İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

UZMAN SİSTEME DAYALI ÜRETİM SİSTEMİ

DOKTORA TEZİ

Y.Müh. H. Engin DEMİRAY

Tezin Enstitüye Verildiği Tarih : 30 Eylül 1994

Tezin Savunulduğu Tarih : 27 Ocak 1995

Tez Danışmanı : Prof. Dr. Eşref ADALI

Jüri Üyeleri : Prof. Dr. Ataç SOYSAL

Prof. Dr. Nükhet YETİŞ

OCAK 1995

ÖNSÖZ

Çalışmamın başlangıcından son gününe kadar, yardımlarını esirgemeyen ve çalışmam boyunca bana destek olan, bitmek tükenmek bilmeyen sorularıma sabırla cevap veren, hocam sayın Prof. Dr. Eşref ADALI'ya, tezin yazımı sırasında bana yardımcı olan, sayın Y. Müh. Bülent MUŞLU'ya teşekkürü bir borç bilirim.

Ayrıca, çalışmalarım süresince daima yanımda olan, benimle bu zor ve sıkıntılı günleri paylaşan, sevgili eşim Rana'ya, çocuklarım Deniz ve Derya'ya, gösterdikleri bu engin sabır ve destekten dolayı kucak dolusu teşekkür ederim.

Küçükyalı
Ocak 1995

H.Engin DEMİRAY

İÇİNDEKİLER

NOTASYONLAR

m	işlem sayısı
n	üretimdeki model sayısı
nn	toplam model sayısı
d_j	üretimdeki bir model içindeki ürün sayısı (j. model)
dd_j	bir model içindeki toplam ürün sayısı (j. model)
q	bir anda kullanılan üretim araçları sayısı
qq	üretim ortamındaki tüm üretim araçları sayısı
c	bir anda çalışan işçi sayısı
cc	toplam işçi sayısı
k	kullanım parametresi
g	temel işlem
p	işlem adımı
t	işlem süresi
t_a	boşaltma süresi
$t_ç$	çalışma süresi
t_y	yerleştirme süresi
t_i	işlem adımı süresi (işleme süresi)
t_h	en hızlı çalışanın işlem süresi
t_o	insan etkisinin neden olduğu süre
S_i	ürün cinsi (i. iş)
T_i	toplam çalışma süresi (i. işlem için)
Z	günlük çalışma süresi (8 saat = 480 dak)
B_b	bir bedenden ne kadar üretileceği (b = beden)
W_b	bir modelin bir bedeninden üretilen miktar (b = beden)
N_v	bir modeldeki varyant sayısı
N_{bs}	bir model ürünün, bir bedeninin gerektirdiği sepet sayısı
$N_{sü}$	bir sepet içindeki ürün sayısı (genelde sabit ve 12 adet)

ŞEKİL LİSTESİ

TABLO LİSTESİ

ÖZET

Sanayi devriminin başladığı günlerde üretim tek tek yapılmaktaydı. Ancak günümüzde 4 ana kümede toplayabileceğimiz şekilde yapılmaktadır.

1. Sürekli Akan Süreç Tipi Üretim (Continous-Flow Proseses)
2. Tek Tek Parçaların Toplu Üretimi (Mass Production of discrete Products)
3. Partiler Halinde Yapılan Aralıklı Üretim (Batch Production)
4. Proje Tipi Üretim (Jop Shop Production)

Üretimin akışı içinde, üretim araçlarının yeniden düzenlenebilme imkanı olduğu için, *'Partiler Halinde Yapılan Aralıklı Üretim Yöntemi'* tez çalışmasına hedef seçilmiştir.

Partiler halinde yapılan aralıklı üretim otomasyonunda bilinen klasik üretim yöntemleri ile yapılan çalışmaların sonucunda, üretim ortamında oluşan kuyrukların ve beklemelerin azaltılması için geliştirilen USDUS yöntemiyle, üretim araçlarıyla operatörler arasında bir köprü kurulmaya çalışılmış ve bir simülasyon ile de, sorunun nasıl çözümlendiği gösterilmiştir. Hedef, bilgisayar mühendisliği gözü ile, üretim ortamının iyileştirilmesine katkı sağlamaktır.

Tezin 1. bölümünde, bu tez çalışması ile önerilen yeni yöntemin; "Uzman Sisteme Dayalı Üretim Sistemi : USDUS" ana hatları tanıtılmıştır. USDUS yönteminde, statik olarak planlaması yapılmış bir üretim ortamının, sürekli izlenerek dinamik olarak yeniden planlanması sağlanmaktadır. Aynı bölümde, yakın konularda yapılan diğer çalışmalar tanıtılmıştır. Ayrıca, USDUS'un katkıları açıklanmıştır.

2. Bölümde, önerilen yöntemle ilgili kuramsal çalışmalara yer verilmiştir. Bu bölümde, uzman(expert) sistemlerin genel bir tanıtımı yapılarak, tezde kullanılan uzman sistem yöntemi anlatılmış ve ayrıntıları 3. bölümde verilen uzman sistem algoritmasının oluşturulmasına zemin hazırlanmıştır.

3. Bölüm, bu tez ile önerilen USDUS yönteminin uygulamasına ayrılmıştır. Yöntemin, en iyi uygulanabileceği üretim ortamı örneği olarak, konfeksiyon fabrikası seçilmiştir. Daha sonra, bir konfeksiyon üretiminin ana hatları incelenmiş ve USDUS yönteminin uygulanmasına geçilmiştir. Yöntemin denenmesi amacıyla geliştirilen yazılım paketi ve simülasyon paketi kullanılarak, önerilen yöntemin doğruluğu gösterilmiştir.

4. Bölüm, bu çalışma için geliştirilmiş bulunan yazılım paketinin teknik yönünün tanıtımına ayrılmış olup, ayrıca üretim ortamının simülasyonları sonucu elde edilen grafiklerle, önerilen USDUS'nin üretim ortamına sağladığı üstünlükler gösterilmiştir.

5. Bölüm sonuç ve önerilere ayrılmıştır.

SUMMARY

Production Technologies From The Past To The Present

Beginning with individual handicrafts hundred's of years ago, the production was carried out later, first in workshops then in the plants.

Due to restricted transport and communication technologies and facilities in many countries, in the past the manufactured goods were for local consumption only. But there is a hard competition for the manufacturers now as these facilities have become available. To be able to stand to this hard competition the manufactures must produce quality goods at cheaper prices and these goods must satisfy the requirements of the consumers.

Thus, a product must provide the following conditions in the present in order to be able to compete with its competitors in the world markets:

- The goods must be up to date. To provide this, it should be manufactured fast and be presented into the markets before its competitors.
- The goods must be satisfy the requirements of different customers, hence production should be flexible.
- There should be a variety of the goods.
- The goods must be competitive on the prices, even if it is manufactured in small lots.
- The goods should keep its actuality (small or no stock).
- The raw material needed by the production should be on a minimum level, so that it does not cause any stop in the production.
- First of all 'A Production of Quality and Standard Conformity', must be achieved.

The conditions above require the manufacturers to develop and to use new and up to date technologies. As an example: it was not a loss, in the past, to manufacture goods in a workshop or in a factory with more than the requirements of the customers; it was a profitable investment for the manufacturers.

But at the present time it is running just contrary due to a hard competition. The question also, should we manufacture as much as we sell or should we sell as much as we manufacture shows clearly that the production capacity depends on the market conditions, on the cost price and on the quality assurance.

The developed technologies for the conformity for the production on the present conditions are divided into two groups : 'Management Technologies and Production Technologies'. These two applications seem first as they were different, but they are gradually affected from each other and involved one within the another.

At the beginning of the industrial revolution production was made singly. But the production at present can be compiled into four groups :

- Continuous - Flow Processes
- Mass Production of discrete Products
- Batch Production
- Job Shop Production

Short explanations are given below for these processes:

Continuous-Flow Processes

This is suitable for production of big lots manufactured continuously. The production process is inspected from the beginning to the end. In other words, each point of the process is sensitively measured and the complete control elements are managed. For this reason the complete production system should be equipped with automation. Oil refineries, chemical processes, continuous rolling are examples of this process.

Mass Production Of Discrete Products

This is a production process in where the goods are manufactured in great numbers and by mass production. The production of white-goods and cars are of this type. As it is seen from these examples, the product varieties are small, but the quantity of goods is quite high. In these production processes full-automatic or semi- automatic production lines are used. In the production lines there is a wide spread use of the robots.

Batch Production

Production of one item or of one part in certain numbers and at one time is the main goal of this process. The same item or part can be later manufactured in certain numbers, too. That is also, there is a question of a re-production. The production rate usually remains within middle numbers. Besides there is a wide spread use of the robots and CAM application where the product flow is achieved automatically or by workers. Printing books and textile manufacturing can be samples for this production type.

Job Shop Production

In this type production process the production rate is quite small. Usually goods specified by the customers and goods require a special or a high technology are placed in this classification. Planes, machine tools and lines are samples of this production. In this production machines with CAM - Application are used.

From the production processes described above the 'Batch Production' is chosen as a goal of this thesis study. The reason of this choice is that in the 'Continuous-Flow Processes' and in the 'Mass Production of Discrete Products' there is no possibility in the flow process of the production to reorganize the production instruments, and in the 'Job Shop Production' there is no continuity. In the following parts of this thesis, the place and procedure of the 'Batch Production will be understood when discussing the background or the procedure of the production.

Production Phases

Whatever the kind of the goods is manufactured the production is subject to the following stages. Before discussing these stages we shall summarize the production technologies used:

1. Customer or Market Requirements
2. Developing the Production Concept
3. Design Applications
4. Prototype + Draft Applications
5. Production Planning
6. Production Timing
7. Acquisition of the Production Instruments
8. Production
9. Quality Control

- To begin a production requires either a idea and wish or a certain market research and or an order given by a customer.
- In the second phase a cost calculation will be made by that will be searched the possibility of such a production and the profitability of this production.
- In the third phase comes the design and engineering. Then the tests on the prototype and finally decision of the production.
- The first stage of the actual production work is production planning and timing. Parallel to these tools needed for the plant, is determined.
- Then comes the point where the *'Real Production'* will be made.
- The production must already controlled regarding the quality. Because of production and control are affected from each another.

Taking into account the use of computers in the present it should be expressed also that CAD and CAM can be involved into the production.

The new production circle equipped with CAD/CAM can be explained as follows :

- To begin, a production depends on the customer or market requirements again.
- At the production concept, design and engineering work, the CAD/CAM are used effectively and the results of this use are quite advantageous.

- The preparation of the prototypes is easier with CAD and Computer Simulation.
- The use of packet software's prepared for the production planning and timing are very useful.
- The Production plants in the present are mostly equipped with the actual technologies, also with CNC's and Robots. Related with this type of production the complete documents are prepared by the computers, too. So, the software's to be used for the operation of the Robots and CNC-Machines are acquired.
- Finally in a environment equipped with the computers, the 'Quality Control' will be carried out with these instruments of the high technologies, too.

By going from the *'Batch Production'* to the *'Continuous-Flow Processes'* the numbers and density of the automation and CAD/CAM are increased. Besides, by increasing the manufactured goods, and decreasing the variety of the production the production procedures of the *'Continuous-Flow Processes'* will be suitable.

In contrary, by increasing the production variety and decreasing the production numbers the *'Job Shop Production'* is adopted.

Also, it is understood that parallel to the changes on the market and market conditions the customer requirements changes too. And it is safe to say, according to the changing customer requirements, the market and market conditions has become harder.

In the first part of this thesis the main lines of the new procedure suggested with this work is introduced depending on the up to date production procedures CAD/CAM. The latest developments in CAD/CAM technologies has put the possibility into the production instead of the classic production technologies.

The organization and management of the production background are fundamentally the occupation of the industrial engineers. But the CAD/CAM applications have given a possibility to the computer engineers to move this area, too. Of course, this is not a competition between the industrial engineers and the computer engineers , but the mutual support of both parties will bring more success in the industry and in the human life.

The main goal of this thesis is to represent the function of the observation and management of the production regarding the computer engineering, and it is intended to realize a production under the support of the computers for the industry engineers.

This thesis includes the necessary procedures and algorithm's which are used actually and the others, which are developed during this study. The USDUS procedure developed by this thesis is developed especially 'Batch Production. The suggestion is also *'Production System Based on Expert Systems = PSBES' (Uzman Sisteme Dayalı Üretim Sistemi = USDUS).* The foundations of USDUS Procedure may be summarized as follows :

- Here it is assumed that the production background is founded by the master persons. Because it is assumed that the plant investment is correct, and the correct machine row is selected for the process.
- The production machines and their operators are evaluated as one object. For this purpose a mathematical model is founded for each object, and this model is used on the computer. During the foundation of this models, a stochastic model is chosen.
- Information related to production such as the machine situations, the machine work at the actual time and the workers state of mind will be collected at once and continuously. For this purpose, it is suggested to use the new technologies of the data acqusition system.
- In this USDUS procedure data are collected from the production continuously. And these data's are compared with the production results continuously. During this evaluation the capability of the production place is taken into account. After this comparison a final decision is reached.
- At the decision phase the expert system technologies are used. The decisions are searched by the simulated technologies and the positive one is determined. And these useful decisions are later applied to the production.
- The results concluded at the decision phase are transmitted to the production place together with the new technologies suggested by this thesis.

The theoretical structure of the USDUS procedure suggested by this thesis depends on the following principles:

- A production plant is constituted in production instruments of definite numbers. On an production instrument there can be an operator or not. One operator is capable to operate more than one instrument. Besides the number of instruments is already more than operators.
- For USDUS - Procedure each production instrument is an object. To introduce each object a model is founded. Because of the relations between output and input of each object are defined, in other words the transfer function of the object is defined.
- If there is an operator to operate a production instrument, this is evaluated as an object, too. Therefore, each operator a model and a transfer function are defined.
- The production process defines the production instruments through which a good to be produced in the production background is passed. The approach suggested by this thesis the production process is changed into the relation between objects.
- As described previously, USDUS procedure developed for the procedures of the *'Batch Production'* is a decision and management system aided by the computer. Again it is assumed here, too, that the quantity and quality of the production instruments placed on the production area are established by the related engineer in the best way.

- The first step for a functional procedure is the foundation of a data collection system. To collect data, a data input terminal will be placed on to each production instrument. The equipment, function and connection of the units called 'Data Input Terminal' are explained in the 3d part.
- From complete instruments placed on the production area are used only the certain one within work time. Assumed that the workers are average, it can be calculated by means of table 2,2 how many has to be used from which production instrument. Then for each production instrument a worker will be assigned. By this way, will be received the numbers of the workers and the instruments. This picture is known by the management system.
- The founded production system will be tested within simulation environment. After this test the production system founded on middle values has to be tested within theoretical process. In the simulation process the function times, outage of the instruments and release of the work by the workers will be taken into account. At the end of simulation, process the numbers of the production instruments may change. According to this change the necessary correction is made and then the production process begins.

After the beginning of the production process complete instruments are observed. During observation the following are detected:

- Whether an instrument is defective or not defective
- When defective, is it repaired or not
- Which function it does
- It's operating times, and time comparison
- Whether the worker is busy or not
- Whether the worker has a reserve work or not
- Whether the work completed by a worker is put to the end or not for the following work station.

The tails are an other point to be followed in the production process. As discussed previously, within production background there are of three types of queues:

- Input Queue
- Inter Queue
- Output Queue

If there are not coming inter-queues in the production and if there are not production points which are free in awaiting for works, then can be said that the production plants is organized in the best way.

In a contrary organization to this is also a question of re-organization of that plant.

The reasons of inter queues may also be set up in following order:

- The production instruments may be defective or out
- One of the workers might leave his work
- The capability of some workers might decrease
- The manufactured product may be problematic

A reason on a previous production point may cause to an inter queue, too. For example :

- The capability of the workers may be increase to unforeseen levels.

Situations coming out at previous production points as it is described below, may cause that the workers remain work-free or they are in awaiting for a new work:

- Defect or trouble production instruments
- Work release by some workers for some any reasons
- Decrease in capability of the workers by some any reason
- Problematic production goods

Of course it is not requested that inter-queues are begin formed in the production process.

- To avoid such problems is only possible by changing the work places of some workers.

