der Bundeswehr
Universität München

# DEVELOPMENT AND INTRODUCTION OF SEMICONDUCTOR PRODUCTION CONTROL ENHANCEMENTS FOR CUSTOMER ORIENTED MANUFACTURING

**Mike Gißrau**

## DISSERTATION

Vollständiger Abdruck der von der Fakultät für Informatik der Universität der Bundeswehr München zur Erlangung des akademischen Grades eines

**Doktors der Naturwissenschaften (Dr. rer. nat.)**

genehmigte Dissertation.

**Gutachter:**
1. Prof. Dr. rer. nat. Oliver Rose
2. Prof. Dr. Stéphane Dauzère-Pérès

Die Dissertation wurde am 25.06.2013 bei der Universität der Bundeswehr München eingereicht und durch die Fakultät für Informatik am 28.06.2013 angenommen. Die mündliche Prüfung fand am 17.12.2013 statt.

# Kurzfassung

Produktionssteuerung im Bereich der kundenorientierten Halbleiterfertigung ist heutzutage eine sehr komplexe und zeitintensive Aufgabe. Verschiedene Anforderungen bezüglich der Fabrikperformance werden seitens der Kunden als auch des Fabrikmanagements definiert. Diese Anforderungen stehen oftmals in Konkurrenz. Dadurch ist eine effiziente Strategie zur Kompromissfindung nicht einfach zu definieren.

Heutige Halbleiterfabriken mit ihren verfügbaren Produktionssteuerungssystemen nutzen oft prioritätsbasierte Lösungen zur Definition der Wichtigkeit eines jeden Produktionsloses. Anhand dieser Prioritäten werden die Produktionslose sortiert und bearbeitet. In der Literatur existiert eine große Bandbreite verschiedener Algorithmen. Im Bereich der kundenorientierten Halbleiterfertigung wird eine sehr flexible und anpassbare Strategie benötigt, die auch den aktuellen Fabrikzustand als auch die wechselnden Kundenanforderungen berücksichtigt. Dies gilt insbesondere für den hochvariablen geringvolumigen Produktionsfall. Diese Arbeit behandelt eine flexible Strategie für den hochvariablen Produktionsfall einer solchen Produktionsstätte. Der Algorithmus basiert auf einem detaillierten Fabriksimulationsmodell mit Rückgriff auf Realdaten. Neben synthetischen Testdaten wurde der Algorithmus auch anhand einer realen Fertigungsumgebung geprüft. Verschiedene Steuerungsregeln werden hierbei sinnvoll kombiniert und gewichtet. Wechselnde Anforderungen wie Linienbalance, Durchsatz oder Liefertermintreue können adressiert und optimiert werden. Mittels einer definierten Zielfunktion erlaubt die automatische Modellgenerierung eine Optimierung anhand des aktuellen Fabrikzustandes. Die Optimierung basiert auf einen genetischen Algorithmus für eine flexible und effiziente Lösungssuche.

Die Strategie wurde mit Realdaten aus der Fertigung einer typischen hochvariablen geringvolumigen Halbleiterfertigung geprüft und analysiert. Die Analyse zeigt ein Verbesserungspotential von 5% bis 8% für die bekannten Performancekriterien wie Cycletime im Vergleich zu gewöhnlichen statischen Steuerungspolitiken. Eine prototypische Implementierung realisiert diesen Ansatz zur Nutzung in der realen Fabrikumgebung. Die Implementierung basiert auf der JAVA-Programmiersprache. Aktuelle Implementierungsmethoden erlauben den flexiblen Einsatz in der Produktionsumgebung.

Neben der Fabriksteuerung wurde die Möglichkeit der Reduktion von Messoperationszeit (auch bekannt unter Sampling) unter gegebenen Randbedingungen einer hochvariablen geringvolumigen Fertigung untersucht und geprüft. Oftmals ist aufgrund stabiler Prozesse in der Fertigung die Messung aller Lose an einem bestimmten Produktionsschritt nicht notwendig. Diese Arbeit untersucht den Einfluss dieses gängigen Verfahrens aus der Massenfertigung für die spezielle geringvolumige Produktionsumgebung. Die Analysen zeigen insbesondere in Ausnahmesituationen wie Anlagenausfällen und Kapazitätsengpässe einen positiven Effekt, während der Einfluss unter normalen Produktionsbedingungen aufgrund der hohen Produktvariabilität als gering angesehen werden kann.

Nach produktiver Einführung in einem typischen Vertreter dieser Halbleiterfabriken zeigten sich schnell positive Effekte auf die Fabrikperformance als auch eine breite Nutzerakzeptanz. Das implementierte System wurde Bestandteil der täglichen genutzten Werkzeuglandschaft an diesem Standort.

# Abstract

Production control in a semiconductor production facility is a very complex and time-consuming task. Different demands regarding facility performance parameters are defined by customer and facility management. These requirements are usually opponents, and an efficient strategy is not simple to define.

In semiconductor manufacturing, the available production control systems often use priorities to define the importance of each production lot. The production lots are ranked according to the defined priorities. This process is called dispatching. The priority allocation is carried out by special algorithms. In literature, a huge variety of different strategies and rules is available. For the semiconductor foundry business, there is a need for a very flexible and adaptable policy taking the facility state and the defined requirements into account. At our case the production processes are characterized by a low-volume high-mix product portfolio. This portfolio causes additional stability problems and performance lags. The unstable characteristic increases the influence of reasonable production control logic.

This thesis offers a very flexible and adaptable production control policy. This policy is based on a detailed facility model with real-life production data. The data is extracted from a real high-mix low-volume semiconductor facility. The dispatching strategy combines several dispatching rules. Different requirements like line balance, throughput optimization and on-time delivery targets can be taken into account. An automated detailed facility model calculates a semi-optimal combination of the different dispatching rules under a defined objective function. The objective function includes different demands from the management and the customer. The optimization is realized by a genetic heuristic for a fast and efficient finding of a close-to-optimal solution.

The strategy is evaluated with real-life production data. The analysis with the detailed facility model of this fab shows an average improvement of 5% to 8% for several facility performance parameters like cycle time per mask layer.

Finally the approach is realized and applied at a typical high-mix low-volume semiconductor facility. The system realization bases on a JAVA implementation. This implementation includes common state-of-the-art technologies such as web services. The system replaces the older production control solution.

Besides the dispatching algorithm, the production policy includes the possibility to skip several metrology operations under defined boundary conditions. In a real-life production process, not all metrology operations are necessary for each lot. The thesis evaluates the influence of the sampling mechanism to the production process. The solution is included into the system implementation as a framework to assign different sampling rules to different metrology operations. Evaluations show greater improvements at bottleneck situations.

After the productive introduction and usage of both systems, the practical results are evaluated. The staff survey offers good acceptance and response to the system. Furthermore positive effects on the performance measures are visible. The implemented system became part of the daily tools of a real semiconductor facility.

# Acknowledgment

First of all, I am indebted to my doctoral adviser Prof. Dr. rer. nat. Oliver Rose. After interesting studies at the Dresden University of Technology, he gave me the chance to pursue an external Ph.D. under his supervision with cooperation of the X-FAB Dresden semiconductor foundry. For the whole duration of the Ph.D., with its ups and downs, he gave great support in profound discussions. Several conference papers were created with his support.

I am also indebted to the X-FAB Dresden offering the chance to implement my ideas and proposals. Without the support of the great colleagues including collaboration and technical support, this thesis would not have been possible. Of course, it was very difficult to win over everybody for my ideas. The working atmosphere was great and offered me the chance to fulfill all my ideas. Specifically, I want to thank my manager Jan Orbanz for his confidence in my abilities and his support with various problems and demands. I also want to thank Dr. Matthias Hänisch for his very valuable and productive discussions during the whole work and conferences. His special point of view on semiconductor production processes was fruitful as well as exhilarating at times. Big thanks are also addressed to the whole IT team of the X-FAB Dresden, giving me great support with various problems and tasks.

# Nomenclature

| | |
|---|---|
| ASIC | application specific integrated circuit |
| CAD | computer aided design |
| CAE | computer aided engineering |
| CAM | computer aided manufacturing |
| CD | critical dimensions |
| CM | cycle time per mask layer |
| CR | critical ratio |
| CT | cycle time |
| DLS | dynamic lot sampling |
| EDD | earliest due date |
| ERP | enterprise resource planning |
| FIFO | first in first out |
| FSM | finite state machine |
| HMLV | high-mix low-volume |
| HMLVSF | high-mix low-volume semiconductor facility |
| IDM | integrated device manufacturer |
| IT | information technology |
| JSL | JAVA simulation libary |
| KPI | key performance indicator of the facility |
| LB | line balance |
| MES | manufacturing execution system |
| ODD | operation due date |

| | |
|---|---|
| OTD | on-time delivery |
| RM | recipe management |
| SA | setup avoidance |
| SFB | semiconductor foundry business |
| SF | semiconductor foundry |
| SM | semiconductor manufacturer |
| SPT | shortest processing time first |
| TD | tardiness |
| TH | throughput |
| UML | unified modeling language |
| WIP | work in process |
| XF | X-factor or flow factor |

# Contents

# Part I

# Preliminaries

Business? That's very simple: it's other people's money.

*(Alexandre Dumas (1824-1895))*

# 1 Introduction

## 1.1 Preface

Variation and change are the two main characteristics of the semiconductor foundry business (SFB) in the last years. Variable business models, different technologies and production tasks, changing orders from customers and new technologies have major impact on the whole production process. New customer demands like higher individuality of the products are one of the main trends in the business. The fast change of the conditions supersedes algorithms and solutions for problems from yesterday.

The semiconductor business was not as complicated back then as it is today. Semiconductor applications are in use from the beginning of the 20th century. In the year 1906 Lee Deforest developed the world's first vacuum tube. This invention made way for new media applications like radio, television and other electronic devices. Furthermore the vacuum tube was used in the world's first electronic computer (ENIAC) in 1947. On 23 December 1947, the first device working like a vacuum tube was designed with semiconductor material. This element was called transfer resistor, also known as transistor. In the year 1959, the first integrated circuit was developed by Jack Kilby, an engineer at Texas Instruments. He formed a complete electrical circuit with one piece of semiconductor material. He combined several transistors, diodes and capacitors.

In the following years the semiconductor industry has seen a huge development and improvement of their processes and production technologies. The transistor count per integrated circuit increased while the chip size itself decreased (see Figure 1.1). Times of extreme growth in the 1990s are unsoldered by times of extreme shrinking at beginning of the 21st century. The changes of these major business and process conditions cause a high focus on the product and technology development. Aspects of operational demands are often neglected. As a major result of this development, production control is often inefficient and outdated. Often it does not meet the requirements defined by the management today. Cycle times often exceed 60 to 90 days, which is not acceptable in the area of semiconductor foundries.

The SFB model is characterized by fabrication of customer specific products. Originally the microelectronic devices were designed and produced by one company. A precise knowledge of the design and the production process was necessary. No common standards were established. Later on, with the upcoming standardization of manufacturing processes, microelectronic devices became more and more standardized. As a result the devices could be produced by more than one manufacturer. The electronic design automation was introduced to share design data between different manufacturers. A separation between design and production became possible.

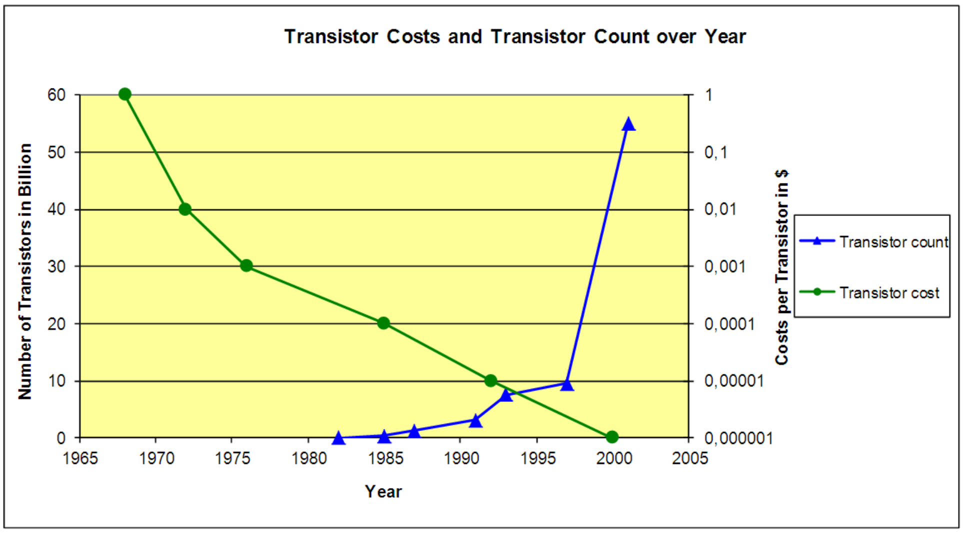


Figure 1.1: Evolution of the transistor costs and count over the last years (source [vZ04])

Currently two business models are available. Some companies continue to design and produce their own devices. Other companies are fabless, only designing their technologies and ordering the products from foundry companies. The switch to a fabless model seems to have some benefits regarding production optimization and cost reduction[1] (for more information see [Doe07, HS01, vZ04]).

Foundries have to deal with higher variety of production processes and customer demands regarding production throughput and quality. This is partly caused by a high mix of different technologies. The current market situation affects the production at a foundry. Short technology cycles, the diversity of customer orders and a short term factory load planning are common practice. In this area each customer has to be satisfied in regard to quality and order compliance. Thus huge varieties in the production process have to be avoided. For more information about the foundry business model see Section 2.1.

Production control is one vital operational task in this kind of environment. Sometimes additional commercial systems are used which are not well understood and not adapted to the specific facility conditions. Often only a simple built-in manufacturing execution system (MES) is used. At our case the MES was used several years without adapting the production control to the variation at the business and production processes. The gap between the production processes and the software support is an major issue in historically grown semiconductor foundries. Short cycle times and fast reaction to market changes are difficult to achieve without reasonable production control approaches.

To provide an efficient solution with regard to the changing manufacturing conditions and business demands, we analyze different solutions and approaches. Different management demands and process characteristics are taken into account to define and implement a general policy. Special demands of the SFB are considered. Finally we evaluate the

[1] The first real foundry was founded in 1987: Taiwan Semiconductor Manufacturing Company (TSMC)

practical benefit of a prototypical implementation. This implementation is applied to a real production control system at a typical high-mix low-volume semiconductor foundry (HMLVSF).

## 1.2 The Goal of the Thesis

This thesis is based on work at the X-FAB Dresden semiconductor foundry. Goal of the thesis is the development and introduction of a new production control system for a facility of a standard foundry business. Our reference model and implementation are applied at the Dresden facility. Different boundary conditions like the inhomogeneous data landscape, a continuous change of the production processes and customer demands are taken into consideration. The current system is mainly driven by an optimized FIFO approach. It should be completely replaced by a new solution. Main objectives of improvement are different performance indicators like CT and OTD (for more information about the KPI see Section 3.2). Flexibility and adaptability of the policy to different factory contexts are the principal objectives. The following three main points are the prime focus of this thesis:

1. Creation of an appropriate representation of the facility as a simulation model:
    a) Definition of the data sources and data collection
    b) Model generation and setup
    c) Model verification and validation
2. Definition and analysis of a flexible and efficient strategy improving the facility-wide production performance:
    a) Specification of the strategy
    b) Evaluation of the strategy with help of the detailed facility model created in (1)
3. Design and implementation of a prototypical control software implementation:
    a) System design and implementation
    b) Evaluation of the practical benefit

The key method in this thesis is the discrete event simulation. It allows performance assessments of different experiments under changing conditions. The method is widely used in the semiconductor industry and commonly accepted by factory owners and managers. The simulation method also allows flexible statistical output from each experiment for a reasonable analysis and evaluation. In addition, changes of the boundary conditions can be performed and tested without influencing the real production process.

The final purpose is the introduction of a new production control system including the proposed approach to a real factory environment. It should be accepted and used as a daily tool in the manufacturing environment with positive influences of the KPI of the facility.

## 1.3 Thesis Organization

The thesis is organized in the following way. In Part I we discuss the background of semiconductor manufacturing. The first chapter of this part includes the different business models and the production process itself. In the following chapter we discuss standardized parts of the information technology infrastructure of each semiconductor facility and acknowledged problems.

Part II deals with a theoretical view on the semiconductor production process. A short introduction of common laws and theorems like Little's Law is presented. It offers a general view on simple wafer manufacturing contexts. Important factory parts influencing the performance of a facility are also introduced. In Part III the way to a detailed factory model is discussed and presented, based on discrete event simulation. This part includes different validation and verification techniques. Furthermore important parts of the factory model influencing productivity are introduced.

In Part IV the proposed control strategy is explained and evaluated. This is done with usage of the detailed facility model. Different scenarios exemplify the positive influence of the approach to various factory performance parameters.

PartV introduces the approach from the design and implementation point of view. Common programming paradigms are introduced. A prototype for daily use at the facility floor is introduced. The part is finished with an evaluation of the practical results. Finally, we present our conclusions in Part VI and give a perspective on future development and research tasks.

# 2 Semiconductor Manufacturing

In this chapter, we give a short overview of the business models in semiconductor manufacturing. In addition, we present an overview about the process of wafer fabrication and the production control system from the view of IT.

## 2.1 The Business Models

Starting with simple laboratory-like production facilities it was a long way to the fully automated semiconductor facilities of today. New customer demands force the manufacturer to develop new technologies and circuits. This process results in a continuous growth of the industry. Different strategies have to be taken into account by facility managers. Management demands like productivity or cost-of-ownership define the economic bottom line of each facility.

As introduced in Section 1.1 foundries are a common business model in today's semiconductor industry. The common definition of a foundry is the customer specific production of electronic designs and variants. If a foundry does not develop its own technology and only produces customer designs, it is also known as pure-play SF. Some large companies shift to the IDM foundry type providing some foundry services which are not conflicting with their own production portfolio.

Typically the introduction of new designs and technologies also requires new production steps and equipment. These processes are very time consuming and expensive. Common prices for state-of-the-art equipment range from some hundred thousands to billions of US-Dollars. At each fab several hundred pieces of equipment are necessary. In the last years many companies sold out their older facilities to foundry companies. No further investments in new equipment and facilities are the consequence. The foundry business can operate more cost efficient with a higher number of customers.

Besides the technological aspects, the business orientation of a foundry is important. At least three main orientations are known (see [Lo07]). A foundry can either have a product orientation, a customer orientation or a combination of both. Each of the strategies needs a customization of the end product by each customer. The customization is done either in a collaborative (in the way of individual products per customer) or adaptive (standardized products with high customization potential by end user) way. Today's foundries serve different customers from all over the world producing electronic devices for automotive, transport, health and other industry sectors. An overview of the ranking of the world largest foundries can be seen in Table 2.1.

| 2011 Ranking | Company | Region | Foundry Type | 2011 (m$) |
|---|---|---|---|---|
| 1 | TSMC | Taiwan | pure-play | 14,600 |
| 2 | UMC | Taiwan | pure-play | 3,760 |
| 3 | Globalfoundries | USA | pure-play | 3,580 |
| 4 | Samsung Semiconductor | South Korea | IDM | 1,975 |
| 5 | SMIC | China | pure-play | 1,315 |
| 6 | TowerJazz | Israel | pure-play | 610 |
| ... | ... | ... | ... | ... |
| 14 | X-FAB | Germany | pure-play | 285 |

Table 2.1: Foundry ranking (revenue) of the year 2011 (source [ELE11])

An effective production control strategy for this business must consider the special demands from native foundries like

- short time to market means low cycle times for short reaction times within the production,
- a good customer rating including parameters like quality or on-time delivery, and
- variable demands from the management regarding certain facility performance parameters like utilization of tools or fast production processes.

## 2.2 From Silicon to the Final Device

In this chapter, we introduce the whole production process from the silicon to the final electronic device. In general the semiconductor manufacturing process can be divided into five stages (see [vZ04]):

1. Material preparation
2. Crystal growth and wafer production
3. Wafer fabrication
4. Packaging
5. Electrical and final test

Normally the five stages are performed at different manufacturing sites around the world. The most time consuming production step is the wafer fabrication. This stage is the first point for improvement of performance. A wide range of different analysis is done to improve different KPI concerning this stage. In this thesis, we only refer to this step of the overall production process.

The wafer fabrication process is a very complex task. There is a huge variety of different types and classes of integrated circuits to be produced during this step. In general all circuits are made of the same few basic structures and therefore using the same basic manufacturing processes. The wafer fabrication contains of four basic processes (see [vZ04]) used in various combinations and very long sequences:

- Layering
- Patterning
- Doping
- Heat treatment

These four basic operations are introduced in the following sections. A general overview is given in Figure 2.1.

### 2.2.1 Layering

The layering operations are used to add thin layers onto the wafer surface. Different processes are used for layering. They are grown or deposited on the surface. Common techniques are oxidation and nitridation. The deposition operations can be divided into

- chemical vapor deposition also called CVD,
- evaporation,
- sputtering, and
- electroplating.

Different materials are used for the layering operations. The most common ones are silicon dioxide providing insulator abilities and metals like aluminum, gold or titanium providing conductor abilities.

### 2.2.2 Patterning

Patterning is a series of operations to remove predefined portions of the surface layers. After the operations, a pattern of the layer can be seen on the wafer surface. Two types of the removed material are defined: the hole and the island.

The patterning process can be divided into photo lithography steps and etching steps. Photo lithography steps normally consist of two steps. At first the exposure step is processed. A specific layer is transferred from a photo mask onto the top of the wafer. The process uses photo resist sensitive to a specific wavelength. Within the second step, the photo resist is developed, and the unwanted layer portions are removed. The etching steps include a removal of the developed photo resist in the unmasked areas.

Patterning is very critical within the production process. Various physical parts like transistors and capacitors are formed. The exact shapes and forms are defined by the electrical design. Errors like distortion or misplacement can change and destroy the electrical properties. Contamination during the process like impact of dust particles can cause defects. This problem is magnified by the fact that these operations are done numerous times during the production process.

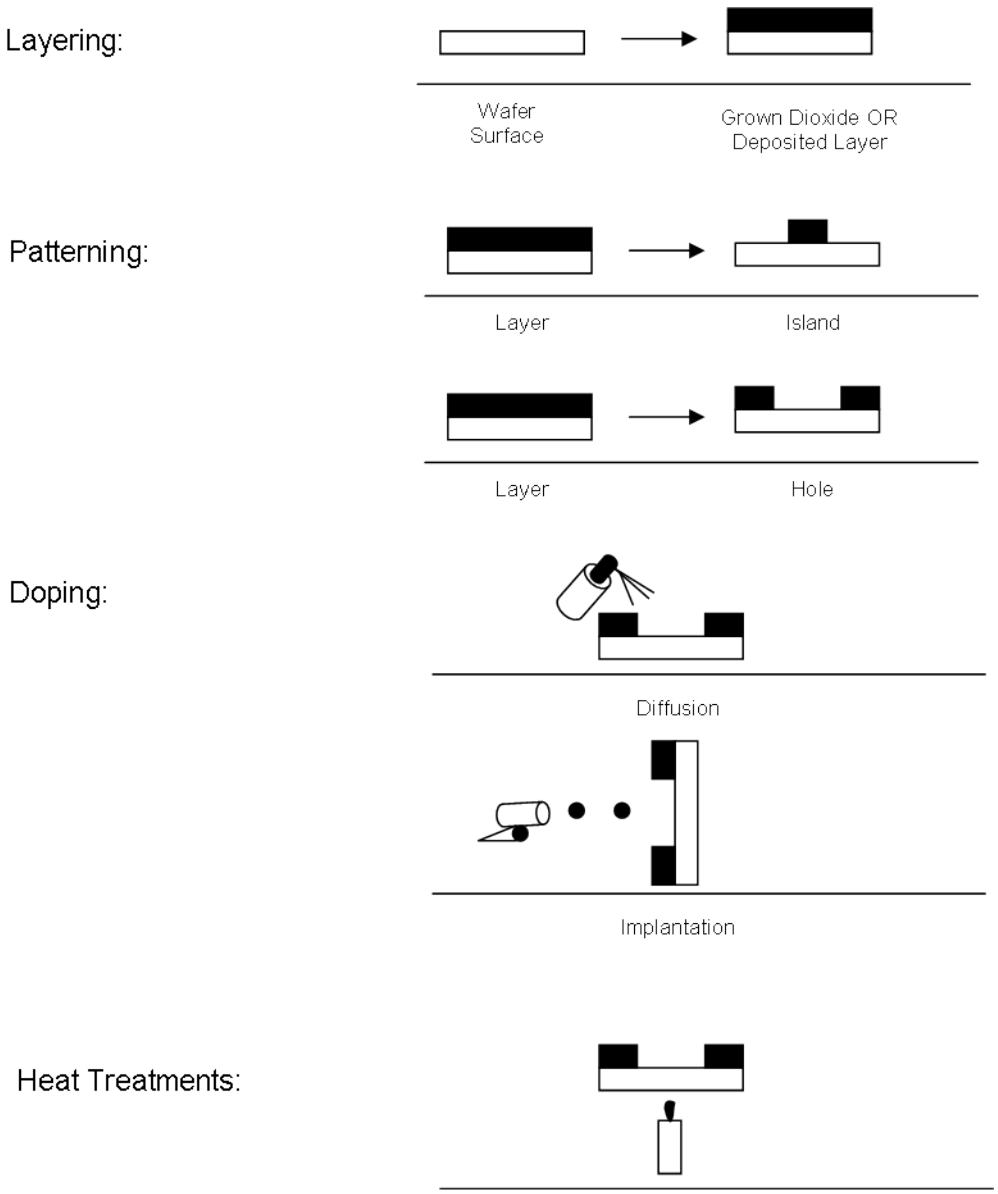


Figure 2.1: Wafer fabrication processes

### 2.2.3 Doping

Doping is a process of changing electrical characteristics of specific areas of the wafer. During the process, electrically active dopants of a specific amount are put onto the wafer surface. Typical amounts of impurities are 1 per 10,000 or 1 per 100,000,000 atoms. The process can be performed by thermal diffusion or by ion implantation.

Thermal diffusion is based on a chemical process. During the process the wafer is heated up to 1000 °C and is exposed to vapors of the proper dopant. The dopant atoms in the vapor move into the exposed wafer surface building a layer on the wafer.

Ion implantation is a physical process. In the implantation equipment, there are the wafers on one side and the dopant source on the other side. At the source end, each dopant atom is ionized and accelerated to a high speed. The high speed of the ions carries them into the wafer surface, like a ball is shot from cannon into a wall of rock.

It is possible to create areas onto the wafer that are either rich in electrons or rich in electric holes. The areas can be used to form different electrically active regions.

### 2.2.4 Heat treatment

During heat treatment operations, the wafer is simply heated up and cooled down. Typical temperatures during the process are 450 °C up to 1000 °C. Usually heat treatment takes place after ion implantation or during pattering and layering operations. The implantation of certain dopant atoms causes several disruptions to the wafer. The heating of the wafer repairs the crystal structures. Another example is the removal of solvents from the wafer with photo resistant layers interfering with the patterning.

## 2.3 IT View on Production Control

At the beginning of the integration of IT structures into manufacturing environments, manually triggered tasks were replaced by computer based services. The management of the accounting, the warehouse administration and production control are some of the tasks IT systems offer today. Production control by central systems is a young part of the historical development. New technologies and production processes force the industry to use efficient management tools.

In relation to each software product, MES solutions run through different development cycles. Until the 1990s, the integration and usage of MES solutions has grown in a steady manner. The first simple solutions only collected production data on manual terminals. In the late 1990s there are first solutions for application of advanced scheduling and planning algorithms. These abilities allow a closed steering loop, e.g., for iterative re-planning, re-scheduling or real-time dispatching. For more information see [Kle07].

### 2.3.1 The Business as IT System

A lot of companies can be defined as information consuming and preparing systems (see [Kle07]). Almost half of the total costs are caused by information tasks today.

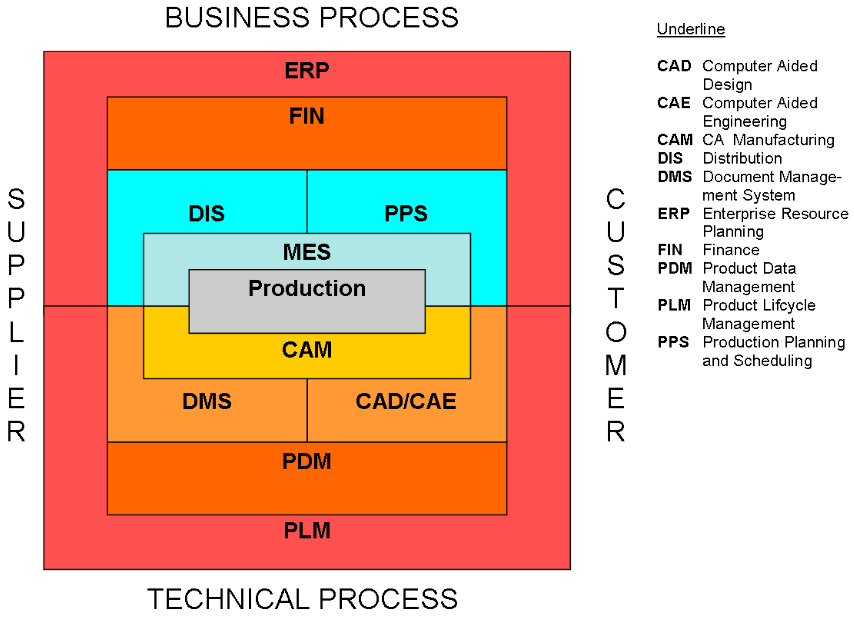

BUSINESS PROCESS
ERP
FIN
DIS
PPS
MES
Production
CAM
DMS
CAD/CAE
PDM
PLM
TECHNICAL PROCESS
SUPPLIER
CUSTOMER
Underline
CAD Computer Aided Design
CAE Computer Aided Engineering
CAM CA Manufacturing
DIS Distribution
DMS Document Management System
ERP Enterprise Resource Planning
FIN Finance
PDM Product Data Management
PLM Product Lifcycle Management
PPS Production Planning and Scheduling

Figure 2.2: Integration of the software landscape (according [Kle07])

The evolution of the information management does not take place at all parts of the production process at the same speed. Often a lag of connection and deficient information processing can be found in some areas. Written collection of data is still used today. This ineffective handling of information causes unreliable data collection and makes a reasonable production control more complicated.

Today's state of the art production control systems deal with a lot of data from other subsystems applied in the semiconductor production process. Figure 2.2 illustrates the main dependencies of the whole software landscape in a production business environment. The MES is one of the core information technology systems in a modern manufacturing environment. A large number of peripherally software solutions operate with data provided by the MES. The ERP system, which is responsible for resource planning and customer order management, is a further important element. In this thesis we use different interfaces and data from MES of a typical SF.

### 2.3.2 Structure of an MES

The architecture of modern MES is aimed at the Business Service Architecture or Enterprise Service Architecture. A special demand is the long life cycle, in which new requirements are introduced to the system regularly. A steady adaption to new requirements is also a main point in designing MES. Often proprietary systems do not have such a consistent architecture and create a higher risk from an IT view.

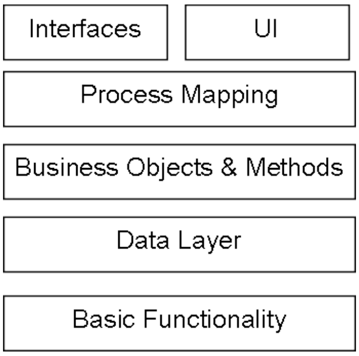


Figure 2.3: Layer of a modern MES architecture

The architecture can be divided into four main parts (according[Kle07]):

- Basic functionality: The basic functionality is a collection of methods, which is required for the operation of the system.
- Data layer: The data layer is the core data model. It is a collection of data model definitions and is responsible for storing all data which is available at the MES context.
- Business objects and methods: The business objects and methods are used for providing the functionality of the system.
- Process mapping: The process mapping represents the business logic of the applying company and offers methods and objects for secondary systems.

A vast amount of different MES products are available today. Different business areas are addressed, besides semiconductor manufacturing, the solar industry, automotive production companies and many other industrial sectors make use of modern MES.

# Part II

# Theoretical Examination of Semiconductor Manufacturing

Start by doing what is necessary, then do what is possible, and suddenly you are doing the impossible.

*(St. Francis of Assisi (1182 – 1226))*

# 3 Terms and Definitions

In this chapter, we introduce the different entities acting jointly in a system of manufacturing processes and define their relations and characteristics.

## 3.1 Production Entities

Different terms and definitions are used in manufacturing environments. The following subsections introduce the main elements of the wafer fabrication.

### 3.1.1 Lot

During the wafer fabrication process, wafers are combined in a lot. A lot is a set of wafers with the same process flow. Classic lot sizes are 24 or 25 wafers, whereas lots with smaller wafer counts are sparsely in use. Each lot is physically transported in a carrier.

### 3.1.2 Process Flow

Each lot follows a predefined route through the facility until the lot is ready for sale. Typically a lot has to pass several hundred steps. Therefore a lot can stay in processing for several months. Each process step is described by a recipe or specification. The recipe defines different parameters of the process conditions at the process equipment.

Each lot corresponds to a certain product. The product defines the electrical circuits and devices to be manufactured. Normally a huge number of the same products (chips) are found on one wafer. Each product refers to a certain technology. The technology defines the basic characteristics of the devices.

### 3.1.3 Manufacturing Operations

Each process step is characterized by a specified recipe or specification. Besides process conditions, the available workstations are defined. In general two types of process steps are available:

- Processing Operation: A process operation changes the characteristics of the wafer by means of chemical or physical processes.

- Metrology Operation: In contrast to process operations, the metrology operations are carried out to control certain parameters of the wafer during the production process. There is no alteration of the characteristics of the wafer. Metrology operations are not mandatory at all. A certain sampling process can be defined, if quality aspects allow this. It specifies which lot can skip the measurement step and which can not. For analysis reasons, a skipping probability $p_{skip}$ can be defined, where $0 \leq p_{skip} \leq 1$. The skipping probability of process operations is, of course, zero.

### 3.1.4 Equipment

The equipment (workstation) is able to perform different processing or metrology operations. There are two characteristics defining the operation of current semiconductor equipment,

- the batching ability, and
- the equipment configuration.

Often the equipment configuration has a major influence on operational decisions, even if the wafer processing requires special equipment setups. Batching is the ability to process more lots at one time with the same recipe. This is often done for long-term processes and requires an efficient strategy to form the batches. Otherwise capacity losses can occur.

In general equipments can be classified as (according [Stu09])

- single wafer tools,
- x-piece-tools,
- cluster tools, or
- batch tools.

Besides this definition, we can divide equipment types into

- fully automated,
- half automated, and
- manual systems.

#### 3.1.4.1 Single Wafer Tools

Single wafer tools process one wafer per time. Usually single wafer tools consist of three parts. The load ports are used to place the lots at the machine. The handler is responsible for moving the wafer to the process chamber from the loading port and back. The process chamber performs the processing tasks on the wafer. Figure 3.1 illustrates an example.

In a semiconductor environment, a huge amount of the equipment machinery are single wafer tools. Besides single wafer tools with one process module, tools with multiple chambers allow the processing of more than one wafer per time. These tools are called cluster tools.

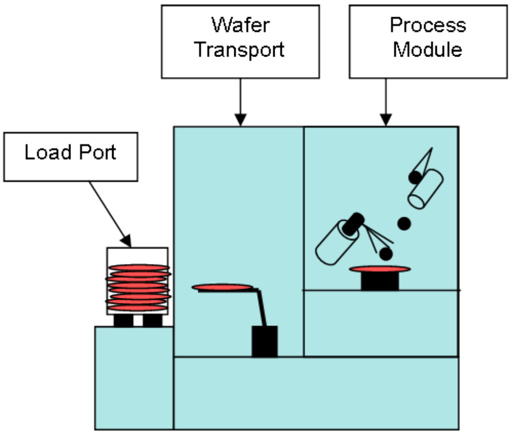


Figure 3.1: Fully automated single wafer tool

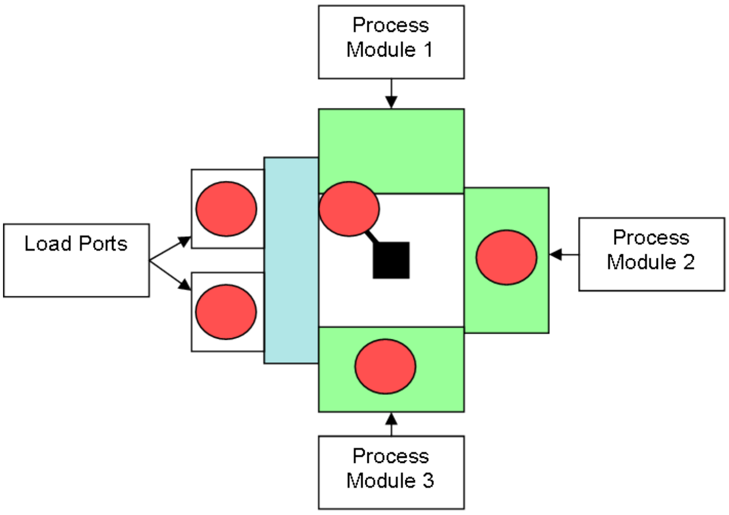


Figure 3.2: Cluster tool with two load ports and three process modules

#### 3.1.4.2 Cluster Tools

Cluster tools are a sub-category of single wafer tools where more than one wafer at a time can be processed. Cluster tools are configured in two types:

- Performing the process step multiple times with multiple process chambers of the same type.
- Integration of steps following each other in analogy to the pipelining principle of modern computer processors.

Figure 3.2 illustrates a cluster tool with three process chambers and two load ports where two lots can be placed. A special characteristic of the cluster tool is the down behavior. In all other equipment types an equipment down causes the whole equipment being unavailable. In case of cluster tools there is the possibility of a failure in only one of the chambers. So the down of one chamber does not block the whole equipment, processing is still possible. In case of cluster tools with the pipelining principle, a down can cause a problem regarding the processing continuation stops with the defect chamber.

Using cluster tools has some advantages. The application of these tools reduces transport times and total space needed in the clean room. Otherwise the variability of the whole system is increased and the internal transport system is introduced as a potential bottleneck. Capacity inequalities of sequential steps lead to unused capacity. Therefore cluster tools have advantages if used properly, but may cause capacity losses if used in a wrong way.

#### 3.1.4.3 X-Piece-Tools

X-piece-tools process batches of x wafers where x is smaller than the standard lot size. The operation performed is similar to the batch tool operation. X-piece-tools generally have several load ports for efficient process execution.

In a state of the art manufacturing system there are only a few equipment types that can be categorized into this class. A well known representative for this kind of tools is the implantation tool using sizes of 13 or 17 wafers per batch. In this case, the batch sizes are defined by the physical process itself. Due to a batch containing a lower number of wafers than the standard lot size, each lot is processed in at least two batches. This can cause different process qualities of the wafers within a lot.

#### 3.1.4.4 Batch Tools

Batch tools are able to process one or multiple lots at the same time. Figure 3.3 illustrates a tool design of a batch tool. In front of the tool there are load ports for several lots. Unlike in other tool groups it is not useful and possible to store all carriers at the load ports through the whole processing. Batch tools normally process larger counts of wafers. The carriers are loaded into an internal buffer area (yellow color) stored through the processing time. Then the wafers are loaded into the processing area for processing. Afterwards the unloading process mirrors the loading process.

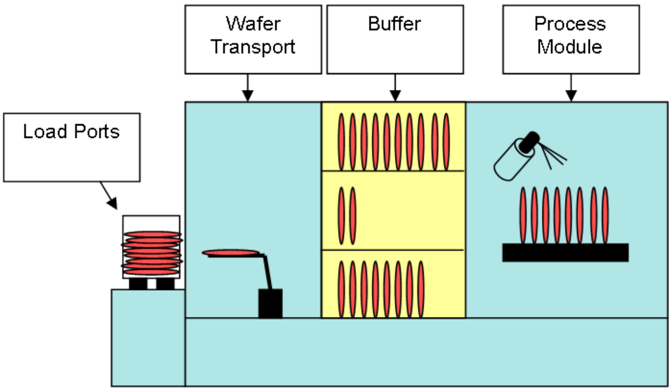


Figure 3.3: Batch tool principle

Typical process areas for batching tools are diffusion and heat treatment processes in furnaces and wet cleans in sinks. Due to long processing times in these areas, batch tools have a higher throughput than single wafer processing. However, batching tools have several disadvantages like:

- The occurrence of high loss in case of processing errors and handling problems.
- The occurrence of long handling times for loading and unloading the equipment.
- The batch building time which is required for forming batches.
- The batch dissolving time at the operations following after the batching operation. Often the operations following a batch step are single wafer operations unable to process the whole batch at the same time.

During cleaning steps there are tools available which can process lot sizes of 25 or 50 the same time. These tools are called mini-batch tools. Mini-batch tools need not to have a buffering area because of the small batch size.

#### 3.1.4.5 The Gap of Automation

In semiconductor environment, the level of automation of a tool basically depends on the age of the tool. We can divide tools into

- fully automated,
- partially automated, and
- manual tools.

Fully automated tools provide a process where no operator interaction is necessary. The interaction with the MES is provided by several interface buses. Different process parameters can be extracted and stored through the processing task.

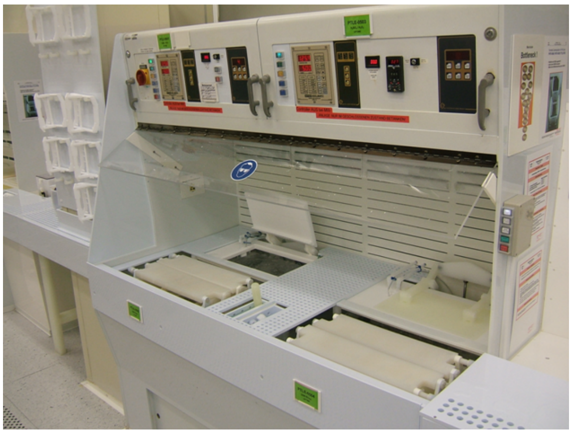


Figure 3.4: Manual wet bench

Partially automated tools can have two deficiencies regarding automation. Often manual user interaction during the process is required. Sometimes access to processing data is not possible. Examples are metrology tools, where defects are also controlled by humans via wafer picture analysis. These kinds of tools are difficult to analyze and model because of the insufficient amount of available data.

Manual tools do not have an automated process data storage and access. These tools are used completely manual by the operating staff. Known tools are manual wet sinks used for etching and cleaning operations (see Figure 3.4). A common problem of these tools is the data availability. There is no data storage and process monitoring possible. Modeling of these tool groups is limited to general assumptions like average process times (see Section 3.1.6).

### 3.1.5 Material Handling and Storage

#### 3.1.5.1 Handling and Transport

In current manufacturing sites, there are two types of transport systems available:

- Manual delivery
- Automated delivery

State of the art semiconductor facilities often have automated lot transport systems. These systems can be floor-based or overhead. Apart from the automated lot transport, smaller foundries often use manual transport. In our case we have a manual lot transport with different special properties. Manual transport can be done in different ways. The operating staff, which is also responsible for the process, performs the transport. In contrast transport operators could be available for transport of the lots through the facility. There are two kinds of transports available. Batch transports move more than one lot a time. In case of single lot transports, only one lot a time is moved.

Batch transports are often used for moving multiple lots over longer distances. Single lot transports are often used within a cluster for shorter distances. At batch transport operations additional waiting times occur. Batch transports are performed not every time a lot arrives, rather than waiting until the transport vehicle is filled or a maximum waiting time has passed.

#### 3.1.5.2 Storage

In a semiconductor facility, there are different types of storage available. The storage is used to place carriers while waiting for the next processing step. In general there are three different types:

- Stockers are common storage places, having a capacity of up to several hundred carriers.
- Under-track or side-track storage provides storage for at least one carrier.
- Storage racks are used in a manual transporting environment allowing the storage of several carriers. Nitrogen storage racks are used in areas where time bounds between operations exist. If a lot is going to violate a defined maximum waiting time between two operations, it can be put in such type of storage preventing unwanted chemical reactions.

### 3.1.6 Human Influence on Manufacturing

This chapter gives a brief overview about human influence on manufacturing illustrated by some examples. The human influence at manually driven facilities is still massive, especially in regard to transport and process tasks.

#### 3.1.6.1 Self Interest and Diversity

Human behavior in factory environments is one of the main sources of variability. Looking at different stages of the factory process, human decisions may not be optimal from a global point of view. Hopp and Spearman [HS01] proposed, that "**People, not organizations, are self-optimizing**". With this statement, they suggest, that individuals make their decisions according to their motives, preferences and goals. In the industry, a variety of examples is known where human act optimal from their point of view, but not optimal from global point of view. For example, extra pay for local targets like processed wafers is one main motive. But this can degrade global performance parameters by processing the lots with the same recipe rather than processing more important lots e.g. regarding due dates.

Unlike linear programs or other optimization techniques, humans acts on the basis of their experience and knowledge. This leads us to the next proposal: "**People are different**"([HS01]). In factory environments, we often try to generalize employee behavior. The proposed statement has some impact on the manufacturing floor. Operators work at different speed and motivation. Manager interacts in different ways with their staff.

In this field humans can have in fact an (uncertain) influence on factory performance parameters. Therefore operators must be supported by IT systems providing useful solutions for operations to decrease the influence of local decisions.

#### 3.1.6.2 Planning and Responsibility

There is a range of fields in factory environments, where the human element causes divergence between planning and reality. For instance, historical data is used to define the operator capacity and speed. Both, the data acquisition and the operator difference cause a deviation from model to reality depending on the point of view.

In semiconductor manufacturing with a large amount of operator influence, several systems are applied to ensure reasonable operator behavior. For example, each operator can have several qualification stages. The stages describe the ability to interact and work with a certain equipment or equipment group. Operators are not allowed to interact with equipment they are not qualified for. Another classification is the area of work of each operator. Some operators work on small areas containing a few tools with higher throughput, other operator can have a more general area of possible equipment interactions (even larger clusters).

#### 3.1.6.3 Summary

Hopp and Spearman ([HS01]) summarized six main points describing the field of human interactions in a factory environment:

1. ”People act according to their self interest.” People’s actions are a consequence of their own subjective incentive.
2. ”People differ.” Each human has different skills, talents, interests and motivations.
3. ”Champions can have powerful positive and negative influences.” Ideas supported by highly respected persons in the management area can have a big impact on manufacturing.
4. ”People can burn out.”
5. ”There is a difference between planning and motivating.” There can be a gap between the current situation and the historical data itself when using historical data for capacity or reliability purposes .
6. ”Responsibility should be commensurated with authority.”

### 3.1.7 Interaction of the Basic Entities

In a SF, there are complex interactions between the entities introduced in Section 3.1. Figure 3.5 exemplifies a re-entrant flow of lots in a semiconductor facility. A typical wafer fabrication process flow has about twenty to thirty processing layers with a total number of about 300 to 500 processing steps. At each layer there is at least one lithography operation.

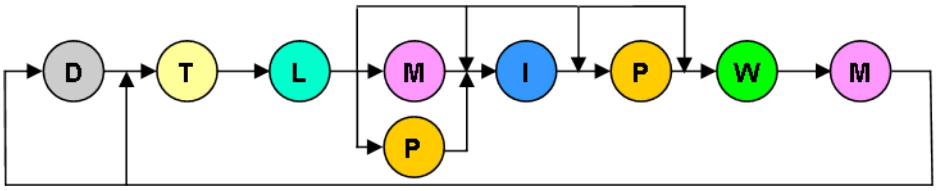
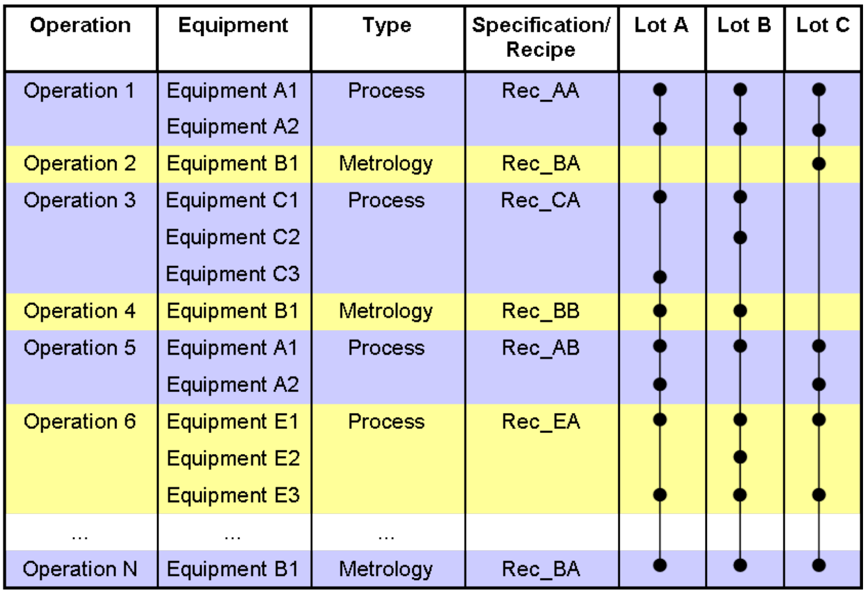


Figure 3.5: Example re-entrant material flow in a semiconductor facility (D: Diffusion, T: Thin Film, L: Lithography, M: Metrology, I: Implant, W: Wet bench, P: Plasma Etch/Strip)



| Operation | Equipment | Type | Specification/ Recipe | Lot A | Lot B | Lot C |
|---|---|---|---|---|---|---|
| Operation 1 | Equipment A1 | Process | Rec_AA | ● | ● | ● |
| | Equipment A2 | | | ● | ● | ● |
| Operation 2 | Equipment B1 | Metrology | Rec_BA | | | ● |
| Operation 3 | Equipment C1 | Process | Rec_CA | ● | ● | |
| | Equipment C2 | | | | ● | |
| | Equipment C3 | | | ● | | |
| Operation 4 | Equipment B1 | Metrology | Rec_BB | ● | ● | |
| Operation 5 | Equipment A1 | Process | Rec_AB | ● | ● | ● |
| | Equipment A2 | | | ● | | ● |
| Operation 6 | Equipment E1 | Process | Rec_EA | ● | ● | ● |
| | Equipment E2 | | | | ● | |
| | Equipment E3 | | | ● | ● | ● |
| ... | ... | ... | | | | |
| Operation N | Equipment B1 | Metrology | Rec_BA | ● | ● | ● |

Figure 3.6: Example for routes of lots of different products

In a semiconductor facility, there are up to 300 different tools. These tools are repeatedly used to fabricate the different layers on the wafer surface. In historically grown facilities there are tools with different age, but the same purpose. In general newer tools have a better process stability and sometimes a faster process speed. Therefore, at each recipe the available tool set influences the KPI of the facility.

Figure 3.6 illustrates the main characteristics of a semiconductor production flow. The individual lots have a predefined production route. Metrology operations can be skipped if the circumstances allow it, such as good process stability.

