, we still can observe some problems such as quite low average capacity utilization and unequally distributed capacity utilization of different machines. Note that M3 is still a *bottleneck*. In addition, our sales department is not satisfied with the output since the potential demand is about 90 units. So the workload balancer has now the task to simulate the impact of following changes:

- Change from *push to pull* by utilizing one-piece flow, CONWIP, and Heijunka dispatching principle
- *Standardization* of manufacturing technology to reduce the processing time variability at different machines
- Better *order management* at the sales site to reduce the variability of incoming orders.

3.3 Experiment 3

For the first experiment, we setup the following data:

- Arrivals defined by interarrival time: `each 2.5 second`
- Agents per arrival: `uniform_discr(2, 3)`
- Force pushing - `False`
- Maximum number of arrivals: `40`
- Delay times:

M1	M2	M3	M4
`triangular(0.5,1,1.5)`	`triangular(0.5,1,1.5)`	`triangular(0.5,1,1.5)`	`triangular(0.5,1,1.5)`

The simulation results are depicted in Fig. 28.

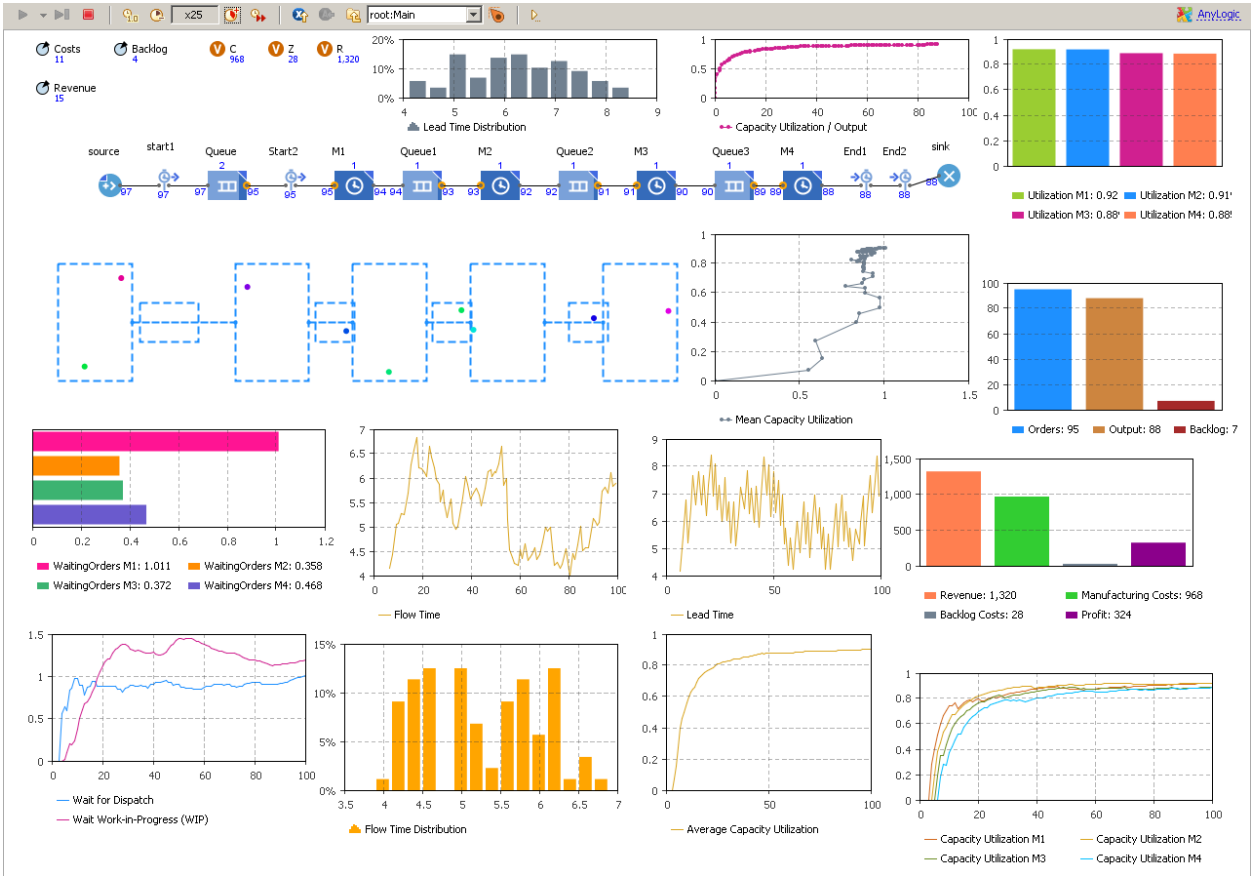


Fig. 28. Simulation results *Experiment 3*

In Table 2, the KPIs for financial and operational performance are compared with experiments 1 and 2.

Let us compare the results and analyse whether our improvement suggestions would allow improving the workload balancing. First, improvement almost in all financial KPIs can be observed (except for unit profit). We achieved total profit growth by 28%, revenue increase by 40%, 40% increase in the output, backlog reduction by 22.3% and service level increase by 4.8%. Unit costs was reduced by 2.6%.

On the operational site, it became possible to reduce average WIP by 75%. We could reduce the average lead time from 12.5 days to 6.2 days as well as the average flow time from 12.3 days to 5.2 days. The average capacity utilization was increased from 61.25% to 89.8%. However, average waiting customer orders increased from 0.1 orders to 1 order.

In general, it can be concluded that we improved significantly in almost all KPIs both on the customer and manufacturing sites. Further improvements in this setting can be seen in a reduction in lead and flow time variability as well as better capacity utilization. However we also should take into account that some idle capacity can potentially increase the company flexibility regarding demand changes or in regard to special customer orders or in light of unpredictable machine breakdowns or capacity disruptions.

Table 2 KPIs for financial and operational performance

KPI	Experiment 1	Experiment 2	Experiment 3	Change (compared Exr. 3 and Exp. 2)
Revenue, $	**480**	**945**	**1,320**	**+40%**
Manufacturing Costs, $	320	693	968	+40%
Backlog Costs, $	1,500	36	28	-22.3%
Total Costs, $	**1,820**	**729**	**996**	**+36.6%**
Total Profit, $	**-1,340**	**252**	324	**+28.6%**
Customer Orders	411	72	95	+32%
Output (orders)	32	63	88	+40%
Backlog (orders)	369	9	7	-22.3%
Service Level, %	**7.1**	**87.5**	**92.3**	**+4.8%**
Unit Costs, $	**59.9**	**11.6**	**11.3**	**-2.6%**
Unit Profit, $	**-41.9**	**4.0**	**3.7**	**-7.5%**
Mean Capacity Utilization,% **Average**	**69.25**	**61.25**	**89.8**	**+28.6%**
M1	99	73	92	
M2	51	71	91	
M3	96	95	88	
M4	31	60	88	
Average Waiting Customer Orders	198.6	0.1	1.0	
Average WIP: **Total**	**7.4**	**4.7**	**1.2**	**-75%**
M2	0	0.2	0.3	
M3	7.3	4.5	0.4	
M4	0	0	0.5	
Lead Time, days				
Average	**51.5**	**12.5**	**6.2**	**-50%**
Min	7.7	5.5	4.2	
Max	95.1	17.4	8.4	
Deviation	26.1	3.7	1.1	
Flow Time, days				
Average	**21.2**	**12.3**	**5.2**	**-57.7%**
Min	7.7	5.5	4.0	
Max	37.9	17.4	6.9	
Deviation	9.3	3.7	0.8	

4. Extensions

In this Section, we will learn how to:

- create interfaces for simulation and output presentation
- create interfaces for real-time experiment control and manual parameter change in order to observe the performance impact of different management decisions
- use parameters, variable, functions, events and datasets to create real-time experiment control interfaces and to collect statistics for KPI dashboard
- introduce the time limits for waiting in the queue (i.e., a customer order can wait 14 days at maximal and if the order is not dispatched after the expiration of this period, the customer withdraws this order which implies opportunity costs)
- use blocks “resources”, “seize”, “release”, and “service” in process flow diagrams
- create optimization experiments and use optimization for setting simulation experiments and resolving multi-criteria decision-making trade-offs.

4.1 How to introduce the time limits for waiting in the queue

In many practical settings, it becomes necessary to introduce the time limits for waiting in the queue (i.e., a customer order can wait 14 days at maximal and if the order is not dispatched after the expiration of this period, the customer withdraws this order which implies opportunity costs). This can be easily done in the “Queue” properties by enabling the option “Enable exit on timeout” setting a value in this field. Additionally, we need to connect the “outTimeoutExit” port of the queue to an external node, e.g., to a new sink node (Fig. 29).

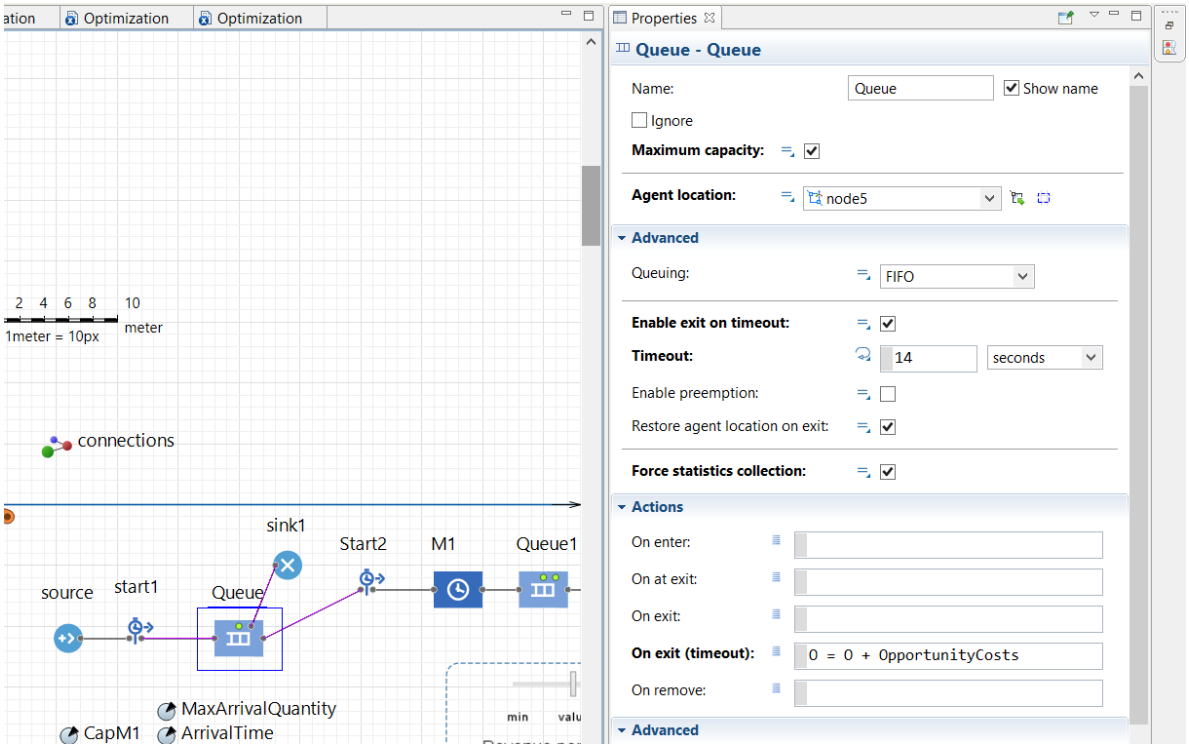


Fig. 29. Enabling “Exit on timeout” function

Now all orders that exceed the time limits for waiting in the queue will disappear. In other words, the customer withdraws this order which implies opportunity costs.

4.2 Interface creation for playing the game: views, sliders, parameters, variables, functions, and events

So far we changed the input data directly in the respective blocks of the model. Now we are going to create interfaces for more convenient playing the game. In addition we are going to learn some new methods for gathering statistics, more accurate and extensive costs analysis, and KPI computation.

4.2.1 Views

You already might notice that we considered two large areas: process logic and output results in the form of statistics and diagrams for performance analysis. In many cases, it might be convenient to separate these views to reduce presentation and result interpretation complexity. This can be easily done. In "Presentation", we select "View" and place the pointer in the desired working area. In our example, we create two views: "Animation & Output" and "Process Logic" (Fig. 30).

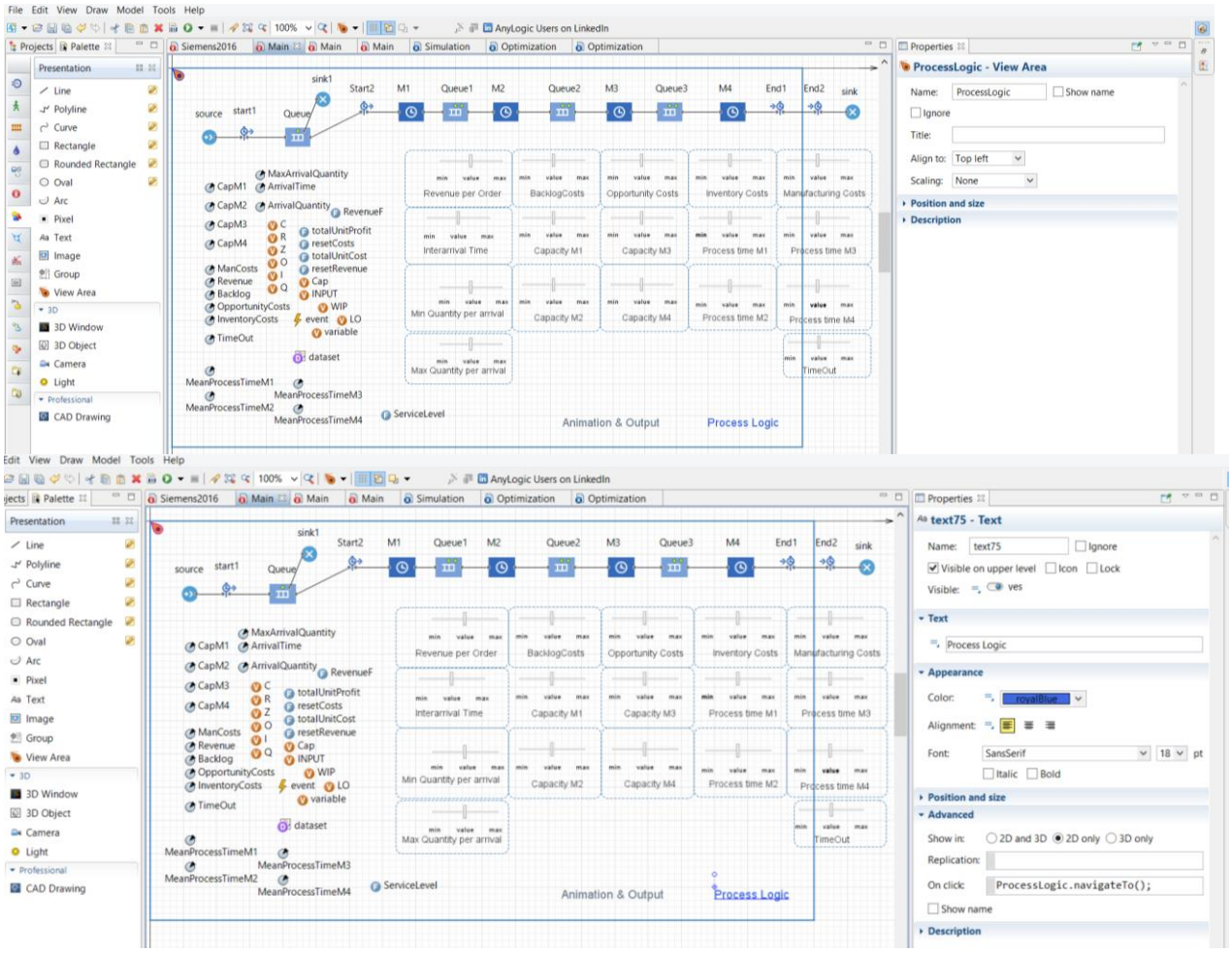


Fig. 30. Creation of different views in the model

To define the actions for switching between the views, we need to define the function "ViewName.navigateTo();" in the field "on click" of the respective text field. In the view "Process Logic" we will present the process flow diagram, define variable, parameters, function, events, and datasets, and create control charts for real-time parameter changes during the simulation experiment later on. In the "Animation & Output view", the process visualisation and KPI dashboard will be placed.

4.2.2 Parameters

We start with the creation of control charts for real-time parameter changes during the simulation experiment. For this purpose we first need to create new parameters (see Fig. 30) in addition to already created parameters “Costs”, “Revenue”, and “Backlog”. We introduce four parameters for machine capacities (CapM1 – CapM4), two parameters for minimal and maximal arrival quantity (MaxArrivalQuantity, ArrivalQuantity), arrival time (ArrivalTime), WIP inventory holding costs (InventoryCosts), opportunity costs (OpportunityCosts), exit on timeout duration (TimeOut), and the processing times at the machines (MeanProcessTimeM1 – MeanProcessTimeM2).

4.2.3 Variables

In the next step, variables need to be defined in order to update the respective values during the experiment. We define the following variables:
C – Manufacturing costs
R – Revenue
Z – Backlog costs
O – Opportunity costs
I – Inventory costs
WIP – WIP inventory
LO – Lost orders
Q – Output quantity
INPUT – the number of incoming orders
Cap – Average capacity utilization

We need to define the places in the flow diagram where the variables will be updated, e.g., the sink mode (Fig. 31).

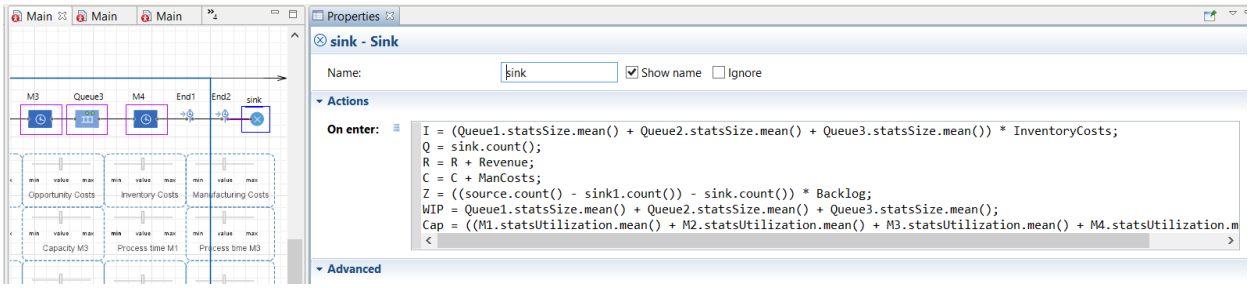


Fig. 31. Variable update definition

Variable LO is defined in the similar way in the “sink1”, and the variable INPUT is defined in the element “Source”.

4.2.4 Functions

In the next, we introduce some functions that are needed to return values for statistics and KPI dashboard (Fig. 32).

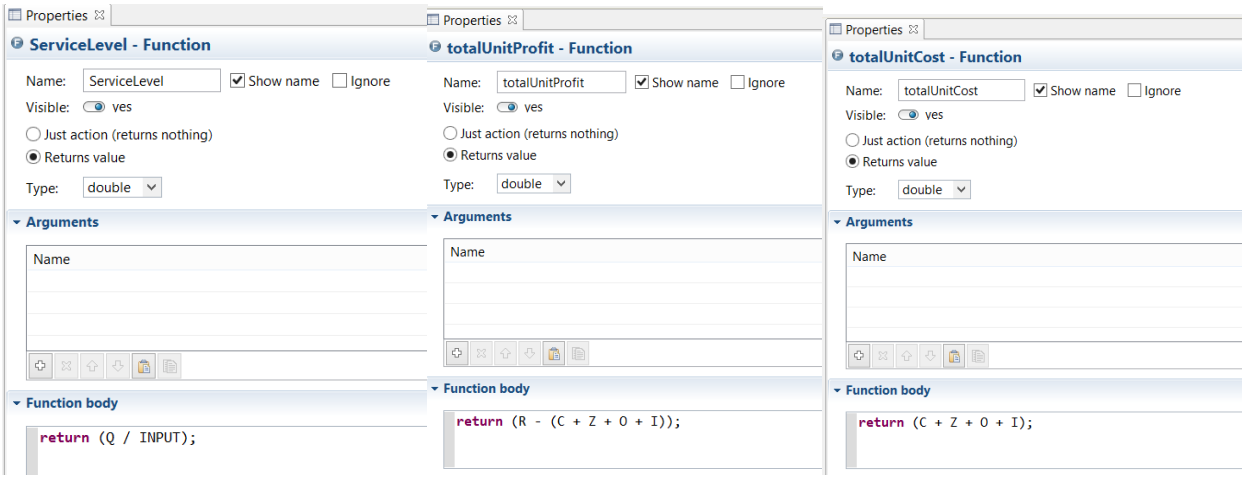


Fig. 32. Function definition

4.2.5 Events

Finally, we would like to introduce new method for statistics collection with the help of events. We need a diagram that would depict dynamics of incoming orders. For doing that, we need an event (named "event"), a dataset (named "Dataset"), and a variable (named "variable") (see Fig. 33). In Fig. 33, we define the rule for the event.

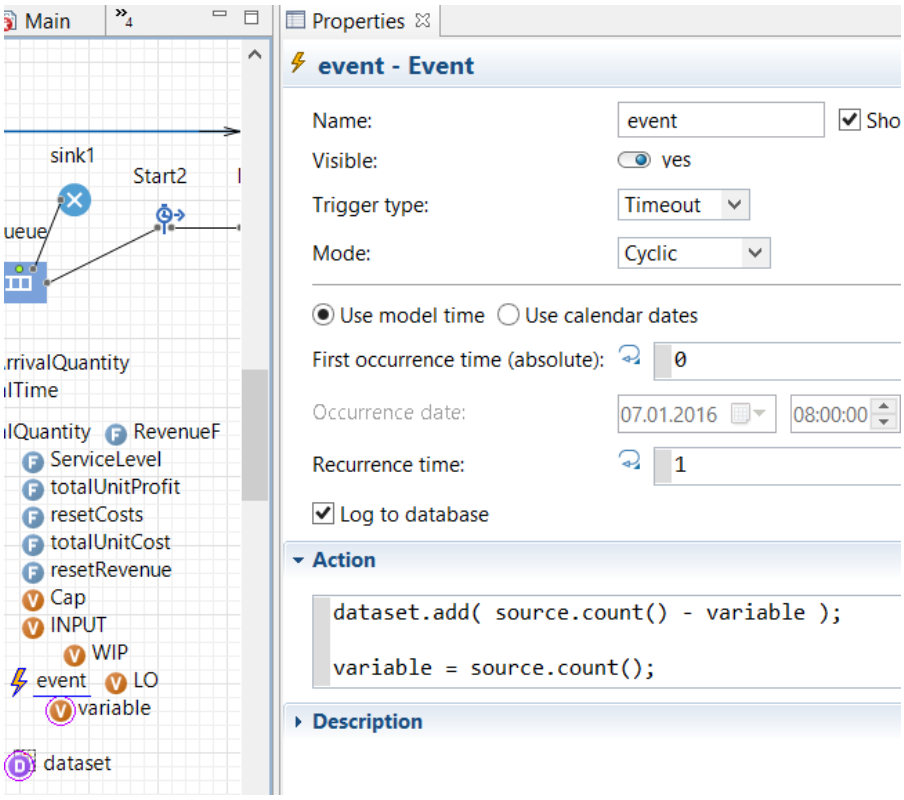


Fig. 33. Event definition

Since the event needs to be repeated (i.e., we want to know how many orders came each day or each week), we define this event as a "Cyclic" event. Setting a value in "Recurrence time" field, we define the cycle duration for updating the dataset. Now we can easily create a time plot for graphical presentation of incoming order dynamics (Fig. 34).

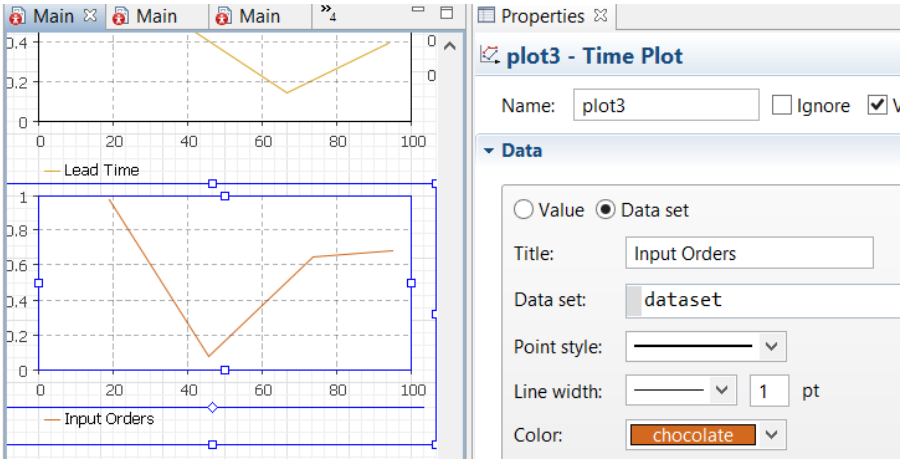


Fig. 34. Input order dynamics evaluation

4.2.6 Sliders

Sliders are needed for manual parameter change during the experiment. We define the sliders for all the parameters (Fig. 35).

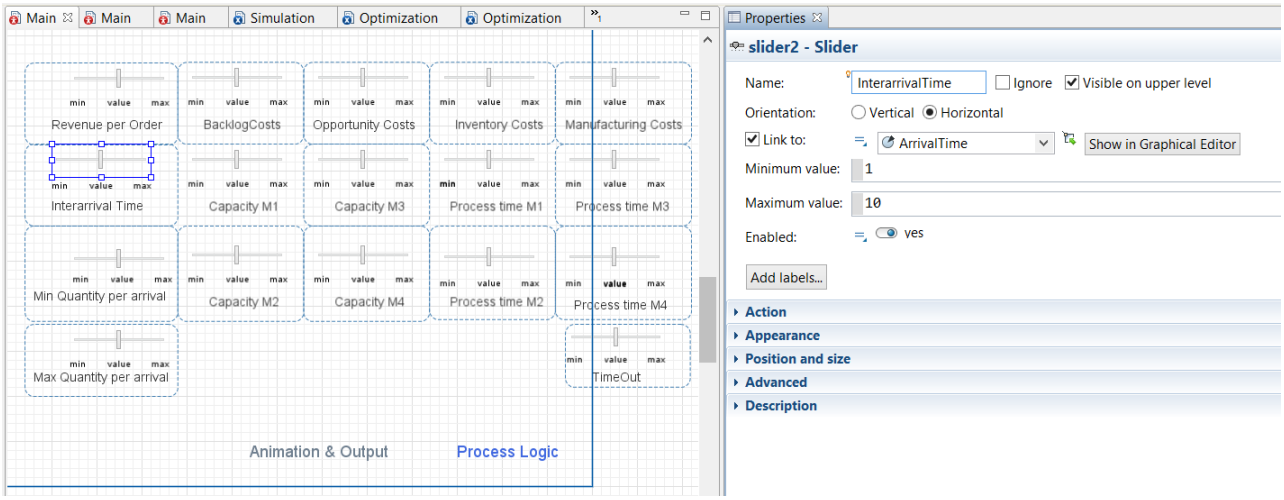


Fig. 35. Slider definition

In the properties, we need to link the sliders to the respective parameter, define its min and max values, and press “Add the labels” to add “min”, “value”, and “max” texts on the slider.

4.2.7 KPI dashboard creation

In the “Animation & Output” view, we now design the KPI dashboard. We use the already created diagrams and extend the presentation by some KPIs (Fig. 36).

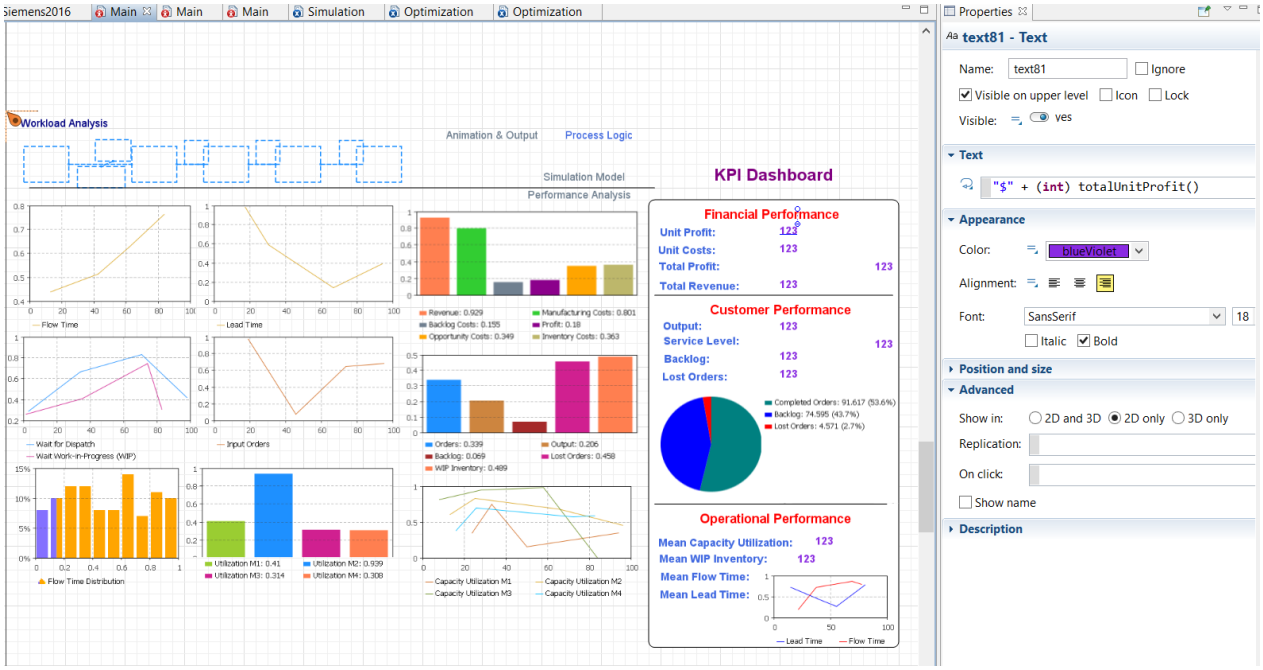


Fig. 36. KPI dashboard

We widely use functions and variables for defining the values of the KPI. An example for KPI "Unit Profit" is shown in Fig. 36.

