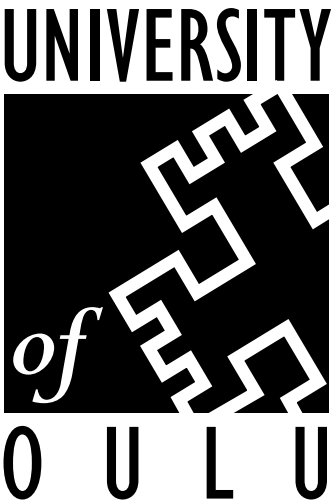

CONTROL ENGINEERING LABORATORY

PRODUCTION OPTIMIZATION ON PCB ASSEMBLY LINES USING DISCRETE-EVENT SIMULATION

Sébastien Gebus, Olivier Martin, Alexandre Soulas,
Esko Juuso

Report A No 24, May 2004

University of Oulu
Department of Process and Environmental Engineering
Control Engineering Laboratory
Report A No 24, May 2004

PRODUCTION OPTIMIZATION ON PCB ASSEMBLY LINES USING DISCRETE-EVENT SIMULATION

Sébastien Gebus, Olivier Martin, Alexandre Soulas, Esko Juuso

Control Engineering Laboratory,
Department of Process and Environmental Engineering,
University of Oulu
Linnanmaa, FIN-90014 University of Oulu, Finland

Abstract: This report describes how discrete-event simulation can be used in production optimisation of electronics assembly lines. Currently, many decisions concerning production are based on workers experience. However, understanding of the parameters influencing production is a challenging task, especially for lines with a wide variety of products. Operation could be improved by analysing bottlenecks and their impact on overall line capacity. Nowadays there are various new tools to extend traditional quality and process-time techniques such as flowcharts or spreadsheets to manage production.

This work focused on how discrete-event simulation could be used in comparing production alternatives to improve production in electronics manufacturing by providing a better understanding of the production environment. It can be used in a straightforward way to test different scenarios for improvement or it can provide information for designing new production facilities. This ability to simulate a real system's behaviour according to some predefined parameters can also be used as an evaluation tool for new control methods. The present simulator can be used as a platform in comparing optimisation and scheduling approaches before implementation.

An experimental framework for electronics manufacturing plant is described, and modelling choices are focused on comparisons of scheduling policies. PKC Group plant producing control cards for the telecommunication was used for the selection and testing of modelling alternatives. Good results were obtained for basic scheduling policies, allowing comparison between different options. The model could now be used to develop intelligent optimisation methods through links with external software. Among others, genetic algorithms have been considered as a possible choice. Any heuristically based method would benefit greatly of the ability that discrete-event simulation has to mimic real processes.

Keywords: production optimisation, discrete-event simulation, scheduling, printed circuit board.

ISBN 951-42-7372-9
ISSN 1238-9390
ISBN 951-42-7517-9 (PDF)

University of Oulu
Control Engineering Laboratory
PL 4300
FIN-90014 University of Oulu

Table of Contents

1. INTRODUCTION

Modelling and simulation (M&S) is a problem-solving methodology for analysing complex systems. [Schriber, 1987] defines simulation as "the modelling of a process that mimics the response of the actual system to events that take place over time". In addition to this, [Pedgen et al., 1995] defines simulation as "the process of designing a model of a real system and conducting experiments with this model for the purpose of understanding the behaviour of the system and/or evaluating various strategies for the operating system". The model can be used to

- analyse current operations and identify problem area, e.g. bottlenecks,
- test various scenarios for improvement,
- design new manufacturing systems.

Simulation models allow to test potential changes in an existing system without disturbing it or to evaluate the design of a new system without building it [Law et al., 1998] [Scriber et al., 1997]. Simulation early in the design cycle is important, because the cost to repair mistakes increases dramatically the later in the product life cycle the error is detected. This methodology also allows comparing new concepts, equipments or scenarios before purchasing.

For some purpose, simulations are better than the analysis of real data. With real data, it is never possible to perfectly know the real-world process that caused a particular measured situation, because of the too complex interactions inherent in large systems. In a simulation, the analyst controls all the factors making up the data and can manipulate them systematically to see directly how specific problems and assumptions affect the analysis. Because simulation software keeps track of statistics about model elements, performance can be evaluated by analysing the model data.

Business processes such as supply chain, customer service and product development are nowadays too complex and dynamic to be understood and analysed with spreadsheets or flowchart techniques. The interactions of resources with processes, products and services result in a very large number of scenarios and outcomes that are impossible to understand and evaluate without the help of a computer simulation model. Old techniques are adequate for answering "*what*" questions, but not for "*how*", "*when*" and "*what if*" questions.

Average costs, quality and process-time techniques are not anymore sufficient to do process design and management. Designing and driving processes based on these average values produce major errors. This can be described with a very simple example: let's imagine we need a robot that will carry an average load of 20 kg (between 10 and 30 kg). The averaging technique will lead us to buy or design a robot able to carry a 22 kg load with security margins), but an average load of 20 kilos means that sometimes the load will be more than 22 kg, and the robot will not be adequate in these cases, i.e. money and time is lost. This example is simple and we all have a feeling that using averaging techniques would not be efficient, but in complex situations, this is not so obvious

although major errors may result from using only averaging techniques. With process simulation techniques, it is possible to test pessimistic scenarios (*what* will happen *if* two machines are down at the same time), to test all kind of *what-if* scenarios, to visualise and understand the processes' flows.

Discrete-event simulation has already been used in many ways to get knowledge about the behaviour of a system under certain conditions. [Estremadoyro et al., 1997] for example presents with its Electronics Manufacturing Simulator a solution for rapid modelling of assembly lines. Companies wanting to purchase new production systems can check the consistency of their choices before agreeing for the investment. Through TAKT time calculation, companies can avoid the purchase of "bottleneck machines". Such a tool however is limited to production lines with simple specifications but not in a problem-solving approach.

A more common use of discrete-event simulation is to replace static analysis in capacity planning [Andersson M et al., 1998] or scheduling of resources. For [Czarmecki et al., 1997] "capacity planning is the process of determining the tooling, personnel, and equipment resources that are required to meet customer demand. Scheduling is the time-sequenced allocation of these resources". Main target for companies using discrete-event simulation is usually to improve the operational control of their manufacturing system and to be able to confirm their production rates [Williams et al., 1997]. In [Harmonosky et al., 1999], simulation is used to shorten the ramp-up period when moving from prototype production type towards mass production. Confirming a production rate is not only a necessity when demand is high, but also very often it is even more important in case of over-capacity of the production tool. A cost-efficient production means that resources should not stay idle. Required increase in flexibility towards external demand can also be achieved through discrete-event simulation [Jackson et al., 1997]. In the same line, [Savsar, 1997] uses discrete-event simulation to analyse the capability of a pull-push system to achieve just-in-time production on an electronics assembly line.

When the production system is as linear as it is in electronics industry, scheduling policies become a very important tool for process optimisation. Machines are standard and interchangeable. They can do each other's work but due to physical limitations, they cannot usually be duplicated in parallel. On this topic, [Savsar et al., 1999] analyses the effect of scheduling policies on serial duplicated machines in order to balance the production rate. An intelligent system for short term scheduling on the basis of preference criteria is presented in [Juuso and Jagdev, 1999].

This report consists of two parts. The first part describes principles of using discrete-event simulation models for representing production systems, contents of a general simulation project and some guidelines for selecting simulation software. The second part presents a case study in an electronics manufacturing process including experimental procedure, modelling and some comparisons of scheduling policies. The study is a part of the Intele (Intelligent methods in electronics manufacturing) project.

2. DISCRETE-EVENT SIMULATION

Discrete-event modelling and simulation is needed for comparing alternatives in analysing, testing and design of production systems in presence of randomness. Flexibility of the modelling approach is important, especially for comparisons of intelligent scheduling policies.

2.1 Production systems

A production system is a system that transforms raw materials into something more elaborate by adding value to the product. Such a system should however not be reduced only to its physical part that cannot work on its own. It is not possible to talk about a production system without a decision subsystem that controls and manages the flow of information and supervises the physical part. Figure 1 gives a simplified representation of a production system.

For both material and information, the control of the flows has become a necessity for companies that aim at improving their competitiveness and their productivity. It can be achieved for example through reduction of delays or greater flexibility towards an increasingly evolving and complex production.

Many parameters can have an influence on the flows and it is therefore difficult to control them. This is of course true for the physical system (factory layout, number of machines, automation level, flexibility of each machine, amount of operators, buffers…) but also for the decision system (strategy of resources assignment, priority rules, maintenance policies, ordonancement…). However, it is even more important that decisions, which are taken at almost every level of the company, also influence these parameters. Through their daily choices, people can directly have an effect on the production without even

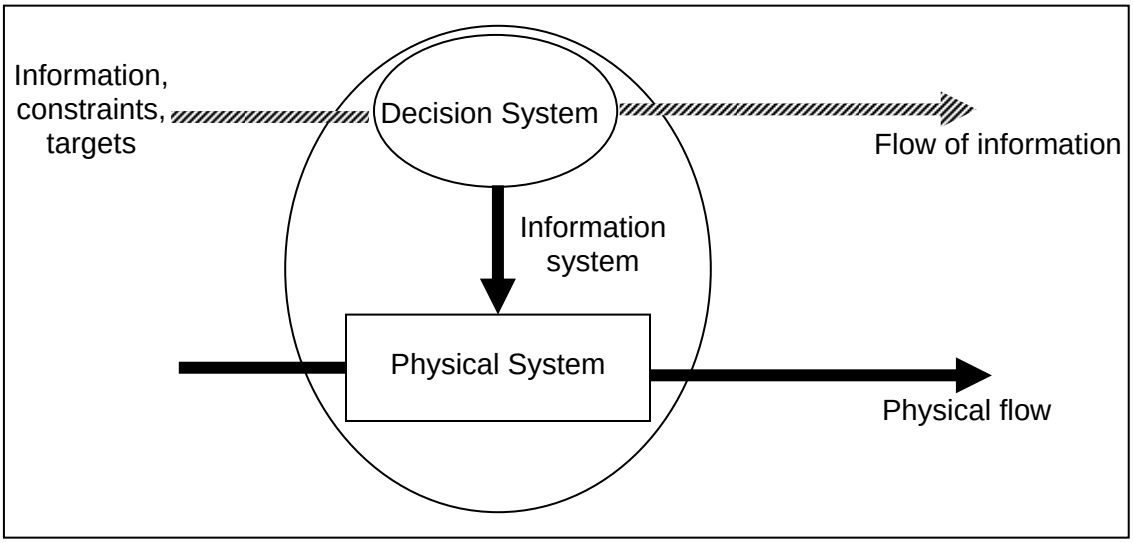

Figure 1. Simplified representation of a production system.

noticing it. This underlines the need for a decision-helping tool aimed at improving technology, organisation and production management in the company.

2.2 Basic principles of simulation

The first aim of simulation is to experiment new methods on a computer model because simulation is cheaper and faster than real experiments, and sometimes it is simply impossible to do the same on a real process. Another advantage is that a new method can be tested and validated without disrupting the real system. Simulation can also be used to create models of non-existent processes when designing new systems or redesigning and reorganizing existing ones.