At tools, lots are processed in different order. Figure 3.7 represents this behavior. In the case that all tools are busy or unavailable, an incoming lot joins the equipment queue. The equipment queue can be divided into several lot groups having the same recipes. Whenever a tool becomes ready to process (and other process resources like operator are available), the next lot or set of lots with the highest priority will be processed. After process completion, the lots move to the next process step. For each lot, two types of

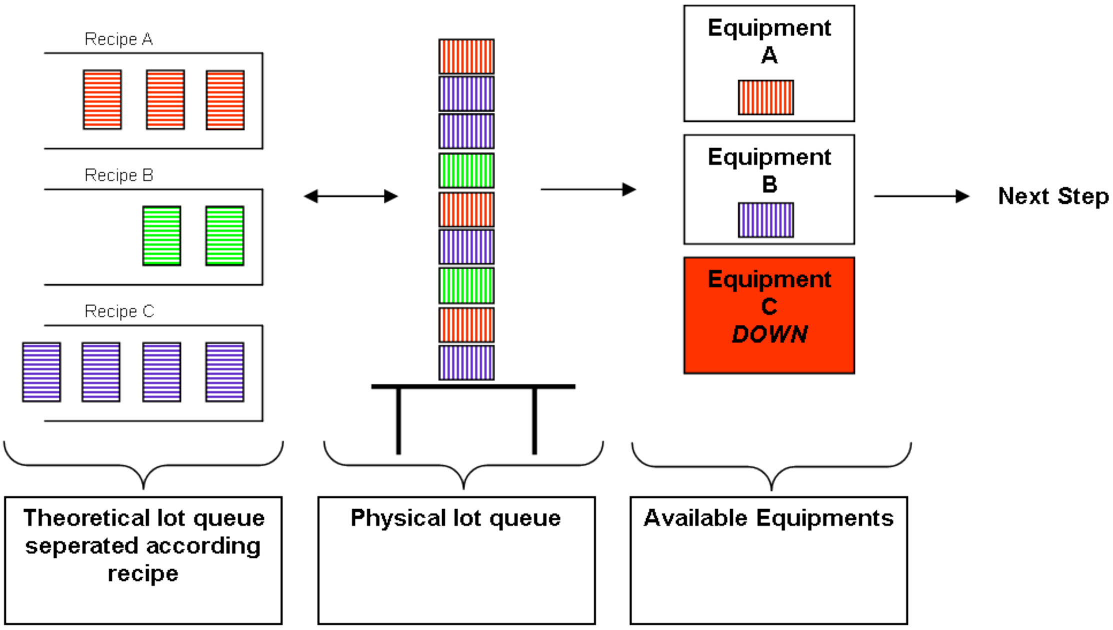


Figure 3.7: Operating behavior at a certain stage

times occur. Besides the necessary processing time defined by the process flow, unwanted waiting times occur. The ideal semiconductor manufacturing has a waiting time close to zero for each lot, which is impossible for normal loads in today's modern facilities.

## 3.2 Factory Characteristics

In semiconductor manufacturing, a number of different KPI are used. In general the performance of a semiconductor facility is not evaluated with a single measure. Different definitions are available. The following sections describe the most important ones.

### 3.2.1 Cycle Time

The $CT$ is the absolute time a lot spends in a facility including the processing and waiting times. With other words, the $CT$ is the sum of all times a lot spend at each stage $c_i$ excluding the shipping operation

$$CT = \sum_{i=1}^{N} c_i \tag{3.1}$$

The cycle time efficiency $E_{CT}$ is the ratio of the best possible $CT_0$ to the actual $CT$:

$$E_{CT} = \frac{CT_0}{CT} \tag{3.2}$$

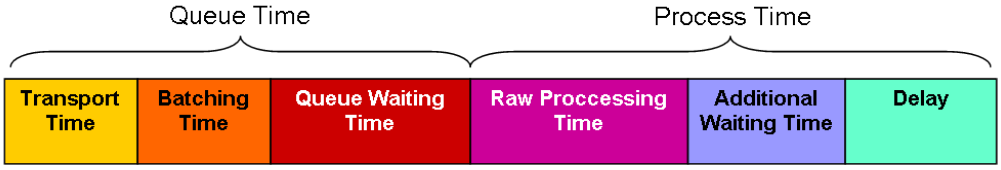


Figure 3.8: Components of the processing step time

The *CT* consists of several components. Each processing step time $c_i$ can be divided into

- Process time $c_{p_i}$
  - Raw processing time $c_{pr_i}$ , denoting the time the lot spent at a processing resource during process at step $i$.
  - Additional waiting time $c_{pw_i}$, where the resource is able to process the next lot, but the former processed lot itself has to wait due to process reasons.
  - Delay $c_{pd_i}$, caused by overlapping processing of consecutive lots.
- Queue time $c_{q_i}$
  - Transport time $c_{qt_i}$, caused by transportation of the lot between two processing steps.
  - Batch building time $c_{qb_i}$, which is caused by waiting for other lots building a batch.
  - Queue waiting time $c_{qw_i}$, which is denoted by the time the lot spend waiting until process start.

The resulting processing step time $c_i$ is defined as (see Figure 3.8):

$$c_i = c_{p_i} + c_{q_i} = (c_{pr_i} + c_{pw_i} + c_{pd_i}) + (c_{qt_i} + c_{qb_i} + c_{qw_i}) \tag{3.3}$$

The raw processing time $c_{pr_i}$ determines the theoretical performance limit for an equipment. For each equipment type, individual processing time types can be defined. Often lots or batches are processed by semiconductor equipment in an overlapping fashion ($c_{pd_i} > 0$). Figure 3.9 introduces an example.

The different equipment types (see Section 3.1.4) cause different processing time behavior for one lot. At batching operations, the processing time is independent from the number of wafers processed. In case of single wafer tools, the processing time can be assumed as linear over the lot size used. The smaller the lot size, the smaller the processing time. X-piece-tool processing time increase at multiples of x in time. In case of a 25 wafer lot and a batch size of 17 wafers, two operations are required. Cluster tools are not easily accessible. Different factors such as the current utilization affect the processing time behavior. There is a qualitative illustration of the dependencies in Figure 3.10.

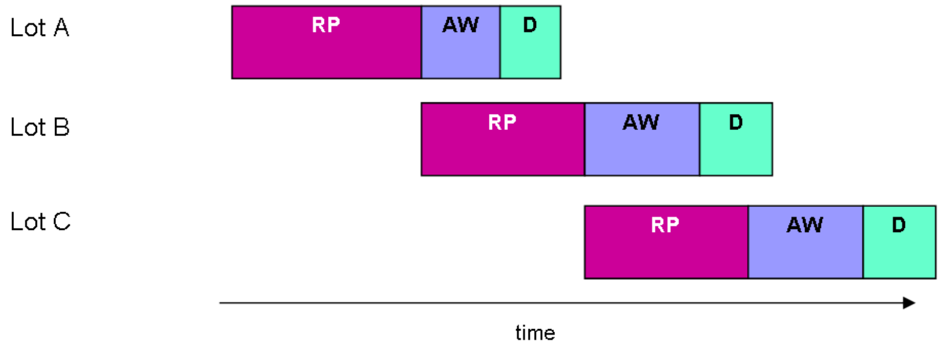


Figure 3.9: Cascading of processing steps on one semiconductor equipment (RP: Raw Processing, AW: Additional Waiting, D: Delay)

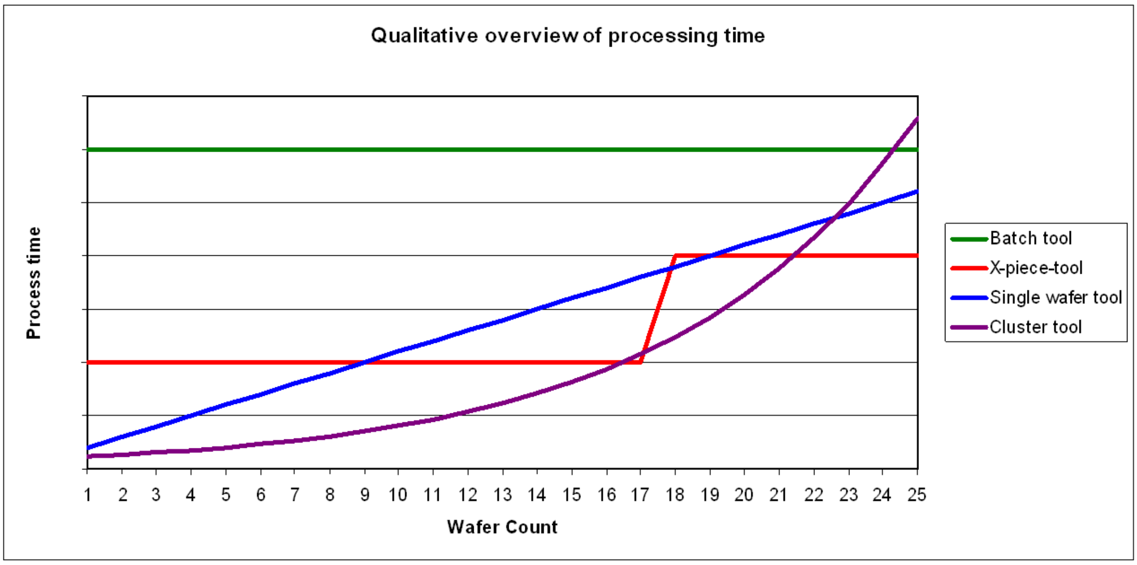


Figure 3.10: Qualitative overview of processing time behavior under different tool types and lot sizes

### 3.2.2 Cycle Time per Mask Layer

The $CM$ is often used in factory environments. While the $CT$ defines the absolute time a lot spends in factory, the $CM$ defines the mean time a lot spends per mask layer in the facility:

$$CM = \frac{CT}{M_{Mask}} = \frac{\sum_{i=1}^{N} c_i}{M_{Mask}} \tag{3.4}$$

The $CM$ allows a better comparison of different products at a SF. Products have different numbers of mask layers to be processed. With a higher complexity of a product, the number of mask layers increases.

### 3.2.3 Work in Process

The $WIP$ defines the number of wafers $w_0$ at a certain position or area in the facility. There are at least three points of view determining useful work in process counts:

- The facility level determining how many wafers are currently between start and finish
- The area level determining how many wafers are currently in a certain area or cluster (e.g., lithography)
- The equipment level, determining how many wafers are currently in the queue of the equipment

### 3.2.4 Throughput

The $TH$ is the quantity of wafers manufactured per time unit:

$$TH = \frac{N_{Wafer}}{t} \tag{3.5}$$

The demand $D$ of a facility defines the number of wafers required per time to fulfill all customer orders. The throughput efficiency $E_{TH}$ can be defined as:

$$E_{TH} = \frac{\min\{TH, D\}}{D} \tag{3.6}$$

If the throughput is greater or equal than the demand the throughput efficiency equals one.

### 3.2.5 The X-Factor

The $XF$ describes the relation between cycle time $CT$ and the theoretical raw process time $TR = C_{PR} + C_{PW} = \sum_{i=1}^{N}(c_{pr_i} + c_{pw_i})$. The X-factor is defined as

$$XF = \frac{CT}{TR} = \frac{CT}{\sum_{i=1}^{N}(c_{pr_i} + c_{pw_i})} \tag{3.7}$$

### 3.2.6 Delay, Due Date and On-Time Delivery

The $OTD$ describes the compliance to the due dates for orders. The absolute on-time delivery $OTD_0$ is defined as

$$OTD_0 = \frac{M}{N} \tag{3.8}$$

where $M$ is the number of lots finished within the due date and $N$ is the number of all finished lots. The tardiness $TD$ is defined as

$$TD = \frac{\sum_{i=1}^{N} FIN_i - DUE_{i,}}{N} \tag{3.9}$$

where $FIN_i$ is the finishing time of lot $i$ and $DUE_i$ is the due date of lot $i$. Often the on-time delivery is generated by applying weekly buckets:

$$OTD_{wb_k} = \frac{\sum_{i=1}^{N} L_i}{N} \text{ where } L_i = \begin{cases} 1 & T_{Lower} < T_{Lot_{i,Target}} \leq T_{Upper} \\ 0 & otherwise \end{cases}$$

with definition of the range $T_{Lower}$ and $T_{Upper}$. Common ranges are

1. $T_{Upper} = 0$ which means all lots within the due date,
2. $T_{Lower} = 0$ and $T_{Upper} = 7d$ which means all lots within a delay of one week,
3. $T_{Lower} = 7d$ and $T_{Upper} = 14d$ which means all lots within a delay of two weeks,
4. $T_{Lower} = 14d$ and $T_{Upper} = 21d$ which means all lots within a delay of three weeks, and
5. $T_{Lower} > 21d$ which means all lots with a delay greater than three weeks.

### 3.2.7 Utilization and Availability

The availability of the equipment is an important performance parameter. Equipments are not available the whole time for processing operations. Several activities reduce the total equipment availability like

- planned maintenance activities,
- unplanned equipment failures, and
- control and adjustment operations.

The total availability $A_0$ of a equipment can be described as

$$A = \frac{T_{Avail}}{T} \tag{3.10}$$

where $T_{Avail}$ is the total time the equipment is able to process and $T$ is the time span of the given period. The available time of a equipment can be divided into idle and processing time. The utilization $U$ of a tool can be defined as

$$U = \frac{T_{process}}{T_{process} + T_{idle}}$$

Tools with a high utilization generally cause a lot of waiting time and are defined as bottlenecks (see Section 5.2.1.1). A utilization of 100% is only possible in ideal production lines, but not in real SF.

# 4 Fundamental Relations

In this chapter, we describe the most common basic rules for semiconductor environments.

## 4.1 Little's Law

Little's Law is one of the most fundamental rules in manufacturing systems. It describes the relation between the $WIP$, the $TH$ and the $CT$. The average $WIP$ of a stable manufacturing system is equal to the average $TH$ multiplied by the average $CT$:

$$WIP = TH * CT \tag{4.1}$$

The assumption made for this equation is a stationary system. Of course, real production environments are rarely stationary. With usage of a infinite observation time $t_o$ this law also can also be applied to real production systems:

$$WIP = \begin{cases} TH * CT & \text{stationary} \\ TH * CT || t_0 \rightarrow \infty & \text{not stationary} \end{cases} \tag{4.2}$$

For the most practical issues (which means not the infinite case), this law is a good approximation. Some exceptions must be mentioned, where the approximation can not be applied. Huge changes in the product portfolio or factory start-ups cause a high degree of variability. In this case the rule can not be applied.

## 4.2 Relations between Important Factory Characteristics

In a factory environment, one can assume a certain correlation between different factory characteristics. In this section we assume an ideal production line with zero variability and the following parameters:

- Throughput $TH_0$: $TH_0$ is the maximal theoretical throughput that can be achieved in the facility. It is defined by the maximal bottleneck[1] throughput.

- Raw Process Time $T_0$: The raw process time is the sum of all process times of each operation in the line (see definition in Section 3.2.1). Waiting times are not considered here.

[1] The bottleneck in a facility is the equipment group with the highest utilization in a defined long term time frame. For more information see Section 5.2.1.1.

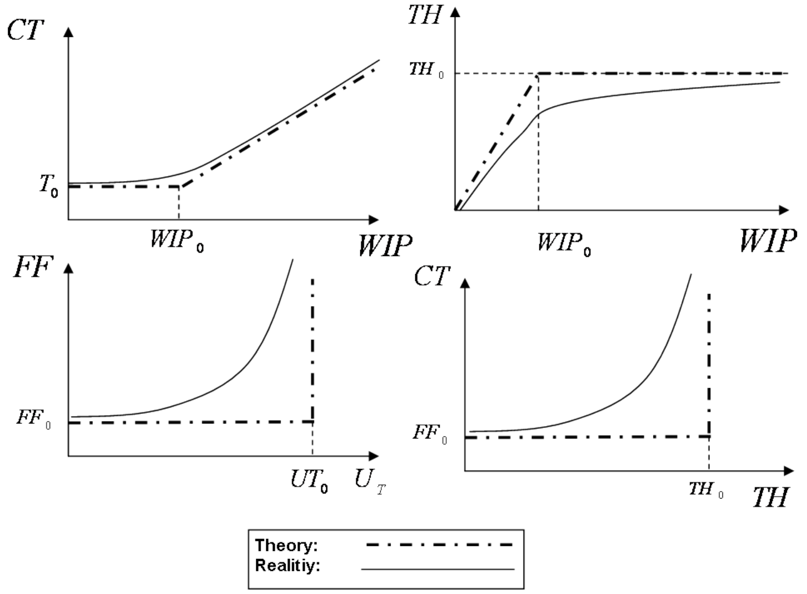


Figure 4.1: Relations between important factory characteristics

- Critical maximal $WIP_0$: The critical maximal $WIP_0$ is defined as the $WIP$ level which is needed to reach the maximal line throughput $TH_0$. It can be calculated according equation 4.1:
$$WIP_0 = TH_0 * T_0 \tag{4.3}$$
- Critical X-factor $XF_0$: $XF_0$ is the largest $XF$ that can be achieved in a factory at stable state. The stable state is defined by a arrival rate $R_a$ which is less or equal to the maximal throughput $TH_0$. In the case of a ideal production line the equation is reduced to $XF_0 = 1$. In this case the bottleneck equipment is utilized with the maximal utilization $U_{max} = 1$.

For the ideal production line the KPI are illustrated in Figure 4.1. In this case the parameters can reach the maximum simultaneously. Looking at real world examples, of course, it is not possible. It is obvious that a reduction of the $TH$ causes a reduction of the other parameters. But in fact a manager is tempted to run a facility at the maximal edge of utilization. One reason is that there is no cost decreasing in reducing throughput (assuming no change of the facility environment). A further way is to reduce the theoretical cycle time. This is very difficult because of physical and chemical processing demands. Therefore the variability is one of the interesting points for reduction. In general there are many reasons causing variability in factory environments with a negative influence on factory performance. In the next section, we will discuss the role of variability in factory environments.

# 5 Variability at Semiconductor Manufacturing

In this chapter we give an overview about variability characteristics in factory environment. In addition important points of performance losses are discussed.

## 5.1 Different Views on Variability

We have to ask ourselves if variability is always bad or sometimes good? Hopp and Spearman [HS01] proposed different points of view. For manufacturing, variability has in general a bad influence on the performance characteristics. The reasons are described in the following sections. In other cases variability can have positive effects. For a short illustration we use the following example from history as described in [HS01].

**Example (source** [HS01]**):**
Henry Ford was a prime example for reducing variability in the car factory environment. The only available color was black, and there were only small changes within the models. He standardized the production process and tried to keep each operation as simple and efficient as possible. Therefore the cars became affordable for the general public. Later on General Motors developed into an important competitor. GM offered a much greater product variety. That was one of the chief reasons that Ford nearly went bankrupt. Naturally the GM production processes and systems included a much higher variability because of the greater product portfolio. The main reason for this development was the point of view. Both companies were not on the market to offer variability reduction solutions, but to make a good return on investment. If an increased product portfolio will raise the revenue made by a company that can also be a good strategy from a business point of view. From the manufacturer point of view, it would not.

There are known potentially good variability examples. Often they are located in the business environment rather than the factory environment. Besides the product variety, changes in technology or demand can elevate a company in a better market position, also affecting the manufacturing sites.

But what is variability? Hopp and Spearman defined it as a ”quality of non-uniformity of a class of entities”. In manufacturing systems, there is a uncountable variety of different examples for variability. Physical aspects like process times, quality of products, temperatures or material properties are known factors. There is a close association between variability and randomness. In fact they are not identical. We can divide variability into two groups:

- Manageable variability
- Random variability

The first type is a direct result of decisions made, like the definition of the produced goods in a factory. Production control strategies can also be dedicated to the manageable variability causes. Many events in a factory environment can not be controlled. These causes of variability can be clusterd into the random type. Material properties, different customer orders or machine outages can not be governed by the factory management. Both types of variation have a negative impact on factory performance outcomes.

In conclusion, Hopp and Spearman proposed a general law describing the influence of variability on the manufacturing floor in a few words: **"Increasing variability always degrades the performance of a production system"**([HS01])**.**

## 5.2 Major Sources of Variability in Semiconductor Manufacturing

### 5.2.1 Equipment

#### 5.2.1.1 The Bottleneck

Bottleneck tools are pieces of equipment with the highest utilization. Changes in the bottleneck control strategy can have a tremendous impact in positive, but also in negative direction. Bottlenecks can be divided into two groups:

- Static bottlenecks
- Dynamic bottlenecks

Static bottlenecks are tools having a high utilization over a long period of time. Dynamic bottlenecks are tools having a huge utilization temporarily. Dynamic bottlenecks move around the facility over the time, which often occurs during of a huge work in process imbalance.

#### 5.2.1.2 Down and Maintenance Events

A massive issue in semiconductor manufacturing are the downtime characteristics of semiconductor equipment. In general, the availability of semiconductor equipment is very low in comparison to other industries. The availability of each tool has a big impact on the factory performance. In factory environment different preventive maintenance policies could be used. In case of a strict reduction of the maintenance activities, an equipment tends to have more unplanned failures causing a high variability and capacity loss. In case of a high number of planned maintenance activities, the unplanned failure rate is lowered by detecting problems before they occur. But there is an additional capacity loss. The qualitative behavior is illustrated in Figure 5.1.

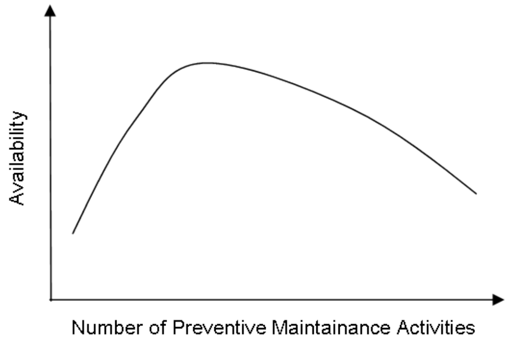


Figure 5.1: Availability in contrast to the numbers of planned maintenance activities

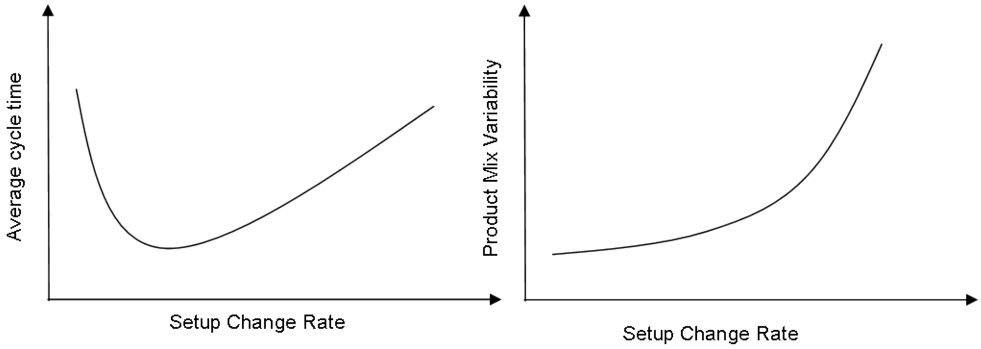


Figure 5.2: Setup rate dependencies

#### 5.2.1.3 Sequence Dependent Equipment Setups

Some types of equipment require a setup for processing lots of a recipe family. The setup causes a down time of the equipment with capacity loss. In addition the order of the lot processing is affected. Lots are processed requiring the same equipment setup at the operation. The other lots have to wait. That causes additional waiting times and variability in the lot order.

The setup change rate depends on the variability of recipes processed on the tool. Some important representatives for equipment setups are

- implantation equipment changing the dopant sources,
- stepper equipment changing the reticles or masks, or
- metallization operations with change of the applied material.

There are also pieces of equipment available that process multiple wafer sizes. In our case, there are combined 6 and 8 inch tools. A switch requires a setup. Figure 5.2 illustrates the qualitative behavior of the setup rate at an operation in contrast to the resulting cycle

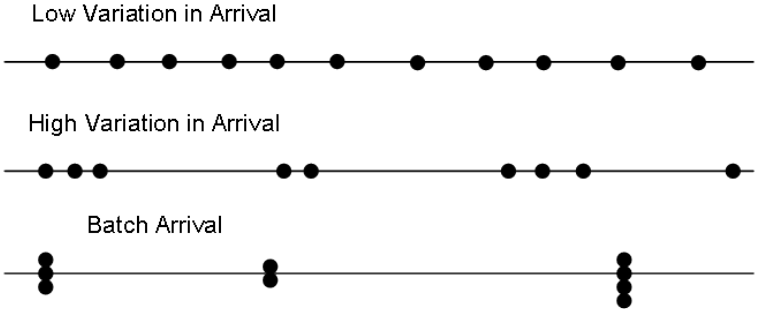


Figure 5.3: Different arrival types causing different amount of variation

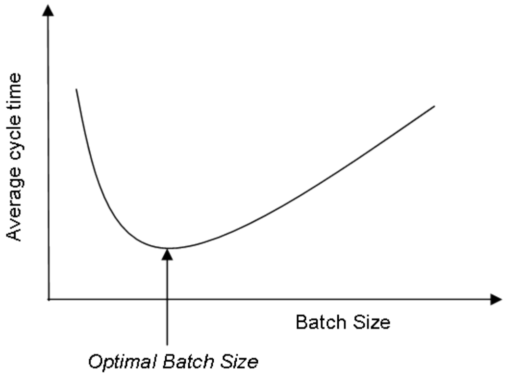


Figure 5.4: Average cycle time versus batch size

time and the dependency of the setup rate from the product mix variability (under the assumption of a sufficient number of lots in the queue of the equipment).

#### 5.2.1.4 Batching

Batching operations cause lot arrival variation at the following processing steps and unwanted batch building times (see definition of *CT* Section 3.2.1). The lots at a batching operation are processed and released together. Often the following operations are not batching, but single wafer operations. At these stages, lot batch arrival causes a higher variability. Figure 5.3 illustrates the different arrival types.

The first element displays a very homogeneous arrival of the goods having a low variation. In contrast, the second element illustrates an even higher variation in arrival. This can cause a waste in capacity. The downstream equipment may be idle because of an empty queue. Batch arrivals even have the highest variability and cause a fluctuating material flow. The next example introduces the batching process from an other point of view (adapted from [HS01]):

**Example:**

We assume that a forklift brings 20 goods per work shift to a workstation. Each work shift has a duration of 12 hours. In this situation, the arrivals occur without any randomness. A possible interpretation of the variability of this example would be zero, also the coefficient of the variation can be expected to be zero.

Now we switch the perspective from the outside to the individual goods of the batch. The interarrival time $t_i$ of the first job in the batch is 12 hours. The other 19 jobs have an interarrival time of zero. Now we can calculate the mean time between the arrivals $t_a$ to $t_a = 12h/20 = 0.6h$. With the mean value it is possible to calculate the variance of these times:

$$\sigma_a^2 = \sum_{i=1}^{N} \frac{t_i^2}{N} - t_a^2 = \left[\frac{1}{20} * 12^2 + \frac{19}{20} * 0^2\right] - 0.6^2 = 6.84$$

Therefore the coefficient of variance $c_a^2 = \frac{6.84}{0.6^2} = 19$. Which value is correct, $c_a^2 = 19$ or $c_a^2 = 0$? There is no precise answer. The batching causes two different effects. The batching itself is one reason and can be described as an inefficient control. The second reason is the variability caused by batch arrivals.

Besides the fluctuating lot arrivals, the batch size has a huge influence on the KPI. Figure 5.4 illustrates a qualitative overview of the relation between batch size and cycle time. In case of a large batch size, the tool has to wait a long time until the batch is full and the process can be started. Otherwise if the batch size is too small, the throughput of the batching operation is very low. The reason is the common long processing time of batch tools.

### 5.2.2 Production Process

#### 5.2.2.1 Operator Variation

Human influence can be a major cause of variation (see Section 3.1.6). Nevertheless, the influence is lower in a fully automated factory than in a non-automated factory with operator interaction. The availability of the operator is variable, depending on holidays and shifts. Different breaks, work schedules and priorities result in a high variation. The operator experience and system knowledge also affect his work. An outcome could be different processing times at the same process for different operators.

#### 5.2.2.2 Scrap and Rework

Scrap and Rework cause variability and additional costs and reduce capacity. Rework is possible in case of faulty process operations like layering. If a wafer is damaged in a way that rework is not possible, it must be scrapped. In this case, a new lot is started into the facility with a very high priority to replace the scraped wafers.

#### 5.2.2.3 Product Mix

The product mix of a facility, even a foundry, is a root cause of variability. Often the routes and process specifications differ significantly. In fact, the product mix in a SF is not stable over time. Different orders and demands lead to a higher variability in processed products.

#### 5.2.2.4 Re-entrant Material Flow

The massive re-entrant material flow in semiconductor manufacturing also leads to more variability in the lot arrivals at the different process stages. Compared to a flow line without re-entrancy, the variability is higher because of no indirect leveling by a previous workstation. The variability of one workstation can affect a number of other pieces of equipment. This is called flow variability.

#### 5.2.2.5 Time Bound Sequences

Maximal time windows between operations are common in semiconductor manufacturing. There are definitions for a maximal waiting time for a lot until it must be processed at the current step. This time window exists for technical and physical reasons. At time bound sequence steps, the wafer surface is affected by native oxidation and contamination. Unwanted formations of undesirable connections between conductive layers are possible. If a sequence is violated, additional capacity is needed for lot rework or scrap. In some facilities there is a restricted number of nitrogen storing places for reducing unwanted oxidation processes (see Section 3.1.5.2).

### 5.2.3 Dispatching and Scheduling

Dispatching and scheduling methods are used to determine the lot sequences at different equipments:

- Scheduling defines a lot schedule for the equipment some time in advance (in general non-real time).
- Dispatching defines a sequence of lots at the equipment in real time.

Scheduling and dispatching are often performed in combination, planning the sequences at the equipments and then, when a new process is started, choosing the right lots for the process. For SFB, the use of dispatching is more common. The complexity of the production processes makes planning very difficult. At human dominated production processes, the influence of stochastic events like equipment failures or operator variance makes current production plans invalid at short time frames. However, some applications of scheduling can be found e.g. in [WQW06, OP09]. Dispatching and scheduling activities are generally done after the detailed production and capacity planning in a SF. They are the last steps in defining a reasonable sequence of lots for an equipment. Figure 5.5 illustrates the general work flow of the lot sequencing.

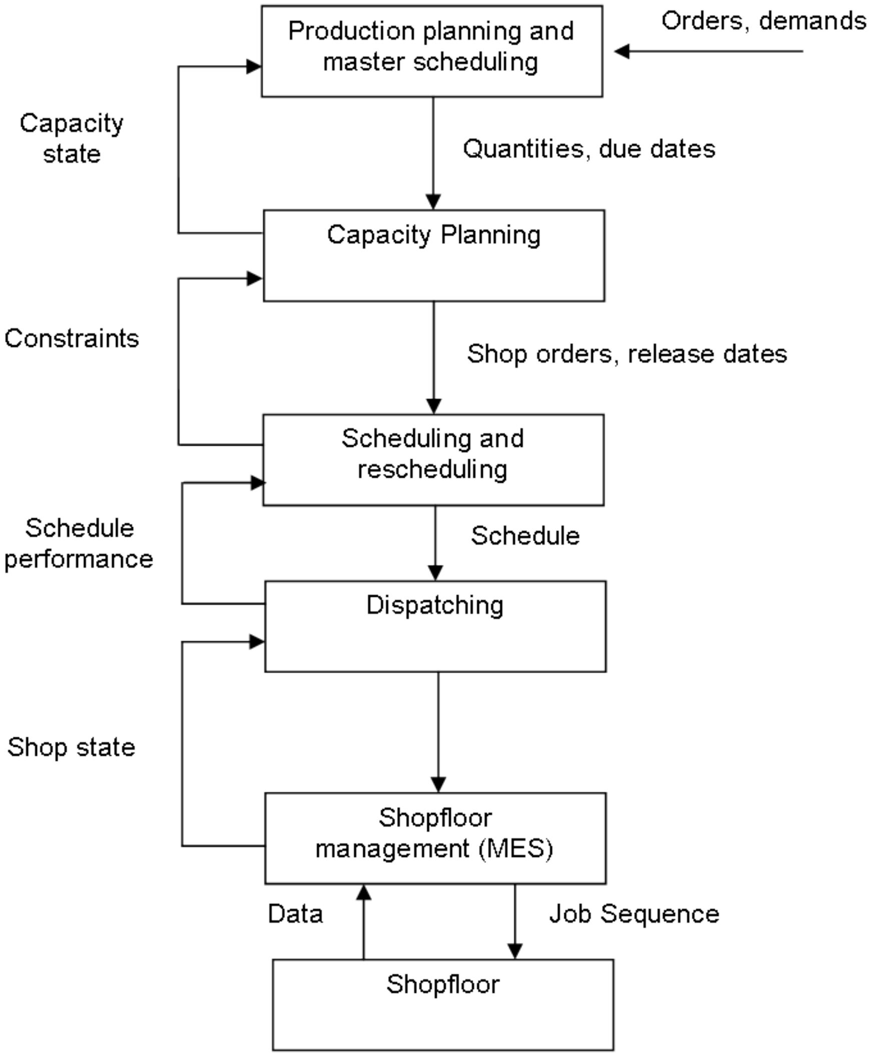


Figure 5.5: Flow of the information in a manufacturing system (source [Pin02])

#### 5.2.3.1 Dispatching

Dispatching approaches are used to calculate lot priorities to define the process order. Simple approaches like FIFO or SPT take only one criterion into account (e.g., see [Ros01]). In semiconductor environment, not only one criterion is sufficient. There is a wide range of more complex dispatching rules available. Depending on the rule specification, Panwalker and Wafik [PW77] categorize rules into:

1. Rules combining several criteria like critical ratio or the apparent tardiness cost rule.
2. Multilevel rules sorting the lots according to several criteria in different steps.
3. Conditional rules optimizing different criteria with dependence on a specific property (e.g., setup rules or batching rules).

For priority determination, different objectives are taken into account. Some of these objectives could be

- short queuing and waiting times,
- on-time delivery,
- short cycle times, or
- high throughput.

The importance of the different objectives and the chosen rules differs depending on the fab profile. Memory facilities with a low variation in the product mix might rather target at a short cycle time whereas a foundry with a higher product variation tends to focus on on-time delivery. Besides the global optimization targets, local targets could be optimized. For example, setup rules for minimization of the total number of setup changes at an equipment can be assigned to this group.

In general the objectives of the dispatching policies are not independent. For example the attempt to shorten cycle times conflicts with the need of an acceptable on-time delivery. There is a need to find a balanced compromise between these objectives. For this, every SM has its own recipe.

A typical dispatch rule applied in the semiconductor manufacturing is the ODD rule, with focus on on-time delivery. It optimizes the local due date of the lot at each stage. But cycle time is not considered. In [MS99] Mittler and Schoemig introduced several common dispatching rules applied in the semiconductor manufacturing. The success of each dispatching rule depends on the field of application. In a modern semiconductor facility, a production without a reasonable dispatching policy is not possible anymore due to different demands from management and customers. For more information about the dispatching policies, see Section 11.1.

#### 5.2.3.2 Lot Priority Classes

Different priority classes for lots are used in general. These priorities are assigned manually to a small share of the total number of lots in the facility. The reason for high priorities is to provide a very short cycle time for critical lots. Critical lots could be lots with high order priority by customers, qualification lots, or development lots for process improvements. Some operational aftermaths of specified priority classes could be:

- Let a tool or load port idle which follows next on the current route for an immediate start of the process at the next step.
- Change the setup state of a workstation although lots for the current state are available.
- Start a batching operation with only a few lots including the high prioritized lot.
- Transportation of the lots using a dedicated operator.

The highly prioritized lots are often called rocket lots or bullet lots. If the amount of these lots increases to a certain level, destructive influences affect the manufacturing site with respect to capacity and performance (e.g., see [Ros08b]). In this thesis, our focus is on normal priority lots which are the largest amount of the available $WIP$ in our case.

# Part III

# Factory Model

Every act of creation is first of all an act of destruction.

*(Pablo Picasso (1881-1973) )*

# 6 Simulation

In this chapter, we introduce simulation as our key method to evaluate production control strategy changes. Besides the basic principles of simulation, discrete-event simulation is analyzed more thoroughly. Finally common tools for simulation are mentioned.

## 6.1 Introduction

A simulation is the imitation or representation of a real world system over time (e.g. see [Ban01, BFS87, CZ04, LK00]). Whether carried out manually or on the basis of a computer software, simulation involves a wide range of different actions for representing a real system as a model. In a simulation model, the system characteristics change over time and can be analyzed. It enables the researcher to answer what-if questions by creating different scenarios. Each simulation model has a system of different entities interacting together. Each entity has a defined mathematical, logical and symbolic relationship to the simulation model. Besides the answer of what-if questions, new systems can be evaluated. Simulation models can be classified according to different characteristics (see [Ros08a], [LK00]):

- Static vs. dynamic
  - Static: Only one point in time is considered or time is not taken into account.
  - Dynamic: The generated model represents the behavior of a real system over time.
- Deterministic vs. stochastic
  - Deterministic: No randomness is included.
  - Stochastic: Randomness affects the model behavior.
- Continuous vs. discrete
  - Continuous: Continuous system states change over time.
  - Discrete: Discrete system states change at discrete points in time.

An overview about the different simulation types is illustrated in Figure 6.1. In our case, we use a detailed simulation model with dynamic character. This includes detailed system knowledge and representation as deterministic or random processes.

Besides the simulation approach, queuing theory is a very powerful tool for analyzing different systems. For the cases where closed analytical formulas exist, exact calculations of the system are possible. For the cases where queuing theory does not provide closed

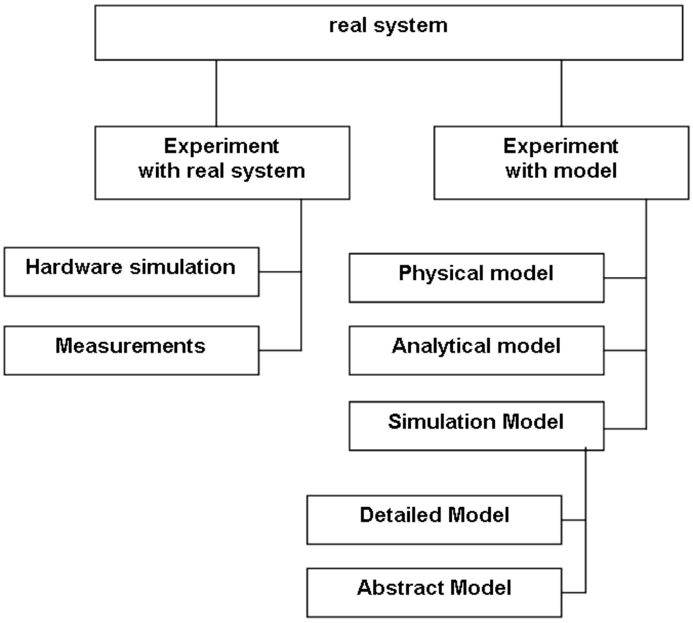


Figure 6.1: Overview over modeling types

analytical models, simulation approaches are used to solve these problems and questions. In our case, queuing theory does not provide sufficient possibilities to solve our large scaled problems. Lots of stochastic effects and processes can not be mapped by the queuing theory. Therefore the simulation approach is used. For more information about queuing theory, see [All90].

### 6.1.1 When Simulation is the Right Tool

Today, there is a wide range of different simulation languages, tools and approaches available. Decreasing cost and steady improvement of simulation tools has transformed simulation into one of the most used and accepted tools in system analysis. According to Banks (see [Ban01]), simulation studies can be used in the following cases:

- The systematic change of simulation input and the observation of the output can provide an insight into variables and their interaction.
- Simulations are used to experiment and test new solutions and designs prior to their implementation.
- The modern system has a very high complexity, thus interactions and system behavior can only be treated by simulation.

Of course, the development, validation and verification of a simulation model is both time and cost consuming. A simulation in the semiconductor environment can have many advantages, like analyzing different operating conditions without influencing the real system.

Alternative system designs can be evaluated. The experimental conditions are configurable, which is difficult to handle in a real system. Simulation is used as a key method for

- evaluation of the impact of dispatching rules on facility performance, e.g., in [AUH$^+$00, Ros01, Ros02, Ros03],
- generation of characteristic curves of an existing or planned manufacturing system,
- usage of simulation based scheduling, e.g., in [Sch00, RM07, PBF$^+$08], or
- optimization of existing manufacturing systems, e.g., in [TFL03, ANG08, DF03].

### 6.1.2 When Simulation is not the Right Tool

Simulation can not be used for any particular application. In each simulation study stochastic models use random variables produced by a random number generator. Valuable outputs are only possible if the input data of a model represents the system correctly. In addition, simulation models need a certain run time rather than using analytical solutions.

The simulation approach should not be used if the problem can be solved by using common sense. An example is a simple facility serving customers. The customers arrive randomly at a rate of 100 per hour and are served with a mean rate of 12 per hour. The minimum number of servers needed is 8.33 (just divide the arrival rate by the service rate). There is no need to use a simulation method if an analytical solution is possible, that will more precise and faster in computation. If real world experiments are simple to do, this route should be taken instead of using a simulation model. The major reason for not generating a simulation model is:

**If there is no data available, not even estimates, simulation is not advisable.**

## 6.2 Discrete-Event Simulation

Discrete-event simulation is a widely used method in simulation of semiconductor facilities. Discrete-event simulation models a system in which the state variables only change at discrete points in time. At these points of time one or multiple events occur. An event is an instantaneous occurrence that can change the state of a system. In a discrete event simulation software, different simulator components must be available (adapted from [LK00, Ban01]):

- **System state variables**: set of variables describing the state of each element in the simulation model.
- **Simulation clock**: is a special variable containing and evolving the simulation time.
- **Event list**: list of points in time containing all events scheduled.
- **Statistical counters**: number of variables containing statistical information about the model and the model elements.

- **Initialization routine**: one method or routine scheduled at the start of the model run at time zero.
- **Time routine**: routine which determines the next scheduled event in time and moves the simulation clock to this event time.
- **Event routine**: routine which changes the system state at a particular event.
- **Random number library**: library for generating random numbers and variates.
- **Report**: routines generating the statistical output of the model run.
- **Main program**: routine scheduling the next events, activating its methods and increasing the time.

Figure 6.2 illustrates a complete flow of a simulation run in the discrete-event environment. There are at least two stopping conditions for a discrete event simulation. The first possibility is an empty event list, thus no further events can be scheduled. The second alternative is the definition of a stopping time of the simulation either determined by the length of the simulation run or the end time.

In discrete-event simulation models, two common approaches can be applied. The rather straightforward way is to use an event oriented model, in which the modeler considers one event after the other. The event itself consumes no simulation time. Thus during the event routine execution the simulation clock is stopped. The second way is to use a process oriented approach. Each process is an ordered sequence of events which are related to a defined model object. While the process is executed, the simulation clock continues. Commercial simulators often use the second approach internally splitting the processes into events.

## 6.3 Simulation Studies

A simulation study consist of different steps, illustrated in Figure 6.3. The first step is the **formulation of the problem**. Naturally, each study has a certain problem or objective which should be analyzed and solved. The model analyst must make clear that the problem is understood. In many cases, only the knowledge about the problem is available, but the nature of the problem is not obvious. The **objective and the project plan** are generated in the second step. The objectives describe the purpose and the goal of the study. In this step, the applied methods and tools of the study are defined. The project plan includes the applied tools and methods, the time schedule, the involved persons and costs. It is useful to define different phases of work with due dates.

Model **conceptualization** can run in parallel to **data collection**. The available data of the systems determines the model detail level. The step consists of the abstraction of the elements of the system which is the object of study. In a semiconductor environment, that could be the elements mentioned in Section 3.1. The data collection for the abstract model must be translated in a form the model can operate with. The **model translation** step

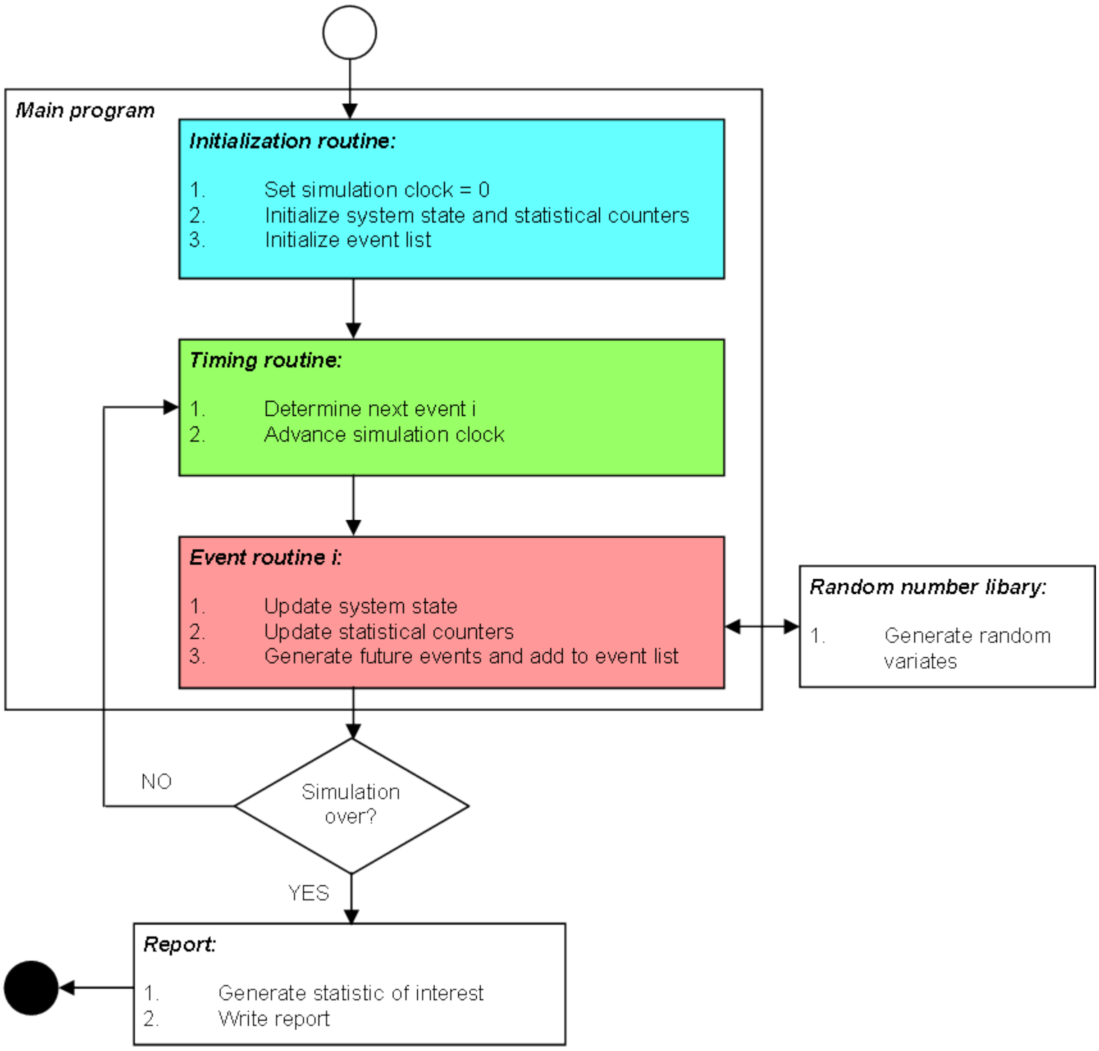

Main program
Initialization routine:
1. Set simulation clock = 0
2. Initialize system state and statistical counters
3. Initialize event list
Timing routine:
1. Determine next event i
2. Advance simulation clock
Event routine i:
1. Update system state
2. Update statistical counters
3. Generate future events and add to event list
Random number libary:
1. Generate random variates
NO
Simulation over?
YES
Report:
1. Generate statistic of interest
2. Write report

Figure 6.2: Complete flow chart of a discrete-event simulation run (adapted from [LK00])

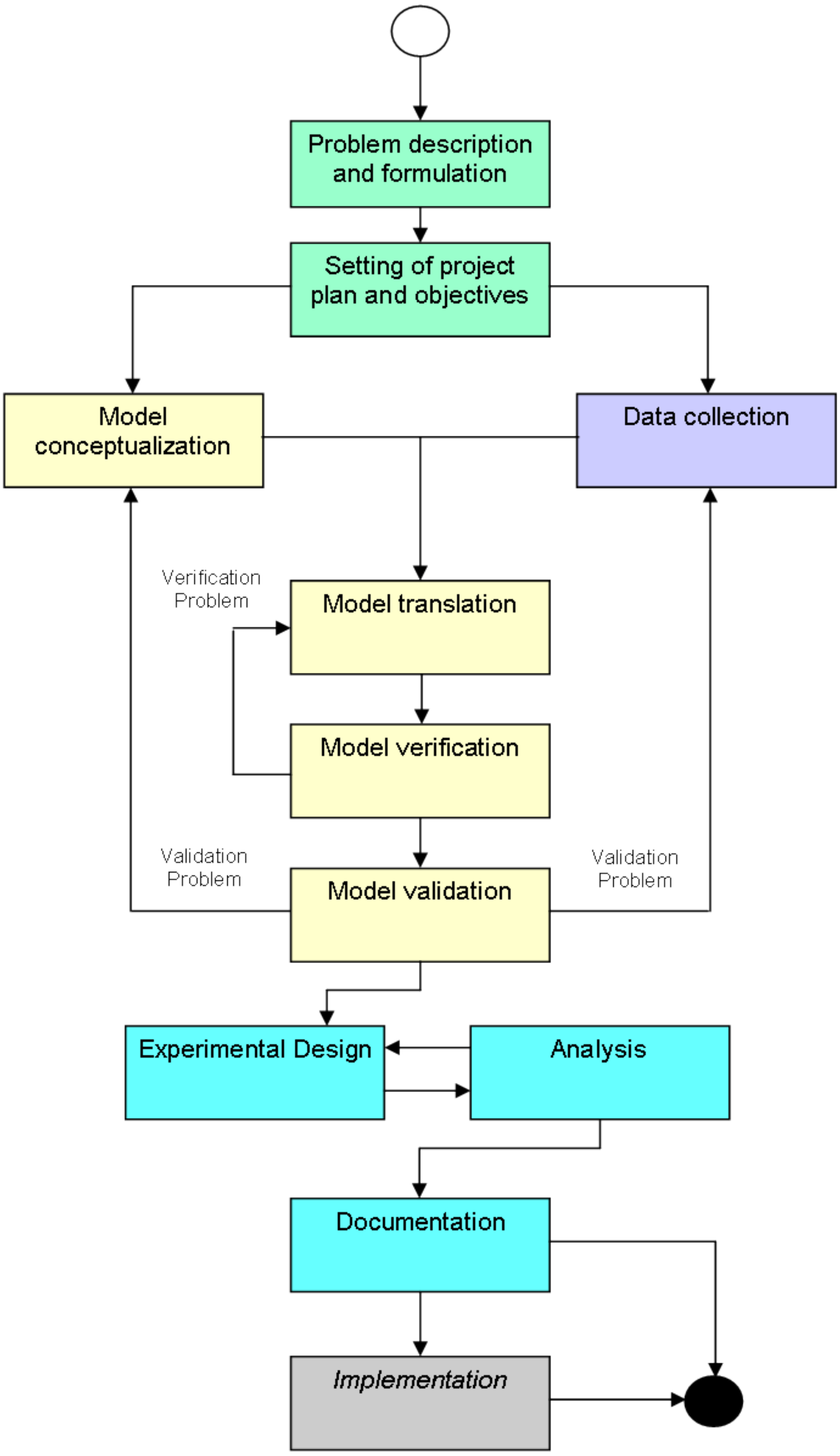


Figure 6.3: Steps in a simulation study (adapted from [Ban01])

transforms the conceptual abstract model into a computer-recognizable format. During this step, the modeling software must be chosen (see Section 6.4). Different characteristics like licensing and function volume are taken into account.

The **model verification** and the **model validation** are very important for simulation studies. The model verification includes initial pilot runs checking the simulation model is working correctly. This step is an iterative one also including debugging tasks. The logical and parameter structure are analyzed for correctness. In case of model verification errors, the model translation process must be changed in order to remove the errors. The validation step is more complex in application. The validation includes the confirmation of the model as an accurate representation of the real system. Performance parameters are checked against the real system. Runs based on historical system data are performed to check if the model behavior is coherent to the historical behavior of the real system. The process is repeated until model accuracy is judged to be acceptable.

In the next step, the **experimental design** and **analysis** phase is used to simulate the alternatives to be determined. The problem is evaluated with the help of the simulation model. This step can be repeated until an acceptable result or solution is found. The phase also includes the definition of the experiments themselves, such as simulation length or number of replications.

The results must, in fact, be **documented** and published. There are at least two types of documentation available. The documentation of the simulation model (including assumptions and the program documentation) and the progress documentation (including the different experiments and results) are the two types. The success of the last step relies on the quality of implementation of the previous steps of the study. The **implementation** step could be taken in form of introduction of the new approach to the real system. A final comparison of the results of the model with the results from the real system may offer new insights.

## 6.4 Simulation Software

A large variety of different simulation software packages is available on the market (see [GR11]). Simulation tools can either be classified regarding their purpose, the available user interface or regarding the business model behind the product. Figure 6.4 illustrates the general classification properties.

Different commercial and non-commercial tools are in use for our simulation experiments. The modeling software Factory Explorer 2.8 from WWK (see [Kel03]) was not sufficient for our purpose. We work extensively with the commercial simulation tool AnyLogic 6 (see [Tec10, MAN11] and with the open source simulation library Java Simulation Library (see [Ros08c]). An overview is illustrated in Table 6.1.

