# UNIVERSITÀ DEGLI STUDI DI PADOVA

DEPARTMENT OF MANAGEMENT AND ENGINEERING

Master's degree in Management Engineering

Master Thesis

## ANALYSIS AND IMPROVEMENT OF A BOTTLING LINE USING A SIMULATION MODELLING APPROACH

**Supervisor: Prof.ssa Daria Battini**

**Co-supervisor: Prof.ssa Ilenia Zennaro**

**Co-supervisor: Prof. Allan Larsen**

**Author: Guido Cortella**

**Student ID: 1156533**

ACCADEMIC YEAR 2018 - 2019

**Abstract**

This project thesis is focused on the study of a bottling production line using a modelling simulation method, through which we analyse the inefficiencies and then improve their performance.

In the first part, we talk about some important theoretical aspects concerning Global Production Effectiveness where we define the main concepts about Total Production Maintenance and an explanation of Overall Equipment Effectiveness (OEE). Then we discuss about automatic production systems, in particular we focus on the sizing of intermediate buffers and their effects in a production line. Moreover, we talk about some theoretical concept of the Buffer Problem Allocation (BAP) with a synthesis of the literature review.

In the second part, the main topic is the simulation modelling. We point out its characteristics and its importance. Then we introduce a Simulation Methodology explaining the steps to follow to analyse and solve a problem of a system. Moreover, we explain the main characteristics of Anylogic that is the software used to study the case study.

At the end, we present the case study concerning the analysis and the improvement of the performance of a bottling line following the steps about the proposed method. The case study ends with the comparison of the results carried out from the simulation with the results obtained from the application of the formula being in the paper of *Battini et al. (2013).*

# Contents

# Introduction

In the industrial world, all companies have as their main focus the effectiveness and efficiency of their operations, that is all that concerns the realization of goods and services. The goal of the companies is therefore to produce at low cost and to guarantee a high flexibility to their own operations in order to satisfy the demands of a market in continuous evolution and increasingly sensitive to the selling price.

In this context, companies, for example in the beverage market, with high sales volumes and low margins have invested in automated production systems. Therefore, monitoring and analysis of production performance in the optical of a continuous improvement in efficiency becomes even more important. The processes are supervised by measuring the KPIs (Key Performance Indicators) that indicate the status of the process. The main KPI used by companies is the OEE (Overall Equipment Effectiveness) which provides information regarding the availability, performance and quality of the production plant (*Nakajima, 1988*). Overall Equipment Effectiveness is such a performance measure, which indicates current status of production with least calculation. It also help to measure losses and corrective actions can be taken to reduce it (*VivekPrabhu et al., 2014*). In particular, it is essential to analyse the downtime of production plants and especially the micro-stops (micro-breakdown or micro-downtime), that are very often not considered because they are difficult to identify and they are evaluated part of the normal functioning of the production process (*Ljungberg, 1998*).

A tool that helps measuring and studying performance improvement is simulation. Simulation has become increasingly important during the years as it allows us to represent the actual situation through models. These models allow the study and analysis of the system behaviour to search and test new scenarios that guarantee an improvement in performance. The simulation is a very useful tool to identify the critical points of a system and to find the most appropriate alternative within a set of proposed configurations. With simulation models, how an existing system might perform if altered could be explored, or how a new system might behave before a modify is really applied, thus saving on cost and time (*Hosseinpuor et al., 2009*). The simulation tool represents an added value that allows to study reality with fewer simplifying hypotheses, allows the rapid modification of variable structure and algorithms to study its effects and provides performance results that are difficult to obtain from other types of approaches.

The thesis project is based on the application of the simulative approach to a case that belongs to this type of context. The case study regards an automatic bottling line characterized by some inefficiencies due to micro-downtimes due to failure, set-up and preventive maintenance. The objective of this project is to optimize the buffer sizing of the critical section (BAP), i.e. the one composed by the Bottle Washer and Labeler. The buffer makes it possible to efficiently decouple the two stations. In particular, intermediate buffers may increase the reliability of the whole system by limiting the consequences of micro-downtime, and saving companies from making inadequate purchases of oversized equipment (*Battini et al., 2009*). The aim is to improve the OEE of the production line and its throughput.

To achieve this kind of results, the project is developed by two kind of methods. At beginning, the case is studied following a simulative approach with AnyLogic software. The method taken into consideration is the *General Methodology for Applying Simulation to Problem Solving* (Rossetti, 2015). The test, used to obtain the optimal buffer size, is performed through a *Parameter Variation Experiment* in order to study the impact on the efficiency performances.

At the end, the bottling case is also studied following an analytical approach by a tool called: *Buffer design for availability (BDFA), Battini et al., 2013.* This formula allow to achieve the optimal buffer size in a simple and quickly way. The project ends with a comparison between the two types of approaches.

# 1 Maintenance and Effectiveness in a Production Line

A growing multitude of variety and an increasing product differentiation has led the sector of machinery and plant engineering to face new challenges.

More customization, shorter product life cycles, uncertainty in demand as well as growing international stress of competition are just some of the reasons that led companies to move towards increasingly automation of their production facilities and ongoing internationalization of their production sites.

Automation, in general, has the core aims of reducing human participation in production systems, introducing machines for doing repetitive and/or complex actions and transforming production to make it as continue as possible. In this type of system, the presence of the operator is necessary only to achieve a correct monitoring of the process, adjustment operations or modification of some production parameters. Production systems effectiveness remains the principal aim of each industry in order to be competitive and achieve success, but it is deeply influenced by the previews market requests.

In this contest, Total Productive Maintenance (TPM) plays a very important role in the industry for plant productivity and operation efficiency. TPM has become one of the most popular maintenance strategies for ensuring high machine reliability and it is regarded as an integral part of lean manufacturing (*Rahman, 2014*). The main index to measure the effectiveness that TPM used is the overall equipment effectiveness (OEE).

## 1.1 Total Productive Maintenance

TPM is a production system that aims to achieve maximum business efficiency. Historically it born to guarantee the maximum efficiency of the single plants, focusing the attention on the activities of the operators, mechanical maintenance and the process technicians. Moreover, activities concerning quality, personnel development, safety and environmental activities and industrialization are also structured.

### 1.1.1 History of TPM

TPM is one of the Japanese production techniques, developed in the 1960s and at the Toyota Motor Corporation and then developed in all the major Japanese companies, thanks to the Plant Maintenance Committee of the JMA (Japan Management Association). Since 1961, it invested his energies in TPM and in 1971, he presented it as a methodology that extended to all operators a role in the operational management of maintenance (for this reason it has been called Total Productive Maintenance).

The recognized "father" of TPM is Seiichi Nakajima, first technical director at Toyota and then (until the end of '80) consultant at JMA and JIPM. Nakajima has been interested since the early fifties in the knowledge developed in the United States regarding preventive maintenance, reliability and maintainability of the plants, life cycle costs and more. When some Americans went to Japan to teach some reference bases in the operational management of the facilities, he acted as an interpreter for his colleagues and continued to work on what had been learned, enriching him with observations and connections. More recently, in 1984, Nakajima came to Italy for the 1st World Maintenance Congress, organized in Venice by AIMAN, the Italian Maintenance Association. During the congress, he illustrated the TPM to the astonishment of the people present. In 1998 the first English version of his book "Introduction to TPM" (the original version in Japanese is from 1984) was released and, in 1992, the first Italian edition was published for the types of ISEDI. FIAT Auto from 1985, with the RDA (Institute for Research and

Intervention in Business Management) and the Telos Group (today Deloitte Consulting) made the first experiences of TPM in Italy.

The Nakajima Prize is entitled to Seichi Nakajima. In particular, he is acknowledged to have been able to insert the various elements learned in an organic vision, making the individual notions elements of a system capable of becoming a true competitive instrument for the companies that apply it. There is also an important recognition known as the TPM Excellence Award given each year to companies that have achieved excellence in the application of the principles that the methodology provides. Established by JIPM as PM Award in 1964, it is still very much coveted by manufacturing companies in the world, not just Japanese.

### 1.1.2 The eight pillars of TPM and the 5s

The main objectives of total productive maintenance concern:

- The reduction of plant stops and their impact on performance;
- The increase in availability and efficiency of the production systems;
- Elimination of losses due to defects, breakdowns and accidents;
- The increase in the useful life of the machines and their reliability.

The achievement of these objectives is based on the introduction of a proactive maintenance approach, with a greater focus on preventive and predictive interventions, in order to move up and avoid plant downtime due to machinery failures. Reducing the reactive aspect of maintenance, restricting interventions to failure, it has a direct effect on the performance of the production system.

Nakajima's philosophy extends to the whole company three fundamental concepts:

- Total efficiency research of the entire production system;
- Implementation of the total maintenance system;
- Total involvement of operators with the participation of all employees, in particular through independent maintenance.

The main aspects of total preventive maintenance can be summarized in 8 fundamental pillars as shown in the following figure 1.

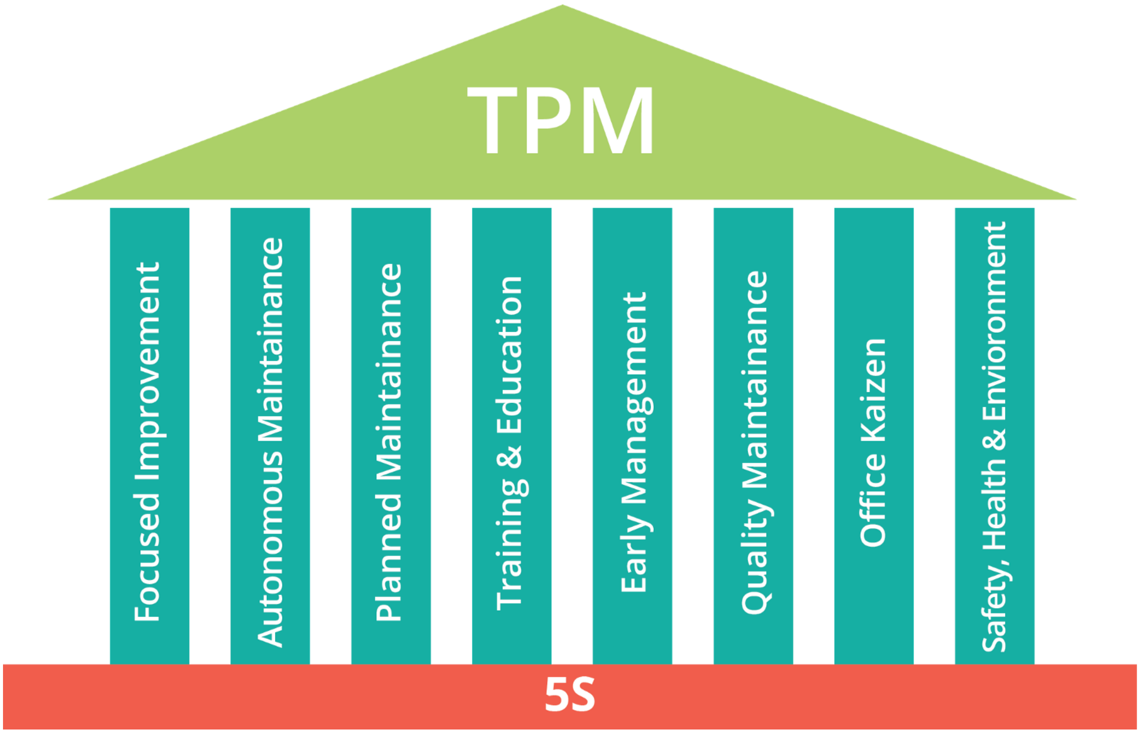



*Figure 1 - The eight pillars of TPM*

1. *Focused Improvement*

Study and achievement of continuous improvement of a production processes based on the Kaizen approach. In particular, the main concepts of the focused improvement approach are:

- Simplification of the processes;
- Simplification of the machines;
- Simplification of the plant;
- Identification of the most critical issues;
- Incremental resolution of problems;
- Continuous improvement according to PDCA cycle.

2. *Autonomous Maintenance*

The core idea of autonomous maintenance is to provide the operators with more responsibility and allow them to carry out preventive maintenance tasks. The autonomous maintenance allows machine operators to carry out directly simple maintenance works (lubrication, bolt tightening, cleaning and inspection) to prevent breakdowns and react faster if a certain failure has been detected.

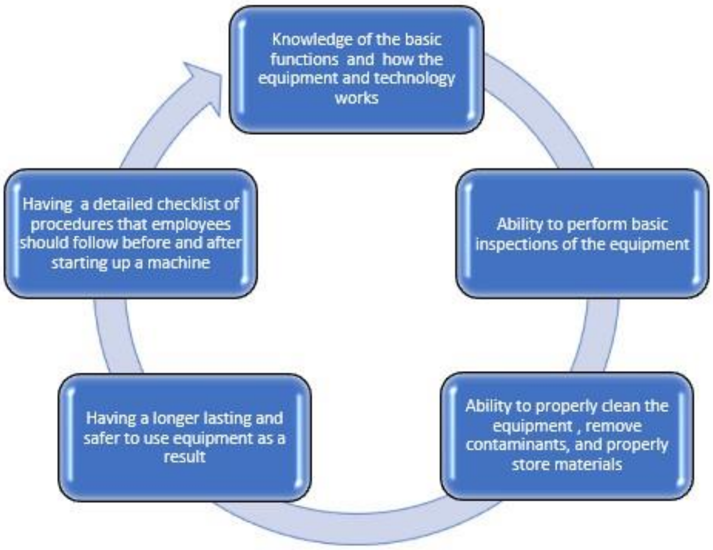



*Figure 2 – Principles of Auonomous Maintenance*

3. *Planned Maintenance*

It is the correct management of the plants by the maintenance.

In particular, it is a proactive approach to maintenance in which maintenance work is scheduled to take place on a regular basis. The type of work to be done and the frequency varies based on the equipment being maintained, and the environment in which it is operating.

4. *Training and Education*

It means the professional growth and the increased skills. It ensures that staff are trained in the skills identified as essential both for their personal development and for the successful deployment of TPM in line with the organisation’s goals and objectives.

5. *Early Management*

Early Management aims to implement new products and processes with vertical ramp up and minimised development lead-time. It is usually deployed after the first four pillars as it builds on the learning captured from other pillars teams, incorporating improvements into the next generation of product and equipment design.

6. *Quality Maintenance*

Quality Maintenance is the elimination of any possible deterioration of the instruments, moving up the wear that could be harmful to the production. Understanding and controlling the process interaction, it is possible to reduce the defects. The key is to prevent defects from being produced in the first place, rather than installing rigorous inspection systems to detect the defect after it has been produced.

7. *Office TPM*

Office TPM concerns the administrative part and the support part to the organization. The aim is to eliminate waste and losses in these company departments.

8. *Safety, Health & Environment*

The final pillar requires a methodology that aims at the occurrence of zero accidents. In other words, it concerns the management of safety and energy waste. This translates into implementing preventive actions to allow employees to work in optimal conditions. The importance of safety and health leads company departments to work transversally in risk management, in creating well-defined standards and in optimizing ergonomics.

In this contest, to support TPM approach, some activities that Lean thinking call 5s, are defined to reduce waste. The methodology for the improvement "5 S" is a methodological approach, born within the logic of Lean Production, which aims to initiate and maintain a process of reduction and elimination of waste present within an organization, thus continuously improving work standards and product quality.

The different steps of the methodology are the following:

1) SEIRI (Sort): to discern and to divide the necessary equipment, materials and instructions from those that are not necessary. The process ends with the elimination of the latter.
2) SEITON (Set in order): to arrange carefully the equipment and parts after identifying them.
3) SEISO (Shine): to clean the workplace thoroughly and extensively.
4) SEIKETSU (Standardize): to standardize work activities to manage problems.
5) SHITSUKE (Sustain): to ensure the standards are regularly applied.


*Figure 3 – 5S*

## 1.2 Overall Equipment Effectiveness (OEE)

The Overall Equipment Effectiveness (OEE) is the traditional evaluation measure of the Total Productive Maintenance (TPM) that has to be maximized and it compares operating level with the ideal potential of the plant performance (*Lanza, Stoll et al., 2013*). It is a widely used global efficiency index in manufacturing, whose function is to control the performance of production plants. An important aspect of this index is to be able to define a target value, to follow as a goal in order to continuous improvement. At the same time, OEE is essential to identify the critical points of a plant and so to focus management's attention on the problems to be solved. The OEE is a synthetic and quantitative index, consisting of a single number, which however is able to contain a large number of information regarding the production plant. Nowadays, the production companies used it because it's very important to remain competitive on the market selling it at minimum cost (*Muchiri and Pintelon, 2008*).

In this contest, the OEE can help to know where improve and the impacts of the improvement. This is because OEE has a particular composition: it is function of three fundamental factors of a plant: availability, performance and quality. This configuration allows identifying the most critical voice or voices, in order to study and propose improvements. The three factors are defined as follows:

- *Availability:* indicates the time in which the plant is actually available for work compared to the planned production time;
- *Performance efficiency:* indicates the actual production time with respect to the time in which the plant is actually available for work;
- *Quality:* indicates, as a percentage, how many compliant products have been produced with respect to the total production.

It is possible to understand where and how to improve the performance of the system and any problems by analysing each of these factors individually.

## 1.2.1 Six Big Losses

Developed in 1971 at the Japanese Institute of Plant Maintenance, the Six Big Losses in manufacturing have been used as a way to categorize equipment-based losses and maximize overall equipment effectiveness (OEE).

In particular, production plants are subject to phenomena that cause loss of time. These disorders can be chronic or sporadic depending on their frequency of occurrence. Usually the chronic phenomena are small, hidden and difficult to identify. On the other hand, sporadic ones are easier to identify as they occur with a high speed and great deviation from the normal state of the system. However, the most significant lost times in a plant are usually the chronic ones since, although minor in terms of duration of the single disturbance, their frequency leads to a low rate of equipment utilization and high costs due to losses (*Nord et al., 1997*).

| Overall Equipment Effectiveness | Recommended Six Big Losses | Traditional Six Big Losses |
|---|---|---|
| Availability Loss | Unplanned Stops | Equipment Failure |
| | Planned Stops | Setup and Adjustments |
| Performance Loss | Small Stops | Idling and Minor Stops |
| | Slow Cycles | Reduced Speed |
| Quality Loss | Production Rejects | Process Defects |
| | Startup Rejects | Reduced Yield |
| OEE | Fully Productive Time | Valuable Operating Time |

*Figure 4 – Traditional Six Big Losses VS OEE*

Anyway, these two types of disturbances lead to a less efficient process, with greater resources consumed, without contributing to any benefit for the final product. The generic losses, which reduce the effectiveness of the equipment, have been grouped and categorized as six big losses, and they are the following: equipment failure, setup and adjustment, idling and minor stops, reduced speed, process defects, reduced yield. As shown in a table above, the Six Big Losses categorize productivity loss from an equipment perspective. They align directly with OEE and provide an actionable level of detail about OEE losses.

EQUIPMENT FAILURE

Equipment failure accounts for any significant period of time in which equipment is scheduled for production but is not running due a failure of some sort. In other words, it is as any unplanned stop or downtime. Equipment failure is an Availability Loss.

The occurrence of these events is of a random nature and it depends on the phase of the life cycle of the system being discussed. The probability of failure is related to the trend in time of the failure rate, described by the "bathtub curve", valid both for a single component and for a complex system.

The typical theory of "bathtub curve" has been widely accepted as an engineering tool. The bathtub shape is 'characteristic of the failure rate curve of many well designed products and components including the human body' (*Oakland, 1992*). The classic bathtub against time has three different period:

- Decreasing failure rate for infant mortality: the initial phase is characterized by a decreasing failure rate and it takes part of the machinery testing period
- Constant failure rate for useful life: the central phase is that of the useful life of the system and is characterized by a constant failure rate and therefore random failures;

- Increasing failure rate for wear-out: the final phase begins with the wear of the components, which leads to an increase in the probability of breakage.

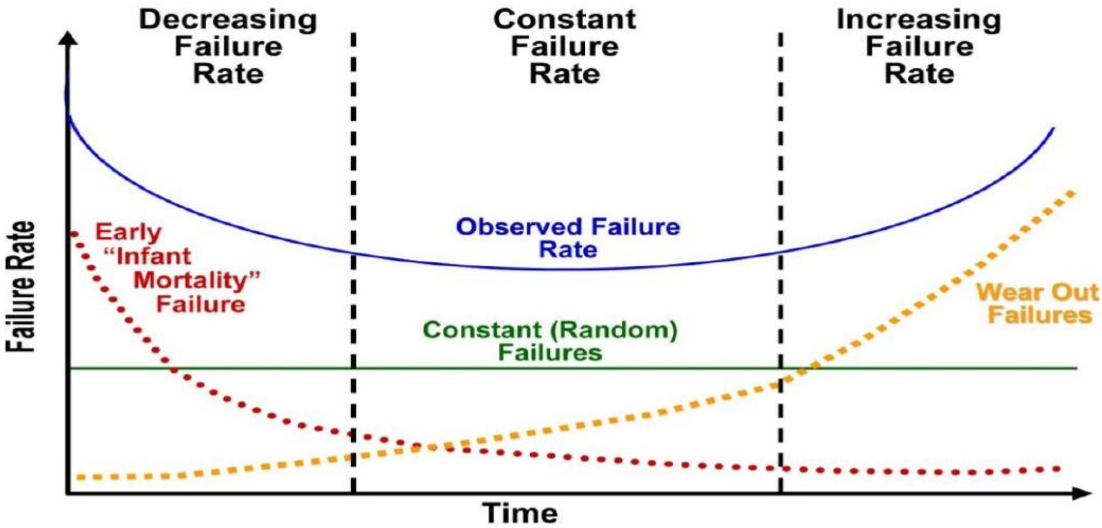



*Figure 5 – The Bathtub Curve*

SETUP AND ADJUSTMENTS

Setup and Adjustments accounts for any significant periods of time in which equipment is scheduled for production but is not running due to a changeover or other equipment adjustment. A more generalized way to think of Setups & Adjustments is as any planned stop. Setup and Adjustments is an Availability Loss. In other words, the Setups performs all the automatic and manual adjustments that take place on the system in order to respond to the previous batch and start the production of a new product. They depend on the production mix. The time of the single setup can be reduced by SMED techniques (Singe Minute Exchange of Die).

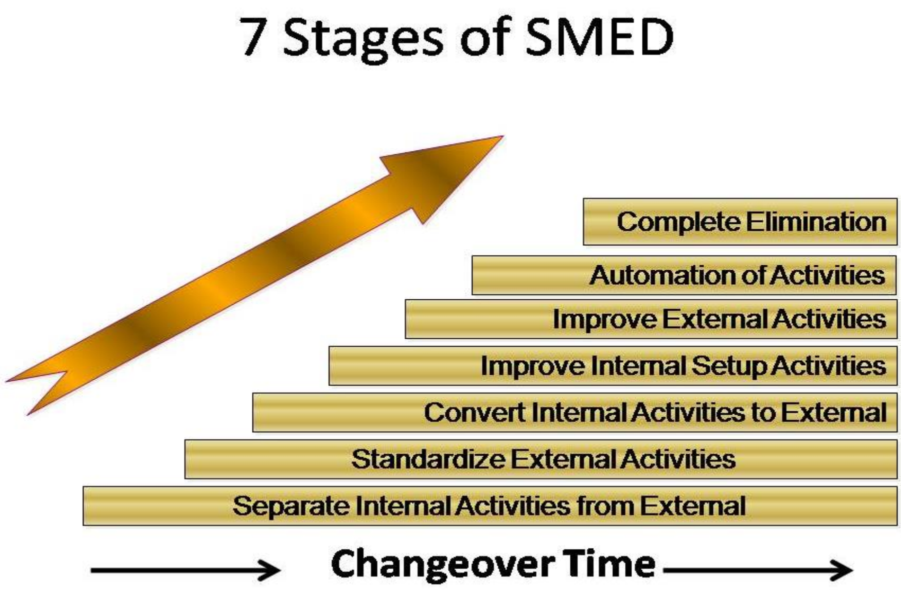



*Figure 6 – SMED*

IDLING AND MINOR STOPS

Idling and Minor Stops accounts for time where the stop resolved by the operator. Another name for Idling and Minor Stops is small stops. Idling and Minor Stops is a Performance Loss. This category is made up of breakdowns under 5 minutes and that require minimum or null personnel service. With a reduced impact, but with a frequency of occurrence that can be very high, the micro-downtimes can represent a heavy performance loss for the production system.

REDUCED SPEED

Reduced Speed accounts for time where equipment runs slower than the Ideal Cycle Time (the theoretical fastest possible time to manufacture one part). Another name for reduced speed is slow cycles. Reduced speed is a Performance Loss. Some common reasons for reduced speed are dirty or worn out equipment, poor lubrication, substandard materials, poor environmental conditions, operator inexperience, start-up, and shutdown.

PROCESS DEFECTS

Defective products manufactured while production is generally stable. In this case, defects include scrapped parts along with those that can be reworked. This is because OEE measures quality according to First Pass Yield (FPY), making this Big Loss a quality criterion.

STARTUP DEFECTS

Start-up Defects are defective parts produced from start-up until stable production is reached. They can occur after any equipment start-up, however, are most commonly tracked after changeovers. Examples include suboptimal changeovers, equipment that needs “warm up” cycles, or equipment that inherently creates waste after start-up (e.g., a web press).

### 1.2.2 Definition of Overall Equipment Effectiveness

The Overall Equipment Effectiveness (OEE) is the total effectiveness measure of a plant. It is an index expressed in percentage points that sums up three very important concepts from the point of view of manufacturing production: the Availability, the Performance and the Quality rate of a plant.

$$OEE = Availability \times Performance \times Quality$$

The OEE is used as a measurement tool in TPM (Total Production Maintenance) and in the Lean Manufacturing programs, where it is able to provide an important key to understanding the effectiveness of the measures adopted while providing support for the measurements of efficiency.

Another way to express OEE is through the ratio between what was manufactured and what could be ideally manufactured or, alternatively, as the fraction of time in which an equipment works at its full operating capacity. The formula that sums up this concept is the following:

$$OEE = \frac{Actual\ Output}{Reference\ Output} = \frac{Cycle\ Time\ \times Valuable\ Operation\ Time}{Cycle\ Time\ \times Loading\ Time}$$

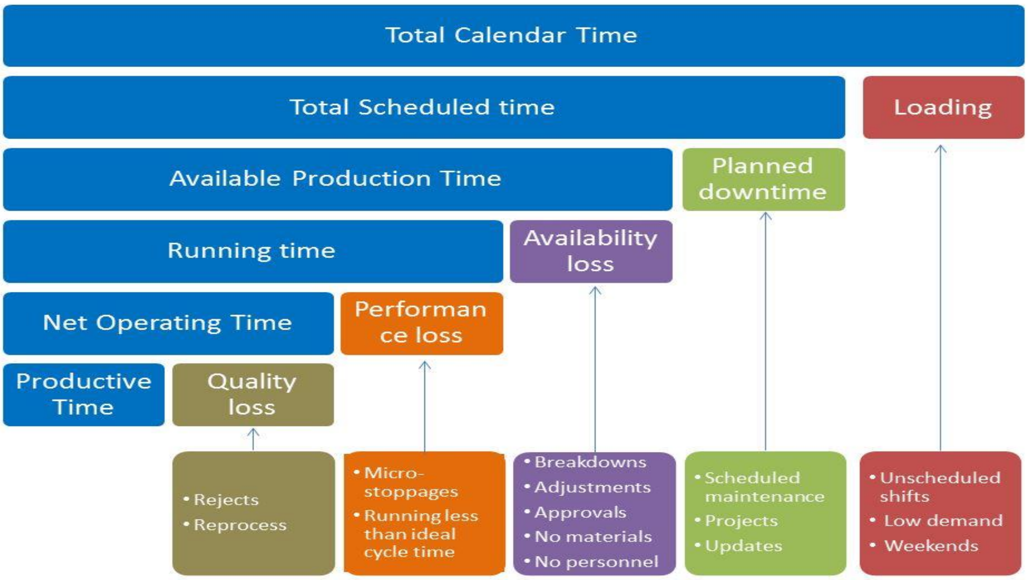



*Figure 7 – Components of effectiveness index*

**Availability**

Availability (A) indicates the ratio between the time actually available for the production (without all planned stops, setups, failures) and the total time in which the plant is potentially in operation and it can be calculated:

$$Availability\ (\%) = \frac{Acutal\ operating\ time\ (mins)}{Planned\ operating\ time\ (mins)} \times 100$$

Where:

- *Planned operating time (mins)* = Total shift time(mins) – Planned maintenance (mins)
- *Actual operating time (mins)* = Planned operating time (mins) - Unplanned maintenance (mins) – Minor stoppages (mins) – Setup changeover (mins).

**Performance rate**

The performance rate represents the relationship between the real production speed of the machine and the theoretical one. These two speed should be the same, but often the machines, for various reason, are subjected to micro-downtimes, which determine inefficiency. The performance rate includes the losses linked to the micro-downtimes and the speed drops, which reduced the production output. It can be calculated:

$$Performance\ rate\ (\%) = Net\ operating\ rate\ \times Operating\ speed\ rate\ \times 100$$

Where:

- $Net\ operating\ rate = \frac{No.produced\ \times Actual\ cycle\ time}{Operation\ time}$
- $Operating\ speed\ time = \frac{Theoretical\ cycle\ time}{Actual\ cycle\ time}$

**Quality**

Quality takes into account manufactured parts that do not meet quality standards, including parts that need rework. The Quality index defines Good Parts as parts that successfully pass through manufacturing process the first time without needing any rework. It is calculated as:

$$Quality = \frac{Good\ count}{Total\ count}$$

### 1.2.3 The advantages and limits of the traditional formula

The OEE is a simple measurement index that allows you to understand and analyse in detail the loss of time through the breakdown into three factors. The goal is to improve the reliability and performance of the machine. It is also possible to quantify the beneficial contribution in terms of performance by comparing the "as is" situation with the "to be" situation to emphasize the actual improvement. Moreover, the OEE has the ability to highlight where the problems of a system reside, to increase production capacity, balance the flow of materials and prevent sub-optimization of processes; it also allows obtaining a systematic method to establish productive targets and a tool for the practical management of the plants (*Garza-Reyes et al., 2009*).

The main applications of the OEE are in automated industries, where the saturation of the production capacity has a high priority and the plant stops are very expensive and lead to a great loss of production (*Dal et al., 2000; Andersson and Bellgran, 2015).*

The OEE is certainly a very used tool in the managerial field due to its ease of understanding and conciseness. Despite all these positive aspects, it presents some limits in the original formula.

One of the main problems that have been highlighted concerns the fact that this formula is not able to explain the performance of an entire process or of the entire production area of the company, but returns the performance of a single machine. When we talk about the production process we mean a set of coordinated machines that interact with each other; hence the birth of the problem as the considerations are made on the single machine without considering their interdependence. The ultimate goal of any company should be a high efficiency of the entire integrated production system and not only have perfect equipment *(Muchiri and Pintelon, 2006)*. Furthermore, while for a perfectly balanced line without buffers, the OEE can be an effective tool to express its performance, in the case of unbalanced processes and with the presence of buffer between the machines the OEE is not able to be a reliable indicator of performance *(Braglia et al., 2008; De Carlo et al., 2014)*. This point is however very controversial because some authors believe instead that the OEE is not suitable to describe the performance of a single machine. Indeed they consider the OEE as a tool that takes into consideration factors external to the machinery, such as material handling, presence of buffers, the efficiency of the logistic system, the lack of products and the block of the line by the downstream machines (*de Ron and Rooda, 2005; Braglia et al., 2008; De Carlo, 2014)*. Another limit is to give equal weight to all three factors making up the OEE. Quality problems should have a different weight compared to the availability and efficiency of the performances and therefore each of the three items should have their own characteristic weight, typical of every company or industry, which allows a more correct evaluation *(Muchiri and Pintelon, 2006; Wudhikarn, 2013)*. Further criticisms indicate the difficulty in defining and understanding the measure due to its composition in three factors. Moreover, there is no clear cause-effect relationship between a change in the values of the three voices and the OEE. Finally, we consider a pre-set ideal cycle time that controls maximum productivity, but the number of people working in the process is not taken into consideration, therefore it does not allow us to adequately assess an improvement in productivity given by the reduction in cycle time or resources used as input *(Andersson and Bellgran, 2015)*.

The *table 1* resume the advantages and limits of the traditional formula.

| **Advantages** | **Limits** |
|---|---|
| Synthetic index<br><br>Information content<br><br>Possibility of evaluating changes in the plants<br><br>Ability to highlight system problems | Some authors believe that it only expresses the performance of a machine and not of the entire system<br><br>Some authors believe that it is not able to express the performance of a single machine as it is affected by factors external to this one<br><br>The three factors of OEE have the same weight<br><br>Lack of consistency in the definition of availability<br><br>Lack of a clear cause-effect relationship between the change in the values of the three items making up the OEE and the OEE itself<br><br>No evaluation of cycle time or plant resources reduction |

*Table 1 – Advantages and limits of traditional formula*

## 1.3 Other performance index

In relation to the limits of the standard formulation of the OEE, some alternative indexes have been proposed in literature with which it is possible to more fully evaluate the progress of a production plant.

The advanced formulations that allow overcoming the main limits of traditional OEE can be divided into two groups:

- Indicators to calculate overall performance at plant;

- Indicators to calculate the performance of the single machine.

In this section, we discuss the following index: OLE and OEEML. Their goals are the extension of the applicability field of the Overall Equipment Effectiveness at the process level.

### 1.3.1 Overall Line Effectiveness

The Overall Line Effectiveness (OLE) is an alternative metric to evaluate the efficiency of a continuous product flow manufacturing system, proposed by Nachiappan and Anantharaman (2006). It is the product of two independent terms, namely the line availability (LA) and the line production quality performance (LPQP):

$$OLE = LA \times LPQP$$

Under the hypothesis of no decouples added between machines, all the operations performed in a manufacturing line are strictly connected together. Indeed, the operating time (OT) of the first machine will be the LT of a second machine and, in analogy, the OT of second machine will be the LT of a third machine and so on, proceeding downward in the line. This concept is shown in next equation where DT and PD stand for downtime and planned downtime, respectively:

$$OT_i = (OT_{i-1} - PD_{i-1}) - DT_i$$

Consequently, LA can be evaluated as the ratio of the OT of the last machine and the LT of the line, as stated by the equation:

$$LA = \frac{OT_n}{LA}$$

At the end, as in the standard OEE definition, LPQP is defined as the ratio of the actual and ideal productive rate of the line and is evaluated applying equation:

$$LPQP = \frac{G_n \times CT_{BN}}{OT_1}$$

where $G_n$ represents the number of items manufactured by the last operation and $CT_{BN}$ is the cycle time of the bottleneck machine.

The problem is that the hypotheses made to evaluate $OT_i$ do not apply when buffer are displaced between machines. Therefore, OLE gives good results just in a continuous production line. Actually, where there are buffer in the line, a DS machine can continue manufacturing even if the preceding machine is down and so, a straight application OLE would underestimate the actual efficiency of the line. Furthermore, both the terms of OLE (i.e. LA and LPQP) regards the operating efficiency of the last machine. This is an additional problem because it is difficult to identify the main criticalities and to detect the points of the line where they actually take place just by monitoring the last machine.

### 1.3.2 Overall Equipment Effectiveness of a Manufacturing Line

The OEEML (Overall Equipment Effectiveness of a Manufacturing Line) is an index proposed by *Braglia et al. (2008)*, which considers the interaction between the various machines of a production system.

The proposed methodology makes it possible to highlight the progressive degradation of the ideal cycle time along the line, and to split the global losses into its main components, making it easier to detect the points where the major problems take place, and to plan the appropriate corrective actions.