Historically, discrete event simulation was a part of operational research. It was used to answer "what if…" questions in order to evaluate the performance of a system. However, it should be kept in mind that this role of performance evaluation doesn't guarantee an optimal solution. Commonly, simulation is used to compare solutions under some parameters and hypothesis. It is up to the user to define if randomness has an influence or not on the result of the simulation. Results obtained by simulations are therefore only observations and a statistical and probabilistic approach is needed to interpret these results.

Other modelling methods

Discrete event simulation is not the only solution and certainly not always the best solution. As modelling can be a time-consuming activity, simulation should be avoided if possible. Other existing methods based on the queue theory or Petri nets have the advantages to be analytical, meaning that they will provide exact instead of approximate results. However, in many cases, these analytical methods are limited to simple systems.

Definition of simulation

Simulation means imitating the behaviour of a dynamic system through time in order to solve a given problem. The model is a simplified representation of this system. It is also created and valid only for this given problem because it is not possible to create a model of a system in all its dimensions.

During the simulation, observations are only estimates, i.e. results can only be given with a confidence interval.

2.3 Discrete-event models to represent production systems

2.3.1 Model Parameters

Attributes

Commonly, models are made of objects or entities that are linked through relationships. Objects are characterized by one or several attributes in order to differentiate them from each other. There are two kinds of object-related attributes:

- Attributes with a fixed value are used to define the nature of an object, e.g. PCB type and number of machines,
- Variable attributes change over time, e.g. number of PCBs in a queue.

In the same way, relationships will also be described by attributes, like for example the processing times for a PCB on each machine. These attributes also have a fixed value.

States

The state of an object at a given time is the value of all its attributes at this given time. In the same way, the sate of the system at a given time will be the state of all the objects at that time. The changes from one state to another will therefore make the dynamic of the model. It is also a criterion for choosing the modelling technique.

2.3.2 Discrete-event and continuous models

Continuous models

A model is called continuous when changes of states happen continuously over time. In such a case, the state of the system can be expressed with a following equation

$$X = f(t, \lambda, X_0),$$

where t is time, λ includes parameters of the model, and X_0 defines initial conditions. This kind of models has a problem of computability because computers cannot work in a continuous way. Continuity can therefore only be kept at a mathematical level but not in the simulation. However, having to use sampled continuous data doesn't mean that the model is discrete.

Discrete-event models

A model is called discrete when changes between states happen only at specific times and in a discontinuous way. An event is what causes the change in state of the system. Attributes can usually only have a certain number of values, and states are countable. Resources affectation problems are a typical use for this kind of model. The activity is the time separating two events. Waiting can therefore also be considered as an activity, even if nothing is happening.

In a discrete model, the process will be a combination of a number of states of different objects, and a succession of activities related to those objects.

2.3.3 Types of discrete-event models

Different approaches are possible for creating a discrete-event model: models can be based on activities, events or processes.

From the point of view of the activity

All the activities must be identified and listed, and the only attribute linked to them is their duration. The real problem is then to find the chronology of those activities. This is done by identifying starting and ending conditions for each activity. In this case, running a simulation means running an event calendar that will start or end the activities.

Major problem with this approach is the calculating power needed to run the simulation since all the conditions have to be checked at every step of the event calendar.

From the point of view of the event

All the events, that can happen, and their effect on the system must be identified. The model consists of algorithms describing the changes of states of the system related to each event. Like in the previous approach, an event calendar has to be used.

From the point of view of the process

Very similar to the previous approach, the process point of view focuses on events rather than activities. A process will consist of several events and the whole system is described through the interactions between the different processes. Figure 2 shows an example of interactions between two processes.

Combined models

In many cases, e.g. manufacturing plants, a discrete model will be enough, whereas continuous models are more adapted to industries where a phenomenon has to be monitored, e.g. chemical industry and pulp and paper mills. As some processes need a combination of both approaches, discrete-event and continuous models are not exclusive and can be combined depending on the aim of the simulation. It is however good to know exactly what has to be simulated and what has to be included in the model in order to avoid an unnecessary combination of modelling techniques. Time management can become complicated with a model that has to be able to “jump” from one event to the next one, and at the same time simulate a specific phenomenon.

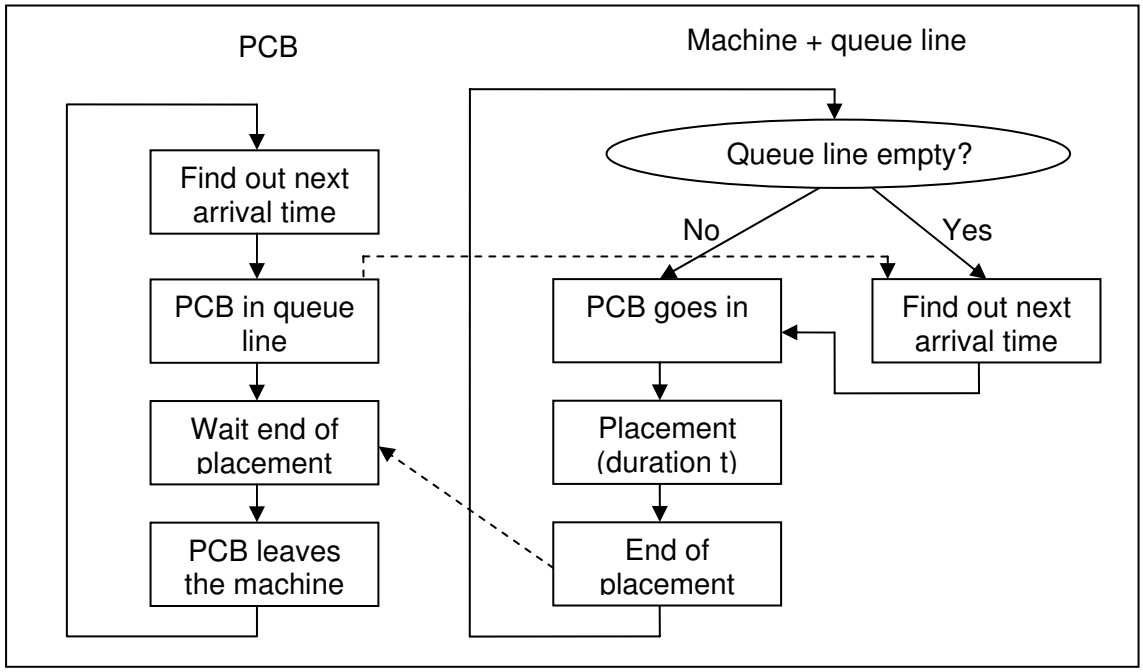

Figure 2. Interaction between two processes.

2.4 Randomness in models

Control over randomness is very important in discrete-event simulation. The accuracy and the repeatability of the simulation, and hence the quality and usability of the results, will depend on that. Two kinds of random variables can be used in simulation. Discrete random variables are commonly used when a choice has to be made between several possibilities. Routing problems for example are often solved using Bernoulli's law. On the other hand, continuous random variables are introduces when dealing with time-related problems, e.g. mean time between failures.

3. SIMULATION PROJECT

Simulation project is based on experimental framework and iterative stages of model construction.

3.1 Experimental framework

The experimental framework is a combination of circumstances under which the system is observed and tested. The framework consists of five parts: optimisation, input variables, initialisation, termination and results.

3.1.1 Optimisation functions

The optimisation functions are defined first. As they are a direct result of the targets of the simulation, they will define the content of the final statistical report. Examples of optimisation function are utilisation rate of machines or time spent by PCBs on the production line (maximum, minimum and average values).

3.1.2 Input variables

Input variables are given to the model but not controlled by it. Scheduling policies or the number of orders for a production line are examples of input variables. They should however not be confused with parameters of the model such as breakdowns or capacities of machines.

3.1.3 Initialisation

Initialisation defines the state of the model at the beginning of the simulation. Alternative initial states are:

- *Empty state.* All the servers, machines, queue lines, etc. are empty. This is the easiest way to simulate but the initial bias can be important.
- *Average state.* In this case, a pilot run is used to define the average state of the different elements of the model. Simulation can then be started with the system close to its stationary state. Initial bias is minimised, however such a method is not always easy to implement. Initial conditions often need to be integer values, whereas stationary state can be for example 0.25 PCBs in average in a certain queue line.
- *Real state.* This method is only used when trying to answer questions like “Can we produce 100 PCBs more before the end of the week?”, otherwise it is useless.

In many cases, initialisation can be arbitrary, but through the initial bias it will have an effect on the performance estimation. Result analysis should be made when the system reaches a stationary state and should not include the ramp-up period. Making long simulation runs in order to observe the system in its stationary state can do this.

3.1.4 Termination

Termination defines the ending conditions of the simulation. Some of those conditions are:

- Fixed simulation time,
- Predefined event,
- Predefined production, e.g. 100000 PCBs,
- Precision reached for an estimator, e.g. average or standard deviation.

More generally, a distinction can be made between terminating and non-terminating simulations. Terminating simulations start and end in a predefined state. In this case, there is no problem in terms of statistics. Independent and identically distributed observations are obtained by changing random generator for each simulation run. Non-terminating simulations start and end in states that are not known before hand. Long simulation runs are used in these cases in order to reach a stationary state of the system with enough precision in the estimation of the results.

3.1.5 Formalisation of results and interpretation

Statistical analysis of the results leads to certain problems and precision is not the least of them. Whether the aim of the simulation is to evaluate the performance of a solution, or to compare different alternatives, being able to calculate precision intervals for the results is a necessity. Furthermore, relevancy of the results is only guaranteed for a given experimental framework

3.2 Project stages

Simulation projects are iterative on various levels. The project starts with problem formulation, functional description of the system and definition of the experimental procedure (Figure 3). The model development is based on the collected data. Collected data may change the functional description of the system. Modelling may require additional data collection on a specific area and even change the functional description.