Factory Explorer was specifically developed for semiconductor manufacturing. It is not restricted to this application. It contains general specifics for semiconductor manufacturing, like rework or scrap, but also has a lack of some specialties in the case of foundry business. The tool provides a MS Excel interface for implementation of the simulation model and producing the statistical output. It is possible to add user specific code such as new

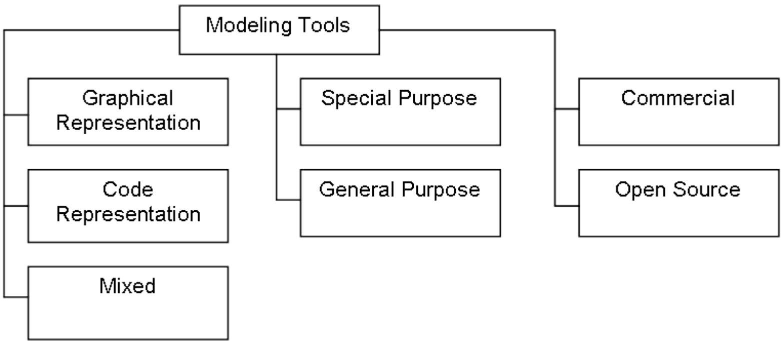


Figure 6.4: Modeling tool overview

| Software | Source | Flexibility | Language | Use Case |
|---|---|---|---|---|
| Factory Explorer | Commercial | Low | MS Excel | first simulation tests |
| Any Logic | Commercial | Medium | JAVA | evaluation of dispatching approaches |
| JSL | Open Source | High | JAVA | controller implementation |

Table 6.1: Applied simulation software

dispatching rules. The substantial advantage of factory explorer is its speed in comparison to other tools. A visualization of the modeling behavior is not possible. Unfortunately there is no option for more profound system changes, not even regarding the operator behavior with their special qualification and shift work system.

Therefore we choose AnyLogic as a general purpose tool allowing different simulation approaches. Besides the discrete-event simulation, an agent based approach is also possible. The tool is based on the JAVA programming language and offers a graphical model development as well as user specific JAVA source code. It allows time saving development of a first facility model for analyzing the different production control tasks. Due to the commercial nature of the tool, there is no direct access to the simulation core and to the defined models from foreign systems.

The JSL library is used for our prototype implementation, which is a purely JAVA based simulation library including all elements for discrete-event simulation. The greatest advantage is the accessible simulation core. Well-known other representatives of the simulation tools are AutoSched AP or the Flexsim simulation software, which are widely used in the semiconductor environment.

# 7 Model Conceptualization and Data Collection

In this chapter, we introduce our baseline simulation model applied at the experimental phase of our research. The model is based on real shop floor data and represents a typical ASIC SF with a broad range of tools and products.

## 7.1 Objectives

Various aspects of a dispatching strategy must be analyzed against

- performance of the facility regarding defined KPI,
- adaptability to factory floor changes,
- definition of different optimization goals, and
- suitability regarding the availability of real data.

For these reasons, a detailed factory model is developed, in which different approaches can be tested and analyzed. The level of detail is determined by the consumptive simulation time per run and the availability of data elements. In real factory environments, the available data is scattered over a wide range regarding quality and availability. The next sections introduce the whole model conceptualization and implementation phase.

## 7.2 Definition of the Model Elements

In semiconductor manufacturing, various strong influences are affecting the factory. Regarding Sections 3.1 and 5 there are several important entities in a semiconductor facility. The following list gives an impression of the fab profile:

- Product profile:
  - Number of different active products types: about 120
  - Flow complexity: 300 to 600 steps
- Process lines:
  - 6 inch line
  - 8 inch line

- Tools: about 300 different pieces of equipment and work centers
    - About 170 6-inch tools
    - About 80 combined 6-inch and 8-inch tools
    - About 50 8-inch tools
- Overall capacity of about 8000 8-inch equivalent wafer starts per month
- Manual transport and high level of operator interaction during processing

In this area different typical semiconductor entities are defined:

- **Equipment**: The equipment is a unique processing station which has to process one or more lots at a time. Optionally it has different setup states corresponding to the different recipes. Different states are possible, like processing and down. In case of a workstation with different unique processing chambers, each chamber is assigned as an extra tool. This is caused by the available data in the shop floor control system. Cluster tools are represented as normal tools having mean processing and setup times. A detailed modeling of these tools is currently not possible because of the lack of data in the data warehouse.
- **Equipment group**: An equipment can be assigned to a certain logical equipment group. It is to be noted that not each tool in a equipment group is eligible for each step assigned to the group. In our case, the tools per step are addressed separately not using equipment grouping.
- **Product**: Each product defines a sequence of different steps for a lot. Each step contains the equipments for processing including deterministic processing times, additional process-related waiting times and operator time consumption. These times are separately assigned per tool. There is a wide range of tools including newer and older ones having the same process capability, but different processing times. The times can be denoted per lot, per wafer or per batch. At each step, the maximal batch size is defined. At some stages, rework happens. Rework means switching in a rework work schedule and then coming back to a previous or the current processing step in the product schedule. If rework is not possible the wafers must be scraped.
- **Lot**: In our case, a typical lot has the size of 25 wafers. Each lot has a state. Most lots are production lots. Besides the production lots, sample and control/experimental lots are available. Sample lots often have a high priority from customer side requiring a short cycle time.
- **Operating staff**: Operators are required for transport and processing operations. The operator has a defined work schedule with an availability per schedule. In case of maintenance operators or senior operators, the productive impact is lower. The qualification of each operator defines their ability to work with different types of equipments.

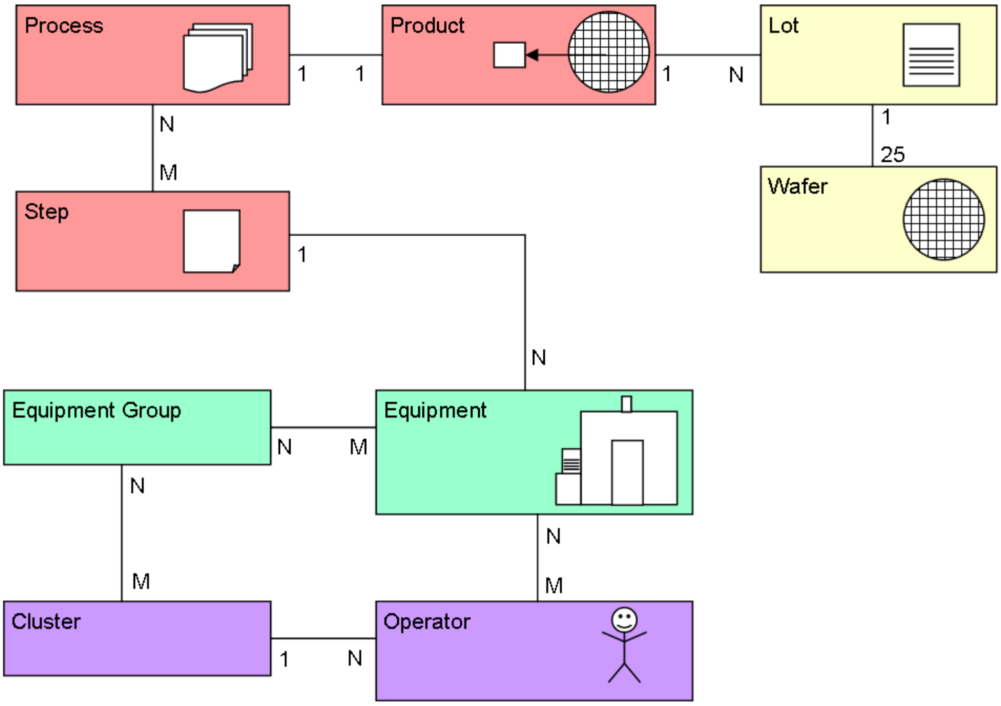


Figure 7.1: Model entities and their relations

The elements mentioned above are the main elements which are represented in the factory model. Figure 7.1 illustrates the main entities with their conceptual relations in a simplified UML[1] notation.

## 7.3 Data Collection and Analysis

For definition and development of a valid facility model, the data collection is a very important element. Low data quality can lead to wrong simulation results and false conclusions. Therefore the data environment of the existing data warehouse is analyzed. Different data sources are evaluated for each entity. The data sources can be divided into the following groups:

- Available, but not accessible (unstructured) data:
  - Empirical information and experience, for example interviews from operating staff about processes and their characteristics.
  - Data stored in proprietary data file formats like MS Excel without any common structure definition.

[1] The Unified Modeling Language is a graphical language for specification, modeling and analysis of software and other technical systems.

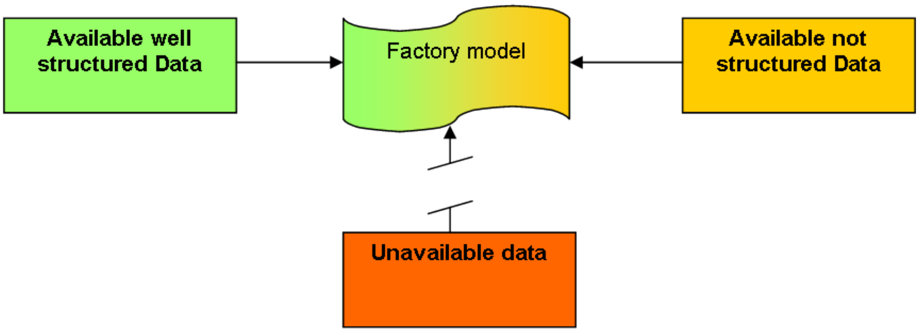


Figure 7.2: Modeling data grouping

- Available and accessible data:
  - Data stored in systems which offer interfaces for access.
  - Data stored in systems not offering interfaces, but the database itself is accessible and the data structure is known.
- Unavailable data:
  - Data which is not available neither in electronic nor in written or oral form, this can include approved methods or unwritten rules without a common agreement.

In this area of data sources, the data collection for a model can be a extremely difficult if no common standards are used. In our case standardized data access is very difficult due to an inhomogeneous historically grown information technology environment and different generations of staff. We had a lot of data collection problems during our research. Unavailable and inconsistent data requires difficult measures for data collection. An overview about this area of conflict is available in Figure 7.2.

In parallel to the definition of our modeling entities, the data access levels are evaluated for each entity. The level of structured accessible data is very variable in contrast to the unstructured data. The following list illustrates the most important data sources used:

- Available non structured data:
  - Setup changes: The sequence dependent setups are currently not being considered in the system. Therefore manual and oral information is collected.
  - Operator management: The operator management includes the operator work schedule and the qualification of the operating staff. Both are done in proprietary data file formats which do not allow structured data access.
  - Equipment down planning system (preventive maintenance): The preventive maintenance management is done in different tools which do not allow structured access. The planning is done with additional proprietary data file formats.
  - Batching: The best practices in batching are available in oral form from the line engineers.

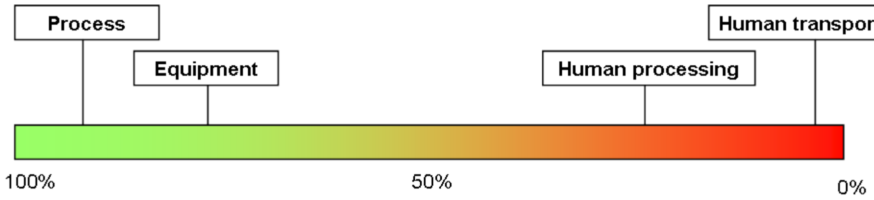


Figure 7.3: Quality level of availability model data

- Available structured data:
  - MES: The manufacturing execution system provides information about historical lot moves, the product steps and current lot states. The transporting times are not provided from the system, so transporting could hardly be modeled. The lot stop regime can be accessed. Lot stops are performed at different steps for a multitude of reasons. The additional lot processing is mapped by the system at a very low level. A modest amount of rework steps can be traced in a historically correct manner.
  - RM: The recipe management system provides information about the products with their processing steps including the planned processing, process-related waiting and operator times. Besides the times, the system provides information about usable tools per step.
  - Equipment monitoring: The equipment monitoring provides historical and actual states and state changes of the equipments regarding maintenance activities, unplanned down time frames, idle and processing states. In our case the state indication is done by operator staff, which results in an inhomogeneous monitoring information. Additional down information like reasons are not structured and therefore not usable in our case.

The model contains a lot of automatically collected data. In some cases, model simplifications in the entity characteristics must be introduced. The average level of detail is illustrated in Figure 7.3.

It is obvious that there is a significant gap between the automatically collected data by the MES (lot moves, equipment states, etc.) and the availability concerning human interactions. Human influences are often not well tracked and traced. The operator staff model uses some model simplifications introduced in the next chapter.

# 8 Model Design

In this chapter, we introduce the basic modeling approach realized with the modeling tool AnyLogic. Different model elements are introduced. The most important flows inside the model are illustrated.

## 8.1 Modeling Elements and Data Source Dependencies

The basic entities provided by AnyLogic are aligned together building a complex system of interactions. The most important elements are discussed in the following sections.

### 8.1.1 Technical Systems

In this section, we describe the most important modeled technical systems.

#### 8.1.1.1 Equipment

The equipment model is one of the most complex elements in the presented simulation solution. Equipment abilities in semiconductor manufacturing are introduced in Section 3.1. At the equipment level, several important components are defined. This components include different characteristics:

- Sequence dependent setups
- Batching characteristics
- Processing
- Equipment states:
  - Planned maintenance
  - Unplanned down
  - Setup
  - Waiting on operating staff
  - Processing

In AnyLogic the entities are implemented with the provided simulator components plus additional JAVA methods. In our case we use a finite state machine to describe the different equipment states. The state machine of the equipment is illustrated in Figure 8.1.

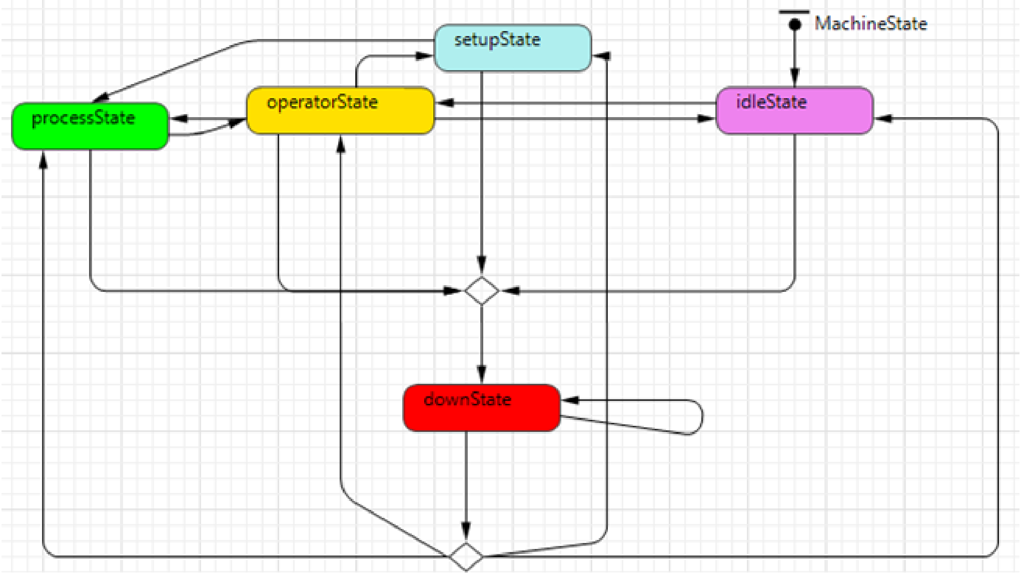


Figure 8.1: State chart of the equipment model

First of all, the available equipment states are defined:

- Setup state (performing setup)
- Operator state (waiting on operator)
- Process state (processing the wafers)
- Idle state
- Down state (in case of maintenance and unplanned downs)

The states are represented as a state chart. At each time in simulation, the equipment has a defined state. The states are triggered by different events. An overview about the most important available events at equipment level is given in Figure 8.2.

Each event causes different state changes in the equipment model, depending on the boundary conditions. If a new lot arrives at the equipment, initially the lot is put into the equipment queue. In case of an idle workstation, the processing of the lot is started. Previously, the appropriate lot or batch (according the dispatching priority, setup states and batching regime) is chosen, and if necessary an operator is called (in case of tools with operator interaction). If the operator is assigned, the lot processing starts. If a setup is required, the setup change is initiated. After the deterministic processing time, the processing is finished and the tool can either process further lots (if more lots are in queue) or change the state to idle. During processing state, the operator resource is released. This happens in case of the defined operator time being over. Therefore an operator is not assigned to the tool for the whole processing time. A more detailed model, including

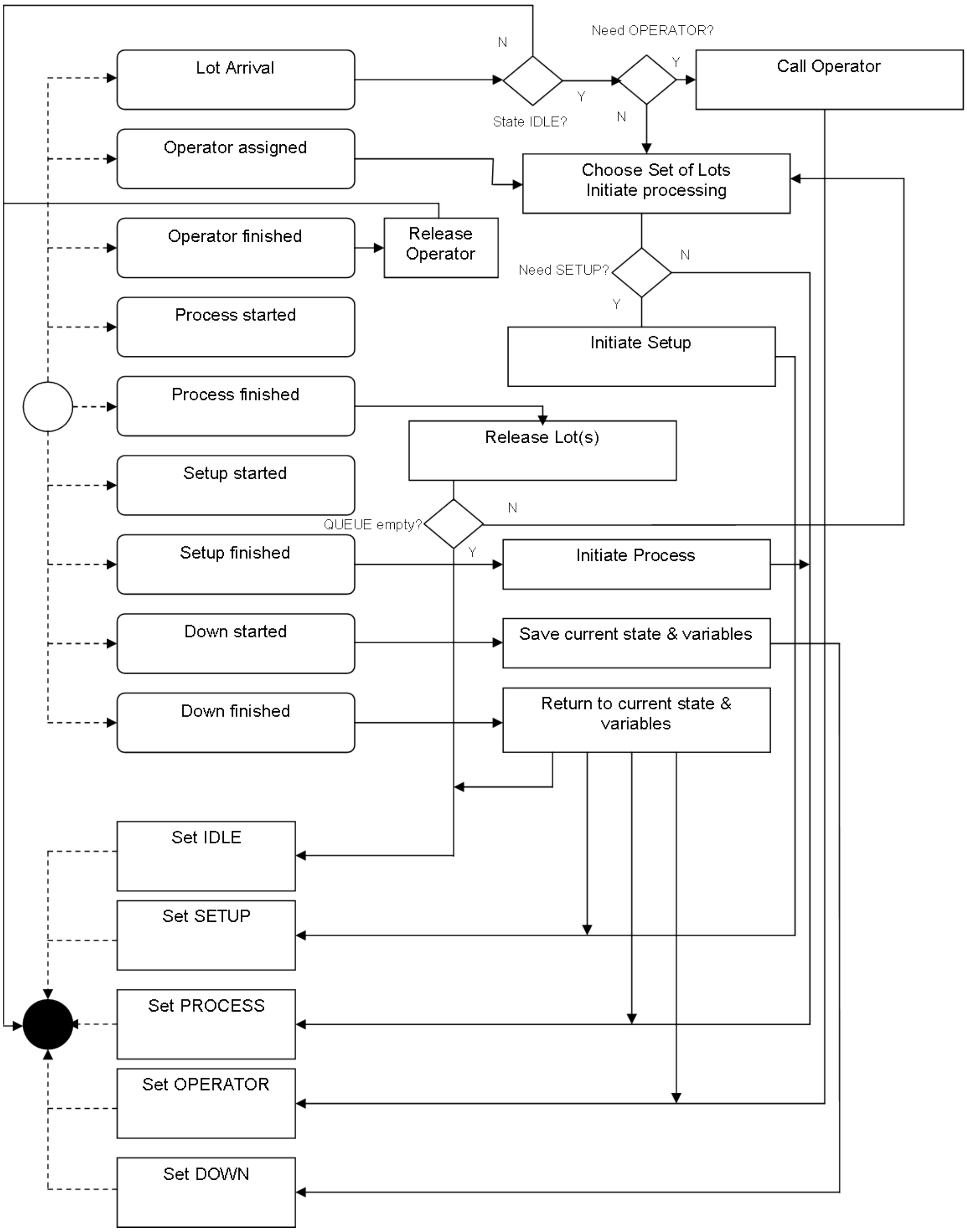


Figure 8.2: Flow chart of equipment dependent events

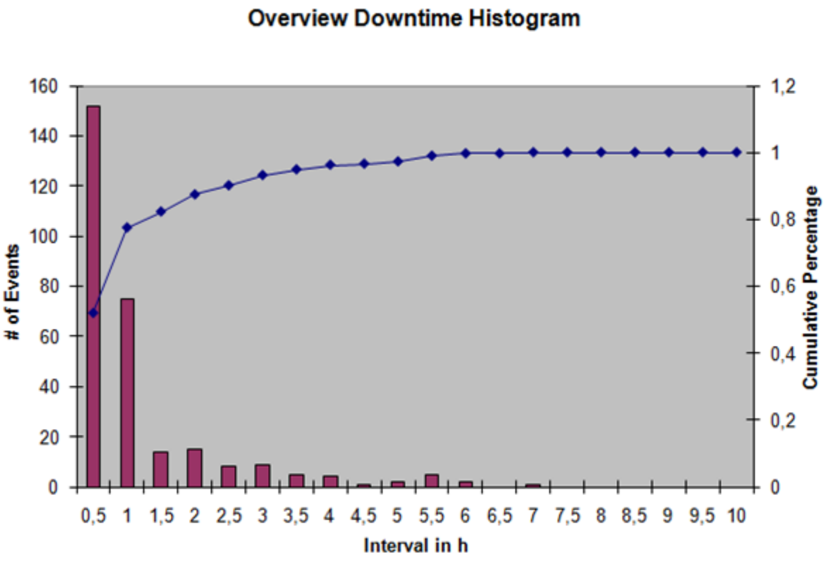


Figure 8.3: Example of a down time histogram for a equipment

loading, unloading and processing times, is not possible. This model simplification is used as there is no data available in regard to loading, unloading and processing times.

The down events are scheduled either by a defined random distribution in both length and occurrence, and by concrete historical down events per equipment. In our research we tend to use historical data in order to verify and validate the model behavior. Large amounts of equipment down events are planned activities like maintenance operations which are naturally not random and independent. In our case maintenance periods for each workstation are not fully available (unstructured data). A manual collection of this massive amount of data is still not possible. The usage of common distributions like exponentially distributed random numbers are not applicable due to the huge variety of down events (example see Figure 8.3). So we used complete historical model runs allowing us to check the model results for reliability and quality concerning the real historical data.

#### 8.1.1.2 Lot, Product and Technology

Each lot in the factory model has several variables and options to represent the natural properties. These variables are the wafer count, the product name, the work schedule and the due date. As mentioned in Section 8.1.1.1, we use historical data including real lot starts and due dates. In our case, there is a vast variety of different products and their variants. The huge amount of different steps accumulate a lot of data. We define product classes with typical product representatives. Each product step contains the available tools, the process times, the process-related waiting times, and the operator times. In addition setup times and states are included. Setups are added to the most important tools if they affect the facility performance.

| | Individual | Group | Society |
|---|---|---|---|
| Strategic | Design decisions | Local markets | Insurgency model |
| Tactical | Selection of next process, route; procedural tasks | Crowd behavior in traffic simulation | Consuming behavior |
| Physical | Skills, Qualification | Crowd movement | Population aging |

Table 8.1: Applications for different levels in human modeling (adapted from [SB10])

### 8.1.2 Human Behavior

In this section, we give an overview about the modeling of human influences and about the operator model realization.

#### 8.1.2.1 Modeling Human Behavior

"Human behavior is the collective set of actions exhibited by human beings, either individually or in groups of various sizes and compositions" (from [SB10]) is the first sentence proposed by Sokolowski and Banks talking about the issue of modeling human behavior. Of course there is a wide range of different theoretical and practical approaches available to model human beings. This field of research has a huge dynamic in its development.

As human beings, we know about ourselves how we tend to generalize and apply our experience to other domains. It is vital to understand the nature of the complexity of human interaction and behavior. The issues during modeling human behavior vary, which depends on the available data and the required level of the human model. Sokolowski and Banks divided the behavior of human being into two levels, the *physical level* and the *tactical level*. At physical level, the human characteristics are driven by physiology and automated processes. The case of the tactical level mainly describes short term actions driven by emotions and decision processes. Long term actions can be summarized as the *strategic level*. This can include complex planning decisions based on emotion, experience and intuition. While modeling humans, there is also a difference regarding modeling individuals or groups. Groups have a so called group dynamic behavior (examples for this levels are illustrated in Table 8.1). Of course every model corresponding to a level deals with different issues. In our case we have to model the tactical and physical behavior of a small operator group. This group can be defined as a group of several individuals. Models of individuals are typically less challenging than the other groups, where the complexity is also driven by the interaction and possible chaotic dynamic nature. Every use case needs other techniques and has other issues with human modeling.

In building human behavior models, common techniques were developed like

- fuzzy Logic control,
- finite state machines (like the case at equipment modeling), and
- RBSs, Pattern recognition, ANNs, HMMs (see [SB10]).

| Operator | Equipment 1 | Equipment 2 | Equipment 3 | Equipment 4 | Equipment 5 |
|---|---|---|---|---|---|
| Operator 1 | OP | | | IHOP | IHOP |
| Operator 2 | | OP | IHOP | | |
| Operator 3 | OP | | | OP | |
| Operator 4 | | | OP | | OP |

Table 8.2: Operator qualification matrix example

In our case we choose a finite state machine to represent the operator's behavior. Fuzzy logic is also taken into account (e.g., see [SB10]), but tends to need a large amount of detailed data which is not available in our case.

#### 8.1.2.2 Operator Interaction

Process tasks at equipments can include different manual operations. The most common ones are the loading process, the unloading process, and the assistance during processing. At purely manual processes like manual wafer inspection steps, the operator has to work throughout the whole processing time. Otherwise operating personal has to do assistance work during the processing operation which is less time consuming. In our case, only a rough estimate of the overall operator time per process step is available. Therefore we estimate this time for operator assignment throughout the whole processing time.

Each operator has his own qualification level. In general, we divide them in senior operating personnel, which has a high qualification level on different equipments in a cluster, maintenance personnel having an additional maintenance qualification level, and the regular operating staff. In our case we specify a qualification matrix per operator, defining the ability to work with different tools or tool groups. At each workstation-operator combination, the ability to perform processing or maintenance tasks is defined. In Table 8.2 there is an example for the assignment. The term OP is defined by the ability to perform the normal processing operations, the term IHOP indicates the additional ability to perform maintenance tasks.

For our model, we compile a general set of operating personnel based on a historical analysis. For each cluster, a set of typical operators with representative qualifications is put in place. The following list illustrates the number of operating personnel per cluster:

- Lithography: about 8 operators
- Etching: about 5 operators
- Metallization: about 2 operators
- High temperature: about 5 operators
- Wet processes: about 5 operators
- End of line control: about 3 operators

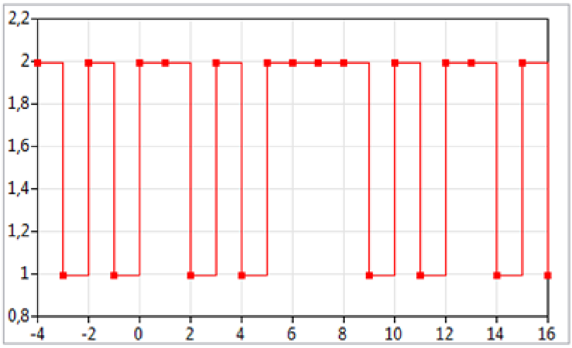


Figure 8.4: Example work shift schedule per operator group

Each operator is assigned to a certain work schedule. In our case the 12 hour work shift is applied. During the working shift, the operator is forced to have several breaks. The break time is very difficult to determine, so a random break distribution per work shift with an average count of three breaks is defined. Besides this, the whole break time of the operating personal is determined to be two hours. In a more general approach, we also tested the definition of fixed break schedules by defining the availability of each operator group. An example is illustrated in Figure 8.4 where the defined operator count in an operator group is two.

The operator state is represented by a state chart. The state chart consists of three states:

1. Idle
2. Processing
3. Not available

State one determines that the operator is able to operate the next requests; currently the operator is doing nothing. The second state defines that the operator is currently active and involved in a processing task. The third state is active when the operator is currently on a break or not available at the work shift. The states are triggered by different events as illustrated in Figure 8.5.

#### 8.1.2.3 Transport

The transport process in a human dominated transportation environment is very difficult to model. In our case we use simplifications. We distinguish two different transportation types:

- Single transport operations
- Batch transport operations

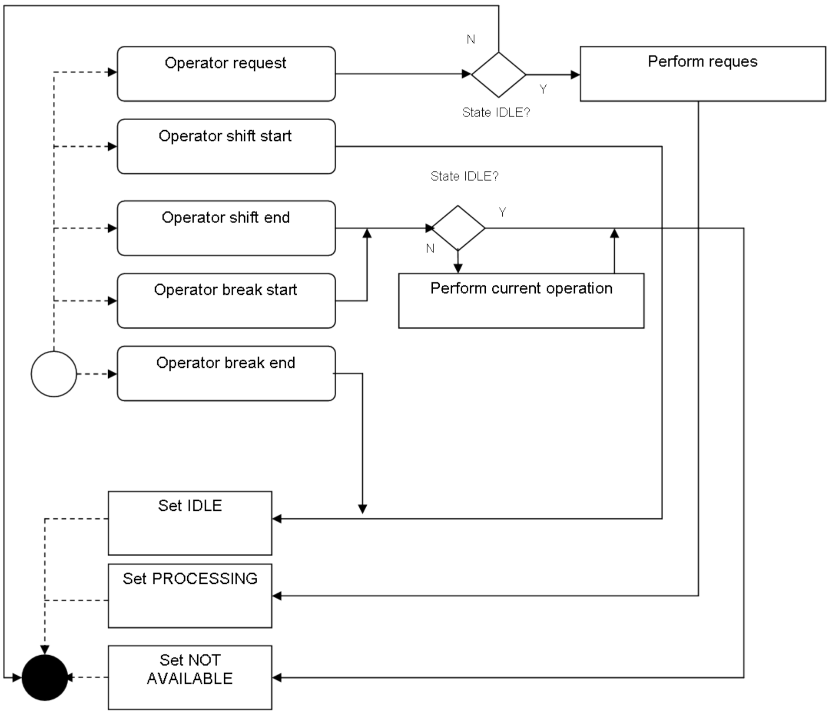


Figure 8.5: Flow chart of the operator staff

Single transport operations happen in the cluster area where an operator takes one lot per time. Batch transports are performed at longer distances between cluster areas. In our case transport operations are not tracked at all, and very little information about general transportation times is available. That is why we use general assumptions to represent the batching transport operations. Batching transport operations are only to be performed between clusters by definition. A general transportation size of at least six lots is used. A maximal waiting time is defined until the batch transport has to start. This time is determined by the following processing steps.

### 8.1.3 Top View on the Model

The factory model is designed as a two-dimensional graphical representation. This representation includes elements like operator, equipment, lot and storage. The graphical representation in our model is a presentable way to illustrate the model behavior. It can be also used for model validation and verification. Figure 8.6 illustrates a part of the whole factory layout. The pieces of equipment are illustrated as light blue boxes, the storage places as orange boxes.

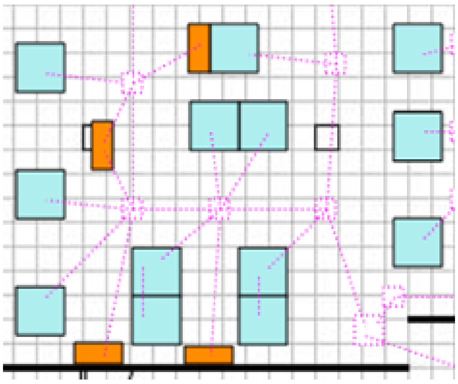


Figure 8.6: Top view on factory model (section)

## 8.2 Modeling Element Flow

In our model, the main model element besides the equipment is the lot entity. There are at least two main stages each lot passes several times. The flow control stage is responsible for the routing of each lot through the facility model. It assigns each lot entity to the available equipment queues per process step. In our case each work station has its own equipment queue to which all available lots are assigned. The reason for this design originates from the circumstance, that clear tool group definitions are not possible, often a mix of different tools of different ages are used for processing. The flow control element has to assign a virtual copy of the entity lot for this at every stage to each equipment queue. Commercial simulation tools like Factory Explorer do not offer this feature in general. There are only work centers with a defined number of available equipments that can be applied.

If a workstation is ready to process the next lot or batch, the second most important element is acting, followed by element control. This control element chooses the best lot or batch for processing. This process contains algorithms using information about setup, the batching information and the currently active dispatching rules. With the selection of the set of lots to be processed next, the virtual copies of the lots in the other possible equipments have to be deleted. Today's simulation tools often use a multi-threaded environment. Therefore a thread safe environment for the method calls has to be established. Otherwise duplicated lot entities or false lot processing are the consequence. Figure 8.7 illustrates the general flow of the lot model representation.

## 8.3 Summary and Open Questions

In this chapter we introduced the classical approach for modeling a semiconductor environment. Several additional characteristics are also taken into account, like lot stop behavior. Some characteristics are often not provided by commercial special purpose tools like a detailed operator qualification or single pieces of equipment instead of work stations. The facility model is developed to find better production control strategies. The large

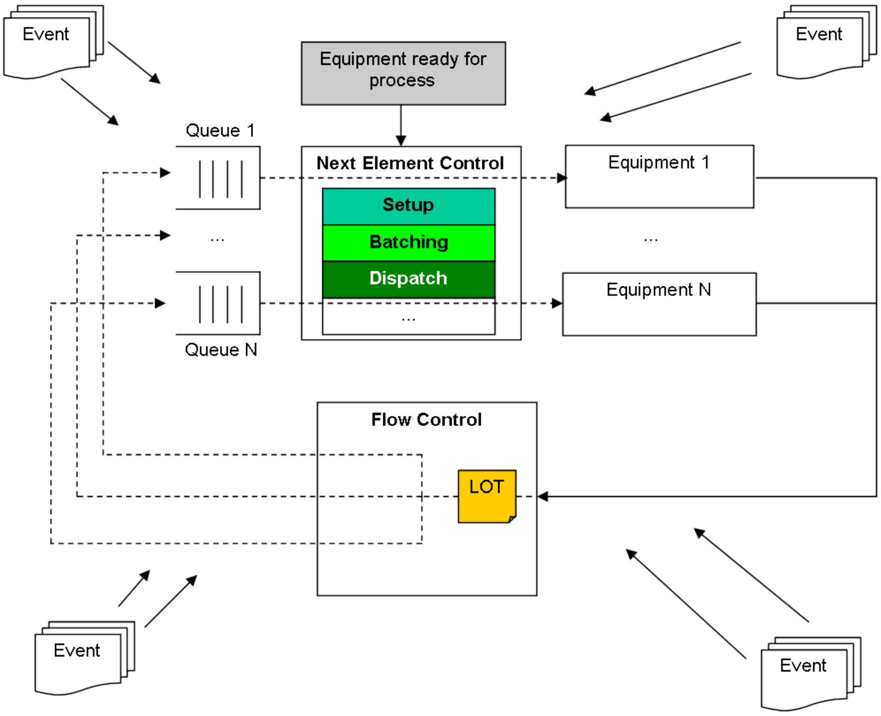


Figure 8.7: General flow of the entity lot

amount of human influence in our case leads to a fuzzier model in comparison to reality. The challenge of a continuous improvement and analysis of this model is still present. Furthermore, the facility in our research is historically grown and offering a huge variety of applied software solutions. This makes data collection very difficult. An automated data collection is advised rather than a manual one. For this a very strict model verification and validation process is carried out which is introduced in the following chapter.

# 9 Model Verification and Validation

In this chapter, we introduce the most important steps of the model verification and validation.

## 9.1 Definitions

Simulation models are used for prediction, comparison and evaluation of established or new real systems. After the model conceptualization and translation phase, the model verification and validation phases are necessary to prove the model's accuracy and correctness. Sufficient model accuracy and correctness can be attained if the model is a substitute for an existing system at a certain level of detail. This substitution is only reasonable for the number of experiments and analysis of the project.

In practice, the steps of verification and validation are often mixed. An iterative way is often used. What is verification and validation in detail? The definitions are extracted from[OR10, LK00, JSC02, Ban01].

**Verification** is the summary of all techniques concerning the right model development and implementation. In common, the conceptual model is compared to the implementation of this model (model translation). In this phase the correctness of the implementation is analyzed. Input and output parameters and the logical structure are examined against the conceptual model. In general the objective is a correct model implementation.

**Validation** is the summary of all techniques concerning the right model representation of the real system. It is checked whether the model is a correct representation of the real system. The validation phase is often called model calibration. Model calibration is an iterative process for decreasing the gap between the real system and the simulation model. This process is repeated until model accuracy is at acceptable level. Figure 9.1 illustrates the whole process.

For our model verification and validation process, we use different notations for the model output parameters $P_i$ introduced in Section 3.2. Within each model parameter, the interesting statistical properties like mean, standard deviation, minimum and maximum are collected. These elements are compared to our model assumptions and historical expectations.

## 9.2 Common Errors in Data Modeling and Simulation

There is a wide range of common well-known errors in data modeling and simulation. Carson [JSC02] proposed four main categories of modeling errors. These categories are

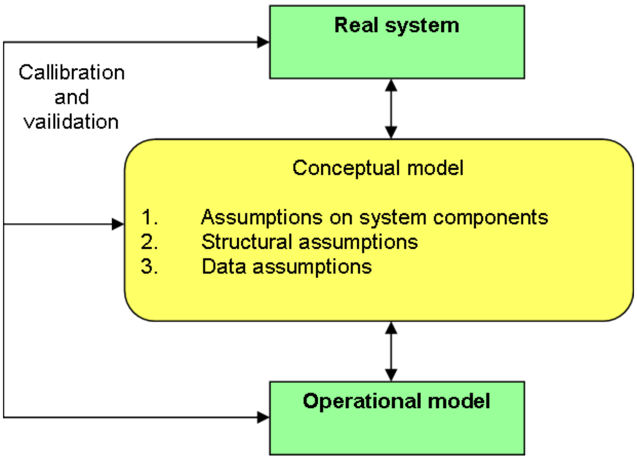


Figure 9.1: Iterative process of model verification and validation (adapted from [Ban01])

**project management** errors, **data** errors, **logical model** errors and **experimentation** errors. We introduce each of these elements in the following sections.

### 9.2.1 Project Management Errors

Errors regarding the project management are often based on communication. In fact, a simulation study requires a team of different persons to succeed. The team consists of the model developers, the customers and engineers. The customer side often includes operator personnel or line engineers. A critical demand is the involvement of all key persons into the whole project. It is important to specify which questions and areas are addressed by the simulation study. The agreement among the people on the questions and study parameters is vital. In reality, this fact is often not considered. This can cause additional model conceptualization work, when key assumptions change. Therefore it is a precondition to involve all people from the beginning.

### 9.2.2 Data Errors

Data errors can be divided into errors regarding the input data and the model data assumptions. Input data errors have their origin in incomplete or inaccurate input data. Carson mentioned different examples for data source errors like:

- Only summary data can be used whereas there is a need of individual values. For example only deterministic process times are available, but the processing time itself varies.

- Data grouping problems often occur. For example the absolute amount of downtime and up-time of an equipment per shift is known, but there is a need for detailed downtime and up-time statistics.

- Inaccurate input data is often caused by human influence. In case of manual equipment state notification, the reasons and real repair times are sometimes not correctly entered.
- Incomplete data is usually a big issue. In many cases data is available that corresponds to simulation output data, but not to input data.

A large variety of data assumption errors can be found. That includes for example:

- Definition and usage of distributions which are not appropriate for the field of use.
- Using average values instead of defining an appropriate random distribution.
- Inappropriate modeling of different model elements, like equipment failures.
- Assuming a statistical independence when it is not possible.

Data errors can be solved by introduction of new data collections systems (in case of incomplete data), or improving work flows in management site like standardized equipment state notifications.

### 9.2.3 Logical Model Errors

Logical modeling errors can be found in specification and implementation of the model elements. Logical model errors can be divided into

- programming language-dependent and
- conceptual

model errors. Conceptual model errors can occur due to project management errors like a faulty project definition. Language-dependent errors can be divided into common errors that can be seen in a wide range of different programming languages and simulation-language dependent errors. For example, faulty indexing of array definitions is one of the most common failures. Logical model faults can be found and removed by a extensive model verification and validation.

### 9.2.4 Experimentation Errors

During the analysis and experimentation phase, further failures can occur. Experimentation errors include for example an insufficient number of experimentation runs, a false detection of the warm-up phase, or false understanding of confidence intervals. This error group is normally not included into the validation and verification process, just a part of the experimentation and analysis phase.

## 9.3 Model Verification

In this section, we introduce the model verification in detail and illustrate the application of the different techniques to our model.

### 9.3.1 Techniques for Model Verification

Verification in the information technology area is a wide field. It contains different techniques and advices for analyzing the implementation in the area of simulation models. The purpose of model verification is to assure that the conceptual model is correctly transformed into a model implementation. In general Banks et al. ([Ban01]) proposed three different groups of techniques:

- Common sense techniques
- Documentation
- Tracing

These techniques are introduced in the following sections.

#### 9.3.1.1 Common Sense Techniques

In the field of common sense techniques, there is a wide range of different suggestions rather than techniques available. The next points introduce some important suggestions mentioned by [Ban01, LK00]:

1. **Four-eyes principle**: In larger simulation projects, it is advisable to apply structured reviews by more than the model developer. In larger organizations, this is also called a *structured walk-through* of the program. This type of review is often a useful and helpful mechanism to avoid errors the programmer himself is not able to see (also called organizational blindness). For structured reviews, a modular and structured implementation is necessary.

2. **Divide and conquer**: This principle is more a design pattern rather than a verification technique. It is very useful during debugging for allocation of errors. Each of the sub-modules should be tested itself before an integration into the main simulation program takes place. This is called the test while implementation approach. In many cases the system is tested after the implementation.

3. **Variation of runs**: The structured definition and application of test cases is essential for a structured verification phase. For each test case, different boundary conditions must be applied and the result has to be checked against the expected result. In discrete event simulation, low detail models can be defined and used for testing entire model elements. For example, a one machine model with one operator can be used to check several implementation aspects. In this case, the results of the run can be calculated and compared to the test results.

4. **A picture says a thousand words**: If possible, a graphical representation of the model activity is useful for finding logical and structural errors. This can be used in network simulation or semiconductor manufacturing simulation. The flow of the lots, the movement of the operators and the equipment states can be visualized to find errors. This can be wrong equipment states, false routes of lots or wrong operator actions.

#### 9.3.1.2 Documentation

One of the most important parts of each study is a sufficient documentation. This includes the model assumptions, model implementation details and test results. The resulting documentation can be integrated into an extended structured review. The documentation can offer structural problems. Besides the model design and implementation, the test coverage can also be analyzed. Some important test cases are often not performed due to only one person being responsible for testing.

#### 9.3.1.3 Trace

For debugging simulation model implementations, the usage of a trace can be helpful. It is one of the most powerful, but also most time consuming techniques available. In a trace, the state of the whole simulation model is printed, after an event occurred. That can include the event list, the different state variables, statistical counters and others. Each of these state changes can be compared to the assumptions made. While the trace technique is applied, extreme unusual situations should be analyzed. Today, an interactive debugger is often used instead of manually printing each trace step. Sometimes, errors only occur after longer simulation runs, or some key information is not available in a trace. An interactive debugger allows the programmer to check each variable in the system at each event during the model run. It is also possible to stop the simulation at a defined point in time. A lot of the modern simulation environments offer this interactive debugging option. Table 9.1 illustrates a partial trace of a single queuing system with one server.

### 9.3.2 Verification of the Factory Model

For verification of our factory model, we define different simple facility models to verify the different model element behaviors. The test setup is illustrated in Figure 9.2. In our case, we use three different simple models containing

- a single queuing system,
- a queuing system with parallel equipments, and
- a queuing system with two equipments in series.

These three simple models are used to perform several tests. These tests include tracing of the model behavior of important facility performance parameters. For this we define simple arrival rates at a constant rate and constant processing times. Throughout the model verification, different additional behaviors like setup, batching, equipment failures, and operator interaction are added and verified. The next two examples illustrate the verification process elements in extracts.

**Example 1:** In our first example, we define a simple single queuing system for verifying basic model element behavior. We assume an arrival rate $\lambda_r = \frac{5lots}{h}$ and a constant interarrival time $t_i = 0.2h$ of identical products and a processing time $t_p = \frac{1}{6}h$ per lot. Now we can simply calculate the utilization $U = \lambda_r t_p = \frac{5}{6}$. The mean queue length is

| Event | Clock | Server State | Queue Length | Event List | | Number of Customers | Server Utilization |
|---|---|---|---|---|---|---|---|
| | | | | Arrive | Depart | | |
| Start | 0 | IDLE | 0 | 0.6 | | 0 | 0 |
| Arrival | 0.6 | PROCESS | 0 | 1.2 | 1.6 | 1 | 0 |
| Arrival | 1.2 | PROCESS | 1 | 1.6 | | 2 | 0. |
| Arrival | 1.6 | PROCESS | 2 | 2.8 | | 3 | 0.625 |
| Departure | 1.6 | PROCESS | 1 | 2.8 | 2.0 | 2 | 0.625 |
| Departure | 2.0 | PROCESS | 0 | 2.8 | 2.4 | 1 | 0.7 |
| ... | ... | ... | ... | ... | ... | ... | ... |

Table 9.1: Partial trace of a single server system with one queue

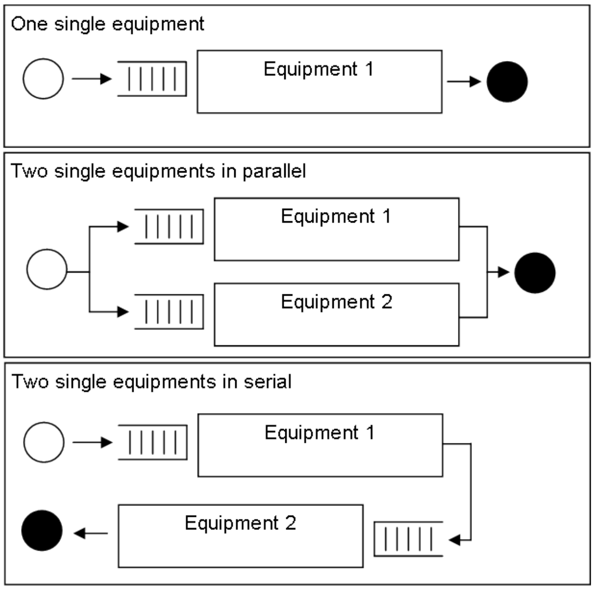


Figure 9.2: Test preparation for facility model verification

zero and the X-factor is $XF = 1$ . The resulting $CT$ can be calculated to $CT = t_p = \frac{1}{6}h$. These expected values are compared to the model results. This simple example is extended by setup times in a two product environment under various start scenarios. In addition operator interaction is added. With this simple setup, the basic model behaviors are verified and tested.

**Example 2:** We assume two workstations in parallel with the following expectations. We define two different product types $p_1$ and $p_2$with a processing time $t_{p_1} = 0.2h$ and $t_{p_2} = 0.3h$, which is equal on both equipments. The arrival rate $\lambda_{r,p_1} = \frac{5lots}{h}$ and $\lambda_{r,p_2} = \frac{1lot}{h}$. For processing of a certain product group, there is a setup needed defined by a constant setup time $t_{s,p_1} = 0.3h$ and $t_{s,p_2} = 0.5h$. There is a mechanism to minimize setup changes, also called setup avoidance. In this simple case, of course, after a short warm up, one of the two equipments would only process product type $p_1$, whereas the other equipment would process $p_2$.The utilization of these equipments can be calculated to $U_{t,ep_1} = 1$ and $U_{t,ep_2} = \frac{1}{3}$.

The examples mentioned are classical examples for a tracing procedure for verification of the basic model behavior. This process is done in a huge variety of different use cases and definitions. Besides these objective tests, the verification procedure of our model also includes the common sense techniques defined in Section 9.3.1. Different meetings are held to verify the basic model behavior. A graphical representation in a two dimensional way is implemented to verify the graphical output during the model run. It is compared with the visual expectations, e.g. the storage utilization at certain points, equipment states, operator movement, and so on (example see Figure 9.3 without operator illustration). At the end of this process our basic model behavior is verified.

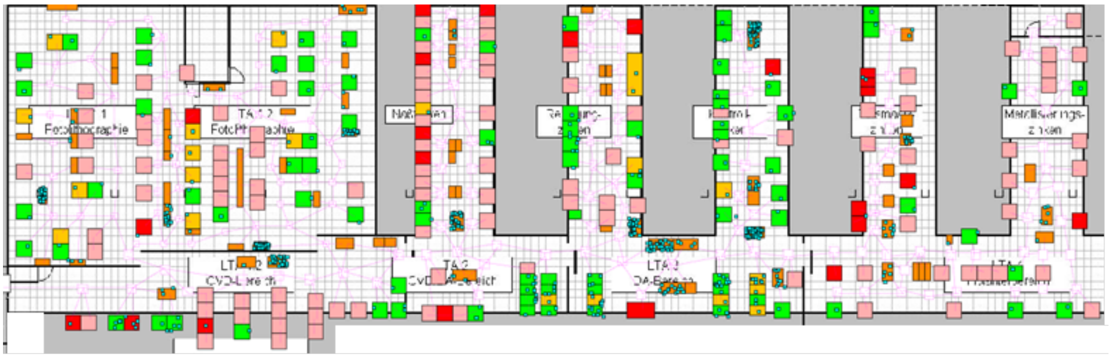


Figure 9.3: Graphical model representation

## 9.4 Model Validation

In this section, we introduce the model validation in detail and illustrate the application of the different techniques to our model.

### 9.4.1 Techniques for Model Validation

As there is a wide range of different verification techniques, a lot of different validation techniques are available. As mentioned before, the validation phase includes model calibration. The validation is a process comparing the model behavior to the behavior of the real system. The comparison can be done by a variety of tests, which include subjective or objective ones. Assuming we have a set of input data $X$. This set is fed in both, the real system (e.g., by historical data), and in the system representation (the model). The goal of the validation is that the resulting set of system variables $Y$ from the real system and $\overline{Y}$ from the model should be nearly the same:

$$H_R(X) = Y = H_M(X) \pm e \tag{9.1}$$

For this the resulting set of the transformation functions $H_R(X)$ and $H_M(X)$ must be equal. Of course this level of equality is not possible at real world simulation models. The resulting error $e$ describes the deviation from the real system to the model and should be in an acceptable region. For this several statistical tests can be performed to analyze the model behavior. The transformation process is illustrated in Figure 9.4.

Banks [Ban01] proposed three steps for model validation. These steps are the **face** validity, the **model assumption** validity and the validation against the **input-output transformations** of the model to the real system. In the next three sections, we introduce each point in more detail. For further information, we refer to [Ban01, BFS87, LK00].

#### 9.4.1.1 Face Validity

The first way to a validated model is the construction of a model, which is reasonable in its form to the model users and developers. The model users and customers should be involved

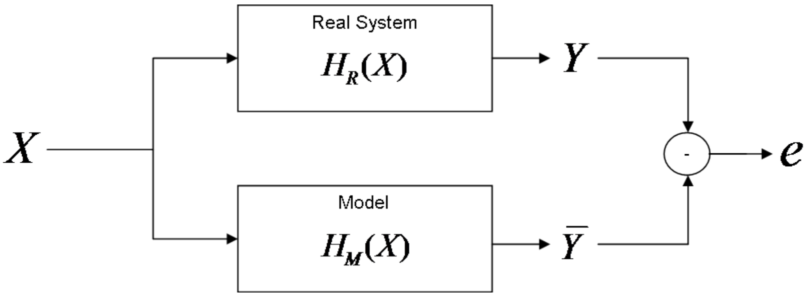


Figure 9.4: Validation in the system theory context

throughout the whole evaluation process. The potential users with their high knowledge of the real system can identify and detect model deficiencies and problems. Besides this, the faith into the model is increased when involving this group of people into the model development phases.