4.2.8 Resources

In many practical settings, the order processing at machines requires some resources. It can be a worker (static resource) or a forklift (moving resource). In these cases, we need to assign resources to the machines with the help of blocks "Resource", "Seize", and "Release" in Process Flow Library (Fig. 37).

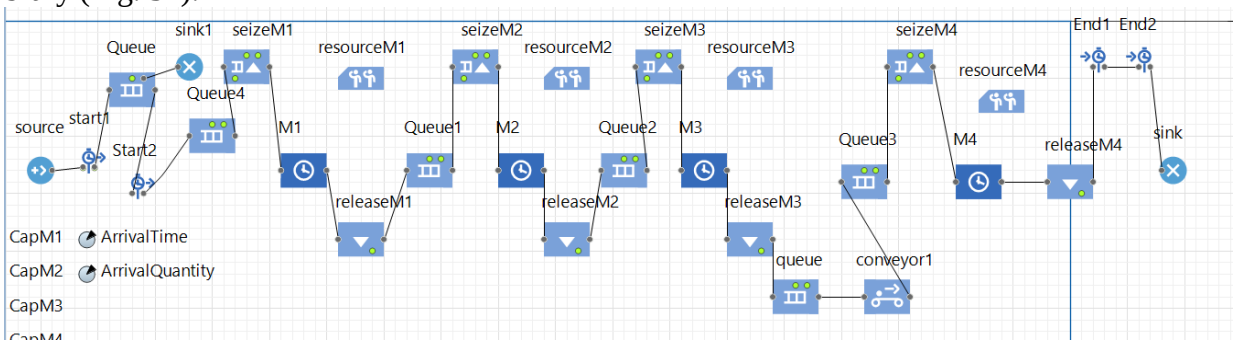


Fig. 37. Process presentation with the help of resource pool

A resource pool defines a set of resource units that can be seized and released by agents using Seize, Release, Assembler and Service flowchart blocks. Any resource unit can be either idle or busy. This object collects utilization statistics, which is continuous time statistics on the percent of busy units. Resource units-objects always collect their individual utilization statistics. If a resource does not need to do anything but execute a delay between seize and release, it can be replaced by the block "Service" that seizes a given number of resource units, delays the agent, and releases the seized units. It is equivalent to a sequence Seize, Delay, Release.

5. Playing the game

5.1 Game rules

Now we are ready to play the game. *The players* are:

- A division head who strives to maximize total profit of the division and unit profits
- A sales manager who strives to maximize revenue, to accept as many customer orders as possible and to complete the orders as quick as possible subject to lead-time
- A manufacturing manager who seeks for the lowest unit costs, high capacity utilization, and low flow time
- A workload balancer who needs to balance the objectives of the division head, the sales manager, and the manufacturing manager in order to achieve maximal possible profit under given constraints.

We play ten rounds. First two rounds are played jointly in order to understand the game rules and basic trade-offs. Other rounds are played in groups. In each round, the group wins that has the highest total profit. At the same time, in each round we will define a winner in each individual category: best manufacturing manager (KPI: lowest unit cost and lowest flow time), best workload balancer (KPI: highest service level), best sales manager (KPI: highest output and lowest lead time), and best division head (KPI: highest total profit). The students are divided into groups each of which contains four participants.

The following setups hold true for all the rounds:

- Each experiment will contain 100 seconds as 100 days of real life
- Queuing rule is FIFO
- Queue capacities are maximum
- Increase in capacity is subject to additional manufacturing costs of $1 for each additional capacity unit
- Capacities can be increased at each machine to 1 unit only (from 1 unit to 2 units). An increase to 3 units means the we use outsourcing as the third capacity unit which implies additional costs of $2 for the third capacity unit

5.2 Round #1

For the first round, the following input data can be suggested:

- Arrivals defined by interarrival time: `2 seconds`
- Agents per arrival: `uniform_discr(2, 5)`
- Force pushing - `True`
- Maximum number of arrivals: `50`
- Timeout: `14 seconds`
- Delay times:

M1	M2	M3	M4
`triangular(0.7,1,1.3)` capacity 1	`triangular(0.7,1,1.3)` capacity 1	`triangular(1.4,2,2.6)` capacity 1	`triangular(0.7,1,1.3)` capacity 1

- Revenue and Costs, $:

Revenue	Manufacturing	WIP inventory	Lost Sales	Backlog
30	10	5	4	4

The simulation results over 100 seconds are shown in Fig. 38.

Fig. 38. Results of the round #1

In the given setting, we achieved total profit of $335, revenue of $1,410, unit profit of $7, and unit costs of $22. We completed 47 orders, lost 70 orders, and have a backlog of 240 orders. This implies a very low service level of 27%.

On the operational site, mean WIP inventory comprises 17 orders, mean capacity utilization is 76%, average flow time (read in the process flow diagram at the End2 block) is 26.3 days, and average lead time is 36 days. We can observe from the visualization that the orders are waiting long at M1 and M3. The reasons are twofold. First, the incoming flow is too large as compared to the machine capacities. Second, the processing time at M3 is twice as high as this is the case at M1, M2 and M4. The M3 is therefore a *bottleneck*.

5.3 Round #2

The division head means that the results of round #1 and their analysis indicate the necessity to reduce the number of incoming orders and to increase the capacity of M3. The sales manager cannot believe that the reduction of the incoming flow (i.e., the number of accepted customer orders) will result in sales (i.e., output) increase. The manufacturing manager is also concerned about increase in total manufacturing costs. Nevertheless, the team agrees to test these two options in the following setting: in the period between [30 – 55] seconds, the maximal arrival quantity will be reduced from 5 to 2 orders, and in the period from [35-60] seconds the M3 capacity will be increased from 1 unit to 2 units which implies manufacturing costs of $11 for this period. The simulation results are presented in Fig. 39.

Fig. 39. Results of the round #2_1

It can be observed that we could improve service level (36.6% instead of 27%), reduce the backlog from 240 to 192 orders, and reduce the lost orders (57 instead of 70). We also improved the output (60 orders instead of 47), total profit ($700 instead of $335) and became better in regard to unit profit ($11 instead of $7) and unit costs ($18 instead of $22). At the same time, mean utilization capacity increased to 78.9%, average WIP inventory was reduced from 17.9 to 10.5 orders, average lead time was reduced from 36 days to 29 days, and average flow time was reduced from 26 days to 18.3 days.

In the diagram "Lost orders" we can observe that the reduction of maximum arrival quantity in the period between [30 – 55] seconds positively affected the number of the lost orders. In the period between [45 – 65] seconds, the function is flat that indicates that no order was lost. Here we can also observe a *time delay (lag)* between making a decision on parameter change and the performance impact of this change. In the diagrams "Flow time" and "Lead time", we also observe positive influence of the decisions to change the input rate and the M3 capacity. In Table 3, the KPIs of all managers in the rounds #1 and #2 are compared.

Table 3 KPI comparison

Manager	KPI	Round 1	Round 2_1
Division Head	Total Profit, $	335	700
Manufacturing Manager	Unit Costs, $	22	18
	Flow time, days	26	18.3
Sales Manager	Output, units	47	60
	Lead Time, days	36	29
Workload Balancer	Service Level,%	27	36.6

The results indicate that we are in the lucky situation where KPIs of all the managers have been improved. It is not difficult to make a decision for running new experiment fully on the basis of new maximum arrival quantity (2 orders) and new M3 capacity (two) with arrivals each two days. The simulation results are presented in Fig. 40.

Fig. 40. Results of the round #2_2

As expected, we could further improve all the KPIs. But the new question arises: what is the maximal possible profit? What parameters values should we use in order to achieve the maximum possible profit? In other words, what is the *optimal solution*? In order to answer this question, we will perform an optimization experiment in AnyLogic.

5.4 Round #2: Optimization experiment

In "Project –> New → Experiment", we select "Optimization" and create new optimization experiment for the data from round #2_2 (Fig. 41).

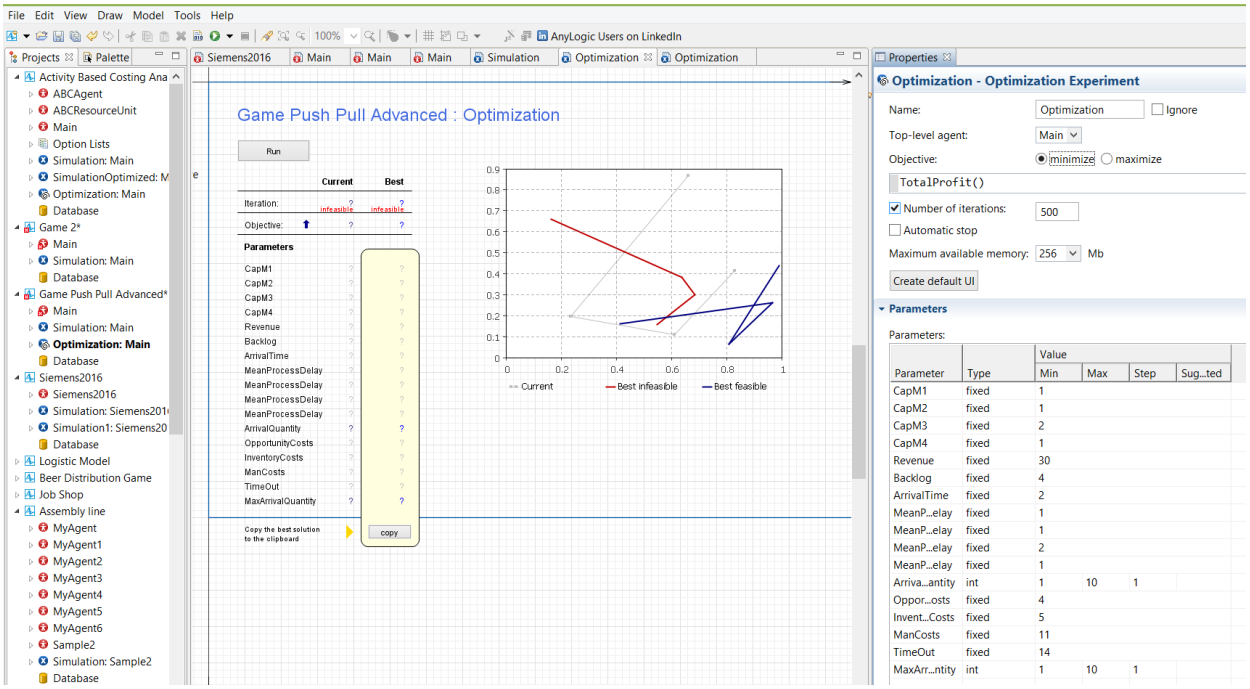


Fig. 41. Optimization experiment design for the data from the round #2_2

The list of parameters with the values in the "Optimization Experiment" is created automatically. In order to create the list of parameters and the graphical representation in the main plot, please click on "Create default UI". Then we need to define the objective function (maximize total profit) and the decision variables (i.e., the parameters the values of which will be varied in some range). In our example, we are going to vary minimal and maximal arrival quantities in the range from [1; 10] with the change step 1. The optimization results are presented in Fig. 42.

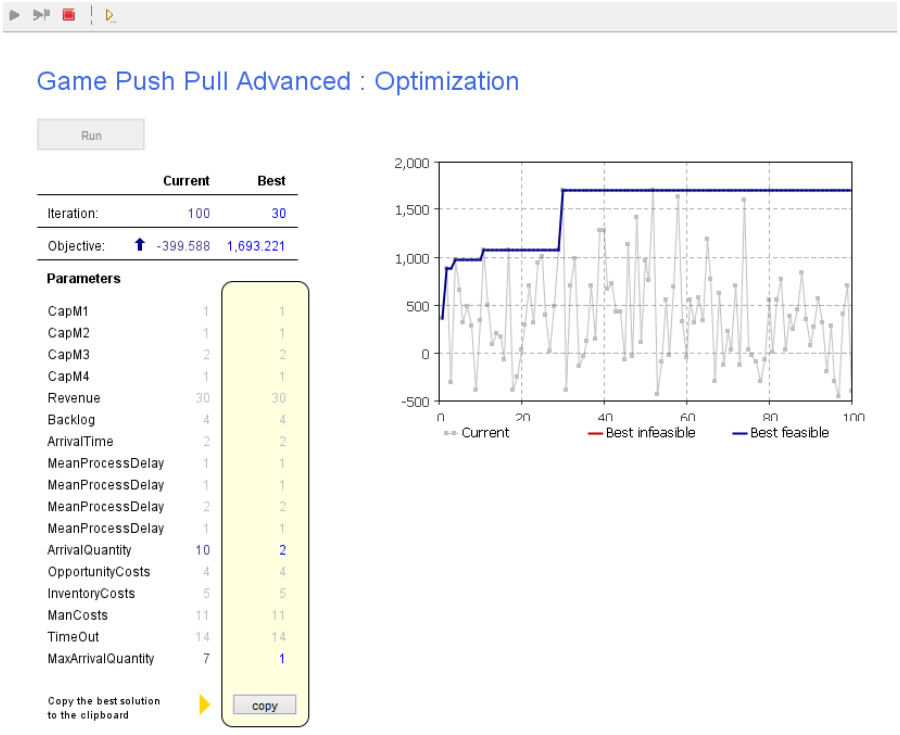


Fig. 42. Optimization results (1) for the data from the round #2_2

It can be observed from Fig. 42 that subject to the given data setting and the considered range of variable alteration, the recommendation is to accept between one and two customer orders each two days.

5.5 Round #2: Optimization-based simulation experiment: multiple objective decision making

Let us perform the simulation experiment on the basis of new maximum arrival quantity (3 orders) and new M3 capacity (two) with arrivals each two days: (see Fig. 43).

Fig. 43. Simulation results for the data from the round #2_Opt(1)

Both division head and sales manager are not satisfied with the results because of decrease in total profit and total sales and suggest increasing the order acceptance range to [1; 3] orders each two seconds. The results are shown in Fig. 44 and compared with the previous simulations #1, 2_1, 2_2, and 2_Opt(1).

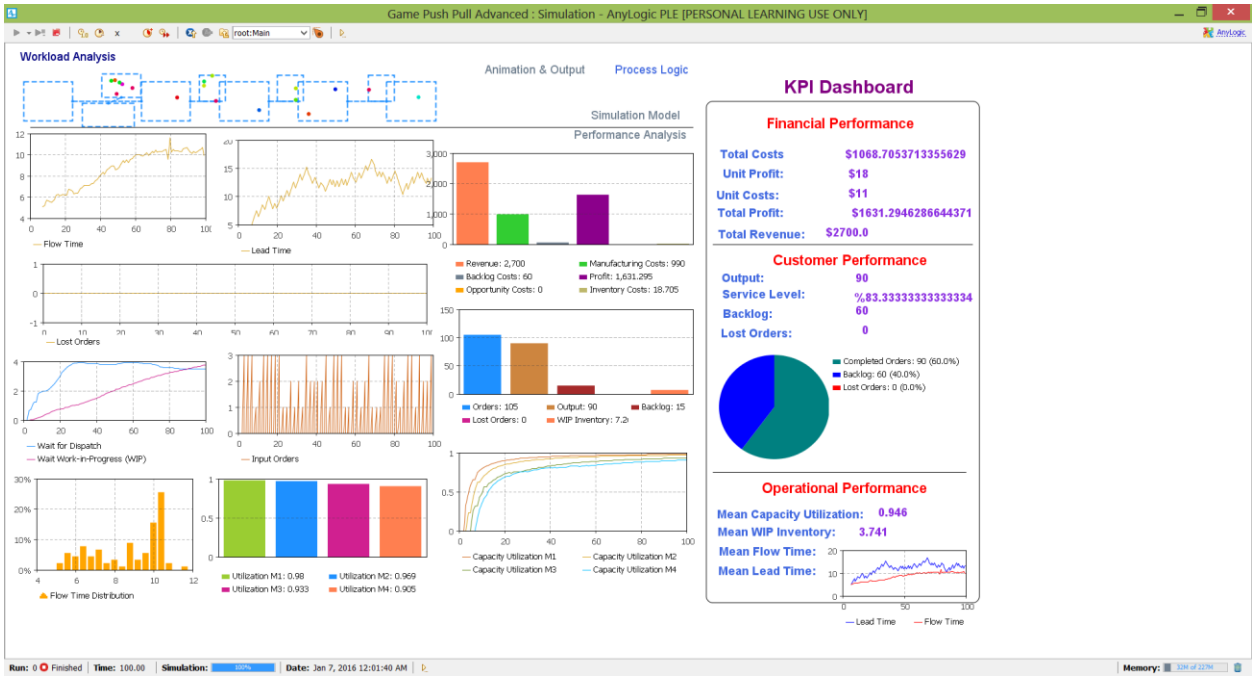


Fig. 44. Simulation results for the data from the round #2_Opt(2)

Table 4 KPIs for financial and operational performance

Manager	KPI	Round 1	Round 2_1	Round 2_2	Round 2_Opt(1)	Round 2_Opt(2)
Division Head	Total Profit, $	335	700	1,409	1,172	**1,631**
	Unit Profit, $	7	11	16	**18**	**18**
Manufacturing Manager	Unit Costs, $	22	18	13	**11**	**11**
	Flow Time, days	26	18.3	26	**5.2**	8.7
	Average WIP inventory	17	10.5	0.5	**0.2**	3.7
Sales Manager	Total Sales, $	1,410	1,800	2,520	1,890	**2,700**
	Output, units	47	60	84	63	**90**
	Lost Orders, units	70	57	22	24	**0**
	Lead Time, days	36	29	36	**5.9**	12.1
Workload Balancer	Service Level,%	27	36.6	65.6	**91.3**	83.3
	Average Capacity Utilization, %	76	78.9	87	65	**95**
	Backlog, units	240	192	88	**0**	60

Let us compare the results and make the final decision on at what parameters we will run our production for the next 100 days. First, it can be observed that the best KPI values can be achieved in the options #2_Opt(1) and #2_Opt(2). However, they are distributed among these two options. The best values for the KPIs of the division head and the sales manager are in the option #2_Opt(2).

On the contrary, the KPIs of the manufacturing manager and the workload balancer (except for average capacity utilization; however it can also be an advantage in regard to flexibility or capacity disruptions) are in the option #2_Opt(1). Here a compromise is needed and the nature of *multiple objective decision making* can be perceived better.

Having applied both simulation and optimization, advantages and disadvantages of simulation and optimization for decision-making in operations and supply chains can be perceived better. Where are these methods different and how can they enrich themselves mutually? Optimization provides us with the best possible values for some decision variables in regard to a selected objective function. However, the experiment is *static* in its nature, since we do not change the parameters during the experiment. Simulation can help to get a *dynamic* view on the system and allow observing the performance impact of parameter deviations at each point of time. It also allows the process visualization. These methods are not contradictory but should be used jointly to enrich the decision-making power and make better decisions.

5.6 Rounds #3 – #10 and game evaluation

In the following rounds, the teams get from the instructor different input data sets and compete against each other subject to total profitability. At the same time, the managers within each team compete against the managers in other teams on individual basis to become the best in regard to their individual KPI. Getting the input data, students have 5-10 minutes time to take decision on how to setup the parameters. During that time they may also use optimization experiments but not the simulation. Having setup the parameters, the teams start simulation experiment. During the experiment, they may pause the experiment as often as they want but total time for all groups need to be limited, say for 5-10 minutes for one round.

The evaluation of the game can be performed in different ways. One possible evaluation method is to evaluate each round independently and assign in each round some scores to each team, e.g., team or respective manager with the best result gets 10 points, and the team with the worst results will get 1 point. Another evaluation method is to combine the results from all the rounds and then compute average values for each KPI to determine the winner.

6. Further possible extensions and other games

The game described in §1-5 can be extended in the considered setting with the linear system or to a networked supply chain system, assembly line, multiple parallel machines, etc. The following extensions can be suggested but are not limited to:

- Two competing products with different priorities
- Limit the length of queues
- Include due dates for orders (or lead time limits)
- Include material availability check prior to customer order confirmation
- Parallel machines and multiple resources for each machine
- Working schedules with calendar
- Other process flow structures such as assembly line or/and batching
- Create a network (supply chain) and include flow synchronisation among different companies (e.g., as JIT system)
- Include intermediate storage (internal or external warehouses)
- Link incoming flow with demand forecasts computed as a result of multi-agent simulation
- Reducing revenue if waiting or lead times increase
- Include inventory of raw and final products
- Include different outsourcing options

- Include bottle-necks and additional constraints by incorporating supply and distribution parts to build an integrated supply chain
- Using Monte-Carlo simulation, compare runs, and sensitivity experiments (available in Professional License only)

Part II Capacity flexibility simulation

7. Business Simulation Game „Capacity flexibility simulation with AnyLogic“

7.1 Learning objectives

- To develop analytical and management skills on capacity demand planning
- To develop soft skills on coordinated team-based decision making
- To develop technical skills on creating discrete-event process simulation models and perform performance measurement analysis on example of AnyLogic multimethod simulation software

7.2 Problem statement

This chapter studies an operational challenge recently faced by a high-tech manufacturing company that is specialized on both rapid prototyping (ETO: engineer-to-order) and serial manufacturing (MTO: make-to-order) of electronic devices. The focus is on tactical-level capacity management which refers to the determination of in-house workforce production capacity in an optimal cost-efficient manner. The simulation model is used in order to reveal the impact of workforce and capacity flexibility on operational performance.

In the considered case-study, the task of a production manager is to determine the manufacturing capacity subject to minimal costs including manufacturing (i.e., internal efficiency) and lost-sales/opportunity costs (i.e., customer perspective in the form of service level). The trade-off consists in having either idle capacity or lost orders.

We consider a single-stage manufacturing process (Fig. 45):

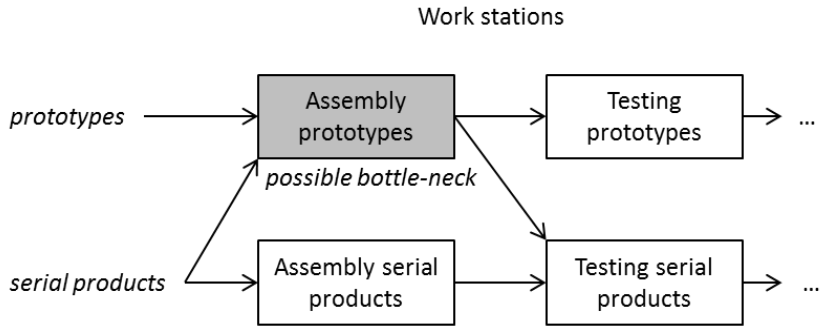


Fig. 45. Single-stage manufacturing system[1]

The surveyed production system of electronical devises deals with a hybrid ETO-MTO production. This means building prototypes is the precondition for a later serial production (MTO). The manufacturing process of prototypes and the serial manufacturing is nearly identical. The first step is the assembly of the devises. After the assembly, all devices in the batch are tested on specific testing work stations. The main difference between serial manufacturing and rapid prototyping is the required qualification of the workers and their labor costs. The workers on the work station for rapid prototyping have a higher level of qualification and can produce the prototypes as well as

[1] Zschorn L., Mueller S., Ivanov D. (2016) Cost analysis of capacity flexibility in a hybrid multiple-line production system at Siemens AG. Proceedings of IFAC MIM Conference, June 28-30, Troyes, France (forthcoming).

the serial products. On the work station for serial manufacturing only the serial products can be assembled.

The available capacity at an assembly work station is determined by the number of workers. We assume that there are no restrictions on available space and no other limitations on the available capacity. The capacity is measured in time units, i.e., in hours. The demand on capacity for rapid prototyping is considered to be sporadically. In order to react faster on the markets and to generate customer orders for serial manufacturing it is necessary to fulfill the desired due dates. This is the requirement from the customer and the business and reflects the service level.

The demand fulfillment subject to the planned due date generates a benefit what can be measured as revenue. The fulfillment depends on the available capacity of the work station assembly for rapid prototyping (RPT). The available capacity can be adapted by the number of workers on the possible bottle-neck. Thus, the amount of qualified workers for the prototype assembly can influence the revenue and profit. If lead time of an RPT order does not exceed 120 time units, we get full revenue. If lead time exceeds 120 time units, revenue is reduced by 20%.

The pool of the qualified workers leads to costs which are constant in the considered period of time. The labor costs of RPT workers are higher than the labor costs for serial manufacturing. On the other hand, a high level of available capacity can lead to a lower degree of capacity utilization in the considered period. Thus the variable costs arise if the expensive work station for the prototype assembly is used for serial production. This represents opportunity costs as difference between the hourly wage rates of an RPT worker and a serial worker. We assume that demand for serial products is permanently high which implies that if the rapid prototype capacity is not utilized with a rapid prototype order, it is utilized with a serial order.

The customer orders arrive at some partially uncertain rate at partially uncertain time. The orders are different regarding the processing times. The order completion brings us revenue, but also creates the costs. If an incoming order cannot be processed because of insufficient assembly capacity, it needs to wait. If the waiting time of an order exceeds 80 time units, the customer withdraws this order. This implies opportunity costs.

Total duration of the planning period is 439 time units.

The management objective is to determine production capacity for rapid prototyping by balancing the incoming customer orders with capacity and the demand in order to achieve maximal possible profit under given constraints.

The students are divided into *teams* each of which comprises the following *players*:

- A division head who strives to maximize total profit of the division and unit profits for the orders
- A sales manager who strives to maximize revenue, to accept as many customer orders as possible and to complete the orders as quick as possible subject to lead-time
- A manufacturing manager who seeks for the lowest unit costs and high capacity utilization,
- A workload balancer who needs to balance the objectives of the division head, the sales manager, and the manufacturing manager in order to achieve maximal possible profit under given constraints.

The teams get from the instructor input data set and compete against each other subject to total profitability. At the same time, the managers within each team compete against each other to become the best from all the teams in regard to their individual KPI.

8. Model building

With the help of AnyLogic multimethod simulation software, the management problem will be transferred to a simulation model that will allow performing experiments in order to understand basic trade-offs and relations in process flow analysis with consideration of financial and operational KPIs (Key Performance Indicators).

8.1 Step 1. Create process model

Hint from simulation professionals: a good modeller tries to use already existing models and adapt them to new problem statements.

We take the model "Call Center" from sample models in AnyLogic 7.2 PLE as a basis for our model development. In this step, we create a process model for our problem statement (Fig. 46).

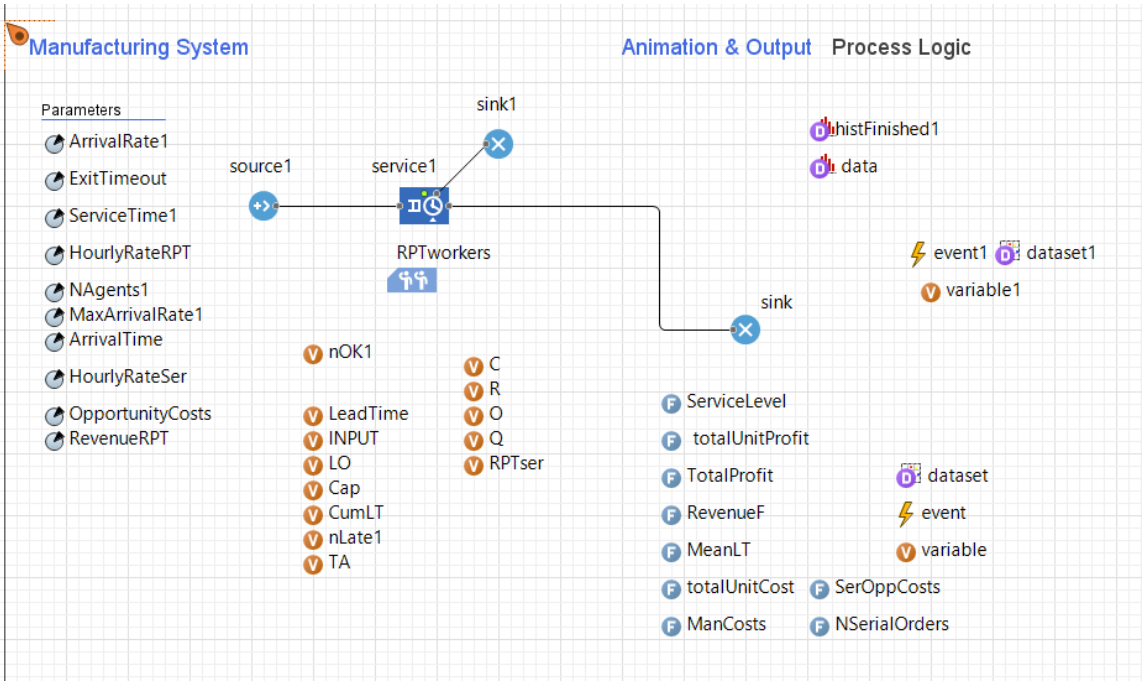


Fig. 46. Process model

The created model contains a source node where the customer orders arrive, the assembly station for prototypes ("service1"), the resource for the assembly station ("RPTworkers"), and two sink nodes ("sink" as the process end and "sink1" for lost orders). The following parameters, variables, events and functions are used (Tables 5-8).