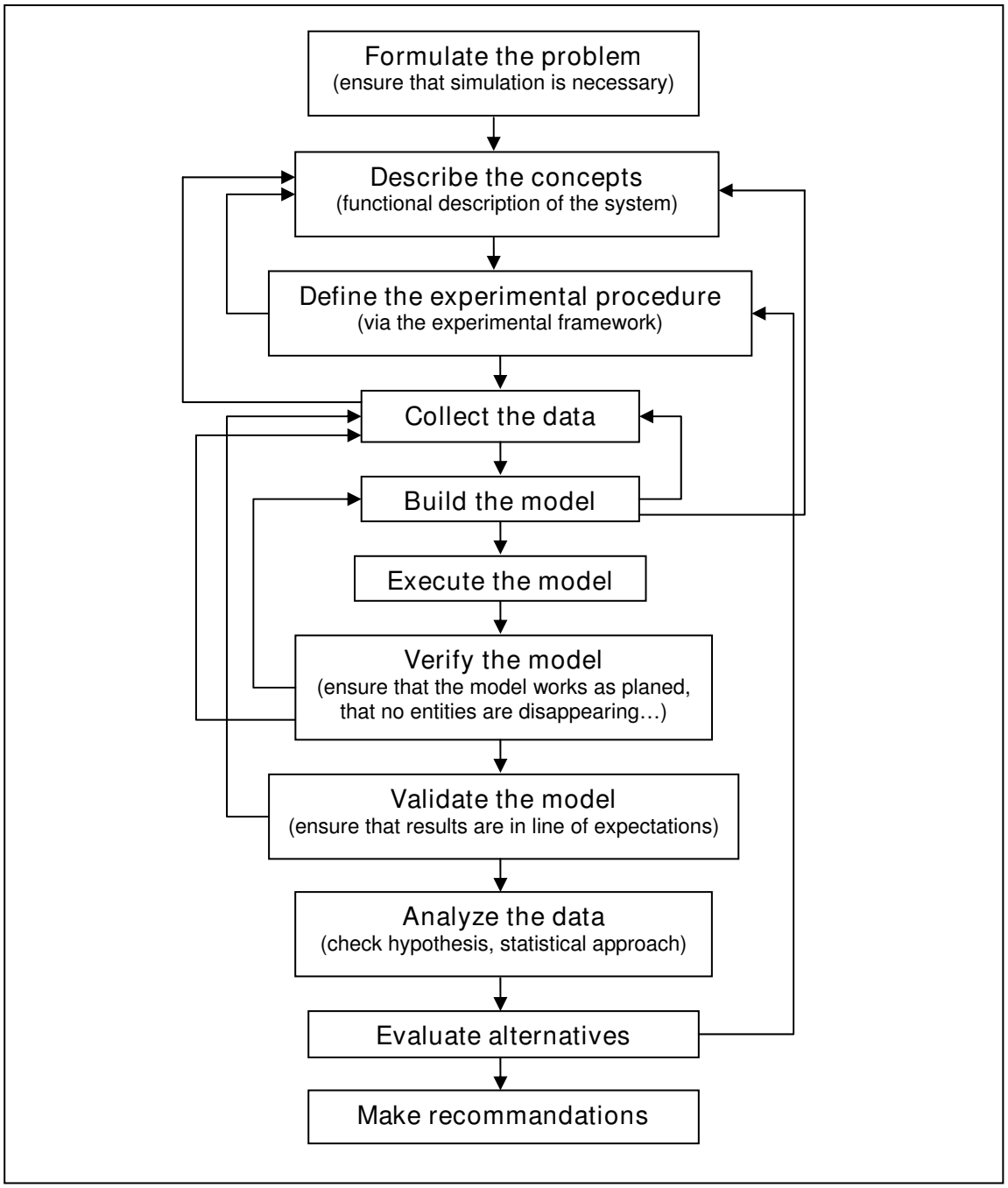

Figure 3. The stages of the simulation project.

Verification and validation of the model requires results obtained by executing the model. The verification ensures that the model works as planned. Modelling is updated if necessary. The validation ensures that the results are in line with expectations. It may require additional data collection.

Evaluation of the alternatives is based on statistical approach used for analysing the data. The evaluation stage may introduce changes to the experimental procedure. Final recommendations can be generated only when the whole sequence can be accepted (Figure 3).

3.3 Model construction

Communication is the first and the best method to get a model that will be as reliable as possible. System Analysis and Design Technic (SADT) or Integrated DEFinition language (IDEF) can be efficient tools to improve communication with system experts by establishing a common language based on a functional point of view. Even if the model is not always built on the same point of view, those methods help improving the understanding of the system. Otherwise, two main construction techniques can be considered.

3.3.1 Construction by increasing model complexity

Low level modelling is used in the beginning, and details are gradually added to increase the realism of the model. This modelling technique is often used when only limited knowledge is available about the process. A draft model must have enough details to get an overall picture of the process, but must be general enough to bring consensus between experts. Once it has been generated, it will be the base of the discussion with those experts.

This technique allows control of the complexity of the model. The difficulty however is to find the right level of details depending on the aim of the simulation

3.3.2 Construction by accumulation of submodels

This technique can be used when it is possible to identify separate and independent submodels. An example could be the model of a hospital. The different services can be modelled and validated separately before aggregating them.

Sub-model accumulation allows control of the complexity since the different parts are independent and can be left out of the final model if necessary. It is also a way to build big and detailed models. One difficulty however with this technique is the management of inputs and outputs and their interactions between the submodels.

4. HOW TO SELECT A SIMULATION SOFTWARE

It is very difficult to choose good software because of the vast amount of software available for discrete-event simulation. Reading some simulation buyer's guides makes the selection process even more complicated, because there are over 50 possibilities. Most of them could be a good choice; this is why a careful software selection decision can sometimes take more than six months! Application related special considerations are essential in choosing the software.

4.1 General considerations

Special requirements of the application must be taken into account first. Then comparison is based on several considerations: feasibility for different input sources, debugging possibilities, processing performance, alternative output possibilities, ease of use and price.

4.1.1 Know exactly what features are needed

Simulation package can include some specific tools, such as cranes or call-centres. Depending on the needs, many of those packages can be useless.

The "yes" and "no" answers.

Can this software model a conveyor? YES, but it doesn't mean that the software can model YOUR conveyors. Other parameters such as the speed of the conveyors or the weight of entities that are being carried could be important as well. Most of the available software provides more or less the same features. Not all of them however give the possibility to customize all the parameters for those features.

4.1.2 Input considerations

Importing/exporting a file

It is often required to import data to use in a simulation (amount of components available in stock, cycle-times depending of the entity to be processed, etc). Usually, the data have to be retrieved from large databases that are generated and updated electronically. Good simulation software allows statistical analysis of real data and it is therefore necessary to be able to import this data from the database. Manual input of all the data into the model would be a waste of time, and would likely be a cause of errors.

Debugger

Even simulation experts make mistakes or commit logical errors when building a model. Therefore, it is very important to have a good debugger for helping to find and correct errors in the following ways:

1. The simulation can be monitored *systematically*. This can be accomplished by running the simulation until a desired time, then displaying model information at that given time. Another possibility is to continue the simulation until a particular condition becomes active, and then display information.

2. Attention can be focused on a particular area of the simulation, or a particular entity. For example, every time an entity enters a specified area, the simulation will gather information, or every time a specified entity becomes active, the simulation will pause.

3. Values of selected model components can be observed. When the simulation has paused, the current value or status of variables, attributes, queues, resources, and counters can be observed.

4. The simulation can be suspended to view information, to reassign values, or to redirect entities.

4.1.3 Processing considerations

Speed

When large models are built, the software speed should not slow down to the point of slow motion.

Run flexibility

It should be possible to generate scenarios, e.g. the speed of a conveyor can range from 0.5 to 2 m/s, or batch a series of runs.

Statistical distributions

There are 12 statistical distributions, which are used in simulation. Most simulation software can generate random values using these 12 distributions.

Independent replications

Multiple replications using different sets of random numbers should be possible. Otherwise, the same results would occur repeatedly. Central Limit Theorem cannot be applied without independent replications.

Randomness control

A model must be able to simulate real world randomness. However, it must be possible to keep control over randomness for debugging purpose. If a mistake occurs in the model, it must be possible to run it again, with the same random values, so that the error will occur at the same time and at the same place.

Portability

This feature enables the software to be run on various classes of computer without any changes in the software.

4.1.4 Output considerations

Standardized reports

Examples of standardized output measures are the average number in queue, average time in queue, and throughput. The software can produce these and other values automatically, or upon request. They are also commonly used as criteria to compare different scenarios.

Business graphics

The software can have an ability to generate high quality bar charts, pie charts, and histograms, which can be used in presentations and reports.

Write to a file

Does the software allow data, events, or system variables to be written to a file whenever desired? This feature allows the analyst to later import the file into a spreadsheet or database program for further customised analysis or manipulation.

4.1.5 Environmental considerations

Ease of use and learning

This feature is important to the casual user, not so important to the frequent or continuous user. For research purpose, power of the software is probably much more important than ease of use. The quality of the available documentation can also be a factor of choice.

Animation capability

Not all animations are created equally. Consider the ease of development, the quality of the picture, the smoothness of movement, and the portability for remote viewing. Good animations can be an efficient way to convince decision makers.

Runtime version

An "execute only" version is a convenient way of creating a stand-alone demonstration/animation of the model or specific scenarios. This capability can be quite important in showing results, without the need to have the licensed software system present at every viewing.

4.1.6 Cost considerations

Prices of software vary from 1.000 to 100.000 euros. Focus only on the price to choose software is therefore not possible. In any case, it’s better to find the most relevant software than the cheapest one!

4.2 Specific Considerations

For INTELE project, research oriented simulation processes had to be considered. Software flexibility will be one of the main factors of choice. Links with tools such as Matlab, Excel, C++, etc, must be possible to conduct research activities. More generally, simulation software must be able to interact with other programs used in research. Three candidates were considered as potential solutions.

4.2.1 ProcessModel

ProcessModel is a very good tool for modelling simple systems. The program is very efficient when working on small systems or complex systems with few details. Figure 4 represents the production line of a car manufacturer and a part of its supply chain. The system is very complex and had to be simplified for modelling purpose. ProcessModel is very useful for managers or for non-expert users of simulation, because it is easy to create and animate a model, and then to discuss around the model and understand the basics of the systems.

4.2.2 Flexim ED (Taylor):

« Flexsim is a PC-based simulation software application used to model, simulate, visualize and monitor any business process. Whether the process is manufacturing, material handling, logistics or administration, Flexsim ED will work because its modelling objects can be customized exactly to match the processes you manage. Flexsim ED can help you determine plant capacity, balance manufacturing lines, manage bottlenecks, solve inventory and WIP problems, test new scheduling practices, optimise production rates, and justify capital expenditures. Every model in Flexsim ED can be viewed in 2D, 3D, and virtual reality animation. If you are looking to improve and optimise the processes you manage, you have found the right place. Flexsim ED is an upgrade developed for Taylor ED™ and Enterprise Dynamics™ Simulation Software. »

Flexim ED is one of this “new” simulation software with great 3D abilities (Figure 5). It is not difficult to build complex models and to animate the system. The software also allows making some internal programming with C++ when the basic features are not enough. There is however no possibility to link Flexim with Matlab, which can be considered as a major problem in our case. It is also not possible to create links with other “home-made” programs. In a research context, this solution was therefore rejected.