Furthermore, the analysis can be done by using common sense. By changing one or more input parameters, the expected results can be analyzed against the model results. A simple test would be the variation of customers in a queuing system, a higher number causes a longer waiting time, in general. By changing the most important input variables, like starting rates and operator numbers, the model can be checked against the expected behavior.

#### 9.4.1.2 Validation of Model Assumptions

The validation of the model assumptions is the second most important step for validation. The model assumptions can be divided into structural assumptions concerning the model elements, and the data assumptions, concerning the input and output data. Structural assumptions include the model elements and their interactions, simplifications and abstractions. For example a simplified equipment model can be mentioned here.

The data assumptions should be based on a set of reliable input and output data. With this data, the model behavior can be defined. For example the processing times are assumed to be constant or not. The analysis of random input data (machine failures and downs, etc.) for finding a correct statistical representation can be done in three steps:

1. Identification of the appropriate probability distribution.
2. Estimating the parameters of the favored distribution (e.g. the mean and the standard deviation).
3. Validating the assumed statistical distribution with known statistical tests like Kolmogorov-Smirnov test.

With the help of these different techniques, the model behavior has also to be validated. For this, the input-output transformations are analyzed.

#### 9.4.1.3 Validation of Model Input-Output Transformations

In this phase of the test, the model is viewed as an input-output transformation process. To do so the model consumes different sets of input variables and generates a defined set of output variables. This transformation process is to be validated. This test can be seen as the only way to test the model behavior in a statistically objective way. In general the transformation process can be validated by using actual data for prediction of the future behavior, but this is not common. Usually two other ways are used for performing transformation tests:

- Validation by using historical input.
- Validation by using Turing tests.

The validation by using historical data is only applicable for the representation of existing systems. By using this technique, the historical system behavior should be reproduced by the model under the same set of input variables. The resulting set can be compared to the resulting historical data. This technique is very reliable and widely used in existing systems with a large amount of historical data and statistical output. Several statistical tests can be used for analyzing the model behavior under the historical input cases. For example the confidence intervals of the model results can be analyzed. With the help of this technique, the number of replications and the stability of the simulation can be evaluated.

If no statistical test can be performed, or the historical data is not available in a sufficient way, knowledge about system behavior can be used for comparison of the model output to the system output. This can be done by generating system output reports and model output reports. Assuming ten reports of the real system and ten reports of the model output, these reports can be handed to an engineer in a random order to analyze which of these reports is real and which is a model output result. If the engineer can not distinguish the reports correctly with a certain consistency, this test will not provide any evidence of model inaccuracy. This type of test is called Turing test.

### 9.4.2 Validation of the Factory Model

In the model validation phase, we followed the three point validation techniques mentioned in the previous sections. Our main point of reference is the usage of historical factory data for the validation of the input-output transformations. For this we define different parameters of interest for each model element. The following list defines the parameters $P_i$ for comparison against the historical data per model element in extracts:

- Shop floor level:
  - Cycle time per mask layer $CM$ and cycle time $CT$
  - Work in process $WIP$
  - X-factor $XF$
  - On-time delivery $OTD$ in weekly buckets and absolute

- Equipment level:
  - X-factor $XF_e$ of the equipment
  - Queue length $L_e$
  - Equipment state statistics
  - Operator interaction level
- Product level:
  - Cycle time per mask layer $CM_p$ and cycle time $CT_p$
  - Work in process $WIP_p$
  - X-factor $XF_p$
  - On-time delivery $OTD_p$ in weekly buckets and absolute
- Operator level:
  - Utilization $UT_o$
  - Availability $A_o$

Each parameter is compared with respect to its statistical properties, like the average, the median, the standard deviation and common quartiles like 50% and 90%. For this, different periods of time are defined and registered. The model input data for the different periods of times is generated automatically in case of well structured data from databases, and manually in case of non-structured data. For our AnyLogic model, we generate different MS Excel files to represent the model input and output data.

The validation itself is done by several manual structured walk throughs, comparing the model results and the historical data. For this, the MES or foreign reporting tools offer varying reporting functionality. Generally we define a validation function $VAL(Q_i, P_i)$, where $Q_i$ is the corresponding historical factory performance parameter. Each parameter $P_i$ has to satisfy the validation function:

$$VAL(Q_i, P_i) = \begin{cases} true & Q_i - T_i < P_i < Q_i + T_i \\ false & \text{otherwise} \end{cases} \tag{9.2}$$

The tolerance interval specified by $T_i$ is defined for each parameter $P_i$ in a manual manner. The validation function is added to the resulting MS Excel files for manual analysis. The definition and specification of the parameters for the validation and its tolerance is a very sensitive task. Validation can only be done at the same level as the data permits. In our case, we have to do several calibration tasks due to the following reasons:

- The operator data is not in a readable form, so manual operator assignment and group definition has to be done.
- The tracking of additional operations on metrology stations is not available, thus the utilization of the model was lower than in reality. For this we added some randomly distributed additional downs for reducing the availability of the tools.

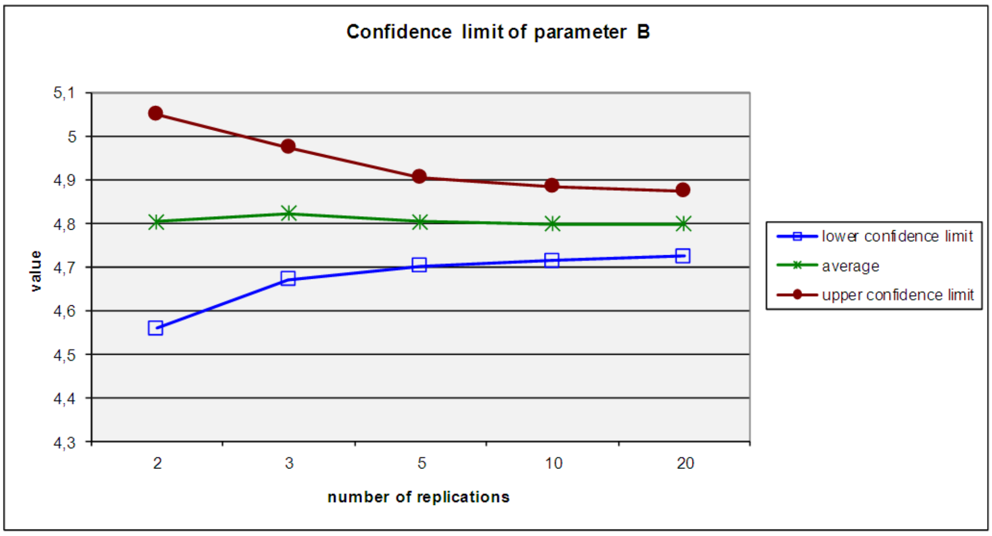


Figure 9.5: Confidence intervals evolution of parameter B

- Maintenance periods and activities per tool are not available in a readable form, thus we are only using historical models with the down periods that actually occurred.
- Equipment setups have to be assigned manually, because to no setup tracking is available.

Most problems are solved by defining the required model input manually through discussion with the line engineers and the responsible staff. We choose different periods of time with at least six months worth of real historical data. At the beginning of the model run period, the historical current facility state is transformed into the model. Therefore a warm up phase of the model is not necessary. Downs are determined by historical information about the equipment. Operating staff break behavior is also defined by fixed break times and work schedules. Therefore the historical validation of the model is done with a relatively low number of replications, due to the low influence of randomness in our historical model.

The number of replications is determined by confidence interval analysis. We took several replication counts into account. We evaluated the confidence intervals for $N = \{5, 10, 20, 50, 100\}$. For the first three elements, Table 9.2 introduces an example of two facility KPI.

The confidence intervals are calculated according to the standard normal distribution and the Student-T-Distribution. The Student-T-Distribution is used in case of a low numbers of experiments ($N < 30$). With an increasing number of replications, the distributions closely align. Figure 9.5 shows the evolution of the confidence intervals for parameter B for the Student-T-Distribution. At a replication count of ten, the confidence interval becomes stable. This behavior is seen at each of the KPI tested. Therefore we choose a replication count of ten for the model validation and the simulation analysis.

| Parameter | Average | Deviation | # Replications | $\alpha$ | t-value | confidence normal | confidence t |
|---|---|---|---|---|---|---|---|
| A | 2.6259 | 0.0203 | 5 | 0.05 | 2.776 | 0.01786 | 0.05658 |
|  | 2.6222 | 0.0209 | 10 |  | 2.262 | 0.01296 | 0.04733 |
|  | 2.6219 | 0.0200 | 20 |  | 2.093 | 0.0088 | 0.04203 |
| B | 4.8036 | 0.0366 | 5 | 0.05 | 2.776 | 0.0322 | 0.1019 |
|  | 4.7983 | 0.0377 | 10 |  | 2.262 | 0.0233 | 0.08532 |
|  | 4.7984 | 0.0357 | 20 |  | 2.093 | 0.0156 | 0.07474 |

Table 9.2: Confidence interval examples

| Model level | normalized model result parameter $P_i$ mean | normalized historical result $Q_i$ mean | Deviation |
|---|---|---|---|
| factory | 0.524 | 0.54 | 3% |
| technology_3EF23 | 0.54 | 0.57 | 5.5% |
| technology_3F | 0.538 | 0.52 | 4.4% |
| technology_4F | 0.478 | 0.48 | 0.4% |
| technology_18A | 0.53 | 0.55 | 3.78% |

Table 9.3: Example validation extract for a given period for the cycle time per mask mean for the main technologies

The validation of the model is illustrated in table 9.3, where the $CM$ is shown for different technologies. The average deviation is in general lower than 10% at factory level and technology level. Of course there are several products with a very low starting rate (about one to ten lots per month), which are having a higher deviation due to manual speed up in the line. The main products having 50 to 1000 lot starts per month are within the mentioned deviation, which is sufficient for our needs.

On equipment level, the main parameters like flow factor and queue length statistics are also within an acceptable deviation. The tolerances here depend upon the equipment type and the available data about setup management, down times and tracking of additional work. Especially manual inspection tools offer a higher deviation between model and reality. This originates from the large amount of additional activities such as line control. The influence of this deviation is low, therefore this deviation is acceptable.

# Part IV

# Production Control Strategy

Large increases in cost with questionable increases in performance can be tolerated only for race horses and fancy women.

*(William Thomson, Lord Kelvin (1824-1907))*

# 10 Introduction

## 10.1 Literature Review

The benefits of several facility performance improvement approaches have been elaborated in a large number of publications. Analytical approaches like scheduling are introduced as a base element for optimization of performance parameters like tardiness. Bruckner and Pinedo [Bru07, Pin02] use different mathematical algorithms to solve standard problems, which are not applicable to real world problems at run time. Real world problems are often too complex for an analytical calculation. Ouelhadj and Petrovic [OP09] classify scheduling algorithms into:

- Reactive scheduling, mostly used in manufacturing environments including dispatching approaches, at run-time.
- Predective-reactive scheduling , which calculates schedules for certain stages at defined events (e.g., equipment failures).
- Robust pro-active scheduling, which uses predetermined schedules based on certain predictability measures.

Today's most common techniques are reactive scheduling and predective-reactive scheduling. In a real environment, complex approaches like pro-active scheduling are often not applicable due to their analytical restrictions and huge calculation efforts. Real manufacturing systems offer a large amount of stochastic system changes and a high system variability over time. Scheduling requires a detailed prediction of the lot moves at the manufacturing area, which is very difficult due to the mentioned system instabilities. Nevertheless some applications could be established. An overview of scheduling algorithms concerning manufacturing is introduced by Ouelhadj and Petrovic [OP09].

Reactive scheduling techniques, particularly dispatching rules and policies are introduced and discussed in a wide range of papers. Wein [Wei88] presents simulation analysis of different lot release strategies in the semiconductor environment. The results show that different strategies have major impact on the KPI of the facility. Mittler and Schoemig [MS99] present performance evaluations for different dispatching rules at large semiconductor wafer fabrication facility models. As a main conclusion, the facility characteristics define the success or failure of a dispatching policy. These characteristics depend on various factors like the available equipment set or the product mix. Thus an individual analysis for the appropriate dispatching solutions is necessary.

For the semiconductor foundry business, the on-time delivery is an important aspect (see Section 10.2). Rose [Ros02, Ros03] analyzes different dispatching approaches optimizing the due date target of each lot. The evaluations show different problems like stability issues of the critical ratio rule. Furthermore Rose [Ros01] introduces the results of throughput oriented rules like shortest processing time first, with unstable evaluation results. The unstable behavior is confirmed by Bansal [Ros01, BHB00]. Larger studies are done e.g. by Ho and Tay [HT03] who apply common dispatching rules to several job shop problems. The papers introduce several advantages and disadvantages of common dispatching approaches. Fast calculation times and simple usage are accompanied by unstable and facility dependent behavior.

In reality more complex global dispatching approaches are rarely used. In contrast local optimizations on several tool groups and types are common. For example local optimizations for batching operations e.g., by Akcali et. al.[AUH$^+$00] are optimizing furnace operations with good success, or simulation based batching heuristic for bottleneck tools are discussed by Mönch and Habenicht [MH03]. Different line balancing approaches with the objective to prevent equipment starvation are introduced e.g. by Zhou [ZR09]. Line balancing without additional parameters like due date consideration shows a larger degradation of several factory performance parameters. Thus it can only be a part of a common approach for the whole facility.

The solutions mentioned here often concentrate on static optimization objectives, but do not take multiple changing targets into account. Often the benefits of these algorithms are shown by different theoretical models rather than using real world data with more variance and flexibility. Most of the time the theoretical models only use a reduced complexity like the often applied MIMAC[1] model suites, containing reduced factory data sets for classical wafer facilities with only a few number of products. The low-volume high-mix characteristic often has not been taken into account. The largest issue is the behavior of a real facility, which is different from the simplified model assumptions. At our facility model, different influences such as an unstable product mix or the fluctuating operating staff are main characteristics, which are not represented in further studies.

The transfer of the mentioned methods to a real environment is often not transparent and simple. As seen in literature sources, the success of the different approaches depends on the facility model characteristics. Boundary conditions like available data sources and management demands are important factors influencing the scheduling and dispatching policies in the real world.

Besides the dispatching policy, metrology operations in semiconductor manufacturing offer a high potential for optimizations. These operations are important elements of the whole production process for detecting problems on wafers. Often the processes are stable enough to allow a certain sampling mechanism to skip these operations. The durations of metrology operations can be in a range of several minutes to several hours.

In the mass production field, the simple definition of a sample rate as mentioned in [Stu09] is sufficient. In general a more complex selection process is necessary. In literature examples refer to certain processing steps or operations (e.g., see [ZS09] with focus on plasma etch). Global solutions for a fab-wide metrology sampling are introduced in [AHG07] by Hohlfeld, Barlovic and Good. The focus in this paper is a mass-production

[1] Measurement and Improvement of Manufacturing Capacities, see www.eas.asu.edu/~masmlab

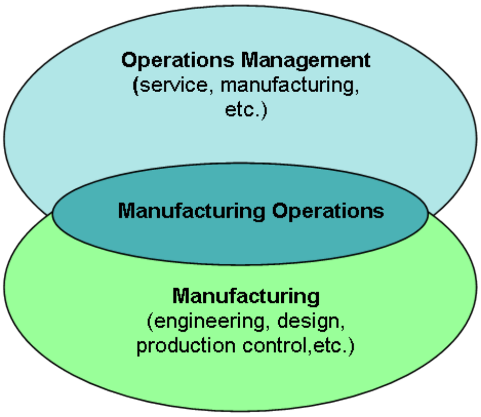

Operations Management
(service, manufacturing, etc.)
Manufacturing Operations
Manufacturing
(engineering, design, production control,etc.)

Figure 10.1: Manufacturing and operations view (according [HS01])

approach, optimizing the utilization of metrology operations and the cycle times of the lots. It is not applicable at the low-volume high-mix characteristics. The context of the sampling operations in a fab wide process control is introduced by Su et al. (see [AJSO08]). It offers a good understanding of the balance between the need of metrology operations for valuable process control and the problems of skipping these operations. In general metrology operation sampling is rather difficult at low-volume high-mix facilities compared to mass production environments.

## 10.2 A Brief Insight into Management Aspects

There are different views on the management of a SF. Hopp and Spearman[HS01] present two different aspects. Figure 10.1 illustrates the both main views of management decision aspects. The operations management is a technical view on the management processes of a company. It has major impact on the decisions at manufacturing level. Operation's influence and importance on the global competition of a company is significant. The operations actions are affected by three dimensions, which are applicable to most situations in manufacturing industries:

- **Cost:** The most traditional dimension of the market competition has been always a part of operations management. One important parameter is the unit cost. Efficient usage and utilization of different equipments, materials and humans are main objectives to keep costs low. Therefore management decisions tend to increase utilization and decrease the number of available resources to save cost.

- **Quality:** Quality is a key parameter for the competition on markets. There is a division between internal quality, not seen by the customer, and external quality, seen by the customer. The internal quality defines the process grades. The external quality is evaluated by the customer and describes the quality of the end product.

- **Speed:** Rapid development of new technologies and products combined with a quick delivery to the customer are the main characteristics of speed. Responsive delivery needs efficient manufacturing production control, reliable processes and an effective use of information technology systems. A low $CT$ and $OTD$ compliance are representatives for important KPI regarding speed.

As introduced in Chapter 1.1, the production control system is influenced by management demands. This results in two views on to the production process. The control perspective and the management perspective interact with their different goals and have to cooperate.

Management decisions are very dynamic and change over time. Different market situations force the management to consider each of the three aspects differently. In situations of excessive utilization of the facility, the attention is directed to speed. At low utilization, the attention is directed to lower cost. Therefore a dispatching strategy must consider these changing demands at a certain level.

# 11 Assessment of Dispatching and Scheduling

In this chapter, we introduce our combined dispatching policy approach solving several requirements of a typical SF with a manual operating and transport system.

## 11.1 Combined Dispatching Policy

### 11.1.1 Dispatching Policy Definition

The dispatching strategy presented here is a result of the analysis of a variety of dispatching rule approaches under usage of the introduced facility model. We consider different simple dispatching approaches like CR as a typical semiconductor rule for our starting point of analysis. Unfortunately these rules focus on single performance criteria and are usually not applicable to other objectives. Especially the CR rule tends to be unstable at high loads (e.g., see [Ros03]). Special approaches defined by Mönch and Habenicht [MH03] are more focused on a local point of view. We are striving for a facility-wide solution including the option of local optimizations. Our research has shown the significant influence of boundary conditions on the quality of results generated by simple dispatching rules. The unstable behavior can be seen at a broad range of different dispatching rules.

Because of the multi-objective characteristics, a combination of several dispatching policies with different policy-specific targets is chosen for a detailed analysis. We define a combined dispatching strategy which has rarely been applied at real systems according to literature (e.g., see [DCFS01, PBF$^+$08]). In case of Dabbas et al. [DCFS01] a mass production example is defined and implemented with significant improvements, but not using freely configurable objectives and dispatching sets. In our case, appropriate rules and strategies are currently almost unavailable, in particular for the high-mix low-volume foundry business.

A variable combination of different rules offers several advantages. Depending on the defined objectives, the influence of each rule can be evaluated to find a semi-optimal solution. A major task is the definition of the influence. In case of a manual definition, the influence to the facility is unknown. In general there is no linear connection between the rule impact and influence of the rule on the facility.

The influence can change over time, depending on the current facility state and the management demands. The influence of the factory state on the optimization solution is often not considered in literature. In order to find optimal solutions for each facility state, optimization periods have to be defined. This element is introduced in Section 11.2.

The rule implementations can be very simple in case of simple strategies like due date (e.g., EDD) or throughput (e.g., SPT) oriented approaches. Each of these dispatching criteria optimizes at least one parameter of interest. The applied dispatching policies are introduced in Section 11.1.2.4.

In our case, the different priorities $P_k$ originating from the different dispatching rules $k$ are combined in a linear way for calculation a final lot priority $P_{Lot}$:

$$P_{Lot} = \sum_{k=1}^{N} w_k P_k \tag{11.1}$$

where $0 \leq P_k \leq 1$, $\sum w_k = 1$ and thus $0 \leq P_{Lot} \leq 1$ . The weight of each dispatching policy can be defined by setting the corresponding weight. A higher weight defines a greater significance for the rule. This allows a high variability in optimizing the performance measures regarding defined objectives, e.g., by management. In reality, many concurrent goals for optimization exist.

## 11.1.2 Main Elements and Definitions

### 11.1.2.1 Equipment Setups

For the simulation analysis we use a time-based setup control mechanism for definition of the preferred setup $S_i$ for equipment $i$:

$$S_i(t) = \begin{cases} S_{Curr,i} & \text{when case 1} \\ S_{L,i} & \text{when case 2} \end{cases} \tag{11.2}$$

The current setup of the equipment $S_{Curr,i}$ is chosen in case of available lots which can be processed within the current setup. In case of lots with a high manual defined priority, these lots are processed next. If another workstation of the recipe has the demanded setup state of a lot, the lot is processed on the other equipment and the next lot is taken.

### 11.1.2.2 Time Bound Sequences

Time bound sequences are a very complicated element of production control. In our case, we use the following policy. For lots in a time bound sequence, the lots are prioritized according the maximal waiting time respecting the move-in time of the lot into the current step. For each of the following steps within the time bound sequence, the current load is calculated. The current load of a step $k$ can be calculated as

$$Load_k = \frac{\sum_{n=1}^{N} \sum_{l=1}^{L} t_{pw} r_w}{N} \tag{11.3}$$

where $N$ is the number of equipments available for the step, $L$ the number of lots in the queue in front of an equipment $n$, $t_{pw}$ is the theoretical raw process time per wafer and $r_w$ is the wafer count of lot $l$. If the calculated load of one step within the time bound sequence is larger than the maximal waiting time of the lot for this step, the lot is stopped

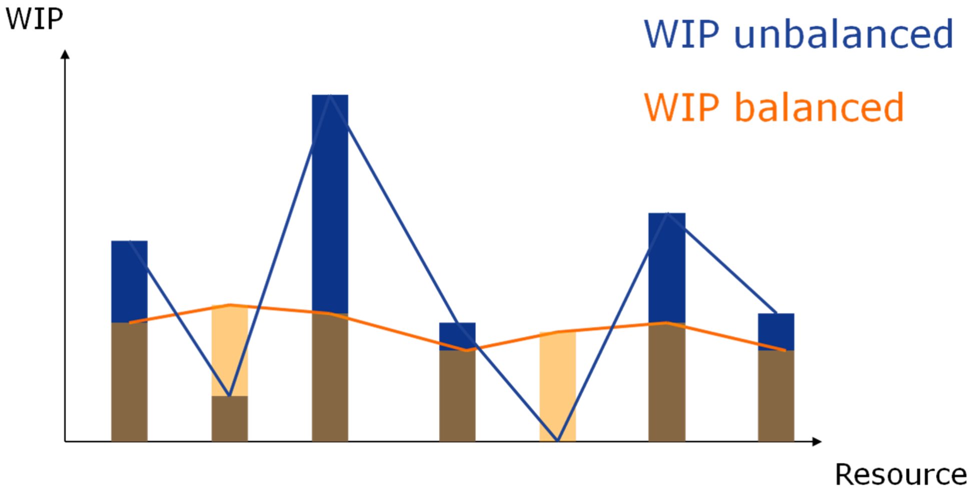


Figure 11.1: General line balance objective

temporarily to avoid violation of the defined maximal waiting time. During the next dispatch action, this is analyzed again.

#### 11.1.2.3 Line Balance

The line balance is important to avoid starvation of tools and is used to reduce $WIP$ fluctuation at each stage or equipment. Figure 11.1 illustrates the main objective of line balance. The blue lines illustrate an unbalanced $WIP$ causing starvation at some tools and an overload at other tools. In general this behavior wastes capacity due to the starved tools that can not process. The goal is to smooth the $WIP$ over all resources (yellow line), to reduce the starvation probability.

We use two different approaches for line balance, the first one defines a goal $WIP_{Goal,e}$ with a minimum level $LWL_e$ and a maximum level $UWL_e$ per equipment $e$. In the following sections, we denote this approach with $LB_{WIP}$. If there is a violation of the defined limits, the priority of the lots at the previous stages $e-1$ is either decreased or increased. In general the normalized priority $P_R$ of the $LB_{WIP}$ rule for lot $L_i$ for equipment $e$ can be calculated as follows:

$$P_{R_{LB}} = \begin{cases} 0.5 + \left(\frac{LWL_{e-1} - WIP_{e-1}}{WIP_{Goal,e-1} - WIP_{e-1}} * 0.5\right) & \exists e-1 : LWL_{e-1} > WIP_{e-1} \\ 0.5 & \forall e-1 : UWL_{e-1} \leq WIP_{e-1} \leq UWL_{e-1} \\ 0.5 - \left(\frac{WIP_{e-1} - UWL_{e-1}}{WIP_{e-1} - WIP_{Goal,e-1}} * 0.5\right) & \exists e-1 : UWL_{e-1} < WIP_{e-1} \end{cases} \tag{11.4}$$

The limits can either be generated from a facility model determining the optimal limits per tool group or by manual definition. At our production characteristics, we use a general $WIP$ goal instead of product dependent specifications as used in [DF03]. Inhomogeneous product load per stage does not allow product specific settings. A main issue for this approach is the determination of the $WIP$ levels per equipment. This is a very sensitive task.

Therefore we use the work load instead of defined $WIP$ targets. We denote this approach with $LB_{Load}$. The workload mechanism is applied to each equipment $e$, where the workload can be calculated as follows:

$$Load_e = \sum_{l=1}^{L} t_{pw,l} r_{w,l} f_{r,l} \tag{11.5}$$

where for each lot $l$ the workload consists of the theoretical processing time per wafer $t_{pw,l}$, the wafer count $r_{w,l}$ and a workload reduction factor $f_{r,l}$. The workload reduction factor reduces the workload caused by the equipment according the number of other equipments available. For each lot a priority can be calculated referring to the workload of the current and scheduled next equipments. A higher priority is calculated in case the workload of the next equipments is lower. A lower priority is calculated in case of the workload of the equipments of the next stage being higher. For this approach a manual definition of $WIP$ levels is not necessary. The differences between both approaches are analyzed in Section 11.1.3.

#### 11.1.2.4 Dispatching Rules

The dispatching policies applied in this work are described in the following list[1]:

- **FIFO**: The first in first out rule is the classical starting point for dispatching rule analysis. The rule offers a very low variability of all parameters of interest and can be described as very fair. The lot with the longest waiting time in the equipment queue is taken next. The normalized priority $P_R$ of the FIFO rule for lot $L_i$ can be calculated as follows:

  $$P_{R_{FIFO}}(L_i) = 1 - \frac{i}{n} \tag{11.6}$$

  where $i$ is the current position of $L_i$ in the queue and $n$ is the current count of all elements in the queue.

- **SPT**: The shortest processing time first rule sort lots with the shortest processing time to be processed next. Therefore the throughput of the equipment and the facility is optimized. The rule can cause stability problems (e.g., see [Ros01]) in case of variation of processing times on a highly utilized equipment. The normalized priority $P_R$ of the SPT rule for lot $L_i$ can be calculated as follows:

  $$P_{R_{SPT}}(L_i) = 1 - \frac{t_i - t_{min}}{t_{max} - t_{min}} \tag{11.7}$$

  where $t_i$ is the processing time of $L_i$.

[1] The standardization of each rule follows according the reverse priority: the lower the priority number, the higher the importance.

- **CR**: The critical ratio rule is widely used in factory environments and is based on the due date and the remaining processing time of the lot. It optimizes two parameters, the throughput and the on-time delivery. The rule tends to be unstable in case of unrealistic due date targets (e.g., see [Ros02, Ros03]). The normalized priority $P_R$ of the CR rule for lot $L_i$ can be calculated as follows:

$$P_{R_{CR}}(L_i) = \begin{cases} N\left(\frac{1+d_{due}-d_{now}}{1+t_{remain}}\right) & d_{due} > d_{now} \\ N\left(\frac{d_{due}-d_{now}}{1+t_{remain}}\right) & \text{otherwise} \end{cases} \tag{11.8}$$

  where $N(x)$ is the normalization function for normalizing the priority values, $d_{due}$ is the due date of the lot, $d_{now}$ is the current date and $t_{remain}$ is the remaining processing time of the lot.

- **EDD**: The earliest due date rule is often applied in semiconductor environments for optimizing the on-time delivery from a global point of view. The lot with the closest global due date is processed next. The normalized priority $P_R$ of the EDD rule for lot $L_i$ can be calculated as follows:

$$P_{R_{EDD}}(L_i) = 1 - \frac{d_i - d_{min}}{d_{max} - d_{min}} \tag{11.9}$$

  where $d_i$ is the due date of $L_i$.

- **ODD**: The operation due date rule is a version of the EDD rule defining local due dates for the lots. The local due date $d_{i,s}$ at stage $s$ can be calculated as

$$d_{i,s} = \sum_{d=1}^{s} t_{p,s} * XF_t = \sum_{d=1}^{s} t_{p,s} * \left(\frac{d_{due} - d_{start}}{t_{RPT}}\right) \tag{11.10}$$

  where $t_{p,s}$ is the theoretical raw processing time of stage $s$, $d_{due}$ the global due date of $L_i$, $d_{start}$ the start date of $L_i$ and $t_{RPT} = \sum_{s=1}^{N} t_{p,s}$ the theoretical raw processing time of the whole process of $L_i$ with $N$ stages. The ODD rule is more stable than EDD (e.g., see [Ros03]). The normalized priority $P_R$ of the ODD rule for lot $L_i$ can be calculated as follows:

$$P_{R_{ODD}} = 1 - \frac{d_{i,s} - d_{min}}{d_{max} - d_{min}} \tag{11.11}$$

- **LB**: The line balance algorithm is described in Section 11.1.2.3.

### 11.1.3 Simulation Analysis

Our simulation analysis is based on models which represent several historical data sets. Several scenarios are evaluated. The most interesting three scenarios are presented in the following sections.

| Number | Experiment | Setup avoidance | Applied Rules | Weights |
|---|---|---|---|---|
| 1 | FIFO Reference | yes | FIFO | 1.0 |
| 2 | ODD | yes | ODD | 1.0 |
| 3 | EDD | yes | EDD | 1.0 |
| 4 | SPT | yes | SPT | 1.0 |
| 5 | LB+FIFO | yes | FIFO | 0.5 |
| | | | $LB_{WIP}$ | 0.5 |
| 6 | FIFO without SA | no | FIFO | 1.0 |

Table 11.1: Scenario 1 - Experimental set

All simulation experiments include 10 replications with historical models. The simulation run length is up to 6 months. The model load ranges between 70% to 98%. All dispatching rules are used in combination with the setup avoidance technique described in Section 11.1.2.1 (with the denoted exceptions). We analyzed both line balancing techniques introduced in Section 11.1.2.3. The model represents the corresponding historical periods with its equipments, operators and lots (see introduction of the simulation model in Section 7.2). As with all simulation results, the numerical values are normalized with respect to the reference simulation run (FIFO Reference). The WIP levels of the $LB_{WIP}$ rule are determined by historical queue length averages per equipment. This approach is used as an approximation for estimating the of WIP targets.

#### 11.1.3.1 Scenario 1

The first scenario represents a typical foundry product mix with 50% of short term products and 50% of long term products. This scenario is used to evaluate the general influence of common dispatching rules to our model. The set of experiments is illustrated in Table 11.1. Besides the dispatching rules, the influence of setup avoidance is analyzed with the FIFO dispatching rule.

The results of the simulation experiments for the 98% load case are illustrated in Table 11.2. The table contains the most important KPI WIP, CM, XF and OTD. The green values illustrate improvements from the reference value, the red values indicate results below expectations. Blue elements depicting the best results for each KPI.

At this scenario, the best results are generated by the ODD dispatching rule and the combined rule of $LB_{WIP}$ and FIFO. In general the ODD rule reduces the deviation of the different KPI's to 30% to 50%. This behavior can also been seen in other literature sources like Rose [Ros03]. Due to the objective of this rule being due date oriented, the rule offers the best OTD performance. The combined approach illustrates a potential regarding multiple objectives. FIFO as a fair rule and $LB_{WIP}$ for avoiding tool starvation offers a better performance than ODD at the non OTD KPI like CM or XF. The WIP is reduced by 5%, the CM by 3% and the XF by 5% in comparison to ODD. With no due date oriented rule being set within the combined dispatching, the OTD is decreased by

| KPI / Experiment | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| WIP average | 1.00 | 0.99 | 1.04 | 1.39 | 0.94 | 3,96 |
| CM average | 1.00 | 0.98 | 1.01 | 1.21 | 0.95 | 3.9 |
| CM deviation | 1.00 | 0.72 | 1.01 | 3.18 | 1.00 | 8.47 |
| XF average | 1.00 | 0.97 | 1.00 | 1.22 | 0.94 | 3.87 |
| XF deviation | 1.00 | 0.54 | 1.11 | 4.55 | 0.97 | 10.92 |
| absolute OTD | 94.4% | 99.3% | 95.3% | 63.4% | 97.4% | 5.9% |

Table 11.2: Scenario 1 - Simulation results for 98% load

1.9%. The EDD dispatching rule with the objective of global due date targets decreases the KPI (except OTD) in comparison to the reference in this case.

Setup avoidance is essential for achieving sufficient performance. Without this mechanism, the performance of the facility is degraded by 300% to 800%. The reasons are the setup equipments, which cause long waiting times due to the numerous setup changes. For example, setup change of the dopant source at implantation equipment has a duration between 0.5 to 1.5 hours in our case. Therefore the performance degradation is very high.

The evolution of the CM over the load is illustrated in Figure 11.2. The average value of the KPI is in close proximity in lower loading scenarios. At higher loads, the SPT rule becomes inefficient. The average CM increases up to 21% above the reference at a 98% load. Deviation is also increased dramatically. This is caused by highly utilized tools with two processing time types. This includes very short and very long processes and can be found at measurement equipment. The short processes are preferred, therefore the more time consuming processes have to wait. Due to the high utilization, short processing lots are preferred most of the time.

The ODD rule offers the lowest deviation in all load cases. The deviation is nearly constant and shows a small increase between 91% to 94%. This load area is the threshold, where batching equipments become bottlenecks and therefore, lot ordering is changed due to batching to a higher degree. This influence is reduced at higher loads, because more lots are available for batch building, therefore lots with higher priority are processed in a more appropriate sequence.

At lower loads, the combined approach offers no improvement against the other dispatching rules at the CM average. With increasing load, the rule becomes more efficient until an improvement of 6% to the reference in case of the maximum load.

The first scenario shows the potential of due date oriented rules, especially the ODD rule shows improvements. The combined approach without any due date oriented rules demonstrates the potential of this approach. Except the due date objective, all KPI improve between 3% to 6% in comparison to the reference.

#### 11.1.3.2 Scenario 2

The second scenario shows the improvements of a combination of multiple dispatching rules. The product mix is taken from real historical data with real due dates of each lot. The

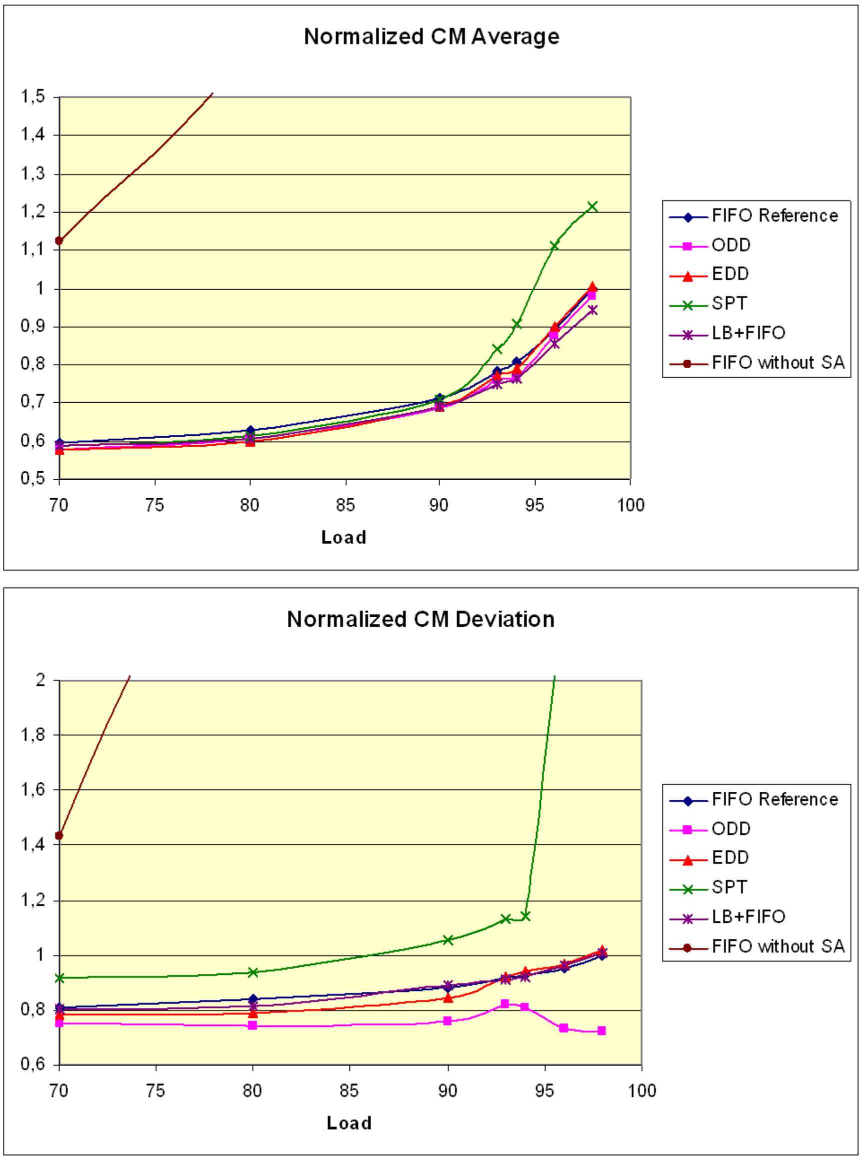


Figure 11.2: Cycle time per mask simulation result - Scenario 1

| Number | Experiment | Setup avoidance | Applied Rules | Weights |
|---|---|---|---|---|
| 1 | FIFO Reference | yes | FIFO | 1.0 |
| 2 | ODD | yes | ODD | 1.0 |
| 3 | CR | yes | CR | 1.0 |
| 4 | Combined Equal | yes | FIFO | 0.2 |
| | | | ODD | 0.2 |
| | | | EDD | 0.2 |
| | | | $LB_{WIP}$ | 0.2 |
| | | | SPT | 0.2 |
| 5 | Combined Optimal | yes | FIFO | 0.1 |
| | | | ODD | 0.2 |
| | | | EDD | 0.5 |
| | | | $LB_{WIP}$ | 0.2 |
| | | | SPT | 0.0 |

Table 11.3: Scenario 2 - Experimental set

set of experiments is illustrated in Table 11.3. In this analysis, we combine five different rules containing the due date objectives (global and local), the line balancing objective (avoid starvation) and the throughput objective (optimize throughput). The influence of the weights is evaluated in two different experiments. The starting point is the assignment of equal weights for each of the rules. For illustrating a more optimal solution, we use the weights calculated in Section 11.2.2. The weights are determined by an optimizer with the objective to minimize the average and the deviation of the KPI CM, XF, WIP. Furthermore the absolute number of late lots is also to be minimized.

The results of the simulation experiments for the 98% load case are illustrated in Table 11.4. The ODD rule performance decreases common KPI like CM or WIP by about 4% to 5%. The OTD is increased to 93.5%. The reference case has 11% late lots. The worst results are generated by the CR rule. At high loads, the rule becomes very unstable. This behavior is familiar from literature. Rose [Ros02] found similar results at high loads. The degree of late lots increased to 42.5%. That amount is not acceptable at real facilities. Only the deviation of the XF is minimized. This results from the objectives due dates and remaining processing time. Therefore the CR rule is not taken into the rule combination.

The best results are generated by the optimized combined solution. In general the combination of rules offer an average performance improvement of 3% to 9% at the various KPI at 98% load. The non-optimized solution shows improvements of 4% of the average of CM and XF. The optimization of the weights shows an improvement of 3% to 5% of the various performance parameters.

During the simulation experiments the equipment behavior is evaluated. To represent the local improvements of equipment level, the average queue length of the model as well as the average queue waiting time is collected and analyzed. A reduction of 18% in the average

| KPI / Experiment | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| WIP average | 1.00 | 1.05 | 1.13 | 0.96 | 0.93 |
| CM average | 1.00 | 1.04 | 1.14 | 0.95 | 0.91 |
| CM deviation | 1.00 | 0.91 | 0.95 | 0.94 | 0.93 |
| XF average | 1.00 | 1.05 | 1.14 | 0.96 | 0.91 |
| XF deviation | 1.00 | 0.88 | 0.86 | 0.97 | 1.00 |
| absolute OTD | 89.0% | 93.5% | 57.5% | 96.4% | 96.8% |
| Queue length average | 1.00 | 0.99 | 1.12 | 0.83 | 0.82 |
| Queue waiting time average | 1.00 | 1.05 | 1.23 | 0.9 | 0.92 |

Table 11.4: Scenario 2 - Simulation results for 98% load

queue length indicates a more stable and balanced line in comparison to the reference. The waiting times of the lots in front of a work station are also reduced by 10%.

The evolution of the CM over the load is illustrated in Figure 11.3. The combination of multiple dispatching rules shows an improvement at each load case simulated. At lower model loads, the average CM is 5% lower than the reference. The performance gap is constant during the different load cases at the non-optimized combined approach. The optimization is reasonable at higher loads. At this scenario, at the model load of 80%, the gap between the optimized and non-optimized case increases. At 98% load, the average CM is about 4%.

The deviation of the CM decreases with higher loads in this scenario. At all loads, the ODD rule has the lowest deviation. A decrease of the deviation is also achieved by the combined approach. The gap between pure ODD and the combination is about 2% to 12%, depending on the use case.

The OTD is one of the most interesting KPI in the foundry business (see Figure 11.4). The CR rule degrades the OTD to an unacceptable level. The ODD rule reduces the amount of lots within one week delay to 1%. At reference case, 5% of the lots are within this level. A differentiation between the non-optimized and optimized case is not visible.

The standardized mean queue length (see Figure 11.5) is an indicator for the WIP distribution within a facility. The lower the queue length, the more balanced the WIP. The combined solutions show a similar behavior like at the CM evolution. At low loads of 70%, the improvement is also about 5%. There is no gap between the ODD rule and the reference. At high loads, the difference between the optimized and non-optimized case is very small.

At equipment level, various reductions are visible (see Figure 11.6). At batching equipments, there are no reductions identifiable rather than a small increase of about 1% to 3%. The increases are a result of the line balancing activities reducing equipment starvation. The batch equipments are fed on a more regular basis with lots. As a result, batch building times increase. For the other tools, e.g. etch operations with low processing times and a high throughput, a huge reduction of the presented parameter of about 70% to the reference is calculated. At implantation tools, a reduction of about 30% is detectable.

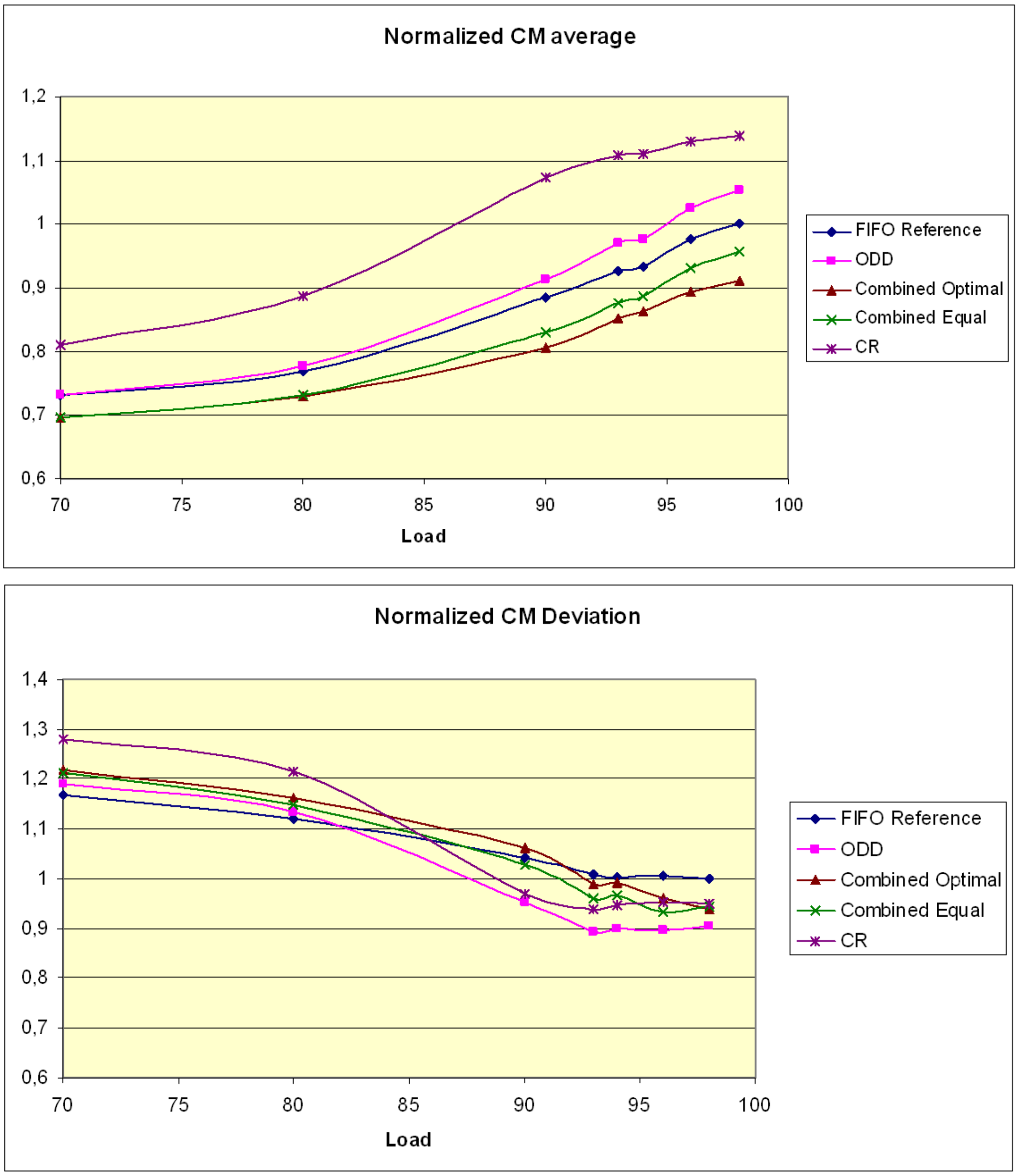


Figure 11.3: CM combined simulation result - Scenario 2

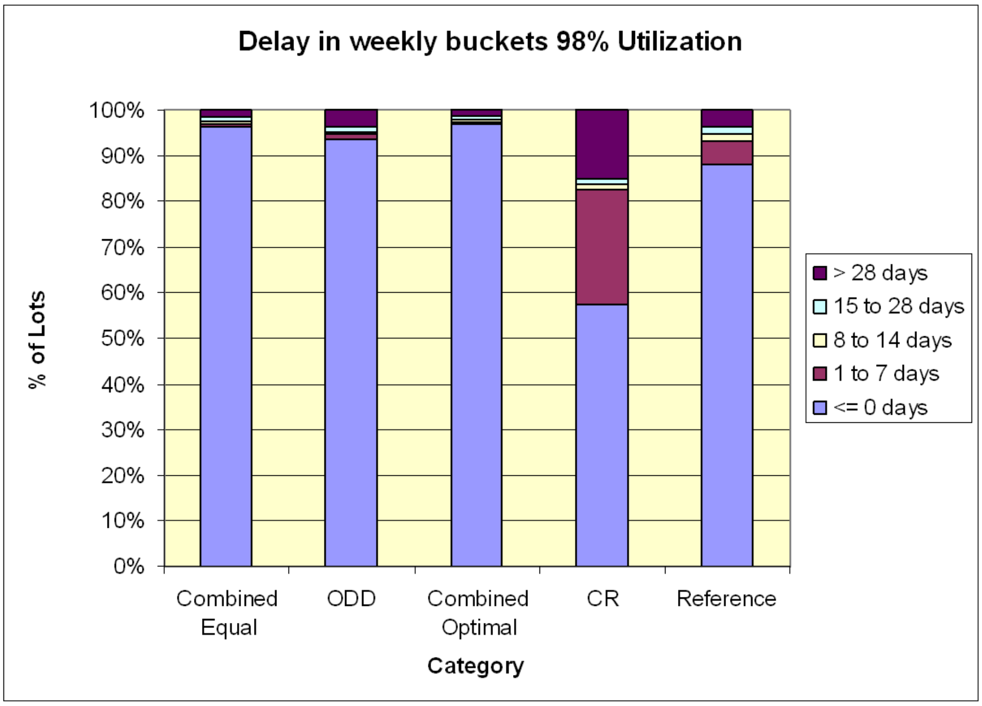


Figure 11.4: Delay in weekly buckets combined simulation result - Scenario 2

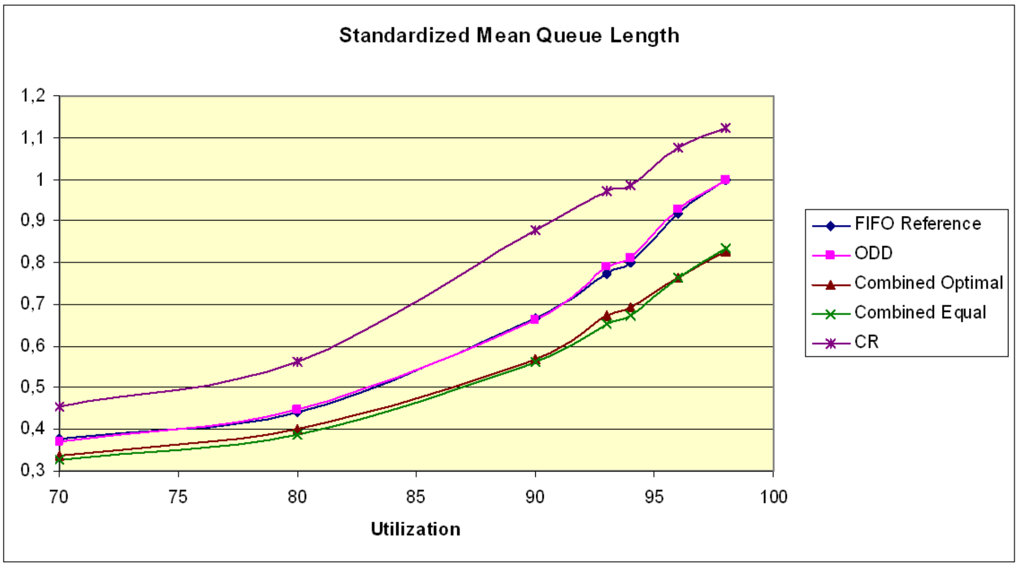


Figure 11.5: Queue length behavior at combined simulation result - Scenario 2

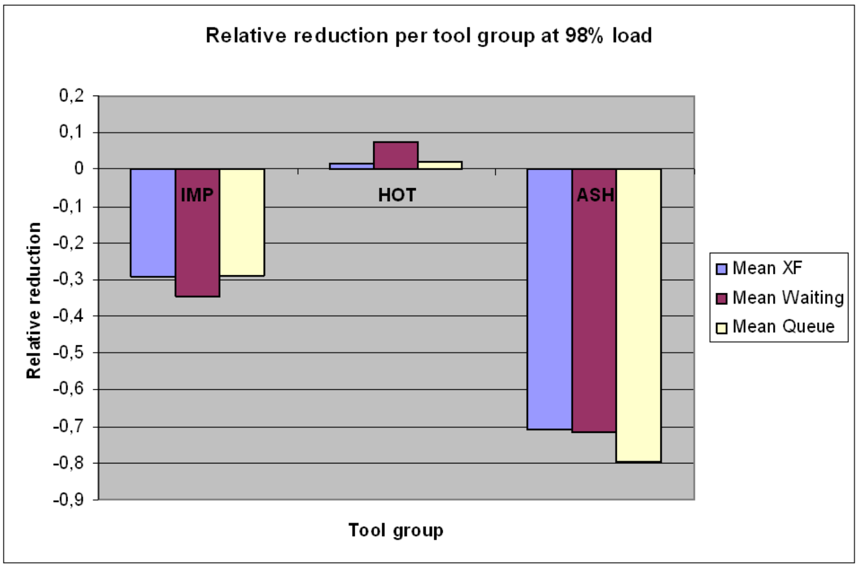


Figure 11.6: Relative reduction of the average queue length, average XF and the average waiting time at the tool groups high temperature (HOT), implant (IMP) and etch (ASH) at 98% - Scenario 2

The second scenario shows the potential of the combined approach. At various KPI, an improvement between 3% to 10% is visible in comparison to the reference case. The optimization allows the adaption of the weights to the current facility state. In general the optimization has more effects at higher loads rather than in the low load area. Improvements of about 3% to 5% are noticeable.