Table 5 Parameters

Parameter	**Meaning**
ArrivalRate1	Minimum arrival rate/quantity
MaxArrivalRate1	Maximum arrival rate/quantity
ArrivalTime	Interarrival time
ExitTimeout	Maximal waiting time for an incoming RPT order that results in a lost order
ServiceTime1	Processing time of one order at the assembly station
HourlyRateRPT	Hourly wage rate for a rapid prototype worker
HourlyRateSerial	Hourly wage rate for a serial product worker
NAgents1	Number of rapid prototype workers
OpportunityCosts	Opportunity costs
RevenueRPT	Revenue for a prototype order

Table 6 Variables

Variable	Meaning
nOK1	Number of on-time delivered (OTD) RPT orders
nLate1	Number of delayed RPT orders (i.e., lead time > 3 weeks = 120 time units)
LeadTime	Lead time for an RPT order
INPUT	Number of arrived RPT orders
LO	Number of lost RPT orders
Cap	Utilization of RPT workers
CumLT	Cumulative lead time for an RPT order
R	Total revenue
Q	Total number of completed RPT orders
O	Opportunity costs because of lost orders
TA	Arrival time of a new order
variable	Auxiliary variable for statistics on incoming orders; used in "event"
variable1	Auxiliary variable for statistics on completed orders; used in "event1"

Table 7 Functions

Function	Meaning
ServiceLevel	Returns service level value as `return (Q / INPUT);`
totalUnitProfit	Returns unit profit value as `return ((R -(ManCosts() + SerOppCosts() + O))/Q );`
totalUnitCosts	Returns unit costs value as `return ((ManCosts() + SerOppCosts() ) / Q);`
TotalProfit	Returns total profit value as `return (R-(ManCosts() + SerOppCosts() + O));`
RefenueF	Returns total revenue value as `return (R);`
MeanLT	Returns average lead time value as `return zidz (CumLT, Q);`
SerOppCosts	Opportunity costs because of using RPT workers for serial orders; `return NAgents1*(HourlyRateRPT - HourlyRateSer) * (439 * (1 - RPTwork-ers.utilization()));`
ManCosts	Total manufacturing costs `return NAgents1*HourlyRateRPT*439;`
NSerialOrders	Number of serial orders completed at the RPT work station; `return 50 * (1 - RPTworkers.utilization());`

;

Table 8 Events

Parameter	Meaning
Event	Collect statistics on incoming orders
Event1	Collect statistics on completed orders

Note: complexity reduction! In the problem statement we assumed that "demand for serial products is permanently high which implies that if the rapid prototype capacity is not utilized with a rapid prototype order, it is utilized with a serial order." For model building this implies that we do not need to represent second assembly line for serial orders at all.

8.2 Step 2. Create custom agent and define rules for order arrival, waiting, and processing: usage of time functions and resource pools

In this task, we are going to apply new methods for KPI computation. In particular, instead of applying blocks "timeMeasureStart" and "timeMeasureEnd", we will measure lead time using

functions and parameters. That is why we need to introduce new custom agent in the "Source" block as a "New Agent" (Fig. 47). For this new agent, we define a parameter "*timeArrived*".

Next, we define the rules for the quantity and frequency of incoming orders, waiting and processing (Figs 47-48).

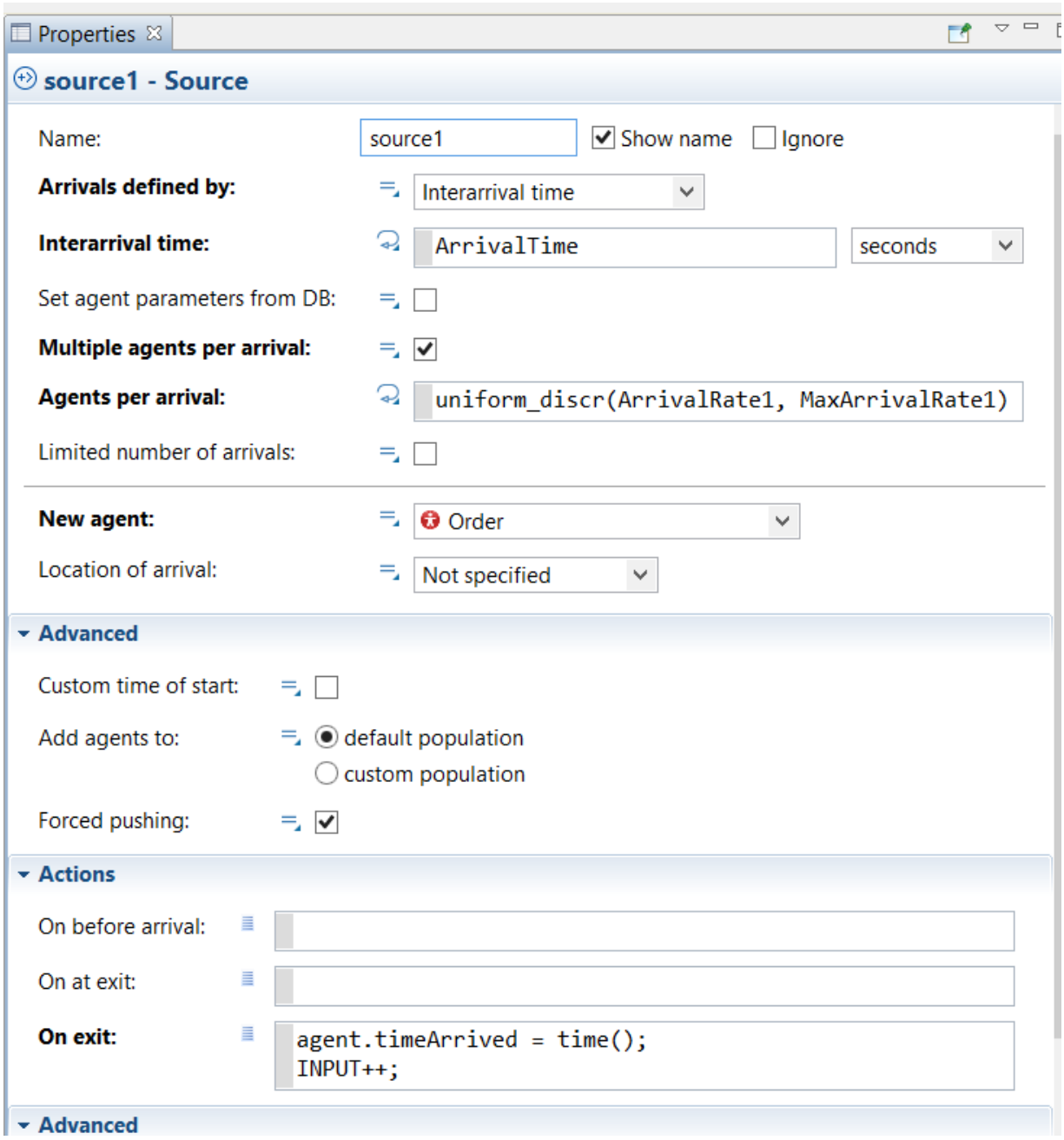


Fig. 47. Defining incoming flow (input)

The definition of the incoming flow is similar to Game 1. On exit from the source block, parameter "timeArrived" gets the current time value with the help of function "time()" and the number of arrived orders is updated with the help of variable INPUT.

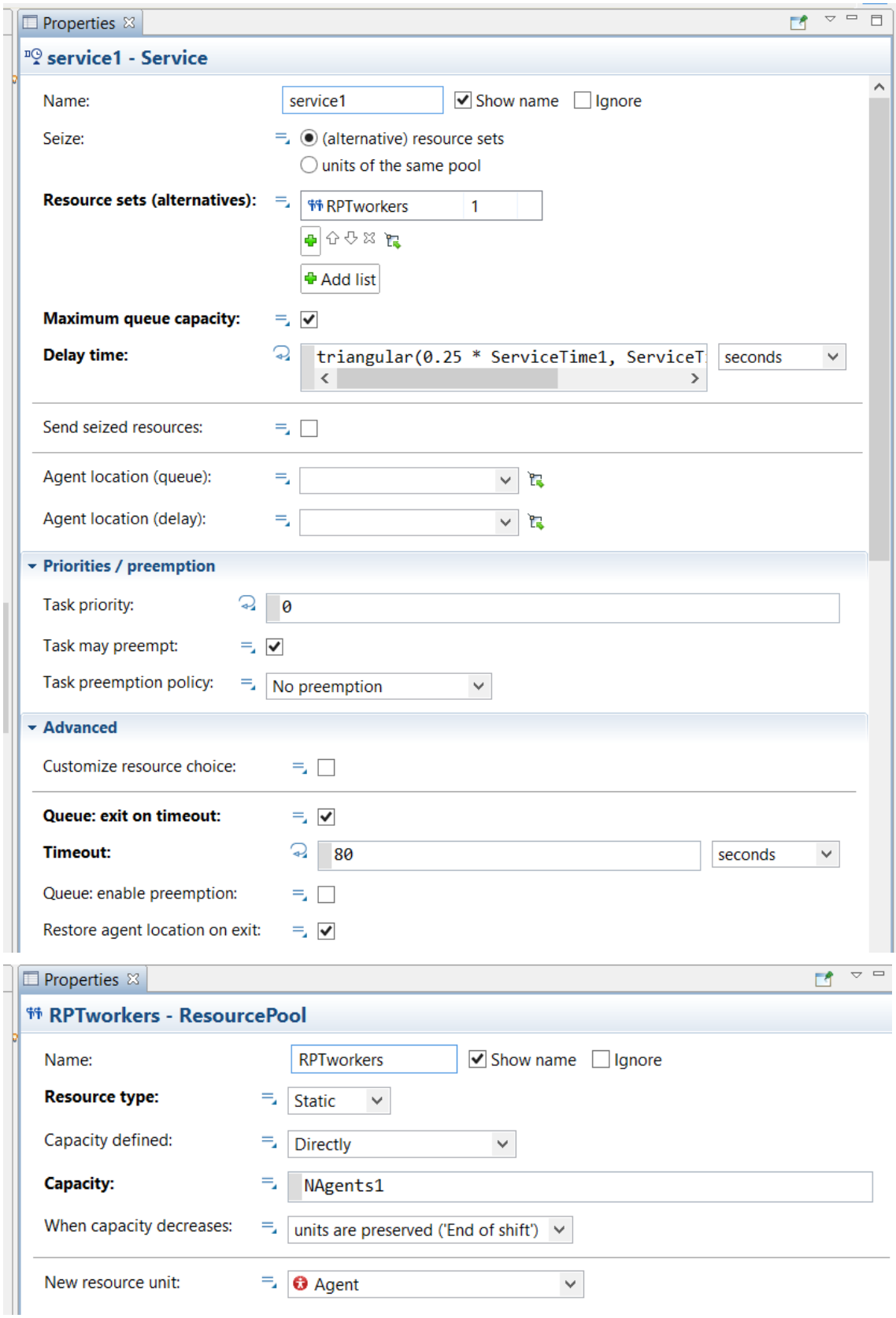


Fig. 48. Defining processing rules in the Service and Resource blocks

In the "Service" block, we define the resource set, delay time, and timeout exit time. The capacity (i.e., the number of RPT workers) is defined in the Resource properties.

8.3 Step 3. Collecting statistics and KPI dashboard design

In this step, we are going to create diagrams to analyse the following KPIs:

- Revenue, costs, profit
- Capacity utilization
- Lead time
- Completed, lost and delayed RPT orders

In Fig. 49, the developed KPI dashboard is presented.

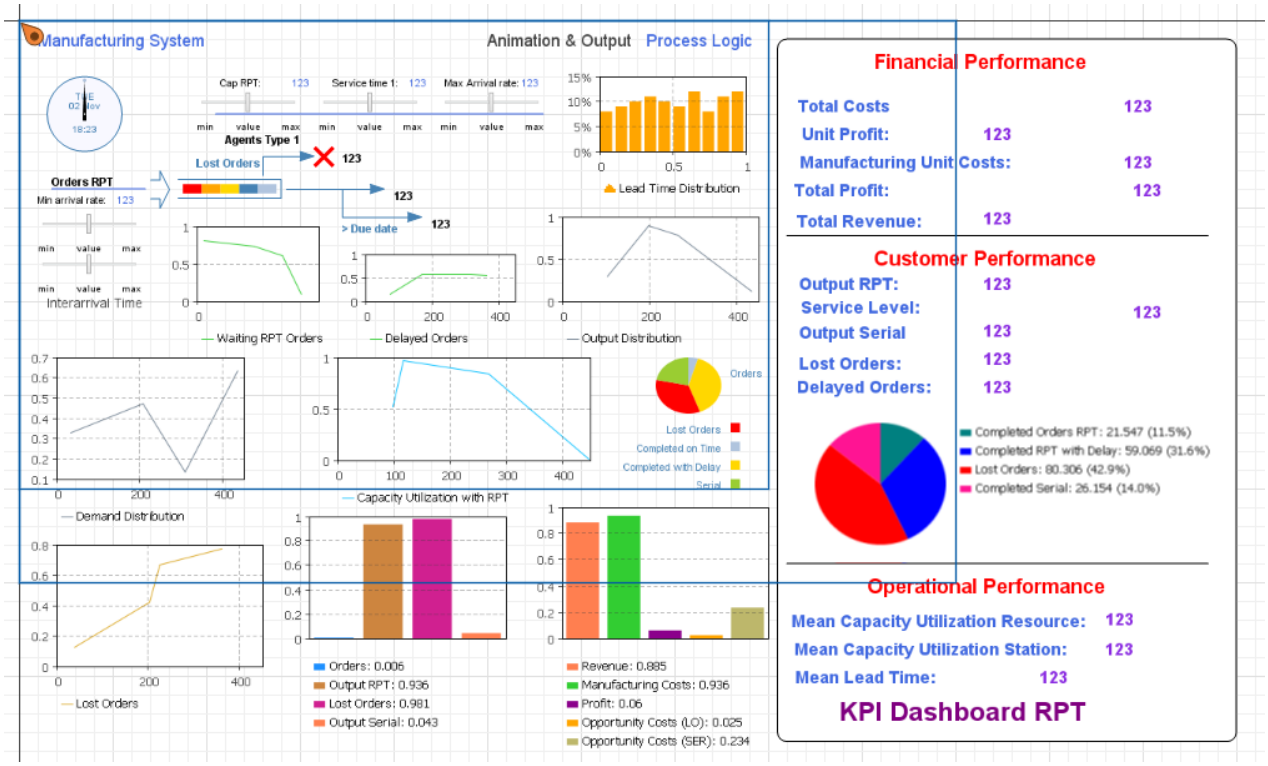


Fig. 49. KPI dashboard

8.3.1 Revenue, costs, profit

To compute financial performance as revenue, costs and profit, we define the rules for Revenue statistics collection in "Sink → Properties" as

```
R = (nOK1 * RevenueRPT) + (nLate1 * 0.8 * RevenueRPT);
```

Total costs comprise manufacturing function ManCosts()) + Opportunity costs because of using RPT workers for serial orders function SerOppCosts()) + Opportunity costs because of lost orders (variable LO that is updated in "Sink1").

Profit is computed in function TotalProfit():

```
return (R-(ManCosts() + SerOppCosts() + O));
```

8.3.2 Resource pool capacity utilization

In the standard "Resource pool" function, the statistics of average capacity utilization can be retrieved using the function [*resourceName*].*Utilization()* (Fig. 50).

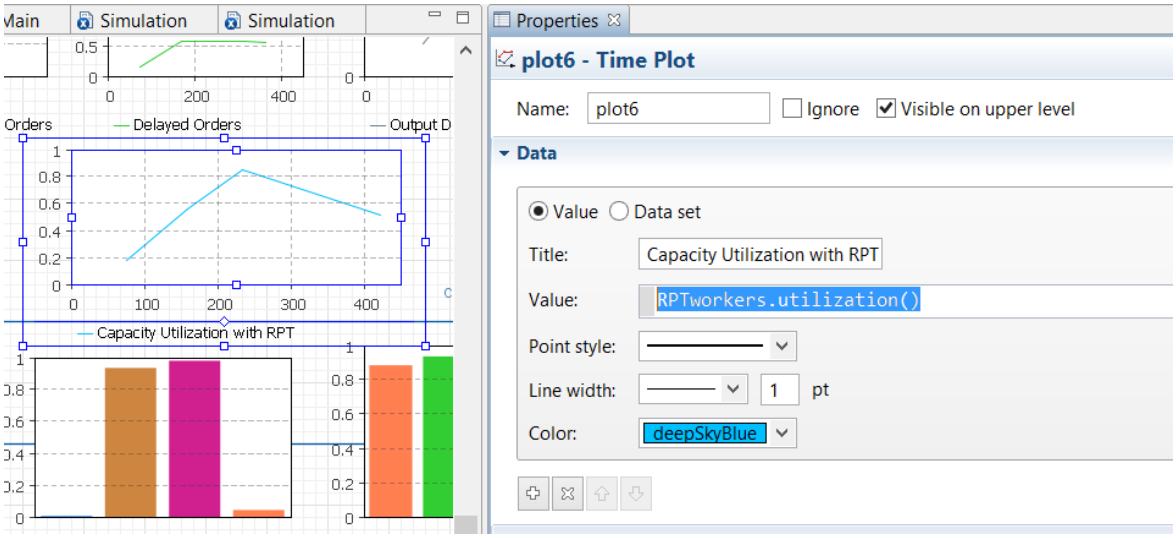


Fig. 50. Statistics update on resource pool capacity utilization

8.3.3 Lead time with the use of time function

In Game 1, in the process flow diagram, we included four points to measure the lead times. More elegant way to measure lead time is to use parameters and time functions. In Fig 50, we show the way to measure lead time and update statistics in Sink node properties as follows:

```
//at what time was the order completed?
double LeadTime = time() – agent.timeArrived;


//update stats for diagram “Lead Time Distribution”
data.add( LeadTime );
//update data for function MeanLT() to return the value “Mean Lead Time” in the KPI dashboard
CumLT = CumLT + LeadTime;
```

Note: average lead time can also be measured using

```
data.mean()
```

For function MeanLT(), please consult Table 7.

8.3.4 Total output, completed on time (OTD), delayed and lost orders: usage of Java code for conditions

Total number of completed orders is updated in “Sink” properties (Fig. 50) as

```
Q++;
```

According to problem statement, “if lead time of an RPT order does not exceed 120 time units, we get full revenue. If lead time exceeds 120 time units, revenue is reduced by 20%.” We define this condition in the “Sink” properties (Fig. 51) as a simple Java code:

```
if( LeadTime <= 120 )
	nOK1++;
else
	nLate1++;
```

So the variable “nOK1” accumulates the number of OTD orders and the variable “nLate1” accumulates the number of delayed orders.

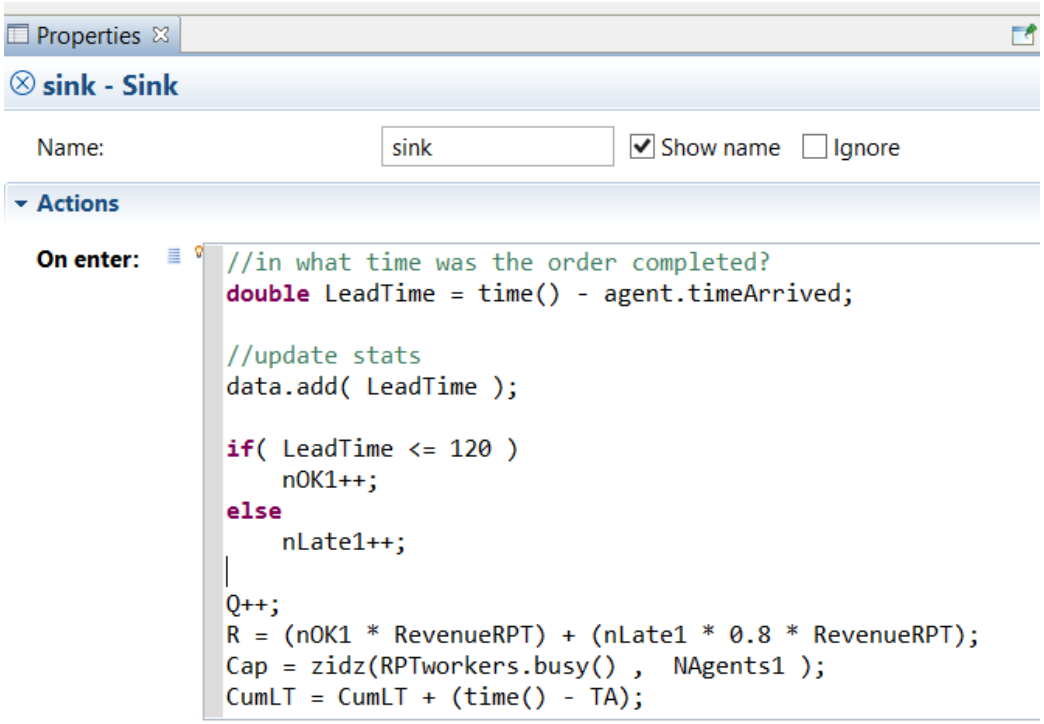


Fig. 51. Statistics update in node “sink”

Lost orders and the respective opportunity costs are measured in the “Sink1” by

```
LO++;
O = O + OpportunityCosts;
```

Finally, the number of serial orders fulfilled at the RPT work station can be measured in function NSerialOrders:

```
return 50 * (1 - RPTworkers.utilization());
```

Here we assume that if no RPT order would arrive, the RPT work station would be able to complete 50 serial orders.

9. Playing the game

The following setups hold true for all rounds:

- Each experiment will contain 439 seconds as 439 working hours of real life (i.e., roughly 11 weeks with 40 working hours per week)
- Queuing rule is FIFO
- Queue capacities are maximum
- Push principle is used

9.1 Game rules

Now we are ready to play the game. *The players* are:

- A division head who strives to maximize total profit of the division and unit profits
- A sales manager who strives to maximize revenue, to accept as many customer orders as possible and to complete the orders as quick as possible subject to lead-time

- A manufacturing manager who seeks for the lowest unit costs and high capacity utilization
- A workload balancer who needs to balance the objectives of the division head, the sales manager, and the manufacturing manager in order to achieve maximal possible profit under given constraints.

The group wins that has the highest total profit. At the same time, in each round we will define a winner in each individual category: best manufacturing manager (KPI: lowest unit cost and highest capacity utilization), best workload balancer (KPI: highest service level), best sales manager (KPI: highest output and lowest lead time), and best division head (KPI: highest total profit).

9.2 Round #1

For the first round, the following input data can be suggested (Table 9):

Table 9 Input data

Parameter	**Value**
ArrivalRate1	3
MaxArrivalRate1	5
ArrivalTime	40
ExitTimeout	80
ServiceTime1	20
Delay Time	`triangular(0.25*ServiceTime1, ServiceTime1, 2*ServiceTime1)`
Agents per Arrival	`uniform_discr(ArrivalRate1, MaxArrivalRate1)`
HourlyRateRPT	90
HourlyRateSerial	30
NAgents1	3
OpportunityCosts	$4,000
RevenueRPT	$9,000

No parameters will be changed during the experiment. The results are presented in Fig. 52 and Table 10.

Fig. 52. Results of the round #1

Table 10 KPI

KPI	**Values**
Service Level, %	100
OTD Orders	37
Delayed Orders	0
Serial Orders	20
Total Output RPT	37
Total Input RPT	37
Lost Orders RPT	0
Unit Profit, $	4,922
Unit Manufacturing Costs, $	4,077
Total Profit, $	182,123
Revenue, $	333,000
Manufacturing Costs, $	118,530
Lost Order Opportunity Costs, $	0
Serial Order Opportunity Costs, $	32,347
Total Costs, $	150,878
Average Lead Time, working hours	24.6
Average Capacity Utilization, %	59.1

9.3 Round #2

The division head means that the results of round #1 and their analysis indicate the necessity to increase the number of incoming orders. The sales and manufacturing managers support this suggestion based on their own KPIs. New minimal and maximum arrival quantities are setup at 5 orders and 7 orders respectively. The simulation results are presented in Fig. 53 and compared with round #1 in Table 11.

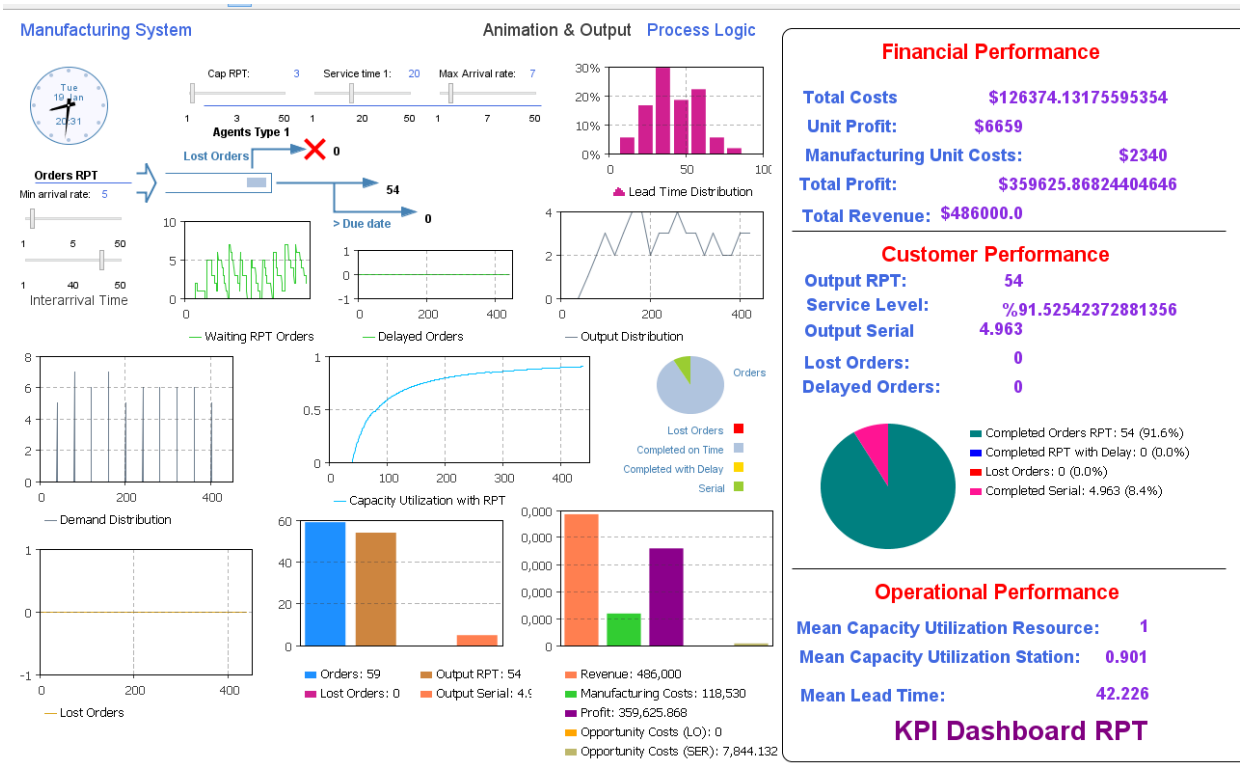


Fig. 53. Results of the round 2_1

Table 11 KPI comparison

Manager	**KPI**	**Round 1**	**Round 2_1**
Division Head	Total Profit, $	182,123	126,374
	Revenue, $	333,000	486,000
Manufacturing Manager	Unit Manufacturing Costs, $	4,077	2,340
	Average Utilization, %	59.1	90.1
	Serial Orders	20	5
	Serial Order Opportunity Costs, $	32,347	7,844
Sales Manager	Intput RPT, orders	37	59
	OTD RPT Orders	37	54
	Delayed RPT Orders	0	0
	Lost Order Opportunity Costs, $	0	0
	Mean Lead Time, days	24.6	42.2
Workload Balancer	Service Level,%	100	91.5

Sales manager indicates the demand on RPT orders is even higher and is in the range [7-10] orders per week. Setting these new parameter values the following result was obtained with the help of simulation model (Fig. 54 and Table 12).