For example, a free worker can be moved to a work point where also an inter-queue has come out. But as simply as it is said. Because when a worker is moved from a point to an other point the stability of complete production process will be rearranged. The useful support provided by this thesis study, may to be seen on this point and for solving such problems is suggested to use the procedure USDUS.

This Study Is Intended To Reach The Following Goals

- Regarding Computer Engineering the production plant and the product inspection will be equipped with the computers.
- This model will bring a possibility of a simulation which will represent the static and dynamic situations of the production.
- It will be learned how a production plant can be organized according to the different criteria and / or which changes can be used when trouble situations are coming in a production plant which is organized on certain criterion's.

Even if the relation between the instrument number and good number is established ideal at the beginning, this relation will be damaged in course of time as this depends on a theoretical structure. Also, even if a production plants is planned in the best way, the plans will deviate in course of time. The main reasons of this are :

- A no-deterministic structure of the production process.
- Variety of the manufactured goods.

If we evaluate the efforts made by the production engineers as static planning the production, then we can say that the USDUS procedure provides a dynamic structure with the support of the computers for the production plant planed on a static base, by observing this plant continuously.

The theoretical explanations are given in part 2. In this part The Expert System Procedure is discussed by introducing the Expert Systems generally, and by that here is made a preparation for the foundation of the Expert System Algorithm its details are given in the 3rd part.

Expert System Procedures Used Within USDUS

The approaches known and the summary above which also are suitable for the 'Batch Production' are used within USDUS. The specifications of Expert System which are useful for USDUS are given below:

Expert Systems Additions

Master Aid : In the production factories taken into hand here as a sample, this matter has been discussed with master persons, and both about the production instruments, a wide information has been collected. These informations have been used for the following parts.

Foreseeing And Modelling : The foundation of USDUS is to model the production background in the best way and to simulate it depending on this model. The production environment is observed continuously. Meanwhile the values out of limits are registered. And according to these findings a decision is given for the necessary changes under the master's opinion. But, at USDUS before this decision is implemented the complete system is simulated and so the future of the system is foreseen. After this simulation the correctness of this decision is confirmed.

Continuity In The Factory : As the informations and capability of the workers are transferred into the computers, here is also reached a system standalone of the masters.

Training Support : By the developed software of simulation, the training of the new workers employed are provided.

Representation Of Informations In Expert System

There are three kinds of data representation:

- Based on rule
- Based on frames
- Based on logic

From these the procedure 'based on rule' is used at USDUS. In this consistency are constituted:

- The 'Data Base' included the specifications of the production environment.
- An 'Order of Rules' included general information about the production environment.
- Developed Procedures for solving processes and softwares suitable for these procedures.

Inference Procedures

At USDUS a forwards inspection of data is used as Data Inference Procedure. According to this procedure:

First, the data's related with production environment are put in, then the processes; pairing, selection and adding are implemented.

Functioning of USDUS

Before this system is operated the following data should be loaded into the management system:

- Varieties of the goods to be produced
- Production Plan
- Selected Production Machines
- Selected Workers.

When operating the system it should be assumed that all things are on the expected level. But the production system should be observed periodically. The observing period of the production environment (t_g) can be calculated depending on sample theory.

During System observing the followings are observed :

- State of the queues
- State of the machines
- State of the workers
- State of the product supply magazine
- State of product collection magazine

Some of the observed states are logical and some are as levels. The natures of the observed states are given below:

Queues	:	As level; for example, at the queue 3 baskets wait to be operated
Machines	:	Defect, at repair, functional, in a waiting work
Workers	:	Ill, short time on vacation, working, a waiting work
Supply Magazine	:	Free, full
Collection Magazine	:	Free, full

The result of each observation will be registered. The results received after observations will be called as the figure of the production. If there is a difference between a present and a previous result the management system begins to work.

At the end of the observation a queue forming can be detected in the step of process i. In this case it comes out that the work (i+1) at the step after the step where the queue is formed was not as quick as expected. For removing this queue there are two options.

- To stop the work at the work at the previous step (i-1) where the queue is formed. In other words, the worker on this work should be moved to the machines at the following step. But, for application of this act the second option should be taken into account.

- To decrease the queue length by increasing the machine numbers at the step i. But, to realize this a suitable machine should be at step i. If this is present, a worker from the process step coming after the step i. can be moved to the step i.

It can be said that a solution will be found if one of the options described above is selected. But, to move a worker forwards or backwards may damage the complete production system, and the damage of the production balance may effect on the flow of the complete system negatively. Because of by USDUS procedure it is suggested to run a simulation before application of the intended change.

In the third part of this study will be discussed the application of the USDUS procedure suggested by this thesis. A textile factory is selected as a best application plant for USDUS procedure. After this selection the production of a ready made cloths is investigated principally, and then was begun with the USDUS applications.

In this application the USDUS procedure has been tried on computers, and the results were positive.

For the purpose to try the USDUS procedure a Software-Packet has been developed. The correctness of the suggested procedure was demonstrated by using this Software Packet and a simulation packet.

The study related with third phase and the results related with these applications are given in third part.

In the fourth part of this thesis are discussed the technical sides of the Software Packet developed for this study. In this part the utilities of the USDUS procedure have been demonstrated by graphics.

In the fifth part of this thesis, results and suggestions are presented.

1. It is assumed that source planning and work planning will be come out as the same, as they are really planned and calculated at a production places organized all the best stochastically.

2. On the other hand, it is accepted that there will be machines waiting for work and work waiting for machines (also queues), in the course of time, when source planning and work planning are calculated statically.

3. Waiting situations and queues will decrease productivity and increase production cost.

4. In this thesis study, the use of expert system technologies is suggested to reduce the waiting situations and to shorten the queues together with an equipment of necessary software and hardware.

5. At the end of an application of the USDUS system, is shown that waiting situations are reduced and queues are shortened.

6. Finally the goals, described at the beginning of this thesis are reached.

1. BÖLÜM

Giriş

Bilgisayar Destekli Tasarım (BDT) ve Bilgisayar Destekli Üretim (BDÜ) tekniklerindeki son gelişmeler, klasik üretim teknikleri yerine, bilgisayara dayalı yeni tekniklerin kullanılmasına olanak sağlamıştır.

Üretim ortamlarının düzenlenmesi ve yönetimi, temelde, endüstri mühendislerinin uğraşı alanıdır. Ancak bilgi sistemlerindeki son gelişmeler, BDT ve BDÜ ile ilgilenen bilgisayar mühendislerine de, bu alanda araştırma ve geliştirme olanakları açmıştır. Bilgisayar mühendislerinin bu alanlardaki çalışmaları, endüstri mühendislerinin çalışmalarını kapsamak (ikame) yerine, onlara daha etkin ve yetenekli bir üretim sistemi kurmayı hedeflemektedir.

Bu tez ile ortaya konulacak olan çalışma, üretim izleme ve yönetme işlevini, bilgisayar mühendisliği açısından ele almaktadır. Bu nedenle, hedefi, endüstri mühendislerinin kullanabileceği, yeni ve gelişmiş bir üretim sistemini, bilgisayar destekli olarak gerçeklemektir. Bu hedefe ulaşabilmek için gerekli olan yöntem ve algoritmalardan, bilinenler ve bu tez çalışması sırasında geliştirilenler, tez içinde yer almaktadır.

Çalışmanın ilk aşamasında, mevcut üretim ortamlarında kullanılan bilgi sistemleri ve bunların yetenekleri ve eksiklikleri ele alınmıştır. Bu aşama ile ilgili araştırma sonuçları bu bölümde sunulmuştur. Aynı bölümün sonunda, bu tez çalışması ile önerilen yeni yöntemin ana hatları tanıtılmıştır.

1.1 Mevcut Üretim Yöntemleri

Geçmiş dönemde üretim tek tek yapılmaktaydı. Ancak günümüzde geçerli olan üretim teknikleri, dört kümeye ayrılmaktadır ve bunlar;

1. Sürekli akan süreç tipi üretim
2. Tek tek parçaların toplu üretimi
3. Partiler halinde yapılan aralıklı üretim
4. Proje tipi üretim

olarak anılmaktadır.

Bu yöntemler hakkında açıklamalar aşağıda kısaca verilmiştir:

Çimento üretimi, demir-çelik üretimi, petrol rafinerisi, sürekli akan süreç tipi üretim için örnek gösterilebilir. Tek tek parçaların topluca üretimi tekniğinde, yapılacak işlemler, tiplerine göre sınıflandırılarak yapılır. Örneğin beyaz eşya ve otomobil üretimi bu gruba girer. Partiler halinde yapılan aralıklı üretim sistemi, parça çeşidi fazla, imal edilecek miktarın orta büyüklükte olması halinde uygulanan bir sistemdir. Konfeksiyon tipi üretim bu yönteme örnek olarak verilebilir. Bir ürün veya ürün grubunun, planlanan bir projeye göre üretimine proje tipi üretim adı verilmektedir. Çoğu zaman üretilen ürün sabit olup, üretim araçları, üretilen ürünün çevresine yerleştirilmektedir. Gemi yapımı, köprü inşaatı proje tipi üretime örnek olarak verilebilir.

1.2 Yakın Çalışmalar

Tez çalışmasının başlangıcında, yakın konulardaki çalışmalar taranmıştır. Bu tarama işlemi sırasında;

- Bilgisayar Destekli Üretim (Computer Aided Manufacturing CAM),
- Esnek Üretim Sistemi (Flexible Manufacturing FMS),
- Bilgisayarla Tümleşik Üretim (Computer Integrated Manufacturing CIM),
- Uzman Sistem (Expert Systems),
- Bilgi Tabanlı Sistemler (Knowledge-Based Systems)

yukarıdaki sözcükler, anahtar sözcük olarak seçilmiş ve Dialog veri bankasından tarama yapılmıştır. Ayrıca, yakın konulardaki ülkemizdeki çalışmalar incelenmiştir. Tüm bu taramaların sonucunda 1000 dolayında bildiri özeti elde edilmiştir. Bu bildiri özetleri incelendiğinde, bazılarının; biribirinin aynı veya yakın olduğu, bazılarının; yetersiz olduğu görülmüştür. Sonuç olarak, ele alınan konuya yakın görülen 14 adet çalışma incelenmiştir.

Çalışmada, yakın ve seçilmiş olan çalışmaların değerlendirmeleri aşağıda verilmiştir.

A. Kusiak ve M. Chen : [1]

Üretimin Programlanması (zamanlaması) *(Process Scheduling)* konusunu, bilgi tabanına dayalı yöntemle çözmeyi önermektedir. Ayrıca, sistemin sürekli olarak gözlenmesini ve her ortaya çıkan durumda, bilgi tabanından yararlanarak; üretim sisteminin programlanmasının yeniden yapılmasını önermektedir.

Jingfan (PAUL) Yung ve Hsu-Pin (BEN) Wang : [2]

Montaj Tipi Üretimlerde, Üretim Planlamasının *(Production Planning)*, otomatik olarak yapılabileceğini göstermektedir. Bu işi yaparken, uzman sistem tekniklerinden yararlanmaktadır.

M.Mehdian ve R.Sagar : [3]

Esnek üretim tekniklerinde, üretim ortamından bilgi toplanmasının, dolayısıyla üretim ortamının gözlenmesinin, yarar sağlayacağı ve bu işlemlere paralel olarak ta, uzman sistem desteğinden yararlanmasını önermektedir.

Dennis E. ve O'Connor : [4]

Digital Equipment Corporation ve Carnegie-Mellon üniversitesinin birlikte sürdürülen araştırma geliştirme projesi sonunda; XCON adını verdikleri uzman sistem esaslı bir yazılım ortaya çıkmıştır. Bu yazılımın geliştirilmesinin nedeni, tüm üretim sistemlerinin planlama ve programlanmasında yakın gelecekte uzman sistemlerin kullanılacağı öngörülmektedir.

John McDermott : [5]

Bu çalışma, Digital Equipment Corporation ve Carnegie-Mellon üniversitesinin birlikte sürdürülen araştırma geliştirme projesini farklı açıdan tanıtmaktadır.

Mark S. Fox ve Stephen F. Smith : [6]

Üretim programlanmasında, bilgiye dayalı tekniklerin kullanıldığı bir çalışmadır. Yazar, üretim ortamının programlanması konusunun karmaşık, çok değişken olduğunu vurgulamaktadır. İncelenen makalesinde, uzman sistem tekniklerine dayalı olarak, Carnegie-Mellon üniversitesinde, geliştirmiş oldukları ISIS yazılımını tanıtmaktadır. Tanıtılan yazılım, programlamayı otomatik olarak yapmaktadır. Ayrıca, çalışma sırasında sistemin performansını arttırmak üzere, yeni programlama durumları üretmektedir. Çok geniş ölçekli olarak düşünülmüş olan, ISIS projesi kapsamında üretim ortamının simülasyonu da bulunmaktadır.

I.P. Tatsiopoulos ve I.A. Pappas : [7]

Bu makalede, üretimin planlanması konusunda, uzman sisteme dayalı bir yöntem önerilmiştir.

A.J. Billington ve A.R. Boucher : [8]

Karmaşık olan üretim ortamının gözlenmesi için, özel donanım ve yazılım gerektiği vurgulanmaktadır. Ayrıca, üretim sisteminin daha verimli olması için, kural tabanlı ve model tabanlı yöntemlerin kullanılmasının yararlı olacağı açıklanmaktadır. Yaptıkları öneriyle uyumlu, uygulamaları makale içinde yer almaktadır.

M.I. Ribeiro, C. Bispo, J. Sentieiro, C.P. Ferreira, L. Carvalho, R. Almeida, J.N. Ferreira : [9]

Bir üniversite işbirliği projesi olarak, Portekizde geliştirilen bu projenin hedefi, esnek üretim sistemleri için, otomatik planlama ve programlama yöntemlerinin geliştirilmesidir. Bu yöntemler geliştirilirken üretim zamanının, minumum yapılması hedeflenmiştir. Makalede geliştirilen planlama ve programlama sistemleri ayrıntılı bir şekilde açıklanmaktadır. Geliştirilen sistemin, %15 ila %20 arasında zamanı kısalttığı, deneysel sonuçlarla gösterilmektedir.

F. Nyhuis ve Wolfgang H. Dumke : [10]

Makalede, üretim ortamının gözlenmeden ve analiz edilmeden, etkin bir şekilde denetlenemiyeceği vurgulanmakta ve üretimin analiz yöntemi tanıtılmaktadır.

J-M le Veaux, M. Fraile, G Mazzocchi : [11]

ESPRIT 504 projesi kapsamında yapılan çalışmaları tanıtan bir makaledir. Çalışma, ana hatları ile üretim ortamının gözlenmesi ve bakımının nasıl yapılacağı üzerinedir.

H. Bera : [12]

İş planlaması için, çeşitli yöntemler bulunmasına karşın, hala optimum çözüm veren bir yöntemin bulunmadığını vurguluyor. Bu bildiri, bekleme süresini ve kuyruk süresini azaltacak bir iş planlama algoritması öneriyor.

Y. Rong : [13]

Statik planlamanın yetersiz kaldığını ve bu nedenle dinamik planlamanın gerekliliği vurgulanmaktadır.

S. Ulfsby : [14]

ESPRIT 2434 projesini tanıtan bu makale, üretim programlamasının dinamik olarak yapılmasını önermekte ve bu konuda yapılan çalışmaları anlatmaktadır.

İncelenen çalışmalardan görüldüğü gibi, üretimin planlanması ve programlanması konuları halen üzerinde çalışılan endüstriyel ve akademik konulardır. İncelenmiş olan çalışmaların birkaçı (Carnegie-Mellon üniversitesinde yapılan çalışmalar) doktora tezi olarak gerçeklenmiştir.

Ele alınan çalışmalar, ağırlıklı olarak üretimin en iyi şekilde planlanması ve programlanmasının nasıl yapılacağı üzerinedir. Üretim öncesi yapılan bu hesaplamaların (statik planlama ve zamanlama) üretim sırasında yetersiz kaldığı belirtilmektedir. Bu kusuru gidermek için, dinamik yöntemlerin geliştirilmesi gerektiği vurgulanmaktadır. Dinamik yöntemlerin geliştirilebilmesi için, üretim ortamının gözlenmesi (monitör edilmesi) gerektiği açıklanmaktadır.