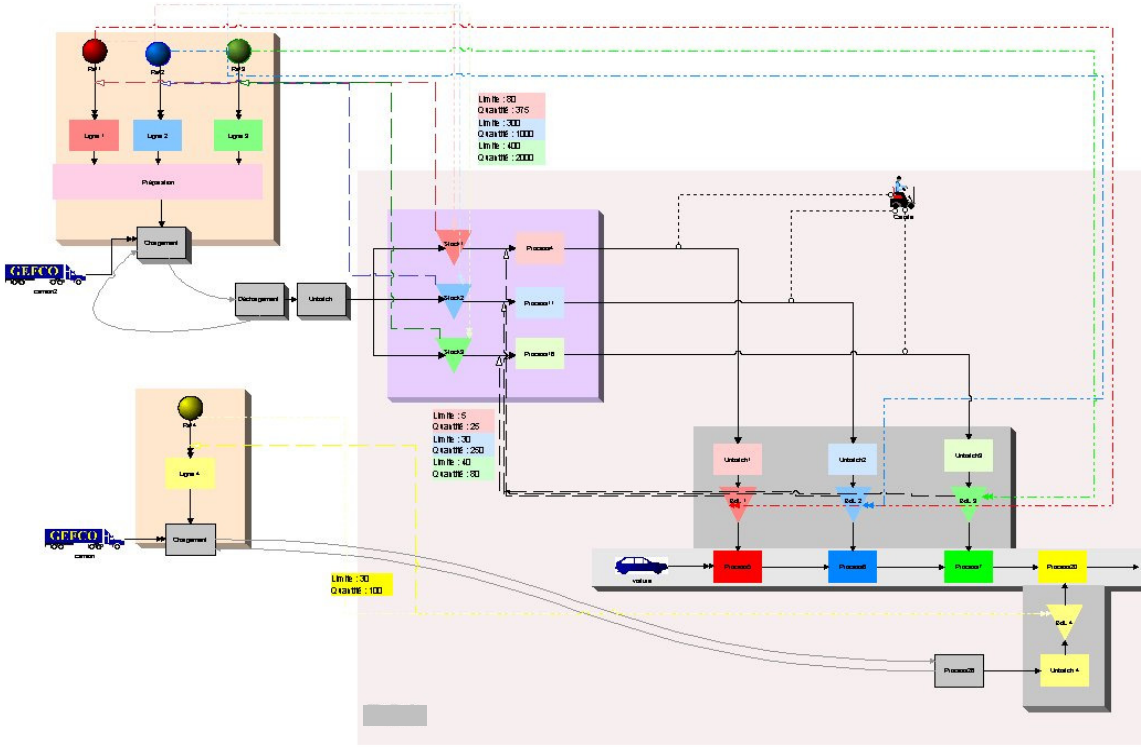

Figure 4. ProcessModel interface.

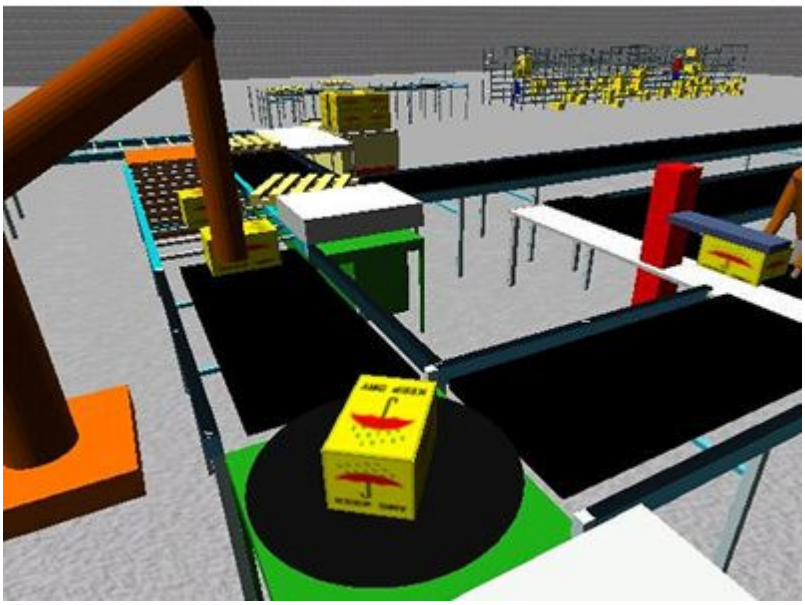

Figure 5. 3D model under Flexim ED.

4.2.3 Arena

For more than ten years, Arena has been the world leader in discrete-event simulation. Arena allows the interactions with other computer tools such as Visual Basic, Excel, etc., and it is very well integrated in Windows environment. With Arena, it is possible to choose the level of complexity by using « basic features » or features that are more specific. It is even possible to create customized tools in the program.

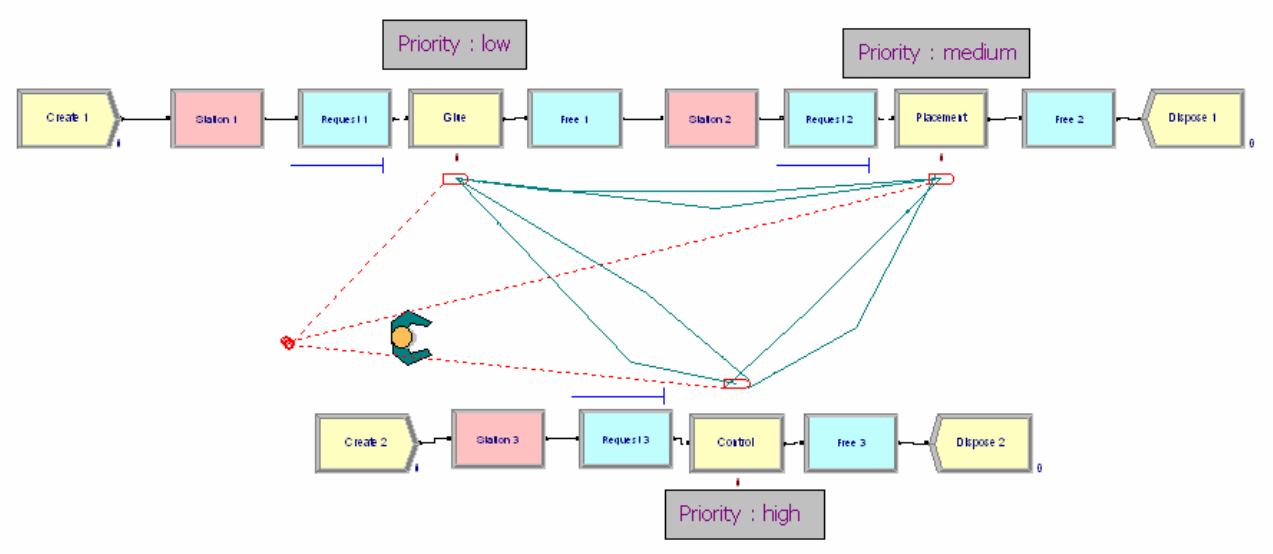

Figure 6. Model for an operator and two production lines with Arena.

Arena is consequently a very good tool for doing research, since it can be linked with other programs in Windows environment and models as complex and accurate as needed can be created. For example, Arena is used in the French Institute for Advanced Mechanics (IFMA) in simulation and process optimisation issues.

4.2.4 Conclusion:

Arena was finally chosen for the purpose of the project because it seems to be the most effective software for doing research with process simulation. Since some researchers already have had prior experience with this software, no special formation is needed. However, never believe that the simulation software can do the job for you. Even if it is very important to choose a good program, it will not compensate for training and experience.

5. CASE STUDY: ELECTRONICS MANUFACTURING

This case study is a simulation project, whose experimental framework and modelling was based on industrial Printed Circuit Boards (PCB) assembly line.

5.1 Electronics manufacturing process

Production lines in electronics manufacturing have already been described in earlier work [Gebus, 2000]. Manufacturing is a linear process composed of a sequence of standard elements, machines linked by conveyors. Machines are mainly used for paste printing, component placement and soldering. More seldom, they are also used for labelling, gluing and automated control of PCBs. In the following, we will however focus only on the most important functions.

Even if modern production systems are increasingly automated, human operators still play an important part on the production. Maintenance and set-up of the production tools are tasks that rely on human workforce to be accomplished. Depending on the type of production, a prompt reaction of the operators to fulfil their tasks is essential in order to have a smooth flow of material and to avoid stoppages. A prototype production type is of course heavily depended on human resources because of the non-negligible time spent in setting up the system at each change of production. However, even in case of mass production constant maintenance is needed due to an increasing complexity of the products and a very high number and variety of components used in modern electronics. For all those reasons, human workforce has to be considered as an independent resource and its scarcity or affectation policy will have a major impact on the production.

Production and human resources management in electronics engineering has taken a new dimension recently. Since a few years electronics industry has become increasingly competitive and many companies are now facing productivity problems. New questions have arisen when the activity started slowing down and when the order books diminished. For this reason, the general idea for this project was to develop some sort of production optimisation method. The production tool needs to become more flexible and adapt to the market demand. This was a good opportunity to introduce discrete events simulation as a tool for evaluating the performance of any optimisation method before implementing it in the factory.

With simulation, it would be possible for researcher at the laboratory to conduct their research work without needing to be constantly testing their solutions on-line and therefore disrupting production. Different solutions could then be studied and depending on the results, the company could choose to implement them or not. Figure 7 shows how a simulation model would have to be integrated in the optimisation process.

The model is based on discrete events simulation. With such a model, it is possible to simulate production over a certain period of time, for example a month, and check the system's reactivity to changes in workload. An optimisation algorithm to define new simulation parameters can then use simulation results. In this way, optimisation methods don't need to rely only on current situation of the production system in order to find an optimum solution that will most probably not be reached in the real process. On contrary, a heuristic approach of production management can help in defining and comparing different solutions and in choosing the one that offers the best results according to predefined parameters.

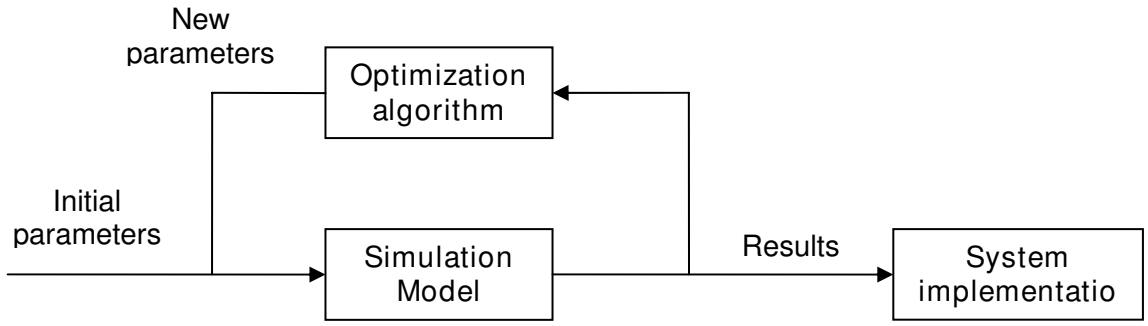

Figure 7. Interactions between optimisation algorithm and simulation model.

5.2 Industrial case

PKC Group is a company involved in the production of Printed Circuit Boards. Earlier work done with this company is presented in [Gebus et al., 2002]. The aim of this project was to develop intelligent decision-helping tools to the specific needs of electronics manufacturing. Two problems were addressed: defect localization on Printed Circuit Boards [Gebus and Juuso, 2002] and production optimisation. According to the first results, the system is successful for new products, even in the ramp-up stage.

Once a system had been developed that can absorb a larger amount of defects, emphasis during the year 2002 was put on production optimisation of the assembly lines. Currently, many decisions are still taken only based on workers experience. Therefore, the work focused on how discrete-event simulation could be used to improve production in electronics manufacturing by providing a better understanding of the production environment and parameters influencing production such as bottlenecks and their impact on overall line capacity. It can be used in a straightforward way to test different scenarios for improvement, but the ability to simulate a real system's behaviour according to some predefined parameters can also be used as an evaluation tool for new control methods.