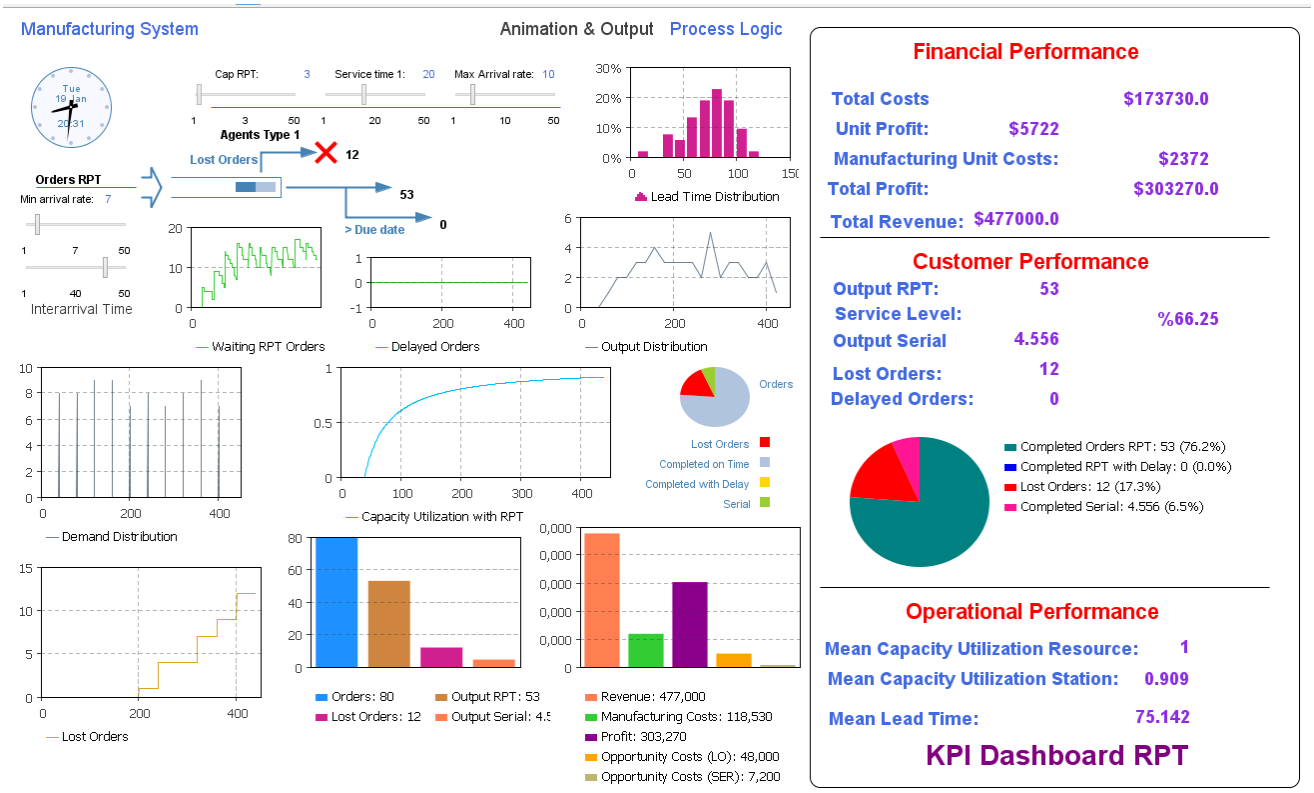


Fig. 54. Results of the round #2_2

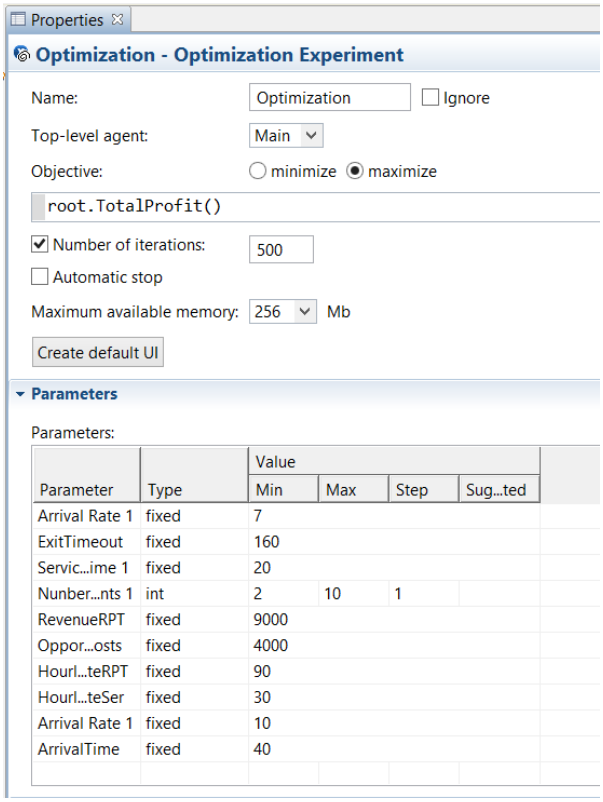
Table 12 KPI comparison

Manager	KPI	Round 1	Round 2_1	Round 2_2
Division Head	Total Profit, $	182,123	359,627	303,270
	Revenue, $	333,000	486,000	477,000
Manufacturing Manager	Unit Manufacturing Costs, $	4,077	2,340	2,372
	Average Utilization, %	59.1	90.1	90.9
	Serial Orders	20	5	4.6
	Serial Order Opportunity Costs, $	32,347	7,844	7.200
Sales Manager	Input PRT, orders	37	60	80
	OTD RPT Orders	37	54	53
	Delayed RPT Orders	0	0	0
	Lost RPT Orders	0	0	12
	Lost Order Opportunity Costs, $	0	0	48,000
	Mean Lead Time, days	23.5	42.2	75.1
Workload Balancer	Service Level,%	100	91.5	66.3

As expected, the capacity of 3 RPT workers is not enough to get along with the increased demand. As known from Game 1, an increase in waiting orders negatively influences the lead time (in our example, 75.1 days instead of 23.5 days in average). In addition, the resulted lost orders increased total costs and decreased total profit. Average capacity utilization was slightly reduced.

9.4 Round #2: Optimization experiment

In "Project → New → Experiment", we select "Optimization" and create new optimization experiment for the data from round #2_2 and solve it optimally. The experiment setup and results are shown in Fig. 55.



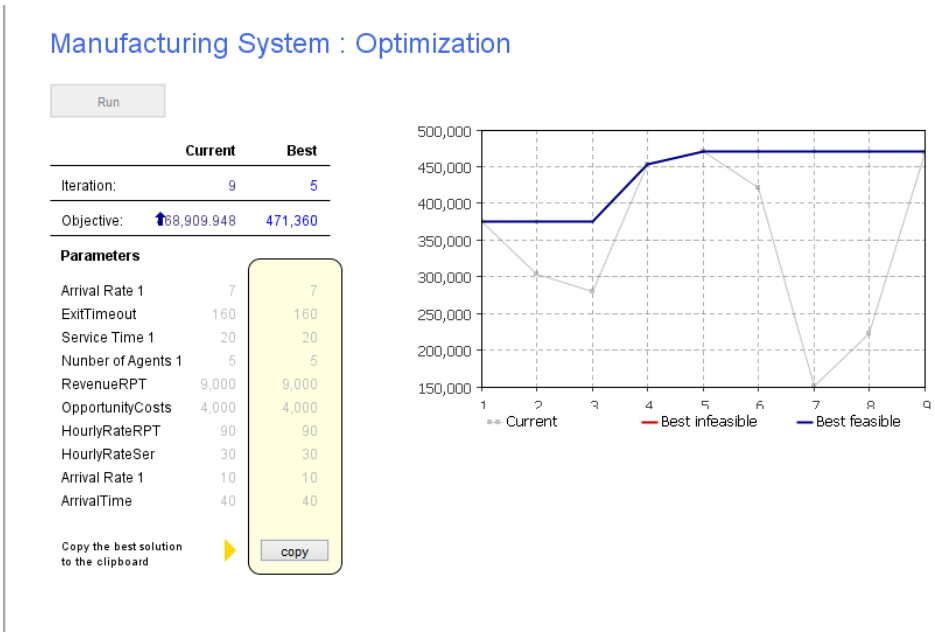


Fig. 55. Optimization experiment result for the data from the round #2_2

It can be observed from Fig. 55 that subject to the given data setting and the considered range of variable alteration, the recommendation is to use five RPT workers and we can expect total profit of $471,360.

9.5 Round #2: Optimization-based simulation experiment

Let us perform the simulation experiment on the basis of new capacity = 6 RPT workers: (see Fig. 56).

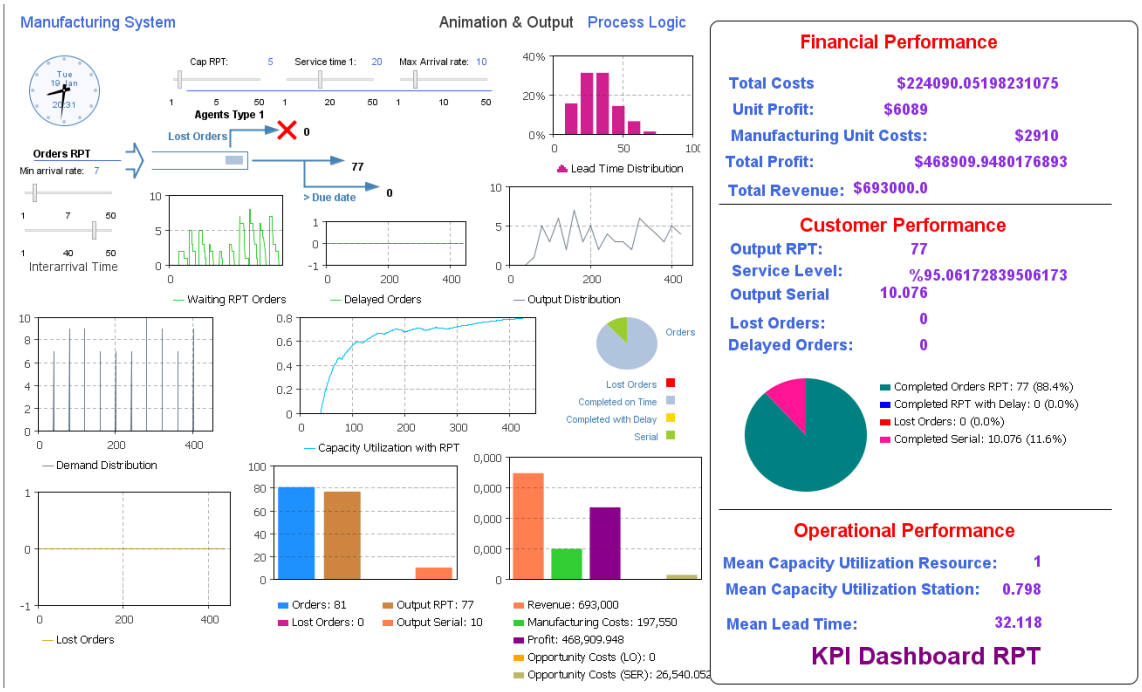


Fig. 56. Simulation results for the data from the round #2_Opt

Indeed, we improved! Total profit is now $468,910 that is very close to the optimization result. Other KPI are compared in Table 13.

9.6 Rounds #3 – #10 and game evaluation

The managers are now about to simulate for three considered demand scenarios with different numbers of RPT workers in the range [2-6].

In the following rounds, the teams compete against each other subject to total profitability. At the same time, the managers within each team compete against the managers in other teams on individual basis to become the best in regard to their individual KPI. Getting the input data, students have 5-10 minutes time to take decision on how to setup the parameters. During that time they may also use optimization experiments but not the simulation. Having setup the parameters, the teams start simulation experiment. During the experiment, they may pause the experiment as often as they want but total time for all groups need to be limited, say for 5-10 minutes for one round.

The evaluation of the game can be performed in different ways. One possible evaluation method is to evaluate each round independently and assign in each round some scores to each team, e.g., team or respective manager with the best result gets 10 points, and the team with the worst results will get 1 point. Another evaluation method is to combine the results from all the rounds and then compute average values for each KPI to determine the winner.

Table 13 KPI comparison

Manager	KPI	Round 1	Round 2_1	Round 2_2	Round 2_Opt
Division Head	Total Profit, $	182,123	359,627	303,270	468,910
	Revenue, $	333,000	486,000	477,000	693,000
Manufacturing Manager	Unit Manufacturing Costs, $	4,077	2,340	2,372	2,910
	Average Utilization, %	59.1	90.1	90.9	79.8
	Serial Orders	20	5	4.6	10
	Serial Order Opportunity Costs, $	32,347	7,844	7.200	26,540
Sales Manager	Input PRT, orders	37	60	80	81
	OTD RPT Orders	37	54	53	77
	Delayed RPT Orders	0	0	0	0
	Lost RPT Orders	0	0	12	0
	Lost Order Opportunity Costs, $	0	0	48,000	0
	Mean Lead Time, days	23.5	42.2	75.1	32.1
Workload Balancer	Service Level,%	100	91.5	66.3	95.1

We can observe some trade-offs. While KPI of division head and sales manager have been improved, KPI of manufacturing manager have decreased.

Table 14 KPI

Demand [3-5]					
KPI	**n=2**	**n=3**	**n=4**	**n=5**	**n=6**
Service Level, %	94.7	100	97.8	97.7	97.7
OTD Orders	36	37	44	43	42
Delayed Orders	0	0	0	0	0
Serial Orders	5	20	23	29	33
Total Output RPT	36	37	44	43	42
Total Input RPT	38	37	45	44	43
Lost Orders RPT	0	0	0	0	0
Unit Profit, $	6,638	4,922	4,266	2,595	852
Unit Manufacturing Costs, $	2,361	4,077	4,733	6,404	8,147
Total Profit, $	238,995	182,123	187,712	111,619	35,793
Revenue, $	324,000	333,000	396,000	387,000	378,000
Manufacturing Costs, $	79,020	118,530	158,040	197,500	237,060
Lost Order Opportunity Costs, $	0	0	0	0	0
Serial Order Opportunity Costs, $	5,985	32,347	50,248	77,831	105,146
Total Costs, $	85,005	150,878	208,288	275,381	342,207
Average Lead Time, working hours	53.0	24.6	21.5	20.0	20.1
Average Capacity Utilization, %	88.6	59.1	52.3	40.9	33.5

Table 15 KPI

Demand [5-7]					
KPI	**n=2**	**n=3**	**n=4**	**n=5**	**n=6**
Service Level, %	63.3	91.5	96.5	96.4	100
OTD Orders	38	54	55	54	61
Delayed Orders	0	0	0	0	0
Serial Orders	5	5	15	22	24
Total Output RPT	38	54	55	54	61
Total Input RPT	60	59	57	56	61
Lost Orders RPT	9	0	0	0	0
Unit Profit, $	5,846	6,659	5,546	4,264	3,823
Unit Manufacturing Costs, $	2,205	2,340	3,453	4,735	5,176
Total Profit, $	222,180	359,626	305,055	230,274	223,226
Revenue, $	342,000	486,000	495,000	486,000	549,000
Manufacturing Costs, $	79,020	118,530	158,040	197,550	237,060
Lost Order Opportunity Costs, $	36,000	0	0	0	0
Serial Order Opportunity Costs, $	4,800	7,844	31,894	58,176	78,714
Total Costs, $	119,820	126,374	189,934	255,726	315,774
Average Lead Time, working hours	72.7	42.2	27.3	23.1	22.5
Average Capacity Utilization, %	90.9	90.1	69.7	55.8	50.2

Table 16 KPI

Demand [7-10]					
KPI	**n=2**	**n=3**	**n=4**	**n=5**	**n=6**
Service Level, %	45.2	66.3	83.5	95.1	98.8
OTD Orders	38	53	71	77	79
Delayed Orders	0	0	0	0	0
Serial Orders	5	5	5	10	16
Total Output RPT	38	53	71	77	79
Total Input RPT	84	80	85	81	80
Lost Orders RPT	27	12	0	0	0
Unit Profit, $	2,952	5,722	6,638	6,089	5,337
Unit Manufacturing Costs, $	2,205	2,372	2,361	2,910	3,662
Total Profit, $	150,180	303,270	471,360	468,910	421,636
Revenue, $	342,000	477,000	639,000	693,000	711,000
Manufacturing Costs, $	79,020	118,530	158,040	197,550	237,060
Lost Order Opportunity Costs, $	108,000	48,000	0	0	0
Serial Order Opportunity Costs, $	4,800	7,200	9,600	26,540	52,304
Total Costs, $	191,820	173,730	167,640	224,091	289,364
Average Lead Time, working hours	74.8	75.2	55.1	32.1	27.1
Average Capacity Utilization, %	90.9	90.9	90.9	79.8	66.9

9.7. Capacity flexibility analysis

Capacity flexibility is our example is measured as the ability of a manufacturing system to cope with changes in demand. In other words, flexibility can be measured as service level. Let us depict the trade-offs between the number of RPT workers and the KPI (Fig. 57-60).

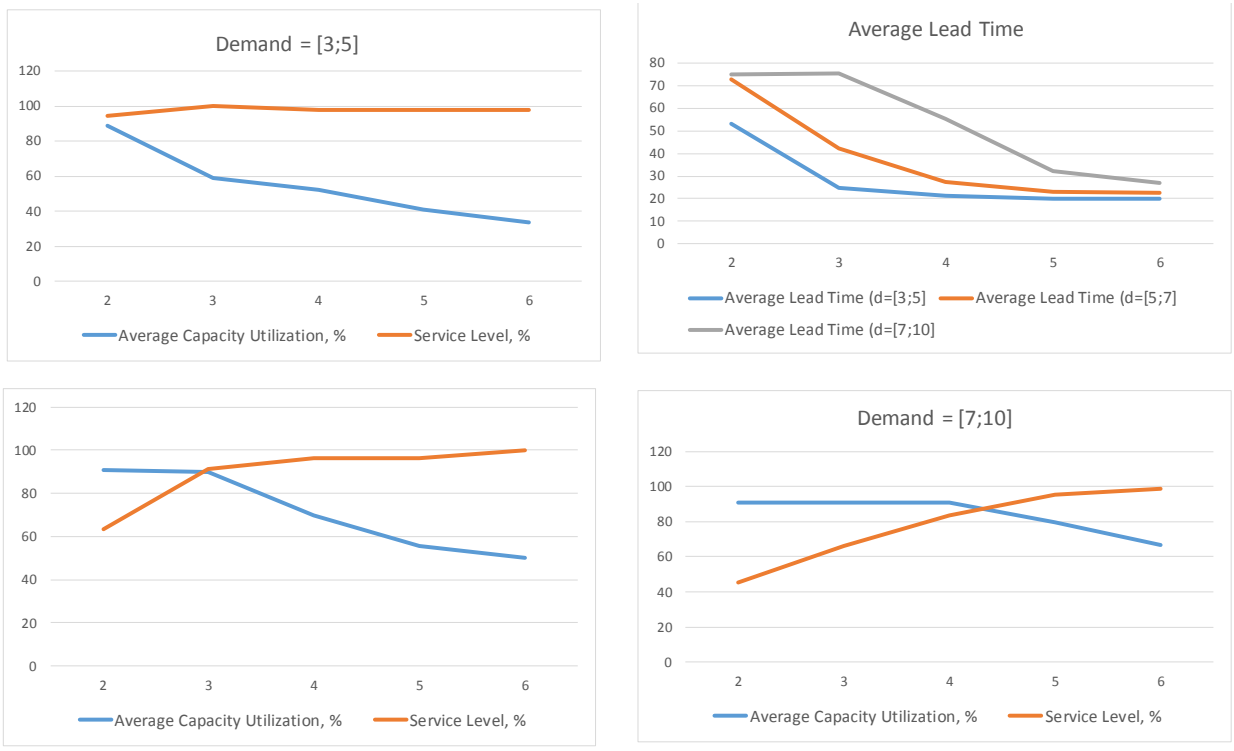


Fig. 57. Operational KPI analysis

It can be observed from Fig. 57 that both service level (i.e., flexibility), lead time, and capacity utilization depend on the number of workers (cf. Fig. 25).

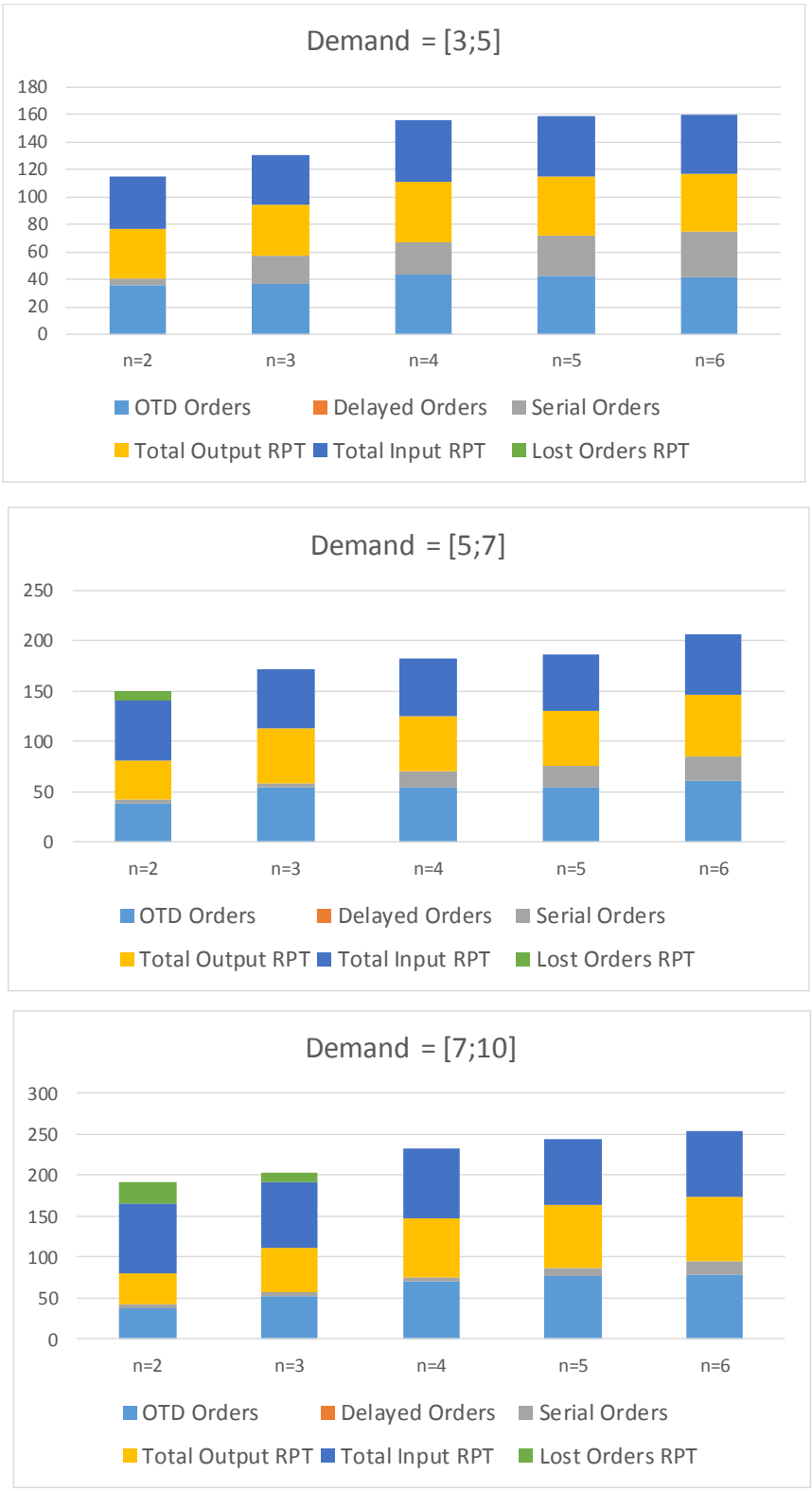
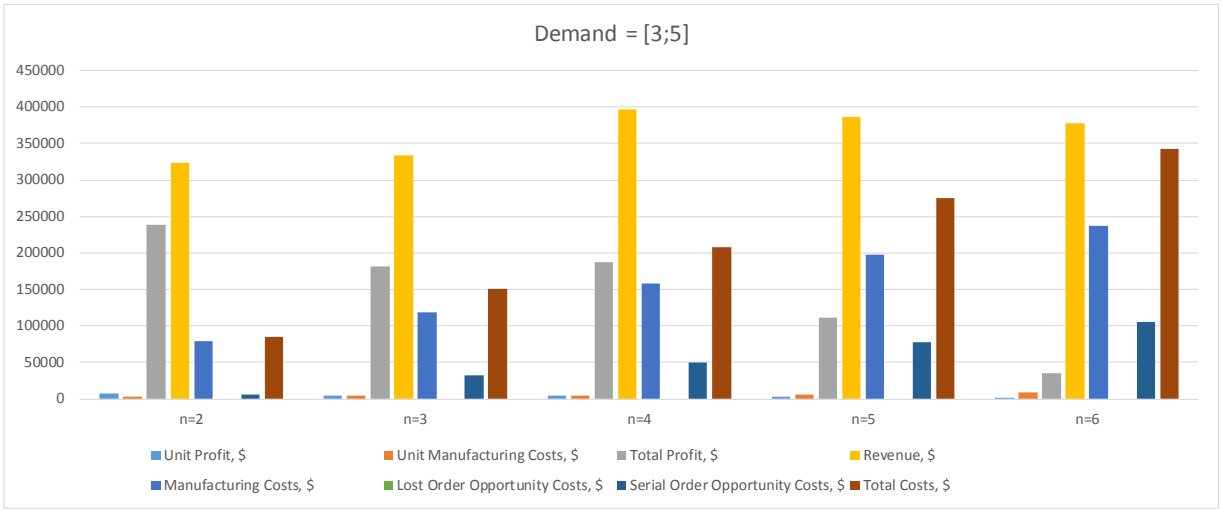
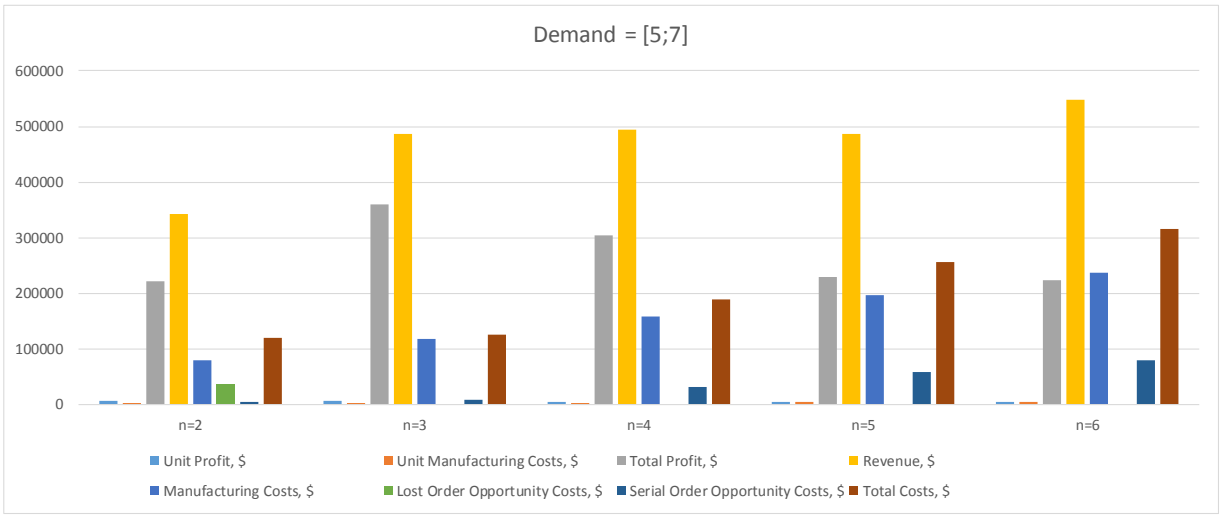


Fig. 58. Customer KPI analysis

Fig. 58 depicts major impacts of the RPT worker number on the customer performance.





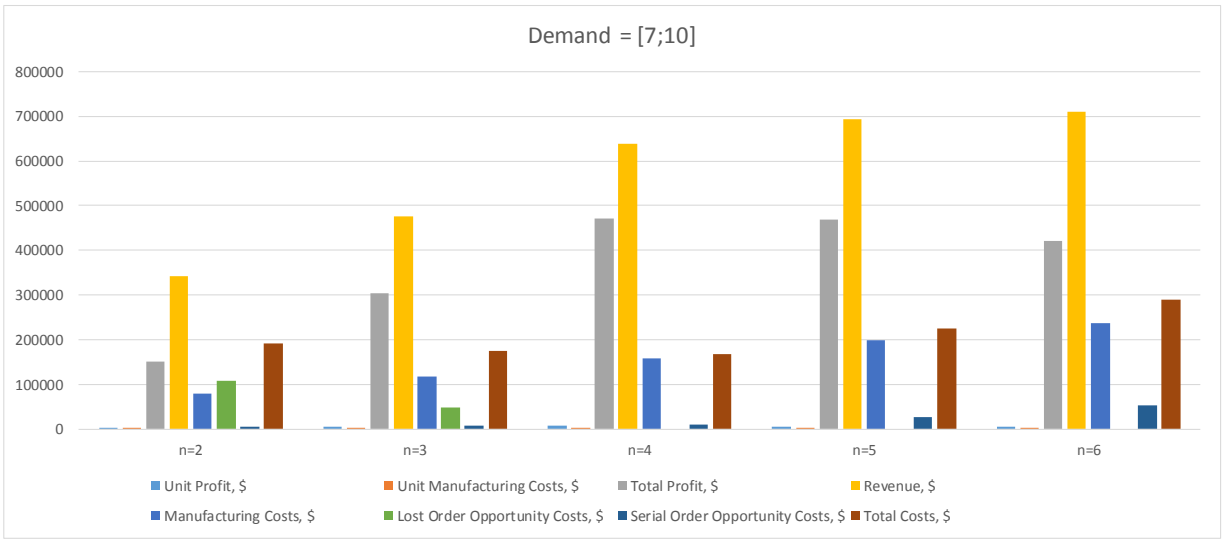


Fig. 59. Customer KPI analysis

Fig. 59 depicts major impacts of the RPT worker number on the financial performance. It can be observed that maximum profits are achieved in the case of 2 workers for low demand, 3 workers for medium demand, and 4 workers for high demand.