Dinamik yöntemlerin, yeni planlama ve programlama yapabilmesi için, bilgiye dayalı uzman sistem tekniklerinin kullanılmasının gerekliliği belirtilmektedir.

1.3 Önerilen Yöntem

Bu tez çalışması ile geliştirilen yöntem, özellikle partiler halinde aralıklı üretim türleri hedef alınarak geliştirilmiştir. Önerilen "Uzman Sisteme Dayalı Üretim Sistemi" kısaca "USDUS" yönteminin ana hatları şöyle özetlenebilir:

- Üretim ortamının, bu konuda uzman sayılan kimseler tarafından, en iyi şekilde kurulmuş olduğu kabul edilmektedir. Dolayısıyla, makine parkında bulunması gereken makinelerin cinsleri ve sayılarının doğru belirlendiği varsayılmaktadır. Ayrıca, makinelerin, üretim sürecine uygun olarak dizildikleri de kabul edilmektedir.

- Üretim makineleri ve bunları kullananlar, birer nesne olarak ele alınmaktadır. Bu nedenle her nesnenin, matematiksel modeli oluşturulmakta ve bu model bilgisayarda kullanılmaktadır. Her makinenin modelinin kurulması sırasında, deterministik verilere dayalı bir model kurma yerine, özellikle stokastik modelinin kurulması benimsenmiştir.

- Üretim ortamından, üretim ile ilgili bilgiler, örneğin; makinelerin durumları, o anda yaptığı iş, çalışanın durumu, gibi bilgiler, gerçek zaman verileri olarak anında ve sürekli olarak toplanacaktır. Bu amaçla, tez çalışması kapsamında geliştirilen yeni veri toplama teknikleri, bu çalışma ile önerilmektedir.

- Önerilen USDUS yönteminde, üretim ortamından anlık veriler, sürekli olarak toplanmaktadır. Toplanan veriler, üretimden beklenen sonuçlar ile sürekli olarak karşılaştırılmaktadır. Karşılaştırma işlemi sırasında, üretim ortamının yetenekleri gözönüne alınmaktadır. Karşılaştırma işleminin ardından, üretimin daha verimli olması için sonuç karar oluşturulmaktadır. Karar aşamasında, uzman sistem tekniklerinden yararlanılmaktadır. Alınmış olan kararlar, simülasyon teknikleri ile irdelenmekte ve sonuçlardan olumlu olanlar belirlenmektedir. Olumlu kararlar daha sonra üretim ortamına aktarılmaktadır.

- Karar verme sürecinde ortaya çıkan sonuçlar, yine bu tez çalışması ile önerilmiş olan yeni teknikler ile üretim ortamına iletilmektedir.

Çalışmanın ikinci aşaması, önerilen yöntemin gerçeklenmesi için gerekli olan kuramsal çalışmaları kapsamaktadır. Kuramsal çalışmalara, bilinen üretim tekniklerinin kısa bir tanıtımı ile başlanmıştır. Bunun ardından, bu tez ile önerilen yönteme zemin hazırlamak üzere geliştirilmiş olan kavramlar tanıtılmıştır. Daha sonra, önerilen yöntemin kuramsal yapısı ortaya konmuştur.

Bu tez ile önerilen USDUS yönteminin kuramsal yapısı temelde aşağıda sıralanan ilkelere dayanmaktadır:

- Bir üretim ortamı, belli sayıda üretim aracından oluşmaktadır. Bir üretim aracının başında, operatör bulunabilir ya da bulunmayabilir. Bir operatör, birden fazla üretim aracını yönetebilecek yetenektedir. Ayrıca, üretim araçlarının sayısı, operatör sayısından, her zaman daha fazladır.

- USDUS yöntemi için her üretim aracı bir nesnedir. Her nesneyi tanımlamak üzere bir model kurulmaktadır. Dolayısıyla, her nesnenin çıkışı ile girişi arasındaki ilişkiler belirlenmekte, bir başka deyişle nesnenin transfer fonksiyonu belirlenmektedir.

- Üretim aracını yöneten bir operatör var ise, bu da bir nesne olarak değerlendirilmektedir. Dolayısıyla, her operatör için bir modeli ve transfer fonksiyonu tanımlanmaktadır.

- Üretim ortamında, üretilecek olan bir ürünün, hangi üretim araçlarından geçeceğini, üretim süreci belirlemektedir. Bu tez ile önerilen yaklaşımla, üretim süreci, nesneler arasındaki ilişkiye dönüştürülmektedir.

- Daha önce belirtildiği gibi, USDUS yöntemi; partiler halinde yapılan aralıklı üretim yöntemleri için geliştirilmiş, bilgisayar destekli karar verme ve yönetme sistemidir.
- Yine, önceden belirtildiği gibi, üretim ortamında yer alması gereken üretim araçlarının nitelik ve niceliklerinin en iyi biçimde, ilgili mühendislerce saptandığı varsayılmaktadır.

Bir üretim ortamı, ne kadar iyi planlanırsa planlansın, zaman içinde, planlardan sapmalar olacaktır. Bunun ana nedenleri;

- Üretim sürecinin tam anlamıyla deterministik yapıda olmaması
- Üretilen ürünlerin çeşitlilik göstermesi

biçiminde açıklanabilir. Üretim ile ilgili mühendislerin yapmış oldukları çalışmalar, üretimin statik planlaması olarak değerlendirilirse, USDUS yönteminin, statik olarak planlaması yapılmış bir üretim ortamının, sürekli izlenerek dinamik olarak yeniden planlanmasını sağlamakta ve bu işlemi yerine getirebilmek üzere bilgisayar desteğinden yararlanıldığı söylenebilir.

Kuramsal çalışmalara ilişkin açıklamalar, 2. Bölümde yer almaktadır. Bu bölümde, uzman (expert) sistemlerin genel bir tanıtımı yapılarak, tezde kullanılan uzman sistem yöntemi anlatılmış ve 3. bölümde ayrıntıları verilen, uzman sistem algoritmasının oluşturulmasına zemin hazırlanmıştır.

Çalışmanın üçüncü bölümü, bu tez ile önerilen USDUS yönteminin uygulamasına ayrılmıştır. USDUS yönteminin en iyi uygulanabileceği bir üretim ortamı örneği olarak, konfeksiyon fabrikası benimsenmiştir. Bu seçimin ardından, bir konfeksiyon üretiminin ana hatları incelenmiş ve USDUS yönteminin uygulanmasına geçilmiştir. 2. Bölümde, kuramsal yapısı tanıtılmış olan USDUS yöntemi, konfeksiyon fabrikasını örnek almak koşulu ile, bilgisayar ortamında denenmiş ve olumlu sonuçlar alınmıştır.

USDUS yönteminin denenmesi amacıyla, bir yazılım paketi geliştirilmiştir. Geliştirilen yazılım paketi ve buna bağlı olan simülasyon paketi kullanılarak, önerilen yöntemin doğruluğu gösterilmiştir. Üçüncü aşamayla ilgili çalışmalar ve bu çalışmalara ilişkin sonuçlar tezin 3. Bölümünde yer almaktadır.

Tezin 4. Bölümü, bu çalışma için geliştirilmiş bulunan yazılım paketinin teknik yönünün tanıtımına ayrılmıştır. Bu bölümde, üretim ortamının çeşitli simülasyonları sonucu elde edilen grafikler (USDUS üretim ortamında var/yok) sayesinde; önerilen USDUS'nin üretim ortamına sağladığı üstünlükler, açık bir şekilde gösterilmiştir.

5. Bölüm sonuç ve önerilere ayrılmıştır.

2. Bölüm

Kuramsal Yaklaşımlar

2.1 Üretim Teknolojisindeki Gelişmeler

Üretim yüzyıllarca önce zenaat biçiminde, bireysel olarak başlamış ve geçen yüzyılın sonunda sanayi devrimi ile önce atölyeye, daha sonra ise fabrikaya dönüşmüştür.

Geçmiş dönemde, ülke içindeki ulaşım ve iletişim tekniklerinin kısıtlı olması nedeniyle, üretilen mallar ancak bölgesel tüketilebilmiştir. Günümüzde ise, hem ulaşım, hem de iletişim olanaklarının artması nedeniyle, üreticiler sert bir rekabet ortamına girmiş bulunmaktadırlar. Bu rekabette ayakta kalabilmenin koşulu, kaliteli, ucuz ve müşterinin isteklerine uygun mal üretebilmektir. Özetlemek gerekirse; Günümüzde ürünün tüm dünyadaki rakipleri ile rekabet edebilip, yaşayabilmesi için, aşağıda sıralanan koşulları sağlaması gerekmektedir:

- Üretilen malın, güncel olması lazımdır. Bunu sağlayabilmek için, ürünün en hızlı şekilde üretilip, piyasaya sunulması gerekmektedir.

- Ürün, değişik müşteri isteklerine (zevklerine) cevap verebilecek (kültürel, bölgesel farklara) nitelikte üretilmelidir.

- Üründe çeşitlilik olmalıdır.

- Üretimin miktarı küçük bile olsa, rekabet edebilecek fiyatlarda imal edilebilmelidir.

- Gereğinden fazla ve güncelliğini kaybedecek seviyede üretim yapılmamalıdır. (Gereksiz stoklara gidilmemeli ve tapon ürün kalmamalıdır)

- Üretimin ihtiyacı olan ham madde, üretimde tıkanmalara yol açmayacak minumum seviyelerde olmalıdır.

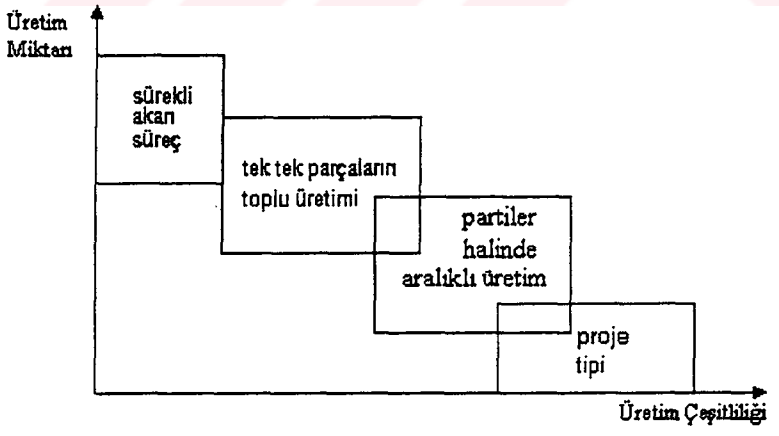
- Üretim kaliteli ve standartlara uygun olmalıdır.

Günümüz koşullarına uygun üretim sağlanabilmesi için, geliştirilmiş olan teknikler, iki kümede toplanmaktadır. Bunlardan birincisi yönetim teknikleri, ikinciside üretim teknikleridir. Birbirinden ayrı gibi gözüken bu iki konu giderek daha içiçe ve daha etkileşimli hale gelmektedir. Yönetim tekniklerinde ulaşılmış olan bugünkü aşama, Üretim Bilgi Sistemi (Production Managment System) olarak anılmaktadır.

Sanayi devriminin başladığı günlerde üretim, tek tek yapılmaktaydı. Ancak, zamanla gelişen üretim ve yönetim tekniklerine bağlı olarak, günümüzdeki üretimler 4 ana kümede toplanabilecek şekilde yapılmaktadır.

1. Sürekli akan süreç tipi üretim
2. Tek tek parçaların toplu üretimi
3. Partiler halinde yapılan aralıklı üretim
4. Proje tipi üretim

Bir üretim ortamında, hangi yöntemin daha kullanışlı olacağına karar verirken, ilk olarak ürün çeşitliliği ve üretim miktarının gözönüne alınması gerekir. Şekil 2.1 de ürün çeşitliliği ile, üretim miktarları arasındaki ilişkiye bakarak, hangi üretim yönteminin seçilmesi gerektiği gösterilmiştir. [15]



Şekil 2.1: Üretim çeşitliliği ile üretim miktarı arasındaki ilişki

Daha önce isimleri belirtilen dört üretim yöntemi, aşağıda kısaca tanıtılmışıtır:

Sürekli Akan Süreç Tipi Üretim

Büyük miktarda ve sürekli üretilen ürünler için uygundur. Süreç baştan sona kadar denetlenmektedir. Bir başka deyişle, sürecin her noktası duyargalar ile ölçülmekte ve tüm kontrol elemanları yönetilmektedir. Bu nedenle, tüm üretim sisteminin otomasyonu gereklidir. Petrol rafinerileri, kimyasal süreçler, sürekli haddehaneler bu tür üretime örnek verilebilir.

Tek Tek Parçaların Toplu Üretimi

Bir ürünün çok sayıda ve toplu olarak üretilmesi yöntemidir. Beyaz eşya ve otomobil üretimi bu sınıfa sokulabilir. Örneklerden de anlaşılacağı gibi, üretilen ürün çeşidi az olmakla beraber, üretim miktarı oldukça yüksektir. Bu tür üretimlerde, tam ya da yarı otomatik üretim bandları kullanılmaktadır. Üretim bandlarında, robot kullanımı oldukça yaygındır.

Partiler Halinde Yapılan Aralıklı Üretim

Bir ürün ya da bir parçanın, belli sayıda ve bir defada üretilmesi bu yöntemin ana hedefidir. Aynı ürün ya da parça daha sonra, yine belli sayıda üretilebilir. Dolayısıyla, üretimde yineleme sözkonusudur. Üretim miktarı, genelde, orta değerlerde kalmaktadır. Partiler halinde yapılan aralıklı üretimde, bilgisayar destekli tezgahlar ve robotlar oldukça yaygın olarak kullanılmaktadır. Ürün akışı, otomatik ya da insanlar tarafından sağlanmaktadır. Bu tür üretime örnek olarak, kitap basımı, konfeksiyon çalışmaları gösterilebilir.

Proje Tipi Üretim

Bu tür üretimde, üretim miktarı çok düşüktür. Genellikle müşteriye özel ve yüksek, ya da özel teknoloji ürünleri bu sınıfa girmektedir. Uçak, takım tezgahı ve gemi üretimi bu tür üretime örnektir. Üretimde, bilgisayar destekli tezgahlar kullanılmaktadır.

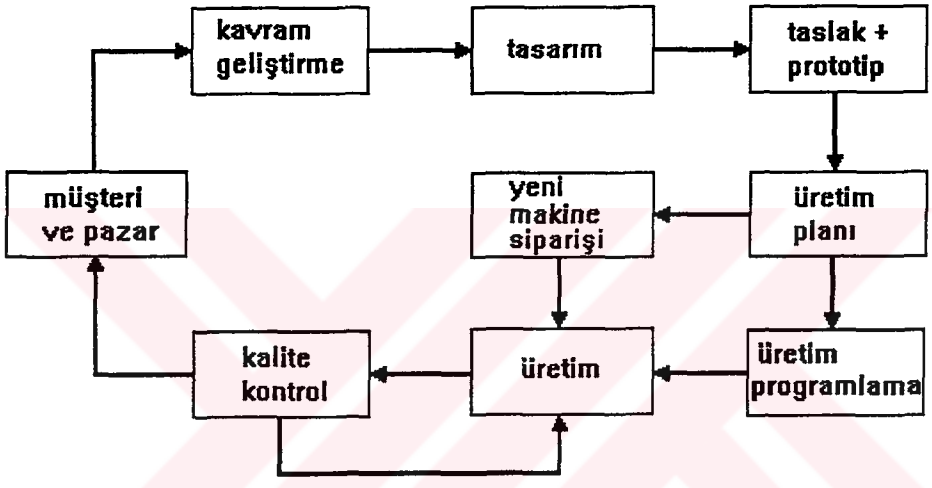
2.1.1 Üretimin Aşamaları

Üretilecek malın türü ne olursa olsun, üretim, şu aşamalardan geçilerek gerçeklenir.

1. Müşteri veya piyasa talepleri,
2. Üretim kavramının geliştirilmesi,
3. Tasarım çalışmaları,
4. Taslak + Prototip çalışması,

5. Üretim planlaması,
6. Üretimin programlanması (zamanlama),
7. Üretim araçlarının sağlanması,
8. Üretim,
9. Kalite kontrol

Bu aşamalar arasındaki ilişkiler, Şekil 2.2 de gösterildiği gibidir [15]. Şekil 2.2 de verilen klasik çevriminin aşamalarını şöyle açıklayabiliriz.