Increasing number of products and fluctuations of customer demand should be taken into account. The production tool is composed of three assembly lines almost identical in their design. For practical reasons, we will use in the rest of this report the description given by Figure 8. Gluing and labelling machines will not be represented in the process. In addition to these material resources, a production line would not be complete without human operators. They take care of the visual inspection of PCBs before their entry to the soldering oven, but they are also in charge of basic maintenance and setup of the machines. Situation during the time of the project was that two operators on each line were in charge of conducting them, and one additional specialized operator was supervising all three lines in case of more severe breakdowns or failures.

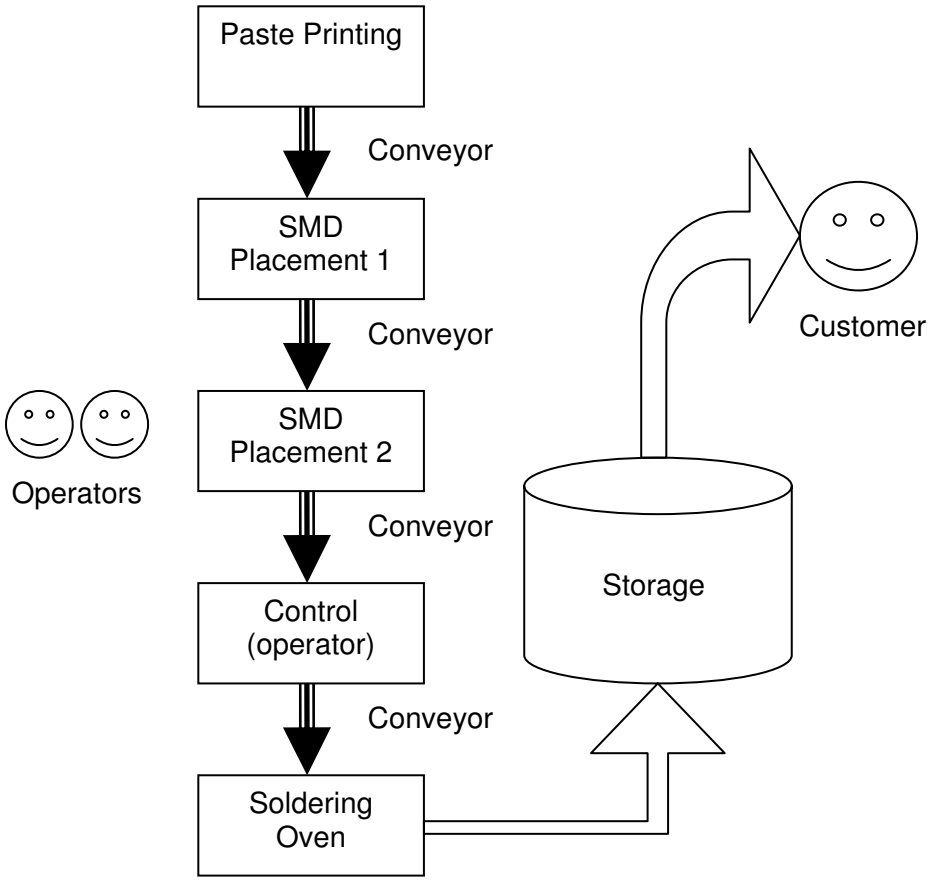

Figure 8. General description of a SMD assembly line.

The company wanted to increase the cost efficiency of their production. In normal times, material costs represent nearly 70% of the costs of a product, but this decreases when machines are not used at maximum capacity, replaced by labour cost. One solution to limit the effect of labour cost would be to increase the flexibility of the production tool and to shut down one of the lines, meaning that fewer operators would be needed for an equivalent output. Therefore, discrete-event simulation has been used as a tool to analyse

the effect of different production parameters on various quality and flexibility indicators. Parameters and indicators are described in details in the experimental framework shown in Figure 8.

5.3 Details of the Experimental Framework

As presented in section 3, the experimental framework describes all the circumstances under which the system is observed and tested. Among other things, it gives a description concerning parameters for the model and ways to interpret simulation results.

5.3.1 Optimisation Functions

The optimisation functions have been chosen according to what can be observed through the simulation model and the general targets defined for the company, namely flexibility increase.

- *Delays:* Probably the biggest production constraint, when the company takes in and accepts an order, the customer assumes that he will get the product at a given time. Delays give information about the ability of the company to respect due dates.
- *Stock levels:* They give a complement of information to the delays. It is not only important to produce before a due date, but also to keep stock levels as low as possible. From an economical point of view, stocks can be considered as immobilized cash in the factory.
- *Resources workload:* In order to have smooth and maximum production, production lines should be balanced and workload distributed equally between resources. The workload of the entire production line is defined by the bottleneck resource. Therefore, a high workload on one of the resources will affect global performance of the line.
- *Buffer levels:* With workload, buffer levels are a second indicator of bottlenecks. Buffers are also used to obtain smooth production and increase flexibility in case of breakdowns.
- *Simulation time:* Simulation time can give a rough overall indication on the performance of a scheduling policy since it is the amount of time needed to process a given set of orders. However, it shouldn't be forgotten that with some scheduling policies, some lines might run empty long before the end of the simulation, affecting simulation statistics.

5.3.2 Input Variables

Only two kinds of input variables have been used in this simulation.

- *Scheduling policies:* They are the different strategies used to affect resources to incoming orders. Scheduling policies will be described together with the results in section 6.
- *Number of operators:* Depending on the number of operators, production will run more or less smoothly. Finding the right amount of operators is a consequence to the main target that is to increase flexibility of the production tool.

Input variables shouldn't be confused with parameters inherent to the model like MTBF (Mean Time Between Failures) and MTTR (Mean Time To Repair). These values, however important, are not a part of any optimisation approach but are considered as constraint relative to the production tool. The same applies to process and setup times, which are constraint relative to a product.

5.3.3 Initialisation

State of the model in the beginning of the simulation has been chosen with all servers, machines and queue lines empty. Reason for this choice is given by the aim of the simulation, which is to compare different scheduling policies. The most important is therefore to have the same initialisation parameters for every simulation run. In order to reduce initial bias, a similar warm-up period is used for every simulation run before starting to collect any statistical data.

5.3.4 Termination

Both terminating and non-terminating simulations are used in order to collect the data necessary for statistical analysis.

Simulation is called terminating if the ending criterion of a run is a predefined state of the system. In our case, the predefined state will be the end of production for a given set of orders. This will be sufficient in most cases to compare the different optimisation functions.

Terminating simulation cannot be used in some cases. Depending on the scheduling policy, some lines may run empty long before the end of the simulation run, which affects statistical results on resource workload and buffer levels for these lines. In these cases, the simulation runs will be stopped after a certain period, corresponding to the stationary state of the system. Simulation is then called non-terminating.

5.3.5 Formalisation of Results and Interpretation

In order to evaluate the performance of different solutions, a comparative analysis has to be made. However, a comparative analysis can only be made if the precision of the simulation results is known. One way to achieve this is to obtain results for a large

number of simulation runs so that Central Limit Theorem can be applied and Gaussian distributions can be used. More details on this subject will be given in section 6.

5.4 The Model

5.4.1 General overview

The model tries to mimic the behaviour of the real workshop. Figure 9 gives a general overview of this model, which consists of three identical production lines. This figure also represents the paths used by operators to move from one machine to another. A path can be limited to the same production line in case of a dedicated operator but it can also link different production lines together when used by the supervising specialist operator. A line is composed of four types of machines:

- 1 Paste printing
- 2 Placement machines
- 1 Control station
- 1 Soldering oven

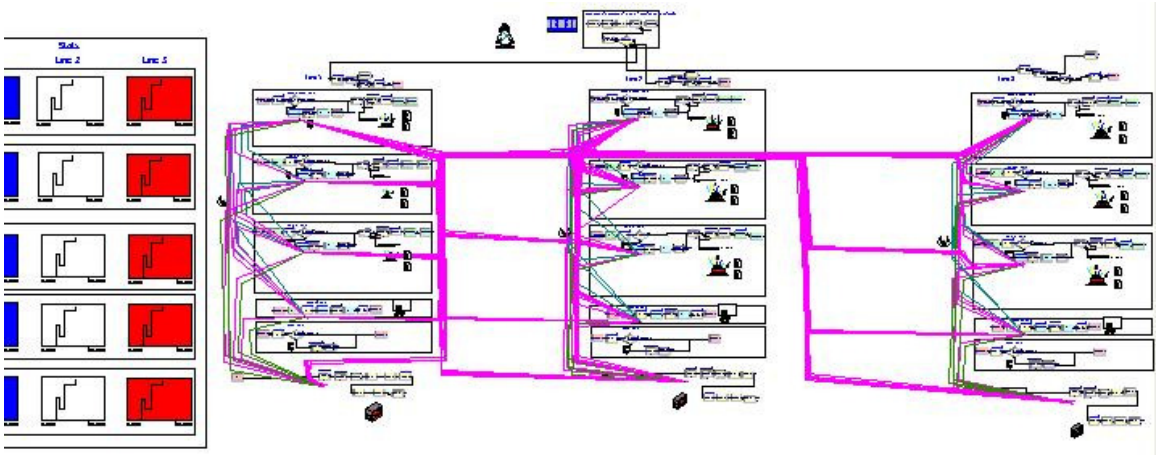

Figure 9. Overall view of the simulation model.

The PCBs enter the specific line by the Paste Printing station and pass successively by the four other stations. The different stations, their components and the logical way they function as well as their failure modes are described in the following sections.

5.4.2 Operators / supervisor

Human resources are composed of two operators per production line and one supervising specialist for the whole workshop. The operators can solve first level failures but the

supervisor is requested for second level failures, which cause more complex breakdowns in production. Here are some other constraints used to model or define the role of the operators:

- Standard operators are dedicated to a specific production line. Only the supervisor can move to other lines when his expertise is needed.
- In addition to first level failures, they are also in charge in quality control, i.e. a PCB will therefore not be released from the control station unless an operator will have checked it.
- Human resources, operators or a supervisor, are modelled with the transporter module of ARENA software. In this way, they can be requested by a resource when needed. As capacity of the transporter depends on the number of operators, it can easily be changed for simulation purposes. Velocity has been fixed at 1m/s that is approximately the speed of a walking human. Distances between the stations have been chosen to correspond to reality.

Figure 10 shows the graphical representation of operators during the simulation. Two different representations have been chosen for standard operator and specialist. This increases understanding of what is happening during the simulation run.

Figure 10. Representation of the operators in the simulation model.