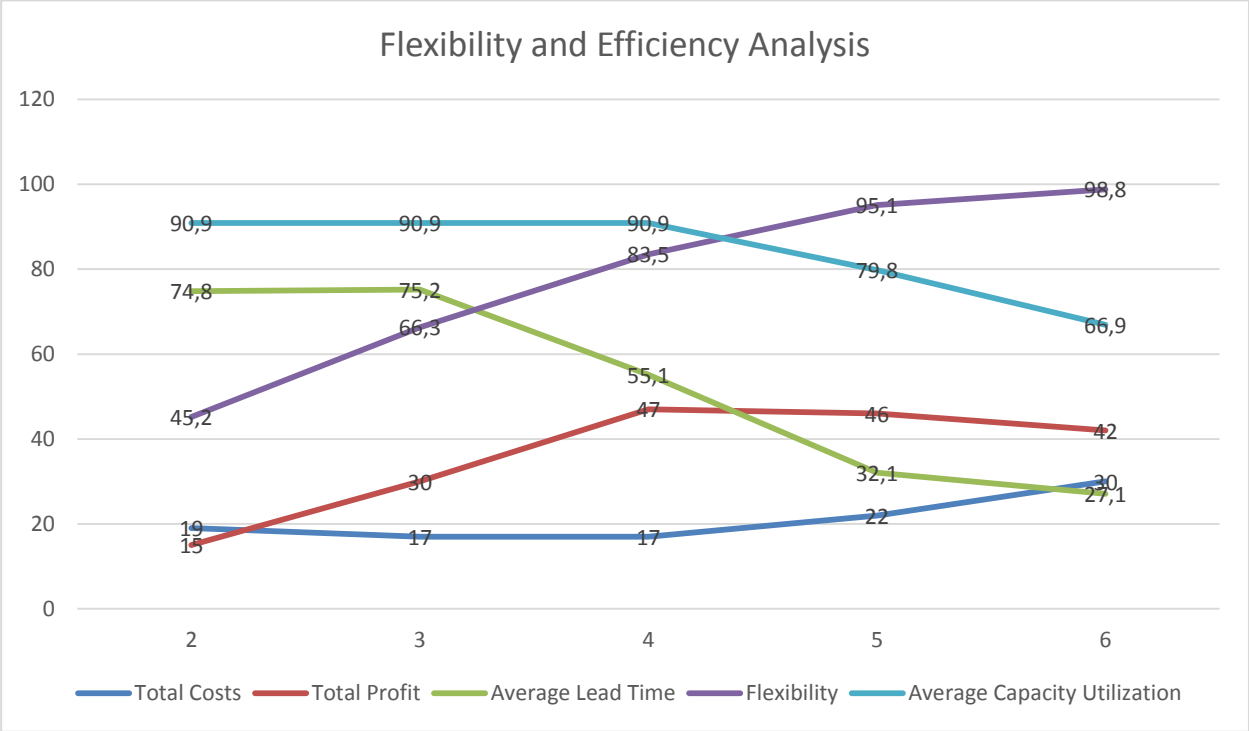


Fig. 60. Flexibility vs analysis: empirical data

Major implications of the RPT worker number, flexibility, and efficiency can be observed from Fig. 60 based on the high demand data. While total costs and profit achieve their best values in the case of 4 workers, service level and lead time can be further improved if increasing the number of RPT workers. In Fig. 61, we summarize our findings.

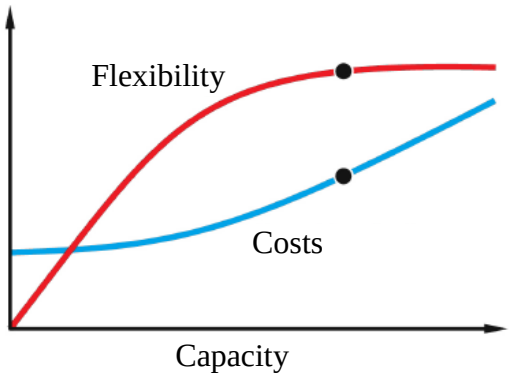


Fig. 61. Flexibility vs analysis: generalization

Final decision on the capacity planning (i.e., how many RPT workers are needed) implies an executive meeting with all the managers. Probably, three or four workers will be suggested, in dependence on other factors such as demand forecast quality, options for worker exchange with other lines, etc.

Part III Supply Chain Coordination

10. Supply Chain Coordination with AnyLogic

10.1 Learning objectives

- To develop analytical and management skills on supply chain coordination
- To develop soft skills on coordinated team-based decision making
- To develop technical skills on creating discrete-event process simulation models for networks and perform performance measurement analysis on example of AnyLogic multimethod simulation software

This example can also be used as a game on the principles similar to two previous examples (Part I and Part II).

10.2 Problem statement

We consider a supply chain network in car manufacturing that comprises an assembly plant, two suppliers, and transportation to customers. At the manufacturer, we have a linear manufacturing process that contains four working stations (M1-M4). M1 produces car bodies. Supplier #1 (S1) delivers doors that are assembled at M2. Supplier #2 (S2) delivers engines that are assembled at M3. M4 is final assembly station. Next the cars go to the packing station. They are delivered to customers in batches of 10 cars. (Fig. 62).

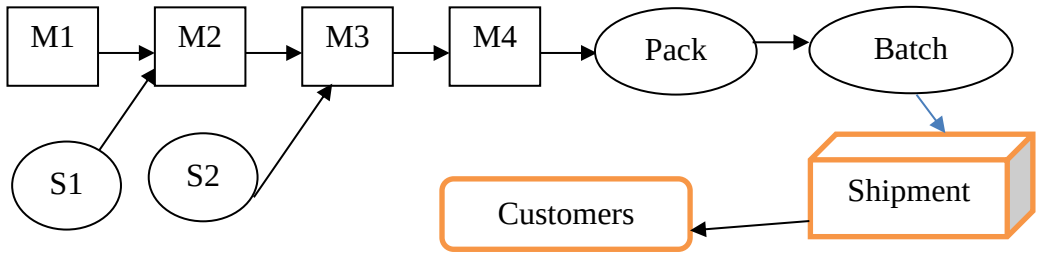


Fig. 62. Problem statement

The customer orders arrive at some partially uncertain rate at partially uncertain time. The orders are different regarding the processing times at the machines but they are processed in the same sequence (i.e., the flow shop) at four subsequent machines. Each order has partially variable processing times at the machines (M1-M4). M1 processes car bodies. M2 is assembly station for doors coming from supplier #1. M3 is assembly station for engines coming from supplier #2. M4 is final assembly station. Subsequently, the cars are packed and batches for delivery are created. Finally, the cars are shipped to the customers.

The order delivery to the customer brings us revenue X, but also creates the costs Y which implies manufacturing, transportation and inventory costs. If an incoming order cannot be processed because of insufficient line capacity, it needs to wait. If a customer order waits more than D-days, it is removed by the customer. This causes lost orders and opportunity costs.

The management objective is to balance incoming customer orders, deliveries from suppliers and shipments to customers with capacity and the demand in order to achieve maximal possible profit under given constraints.

Use KPI dachboard to analyse the results and basic trade-offs between takt time at the JIT (just-in-time) assembly line, production quantities and delivery frequencies at the suppliers, transportation time, batch sizes, and capacity utilization to make better decisions on supply chain coordination in the presence of bottle-necks and uncertainties in demand and lead time.

Total duration of the planning period is 450 time units.

11. Model building

With the help of AnyLogic multimethod simulation software, the management problem will be transferred to a simulation model that will allow performing experiments in order to understand basic trade-offs and relations in process flow analysis with consideration of financial and operational KPIs.

11.1 Step 1. Create process model

Hint from simulation professionals: a good modeller tries to use already existing models and adapt them to new problem statements.

We take the model from Part I as a basis for our model development. In this step, we create a process model for our problem statement (Fig. 63).

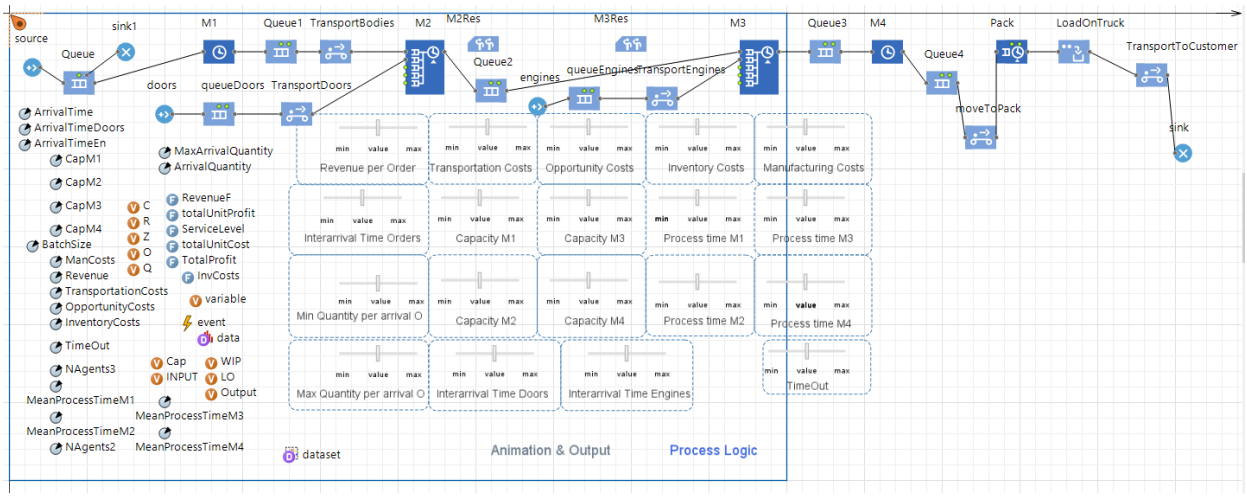


Fig. 63. Process model

The networked supply chain structure is created based on three source blocks for bodies, doors, and engines, respectively. In the process model depicted in Fig. 63, some new elements have been used that need to be explained more detailed.

11.2 Assembler

The object "Assembler" of the Process Modelling Library allows certain number of agents from several sources (five or less) to be joined into a single agent. The type of the new agent, as well as its initialization, is specified by the user. The number of agents for each input required to produce one new agent is also specified using the object parameters (Quantity 1, Quantity 2, etc.). All arrived agents wait inside the object until all required agents arrive. Once the new agent can be build, the assembly operation starts. This operation takes the time specified in the Delay time parameter. Assembly may involve resource usage - in this case you should specify what resource(s) are required to perform the assembly.

In our model, "Assembler" is used for stations M2 and M3. Assembler definition for M2 is shown in Fig. 64.

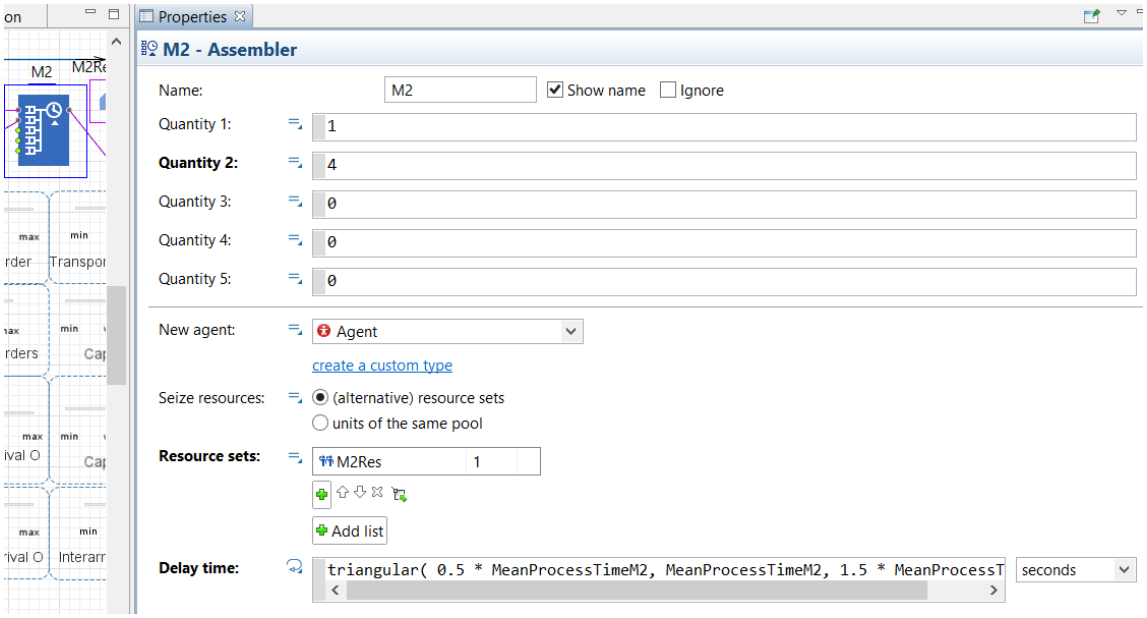
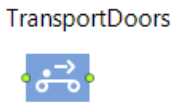


Fig. 64. Assembler definition for M2

For station M2, we define that 1 body and 4 doors are needed for this assembly step. Further, we define the resource "M2Res" for M2 and the delay time based on parameter "MeanProcess-TimeM2".

11.3 Transportation with the help of "Conveyor" or "MoveTo"



Internal and external transportation can be depicted using the blocks "Conveyor" or "MoveTo" from Process Modelling Library. An example for door transportation definition from S1 to assembly plant is shown in Fig. 65.

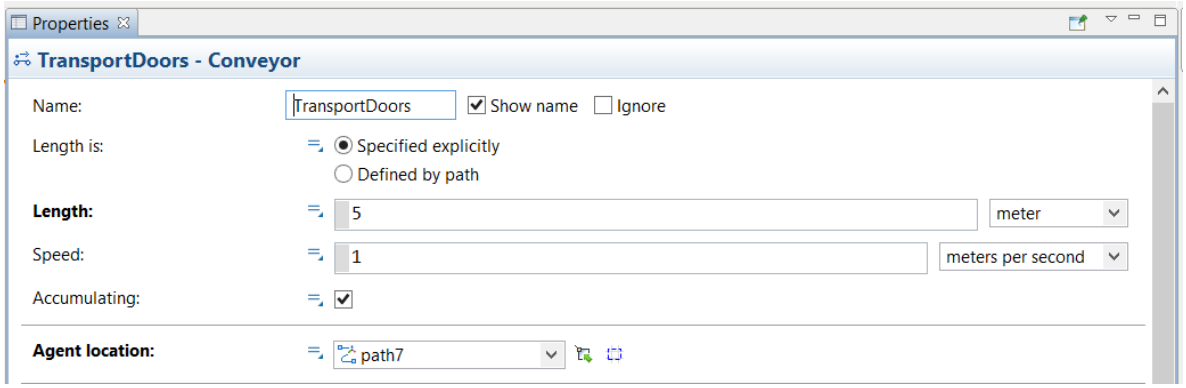
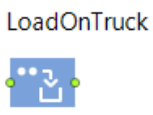


Fig. 65. Door transportation definition from S1 to assembly plant

By setting the values for length and speed of the conveyor, we define transportation time. Under assumption that 1 second of modelling time corresponds to 1 hour of real life, according to Fig. 65, transportation time from S1 to assembly plant will be 5 hours.

11.4 Batch



Batch block is used to accumulate certain number of units before forwarding them to the next block. In our case, we batch 10 cars for transportation to the customers (Fig. 66).

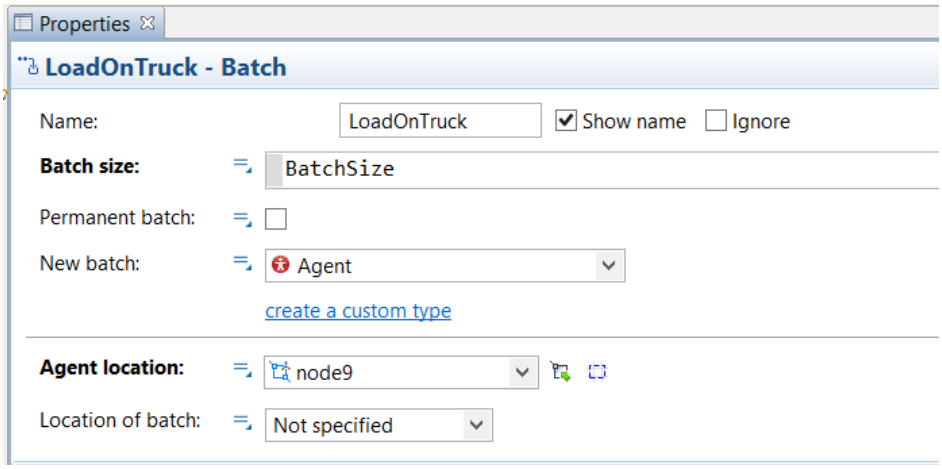


Fig. 66. Batch definition

12. Experiments and managerial insights

The following setups hold true for all rounds:

- Each experiment will contain 450 seconds as 450 working days of real life
- Queuing rule is FIFO
- Queue capacities are maximum
- Push principle is used

The following input data have been used for experiment:

- interarrival times for bodies, doors and engines are synchronized and equal 2 seconds
- processing times of M1-M4 are synchronized and distributed as triangular (0.7 * MeanProcessTimeM1, 1.3 * MeanProcessTimeM1, MeanProcessTimeM1)
- capacities at M1-M4 equal 1
- External transportation times are setup at 5 days
- Internal transportation times are setup at 1 day
- Revenue is $30 for each car, manufacturing costs is $10 for each car, transportation costs is $30 for each delivery from assembly plant to customer, inventory costs for WIP at the assembly plant is $5 for each WIP unit at the end of the experiment, opportunity costs is $4 for each lost customer order.
- KPI computation is based on the same principles as in the examples of Part I and Part II apart from "Service Level" that is computed here as
  SL = `100 * (Output-Lost Orders)/Output`

The simulation result is presented in Fig. 67.

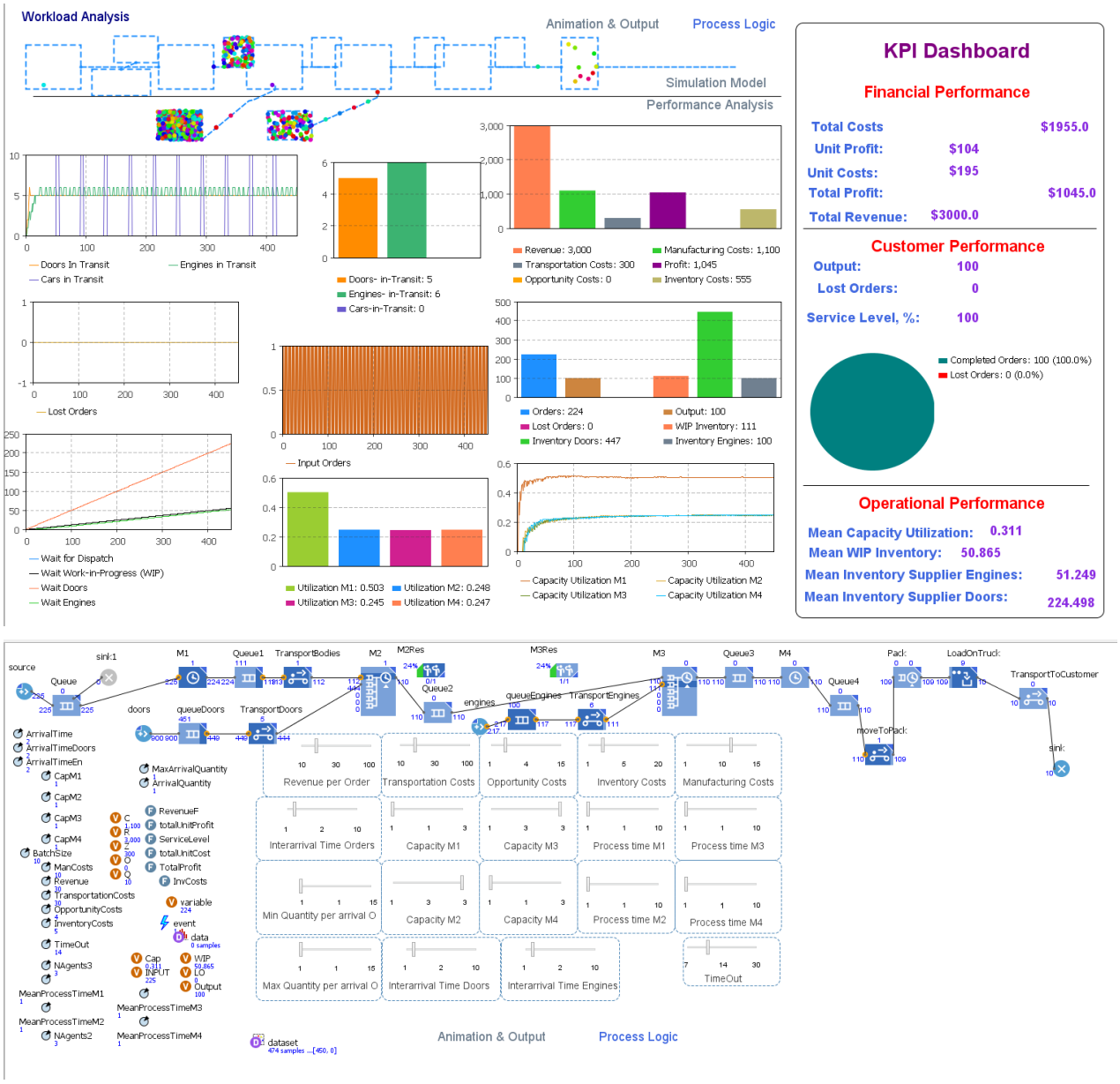


Fig. 67. Simulation results

It can be observed from Fig. 67 that major problem in the current situation is unbalanced production and transportation times. Even if production times are synchronized in the assembly line and with the suppliers (new arrivals each 2 seconds), longer transportation times result in high inventory and queues for bodies, doors, and engines.

Develop improvement suggestions!

Part IV Inventory Control

13. Modelling periodic and continuous review inventory control policies with AnyLogic

13.1 Learning objectives

1) To develop analytical and management skills on decision-making in regard to choice of right inventory control policies for products with different demand and inventory patterns
2) To develop technical skills on creating simulation models for periodic and continuous review inventory control policies and doing performance measurement analysis of those policies on example of AnyLogic multimethod simulation software

13.2 Problem statement

We consider a retailer of automotive spare parts that offers five products: nut-bolts, spark plugs, clutch plate, oil, and coolant. Two types of inventory review systems and four types of inventory control policies are involved with (see Fig. 68):

- Continuous Review Systems (s,q; s,S) and
- Periodic Review Systems (t,q; t,S)

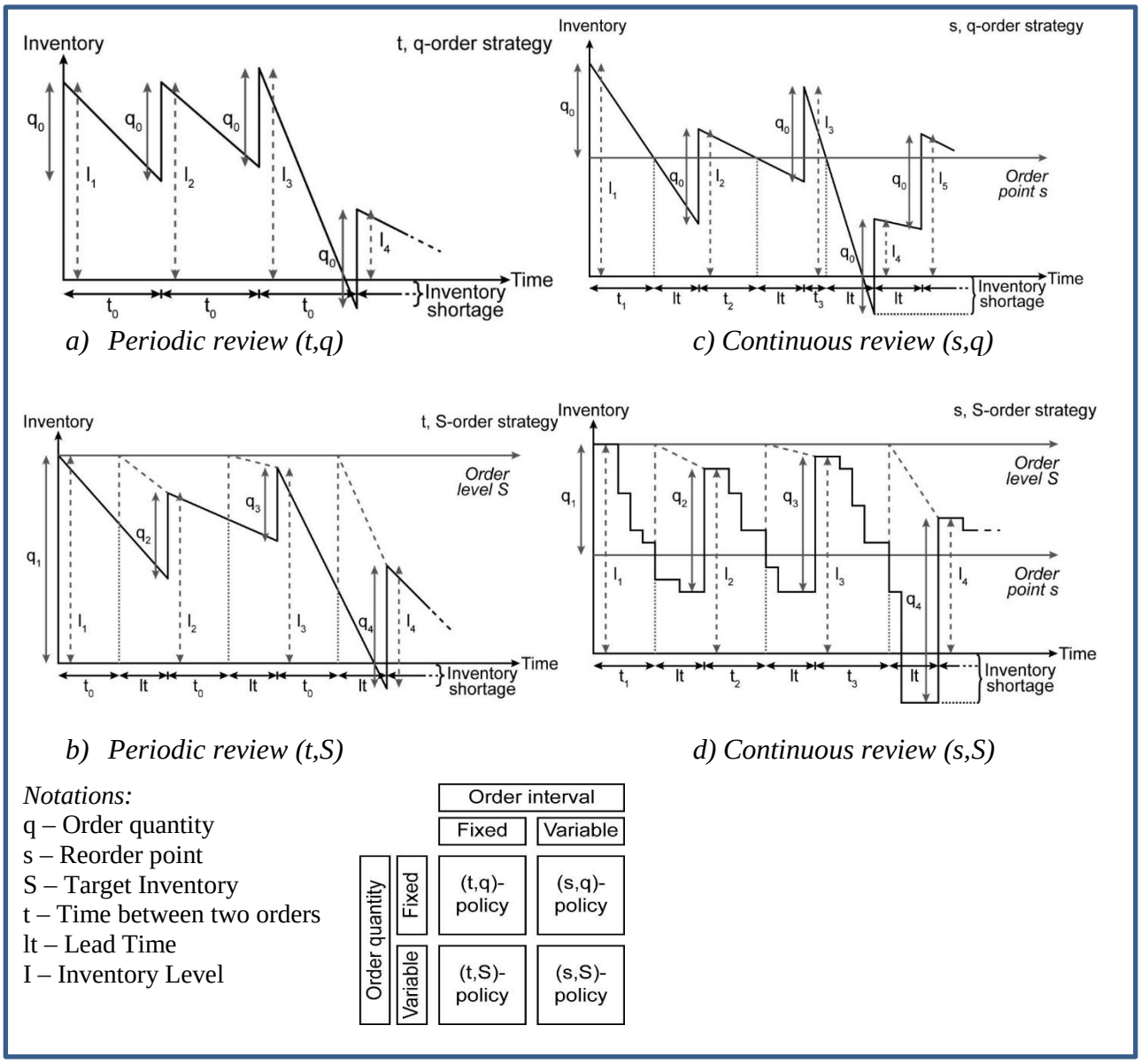


Fig. 68. Inventory control policies (Ivanov et al. 2016)

Five products have different demand and inventory patterns (Table 17 and Fig. 69).

Table 17 Products for analysis

Product Type	Demand	Inventory Level	Items
A	High	High	Engine Oil
B	Medium	Medium	Nuts and Bolts
C	Low	High	Spark Plugs
D	Low	Low	Clutch Plate and brake pads
E	High	Low	Coolant

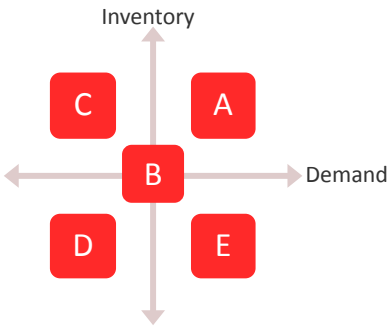


Fig. 69. Products for analysis

The management objective is to perform analysis of different control policies for five products and identify the most efficient policy for each product subject to service level requirements.

14. Model building

With the help of AnyLogic multimethod simulation software, the management problem will be transferred to a simulation model that will allow performing experiments in order to understand basic trade-offs and relations in process flow analysis with consideration of financial and operational KPIs (Key Performance Indicators).

14.1 EOQ model: event-based simulation

In the first step, we learn how to create a simulation model for well-known EOQ problem using events (Rozhkov 2011). For modelling EOQ problem, we create an event "Ordering" to schedule replenishments, two parameters (demand and order quantity), a variable "InventoryLevel" as well as a time plot (Fig. 70).

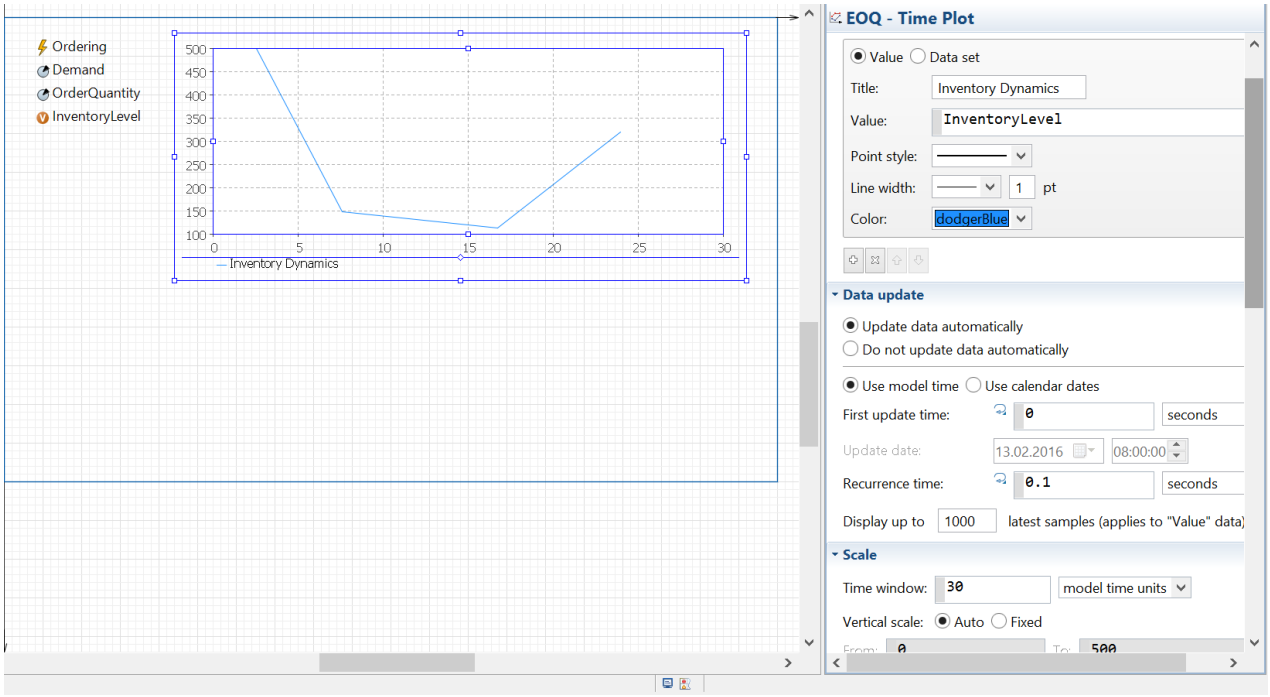


Fig. 70 EOQ Model

In event "Ordering", we describe the logic of EOQ model (Fig. 71).