Şekil 2.2: Klasik üretim yöntemi

- Üretimin başlayabilmesindeki fikir ya da talep, ya belli bir piyasa araştırması sonucunda ortaya çıkabilir, ya da bir müşterinin arzusundan veya siparişinden doğabilir.

- İkinci aşamada böyle bir ürünün, mevcut olanaklarla üretilip, üretilemeyeceği, nasıl üretileceği, karlılık oranları araştırılır. Bir yerde üretimin olurluluğu ortaya çıkartılır.

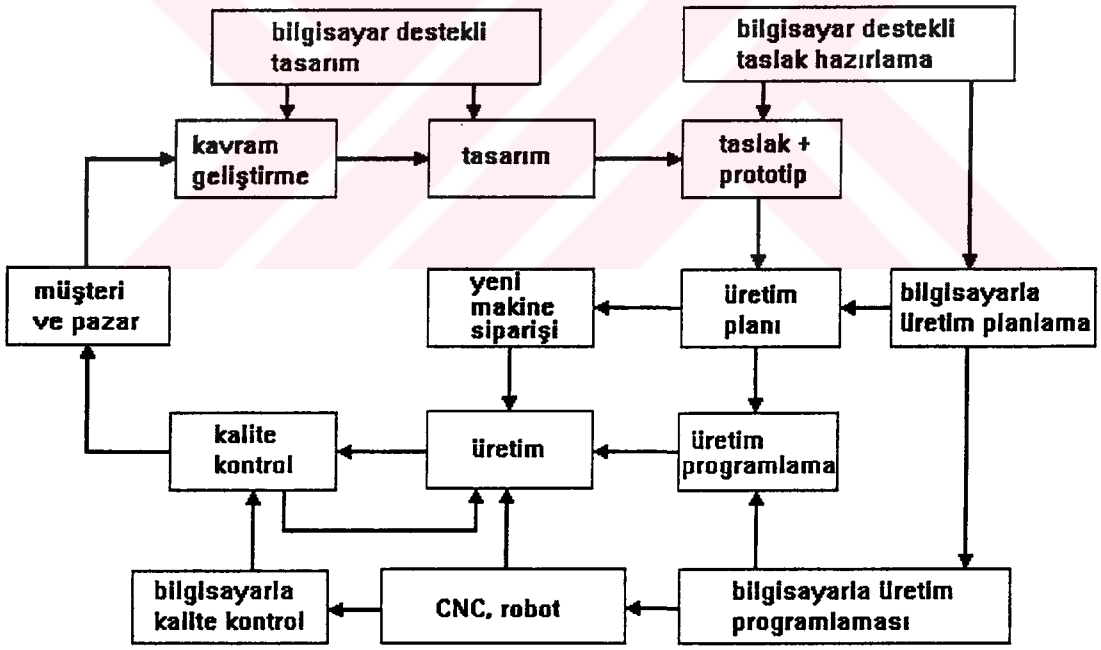
- Üretimin karlı olacağı anlaşıldığı takdirde, tasarım ve mühendislik aşamasına geçilir. Bu aşama proje ve mühendislik çalışmalarını içerir. Mühendislik ve tasarım aşamalarının sonucunda, taslak + prototip çalışmalarına geçilir ve prototip üzerinde gerekli sınamalar yapılır. Prototip aşaması tamamlandıktan sonra, artık üretim çalışmalarına başlanabilir.

- Üretim çalışmasının ilk aşaması, üretimin planlanması ve programlanmasıdır. Üretimin planlanmasına ve programlanmasına paralel olarak, bu üretim için geçerli olan makine parkının oluşturulması gerekir.

- Üretimin bundan sonra geldiği nokta, gerçek üretimin yapılması evresidir.

- Üretim her zaman kalite yönünden kontrol edilmelidir. Dolayısıyla üretimle, kalite kontrol her zaman yakın etkileşimde bulunurlar.

Günümüzde bilgisayar olanakları gözönüne alındığında, klasik üretim çevrimine getirilebilecek katkılar, Şekil 2.3 te gösterilmiştir. Şekil 2.3 te gösterilen düzen, aslında Bilgisayar Destekli Tasarım ve Üretim adını taşımaktadır (BDT/BDÜ ya da ingilizce kısaltmasıyla CAD/CAM : Computer Aided Design/Computer Aided Manufacturing). [15]



Şekil 2.3: Bilgisayar destekli üretim yöntemi

Şekil 2.2 ve Şekil 2.3 karşılaştırıldığında, Şekil 2.3' ün aslında Şekil 2.2' yi de kapsadığı ve ona bazı ekler getirdiği açıktır.

Bilgisayar destekli tasarım ve üretimle desteklenen yeni üretim çevrimi şöyle açıklanabilir:

- Üretim yine, müşteri ya da piyasa talebi ile başlamaktadır.

- Üretim kavramının oluşturulmasında, tasarım ve mühendislik çalışmalarında, bilgisayar destekli tasarım olanakları etkin bir şekilde kullanılmakta ve bu konuda büyük yararlar sağlamaktadır.

- Taslak + prototipin hazırlanması sırasında da, bilgisayar destekli tasarım ve bilgisayarla simülasyon büyük kolaylıklar getirmektedir.

- Üretimin planlanması ve programlanması konusunda hazırlanmış olan paket yazılımlar, üretimin planlanması ve programlanması (zamanlama) çalışmalarını kolaylaştırmaktadır.

- Günümüzün üretim araçları, günün getirdiği teknolojinin sonucu olarak CNC tezgahlar ve Robotlardan oluşmaktadır. Üretilecek mamulün tasarım aşamasından başlayarak, bilgisayar ortamında sürdürülen çalışmalar, bütün dökümantasyonunda bilgisayar ortamında oluşmasını sağlamaktadır. Bunun sonucu olarak, Robot ve CNC tezgahların çalışmasını organize edecek yazılımlar da, bu çalışmaların sonucunda elde edilmektedir.

- Son olarak, bilgisayarla donatılmış böyle bir üretim ortamında Kalite kontrol da, doğal olarak bilgisayar yardımı ile yapılacaktır.

2.2 Bu Çalışmanın Amacı

Şu ana kadar ki açıklamalardan da anlaşılacağı gibi, günümüz üretim teknolojisinde, hem üretimin yönlendirilmesi, hem de üretimin kendisinde bilgisayar desteğinden yoğun biçimde yararlanılmaktadır.

Üretim ortamının planlanması ve işletmeye alınması, temelde endüstri mühendisliğinin çalışma alanı olmakla beraber, bilgisayar ile desteklenmeyen bir işletmenin başarılı olamayacağı görülmektedir. Bu tez çalışmasının amacı;

- Endüstri mühendisleri tarafından, en iyi şekilde kurulmuş ve düzenlenmiş olan bir üretim ortamının, modelini oluşturmak;

- Üretim ortamının sürekli gözlenmesini sağlayacak bir bilgi toplama sistemi tasarlamak ve uygulamak;

- Üretim ortamından toplanan verileri, sistem modeli üzerinde irdelemek ve en iyi çözümü elde etmek; Bu aşamada, uzman sistem kuramlarından yararlanmak;

- En iyi çözüme göre, üretim ortamını yeniden düzenlemek. Bir başka deyişle, üretim araçlarının sayısını değiştirmek;

biçiminde özetlenebilir.

Bölüm 2.1.1 de tanıtılan, üretim yöntemlerinden, *'Partiler halinde yapılan aralıklı üretim'* yöntemi, bu tez çalışması için hedef seçilmiştir. Böyle bir seçim kararının verilmiş olmasının nedeni şöyle açıklanabilir: *'Sürekli akan süreç tipi' ve 'Tek tek parçaların toplu üretimi'* tekniklerinde, üretimin akışı içinde, üretim araçlarının, yeniden düzenlenmesi olanağı yoktur. *'Proje tipi üretimde'* ise, üretimin sürekliliği sözkonusu değildir. Tezin bundan sonraki kısımlarında, üretim ortamı ya da yönteminden sözedildiğinde, *'Partiler halinde yapılan aralıklı üretim'* ortamı ve yöntemi anlaşılmalıdır.

2.3 Temel Tanımlar

Tez çalışmasını, istenen hedefe ulaştırabilmek için, ilk olarak yapılması gereken çalışma, üretim ortamının en iyi şekilde tanımlanması ve ardından modellenmesidir. Bir üretim ortamında üç temel öğe bulunmaktadır:

1. Ürünler
2. Üretim araçları
3. Çalışanlar

Bu tez çalışmasında, ilk aşamada, üretim ortamının bu üç öğesinin modeli kurulmuştur. Ayrıca, bu üç öğe arasındaki ilişkiler tanımlanmıştır. Bu bölüm içinde, modellemenin ve ilişkilendirmenin nasıl yapıldığı anlatılacaktır.

Ürünler

Bir ürünün ortaya çıkabilmesi için, o ürünü oluşturan parçaların, birbirine belli bir sırada eklenmesi veya ürün üzerinde bazı işlemlerin gerçeklenmesi gerekmektedir. Her bir ekleme ve işlem bir iş istasyonunda gerçeklenmektedir.

Bir ürünün oluşmasında, ardarda gerçeklenen işlemler p_j olarak gösterilecektir. Üretim ortamında gerçeklenebilecek işlem çeşidi sayısı m olarak belirlenmiştir. Ancak bir ürünün üretilmesi aşamasında, tüm üretim aşamalarından geçilmesi gerekmemektedir.

Üretim ortamında birbirinin aynı ve/veya farklı ürünler üretilebilir. Birbirinin aynı olan ürünler kümesine, ***model*** denecektir. Üretim ortamında, toplam nn değişik modelin üretildiği varsayılacaktır. Her model içinde, birden fazla ürün bulunacaktır. Bir model içindeki ürün sayısı dd ile gösterilecektir. (Uygulama tekstil endüstrisinden alındığı için, *'mamul'* yerine *'model'* tabiri kullanılmıştır)

Yukarıda değinildiği gibi bir ürün, çok sayıda işlemin peşpeşe gerçeklenmesi ile meydana gelmektedir. Bu durumda, herhangi bir ürünün oluşması şöyle yazılabilir:

$$S_j = p_{j1} + p_{j2} + \cdots + p_{ji} + \cdots \qquad (2\text{-}1)$$

Üretim ortamında üretilecek olan ürünlerin toplam model sayısı nn ve bir model içindeki ürün sayısı ddj olabilir. Ancak, bir anda üretim ortamında bulunabilecek model sayısı n dir. Her bir modelden üretilecek olan miktar ise dj olarak sınırlanmış olabilir.

Yukarıda belirtilen hususların ışığında, ürünler, bilgisayar ortamında modellenecektir.

Üretim Araçları

Üretim ortamında bulunan üretim araçları, belli işleri gerçeklemek üzere kullanılmaktadır. Belli işlemleri gerçeklemek üzere birden fazla üretim aracı kullanılabilecektir. Üretim için gerekli olan üretim araçlarının sayısının, üretim öncesinde hesaplandığı varsayılmaktadır. Üretim ortamında bulunan tüm üretim araçlarının sayısı qq ve belli bir anda kullanılanlarının sayısı ise q ile gösterilecektir.

Bir üretim aracı, birden fazla işlemi gerçeklemek üzere kullanılır ise, bu çalışmada, her çalışma için başka bir üretim aracı olarak değerlendirilecektir.

Her üretim aracı, belli bir sınıf işlemi yerine getirmek üzere gerçeklenir. Ancak, bir üretim aracında yapılan iş, işin niteliği ve karmaşıklığına bağlı olarak kısa ya da uzun sürebilir. Bu görüşlerin ışığında, bir üretim aracında yapılan iş, işlem ve işlem süresi cinsinden şöyle ifade edilmiştir:

$$p = t\,g \qquad (2\text{-}2)$$

Bu bağıntıda görülen **g**, üretim aracında yapılan temel işlemi göstermektedir; dolayısıyla ***temel işlem*** olarak anılacaktır. (2-2) bağıntısında görülen **p** ise, ***işlem adımı*** olarak anılacaktır. Aynı bağıntıda yer alan **t**, ***işlem süresi*** olarak adlandırılmıştır.

Üretim ortamında bulunan tüm üretim araçlarının, bir üretim sürecinde kullanılmayacakları yukarıda söylenmişti. Bir süreç içinde, bir üretim aracının kullanılması ya da kullanılmaması durumu **k** parametresi (***kullanım parametresi***) ile belirlenmiştir (k=1 için kullanılıyor, k=0 için kullanılmıyor). Dolayısıyla (2-2) bağıntısı şöyle yazılabilir:

$p = k\,t\,g$ veya bir üretim aşaması için

$p_j = k_{ji}\,t_{ji}\,g_i$ bu bağıntı (2-1) de yerine konursa;

$S_j = k_{j1}t_{j1}g_1 + k_{j2}t_{j2}g_2 + \cdots + k_{ji}t_{ji}\,g_j + \cdots$ veya toplam üretim aşaması sayısı m olarak alındığına göre;

$$S_j = \sum_{i=1}^{m} k_{ji}t_{ji}g_i$$ yazılabilir. (2-3)

Üretimde kullanılan her üretim aracının, belli bir olasılıkla sağlam ya da bozuk olması sözkonusudur. Bir üretim aracının sağlam ya da bozuk olması olasılığı daha sonra incelenecek ve bağıntılara etkisi ele alınacaktır.

Çalışanlar

Üretim araçları, bu araç için atanmış insanlar tarafından yönetilmektedir. Üretim araçlarının, insan tarafından yönetilmesi, üretimin modellenmesi

sırasında, insan etkisinin de gözönüne alınmasını gerekli kılmaktadır. Bu etki, iki şekilde görülmektedir; işlem süresi ve insanın çalışır durumda olması ya da olmaması.

Özellikle emek yoğun çalışmalarda, bir işlem adımının süresini, üretim aracının süresi ve insanın yetenekleri belirlemektedir. Üzerinde işlem yapılacak olan parçanın üretim aracına yerleştirilmesi t_y, üretim aracının çalışma süresi $t_ç$ ve parçanın üretim aracından alınması t_a ile gösterilirse, işlem adımı süresi şöyle yazılabilir:

$$t_i = t_y + t_ç + t_a \qquad (2\text{-}4)$$

(2-4) bağıntısında görülen t_y ve t_a doğrudan doğruya, üretim aracını kullanan kişinin yeteneklerine bağlı zamanlardır. $t_ç$ ise, hem üretim aracının hem de kullanıcının yeteneklerine bağlı bir değerdir. Üretim aracının, ele alınan işlemi en yüksek hızda tamamlaması durumunda geçen zamana t_h dersek,

$$t_ç = t_h + t_k \qquad (2\text{-}5)$$

bağıntısını yazabiliriz. (2-5) bağıntısında görülen t_k kişinin harcadığı süredir.

Bir çalışanın, birden fazla üretim aracını kullanabildiği varsayılacaktır. Ancak, her üretim aracını aynı yetenek düzeyinde kullanamayacağı da açıktır. Çalışanlar ve üretim araçları arasındaki ilişki Tablo 2.1 deki gibi gösterilecektir.

Tablo 2.1
Çalışanlar ile üretim araçları (ya da işlemler) arasındaki ilişkiler

Çalışan	1.işlem g_1	2.işlem g_2	3.işlem g_3	. . .	m.işlem g_m
C_1	b_{11}	b_{12}	b_{13}		b_{1m}
C_2	b_{21}	b_{22}	b_{23}		b_{2m}
C_3	b_{31}	b_{32}	b_{33}		b_{3m}
. . .					
C_r	b_{r1}	b_{r2}	b_{r3}		b_{rm}

Tablo 2.1 de görülen b_{ji} değerleri, j. çalışanın i. üretim aracını kullanmadaki ***beceri katsayısı***'nı göstermektedir. Beceri katsayısı şöyle hesaplanacaktır: Bir üretim aracını kullanabilen tüm çalışanlara, aynı işlem, birkaç kez yaptırılarak, süre ölçümü yapılacaktır. Ölçüm sonuçlarının normal olasılık kurallarına göre dağılacağı açıktır. Bu ölçümler sonunda, ortalama beceriye sahip bir çalışanın, beceri katsayısı 70 alınıp, diğerleri buna göre değerlendirilecektir. Deneye katılmamış olanların beceri katsayıları sıfır alınacaktır. Beceri katsayısı, üretimin yönlendirilmesi sırasında, çalışanların yerlerinin değiştirilmesi işleminde önem kazanmaktadır.

