Modeling and Performance Evaluation of Production Lines Using the Modeling Language MOSEL

G. Bolch, S. Greiner
IMMD IV – University Erlangen-Nuremberg
Martensstraße 1, D – 91058 Erlangen, Germany
E-mail: {bolch, sngreine}@informatik.uni-erlangen.de
Phone: +49-9131-85-7903, Fax: +49-9131-39388

Abstract

Modeling and performance evaluation of current production systems gains more and more importance because of growing system complexity and production costs. Therefore tools and methods are needed that allow a simple modeling and performance evaluation of systems. The goal of this paper is to give a short introduction into the specification of production systems using the model description language MOSEL (**MO**deling **S**pecification and **E**valuation **L**anguage) [BoHe95] and the analysis tool MOSES (**MO**deling **S**pecification and **E**valuation **S**ystem) [BGJZ94, BGR95] which uses MOSEL as input. Therefore the characteristics of the components of production systems (like batch processing or unreliable machines) are introduced. For each of these characteristics a MOSEL specification is given and explained. On the example of a wafer production system [Rand86] it is shown that using our specification language also realistic production systems can be specified and analyzed.

Keywords
Modeling, performance evaluation, model description language, production lines, Markovian analysis

1 Introduction and Motivation

Performance evaluation of current production systems gains more and more importance because of growing costs and system complexity. Usually it is very difficult or even impossible to do performance evaluation by system measurement. What we need is a system model. For the analysis of such models basically three methods are available:

- Simulation.
- Markov Analysis.
- Analytical methods based on queueing network models.

If we use discrete time simulation as analysis technique then the system can be modeled with a very high degree of accuracy but many resources are needed for the analysis of the system. If we decide to use analytical methods for system analysis [Bolc89] then only simple systems can be modeled because of the restrictions of these methods. It is for example not possible to model a production system with finite buffer, batch processing and unreliable machines using queueing network based analytical methods. But exactly these characteristics are of interest in the field of production systems. Therefore we use Markov analysis techniques for modeling and performance evaluation of production systems since it is faster than simulation and more powerful than analytical methods. To apply Markovian analysis the underlying system of linear equations must be determined and solved. The system of equations describes the behavior of the corresponding system. By solving this system of equations we get the so called state probabilities which are used to determine all interesting performance measures like utilization, throughput or system time. The generation and solution of the Markov system of equations by hand is already for very simple systems difficult and for complex systems nearly impossible. This can be done much easier with the system specification language MOSEL and the Markov analyzer MOSES. Since queueing models offer a very compact and clear possibility to describe a system we use them for system description (not analysis !). These models are specified using the model description language **MOSEL** and analyzed using **MOSES**. MOSES takes the system specification as an input and generates the underlying system of equations automatically. To solve this system of equations MOSES offers six different methods. It is also possible to describe the (production) system using Petri nets but they have the disadvantage of getting very complex (especially for large systems).

This paper is organized as follows. At first the different characteristics of production systems (like batch processing or unreliable machines) are explained and it is shown how to specify these characteristics using MOSEL. Then the photolitographic process of the wafer production system is specified and analyzed. All parameters used in this paper can be found in the following table:

Parameter	Explanation
K	Maximum number of jobs
num	Actual number of jobs
a	Minimum batch size
b	Maximum batch size
mttr	Mean time to repair
mtbf	Mean time between failure
ServiceTime_1	Service time of machine 1
ServiceTime_2	Service time of machine 2
ServiceRate_1	Service rate of machine 1
ServiceRate_2	Service rate of machine 2
Arrival	Arrival rate of jobs at the system

It should be noted that the service rate at a machine is just given by the reciprocal value of the service time.

2 The Model Description Language MOSEL

The model description language MOSEL consists of a series of constructs which can be categorized as follows: definition part, vector description part, rules part, results part, macro definitions and comments. In the following the semantics of each part shall be explained briefly.