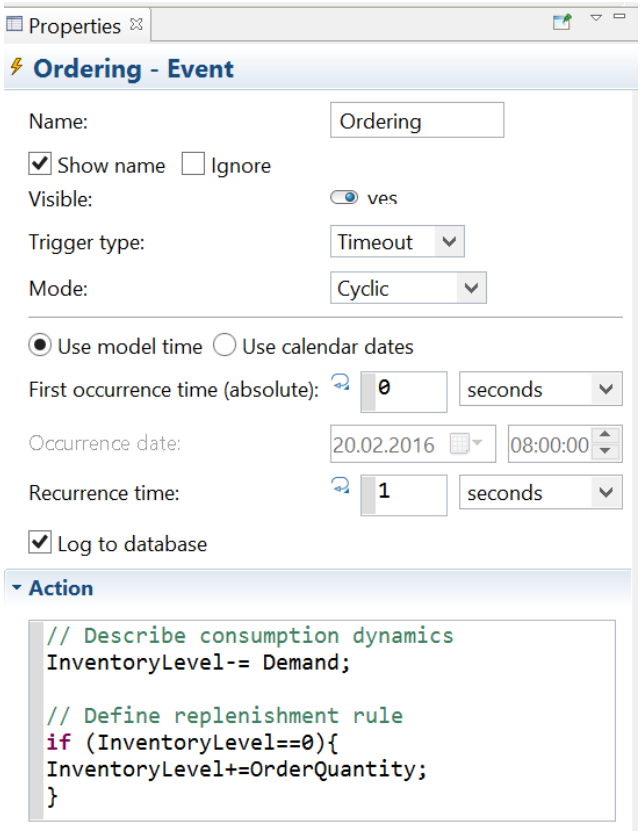


Fig. 71 Event description

The event "Ordering" is a cyclic event with recurrence time 1second (e.g., one day of real life). According to EOQ model, inventory consumption is a linear function from demand that is why we describe inventory dynamics as:

```
InventoryLevel-= Demand;
```

According to EOQ, if inventory level is zero, new order of the same quantity (q) arrives immediately. That is why we describe replenishment rule using Java as follows:

```
if (InventoryLevel==0){
InventoryLevel+=OrderQuantity;
}
```

Setting „Demand“ = 80 and „OrderQuantity“ = 400, we run the experiment and observe inventory dynamics (Fig. 72).

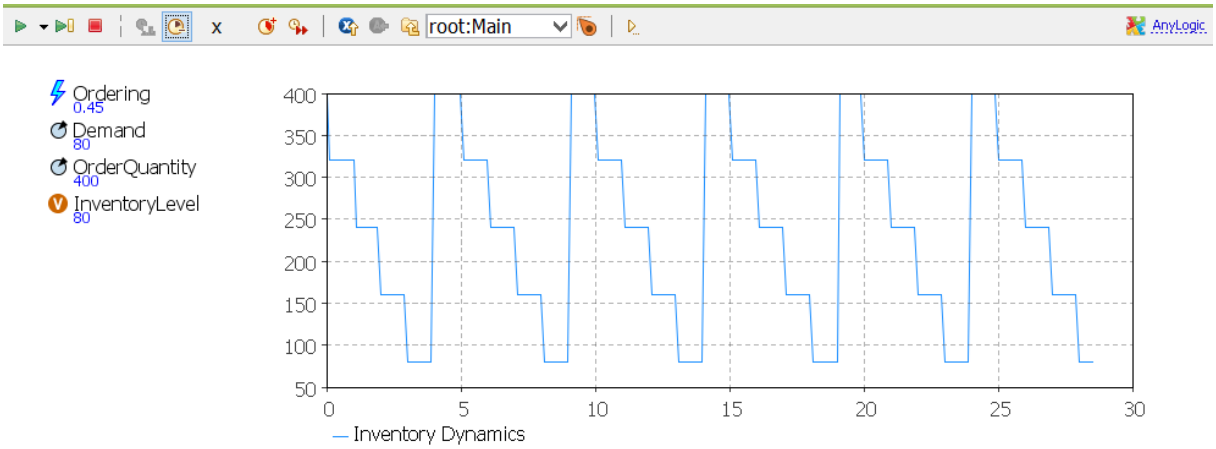


Fig. 72 EOQ inventory dynamics

It can be observed from Fig. 72 that each day inventory level is reduced by demand. At the fifth day inventory level is zero and immediate replenishment is generated. That is why the step from 80 to 0 is missing in the diagram in Fig. 72.

14.2 Modelling stochastic demand: periodic and continuous review policies

Assume that daily demand and/or lead time are normally distributed. We need to define additional parameters for lead time, mean demand, mean lead time, standard demand deviation and standard lead time deviation. In “Ordering” event, we add (Fig. 73):

```
Demand = round( max (normal(DevDemand,MeanDemand),0));
LeadTime = round( max (normal(DevLT,MeanLT),0));
```

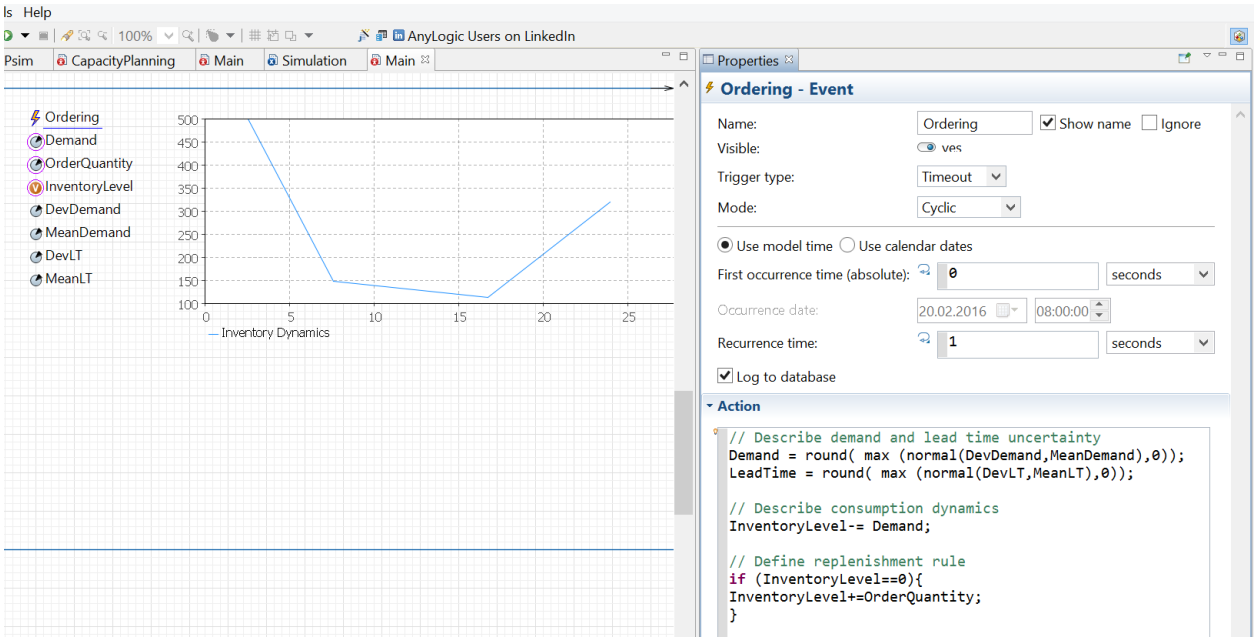


Fig. 73 Stochastic inventory management simulation

First, consider the case without lead time. The simulation result for mean demand = 80 units and standard demand deviation = 16 is presented in Fig. 74.

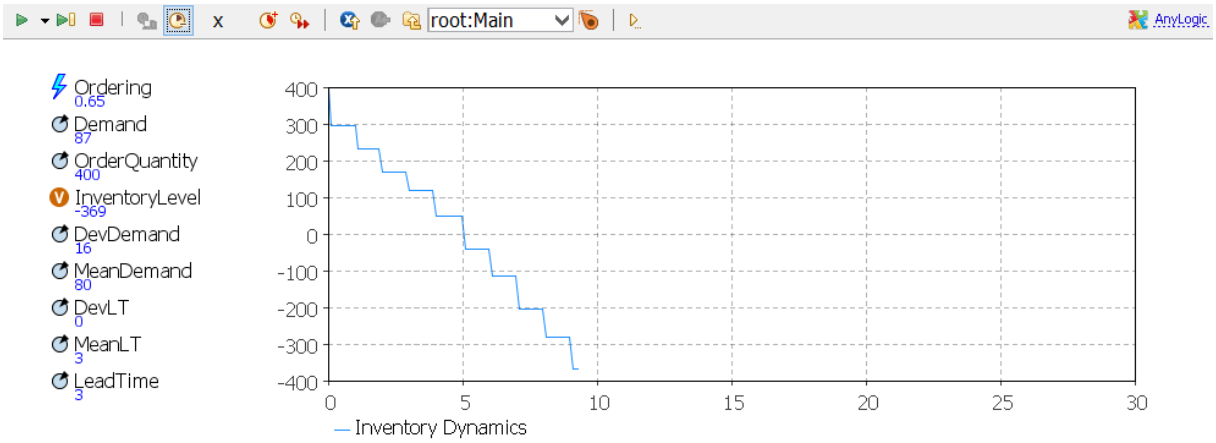


Fig. 74 Stochastic inventory management simulation results

It can be observed from Fig. 74 that no replenishment actions are performed and the model works wrong. The reason is the condition for new replenishment defined in "Ordering" event:

```
// Define replenishment rule
if (InventoryLevel==0){
InventoryLevel+=OrderQuantity;
}
```

Due to demand deviations, the condition InventoryLevel = 0 will not be fulfilled. That is why we have to modify this condition.

For *periodic review system*, this condition is not needed at all since we replenish at fixed dates. Thus we just modify the recurrence time of the event "Ordering" by defining it through the parameter "t_Period" (Fig. 75).

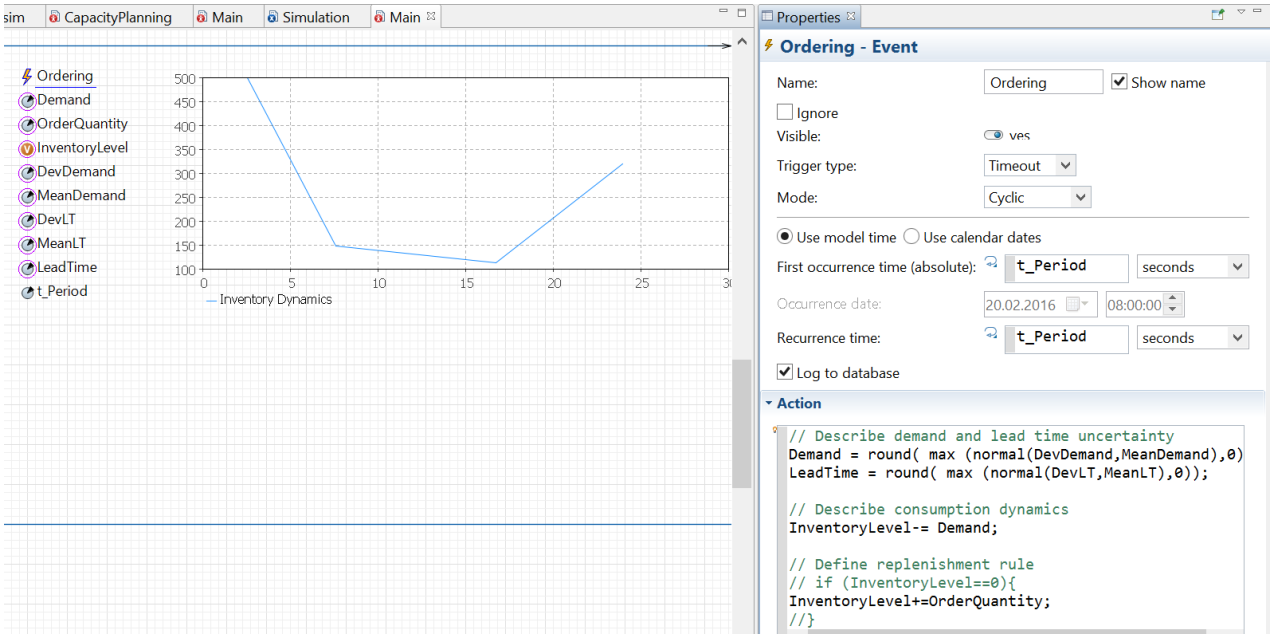


Fig. 75 Periodic review policy *t,q*

Fig. 75 describes periodic review policy *t,q*. For *t,S*-policy, the line

```
InventoryLevel+=OrderQuantity;
```

needs to be re-written as

```
InventoryLevel+=TargetInventory;
```

where TargetInventory is new parameter to define *S*. The value of this parameter can be fixed by a number or defined dynamically by a condition (see further in this chapter).

For *continuous review system*, replenishment rule can be defined with the help of re-order point (ROP). Thus we modify the condition

```
// Define replenishment rule
if (InventoryLevel==0){
InventoryLevel+=OrderQuantity;
}
```

as new condition using new parameter "ROP"

```
// Define replenishment rule
if (InventoryLevel <= ROP){
InventoryLevel+=OrderQuantity;
}
```

As such, *s,q*-policy is defined. The simulation results for ROP = 120 units, mean demand = 80 units and standard demand deviation = 16 are presented in Fig. 76.

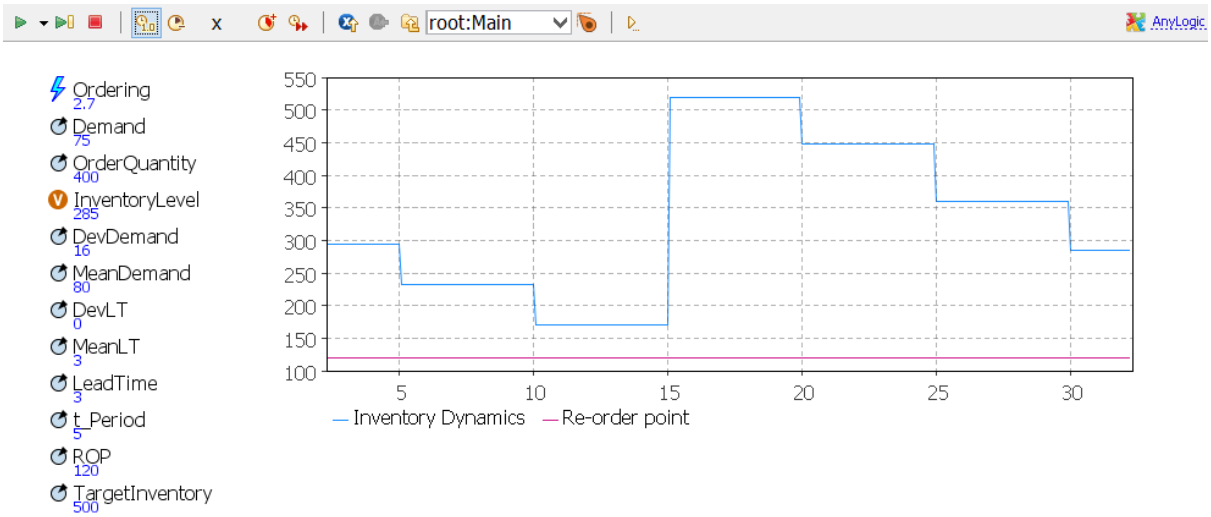


Fig. 76 Continuous review policy *s,q*

In order to define the *s,S*-policy, the line

```
InventoryLevel+=OrderQuantity;
```

needs to be re-written as

```
InventoryLevel+=TargetInventory;
```

where TargetInventory is parameter to define *S*. The value of this parameter can be fixed by a number or defined dynamically by a ROP-dependent condition subject to changes in demand and lead time (see further in this chapter).

14.3 Modelling stochastic demand and lead time: re-order point using Java

The real meaning of re-order point becomes obvious if we extend our models by lead time. In addition we will include safety stock in our model.

For modelling lead time, we define new event "NewOrder" and new variable "OrderReceived" of type "Boolean" (with initial value "false") and re-write replenishment rule in the event „Ordering" as follows (Fig. 77):

```
// Define replenishment rule
if (InventoryLevel<=ROP && !OrderReceived)
{NewOrder.restart( LeadTime );
      OrderReceived = true;
}
```

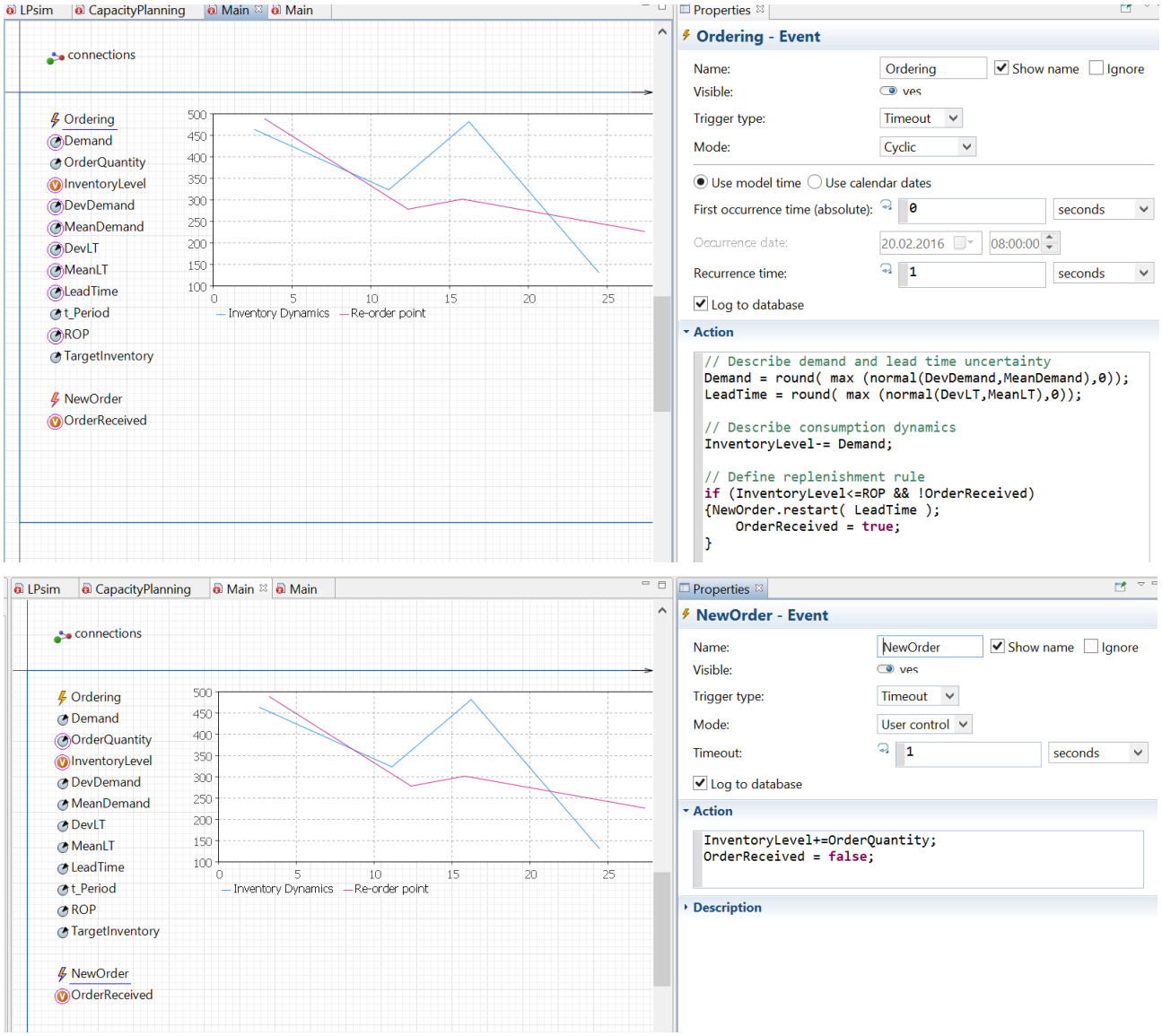


Fig. 77 Continuous review policy *s*,*q* with lead time

In "Ordering", we defined that

```
if (InventoryLevel<=ROP && !OrderReceived)
{NewOrder.restart( LeadTime );
      OrderReceived = true;
```

Here we used two new Java operators:

«&&» - logical "and"

«!» - logical "not"

If inventory level reaches the re-order point, new replenishment order is generated and new order arrived in X-days defined in LeadTime parameter. That is why we let the event "NewOrder" start in X-days defined in LeadTime parameter. At the time of order entry and inventory level increase, the logical variable "OrderReceived" gets the value "false". This means that no other orders are

received by our supplier. Next order will be generated when inventory level reaches the re-order point again. We need the component "`!OrderReceived`" in the condition `if (InventoryLevel<=ROP && !OrderReceived)`, since new order is generated if both re-order point is reached and there are no other orders in-transit.

The simulation result for re-order point = 240 units and mean LT = 3 days (without deviation) is presented in Fig. 78.

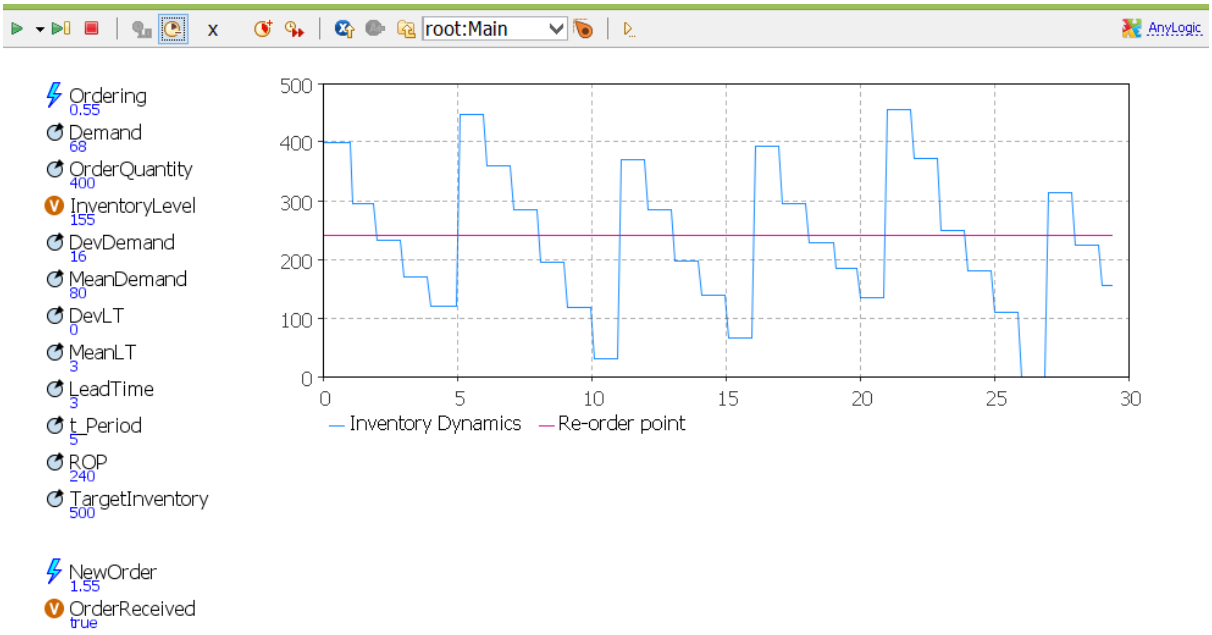


Fig. 78 Simulation of continuous review policy *s*,*q* with lead time

14.4 Inventory holding, ordering and stockout costs

In order to estimate efficiency of different ordering policies, we need to estimate inventory holding, ordering and stockout costs. First, new parameters and variables for these three types of costs are needed. Second, diagrams need to be constructed as bar charts and time plots.

We define three new parameters "HoldingCost", "OrderingCost" and "StockoutCosts" and three new variables "HoldingCosts", "OrderingCosts" and "StockoutCosts". Further three new variables "ServiceLevel", "TotalSales" and "RetailShortage" need to be defined.

In event "Ordering", we add new condition:

```
// Compute sales
if (InventoryLevel > 0)
TotalSales+= Demand;
```

Then we create new event "CostUpdate" as shown in Fig. 79.

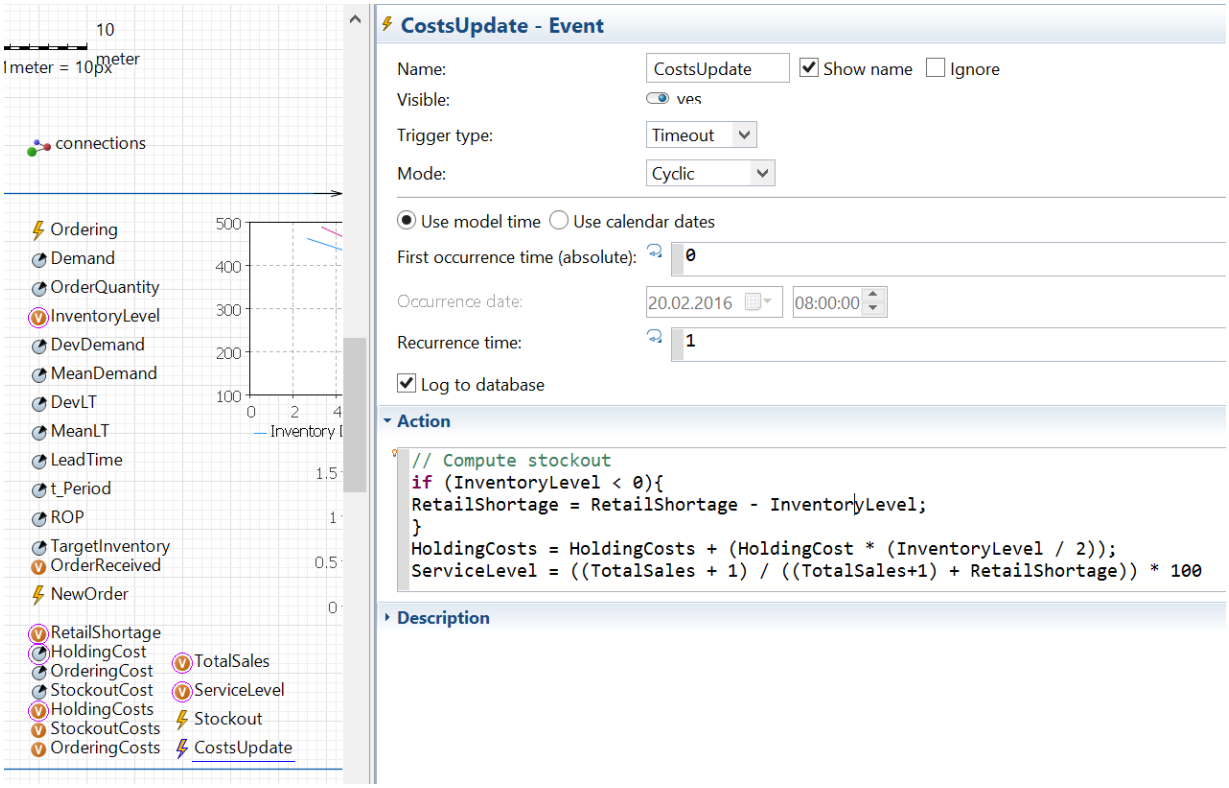


Fig. 79 Costs, shortage and service level update event

Costs, shortage and service level are updated in the event "CostsUpdate". Holding costs is measured for average inventory on-hand (InventoryLevel / 2).

Next, we create event "Stockout" where stockout costs will be computed at the end of planning horizon (in our case it will be 30 days, so we update at point of time = 29 sec) (Fig. 80).

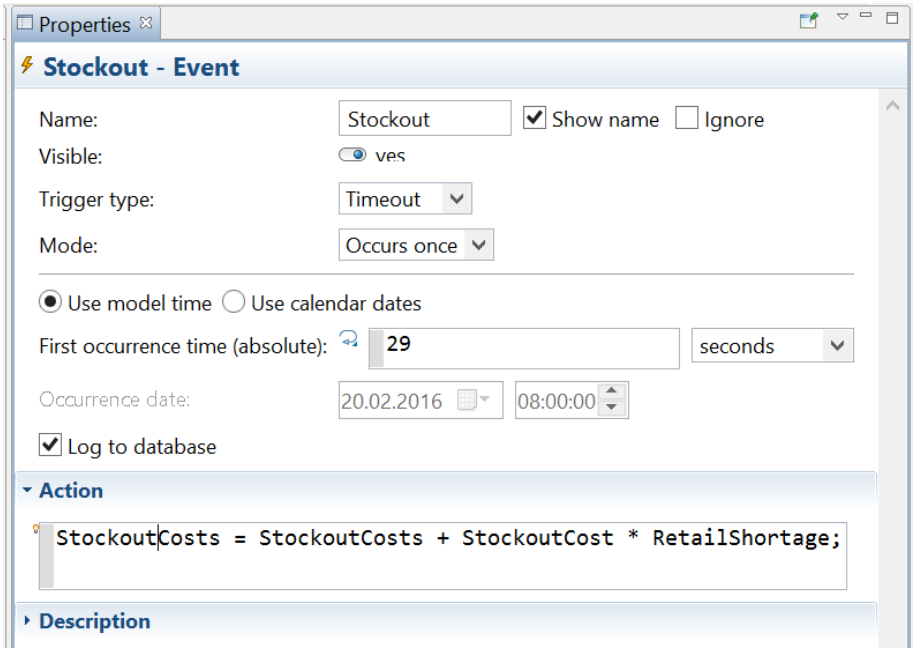


Fig. 80 Stockout costs update event

Finally, we create two time plots for variables "ServiceLevel" and "RetailShortage" and a stack chart for holding, ordering, and stockout costs.