Beceri katsayısı, işlem süresi içinde yer alan t_o süresini belirleyecektir. Beceri katsayısı 70 olan kişi için bu süre birim süre olarak alınırsa, beceri katsayısı 100 olan bir kişi için, bu süre 70/100 olacaktır. Beceri katsayısı, aynı zamanda tempo faktörü olarakta anılmaktadır. Bir çalışanın, o üretim aracını kullanma temposunu (hızını) belirler. Dolayısıyla (2-5) bağıntısı şöyle yazılacaktır:

$$t_ç = t_h + t_o(70/b) \qquad (2\text{-}6)$$

dolayısıyla (2-4) bağıntısı da şöyle yazılacaktır:

$$t_i = t_y + t_h + t_o(70/b) + t_a \qquad (2\text{-}7)$$

Üretim ortamında çalışanların toplam sayısının (cc), her zaman, üretim aracı sayısından küçük olduğu varsayılacaktır. Bir anda üretim ortamında bulunan çalışan sayısı c ile gösterilecektir. Bir işlem adımı sırasında, üretim aracını kullanan kişinin var olup olmaması ya da rahatsızlanması durumu da sözkonusudur. Bu durum daha sonra ele alınacaktır.

2.3.1 Ürünler - Üretim Araçları ve Çalışanlar Arasındaki İlişkiler

Bir üretim ortamının üç temel öğesinin neler olduğu ve bu öğelerin temel tanımları yukarıda verilmiştir. Bundan sonra bu öğeler arasındaki ilişkiler ortaya konacaktır.

Üretim ortamında, ele alınan zaman aralığında, bulunan ve kullanılan üretim araçları çeşitinin sayısı m (bu sayı işlem sayısına eşit olacaktır), üretilen model sayısı n ise, Tablo 2.2 deki durum ortaya çıkmaktadır. Üretim ortamında, aynı tür üretim aracından birden fazla olabilir. Benzer şekilde, aynı üründen birden fazla üretilebilir.

Tablo 2.2

İşler ve üretim araçları arasındaki ilişkiler

Ürün cinsi	1.işlem g_1	2.işlem g_2	3.işlem g_3	. . .	m.işlem g_m
1. iş (S_1)	t_{11}	t_{12}	t_{13}		t_{1m}
2. iş (S_2)	t_{21}	t_{22}	t_{23}		t_{2m}
3. iş (S_3)	t_{31}	t_{32}	t_{33}		t_{3m}
. . .					
n. iş (S_n)	t_{n1}	t_{n2}	t_{n3}		t_{nm}

Tablo 2.2'ye bakarak, aşağıdaki sonuçlara varılmaktadır:

Tablo 2.2 de görülen t_{ji} işlem süreleri, ortalama beceriye sahip bir çalışana ilişkin sürelerdir; bu nedenle ortalama işlem süresi olarak varsayılacaktır. Bir modelden bir parçanın üretimi için (2-8) deki bağıntı yazılabilir:

$$S_j = \sum_{i=1}^{m} k_{ji} t_{ji} g_i = k_{j1} t_{j1} g_1 + k_{j2} t_{j2} g_2 + k_{j3} t_{j3} g_3 + \ldots + k_{jm} t_{jm} g_m \qquad (2\text{-}8)$$

olarak değişecektir. İşlemler ile çalışanlar arasındaki ilişki tablo 2.1 de verilmiştir. Daha önce değinildiği gibi, işlem süresi çalışanların beceri katsayılarına bağlı, dolayısıyla (2-8) bağıntısında görülen t_{ji} değerleri Tablo 2.1 'e uygun olarak hesaplanacaktır.

Her bir model içinde birden fazla parça olacağı ve bir anda üretim ortamında bulunan sayının d ile gösterileceği daha önce belirtilmişti. Bu nedenle S_j işlemi d_j kadar tekrarlanacaktır. Dolayısıyla (2-8) bağıntısı şöyle yazılabilir:

$$S_j d_j = \sum_{i=1}^{m} k_{ji} t_{ji} g_i d_j \qquad (2\text{-}9)$$

Tablo 2.2 nin bir sütunu, bir işlem için toplam süreyi belirlemektedir. Bir başka deyişle, bir tür çalışma için toplam çalışma süresini belirlemektedir. Bu değer, ele alınan iş için gerekli olan üretim aracı sayısının belirlenmesi için kullanılır. i. işlem için, parça sayıları da gözönüne alınarak (2-10) bağıntısı yazılabilir:

$$T_i = \sum_{j=1}^{n} d_i t_{ji} \qquad (2\text{-}10)$$

T_i i. işlem için gereken toplam süredir. Üretim ortamında, çalışma süresinin sınırlı olduğu bilindiğine göre, T_i toplam süresinin, Z günlük çalışma süresine oranı, i. işlemi gerçeklemek için gerekli olan üretim aracını belirleyecektir. Dolayısıyla i. işlem için gerekli olan üretim aracı sayısı;

$$q_i = \frac{T_i}{Z} \qquad (2\text{-}11)$$

2.3.2 Üretim Araçlarının Sağlamlığı

Her üretim aracının güvenilirliğini belirleyen iki büyüklük vardır; bunlar, iki bozulma arasında geçen ortalama süre (Ortalama Bozulma Süresi : OBS) ve bir bozulmanın giderilmesi için gerekli olan ortalama süre (Ortalama Onarım Süresi : OOS). Üretim ortamında bulunan her üretim aracı için bu iki büyüklük bellidir. OBS değerine bakarak, bir üretim aracının ne zaman bozulacağı, olasılık kurallarına göre hesaplanabilir. Üretim araçlarının bozulması olasılığı, Poisson dağılımına uygun olmaktadır. [16]

$$p(x;\mu) = \frac{e^{-\mu}\mu^x}{x!} \qquad x = 0,1,2,3, \ldots \qquad (2\text{-}12)$$

Bu bağıntıdaki, x Poisson rassal değişkeni olarak anılır ve ele alınan zaman aralığında, olayın kaç kez yineleneceğini belirler. μ ortalama değer olarak anılır. Dolayısıyla, bir üretim aracının, ele alınan zaman aralığında bir kez bozulması olasılığı için

$x = 1$ ve μ = OBS olacaktır. Dolayısıyla (2-12) bağıntısı

$p(1;\mu) = e^{-\mu}\mu$ biçiminde yazılabilir. [17] (2-13)

Bir üretim aracının onarım süresi, normal dağılıma uygun olarak hesaplanabilir. [18]

$$n(x;\mu,\sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} \qquad -\infty < x < \infty \qquad (2\text{-}14)$$

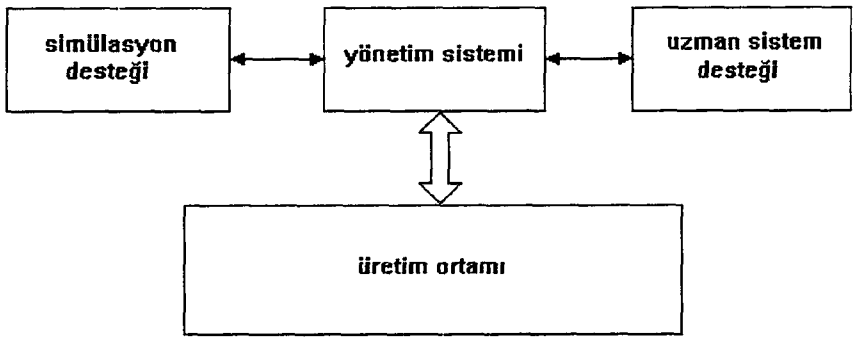
2.3.3 Çalışanların Varlığı

Üretim ortamında çalışan insanlar, hergün çalışmaya gelmemektedirler. Çalışmaya gelen insanlardan bazıları da gün içinde rahatsızlanma ya da kaza nedeniyle çalışamaz duruma düşmektedirler. Ele alınan üretim ortamı biçiminde, uzun yıllar yapılan gözlemler sonunda, günlük gelmeme oranının %5 olduğu saptanmıştır. Çalışma süresi içindeki çalışamama durumu ise ortalama %2 dolayındadır.

2.3 altbölümü içinde, tezin bundan sonraki bölümlerinde kullanılacak olan bazı temel tanımlar tanıtılmıştır. Bu temel tanımlara uygun olarak geliştirilen yöntem 2.4 altbölümünde tanıtılmıştır.

2.4 Uzman Sisteme Dayalı Üretim Yönetimi

Daha önce belirtildiği gibi bu tez çalışmasının amacı, en iyi biçimde düzenlenmiş olan bir üretim dizgesini, önce bilgisayar ile donatmak, ardından uzman sistem yöntemlerinden de yararlanarak yönetmektir. Bu amaca ulaşmak için, önce, üretim ortamından veri toplayacak donanım tasarlanıp kurulacaktır. Üretim ortamından toplanan veriler, yönetim için karar oluşturma sırasında kullanılacaktır. Karar verme sürecinde, uzman sistem yöntemlerinden yararlanılacaktır. Bu tezde, uzman sistemlere ilişkin genel hususlar, bölüm 2.5'te bulunmaktadır. Geliştirilen "Uzman Sisteme Dayalı Üretim Sistemi : USDUS yönteminin ana hatları Şekil 2.4 te verilmiştir.



Şekil 2.4: Uzman sisteme dayalı üretim sisteminin ana hatları ile görünümü

Şekil 2.4 ten de görüldüğü gibi, geliştirilen sistemin altı ana bileşeni vardır:

1. Üretim ortamı
2. Üretim ortamından veri toplama donanımı
3. Üretim ortamına veri iletme donanımı
4. Yönetim sistemi
5. Simülasyon ortamı
6. Uzman sistem desteği

Bu tez çalışması için ele alınan üretim biçimi, partiler halinde yapılan aralıklı üretim yöntemidir ve hedef seçilmiştir. Bu tür üretim yöntemi için geçerli olan bazı temel kurallar aşağıda belirtilmiştir:

1. Üretimin yapıldığı aralık içinde, üretim süreklidir.

2. Üretilmesi planlanan ürünlerin çeşit sayısı nn dir. Ancak, bir anda üretim ortamında yer alan ürünlerin sayısı n dir. Üretim ortamında bir anda bulunacak olan ürün çeşitleri öyle düzenlenmelidir ki, üretim ortamında bulunan araçlar ve insanlar yeterli olsun.

3. Aynı işi yapacak birden fazla üretim aracı bulunabilir. m değişik üretim aracı bulunmasına karşılık her üretim aracı türünden farklı sayıda bulunabilir. Üretim aracı sayısı q_i olarak gösterilmiştir.

4. Üretim araçlarının sayısı, her zaman, çalışanların sayısından fazladır.

5. Bir kişi, birden fazla üretim aracını kullanabilir. Ancak, her aracı aynı beceri derecesi ile kullanamaz.

6. Üretimin akışına uygun olarak, üretim araçlarının sayısı değişebilir. Ancak bu arada, çalışan sayısı sabit olduğundan, toplam üretim aracı sabit kalır. Dolayısıyla, bir çalışan, bir araçtan kalkıp diğerine gidecektir.

Yukarıda, temel ilkeleri belirtilen partiler halinde yapılan aralıklı üretim yönteminde, üretim süreci aşağıda belirtildiği gibi akacaktır:

- Belli bir zaman kesitinde, örneğin bir hafta içinde, üretilecek olan ürünler, giriş kuyruğuna, üretim öncelikleri gözönüne alınarak, sıralı biçimde yerleştirilir.

- Üretim başladığında, giriş kuyruğunda bulunan ürünler, üretim önceliklerine bakılarak, üretim ortamına gönderilir. Bir ürünün, hangi üretim aşamalarından geçeceği, Tablo 2.2 de gösterilmiş ve (2-8) bağıntısı ile formule edilmiştir.

- Bir üretim aşaması tamamlanmış olan bir ürün, bir sonraki üretim aşamasının kuyruğuna girer.

- Her üretim noktasında, üç türlü iş bulunması olasılığı vardır:

  - Yedek iş
  - Üzerinde çalışılan iş
  - Tamamlanmış iş

- Tüm üretim aşamalarından geçmiş olan bir iş, çıkış kuyruğuna alınır ve buradan da depoya gönderilir.

Yukarıda açıklanan temel ilkelere bağlı olarak, bu tez çalışması ile önerilen USDUS yönteminin nasıl işleyeceği aşağıda adım adım anlatılmıştır:

2.4.1 USDUS Yönteminin Temel İlkeleri

- Yöntemin çalışabilmesi için ilk adım, veri toplama dizgesinin kurulmasıdır. Veri toplamak üzere her üretim aracı üzerine bir veri giriş terminali yerleştirilecektir. Veri giriş terminali adı verilen bu birimlerin donanımı, çalışması ve tüm sistem ile bağlantısı 3. Bölümde anlatılmıştır.

- Çalışanlara, belli mesajların iletilebilmesi için, her üretim aracının üzerinde, bir adet bilgi gösterim terminali bulunacaktır. Bilgi gösterim terminallerinin donanımı, çalışması ve tüm sistem ile bağlantısı 3. Bölümde anlatılmıştır.

- Üretim ortamında yer alan tüm üretim araçlarından belli bir kısmı, ele alınan zaman kesiti içinde kullanılmaktadır. Hangi üretim aracından kaç tane gerektiği, çalışanların ortalama yetenekli oldukları varsayımı ile, Tablo 2.2 ye bakılarak hesaplanacaktır. Ardından, her üretim aracı için bir çalışan atanacaktır. Böylece üretim ortamının resmi (kullanılan üretim araçları, yerleşme düzeni ve her araca atanmış olan kişi) ortaya çıkacaktır. Bu resim, yönetim sistemi tarafından bilinmektedir.

- Kurulan üretim dizgesi, simülasyon ortamında denenecektir. Bu deneme sonunda, ortalama değerler üzerine kurulmuş olan üretim dizgesi, stokastik süreç anlamında denenmiş olacaktır. Simülasyon sürecinde, işlem süreleri, üretim araçlarının bozulması ve çalışanların işi bırakması gibi olasılıklar gözönüne alınacaktır. Simülasyon süreci sonunda, üretim araçlarının sayılarında değişme olabilir. Bu durum üretim ortamına, aktarılarak gerekli düzeltme yapılır ve üretim süreci başlar.

- Üretim süreci başladıktan sonra, tüm üretim araçları izlenir. Bu izlemeler sırasında;

  - Bir üretim aracının sağlam olup olmadığı,
  - Arıza durumunda ise, onarımın tamamlanıp tamamlanmadığı,
  - Ne iş yaptığı,
  - Harcadığı sürenin, daha önce tanımlanmış olandan faklı olup olmadığı,
  - Çalışanın elinde iş olup olmadığı,
  - Çalışanın yedek işi bulunup bulunmadığı,
  - Çalışanın tamamlamış olduğu işin bir sonraki iş istasyonu için kuyruğa alınıp alınmadığı

- Üretim sürecinde izlenen bir başka nokta, kuyruklardır. Daha önce değinildiği gibi, üretim ortamında üç sınıf kuyruk sözkonusudur.
  - Giriş kuyruğu
  - Ara kuyruk
  - Çıkış kuyruğu

- Üretim ortamının en iyi biçimde düzenlendiği, ara kuyrukların oluşmaması ve iş bekleyen üretim noktası bulunmamasından anlaşılacaktır.

Ara kuyrukların oluşması ya da bir üretim noktasının iş bekler duruma düşmesi, üretim ortamının yeniden düzenlenmesinin gerektiğini, ortaya koymaktadır.

Ara kuyruğun oluşması, bir sonraki üretim noktasında olabilecek şu nedenlerden kaynaklanabilir:

- Üretim araçlarında bozulma ya da arıza olabilir.
- Çalışan kişilerden biri, bir nedenle işi bırakmış olabilir.
- Çalışan kişilerin beceri katsayıları, bir nedenle düşmüş olabilir.
- Üzerinde çalışılan üründe bir sorun yaşanıyor olabilir.