Declaration part: In this optional part constants and variables can be declared. For the definition of constants we use the **#define** and **enum** constructs. The variable types **int**, **double** and **prob** (probability) are available.

Vector description part: In this part the components of the state vector are declared. For each component its name and capacity has to be given.

Rules part: In this part the possible state transitions in the system are described. Each rule consists of a global part and optionally of a local part in brackets. The keywords **FROM, TO, W, P, IF** are used to describe the state transitions.

Results part: In the results part the performance measures of the system are specified. Basic performance measures like blocking probability or utilization can be computed using these state probabilities which are automatically computed by MOSES.

Macro definitions: MOSEL allows the definition of text macros by using the keyword **#string** *macroname macrotext.* Whenever *macroname* is found in the specification it is substituted by *macrotext.*

3 Application of MOSEL to Simple Production Systems with two Machines

We want to show now how to specify unreliable machines, systems with finite buffers and batch processing using MOSEL. Our examination is based on the model shown in Fig. 1.

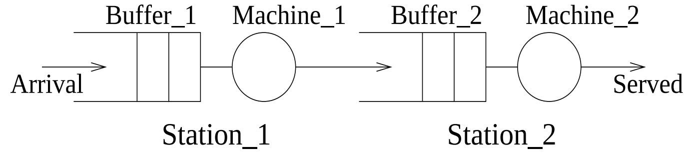

Figure 1: Basic model

As we can see each Station consists of a *Buffer* and a *Machine*. In the now following specification it is sometimes necessary to model the behavior of a *Buffer* and a *Machine* separately (*Machine_i*, *Buffer_i*, i=1,2) while sometimes it is also enough if we model the behavior of a *Station* as a whole. We model all systems as open systems which means that jobs arrive with rate *Arrival* and leave the system as soon as they are served. All service times used in this article are exponentially distributed and are measured in *units* (msec, sec, min, ...).

3.1 Specification of a Simple Production Line

Let us consider the simple production line shown in Fig. 1. To give the reader a short introduction into the model description language MOSEL we specify the whole system and explain each line. In this sense the reader gets more familiar how to use the MOSEL constructs.

```
/* Declaration part */

1  enum machine_state{idle, busy}                         ;

/* Vector description part */

2  NODE Buffer_1 [K]             = 0                      ;
3  NODE Machine_1[machine_state] = idle                   ;
4  NODE Station_2[K]             = 0                      ;
5  NODE num[K]                   = 0                      ;

/* Rules part */

6  FROME TO Buffer_1, num W Arrival                       ;

7  FROM Buffer_1 TO Machine_1[busy] IF Machine_1[idle]   ;
8  FROM Machine_1[busy] TO Machine_1[idle], Station_2
        W ServiceRate_1                                   ;

9  FROM Station_2, num TOE W ServiceRate_2                ;

/* Results part */

10 RESULT>> IF(Station_2 > 0) rho_2 += PROB               ;
11 RESULT>> WIP = MEAN num                                ;
12 RESULT>> T = WIP / Arrival                             ;
```

Figure 2: MOSEL code for a simple production system

These lines can be interpreted as follows:

Line 1: The state of a machine is defined to be *idle* or *busy*.

Line 2-5: The elements of the production system are defined. Our simple system consists of *Buffer_1*, *Machine_1* (which form together *Station_1*), *Station_2* and the help state *num* which is used to control the number of jobs in the system.

Line 6: Jobs arrive from external at *Buffer_1* with the rate *Arrival*. At the same time *num* is increased by one.

Line 7: If there is at least one job in *Buffer_1* and *Machine_1* is in the state *idle* then the machine is set to *busy*.

Line 8: If $Machine\_1$ is busy then the job is served with rate $ServiceRate\_1$ and moved to $Station\_2$. In the meantime $Machine\_1$ is set to $idle$ again.

Line 9: If there is at least one job at $Station\_2$ then a job is served and leaves the system. At the same time num is reduced by one.