In a production line, material flow, buffers, transportations and queues have a direct impact on equipment performance and vice versa. The calculation of the OEEML starts from the division of time lost in two macro-categories:

- Equipment dependent losses (EDL): losses linked to the operation of the single machine;
- Equipment independent losses (EIL): losses linked to the interaction of the systems in the plant.

The EDL can be eliminated through improvement interventions, while the EIL must be managed acting directly on the productive environment. In addition to this difference in the allocation of downtime times, planned stops are also considered in the OEEML, in particular preventive maintenance, as suggested by De Groote (1995).

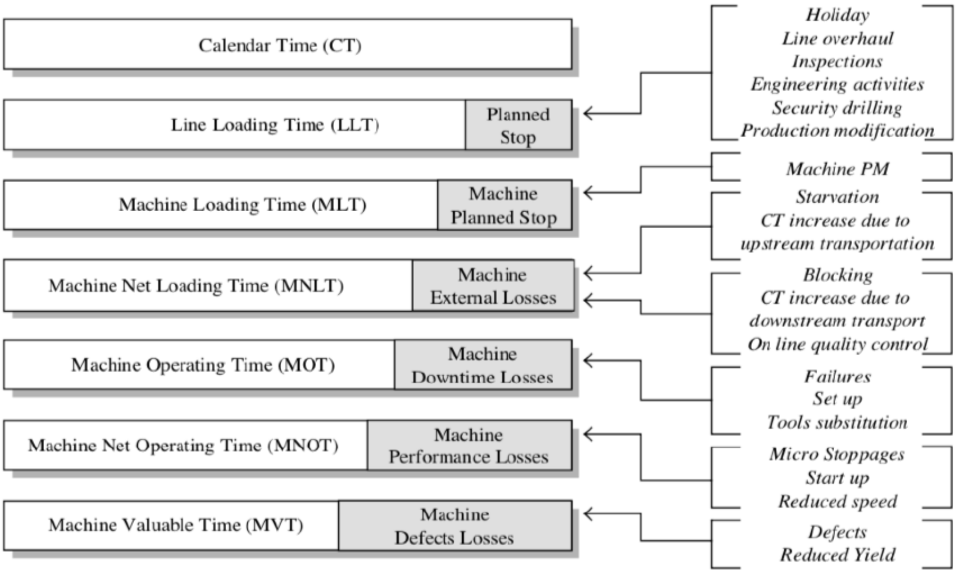



*Figure 8 – Step to calculate OEEML (Braglia et al., 2008)*

Referring to the structure of losses shown in the figure, it is possible to calculate the value of Total Overall Equipment Effectiveness (TOEE), which takes into account all the production losses of the considered plant.

$$TOEE = \frac{MVT}{LLT}$$

where:

- MVT = Machine Valuable Time;
- LLT = Line Loading Time.

Through the definition of TOEE, the evaluation of the OEEML is straightforward, in particular:

$$OEEML = \frac{Actual\ Output}{Reference\ Output} = \frac{O_{LM}}{{LLT}/{CT_{BN}}}$$

where:

- $O_{LM}$ = the output released by the last machine of the line;
- LLT = Line Loading Time;
- $CT_{BN}$ = The ideal cycle time of the bottleneck machine.

From the definition of machine valuable time (MVT), the output released by the last machine can also be expressed as the product of the ideal cycle time and the valuable time of the last machine of the line:

$$O_{LM} = CT_{LM} \times MVT_{LM}$$

Put into this form, OEEML can now be expressed as a function of the TOEE of the last machine:

$$OEEML = \frac{MVT_{LM}/CT_{LM}}{LLT/CT_{BN}} = \frac{CT_{BN}}{CT_{LM}} \times TOEE_{LM}$$

Although the OEEML of the line can be evaluated using the total OEE of the last machine only, it is evident that the last machine does account only for a little share of the total inefficiency of the line.

# 2 Buffers in the production systems

The availability improvement is one of the most challenging problem for operations managers. High availability in productions system brings to high productivity and quick response to market changes (*Battini et al., 2013*). Machine breakdowns are important causes of variability increase in process times and flows of production system, leading to reduced manufacturing performance (*Hopp and Spearman, 2000*). The buffer design problem is common in different industrial sectors (*Gonzàlez and Alarcòn, 2009*).

In particular case, when the production system is an automated high throughput production system, the buffer allocation and sizing can be used to improve the availability of the system, especially if the breakdowns are of short duration (defined as “micro-downtimes”).

Some studies address the optimization of buffer capacity from an analytical point of view, using probabilistic modelling methods of the system and proposing solution algorithms that are processed through linear programming. In this case, the models involve complex and articulated formulations that often make the approach unsuitable for industrial application. For this reason, many authors believe that the simulative method using a software is the best way to face the BAP and to carry out important managerial choices in a short time.

The goal of this chapter is to discuss in detail the dynamics that take place in a system in the production phase.

The first part talks about the theoretical concepts useful for the description of a productive system, with focus on the role of buffers and the Buffer Problem Allocation (BAP).

The second part describes a buffer sizing method by simulation approach based to fundamental reliability parameters, like MTTR and MTBF (Battini et al., 2013).

## 2.1 Literature Review of BAP

The buffer allocation problem (BAP) concerns the size and location of storage between the stages of a flow line. This is a critical research area in the design of production lines. A production line is defined as a series of workstations that are linked from one station to the other (*Ameen et al., 2018*). Flow lines are typically affected by breakdowns due to the variation in the processing times and failure of the workstations. Thus, a solution to reduce these negative effects is to allocate buffers in between the machine. The buffer storage increases the throughput of the production line by reducing the blocking and starving time of the workstations and in the same time, it improves the flow line efficiency. The buffer allocation problem translate into an optimization problem of stochastic system involving many variables (*Tezcan and Gosavi, 2001*). In production line n-1 storage buffers exist for a n-stage line, as shown in figure 9 (*Roser et al., 2004*).

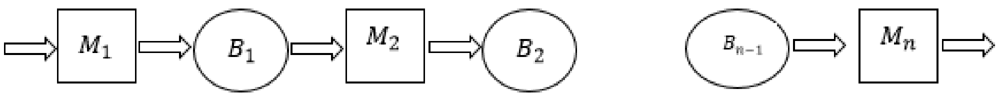



*Figure 9 - Buffer storage in flow lines*

The literature on BAP can be classified based on the following criteria (*Weiss et al., 2018*):

1. The characteristic of the flow line: for example, assumptions about whether stations are reliable or unreliable and whether the stations in the line are identical, a balanced line or not;
2. The considered objective function and contrains: for example, the throughput, the Work-In-Process inventory (WIP), and the cycle time.
3. The solution method: it is used to achive the buffer allocation. There are some different method, with different approach. It is possible to identify integrate optimization methods, and iterative optimization methods.

In 1959, Koenigsberg described first the optimization problem of allocation buffer capacities. Quantitative decision support for the BAP dates back to Barten (1962) and has since attracted the attention of many researchers. There are two reasons why this is a difficult problem to solve. First, the exact performance evolution of flow line is only possible for small system under specific assumptions; second, the allocation problem of buffer capacities in an NP- hard optimization problem (*Smith and Cruz, 2005*). This means that currently, there are no well-defined algorithms that can optimize in polynomial time (*Smith and Cruz, 2005*). Therefore, the exact solution for the BAP exist only for special cases by approximate solution methods.

Over the years, a large amount of research has been devoted to the production line. Much of this study and analysis has involved the design of manufacturing systems when there is a considerable inherent variability in the processing times at the various station, a common situation with human operators/assemblers (*Papadopoulos and Vidalis, 2001*).

The literature on the modelling of production line is very extensive. Thus far, Demir *et al.* (2014), Gershwin and Schor (2000), Hudson *et al.* (2015), Papadopoulos *et al.* (2009), and Weiss *et al.* (2015) have given reviews of subsets of the available literature on the BAP. With respect to the three classification criteria introduced above, Weiss *et al.* (2018) have grouped the researches made in the past in the following way:

1. Demir *et al.* (2014), Gershwin and Schor (2000), and Papadopoulos *et al.* (2009) discern whether stations are reliable or may fail. Papadopoulos *et al.* (2009) further distinguish short lines with up to six stations from longer lines. Demir *et al.* (2014) make a distinction between serial line and more complex network. Gershwin and Schor (2000) state the probability distributions used in the reviewed references. Weiss *et al.* (2015) also provide the probability distribution but just for unreliable line. Hudson *et al.* (2015) focus on unbalanced flow lines.

2. Gershwin and Schor (2000) and Weiss *et al.* (2015) talk about the Primal, the Dual, and the Profit Problems. Papadopoulos *et al.* (2009) describe the Primal, the Dual, and WIP minimization problems. Demir *et al.* (2014) and Hudson *et al.* (2015) categorize the optimization problems based only on the performance measures considered in the objective function.
3. Gershwin and Schor (2000) use the performance evaluation method applied in the optimization methods to split between methods with exact numerical evaluation, simulation-based evaluations, and other methods. Demir *et al.* (2014), Papadopoulos *et al.* (2009) and Weiss *et al.* (2015) classify all methods into evaluative and generative parts. However, no further classification of the solution methods described in the references under review is provided. Hudson *et al.* (2015) do not provide a description of the solution methods.

Anyway, this work try propose a simulative method to achieve a buffer size considering the different kind of failures, i.e., considering the different lifetime phase in which the machines are working, especially aims to the failure that are not predictable. Then, the results of a simulation that carry out a buffer size, is compared with  Battini formula proposed in the Battini, Faccio and Persona paper (2013). In this regard they propose a new exhaustive matrix, that can help designers understand the methodology potetials according to different reliability parameters depending on the types of machines considered, as well as determine the optimal buffer size in order to maximise the throughput and minimise downtime costs.

## 2.2 Some general concepts of an automatic flow production line

Production plants based on automated lines can be considered as flow systems consisting of material, work areas and storage areas with a certain throughput. It is important to mention to the following concepts:

- Material: raw material or a semi-finished product that undergoes a series of transformations and exit the system as a finished product;

- Workstation: machines that perform certain operations on the material in a certain time. The time spent in the work areas is given by the process time, the duration and frequency of the micro-downtimes and the repair time;

- Storage area: transport systems or storage systems, with a maximum limit of capacity. These stock areas are called buffers;

- Production capacity: is the maximum output that can be produced in a business with available resources.

## 2.3 The analysis of internal dynamic

The main parameter to describe the system is the throughput of the production line, understood as the maximum flow of material leaving the line in a unit of time. To evaluate the throughput performance, also known as mechanical efficiency, it is necessary to compare it with the nominal potential of the system and reflect in terms of efficiency.

The efficiency index of the system depends on the internal dynamics of the system itself and on the specific characteristics of the line. In particular, efficiency is influenced by:

- Characteristics of the micro-downtimes of each machine, expressed in terms of efficiency (isolated efficiency);
- Buffer capacity, which translates into the level of decoupling that they are able to provide to the section.

### 2.3.1 The starving and blocking condition of a production line

With reference to the ASME notation for the representation of production systems, the following diagram represents an automated line consisting of machines and i-1 buffers.

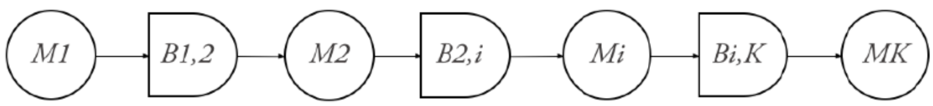



*Figure 9 – Scheme of a production line*

The scheme above illustrates how the incoming material crosses the line in the direction indicated, passing from machine 1 to machine i until leaving the system. In standard conditions, when the machine $M_i$ works, the level of buffer $B_{i-1;i}$ decreases and the level of buffer $B_{i;i+1}$ increases. Similarly, when $M_i$ is stopped, the level of buffer $B_{i-1;i}$ tends to increase, while the level of $B_{i;i+1}$ tends to decrease.

If the stop of the work of the machine $M_i$ persists, the buffer upstream can get filled up to saturation and the buffer downstream can be completely emptied.

These conditions have been described in depth by Dallery and Gershwin (1992), defining them as a state of starving and a state of blocking. Considering a section composed of machines A and B separated by a buffer of a certain capacity, the starving and blocking times are defined as following:

- The starving time: the downstream machine (B) cannot operate due to lack of incoming material. After the interruption in the work of the machine A, the buffer was completely emptied;
- The blocking time: the upstream machine (A) cannot operate due to lack of unloading space. After the interruption in the work of the machine B, the intermediate buffer arrived at complete saturation.

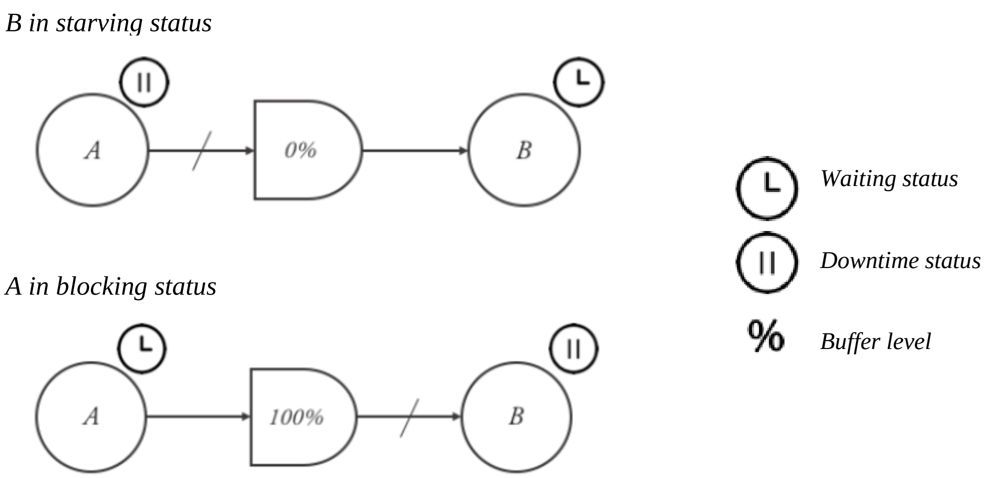



*Figure 11 – Starving and blocking conditions*

The blocking and starving conditions represent the origin of the loss of efficiency of the system, because the interruption of the work of a single machine leads to the inactivity of the adjacent machinery and is then transmitted to the entire line.

If the buffer is correctly sized with respect to the uptime and downtime of the machines, the overall productivity of the system does not undergo some reductions, because the lung is able to compensate for the effect of the micro-breakdowns (Dallery et al., 1992).

### 2.3.2 Nominal throughput of the system

A production plant consists of different systems that perform a specific processing on the product and require a certain process time, defined as cycle time. The cycle time, typically expressed in s / pcs, is the minimum time for the execution of a single product and represents the nominal speed of the machinery. The stadium with the greatest cycle time is defined as the bottleneck of the system and determines the production rate of the entire line. It follows that, in standard operating conditions and without process stops, the bottleneck speed represents the nominal throughput of the system.

$$Q_S(t) = Q_{\max BN}(t)$$

where:

- $Q_S(t)$ = The nominal throughput of the system;
- $Q_{\max BN}(t)$ = The throughput of the bottleneck of the system

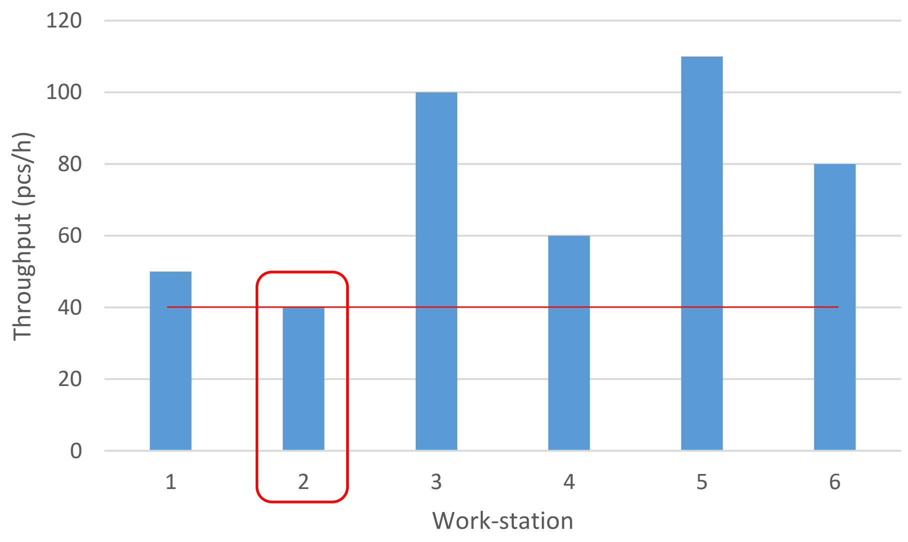



*Figure 12 – Histogram of nominal throughput*

If we want to evaluate the performance of the system in terms of efficiency, the mechanical efficiency is defined as:

$$E_M = \frac{Q_{BN}(t)}{Q_{max}}$$

where:

- $Q_{BN}(t)$ = The throughput of the bottleneck of the system
- $Q_{max}$ = The maximum throughput level

It is clear that if the bottleneck is not subject to any stops in the process, and mechanical efficiency is equal to 100%.

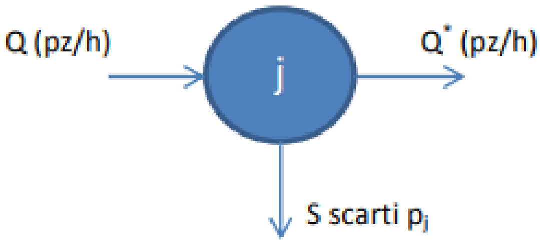
The system bottleneck can be identified by a load histogram, which shows the cycle times of the different stages of the line. This kind of bottleneck is defined theoretical bottleneck, due to the nominal potential of the machines has been considered. Knowing how to identify the nominal bottleneck is essential to establish which machine determines the maximum throughput and consequently define the target speed value with which to measure the performance of the plant. However, in real operating conditions, it may happen that the real bottleneck is different than the nominal one. The real bottleneck (BN) is the machine, which reduces the productivity of the system to a greater extent than the others. Identifying it and improving its performance are some of the most important activities in a productive environment with a view to continuous improvement (Chiang et al., 1998).

### 2.3.3 Analysis of the throughput of a single work-station

Under real conditions, all the machines are subject to arrests or slowdowns, which we have defined as micro-downtimes, according to the Six Big Losses theory introduced in the TPM by Nakajima. The micro-downtimes are short stops that occur during the normal operation of the system. They have certain characteristics of frequency and duration depending on the cause from which they are generated.

In general, with reference to an operating machine or station j, engaged in a phase of the technological cycle, the production of that phase is:

$$\frac{Q_{nom}}{Q_{real}} = \eta$$




with:

$$\eta = K_1 \times K_2 \times K_3 \times K_4$$

where:

- $Q_{nom}$ = the nominal throughput of the machine;
- $Q_{real}$ = the real throughput of the machine;
- $K_1$ = the scrap coefficient;
- $K_2$ = the availability coefficient;
- $K_3$ = the operator performance coefficient;
- $K_4$ = coefficient of the effective use of the machine.

Therefore, in general the throughput of the line, the station or the machine to be sized is:

$$Q_{real} = Q_{nom} \times \eta = Q_{nom} \times K_1 \times K_2 \times K_3 \times K_4$$

The $K_2$ parameter can be also calculated in function of the MTTF and MTTR parameters. If we consider the MTTF and MTTR constants, the impact of the micro-downtimes on the throughput of a single work-station is the following:

$$E_j = K_2 = \frac{MTTF_j}{MTTF_j + MTTR_j} = \frac{1}{1 + (\lambda_j \times MTTR_j)}$$

with:

- $MTTF_j$= Mean Time to Failure of work-station j;
- $MTTR_j$= Mean Time to Repair of work-station j;
- $\lambda_j$= Downtime ratio.

Therefore, it is possible to recalculate the throughput and the real process time of each work-station, taking into account only the efficiency decrease due to the micro-downtimes:

$$Q_{real} = Q_{nom} \times E_j$$

$$T_p = \frac{T_C}{E_j} = T_C \times (1 + \frac{MTTR_j}{MTTF_j})$$

From the figure below, considering the throughput of a single work-station, the real bottleneck of the system consists of a different machine with respect to the nominal bottleneck.

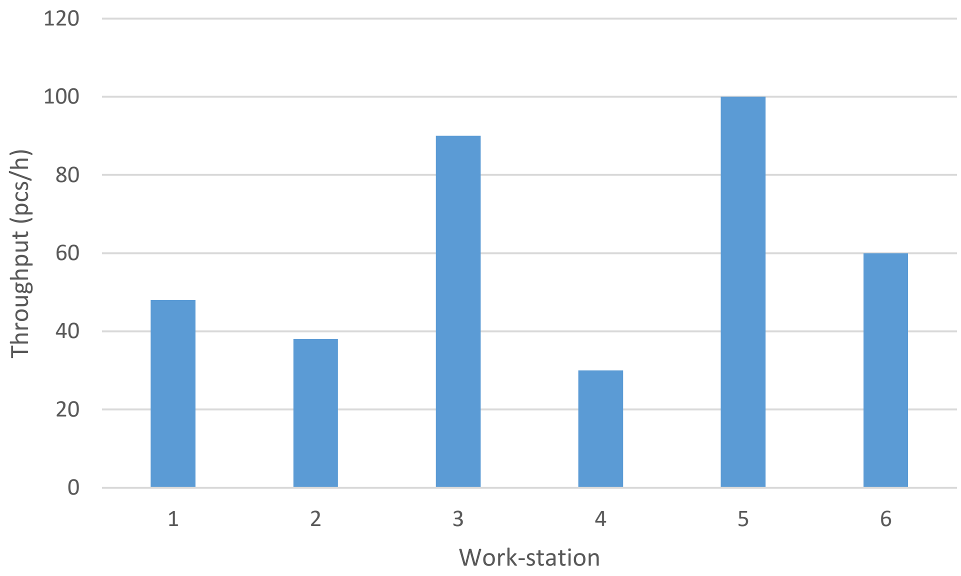



*Figure 13 – Histogram of real throughput*

## 2.3.4 The buffer role

In a production plant the ideal objective would be to eliminate the micro-downtimes even if a more realistic hypothesis is certainly the reduction of the micro-downtimes themselves because a part of these can be considered physiological and inherent to the production process.

Regardless of the number and duration of the micro-breakdowns, the buffers play a role of fundamental importance to reduce the impact of these on the OEE and to increase the reliability of the system. Buffers are storage zone placed between two machines in an automated production line; their task is to decouple the equipment in series and to manage small machine stops or delays of the upstream machine, so that the downstream machine can continue its regular productive function.

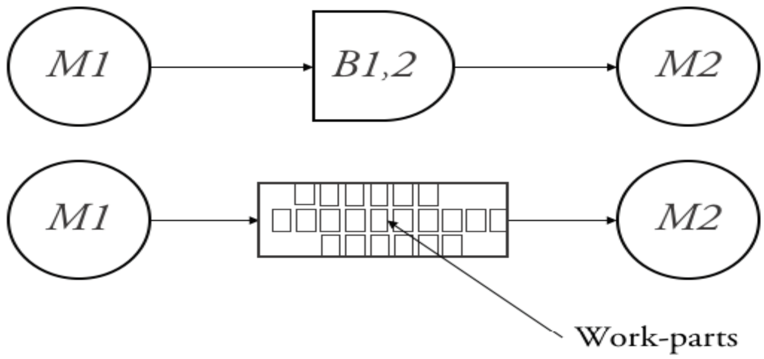



*Figure 14 – Exemple of interoperational buffer*

Furthermore, the use of buffers allows setting different speeds for the various machines, in order not to work with a fixed-rate line where the production rate is determined by the slowest machine and all line must adapt to that speed; without the use of buffers, a machine stop would block the entire process.

The insertion of storage zones prevents companies from improperly purchasing oversized equipment and makes it possible to better balance a production system (*Battini et al., 2006*).

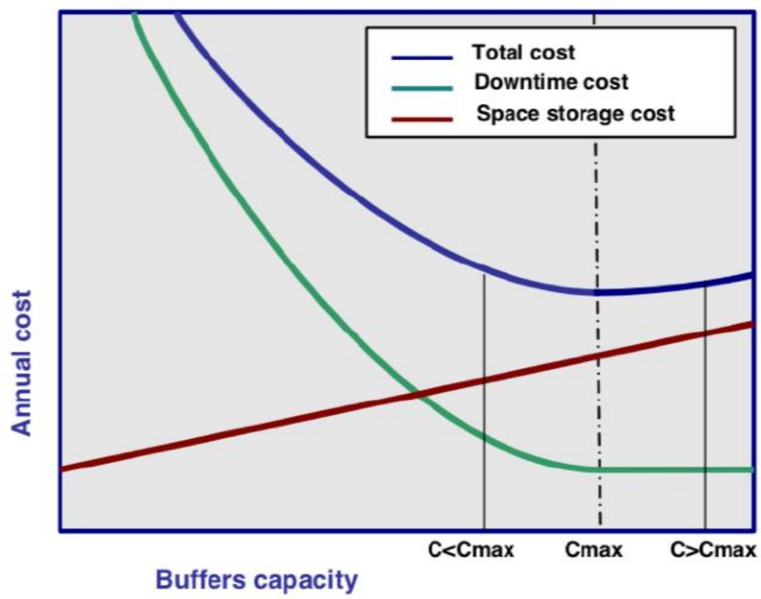



*Figure 15 – Annual cost – Buffers capacity (Battini et al., 2006)*

Buffers also allow a workstation to not be affected by all the losses of previous stations. The main errors in the design of a buffer are linked to the lack of consideration of downtime costs, with attention just on storage costs, with a buffer size often constrained by physical limits of space. Even considering the downtime costs, a typical mistake is to estimate the reliability of the production systems only with average values, based on the manufacturers' data and without considering the phase of the machine life cycle or the downtimes (*Faccio et al., 2013*). Furthermore, even not considering the reliability parameters correctly, can lead to an underestimation of downtime costs, with a consequent under-sizing of the buffer.

## 2.4 Buffer design for availability (BDFA) - Buffer sizing method

The study carried out by *Battini et al. in 2009* it proposes a simulation model that, starting from the reliability parameters of the machinery, provides an effective tool for sizing the buffer, with a view to maximizing system productivity. This study introduced a new approach to the study of buffers with a view to maximizing machine availability, with the advantage of being able to describe the system using experimental formulas and then using a synthesis matrix to derive quickly design values.

The Buffer Design For Availability (BDFA) approach is developed by initially considering only systems in the final phase of the life cycle, subject to failures caused by wear and with increasing failure rate λ(t).

However, since this hypothesis did not allow the application of the BDFA model to all industrial contexts, *Faccio et al. (2013)* extend this sizing method, also considering failures and stops of the early failures type (decreasing λ(t)) and random failures (constant λ(t)). This type of stops is typical of modern production systems, where continuous setups and adjustments tend to cause micro-breakdowns with random and unpredictable trends. The study focuses on high-productivity automatic systems, typical of industrial sectors in which buffer optimization is a critical element due to the high downtime costs.

The simulation carried out concerns two stations, which are part of the critical section of the line, since a stop in this section causes a loss of efficiency for the entire system. A buffer with a certain level of capacity is placed between the two stations, in order to reduce downtime costs and improve plant productivity. The reliability parameters considered are the following:

- MTTF = Mean time to failure
- MTTR = Mean time to repair
- $A = \frac{MTTF}{MTTF+MTTR}$ = Average availability

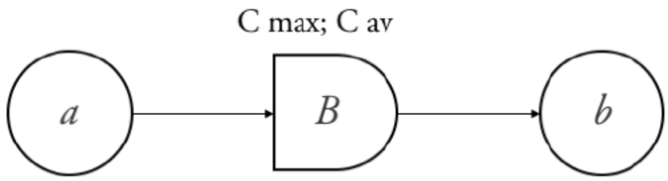



*Figure 16 – ASME representation*

The simulation analysis was carried out by independently modifying the parameters of each station, in order to simulate different trends for up time and down time. Furthermore, the critical section can be described by parameters G, P and R (Battini et al, 2009).

$$G = MTTR_{max} = Max\ [MTTRa; MTTRb]$$

$$P = \frac{A(a)}{A(b)}$$

$$R = \frac{MTTR(a)}{MTTR(b)}$$

The capacity of the intermediate buffer is set at a level that can not be reached during the simulation process. The duration of the process simulation is 10,000 minutes.

The output of the model are:

- Cmax = maximum level of buffer capacity
- C av = average level of buffer capacity

The uptime trend has been described by the Weibull distribution, defined by the shape β and scale α parameter. The use of this probability distribution made it possible to simulate the process in the three types of stops, modifying the parameters α and β.

- Early failures with decreasing λ(t) and β<1;
- Random failures with constant λ(t) and β=1;
- Wear out zone with increasing λ(t) and β>1.

| *Model inputs* | | | | | | | | | | | | | | | *Model outputs* | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| *WSa* | | | | | | | *WSb* | | | | | | | | | | |
| | | *Up times* | | | *Down times* | | | | *Up times* | | | *Down times* | | | | | |
| *Q'a (pieces/hour)* | $A_1$ | $\alpha_a$ | $\beta_a$ | *$MTTF_a$ (min)* | *$MTTR_a$ (min)* | $\sigma_a$ | *Q'b (pieces/hour)* | $A_2$ | $\alpha_b$ | $\beta_b$ | *$MTTF_b$ (min)* | *$MTTR_b$ (min)* | $\sigma_b$ | | *Cmax (pieces)* | *Cav (pieces)* | *Q (pieces/hour)* |
| 90.00 | 0.954 | 0.3 | 20 | 187.05 | 9.11 | 0.09 | 89.62 | 0.958 | 0.3 | 20 | 187.79 | 8.32 | 0.03 | | 148 | 44.64 | 81.68 |
| 90.00 | 0.991 | 0.4 | 20 | 109.13 | 0.94 | 0.07 | 90.84 | 0.982 | 0.4 | 20 | 108.61 | 1.95 | 0.15 | | 64 | 19.86 | 88.15 |
| 89.69 | 0.985 | 0.5 | 20 | 46.75 | 0.70 | 0.03 | 90.00 | 0.982 | 0.5 | 20 | 47.20 | 0.87 | 0.11 | | 22 | 3.48 | 87.93 |
| 88.01 | 0.994 | 0.6 | 20 | 33.71 | 0.21 | 0.01 | 90.00 | 0.972 | 0.6 | 20 | 33.01 | 0.96 | 0.02 | | 15 | 2.79 | 87.31 |
| 91.01 | 0.976 | 0.7 | 20 | 25.74 | 0.63 | 0.07 | 90.00 | 0.987 | 0.7 | 20 | 25.58 | 0.34 | 0.02 | | 10 | 1.33 | 88.61 |
| … | … | … | … | … | … | … | … | … | … | … | … | … | … | | … | … | … |

*Figure 17 – Exemple of simulation output (Faccio et al., 2013)*

The downtime trend has been described with a normal distribution, defined by a given value of MTTR and of σ. It is also assumed that the downtime costs, are much higher than the costs of the WIP being on the buffers. For this reason the costs of stocks management  are negligible compared to downtime costs and the optimal buffer capacity is the maximum level (Cmax) obtained during the simulation.

The result of the study can be expressed by the following formula which links the optimal level of buffer capacity and the parameters of the critical section:

$$C_{max} = K(P, R, \beta) \times G \times Q$$

where:

- Cmax is the maximum capacity buffer level (pieces), and it is also the optimal assumption level;
- Q is the level of production throughput in pieces/hours;

- G is definied as *max* MTTR between the two WSs (MTTRa, MTTRb) and it is measured in hours;
- K is a "safety factor", function of P and R (*Battini et al., 2009*), but also function of β, shape parameter of the Weibull distribution.

| | | | *β* | | | | |
|---|---|---|---|---|---|---|---|
| | | | *First infant failures* | *Last infant failures and random failure* | *First wearout failures* | | *Predictable failures (Battini et al., 2009)* |
| | | | $\beta < 0.7$ | $0.8 \leq \beta \leq 1$ | $\beta = 2$ | $\beta = 3$ | $\beta > 3$ |
| *R* | $R < 0.95$ | 6.2 | 7.5 | 6.2 | 3.6 | 2.6 | 1.8 |
| | $0.95 \leq R \leq 1.05$ | 6.1 | 6.2 | 6.1 | 3.3 | 2.4 | 1.1 |
| | $R > 1.05$ | 5.9 | 6 | 5.8 | 3.2 | 2 | 1.5 |

*Figure 18 – Cross matrix of K (Faccio et al., 2013)*

Starting from the matrix in the figure above, it is possible to derive the safety factor according to the characteristics of the micro-breakdowns, described by the parameter R, and the life cycle phase in which the machines are located (value β in the curve bathtub).

# 3 Simulation modelling

The systems can be seen as a number of interconnected processes. Therefore, in order to improve the performance of an organization it is necessary to study the design of these processes and the resources they consume need to be studied. Simulation provides a way of experimenting with a model (i.e. simplified representation) of a system in order to understand its behaviour under a number of scenarios.

In this section, we introduce the nature of simulation modelling, defining some central concepts as simulation, system and model. We then propose a classification of the different types of simulation to introduce Discrete Event Simulation. Subsequently, a simulative method is proposed and explained that has been applied step by step to carry out the case study (chapter 6). The method taken into consideration is the *General Methodology for Applying Simulation to Problem Solving* (Rossetti, 2015) that identifies five major phases: problem formulation, simulation model building, experimental design and analysis, evaluation and iteration, implementation. They will be discussed in detail gradually by identifying the steps to follow the good execution of the methodology.

## 3.1 What is simulation modelling

This thesis is based on the principles and methods of simulation modelling that is a mathematical business model, which combines both mathematical and logical concepts to try to emulate a real life system through the use of computer software. The models that are built and employs are called simulation models. Therefore, when you execute a simulation model you are performing a simulation.

Before speaking specifically of the simulation, we define the following important concepts. The system is the facility or process of interest. To study it, we have often to make some assumptions about how it works. These assumptions, which usually take the form of mathematical and logical relationships, constitute the model that is used to try to gain some understanding of how the corresponding system behaves.

For models simple enough, it may be possible to use an analytic solution that use some mathematical methods to obtain exact information on question of interest. However, most real-world systems are too complex and then they must be studied by means of simulation. In a simulation, we use a computer to evaluate a model numerically and it is essential to collect data to estimate the characteristics of the model being represented. Either way, the first rule to remember about simulation is that it is only a representation of real thing but it is not the real thing.

*Ravindran et al. (1987)* defined computer simulation as "A numerical technique for conducting experiments on a digital computer which involves logical and mathematical relationships that interact to describe the behaviour of the system over time".

In other words, a simulation is the imitation of the operation of a real-world process or system over time. The behaviour of the system is analysed by developing a simulation model. From a set of assumption about the operation of the system, the model is carried out. These assumption consist of mathematical, logical, and symbolic relationship between the entities, or objects of interest, of the system. The following steps are the development and the validation of the simulation. After these, a model can be used to investigate a wide variety of "what if" questions about the real-world system. It is possible to simulate potential changes of system, in order to predict their impact on a system performance. Moreover, the simulation can also be used to analyse in the design stage. Thus, simulation modelling can be used for two different tools: at first, as an analysis tool for predicting the effect of changes to existing systems; then, as a design tool to predict the performance of new system under varying sets of circumstances.

Sometimes the model can be developed using mathematical methods like differential calculus, probability theory, algebraic methods, or other mathematical techniques. At the end, the evaluation of the system is achieved by a measurement of the performance of the system itself using one or more numerical parameters. However, many real system are so complicated to be model and then to be used to solve mathematically.

In these instances, numerical, computer based simulation can be used to represent the behaviour of the system over the time. From the simulation, data are collected as if a real system were being observed. This simulation-generated data is used to evaluate the measures of performance of the system.