#### 11.1.3.3 Scenario 3

This scenario has its focus on the different line balancing techniques. We compare the $LB_{WIP}$ algorithm with the $LB_{Load}$ algorithm. The set of experiments is illustrated in Table 11.5. Further simulations show, that a single application of the line balancing approaches degrades the KPI at different levels in comparison to the other dispatching rules. Therefore we combine both approaches with FIFO. In addition, we evaluate the influence of the optimization in comparison to the different LB approaches. The weights presented in the table are a result of two optimizations with the same objective function used in Scenario 2. The test scenario is based on historical data from a different historical time period as in Scenario 2.

The results of the simulation experiments for the 98% load case are illustrated in Table 11.6. The differences between the WIP based and Load based dispatching approaches are very small. In combination with FIFO, the deviation of the $LB_{Load}$ rule is about 10% higher in case of CM and XF. The optimization reduces the differences between both rules. The results are nearly equal. Therefore the advantage of using of the equipment load

| Number | Experiment | Setup avoidance | Applied Rules | Weights |
|---|---|---|---|---|
| 1 | FIFO Reference | yes | FIFO | 1.0 |
| 2 | FIFO + $LB_{WIP}$ | yes | FIFO | 0.5 |
| | | | $LB_{WIP}$ | 0.5 |
| 3 | FIFO + $LB_{Load}$ | yes | FIFO | 0.5 |
| | | | $LB_{Load}$ | 0.5 |
| 4 | Optimal $LB_{WIP}$ | yes | FIFO | 0.2 |
| | | | ODD | 0.0 |
| | | | EDD | 0.45 |
| | | | $LB_{WIP}$ | 0.35 |
| | | | SPT | 0.0 |
| 5 | Optimal $LB_{Load}$ | yes | FIFO | 0.25 |
| | | | ODD | 0.0 |
| | | | EDD | 0.6 |
| | | | $LB_{Load}$ | 0.15 |
| | | | SPT | 0.0 |

Table 11.5: Scenario 3 - Experimental set

| KPI / Experiment | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| WIP average | 1.00 | 0.99 | 0.99 | 0.92 | 0.92 |
| CM average | 1.00 | 0.98 | 0.98 | 0.87 | 0.87 |
| CM deviation | 1.00 | 1.06 | 1.14 | 0.85 | 0.85 |
| XF average | 1.00 | 0.99 | 0.98 | 0.87 | 0.88 |
| XF deviation | 1.00 | 1.03 | 1.13 | 1.09 | 1.11 |
| absolute OTD | 86.6% | 86.1% | 86.2% | 99.1% | 99.0% |

Table 11.6: Scenario 3 - Simulation results for 98% load

prevails. No WIP targets have to be defined, therefore this rule is to be used in further implementations.

The evolution of the XF average shows a similar behavior (see Figure 11.7). The combined approaches offer a performance improvement of about 3% at low load and 12% at high loads. A combination with FIFO only does not provide massive improvements. Here the performance gap is between 2% and 4% in comparison to the reference.

### 11.1.4 Summary

In general, the combined approach allows the usage of several different dispatching rules with a defined weight. Several simulation studies show average improvements of 3% to 10% at various cases to the reference implementation. Bottlenecks and non-bottleneck

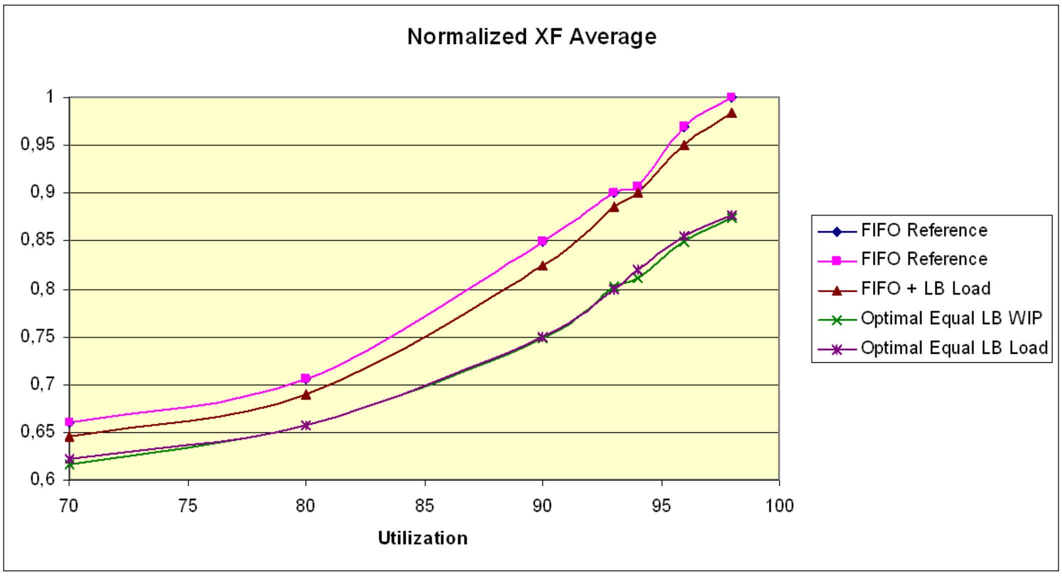


Figure 11.7: XF result - Scenario 3

tools are balanced in a reasonable way. The stability of the whole line increases. The gap between the non-optimized (equal) and optimized weight determination is about 3% to 5%. Therefore it is reasonable to use an optimization algorithm to calculate weight combinations on the basis of an objective function.

## 11.2 Optimization of the Combined Approach

In this section, we introduce optimization aspects of the dispatching approach.

### 11.2.1 Introduction

Optimization is used in a wide field of mathematics, science and management for providing and generating optimal solutions for a given problem. Optimization is a key tool for various problems with a wide range of different well-developed algorithms. Often the research regarding optimization deals with tasks in which all aspects of the system are known in detail. Therefore the performance of the system under research can be evaluated exactly. In contrast simulation operates with random variables. Thus results of a simulation are random values.

An optimization problem consists of the following elements (which can also be applied to simulation studies, see [Ros08a]):

- The objective function $Y = O(X)$ in abstract or defined way.
- The set of changeable input variables $X$, also called search space.

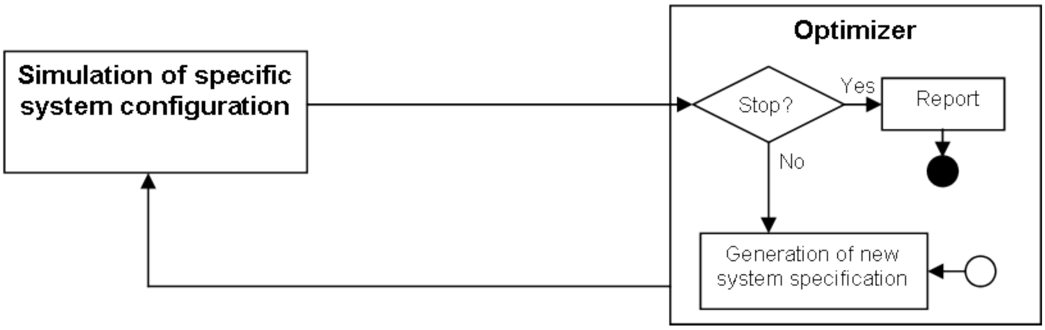

Simulation of specific system configuration
Optimizer
Stop?
Yes
Report
No
Generation of new system specification

Figure 11.8: Interaction of simulation and optimization

- The set of the output variables to be optimized $Y$, also called candidate solutions.
- The optimization result function $V(X, Y)$ determining the level of optimality of each solution, it describes the space in which the results are compared to each other.
- The set of external constraints $C$ which define constraints regarding the input and output variables $X$ and $Y$.

Optimization tasks can be divided into several groups according to their properties:

- Structure of the search space $X$, e.g. a discrete or continuous search space.
- Structure of the optimization problem, e.g. a convex surface.
- Structure of the solution space.

### 11.2.2 Optimization at Simulation Systems

Methods for optimization generally involve a simulation of a certain sequence of system configuration variables in a model, the evaluation of the result and the calculation of new combinations. There is only a very practical issue, involving questions like how to manage the several sets of input variables and setting the next configurations. Performing this manually is obviously not possible. Thus a large amount of today's simulation packages offer optimum-seeking mechanisms within the software. Figure 11.8 illustrates the interaction between these two elements.

In general optimization packages are separated from the simulation program. The optimizer is responsible for choosing the sets of variables for model input, the evaluation of the model results and for termination of the whole process. The simulation system is responsible to generate the model result for a given configuration. A wide range of different optimization packages is used in simulation software today. A short overview is given in Table 11.7.

In our research, we use the OptQuest optimization package from AnyLogic for the first evaluation of the optimization potential. It uses a high variety of different optimization techniques (for more detail we refer to [MAN11, Lag11]). Unfortunately, it is not apparent

| Package | Support by | Algorithms |
|---|---|---|
| AutoStat | AutoMod, AutoSched | Evolution strategies |
| OptQuest | Arena, AnyLogic | Tabu search, scatter search, neural network |
| SimRunner2 | ProModel, ServiceModel | Evolution strategies, genetic algorithm |

Table 11.7: Optimization packages (adapted from [LK00])

which algorithms are used in detail. Furthermore, access to the simulation core is not possible from foreign systems, which is required at system implementation. Therefore we switch to our own implementation of an optimizer in combination with the open source package JSL simulation library (see Chapter 13).

#### 11.2.2.1 Optimization with OptQuest

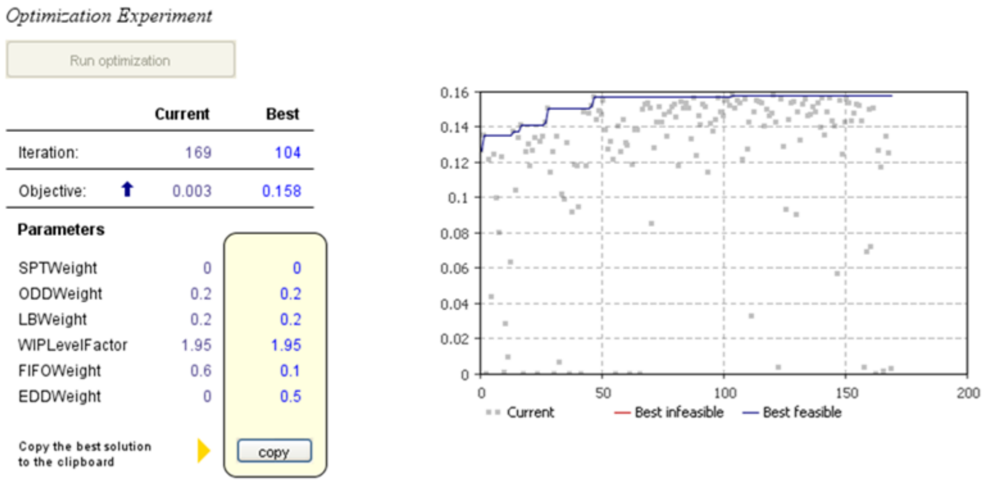
For illustration of the optimization potential, we first evaluate our approach with OptQuest. For example we define an objective function to minimize the $WIP$, the average and the deviation of the $CM$, the average and the deviation of the $XF$ and the absolute number of late lots $L$:

$$\begin{aligned} Y = O(WIP, CM_{Mean}, CM, XF_{Mean}, XF_{Deviation}, L) = \\ w_1 N(WIP) + w_2 N(CM_{Mean}) + w_3 N(CM_{Deviation}) + \\ w_4 N(XF_{Mean}) + w_5 N(XF_{Deviation}) + w_6 N(L) \quad (11.12) \end{aligned}$$

where the function $N(X)$ normalizes the given values to the reference run with FIFO dispatching at 98% load. The weights $w_i$ determine the importance of each factor, where $\sum w_i = 1$. In our example, we choose $w_i = \frac{1}{6} \forall w_i \epsilon W$, which means their influence is equal. For better illustration of the optimization process, we calculated the normalization function $N(X)$ in a way that this function is to be maximized. Figure 11.9 illustrates the example. The OptQuest engine calls the model about 170 times, the resulting weights are:

- $w_{SPT} = 0.0$
- $w_{ODD} = 0.2$
- $w_{LB} = 0.2$
- $w_{FIFO} = 0.1$
- $w_{EDD} = 0.5$

The average improvement of this optimization compared to the equal weight experiment is about 3% to 4% at several performance measures in this case with the mentioned objective function (see simulation experiments in Section 11.1.3).



| | Current | Best |
|---|---|---|
| Iteration: | 169 | 104 |
| Objective: ⬆ | 0.003 | 0.158 |
| **Parameters** | | |
| SPTWeight | 0 | 0 |
| ODDWeight | 0.2 | 0.2 |
| LBWeight | 0.2 | 0.2 |
| WIPLevelFactor | 1.95 | 1.95 |
| FIFOWeight | 0.6 | 0.1 |
| EDDWeight | 0 | 0.5 |

Figure 11.9: Optimization procedure in AnyLogic

#### 11.2.2.2 Optimization with the Evolutionary Approach

**Introduction** For problem optimization, a wide range of different solutions is available. Spall [Spa05] and Chong et al.[CZ04, Spa05] define different classes and algorithms for optimization:

- Deterministic
  - Numerical Algorithms
  - Branch and Bound
- Probabilistic
  - Monte Carlo Algorithms
    - Evolutionary Algorithms (e.g., Genetic Algorithm)
    - Simulated Annealing
    - Tabu Search
  - Artificial Intelligence

Deterministic approaches are not applicable to our problem. The complexity as well as the stochastic influences are not representable with these kind of solutions. Stochastic heuristics are often applied in the simulation applications. In literature, evolutionary algorithms are widely used to solve scheduling problems in the manufacturing area. Frequent reference in literature can be found regarding job shop problems. For example ElMaraghy et al. [EPA00] use a genetic algorithm to solve staff assignment problems in the manufacturing area. Ho and Tay [HT03] applied an evolutionary algorithm to find new analytical dispatching rule sets based on simple mathematical operations to solve theoretical job

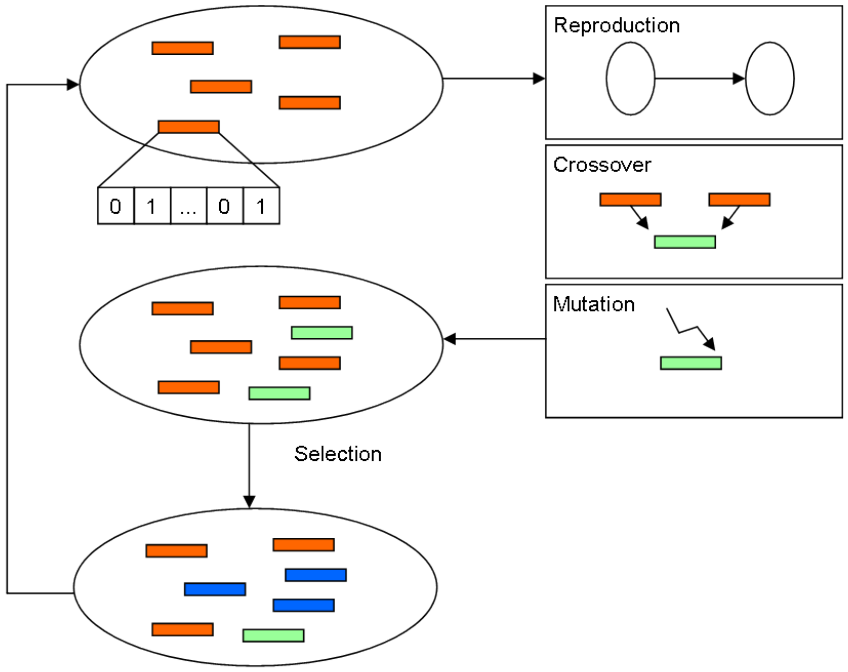


Figure 11.10: Flow chart of the genetic algorithm

shop problems. The new analytical dispatching rules offers improvements regarding different facility performance parameters, but uses static model assumptions. Solving the combination of defined dispatching rule sets is not applied in literature by genetic algorithms over time.

Besides the evolutionary approaches, solutions like tabu search or simulated annealing are also widely adopted. For our simulation problem, we choose the genetic approach. Genetic-algorithm based optimizers are simple to use for a wide range of different optimization problems including simulation. The realization of a genetic based algorithm is carried out on a wide range of different manufacturing tasks and problems. Koza [Koz92] mentioned seven basic features for using conventional optimization methods: correctness, consistency, justifiability, certainty, orderliness, parsimony, and decisiveness. He argued that a genetic algorithm embodies none of these principles. For our problem space, stochastic influences and the wide range of events do not allow us to use conventional solutions. He also argued that the right algorithm for a certain problem does not exist. Often different optimization algorithms can solve the same problem. We choose this algorithm class for our optimization problem because of the widely approved usage of genetic algorithms in the manufacturing context.

**Genetic Algorithms** Genetic algorithms are popular representatives for heuristics used in simulation environments. The basic idea is taken from nature. The algorithm takes the root concept of biological evolution. It offers a search strategy for improvement on basis of survival of the fittest solution for the problem. On each iteration, the genetic algorithm operates on a population $N$ of $n$ solutions. Thus the population $N(i)$ of the $i$ the iteration can be defined by $N(i) = \{n_1(j), ..., n_N(j)\}$. The population space can contain solutions discovered at current and previous iterations. At each iteration, the best solutions tend to survive and are combined to create better ones. The less effective solutions tend to be removed from the population. In general the genetic algorithm passes through the following steps:

1. Definition of objective function (fitness function) $F(n)$ for comparing the solution alternatives and definition of a string representation of the problem.
2. Selection of a initial population of $N$ solutions: $N(0) = \{n_1(0), ..., n_N(0)\}$ and calculation of the corresponding objective function (e.g., via simulation experiments).
3. Selection of the $n$ best elements of $N$ in a way that the solutions with a small $F(n)$ is taken is more likely (reproduction).
4. Recombination of the solutions via crossover (joining of two solutions to produce a new one) and mutation (random change of elements of a solution).
5. Determining the fitness of the new population members and taking the best $n$ elements as in step 3 for building a new population $N(i+1)$.
6. Got back to step 3 if no stopping is advised.

An illustration of these algorithm can be seen in Figure 11.10. There are several advantages known, like the robustness and the simple operations used. The most common disadvantages are assigned to the parameter encoding and the crossover and mutation operations, which should be adequate. Often a large number of runs is required to find the optimum.

**Optimization Algorithm** Our approach is fitted to the combined dispatching approach for finding the optimal weight combination $W_o$ under the given restriction $\sum w_k = 1$ and $0 \leq w_k \leq 1$ and a given set of dispatching policies $D = \{d_1, ..., d_k\}$. The ordinary encryption of the population members in the way of a binary representation is replaced by the vector

$$W = \begin{pmatrix} w_1 \\ ... \\ w_K \end{pmatrix} \tag{11.13}$$

where $w_k$ is the decimal value of the weight referenced to the dispatching rule $d_k$. A binary representation of the decimal weight values is not reasonable. The crossover operation is defined by the following equation:

$$F_{CROSS}(W_1, W_2) = W_1 \circ W_2 = \begin{pmatrix} w_{1,1} \\ ... \\ w_{1,K} \end{pmatrix} \circ \begin{pmatrix} w_{2,1} \\ ... \\ w_{2,K} \end{pmatrix} = \begin{pmatrix} (w_{1,1} - w_{2,1})z + w_{2,1} \\ ... \\ (w_{1,K} - w_{2,K})z + w_{2,K} \end{pmatrix} \tag{11.14}$$

The variable $z$ is a random number with a $U(0,1)$ distribution. There are several restrictions for z:

- $z \neq 0$
- $z \neq 1$
- $0 < z < \min\left(\frac{w_{1,k}-w_{2,k}}{w_{2,k}}\right)$

The restrictions for $z$ are a result of the precondition defined at the start, that $\sum w_{i,k} = 1$. The crossover operation must preserve the precondition, so if $\sum w_{1,k} = 1$ and $\sum w_{2,k} = 1$, the crossover result should also be $\sum w_{New,k} = 1$ . That can easily be proved by the following proof:

**Proof:**

We assume two weight combinations $W_1$ and $W_2$ with $n$ elements. We want to show that if $\sum w_{1,k} = 1$ and $\sum w_{2,k} = 1$, after the crossover operation $W_{New} = W_1 \circ W_2$ the resulting weight vector $W_{New}$ is also within the condition $\sum w_{New,k} = 1$. For this we apply the crossover operation

$$W_{New} = W_1 \circ W_2 \tag{11.15}$$

as defined in equation 11.14. Now we can calculate the sum of each weight element $W_{New}$:

$$\begin{aligned}
\sum_{k=1}^{K} w_{New,k} &= 1 && (11.16)\\
\sum_{k=1}^{K}((w_{1,k} - w_{2,k})z + w_{2,k}) &= 1 && (11.17)\\
\underbrace{\sum_{k=1}^{K} w_{1,k}}_{1} z - \underbrace{\sum_{k=1}^{K} w_{2,k}}_{1} z + \underbrace{\sum_{k=1}^{K} w_{2,k}}_{1} &= 1 && (11.18)\\
z - z + 1 &= 1 && (11.19)\\
1 &= 1 && (11.20)
\end{aligned}$$

With this equation, it is shown that after each crossover operation the resulting new weight vector $W_{New}$ does not violate the defined precondition. The mutation operation is defined by

$$F_{MUT}(W_1) = \begin{pmatrix} \widetilde{w_1} \\ ... \\ \widetilde{w_K} \end{pmatrix} \tag{11.21}$$

where $w_i = \widetilde{w_j}$ with $i \neq j$, which means a simple position change of the weights $w_i$ inside the vector. This is done by calculation of two uniform $U(0,1)$ distributed random numbers, determining the positions to be changed. In general the algorithm is performed through the following steps:

1. Calculation and evaluation of a uniform distributed grid of points on the search area for initial calculation of the start population $P_S$, which includes the calculation of
   a) The starting weight combination $p_s$ as a reference for evaluation
   b) Edge points $p_e$ where $w_k = 1$ for one element, the other elements are zero
   c) Grid points according the restriction of $\sum w_k = 1$
2. Take the $k$ best elements of the population regarding the defined objective function and remove the rest from the population $P_S$, where $k$ is the problem complexity by the number $k$ of dispatching rules available. Additionally save the deleted population elements by their weight combination for preserving a reevaluation of already evaluated combinations.
3. Perform the crossover operation $F_{CROSS}$ for each of the elements in the population $P_S$, skip weight combinations already evaluated
4. Perform the mutation operation for each newly calculated element if the random number $z_{MUT}$ is within the defined mutation probability $P_{MUT}$ (for each newly calculated element, a recalculation of $z_{MUT}$ is performed)
5. Evaluate the generated population members by simulation model run
6. Return to point 2 if
   a) The percentage of improvement of the best element $p_{best}(g-1)$ of the last iteration $g-1$ compared to the current best element $p_{best}(g)$ is lower than the defined minimal improvement $I_{IMP}$ (optional definition) for this iteration round
   b) The maximal number of iterations $g_{Max} > g$
   c) The maximal optimization time $t_{Max} > t$
7. Generate a report of the best element $p_{best}$

For a better view inside, we introduce the following example of optimization:

**Example**:

We assume an optimization problem with $k = 3$ dispatching weights to be optimized. Figure 11.11 illustrates the search space of the problem. In our case we have to deal with a triangle as search space.

Figure 11.12 illustrates the algorithm evaluation. At the first step the edge points are checked and evaluated, the second step includes a point distribution in form of a regular grid. The third step is the iterative improvement of the points by crossover and mutation. The gray points indicate worse results, the green ones indicates the actual population with the best elements, and the red points are the newly generated points, which have to be evaluated. The process of optimization is exemplified here by a reduced facility model (usage of JSL simulation library for model implementation). It includes about 200 equipments and 50 product types and a standard set of operator personal.

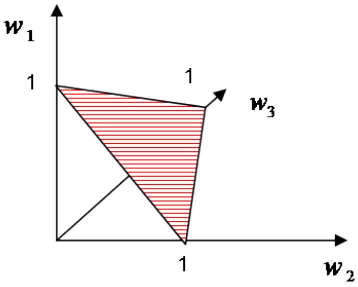


Figure 11.11: Search space for $k = 3$

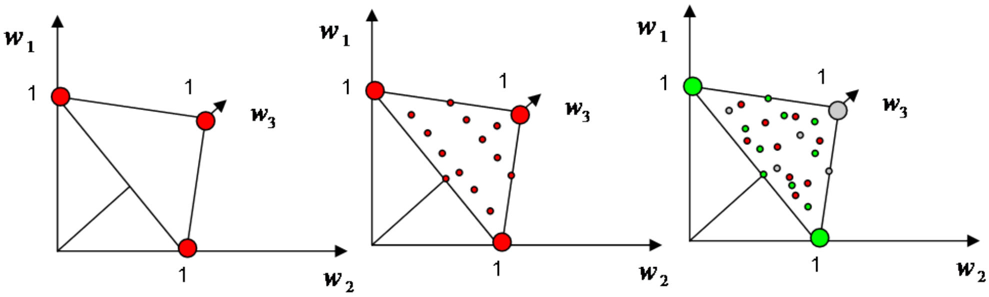


Figure 11.12: Qualitative algorithm illustration for $k = 3$

We exemplify the following case:

- The objective function is defined by $O(XF_{Mean}, OTD, WIP_{Mean}, CM_{Mean}) = 0.2 * N(XF_{Mean}) + 0.3 * N(OTD) + 0.2 * N(WIP_{Mean}) + 0.3 * N(CM_{Mean})$, which has to be minimized.

- Usage of the dispatching policies EDD, ODD, SPT, FIFO and LB.

- We use three different starting points $p_s$ of evaluation:
  - $p_{s,1}$: $w_{Fifo} = 1$
  - $p_{s,2}$: $\forall w_k : w_k = 0.2$
  - $p_{s,3}$: $w_{FIFO} = 0.1$, $w_{LB} = 0.2$, $w_{SPT} = 0.0$,$w_{EDD} = 0.5$,$w_{ODD} = 0.2$

The stopping condition is defined by a maximal iteration count of 300. The results are shown in Figure 11.13. The results illustrate a very similar behavior of the algorithm in case of different starting points. The different experiments show similar results. The general linear trend of each example case shows, as expected, a decrease of the objective function value for more iterations. The algorithm does not terminate in a local minimum and produces as results nearly the same optimal weight combinations. Therefore the starting point has no great impact onto the resulting weight combinations.

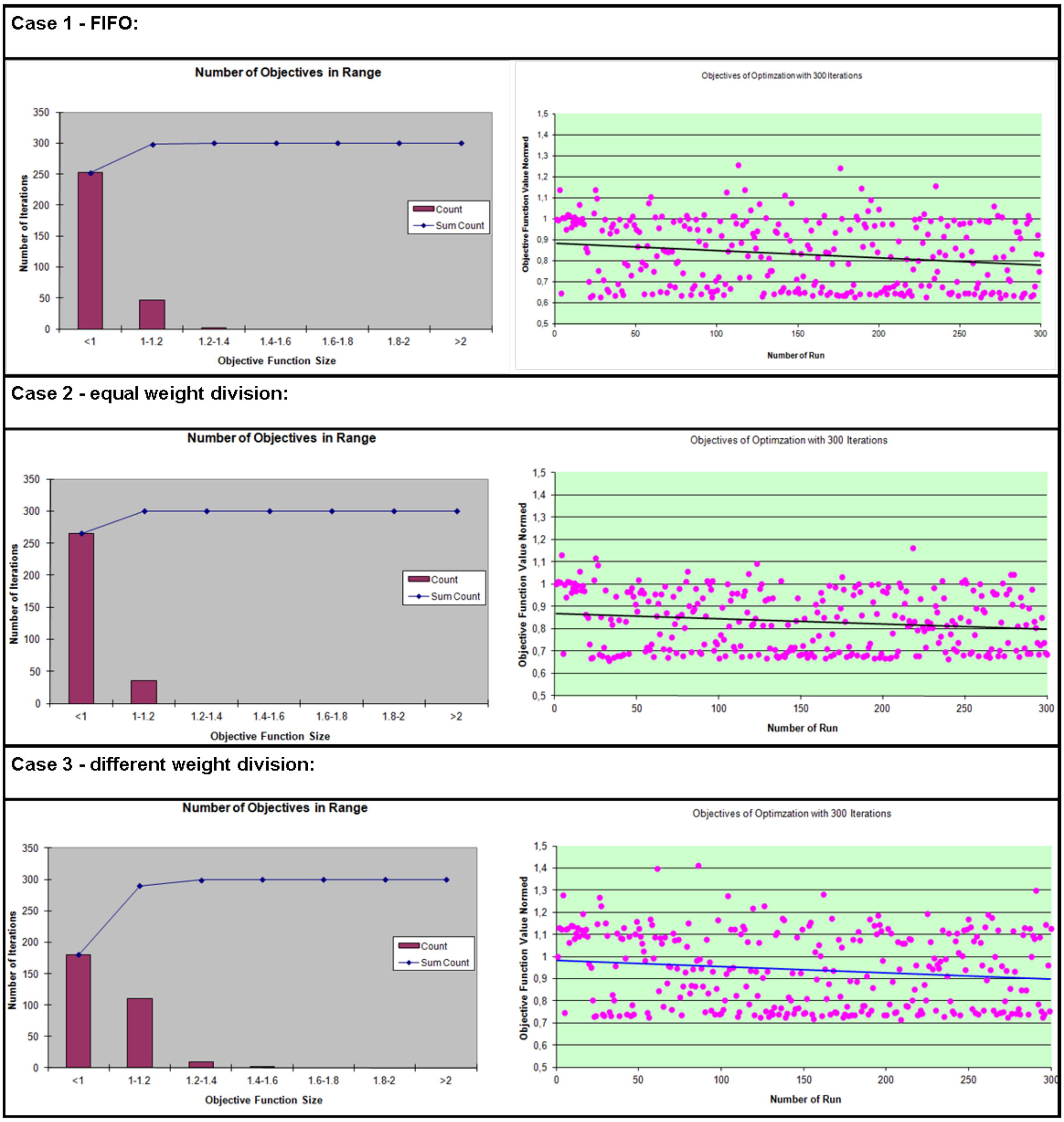


Figure 11.13: Example optimization processes with different starting points and 300 iterations

**Optimization over Time** Due to changing facility conditions, the optimization of the dispatching rule combination is repeated at fixed intervals. Our analysis contains different optimization intervals $T_{opt}$. For practical reasons, we choose a monthly optimization (OPM), a weekly optimization (OPW), a daily optimization (OPD) and a optimization twice per day (TPD). Besides the frequency of the optimization calls, the optimization preview time horizon $t_{prev}$ has to be defined. For our experiments, we take the preview time frames of one month (M), one week (W), one day (D) and half a day (HD) into consideration. With this definition, we can divide two general cases:

- $T_{opt} = t_{prev}$: In this case, after the end of the preview time frame, a new optimization is started (for example $OPD_D$).
- $T_{opt} < t_{prev}$: In this case, within the preview time frame, a new optimization is started (for example $OPD_W$).

For the analysis of the change of the weight sets, we introduce an adaption factor $a \epsilon \{0;1\}$ which defines the impact of the newly calculated weight $w_{k,i}$ of the dispatch rule $k$ at the optimization $i$:

$$w_{k,i} = w_{k,i-1} + (w_{k,i} - w_{k,i-1}) * a \tag{11.22}$$

The adaption factor is applied to avoid big fluctuations for the dispatching weights during the periodic optimizations. For the optimization operation, the restriction $\sum w_{k,i} = 1$ also holds:

$$\begin{aligned}
\sum_{k=1}^{K} w_{k,i} &= 1 && (11.23)\\
\sum_{k=1}^{K} (w_{k,i-1} + (w_{k,i} - w_{k,i-1}) * a) &= 1 && (11.24)\\
\underbrace{\sum_{k=1}^{K} w_{k,i-1}}_{1} + a\sum_{k=1}^{K} w_{k,i} - a\underbrace{\sum_{k=1}^{K} w_{k,i-1}}_{1} &= 1 && (11.25)\\
1 - a + a\sum_{k=1}^{K} w_{k,i} &= 1 && (11.26)\\
\sum_{k=1}^{K} w_{k,i} &= 1 && (11.27)
\end{aligned}$$

The experiments are done with different representative model sets of a one month time frame with standard product set and a standard operator set. The experimental set up is illustrated in Table 11.8. Preview times larger than one week are currently not possible in reality due to insufficient data and the large amount of stochastic effects in the real system. Nevertheless, for our experiments, these preview times offer a good insight into

| Frequency/Preview | M | W | D | HD |
|---|---|---|---|---|
| OPM | $OPM_M$ | | | |
| OPW | $OPW_M$ | $OPW_W$ | | |
| OPD | | $OPD_W$ | $OPD_D$ | |
| TPD | | $TPD_W$ | $TPD_D$ | $TPD_{HD}$ |

Table 11.8: Optimization experiments overview

| KPI / Experiment | Reference | $OPM_M$ | $OPW_M$ | $OPW_W$ | $OPD_W$ | $OPD_D$ | $TPD_D$ |
|---|---|---|---|---|---|---|---|
| WIP average | 1.00 | 0.96 | 1.01 | 1.00 | 0.98 | **0.98** | 0.99 |
| CM 50% | 1.00 | 0.93 | 1.01 | 1.00 | 0.98 | **0.97** | 0.97 |
| CM 90% | 1.00 | 0.93 | 1.02 | 0.99 | 0.96 | **0.94** | 0.95 |
| XF 50% | 1.00 | 0.94 | 1.02 | 1.00 | 0.98 | **0.97** | 0.98 |
| XF 90% | 1.00 | 0.88 | 1.03 | 0.99 | 0.97 | **0.95** | 0.96 |
| absolute OTD | 88.9% | 91.0% | 87.8% | 89.6% | 89.9% | **90.4%** | 89.8% |
| Lots finished | 1.00 | 1.16 | 0.97 | 1.01 | 1.01 | **1.06** | 1.05 |

Table 11.9: Scenario O - KPI results for $a = 1$

the mechanism of the weight change over time. For presentation of the results, we show some example experimental results.

**Optimization without Adaption ($a = 1$):** Table 11.9 illustrates the results for the 98% load case and $a = 1$. For this experiment, we use a one month historical facility model at a 98% load with the corresponding equipment downs and operator attendance. The reference is described by the FIFO policy. The objective function is defined as

$$O(XF_{90}, OTD, WIP_{Mean}, CM_{90}) = 0.1 * XF_{90} + 0.5 * OTD + 0.1 * WIP_{Mean} + 0.3 * CM_{90} \tag{11.28}$$

The best result is generated by the monthly optimization with a preview period of one month. Improvements of about 4% to 6% can be achieved. The throughput can be increased by 16% in regards to the reference dispatch rule. The monthly preview time frame with a weekly optimization performs worse than the reference. One reason is the overlapping time frame, in which the optimization targets of the further optimization are replaced by the newer one. Therefore a degradation of the performance can be observed in this case. An increase of the optimization frequency tends to have a better result down to a daily optimization. The daily optimization with a preview time frame of one week works less efficient than the preview time frame of one day. Nevertheless both results outperform the reference. The effect is virtually the same as with the overlapping time frames of the month preview time frame. An improvement by increasing the optimization frequency more than one per day is not observable in our case.

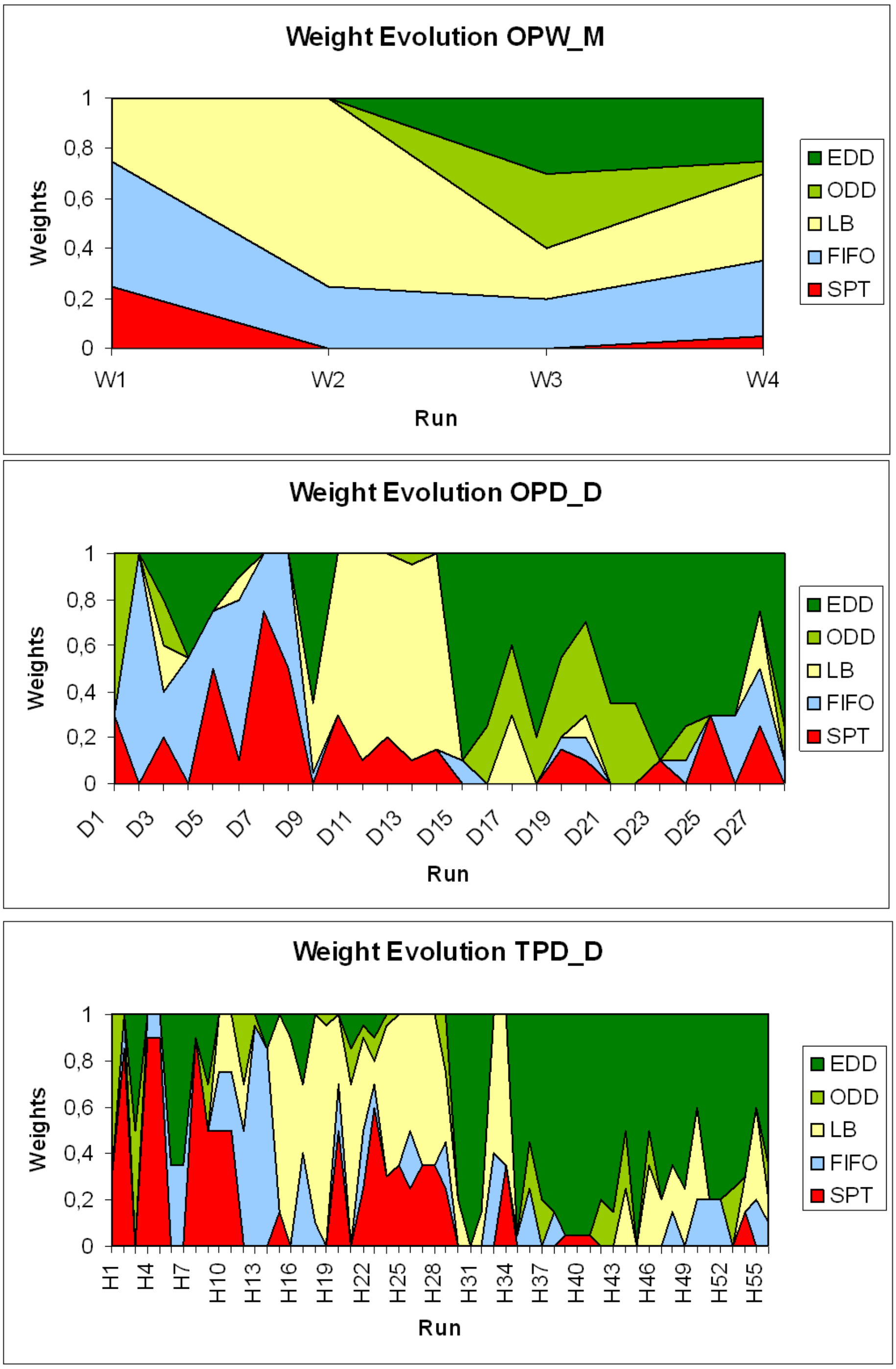


Figure 11.14: Scenario O - Weight Division of $OPW_M$, $OPD_D$ and $TPD_D$ for $a = 1$

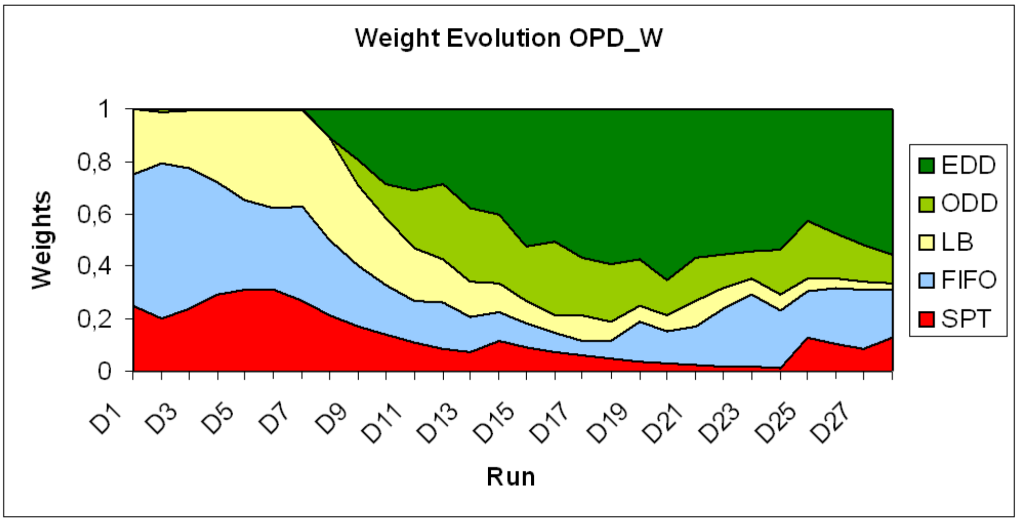


Figure 11.15: Scenario O - Weight Division of $OPD_W$ for $a = 0.2$

Figure 11.14 illustrates the weight evolution over time for the cases $OPW_M$, $OPD_D$ and $TPD_D$. During the first two weeks, the dispatching rules SPT, LB and FIFO have the greatest impact on the resulting priorities. The reasons can be found in larger equipment downs in this time and an initialization of the dispatching policy itself. After the two weeks, the due date oriented rules have the most influence. The line is stabilized to a balanced state, where the due dates are appointed with the largest priority by optimization.

The weekly optimization, of course, has the most inaccurate weight split over time. Even the amount of the due date oriented rules at the last two weeks degrades the performance. The daily optimization and the optimization twice per day have a similar weight split over time. Smaller differences can be observed at the start of the time period, where the SPT rule is more prioritized by the $TPD_D$ solution than in the $OPD_D$ solution.

**Optimization with Adaption ($a\epsilon\{0;1\}$):** Our experiments indicate improvements in using the adaption factor. Even in cases of overlapping preview time frames, where $T_{opt} < t_{prev}$, improvements are obvious at our experiments. The reason for this behavior is the overlapping optimization time frame, where $t_{prev,i-1}$ of optimization $i - 1$ within $t_{prev,i}$ of optimization $i$. Therefore the preview optimization is also valid for a part of the next optimization. Figure 11.15 illustrates the weight evolution for $a = 0.2$. Figure 11.16 illustrates the KPI results for $OPD_W$ with different $a$.

The influence of the adaption factor $a$ for the throughput at case $OPD_W$ are obvious for $0.2 \leq a \leq 0.4$. Here the improvement to $a = 1$ is about 5%. In this area, $OPD_W$ outperforms also $OPD_D$ in the $CM50$ and $CM90$ KPI. For $OPD_D$, the influence of the adaption factor $a$ is very low as it was expected. The higher the $a$, the better the KPI results turn out for $OPD_D$.

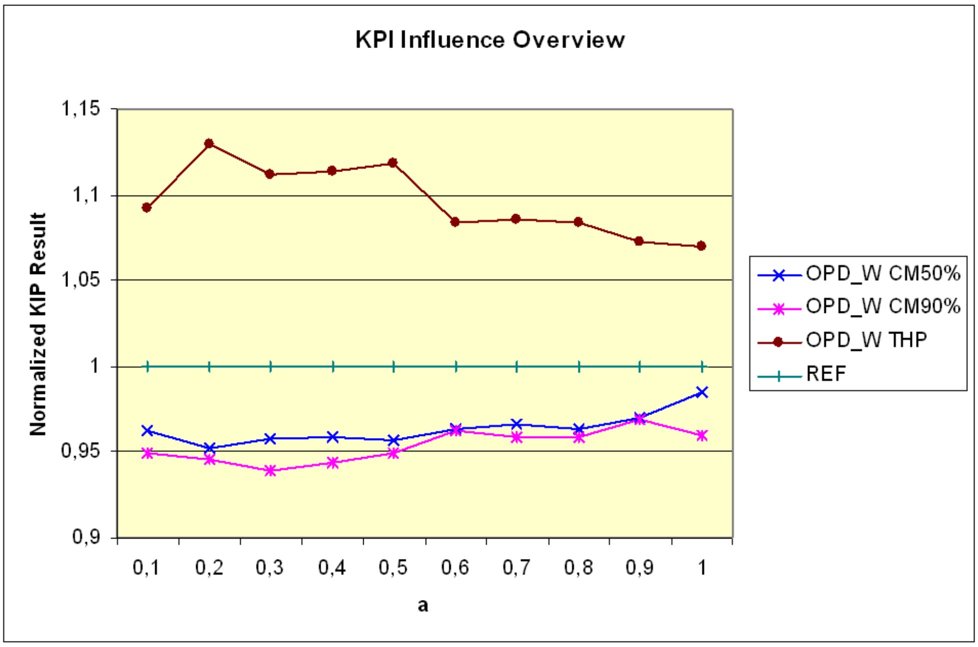


Figure 11.16: Scenario O - KPI results for $0.1 \leq a \leq 0.9$

**Conclusions:** In general, our experiments tend to prefer practical optimization frequencies on daily basis with time frames between one day and one week. The differences between these results are very low within one to two percent in case of $a = 1$. With usage of the adaption factor, improvements for the case $T_{opt} < t_{prev}$ are realizable. Our experiments show the best results for our case for $0.2 \leq a \leq 0.4$. $OPM_M$ is not reasonable at our case, but offers the best performance results at the most experiments.

# 12 Assessment of Metrology Operations

In this chapter, we introduce a freely configurable dynamic sampling application approach for usage in a foundry with high-mix low-volume facility profile.

## 12.1 Dynamic Lot Sampling at Metrology Operations

### 12.1.1 Motivation

In semiconductor manufacturing, metrology tools could be very expensive. Capacity restrictions are the consequence. The full measurement of all lots is often not applicable due to the limited capacity. In mass production environments, sampling rates are defined by a certain sampling frequency, e.g. $f_s = 5$, which defines a measurement operation every five lots (e.g., see [Stu09] ). Figure 12.1 illustrates an example. At a typical foundry, where product volumes can vary from one lot per month to several hundred lots per month, the definition of a sampling rate for the lots is not sufficient.

Cost and quality are the main parameters for a sampling algorithm, which are taken into account. Figure 12.2 illustrates the gap between measurement costs and quality. In our case the quality considerations have a greater importance than the metrology tool utilization. In general we propose two different sampling algorithm classes regarding quality:

- **Quality saving**: The reduction of metrology effort should not result in a reduction of the end line quality. The save prediction and detection of failures are the main tasks.

- **Non-quality saving**: A decrease of the quality by reduction of the metrology effort is tolerated and compensated by the higher throughput at these operations.

In our case the reduction of the quality is not accepted. Our focus is to provide a practical framework for a SF in order to select the right lots for metrology operations from the point of assuring quality. In literature there is a wide range of different policies and solutions

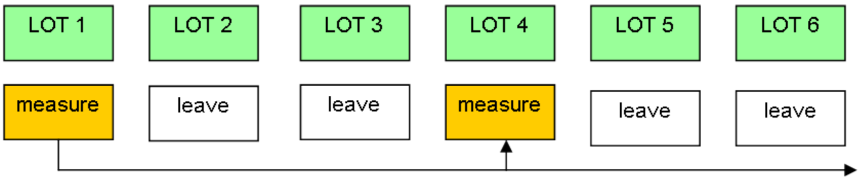


Figure 12.1: Simple sampling frequency measurement

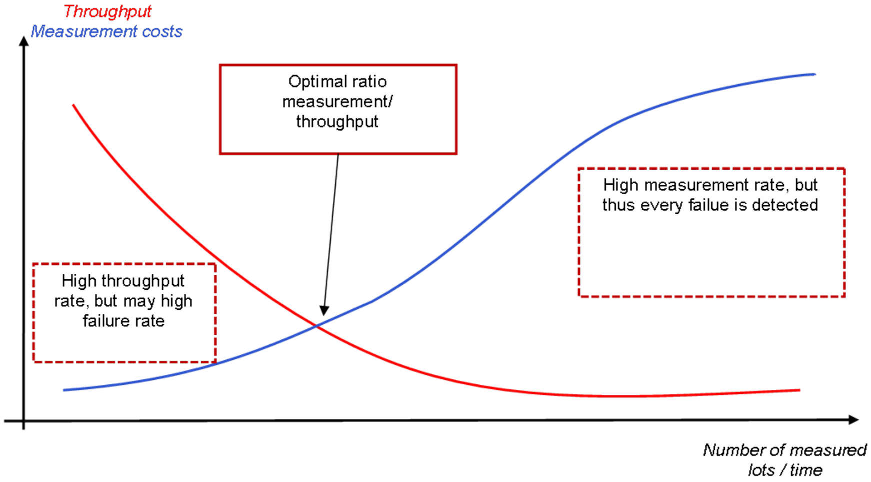


Figure 12.2: Qualitative view of quality vs. measurement costs

available, especially for the mass production case. Nduhura-Munga et al. [NMRVDP+13] defines three different sampling types:

- Static sampling
- Adaptive sampling
- Dynamic sampling

Whereas static sampling rules mostly are reasonable in use at mass production cases, adaptive as well as dynamic sampling has a high potential also in the area of high-mix production facilities. A wide range of different global and local sampling policies are available, mostly examined with academic data. Only a few solutions are realized at real facility environments. Unfortunately, a comparison is often not available due to the different basic conditions. The potential in the area of high-mix low-volume production is not obvious and depends mainly on the product mix. In the next sections, we want to introduce a flexible global framework allowing a dynamic usage of different policies. Furthermore we investigate the influence of sampling at a typical high-mix low-volume production area.

### 12.1.2 Policy Definition

At our approach the sampling decision $S_d(E, P_L)$ for a certain step or equipment $E$ and a lot or wafer $L$ with the properties[1] $P_L$ consists of several sub decisions $s_i$ done by sampling modules. The sampling modules are process specific and represent the demands of the

[1] The properties $P_L$ of a lot $L$ contain all process relevant parameters like lot number, product number and technology.

line engineering. A free configurable set of these modules per equipment or stage can be defined. For example the modules can represent

- the current process capability $CPK$ for the current process,
- the minimal sampling frequency $f_s$ at which a lot should be measured,
- dependencies regarding passed equipments or processes in the past, or
- methods for analyzing the demand for an additional measurement.

In general the decision can be calculated as

$$S_d(E, P_L) = \begin{cases} s_1 \circ s_2 \circ ... \circ s_n & P_L \in P_S \\ \text{default action} & P_L \notin P_S \end{cases} \tag{12.1}$$

where $s_i \in \{m_s, m_m, m_a\}$ ($m_s$ indicates a skip of the operation, $m_m$ a standard measurement and $m_a$ a additional measurement) . The operator $\circ$ can be defined as follows:

$$\bigcirc = \begin{cases} m_m & \exists s_i : s_i = m_m \\ m_a & \exists s_i : s_i = m_a \wedge \forall s_i : s_i \neq m_a \\ m_s & \forall s_i : s_i = m_s \end{cases} \tag{12.2}$$

In the following section, we introduce the influences of lot sampling to our model.