Ara kuyruğun oluşması, bir önceki üretim noktasında olabilecek şu nedenden de kaynaklanabilir:

- Çalışan kişilerin beceri katsayıları, öngörülenden yukarı çıkmış olabilir.

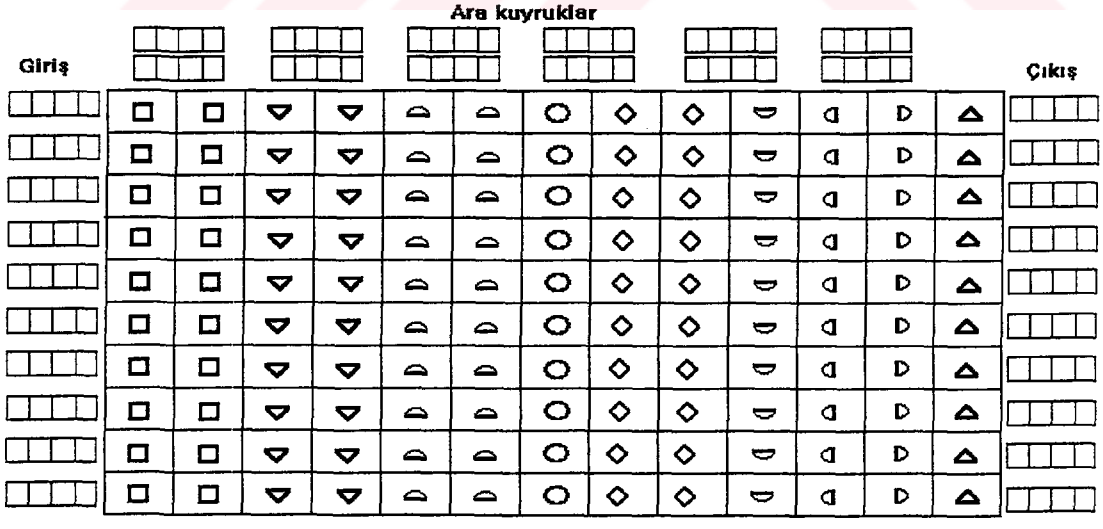
Yeni iş bekleme ya da işsiz kalma durumunun oluşması, bir önceki üretim noktasında olabilecek şu nedenlerden kaynaklanabilir:

- Üretim araçlarında bozulma ya da arıza olabilir.
- Çalışan kişilerden biri, bir nedenle işi bırakmış olabilir.
- Çalışan kişilerin beceri katsayıları, bir nedenle düşmüş olabilir.
- Üzerinde çalışılan üründe bir sorun yaşanıyor olabilir.

Yeni iş bekleme ya da işsiz kalma durumunun oluşması, işsiz kalan üretim noktasında olabilecek şu nedenlerden kaynaklanabilir:

- Çalışan kişilerin beceri katsayıları, öngörülenden yukarı çıkmış olabilir.

Üretim ortamının sembolik bir görünümü Şekil 2.5 te verilmiştir. Burada gösterilen her bir sembol grubu, farklı makina gruplarını ve şeklin kenarlarında ve üstünde bulunan 4 gözlü kutular sepetleri temsil etmektedir.



Şekil 2.5: Üretim ortamının sembolik gösterimi

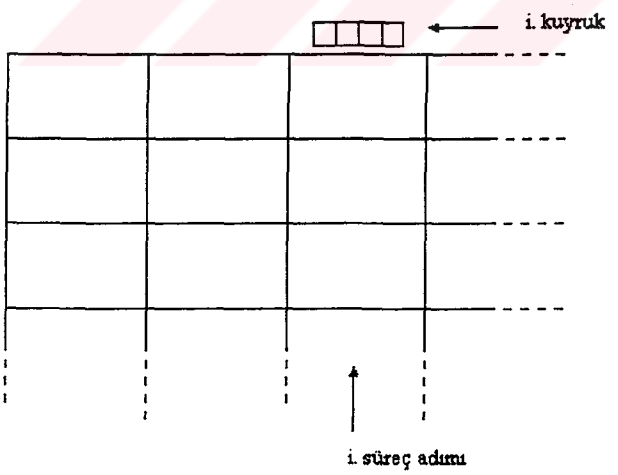
Üretim süreci içinde, ara kuyrukların oluşması ve işsiz kalınması, istenmeyen durumlardır. Bu durumların giderilmesi, ancak bazı çalışanların çalışma yerlerinin değiştirilmesi ile mümkündür.

Bir ara kuyruğun oluşması, bir sonraki üretim aşamasına, iş bekleyen bir çalışanın kaydırılması ile giderilebilir. Ancak, konu bu kadar basit değildir. Çünkü, bir çalışan bir noktadan diğer bir noktaya kaydırıldığında, tüm üretim sürecinin dengesi bozulacaktır.

Bu tez çalışması ile getirilen önemli bir katkı bu aşamada görülmektedir ve bu konunun çözümü için uzman sistem yöntemlerinden yararlanılmaktadır.

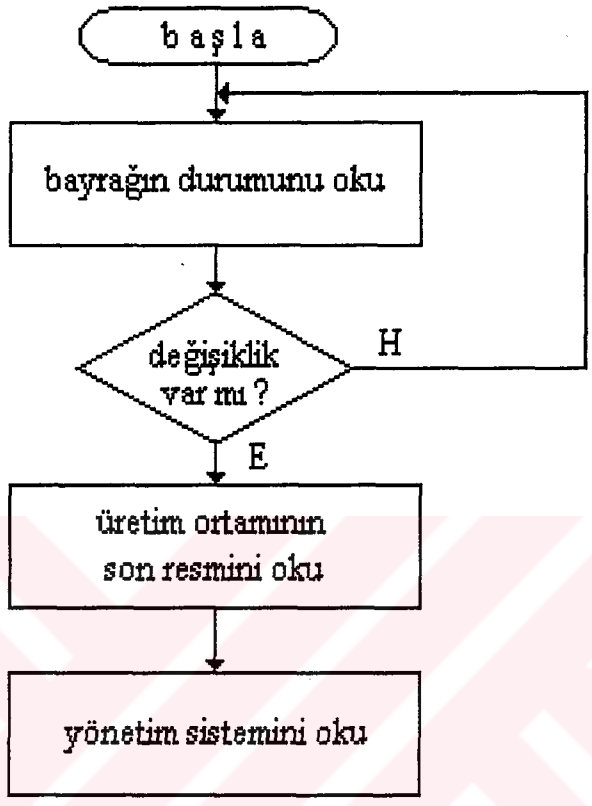
Bu çalışma ile önerilen, uzman sistem yaklaşımının ana hatları akış diyagramlar biçiminde gösterilmiştir.

Bu diyagramlarda kullanılan notasyonlar Şekil 2.6 da gösterilmiştir.



Şekil 2.6: Uzman sistem algoritmalarında temel alınan notasyon

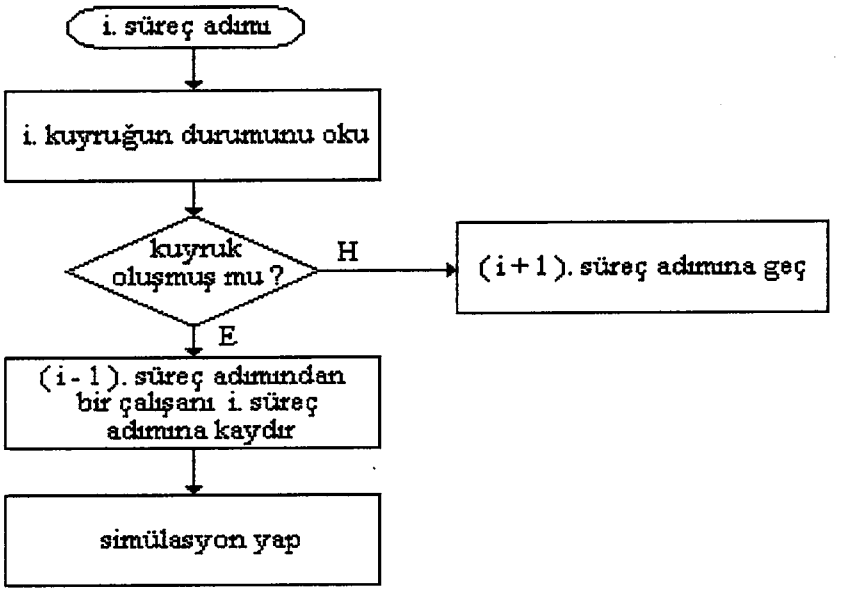
Uzman sistem ile ilgili algoritmalar Şekil 2.7, Şekil 2.8 ve Şekil 2.9 da verilmiştir.



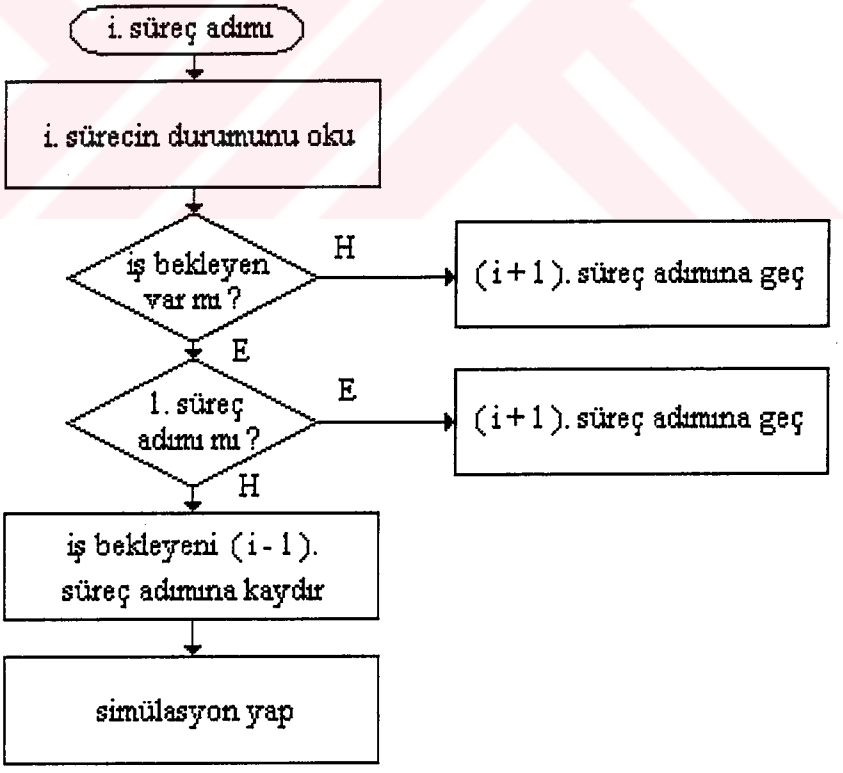
Şekil 2.7: İzleme süreci

- Uzman sistem yöntemlerinin katkılarıyla, üretim ortamında gerekli kaydırmaların nerelerde yapılması gerektiği ortaya konmaktadır.

- Böylece ortaya çıkan yeni resim, üretim ortamına aktarılmadan önce, simülasyon ortamında irdelenmekte ve eğer, sonuç olumlu ise, üretim ortamında gerekli değişikliklere gidilmektedir.



Şekil 2.8: Kuyruk sorununun çözümü için uzman sistem yaklaşımı



Şekil 2.9: İş bekleme sorununun çözümü için uzman sistem yaklaşımı

2.5 Uzman Sistem Yaklaşımı

Uzman, eğitimi ve deneyimi sonucu; pek çok kişinin yapamayacağı işleri yapabilecek duruma gelmiş olan kişidir. Yaptığı işte becerili olma dışında verimli ve düzenlidir. Uzman pek çok ilgisiz bilgi ve veri arasından sorunların çözümüne yarayacak olanları seçebilir. Bunun için, büyük bir ustalık ve bilgi birikimine sahiptir.[19] Son yıllarda, yapay zeka araştırmacıları "*bir programı akıllı yapabilmek için, çalışma ortamına olabildiğince nitelikli ve özel bilgilerin sağlanması gerekir*" ilkesini benimseyerek, bu yaklaşıma dayalı sistemlere, uzman sistem adını vermişlerdir. [20]

Uzman sistemlerin faydaları şöyle özetlenebilir:

- **Uzman Katkısı :** Bir konuda, en üst düzeyde bilgi ve beceriye sahip olan uzmanın, sisteme sağlayacağı destek, tabiatıyla bu alandaki olayların çözümünde en uygun ve en verimli yardımcı olacaktır.

- **Öngörme ve Modelleme :** Sistemin modellenmesi ve ardından simülasyonu sayesinde, sistem içinde ileride ortaya çıkabilecek durumlar önceden öngörülmektedir.

- **İşletmede Süreklilik :** İşletmede çalışan elemanların, bilgi ve beceri birikimlerini derleyerek; daha sonra bu bilgilere kolayca erişilebilmeyi sağlar. Elemanlar, işletmeden ayrılsalar dahi, bu bilgiler işletme içinde kalır.

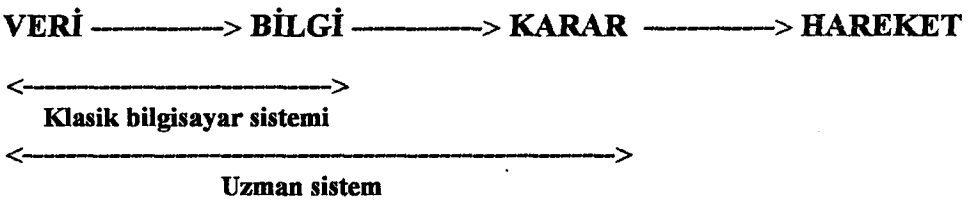
- **Eğitim Desteği :** Uzman sistem sayesinde, işletmeye yeni giren deneyimsiz elemanların eğitilmesi işini kolaylaştırır.

2.5.1 Uzman Sistemlerle, Klasik Bilgi Sistemleri Arasındaki Farklar

Uzman sistemler, klasik bilgi sistemlerine göre aşağıda belirtilen katkıları sağlarlar:

- Kullanıcının iş yükünü hafifletir.
- Kullanıcıya, karar verme sürecinde yardımcı olur. Hatta; kararı bilgisayar oluşturur, kullanıcı onaylar.

Aşağıdaki şekilde, sistemler arasındaki fark sembolik olarak gösterilmiştir. [21]



Şekil 2.10: Bilgi tabanlı haberleşme sisteminin, bilgisayar sistemi ile mukayesesi

Uzman sistemlerde, diğer bilgi sistemlerinden farklı diller kullanılır. Bunlar genelde, **Lisp** (1950'lerden sonra geliştirilen, sembole dayalı) ve **Prolog** (lojik tabanlı sistem) dilleridir.

2.5.2 Uzman Sistemlerin Temel İlkeleri

- **Deneme Yanılmaya Dayalı Uzmanlık :** Günlük yaşamda bir çok bilgi, deneme yanılma yoluyla öğrenilir ve sonuçta elde edilen bilgiler kesin bilgiler olmayabilir. Ancak insan, benzer olaylar karşısında, benzer kararlar üretir.
- **Açıklayabilme :** Düşünceyi açıklayabilme ve bilgi tabanına yöneltilen sorulara cevap verebilme özelliğidir.
- **Esneklik :** Veri tabanındaki mevcut veri ile, yeni elde edilen verileri ilişkilendirebilme özelliğidir.

2.5.3 Uzman Sistemlerin Kullanıldığı Alanlar

- **Tasarım :** Tasarım anında, sistemin oluşturulması için gereken yapı elemanlarının seçilmesi ve biribirleri ile bağlanması esnasında, karar verme görevini üstlenirler.
- **Hata Bulma ve Onarım :** Sistemde hataya neden olan parçaları tespit edip, yapılması gereken değişiklikler veya alınması gereken önlemler konusunda, kullanıcıya yardımcı olmak.

- **Denetim :** Sistemden gelen verileri sürekli olarak yorumlayıp, alınması gereken önlemler konusunda önerilerde bulunmak.

- **Öğretim :** Sistem, kullanıcıların bir bilgi alanında eğitilmesi işini üstlenir.