Simulation is obviously appealing to a customer because it allows to imitate what happens in a real system or what is made out for a system that is in a design stage. There are many reasons that push to choose the simulation as a technique in problem solving, for example, the output data should represent the outputs of the real system. Another example is the possibility to achieve a simulation model with precise assumptions.

Simulation has many advantages, but also some disadvantages. There are a list by *Pegden, Shannon, and Sadowski (1995).* Some advantages are:

1) New policies, operating procedures, decision rules, information flows, organizational procedures, and so on can be explored without disrupting ongoing operation of the real system;
2) New hardware design, physical layouts, transportations system, and so on can be tested without committing resources for their acquisition;
3) Hypothesis about how or why certain phenomena occur can be tasted for feasibility;
4) Time can be compressed or expanded to allow for a speed-up or slow-slow down of the phenomena under investigation;
5) Insight can be obtained about the interaction of variables;
6) Insight can be obtained about the importance of variables to the performance of the system;
7) Bottleneck analysis can be performed to discover where work in process, information, materials, and so on are being delayed excessively;
8) A simulation study can help in understanding how the system operates rather than how individuals think the system operates:
9) "What if" questions can be answered. This is particularly useful in the design of new systems.

Some disadvantages are these:

1) Model building requires special training. It is an art that is learned over time and through experience. Furthermore, if two models are constructed by different competent individuals, they might have similarities, but it is highly unlikely that they will be the same;
2) Simulation results can be difficult to interpret. Most simulation outputs are essentially random variables (they are usually based on random inputs), so it can be hard to distinguish whether an observation is a result of system interrelationships or of randomness;
3) Simulation modelling and analysis can be time consuming and expensive. Skimping on resources for modelling and analysis could result in a simulation model or analysis that is not sufficient to the task;
4) Simulation is used in some cases when an analytical solution is possible. This might be particular true in the simulation of some waiting lines where closed-form queueing models are available.

Simulation is a useful and powerful tool for many application areas:

- Designing and analysing manufacturing system;
- Evaluating hardware and software requirements for a computer system;
- Evaluating a new military weapons system or tactic;
- Determining ordering policies for an inventory system;
- Designing communications system and message protocols for them;
- Designing and operating transportation facilities such as freeways, airports, subways, or ports;
- Evaluating designs for service organizations such as hospitals, post offices, or fast-food restaurants;
- Analysing financial or economic system.

# 3.2 Type of simulation (description of a general type of system)

When we referring to simulation modelling, we need to establish what we mean by system. A system is defined as a set of interrelated components working together toward a common objective (*Blanchard and Fabryckty, 1990*).

In engineering literature, there is also a broader definition of system: "A system is a composite of people, products and processes that provide a capability to satisfy stated needs. A complete system includes the facilities, equipment (hardware and software), materials, services, data, skilled personnel, and techniques required to achieve, provide and sustain system effectiveness." Air Force Systems Command (1991).

In practice, what we mean with "system" depends on the objective of a particular study. In fact, some items that compose an entire system might be just a subset of another system. We define the state of a system as a collection of variables necessary to describe a system at a particular time, relative to the objectives of a study.

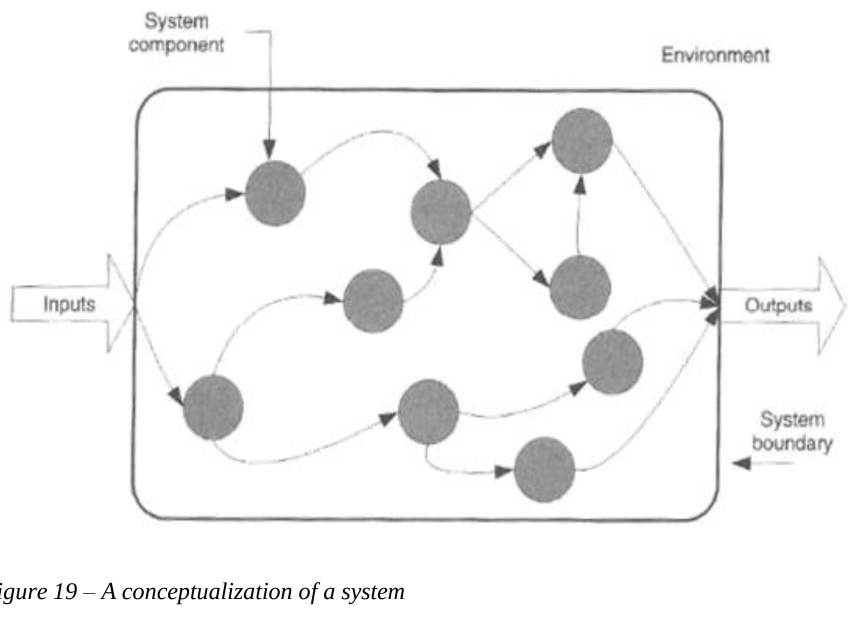



*Figure 19 – A conceptualization of a system*

As illustrated in the figure, the environment is made up of many systems connected to each other, and typically, some input is needed to understand some output using internal components. The intended use of the model and how you fell the system will influence the composition of the model.

It is worthwhile discussing some general system classifications because it conditions the modelling. We talk about stochastic system if stochastic or random behaviour is an important component of the system; otherwise, it is called deterministic system. Moreover, stochastic or deterministic systems can be each divided each into static or dynamic systems: if a system does not change significantly with respect to time, it is said to be static system. When we talk about dynamic system, we might want to consider how it evolves with respect to time. Then dynamic system is divided by another classification: if the state of the system changes at discrete points in times, it is called discrete; otherwise if the system changes continuously with time, it takes the name of continuous.

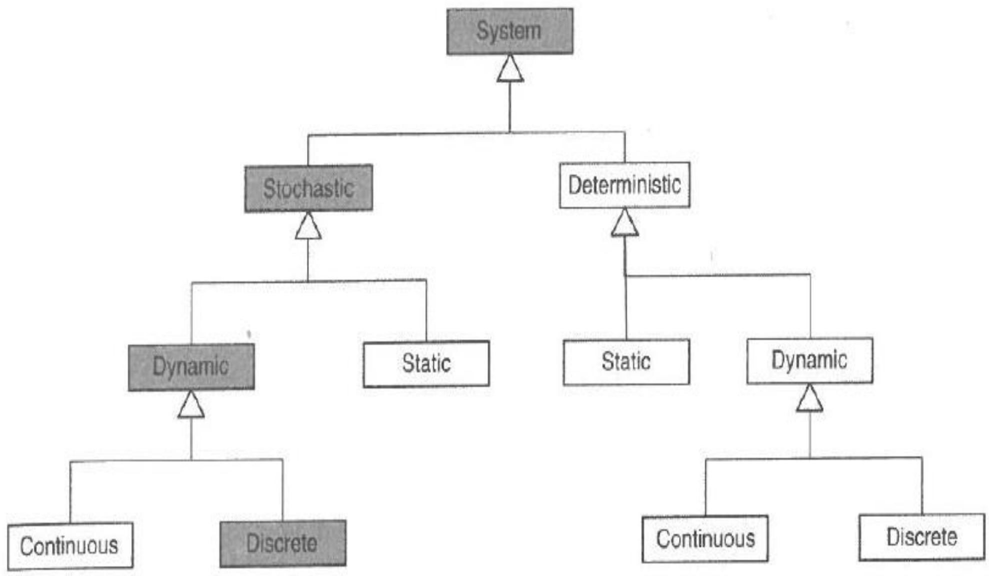



*Figure 20 – General type of system*

The main propose of a simulation model is to allow observations about a particular system to be gathered as a function of a time. From that standpoint, there are two distinct types of simulation models: discrete event and continuos.

Just as discreate systems change at a discrete points in time, in a discrete-event simulation observations are gathered at selected points in time when certain changes take place in the system. These selected points in time are called events. On the other hand, continuos simulation requires that observations be collected continuously at every point in time (or at least that the system is described for all point in time). In the first case, system does not need to be osserved on a contiuous basis. The system need only be osserved at selected discrete points in time, resulting in the applicability of a discrete- event simualation model. In the second case, a model of the system must describe the rate of flow over time and the output of the model is presented as a function of time. System such as thede are often modeled using differential equations. The solution of these equations involves numerical methods that integrate the state of modeled system over time. This, in essence, involves dividing tie into small equal intervals and stepping through time. Often both the discrete and continuous viewpoints are relevant in modeling a system.

## 3.3 Randomness of the simulation

Normally many situations that happen every day can be defined as an event or processes that happen randomly. The fact that this random component is present in these processes does not mean that randomness cannot be modelled or described. The difficulty and at the same time the fundamental point of the simulation lies in representing this randomness.

One of the ways to model this randomness is to describe the phenomenon as a random variable governed by a particular probability distribution. In this sense, it is very important to gather information through direct observation of the system or the use of historical data. If neither source of information is available, then some plausible assumptions must be made to describe the random process by a probability model. If historical data is available, there are two basic choices for how to handle the modelling:

1) To develop a probability model, given data;
2) To try to drive the simulation directly from the historical data.

The latter approach is not recommended. First, it is extremely unlikely that the captured data will be in a directly usable form. Then, it is even more difficult for data to correctly represent situations that are then simulated through modelling. Following the development of the probability model, statistics intervenes as a tool used to obtain a uniform distribution of random numbers in the interval (0,1). These random samples are then used to map the future occurrence of an event on the time scale.

## 3.4 Discrete event simulation (DES)

Discrete event simulation (DES) is a form of computer-based modelling that provides an intuitive and flexible approach to representing complex systems. In particular, DES represent complex behaviour within, and interactions between individuals, populations, and their environment. The main features of DES consist in the fact that it moves forward in time at discrete intervals and that the events can be considered mutually exclusive. In this way, DES appears to be characterized by good flexibility and efficiency and thus, enables to solve a large number of problems.

DES was developed in the 1960s in industrial engineering and operations research to help analyse and improve industrial and business processes (*Karnon et al., 2012*).

In the discrete event simulation, the system is represented, in its evolution over time, with variables that instantaneously change their value in well-defined instants of time belonging to a countable set. These moments are those in which events occur. It is clear that, as these models are of a dynamic nature, it is necessary to record, or keep memory, of the (simulated) time that proceeds. In particular, it will be necessary to define a time advancement mechanism to make the simulated time proceed from one value to another. The variable that in a simulation model provides the current value of the simulated time is called "simulation clock", and there are two ways to define its progress:

- progress of the time to the next event;
- progress of the time in pre-set increments.

The first is certainly the most widespread. In this case, the "simulation clock" is initialized to zero and is moved forward at the time of the first of the future events; then the system is updated taking into account the event that occurred, the timing of future events is updated and the procedure is repeated. Unlike progress in pre-set increments, periods of inactivity are not considered.

The core concepts of DES are entities, attributes, events, resources, queues, and time.

**Entities**

Entities are individual elements of the system that must be defined. Entities is defined dynamic entities when it flows inside the system. Otherwise, they are defined static entities. Events are characterized by attributes, experience events, consume resources, and enter queues, over time. They can be created whenever it is appropriate to the problem and the time of a relevance to an entity may be a subset of the situation time. Entities can be grouped into classes that are sets of entities of the same type.

**Attributes**

Attributes are features specific, which provide a value of a data assigned to the entity itself. They allow entity to carry information and to understand how an entity responds to a give data set of circumstances. Attributes can be modified during the simulation and also analysed further outside of the simulation itself.

**Events**

An event is defined as any instantaneous circumstance that causes the value of at least one of the state variables to change. There are events outside the system (exogenous events) and internal events (endogenous events). In other words, events are things that can happen to an entity or the environment.

**Resource**

Resources are elements of the system that provide a service to entities. An entity can request one or more resource units and if this is not available the entity will have to, for example, put itself in a queue waiting for it to become available, or take another action. If instead the resource is available, it is "captured" by the entity, "retained" for the necessary time and then "released".

There are also other important concepts about discrete event simulation. In particular, if for example, the resources are occupied and the entities cannot therefore be taken, the queues are formed. *Queues* can have a maximum capacity, and alternative approaches to calling entities from queues can be defined: first-in-first-out and last-in-first-out.  Another example is *interaction* that happened when two or more entities compete over a resource.

## 3.5 Simulation methodology

A methodology is defined as a series of steps to follow. The aim of the simulation methodology is to analyse the system through the general precepts of solving problem. A general methodology for solving problem can be stated as follows:

1) Define the problem;
2) Establish measures of performance for evaluation;
3) Generate alternative solution;
4) Rank alternative solution;
5) Evaluate and iterate during process;
6) Execute and evaluate the solution.

This methodology can be define DEGREE methodology referring to the first letter of each step. It represents for problem solving a series of steps that can be used during the problem-solving process. The first step helps to confirm that you are solving the right problem. The second step helps to ensure that you are solving the problem for the right reason, that is, your metrics must be coherent with your problem. Steps 3 and 4 make confirm that the analyst looks at and evaluates multiple solutions to the problem. In step 5, the analyst evaluates how the process is proceeding and allows for iteration. Iteration is an important concept; in particular, it recognizes that the problem-solving process must be repeated until the desired degree of modelling fidelity has been achieved. Start the modelling at a level that allows it to be initiated and do not try to address the entire situation in each of the steps. Start with small models that work and build them up until you have reached your desired goals. It is important to get started and get something established on each step and continually go back in order to ensure that the model is representing reality in the way that you intended. The last step, step 6, indicates that if you have the opportunity, you should execute the solution by implementing the decisions. Finally, you should always follow up to confirm that the projected benefits of the solution were obtained. The DEGREE problem-solving methodology should serve you well; however, simulation involves certain unique actions that must be performed during the general overall problem-solving process.

When applying DEGREE to a problem that may require simulation, the general DEGREE approach needs to be modified to explicitly consider how simulation will interact with the overall problem-solving process.

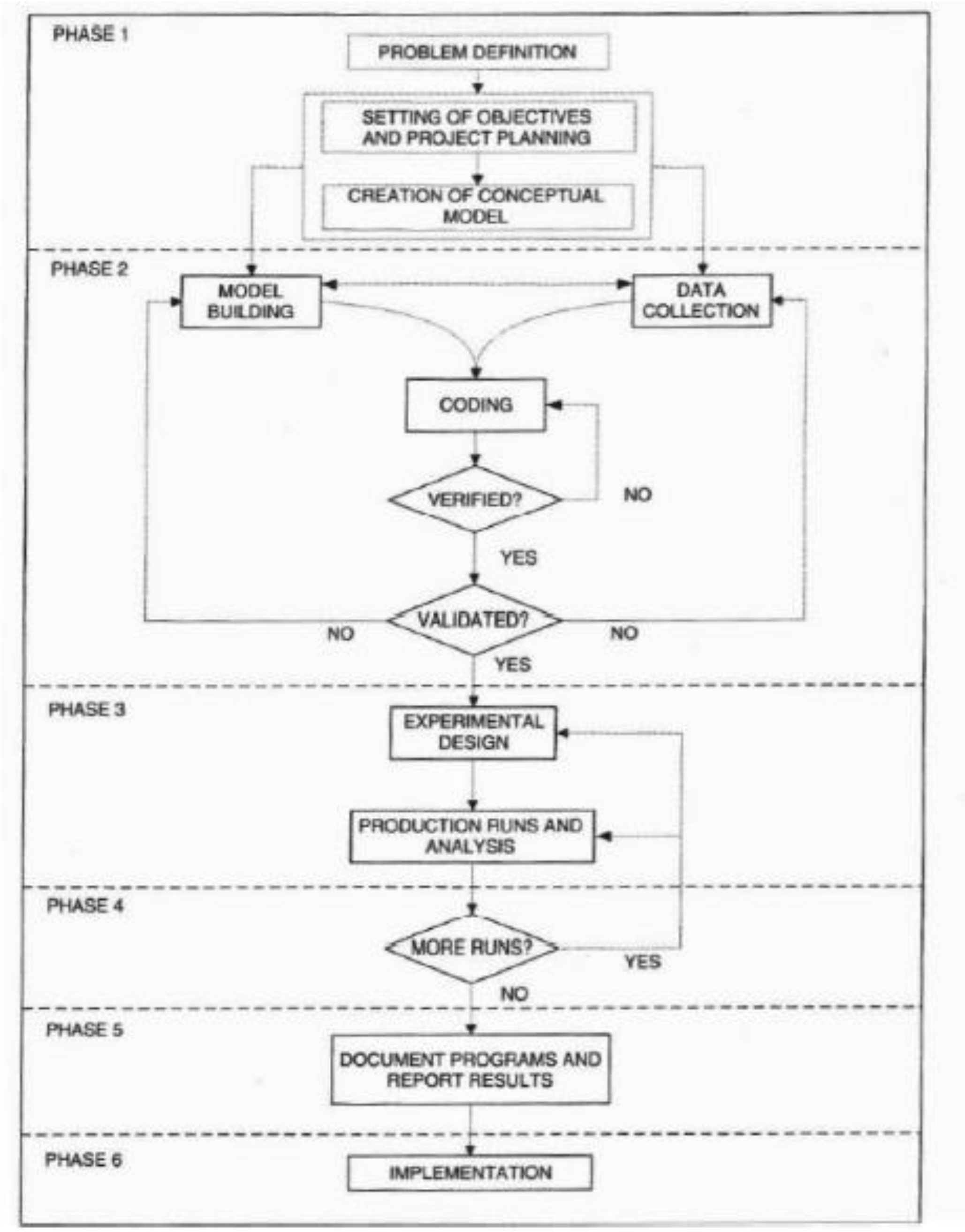



*Figure 21 – General simulation methodology*

The figure above shows the general methodology for applying simulation to problem solving. Specificly, the general methodology is organized as following:

1) *Problem formulation*
    a. Define the problem
    b. Define the system
    c. Establish performance metrics
    d. Build conceptual model
    e. Document model assumption

2) *Simulation model building*
    a. Model translation
    b. Input data modeling
    c. Verification
    d. Validation

3) *Experimental design and analysis*
    a. Preliminary runs
    b. Final experiments
    c. Analysis of results

4) *Evaluate and iterate*
    a. Documentation
    b. Model manual
    c. User manual

5) *Implementation*

## 3.5.1 The problem formulation

The problem formulation phase of the study captures the essence of the first two steps of the DEFREE process, that are “define the problem” and “establish measures of performance for evaluation”. It consists in five main steps to follow:

1) Defining the problem
2) Defining the system
3) Establishing performance metrics
4) Building conceptual models
5) Documenting modelling assumption

The study of the problem starts always with a need that must be try to achieve. The basic output of the problem definition activity is a problem definition statement that is necessary to represent the problem in a synthetic way for the analyst and the problems stakeholders. This should take in all the required assumption made during the modelling process. The assumption effects are analysed during the verification, validation, and experimental analysis steps. When all these steps are over, it means that the problem is well understood and it is ready to continue to be examine in depth.

The general goals of a simulation study often include:

- Comparison: to compare system alternatives and their performance measures across various factors with respect to some objectives;
- Optimization: to find the system configuration that optimizes performance subject to constrains;
- Prediction: to predict the behaviour of the system at some point in time;
- Investigation: to learn about and gain insight into the behaviour of the system, given various inputs.

Therefore, the problem definition is composed a detailed description of the object of the study, the desired output of the problem and the types of scenarios to be examined or decision to be made.

The second point of this phase is the definition of the system. It is a narrative that tells about a representation of the major elements of the system where the boundaries are also defined. This confirm that the study is focused on the correct areas of interest to the stakeholders and that the scope of the project is well understood.

The third part of this phase develops an understanding of how to measure system performance. The analyst has to define the required performance measures for the model. To meaningfully compare alternatives scenarios, objective and measurable metric describing the performance of the system are necessary. The performance metrics should be composed by quantitative statistical measures, quantitative measures from the system and quality assessments. The focus should be placed on the performance measures that are considered to be the most important to system decision makers and tied directly to the objectives of the simulation study. Evaluation of alternatives can than proceed in an objective and unbiased manner to determine which system scenario performs the best according to the decision maker's preferences.

After a good study of the system and its measure performance, the first step expects the model formulation. The conceptual model tools conveys a more detailed system description to allow the model to be translated into a computer representation. Some relevant diagramming constructs include the following:

- *Context diagram:* the pictorial representation of the system than often includes flow patterns typically encountered. Anyway, there are no rules to draw up a context diagram.
- *Activity diagram:* the pictorial representation of the process for an entity and its interaction with resources. The activity diagram can be an activity flow diagram (the entity is a temporary entity), or an activity cycle diagram (the entity is permanent entity). In particular the activity diagram is composed by:
    - Queues: shown as a circle with queue labeled inside;
    - Activities: shown as a rectangle with appropriate label inside;

- Resources: shown as small circles with resource labeled inside;
- Lines/arcs: indicate flow (precedence ordering) for engagement of entities in activities or for obtaining resources. Dotted lines used to indicate the seizing and releasing of resources;
- Zigzag lines: indicate the creation or destruction of entities.

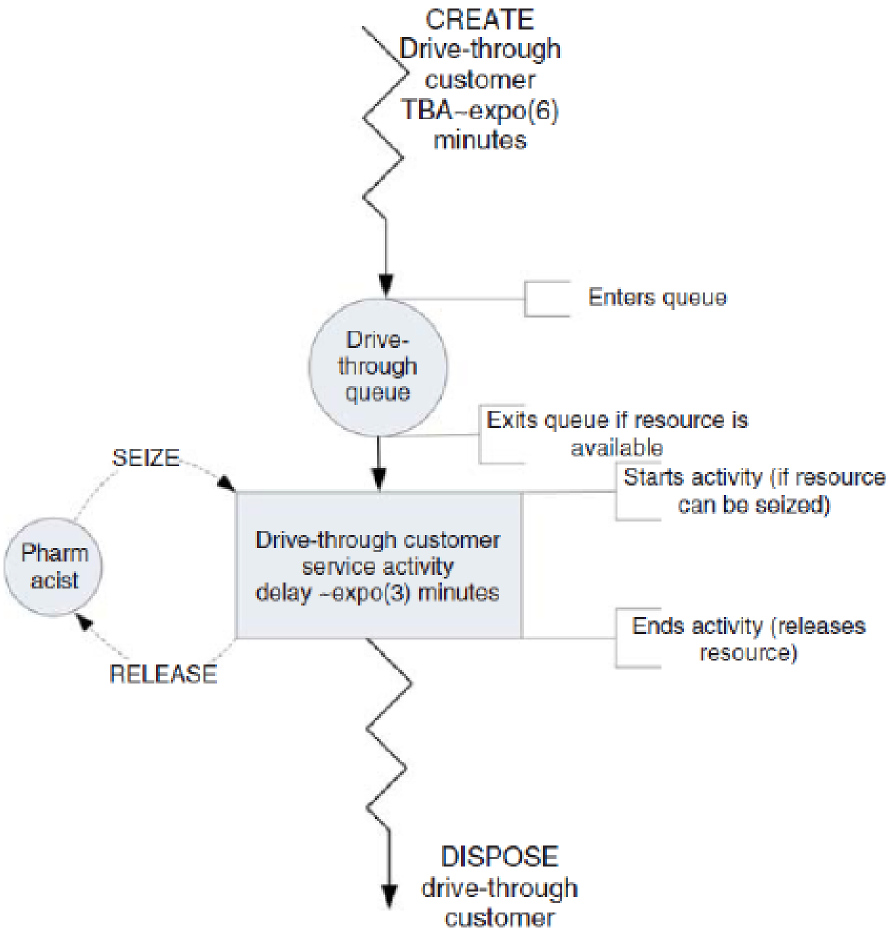



*Figure 22 – Example of activity diagram*

- *Software Engineering Diagrams:* the wide variety of software engineering diagramming technisque to provide information for the model builder. The diagrams are for example flow charts, database diagrams, IDEF (ICAM Definition language) diagrams, UML (unified modelling language) diagrams, and state charts.

At the beginning, the model is developed with an easy conceptual model that captures the basic characteristics and behaviours of the system. Then, some details can be added considering more specifically functionality.

## 3.5.2 Simulation model building

After developing a solid conceptual model of the situation, simulation model building can begin. During the simulation model building phase, alternative system design configuration are developed based on the previously developed conceptual models. Additional project planning is also performed to yield specifications for the equipment, resources, and timing required for the development of the simulation models. The simulation models used to evaluate the alternative solutions are then developed, verified, validated and prepared for analysis.

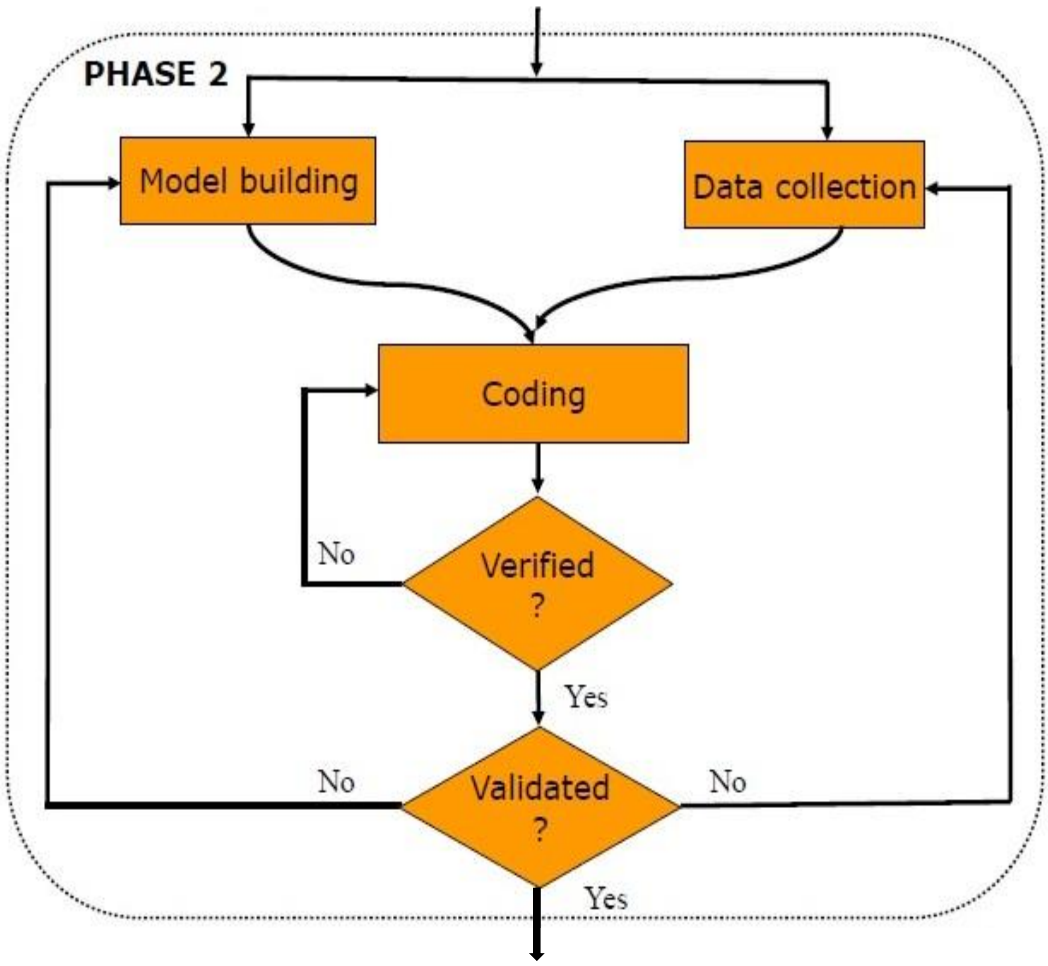



*Figure 23 – Simulation modeling building*

Within the context of a simulation project, this process includes the following:

- *Input data preparation:* input data is analysed to determine the nature of the data and further data collection needs. Necessary data is also classified into several areas. This classification established different aspects of the model that are used in model development.
- *Model translation:* description of the procedure for coding the model, including timing and general procedures and the translation of the conceptual models into computer simulation program representation.
- *Verification:* verification of the computer simulation model performed to determine whether or not the program performs as intended. To perform model verification, model debugging is performed to locate any errors in the simulation code. Errors of particular importance include improper flow control or entity creation, failure to release resources, and logical/arithmetic errors or incorrectly observed statistics. Model debugging also includes scenario repetition utilizing identical random number seeds, "stressing" the model through a sensitivity analysis (varying factors and their levels) to ensure compliance with anticipated behaviour, and testing of individual modules within the simulation code.
- *Validation:* validation of the simulation model is performed to determine whether or not the simulation model adequately represents the real system. The simulation model is shown (of various level) associated with the system in question. Their input concerning the realism of the model is critical in establishing the validity of the simulation. In addition, further observations of the system are performed to ensure model validity with respect to actual system performance. A simple technique is to statistically compare the output of the simulation model to the output from the real system and to analyse whether there is a significant difference between the two.

### 3.5.3 Experimental design and analysis

After you are confident that your model has been verified and validated to suit your purposes, you can begin to use the model to perform experiments that investigate the goals and the objectives of the project.

Preliminary simulation experiments should be performed to set the statistical parameters associated with the main experimental study. The experimental method should use the simulation model to generate benchmark statistics of current system operation. The simulation model is then altered to conform to a potential scenario and is rerun to generate comparative statistics. This process is continued cycling through suggested scenarios and generating comparative statistics to allow evaluation of alternative solutions. In this manner, assessments of alternative scenarios can be made.

For a small set of alternatives, this “one at a time” approach is reasonable; however, often there are a significant number of design factors that can affect the performance of the model. In this situation, the analyst should consider utilizing formal experimental design techniques. This step should include a detailed specification of the experimental design and any advanced output analysis techniques that may be required during the execution of the experiments. During this step of the process, any quantitative models developed during the previous steps are exercised. Within the context of a simulation project, the computer simulation model is exercised at each of the design points within the stipulated experimental design.

Utilizing the criteria specified by system decision makers, and utilizing the simulation model’s statistical results, alternative scenarios should then be analysed and ranked. A methodology should be used to allow the comparison of the scenarios that have multiple performance measures that trade-off against each other.

### 3.5.4 Evaluate and iterate

If the simulation has achieved the objectives, then the recommended solutions should be documented and implemented. If not, it is necessary to iterate and determine if any additional data, models, experimentation, or analysis is needed to achieve modelling experimentation. Good documentation should consist of a least two parts: a technical manual, which can be used by the same analyst or by other analysts, and a user manual. A good technical manual is very useful when the project has to be modified, and it can be very important contribution to software reusability and portability.

### 3.5.5 Implementation

When the simulation satisfies the goals of the study, it is time to document and implement the recommended solutions. Afterwards, the project should be evaluated as to whether or not the proposed solution met the intended objectives.

# 4 AnyLogic ®

Simulation is a very useful tool for identifying the critical points of a system and for finding the most appropriate alternative within a set of configurations proposed by a decision maker. However, it is not suitable for contexts in which the optimum condition is required, so there may be better solutions not yet analysed.

The simulation models allow to consider the temporal distributions of the values of the variables and to hypothesize different solutions without realizing them physically, thus reducing the implementation costs and the risks deriving from a bad choice. Once the model is built, it must be translated into a computer program. It is possible to use general purpose languages (Pascal, C, C ++) or specialized languages (SIMSCRIPT, MODSIM, GPSS). An alternative is to use interactive applications for simulation, including: AutoMod, Simul8, Arena Simulation, Witness, Extend, Micro Saint and AnyLogic. These applications are easy to use and therefore very suitable for quickly building models, even sophisticated ones, but they are less versatile and powerful than previous languages. In particular, AnyLogic is a virtual modelling environment for discrete, continuous and hybrid systems. With this tool, it is possible to create system prototypes during the phases of study, design or development, through which to explore aspects and details of the design or implementation of the relative systems in a simple and risk-free way. AnyLogic allows programming using the Java language, or, alternatively, a faster modelling style can be used, based on the drag and drop of elements belonging to the libraries provided. The animation environment made available by AnyLogic allows the construction of sophisticated interactive animations (implemented in Java), built modularly, using hierarchical structures of the model.

# 4.1 History of AnyLogic ®

At the start of 1990 there was a big interest in the mathematical approach to modelling and simulation of parallel processes. This approach may be applied to the analysis of correctness of parallel and distributed programs. The Distributed Computer Network (DCN) research group at Saint Petersburg Polytechnic University developed such a software system for the analysis of program correctness; the new tool was named COVERS (Concurrent Verification and Simulation). This system allowed graphical modelling notation for system structure and behaviour. The tool was applied for the research granted by Hewlett-Packard.

In 1988 the success of this research inspired the DCN laboratory to organize a company with a mission to develop a new age simulation software. The emphasis in the development was placed on applied methods such as simulation, performance analysis, behaviour of stochastic systems, optimization and visualization.

New software released in 2000 was based on the latest advantages of information technologies: an object-oriented approach, elements of the UML standard, the use of Java, a modern GUI, etc. (Molderink et al., 2009). The tool was named AnyLogic, because it supported all three well-known modelling approaches: system dynamics, discrete event simulation, agent-based modelling, and any combination of these approaches within a single model (Borshchev and Filippov, 2004; Bazan and German, 2012).

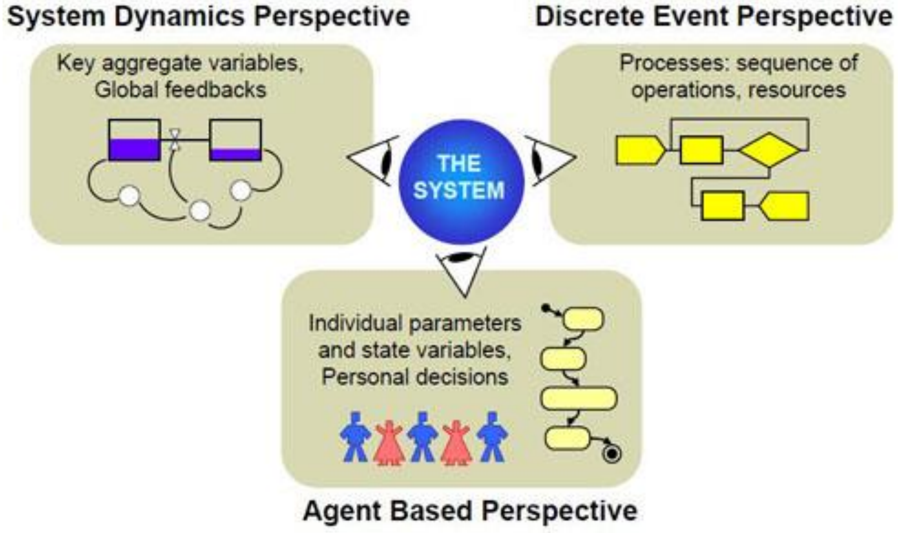



*Figure 24 – Three buisiness simulation approach*

The first version of AnyLogic was AnyLogic 4, because the numbering continues the numbering of COVERS 3.0. A big step was taken in 2003, when AnyLogic 5 was released. New version was focused on business simulation in different industries.

*Figure 25 – Anylogic symbol*

AnyLogic 7, was released in 2014. It featured many significant updates aimed at simplifying model building, including enhanced support for multimethod modelling, decreased need for coding, renewed libraries, and other usability improvements. AnyLogic 7.1, also released in 2014, included the new GIS implementation in the software: in addition to shapefile-based maps, AnyLogic started to support tile maps from free online providers, including OpenStreetMap. 2015 marked the release of AnyLogic 7.2 with the built-in database and the Fluid Library. Since 2015, AnyLogic Personal Learning Edition (PLE) is available for free for the purposes of education and self-education. The PLE license is perpetual, but created models are limited in size. The new Road Traffic Library was introduced in 2016 with AnyLogic 7.3. AnyLogic 8 was released in 2017. Beginning with Version 8.0, the AnyLogic model development environment was integrated with AnyLogic Cloud, a web service for simulation analytics. The platform for AnyLogic 8 model development environment is Eclipse.