5.4.3 Initialisation of the simulation model

The first part of the simulation model shown in Figure 11 is used for initialising the different variables. At the start of the simulation, one single entity is generated. This entity has no other purpose than activating the modules that will initialise general variables and read the order information, e.g. setup times for the machines or for the oven and number of order. Variables are used to control the sequences of the simulation. There are 12 of them, 4 for each line. For example, the variables defined for line 1 are following:

- *Setup stencil1*: This variable has zero for initial value. It is used to detect a change of production and therefore know if setup is needed for the paste printing resource. Afterwards, last PCB type is assigned to this variable and no setup is needed until next change of production.
- *Setup smd1l1*: This variable has zero for initial value. It is used the same way than *Setup stencil1* variable but for the first placement station. *Setup smd1l1* is also indexed on the PCB type.
- *Setup smd2l1*: Identical to *Setup smd1l1*, this variable is used to detect changes of production on the second placement station.
- *Oven1 config*: This variable is also used to detect a change in production parameters. This time however, it will not be indexed on the PCB type but on the required oven temperature since many types of PCBs will require the same oven settings.

Once the variables initialised, a module is used to generate individual orders. First entity will be duplicated in as many new entities as there are orders to be processed before eventually being disposed.

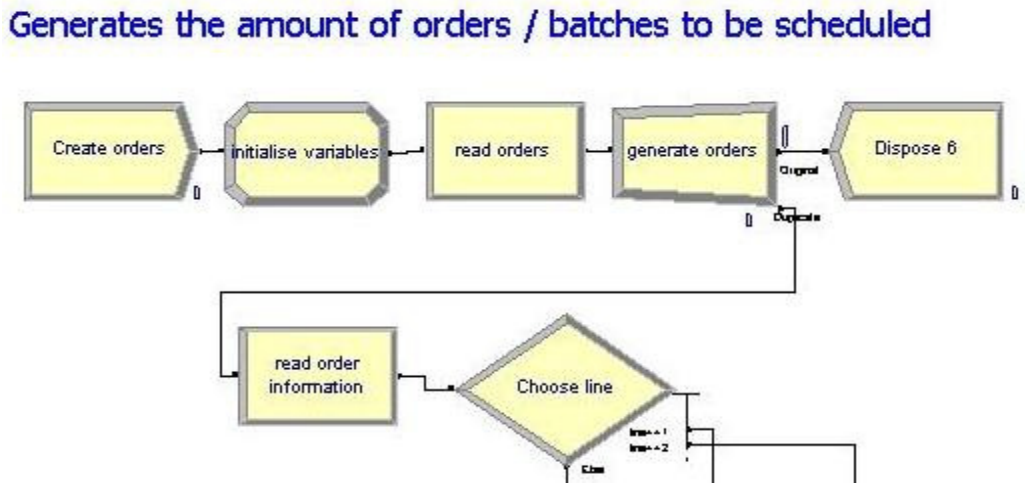

Figure 11. Initialisation procedure of the simulation model.

At this stage of the simulation, the right amount of orders has been generated. Initialisation however is not yet complete and next step is to read the information relative to each individual order, e.g. processing time and due date. Only after this step, orders can be directed to one of the production lines according to any specific scheduling policy and using a decide module.

5.4.4 Beginning of a production line

The initialisation procedure of a production line is shown in Figure 12. Entities coming in are orders directed to this line by the previous decision module. In the same way as the single initialisation entity was duplicated into orders, each order is now duplicated into the right amount of individual PCBs before being disposed.

For modelling purpose, orders are being processed sequentially and are therefore not duplicated into PCBs unless the queue line for this specific line is empty. In this way, the number of entities simultaneously in the model is kept as low as possible and simulation can be speeded up. The same kind of hold modules is being used at several places of the model. Their purpose then is to limit the amount of PCBs on conveyors or other buffers at a predefined level by blocking entities as long as queue lines are at this maximum level. In Figure 12 for example, the second last module will block all PCBs as long as there is no space in the queue line for paste printing station.

Initialisation is now complete at a PCB level and production is ready to start. Last module is a routing module, used like the convey module, to move entities from one station to another. The notion of station is very important: a resource or a group of resources can be represented with it when information is needed about the physical localisation of those resources. Distances between stations are defined as parameters of the model making it possible for conveyors or transporters to move from one station to another. As seen in section 5.4.2, transporters have already been used when modelling operators. Transfer of PCBs will be made using conveyors.

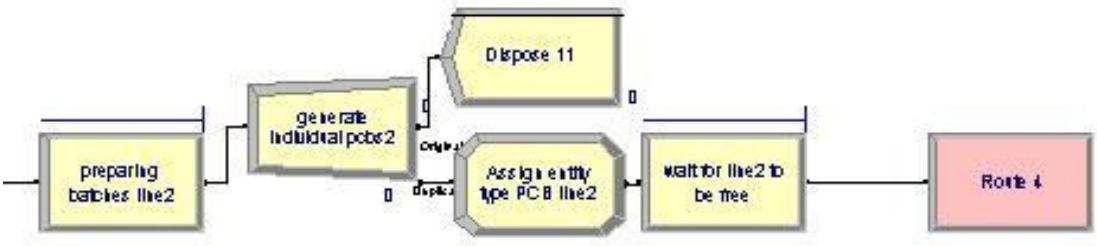

Figure 12. Initialisation procedure for one of the production lines.

5.4.5 Paste Printing and Placement Stations

Paste printing and placement stations have similar modelling structures: they differ only in the model parameters. The general structure of this model is shown in Figure 13.

An entity will arrive to the station from a previous routing or conveying module as long as the queue line for this station is not full. It will then stay in this queue line until the required resource (paste printing machine or placement machine) becomes available. A resource is considered available if it is in idle state and if no setup is necessary. That's where variables defined earlier (*setup stencil1*, *setup smd1l1*…) come into use. A decide module compares the proper setup variable with the PCB type and chooses whether setup is necessary or if the entity can be sent straight to the resource.

If setup is necessary, a request is sent for an operator who will have to make all the adjustments to the machine. Since operators are modelled using transporters, time before the intervention will depend on his last position in the workshop. Setup time for the resource however is modelled with a virtual resource and a processing time defined by the kind of intervention. When setup is complete, operator is released and made available for other tasks and the entity can now enter the resource. The new PCB type is also assigned to the setup variable, meaning that from now on and until next change in production, all PCBs will be directed straight to the resource.

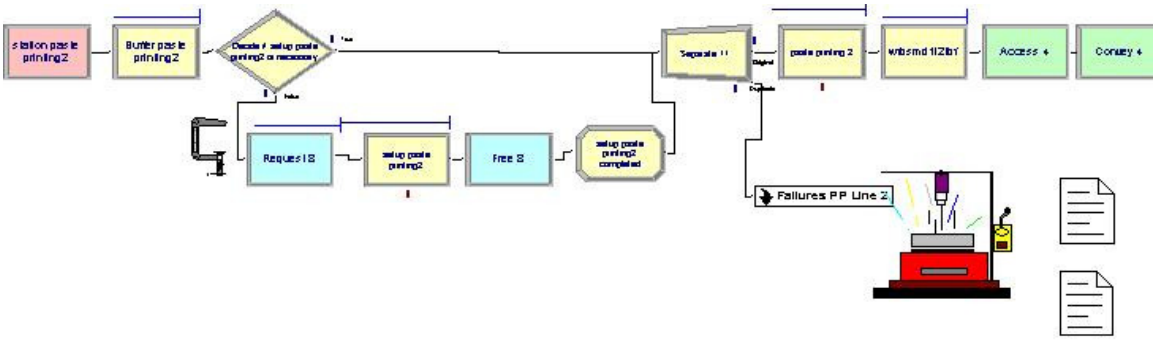

Figure 13. Model of a Paste Printing Station with Arena.

If the first part of the station was dedicated to the setup, the second part represents the process taking place when a PCB enters the resource. A separate entity is generated that will be used to test the state of the resource and simulate the different failure modes and breakdowns described in section 6.3.8. Meanwhile, the original entity goes to the resource to be processed. Figure 14 shows the icons that are displayed during the simulation to represent the different states of the system.

When the process is over, the resource can be released and the entity is ready to be shipped to the next station, but not before checking again that queue lines are not saturated. A conveyor will be requested as soon as those conditions are fulfilled.

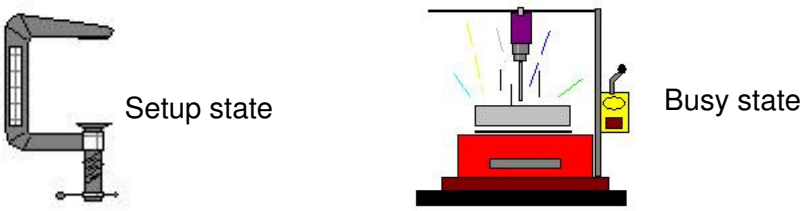

Figure 14. Icons used during simulation to show state of a resource.

5.4.6 Control Station

A control station, represented on Figure 15, is mainly a simplified form of the previous paste printing or placement stations. The main difference is that the resource is an operator instead of a machine. Therefore, neither setup nor maintenance is needed and the resource has only two states, busy or idle as represented in Figure 16. Failure state doesn't need to be considered.

When a PCB is ready to be inspected, a request will be sent for the nearest operator. An operator has to be made available for the control process to begin and he will be released at the end of the inspection.

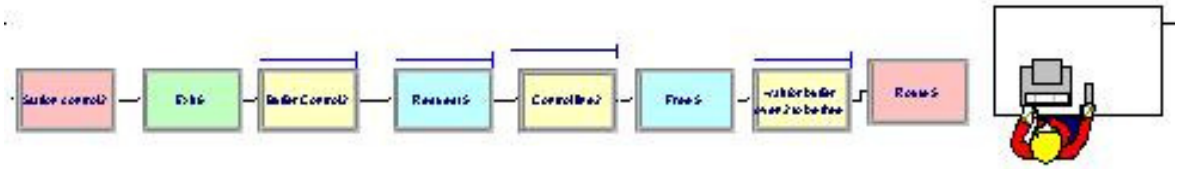

Figure 15. Model of a Control Station with Arena.

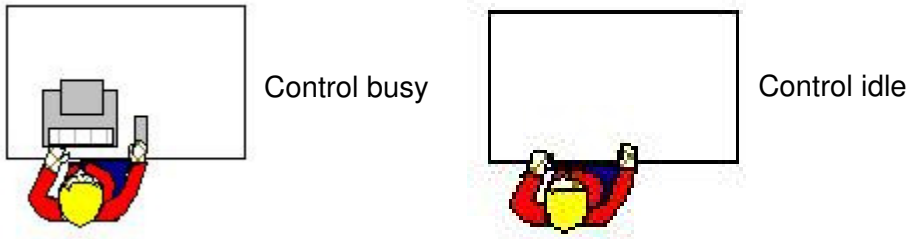

Figure 16. Representation of running and idle control stations.

5.4.7 Soldering Oven Station

The first part of the model for the oven, represented in Figure 17, is similar to the model for paste printing and placement stations. Need for a setup has to be checked using *oven1 setup* variable in the same way than for previous stations. Oven settings however don't need to be changed for each new order, since they depend only on the soldering temperature that can be similar for several PCB types. Furthermore, changing the temperature of the oven is a long process that is only done for specific products.