2.5.4 Uzman Sistemlerde Bilgi Gösterimi

Bilgi gösteriminde, üç temel öğe vardır. Bunlar :

- **Genişleyebilme :** Veri yapıları, veri tabanı genişletilirken, büyük değişiklikler gerektirmeyecek kadar esnek olmalıdır. Veri tabanı, uzmanların deneyimlerine dayanan bilgiler içermektedir.

- **Basitlik :** Esnekliğin sağlanabilmesi için, veri yapılarının, basit ve bir düzende olması gerekecektir. Böylece, erişim işlemlerinde, kolaylık sağlanır.

- **Kapsama :** Bilgi tabanındaki uzman bilgisinin, sorunu çözmeye yetip-yetmeyeceği ile ölçülür.

Uzman sistemde bilgi gösterimi için, üç çeşit gösterim yöntemi kullanılır.

- Kurala dayalı,
- Çerçevelere dayalı,
- Mantığa dayalı

Kurala Dayalı Gösterim Yöntemi

Uzman sistemler, daha genel bir bilgi işlem yöntemi olan üretim sistemlerinden türetilmiştir. Klasik bir üretim sistemi üç temel öğeden oluşmaktadır.

- Çözümü aranan üretim ortamının, özelliklerini içeren veri tabanı,
- Üretim ortamı hakkında, genel bilgileri içeren bir kural tabanı,
- Çözüm süreçlerini gerçekleştiren yorumlayıcı

Kural yorumlayıcı, hangi kuralın uygulanacağına karar vermekle yükümlüdür. Yorumlayıcı ise, seçilen kuralın koşullarının, veri tabanıyla nasıl ilişkilendireceğinin kararını verip, sorun çözümü sürecini yönetir.

Çerçevelere Dayalı Gösterim Yöntemi

Çerçevelere dayalı bilgi gösteriminin temeli; insanların, tanıdık ve bildikleri durumlar karşısında, benzer davranışlar gösterdiklerini savunan, psikoloji kuramlarına dayanır. Marvin Minsky'in geliştirdiği bu yöntem; tanımların ve yordamların, tek bir bilgi gösterim ortamında birleştirilmesi amacıyla ortaya konmuştur.

Bu yöntemde bilgi, çerçeveler içinde tanımlanmış bölümler ve bu bölümlere atanan değerlerle gösterilir. Bu yöntemin en belirgin özelliği; biribirlerine, alt-üst ilişkileriyle bağlı çerçeveler yoluyla, nesne gruplarının, sınıflandırılmış bir gösteriminin elde edilmesine imkan vermesidir. Yani, bir sınıfa ait olan çerçevenin, bu sınıfın ortak bölüm tanımlarını ve değerlerini doğrudan içermesidir. Bu yöntemin bir diğer avantajı; benzerliğe dayalı çıkarımdır. Yani, aralarında doğrudan ilişki bulunmadığı halde, göze tanımları birbirine benzeyen çerçeveler arasında bilgi aktarımına imkan vermesidir.

Mantığa Dayalı Gösterim Yöntemi

Bu yöntemde bilgi, basit veya karmaşık önermelerle ifade edilir. Çerçevelerle ifade edilebilen bilgiler, mantığa dayalı yöntemle de ifade edilebilir.

2.5.5 Uzman Sistemlerde Çıkarım Yöntemleri

Çıkarım, bilgi tabanındaki bilinenler ve kurallar arasında bir bileşke ortaya çıkartmaya çalışır. Yani, temel anlamıyla sezgisel bilgiye dayalı bir arama yöntemidir. Bu arama yöntemi her zaman mükemmel çözümlere ulaşamayabilir, ancak arama alanını ve zamanını azaltabilir. Çıkarım sırasında kullanılan denetim yolları aşağıda verilmiştir:

1. **İleriye Doğru Denetim :** Öncelikle, sorunla ilgili bilgilerin veri tabanına girilmesi gerekir. Kurallar, koşulları sağlandığında hemen işletilir. İleriye doğru denetim yönteminin aşamaları aşağıda gösterilmiştir.

    - **Eşleme :** Bilgi tabanında, koşulları sağlayan bütün kurallar belirlenir.

- **Seçme :** Eşlemeden geçen kuralların işletilecek olanları seçilir. Çeşitli seçme yöntemleri vardır.
  - sırayla işlet
  - tek birini işlet
  - son eşleneni işlet
  - en özelini işlet
  - tümünü işlet
- **Ekleme :** İşletilen kuralların sonuçları, bilgi tabanına eklenir.

2. **Geriye Doğru Denetim :** Bir amaca ulaşılmış olması, bu amaca ulaşmak için gerekli olan kuralların sağlanmış olduğunun kanıtıdır.
3. **İleri-Geri Denetim :** Yukarıda tanıtılan ileriye ve geriye doğru denetim yöntemlerinin, iyi yönlerinin kullanıldığı bir yöntemdir.

2.5.6 Uzman Sistem Geliştirme Aşamaları

Uzman sistem gelişimi, biribirleriyle ilişkili evrelerden oluşur. Bunlar;

- **Tanımlama :** Bu evrede uzman, sorun alanının temel özelliklerini belirler. Tanımlama evresi iki altevreye ayrılabilir. Bunlar;
  - Olabilirlik Çözümlemesi
  - Yapısal Çözümleme'dir.
- **Kavramsal tasarım :** Bu evrede, sorun çözümünü tanımlayacak ilişkiler ve denetim düzenekleri belirlenir.
- **Biçimlendirme :** Seçilmiş olan uzman sistem geliştirme aracının sorun alanına uygun olup olmadığı ortaya çıkar ve gerekirse bir önceki evreye geri dönülür.
- **Gerçekleştirme :** Bu evrede, biçimlendirmiş bilgi, izlenceye dönüştürülür.
- **Sınama ve Doğrulama :** Bu evrede, izlencenin kullanılabilirliği, performansı, sorun çözümündeki doğruluğu sınanır. Sınama sonucu; sistemin doğruluk yüzdesi ve güvenilirliği ortaya çıkartılır.

2.6 USDUS'de Kullanılan Uzman Sistem Yöntemleri

Bilinen ve yukarıda özetlenen uzman sistem yaklaşımlarından, topluca üretim için uygun olanlar, USDUS içinde kullanılmıştır. USDUS için yararlanılan uzman sistem özellikleri aşağıda verilmiştir:

Uzman sistem katkıları:

- **Uzman katkısı :** Ele alınan üretim fabrikasında, konusunda uzman olmuş kişilerle görüşülmüş ve gerek üretim ortamı, gerekse üretim araçları hakkında geniş bilgi toplanmıştır. Bu bilgiler, daha sonraki aşamalarda kullanılmıştır.

- **Öngörme ve modelleme :** USDUS'nin temeli, üretim ortamını en iyi şekilde modellemek ve bu modele dayalı olarak simüle etmektir. Üretim ortamı sürekli izlenmektedir. Bu arada, ortaya çıkan eşik değerlerini aşmış durumlar, ortaya çıkarılmaktadır. Bu durumlarda, uzman görüşleri ışığı altında üretim ortamında değişiklik yapılmasına karar verilmektedir. Ancak, USDUS de bu karar uygulanmadan önce, sistemin tamamı simüle edilmekte ve böylece sistemin geleceği öngörülmektedir. Bu simülasyon sonunda, uzman görüşüne dayalı olarak alınmış olan kararın doğruluğu irdelenmektedir.

- **İşletmede süreklilik :** İşletmede çalışan elemanların, bilgi ve beceri birikimleri bilgisayara aktarıldığı için, uzmanlardan bağımsız bir sisteme ulaşılmıştır.

- **Eğitim desteği :** Geliştirilen simülasyon yazılımı ile, işletmeye yeni giren deneyimsiz elemanların eğitilmesi sağlanmıştır.

2.6.1 USDUS'de Bilgi Gösterimi

Daha önce anlatıldığı gibi, uzman sistemlerde üç türlü bilgi gösterimi sözkonusudur:

- Kurala dayalı,
- Çerçevelere dayalı,
- Mantığa dayalı

USDUS da bu gösterimlerden, kurala dayalı yöntem kullanılmıştır. Bu bağlamda,

- Üretim ortamının, özelliklerini içeren veri tabanı oluşturulmuştur.
- Üretim ortamı hakkında, genel bilgileri içeren bir kurallar dizisi oluşturulmuştur.
- Çözüm süreçlerini gerçekleştiren yöntemler geliştirilmiş ve bu yöntemlere uygun yazılımlar gerçeklenmiştir.

2.6.1.1 USDUS'de Bilgi Çıkarım Yöntemi

USDUS de bilgi çıkarım yöntemi olarak ileriye doğru denetim yöntemi kullanılmıştır. Bu yönteme göre;

Önce, üretim ortamına ilişkin bilgiler veri tabanına girilmiştir. Ardından eşleme, seçme ve ekleme işlemlerine geçilmiştir.

2.7 USDUS'nin Çalışması

Şekil 2.7, Şekil 2.8 ve Şekil 2.9 da akış diyagramı olarak tanıtılan, uzman sistem yaklaşımı üzerine açıklamalar aşağıda verilmiştir.

Sistem çalıştırılmaya başlamadan önce, yönetim sistemine aşağıdaki veriler yüklenecektir:

- Üretilecek ürün çeşitleri
- Üretim planı
- Seçilmiş üretim makineleri
- Seçilmiş çalışanlar

Sistem işletmeye alındığında, her şeyin beklenen düzeyde olduğu varsayılacaktır. Ancak periyodik olarak üretim sistemi gözlenecektir. Üretim ortamının gözleme periyodu (t_g), örnekleme kuramına bağlı olarak hesaplanabilir. Üretim sisteminin gözleme periyodunun nasıl hesaplanacağı 2.7.1 de açıklanmıştır.

Sistemin gözlenmesi sırasında, gözlenen durumlar şunlardır:

- Kuyrukların durumu
- Makinelerin durumu
- Çalışanların durumu
- Ürün besleme magazinlerinin durumu
- Ürün toplama magazinlerinin durumu

Gözlenen durumların bazıları mantıksal bazıları ise seviye biçimindedir. Aşağıda gözlenen durumların özellikleri açıklanmıştır:

Kuyruklar : Seviye biçiminde; örneğin kuyrukta 3 sepet iş bekliyor

Makineler : Bozuk, tamirde, çalışıyor, iş bekliyor

Çalışanlar : Hasta, kısa süreli izinli, çalışıyor, iş bekliyor

Besleme magazini : Boş, dolu

Toplama magazini : Boş, dolu

Her gözleme sonucu, yönetim sisteminde tutulacaktır. Gözleme sonunda elde edilen durum, üretim ortamının resmi olarak anılacaktır. Bir gözleme sonucu, bir önceki gözleme sonucundan farklı ise, yönetim sistemi çalışmaya başlayacaktır.

Şekil 2.8 den görüldüğü gibi, gözleme sonunda, i. süreç adımında bir kuyruk oluştuğu saptanabilir. Bu durumda, kuyruğun oluştuğu adımdan bir sonraki adımda (i+1) çalışmanın beklenen hızda yürümediği ortaya çıkmaktadır. Kuyruğu yok etmek için iki seçenek vardır:

1. Bir önceki süreç adımında (i-1) kuyruğu oluşturan çalışmayı durdurmak. Bir başka deyişle, bu işte çalışanı kaldırıp bir sonraki süreç adımındaki makinelere kaydırmaktır. Ancak bu çözüm benimsenmeden önce 2. seçenek düşünülmelidir.

2. i. süreç adımında, makine sayısını artırarak, kuyruğu azaltmak. Bu çözüme ulaşmak için, herşeyden önce, sürdürülen işi gerçekleyebilecek bir üretim makinesinin, i. süreç adımında bulunması gerekir. Bu durum mevcut ise, i. adımdan sonra gelen bir süreç adımından, bir çalışan alınıp i. süreç adımına kaydırılabilir.

Yukarıda verilen iki seçeneğin birinin seçilmesi durumunda, kuramsal olarak çözüm bulunacağı söylenebilir. Ancak, bir çalışanın, ileri ya da geri süreç adımına kaydırılması, tüm üretim sisteminin dengesini bozabilir. Bu denge bozukluğu, tüm sistemin akışını kötü yönde etkileyebilir. Bu nedenle, hedeflenen değişikliği yapmadan önce, simülasyonunun yapılması USDUS yöntemi ile önerilmektedir. Dolayısıyla, uygulamaya geçmeden önce, değişiklik simülasyon çalışması ile irdelenmeli ve eğer, sonuç tatminkar ise uygulamaya konmalıdır.

2.7.1 Üretim Ortamının Gözlenme Periyodu

Frekansı f olan harmonik bir değişimin örnekleme frekansının en az 2f olacağı örnekleme kuramı ile verilmiştir [22]. Üretim ortamında, gözlenmesi gereken ve birbirinden farklı hızlarda değişen olaylar bulunmaktadır. Ancak bu değişimlerden en hızlı olanına göre bulunacak örnekleme zamanı, diğerleri için de geçerli olacaktır.

Yukarıda ortaya konan kuramsal sonuçlara göre, ilk olarak, üretim ortamında en hızlı gözlenmesi gereken işlem belirlenmelidir. Bu işlemin süresi t ise, örnekleme zamanı; bu sürenin yarısı olarak seçilmelidir.

3. Bölüm

Uygulama

2. Bölümde, temel yapısı ve kuramsal yaklaşımları verilmiş olan USDUS'nin uygulanmasına yönelik çalışmalar bu bölüm içinde yer almıştır.

USDUS'nin temel yapısı içinde yer alan ve uygulanacak olan konular aşağıda sıralanmıştır:

1. Üretim Ortamı
2. Simülasyon destek sistemi
3. Uzman sistem desteği
4. Yönetim sistemi

Yukarıda adları verilen dört konu, bu bölüm içinde ele alınmıştır.

Daha önceki bölümlerde ayrıntıları verilen üretim ortamından verilerin toplanabilmesi için DBD (Dağıtılmış Bilgisayarlar'la Denetim) sistemine benzer bir yöntem uygulanmıştır. Zira, ele alınan üretim ortamında çok sayıda üretim aracı bulunmakta ve bir ürün, birden fazla üretim tezgahlarında işlem görerek oluşmaktadır. Bu üretim sürecinde, üretim ortamı ve de dolayısıyla üretim araçları ile ilgili bilgileri öğrenebilmemiz için sürekli olarak üretim ortamının gözlenmesi gerekmektedir.

Bunun için, üretim ortamındaki her üretim aracına, bir bilgisayar terminali yerleştirilmesi hedeflenmiştir. Yani tasarlamaya çalıştığımız yapı, birden fazla bilgisayarlı denetim dizgesini içeren ve bilgisayarlar arasında, bilgi iletişim bağlantısı bulunan, dağıtılmış bilgisayarlı bir dizgedir.

Üretim araçlarına eklenen terminallere "**yamak**" ve tüm üretim ortamını gözleyen bilgisayara "**usta**" bilgisayar adı verilmiştir.

Yamak bilgisayarın görevi, üzerinde bulunduğu üretim aracının durumunu belirlemek ve çalışana bilgi aktarmaktır. Dolayısıyla işlevi ana hatları ile şöyledir:

- Yedek ürün var/yok
- İşlenen ürün var/yok
- İşlenmiş ürün var/yok
- Tamamlanmış ürün var/yok
- Üretim aracı sağlam/arızalı/onarımda
- Üretim aracında çalışan var/yok
- Üretim aracında çalışan kişinin kimliği
- O anda işlenen ürünün kimliği
- Çalışanların ürettiği miktarları göstermek
- Çalışana yönetim bilgisi sunmak

Tüm üretim ortamının yönetiminden sorumlu olan usta bilgisayarın temel görevleri ise şöyledir:

- Üretim ortamını gözlemek
- Ürün akışını gözlemek
- Uzman sistem yaklaşımı ile üretim ortamını yönetmek
- Çalışanların değerlendirmesini yapmak
- Durum raporu hazırlamak

3.1 Üretim Ortamı

Üretim ortamı başlığı altında, üretim ortamı için gerekli olan veri iletişimi teknikleri ve bir üretim ortamının, bilgisayar açısından tanıtımına yer verilmiştir.