### 12.1.3 Simulation Analysis

#### 12.1.3.1 Scenario 1

The first scenario includes a performance evaluation of lot sampling. We define common sampling rates per lot of 5%, 10% and 25% at a single metrology operation, which is a known bottleneck. The second experiment includes these sampling rates at each metrology at the lithography area operation of the facility. Sampling rates above 25% are not applicable at this model, due to the high mix of products. The first experiment illustrates the influence on a single introduction of the system into the facility, the second one illustrates the introduction at all metrology steps in the lithography area. In the foundry business, not every product is a possible candidate for sampling. In our case several products are not allowed to sample, like automotive technologies. For examination of the potential of lot sampling in our case, we do not take this fact into account. The decision to skip is determined by a random number, which is $U(0, 1)$ distributed. The model experiments are done at 98% facility load with the FIFO dispatching rule and a strict setup avoidance policy. Figure 12.3 illustrates the XF of the different cases at 98% load.

The single operation skip experiment illustrates a very low influence on the XF. The influence to skip one defined metrology operation through the overall wafer processing is very low. Lots passing this operation are forced to wait at the following steps. Thus the introduction of the system at a single operation has no effect on the KPI. In the case of an

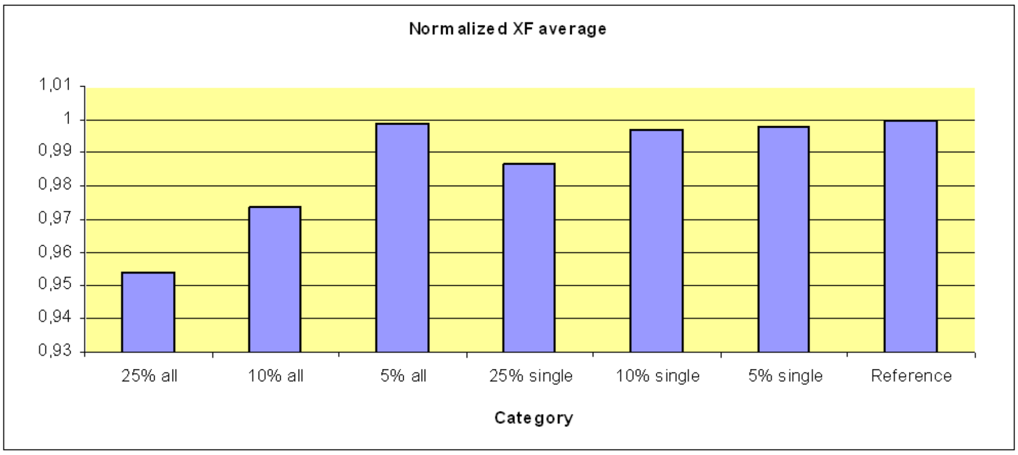


Figure 12.3: Cycle time per mask lot sampling simulation result

area-wide introduction, a 10% skipping rate reduces the XF average by about 2% to 3%. In case of 25% skip rate, the reduction is about 5%. For a global effect, the sampling has to be introduced in an area-wide way. The same behavior can be detected at the other KPI, which are also reduced to the same level. The on-time delivery is not affected by this policy, in almost 25% of the case an improvement of about 1% is detectable.

#### 12.1.3.2 Scenario 2

This scenario deals with a hypothetical tool down in the CD measurement area of one tool over the whole modeling period. In our case, metrology tool downs are the main application area of lot sampling. The sampling is used to preserve the whole facility from KPI degradation in order to choose the right lots for measurement systematically. The set of experiments presented here is shown in Table 12.1.

The simulations are performed with historical facility data and a standard product. The CD fail example illustrates the model behavior without lot sampling and any corrective actions. The CD fail with DLS+Combined includes the lot sampling mechanism at the CD measurement steps and the usage of the combined dispatching approach. As in the last case, the combined approach with lot sampling and without tool down is shown (see Section 11.1.3). The results are presented in Table 12.2.

The performance degradation of the fail experiment without any corrective action is very high. The performance loss is about 20% to 40% with regard to the average KPI. Therefore corrective actions are necessary. The performance degradation with the combined dispatching and a sampling rate of 5% is about 3% to 5%. The gap between these both experiments is about 20% to 30%.

The usage of lot sampling in combination with the combined dispatching without a failure at the measurement tool shows improvements of about 8% to 15% at various KPI to the reference case. The evolution of the CM average and deviation is illustrated in Figure

| Number | Experiment | Comment | Applied Rules | Weights |
|---|---|---|---|---|
| 1 | FIFO Reference | | FIFO | 1.0 |
| 2 | Combined Reference | With lot sampling (5%) | FIFO | 0.2 |
| | | | ODD | 0.2 |
| | | | EDD | 0.2 |
| | | | $LB_{Load}$ | 0.2 |
| | | | SPT | 0.2 |
| 3 | CD Fail | Fail of one CD Tool | FIFO | 1.0 |
| 4 | CD Fail with DLS+Combined | Like (3) with lot sampling (5%) and (2) | FIFO | 0.2 |
| | | | ODD | 0.2 |
| | | | EDD | 0.2 |
| | | | $LB_{Load}$ | 0.2 |
| | | | SPT | 0.2 |

Table 12.1: Scenario 2 - Experimental set

| KPI / Experiment | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| WIP average | 1.00 | 0.92 | 1.24 | 1.03 |
| CM average | 1.00 | 0.88 | 1.39 | 1.04 |
| CM deviation | 1.00 | 0.85 | 1.79 | 1.36 |
| XF average | 1.00 | 0.87 | 1.40 | 1.03 |
| XF deviation | 1.00 | 1.03 | 1.80 | 1.11 |

Table 12.2: Scenario 2 - Simulation result for 98% load

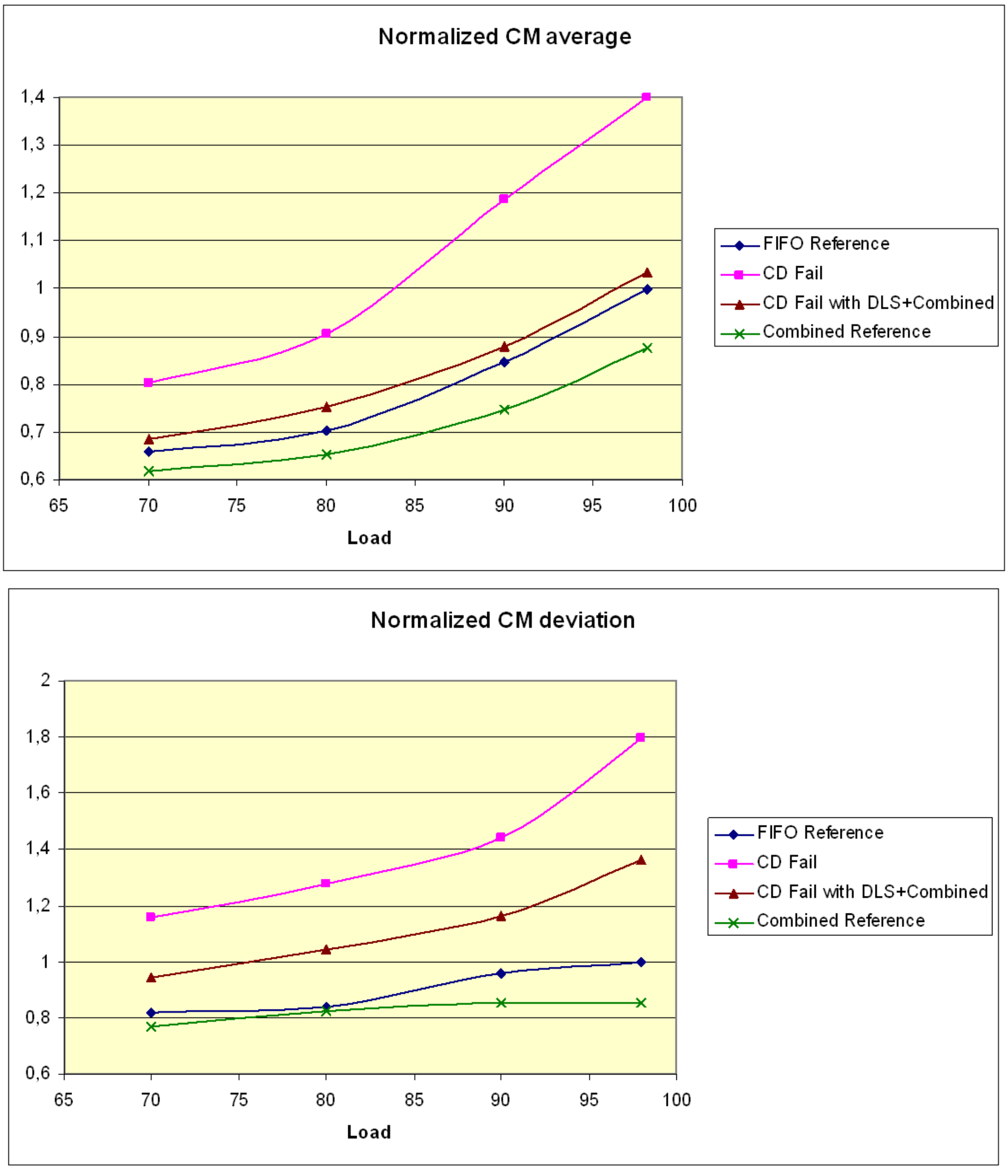


Figure 12.4: The CM at CD measurement tool down

12.4. The gap between experiment (3) and the reference increases with increasing load. A higher load causes a higher variability with longer waiting times at the measurement bottleneck. The gap between experiment (4) and the reference is nearly equal at all load cases. The reasons are found in the load balancing mechanism and the sampling which improves the bottleneck utilization. The deviations of the experiments illustrate a similar behavior. The slope of the curve shows an increasing behavior at higher loads at the fail cases. This results from the high variability during the waiting times at the measurement bottleneck tool.

## 12.2 Conclusions

For performance optimization aspects, the lot sampling mechanism has to be applied at many stages on the factory floor to achieve measurable global performance enhancement. The more interesting use case is the bottleneck capacity increase in case of tool downs. In this case, lot sampling is generating a great benefit by offering an automated reduction of the measurement effort of these tools, including the reduction of waiting times. In combination with an optimized dispatching strategy, the tool down effect can be reduced to a 5% increase of the overall factory performance measurement at our example.

# Part V

# Controller Realization

Imagination is more important than knowledge.

*(Albert Einstein (1879-1955))*

# 13 Controller Design and Implementation

## 13.1 Introduction

In this chapter, we introduce a very flexible and adaptable concept for a software design of the production control approach. Older foundries with a historically grown IT infrastructure have to put up with inhomogeneous systems and solutions. These software solutions offer a high variation of different interfaces and access options for data manipulation. We apply the well known standard of web services, which offers a high flexibility. Web services are briefly introduced in the next section. The whole prototype implementation is based on the JAVA programming language. The selection of a JAVA programming language has several reasons:

- Well known best practices
- Comprehensive documentations
- Comprehensive support of web service techniques[1]
- Native techniques for transactional safety in database environment

We divide the project into two parts:

- The dispatch controller part
- The lot sampling controller part

For these two parts, independent controller structures and user interfaces are designed and implemented. The user interfaces are based on common WEB 2.0[2] techniques without the need of local client installations.

## 13.2 Global System View

For our design, there is a need of an independent application structure (see Figure 13.1). The structure is divided into the data interface and the application itself. The application

[1] Web services are a grown element of interoperable communication and interaction between systems over the network. Many applications localized on a server system can also be published, used and changed via web service techniques.

[2] The term of WEB 2.0 is associated with a wide range of new collaboration and interoperability options available at common web standards newer generations. WEB 2.0 pages allow an active user interaction and collaboration with different elements of the web site.

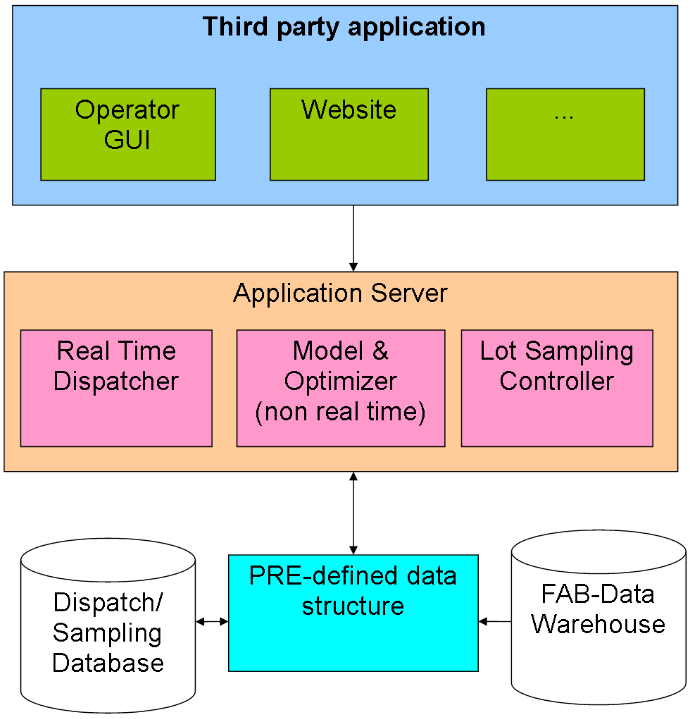


Figure 13.1: Big picture of the controller system

consists of three logical elements. The **real time dispatcher** is responsible for the calculation of dispatching priorities and the generation of equipment dependent lot lists. The real time characteristic requires a high performance for dispatching. The **model and optimizing** element realizes the approach introduced in Section 11.1. For this, an automated model generation part is implemented. It generates a model representation of the current facility state. The optimization task determines the optimal weight combination for each dispatch policy for the given objective function. The **lot sampling** controller element is responsible for determining the sampling decisions at defined steps in the process work flow. The decisions are done in a real-time environment, so time consumption must be very low.

The data for dispatching and lot sampling is collected by an interface to the data management system of the facility. A data structure is defined, to which the facility data warehouse is adapted and connected. In addition the sampling and dispatching system defines database elements for a local storage.

Each of the controller elements has defined interface methods over which third party applications can access the functionality. The application is allotted to an application server system, which is independent from other systems in the facility environment.

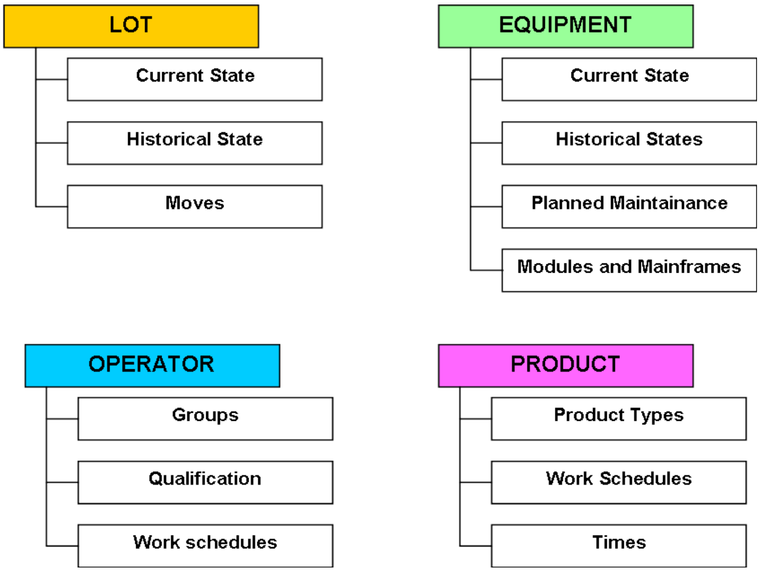


Figure 13.2: Data domains for dispatching and modeling

## 13.3 Data Access

Figure 13.2 illustrates the main data domains used in the dispatching and modeling system. We define four domains including the main data structures and definitions for the system:

- **Lot**: The lot data domain includes state data of the lot at the current point of time and historical points of time. The historical lot data covers historical lot moves with the corresponding dates (process start, process finish, steps, etc.). Besides the historical information, the current state of the lot is represented. The state includes current wafer counts, the current step, and information about the lot stop behavior.

- **Equipment**: The equipment data domain includes all equipment dependent data. This domain contains historical downs and maintenance activities and planned activities for the future. Each available workstation is specified in a mainframe overview with their modules, if available (in case of cluster or multi-chamber tools).

- **Operator**: The operator data domain covers all data of the operator staff. Each operator has different qualification levels. The levels define the ability to interact with different pieces of equipment during the process. In addition, each operator has a work schedule with holiday and break information.

- **Product**: The most important data domain is the product data domain. The product data domain includes all available products with their work schedules. Each process step of the schedule defines several corresponding times like processing time, process dependent waiting time, operator time and, if available, the setup time with the required setup state. The times are equipment dependent.

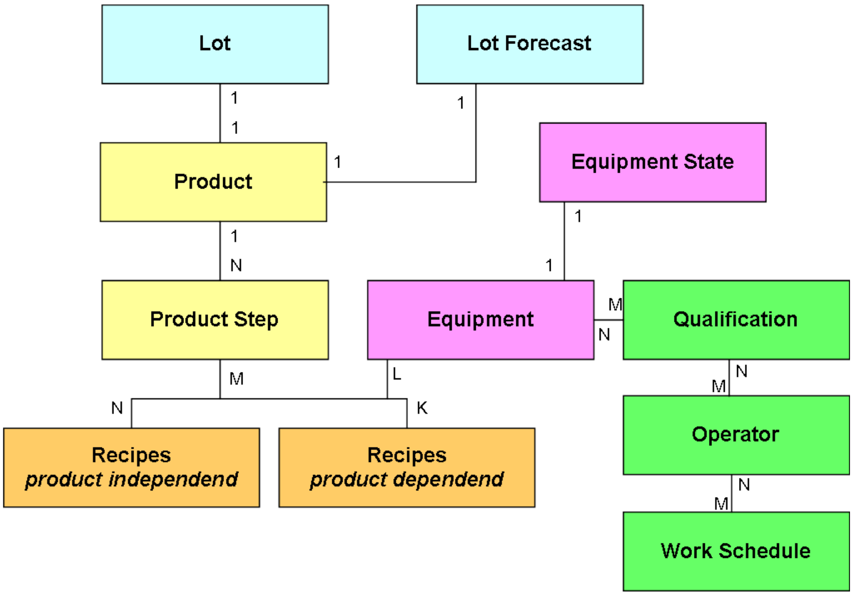


Figure 13.3: Database structure

Each of the domains is represented in the data access interface structure with a high-performance access. This can be realized by efficient database tables and views or by data interfaces. In our case, we define standardized data interfaces as illustrated in Figure 13.3. The domains mentioned are represented in the tables and views. To avoid performance issues in accessing different database and data interface systems, a central database scheme is established. This scheme provides all necessary data domain information.

The tables and views provided by this scheme can be divided into different refresh time classes. We use the possibility of materialized database views (where it is possible), which allow high performance. Materialized database views create a snapshot of the whole content and store it in a temporary table space with defined refresh times. Data, where the change frequency is not high, like the product data domain (excluding recipes) are examples for the application of the method. Real time data like the current lot states or the current recipe settings are not represented with materialized views. The current and actual data is required in this case.

## 13.4 Real Time Dispatch and Optimization

In this section, we concentrate on the design and implementation of the real time dispatch and optimization.

### 13.4.1 Use Cases

At the start of the design process, Use-Case-Diagrams according the UML notation are useful to define the general application aspects. Figure 13.4 illustrates the top view use

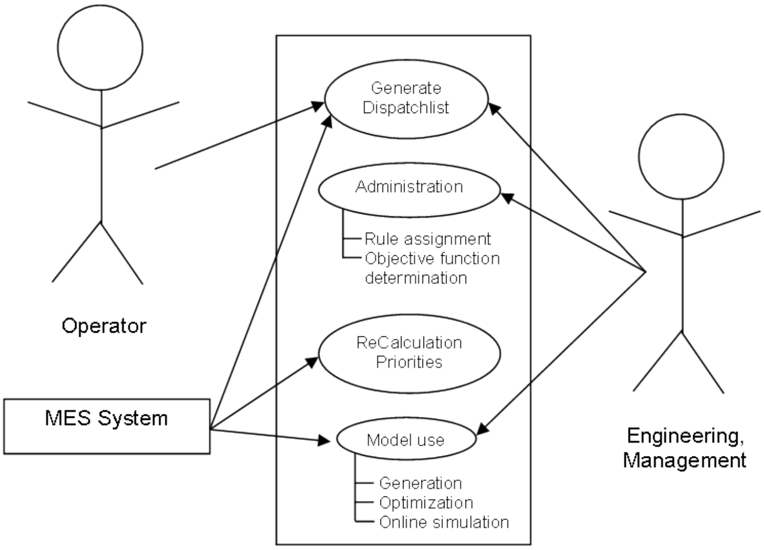


Figure 13.4: Use-Case-Diagram of the dispatching system

case diagram on the system. We can define three entities interacting with the system:

- **Operator** staff, who is interested in a reasonable dispatching list (either in the operator GUI or on a web page).
- **Engineering and management** staff, who has more privileges than the operator staff, like the administration of the whole system and the model usage.
- **MES**, which automatically uses different methods of the system like rule priority calculation, simulation model optimization and dispatching list generation.

In the next sections, we introduce the most important use cases in detail.

#### 13.4.1.1 Dispatch List Generation

The dispatch list generation is one of the most time-critical operation. There are only a few seconds of time until the dispatching list has to be presented. The main flow is illustrated in Figure 13.5.

The first step includes the generation of the available lot list for the equipment. In this case the lot list is not provided by call, it must be generated by checking the recipes of each of the available lots in production. The second step calculates the currently optimal setup for the equipment, if required. This is done by a implemented setup rule for each equipment or equipment group. The next step is to recalculate the lot priorities, if necessary. Due to performance reasons, lot priority calculations should be done at the lot move point to the next stage in an asynchronous way, rather than at dispatching time. Therefore each rule can define a calculation point. This is applicable for rules without the need for calculation at dispatching time like the EDD rule or the ODD rule.

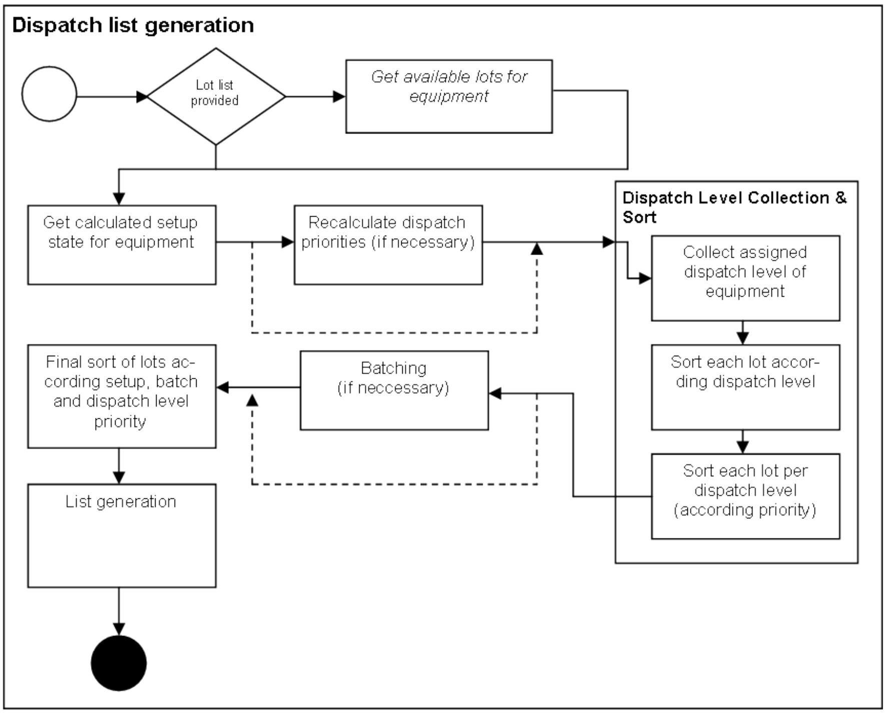

Dispatch list generation
Lot list provided
Get available lots for equipment
Get calculated setup state for equipment
Recalculate dispatch priorities (if necessary)
Dispatch Level Collection & Sort
Collect assigned dispatch level of equipment
Sort each lot accor-ding dispatch level
Sort each lot per dispatch level (according priority)
Final sort of lots ac-cording setup, batch and dispatch level priority
Batching (if neccessary)
List generation

Figure 13.5: Flow chart of the dispatching list generation use case

At each equipment, there is either a global or local dispatching level hierarchy defined. Dispatching levels are used in order to represent manually assigned priorities by the MES. Hot lot or bullet lot states are provided by system, which have a higher priority than the standard dispatching priority. At each level, the origin of the priority per lot can be identified, which can be **system provided**, a **single** dispatching rule, or a **combined** dispatching rule.

Several levels can be defined globally or locally in descending order. The levels are arranged according to a defined hierarchy $O(D) = (D_1, D_2, ..., D_n)$. In this case $D_1$ has the highest priority, then $D_2$ and so on. If there is the same priority at one level, the next lower level defines the lot order. The following example illustrates the lot list generation. We assume three lots $L_1$, $L_2$ and $L_3$ are given. Each of the lots has four defined dispatch levels (see Table 13.1):

- $D_1$ represents the bullet lot state with the values $D_1 \in \{0; 1\}$.
- $D_2$ represents the hot lot status with the values $D_2 \in \{0; 1\}$.
- $D_3$ indicates the batch accelerating rule the with the values $D_3 \in \{0; 1\}$.
- $D_4$ represents the combined dispatch rule with the values $0 \leq D_4 \leq 1$.

After the calculation of the order of the lots, the lots are sorted according to the batches, if required. The lots with the highest priorities are collected in the batch at the top of

| Lot | Dispatch level | Order | Comment |
|---|---|---|---|
| $L_1$ | $\{0;0;1;0.45\}$ | 2 | Batch accelerated lot |
| $L_2$ | $\{1;0;0;0.23\}$ | 1 | Bullet Lot |
| $L_3$ | $\{0;0;0;0.42\}$ | 3 | Normal lot |

Table 13.1: Dispatch level ordering example

the dispatch list. During this step, the setup state is taken into account. Lots with the same setup as determined by the setup rule are placed at the top of the list. The final list containing the setup and batch information is sent back to the request.

#### 13.4.1.2 Priority (Re)Calculation

The calculation of the lot priorities is done at lot moves in an asynchronous way or at dispatching time. The point of time is defined by the dispatching rule. From the performance point of view, each rule is forced to calculate their rule priority during the move of the lot to the next stage in an asynchronous way. For priority calculation, the set of the dispatching level $O(D_i)$ is read from system database according to the equipment group of the next stage (during move calculation) or from the current state (at dispatch time). For each dispatch rule, the non-normalized lot priority $P_{i,U}(L_n)$ of the lot $L_n$ is calculated and stored. After recalculation, a normalization of the rule priorities is calculated, so that $0 \leq P_{i,U}(L_n) \leq 1$ .

#### 13.4.1.3 Dispatch Weight Optimization

The model optimization of the corresponding dispatch weights is realized by an automated facility model with the defined optimization strategy for the dispatch weights. A data collection module collects all required data for this process. After data collection, a data consistency check and a model validation procedure is carried out to avoid problems at model calculation. After that the model optimization is performed. This task contains a model initialization run, model validation and verification. For more details, we refer to the implementation at Section 13.4.3.2.

### 13.4.2 Dispatch Controller Class Structure Diagram

The class structure diagram (see Figure 13.6) of the project gives an impression of the number of packages and objects defined for implementation. The package structure consists of five main elements:

- **Fab model system:** The fab model system offers exhaustive functionality for running simulation models. We apply the JSL simulation library to implement our requirements for the controller system. The fab model data package includes the necessary data for the simulation. The fab model entity package includes all simulation entities realized with JSL (see Section 13.4.3). After a simulation run, the simulation statistics are collected for each entity available. Each statistical object has the ability to collect and calculate different statistical parameters.

- **Data interaction:** The data interaction package is responsible for collecting all necessary data for the model and dispatching functionality from various databases and interfaces. To do so, the system has access to the defined data interface (see data structure at Figure 13.3).
- **Modeling:** The modeling package uses the fab model system and provides various functionality for simulation, optimization and verification. The simulation element includes simulation runs of different input data sets supporting direct database and file access for the simulation data.
- **Dispatching:** The dispatching element collects all available dispatching and setup rules. For this purpose a standard class interface is available. This interface defines the structure of a dispatching and setup rule definition for simple extension by new rules.
- **Core and Webapp:** The core interface provides complete functionality of the dispatching system including the optimization and simulation options. All methods called by the web service are part of the core interface.

We present some short extracts of important implementation aspects in the next sections.

### 13.4.3 Simulation Aspects

#### 13.4.3.1 Simulation Model Structure

The model entities are implemented as different class objects. Figure 13.7 illustrates a brief extract of the implemented classes. For each entity, a set of events is defined. The events describe the behavior of the model elements. For each entity, the events are implemented as inner classes. The operator entity, representing the operator behavior has three main states (according Section 8.1.2.2). The states are the idle state, the busy state and the inactive state (in case of break or no work schedule). A state change is triggered by the events pause start and pause end as well as the work schedule start and end. In the idle state, an operator can address a new request from a certain equipment or transport task.

The equipment station, one of the most important elements in this model, has several events to be triggered. Each equipment station can have the states defined in Section 8.1.1.1. These states are the down state, the busy state, the idle state, the waiting-for-operator state and the setup state. Each of these states is triggered by various events. The down start event listener indicates an equipment down with an immediate change to the down state. When the down state is over, the down end listener changes either to idle or process, depending on the previous state. When a process is finished or a new lot arrives and the equipment state is idle, an operator is called (if necessary) and then the process is to be started when the operator arrives. After finishing the process, the lot is released to the additional waiting time (e.g., for cooling). After that, it can be moved to the next step.

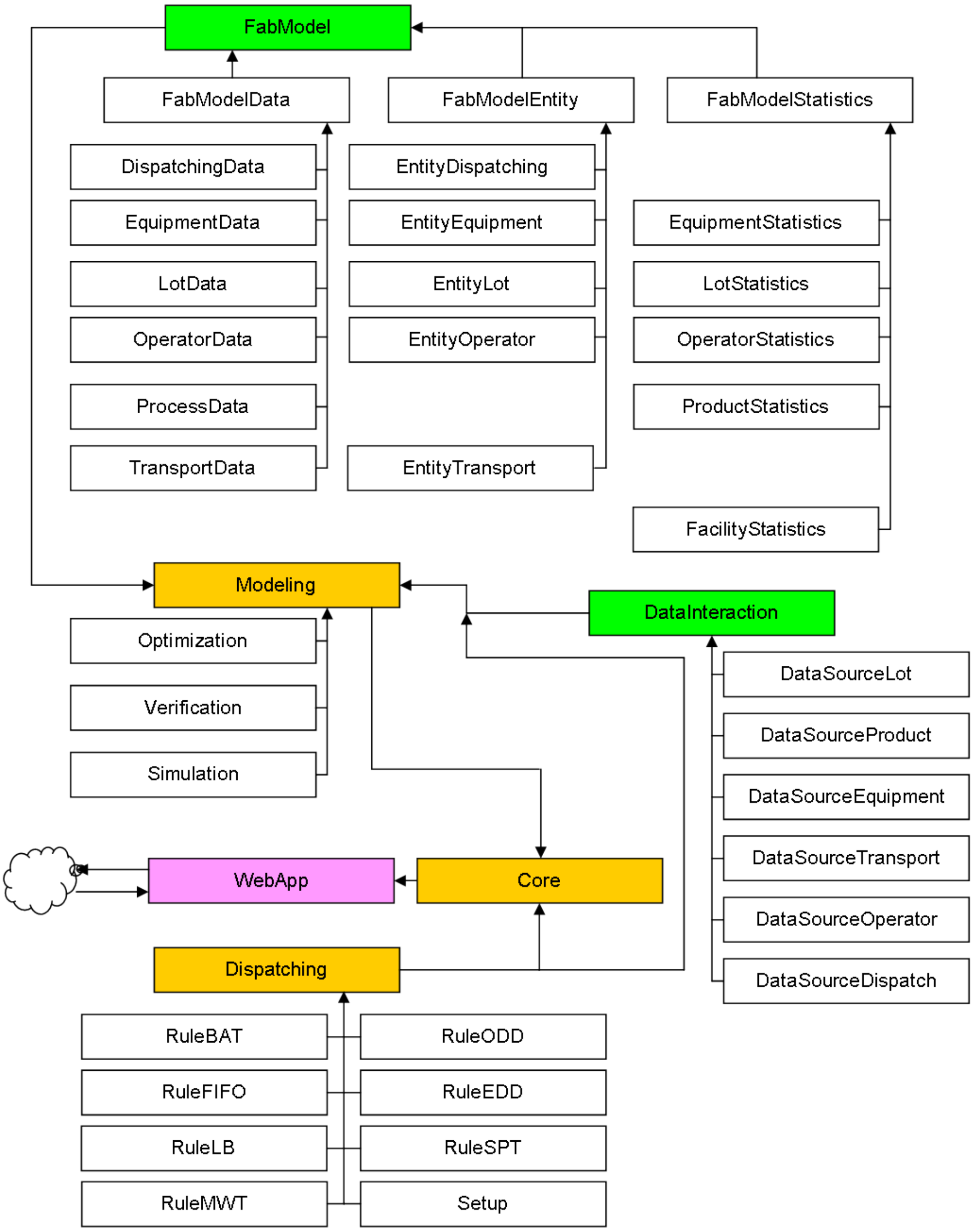


Figure 13.6: Implementation class structure diagram

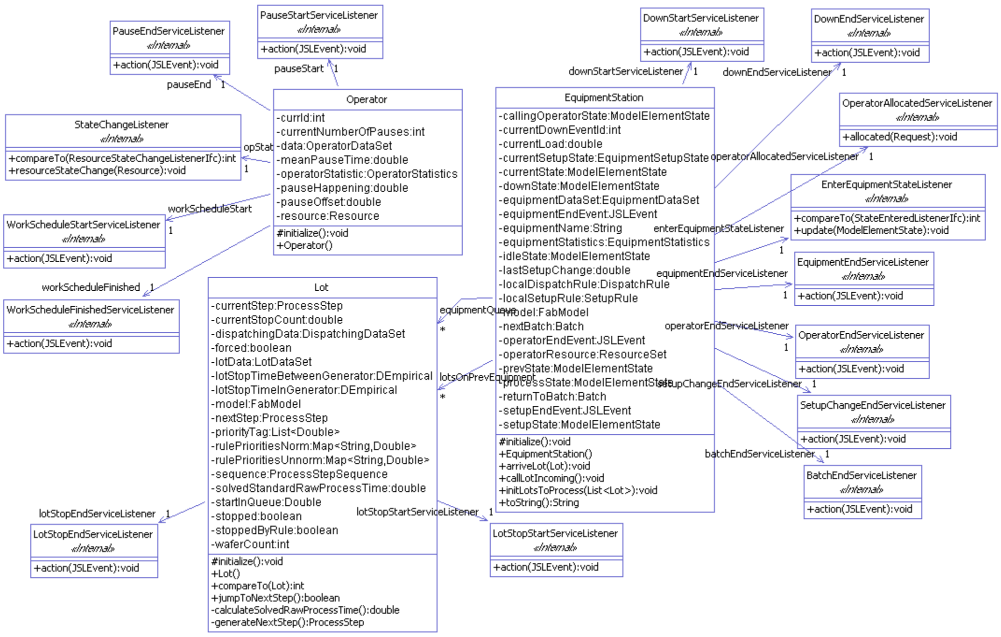

PauseEndServiceListener
«Internal»
+action(JSLEvent):void
PauseStartServiceListener
«Internal»
+action(JSLEvent):void
pauseStart 1
pauseEnd 1
Operator
-currId:int
-currentNumberOfPauses:int
-data:OperatorDataSet
-meanPauseTime:double
-operatorStatistic:OperatorStatistics
-pauseHappening:double
-pauseOffset:double
-resource:Resource
#initialize():void
+Operator()
StateChangeListener
«Internal»
+compareTo(ResourceStateChangeListenerIfc):int
+resourceStateChange(Resource):void
opStat
1
workScheduleStart
1
WorkScheduleStartServiceListener
«Internal»
+action(JSLEvent):void
workScheduleFinished 1
WorkScheduleFinishedServiceListener
«Internal»
+action(JSLEvent):void
Lot
-currentStep:ProcessStep
-currentStopCount:double
-dispatchingData:DispatchingDataSet
-forced:boolean
-lotData:LotDataSet
-lotStopTimeBetweenGenerator:DEmpirical
-lotStopTimeInGenerator:DEmpirical
-model:FabModel
-nextStep:ProcessStep
-priorityTag:List<Double>
-rulePrioritiesNorm:Map<String,Double>
-rulePrioritiesUnnorm:Map<String,Double>
-sequence:ProcessStepSequence
-solvedStandardRawProcessTime:double
-startInQueue:Double
-stopped:boolean
-stoppedByRule:boolean
-waferCount:int
#initialize():void
+Lot()
+compareTo(Lot):int
+jumpToNextStep():boolean
-calculateSolvedRawProcessTime():double
-generateNextStep():ProcessStep
lotStopEndServiceListener 1
LotStopEndServiceListener
«Internal»
+action(JSLEvent):void
lotStopStartServiceListener 1
LotStopStartServiceListener
«Internal»
+action(JSLEvent):void
equipmentQueue
*
lotsOnPrevEquipment
*
DownStartServiceListener
«Internal»
+action(JSLEvent):void
DownEndServiceListener
«Internal»
+action(JSLEvent):void
downStartServiceListener 1
downEndServiceListener 1
EquipmentStation
-callingOperatorState:ModelElementState
-currentDownEventId:int
-currentLoad:double
-currentSetupState:EquipmentSetupState
-currentState:ModelElementState
-downState:ModelElementState
-equipmentDataSet:EquipmentDataSet
-equipmentEndEvent:JSLEvent
-equipmentName:String
-equipmentStatistics:EquipmentStatistics
-idleState:ModelElementState
-lastSetupChange:double
-localDispatchRule:DispatchRule
-localSetupRule:SetupRule
-model:FabModel
-nextBatch:Batch
-operatorEndEvent:JSLEvent
-operatorResource:ResourceSet
-prevState:ModelElementState
-processState:ModelElementState
-returnToBatch:Batch
-setupEndEvent:JSLEvent
-setupState:ModelElementState
#initialize():void
+EquipmentStation()
+arriveLot(Lot):void
+callLotIncoming():void
+initLotsToProcess(List<Lot>):void
+toString():String
OperatorAllocatedServiceListener
«Internal»
+allocated(Request):void
operatorAllocatedServiceListener 1
EnterEquipmentStateListener
«Internal»
+compareTo(StateEnteredListenerIfc):int
+update(ModelElementState):void
enterEquipmentStateListener
1
EquipmentEndServiceListener
«Internal»
+action(JSLEvent):void
equipmentEndServiceListener
1
OperatorEndServiceListener
«Internal»
+action(JSLEvent):void
operatorEndServiceListener
1
setupChangeEndServiceListener 1
SetupChangeEndServiceListener
«Internal»
+action(JSLEvent):void
batchEndServiceListener 1
BatchEndServiceListener
«Internal»
+action(JSLEvent):void

Figure 13.7: Implemented simulation entity classes

Besides the important model entities, the model is fed by certain data object elements describing the behavior of the model elements. These objects include for example the lot data (current lot states), the equipment data (current state, planned down activities), the operator data, the product data, and the transport data. The objects are generated by the data interaction interface of the system.

The statistical output of the model runs is generated by several statistical objects describing several points of view like

- the facility view,
- the product view,
- the lot view and
- the equipment view.

Each statistical object has a large variety of statistical parameters like the average, the median, the min value, the max value and the deviation. In addition, the 50th and 90th percentile are available. After each simulation, there is an option to generate several simulation outputs as XML or MS Excel files.

### 13.4.3.2 Automated Model Generation

The automated model generation is the most important step in estimating a reasonable dispatch weight combination. The model generation starts with the data collection. In our case, we define several data collection methods gathering data from the entities mentioned in Section 13.4.2. The current state information of the lots, the pieces of equipment and the current schedules are the prime points of interest. The optimization task is accomplished within a time frame of about one week from the current time to the future. This includes planned down events of equipments, planned work schedules of operator staff and new lot starts. This data collection procedure provides the raw data for the simulation model. During the model generation process, two stages of system checks are performed:

1. Model data correctness and plausibility
2. Model validation

The first point includes the correct collection of valid data. Therefore the model domains lot, equipment, operator and product are analyzed for their inner and outer correctness and plausibility. Inner correctness means, that all data inside a domain is consistent. For example for each product step there is at least one equipment mapping with the corresponding times. We define several of these plausibility rules $P(D)$ of each domain $D$. Outer correctness defines the correctness when comparing several data domains. For example, a workstation is defined in a product step, but not in the equipment domain. At least two reactions are possible. In case of non-critical plausibility problems (as in more equipments available than required for the products), the data collection can continue. Critical problems are problems where the data collection process must be aborted, e.g. there is no operator for an equipment group which is used in a product step. For the inner correctness of the product domain, several examples are illustrated in Table 13.2.

After the model data consistency checks, the model is validated by historical model runs. The corresponding historical result parameters are collected, including performance measures. The historical results are compared with the model result. For the validation run, a maximal difference between the model and the historical result is defined. In case of a difference above the maximum, the model generation process is aborted.

#### 13.4.3.3 Optimization

After the successful generation and validation of the model data, the optimization procedure can be commenced. The involved classes of the implementation are presented in Figure 13.8. The classes can be divided into the algorithmic objects and data objects. The data objects contain required information about the optimization process and the algorithm. The algorithm implementation is realized with the abstract class OptimizationAlgorithm. This class provides a defined structure which each optimization algorithm has to implement. Therefore the usage of different implementations is possible. To complete the algorithm implementation, three methods have to be applied:

- changeParameterForIteration: generation of new weight combination
- objectiveFunction: calculation of a numeric evaluation of the simulation run
- checkIterationToBeTheBest: checks if the iteration is the best one

| Division | Relation to data domain | Rule name | Rule description | Description |
|---|---|---|---|---|
| Inner | | step completeness | $P_{ICS}(D)$ | At each stage a complete set of times and equipments is available (at least one element) |
| | | product completeness | $P_{ICP}(D)$ | All standard flow steps for a product are well defined in the right order |
| | | stage time plausibility | $P_{IPS}(D)$ | e.g. $t_p + t_w \leq t_o$ ($t_p$- process time, $t_w$ - additional waiting time, $t_o$ - operator time);<br>$t_p > 0$ (no zero process time) |
| | | ... | ... | ... |
| Outer | Equipment | equipment completeness | $P_{OCE}(D)$ | Each equipment defined at the product stages must be available in equipment data domain |
| | Operator | operator completeness | $P_{OCO}(D)$ | At least one operator must be available in operator data domain for solving step (or $t_o = 0$) |
| | ... | ... | ... | ... |

Table 13.2: Inner product data domain correctness and plausibility rules

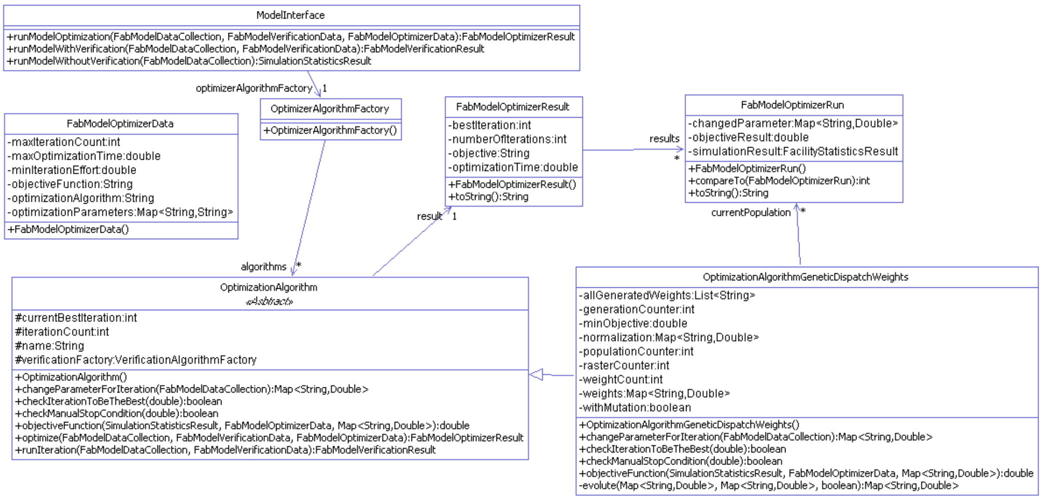

ModelInterface
+runModelOptimization(FabModelDataCollection, FabModelVerificationData, FabModelOptimizerData):FabModelOptimizerResult
+runModelWithVerification(FabModelDataCollection, FabModelVerificationData):FabModelVerificationResult
+runModelWithoutVerification(FabModelDataCollection):SimulationStatisticsResult
optimizerAlgorithmFactory
1
OptimizerAlgorithmFactory
+OptimizerAlgorithmFactory()
FabModelOptimizerData
-maxIterationCount:int
-maxOptimizationTime:double
-minIterationEffort:double
-objectiveFunction:String
-optimizationAlgorithm:String
-optimizationParameters:Map<String,String>
+FabModelOptimizerData()
FabModelOptimizerResult
-bestIteration:int
-numberOfIterations:int
-objective:String
-optimizationTime:double
+FabModelOptimizerResult()
+toString():String
results
*
FabModelOptimizerRun
-changedParameter:Map<String,Double>
-objectiveResult:double
-simulationResult:FacilityStatisticsResult
+FabModelOptimizerRun()
+compareTo(FabModelOptimizerRun):int
+toString():String
result
1
currentPopulation
*
algorithms
*
OptimizationAlgorithm
«Abstract»
#currentBestIteration:int
#iterationCount:int
#name:String
#verificationFactory:VerificationAlgorithmFactory
+OptimizationAlgorithm()
+changeParameterForIteration(FabModelDataCollection):Map<String,Double>
+checkIterationToBeTheBest(double):boolean
+checkManualStopCondition(double):boolean
+objectiveFunction(SimulationStatisticsResult, FabModelOptimizerData, Map<String,Double>):double
+optimize(FabModelDataCollection, FabModelVerificationData, FabModelOptimizerData):FabModelOptimizerResult
+runIteration(FabModelDataCollection, FabModelVerificationData):FabModelVerificationResult
OptimizationAlgorithmGeneticDispatchWeights
-allGeneratedWeights:List<String>
-generationCounter:int
-minObjective:double
-normalization:Map<String,Double>
-populationCounter:int
-rasterCounter:int
-weightCount:int
-weights:Map<String,Double>
-withMutation:boolean
+OptimizationAlgorithmGeneticDispatchWeights()
+changeParameterForIteration(FabModelDataCollection):Map<String,Double>
+checkIterationToBeTheBest(double):boolean
+checkManualStopCondition(double):boolean
+objectiveFunction(SimulationStatisticsResult, FabModelOptimizerData, Map<String,Double>):double
-evolute(Map<String,Double>, Map<String,Double>, boolean):Map<String,Double>

Figure 13.8: Classes for the optimization process

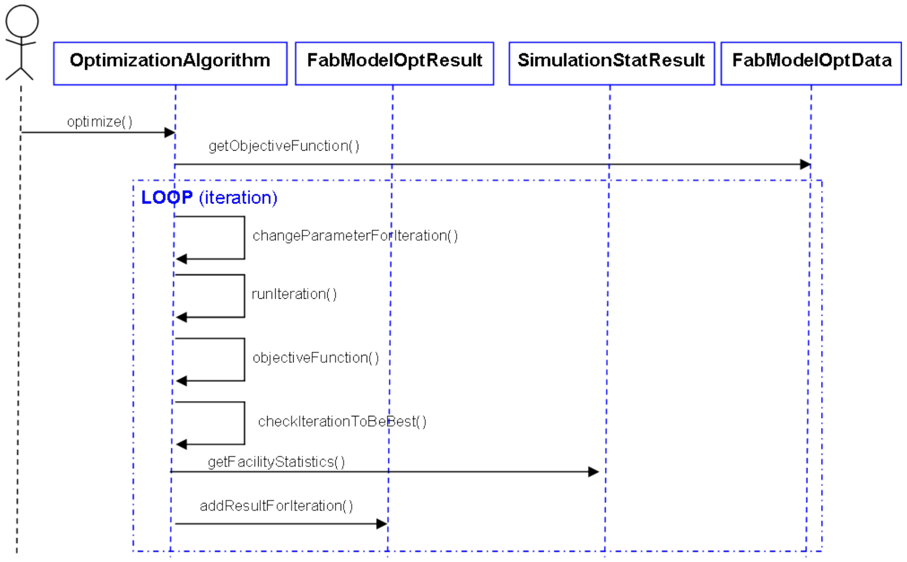


Figure 13.9: Sequence diagram the optimization task procedure

The optimization can be started by calling the runModelOptimization method of the model interface class. After each iteration, a data element is generated providing basic information of the optimization run (FabModelOptimizerRun). When the process is finished, a result object (FabModelOptimizerResult) provides all necessary information about the optimization process including the best weight combination for the given objective. Figure 13.9 introduces the sequence of method calls. In general five steps per iteration have to be followed:

1. Generate a new weight combination which represents a new population member according to the genetic principle.
2. Run the simulation with the new weight combination.
3. Generate the objective function value representing the simulation result.
4. Adding the result to the current population and check if the result is the best one.
5. Go to (1) if iteration stop conditions are not true.

The optimization task can be called manually by using the user interface implementation (see Section 13.4.5) or periodically at fixed time frames. Both methods allow a very flexible usage of the optimization task.

### 13.4.4 Web Service Implementation

The implemented web service functionality can be divided into dispatching and simulation aspects. The system provides the following methods:

- Methods to determine lot moves for an internal update of the lot priorities.
- Methods for system initialization and configuration.
- Methods for simulation and optimization of historical and current facility data sets.
- Methods for generation of a dispatching list for workstation.

The web service calls can be divided into asynchronous calls without direct response and synchronous calls with a defined response. Asynchronous calls are used for time consuming operations. These operations include recalculation of the lot priorities or simulation and optimization tasks. The synchronous calls are used for generating a defined server response. This type of call includes for example the generation of a dispatching list for an equipment. An example of a typical web service method definition is illustrated in Figure 13.10.

For each method, several annotations are used (@ notation), which describe the type of web service method. In our example, the name of the web service (@Path), the type (@GET or @POST) and the parameters are defined. For a closer look into details, we introduce the web service call of a lot priority update during the lot movement to the next processing step. This example is called asynchronously to avoid influences on other sub

```
@Path("")
public interface GDCServices {

/**
* WebService for getting the dispatch list for an equipment
* @Param equipment the equipment name
* @return the dispatch object with all information for the dispatch
* @throws Exception
*/
@GET
@Path("dispatchlistforequipment")
@Produces(MediaType.APPLICATION_JSON)
public ClientResponse<DispatchObject> getDispatchListForEquipment(
@QueryParam("equipment") String equipment) throws Exception;

/**
* WebService starting an optimization of the weights
* @param asynchronious if true asynchronious call default is true
* @throws Exception
*/
@POST
@Path("optimize")
public Response.Status runOptimization(
@QueryParam("asynch") @DefaultValue("true") Boolean asynchronious) throws Exception;

//further methods...
```

Figure 13.10: Typical web service method interface declaration

systems by waiting for the response. Figure 13.11 illustrates the cut sequence flow of the whole method.

The web service, in our case the movement trigger of the MES, indicates a lot move. During the trigger execution, the web service for the priority calculation is called. Inside the web service, several operations are executed. The outdated priorities of the lot are removed first. For each production lot in the facility, several priorities for the different dispatching rules are stored. For each rule, a not-normalized and a normalized priority is stored. These priorities are used by the dispatching system to sort the lots effectively according to their weighted importance through the combined dispatching approach. After removal, the priorities are recalculated by the dispatching rule implementations. The implementations of the dispatching rules are provided by a rule factory using the *factory pattern*[3]. The factory pattern offers the possibility of using different implementations from the same problem domain. This can be applied to the dispatching approaches as well.

The normalization procedure is activated if the calculated priorities of the moved lot exceed the former maximal or minimal priorities for the dispatching rule. In this case, all normalized priorities are changed due to the new normalization limits. Finally the resulting priorities for the lot are written back to the internal priority database of the dispatching system.

The the largest fraction of the web services interacts with database objects. Transactional safety is very important. Modifications to the system should be made in an orderly manner. Multiple changes of the system at the same time are not possible. The whole mechanism is provided by common JAVA based transactional solutions like the JPA[4] implementation.