The simulation results for ROP = 120 units, holding costs = $0.1 per day per unit, stockout costs = $0.2 per day per unit, and ordering costs = $10 per order are presented in Fig. 81.

Fig. 81 Efficiency analysis for *s*,*q*-policy

It can be observed that total costs equals $584.8, service level decreases during the experiment and is 59.4% at the end of the month, stockout increases during the experiment and is 838 units at the end of the month.

14.5 Re-order point and safety stock

Let us enhance the analysis from Fig. 81 by computing re-order point with the use of safety stock. For example, 0.05 probability of stock-out corresponds to a 95% service level. In a situation of demand and /or lead time uncertainty**,** *safety inventory* is introduced. In order to calculate service level, Eq. (1) is used:

$$SS = z \times \sigma_{dLT}\,, \qquad (1)$$

where *ss* is safety stock, σ_{dLT} is standard deviation of demand during lead-time and *z* is the number of standard deviations.

Demand deviation can be gleaned, e.g., from the analysis of demand forecasts and actual sales in the past. For example, $\sigma = 1.25MAD$ is a typical value. Z-value can easily be determined (see Table 18).

Table 18. Table of normal distribution

Z	0.00	0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09
0.0	0.5000	0.5040	0.5080	0.5120	0.5160	0.5199	0.5239	0.5279	0.5319	0.5359
0.1	0.5398	0.5438	0.5478	0.5517	0.5557	0.5596	0.5636	0.5675	0.5714	0.5753
0.2	0.5793	0.5832	0.5871	0.5910	0.5948	0.5987	0.6026	0.6064	0.6103	0.6141
0.3	0.6179	0.6217	0.6255	0.6293	0.6331	0.6368	0.6406	0.6443	0.6480	0.6517
0.4	0.6554	0.6591	0.6628	0.6664	0.6700	0.6736	0.6772	0.6808	0.6844	0.6879
0.5	0.6915	0.6950	0.6985	0.7019	0.7054	0.7088	0.7123	0.7157	0.7190	0.7224
0.6	0.7257	0.7291	0.7324	0.7357	0.7389	0.7422	0.7454	0.7486	0.7517	0.7549
0.7	0.7580	0.7611	0.7642	0.7673	0.7704	0.7734	0.7764	0.7794	0.7823	0.7852
0.8	0.7881	0.7910	0.7939	0.7967	0.7995	0.8023	0.8051	0.8078	0.8106	0.8133
0.9	0.8159	0.8186	0.8212	0.8238	0.8264	0.8289	0.8315	0.8340	0.8365	0.8389
1.0	0.8413	0.8438	0.8461	0.8485	0.8508	0.8531	0.8554	0.8577	0.8599	0.8621
1.1	0.8643	0.8665	0.8686	0.8708	0.8729	0.8749	0.8770	0.8790	0.8810	0.8830
1.2	0.8849	0.8869	0.8888	0.8907	0.8925	0.8944	0.8962	0.8980	0.8997	0.9015
1.3	0.9032	0.9049	0.9066	0.9082	0.9099	0.9115	0.9031	0.9147	0.9162	0.9177
1.4	0.9192	0.9207	0.9222	0.9236	0.9251	0.9265	0.9279	0.9292	0.9306	0.9319
1.5	0.9332	0.9345	0.9357	0.9370	0.9382	0.9394	0.9406	0.9418	0.9429	0.9441
1.6	0.9452	0.9463	0.9474	0.9484	0.9495	0.9505	0.9515	0.9525	0.9535	0.9545
1.7	0.9554	0.9564	0.9573	0.9582	0.9591	0.9599	0.9608	0.9616	0.9625	0.9633
1.8	0.9641	0.9649	0.9656	0.9664	0.9671	0.9678	0.9686	0.9693	0.9699	0.9706
1.9	0.9713	0.9719	0.9726	0.9732	0.9738	0.9744	0.9750	0.9756	0.9761	0.9767
2.0	0.9772	0.9778	0.9783	0.9788	0.9793	0.9798	0.9803	0.9808	0.9812	0.9817
2.1	0.9821	0.9826	0.9830	0.9834	0.9838	0.9842	0.9846	0.9850	0.9854	0.9857
2.2	0.9861	0.9864	0.9868	0.9871	0.9875	0.9878	0.9881	0.9884	0.9887	0.9890
2.3	0.9893	0.9896	0.9898	0.9901	0.9904	0.9906	0.9909	0.9911	0.9913	0.9916
2.4	0.9918	0.9920	0.9922	0.9924	0.9927	0.9929	0.9931	0.9932	0.9934	0.9936
2.5	0.9938	0.9940	0.9941	0.9943	0.9945	0.9946	0.9948	0.9949	0.9951	0.9952
2.6	0.9953	0.9955	0.9956	0.9957	0.9958	0.9960	0.9961	0.9962	0.9963	0.9964
2.7	0.9965	0.9966	0.9967	0.9968	0.9969	0.9970	0.9971	0.9972	0.9973	0.9974
2.8	0.9974	0.9975	0.9976	0.9977	0.9977	0.9978	0.9979	0.9979	0.9980	0.9981
2.9	0.9981	0.9982	0.9982	0.9983	0.9984	0.9984	0.9985	0.9985	0.9986	0.9986

For example, z=1.65 for service level 95%, z=2.33 for service level 99%, and z=1.28 for service level 90%. If standard deviation of demand during lead time (L) is 10, then safety stocks equals 1.65 x 10 = 16.5; 2.33 x 10 = 23.3; 1.28 x 10 = 12.8. We can observe that the increase of service level from 90% to 99% results in doubling the safety stock.

The inclusion of safety stock changes the calculation of ROP (see Eq. 2):

$$ROP = d * L + ss \quad (2)$$

In computing ROP, three situations are possible:

- daily distribution of demand is given (i.e., demand is variable) and lead time is constant (Eq. 3)
- daily demand is constant and lead time is variable (Eq. 4),
- both demand and lead time are variable (Eq. 5).

$$ROP = d \times L + z \times \sigma_d \times \sqrt{L} \quad (3)$$

$$ROP = d \times L + z \times d \times \sigma_L \quad (4)$$

$$ROP = d \times L + z\sqrt{L \times \sigma_d^2 + d^2 \times \sigma_L^2} \quad (5)$$

Let us compute ROP for the case from Fig. 81 subject to service level 95% using Eq. (3):

ROP = 80 * 3 + 1.65 * 18 * $\sqrt{3}$ = 240 + 52 = 292 units, where 52 units is safety stock. For computing order quantity, we use EOQ formula:

$$q = \sqrt{\frac{2 \times 2400 \times 10}{3}} = 127 \text{ units} * 3 \text{ days} = 381 \text{ units}$$

The simulation results with new s =292 units and q=381 units are presented in Fig. 82.

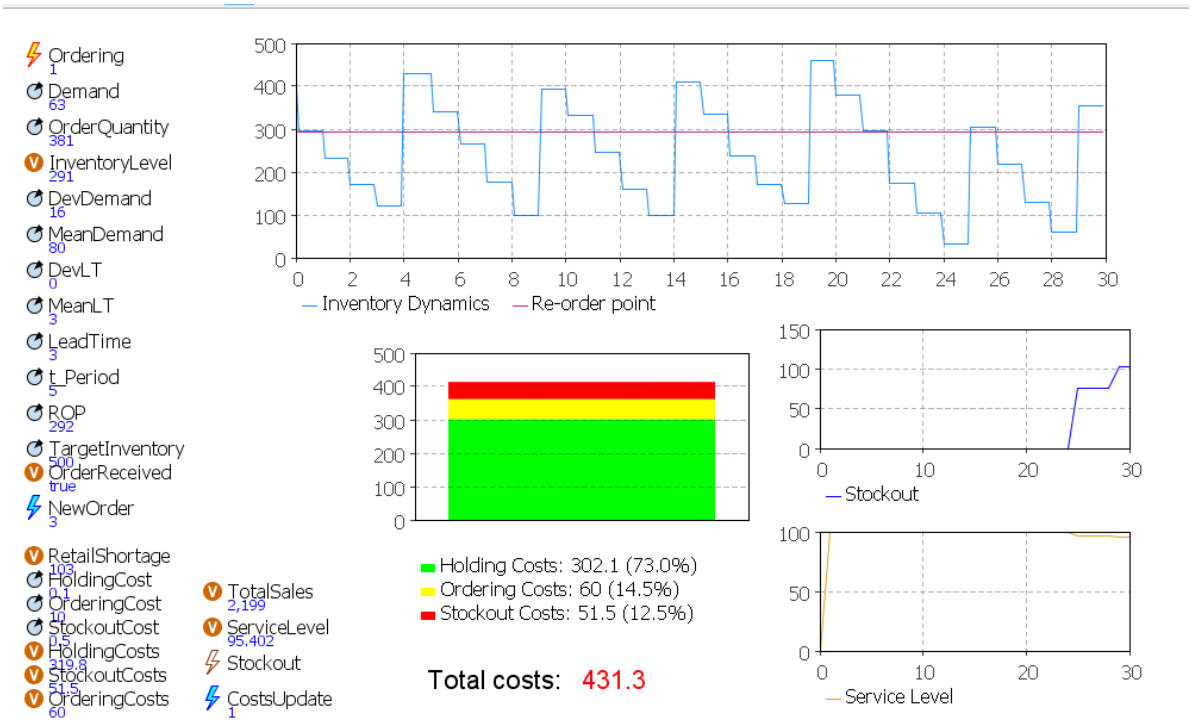


Fig. 82 Efficiency analysis for optimized *s*,*q*-policy

It can be observed from Fig. 82 that total costs equals $431.3 (as compared to $584.8 in the initial situation), service level increased from 59.4% to 95% at the end of the month, stockout decreases from 838 units to 103 units in total. In order to avoid stockouts, we need to increase order quantity. One option is to consider in the *q* computation not the mean demand, but e.g., d = $d_{mean} + \sigma_d$) = 80 + 18 = 98 units * 30 days = 2940 units:

$$q = \sqrt{\frac{2 \times 2940 \times 10}{3}} = 140 \text{ units} * 3 \text{ days} = 420 \text{ units}$$

Results are presented in Fig. 83.

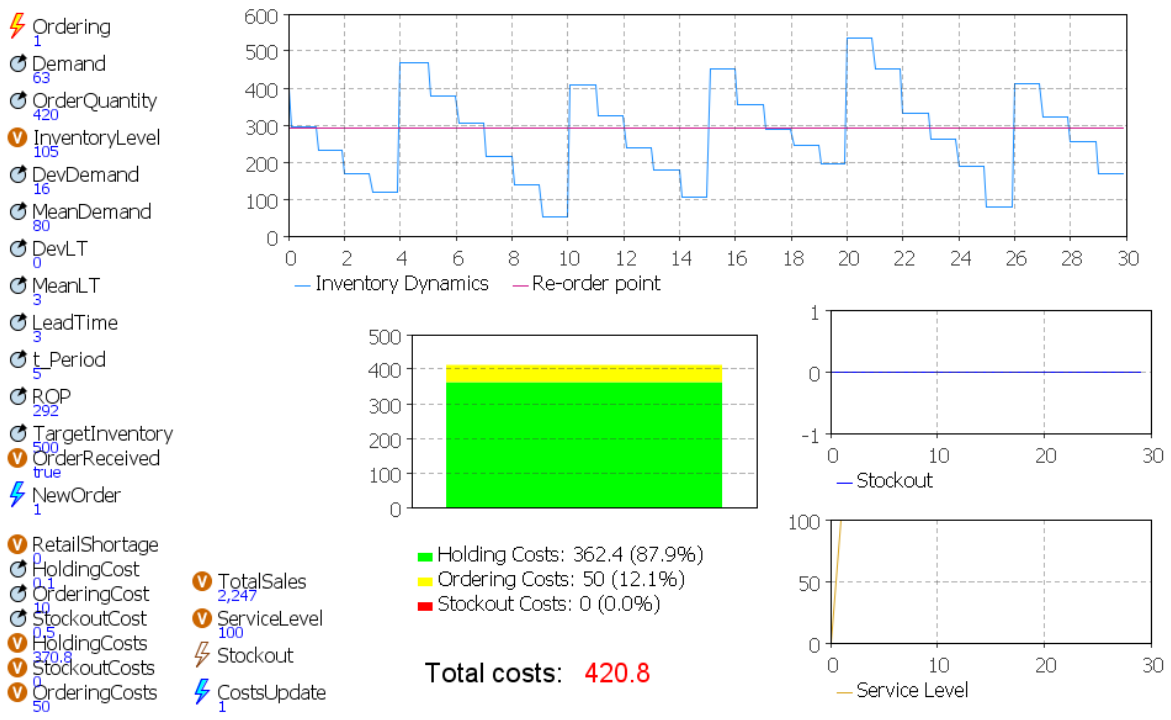


Fig. 83 Efficiency analysis for optimized *s*,*q*-policy

It can be observed from Fig. 83 that total costs equals $420.8 (instead of $431.3 after first optimization), service level increased from 95% to 100% at the end of the month, stockout decreases from 103 units in total to zero.

15. Experiments and managerial insights

15.1 Preparing experiment: using Action Charts

In order to decouple inventory replenishment logic for different control policies we introduce action charts for continuous review policies instead of placing all the codes in event "Ordering".

Action charts are created in "Palette → Actioncharts" (Fig. 84).

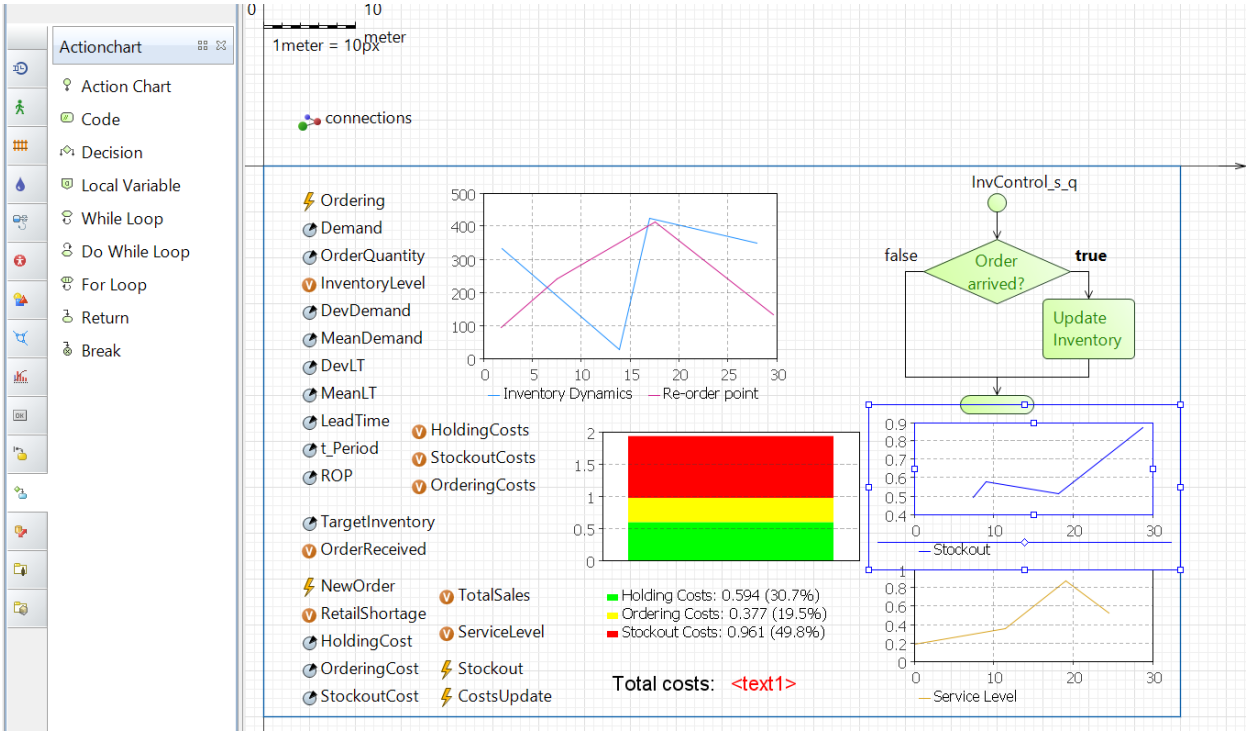


Fig. 84 Action chart for *s*,*q*-policy

In action chart, we can define a condition for activating the action chart and the code (action) to be performed if the condition is true or false. In our case, we define a condition "Order arrived?" as

```
InventoryLevel <= ROP && !OrderReceived
```

and the code to be executed for "Update Inventory" if the condition "Order arrived?" is true as

```
NewOrder.restart( LeadTime );
      OrderReceived = true;
```

Previously we had this definition of replenishment rule directly in the event "Ordering" (cf. Fig. 77):

```
// Define replenishment rule
if (InventoryLevel<=ROP && !OrderReceived)
{NewOrder.restart( LeadTime );
      OrderReceived = true;
}
```

Usage of actions charts allows decentralizing different replenishment actions and therefore decoupling inventory dynamics in different control policies. Action charts may have much more complex structure with numerous conditions, branches and actions.

For our case, we create two action charts for *s*,*q*- and *s*,*S*-policies calling already existed event "NewOrder" for *s*,*q*-policy and new event "NewOrder1" for *s*,*S*-policy. The event "NewOrder1" is described as:

```
InventoryLevel1+=s_TargetInventory + InventoryLevel1;
OrderingCosts1 = OrderingCosts1 + OrderingCost;
OrderReceived1 = false;
```

For *t,q*- and *t,S*-policies no action charts are needed. We just create new event t_OrderingPolicy with the following code:

```
// Periodic Review - Fixed Quantity Model
InventoryLevel2+=OrderQuantity;
OrderingCosts2 = OrderingCosts2 + OrderingCost;

// Periodic Review - Target Inventory Model
InventoryLevel3+=(t_TargetInventory - InventoryLevel3);
OrderingCosts3 = OrderingCosts3 + OrderingCost;
```

15.2 Preparing experiment: dynamic target inventory

In §15.1, we introduced new parameters s_TargetInventory and t_TargetInventory in order to define target inventory. The target inventory can be fixed or dynamic. If we fix the value of target inventory, this value will be used over the whole experiment. From management point of view, such a policy can be quite efficient in regard to fixed warehouse area and transportation planning. However, if demand changes over the considered standard deviation, shortages can occur. The same holds for re-order point.

That is why it may become sensible to make change both re-order point and target inventory dynamically. One possible option is to establish a relation between re-order point and target inventory. Since we have different inventory dynamics in *s,q*- and *s,S*-policies, we introduce additional parameter ROP1 for *s,S*-policy whilst ROP will be used for *s,q*-policy.

We also introduce new parameters s_S_inventory that will keep the current inventory level (variable InventoryLevel1) at the time of new order launching.
Now we update the code "UpdateInventory" in action chart InvControl_s_S as follows:

```
s_S_Inventory = InventoryLevel1;
s_TargetInventory = (ROP1-s_S_Inventory) + ((MeanDemand + DevDemand)*LeadTime);
NewOrder1.restart( LeadTime );
       OrderReceived1 = true;
```

The rule

```
s_TargetInventory = (ROP1-s_S_Inventory) + ((MeanDemand + DevDemand)*LeadTime);
```

for setting target inventory in *s,S*-policy is our subjective estimation. It can be adjusted during experiments. For theory on defining this parameter analytically, we recommend literature on inventory control (Zipkin 2000, Axsaeter 2010).

Note that we also added new logical variable OrderReceived1 for *s,S*-policy to decouple *s,q*- and *s,S*-policies.

The event "Ordering" has now the following form (Fig. 85).

s_OrderingPolicy - Event

Log to database

Action

```
Demand = round( max (normal(DevDemand,MeanDemand),0));
LeadTime = round( max (normal(DevLT,MeanLT),0));

// Describe consumption dynamics
InventoryLevel-= Demand;
InventoryLevel1-= Demand;
InventoryLevel2-= Demand;
InventoryLevel3-= Demand;

// Continuous Review - Fixed Quantity Model
InvControl_s_q();
// Continuous Review - Target Inventory Model
InvControl_s_S();


// Dynamic re-order point
ROP = MeanDemand * LeadTime + z * DevDemand * sqrt(LeadTime);
ROP1 = MeanDemand * LeadTime + z * DevDemand * sqrt(LeadTime);

// Compute stockout
if (InventoryLevel < 0)
RetailShortage = RetailShortage - InventoryLevel;
if (InventoryLevel1 < 0)
RetailShortage1 = RetailShortage2 - InventoryLevel1;
if (InventoryLevel2 < 0)
RetailShortage2 = RetailShortage2 - InventoryLevel2;
if (InventoryLevel3 < 0)
RetailShortage3 = RetailShortage3 - InventoryLevel3;

// Write data
fileInventory.println(InventoryLevel);
```

Fig. 85 Event "Ordering"

For dynamic updates of ROPs, we use Eq. (3). For other cases, these lines need to be changed according to Eqs (4) or (5). Due to four policies, we need to separate logic and output analysis views (Figs 86 and 87).

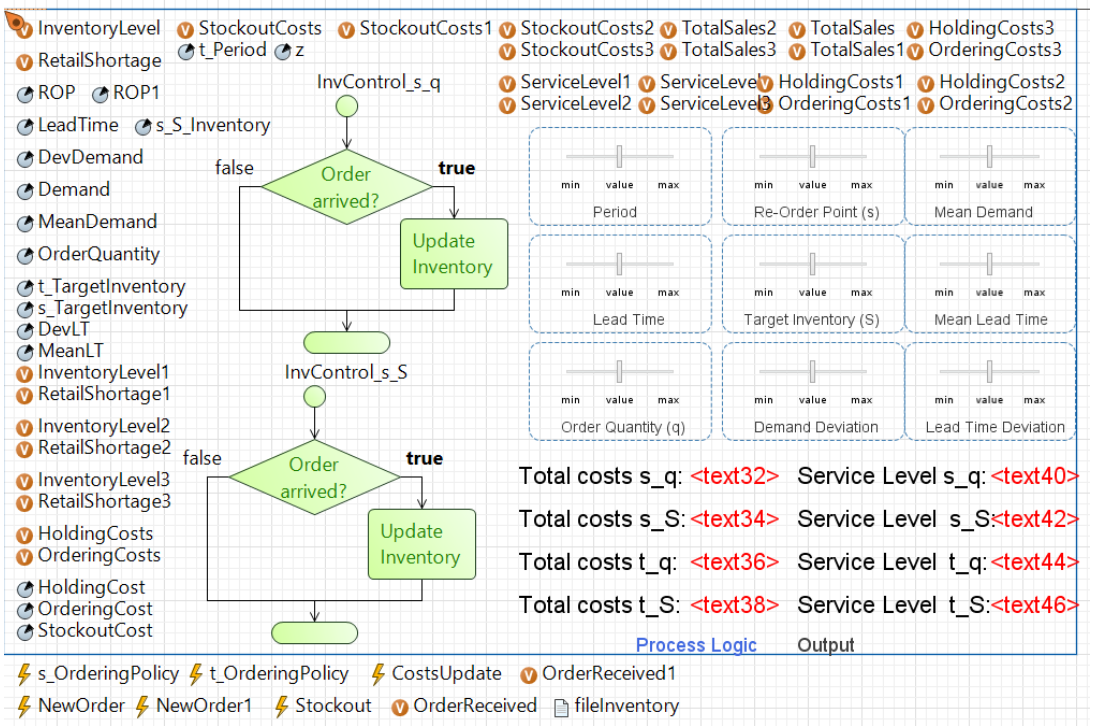


Fig. 86 Logic view

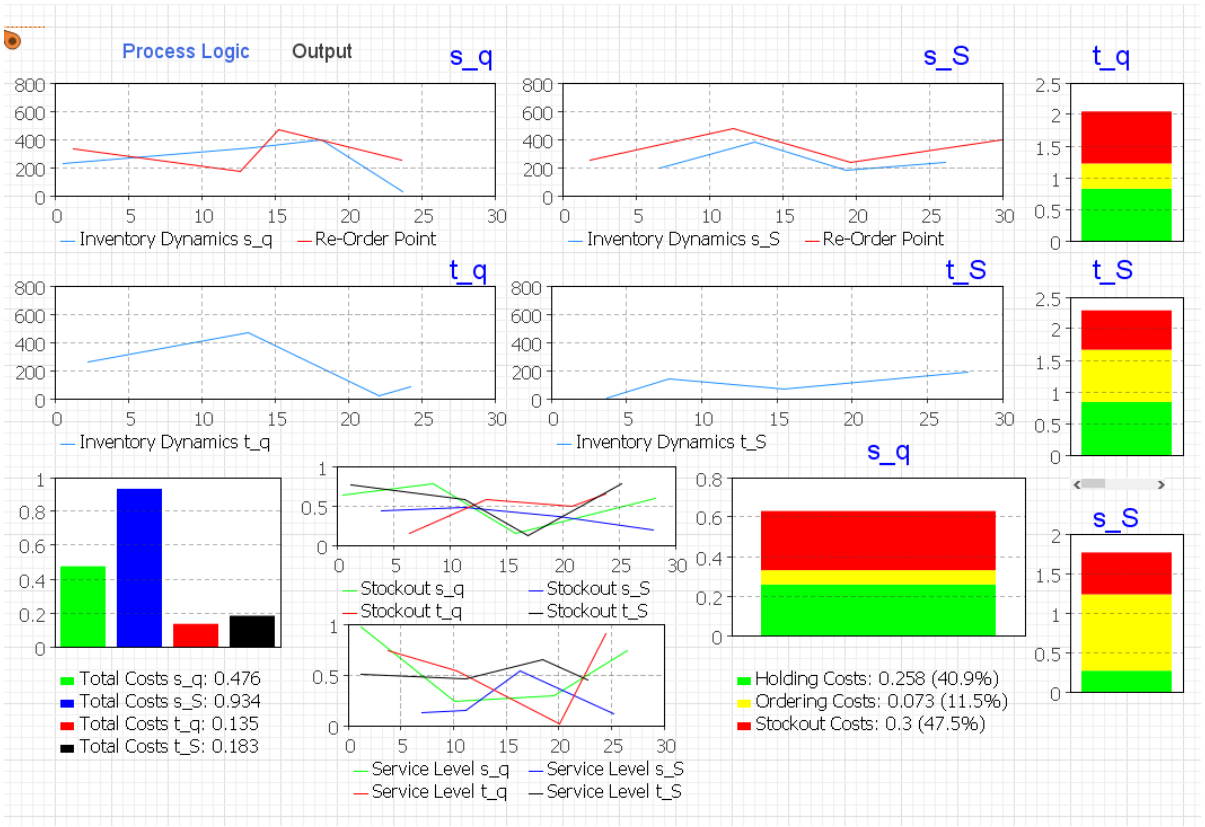


Fig. 87 Analysis view

Recall case study from §14.1. Let us specify data (Table 19).

Table 19 Products for analysis

Product Type	Daily Demand	Inventory Level	Items
A	200	1000	Engine Oil
B	120	1000	Nuts and Bolts
C	80	1000	Spark Plugs

We are going to compare four inventory control policies for each product. We will compare for all products:

- *t*,*q*-policy with fixed period and quantity
- *t*,*S*-policy with fixed period and target inventory
- *s*,*q*-policy with dynamic re-order point and fixed order quantity
- *s*,*S*-policy with dynamic re-order point and target inventory

In all experiments, demand will be assumed to be variable and lead time will be assumed to be constant. z-value will be 1.65 (according to service level = 95%). Holding costs is $0.1 per unit per day, ordering costs is $10 per order, and stockout costs is $0.2 per day per unit.

15.3 Experiment 1 for engine oil: impact of demand dynamics

The following input data is used (Table 20).

Table 20 Input data for experiment 1

Parameter	Value
Mean daily demand, units	200
Standard deviation of demand per day, units	40
Lead time, days	3
Fixed replenishment period, days	5
Order quantity, units	1,000
t_Target Inventory	1,000
s_Target Inventory	`(ROP1-s_S_Inventory) + ((MeanDemand + DevDemand)*LeadTime)`
Initial inventory level	1,200

The simulation results are presented in Figs 88 and 89.