## 4.2 AnyLogic ® & Java

AnyLogic includes a graphic modelling language and also allows the user to extend simulation models with Java language. AnyLogic's Java nature lends itself to extensions of the custom model using Java code, as well as the creation of Java applet that can be opened with any standard browser.

These applets make AnyLogic models very easy to share or put on websites. In addition to the Java applets, the Professional version allows the creation of Java Runtime applications, which can be distributed to users. These pure Java applications can be a basis for the decision support tool.

Even if AnyLogic is a simulation tool that is programmed using Java, this does not means that the user is supposed to be skilled programmer to use AnyLogic. All the advanced coding has already been done in the blocks, the user is using to create their model. However in some cases the blocks themselves are not enough and the user needs to specify what is needed to run the model as intended. To do this, an *if-statement* might be needed when an agent is leaving or entering a block. When creating functions a *for-* or *while-loop* might be needed to calculate the result the user wants to present.

### 4.2.1 If statements

A *if-statement* is a basic way of controlling the flow in your model. Often coding it is desired to run a section of code depending on whether or a not a condition is true. When dealing with if statement there is two main types:

- IF-THEN: these statements run the wanted section of code if the condition is true;
- IF-THEN-ELSE: these statements run the wanted section of code if the condition is true, however if the condition is false then it will run the code written in the else section.

The syntax of the if-statement is shown below:

```
if(condition){
//code if condition is true
}else{
//code if condition is false
}
```

*Figure 26 – if-statement*

When we writing statements and loops, certain operators might be needed to state the condition correctly.

Below the most common operators are explained and the java syntax is shown:

| Operators | |
|---|---|
| <,>,>=,<= | Greater, lower, greater or equal and lower or equal. |
| == | Compares if 2 expressions are exactly the same, and will be true if they are (e.g. 2 == 2 will return true and "2" == 2 will return false, because "2" is a string) |
| != | Compares if 2 expressions are not the same ("2" != 2 will return turn) |
| \|\| | The **or** operator compares if 1 of 2 or both statements are true. |
| && | The **and** operator compares if 2 conditions are true or not (e.g 2 == 2 && "2" != 2 will return true) |
| % | The Modulus operator is known as the remainder operator. Meaning it will return the remainder of 2 numbers. (e.g. 10 % 3 = 1. 10-3-3-3=1). This is not a boolean operator like the others, but very useful in some cases. |

*Figure 27 – Operators*

To show how an if-statement can work, an example from a “hold” block in the Job Shop is shown:

```
if (self.in.count() % BlueAssemblerCap == 0 ){
        self.setBlocked(true);
}
```

*Figure 28 – Example of if-statement*

What this code does is that it blocks itself (**self.setBlocked(true)**) if the remainder of the amount of agents (**self.in.count()**) and the capacity of blue agents (**BlueAssemblerCap**) is 0. This is useful if a block cannot contain more than a certain amount of agents while processing. This way the processing block will not get overloaded and crash the simulation. However in this case you also need to unblock it when the agents leave the processing block, otherwise the whole system will be blocked (**holdBlockName.setBlocked(false)**).

## 4.2.2 For-loop

A for-loop will most likely be used to calculate an output that AnyLogic does not produce. To make a for-loop an initialization needs to made, which typically is the index of an array the loop starts in.

This number keeps getting updated every time the loop has run once. The loop will keep running until a specified condition no longer holds.

```
for(initialization; condition; update) {
    // Statements
}
```

*Figure 29 – For- loop*

An example of a for-loop can be written as:

```
int result = 0;
for(int i = 0; i<Data.size(); i++){
        result += Data.getY(i);
}
return result;
```

*Figure 30 – Example of for-loop*

This is a for-loop from AnyLogic that calculates the sum of a data set created in **AnyLogic** (**Data**). The loop will run until "i" is equal to the number of the size of the data set (**Data.size()**). "i" will start at 0 (**int i = 0**). 0 is the starting index of an array in Java, not 1. "i" will increase by 1 each time the loop has run once (**i++**). "result" is the sum of the data set, and it is calculated by using the += operator, that adds the previous value of "result" + the new value given from the data set in the i'th place (**Data.getY(i)**). This is just another way of writing *result* = *result* + *Data.getY* (*i*).

### 4.2.3 While-loop

A while-loop is another way of writing a loop. If you know the exact number of times you want to run the loop a for-loop is prefered, however a while-loop is prefered if the exact number is not known.
A while loop runs until the boolean expression written in the loop is no longer true.

```
while(Boolean_expression) {
    // Statements
}
```

*Figure 31 – While-loop*

However, in this course a for-loop is the most useful. To give an example how a while-loop can be used, the sum function from above is written in a while-loop:

```
int result = 0;
int i = 0;
while(i<Data.size()){
    result += Data.getY(i);
    i++;
}
return result;
```

*Figure 32 – Example of while-loop*

# 4.3 Model of multi-method simulation

The multimethod modelling aim to seamlessly integrate different methods of modelling and simulation to overtake the limits of individual approaches and takes advantages of each one. Combining different methods leads to efficient and manageable models without workarounds.

There are three major methodologies used to build dynamic business simulation models: system dynamics, discrete event modelling, and agent based modelling.

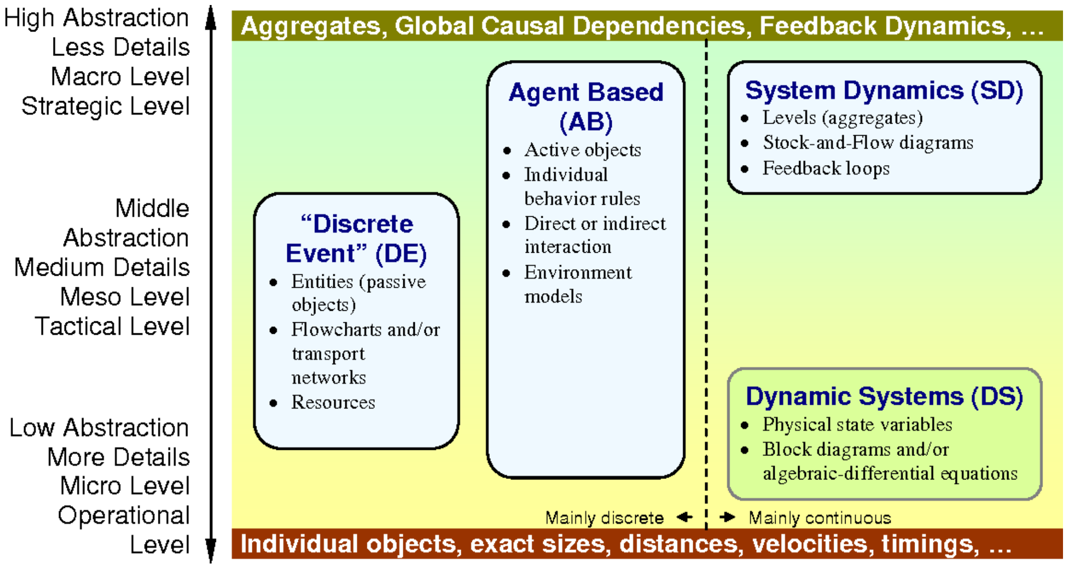



*Figure 33 – Multimethod simulation modelling*

The system dynamics method assumes a high abstraction level and is primarily used for strategic level problems, such as market adoption rates and social process dependency.

Discrete event modelling is mainly used at operational and tactical levels, like manufacturing processes and equipment investment evaluation.

Agent-based models are used at all levels, with the agents possibly being any active entity. Example applications include supply chain optimization and epidemiology.

Building a model requires a level of simplification.

Using a single method, it can be difficult to model at the appropriate level of abstraction. It may be possible to model the actions of autonomous entities via system dynamics, but unnecessary when agent based tools avoid the need for additional abstractions and assumptions. Similarly, discrete methods are inefficient for modelling continuous variables when system dynamics methods are available.

Most real-world cases are complex, and it is convenient to describe different parts of a system with different methods. The ability to get business systems with their real complexity and interactions can be seriously limited using only one method. Some system elements will have to be excluded or a workaround developed.

- If there are many independent objects, use an agent-based approach.
- If there is only information about global dependencies, use system dynamics.
- If a system is easily described as a process, use a discrete-event approach.
- If your system has all those aspects, you should consider combining all three methods.

Having access to all methods simultaneously gives the flexibility needed to successfully solve the problem at hand.

## 4.3.1 Agent-Based Modelling 

Agent based modelling focuses on the individual active components of a system. This is a differentiation to both the more abstract system dynamics approach, and the process-focused discrete event method.

With agent based modelling, active entities, known as agents, must be identified and their behaviour defined. They may be people, households, vehicles, equipment, products, or companies, whatever is relevant to the system. Connections between them are established, environmental variables set, and simulations run. The global dynamics of the system then emerge from the interactions of the many individual behaviours.

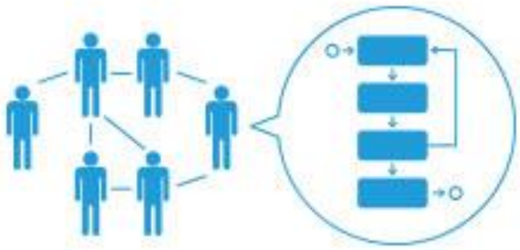


*Figure 34 – Agent based modelling*

The agent based modelling approach is free of some limitations because it is focused directly on individual objects, their behaviour, and their interaction. As such, an agent based simulation model is a set of interacting objects that reflect relationships in the real world. The results make agent based simulation a natural step forward in understanding and managing the complexity of today's business and social systems.

## 4.3.2 System Dynamic

System dynamics is a highly abstract method of modeling. It ignores the fine details of a system, such as the individual properties of people, products, or events, and produces a general representation of a complex system. These abstract simulation models may be used for long-term, strategic modeling and simulation.

AnyLogic supports the design and simulation of feedback structures such as, stock and flow diagrams, array variables (subscripts) in a way most system dynamics modelers are familiar. System dynamics is supported by several tools that are very much alike.

AnyLogic inherently offers all the benefits of the object-oriented approach to system dynamics modeling. Complex models can be defined in a hierarchical manner with objects only exposing interface variables as inputs and outputs.

Moreover, a frequently met system dynamics pattern may be saved as a library object and reused within one simulation model or across different models.

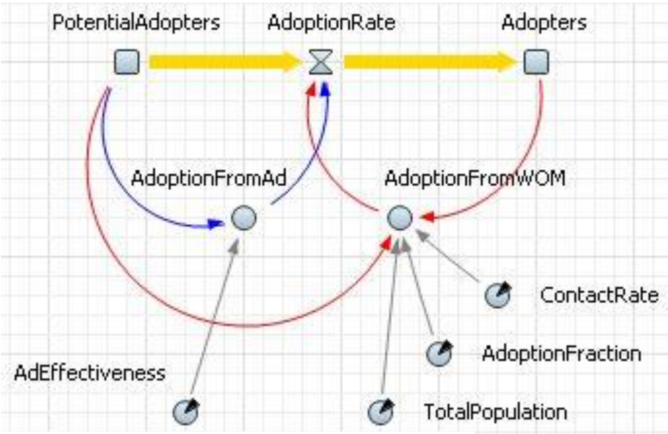



*Figure 34 – Example of system dynamic*

AnyLogic users also benefit from advantages such as model export, cloud model execution, sophisticated animation, and interoperability with other software tools.

### 4.3.3 Discrete Event Modelling

Most business processes can be described as a sequence of separate, discrete, events. Using discrete event simulation modelling, the movement of a train from point A to point B is modelled with two events, namely a departure and an arrival. The actual movement of the train would be modelled as a time delay between the departure and arrival events. These events and movement between them can be smoothly animated.

Discrete event simulation focuses on the processes in a system at a medium level of abstraction. Typically, specific physical details, such as car geometry or train acceleration, are not represented. Discrete event simulation modelling is widely used in the manufacturing, logistics, and healthcare fields.

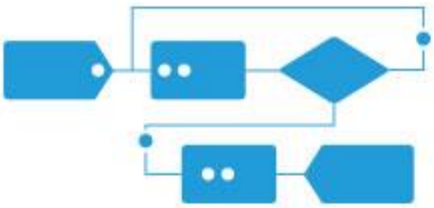


*Figure 36 – Discrete Event Simulation*

## 4.4 How to model in AnyLogic

This section briefly summarizes the basic concepts to be able to perform a good modeling with AnyLogic by referring to agents, blocks, state diagrams, diagrams, parameters, variables, etc. The software presents an iterative guide called *AnyLogic Help* which is very useful for solving doubts and difficulties during the drafting of the model. Despite being a fairly complete guide, it alone is not enough. AnyLogic can be considered a learning software. The modeler presents a wide variety of ways to represent and characterize his system in the simulation environment. Furthermore, knowledge of the Java programming language through which commands are given to perform the functions are required.

The elements used to perform modeling in AnyLogic are found in the *palette* divided into a number of stencils. Through the cursor, select the element in the palette and drag it into the diagram. *Libraries* are the basics of simulation and where the system acts. AnyLogic consists of six main libraries that can be used simultaneously. The *System Dynamics* palette contains elements frequently used by system dynamics modelers while the *Statechart* palette contains elements of statecharts. Statecharts are schemes that work through functions that allow us to represent the behaviors of the event and those based on time. The *Agent* pallet includes those elements that allow you to represent the model, its structure and data as parameters, variables and more. The *Space Markup* palette is composed of elements that allow to represent the marking up of the space to define the positions of the agent. The *Analysis* pallette contains elements that allow you to collect, view and analyze output data. In addition there is the possibility of modeling the graphic representation through the *Presentation* pallette and the *3D Object* (set of 3D images).

Finally, the *Projet* view allows you to view and open anylogic models. The *Properties* view allows instead to directly intervene on the object directly selected. There is also a 3D animation tool.

AnyLogic's camera objects allow to define the view that displays in the 3D window. The chapter continues with a more detailed look at the main items contained in the palettes.

### 4.4.1 Agent and its characteristics

Agents may represent very diverse things: vehicles, units of equipment, projects, products, ideas, organizations, investments, pieces of land, people in different roles, etc. Agents are main building blocks of AnyLogic model. It is a unit of model design that can have behavior, memory (history), timing, contacts, etc. Within an agent you can define variables, events, statecharts, System Dynamics stock and flow diagrams, you can also embed other agents, add process flowcharts. You can define as many agent types in your model as there are different types of agents. Design of an agent typically starts with identifying its attributes, behaviour and interface with the external world. In case of large number of agents with dynamic connections (such as social networks) agents can communicate by calling functions. The agent internal state and behaviour can be implemented in a number of ways. The state of the agent can be represented by a number of variables, by the statechart state, etc. The behaviour can be so to say passive (e.g. there are agents that only react to message arrivals or to function calls and do not have their own timing), or active, when internal dynamics (timeouts or system dynamics processes) of the agent causes it to act. In the latter case, agents most probably would have event and/or statechart objects inside.

**Parameters and variables**

Agent may have parameters or variables. *Parameters* are frequently used for representing some characteristics of the modelled object. They are helpful when object instances have the same behaviour described in class, but differ in some parameter values. All parameters are visible and changeable throughout the model execution. Thus, you can simply adjust your model by changing parameters at runtime. If you need, you can define action to be executed on a parameter change.

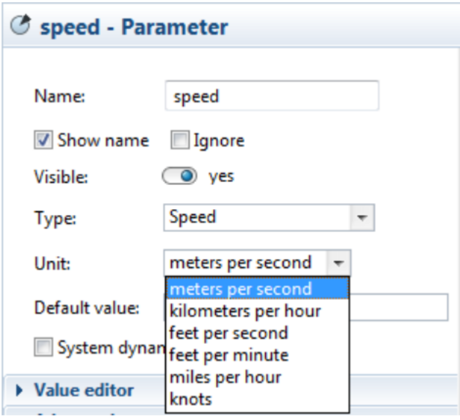



*Figure 37 – Example of parameters*

*Variables* are generally used to store the results of model simulation or to model some data units or object characteristics, changing over time. AnyLogic supports two types of variables:

- Collection: used for defining data objects that group multiple elements into a single unit;
- Variable: a simple variable of an arbitrary scalar type or Java class. It always has some value assigned. You specify the variable's initial value in the Initial value property of the variable. If an initial value is not specified, Java rules apply, for example a variable of type double is set to 0.

There is a clear difference between variables and parameters. A variable represents a model state, and may change during simulation. A parameter is commonly used to describe objects statically. A parameter is normally a constant in a single simulation, and is changed only when you need to adjust your model behaviour.

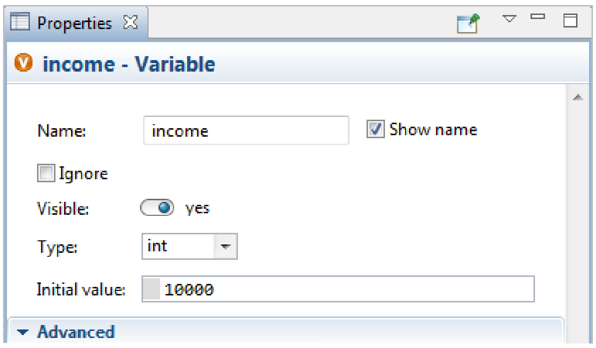



*Figure 38 – Example of variable*

### Statechart

While using events is quite clear, sometimes you may need to define some more sophisticated behaviour that cannot be defined using events and dynamic events. This can be done using statecharts. Statechart is the most advanced construct to describe event- and time-driven behaviour. For some objects, this event- and time-ordering of operations is so pervasive that you can best characterize the behaviour of such objects in terms of a state transition diagram – a statechart. It has states and transitions. Transitions may be triggered by this user-defined conditions:

- Timeouts or rates;
- Messages received by the statechart;
- Boolean conditions;

Transition execution may lead to a state change where a new set of transitions becomes active. States in the statechart may be hierarchical, i.e. contain other states and transitions. Statechart is used to show the state space of a given algorithm, the events that cause a transition from one state to another, and the actions that result from state change. By using statecharts you can visually capture a wide variety of discrete behaviours, much more rich than just idle/busy, open/closed, or up/down status offered by most block-based tools.

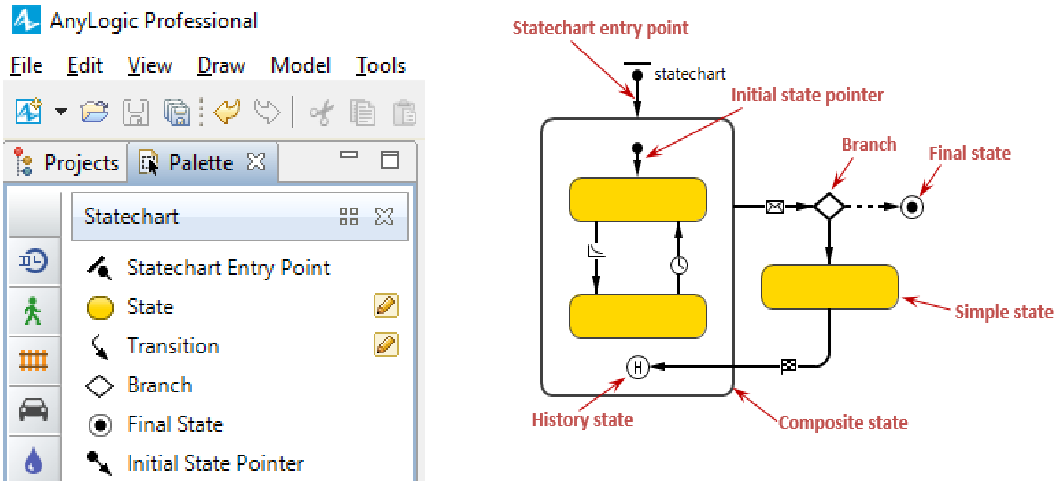



*Figure 39 – Statechart*

## 4.4.2 Space Markup and Resources

AnyLogic uses *Space Markup* for the visual representation of the model. It contains elements for marking up the space in models to define, for instance, agent locations.

*Paths* and *nodes* are space markup elements that define the locations of agents in the space:

- Path: it defines a movement path for agents;
- Node: it defines a place where agents can reside.

In the figure below, a classification of the Space Markup is shown:


*Figure 40 – Space Markup*

**Resource**

The Resource Pool defines a set of agents that can be used to perform a given task in the model. The resource pools agents can be:

- Static resources are bound to a particular location (i.e. node) within the network and cannot move or be moved. An example of a static resource would be an X-Ray room or a weighbridge;
- Moving resources can move on their own, they can represent staff, vehicles, etc.;
- Portable resources can be moved by agents or by moving resources. A portable U-Sound device or a wheelchair would be an example of a portable resource.

Moving and portable resources have their home locations where they can optionally return or be returned. The resource units in one pool can have individual properties, can be animated, collect unit-based statistics, etc. You can define your own resource types representing staff, equipment, etc. The agent uses the pool name to refer to the resource units, and can pick a particular unit by analysing the unit attributes. 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 always collect their individual utilization statistics.

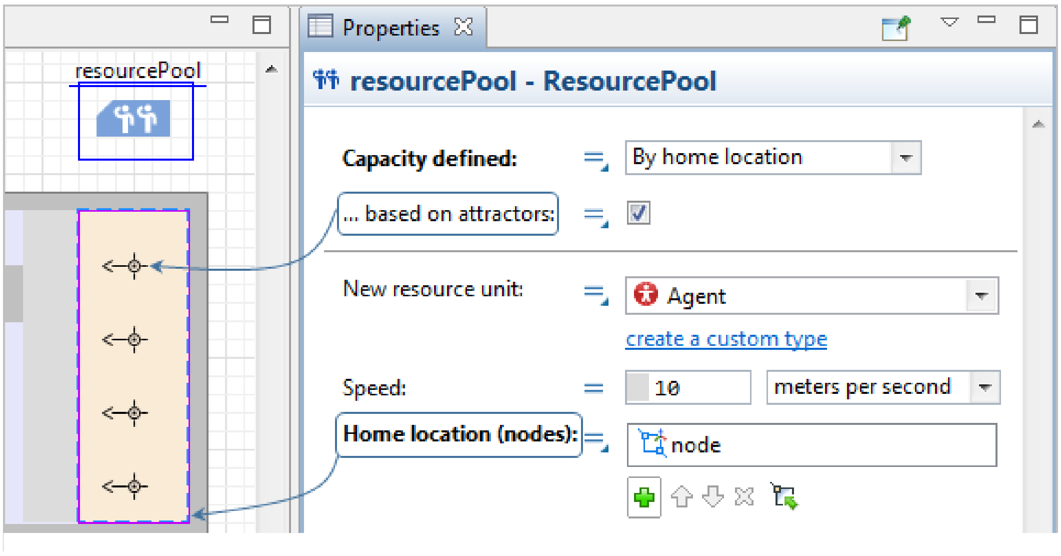



*Figure 41 – Example of ResourcePool*

### 4.4.3 AnyLogic Library

In AnyLogic we can find some pre-set libraries. These *Libraries* are collections of agents developed for some particular application area or modelling task. You can develop a set of reusable agents and Java classes for a particular application area, package them and save as a library. Such custom library can be opened in the palette view along with the standard ones. This way you can share a customized simulation solution within a team of modellers, or deliver it to your clients.

AnyLogic includes the following standard libraries:

- The Enterprise Library is designed to support ED simulation in the areas of Production, Supply Chain, Logistics and Health. Using Enterprise Library objects it is possible to model real systems in terms of entities (operations, customers, products, parts, vehicles, etc.), processes (sequences of operations that generally involve queues, delays, the use of resources), and resources. Processes are represented through flow charts.
- The Pedestrian Library is used to simulating pedestrian flows in a physical environment. It allows to create models of buildings of pedestrians (such as metro stations, security checkpoints, etc.) or roads. The models support a collection of pedestrian density statistics in different areas. This guarantees acceptable performance of service points with a hypothetical load (of people), estimates the duration of staying in specific areas, and detects potential problems due by internal geometry - such as the effect of adding too many obstacles - and others applications. In models created with the Pedestrian Library, pedestrians move in continuous space, reacting to different types of obstacles (walls, various areas), as well as to other pedestrians. Pedestrians are simulated as actors interacting with complex behaviour, but the Pedestrian Library of AnyLogic provides a high-level interface for the rapid creation of pedestrian models through flow diagrams.
- The Material Handling Library supports the modelling, simulation and visualization of factories and warehouses. The library is composed by conveyors, transporters, and other elements.

- The Rail Yard Library supports the modelling, simulation and visualization operations of a railway yard of any complexity and size. The railway yard models can be combined with Discrete Event and Agent-Based modelling related to: loading and unloading, resource allocation, maintenance, market processes, and other transportation activities.
- The Road Traffic Library allows you to model, simulate and visualize vehicle traffic. The library supports detailed yet highly efficient physical level modelling of vehicle movement. It is suitable for modelling highway traffic, street traffic, on-site transportation at manufacturing sites, parking lots, or any other systems with vehicles, roads, and lanes. This library includes visual space markup shapes (road, intersection, bus stop, parking lot, stop line) to draw road networks; driver behaviour: speed control, choosing less busy lane, giving way when lanes merge, avoiding and detecting collisions on crossroads; support of user-defined car types with custom animation and attributes.

Next to these standard libraries, the user can create his own libraries and distribute them.

In chapter 5 we will use some blocks from the Process Modelling Library to model a bottleneck of an automated bottling line, so we will explain in that chapter the blocks used for the case study.

### 4.4.4 Output Analysis

Referring to chapter 7 of "Simulation Modelling and Arena" by Rossetti (2016), we know that both the simulation inputs and the outputs are random. There is a specific panel in the simulation experiments that allows me to keep this randomness under control.

There are three possibilities for modeling the randomness in AnyLogic:

- Random seed: the software initializes a different seed for each experiments performed. Hence, the model runs cannot be reproducible.
- Fixed seed: the user sets a fixed seed for the randomness. This option is very valuable in the development phase as the simulations are reproducible.
- Custom generator: AnyLogic also permits the user to create his own random class.

Performing the output analysis, a simulation experiment aims to observe a set of outputs (over the time or at the end of the experiments) based on the input parameter values. If your model is stochastic, the results of the single model run may not be representative and they change over the random number.

Therefore, more replications with independent random number should be run in order to take valid conclusions.

AnyLogic affords an opportunity to run model with different model parameters and analyze how some certain parameters affect the model behavior. You do not need to run your model several times one by one, and change parameter values manually after each model run, trying to remember the results of these runs and compare them. Using the "Parameter Variation" experiment you can configure the complex model simulation comprising several single model runs, varying one or more root object parameters. Running this experiment with fixed parameter values you can also assess the effect of random factors in stochastic models.

## 4.4.5 Experiments in AnyLogic

An experiment stores a set of configuration parameters of a model. AnyLogic supports several types of experiments meant for different simulation tasks. When a new project is created, one experiment is created automatically. It's a simulation experiment named Simulation. It runs model simulation with animation displayed and model debugging enabled. Simulation experiment is used in most cases. Other AnyLogic experiments are used only when the model parameters play a significant role and you need to analyse how they affect the model behaviour, or when you want to find optimal parameters of your model.

**Types**

AnyLogic supports the following types of experiments:

- Simulation experiment runs model simulation with animation displayed and model debugging enabled. It is used in most cases. Other AnyLogic experiments are used only when the model parameters play a significant role, or when you need to configure a complex simulation comprising several simple model runs.
- Parameters variation experiment performs the complex model simulation comprising several single model runs varying one or more root object parameters. Using this experiment you can compare the behaviour of model with different parameter values and analyse how some certain parameters affect the model behaviour. Running this experiment with fixed parameter values allows to estimate the influence of stochastic processes in your model.
- Optimization experiment finds the optimal combination of parameters that results in the best possible solution. Using the optimization experiment you can observe system behaviour under certain conditions, as well as improve system performance.

- Monte Carlo experiment obtains and displays a collection of simulation outputs for a stochastic model or for a model with stochastically varied parameter(s). Both regular and 2D histograms may be used.
- Compare Runs experiment enables you to interactively input different parameter values and run the model multiple times. It visually compares outputs of simulation runs in both scalar and dataset forms.
- Sensitivity Analysis experiment runs the model multiple times varying one of the parameters and shows how the simulation output depends on it.
- Calibration experiment uses optimizer to find the model parameter values that correspond to the simulation output best fitting with the given data. The data may be both in scalar and dataset form. Coefficients may be used in case of multiple criteria. The calibration progress and fitting of each criterion are displayed.
- Custom experiment runs experiment with custom scenario entirely written by user. Custom experiment gives you maximum flexibility with setting parameters, managing simulation runs, making decisions. It simply gives you a code field where you can do all that (and a lot more) by using a rich Java API of AnyLogic engine (functions like run(), stop(), etc.). This experiment has no built-in graphical interface as well as no predefined behaviour.

# 5 Case Study: Automated Bottling Line

This chapter is focused on the topics prior discussed into a case study. The steps followed to analyse the case study are the ones of the *General Methodology for applying Simulation to Problem Solving* presented in chapter 3. In a first phase, the project consider the simulative representation of the bottleneck of an automatic bottling line through the AnyLogic software. The simulation, then, aims to optimally size the buffer of the same line. Finally, a comparison is made between the results deriving from the simulative approach with the results deriving from an analytical approach applying the formula present in the paper "*Buffer design for availability: a new simulative study in case of infant and random failures"(Battini and al., 2013).*

## 5.1 Problem formulation

### 5.1.1 Define the problem

The case study aims to improve the production rate of an automated bottling line based on the production of glass bottles starting from second-hand ones. The ASME scheme of the line can be seen in the figure below.

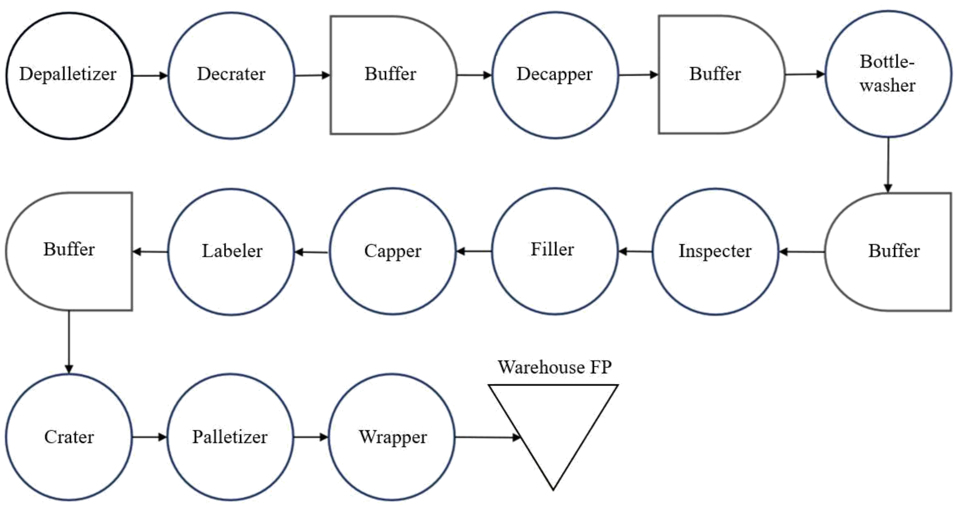



*Figure 42 – ASME scheme of the line*

An important measure performance related to the throughput of the line and its efficiency is the OEE (Overall Equipment Effectiveness). The higher it is, the higher the production rate results. The factors that affect the line, and so the OEE, are failures and set-ups. The central aspect of this work aims to achieve an optimal buffer size in order to raise the OEE of the line. The way of doing it involves the development a simulation model of the critical section of the line. The critical section has already been found out. It is determined by the work-stations (or group of them) with the lowest production rate.

### 5.1.2 Define the system

The system taken into consideration is the critical section of the line. Since the nominal production rate of the work-stations included in the system is the same and they are both affected by failures, the real bottleneck can be either the bottle-washer or the group of work-stations from the bottle-washer to the labeller. The latter works in sync, so during our analysis we can consider only the last work-station: the labeller.

The system is therefore composed by two conveyors - the buffers - and two work-stations; we will refer to them as elements of the line.

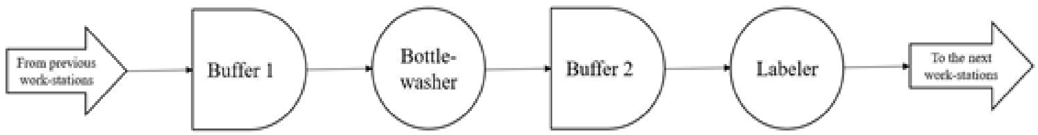



*Figure 43 – ASME scheme of the system*

The bottles arrive at the entry point with a rate of 33000 bottles per hour after that they have been subjected to other works. They appear on a first conveyor that acts as buffer with a capacity of 2500 pieces. The buffer feeds the bottle-washer, a station with a production rate of 25000 pieces per hour, followed by a second buffer that makes the bottles ow until a second work-station. The second buffer of the section has a capacity of 600 pieces.

Both the work-stations can be the real bottleneck since their availability is strictly in influenced by the micro down-times. The output of this piece of line that corresponds with the exit from our system is represented by the bottles with the stuck labels.

The main problem of this bottleneck is the capacity of the buffers, considered as not enough to let the flow of bottles continuous even when a micro-downtime occurs, that it should be its scope. The failures can affect both the conveyors and the work-stations, their causes are many and each element might have different problems. They happen after a certain time (TTF) and then it takes a certain amount of time to the machine (or conveyors) to restart working (TTR). Once a fail occurs the elements of the line affected by it stops to work, so TTF can also be seen as up-time and TTR as down-time. In our study, the second ones are considered micro-downtimes because the time that the machine stalls is lower than 5 minutes.

The table below contains the elements of the line with their failures and their symbols.

| *Machine* | *Failure* | *Symbol* |
|---|---|---|
| Buffer 1 | Lateral stuck | A |
| | Fallen bottle | C |
| Bottle-washer | Stuck load | A |
| | Stuck unload | D |
| | Out of sync machine | E |
| Buffer 2 | Bottles block | B |
| Labeler | Unstuck labels | A |
| | Wrong positioning | B |

*Figure 44 – Line elements with their failures*

In addition, set-ups also stop the entire line: one that happens with a rate between 6 to 12 times a week for a change of the format of the bottles and the other one it occurs once a week for predictive maintenance.

Our inputs are:

- ASME scheme of the line taken into the study. ASME (American Society of Mechanical Engineers) is a symbolism used to represent production systems;

- Rate of bottles that enter our system per hour. They arrive from the previous section of line;

- Nominal production rate of the buffers and of the work-stations. It is expressed in pieces per minute;

- Length and width of the buffers;

- Data sets of Time to Failure and Time to Repair of the buffers and the work-stations for each kind of failure. They are present in the appendix at the end of the thesis;

- Set-up rates and times;

- Values of Overall Equipment Effectiveness of the as is situation of the real line;

- Shift timetable. During a year, the line works 24 hours a day for 8 months and 16 hours a week during the rest months.

### 5.1.3 Establish performance metrics

Different types of performance metrics can evaluate the situation of the system and they are generally resumed under the five headings of quality, speed, dependability, flexibility and cost.

Thus, the performance metrics taken into consideration are:

- Throughput: job exiting from the production line per unit time;

- Overall Equipment Effectiveness (OEE) that reflects the six major losses based on its Availability, Performance and the Quality rate of the output.

As seen in chapter 1, the most practice way to calculate the Overall Equipment Effectiveness value for the entire plant (or production line) implies the using of the following formula presented. This will be used to calculate the OEE of the critical section of the line during our simulation study. The elements of the formula are prior explained in chapter 1.

$$OEE_{line} = \frac{Production\ rate}{Nominal\ production\ rate} = \frac{\frac{Throughput}{Production\ Time}}{25000}$$

*Figure 45 – Formula used to calculate the OEE of the line*

## 5.1.4 Build conceptual model

To depict our system we use an activity diagram where each shape represents something different.