[3] In the object oriented programming environment, several programming patterns are known for solving common problems and offering best practice solutions

[4] The JAVA Persistence API is a library offering various techniques in object oriented access and interaction with relational database systems

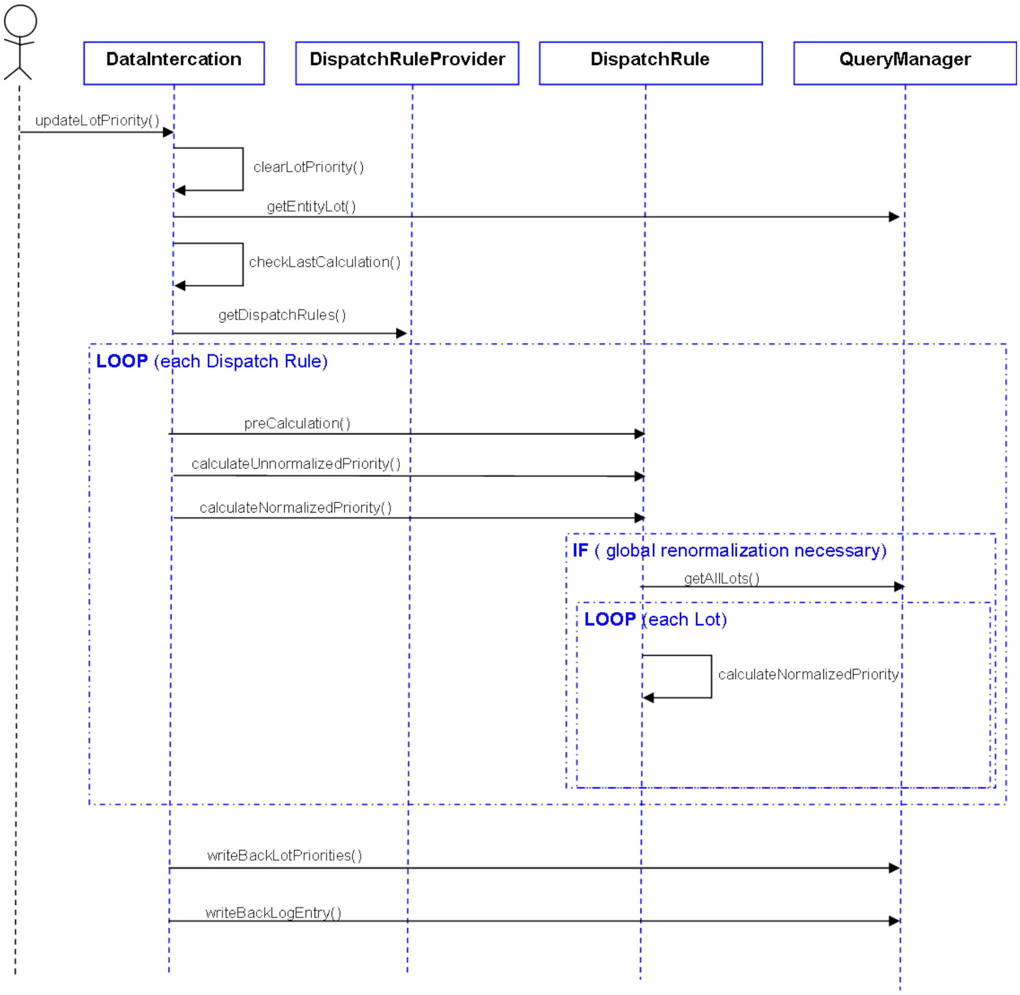
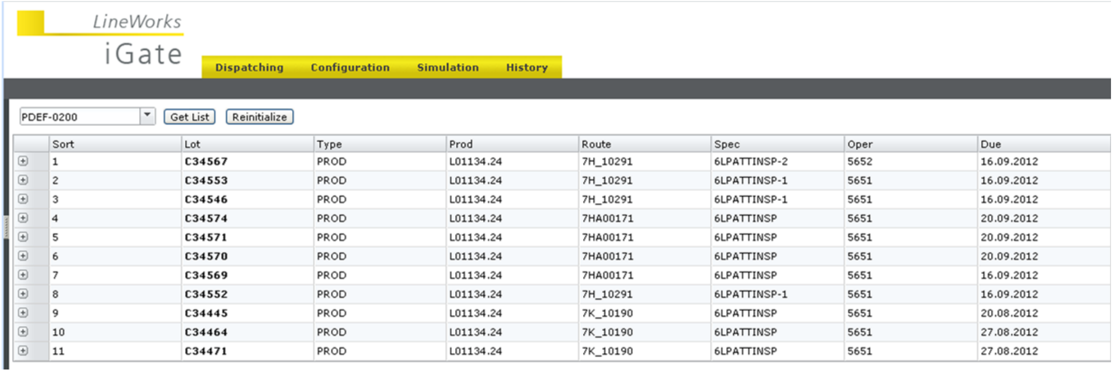


Figure 13.11: Sequence diagram of the lot update method



| | Sort | Lot | Type | Prod | Route | Spec | Oper | Due |
|---|---|---|---|---|---|---|---|---|
| ⊞ | 1 | **C34567** | PROD | L01134.24 | 7H_10291 | 6LPATTINSP-2 | 5652 | 16.09.2012 |
| ⊞ | 2 | **C34553** | PROD | L01134.24 | 7H_10291 | 6LPATTINSP-1 | 5651 | 16.09.2012 |
| ⊞ | 3 | **C34546** | PROD | L01134.24 | 7H_10291 | 6LPATTINSP-1 | 5651 | 16.09.2012 |
| ⊞ | 4 | **C34574** | PROD | L01134.24 | 7HA00171 | 6LPATTINSP | 5651 | 20.09.2012 |
| ⊞ | 5 | **C34571** | PROD | L01134.24 | 7HA00171 | 6LPATTINSP | 5651 | 20.09.2012 |
| ⊞ | 6 | **C34570** | PROD | L01134.24 | 7HA00171 | 6LPATTINSP | 5651 | 20.09.2012 |
| ⊞ | 7 | **C34569** | PROD | L01134.24 | 7HA00171 | 6LPATTINSP | 5651 | 16.09.2012 |
| ⊞ | 8 | **C34552** | PROD | L01134.24 | 7H_10291 | 6LPATTINSP-1 | 5651 | 16.09.2012 |
| ⊞ | 9 | **C34445** | PROD | L01134.24 | 7K_10190 | 6LPATTINSP | 5651 | 20.08.2012 |
| ⊞ | 10 | **C34464** | PROD | L01134.24 | 7K_10190 | 6LPATTINSP | 5651 | 27.08.2012 |
| ⊞ | 11 | **C34471** | PROD | L01134.24 | 7K_10190 | 6LPATTINSP | 5651 | 27.08.2012 |

Figure 13.12: User interface of the dispatch control system

### 13.4.5 User Interface

The user interface for the dispatch control mechanism uses a web-based solution. The user interface is structured according to the following elements:

- Dispatching information:
  - Dispatch list generation: offers the functionality for a dispatch list generation per equipment with an extended batch preview and setup mechanism.
  - Lot priority overview: shows the calculated dispatching priorities per lot for each dispatching rule.
- Configuration:
  - Dispatch level administration: The dispatching levels define a sort direction for each priority provided by the system or the dispatching controller. At least three level types are available. The first one indicates a priority provided by the MES. The second level type indicates a single dispatching rule. The third level type is used for a combined dispatching policy. Each level has a defined sort index according to which the lots are finally sorted.
  - Dispatch rule administration: allows adding and editing dispatching rules.
  - Optimization administration: allows setting various optimization parameters such as the objective function or the model parameters such as run length and number of replications.
- Historical output: The historical output offers the ability to obtain the complete system log.

Each element is protected by a user right, which is required to modify the parameters of the system. An example of the layout is illustrated in Figure 13.12.

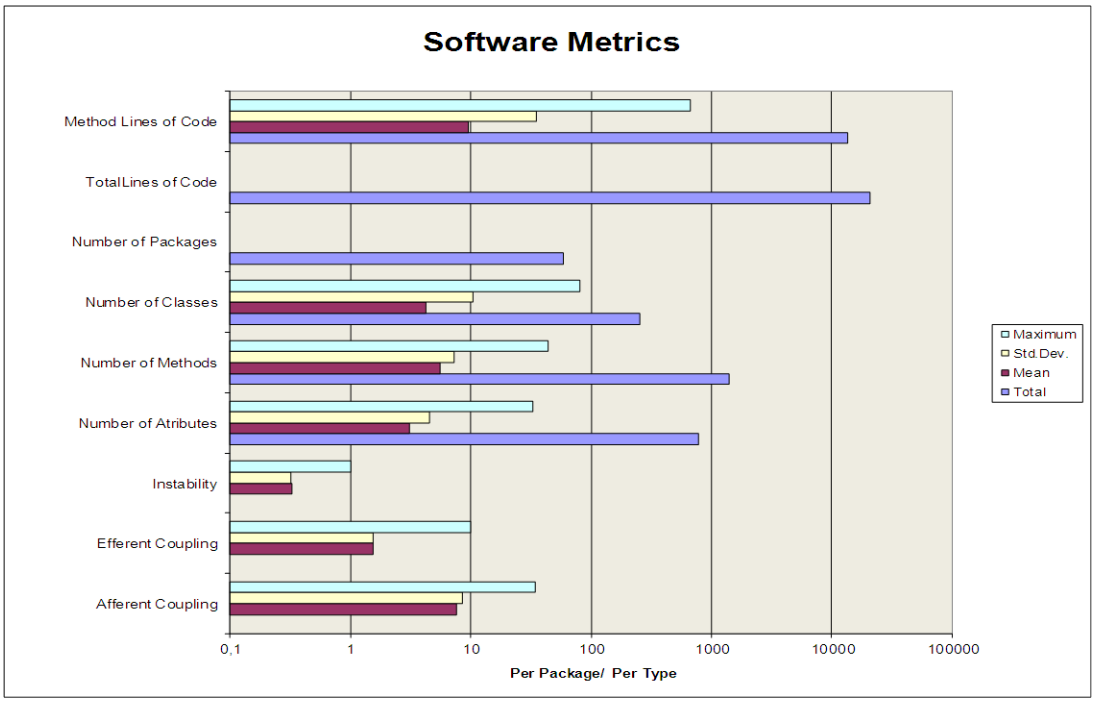


Figure 13.13: Software metrics

### 13.4.6 Important Software Metrics

Software metrics are mostly mathematical functions which quantify a property of a software system. Goodman (see [Goo93]) defines software metrics as ”the continuous application of measurement-based techniques to the software development process and its products to supply meaningful and timely management information, together with the use of those techniques to improve that process and its products”. In Figure 13.13, a short overview is given about common metrics of the entire controller implementation. The total number of lines of code is 20880. There are 59 packages at the project with 251 classes and interfaces. Furthermore other metrics are mentioned. The afferent coupling (CA) is the number of further packages that depends upon classes within the package. It is an indicator of the package responsibility. The efferent coupling (CE) is an indicator for the package independence and defines the number of other packages that classes in the package depend upon. With these both values, the instability (I) can be calculated as

$$I = \frac{CE}{CE + CA} \tag{13.1}$$

where $I = 1$ indicates a completely unstable package and $I = 0$ marks a completely stable package. This metric defines an indicator of the package resilience to change. In our case the average instability is about 0.3, which indicates a good resilience to changes. For the definition of further metrics we refer to [Goo93].

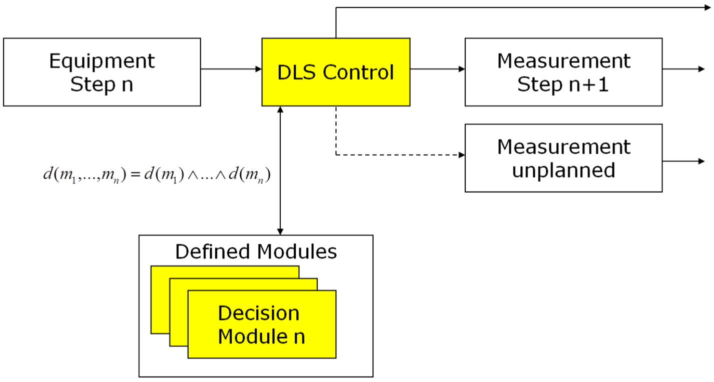


Figure 13.14: Overview of the lot sampling procedure

## 13.5 Lot Sampling at Metrology Operations

### 13.5.1 Global System View

The lot sampling procedure is designed in the following way. Figure 13.14 introduces the general approach. As introduced in Section 12.1.2, the skip of planned measurement operations (type A) and the forcing of additional unplanned measurement steps (type B) are the main application areas of the approach.

For sampling type A we define a first set of modules for the lithography environment[5] as follows:

- **Process capability index (CPK) observation module**: The CPK observation module monitors the process capability index of the corresponding process. The CPK can be calculated as follows:

$$CPK = \frac{\min(\mu - LSL; USL - \mu)}{3\sigma} \tag{13.2}$$

  The $LSL$ defines the lower specification limit, the $USL$ the upper specification limit. A minimal limit $CPK_{Min}$ is defined. In case of a lower value, which indicates an unstable process, the measurement is forced. Today's common values for stable processes range between 1 and 2 (see Six Sigma approach in [Pyz97]). With this module very stable processes can be identified. Sampling of unstable processes is not reasonable, thus full measurement is forced. For illustration of the module behavior, we run several historical tests with real facility data on a process with a high degree of stability (with $CPK_{Min} = 2$). The results are illustrated in Figures 13.15 and 13.16. In the middle of the historical CPK curve, there is a process stability problem which decreases the CPK value under $CPK_{Min} = 2$. In this case the module enforces a full measurement. Otherwise, in combination with the sampling frequency module, a normal measurement every five lots of the product group is enforced.

[5] The set of modules of type A represents a first use case. Further modules can be easily extended.

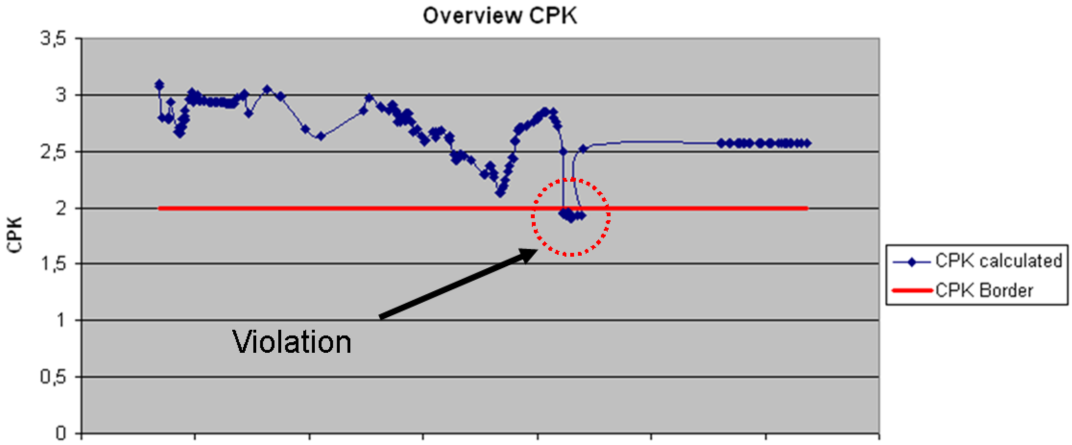


Figure 13.15: CPK module illustration at historical data - CPK value

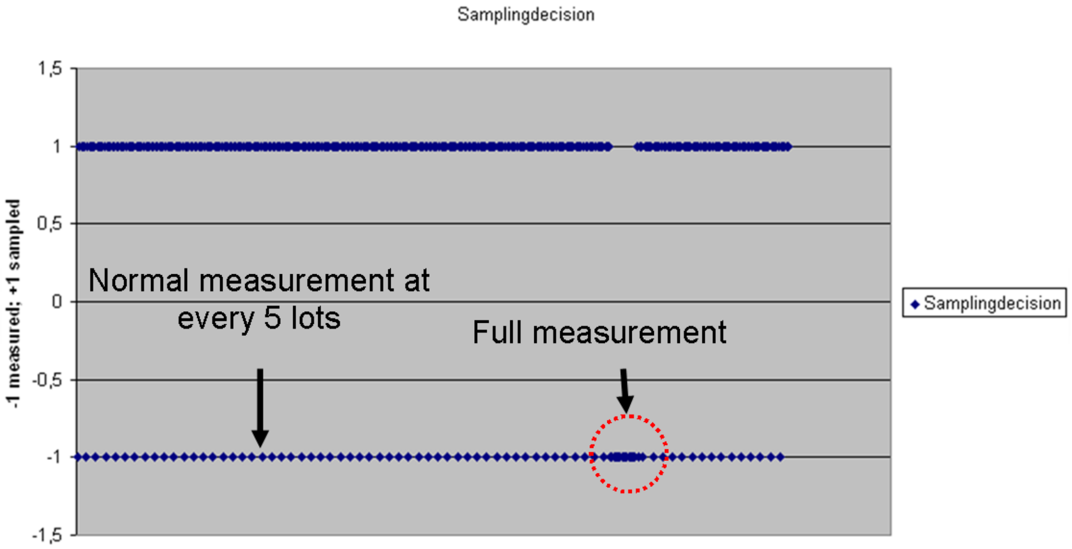


Figure 13.16: CPK module illustration at historical data - Sampling decision

- **Sampling frequency observation module**: In order to obtain an adequate amount of values for the calculation of process capability curves and functions, a steady measurement at a defined rate is necessary. Furthermore besides the defined minimal rate, a maximal time between two measurements at the product group is defined. In a foundry, at products with a low starting rate of about one to twenty lots per month, the process properties can change dramatically. This may not be detected without definition of a maximal time. Therefore this module guarantees a minimal process observation.

- **Equipment correlation observation module**: In historically grown semiconductor facilities, there are different pieces of equipment available for performing the same processes. Some tools are older showing more unstable process behavior. The newer ones have an improved process behavior. So processing at certain pieces of equipment must be observed strictly to avoid problems. In case of a lot, which is historically processed by a defined equipment or group, the measurement is forced.

- **Lot irregularity observation module**: Notified lot problems which cause rework loops or other issues need specific observation. These lots cannot skip a planned measurement step to avoid further problems in the process.

Further modules can simply be added and implemented to this approach, like an extended metrology module for measurement value prediction. For sampling lot type B we define a general module type for line product observation:

- **Additional measurement observation module**: In a semiconductor production line, the line stability is observed by several measurement objectives through the whole line. There are defined specific product types representing the wide range of different products in the line. The parameters and last measurements have to follow a defined time frame, in which an additional measurement has to be taken. For this, the module type can be used to define measurement objectives and time frames for enforcing additional measurements.

The defined approach allows a very wide use at different process stages and operations with a predefined set of modules. In general the definitions and options of each module have to be defined by engineering personal. The implementation can also be done by the engineering personal. Due to the very inhomogeneous IT landscape, each module has to define the data access to the input data by itself. A disconnection of the module from the controller design is applied.

### 13.5.2 System Implementation Aspects

The lot sampling controller system consists of two main elements, the controller logic and the module implementations are provided as JAVA classes. The communication is possible via web service calls. At the administration interface, a sampling decision step can be defined by setting various parameters such as the operation, the recipe or specification, the equipment name, the technology identifier, or the product identifier.

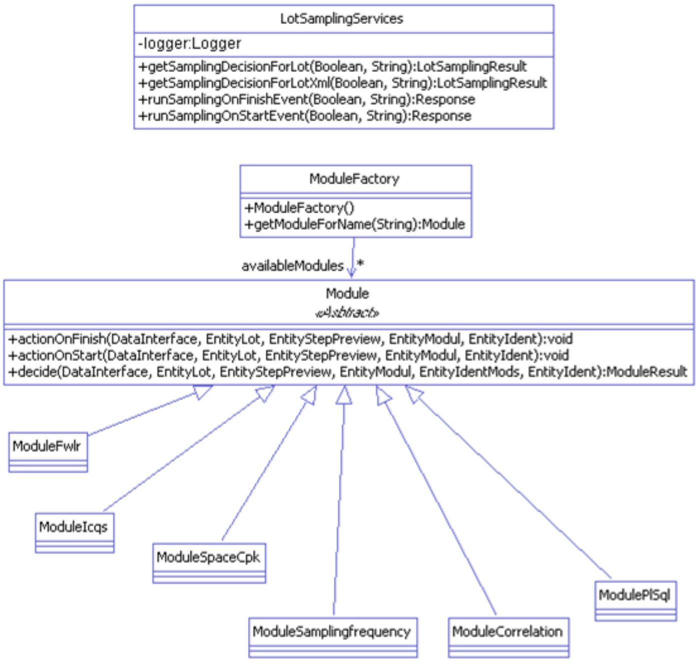

LotSamplingServices
-logger:Logger
+getSamplingDecisionForLot(Boolean, String):LotSamplingResult
+getSamplingDecisionForLotXml(Boolean, String):LotSamplingResult
+runSamplingOnFinishEvent(Boolean, String):Response
+runSamplingOnStartEvent(Boolean, String):Response
ModuleFactory
+ModuleFactory()
+getModuleForName(String):Module
availableModules
*
Module
«Abstract»
+actionOnFinish(DataInterface, EntityLot, EntityStepPreview, EntityModul, EntityIdent):void
+actionOnStart(DataInterface, EntityLot, EntityStepPreview, EntityModul, EntityIdent):void
+decide(DataInterface, EntityLot, EntityStepPreview, EntityModul, EntityIdentMods, EntityIdent):ModuleResult
ModuleFwlr
ModuleIcqs
ModuleSpaceCpk
ModuleSamplingfrequency
ModuleCorrelation
ModulePlSql

Figure 13.17: Module class overview

For each sampling step, a number of different modules is implemented for providing the functionality for the sampling demands. Figure 13.17 illustrates the main module implementations. Each of the modules is a member of the abstract module class providing the required method definitions. Besides modules providing pure JAVA implementations like the CPK calculation, it is also possible to use database functions with a predefined function header. At least the decision method has to be implemented to generate a new sampling module. Furthermore, there are possibilities for pre-calculation of certain events. The methods actionOnFinish and actionOnStart are called at lot finish and lot start. During this function call, larger database operations can be carried out to save operational time during the decision process.

The decision process is illustrated in Figure 13.18. If a web service call is applied, the first step is to search for steps for sampling of the lot in the future. If this includes the next or current step, the sampling system next identifies the administration sequence corresponding to the step. In the case of no administration, the sampling process is stopped and the normal schedule of the lot is continued. Otherwise, the modules corresponding to the administration are loaded. For each of the loaded modules, the decision method is called generating module specific results. These results are combined to a final decision for the step.

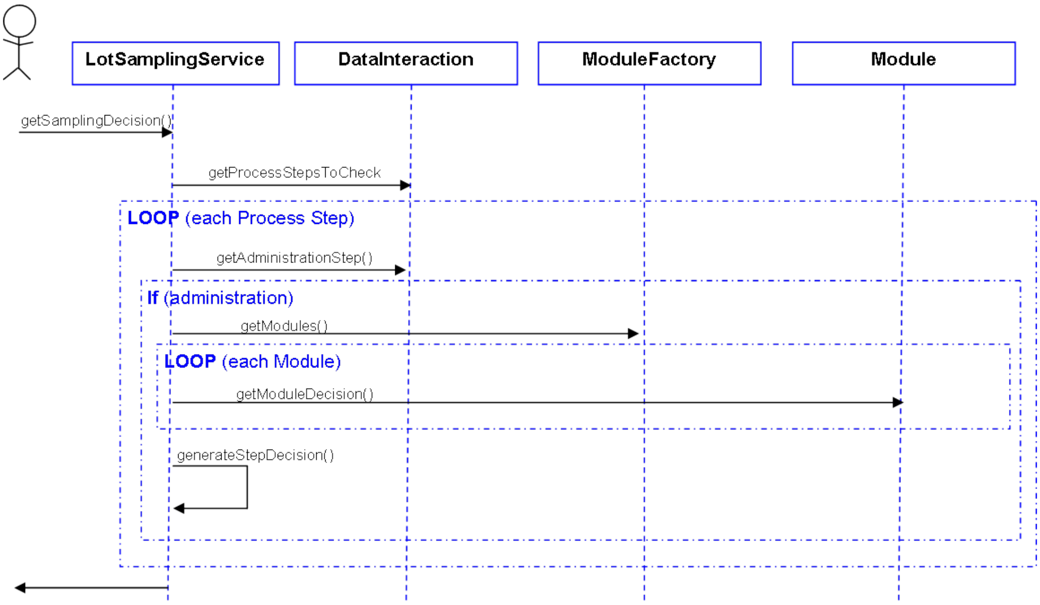


Figure 13.18: Sequence of the sampling decision process

# 14 Prototype Evaluation

## 14.1 Test and Roll-Out Scenarios

In this section, we introduce our system test and roll-out procedure for the final productive usage of the system.

### 14.1.1 System Test Procedure

For a sufficient system test procedure, we define three stages of detail:

1. Local method-dependent unit tests
2. Basic web service tests
3. Performance and stability tests

The first two stages include a detailed unit and method testing with common techniques like the usage of JUNIT[1]. These tests are done to avoid logical and structural system problems and to guarantee the valid functionality of the whole system.

The performance and stability tests are very vital to avoid problems due to long waiting times or database instabilities. The performance and stability tests are done for all elements of the system including the real-time dispatching module of the dispatch controlling system. In this case the dispatch list generation and priority recalculation at lot move are analyzed. These two elements are exemplified in the next section.

#### 14.1.1.1 Synthetic Stability Test Example

The stability tests at the dispatch case are performed on a closed test system including a copy of real factory data. Besides the hardware configuration, the test system represents the real facility data infrastructure.

We run different test cases at various calling frequencies. Each frequency describes the average value $f_{avg}$ of a $U(0; 2 * f_{avg})$ distributed random variable to represent non-static calling events. The usage of static frequencies for the test case is not reasonable. In reality, the web service calls happens in a random manner. The stability test has to simulate the real calling occurrence as life-like as possible. We use the frequencies $f_{call} = \left\{\frac{1}{2s}; \frac{1}{5s}; \frac{1}{10s}; \frac{1}{15s}; \frac{1}{25s}\right\}$. At a very high facility load, the frequencies can be approximated to $f_{move} \approx \frac{1}{15s}$ and $f_{dispatch} \approx \frac{1}{5s}$. To verify the system stability, we use an unrealistic calling

[1] JUNIT is a useful framework for automated testing of certain classes or methods in the JAVA programming environment.

frequency of $f = \frac{1}{2s}$. The three test cases include the dispatch list generation, the priority calculation and a combination of both. Figure 14.1 illustrates the method call duration with the minimum, maximum, median and the 90th quartile of the test run. Each test is run for 30 minutes.

In the case of the lot priority only stability test, the average duration of each call is approximately 2.5 seconds. The calls are made asynchronously, the system performance of the MES is not influenced. The maximal duration growth of 60% from the lowest up to the highest frequency level is a result of several short inter-event times. These events decrease the system performance. The average value is also increasing by about 7%. The dispatch list generation, which is called synchronously, has an average duration of about 1.5 seconds. The customer has to wait that time until the dispatch list is shown. Besides the database interaction time, the number of lots influences this duration. At all test cases, we use a 10 lot dispatch list length on average.

The most interesting test case is the third one, were both elements are tested simultaneously. We divide three different scenarios, the $f_{call} = \frac{1}{10s}$ describes a low utilized facility, the $f_{call} = \frac{1}{5s}$ a facility with a very high load, and the $f_{call} = \frac{1}{2s}$ is used to verify system stability. Both first realistic cases show a very stable average calling duration. Of course, the maximal duration grows with an increased calling frequency. The increases remain reasonable and do not exceed 10%. The unrealistic third case shows the limits of the system. The average call duration increases by 25% in both cases. The maximal value is increased dramatically. Higher calling frequencies lead to an unstable system.

#### 14.1.1.2 Real Stability Test Example

Besides the synthetic tests at the test system, a test with real staff is performed. An exemplary section of the work flow of a certain product is used. Different persons take different equipments[2] along this work flow and perform the “real” production tasks, including dispatch list generation and virtual lot movements. This “production tasks” are done individually by each person at a random manner. The work flow covers 10 production steps. The overall test time is about one hour, including 294 server methods calls. Each person is assigned two different equipment groups:

- Person 1: Implantation
- Person 2: High Temperature
- Person 3: Microscope Control
- Person 4: Cleaning
- Person 5: Etch
- Person 6: Lot Reordering

[2] The equipments are simulated virtually at the test system allowing virtual lot preparation and movement like at the real system.

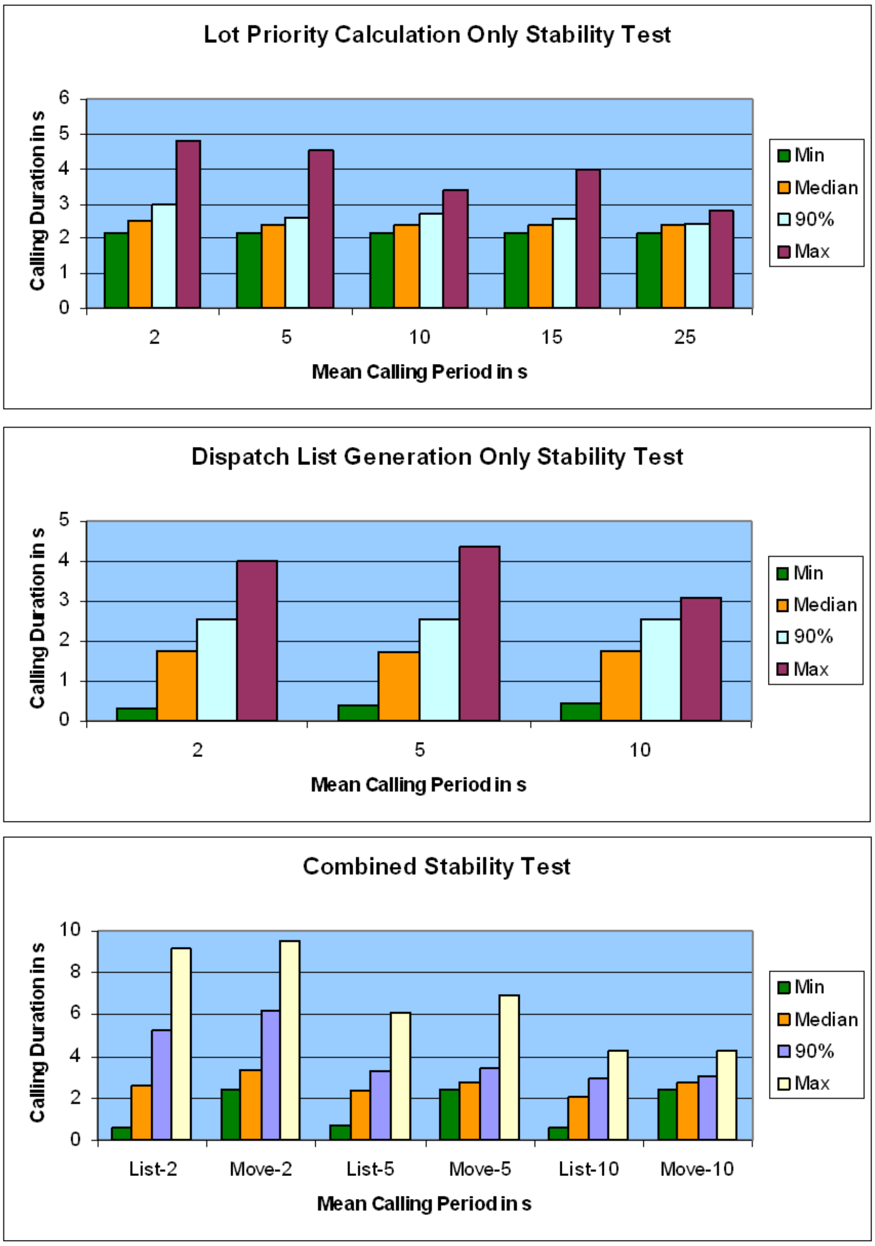


Figure 14.1: Synthetic stability test results for real time dispatcher

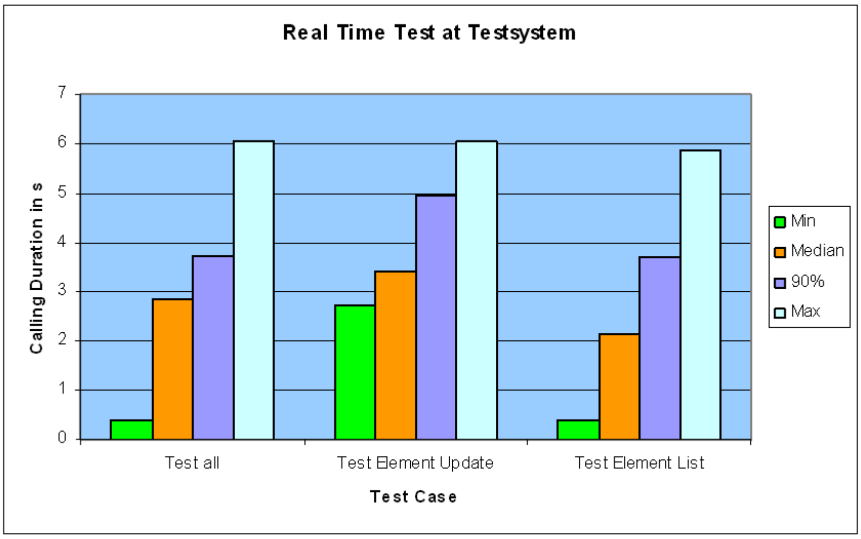


Figure 14.2: Real time test results for real time dispatcher

The last person is responsible to assign new lots to the first step of the work flow section chosen. These lots are generated and assigned by the MES manually at the test system.

In general (see Figure 14.2) the behavior is similar to the synthetic tests. At certain time frames, several calls are generated simultaneously, resulting in longer durations of the method calls. Thus variation is about 50% higher than in the synthetic tests.

#### 14.1.1.3 Conclusion

The system tests are including a wide range of different performance and stability analysis results in a stable and reliable system. The Unit-Tests at code level guarantee a minimization of logical and structural code errors. The representation of the productive environment by a complete copy of the productive system also allows us to analyze the system under different conditions. At the productive system, it is estimated to lower the average method durations due to a more powerful hardware configuration.

### 14.1.2 System Roll-Out Procedure

After the successful system test of the whole application, the system introduction to the productive environment is described. For introduction of the new systems to an existing IT landscape, several scenarios are known and used. In general we can divide two main policies for introduction:

1. Full introduction at a defined date

2. Smooth introduction with several defined mile stones at certain dates

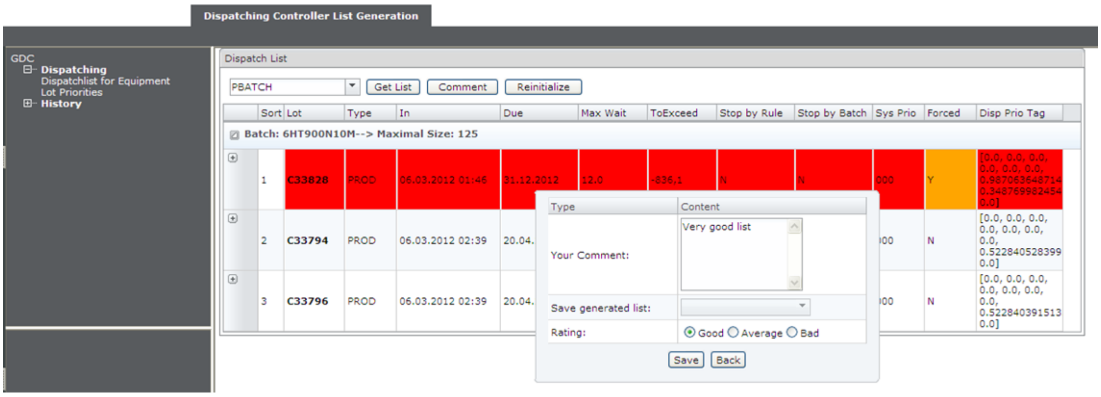
The first point is chosen in case of very complex IT landscapes and has several benefits. The reduced effort of managing two systems in parallel (the old one and the new one), or the complete availability of the system to the whole facility are two advantages. This scenario requires a very detailed preparation and planning period. In case of problems and failures the consequences are more dramatic than in case two. The system to be replaced is not available any more and can only be recovered by defined roll back scenarios. The second case is often used if all conditions and possible problems are not known until the introduction to the real system. The time until the productive release tends to be longer than in point one. Another advantage of point two is a better test period with several defined milestone dates in the real system.

For introduction of the dispatching and lot sampling system, we choose the second roll-out scenario. The effects to the real system are not known to a detailed level for scenario one. For the dispatching system, the following milestones are defined:

1. Evaluation at test system
    a) Set up of the whole control system at test system
    b) Definition and evaluation of test cases:
        i. Implantation tools
        ii. Batching area
        iii. Time bound sequences
2. Evaluation at productive system
    a) Set up of the control system at productive system
    b) Shadow evaluation with a parallel web based dispatching system not influencing the real production line
        i. Implantation tools
        ii. Batching area
        iii. Time bound sequences
    c) Introduction to the real dispatching system
    d) Fab wide roll out

The first evaluation step has the objective to provide a valuable statement about the correct system behavior. For this a test system is used providing all productive data without any influence on the real system (see Section 14.1.1). We define three test cases for the first evaluation step, the setup control at implantation tools, the batching ability at a high-temperature heat treatment tools and the time bound sequence steering. These steps provide a significant difference to the existing system without this mechanism. In addition, we choose these elements because the changes are very obvious for the user.

With step two, the whole system is transferred to the productive environment. During this step, dispatcher and operator evaluate the dispatching lists. Initially a shadow system



Batch: 6HT900N10M--> Maximal Size: 125

| Sort | Lot | Type | In | Due | Max Wait | ToExceed | Stop by Rule | Stop by Batch | Sys Prio | Forced | Disp Prio Tag |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | C33828 | PROD | 06.03.2012 01:46 | 31.12.2012 | 12.0 | -836,1 | N | N | 000 | Y | [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.987063648714 0.348769982454 0.0] |
| 2 | C33794 | PROD | 06.03.2012 02:39 | 20.04. | | | | | 00 | N | [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.522840528399 0.0] |
| 3 | C33796 | PROD | 06.03.2012 02:39 | 20.04. | | | | | 00 | N | [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.522840391513 0.0] |

| Type | Content |
|---|---|
| Your Comment: | Very good list |
| Save generated list: | |
| Rating: | ● Good ○ Average ○ Bad |

Figure 14.3: Evaluation tool for dispatch list generation

is applied offering a web based interface for dispatch list generation for comparison of the new and old dispatching lists. An example is illustrated in Figure 14.3. With this step, an evaluation is done by the operating staff. Within the web interface, the results can be manually quantified and comments are possible. The results of the productive usage are illustrated in Section 14.2.

A similar way is defined for the introduction of the sampling controller system. We use three application areas for the evaluation of the system, the skipping of a regular inspection step at CD measurements at the lithography area, the enforcing of additional measurements at the back end line control and the enforcing of a manual lot control at in-line defect control. These steps are tested beforehand at the test system and then transferred to the productive system. This includes a test period of a few weeks only advising the operating staff to skip or force measurements without changing the work schedule at the manufacturing execution system. The results of the roll out of both systems on to the real facility environment are introduced in Section 14.2.

## 14.2 Evaluation of the Practical Benefit

In this chapter, we describe the practical results of the usage of both elements in the production environment, the lot sampling and dispatching system.

### 14.2.1 Real-Life Evaluation of Lot Sampling

The evaluation of the practical impact of the lot sampling system is done at the lithography area (CD measurement) with the definition of one representative product group for sampling. During this test period, the utilization of the 6 inch line is very low. Therefore the estimated impact of the sampling result is low as well.

For the evaluation of the results, we focused on a test period of one month with an absolute sampling amount of about 20% for a special metrology task. Figure 14.4 illustrates the median of the step time including queue time and process time. The 30d median is

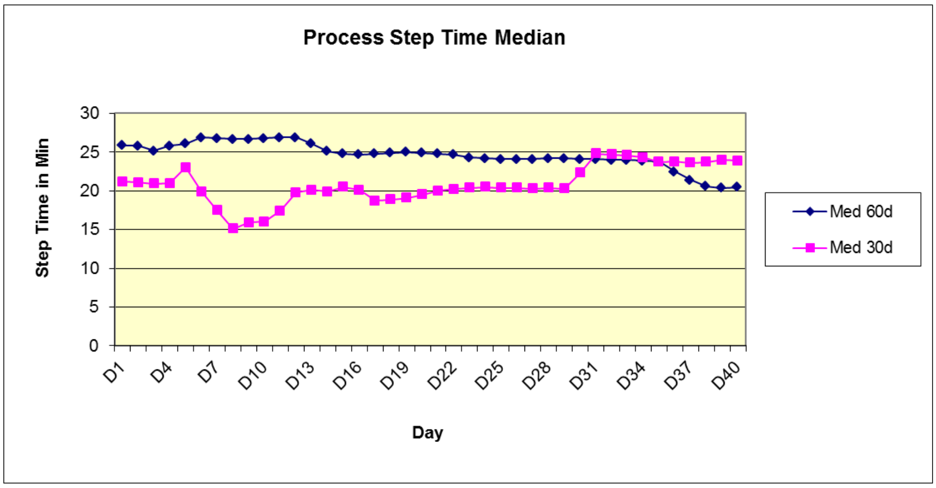


Figure 14.4: Process step time for sampled metrology tool

calculated for the last 30 days, the 60d median for the last 60 days. Between D5 and D13, where the sampling amount is high, the step time could be reduced by about 25%. The 60d median illustrates the negative trend of the step time over the whole time frame.

In general, the sampling mechanism has a larger impact at higher utilized facilities. The example illustrates the potential when sampling is used during more metrology operations throughout the whole facility. Due to the high product mix and the low product volume, the metrology reduction potential is lower than at mass production applications. Our simulation experiments (see Section 12.1.3) show a larger potential in equipment down scenarios and bottleneck situations.

### 14.2.2 Real-Life Evaluation of Dispatch Control System

The evaluation of the dispatching controller system is undertaken for a test period of 4 months. During this time, the utilization of the 6 inch and 8 inch line are low. Due to the low load, impacts of the dispatching system are also low, improvements of the facility KPI can rather be achieved than in the high load case of the simulation model analysis.

#### 14.2.2.1 System Performance

The system performance, which means the measured response and waiting time as well as the verdict from the operating staff, is very vital in a human based production process. To that end, we observe the list generation time statistics as well as the lot update time statistics to represent the major measured performance parameters of our system. Figures 14.5 and 14.6 illustrate the time statistics of the list generation and lot update procedures for the test period (including the start of the productive test). The statistic is collected twice per day for each shift. The lot update procedure, called during the lot movement operation at the MES, shows a very stable behavior over time. At most times, the 90%

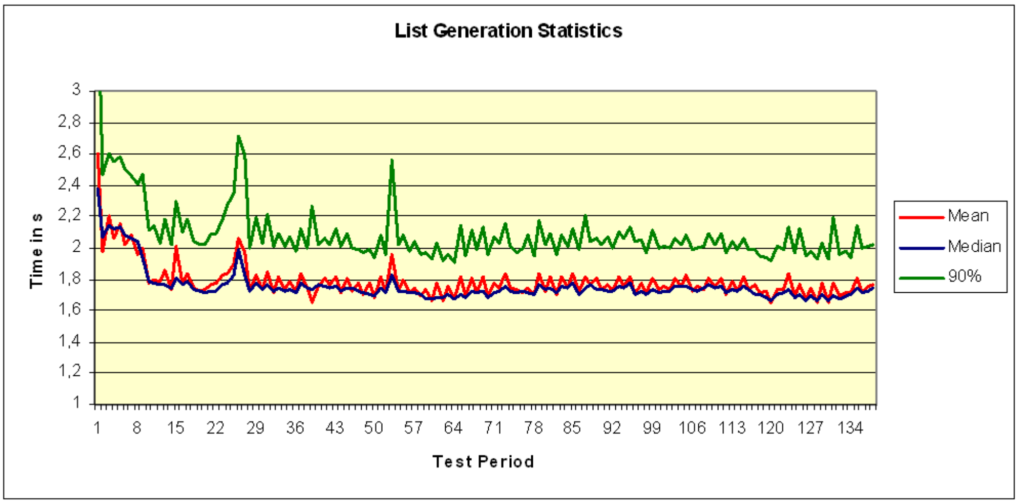


Figure 14.5: List generation statistics of one month per day

percentile is below the two second level. In some cases, the value is above the level, which indicates different simultaneous events slowing down the execution. Furthermore, at the start time period of the system, some optimizations regarding database interaction cause major performance improvements.

The more important statistic is the required time for list generation. This data is acknowledged by the operating staff. Extended waiting times during list request cause dissatisfaction. In our case, the median value is below 2 seconds. In 90% of the cases, the value is below the 2.5 second level at most days during the test period. The fluctuations result from the different load situations of the application cluster and the database system. Values above 3 seconds are not accepted by the operating staff regarding the 90% percentile. The list generation offers the required stability. The waiting times are accepted by the operating staff. The most time consuming tasks are related to database operations. Therefore an optimization of the database tables and structures will be beneficial for improving the procedure times. This can include the usage of materialized database views to reduce calling time requirements, but degrade the up-to-dateness of the data.

#### 14.2.2.2 Automated Model Optimization Results

As shown in the simulation analysis chapter (see Section 11.1.3), the optimization of the dispatching weights has only significant benefit in facility loads above 80% in our case. Optimization benefits are not expected with regards to equal weight combinations because of the low utilization of the facility during the test period.

During the test period, we run optimization tasks with maximal 300 iterations and a maximal duration of one hour. The first test excludes the usage of the generated weights (without write back), therefore the weight fluctuation is estimated to be even higher than in the second case, where the generated weights are used for dispatching. The objective

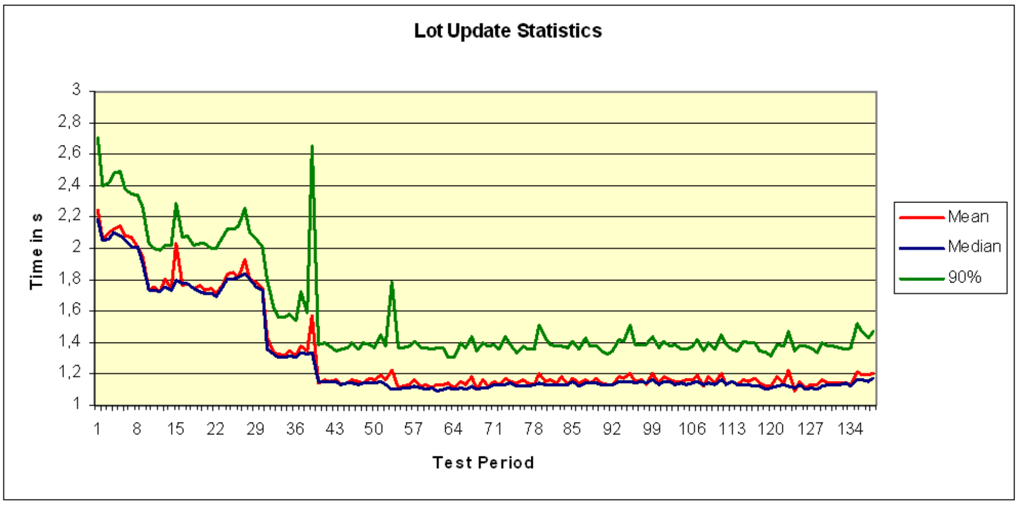


Figure 14.6: Lot update statistics of one month per day

function of the optimization is defined to

$$\begin{aligned} O(XF_{P90}, OTD, WIP, CM_{P90}, CM_{P50}) = \\ 0.1 * N(XF_{P90}) + 0.4 * N(OTD) + 0.1 * N(WIP) + \\ 0.3 * N(CM_{P90}) + 0.1 * N(CM_{P50}) \end{aligned} \quad (14.1)$$

with focus on $OTD$ and $CM$. These two parameters are also the main KPI for evaluating the performance of the facility in our case. In case of the $CM$ the 50% percentile and the 90% percentile are taken into account.

**Optimization without Write Back** For the optimization without write back, the Figures 14.7 and 14.8 illustrate an example time frame of one month of optimization with $a = 1$. Within this time frame, the reference is the equal weight combination. For test cases, two optimizations per day are done. The optimizations are run before any influence of the dispatch system during the silent test.

The average improvement of each optimization run varies between 0,5% to 3% in regards to the objective value result. Improvements are often not significant due to the low facility load. Improvements are only possible in case of huge line imbalances (see iteration 39 to 50) or equipment failures (e.g., iteration 7 or iteration 19) at low facility utilization.

The calculated weight combinations show a high variation over time in contrast to our simulation study (see Section 11.2.2.2). A vast amount of the variation is caused by the test procedure. During the test time frame, the new dispatching policy is evaluated without active usage of the proposed weight combinations. Therefore lot movements are made according the standard FIFO dispatching policy. This changes the situation for

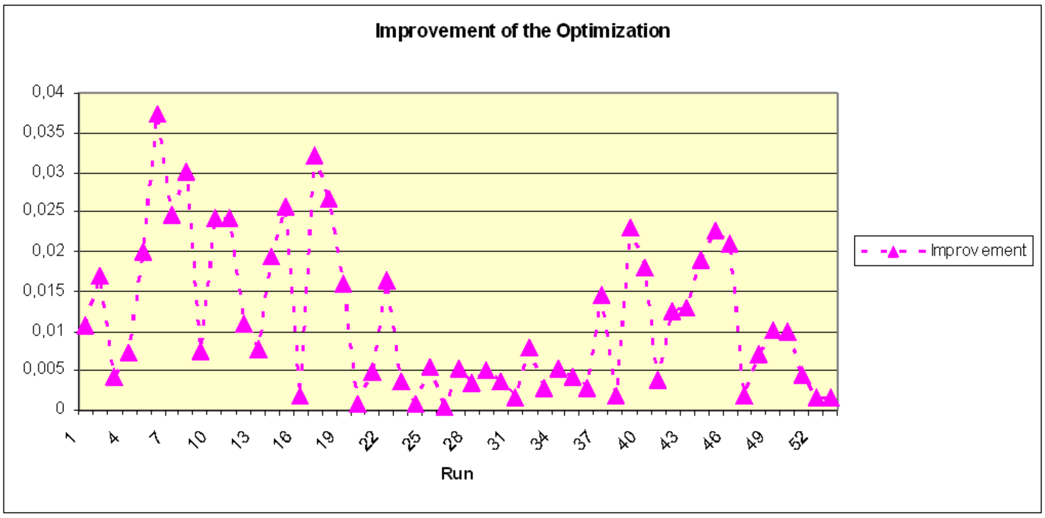


Figure 14.7: Calculated improvement per optimization against equal weight combination

dispatching at each optimization call, where other movements are run than indicated by the combined policy which can be seen in Figure 14.8. The due date oriented dispatching rules ODD and EDD (green colors) always have a huge weight amount. This results from the high importance of the on-time delivery from the objective function. At optimization call number 43 to 51, line balancing problems occur due to equipment down activities. At iteration 35, the SPT rule is applied to reduce the large amount of WIP in front of bottleneck tools.

**Optimization with Write Back** For the write back test, we restricted the minimal improvement of 1.0% of the objective function to the actual reference run to minimize fluctuations of the dispatching weights. Improvements lower than 1.0% during the optimization regarding the objective function are not considered. The optimization is run every day with a 4 day preview time frame. Due to the low load of the line during the test period, the optimization potential is very low. We expect a lower variation than in the test case without write back, indicated by our experiments in Section 11.2.2.2. The adaption factor applied was $a = 0.5$.

Figure 14.9 illustrates the results from two months out of the whole test time period. The variation of the calculated weight combinations is equal to the test without write back. The variation is higher in contrast to our simulation experiments. The main reason for this behavior is the average compliance level of the dispatching list provided by the system. In reality, operating staff sometimes choose other lots than the first ones for processing. This can have process or experience reasons. The results of the dispatch compliance are illustrated in Section 14.2.2.4.