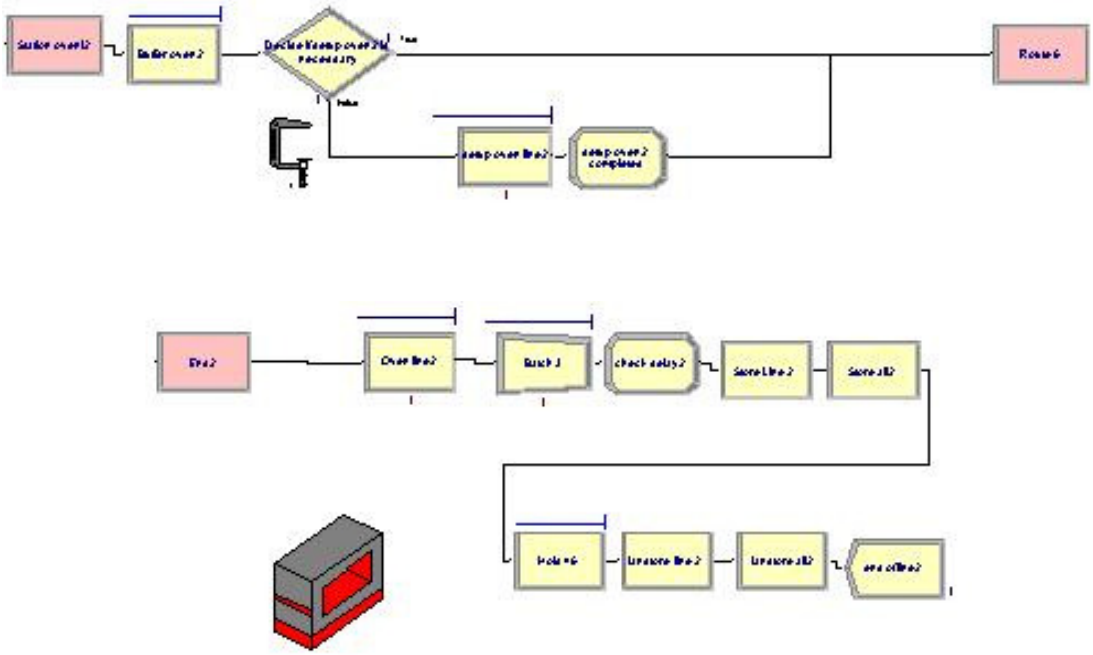

Figure 17. Model of a Soldering Oven Station with Arena.

The second part of the model represents what is taking place inside the soldering oven. Since no process is actually taking place inside the soldering oven, it can be modelled simply using a conveyor. The icon shown on Figure 18 will be displayed as long as there is at least one PCB on this conveyor.

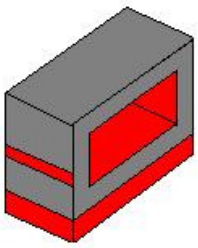

Figure 18. Icon displayed when soldering oven is busy.

A third part, namely the exit of the production line has been added to this model even if it doesn't belong anymore to the soldering oven. PCBs are batched back into orders so that the total amount of entities simultaneously in the model is kept as low as possible. Another reason is also that PCB level information is not anymore needed. An order can have assigned a variable representing the production delay and put to storage until its due date. Statistics can then be collected relative to storage levels and management of just in time production.

5.4.8 Failures and Maintenance

Stoppages on the production lines can have different origins. Some of them are linked to a change in production and are therefore included in the setup times for each machine as described in sections 6.3.5 and 6.3.7. Some stoppages such as failures and maintenance are taking place on a more random base during the production sequence. Both however have in common that there is a need for an operator intervention in order to start again production. For this reason, all stoppages of this kind will be represented as failures in the Arena model. Figure 19 shows the sub-model used for failure management on these stations. Sub-models are being used to limit the number of modules on the main simulation page and therefore keep a clear and understandable main model.

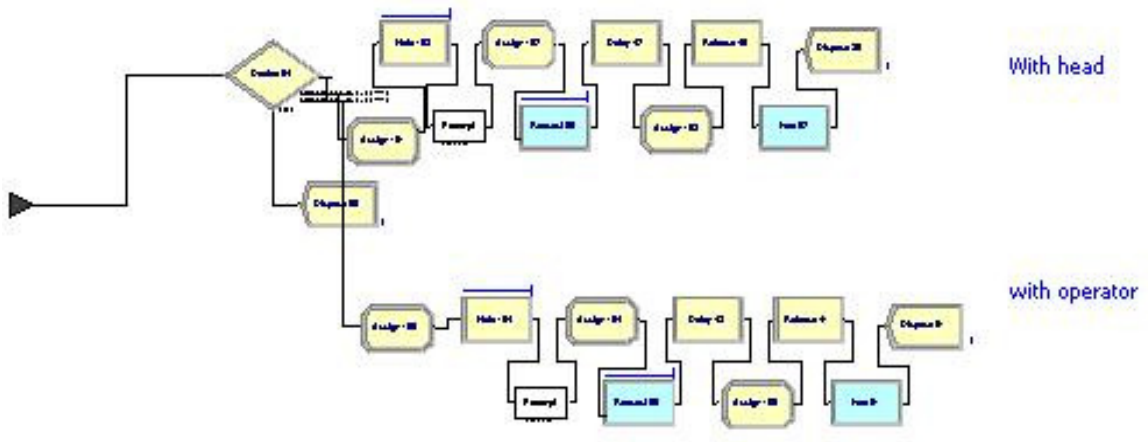

Figure 19. Sub-model for failure management with Arena.

Paste printing and placement stations have two types of failures. For each type of failures, an operator is needed before production can continue. First type of failure requests a standard operator for general maintenance acts such as changing component feeders on placement machines or reloading paste on paste printing machine. Second type of failure represents serious breakdowns requesting an intervention from a specialized operator. Downtime and frequency for breakdowns and maintenance have been chosen according to real data.

Duplicating an entity just before it enters a resource and sending it to the failure management submodel activate failure management in the Arena model. This prevents failures from happening when a line is stopped. Both failure types are based on the same principle. When an entity arrives, the next failure is planned according to a chosen statistical distribution. The entity is then retained until the given failure time. When released two new modules are activated. The first one pre-empts the resource and blocks production whereas the second one calls for an operator. The entity is then again retained for a time corresponding to the downtime of the resource. When released, the resource is re-activated for production to continue and the operator is freed.

During the downtime of the resource, its state is switched to down. This allows using the different graphical representations presented in Figure 20 to show the state of a resource when the simulation is running in animated mode. In addition to this, resource utilisation is collected for each station during the simulation and displayed on histograms making it possible to track over a certain time period the effect of breakdowns.

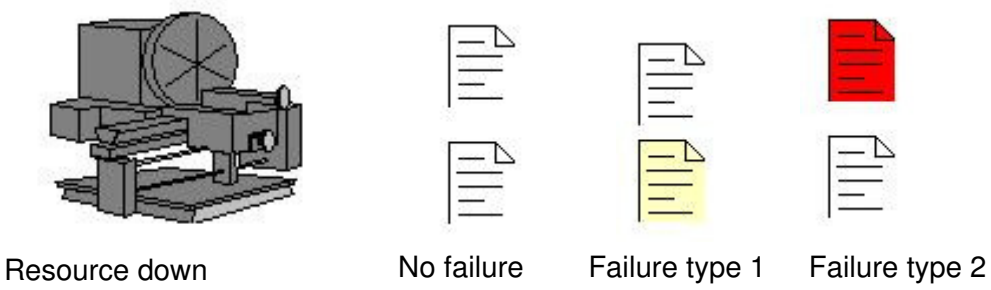

Figure 20. Graphical representations of the state of a resource.

6. COMPARISON OF SCHEDULING POLICIES

Discrete-event simulation is a tool for comparing production alternative. Handling part of the constraints as preferences and priorities improves performance of the scheduler in changing operating conditions. Intelligent methods are suitable for smooth adaptation of scheduling rules to the changing operation conditions. An optimisation-based inference engine handles an aggregate optimisation problem for the Mid Term Scheduling. For the short term scheduling, some preference criteria can be used to select from several feasible alternatives. [Juuso and Jagdev, 1999]

Functionality of the simulation models developed in the case study has been tested with comparisons of three scheduling strategies by running the same simulations for each of these strategies.

6.1 Comparison of scheduling policies

Several very simple scheduling policies have been tested on the simulation model. This is the easiest way to use simulation and implementing them would need only minor adaptation of the production tools since in the current state of the system they mainly represent changes in the operator's behaviour. It is therefore interesting to see how those changes would affect the process output.

Downside however is that the optimisation methods are limited and have to stay simple. In the following part, four different scheduling policies are presented and three of them have been tested with the simulation model. Evaluation has been limited to the three first policies because they could be tested on almost similar models. Mixed strategy however would have needed to change the structure of the model and use external software to run the optimisation algorithm. For the same reason, more complex optimisation methods based for example on evolutionary algorithms [Pierreval et al 1998] have not been selected. The scheduling system presented in [Juuso and Jagdev, 1999] has been implemented in Matlab© environment. Enhanced methods involving interconnection between different software are left for later studies.

6.1.1 Strategy A: First free line

This is probably the simplest of all scheduling policies but unfortunately in many cases not a very efficient one. It has been considered in this work as a base for comparison with other, more advanced, strategies.

First free line strategy will order the batches by increasing estimated start time. These batches will then be sent to the first production line becoming available. The choice of the production line doesn't depend on any specific criteria. This is the worst-case scenario and therefore only poor results are expected.

6.1.2 Strategy B: Priority to oven temperature

This strategy is slightly more advanced than the first free line strategy: priority is given to the reduction of setup times for the production lines. Longest setup times appear to be for the soldering oven, where temperature changes may take up to an hour. It makes sense therefore to sort batches out according to their required soldering temperature.

Strategy B categorises batches according to their soldering temperature and in each category, those batches are ordered by increasing start time. Production lines are then affected to a specific oven temperature in order to minimise setups.

6.1.3 Strategy C: Improved oven temperature priority

Based on the strategy B, this strategy keeps the characteristics described above. Main improvement is made on the simulation termination, where strategy B had the disadvantage to let some lines run empty much earlier than other ones, depending on the workload for a specific oven setting.

With strategy C, when a production line runs empty, state of the workshop is observed. If another production line has an important remaining workload and several orders in its queue line, then part of this production will be rerouted to the empty production line. This way, oven setup is only done when no other batch is available for a given production line.

6.1.4 Strategy D: Mixed strategy

Mixed strategy is a result of observations made on the previous strategy. Changing the oven temperature is a long process but doesn't necessarily penalise production when done in an intelligent fashion. Production lines rarely run completely empty for a specific temperature setting, so strategy C would never apply in a real case. However it is not considered clever for example to run a production line when only few orders are scheduled at a much later date. For this reason it is possible to introduce time windows in which strategy C would apply.

Mixed strategy starts production based on an oven temperature priority. However, when workload diminishes and a production line runs empty in a given time window, then orders will be rerouted from the production line with the heaviest workload. After completion, production can go back to normal.

6.2 Simulation results

Two sets of simulation results are presented in Table 1. The first set presents the results of simulating the three first strategies with a given set of orders and comparing the utilisation of each production line and the time to complete the given production obtained the first one. Using the reference strategy and varying the amount of operators have obtained the second set of results. Observations for the second set have been made on the production line utilisation. For both sets, 40 replications have been used to ensure good confidence intervals and relevancy of the results.