Line 10: Specification of the performance measure rho_2 (the utilization of $Station\_2$). We add all probabilities (+=PROB) for the states where at least one job is at $Station\_2$.

Line 11: WIP (work in process) is given as the mean value (MEAN) of the number of jobs in the system (num).

Line 12: The system time of a job in the system is given by Little's law T = WIP / Arrival.

As we can see it is very straightforward to specify this example system. In the following we want to consider more complex examples.

3.2 Systems with Several Machines at a Station

In this section we want to consider a system where $Station\_1$ consists of several machines as shown in Fig. 3. $Station\_2$ again consists only of one machine.

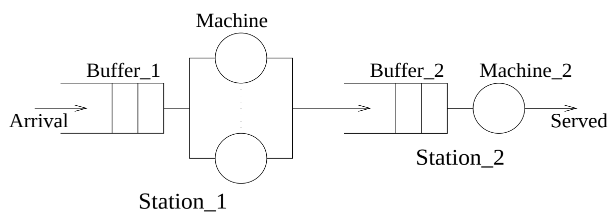

Figure 3: System with several machines at a station

In the specification shown in Fig. 4, $Station\_1$ consists of 4 machines and $Station\_2$ consists only of one machine.

```
1  #define m 4                                          ;

2  NODE Buffer_1[K]                                     ;
3  NODE Machine[m]                                      ;
4  NODE Station_2[K]                                    ;
5  NODE num[K]                                          ;

6  FROME TO Buffer_1, num W Arrival                     ;

7  FROM Buffer_1 TO Machine                             ;
8  <1..m> FROM Machine TO Station_2 W #*ServiceRate_1
               IF Machine == #                          ;

9  FROM Station_2,num TOE W ServiceRate_2               ;

10 RESULT>> IF(Station_2 > 0) rho_2 += PROB             ;
```

Figure 4: MOSEL core for a station with several machines

The core of this specification can be interpreted as follows:

Line 6: Jobs arrive at $Buffer\_1$ with the rate $Arrival$.

Line 7: If there are jobs in the buffer and at least one machine is available then a job is removed from $Buffer\_1$ and moved to a machine.

Line 8: Depending on the number of active machines the movement of a job to *Station_2* takes place with rate # ∗ *ServiceRate_1* where # can either be 1, 2, 3 or 4.

3.3 Batch Processing

Batch processing is an integral part of many production systems. In this paper we want to consider the full batch processing mode. In this case the machine starts to work only if the batch is full (assuming a batch size *b*). As soon as the batch is full all *b* jobs are processed in parallel. The core of the MOSEL specification can be seen in Fig. 5.

```
1  FROME TO Buffer_1, num W Arrival                                        ;

2  FROM Machine_1[idle] TO Machine_1[busy] IF Buffer_1 >= b                ;
3  FROM Machine_1[busy], Buffer_1(b) TO Machine_1[idle], Station_2(b)
   W ServiceRate_1                                                         ;

4  FROM Station_2, num TOE W ServiceRate_2                                 ;
```

Figure 5: MOSEL core for the full batch system

These lines can be interpreted as follows:

Line 1: Jobs arrive at *Buffer_1* with rate *Arrival*. As soon as a job arrives *num* is increased by 1.

Line 2: If *Machine_1* is *idle* and there are at least *b* jobs in *Buffer_1* then *Machine_1* is set *busy*.

Line 3: *b* jobs are moved from *Buffer_1* to *Station_2* with rate *ServiceRate_1*.

Line 4: If there is at least one job at *station_2* it is served and leaves the system. At the same time *num* is decreased by 1.

3.4 Systems with Unreliable Machines

3.4.1 Description of Systems with Unreliable Machines

Up to now we have assumed that no machine fails but in general for production systems we also have to consider machine failures. This means that as soon as a machine fails no job is served any more until the machine is up again. But while the machine is being repaired jobs can still arrive at the buffer. The job just served when a machine goes down has to be served again. In order to indicate that a machine is up or down we use the two states *up* and *down*.