The average improvement of each optimization run is very low. The most cases are below 1.0 % in contrast to the reference run. In general, due to the low facility load, the

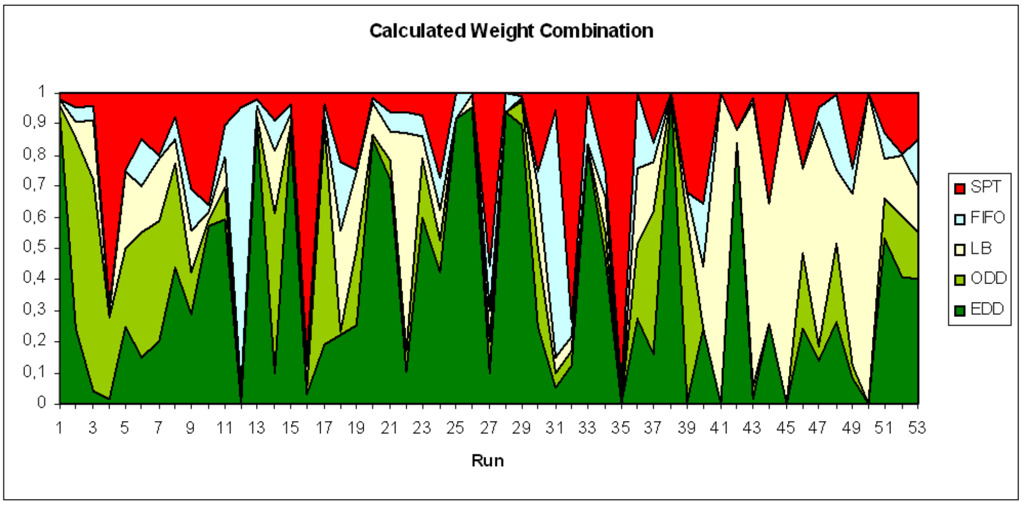


Figure 14.8: Calculated weight combination

optimization procedure has a low impact on the resulting facility KPI.

**Conclusions** In general, the optimization procedure is only reasonable in cases with higher utilization. Unfortunately, during the test time period it is not possible to provide this state in order to calculate more appropriate improvements. Our experiments in Section 11.2.2.2 indicate the optimization potential at higher loads.

#### 14.2.2.3 Automated Model Forecast Example

For illustration of the forecast accuracy of the generated facility model, we exemplify one test case. The example deals with the prediction of weekly facility KPI. The model is generated automatically with a one week preview time frame. For this time frame, sufficient data is available from the facility data warehouse. The model predictions are evaluated against the real facility KPI every week. Therefore we run several forecast scenarios. The following scenario illustrates the main results:

- Every day one forecast is run for the next week at 6:00 am
- Test time period of 14 days
- Replication count of 10
- Automated model generation with data from the facility data warehouse

Figure 14.10 illustrates the deviation of the facility KPI CM 50% percentile, CM 90% percentile and WIP average for this scenario. Each run represents a one week forecast with the overall WIP and CM results. The average deviation of common facility KPI at the forecast varies between -10% to 10%. Most of the prominent outliers are a result of

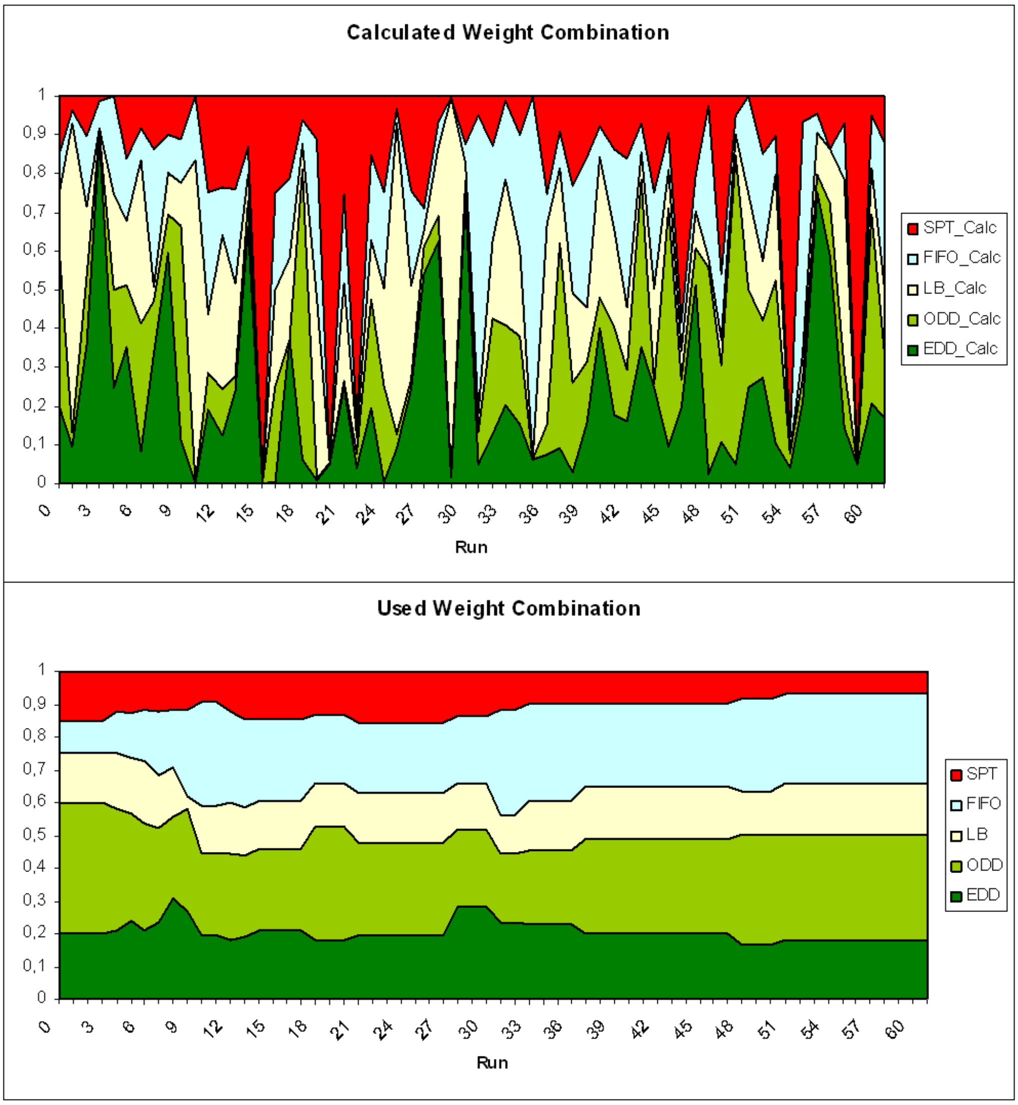


Figure 14.9: Optimization result with write back

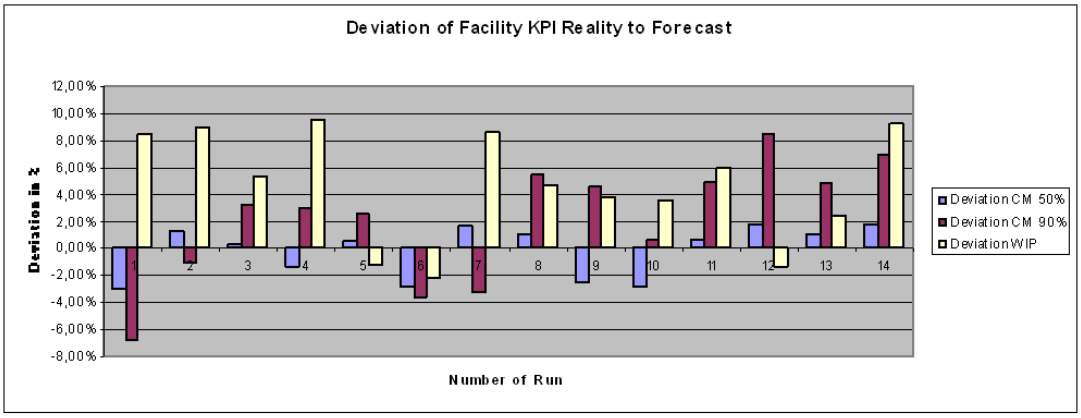


Figure 14.10: Deviation of model prediction to reality

unknown equipment down activities during the forecast period. Therefore data quality improvements will improve the forecast quality. In general the model prediction tends to be faster in various KPI than in reality. This is an result of unplanned activities of the manufacturing process like rework loops or additional metrology operations. The actual break behavior of the operating personal is also not available in any system, therefore the random estimations distort the results. In some cases, lots are prepared for release to the system (with system indication), but actually waiting at the first stage for real start. This can have several process and human related reasons.

At equipment level, the KPI has a larger deviation at some stages where queue length average, flow factor average and queue waiting time average differ from 5% to 20% to reality. At manual equipment without any data available, deviations are higher and reach levels of about 30 to 40% in some cases. The amount of these tools in the whole production is very low, therefore the influence on the overall result is low, too.

In general the low utilization of the real facility reduces the forecast accuracy. One reason is that single lot moves have a greater impact on the overall simulation results here than in a highly loaded facility. Therefore, we will have a higher forecast accuracy at high loaded facilities than in the current context. In general, the level of prediction correctness is sufficient for optimization purposes in our case. Improvements to the model data availability and quality are still in progress to improve the prediction level. With the improved prediction level, further applications like WIP prediction at cluster or equipment level are possible.

#### 14.2.2.4 KPI Influences and Results

During the test period, different KPI at equipment level and global level are analyzed against the ones from previous months of production without the combined dispatch approach. The utilization and the product portfolio change over the year, therefore only limited reasonable comparisons are possible. The operating staff is instructed to adhere to the generated dispatching lists.

**S-Curve of Cycle Time Per Mask Layer** The S-curve of the cycle time per mask layer is one main indicator for the performance of the facility. It represents the cumulative distribution of each cycle time per mask layer value of each finished lot.We use two different normalization approaches for the CTPM S-curve:

- Absolute normalization: using the global maximal and minimal values of the distribution for all lots.
- Relative normalization: using technology dependent maximal and minimal values for each lot.

The first normalization approach shows the absolute facility performance during the different quarters without consideration of natural differences of the CTPM between the different technologies. For detecting the relative improvement, we use the relative normalization which filters out natural variances.

Figures 14.11 and 14.12 illustrate the absolute CTPM distribution of all quarters in 2012. The technology mix changes during Q1 and Q2 as well as the load of the facility decreases. Therefore a comparison is only reasonable between Q3 and Q4. The test period starts in the middle of Q3. Due to the process times ranging from one to three months, impacts are visible in Q4 at the earliest. The overall result shows two facets. Up to the 75% value, improvements are obvious and range between 1% and 5% between Q3 and Q4. The gradient of the curve in the 0% and 20% area is also sharper than in Q2 and Q1 showing a lower deviation. At values above 75%, the performance losses are obvious. These losses are caused by a huge amount of engineering lots and some exotic technologies naturally having a bad performance due to process reasons. The product mix changes in Q4 to a higher product variety with a higher load (~5%) and 15% lower output of the main technology.

The main technologies[3] shows a large increase of the CTPM performance result. The sharper curve in Q4 in comparison to Q3 illustrates a lower deviation. Furthermore, the median as well as the 90% mark are clearly lower compared to Q3. This is partly caused by the new dispatch control algorithm, but also a result of the lower output of the main technology in Q4.

Figure 14.13 illustrates the relative CTPM distribution for the four quarters of 2012. By removing the CTPM variances between the different technologies, the improvements are better accountable than through the use of absolute values. Improvements at the several technologies range between 3% to 10% between Q3 and Q4.

In general, performance increases range between 1% to 5% during the whole test period, depending on the produced technology. The relative S-Curve at Figure 14.13 shows the positive impact of the controller system by removing technology dependent influences on the CTPM. Due to the low load of the facility during the test period, the results are not as clear as expected in case of a higher utilization.

[3] The main technology represents about 50% of the whole product mix during the test period.

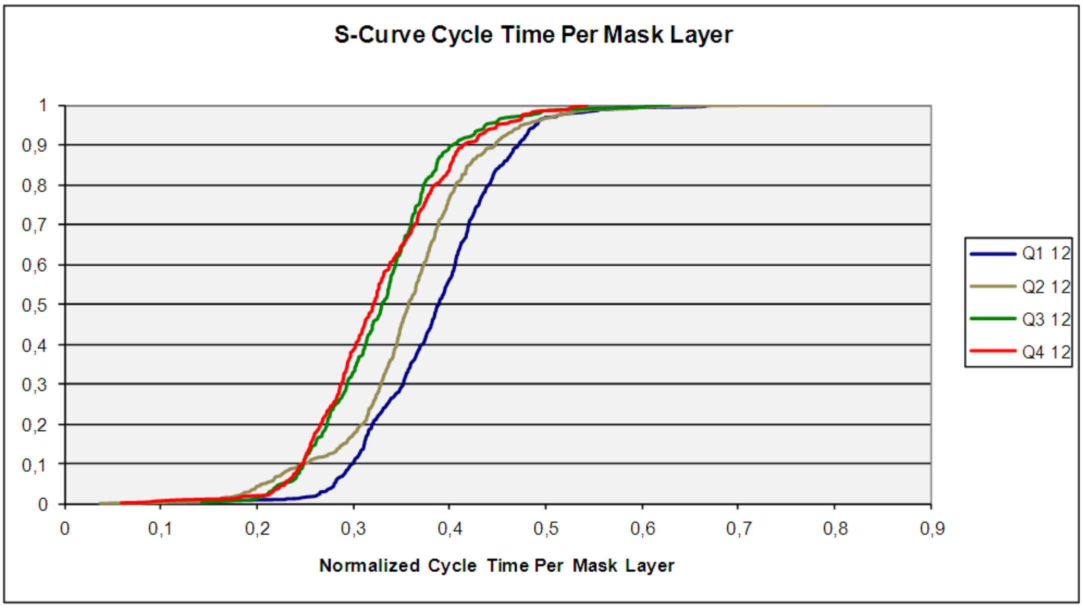


Figure 14.11: Absolute S-Curve of CTPM for all technologies

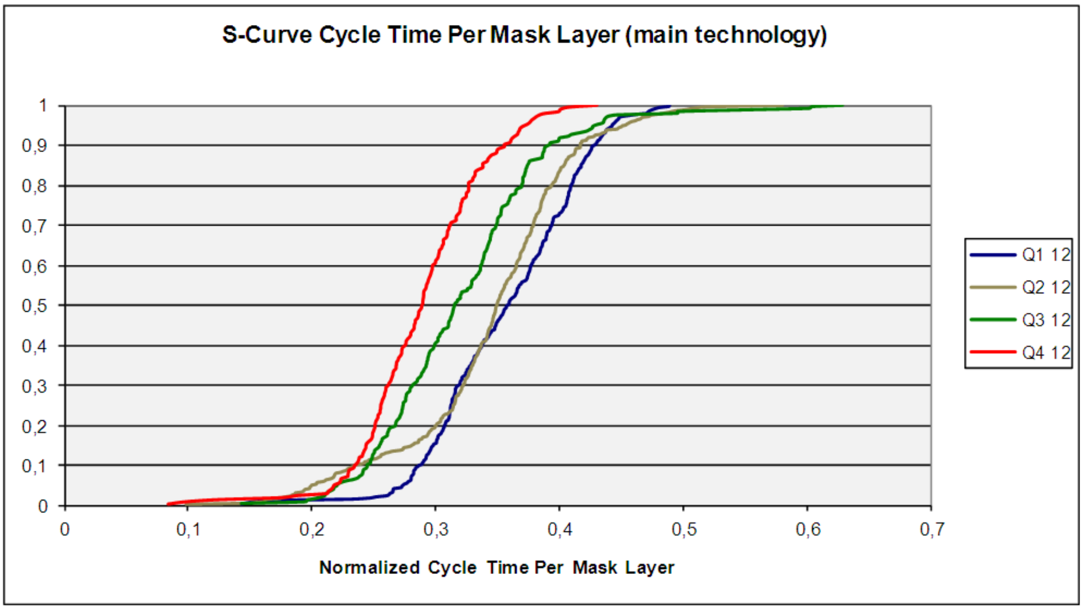


Figure 14.12: Absolute S-Curve of CTPM for the main technology group

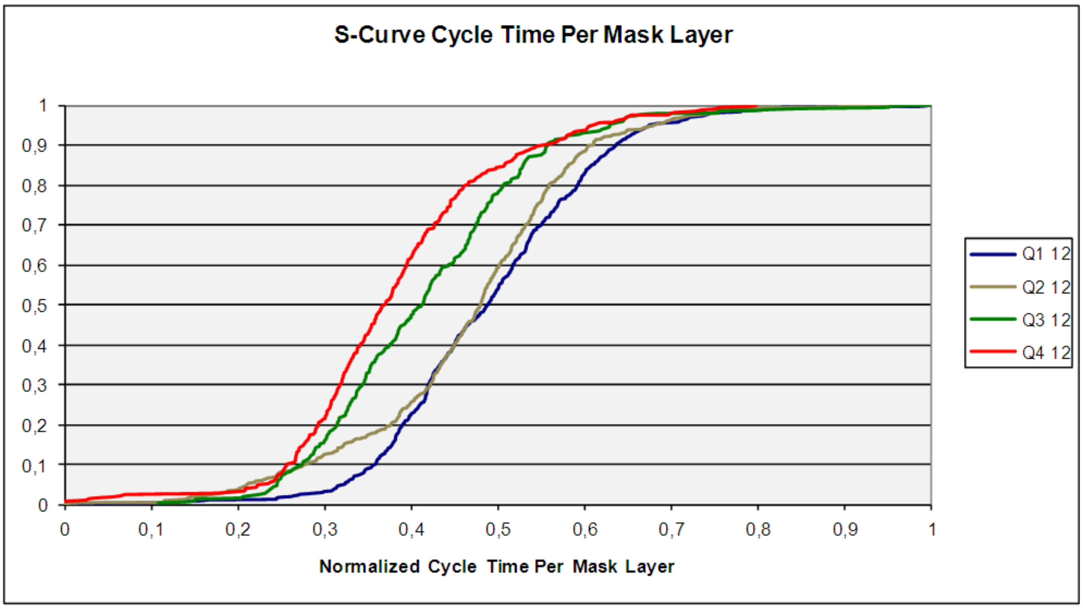


Figure 14.13: Relative S-Curve of CTPM for all technologies

**Absolute KPI Performance Progress** The monthly KPI resulting from each finished lot during the year 2012 is illustrated in Figure 14.14. The results are normalized to the month August (M8). In September (M9), a partly fab shut down happened. Therefore the operator count was lower than in the other months. The degradation of the WIP is obvious, due to the low facility utilization. During this time period, the degradation of the CM 50% and CM 90% values is not as high as the WIP decrease due to several reasons:

- Deactivation of tools in tool groups
- Decrease of operator count in several clusters

These modifications are normal behavior in low utilized facilities to save money and unused capacity. To detect the dependencies between the different KPI, a correlation analysis is done (see Table 14.1). All KPI results have a positive correlation. The correlation between the operator count and the CM values is lower than expected. Due to the huge amount of deactivated tools during the test period (M9 to M12), a higher impact is expected.

The correlation between the WIP and the CM 90% is lower than the correlation between the WIP and the CM 50%. The decrease could be explained by the dispatch system, which is responsible to provide a low CM 90% value at any circumstances. The stable evolution of the CM 50% and CM 90% during the test period with a fluctuating operator count and an increasing amount of deactivated tools and new products indicate the positive impact of the dispatch control system.

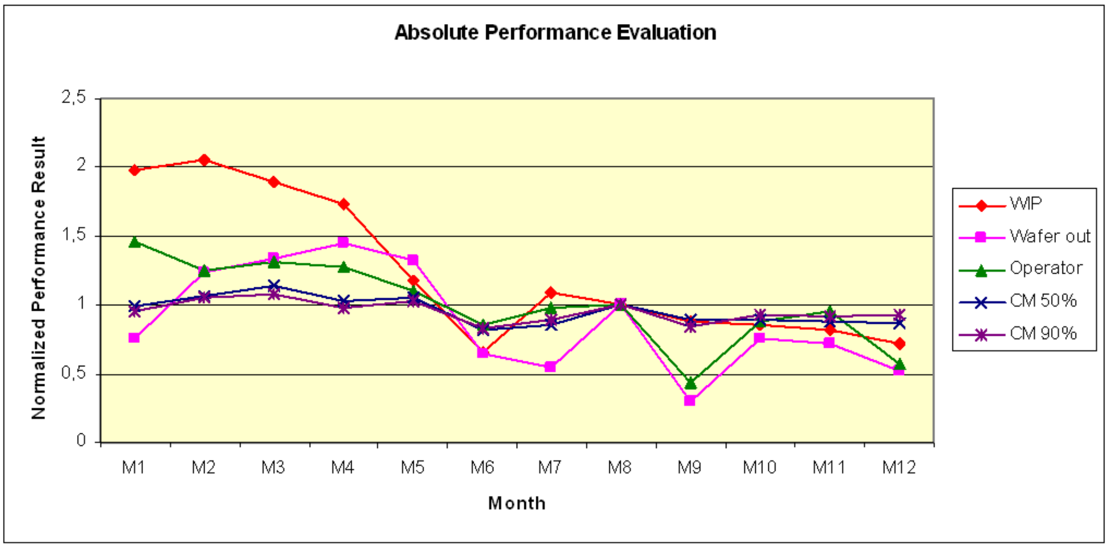


Figure 14.14: Absolute normalized monthly KPI evaluation

| | WIP | Wafer Out | Operator | CM 50% | CM 90% |
|---|---|---|---|---|---|
| WIP | 1 | 0,656 | 0,839 | 0,804 | 0,689 |
| Wafer Out | | 1 | 0,738 | 0,857 | 0,863 |
| Operator | | | 1 | 0,697 | 0,669 |
| CM 50% | | | | 1 | 0,926 |
| CM 90% | | | | | 1 |

Table 14.1: Correlation matrix of monthly KPI results

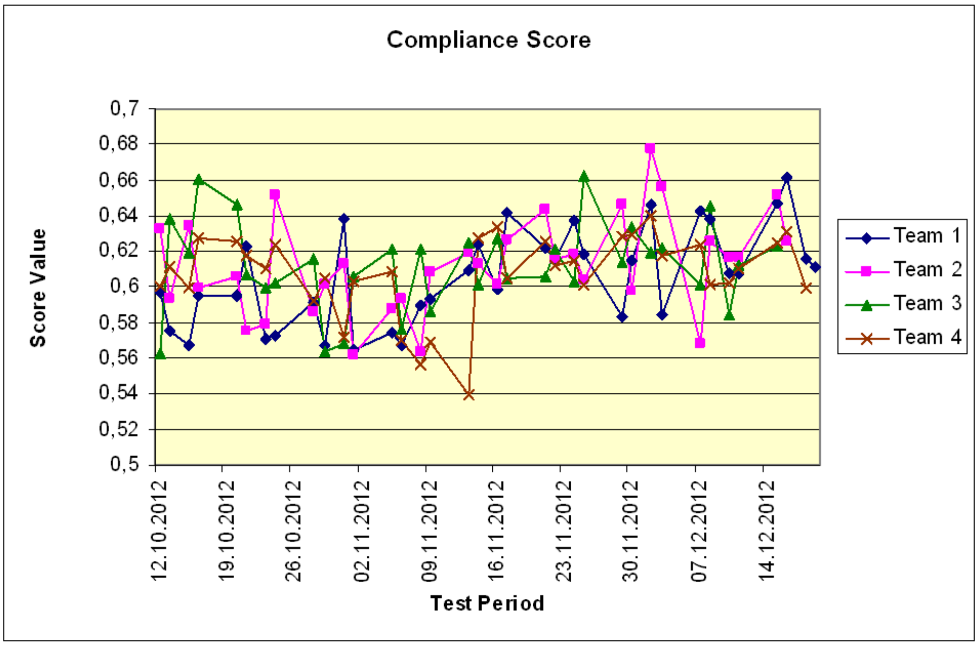


Figure 14.15: Compliance Score for the Dispatch Compliance

**Dispatch Compliance** Besides the system performance, the dispatch list compliance is an important indicator for the acceptance of the system by the operating staff and the fulfillment of all requirements by the dispatching system. Thus, several statistical analyses were carried out to obtain an overview about the compliance development at the facility. We define four compliance parameters:

- Correlation of waiting time to dispatch priority – a negative correlation is expected.
- Absolute dispatching compliance – which describes the number of cases the first lot of the list is taken for processing.
- Compliance score – it describes the relative dispatch compliance by generating a score for each processed lot relative to its list position. The value is between zero and one.
- Average dispatch list index – describes the average dispatching list index.

Each parameter is calculated during each work shift for dispatch lists having more than one lot. Figure 14.15 illustrates the compliance score over several shift periods. Figure 14.16 illustrates the correlation of the waiting time of the lots to the lot priority.

The compliance score is lower than expected, but exhibits a positive trend. After a detailed analysis, several reasons for this behavior can be identified:

- Manual batch transport of lots between clusters: Lots are physically not available at the next operation but waiting for transport.

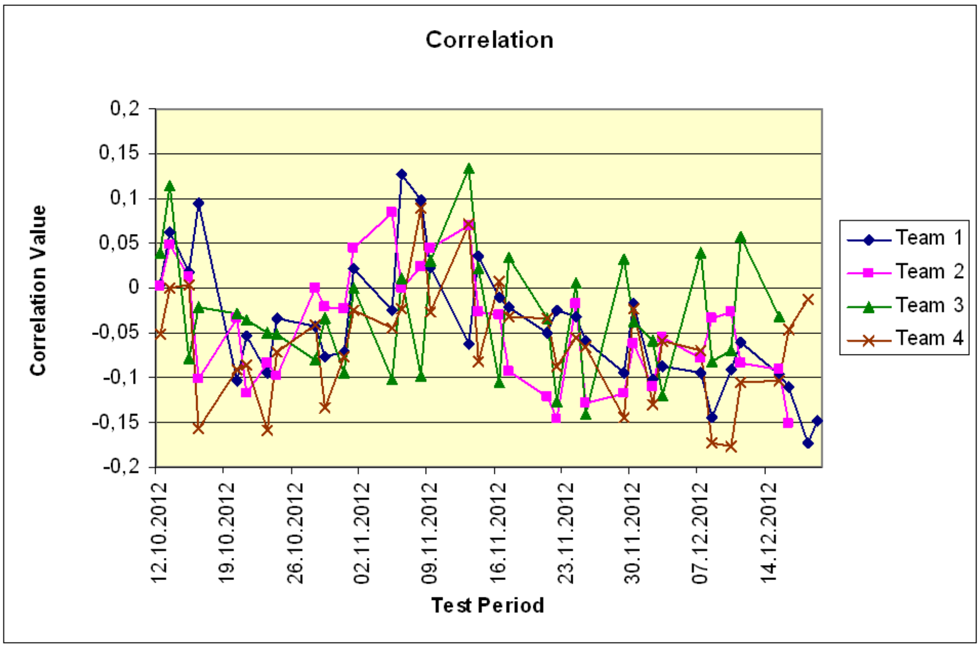


Figure 14.16: Correlation of Waiting Time to Lot Priority

- Work organization: Several small storage buffers in front of the equipment lead to preference of FIFO processing.

- Equipment characteristics like cluster tools requiring special order of lots according to the recipe properties:
  - Unknown cluster tool chamber to recipe matching and number of available load ports (not available in database).
  - Unknown control test state of the equipment required for recipe processing (not available in database).
  - Weak MES administration in some cases (recipe to equipment matching).

Especially the influence of the third point can be reduced by defining further equipment dependent setup states if the data is available for that purpose. The correlation value is higher than expected with a negative trend. The correlation value is biased by the low utilization of the tools and the whole facility resulting in small dispatch lists. Therefore waiting times of low prioritized lots are not significantly higher than of higher prioritized lots.

In general, the targeted compliance score value is about 0.75. The huge amount of different products and the high degree of manual operator interaction does currently not allow higher values due to the aforementioned reasons.

#### 14.2.2.5 Staff Survey

After the end of the test period, a web based staff survey is performed to obtain an overview of the personal opinions from leading staff. The questions are divided into four areas:

1. Evaluation of the current dispatching procedure
2. Opinions about improvements and required changes for the current dispatching system
3. Evaluation of the new dispatching procedure
4. Explanatory notes

The period for the survey was 14 days. The survey was attended by 12 persons. These persons include the shift leader and the dispatching personnel. Operators do not participate, due to reduced working hours and the small number of operators available at each cluster. The following points show the main results of the survey:

1. Current dispatch procedure:
   - 9 persons describe the existing FIFO solution as not sufficient for a modern production control system.
   - 11 persons rate the FIFO policy and the manual priorities as not sufficient for the foundry business with high-mix low-volume characteristics.
   - All persons range the existing policy with a general rating worse than 2 (1. . . perfect, 6. . . worst case), 5 worse than 3.
2. Required changes and needs:
   - All persons rate the need of automated production control systems as very important.
   - 7 persons define the manual priority solution rather as a corrective action than as a planning action.
   - 11 persons see the need for a new production control approach.
3. New dispatch approach:
   - 9 persons evaluate the dispatching lists as more reasonable.
   - Only 6 persons say that the processing follows the lists each time.
   - 8 persons see the new approach as a useful improvement.
   - Only 4 persons accept larger performance decreases.
   - 10 persons range the new policy with a general rating better than 3 (1. . . perfect, 6. . . worst case).

In general, all interviewed persons see a general improvement with regard to the existing solution. In addition to the results of the dispatch compliance, the survey also shows improvement potential at a few local work stations as well as the work organization. A general awareness for the importance of a reasonable production control exists. But performance decreases are not widely accepted. Every person acknowledges a larger improvement in regard to the existing policy definition.

# Part VI

# Conclusions

Expect the worst and your surprises will always be pleasant ones.

*(Louis E. Boone (1941- ))*

# 15 Achievements of this Thesis

## 15.1 Conclusions

In this thesis, we introduce a production control strategy for the demands and requirements of a low-volume high-mix SF with a large amount of operator influence. With the detailed facility model, that is based on real-life production data, several analysis are conducted to find an adaptive and flexible strategy to comply with a multitude of changing demands from management and customers. The detailed facility model creation relies on several simplifications and assumptions about the real production system due to a problematic data quality as well as the absence of important data. Nevertheless, the model verification and validation process documents a good approximation of the real system behavior and allows for valuable analysis.

The combined dispatching approach, which allows the allocation of different dispatching rules with different objectives, is introduced and offers a significant improvement to our reference case, an extended FIFO policy. In our research, we use the dispatching rules EDD for global due date optimization, the ODD rule for local due date optimization, the SPT rule for throughput optimization, a line balancing approach to avoid equipment starvation as well as the mentioned FIFO rule to apply fairness to the decisions. These rules are combined by a linear weight combination for calculation of the final lot priorities. These weights are determined by a detailed facility model under the usage of a genetic optimization algorithm. The different demands and objectives, which change over time, can be transformed into an objective function which is the base for the optimization algorithm. Our analysis shows about 8% improvement of several facility parameters of interest by usage of the approach with optimization. The equal weight case without optimization setting all weights of the $N$ dispatching rules to $w_i = \frac{1}{N}$ shows an average improvement of about 5%. The main requirement of inclusion of the changing demands in the foundry business is met.

Besides the dispatching, the metrology operation skipping is analyzed in order to figure out the influence of a fab-wide sampling procedure. Of course, in a low-volume high-mix facility, sampling is very difficult and our analysis shows a low impact on the resulting facility performance parameters at normal operation. In case of metrology operation bottleneck down, the implementation of the approach is very useful.

Besides the analytical examination, the realization of a prototypical implementation was the main aim of this thesis. Under the usage of common technologies like web services, a JAVA based independent system is build offering the required high flexibility and decoupling for usage in an inhomogeneous IT infrastructure. In this project, the automated model generation task for optimization of the weights for each rule is presented. At this stage, several data quality problems occur which complicate the generation of a valid model. Out of these circumstances a steady data improvement project is commenced to fill in the unavailable data. In general, the data availability and quality is still a massive issue in semiconductor manufacturing.

Finally, the implemented prototype is applied to the real facility environment according to a defined roll-out policy. The policy allows a steady improvement of the current system as well as the introduction of new dispatching rules for the future. The practical results offer improvements at various stages of the production process. Reduced line balancing problems and better on-time delivery compliance can be noticed during our test period of six months even in the low facility utilization case.

## 15.2 Future Areas of Analysis

The policy described in this thesis is a starting point to introduce other dispatching rules and analyze their influence onto the facility performance behavior. The automated model generation and verification process is still a prototypical implementation and further research in this area is required. The handling of low quality data or the absence of data are priority topics for future research. An automated data replacement or completion technique could be useful here to avoid problems. The influence of an automated transport system could also be an element of a further study, currently we work with manual transportation. The application of the approach to other business models such as high-volume logic facilities and their analysis could also be of interest in further studies.

In our case, only dispatching is taken into consideration due to the huge complexity of the low-volume high-mix characteristics of our modeled facility. Scheduling could play an important role for future production control strategies where further improvements are no longer achievable by pure dispatching approaches. Scheduling can predetermine optimal routes for each lot as well as take additional information into account like optimal recipe-equipment allocations in order to improve the quality the predictions.

In our analysis and realization, we only focus on the operational benefits in various performance parameters. A further look into financial aspects brought on by these reductions is also useful to quantify the improvement for business analysts and the management.

In the area of human dominated production areas, a more detailed focus on the human influence on the production process is still pending to quantify and qualify the dependencies. The human influence on production control is often not considered at the required level of detail at common simulation models. A more detailed model of the operator behavior e.g. with fuzzy logic will bring more benefit to the model accuracy. The detailed operator model in combination with a powerful scheduling approach has the potential for further factory performance improvements.

# Part VII

# Appendix

# List of Publications

[GR11] Mike Gißrau and Oliver Rose. A detailed model for a high-mix low-volume asic fab. In S. Jain, R. R. Creasey, J. Himmelspach, K. P. White, and M. Fu, editors, *Proceedings of the 2011 Winter Simulation Conference*, pages 1953–1963, 2011.

[GR12a] Mike Gißrau and Oliver Rose. Development and introduction of a combined dispatching policy at a high-mix low-volume asic facility. In R. Pasupathy O. Rose C. Laroque, J. Himmelspach and A. M. Uhrmacher, editors, *Proceedings of the 2012 Winter Simulation Conference*, 2012.

[GR12b] Mike Gißrau and Oliver Rose. A java based dynamic lot sampling framework for a low-volume high-mix asic facility. In *12th European Advanced Process Control and Manufacturing Conference*, 2012.

[GR12c] Mike Gißrau and Oliver Rose. A model-based combined dispatching approach for a low-volume high-mix asic facility. In *12th European Advanced Process Control and Manufacturing Conference*, 2012.

[GR12d] Mike Gißrau and Oliver Rose. A model-based combined dispatching approach for a low-volume high-mix asic facility, 2012. Invited Talk, Semicon Europe 2012, October 08–10.

[GR13a] Mike Gißrau and Oliver Rose. Evaluation of a model-based combined dispatching approach at a low-volume high-mix asic facility. In *13th European Advanced Process Control and Manufacturing Conference*, 2013.

[GR13b] Mike Gißrau and Oliver Rose. Practical assessment of a combined dispatching policy at a high-mix low-volume asic facility. In A. Tolk R. Hill R. Pasupathy, S.H. Kim and M. E. Kuhl, editors, *Proceedings of the 2013 Winter Simulation Conference*, 2013.

[GR13c] Mike Gißrau and Oliver Rose. Real life evaluation of a model-based combined dispatching approach at a low-volume high-mix asic facility. In Alexander Klaas Wilhelm Dangelmaier, Christoph Laroque, editor, *Simulation in Produktion und Logistik 2013.*, volume 316 of *HNI-Verlagsschriftenreihe, Paderborn.* Heinz Nixdorf Institut, Oct 2013.

# References

[AHG07] R. Barlovic A. Hohlfeld and R.P. Good. A fab-wide apc sampling application. *IEEE Transactions on Semiconductor Manufacturing*, 20:393–399, 2007.

[AJSO08] Cheng-Ching Yu An-Jhih Su, Chen and Babatunde A. Ogunnaike. On the interaction between measurement strategy and control performance in semiconductor manufacturing. *Journal of Process Control*, 18:266–276, 2008.

[All90] Arnold O. Allen. *Probability, statistics, and queueing theory : with computer science applications.* Acad. Press, 2nd edition, 1990.

[ANG08] Marcus Andersson, Amos H.C. Ng, and Henrik Grimm. Simulation optimization for industrial scheduling using hybrid genetic representation. In S. J. Mason, R. R. Hill, L. Mönch, O. Rose, T. Jefferson, and J. W. Fowler, editors, *Proceedings of the 2008 Winter Simulation Conference*, pages 2004–2011, 2008.

[AUH$^+$00] Elif Akcali, Reha Uzsoy, David G. Hiscock, Anne L. Moser, and Timothy J. Teyner. Alternative loading and dispatching policies for furnace operations in semiconductor manufacturing: A comparison by simulation. In J. A. Joines, R. R. Barton, K. Kang, and P. A. Fishwick, editors, *Proceedings of the Winter Simulation Conference*, pages 1428–1435, 2000.

[Bal97] Osman Balci. Verification, validation and accreditation of simulation models. In S. Andradottir, K. J. Healy, D. H. Withers, and B. L. Nelson, editors, *Proceedings of the Winter Simulation Conference*, pages 135–141, 1997.

[Ban01] Jerry Banks. *Discrete event system simulation.* Prentice Hall, 3. ed. edition, 2001.

[BFS87] Paul Bratley, Bennet L. Fox, and Linus E. Schrage. *A guide to simulation.* Springer, 2. ed. edition, 1987.

[BHB00] Nikhil Bansal and Mor Harchol-Balter. Analysis of srpt scheduling: Investigating unfairness. Technical report, School of Computer Science Carnegie Mellon University, 2000.

[Bru07] Peter Bruckner. *Scheduling Algorithms.* Springer, 5. ed. edition, 2007.

[CH00] Yon-Chun Chou and I.-Hsuan Hong. A methodology for product mix planning in semiconductor foundry manufacturing. *IEEE Transactions on Semiconductor Manufacturing*, 13:278–285, 2000.

[CJ09] Jaewoo Chung and Jaejin Jang. A wip balancing procedure for throughput maximization in semiconductor fabrication. *IEEE Transactions on Semiconductor Manufacturing*, 22:381–390, 2009.

[CZ04] Edwin K. P. Chong and Stanislaw H. Zak. *An Introduction to Optimization.* John Wiley & Sons, Inc., 2. ed. edition, 2004.

[DCFS01] Russ M. Dabbas, Hung-Nan Chen, John W. Fowler, and Dan Shunk. A combined dispatching criteria approach to scheduling semiconductor manufacturing systems. *Computers and Industrial Engineering*, 39:307–324, 2001.

[DF03] Russ M. Dabbas and John W. Fowler. A new scheduling approach using combined dispatching criteria in wafer fabs. In *IEEE Transactions on Semiconductor Manufacturing*, pages 501–509, 2003.

[Doe07] Robert Doering. *Handbook of Semiconductor Manufacturing Technology.* CRC Press, 2. ed. edition, 2007.

[ELE11] 2011 major ic foundries, 2011. accessed on www.semimd.com at 11.12.2012.

[EPA00] Hoda ElMaraghy, Vishvas Patel, and Imed Ben Abdallah. Scheduling of manufacturing systems under dual-resource constraints using genetic algorithms. *Journal of Manufacturing Systems*, 19:186–201, 2000.

[ES98] Erdal Erel and Subash C. Sarin. A survey of the assembly line balancing procedures. *Production Planning and Control*, 9:414–434, 1998.

[FR95] J. Fowler and J. Robinson. Measurement and improvement of manufacturing capacities (mimac): Final report. Technical Report 95062861A-TR, SEMATECH„ Austin, 1995.

[FRC93] Hsiao-Lan Fang, Peter Ross, and Dave Corne. A promising genetic algorithm approach to job-shop scheduling, rescheduling, and open-shop scheduling problems. In S. Forrest, editor, *Proceedings of the International Conference on Genetic Algorithms*, pages 375–382, 1993.

[Fu02] Michael C. Fu. Optimization for simulation: Theory vs. practice. *Informs Journal on Computing*, 14(3):192–215, 2002.

[GELK79] R.L. Graham, J.K. Lenstra E.L. Lawler, and Rinnooy Kan. Optimizaion and approximation in deterministic sequencing and scheduling: A survey. *Annals of Discrete Mathematics*, 5:287–326, 1979.

[GLM00] Fred Glover, Manuel Laguna, and Rafael Marti. Scatter search. Technical report, University of Colorado and Universidad de Valencia, 2000.

[Goo93] Paul Goodman. *Practical Implementation of Software Metrics.* McGraw-Hill, 1993.

[Hau89] R. Haupt. A survey of priority rule-based scheduling. *OR Spektrum*, 11:3–16, 1989.

[Hel00] Keld Helsgraun. Discrete event simulation in java. Technical report, Roskilde University, 2000.

[HM05] Harald Hungenberg and Jürgen Meffert. *Handbuch strategisches Management.* Gabler, 2005.

[HS01] Wallace J. Hopp and Mark L. Spearman. *Factory physics : foundations of manufacturing management.* Irwin McGraw-Hill, 2. ed. edition, 2001.

[HT03] Nhu Binh Ho and Joc Cing Tay. Evolving dispatching rules for solving the flexible job-shop problem. Technical report, Nanyang Technological University, 2003.

[JSC02] II John S. Carson. Model verification and validation. In E. Yücesan, C.-H. Chen, J. L. Snowdon, and J. M. Charnes, editors, *Proceedings of the Winter Simulation Conference*, pages 52–58, 2002.

[Kel03] Inc. Wright Williams & Kelly. *User Manual Factory Explorer 2.8*, 2003.

[Kle07] Jürgen Kletti. *MES - Manufacturing Execution System.* Springer, 2007.

[Koz92] John R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection.* MIT Press, 1992.

[Lag11] Manuel Laguna. *OptQuest - Optimization of Complex Systems.* Opttek Systems, INC., 2011.

[LK00] Averill M. Law and W. David Kelton. *Simulation modeling and analysis.* McGraw-Hill, 3. ed. edition, 2000.

[Lo07] Mei-Chen Lo. Wafer foundrys business strategies on mass production or mass customization. In *Proceedings of the IMCM Conference*, 2007.

[LRK94] Steve C. H. Lu, Deepa Ramaswamy, and P. R. Kumar. Efficient scheduling policies to reduce mean and variance of cycle-time in semiconductor manufacturing plants. *IEEE Transactions Semiconductor Manufacturing*, 7:374–388, 1994.

[LTC01] Loo Hay Lee, Loon Ching Tang, and Soon Chee Chan. Dispatching heuristic for wafer fabrication. In B. A. Peters, J. S. Smith, D. J. Medeiros, and M. W. Rohrer, editors, *Proceedings of the Winter Simulation Conference*, pages 1215–1219, 2001.

[MAN11] Simulation modeling with anylogic: Agent based, discrete event and system dynamics methods, March 2011. accessed on www.anylogic.com at 21.3.2011.

[Mar94] Robert Marting. Oo design quality metrics. Technical report, OMA London, 1994.

[MFDP$^+$11] Lars Mönch, John W. Fowler, Stephane Dauzere-Peres, Scott J. Mason, and Oliver Rose. A survey of problems, solution techniques, and future challenges in scheduling semiconductor manufacturing operations. *J Sched*, 14:583–599, January 2011.

[MH03] Lars Mönch and Ilka Habenicht. Simulation-based assessment of batching heuristics in semiconductor manufacturing. In S. Chick, P. J. Sanchez, D. Ferrin, and D. J. Morrice, editors, *Proceedings of the Winter Simulation Conference*, pages 1338–1345, 2003.

[MS99] Manfred Mittler and Alexander K. Schoemig. Comparision of dispatching rules for semiconductor manufacturing using large facility models. In P. A. Farrington, H. B. Nembhard, D. T. Sturrock, and G. W. Evans, editors, *Proceedings of the Winter Simulation Conference*, pages 709–713, 1999.

[MS06] Gary S. May and Costas J. Spanos. *Fundamentals of Semiconductor Manufacturing and Process Control.* John Wiley & Sons, Inc., 2006.

[NMRVDP$^+$13] Justin Nduhura-Munga, Gloria Rodriguez-Verjan, Stephane Dauzere-Peres, Philippe Vialletelle Claude Yugma, and Jacques Pinaton. A literature review on sampling techniques in semiconductor manufacturing. *IEEE Transactions on Semiconductor Manufacturing*, 26(2):188–195, May 2013.

[NR08] Daniel Noack and Oliver Rose. A simulation based optimization algorithm for slack reduction and workforce scheduling. In S. J. Mason, R. R. Hill, L. Mönch, O. Rose, T. Jefferson, and J. W. Fowler, editors, *Proceedings of the Winter Simulation Conference*, pages 1989–1994, 2008.

[OP09] Djamila Ouelhadj and Sanja Petrovic. A survey of dynamic scheduling in manufacturing systems. *Journal of Scheduling*, 12:417–431, 2009.

[OR10] L. William Oberkampf and Christopher J. Roy. *Verification and Validation in Scientific Computing.* Cambridge University Press, 2010.

[PBF$^+$08] Michele E. Pfund, Hari Balasubramanian, John W. Fowler, Scott J. Mason, and Oliver Rose. A multi-criteria approach for scheduling semiconductor wafer fabrication facilities. *Journal of Scheduling*, 11:29–47, 2008.

[PC01] Raffaele Pesenti and Lorenzo Castelli. Scheduling in a realistic environment using autonomous agents: A simulation study. Technical report, Universita degli Studi di Palermo, Universita degli Studi di Trieste, 2001.

[Pin02] Michael Pinedo. *Scheduling : theory, algorithms, and systems.* Prentice Hall, 2. ed. edition, 2002.

[PW77] S. S. Panwalker and Iskander Walfik. A survey of scheduling rules. *Operations Research*, 25(1):45–61, 1977.

[Pyz97] Tom Pyzdek. Motorola's six sigma program, 1997. accessed on http://www.qualitydigest.com/magazine/1997/dec/article/motorolas-six-sigma-program.html at 13.06.2013.

[RDS00] Oliver Rose, Matthias Dümmler, and Alexander Schömig. On the validity of approximation formulae for machine downtimes. Technical report, University of Würzburg, 2000.

[RM07] Dirk Reichelt and Lars Mönch. Multiobjective scheduling of jobs with incompatible families on parallel batch machines. Technical report, Technical University of Ilmenau, 2007.

[Rob97] Stewart Robinson. Simulation model verification and validation: Increasing the users confidence. In S. Andradottir, K. J. Healy, D. H. Withers, and B. L. Nelson, editors, *Proceedings of the Winter Simulation Conference*, pages 53–59, 1997.

[Ros01] Oliver Rose. The shortest processing time first dispatch rule and some variants in semiconductor manufacturing. In B. A. Peters, J. S. Smith, D. J. Medeiros, and M. W. Rohrer, editors, *Proceedings of the Winter Simulation Conference*, pages 1220–1224, 2001.

[Ros02] Oliver Rose. Some issues of the critical ratio dispatch rule in semiconductor manufacturing. In E. Yücesan, C.-H. Chen, J. L. Snowdon, and J. M. Charnes, editors, *Proceedings of the Winter Simulation Conference*, pages 1401–1405, 2002.

[Ros03] Oliver Rose. Accelerating products under due date oriented dispatching rules in semiconductor manufacturing. In S. Chick, P. J. Sanchez, D. Ferrin, and D. J. Morrice, editors, *Proceedings of the Winter Simulation Conference*, pages 1346 – 1350, 2003.

[Ros04] Oliver Rose. Modeling tool failures in semiconductor fab simulation. In *Proceedings of the Winter Simulation Conference*, 2004.

[Ros07] Oliver Rose. Improving the accuracy of simple simulation models for complex production systems. In *Proceedings of the INFORMS Simulation Society Research Workshop*, 2007.

[Ros08a] Oliver Rose. Introduction into simulation, 2008. Lecture.

[Ros08b] Oliver Rose. Production management in semiconductor manufacturing, 2008. Lecture.

[Ros08c] Manuel D. Rossetti. Java simulation library (jsl): an open-source object-oriented library for discrete-event simulation in java. *International Journal of Simulation and Process Modelling*, 4(1):69–87, October 2008.

[Ros09] Oliver Rose. Scheduling problems and solutions, 2009. Lecture.

[RR99] Simone Riedmiller and Martin Riedmiller. A neural reinforcement learning approach to learn local dispatching policies in production scheduling. Technical report, University of Karlsruhe, 1999.

[Sad91] Norman Sadeh. *Look-Ahead Techniques for Micro-Opportunistic Job Shop Scheduling*. PhD thesis, Carnegie Mellon University, 1991.

[Sar05] Robert G. Sargent. Verification and validation of simulation models. In M. E. Kuhl, N. M. Steiger, F. B. Armstrong, and J. A. Joines, editors, *Proceedings of the Winter Simulation Conference*, pages 130–143, 2005.

[SB10] John A. Sokolowski and Catherine M. Banks. *Modeling and Simulation Fundamentals : Theoretical Underpinnings and Practical Domains.* John Wiley & Sons, Inc., 2010.

[Sch00] Günter Schmidt. Scheduling with limited machine availability. *European Journal of Operational Research*, 121:1–15, 2000.

[Siv99] Appa Iyer Sivakumar. Optimization of cycle time and utilization in semiconductor test manufacturing using simulation based, on-line, near-real-time scheduling system. In P. A. Farrington, H. B. Nembhard andD. T. Sturrock, and G. W. Evans, editors, *Proceedings of the Winter Simulation Conference*, pages 727–735, 1999.

[Ski84] Wickham Skinner. Operations technology: Blind spot in strategic management. *Interfaces*, 14:116–125, 1984.

[Spa05] James C. Spall. *Introduction to Stochastic Search and Optimization : Estimation, Simulation, and Control.* John Wiley & Sons, Inc., 2005.

[Stu09] Kilian Stubbe. *Development and Simulation Assessment of Semiconductor Production System Enhancements for Fast Cycle Times.* PhD thesis, Dresden University of Technology, 2009.

[SVW11] Subhash C. Sarina, Amrusha Varadarajana, and Lixin Wanga. A survey of dispatching rules for operational control in wafer fabrication. *Production Planning & Control: The Management of Operations*, 22:4–24, 2011.

[Tec10] XJ Technologies. *User Manual Anylogic 6.* XJ Technologies, 2010. Online Resource accessed on http://www.anylogic.com at December 2011.

[TFL03] Chih-Hung Tsai, Yun-Min Feng, and Rong-Kwei Li. A hybrid dispatching rules in wafer fabrication factories. *International Journal of The Computer, The Internet and Management*, 11:64–72, 2003.

[TH08] Joc Cing Tay and Nhu Binh Ho. Evolving dispatching rules using genetic programming for solving multi-objective flexible job-shop problems. *Computers & Industrial Engineering*, 54:453–473, 2008.

[vZ04] Peter van Zant. *Microchip fabrication : a practical guide to semiconductor processing.* McGraw-Hill, 5. ed. edition, 2004.

[WB89] Richard B. Whitner and Osman Balci. Guidelines for selecting and using simulation verification techniques. Technical report, Virginia Polytechnic Institue and State University, 1989.

[Wei88] Lawrence M. Wein. Scheduling semiconductor wafer fabrication. *IEEE Transactions on Semiconductor Manufacturing*, 1:115–130, 1988.

[Win99] Thomas Winter. *Online and Real-Time Dispatching Problems.* PhD thesis, Technical University of Braunschweig, 1999.

[WQW06] Zuntong Wang, Fei Qiao, and Qidi Wu. Scheduling semiconductor wafer fabrication with optimization of multiple objectives. In *Proceeding of the IEEE International Conference on Automation Science and Engineering*, 253-258, 2006.

[ZR09] Zhugen Zhou and Oliver Rose. A bottleneck detection and dynamic dispatching strategy for semiconductor wafer fabrication facilities. In M. D. Rossetti, R. R. Hill, B. Johansson, A. Dunkin, and R. G. Ingalls, editors, *Proceedings of the Winter Simulation Conference*, pages 1646–1656, 2009.

[ZS09] Dekong Zeng and Costas J. Spanos. Virtual metrology modeling for plasma etch operations. *IEEE Transactions Semiconductor Manufacturing*, 22(4):419–431, 2009.