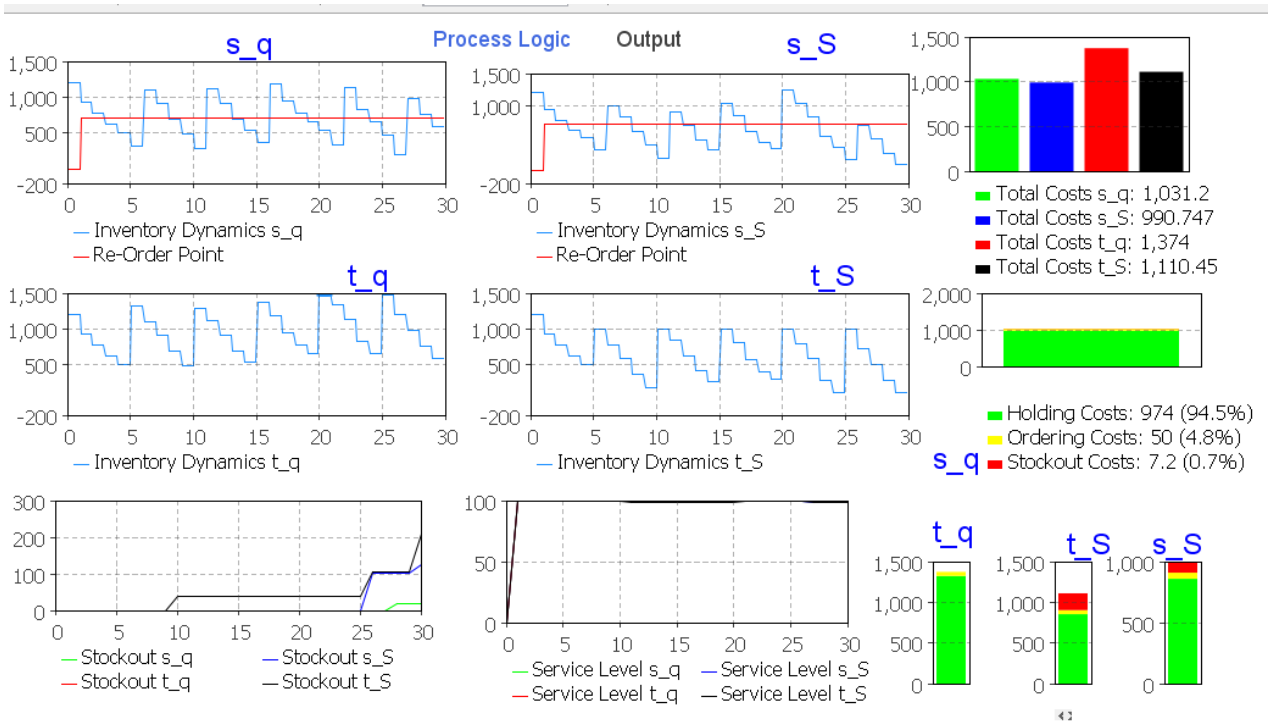


Fig. 88 Experiment #1: Analysis view

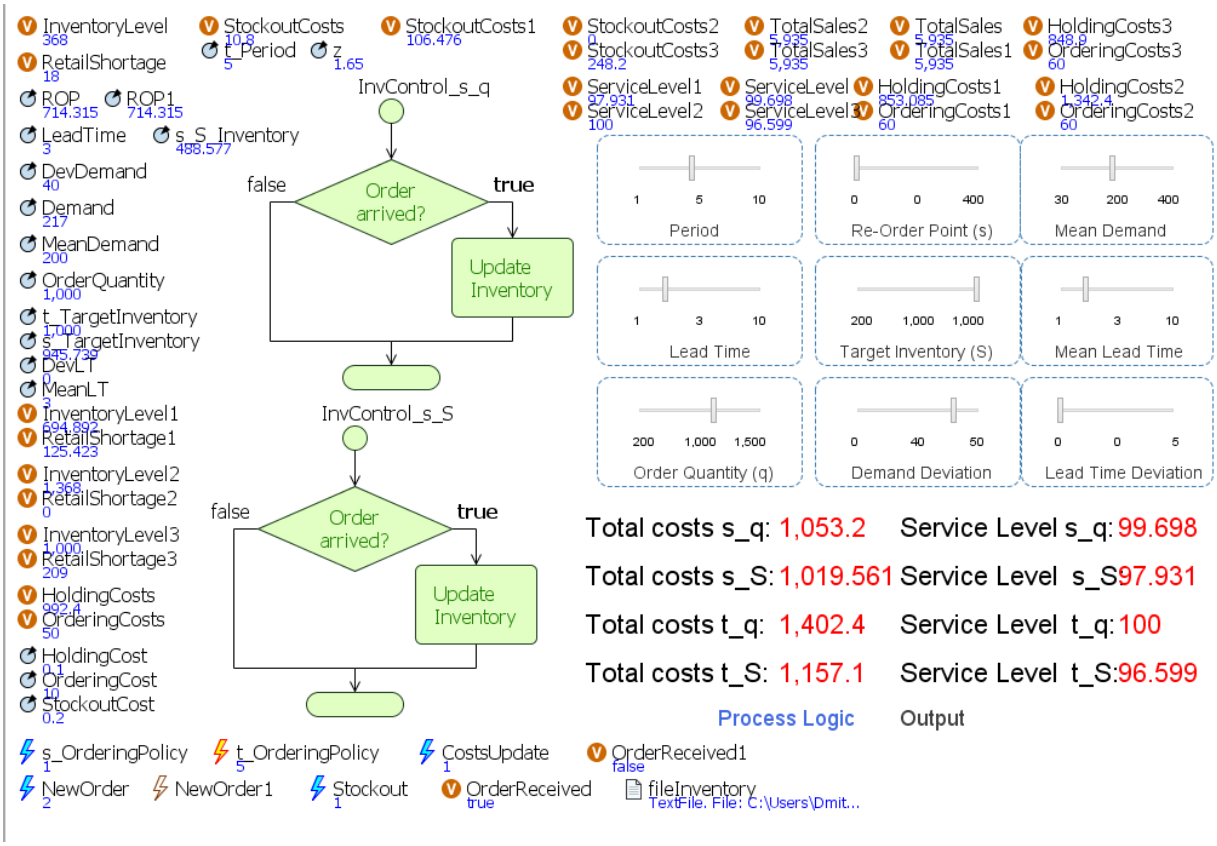
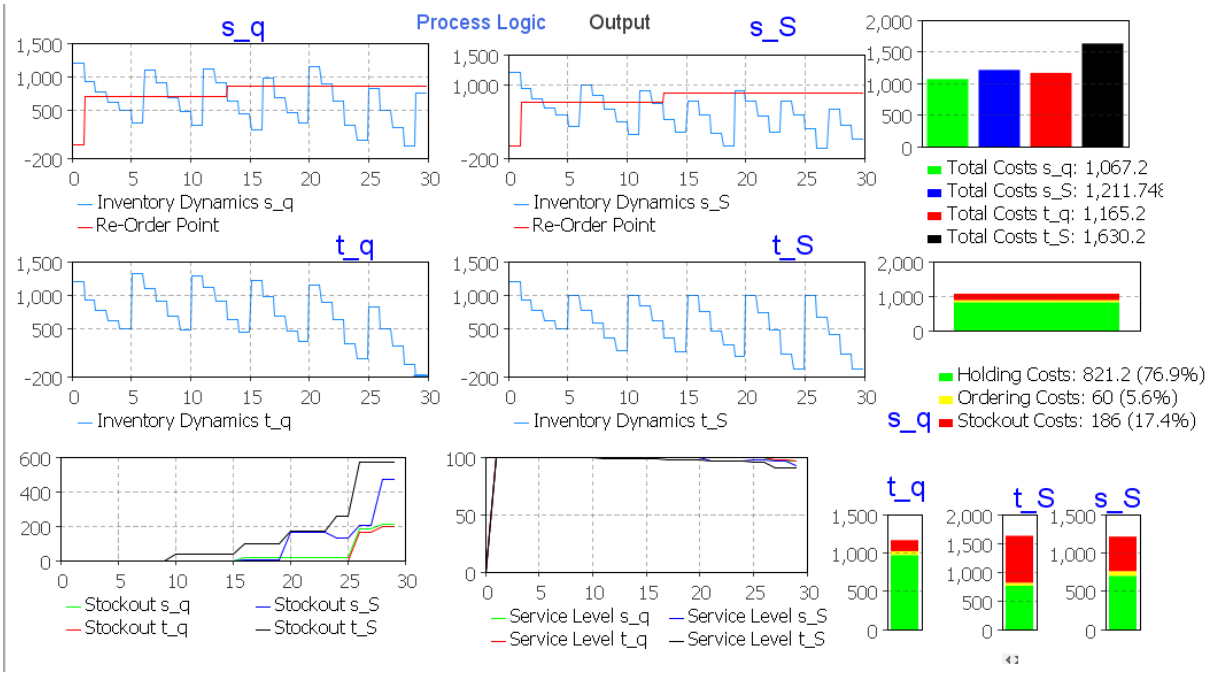


Fig. 89 Experiment #1: Logic view

It can be observed from Figs 88 and 89 that *s,S* is the most efficient policy with costs of $990.8 while periodic review policy *t,q* provides 100% service level but at the same time has the highest costs of $1,374.

Now assume unpredictable change in demand: after 10 days mean demand increases from 200 to 250 units since our competitor in the neighbourhood is closed for the rest of the month due to unforeseen technical problems.

The simulation results are presented in Fig 90.



Service Level s_q: 92.906

Service Level s_S90.97

Service Level t_q: 86.227

Service Level t_S:85.72

Fig. 90 Experiment #1 with changed demand

It can be observed from Fig. 90 that different policies performed differently in regard to the considered demand change. Table 21 compares the results.

Table 21 Result comparison for experiment 1 with and without demand changes

Policy	**Costs**		**Service Level**	
	No changes in demand	Change in demand	No changes in demand	Change in demand
t,q	1,374	1,165	**100**	86.2
t,S	1,110	1,630	96.6	85.7
s,q	1,031	**1,067**	99.7	**92.9**
s,S	**991**	1,212	97.9	90.1

It can be observed from Fig. 90 and Table 21 that continuous review policies with dynamic changes in re-order point and target inventory allow reacting faster to demand fluctuations.

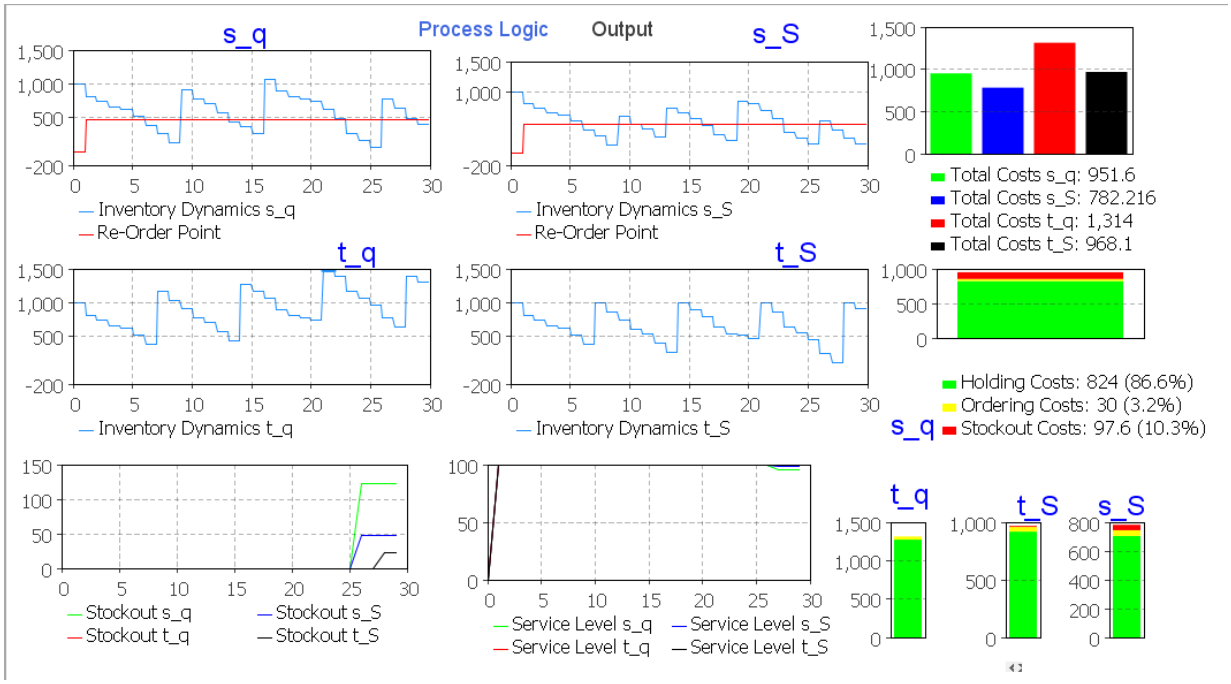
15.4 Experiment 2 for nuts and bolts: impact of lead time dynamics

The following input data is used (Table 22).

Table 22 Input data for experiment 2

Parameter	Value
Mean daily demand, units	120
Standard deviation of demand per day, units	40
Lead time, days	3
Fixed replenishment period, days	7
Order quantity, units	900
t_Target Inventory	1,000
s_Target Inventory	`(ROP1-s_S_Inventory) + ((MeanDemand + DevDemand)*LeadTime)`
Initial inventory level	1,000

The simulation results are presented in Fig. 91.



Service Level s_q: 96.121

Service Level s_S 98.467

Service Level t_q: 100

Service Level t_S: 99.245

Fig. 91 Experiment #2

It can be observed from Fig. 91 that *s,S* is the most efficient policy with costs of $782.2 while periodic review policy *t,q* provides 100% service level but at the same time has the highest costs of $1,314.

Now assume unpredictable change in lead time: after 10 days mean lead time increases from three days six days because of a natural disaster that interrupted the transportation route for the rest of the month. This affects continuous review policies only since the supplier in periodic review policies has a period of seven days between two deliveries.

The simulation results are presented in Fig 92.

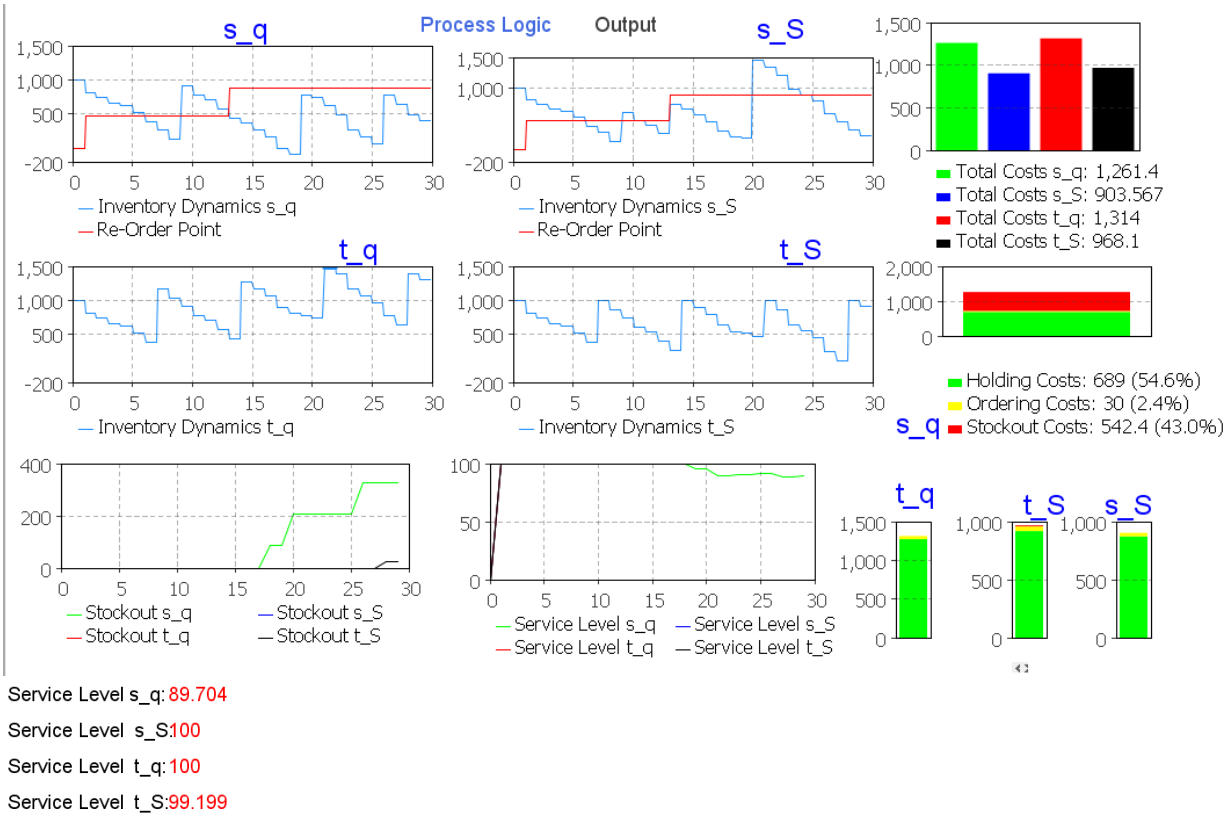


Fig. 92 Experiment #2 with lead time change

It can be observed from Fig. 92 that different policies performed differently in regard to the considered lead time change. Table 23 compares the results.

Table 23 Result comparison for experiment 2 with and without lead time changes

Policy	**Costs**		**Service Level**	
	No changes in lead time	Change in lead time	No changes in lead time	Change in lead time
t,q	1,314	1,314	**100**	**100**
t,S	968	968	99.2	99.2
s,q	952	1,261	96.1	89.7
s,S	**782**	**904**	98.5	**100**

It can be observed from Fig. 92 and Table 23 that target inventory policies are more robust in regard to lead time fluctuations as compared to order quantity-driven policies.

Consider results from first experiment in this group (without lead time changes, Fig. 91). What can be improve in different policies?

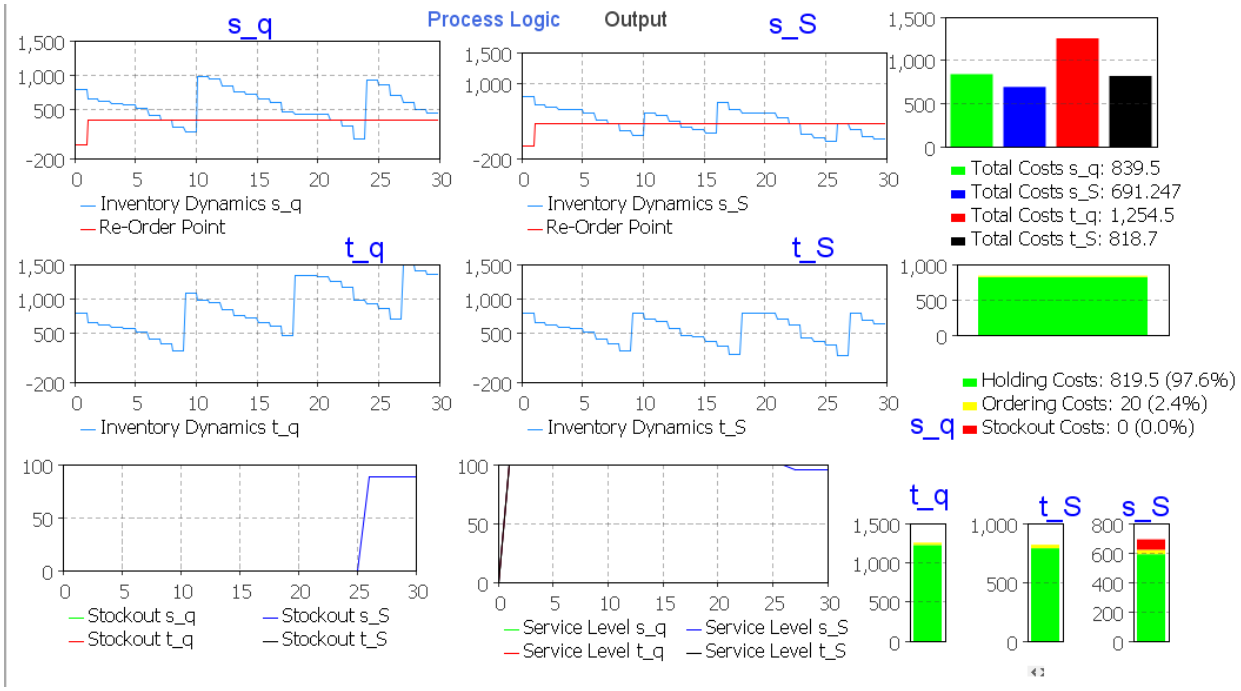
15.5 Experiment 3 for nuts and bolts: impact of order quantity

The following input data is used (Table 24).

Table 24 Input data for experiment 3

Parameter	Value
Mean daily demand, units	80
Standard deviation of demand per day, units	10
Lead time, days	3
Fixed replenishment period, days	9
Order quantity, units	900
t_Target Inventory	800
s_Target Inventory	`(ROP1-s_S_Inventory) + ((MeanDemand + DevDemand)*LeadTime)`
Initial inventory level	800

The simulation results are presented in Fig. 93.



Service Level s_q: 100

Service Level s_S 95.879

Service Level t_q: 100

Service Level t_S: 100

Fig. 93 Experiment #3

It can be observed from Fig. 93 that *s*,*S* is the most efficient policy with costs of $691.2 while all other policies provides 100% service level (*s*,*S*-policy performs with 95.9% service level).

Let us analyse possible reasons for efficiency and service level of different policies. The following changes can be suggested:

- *t*,*q*-policy has to high average inventory on-hand → reduce order quantity or increase the re-order period; we will reduce order quantity from 900 to 700 units.
- *t*,*S*-policy performs well → no changes

- *s,q*-policy has to high average inventory on-hand → reduce order quantity or increase the lead time; we will reduce order quantity from 900 to 700 units.
- *s,S*-policy performs with shortage → increase re-order point or target inventory; we will increase target inventory as follows: `s_TargetInventory = (ROP1-s_S_Inventory) + (1.5*(MeanDemand + DevDemand)*LeadTime) ;`

The simulation results for the changed parameters are presented in Fig 94.

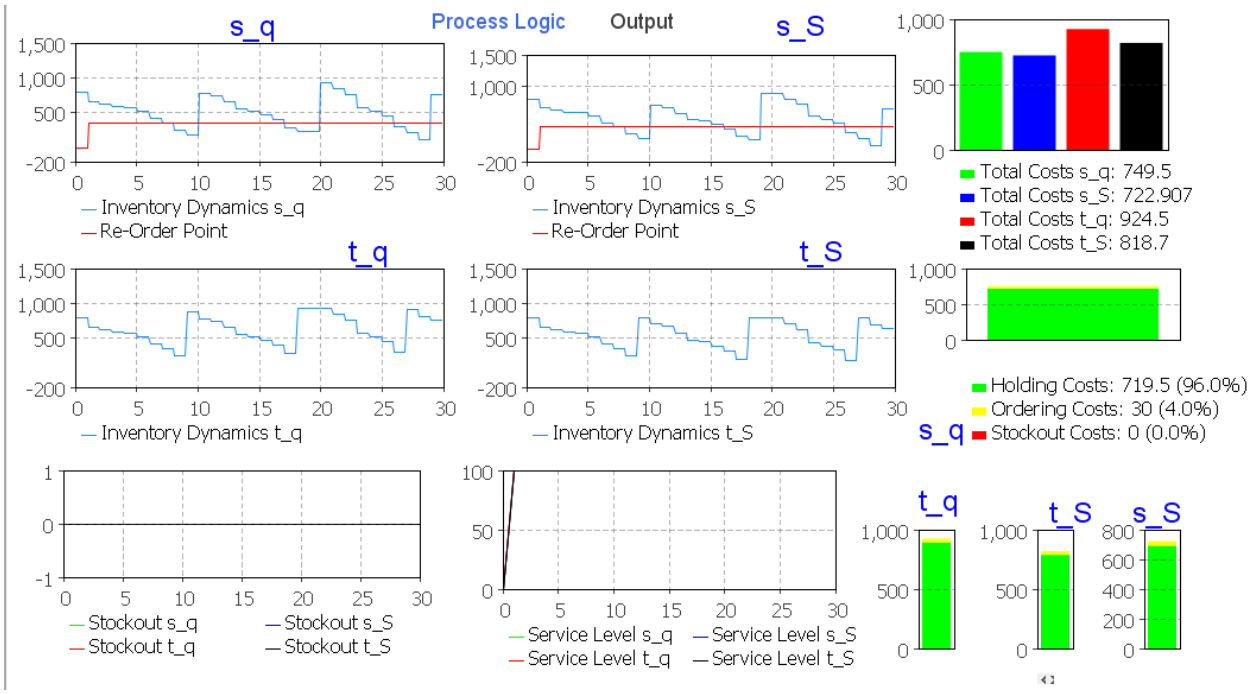


Fig. 94 Experiment #3 with parameter adjustments

It can be observed from Fig. 94 that different policies performed differently in the initial situation and after parameter changes. Table 25 compares the results.

Table 25 Result comparison for experiment 3 with and without parameter adjustment

Policy	**Costs**		**Service Level**	
	No changes	Changes	No changes	Changes
t,q	1,255	923	100	100
t,S	819	819	100	100
s,q	840	750	100	100
s,S	**691**	**723**	95.6	100

It can be observed from Fig. 94 and Table 25 that an increase in target inventory positively influenced the *s,S*-policy in regard to service level whilst efficiency slightly decreased ($723 instead of $691). Here a trade-off "service level vs efficiency" can be nicely observed. Nevertheless, *s,S*-policy still remains the most efficient one. Changes in order quantity also positively influenced efficiency of *t,q*- and *s,q*-policies.

Perform simulation analysis for two other products from our case study. These products are characterized by low target inventory. This will imply higher delivery frequency. Analyse impact of changes in demand, lead time, and order quantity similar to experiments 1-3. Can you observe any differences as compared to products A, B, and C with high target inventory?

Of course, sensitivity analysis is needed here in regard to different values of parameters in all the control policies and the costs in order to make generalized recommendations. In addition, we need to take into account that periodic policies require less supplier flexibility and therefore may allow negotiating lower unit prices. This would result on lower inventory holding costs as compared to continuous review policy which require higher flexibility. Continuous review policy are also more complex and require sophisticated software to monitor all SKU (stock keeping units) the number of which can reach millions of items in distribution centers. Literature provides a number of useful techniques to determine basic parameters of inventory control policies analytically (Zipkin 2000, Axsater 2010)

16. Extensions

In §14-15, we considered basic inventory control policies. We assumed that no constraints on supply side exist and demand can be fairly estimated with the help of statistical methods. In reality, we cannot order more than the suppliers can produce or the distributors have on stock. Market developments cannot always be fairly estimated with the help of statistical methods. In this final paragraph, we introduce two new simulation methods: system dynamics and agent-based modelling. This is unique feature of AnyLogic to allow creating simulation models using all three simulation methods (discrete events, agents, and system dynamics) jointly.

16.1 Inventory management in supply chains with production and transportation considerations: system dynamics

In supply chains, behaviour of producers, distributors, retailers, and suppliers depend on each other. This implies that production, inventory and transportation planning need to be integrated. Such integration is known as production-inventory, inventory-routing and production-inventory-routing problems. The major idea of such integrated problems is to find *balanced states* in regard to production and sourcing quantities. As such, system dynamics (Sterman 2000) is a popular method for simulating the integrated problems.

"The System Dynamics (SD) methodology is typically used in long-term, strategic models and assumes a high level of aggregation of the objects being modeled. People, products, events, and other discrete items are represented in SD models by their quantities so they lose any individual properties, histories or dynamics. If this level of abstraction is appropriate for your problem, SD may be the right method to use. However if individual details are important, you can always re-conceptualize all or part of your model using Agent Based or Discrete Event (process-centric) methods without ever leaving the AnyLogic environment." (AnyLogic.com).

16.2 Agent-based modelling the market demand

"AnyLogic is the only tool that allows you to combine SD model components with components developed using agent based or discrete event methods. This can be done in a number of different ways. For example, you can model the consumer market using SD and the supply chain using the AB approach. You can model the population of a city in a disaggregated way (as agents) and the underlying economic or infrastructural background in an SD style. You can even put SD diagrams inside agents. For example, SD can model the production processes inside a company, whereas the company may be an agent at a higher level. Technically, interfaces and feedbacks between SD and AB or DE are very easy. Some SD variables can be used in the decision logic of agents or be parameters of process flowcharts, and the latter in turn may modify other SD variables." (AnyLogic.com).

We recommend to consider the example models in AnyLogic PLE software "AB Market and SD Supply Chain", "Adaptive Supply Chain", "Supply Chain", "Two Stocks Problem", and "Stock

Management" to understand agent-based simulation, system dynamics, and advanced inventory control policies.

17. Literature

AnyLogic.com

Axsater S (2010) Inventory control. 2nd edn. Springer, New York

Borshchev A. (2013). The Big Book of Simulation Modeling: Multimethod Modeling with Anylogic 6. AnyLogic North America

Grigoryev I. (2015). AnyLogic 7 in Three Days: A Quick Course in Simulation Modeling. CreateSpace Independent Publishing Platform, 2nd Ed.

Ivanov D., Tsipoulanidis A., Schönberger J. (2016). Global Suppy Chain and Operations Management. Springer, 1st edition.

Rozhkov M. (2011). Designing simulation models for inventory management analysis in supply chains. Available at www.anylogic.com (in Russian)

Runthemodel.com

Sterman J. (2000). Business Dynamics: Systems thinking and modeling for a complex world. McGraw Hill.

Zipkin PH (2000) Foundations of Inventory Management. McGraw-Hill, Boston et al.

18. Discussion

- What roles play the blocks "Wait", "Delay", "Resource", "Seize", "Release", and "Service" in Process Modelling Library in regard to this case-study? What other blocks in the Process Modelling Library do you know? Think of other possible applications of these blocks in manufacturing or logistics planning and control setting!
- Explain basic trade-offs between capacity utilization, lead-time, and the number of waiting orders!
- Explain basic trade-offs in coordinated decision making to balance the objectives of the division head, the sales manager, and the manufacturing manager in order to achieve maximal possible profit under given constraints.
- What KPIs are important for a control dashboard of a workload balancer?
- Discuss on the difference between push and pull dispatching principles!
- What can you say about bottlenecks?
- Discuss on the importance of sales plan and capacity availability alignment!
- What is the objective of workload balancing?
- What KPI are typically used for financial, customer and operational performance measurement? How to compute them in AnyLogic?
- What elements are needed to build a network structure such as supply chain?
- What is important for JIT implementation in the supply chain?
- In which inventory review system higher safety stock is required?
- What kinds of products can be ordered in both continuous and periodic review systems?
- Which review system has higher uncertainty?
- In which inventory control policy you can observe higher stock-outs?
- Discuss about advantages and disadvantages of simulation and optimization for decision-making in operations and supply chains! Where are these methods different and how can they enrich themselves mutually?