3.4.2 Specification of a System with Unreliable Machines

The core of the MOSEL specification of a system with unreliable machines can be seen in Fig. 6. *Station_1* (consisting of *Buffer_1* and *Machine_1*) is assumed to be an unreliable station while *Station_2* is assumed to be reliable.

These lines can be interpreted as follows:

Line 1: Specification of the arrival of jobs at the system.

Line 2: To model the failure of machines we use the two states *up* and *down*. After an exponentially distributed time *mtbf* (mean time between failure) the *up* state is set to *down*.

Line 3: If the machine is in the state *down* then it is repaired in an exponentially distributed time *mttr* (mean time to repair) and goes back to the *up* state again.

```
1  FROME TO Buffer_1, num W Arrival                                ;

2  FROM State_Machine_1[up] TO State_Machine_1[down] W 1/mtbf      ;
3  FROM State_Machine_1[down] TO State_Machine_1[up] W 1/mttr      ;

4  FROM Buffer_1, Machine_1[idle] TO Machine_1[busy]
     IF State_Machine_1[up]                                        ;
5  FROM Machine_1[busy] TO Machine_1[wait] W ServiceRate_1
     IF State_Machine_1[up]                                        ;
6  FROM Machine_1[wait], Buffer_1 TO Station_2, Machine_1[idle]
     IF State_Machine_1[up]                                        ;

7  FROM Station_2, num TOE W ServiceRate_2                         ;
```

Figure 6: MOSEL core for a system with unreliable machines

Line 4 : As soon as there are jobs in *Buffer_1* and *Machine_1* is *idle* and not *down* then *Machine_1* is set to *busy*.

Line 5: Now we use the pseudo state *wait* in order to serve a job. If during the service time *Machine_1* goes down then the job just served has to wait until the machine is repaired again.

Line 6: If a job has been served successfully it is moved to *Station_2*.

Line 7: At *Station_2* jobs are served with the rate *ServiceRate_2* and finally leave the system.

3.5 Combination of Different Characteristics

Up to now we have considered the different node types only separately. Now we want to show how to combine different node types. Let us for example consider the combination of a full batch system with unreliable machines. The corresponding MOSEL core can be seen in Fig. 7. in this case *Station_1* (consisting of *Buffer_1* and *Station_1*) is an unreliable Station with full batch strategy while *Station_2* is an ordinary service station.

```
1  FROME TO Buffer1, num W Arrival                                   ;

2  FROM State_Machine_1[up] TO State_Machine_1[down] W 1/mtbf        ;
3  FROM State_Machine_1[down] TO State_Machine_1[up] W 1/mttr        ;

4  FROM Machine_1[idle] TO Machine_1[busy]
     IF (Buffer_1 > b) AND State_Machine_1[up]                       ;
5  FROM Machine_1[busy] TO  Machine_1[wait] W ServiceRate_1
     IF State_Machine_1[up]                                          ;
6  FROM Machine_1[wait], Buffer_1(b) TO Station_2(b), Machine_1[idle] ;
     IF State_Machine_1[up]

7  FROM Station_2, num TOE W ServiceRate_2                           ;
```

Figure 7: MOSEL core for a system with unreliable machines and full batch strategy

As we can see at the structure of this specification it is basically the same as for the two separate specifications. The only difference is that if we want to use *Machine_1* we first have to check if it is *up*. This check is done in the lines 4, 5 and 6. If the machine is available it behaves like the two separate stations (full batch and unreliable machine). If *Machine_1* is not available no job is moved to *Station_2*. After an exponentially distributed time *mttr* the defect machine is repaired and can start working again.