*Zigzag lines* indicate the creation or destruction of entities.

The *queues* are shown as a circle and stand for the buffers of the line, instead of the activities shown as rectangles.

The *resources* that interact with the agents, in our case the working machines that work on them are represented by small circles.

*Lines/arcs* indicating flow (precedence ordering) for engagement of entities in activities or for obtaining resources. Dotted lines are used to indicate the seizing and releasing of resources.

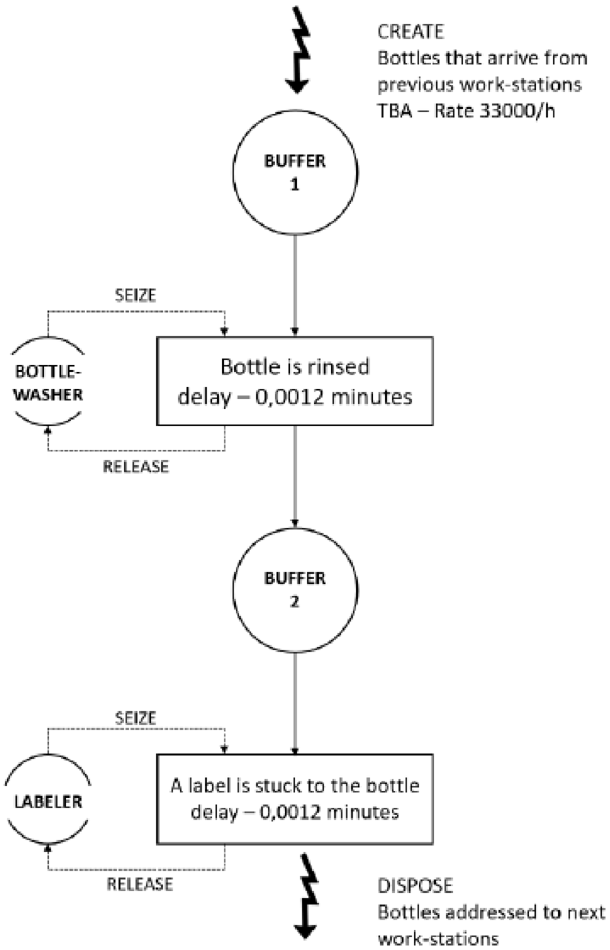



*Figure 46 – Activity diagram of the system*

### 5.1.5 Document model assumptions

The model requires some assumptions due to lacks of some data or to the level of detail desired, that could not require representing every aspect of the real line. The assumption made to build the model are the following:

1. The layout of the line in the model does not follow the real one;

2. The time spent by the agent on the conveyor is the same of its next work-station's cycle time;

3. The speed of the conveyors it has been calculated using the data about the cycle time of the line we had as input and the length of the real conveyors;

4. To represent the grouping of the bottles when they ow on the real line, we use a statistical approach to determine the batch size;

5. The buffer is considered as a unique conveyor with a width of 0,6 meters in order to simplify the simulation. Calculation are done to avoid the distortion of the real working of these conveyors;

6. Conveyors and work-stations' repairs and failures are randomly distributed. Set-up rates and times too;

7. The value for the parameter quality in the calculation of OEE is the same of the one that we have in input up to 99% since the study does not consider scraps;

8. There are not physical constraints to consider when performing the optimal sizing;

9. The time of a model run is set to 10080 minutes, equal to a week of work.

## 5.2 Simulation Model Building

As stated in the General simulation methodology for applying simulation to problem solving, the phases of model building and data collection are strictly linked in order to realize a model as closest as possible to the real system (Rossetti, 2009). In fact, the model building has to depict the best trade-o between all the inputs that affects the study and the investigated level of detail. Since our study aims to improve the productivity on the line, a focus on the efficiency of the model rather than the graphical aspect is preferred. There are many elements and variables that interact between them in the model, therefore when building it, it is important to have a look also at the general set while working on the data preparation of the singular one. For this purpose, assumptions and calculations are made and repeatedly modified, moving up and down through the phases of the iterative process proposed by the methodology.

Our decisions and assumptions are a tempt to make the model as more realistic as possible and they regard the points developed in this chapter. The distributions for the TTF and TTR have been figured out with the use of the statistics. The information about the physics element of the line are important to make the line behaviour as realistic as possible, just like cycle times and buffer capacities.

The approach used for this phase can be seen as a data funnel: the inputs of the case study enter the funnel, where they are worked to exit it as inputs for the simulation model. The funnel reflects the phases of Input Data Preparation and Model Translation, as represented in image below.

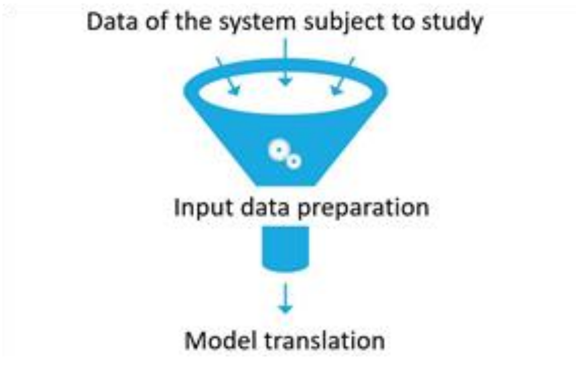



*Figure 47 – Funnel of the work done on the data*

## 5.2.1 Input data preparation

This phase consists in starting from the input data taken from the case study has been analysing and working them to use them as inputs for the in-building model. To reach this aim, various techniques are used. Statistical approaches and methods are needed to develop the proper distributions for Times to Fail and Times to Repair of both the work-stations and the buffers. The Cumulative Distribution Function helps to find out a suitable batch size for representing the flown of the bottles in the model, combined with logical considerations. Information about the physical elements of the line as work-stations and buffers are to be collected and managed with their production speed and lengths in order to be useful for the model.

**Fit the availability parameters with a probability distribution**

To achieve randomness in simulation we need to find the proper probability distribution for the data samples regarding the Times to Failure and Times to Repair of the different causes of down-times of the line. This work is very important because the quality of the data used in a simulation study is vital for the validity of the result. That is the reason for which a thorough analysis of the data with the software Minitab is needed. The result of the analysis is the set of distributions to put into the software AnyLogic to manage the failures.

The first thing to do when analysing some data is to create a histogram and collect descriptive statistics. The histogram can show the frequency distribution of the data and then find out an appropriate distribution with its parameters at first sight. The next step brings to the use of the Anderson Darling Goodness of Fit test to investigate which probability distribution would t the most with the data samples. The kind of distributions taken into consideration for the test are the continuous distributions because they can be used to situations where the set of possible values occurs in an interval or set of intervals. Furthermore, within discrete-event simulation they are often used for modelling time to perform a task *(Rossetti, 2009)*.

The method used to investigate the distributions for Times to Failure and Times to Repair is the same.

It is composed by due steps:

1. Create a histogram for the data sample;
2. Identify the probability distribution that fits the most the data through the Anderson-Darling test.

**The Anderson-Darling test**

The Anderson-Darling test measures how well the data follow a particular distribution. For a specific data set and distribution, the better the distribution fits the data, the smaller this statistic will be. It is defined as:

- $H_0$: the data follow a specific distribution;
- $H_a$: the data do not follow the specific distribution.
- Test Statistic: $A^2 = -N - S$

Where: $S = \sum_{i=1}^{N} \frac{(2I-1)}{N} [lnF(Y_i) + ln(1 - F(Y_N + 1 - i))]$

F is the cumulative distribution function of the specified distribution. Note that $Y_i$ are the ordered data.

The Anderson-Darling statistic ($A^2$) measures the area of the expected model (based on the chosen distribution) and the empirical distribution function. More precisely, it is a squared distance that will have a greater weight in the tails of the distribution. Low values of the Anderson-Darling statistic mean that the hypothesized distribution fits the data well.

Use the corresponding p-value (when available) to test if the data come from the chosen distribution. If the p-value is less than a chosen (usually 0.05 or 0.10), then reject the null hypothesis that the data come from that distribution. Thus, to choose the right distribution, is needed to look in order at:

- AD value: the less it is, the better the distribution is;

- p-value: it should be >0.05 to make the distribution be considerable. The p-value is also used to choose the right distribution when the AD values of two alternative distributions are very close. In these cases it is picked the distribution with a higher p-value.

In our case, the Anderson-Darling test is made for the first reason of failure of the bottle-washer, denoted with the symbol "A". Before it, a histogram for the Times to Failure of the failure A of the work-station bottle-washer is created. This failure is caused by a lateral stuck of the bottles inside the machine.

The histogram displays statistical information with rectangles to show the frequency of data items in successive numerical intervals of equal size. The Times to Failure expressed in minutes that stay in the same interval class are grouped together and it therefore allows to identify the possible distributions for the sample. They are then checked with the Anderson-Darling Test for the data sample.

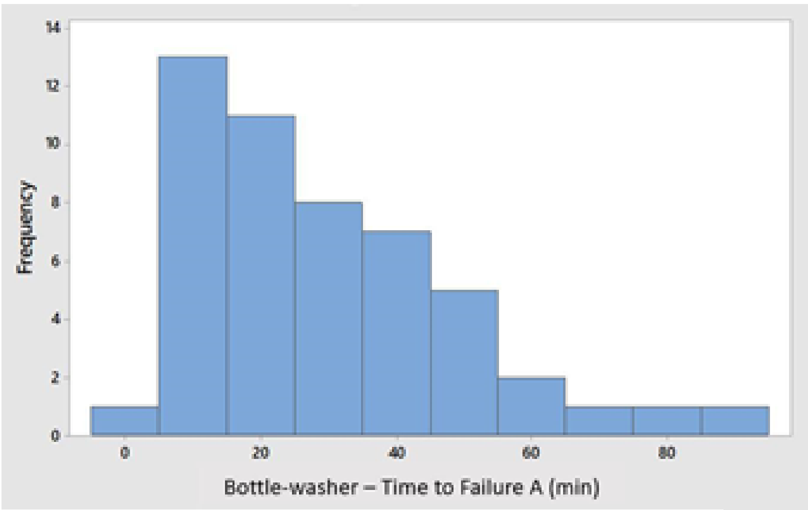



*Figure 48 – Histogram for the TTF A of the bottle-washer*

It is followed by some statistical information about the data sample.

Bottle-washer Time to Failure "Lateral Stuck": Descriptive statistics

| N | Mean | StDev | Median | Minimum | Maximum | Skewness | Kurtosis |
|---|---|---|---|---|---|---|---|
| 50 | 29,6482 | 19,45 | 24,9622 | 4,7915 | 86,2514 | 1,07299 | 0,803603 |

*Figure 49 – Descriptive statistics for Bottle-Washer Time to Failure A*

The Anderson-Darling Goodness of Fit test shows that the Lognormal distribution fits the best the data sample for the Time to Failure of the failure A of the bottle-washer. The reason is that its AD value of 0,246 is the of the test results. Afterwards the parameters of the chosen distribution are pointed out and they will be the input of model for the Time to Failure related to the failure A of the work-station bottle-washer.

Goodness of Fit Test

| Distribution | AD | P |
|---|---|---|
| Normal | 1,284 | <0.005 |
| Box-Cox Transformation | 0,246 | 0,747 |
| Lognormal | 0,246 | 0,747 |
| Exponential | 2,962 | <0.003 |
| Weibull | 0,308 | >0.250 |
| Smallest Extreme Value | 2,801 | <0.010 |
| Largest Extreme Value | 0,471 | 0,24 |
| Gamma | 0,217 | >0.250 |
| Logistic | 0,999 | 0,005 |
| Loglogistic | 0,299 | >0.250 |
| Johnson Transformation | 0,138 | 0,974 |

*Figure 50 – Anderson-Darling test for TTF A*

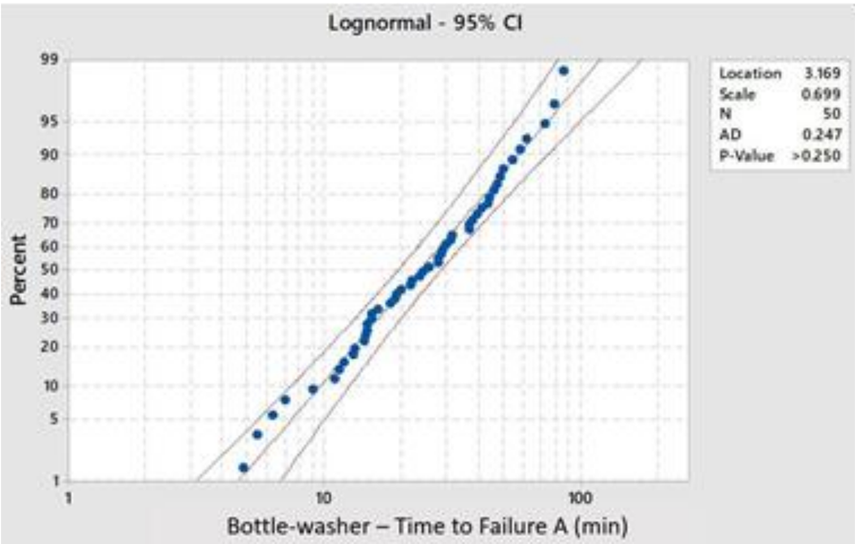



*Figure 51 – Probability Plot for TTF A*

**Estimates of Distribution Parameters**

| Distribution | Location | Shape | Scale |
|---|---|---|---|
| Normal* | 29,64824 | | 19,44999 |
| Box-Cox Transformation* | 3,1694 | | 0,69925 |
| Lognormal* | 3,169 | | 0,699 |
| Exponential | | | 29,64824 |
| Weibull | | 1,63066 | 33,2963 |
| Smallest Extreme Value | 40,12026 | | 22,78402 |
| Largest Extreme Value | 21,06857 | | 13,96618 |
| Gamma | | 2,42638 | 12,21914 |
| Logistic | 27,43989 | | 10,693 |
| Loglogistic | 3,19095 | | 0,40372 |
| Johnson Transformation* | -0,01188 | | 0,96352 |

* Scale: Adjusted ML estimate

*Figure 52 – Distribution parameters for TTF A*

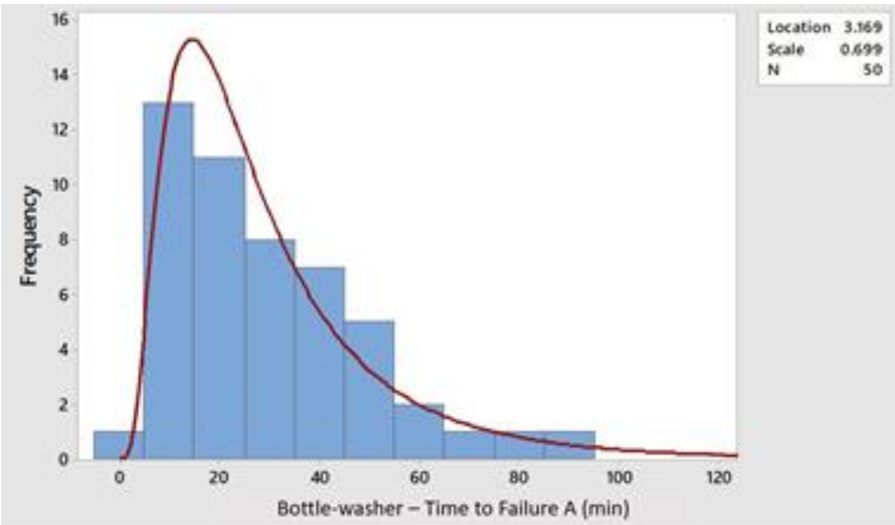



*Figure 53 – Histogram with the fitted distribution for TTF A*

The steps are repeated for all the availability parameters. The results of the all analysis and the chosen distributions are shown in the tables below.

| *Machine* | *Symbol* | *Micro-downtime* | *TTF Distribution* | *Location* | *σ* | *Shape α* | *Scale β* |
|---|---|---|---|---|---|---|---|
| Buffer 1 | A | Lateral stuck | Weibull | | | 1,224 | 46,924 |
| | C | Fallen bottle on FT | Lognormal | 3,765 | 0,375 | | |
| Bottle-washer | A | Stuck load | Lognormal | 3,169 | 0,699 | | |
| | D | Stuck unload | Weibull | | | 1,252 | 38,541 |
| | E | Out of sync machine | Lognormal | 4,044 | 0,420 | | |
| Buffer 2 | B | Bottles block | Weibull | | | 1,127 | 53,908 |
| Labeler | A | Unstuck labels | Weibull | | | 0,996 | 58,335 |
| | B | Wrong positioning | Lognormal | 3,159 | 1,468 | | |

*Figure 54 – Fitted distribution of  the Time to Failure*

| *Machine* | *Symbol* | *Micro-downtime* | *TTR Distribution* | *Location* | *σ* | *Shape α* | *Scale β* |
|---|---|---|---|---|---|---|---|
| Buffer 1 | A | Lateral stuck | Lognormal | 0,557 | 0,459 | | |
| | C | Fallen bottle on FT | Weibull | | | 2,568 | 2,454 |
| Bottle-washer | A | Stuck load | Lognormal | 0,403 | 0,547 | | |
| | D | Stuck unload | Lognormal | 0,662 | 0,325 | | |
| | E | Out of sync machine | Normal | 2,006 | 0,833 | | |
| Buffer 2 | B | Bottles block | Normal | 1,399 | 0,340 | | |
| Labeler | A | Unstuck labels | Lognormal | 0,628 | 0,544 | | |
| | B | Wrong positioning | Weibull | | | 2,852 | 3,416 |

*Figure 55 – Fitted distribution of the Time to Repair*

### Set-up

A 90 minute predictive maintenance is carried out once a week. Therefore, it has been modelled in the following way:

- rate: event that happens once during the simulation;
- duration time: 90 minutes.

To choose the distributions for the set-ups due to a change of the format we analysed the histogram of the frequency and the duration time from a data sample. The rate of occurrence is set at 6 to 12 times a week.

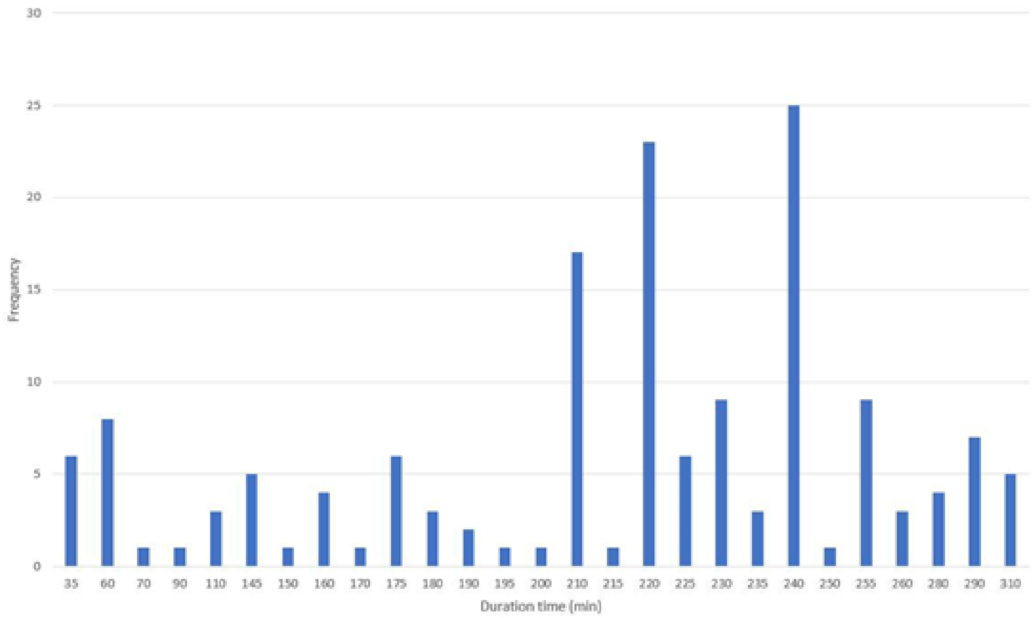



*Figure 56 – Histogram of set-up*

Looking at the histogram, it is easy to notice that a triangular distribution can fit the trend of the duration time of the data sample. The values can be taken from the plot: minimum 35, maximum 310 and mode 240. The distributions to set in the model are the following:

- rate: uniform(6, 12) a week;
- duration time: triangular(35, 310, 240).

**Batch size**

In automated bottling lines the bottles do not pass one by one but flow together on the conveyors and can enter the work-station grouped in the same way. This aspect has been subjected to a particular analysis. Therefore, the size of the group, referred to as batch size, must be decided in order to best fit the similarity with the real line. A trade off between the real line and a statistical explanation of the choice is also needed, without distorting its normal function.

To begin with, it is assumed that 50 bottles, represented as one agent, can be considered a reasonable number for the desired batch size. The analysis is based on some calculations: the input of the line is 33000 bottles per hour, equivalent to 550 bottles per minute. Thus, 50 bottles are generated in 0,09 minute.

The aim is to confirm that the batch size of 50 bottles is reasonable for the buffer capacity and guarantees that the likelihood of finding a failure within a time span of 0,09 minute is less than 1%. The latter is the hypothesis of the analysis.

The tool used to evaluate the hypothesis is the Cumulative Distribution Function. Therefore, the Cumulative Distribution Function (CDF ) of a distribution function of a real-valued random variable X is the function given by:

$$F_x(X) = P(X \leq x)$$

where the right-hand side represents the likelihood that the random variable X takes on a value less than or equal to x. In the case subject to study:

- X is the probability distribution for the Times To Failure;
- x is equal to 0,09 minute, rate of arrival of 50 bottles.

In the diagram below, we can see that for the bottle-washer there is a likelihood of 1% to find a failure stuck load within 2,72 minutes. This amount of time is higher than 0,09 minutes, therefore the likelihood of finding a failure in 0,09 is less than 1% and our hypothesis is valid for this Time To Failure. If this occurs for all the Times To Failure then our hypothesis is confirmed for all and we can use a batch size of 50 bottles.

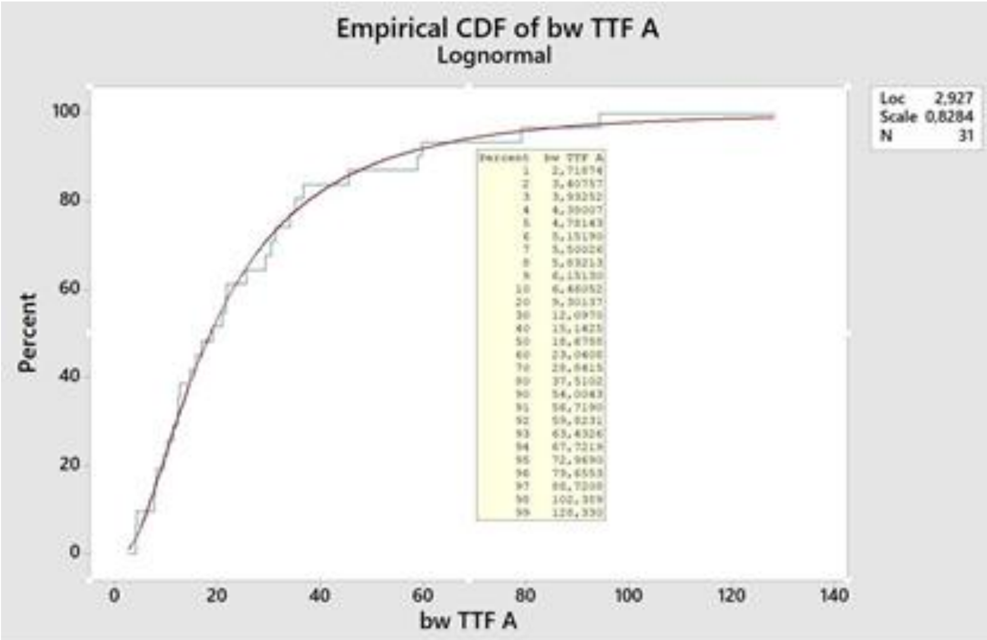



*Figure 57 – Plot of the Cumulative Distribution Function of TTF A*

The results of the analysis have been collected in the following table.

| | *Failure* | *Distribution for X* | $P\{X \leq 0,18\}$ | *Hypothesis confirmed?* |
|---|---|---|---|---|
| Buffer 1 | Lateral stuck | Lognormal(0,49; 2,87) | 0 % | Yes |
| | Fallen bottle | Weibull(1,93; 19,29) | 0,0121 % | Yes |
| Bottle-washer | Stuck load | Triangular(34,95; 28,97, 72;38) | 0 % | Yes |
| | Stuck unload | Gamma(1,68; 25,96) | 0,00155 % | Yes |
| | Out of sync machine | Lognormal(0,36; 3,60) | 0 % | Yes |
| Buffer 2 | Bottles block | Gamma(1,88; 26,42) | 0,0047 % | Yes |
| Labeler | Unstuck labels | Exponential(2,1; 44,39) | 0,4047 % | Yes |
| | Wrong positioning | Lognormal(1,34; 3,12) | 0,0154 % | Yes |

*Figure 58 – Results of the CDF test*

We can confirm that the batch size of 50 bottles, which represents a reasonable value with buffer capacity of 600 and 2500 pieces, is also valid from a statistical point of view. Therefore, the cycle time of the elements of the line changes as following:

$$Cycle\ time = \frac{1}{25000/60} \times Batch\ Size = 0{,}12\ [minutes/batch]$$

From here on, one agent will represent an amount of bottles equal to the batch size. In the model one agent represents 50 bottles.

The rate of bottles that enter the system is 33000 bottles/hour so the time of arrival of one agent is set as:

$$Interarrival\ time = \frac{1}{33000/60} \times Batch\ Size = 0{,}091\ [minutes/batch]$$

**Physical elements of the line**

The calculations on the parameters of the conveyors to be used in the model are driven by the fact that the inputs regarding the speed of the conveyor were missing. Thus, as we had the data on the production rate, which is the same for the entire section of the line, the speed of the two conveyors was obtained by a ratio between length and cycle time. The cycle time of the conveyor refers to the time spent by one agent to move along the entire length of the conveyor in a situation of normal functioning. The formula used is:

$$Speed_{conveyor} = \frac{Conveyor\ Length}{Cycle\ time}$$

The table shows the data on the conveyors and their obtained speed.

| Conveyor | Length [meters] | Width [meters] | Cycle time [minutes] | Speed [meters/second] |
|---|---|---|---|---|
| Buffer 1 | 33,6 | 0,6 | 0,12 | 4,7 |
| Buffer 2 | 8,1 | 0,6 | 0,12 | 1,12 |

*Figure 59 – Physical parameters of the buffer*

The second aspect to be analysed is the length of one batch. This is because, in the AnyLogic model, as in real cases, the capacity of the buffer is directly proportional to its length: the longer the buffer, the greater the number of bottles that the buffer can carry (its) capacity. For this reason, every batch that is added increases the buffer capacity of 50 bottles. To calculate the length of the batch we have used the assumption made on the diameter of one bottle: 0,09 meters.

The procedure to calculate the length of the batch is the following:

1. Consider the width of the conveyor of 0,6 meters and the diameter of one bottle in order to calculate how many bottles fit in a row;

2. Calculate the number of rows that form a batch;

3. The length of one batch is obtained from the multiplication of the diameter of one bottle and the number of rows that form a batch.

| ***Diameter [m]*** | ***[Bottle in a row]*** | ***[Row]*** | ***Batch length [m]*** |
|---|---|---|---|
| 0.09 | 6 | 9 | 0.81 |

*Figure 60 – Procedure to calculate the length of the batches*

Therefore, an increase of 0,81 meters to the length of the conveyors corresponds to an increase of buffer capacity of 50 bottles.

The capacity of the buffers derives from the real capacity and the batch size:

$$Capacity_{Buffer1} = \frac{Real\ Capacity_{Buffer1}[bottles]}{Batch\ Size\ [bottles]} = \frac{2500}{50} = 50\ [batches]$$

$$Capacity_{Buffer2} = \frac{Real\ Capacity_{Buffer2}[bottles]}{Batch\ Size\ [bottles]} = \frac{600}{50} = 50\ [batches]$$

After the input data preparation phase, the data to be inserted into the model as input are:

- Rate of arrival: 0,0024 minutes/bottle;

- Batch size: 50 bottles;

- Length of one batch: 0,81 meters;

- Work-stations cycle time: 0,12 minutes/batch;

- Probability distributions for machines and conveyor failures and repairs and for the set-ups;

- Parameters of the conveyors: length and speed;

- Capacity of the buffers;

- Simulation time: 10080 minutes.

## 5.2.2 Model translation

This phase entails the description of the procedure carried out to code the model, in other words, to represent the real system with the software Anylogic. The aspects explained in this section regard the translation of the conceptual model into computer simulation program representations. The model aims to represent the critical section of the bottling line of our case study. It is composed of three main parts that will be introduced and explained in the following sections: graphics, flowchart and statecharts.

**Graphics**

The software Anylogic allows the model to have different graphical levels of detail, linked with the aimed level of detail of the study. The graphical issue in this case study is not that important since the main purpose is to increase the performance of the system. It can be represented by a basic design. For this reason, few elements of the Space Markup palette have been used to show the critical section of our bottling line in an simple manner. A bottle from 3D Object was taken to represent the batch of 50 bottles. The buffers are therefore represented by paths and the work-stations by rectangular nodes.

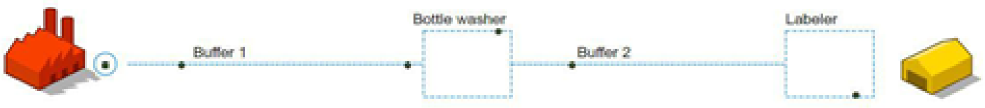



*Figure 61 – Critical section of the bottling line in the model*

**Flowchart of the system**

The simulation model works following the logic created by the user, called flowchart. This is composed of the blocks of the Process Modelling Library seen in chapter 4.

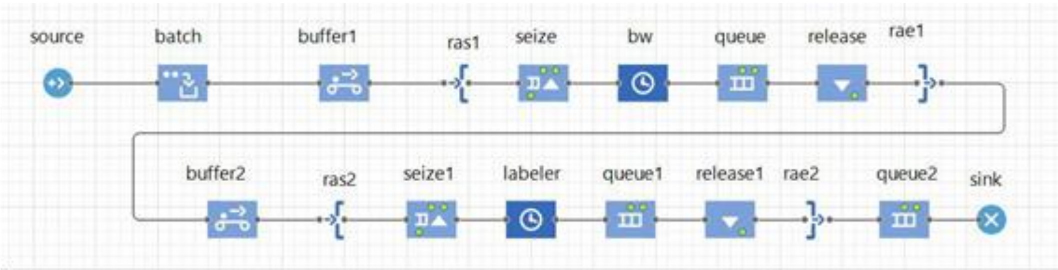



*Figure 62 – Flowchart of the model*

The agents are generated in the block source with an arrival rate of 33000 bottles/hour and a rate of 0,00182 minutes/bottle in the model. Immediately after, the batch aggregates 50 agents into one to create the batch of a previously chosen size. The system starts here. The agent is moved along the first buffer by the block buffer1, a Conveyor block, to get to the first work-station. The work-stations are inserted in the model as resources initialized by the Resource Pool block.


*Figure 63 – Work-stations*

When the agent arrives at the work-station, it enters the restricted area and is then seized by the resource. It reaches the delay block and waits there for a delay time that corresponds to the cycle time of the machine, set at 0,12 minutes/batch. When the operation is completed, the agent is released and exits the restricted area to proceed along the line. It flows along the second buffer until it reaches the work-station labeller. The functioning of the operation is the same as the previous station. Once the operation of labelling is completed, the agent exits the system through the block exit. On the real line, it will go to other work-stations but the system of the case study finishes here. In conditions of normal functioning, the agents flow along the line with a nominal speed of 0,12 minutes/batch, whether they are on buffers or work-stations. This is not always the case as the elements can be subject to failures that stop their functioning for a certain amount of time, until they are repaired. These failures are managed with statecharts. The management of failure is discussed in the next paragraph.

**Managing the failure with statechart**

A thorough analysis of the data sample of the Times to Failure and Times to Repair has been carried out since this is a critical aspect of the system. Failures influence the system and decrease the performance of the line. The more failures that occur, the more time the elements of the line, subject to failure, are stopped and the production output rate decreases. The bottleneck is characterized by many causes of failure and some elements are subject to more than one cause of failure too, as seen in 5.3. For example, the bottle-washer can be stopped by a stuck either while loading the bottles into the machine or unloading after the operation of rinsing, but also due to synchronization problem. Instead, the labeller can stop due to a failure caused by wrong positioning or by an unstuck label.

For these reasons, we have created a statechart for each kind of failure, for each work-station and buffer. The set-ups are also managed by statecharts. Moreover, we have added functions, parameters and variables; they will be explained further on.

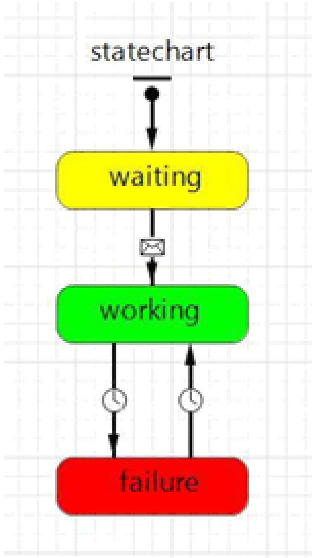



*Figure 64 – Statechart of a failure with its states*

In our case, the statechart is basically composed of three different states. The first, waiting, is active before the system starts. When the first agent is created, a trigger activates the working state and the machine (or the buffer) starts to run. After the Time to Failure, the state failure is triggered and a code *suspend()* (or *stop()* for the buffer) stops the machine. A new code *resume()* activates when the Time to Repair is over and the machine starts working again.

The situation discussed is a simple case when an element of the line is subject to just one failure. This is the case of the second buffer, where only one cause of failure is observed, due to a bottle block during the flow on the conveyor. For the rest of the elements of the system, at least two causes of failure have been detected: two for buffer1 and for the labeller, three for the bottle-washer. These sets of failures are managed with more statecharts in a single agent diagram and some codes and functions both in the states and in the transitions. The management of more than one failure, set in our simulation environment, follows this logic: every time a failure occurs, the system verifies whether there are other failures active or not. In the latter case, the last failure to occur is the dominant one and will manage the resume(); otherwise, a comparison is made between the Time to Repair of the occurred failure and the remaining repair time of the previous failure (failures). The longest time establishes the dominant failure, which will be the one responsible for the resume() of the machine (buffer).

The logic is shown in the flowchart below, where there are two failures (A, B) with their Time to Repair and Time to Fail (TTF A, TTR A, TTR B), that act on the machine.