Concerning the choice of a scheduling strategy, as expected strategy A has the poorest performances since no optimisation criteria has been chosen. With both strategies B and C however, we have a dramatic increase in performance. Utilisation becomes much higher as soon as oven temperature is included as a criterion for production line affectation allowing saving up to 60% of production time by lowering maintenance and setup operations. This shows the importance that one single criterion can have on overall performance of a production line. When switching from strategy B to strategy C however, it was not observed the expected smoothening of utilisation levels. Reason for this can be the size of the chosen batch but strategy D could not be tested on the similar model to confirm this idea.

Table 1. Simulation Results.

Utilisation of the production lines with different scheduling strategies

	Strategy A	Strategy B	Strategy C
Line 1	0.30	0.45	0.26
Line 2	0.29	0.39	0.47
Line 3	0.29	0.63	0.64
Time to complete the batches	163 hours	128 hours	104 hours

Utilisation of the production lines with different operator configurations

	2 op. 1 sup.	3 op. 1 sup.	3 op. 2 sup.
Line 1	0.30	0.29	0.29
Line 2	0.29	0.29	0.29
Line 3	0.29	0.29	0.29

Concerning the different operator configurations, it has been shown that adding operators or supervisors doesn't have any perceptible effect on overall performance. This would mean that with current failure and breakdown parameters, operator configuration already provides optimum performance.

7. CONCLUSION

This study introduces new modelling techniques in the field of electronics manufacturing. Focus was put on how those new tools could be used to develop control methods. Discrete-event simulation has proven to be a good tool to evaluate performances of different optimisation methods. Good results were obtained in the case of various basic scheduling policies, allowing comparison between different options. The model could now be used to develop intelligent optimisation methods through links with external software such as Matlab©. Among others, genetic algorithms have been considered as a possible choice. Any heuristically based method would benefit greatly of the ability that discrete-event simulation has to mimic real processes.

REFERENCES

/1/ Andersson M. & Olsson G., 1998, "*A Simulation Based Decision Support Approach for Operational Capacity Planning in a Customer Order Driven Assembly Line*". In: Proceedings of 1998 Winter Simulation Conference, Ed. D J Medeiros, E F Watson, J S Carson & M S Manivannan.

/2/ Czarmecki H., Schroer B. J. & Rahman M. M., 1997, "*Using Simulation to Schedule Manufacturing Resources*". In: Proceedings of the 1997 Winter Simulation Conference, Ed. S Andradóttir, K J Healy, D H Withers & B L Nelson.

/3/ Estremadoyro D. N., Farrington P. A., Schroer B. J. & Swain J. J., 1997, "*Simulation of Memory Chip Line using an Electronics Manufacturing Simulator*". In: Proceedings of the 1997 Winter Simulation Conference, Ed. S Andradóttir, K J Healy, D H Withers & B L Nelson.

/4/ Gebus S., 2000, "*Process Control Tool for a Production Line at Nokia*". Report A No 13, December 2000. University of Oulu, Control Engineering Laboratory, 27p. ISBN 951-42-5870-3

/5/ Gebus, S. & Juuso, E., 2002, "Industrial Utilization of Linguistic Equations for Defect Detection on Printed Circuit Boards". In Proceedings of the IECON'02 Conference of IEEE Industrial Electronics Society, Seville, Spain, November 5-8, 2002, pp. 1887-1892.

/6/ Gebus S., Lorillard S. & Juuso E., 2002, "*Defect Localization on a PCB with Functional Testing*". Report A No 20, May 2002. University of Oulu, Control Engineering Laboratory, 44 p. ISBN 951-42-6731-1

/7/ Harmonosky C. M., Miller J. L., Rosen S. L., Traband M. T., Tillotson R. & Robbie D., 1999, "*Interfacing Simulation with Costing Software to Drive the Transformation from Prototype Manufacturing to High Volume Manufacturing*". In: Proceedings of the 1999 Winter Simulation Conference, Ed. P A Farrington, H B Nembhard, D T Sturrock & G W Evans.

/8/ Jackson M. & Johansson C., 1997, "*Real Time Discrete-Event Simulation of a PCB Production System for Operational Support*". In: Proceedings of the 1997 Winter Simulation Conference, Ed. S Andradóttir, K J Healy, D H Withers & B L Nelson.

/9/ Juuso, E. K. and Jagdev, H. S., 1999, "Intelligent Scheduling of Semi-Continuous Systems". In: Leena Yliniemi and Esko Juuso (eds.) Proceedings of TOOLMET'99 Symposium - Tool Environments and Development Methods for Intelligent System. Oulu, Finland, pp. 182-193.

/10/ Law A. M. & McComas M. G., 1998, "*Simulation of Manufacturing Systems*". In: Proceedings of 1998 Winter Simulation Conference, Ed. D J Medeiros, E F Watson, J S Carson & M S Manivannan.

/11/ Pedgen C., Sadowski R. & Shannon R., 1995, "*Introduction to Simulation using SIMAN*". 2nd ed., McGraw-Hill, Singapore.

/12/ Pierreval H & Plaquin M F, 1998, "*An Evolutionary Approach of Multicriteria Manufacturing Cell Formation*". International Transactions in Operational Research Vol.5, pp. 13-25.

/13/ Savsar M., 1997, "*Simulation analysis of a pull-push system for an electronic assembly line*". International Journal of Production Economics 51 (1997), pp.205-214

/14/ Savsar M. & Allahverdi A., 1999, "*Algorithms for scheduling jobs on two serial duplicate stations*". International Transactions in Operational Research Vol.6, pp. 411-422.

/15/ Schriber T. J., 1987, "*The Nature and Role of Simulation in the Design of Manufacturing Systems*". In: Simulation in CIM and Artificial Intelligence Techniques, Ed. Retti J & Wichmann K E, Society of Computer Simulation, pp. 5-18

/16/ Schriber T. J. & Brunner D. T., 1997, "*Inside Discrete-Event Simulation Software: How it works and Why it matters*". In: Proceedings of the 1997 Winter Simulation Conference, Ed. S Andradóttir, K J Healy, D H Withers & B L Nelson.

/17/ Williams E. J. & Gevaert A., 1997, "*Pallet Optimization and Throughput Estimation via Simulation*". In: Proceedings of the 1997 Winter Simulation Conference, Ed. S Andradóttir, K J Healy, D H Withers & B L Nelson.

Notes about SADT and IDEF methods:

- System Analysis and Design Technic (SADT) is a trademark of SofTech Inc.
- Integrated DEFinition language methods have been developed by the Computer Systems Laboratory of the National Institute of Standards and Technology (NIST). In December 1993, IDEFØ has been released as a standard for Function Modeling in FIPS Publication 183A. More information about IDEF methods can be found at http://www.idef.com/

ACKNOWLEDGEMENT

The study of the Process Control Tool was done within the Socrates exchange programme between *University of Oulu* and *Institut Francais de Mecanique Avancee (IFMA)* in connection to the INTELE project funded by TEKES, PKC Group, Filtronic and JOT Automation.

ISBN 951-42-7372-9
ISSN 1238-9390
University of Oulu
Control Engineering Laboratory – Series A
Editor: Leena Yliniemi

1. Yliniemi L, Alaimo L & Koskinen J, Development and tuning of a fuzzy controller for a rotary dryer. December 1995. ISBN 951-42-4324-2.
2. Leiviskä K, Simulation in pulp and paper industry. February 1996. ISBN 951-42-4374-9.
3. Yliniemi L, Lindfors J & Leiviskä K, Transfer of hypermedia material through computer networks. May 1996.ISBN 951-42-4394-3.
4. Yliniemi L & Juuso E (editors), Proceedings of TOOLMET'96 – Tool environments and development methods for intelligent systems. May 1996. ISBN 951-42-4397-8.
5. Lemmetti A, Leiviskä K & Sutinen R, Kappa number prediction based on cooking liquor measurements. May 1998. ISBN 951-42-4964-X.
6. Jaako J, Aspects of process modelling. September 1998. ISBN 951-42-5035-4.
7. Lemmetti A, Murtovaara S, Leiviskä K & Sutinen R, Cooking variables affecting the craft pulp properties. June 1999. ISBN 951-42-5309-4.
8. Donnini P A, Linguistic equations and their hardware realisation in image analysis. June 1999. ISBN 951-42-5314-0.
9. Murtovaara S, Juuso E, Sutinen R & Leiviskä K, Modelling of pulp characteristics in kraft cooking. December 1999. ISBN 951-42-5480-5.
10. Cammarata L & Yliniemi L, Development of a self-tuning fuzzy logic controller (STFLC) for a rotary dryer. December 1999. ISBN 951-42-5493-7.
11. Isokangas A & Juuso E, Fuzzy modelling with linguistic equation methods. February 2000.33 p. ISBN 951-42-5546-1.
12. Juuso E, Jokinen T, Ylikunnari J & Leiviskä K, Quality forecasting tool for electronics manufacturing. March 2000. ISBN 951-42-5599-2.
13. Gebus S, Process Control Tool for a Production Line at Nokia. December 2000. 27 p. ISBN 951-42-5870-3.
14. Juuso E & Kangas P, Compacting Large Fuzzy Set Systems into a Set of Linguistic Equations. December 2000. 23 p. ISBN 951-42-5871-1.
15. Juuso E & Alajärvi K, Time Series Forecasting with Intelligent Methods. December 2000. 25 p. Not available.
16. Koskinen J, Kortelainen J & Sutinen R, Measurement of TMP properties based on NIR spectral analysis. February 2001. ISBN 951-42-5892-4.
17. Pirrello L, Yliniemi L & Leiviskä K, Development of a Fuzzy Logic Controller for a Rotary Dryer with Self-Tuning of Scaling Factor. 32 p. June 2001. ISBN 951-42-6424-X.
18. Fratantonio D, Yliniemi L & Leiviskä K, Fuzzy Modeling for a Rotary Dryer. 26 p. June 2001. ISBN 951-42-6433-9.
19. Ruusunen M & Paavola M, Quality Monitoring and Fault Detection in an Automated Manufacturing System - a Soft Computing Approach. 33 p. May 2002. ISBN 951-42-6726-5.

20. Gebus S, Lorillard S & Juuso E, Defect Localization on a PCB with Functional Testing. 44 p. May 2002. ISBN 951-42-6731-1.
21. Saarela U, Leiviskä K & Juuso E, Modelling of a Fed-Batch Fermentation Process. 23 p. June 2003. ISBN 951-42-7083-5.
22. Warnier E, Yliniemi L & Joensuu P, Web based monitoring and control of industrial processes. 15 p. September 2003. ISBN 951-42-7173-4.
23. Van Ast J M & Ruusunen M, A Guide to Independent Component Analysis – Theory and Practice. 53 p. March 2004. ISBN 951-42-7315-X.
24. Gebus S, Martin O, Soulas A. & Juuso E, Production Optimization on PCB Assembly Lines using Discrete-Event Simulation. 37 p. May 2004. ISBN 951-42-7372-9.