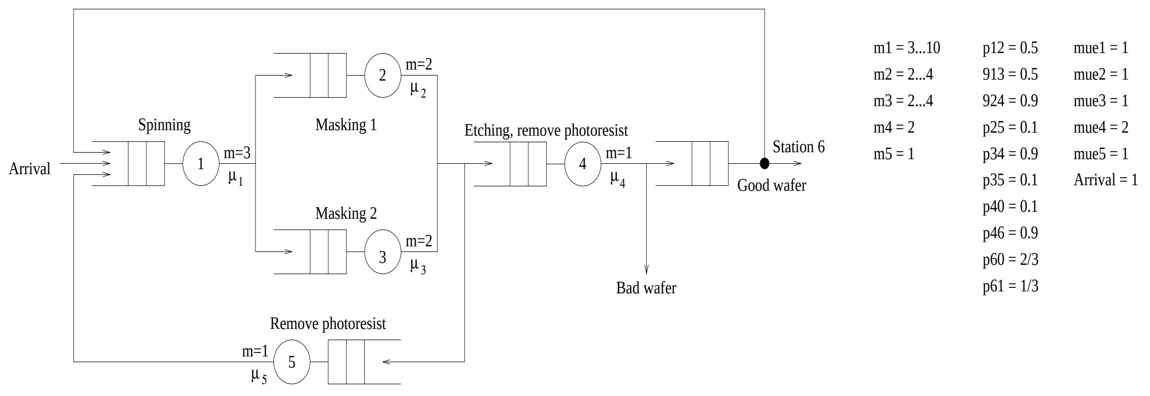

Figure 8: Model of the wafer production system

4 Specification and Analysis of a Wafer Production System

In this section we want to apply MOSEL to the photolitographic process of a wafer production system which is specified and analyzed. In [Rand86] the photolitographic process is divided into the following five steps:

- Cleaning of the wafer.
- Put photoresist on.
- Bake wafer and remove solvent by evaporation.
- Allign mask and illuminate and develope the wafer.
- Etching the wafer and removing photoresist.

The photolitographic process is done by two machine types:

- The spinning machine does the first three production steps.
- The masking machine does two steps.

From the first machine type 3 machines are available. From the second machine type (which does step four) we have machines from different producers. These machines are called masking 1 and masking 2. An additional machine type does the steps etching and removing the photoresist. The job of removing the photoresist is done by machine five in the case that the wafer has not been produced correctly. Since three layers of photoresist have to be put on, the wafer has to be processed again after a successful masking and spinning. In Fig. 8 the model of the wafer production system is shown.

- m_i is the number of machines at station i.
- p_{ij} is the probability that a job after being served at station i moves to station j.
- mue_i is the service rate at station i.
- $Arrival$ is the arrival rate of jobs at the system.

The MOSEL specification of the system is shown in Fig. 9.

Now we want to do some system analysis using the parameters given in Fig. 8 In the plot shown in Fig. 10 the mean total service time of a job is plotted as a function of the number of machines.

```
/*---------------Nodes--------------------------------------------*/

  <1..5> NODE Buffer_# [K]                                               ;
  <1..5> NODE Active_# [m_#]                                             ;
         NODE Station_6 [K]                                              ;
         NODE num  [K]                                                   ;



/*--------------Rules---------------------------------------------*/


          FROME  TO Buffer_1, num  W Arrival                             ;

<1..5>    FROM Buffer_#  TO Active_#                                     ;

<1..m_1>  FROM Active_1 TO Buffer_2  W #*mue_1 P p12 IF Active_1 == #    ;
<1..m_1>  FROM Active_1 TO Buffer_3  W #*mue_1 P p13 IF Active_1 == #    ;

<1..m_2>  FROM Active_2 TO Buffer_4  W #*mue_2 P p24 IF Active_2 == #    ;
<1..m_2>  FROM Active_2 TO Buffer_5  W #*mue_2 P p25 IF Active_2 == #    ;

<1..m_3>  FROM Active_3 TO Buffer_4  W #*mue_3 P p34 IF Active_3 == #    ;
<1..m_3>  FROM Active_3 TO Buffer_5  W #*mue_3 P p35 IF Active_3 == #    ;

<1..m_4>  FROM Active_4, num TOE    W #*mue_4 P p40 IF Active_4 == #     ;
<1..m_4>  FROM Active_4 TO node_6   W #*mue_4 P p46 IF Active_3 == #     ;

          FROM Active_5 TO Buffer_1 W mue_5                              ;

          FROM Station_6, num  TOE P p60                                 ;
          FROM Station_6 TO Buffer_1 P p61                               ;



/*--------------Results------------------------------------------*/

/*-------Mean number of active machines-------*/

<1..5> RESULT>> A_# = MEAN Active_#                                      ;


/*-------Utilization of the machines-------------*/

<1..5> RESULT>> rho_# = A_#/m_#                                          ;


/*-------Mean buffer length---------*/

<1..5> RESULT>> Q_# = MEAN Buffer_#                                      ;

/*-------Mean number of wafers (WIP) ------*/

RESULT>> WIP = MEAN num                                                  ;

/*-------Mean response time---*/

RESULTS>> T = WIP / Arrival                                              ;
```