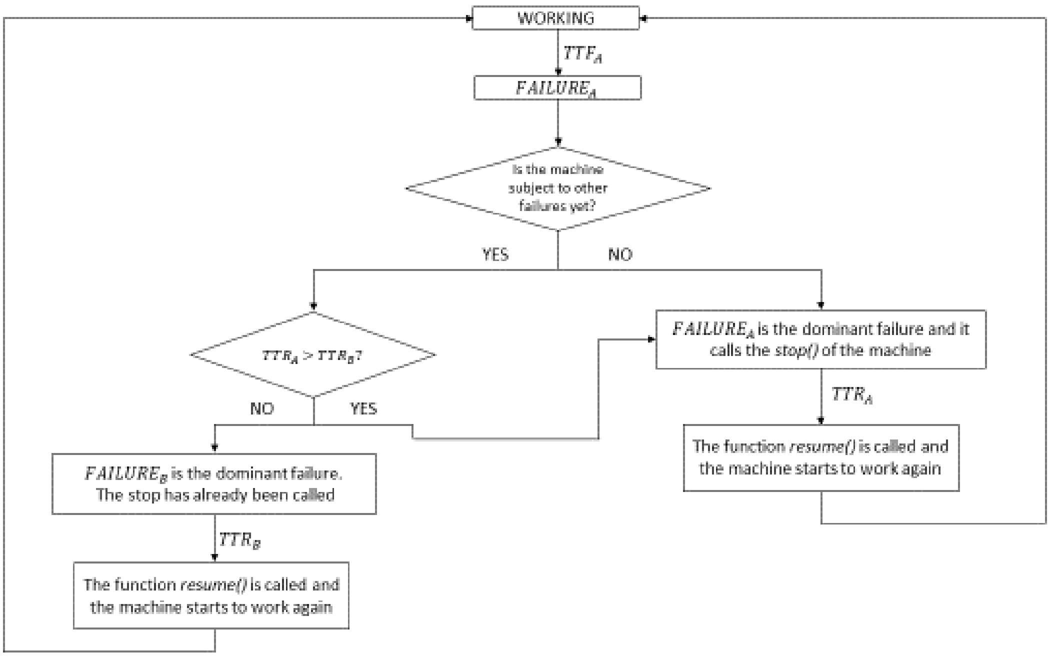



*Figure 65 – Flowchart of the operation of the statecharts*

As stated before, the computational work implies functions and parameters that are key elements in the creation of a model with the software Anylogic. They are written in a Java language and are needed to operate all the logic explained before regarding failures. Moreover, codes are also written inside the transitions and the states. The functions that we have used are:

- *updateTTR.* When actioned, it swaps the Time to Repair present in the collection ttr with the remaining time to the end of the repair through function restTime. The collection ttr contains the values of the TTR of the failures: 0 if the failure is not active, otherwise a certain amount of time.

The code for this function is:

```
for (int i=0; i<ttr.size(); i++)
{
if (ttr.get(i)!=0)
    {
    ttr.remove(i);
    ttr.add(i, restTime(i));
    }
}
```

*Figure 66 – Funtion updateTTR*

- *restTime.* When activated, it initializes a double variable rest to which it assigns a value if a failure is already ongoing. This value is equal to the remaining time of the Time to Repair related to the ongoing failure. The code for this function is:

```
double rest = 0;

if (i==0)
    rest = ttrA.getRest();
else if (i==1)
    rest = ttrB.getRest();

return rest;
```

*Figure 67 – Function RestTime*

Though in our model the coding method has been repeated for all the work-stations and the buffers, the following procedure represents the path used to code the statecharts that manage the failures of the flowchart (*figure 65*). Thus, it is a generic approach that can also be used in other systems.

Each statechart is made by:

- *States:* waiting, working and failure;

- *Transitions ttf and ttr*. They are triggered by a timeout that follows the probability distribution related to that availability parameter;

- *Variables: restartA* and *restartB* are used to check which is the dominant failure when it is time to restart the machine; maxTTR is a boolean variable used to store the value of the current maximum Time to Repair.

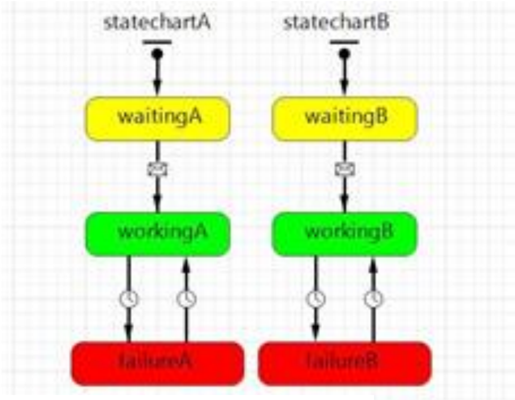



*Figure 68 – Statecharts of Failure A and Failure B of the machine*

The codes are written as entry actions inside the status *failure* and inside the transition of the Time to Repair, that links the *failure* state to the *working* one. The meaning and the function of statechart of failure A are shown taking into consideration that the steps for coding failure B are the same.

When a failure occurs, the function *updateTTR* is activated and the maximum *ttr* is calculated between the active ones. Position 0 of the *ttr* collection is also initialized with the value of *ttrA*. An *if-else* statement states which action is carried out. If there is no other ongoing TTR transition, the function *suspend()* is activated and the machine stops running, the double variable *restartA* is assigned to be true and the *dominantTTR* is declared to be the one that occupies position 0 inside the trr collection (equal to failure A; while the Time to Repair of the failure B occupies the position 1).

Instead, if the transition *ttrB* is active (failure B has already stopped the machine), the Time to Repair A is checked in order to identify which is the longest in respect to the other ongoing failure. The Time to Repair of the ongoing failure B, during the comparison, is considered as the time remaining to the end of *ttrB*. Thus, a *true* value is assigned to the restart variable related to the failure with the maximum Time to Repair and *false* to other.

```
updateTTR();
ttr.remove(0);
ttr.add(0, ttrUnstuck);

maxTTR = Collections.max(ttr);

if(ttrB.isActive()==false)
    {
    machine.suspend();
    restartA = true;
    dominantTTR=0;
    }
else
    {
    if (ttrA == maxTTR)
        {
        restartA = true;
        restartB = false;
        dominantTTR=0;
        }
     else
        {
        restartB = true;
        }
     }
```

*Figure 69 – Code of the state failure A*

The transition $ttr_A$ contains the code that restarts the machine if the variable *restA* is true. More-over, it initializes the *boolean* variable $rest_A$ to *false* which is its default value and to 0.0 the position related to the $ttr_A$ in the collection *ttr*.

```
ttr.remove(0);
ttr.add(0, 0.0);

if (restartA==true)
{
 machine.resume();
}
 restartA=false;
```

*Figure 70 – Code of transition ttrA*

Since the object of study is the performance of the system, we have also added to the model variables and elements of the Analysis palette to collect statistics during the runs. They are useful to:

- Calculate the Overall Equipment Effectiveness of the machines;
- Collect availability and efficiently parameters of machines: availability, performance, MTTF, MTTR;
- Create a plot of the utilization of the machines;
- Achieve other performance measures: throughput of the machines and of the line.

### 5.2.3 Verification

The verification-phase involves running the model many times to be assured that the codes work and that the model does what it is supposed to do. It has also helped to change and improve the model during the whole construction.

Before the validation, 1000 replications of the model were carried out. In fact, since the model is stochastic, the result of a single model run might not be representative of the system. This is due to the randomness of the simulation. For this reason a proper number of replications is required with independent random numbers in order to make valid conclusions. The practical approach used to determine the right number of replications for a simulation requires creating a steady-state plot. A steady-state plot is a plot of the average of the number of replications. It graphically shows the number of replications from which the average result is stable (it does not change much replication after replication).

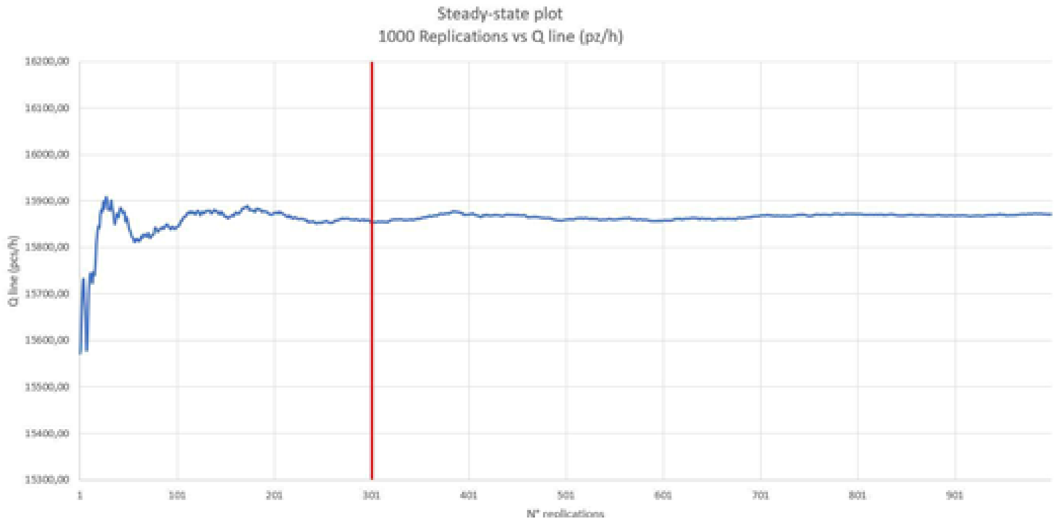



*Figure 71 – Steady-state plot to investigate the number of replication*

In the plot, the steady-state can be seen. The plot shows that the values of the production rate become approximately constant with about 300 replications.

### 5.2.4 Validation

In order to achieve the validation, parameters and performance values are checked to determine whether the simulation model adequately represents the real system. This technique is applied to statistically compare the output of the simulation model to the output from the real system. The output taken into consideration are the OEE of the line and production rate (expressed in bottles per hour). The OEE value should be between 62% and 65% and the production rate between 150000 and 16000 bottles/hour. The results obtained from the model runs after 300 replications are shown in the following table and compared with the real ones. The AS IS situation shows the values of the real line.

| | ***AS IS*** | ***SIMULATION RESULTS*** |
|---|---|---|
| OEE of the line | 62-65 % | 63,49 % |
| Production rate (bottles/hour) | 15000 - 16000 | 15871,5 |

*Figure 72 – Comparison between AS IS values and Simulation results*

These results can be considered as satisfactory and therefore the model can be validated and used to run experiments. In fact, the results show that the proposed model has an acceptable level of confidence in the performances processing assumed.

# 6 Experimental analysis and improvements

The model are run many times to see the *as is* situation that it is described mainly with the value of the Overall Equipment Effectiveness and the production rate. The test is a *Parameter Variation* experiment where buffer length is a parameter to be varied in a range, highlighting the different scenario. The purpose is precisely to efficiently decouple the bottle washer with the labeler, finding the optimal sizing of the buffer. The buffer has indeed the aim of allowing process continuity and should be placed between two critical areas from the point of view of the micro-downtimes, making it possible for each machine to continue operating also after the interrupting of the adjacent machines (*Gershwin,1992*). After that, different improvements and possible solution are shown to improve the OEE and maximize the throughput of the line. At the end a comparison between the simulation approach and the analytic approach of Battini et al. paper (proposed in chapter 2) is made to underline the possible differences. The chapter also presents a brief economic analysis to verify the feasibility and the investment costs of the possible modifications of the line.

The following figure summarizes the procedure used:

- at first the AnyLogic software was used to represent the current situation of the line and then evaluate possible improvements;
- later an analytical approach was applied, the results of which were compared with those obtained from the simulation approach;
- finally it ended with a brief economic analysis of the most significant scenario through the Net Present Value (NPV).

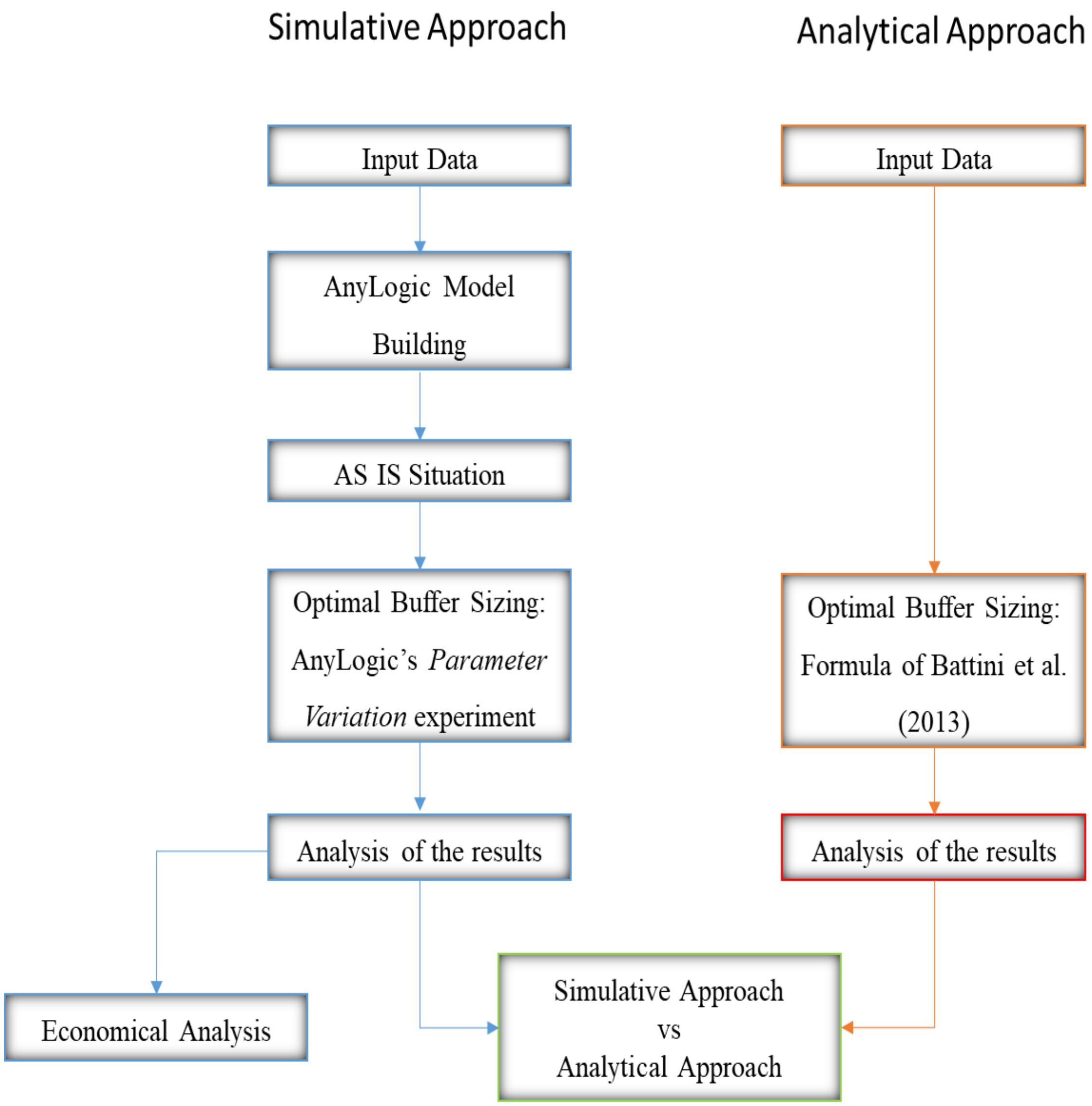



*Figure 73 – Experimental Methodology*

## 6.1 Optimal Buffer Sizing

After reaching the desired *as is* values, the experimental phase starts. The goal of the study is to achieve new possible scenarios regarding the capacities of the buffers to analyse the best solution that optimizes its size. An optimal buffer size allows improving the OEE and maximizing the throughput of the line.

The second buffer is not able to effectively decouple the two station, bottle washer and labeler. For this reason, we proceed with the analysis of this buffer. In fact, it must be able to reduce the negative effects caused by failures and micro-downtime. The test is performed through a *Parameter Variation Experiment* of AnyLogic. It affords an opportunity to run model with different model parameters and analyse how some certain parameters affect the model behaviour. The figure below shows the setting of the Parameter Variation in our case.

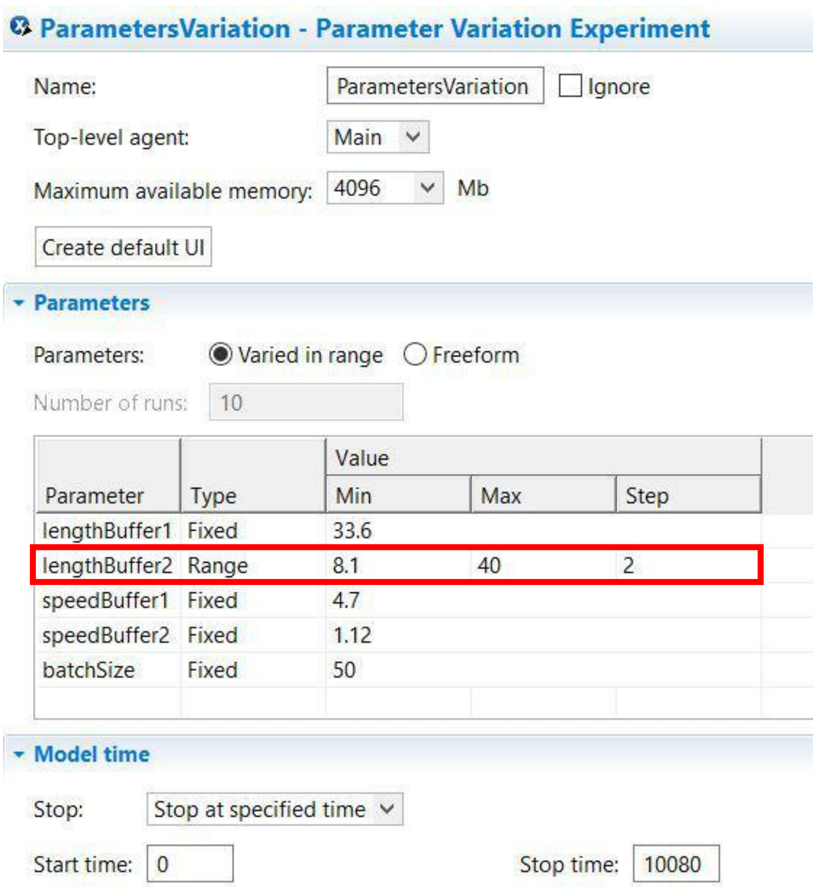



*Figure 74 – Parameter Variation in AnyLogic*

As stated in 5.1.5, the simulation time is 10080 minutes and the number of replications per iteration is set to 300. This number of replications derives from the analysis of the steady-state plot in figure 71. The parameter set to vary in our test is the length of the buffers, depending on the scenario.

SCENARIOS

**Scenario 1**

The length of the buffer1 is fixed at 33,5 meters while the length of buffer2 varies between 8 and 56 meters. The step this parameter will increase its value to reach the maximum is set to 8.

**Scenario 2**

After founding the optimal length for buffer2, we set the length of buffer2 fixed at the optimal found with the previous test and the length of buffer1 to vary between 25.5 to 57.9 meters, with a step of 8.1. This test is done to see whether an adjustment of the length of the buffer1 might be interesting too.

## 6.2 Analysis of the results

As we have seen in the previous paragraph, some scenarios have been simulated that provide for the optimization of the sizing of the line buffer. Therefore, it can be assessed whether changing the buffer can guarantee a significant increase in productivity. In particular, this analysis was focused on the Bottle Washer-Labeler section, because the buffer consisting of the "buffer2" storage systems has a very small capacity and involves continuous interruptions between the two machines.

After collecting the data in a table, through a histogram we can see the trend of the OEE and of the throughput according to the length of the buffer.

| Lenght | Capacity | OEE | Q (pcs/h) | U bottle-washer | U labeler |
|---|---|---|---|---|---|
| 8.1 | 500 | 63.50% | 15881 | 0.92989499 | 0.818723 |
| 16.2 | 1000 | 64.70% | 16187 | 0.92942698 | 0.834859 |
| 24.3 | 1500 | 65.30% | 16328 | 0.929179058 | 0.842066 |
| 32.4 | 2000 | 65.60% | 16402 | 0.930759819 | 0.84727 |
| 40.5 | 2500 | 66.20% | 16538 | 0.930000124 | 0.853026 |
| 48.6 | 3000 | 66.20% | 16541 | 0.929844597 | 0.853229 |
| 56.7 | 3500 | 66.20% | 16556 | 0.930094712 | 0.855943 |

*Figure 75 – "Scenario 1" results (table)*

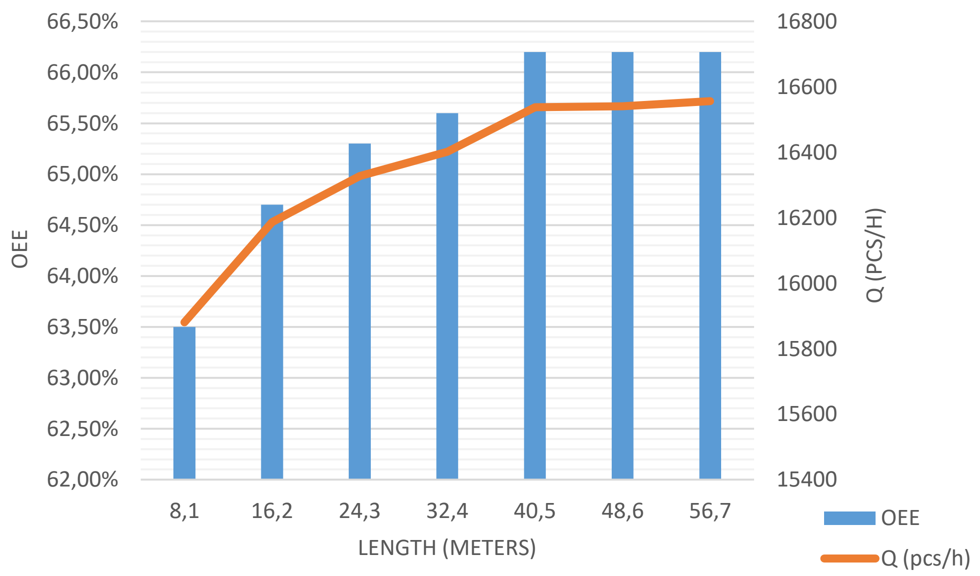



*Figure 76 - "Scenario 1" results (graph)*

The histogram in the figure above, shows as the increase of length of the second buffer involves a significant effect on line efficiency from 24.3 meters on. The average OEE value changes from 63.5% with 8.1 meters of buffer length to 65.3% with a length of the second buffer equal to 24.3 meters. After 40.5 meters of length, corresponding with 66.2% of OEE and a productivity around 16600 bottles/hour, a further increase in the buffer size results into a null or only marginal increase in efficiency. Since the aim is to maximize the throughput of the line, a buffer size of 40.5 meters is considered as the optimal one.

| Lenght | Capacity | OEE | Q (pcs/h) | U bottle-washer | U labeler |
|---|---|---|---|---|---|
| 25.5 | 1574 | 66.06% | 16515 | 0.9294 | 0.8508 |
| 33.6 | 2074 | 66.03% | 16507 | 0.9313 | 0.8521 |
| 41.7 | 2574 | 65.87% | 16469 | 0.9304 | 0.854 |
| 49.8 | 3074 | 66.05% | 16512 | 0.9305 | 0.8508 |
| 57.9 | 3574 | 66.09% | 16523 | 0.9295 | 0.8501 |

*Figure 77 – "Scenario 2" results (table)*

As can be seen from the table in Figure 77, a change in the length of the buffer 1 is not significant for the purpose of further improving the OEE value of the line. Indeed, OEE and throughput values that justify an increase in length and therefore in buffer capacity are not noticed. So, the analysis of scenario 2 makes us understand that changing the capacity of buffer 1 does not lead to substantial improvements.

At the end of this analysis, if we consider the optimal size of the 40.5 meter buffer, the following improvements are evident:

- The *OEE* grows from 63.5% to 66.2%, so it undergoes an increase of 2.7%;
- The *throughput* grows from 15871.5 pcs / h to 16538 pcs / h, so 667 pcs / h are produced which correspond to 4.2% more.

The improvements of the results are shown in the following figures.

| | AS IS | Buffer Sizing |
|---|---|---|
| **OEE** | 63.50% | 66.20% |
| **Q (pcs/h)** | 15871.5 | 16538 |

*Figure 78 – Improvement of the results (table)*

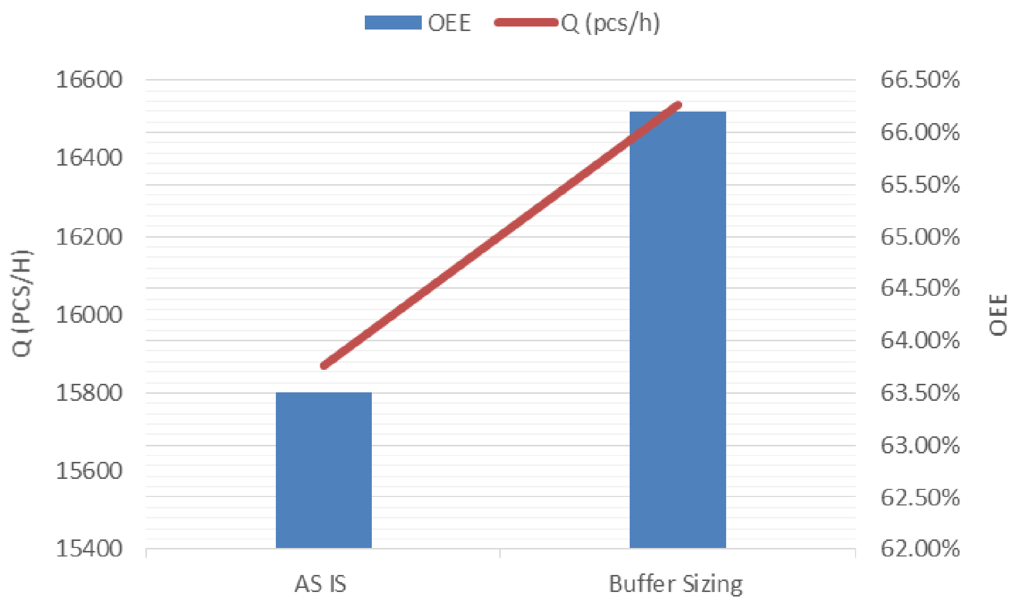



*Figure 79 - Improvement of the results (graph)*

## 6.3 Economical Analysis

The economic analysis proposed is the last step and allows to evaluate the improvement proposal in economic terms. Economy data are fictitious. The solution proposed through the simulation method consists in optimizing the buffer size. An optimally sized buffer improves OEE and throughput of the line.

The results we have obtained indicate that to have an effective improvement in performance, the length of the second buffer must increase from its 8.1 meters to 40.5 meters. The cost of investment and the recoverable OEE as well as the pay back period have been calculated.

The cost of the investment to increase the length of the second buffer is assumed to be € 1000,00/m. The labor cost of the project and the rearrangement of the layout are assumed to amount to € 8000,00 and about € 10000,00, respectively. The modification of the buffer length brings additional fixed costs of the period; these cost items form the negative factor of the cash flow of the period in the NPV formula.

| INVESTMENT COST | | INCREMENTAL FIXED COST (year) | |
|---|---|---|---|
| *Cost item* | *Cost (€)* | *Cost item* | *Cost (€)* |
| Additional conveyor | 32400 | Maintenance | 2785 |
| Project | 8000 | Cleanings | 3360 |
| Building Cost | 9600 | WIP | 95 |
| | 50000 | Utilities | 1920 |
| | | Other costs | 2000 |
| | | | 10160 |

*Figure 80 – Cost of new scenario*

Given an increase of 667 bottles/hour from the optimized buffer sized, the annual increase amount to about 4.012.772 bottles. The contribution margin for the first level has been calculated as the production for the unitary contribution without fixed costs. The unitary contribution it is considered with a unitary contribution margin range that varies from € 0,01 to € 0,25 per bottle. For the second level incremental fixed costs such as cost for maintenance, cleanings, utilities, work-in-process and others have also been considered.

| | CM 1 | CM 2 | CM 3 | CM 4 | CM 5 | CM 6 | CM 7 | CM 8 |
|---|---|---|---|---|---|---|---|---|
| **Investment cost (€)** | 32,400.00 | 32,400.00 | 32,400.00 | 32,400.00 | 32,400.00 | 32,400.00 | 32,400.00 | 32,400.00 |
| **Cost of the project (€)** | 17,600.00 | 17,600.00 | 17,600.00 | 17,600.00 | 17,600.00 | 17,600.00 | 17,600.00 | 17,600.00 |
| **Total (€)** | 50,000.00 | 50,000.00 | 50,000.00 | 50,000.00 | 50,000.00 | 50,000.00 | 50,000.00 | 50,000.00 |
| **CM 1^ Level (€/pcs)** | *0.01* | *0.02* | *0.03* | *0.05* | *0.10* | *0.15* | *0.20* | *0.25* |
| **Production increase (pcs/year)** | 4,012,672 | 4,012,672 | 4,012,672 | 4,012,672 | 4,012,672 | 4,012,672 | 4,012,672 | 4,012,672 |
| **CM 1^ Level (€)** | 40126.72 | 80253.44 | 120380.16 | 200633.6 | 401267.2 | 601900.8 | 802534.4 | 1003168.0 |
| **Fixed costs (€)** | 10,160.00 | 10,160.00 | 10,160.00 | 10,160.00 | 10,160.00 | 10,160.00 | 10,160.00 | 10,160.00 |
| **CM 2^ Level (€)** | 29,966.72 | 70,093.44 | 110,220.16 | 190,473.60 | 391,107.20 | 591,740.80 | 792,374.40 | 993,008.00 |

*Figure 81 – Contribution margin calculation*

The payback period was calculated starting from the second level margin contribution using the *Net Present Value* index, defined as follows:

$$NPV_t = -C_0 + \sum_{t=0}^{N} \frac{C_t}{(1+i)^t}$$

Where:

- $C_0$ = The initial investment;
- C = The cash flow;
- i = The interest rate.

| 2^ Level Cash Flow | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Year | 0.01 € CM | 0.02 € CM | 0.03 € CM | 0.05 € CM | 0.10 € CM | 0.15 € CM | 0.20 € CM | 0.25 € CM |
| 0 | - 50,000.00 € | - 50,000.00 € | - 50,000.00 € | - 50,000.00 € | - 50,000.00 € | - 50,000.00 € | - 50,000.00 € | - 50,000.00 € |
| 1 | - 21,460.27 € | 16,755.66 € | 54,971.58 € | 131,403.43 € | 322,483.05 € | 513,562.67 € | 704,642.29 € | 895,721.90 € |
| 2 | 5,720.43 € | 80,332.47 € | 154,944.52 € | 304,168.60 € | 677,228.81 € | 1,050,289.02 € | 1,423,349.22 € | 1,796,409.43 € |
| 3 | 31,606.81 € | 140,881.82 € | 250,156.83 € | 468,706.86 € | 1,015,081.91 € | 1,561,456.97 € | 2,107,832.02 € | 2,654,207.08 € |
| 4 | 56,260.51 € | 198,547.87 € | 340,835.23 € | 625,409.96 € | 1,336,846.77 € | 2,048,283.59 € | 2,759,720.40 € | 3,471,157.22 € |
| 5 | 79,740.22 € | 253,467.91 € | 427,195.61 € | 774,651.01 € | 1,643,289.50 € | 2,511,927.99 € | 3,380,566.48 € | 4,249,204.97 € |

*Figure 82 – Payback Period calculation*

Analysing the results, we can see that the payback period decreases as the contribution margin increases. In particular, with a contribution margin of 0.01 € the period of time required to recoup the funds expended in the investment is equal to 1 year and 9 months. This period decreases if the contribution margin increases.

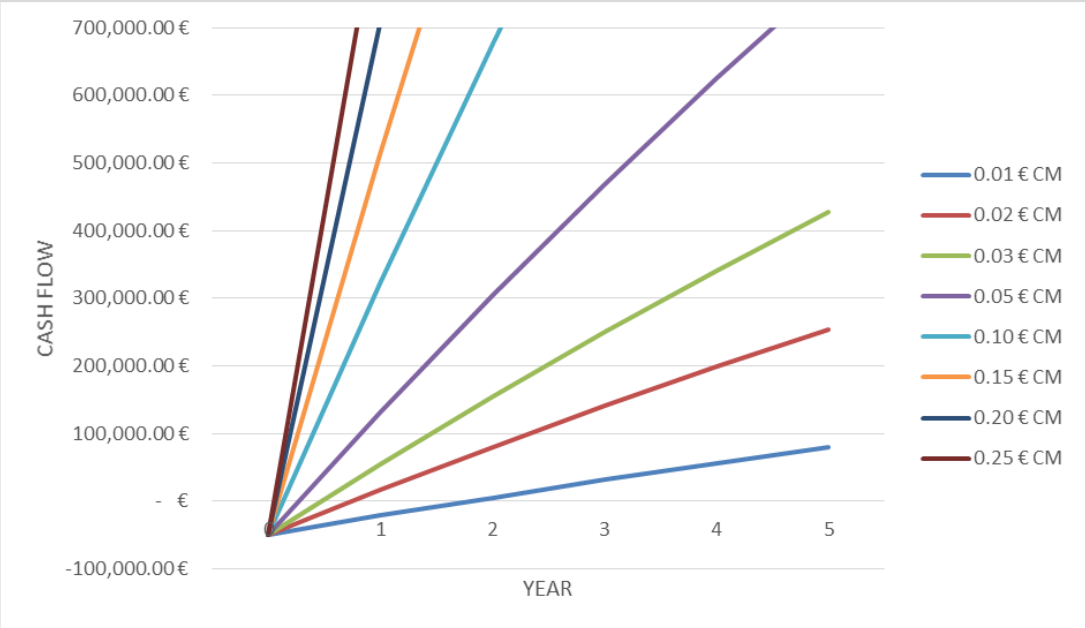



*Figure 83 – Payback Period graphic*

## 6.4 Simulative Approach vs Analytical Approach

Simulation is a tool used for processing information and data and predicting the responses of a real system to specific inputs, thus becoming an effective support in analysis, performance evaluation and decision processes. In our case, this type of approach has allowed us to represent the *as is* situation (the current state of the line), and then, through the Parameter Variation we went to study different scenarios. In particular, we have studied the intermediate buffer sizing to improve the performance of the line in terms of OEE and throughput.

Intermediate buffers built between the various machines in an asynchronous automatic (or semi-automatic) production line may increase the reliability of the whole system by limiting the consequences of micro-downtime, and saving companies from making inadequate purchases of oversized equipment (*Battini et al., 2009*).

Although the simulative approach returns us as very similar results to the real ones, it is often very expensive in terms of time and costs. In fact, the construction of a model requires a big investment in terms of time, since these software are very structured, they can carry out an enormous number of functions and often it takes a long time to represent correctly the reality.

As an alternative to the simulation method, the optimal sizing of the buffers can be carried out through an analytical approach that involves the use of a formula. This analytical sizing tool is called: *Buffer design for availability (BDFA), Battini et al., 2013.* This formula allows to describe the system in question using experimental formulas and then use a synthesis matrix to quickly obtain the project values. Unlike the simulative approach, this type of method is immediate and easy to apply.

After having explained it theoretically in chapter 2, in this section we will apply the formula in two different scenarios, then comparing the results with those of the simulative approach.

The formula to be applied for the optimal sizing of the buffer is as follows:

$$C_{max} = K(P, R, \beta) \times G \times Q$$

where:

- Cmax is the maximum capacity buffer level (pieces), and it is also the optimal assumption level;
- Q is the level of production throughput in pieces/hours;
- G is definied as *max* MTTR between the two WSs (MTTRa, MTTRb) and it is measured in hours;
- K is a “safety factor”, function of P and R (*Battini et al., 2009*), but also function of β, shape parameter of the Weibull distribution.

| | | | *β* | | | | |
|---|---|---|---|---|---|---|---|
| | | | *First infant failures* | *Last infant failures and random failure* | *First wearout failures* | | *Predictable failures (Battini et al., 2009)* |
| | | | $\beta < 0.7$ | $0.8 \leq \beta \leq 1$ | $\beta = 2$ | $\beta = 3$ | $\beta > 3$ |
| *R* | $R < 0.95$ | 6.2 | 7.5 | 6.2 | 3.6 | 2.6 | 1.8 |
| | $0.95 \leq R \leq 1.05$ | 6.1 | 6.2 | 6.1 | 3.3 | 2.4 | 1.1 |
| | $R > 1.05$ | 5.9 | 6 | 5.8 | 3.2 | 2 | 1.5 |

*Figure 84 – Cross matrix of K (Faccio et al., 2013)*

## 6.4.1 Application of the formula considering all the data

The method indicates that the two work-stations must be considered before and after the Buffer. In this case, as in the simulative approach we will consider the *Bottle-washer* and the *Labeler*. In this first application of the formula we consider every micro-downtimes of the work-stations. The following table summarizes the MTTF and MTTR of each station:

| *Machine* | *Symbol* | *Micro-downtimes* | *MTTF* | *MTTR* |
|---|---|---|---|---|
| Buffer 1 | A | Lateral stuck | 43.620 | 1.940 |
| | C | Fallen bottle on FT | 83.355 | 2.179 |
| Bottle-washer | A | Stuck load | 29.648 | 1.828 |
| | D | Stuck unload | 35.924 | 2.042 |
| | E | Out of sync machine | 62.071 | 2.006 |
| Buffer 2 | B | Bottle block | 51.775 | 1.259 |
| Labeler | A | Unstuck labels | 58.429 | 1.847 |
| | B | Wrong positioning | 54.858 | 3.593 |

*Figure 85 – MTTF and MTTR of the work-stations*

Afterwards, the data of each station were fitted according to a *Weibull distribution* to get the shape and scale parameters, using the statistical software Minitab.

**Bottle Washer**

Descriptive Statistics

| N | N* | Mean | StDev | Median | Minimum | Maximum | Skewness | Kurtosis |
|---|---|---|---|---|---|---|---|---|
| 150 | 0 | 1.95864 | 0.924201 | 1.83151 | 0.334614 | 8.13752 | 2.09399 | 12.1253 |

*Figure 86 – Descriptive statistics of Bottle washer*

ML Estimates of Distribution Parameters

| Distribution | Location | Shape | Scale | Threshold |
|---|---|---|---|---|
| Normal* | 1.95864 | | 0.92420 | |
| Box-Cox Transformation* | 1.36396 | | 0.31451 | |
| Lognormal* | 0.56533 | | 0.48405 | |
| 3-Parameter Lognormal | 1.02725 | | 0.29429 | -0.96026 |
| Exponential | | | 1.95864 | |
| 2-Parameter Exponential | | | 1.63493 | 0.32371 |
| Weibull | | 2.19146 | 2.20776 | |
| 3-Parameter Weibull | | 1.89042 | 1.89068 | 0.27663 |
| Smallest Extreme Value | 2.49492 | | 1.61694 | |
| Largest Extreme Value | 1.55706 | | 0.71436 | |
| Gamma | | 4.83678 | 0.40495 | |
| 3-Parameter Gamma | | 5.79822 | 0.36484 | -0.15677 |
| Logistic | 1.90116 | | 0.47790 | |
| Loglogistic | 0.59561 | | 0.26453 | |
| 3-Parameter Loglogistic | 1.06116 | | 0.16107 | -1.04329 |
| Johnson Transformation* | -0.04464 | | 0.98207 | |

*Figure 87 – Distribution parameter of Bottle-Washer*

## Labeler

Descriptive Statistics

| N | N* | Mean | StDev | Median | Minimum | Maximum | Skewness | Kurtosis |
|---|---|---|---|---|---|---|---|---|
| 100 | 0 | 2.60223 | 1.25719 | 2.31035 | 0.592028 | 6.77914 | 0.788185 | 0.653613 |

*Figure 88 – Descriptive statistics of Labeler*

ML Estimates of Distribution Parameters

| Distribution | Location | Shape | Scale | Threshold |
|---|---|---|---|---|
| Normal* | 2.60223 | | 1.25719 | |
| Box-Cox Transformation* | 1.56620 | | 0.38828 | |
| Lognormal* | 0.83306 | | 0.51977 | |
| 3-Parameter Lognormal | 1.20211 | | 0.35236 | -0.93540 |
| Exponential | | | 2.60223 | |
| 2-Parameter Exponential | | | 2.03051 | 0.57172 |
| Weibull | | 2.21512 | 2.94609 | |
| 3-Parameter Weibull | | 1.73423 | 2.37703 | 0.48112 |
| Smallest Extreme Value | 3.26792 | | 1.46415 | |
| Largest Extreme Value | 2.02264 | | 1.00358 | |
| Gamma | | 4.21421 | 0.61749 | |
| 3-Parameter Gamma | | 3.80215 | 0.65607 | 0.10774 |
| Logistic | 2.51077 | | 0.71130 | |
| Loglogistic | 0.85669 | | 0.29677 | |
| 3-Parameter Loglogistic | 1.07471 | | 0.23541 | -0.54423 |
| Johnson Transformation* | -0.08414 | | 0.97527 | |

*Figure 89 – Distribution parameter of Labeler*

The following table summarizes all the data necessary for the application of the formula for both work-stations.

| | *shape (β)* | *scale (α)* | *AD* | *P-Value* | *MTTR (min)* | *MTTR (h)* |
|---|---|---|---|---|---|---|
| **bw_ttr (station A)** | 2.191 | 2.208 | 1.276 | < 0.010 | 1.95864 | 0.032644 |
| **l_ttr (station B)** | 2.215 | 2.946 | 0.437 | > 0.250 | 2.60223 | 0.0433705 |

*Figure 90 - Summary table for both work-stations*

Now all the data are ready and you can proceed with the application of the formula by calculating the various parameters:

$$C_{max} = K(P, R, \beta) \times G \times Q = 2583\ pcs$$

Where:

- G=max[MTTRa,MTTRb] = 0.0433705 h;
- $R = {MTTRa}/{MTTRb}$ = 0.752678;
- $K(P, R, \beta)$ = 3.6 (*first wearout failure with R<0.95*);
- Q = 16538 pcs/h.

After applying the formula, the result obtained tells us that the optimal buffer of the section of the line between the Bottle Washer and the Labeler has a capacity equal to 2583 pcs.

| | **SIMULATIVE APPROACH** | **ANALYTIC APPROACH** |
|---|---|---|
| ***Buffer Capacity (pcs)*** | 2500 | 2583 |

*Figure 91 – Comparison of results: Simulative Approach vs Analytic Approach*

As shown in the table in *Figure 91*, there is a difference of 83 bottles between the result of the simulation and that of the formula. This difference amounts to 3.4%.

It is interesting to observe how the costant K of the *figure 84* varies if we want to get a buffer of 2500 pcs.

$$K(P,R,\beta) = \frac{Capaciry_{reuired}}{G \times Q} = \frac{2500}{0.0433705 \times 16538} = 3{,}5$$

After applying the inverse formula, we note that the K required to obtain a buffer of 2500 pcs is equal to 3.5.

| K (Battini et al., 20113) | K (new) |
|---|---|
| 3.6 | 3.5 |

*Figure 92 – New K of formula*

Therefore, we can state that to obtain the same result of the simulation, the K of the table of the paper by *Battini et al. (2013)*, must be reduced by a percentage equal to 2.8%.

## 6.4.2 Application of the formula considering the most impacting micro-downtimes

In this second application of the formula we try to see what happens if instead of considering all the causes of micro-downtimes, we consider only the most impacting ones.

To achieve this goal, given the input data, we try to apply the formula considering only 80% of the causes of micro-downtimes. As the Labeler is subject to only two causes of micro-downtimes, this study does not make sense for this work-station and therefore it will be applied only to the Bottle Washer.

| *Machine* | *Symbol* | *Micro-downtimes* | *MTTF* | *MTTR* |
|---|---|---|---|---|
| Buffer 1 | A | Lateral stuck | 43.620 | 1.940 |
| | C | Fallen bottle on FT | 83.355 | 2.179 |
| Bottle-washer | A | Stuck load | 29.648 | 1.828 |
| | D | Stuck unload | 35.924 | 2.042 |
| | E | Out of sync machine | 62.071 | 2.006 |
| Buffer 2 | B | Bottle block | 51.775 | 1.259 |
| Labeler | A | Unstuck labels | 58.429 | 1.847 |
| | B | Wrong positioning | 54.858 | 3.593 |

*Figure 93 – MTTF and MTTR of work-station*

We can see from the data in the table in *Figure 86*, that the micro-stops A and D occur much more often than the micro-downtime E. The respective MTTF are indeed 29.648 min for the micro-downtime A and 35.924 min for the micro-downtime D, unlike the micro-downtime E which is equal to 62.021 min. We calculated the relative frequency and the percentage frequency on a working hour and the following table shows the results obtained:

| MICRO-DOWNTIME | COUNT | FREQUENCE | FREQUENCE % | |
|---|---|---|---|---|
| A | 2.02 | 0.434 | 43.4% | 79.3% (A + D) |
| D | 1.67 | 0.359 | 35.9% | |
| E | 0.966 | 0.207 | 20.7% | |
| TOT | 4.656 | 1.00 | 100% | |

*Figure 94 – Micro-downtime frequencies of Bottle Washer*

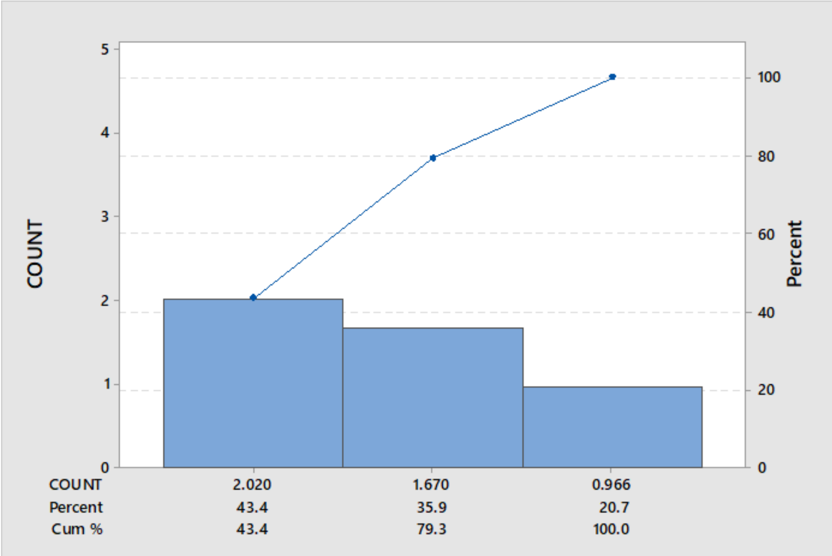



*Figure 95 – Frequencies graph of micro-downtimes BW*

Both from the table and from the graph, we can see how considering only the A and D micro-downtimes we cover about 80% of the frequency of occurrence.

Now we try to evaluate the sizing of the buffer through the analytical method considering only the micro-downtimes that cover 80% of the frequency of occurrence.

Also in this case the first step is to fit the data through the *Weibull distribution* with Minitab to obtain the parameters necessary for the application of the formula.

**Bottle Washer**

Descriptive Statistics

| N | N* | Mean | StDev | Median | Minimum | Maximum | Skewness | Kurtosis |
|---|---|---|---|---|---|---|---|---|
| 100 | 0 | 1.93515 | 0.969863 | 1.77267 | 0.334614 | 8.13752 | 2.82055 | 15.8290 |

*Figure 96 – Descriptive Statistics of Bottle Washer*

ML Estimates of Distribution Parameters

| Distribution | Location | Shape | Scale | Threshold |
|---|---|---|---|---|
| Normal* | 1.93515 | | 0.96986 | |
| Box-Cox Transformation* | 0.55759 | | 0.45953 | |
| Lognormal* | 0.55759 | | 0.45953 | |
| 3-Parameter Lognormal | 0.77168 | | 0.36480 | -0.38273 |
| Exponential | | | 1.93515 | |
| 2-Parameter Exponential | | | 1.61670 | 0.31845 |
| Weibull | | 2.08198 | 2.18419 | |
| 3-Parameter Weibull | | 1.79347 | 1.83929 | 0.29925 |
| Smallest Extreme Value | 2.52082 | | 1.74958 | |
| Largest Extreme Value | 1.54448 | | 0.66780 | |
| Gamma | | 5.03452 | 0.38438 | |
| 3-Parameter Gamma | | 4.54130 | 0.40766 | 0.08386 |
| Logistic | 1.84205 | | 0.46617 | |
| Loglogistic | 0.56748 | | 0.24970 | |
| 3-Parameter Loglogistic | 0.71325 | | 0.21411 | -0.26689 |
| Johnson Transformation* | -0.05520 | | 0.91896 | |

*Figure 97 – Distribution Parameter of Bottle Washer*

## Labeler

The parameters of the Labeler are the same of the case discussed in the paragraph 6.4.1.

The following table summarizes all the data necessary for the application of the formula for both work-stations.

| | *shape (β)* | *scale (α)* | *AD* | *P-Value* | *MTTR (min)* | *MTTR (h)* |
|---|---|---|---|---|---|---|
| **bw_ttr (station A)** | 2.082 | 2.184 | 2.015 | < 0.010 | 1.93515 | 0.0322525 |
| **l_ttr (station B)** | 2.215 | 2.946 | 0.437 | > 0.250 | 2.60223 | 0.0433705 |

*Figure 98 – Summary tables for both work-station*

At the end, we proceed to the calculation of the buffer capacity by applying the formula as in the case seen in the previous paragraph.

$$C_{max} = K(P, R, \beta) \times G \times Q = 2583\ pcs$$

Where:

- G = max[MTTRa,MTTRb] = 0.0433705 h;
- $R = {}^{MTTRa}/_{MTTRb}$ = 0.743651;
- $K(P, R, \beta)$ = 3.6 (*first wearout failure with R<0.95*);
- Q = 16538 pcs/h.

From this study we note how, considering the 80% of the frequency of occurrence of the micro-downtimes referring to the Bottle-Washer, the optimal buffer size does not change.

It can be seen that the Labeler has the greatest weight between the two work-stations, since when we go to establish parameter G in both cases the maximum corresponds to the MTTR of the Labeler. Thus, the result does not change even if the number of micro-downtimes causes of the Bottle-Washer is reduced. In terms of time, therefore, it is advisable to use only the most impactful micro-downtimes, ie those that cover 80% of occurrence, in order to carry out optimal sizing of the buffer of the section studied.

# Conclusion

Nowadays the management of companies is based on the optimization of production processes, in particular, the aim is to reduce costs, increase flexibility and maximize efficiency. The market is characterized by ever-increasing variability in customer demands and, as a result, companies must be ready for a rapid response to satisfy the demand. A complete integration between the business functions is fundamental in order to create an efficient supply chain throughout its chain. To ensure speed and effectiveness of the system, a shared company performance control system based on the KPI measurement is required. This function of control and improvement of production processes is carried out by the production area, which is therefore fundamental not only at an operational level, but also strategically for the entire company; perform effectively and efficiently in this area brings great benefits for the entire business.

In sectors characterized by highly automated processes, such as food and beverage, measuring the performance of the production plant, by indicators such as the OEE, make possible to assess the current condition of the equipment and to understand where and how to improve. This also leads to the possibility of quantifying the extent of the improvement, to understand the real benefits and justify any interventions.

The simulation is a very useful tool for the improvement of the production systems since it allows to represent through a model the real situation on which then we go to study its criticalities and possible improvements.

The present thesis project aimed to optimize the size of the buffers of the critical section of the line through a simulation approach to improve performance and productivity of the line. The results obtained were then compared with those deriving from the application of an analytical approach to analyse any differences. The goal is to operate on the losses of inefficiencies that affect the line, which are due to failures, set-up for predictive maintenance, reduced buffer capacity between workstations.

The case study was carried out following the General Methodology for Applying Simulation to Problem Solving (Rossetti, 2015). The input data were studied and analysed to be used in a simulation model based on discrete events in AnyLogic that would reflect the behaviour of the initial one. The most interesting features of the simulation model created are the flow chart that controls the actions of the bottles along the line and the state diagrams that allow you to accurately manage the micro-downtimes that affect the different workstation of the line. Then the model obtained the validation and it was ready to be studied for the improvements.

The first simulation resulted in the current situation of the line: it has an OEE of 63.49% with a productivity of 15871.5 bottles / hour. The improvement proposal consisted in efficiently decoupling the two stations of  the critical section of the line production through an optimal sizing of the intermediate buffer. To make that it was performed a test called *Parameter Variation* in AnyLogic. The goal was to see the effects in terms of OEE and productivity by increasing and decreasing the buffer length of the critical section, ie the one between the Bottle Washer and the Labeler.

The simulation result shows that we get the optimal size of the buffer with a length of 40.5 meters. This length corresponds to a capacity of 2500 pieces and guarantees an increase in efficiency up to 66.2% of OEE and productivity up to 16538 pcs.

| | **AS IS** | **Buffer Sizing** |
|---|---|---|
| **OEE** | 63.50% | 66.20% |
| **Q (pcs/h)** | 15871.5 | 16538 |

*Figura 99 – Improvements of the line (table)*

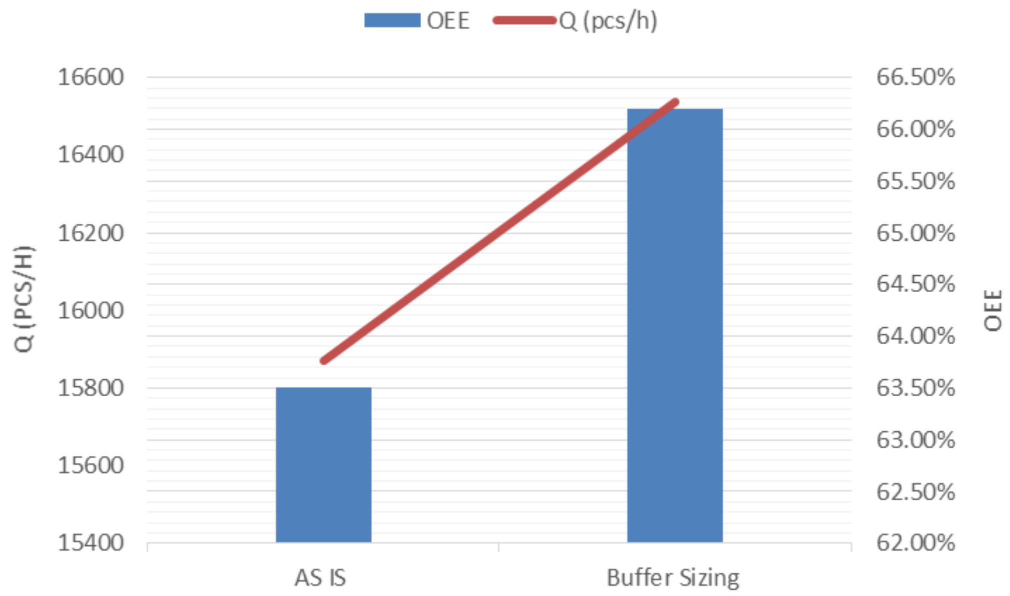



*Figura 100 – Improvement of the line (graph)*

So as we can see from the graph above, the layout change leads to the following improvements:

- + 2,7 % in terms of OEE;
- + 4,2 % in terms of throughput (pcs/h).

In addition to the simulative approach, optimal buffer size was studied using an analytical sizing tool is called: *Buffer design for availability (BDFA), Battini et al., 2013.* This formula makes it possible to obtain an optimal size of the buffer, saving time and costs, thus being very functional. It allows us to obtain the optimal capacity by multiplying three parameters: the productivity Q, the maximum MTTR between the two stations G, and a constant K obtained from a cross-matrix which considers three parameters (P, R, β).

The results of the simulative approach were then compared with those of the analytical approach.

| | SIMULATIVE APPROACH | ANALYTIC APPROACH |
|---|---|---|
| ***Buffer Capacity (pcs)*** | 2500 | 2583 |

*Figura 101 – Comparison between two different approach*

As shown in the table in *Figure 101*, there is a difference of 83 bottles between the result of the simulation and the result of the formula. This difference amounts to 3.4%. After applying the inverse formula to obtain a buffer of 2500 pcs, we note that the K is equal to 3.5.

| K (Battini et al., 20113) | K (new) |
|---|---|
| 3.6 | 3.5 |

*Figure 92 – New K of the formula*

Therefore, we can state that to obtain the same result of the simulation, the K of the table of the paper by *Battini et al. (2013)*, must be reduced by a percentage equal to 2.8%.

Finally, we tried to analyse the application of the formula by considering only the most impacting microscopes, in particular those that covered 80% of the frequency of occurrence. This type of study was only possible on the Bottle Washer as the Labeler presented only two causes of micro-downtimes. The result obtained, considering only the micro-downtimes A and D of the Bottle Washer is the same as in the previous case. This is justified by the fact that in this case, in the calculation of the parameter G, the MTTR that mostly impacts is MTTR of the Labeler because it is greater. So even considering only the micro-screens that cover 80% of the occurrence frequency for the Bottle Washer, the buffer is optimized.

Moreover, it could be interesting to carry out the same study, therefore comparing the simulative approach with the analytical approach, on a section of a line that presents two machines subject to many causes of micro-downtimes. This case would allow to analyse how the size of the buffer varies, trying to consider the micro-screens that cover 80% of the stops occurring for both stations examined, in particular the behaviour of the K parameter of the formula.

# Appendix A: Data Samples

| *Bottle Washer* | | | *Buffer 1* | | *Buffer 2* | *Labeler* | |
|---|---|---|---|---|---|---|---|
| **A** | **D** | **E** | **A** | **C** | **B** | **A** | **B** |
| 1.245464 | 1.283387093 | 3.081384 | 1.376329628 | 2.238202 | 1.608464 | 2.037356 | 3.172079 |
| 3.28956 | 2.809593612 | 0.675434 | 1.640766305 | 4.460319 | 1.574835 | 1.650914 | 3.712083 |
| 1.096435 | 3.33356832 | 1.635327 | 1.196542987 | 3.807615 | 1.581796 | 3.322846 | 3.163333 |
| 0.49159 | 1.365147993 | 2.997678 | 1.269679816 | 2.771791 | 1.353447 | 1.145107 | 2.251163 |
| 0.934495 | 3.882275339 | 2.149491 | 2.809527307 | 1.319552 | 1.025252 | 4.311681 | 2.714438 |
| 0.334614 | 1.573813196 | 3.250094 | 1.942726738 | 2.188094 | 2.05819 | 1.222086 | 1.988439 |
| 1.585382 | 1.542885553 | 2.505693 | 2.806288179 | 2.119252 | 0.906511 | 0.692719 | 3.273415 |
| 1.379202 | 2.897258455 | 1.874736 | 3.232790164 | 3.460396 | 1.724093 | 2.241435 | 2.923639 |
| 1.395179 | 3.690847897 | 2.952961 | 2.96732766 | 1.989991 | 1.813097 | 1.749136 | 2.405676 |
| 1.109345 | 2.092782871 | 1.882869 | 1.165723803 | 2.059304 | 1.121375 | 0.968819 | 3.809995 |
| 1.781375 | 2.621315316 | 2.721399 | 2.014575354 | 3.587357 | 1.558435 | 3.795619 | 4.543872 |
| 2.331879 | 1.832021623 | 0.858069 | 1.97680212 | 3.081617 | 1.561919 | 3.318421 | 1.761649 |
| 1.660493 | 2.625309357 | 1.840823 | 0.710724178 | 3.43852 | 1.523197 | 6.484849 | 1.562408 |
| 8.137522 | 1.383958745 | 2.846349 | 1.469391234 | 1.314763 | 1.001774 | 2.820846 | 2.134472 |
| 2.639687 | 1.791095939 | 1.215761 | 0.956508639 | 2.857336 | 1.407223 | 2.289473 | 3.338802 |
| 1.203416 | 1.062739029 | 1.303902 | 1.972997897 | 2.321227 | 1.854241 | 1.789773 | 3.271223 |
| 2.80653 | 2.636520446 | 0.579813 | 1.054896481 | 1.470107 | 1.371688 | 2.495083 | 3.080125 |
| 1.596475 | 2.779719899 | 2.154071 | 3.608984607 | 2.117872 | 1.34441 | 0.622696 | 5.213718 |
| 0.760213 | 1.109881637 | 3.545559 | 4.033377593 | 1.506712 | 1.231782 | 1.638574 | 3.249236 |
| 1.944204 | 1.310491587 | 1.493659 | 1.505745552 | 1.08825 | 0.698326 | 1.676054 | 4.527682 |
| 1.310278 | 1.831005517 | 0.626249 | 1.688572639 | 2.781076 | 1.02906 | 1.560809 | 3.783708 |
| 1.514655 | 2.017894662 | 0.978508 | 5.366147115 | 2.880846 | 0.767985 | 1.68045 | 2.534738 |
| 2.887852 | 2.2967001 | 0.354669 | 3.124216373 | 0.33901 | 1.41529 | 3.828712 | 2.936864 |
| 1.049547 | 1.080177678 | 0.886556 | 3.109317176 | 2.018227 | 1.429231 | 0.592028 | 2.030211 |
| 0.99023 | 2.67721977 | 0.919243 | 1.041061672 | 1.039358 | 0.919748 | 2.0668 | 2.33122 |
| 3.366958 | 2.072412672 | 1.971868 | 0.952053588 | 2.450516 | 1.983099 | 3.943056 | 2.969754 |
| 0.631066 | 1.282659309 | 2.264908 | 1.474284786 | 2.20527 | 1.057087 | 1.093758 | 2.554809 |
| 2.320996 | 2.920662418 | 1.775882 | 1.499334684 | 1.070904 | 0.912209 | 1.325091 | 4.131528 |
| 2.259452 | 1.950731328 | 2.230803 | 0.689416675 | 3.036055 | 1.801549 | 2.221562 | 3.441681 |
| 2.358945 | 1.603748811 | 1.670306 | 1.104161562 | 0.475934 | 1.205541 | 2.349219 | 1.518255 |
| 2.547531 | 1.905430211 | 2.821788 | 3.181343934 | 2.255054 | 1.525848 | 4.537402 | 1.361215 |
| 1.754204 | 1.557736366 | 1.458309 | 2.358650485 | 1.820404 | 1.78365 | 2.171817 | 4.245538 |
| 2.145574 | 1.292320241 | 0.580736 | 1.648450843 | 2.920861 | 1.070252 | 0.677043 | 1.381353 |
| 1.213757 | 2.115997953 | 2.538251 | 1.671224402 | 2.430906 | 1.891312 | 2.651461 | 1.226546 |
| 1.791799 | 1.860404618 | 2.356519 | 1.853598031 | 2.023985 | 1.699838 | 0.940663 | 2.406985 |
| 1.534626 | 1.776800914 | 2.067629 | 2.231464855 | 1.154009 | 1.398814 | 1.303159 | 3.751928 |
| 1.186429 | 2.310298765 | 2.868629 | 2.012519828 | 1.888405 | 1.324895 | 2.12358 | 2.184495 |
| 2.97511 | 1.393561077 | 2.150079 | 0.900920852 | 2.088685 | 1.033576 | 1.998495 | 3.886708 |
| 1.096773 | 2.211516772 | 2.210352 | 1.76534098 | 0.784454 | 1.556059 | 2.054382 | 3.34653 |
| 1.191015 | 1.158077914 | 2.751203 | 1.854905196 | 2.803791 | 1.831458 | 1.910249 | 0.992272 |
| 1.702609 | 2.568377689 | 1.616757 | 1.33890947 | 1.430622 | 1.044801 | 1.86045 | 3.237056 |
| 2.844296 | 1.768542204 | 1.467362 | 3.160777576 | 3.554486 | 1.613803 | 1.332572 | 3.807865 |
| 0.899814 | 2.196397721 | 3.241447 | 2.131614279 | 1.439397 | 1.877943 | 4.976954 | 2.031824 |
| 0.823557 | 2.858975186 | 2.052039 | 1.226409982 | 1.113191 | 1.24927 | 1.685712 | 3.701848 |
| 1.423373 | 1.496940613 | 2.489586 | 0.938541691 | 1.311237 | 0.957284 | 1.493356 | 1.599197 |
| 3.654607 | 2.02329043 | 1.730992 | 2.777872601 | 2.688181 | 1.751886 | 1.742912 | 4.806167 |
| 1.290848 | 1.925605984 | 2.851935 | 1.540486341 | 2.745585 | 1.567593 | 3.247125 | 6.77914 |
| 1.640106 | 2.557090502 | 1.613172 | 1.782946659 | 3.69009 | 1.259327 | 0.827657 | 2.925748 |
| 1.742453 | 2.001380418 | 2.80155 | 1.588702362 | 2.054931 | 1.268775 | 1.84728 | 4.602864 |
| 1.549577 | 1.756847963 | 3.369663 | 1.261677359 | 1.272459 | 1.389414 | 1.707482 | 3.593264 |

*Figure 103 – Time to Failure (minutes)*

| Bottle Washer | | | Buffer 1 | | Buffer 2 | Labeler | |
|---|---|---|---|---|---|---|---|
| A | D | E | A | C | B | A | B |
| 1.245464 | 1.283387 | 3.081384 | 1.37633 | 2.238202 | 1.608464 | 2.037356 | 3.172079 |
| 3.28956 | 2.809594 | 0.675434 | 1.640766 | 4.460319 | 1.574835 | 1.650914 | 3.712083 |
| 1.096435 | 3.333568 | 1.635327 | 1.196543 | 3.807615 | 1.581796 | 3.322846 | 3.163333 |
| 0.49159 | 1.365148 | 2.997678 | 1.26968 | 2.771791 | 1.353447 | 1.145107 | 2.251163 |
| 0.934495 | 3.882275 | 2.149491 | 2.809527 | 1.319552 | 1.025252 | 4.311681 | 2.714438 |
| 0.334614 | 1.573813 | 3.250094 | 1.942727 | 2.188094 | 2.05819 | 1.222086 | 1.988439 |
| 1.585382 | 1.542886 | 2.505693 | 2.806288 | 2.119252 | 0.906511 | 0.692719 | 3.273415 |
| 1.379202 | 2.897258 | 1.874736 | 3.23279 | 3.460396 | 1.724093 | 2.241435 | 2.923639 |
| 1.395179 | 3.690848 | 2.952961 | 2.967328 | 1.989991 | 1.813097 | 1.749136 | 2.405676 |
| 1.109345 | 2.092783 | 1.882869 | 1.165724 | 2.059304 | 1.121375 | 0.968819 | 3.809995 |
| 1.781375 | 2.621315 | 2.721399 | 2.014575 | 3.587357 | 1.558435 | 3.795619 | 4.543872 |
| 2.331879 | 1.832022 | 0.858069 | 1.976802 | 3.081617 | 1.561919 | 3.318421 | 1.761649 |
| 1.660493 | 2.625309 | 1.840823 | 0.710724 | 3.43852 | 1.523197 | 6.484849 | 1.562408 |
| 8.137522 | 1.383959 | 2.846349 | 1.469391 | 1.314763 | 1.001774 | 2.820846 | 2.134472 |
| 2.639687 | 1.791096 | 1.215761 | 0.956509 | 2.857336 | 1.407223 | 2.289473 | 3.338802 |
| 1.203416 | 1.062739 | 1.303902 | 1.972998 | 2.321227 | 1.854241 | 1.789773 | 3.271223 |
| 2.80653 | 2.63652 | 0.579813 | 1.054896 | 1.470107 | 1.371688 | 2.495083 | 3.080125 |
| 1.596475 | 2.77972 | 2.154071 | 3.608985 | 2.117872 | 1.34441 | 0.622696 | 5.213718 |
| 0.760213 | 1.109882 | 3.545559 | 4.033378 | 1.506712 | 1.231782 | 1.638574 | 3.249236 |
| 1.944204 | 1.310492 | 1.493659 | 1.505746 | 1.08825 | 0.698326 | 1.676054 | 4.527682 |
| 1.310278 | 1.831006 | 0.626249 | 1.688573 | 2.781076 | 1.02906 | 1.560809 | 3.783708 |
| 1.514655 | 2.017895 | 0.978508 | 5.366147 | 2.880846 | 0.767985 | 1.68045 | 2.534738 |
| 2.887852 | 2.2967 | 0.354669 | 3.124216 | 0.33901 | 1.41529 | 3.828712 | 2.936864 |
| 1.049547 | 1.080178 | 0.886556 | 3.109317 | 2.018227 | 1.429231 | 0.592028 | 2.030211 |
| 0.99023 | 2.67722 | 0.919243 | 1.041062 | 1.039358 | 0.919748 | 2.0668 | 2.33122 |
| 3.366958 | 2.072413 | 1.971868 | 0.952054 | 2.450516 | 1.983099 | 3.943056 | 2.969754 |
| 0.631066 | 1.282659 | 2.264908 | 1.474285 | 2.20527 | 1.057087 | 1.093758 | 2.554809 |
| 2.320996 | 2.920662 | 1.775882 | 1.499335 | 1.070904 | 0.912209 | 1.325091 | 4.131528 |
| 2.259452 | 1.950731 | 2.230803 | 0.689417 | 3.036055 | 1.801549 | 2.221562 | 3.441681 |
| 2.358945 | 1.603749 | 1.670306 | 1.104162 | 0.475934 | 1.205541 | 2.349219 | 1.518255 |
| 2.547531 | 1.90543 | 2.821788 | 3.181344 | 2.255054 | 1.525848 | 4.537402 | 1.361215 |
| 1.754204 | 1.557736 | 1.458309 | 2.35865 | 1.820404 | 1.78365 | 2.171817 | 4.245538 |
| 2.145574 | 1.29232 | 0.580736 | 1.648451 | 2.920861 | 1.070252 | 0.677043 | 1.381353 |
| 1.213757 | 2.115998 | 2.538251 | 1.671224 | 2.430906 | 1.891312 | 2.651461 | 1.226546 |
| 1.791799 | 1.860405 | 2.356519 | 1.853598 | 2.023985 | 1.699838 | 0.940663 | 2.406985 |
| 1.534626 | 1.776801 | 2.067629 | 2.231465 | 1.154009 | 1.398814 | 1.303159 | 3.751928 |
| 1.186429 | 2.310299 | 2.868629 | 2.01252 | 1.888405 | 1.324895 | 2.12358 | 2.184495 |
| 2.97511 | 1.393561 | 2.150079 | 0.900921 | 2.088685 | 1.033576 | 1.998495 | 3.886708 |
| 1.096773 | 2.211517 | 2.210352 | 1.765341 | 0.784454 | 1.556059 | 2.054382 | 3.34653 |
| 1.191015 | 1.158078 | 2.751203 | 1.854905 | 2.803791 | 1.831458 | 1.910249 | 0.992272 |
| 1.702609 | 2.568378 | 1.616757 | 1.338909 | 1.430622 | 1.044801 | 1.86045 | 3.237056 |
| 2.844296 | 1.768542 | 1.467362 | 3.160778 | 3.554486 | 1.613803 | 1.332572 | 3.807865 |
| 0.899814 | 2.196398 | 3.241447 | 2.131614 | 1.439397 | 1.877943 | 4.976954 | 2.031824 |
| 0.823557 | 2.858975 | 2.052039 | 1.22641 | 1.113191 | 1.24927 | 1.685712 | 3.701848 |
| 1.423373 | 1.496941 | 2.489586 | 0.938542 | 1.311237 | 0.957284 | 1.493356 | 1.599197 |
| 3.654607 | 2.02329 | 1.730992 | 2.777873 | 2.688181 | 1.751886 | 1.742912 | 4.806167 |
| 1.290848 | 1.925606 | 2.851935 | 1.540486 | 2.745585 | 1.567593 | 3.247125 | 6.77914 |
| 1.640106 | 2.557091 | 1.613172 | 1.782947 | 3.69009 | 1.259327 | 0.827657 | 2.925748 |
| 1.742453 | 2.00138 | 2.80155 | 1.588702 | 2.054931 | 1.268775 | 1.84728 | 4.602864 |
| 1.549577 | 1.756848 | 3.369663 | 1.261677 | 1.272459 | 1.389414 | 1.707482 | 3.593264 |