Figure 9: MOSEL spezification for the wafer production system

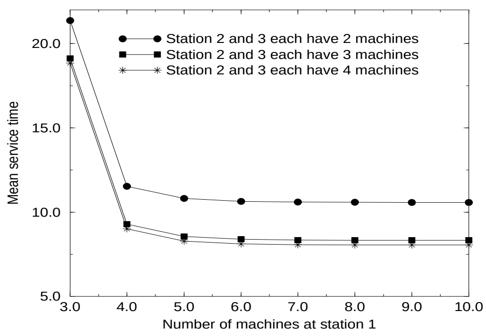

Figure 10: Results for different number of machines

As we can see in Fig. 10 it does not make any sense to have more than 4 spinning machines of each type for the given configuration. It is very easy to extend the whole model to contain also unreliable machines and finite buffer.

5 Conclusion

In this paper the analysis tool MOSES [BGJZ94] and the specification language MOSEL [BoHe95] are introduced and it is shown that it is relatively easy to specify and analyze production systems. We have shown as an example how to specify systems with unreliable machines and batch processing. The modeler is free to combine these components as necessary or to build more complex systems consisting of the presented elements. Right now macros for production systems are implemented which allow a much easier system description. The MOSEL specification is then generated automatically only by specifying the different characteristics of a station. In the example of the wafer production system we have shown how to specify a realistic system elegantly using MOSEL. Since the system analysis is fast compared to simulation it is possible to analyze the model with a whole set of parameters within a short time. Therefore MOSES is also suited for parameter optimization.

References

[AVDD90] Alvarez-Vargas R., Dallery Y., David R.: *A Study of the Continuous Flow Model of Production Lines with Unreliable Machines and Finite Buffers*, Journal of Manufacturing Systems, Volume 13/No.3.

[Bolc89] Bolch, G.: *Leistungsbewertung von Rechensystemen mittels analytischer Warteschlangenmodelle*, Teubner, Stuttgart, 1989. Springer, Berlin, 1965.

[BGJZ94] Bolch G., Greiner S., Jung H., Zimmer R.: *The Markov Analyzer MOSES*, Technical Report TR-I4-10-94, Institute of Mathematical Machines and

[BGR95] Bolch G., Greiner S., Jung H., Roessler M., and Zimmer R.: *Leistungsbewertung mit PEPSY-QNS und MOSES* , it+ti–Informationstechnik und Technische Informatik, pp 28-33, Vol. 37, 3/95.

[BoHe95] Bolch G., Herold H.: *MOSEL–A New Language for the Markov Analyzer MOSES* Technical Report TR-I4-95-02, Institute of Mathematical Machines and Data Processing IV University Erlangen-Nuernberg, March 1995.

[Rand86] Randhawa S.U.: *Simulation of a Wafer Fabrication Facility Using Network Modeling*, State University, Corvallis, USA, Jour. Paper Society of Manufac. Engineers 1986.