*Figure 104 – Time to Repair (minutes)*

# Appendix B: Output Statistical Analysis Minitab®

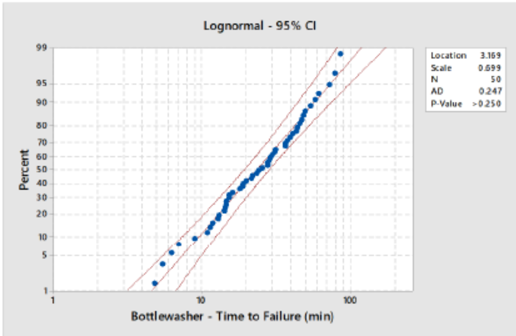
### Distribution ID Plot: TTF (min) Bottle-washer A

**Descriptive Statistics**

| N | Mean | StDev | Median | Min | Max | Skewness | Kurtosis |
|---|---|---|---|---|---|---|---|
| 50 | 29,6482 | 19,45 | 24,9622 | 4,7915 | 86,2514 | 1,07299 | 0,803603 |

**Goodness of Fit Test**

| Distribution | AD | P |
|---|---|---|
| Normal | 1,284 | <0.005 |
| Box-Cox Transformation | 0,246 | 0,747 |
| Lognormal | 0,246 | 0,747 |
| Exponential | 2,962 | <0.003 |
| Weibull | 0,308 | >0.250 |
| Smallest Extreme Value | 2,801 | <0.010 |
| Largest Extreme Value | 0,471 | 0,24 |
| Gamma | 0,217 | >0.250 |
| Logistic | 0,999 | 0,005 |
| Loglogistic | 0,299 | >0.250 |
| Johnson Transformation | 0,138 | 0,974 |



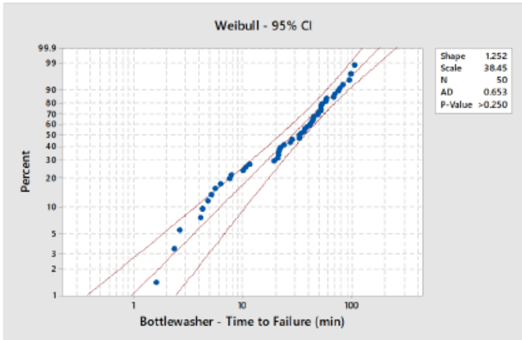

### Distribution ID Plot: TTF (min) Bottle-washer D

**Descriptive Statistics**

| N | Mean | StDev | Median | Min | Max | Skewness | Kurtosis |
|---|---|---|---|---|---|---|---|
| 50 | 35,924 | 27,0796 | 33,3865 | 1,60794 | 103,951 | 0,707773 | -0,13075 |

**Goodness of Fit Test**

| Distribution | AD | P |
|---|---|---|
| Normal | 0,859 | 0,025 |
| Box-Cox Transformation | 0,477 | 0,228 |
| Lognormal | 1,579 | <0.005 |
| Exponential | 0,998 | 0,116 |
| Weibull | 0,653 | 0,086 |
| Smallest Extreme Value | 1,911 | <0.010 |
| Largest Extreme Value | 0,654 | 0,085 |
| Gamma | 0,728 | 0,07 |
| Logistic | 0,751 | 0,028 |
| Loglogistic | 1,396 | <0.005 |
| Johnson Transformation | 0,433 | 0,292 |




### Distribution ID Plot: TTF (min) Bottle-washer E

**Descriptive Statistics**

| N | Mean | StDev | Median | Min | Max | Skewness | Kurtosis |
|---|---|---|---|---|---|---|---|
| 50 | 62,0714 | 25,5789 | 57,4527 | 24,7544 | 131,444 | 0,63317 | -0,23415 |

**Goodness of Fit Test**

| Distribution | AD | P |
|---|---|---|
| Normal | 0,682 | 0,07 |
| Box-Cox Transformation | 0,303 | 0,56 |
| Lognormal | 0,303 | 0,56 |
| Exponential | 8,158 | <0.003 |
| Weibull | 0,447 | >0.250 |
| Smallest Extreme Value | 1,595 | <0.010 |
| Largest Extreme Value | 0,356 | >0.250 |
| Gamma | 0,316 | >0.250 |
| Logistic | 0,66 | 0,049 |
| Loglogistic | 0,39 | >0.250 |
| Johnson Transformation | 0,159 | 0,947 |

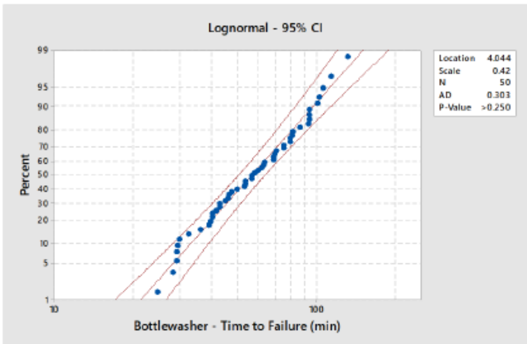



*Figure 105 – Probability distribution Identification for TTFs and TTRs*

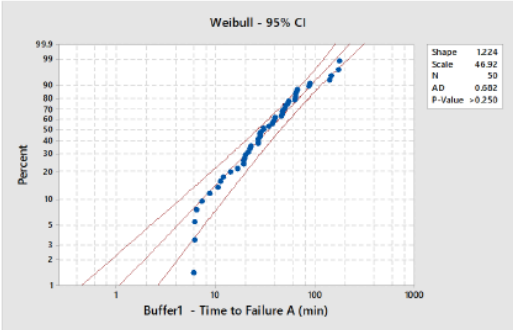
### Distribution ID Plot: TTF (min) Buffer 1 A

**Descriptive Statistics**

| N | Mean | StDev | Median | Min | Max | Skewness | Kurtosis |
|---|---|---|---|---|---|---|---|
| 50 | 43,6203 | 40,0247 | 30,2254 | 6,00177 | 174,7 | 1,97634 | 3,86575 |

**Goodness of Fit Test**

| Distribution | AD | P |
|---|---|---|
| Normal | 3,449 | <0.005 |
| Box-Cox Transformation | 0,338 | 0,489 |
| Lognormal | 0,338 | 0,489 |
| Exponential | 1,288 | 0,053 |
| Weibull | 0,682 | 0,073 |
| Smallest Extreme Value | 6,028 | <0.010 |
| Largest Extreme Value | 1,041 | <0.010 |
| Gamma | 0,546 | 0,192 |
| Logistic | 1,893 | <0.005 |
| Loglogistic | 0,3 | >0.250 |
| Johnson Transformation | 0,283 | 0,62 |



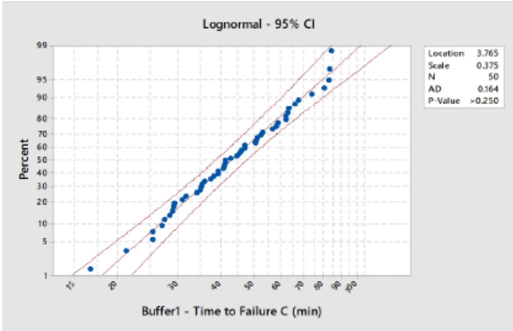

### Distribution ID Plot: TTF (min) Buffer 1 C

**Descriptive Statistics**

| N | Mean | StDev | Median | Min | Max | Skewness | Kurtosis |
|---|---|---|---|---|---|---|---|
| 50 | 46,1522 | 17,0646 | 42,067 | 16,6907 | 84,3301 | 0,639071 | -0,2452 |

**Goodness of Fit Test**

| Distribution | AD | P |
|---|---|---|
| Normal | 0,73 | 0,053 |
| Box-Cox Transformation | 0,165 | 0,937 |
| Lognormal | 0,165 | 0,937 |
| Exponential | 9,474 | <0.003 |
| Weibull | 0,571 | 0,146 |
| Smallest Extreme Value | 1,897 | <0.010 |
| Largest Extreme Value | 0,189 | >0.250 |
| Gamma | 0,241 | >0.250 |
| Logistic | 0,622 | 0,068 |
| Loglogistic | 0,234 | >0.250 |
| Johnson Transformation | 0,155 | 0,952 |



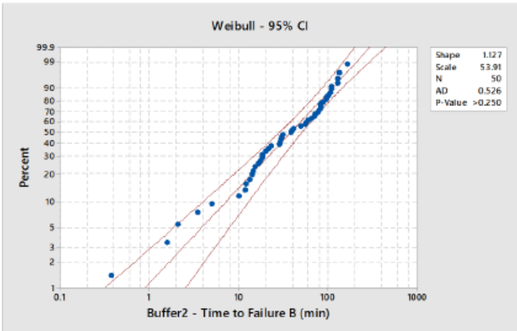

### Distribution ID Plot: TTF (min) Buffer 2 B

**Descriptive Statistics**

| N | Mean | StDev | Median | Min | Max | Skewness | Kurtosis |
|---|---|---|---|---|---|---|---|
| 50 | 51,775 | 41,9155 | 38,4746 | 0,376331 | 165,029 | 0,739693 | -0,35375 |

**Goodness of Fit Test**

| Distribution | AD | P |
|---|---|---|
| Normal | 1,483 | <0.005 |
| Box-Cox Transformation | 0,563 | 0,138 |
| Lognormal | 1,317 | <0.005 |
| Exponential | 0,592 | 0,375 |
| Weibull | 0,526 | 0,188 |
| Smallest Extreme Value | 2,126 | <0.010 |
| Largest Extreme Value | 1,159 | <0.010 |
| Gamma | 0,524 | 0,216 |
| Logistic | 1,448 | <0.005 |
| Loglogistic | 0,907 | 0,01 |
| Johnson Transformation | 0,264 | 0,683 |




### Distribution ID Plot: TTF (min) Labeler A

**Descriptive Statistics**

| N | Mean | StDev | Median | Min | Max | Skewness | Kurtosis |
|---|---|---|---|---|---|---|---|
| 50 | 58,4294 | 58,8403 | 31,975 | 0,668593 | 273,772 | 1,62865 | 2,88675 |

**Goodness of Fit Test**

| Distribution | AD | P |
|---|---|---|
| Normal | 2,746 | <0.005 |
| Box-Cox Transformation | 0,269 | 0,668 |
| Lognormal | 0,515 | 0,183 |
| Exponential | 0,305 | 0,812 |
| Weibull | 0,3 | >0.250 |
| Smallest Extreme Value | 4,496 | <0.010 |
| Largest Extreme Value | 1,64 | <0.010 |
| Gamma | 0,311 | >0.250 |
| Logistic | 2,216 | <0.005 |
| Loglogistic | 0,419 | >0.250 |
| Johnson Transformation | 0,217 | 0,833 |

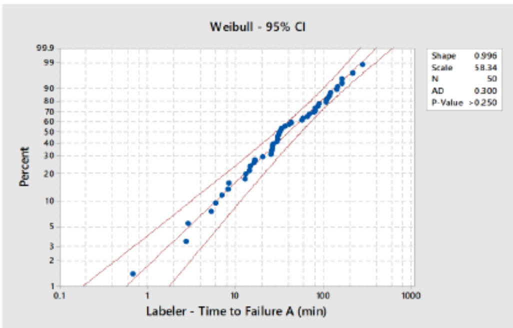



*Figure 106 - Probability distribution Identification for TTFs and TTRs*

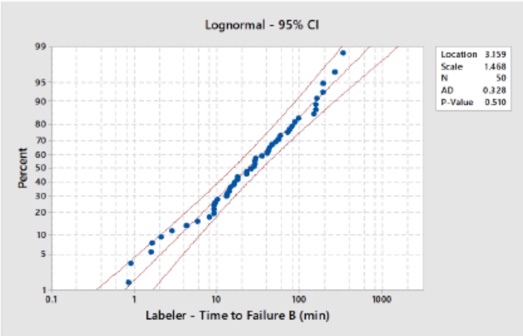
## Distribution ID Plot: TTF (min) Labeler B

### Descriptive Statistics

| N | Mean | StDev | Median | Min | Max | Skewness | Kurtosis |
|---|---|---|---|---|---|---|---|
| 50 | 54,8577 | 71,7573 | 26,7837 | 0,843529 | 326,587 | 2,07525 | 4,30451 |

### Goodness of Fit Test

| Distribution | AD | P |
|---|---|---|
| Normal | 4,966 | <0.005 |
| Box-Cox Transformation | 0,328 | 0,51 |
| Lognormal | 0,328 | 0,51 |
| Exponential | 2,175 | 0,006 |
| Weibull | 0,372 | >0.250 |
| Smallest Extreme Value | 6,602 | <0.010 |
| Largest Extreme Value | 3,168 | <0.010 |
| Gamma | 0,575 | 0,172 |
| Logistic | 3,835 | <0.005 |
| Loglogistic | 0,273 | >0.250 |
| Johnson Transformation | 0,2 | 0,878 |



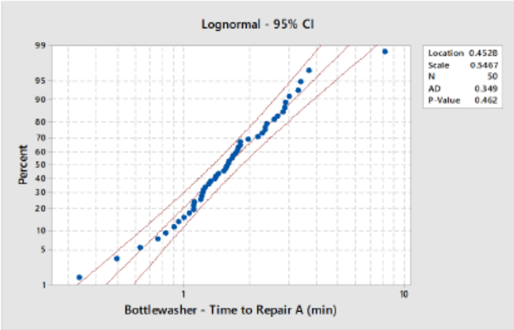

## Distribution ID Plot: TTR (min) Bottle-washer A

### Descriptive Statistics

| N | Mean | StDev | Median | Min | Max | Skewness | Kurtosis |
|---|---|---|---|---|---|---|---|
| 50 | 1,82842 | 1,19515 | 1,56748 | 0,334614 | 8,13752 | 3,17325 | 15,2166 |

### Goodness of Fit Test

| Distribution | AD | P |
|---|---|---|
| Normal | 2,576 | <0.005 |
| Box-Cox Transformation | 0,349 | 0,462 |
| Lognormal | 0,349 | 0,462 |
| Exponential | 6 | <0.003 |
| Weibull | 1,399 | <0.010 |
| Smallest Extreme Value | 7,411 | <0.010 |
| Largest Extreme Value | 0,393 | >0.250 |
| Gamma | 0,572 | 0,16 |
| Logistic | 1,039 | <0.005 |
| Loglogistic | 0,188 | >0.250 |
| Johnson Transformation | 0,153 | 0,956 |



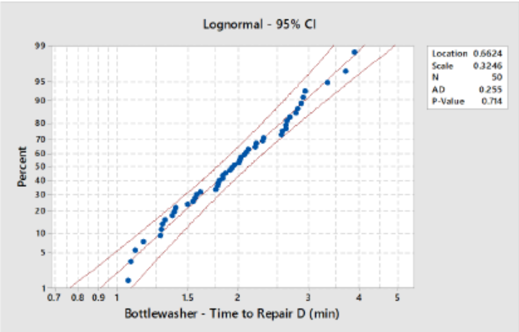

## Distribution ID Plot: TTR (min) Bottle-washer D

### Descriptive Statistics

| N | Mean | StDev | Median | Min | Max | Skewness | Kurtosis |
|---|---|---|---|---|---|---|---|
| 50 | 2,04187 | 0,669946 | 1,93817 | 1,06274 | 3,88228 | 0,720388 | 0,224218 |

### Goodness of Fit Test

| Distribution | AD | P |
|---|---|---|
| Normal | 0,624 | 0,098 |
| Box-Cox Transformation | 0,255 | 0,714 |
| Lognormal | 0,255 | 0,714 |
| Exponential | 10,824 | <0.003 |
| Weibull | 0,6 | 0,118 |
| Smallest Extreme Value | 1,848 | <0.010 |
| Largest Extreme Value | 0,292 | >0.250 |
| Gamma | 0,295 | >0.250 |
| Logistic | 0,55 | 0,111 |
| Loglogistic | 0,326 | >0.250 |
| Johnson Transformation | 0,208 | 0,859 |




## Distribution ID Plot: TTR (min) Bottle-washer E

### Descriptive Statistics

| N | Mean | StDev | Median | Min | Max | Skewness | Kurtosis |
|---|---|---|---|---|---|---|---|
| 50 | 2,00564 | 0,832743 | 2,05983 | 0,354669 | 3,54556 | -0,17904 | -0,82592 |

### Goodness of Fit Test

| Distribution | AD | P |
|---|---|---|
| Normal | 0,35 | 0,46 |
| Box-Cox Transformation | 0,35 | 0,46 |
| Lognormal | 1,695 | <0.005 |
| Exponential | 7,334 | <0.003 |
| Weibull | 0,537 | 0,178 |
| Smallest Extreme Value | 0,417 | >0.250 |
| Largest Extreme Value | 0,942 | 0,016 |
| Gamma | 1,055 | 0,01 |
| Logistic | 0,401 | >0.250 |
| Loglogistic | 1,234 | <0.005 |
| Johnson Transformation | 0,35 | 0,46 |

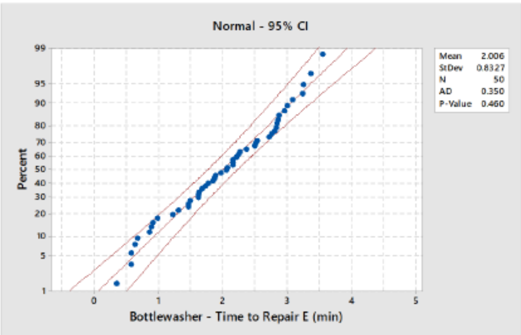



*Figure 107 - Probability distribution Identification for TTFs and TTRs*

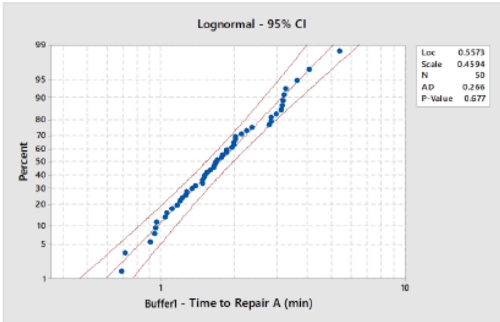
## Distribution ID Plot: TTR (min) Buffer 1 A

**Descriptive Statistics**

| N | Mean | StDev | Median | Min | Max | Skewness | Kurtosis |
|---|---|---|---|---|---|---|---|
| 50 | 1,93973 | 0,948443 | 1,6799 | 0,689417 | 5,36615 | 1,33631 | 2,25691 |

**Goodness of Fit Test**

| Distribution | AD | P |
|---|---|---|
| Normal | 1,53 | <0.005 |
| Box-Cox Transformation | 0,266 | 0,677 |
| Lognormal | 0,266 | 0,677 |
| Exponential | 7,236 | <0.003 |
| Weibull | 0,95 | 0,016 |
| Smallest Extreme Value | 3,419 | <0.010 |
| Largest Extreme Value | 0,461 | >0.250 |
| Gamma | 0,515 | 0,21 |
| Logistic | 1,177 | <0.005 |
| Loglogistic | 0,306 | >0.250 |
| Johnson Transformation | 0,212 | 0,849 |



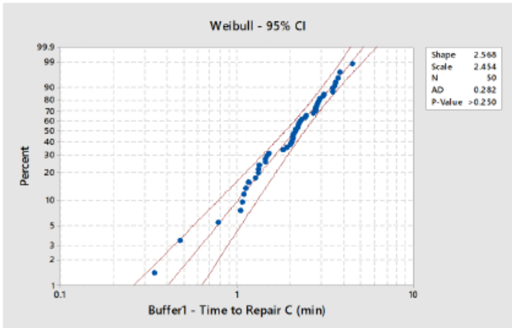

## Distribution ID Plot: TTR (min) Buffer 1 C

**Descriptive Statistics**

| N | Mean | StDev | Median | Min | Max | Skewness | Kurtosis |
|---|---|---|---|---|---|---|---|
| 50 | 2,17932 | 0,923037 | 2,11856 | 0,33901 | 4,46032 | 0,216134 | -0,37403 |

**Goodness of Fit Test**

| Distribution | AD | P |
|---|---|---|
| Normal | 0,323 | 0,517 |
| Box-Cox Transformation | 0,368 | 0,417 |
| Lognormal | 1,004 | 0,011 |
| Exponential | 7,415 | <0.003 |
| Weibull | 0,282 | >0.250 |
| Smallest Extreme Value | 0,95 | 0,016 |
| Largest Extreme Value | 0,507 | 0,206 |
| Gamma | 0,529 | 0,198 |
| Logistic | 0,385 | >0.250 |
| Loglogistic | 0,71 | 0,038 |
| Johnson Transformation | 0,323 | 0,517 |



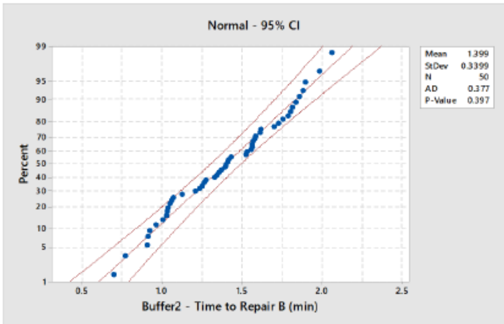

## Distribution ID Plot: TTR (min) Buffer 2 B

**Descriptive Statistics**

| N | Mean | StDev | Median | Min | Max | Skewness | Kurtosis |
|---|---|---|---|---|---|---|---|
| 50 | 1,39873 | 0,339884 | 1,40302 | 0,698326 | 2,05819 | -0,08546 | -0,85424 |

**Goodness of Fit Test**

| Distribution | AD | P |
|---|---|---|
| Normal | 0,377 | 0,397 |
| Box-Cox Transformation | 0,377 | 0,397 |
| Lognormal | 0,667 | 0,077 |
| Exponential | 13,142 | <0.003 |
| Weibull | 0,344 | >0.250 |
| Smallest Extreme Value | 0,52 | 0,194 |
| Largest Extreme Value | 0,768 | 0,043 |
| Gamma | 0,542 | 0,183 |
| Logistic | 0,47 | 0,199 |
| Loglogistic | 0,674 | 0,046 |
| Johnson Transformation | 0,377 | 0,397 |




## Distribution ID Plot: TTR (min) Labeler A

**Descriptive Statistics**

| N | Mean | StDev | Median | Min | Max | Skewness | Kurtosis |
|---|---|---|---|---|---|---|---|
| 50 | 2,16049 | 1,21494 | 1,85387 | 0,592028 | 6,48485 | 1,42223 | 2,41991 |

**Goodness of Fit Test**

| Distribution | AD | P |
|---|---|---|
| Normal | 1,815 | <0.005 |
| Box-Cox Transformation | 0,393 | 0,363 |
| Lognormal | 0,393 | 0,363 |
| Exponential | 5,744 | <0.003 |
| Weibull | 0,912 | 0,019 |
| Smallest Extreme Value | 3,889 | <0.010 |
| Largest Extreme Value | 0,48 | 0,232 |
| Gamma | 0,515 | 0,212 |
| Logistic | 1,208 | <0.005 |
| Loglogistic | 0,315 | >0.250 |
| Johnson Transformation | 0,307 | 0,551 |

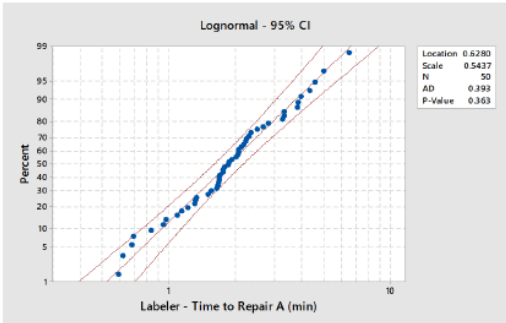



*Figure 108 - Probability distribution Identification for TTFs and TTRs*

**Distribution ID Plot: TTR (min) Labeler B**

**Descriptive Statistics**

| N | Mean | StDev | Median | Min | Max | Skewness | Kurtosis |
|---|---|---|---|---|---|---|---|
| 50 | 3,04398 | 1,14849 | 3,12173 | 0,992272 | 6,77914 | 0,591773 | 0,98762 |

**Goodness of Fit Test**

| Distribution | AD | P |
|---|---|---|
| Normal | 0,274 | 0,649 |
| Box-Cox Transformation | 0,282 | 0,624 |
| Lognormal | 0,649 | 0,085 |
| Exponential | 9,174 | <0.003 |
| Weibull | 0,265 | >0.250 |
| Smallest Extreme Value | 1,667 | <0.010 |
| Largest Extreme Value | 0,498 | 0,215 |
| Gamma | 0,374 | >0.250 |
| Logistic | 0,255 | >0.250 |
| Loglogistic | 0,571 | 0,094 |
| Johnson Transformation | 0,274 | 0,649 |

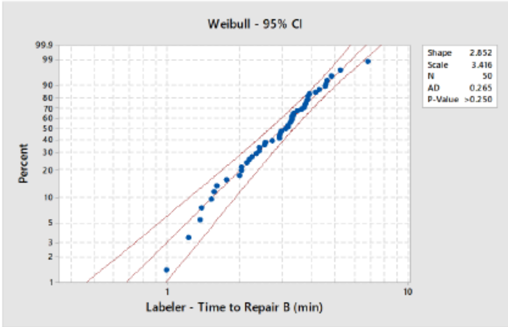



*Figure 109 - Probability distribution Identification for TTFs and TTRs*

# Bibliography

[1] Aamen, W., AlKahtani, M., Mohammed, M.K., Abdulhameed, O. and El-Tamimi, A.M. (2018) *Investigation of the effect of buffer storage capacity and repair rate on production line effciency*, Journal of King Saud University-Engineering Sciences, Vol.30, pp. 243-249.

[2] Allahverdi, A. and Soroush, H.M. (2008) *The significance of reducing setup times/setup costs*, European Journal of Operational Research, 187, 978984.

[3] Anderson, T.W. and Darling, D.A. (1952) *Asymptotic theory of certain "goodness-of-fit" criteria based on stochastic processes*, Annals of Mathematical Statistics 23: 193-212.

[4] The AnyLogic Company (2018) *AnyLogic Help*, available from: http://www.anylogic.com/anylogic/help/"

[5] Battini D., Manzini R., Persona A. and Regattieri A., (2006) *TPM Approach and new buffer design paradigm in plant layout*, 12th ISSAT International Conference on Reliability and Quality in design. Chicago, August 3-5, 2006.

[6] Battini, D., Persona, A., Regattieri, A. (2009) *Buffer size design linked to reliability performance: a simulative study*, Computers & Industrial Engineering, Vol. 56, No. 4, pp. 1633–1641.

[7] Battini, D., Faccio, M., Persona, A., Regattieri, A. (2013) *Buffer design for availability: a new simulative study in case of infant and random failures*, Int. J. Services and Operations Management, Vol. 14, No. 2, pp.157–174.

[8] Bazan, P. and German, R. (2012) *Hybrid Simulation of Renewable Energy Generation and Storage Grids*, Proceedings of the 2012 Winter Simulation Conference.

[9] Borshchev A. and Filippov, A. (2004) *From System Dynamics and Discrete Event to Practical Agent Based Modeling: Reasons, Techniques, Tools,* The 22nd International Conference of the System Dynamics Society, July 25-29, 2004, Oxford, England.

[10] Braglia, M., Frosolini, M., and Zammori, F. (2009) *Overall equipment effectiveness of a manufacturing line (OEEML): an integrated approach to assess systems performance*, Journal of Manufacturing Technology Management, 20(1), 8-29.

[11] Braglia, M., Carmignati, G., Bertolini, M., *An FMECA-based approach to process analysis*, Int. J. Process Management and Benchmarking, Vol. 1, No. 2, 2006.

[12] Buzacott, J.A. and Hanin, L.E. (1978) *Automatic transfer lines with buffer stocks*, International Journal of Production Research, 5(3): 183-200.

[13] Chung, C.A. (2004) *Simulation Modeling Handbook: A practical Approach*, CRC Press LLc.

[14] Dallery, Y. and Gershwin, S.B. (1992) *Manufacturing flow line systems: a review of models and analytical results*, vol. 12, pp. 3-94, 1992.

[15] DeRon, A.J. and Rooda, J.E. (2006) *OEE and equipment effectiveness: an evaluation*, International Journal of Production Research, Taylor.

[16] Drury, H.B. (1918) *Economy Scientic Management*, seconded., rev., Columbia University Press, New York, NY, 1918, p. 125126.

[17] Du, J. and Leung, J.Y.T. (1993a) *Minimizing Mean Flow Time in Two-Machine Open Shops and Flow Shops*, Journal of Algorithms, Vol. 14, pp. 24-44.

[18] Fabricky, W.J. and Blanchard, B.S. (1991) *Life-Cycle Costs and Economic Analysis*, PrenticeHall.

[19] Flynn, B.B. (1987) *The effects of setup time on output capacity in cellular manufacturing.* International Journal of Production Research, 25, 17611772.

[20] Grygoryev, I. (2014) *AnyLogic in Three Days*, available from: https://www.anylogic.com/resources/books/free-simulation-book-and-modeling-tutorials/

[21] Hecker, F., Hussein, W. and T. Becker (2010*) Analysis and optimization of a bakery production line using ARENA*, International Journal of Simulation Modeling, vol.9, no. 3, pp. 108-216, Dec. 2010.

[22] Heshmat, M., El-Sharief, M.A. and El-Sebaie, M.G. (2013) *Simulation Modeling of Automatic Production Lines with Intermediate Buffers*, International Journal of Scientific Engineering Research, vol. 4(7), Jul. 2013, pp. 2528-2535.

[23] Hosseinpour, F. and Hajihosseini, H. (2009) *Importance of Simulation in Manufacturing, World Academy of Science*, Engineering and Technology International Journal of Economics and Management Engineering Vol: 3, No: 3, 2009.

[24] Huang, S.H. (2003) *Manufacturing productivity improvement using effectiveness metrics and simulation analysis*, International Journal of Production Research, 41 (3), 513-527.

[25] Jeong, K-Y. and Phillips, D.T. (2001) *Operational effciency and effectiveness measurement*, International Journal of Operations Production Management, Vol. 21 No.11, pp. 1404-1416.

[26] Jonsson, P. and Lesshammar, M. (1999) *Evaluation and improvement of manufacturing performance measurement systems-the role of OEE*, International Journal of Operations and Production Management 1999, 19, 55-78.

[27] Kelton, W.D., Sadowski, D.T. and Sturrock, R.P. (2007) *Simulation with ARENA*, Fourth Edi. McGraw-Hill NewYork.

[28] Krajewski, L.J., King, B.E., Ritzman, L.P. and Wond, D.S. (1987) *Kanban, MRP and shaping the manufacturing environment*, Management Science, 33, 3957.

[29] Lageweg, B.J., Lawler, E.L., Lenstra J.K. and RinnooyKan, A.H.G. (1982) *Computer Aided Complexity Classification of Combinatorial Problems*, Communications of the ACM, Vol. 25, pp 817-822.

[30] Land A.H. and Doig, A.G. (1960) *An Automatic Method of Solving Discrete Programming Problems*, *Econometrica*, Vol. 28, No.3 (Jul.,1960), pp. 497-520.

[31] Law, A.M. and McComas, M.G. (1997) *Simulation of manufacturing systems*, Proceeding of the 1997 winter simulation.

[32] Law, A.M. and Kelton, W.D. (1991) *Simulation Modeling and Analysis*, McGraw-Hill, NewYork.

[33] Levner, E.M. (1969) *Optimal Planning of Parts Machining on a Number of Machines,* Automation and Remote Control, Vol. 12, pp. 1972-1981.

[34] Molderink, A., Bosman, M.G.C., Bakker, V., Hurink, J.L. and Smit, G.J.M. (2009) *Simulating the Effect on the Energy Effciency of Smart Grid Technologies*, Proceedings of the 2009 Winter Simulation Conference.

[35] Mourtzis D. and Doukas M., (2012) *Decentralized Manufacturing Systems Review: Challenges and Outlook*, Logistics Research, Springer. 1865-0368.

[36] Mussellman, K.J. (1992) *Conducting a Successful Simulation Project*, Proceedings of the 1992 Winter Simulation Conference (Ed.: Crain, Wilson, Swain, and Goldsman) Arlington, Virginia, pp. 115-121.

[37] Nachiappan, R.M. and Anantharam, N.(2006) *Evaluation of overall line effectiveness (OLE) in a continuous product line manufacturing system*, Journal of Manufacturing Technology Management, 17(7), 987-1008.

[38] Nakajima, S. (1988) *Introduction to TPM: total productive maintenance*, Cambridge, MA: Productivity Press Inc.

[39] Nord C., Pettersson B. and Johansson B. (1997) *TPM: Total Productive Maintenance in the Volvo company*, Idrottens Graska I Goteborg AB, Molnlycke.

[40] Pandey, D.S. and Raut, N. (2016) *Implementing TPM by doing RCA*, International journal of advanced research in science, engineering and technology, vol.3, no. 2, 2016.

[41] Papadopoulos H.T. (1996) *Queuing theory in manufacturing systems analysis and design: A classification of models for production and transfer lines*, European Journal of Operational Research, vol. 92, pp. 1-27, 1996.

[42] Papadopuolos H.T., M.I. Vidalis (2001), *A heuristic algorithm for the buffer allocation in unreliable unbalaced production lines*, Computer & Industrial Engineering 41 (2001) 261-277.

[43] Prombanpong S., Kaewyu J., Thanadulthaveedech N. and Matwangsang M. (2013) *A Buffer Design for Mitigation Downtime Effect in an Automated Transfer Line*, Internation Journal of Innovation, Management and Technology, Vol. 4, No. 1, February 2013.

[44] Robinson, C. (2004) *Calculating line or process OEE*, Maintenace Technology, available at: www.mt-online.com/newarticles2/06-94mm.cfm

[45] Rossetti, M.D. (2015) *Simulation Modeling and Arena*, Wiley Publishing.

[46] Sadowski, R.P. (1989) *The Simulation Process: Avoiding the Problems and Pitfalls*, Proceedings of the 1989 Winter Simulation Conference (Ed.: Mac Nair, Musselman, and Heldelberger) Washington, D.C., pp. 72-79.

[47] Seraj, Y. (2008) *A Simulation Study To Increase The Capacity Of A Rusk Production Line*, Department of Industrial Engineering, vol.5, no.9, pp. 1395-1404, 2008.

[48] Smith, W.E. (1956) *Various Optimizers for Single Stage Production*, Naval Research Logistics Quarterly, Vol.3, pp.59-66.

[49] Sophie Weiss, Justus Arne Arne Schwarz & Raik Stolletz (2018): *The buffer allocation problem in production lines: Formulations, solution methods, and instances*, IISE Transactions.

[50] Trovinger, S.C. and Bohn, R.E. (2005) *Setup time reduction for electronics assembly: Combining simple (SMED) and IT-based methods*, Production and Operations Management, 14, 205217.

[51] Ulgen, O.M. (1991) *Proper Management Techniques are Keys to a Successful Project*, Industrial Engineering, pp. 37-41.

[52] Weimer, C.W., Miller, J.O. and Hill, R.R. (2016) *Agent- Based Modeling: an Introduction and Primer,* Proceedings of the 2016 Winter Simulation Conference.

[53] Zennaro I., Battini D., Sgarbossa F., Persona A. and De Marchi R. (2018) *Microdowntime: Data collection, analysis and impact on OEE in bottling lines the San Benedetto case study*, International Journal of Quality Reliability Management, Vol.35 Issue: 4, pp. 965-995.

[54] Zhang M., Matta A., Pedrielli G. (2016) *Discrete optimization: workstation and buffer allocation problem in manufactoring flow lines*, Proceedings of the 2016 Winter Simulation Conference.