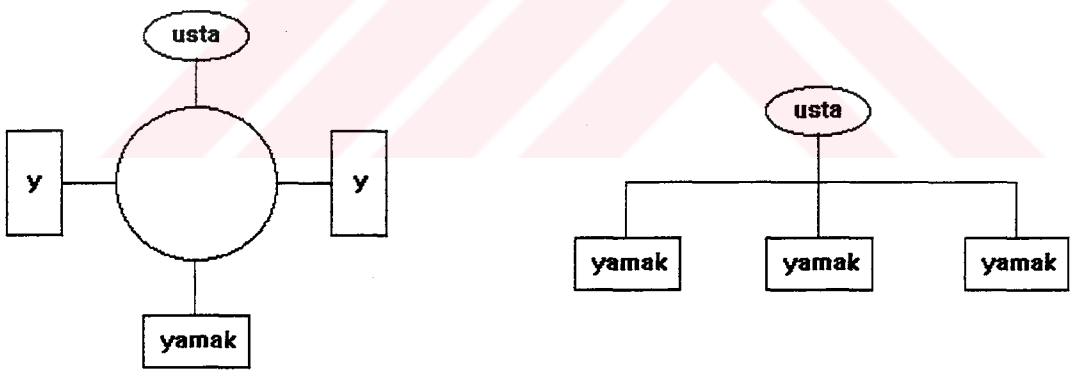
3.1.1 Veri İletişimi (Veri Toplama ve İletme)

Yukarıda değinildiği gibi, üretim ortamında, dağıtılmış bilgisayarla denetim mimarisine uygun bir dizge kurulacaktır. DBD dizgelerinde iletişim ortamı iki türlü kurulabilmektedir: [23]

1. Halka biçimi
2. Yıldız biçimi

DBD dizgelerinin üçüncü kuruluş biçimi halka ve yıldız biçimlerinin birleşmesinden oluşan karma biçimdir.

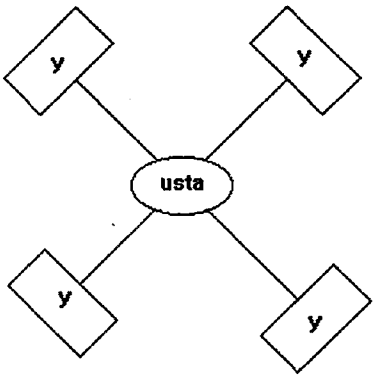
Halka biçimi iletişim ortamının kuruluşu iki türlü olabilir. Bunlar kapalı halka ve açık halka biçimleridir. Şekil 3.1'de kapalı halka ve açık halka biçimi iletişim ortamı gösterilmiştir.



Şekil 3.1: Kapalı ve açık halka biçimi iletişim yöntemi

Halka biçiminde, bilgisayarlar arasındaki bilgi iletişimi aynı yol üzerinden sağlanır. Bir başka deyişle, tüm yamaklar ve usta bilgisayar aynı iletişim yoluna bağlanmıştır.

Yıldız biçiminde usta bilgisayar, yamak'ların ortasında yer alır. Durum Şekil 3.2 de gösterilmiştir. Bu kuruluş biçiminde, halka biçiminden farklı olarak, herbir yamağın kendisine ayrılmış bir iletişim hattı ile ustaya bağlandığı görülür.



Şekil 3.2: Yıldız biçimi iletişim yöntemi

3.1.1.1 Halka ve Yıldız Biçimi DBD'lerin Karşılaştırılması

Halka ve Yıldız biçimi iletişim ortamlarının özelliklerinin tanıtılması ve karşılaştırılması beş ayrı açıdan yapılacaktır. Böylece, iki yöntemin biçiminin biribirine göre üstünlüklerinin, nasıl ve nereden doğacağı vurgulanmış olacaktır.

1. **Ara Bağlantılar Yönünden:** Halka biçimi, yıldız biçimine oranla daha az ara bağlantısı gerektirir. Zira, halka biçimi uygulamada tüm yamaklar ve usta için ortak bir iletişim yolu bulunur iken, yıldız biçiminde her yamak için ayrı bir iletişim hattı ayrılmıştır. Usta ve yamağın biribirlerinden uzak bulunduğu uygulamalarda (petrol rafineleri, demiryolu taşımacılığı ve trafik denetimleri gibi) kablo yatırım giderlerini azaltmak, sistemin maliyeti açısından büyük önem taşır.
2. **İletişim Yolunun Donanımı Yönünden :** Halka biçiminde usta ve yamaklar arasında tek bir iletişim yolunun bulunması, istenen bir yamağa ulaşmada bazı yan birimler gerektirecektir. Bu yan birimler Adres Çözücü ve Protokol Denetleyicisi (AÇPD) olarak anılacaktır. Oysa, yıldız biçiminde, usta bilgisayarın doğal olarak donanımında bulunan giriş-çıkış arabirimleri, usta yamak arası iletişim için yeterlidir.

Usta bilgisayarın giriş-çıkış arabirimlerinin sayısı az ise, bunu arttırmak, bir başka deyişle, yıldız biçiminde yeni bir yamağın eklenmesinin getireceği ek yatırım; halka biçiminde aynı işlem için gerekenden 5-10 defa daha ucuzdur.

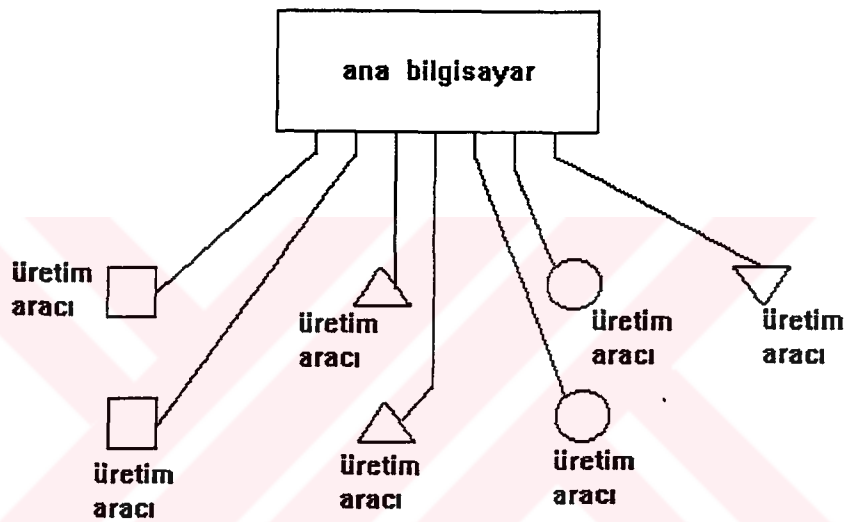
3. **Esneklik Yönünden :** Halka biçiminde yeni bir yamak veya gözleme aygıtı iletişim yolunun herhangi bir noktasına kolayca bağlanabilir. Bu işlem, tarlaları boydan boya geçen su kanalının istenen her noktasından su alabilmeye benzetilebilir. Halka biçiminde iletişim yoluna bağlanabilecek yamakların sayısı, ustanın adresleyebilme yeteneği ile sınırlıdır. Yıldız biçiminde ise yamakların sayısı, usta bilgisayarın giriş-çıkış birimi sayısı ile sınırlıdır. Her iki yöntemde de adreslenebilecek yamak bilgisayar sayısı, bu tezde, örnek üretim ortamı olarak ele alınmış olan sanayi kolları için, bir sınır teşkil etmemektedir. Bağlanabilecek yamak bilgisayar sayısı, yüzbinlerin üzerindedir. Yıldız biçiminin esneklik açısından en büyük kusuru; her yeni yamak için, yeni bir iletişim hattını gerektirmesidir. Bu tezde sözü edilen üretim ortamının boyutları kablo maliyetlerini aşırı arttırmayacak seviyede olması nedeniyle ve ortamda sürekli hareket halinde olan bazı taşıma araçlarının mevcudiyeti, bizi yıldız biçimi bir donanım seçmeye yöneltmiştir.

4. **İletişimin Yazılımı Yönünden :** İletişim yazılımı, yıldız biçimi için, halka biçimine oranla daha kolay olacaktır. Bunun nedeni, iletişim hattında ek aygıt gerektirmemesidir.

5. **Güvenirlik Yönünden :** Bir sistemde, bir yamağın veya ustanın aksaması durumunda, diğer birimlerin etkilenmemesi temel koşuldur. Yıldız biçiminde her yamak, özel bir iletişim hattı ile ustaya bağlandığından, herhangi bir yamağın aksaması veya bu yamağa ilişkin iletişim hattının aksaması ya da kopması veya kısa devre olması, ne ustayı ve ne de diğer yamakları etkiler. Halka biçiminde, herhangi bir yamağın aksamasının tüm dizgeye yansımaması, alınacak önlemlerle sağlanabilir. Ancak, iletişim yolundaki bir kopukluk, kopukluğun ötesinde kalan yamaklarla usta arasındaki iletişimi keser. Bundan daha kötüsü, iletişim yolundaki bir kısa devre dizgedeki tüm iletişimi aksatabilir.

Bu nedenledir ki, halka biçiminde iletişim yolu kesinlikle desteklenmelidir.

Yukarıda ayrıntıları verilen iki sistemden, yıldız biçimindeki iletişim ortamı ele alınan uygulamaya daha uygun düşmektedir. Zira, bu tezde sözü edilen üretim ortamının boyutlarının küçük olması nedeniyle, kablo maliyetlerini aşırı artırmayacaktır. Diğer taraftan, üretim ortamında sürekli hareket halinde olan bazı taşıma araçlarının mevcudiyeti, ortalıkta dolaşan kablo donanımının emniyetle korunmasını zorlaştıracağından dolayı, yıldız biçimi bir donanımın seçilmesi gerekmiştir. Bu düşüncelerin ışığında gerçeklenmiş olan veri iletişim ortamı Şekil 3.3 te gösterilmiştir.



Şekil 3.3 : Üretim ortamının donanımının şematik gösterimi

3.1.2 Üretim Aracı

Bir üretim aracı aynı işlevi gören makine grubu içinde konumlandırılmıştır. Üretim aracının her iki yanında ikişer sepet bulunmaktadır.

- Yedek sepet
- İşlenecek sepet
- İşlenmiş sepet
- Bitmiş sepet

Yedek Sepet

İçerisinde işlem görmemiş ürünlerin bulunduğu (12 adet ürün bulunan) bir sepettir. Yedek sepetin olup, olmadığını anlamak için, sepetin altına bir kontak sistemi yerleştirilmiştir.

İşlenen Sepet

Üretim aracında, o anda işlenmekte olan ürünlerin bulunduğu sepettir. Herhangi bir t_o anında, içerisinde en fazla 12 adet veya daha az işlenecek ürün bulunur.

İşlenmiş Sepet

Üretim aracında, o anda işlenmekte olan ürünlerin bulunduğu sepettir. Herhangi bir t_o anında, içerisinde en fazla 12 adet veya daha az işlenmiş ürün bulunur.

Bitmiş Sepet

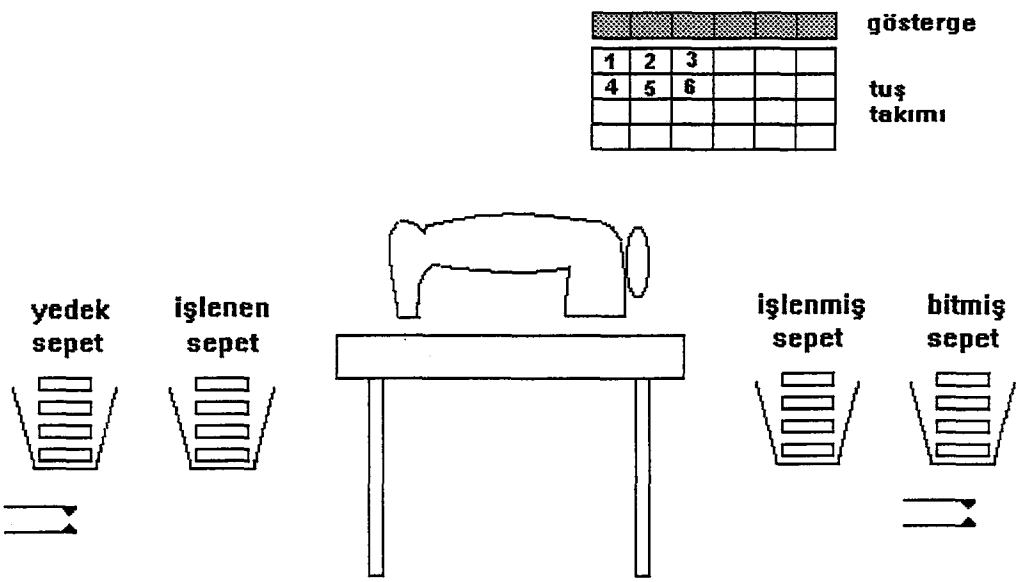
İçerisinde işlem görmüş ve bir diğer işlem için, başka bir üretim noktasına hareket etmeye hazır ürünlerin bulunduğu (12 adet ürün bulunan) bir sepettir. Bitmiş sepetin olup, olmadığını anlamak için, sepetin altına bir kontak sistemi yerleştirilmiştir.

Üretim aracının üzerinda, yamak bilgisayar adı verilen, bir bilgisayar bulunmaktadır. Yamak bilgisayarda, bilgilerin girilebilmesi için tuş takımı ve sonuçların görülebilmesi için bir gösterge mevcuttur. Yamak bilgisayar, üretim sistemine sürekli olarak bilgi gönderir ve buradan gelen verileri çalışanlara iletir.

Bu bilgiler, aşağıda sıralanmıştır:

- İş başladığından itibaren o ana kadar işlemi bitirilen ürün adedi,
- Ne kadar para kazandığı (prim),
- Başındaki operatör çalışıyor / kısa veya uzun süreli izinli,
- Üretim aracı sağlam / bozuk,
- Üretim aracı tamirde,
- Tamiri yapılan sağlam üretim aracı

Tanıtılan üretim aracının temsili resmi Şekil 3.4 te gösterilmiştir.



Şekil 3.4 : Üretim aracının sembolik gösterimi

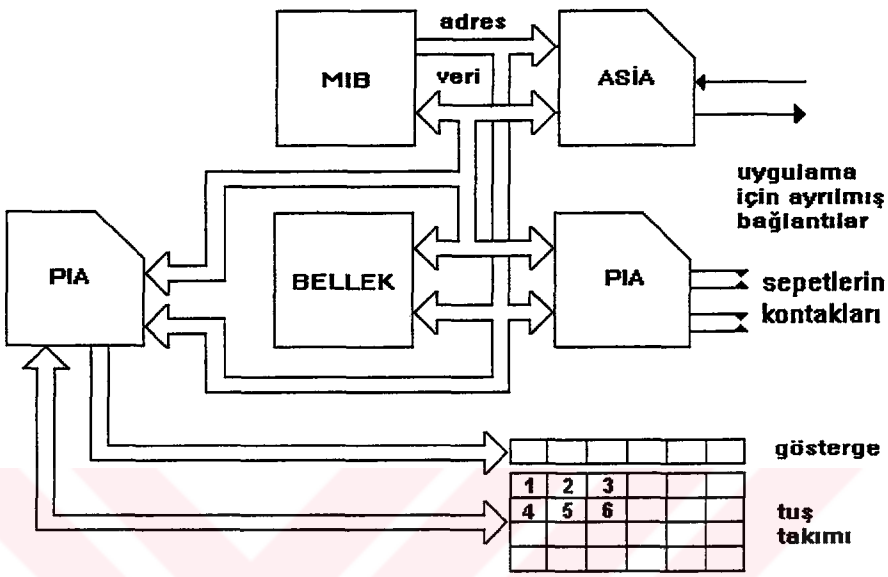
3.1.3 Bilgisayar Donanımı

Üretim aracının yan tarafında bulunan yamak bilgisayarın mimarisi aşağıdaki gibi tasarlanmıştır. Buna göre, tasarlanan bilgisayar;

- MİB (Merkezi İşlem Birimi),
- Bellek,
- Tuş Takımı/Gösterge,
- ASİA (Asenkron Seri İletişim Arabirimi),
- PİA (Paralel İletişim Arabirimi),

gibi elemanlardan meydana gelmiştir. Sistemin sembolik gösterimi, Şekil-3.5'te gösterilmiştir. ASİA, yamak bilgisayarla, usta bilgisayar arasındaki veri iletişimini sağlar. Sistemde iki tane PİA vardır. Bunlardan birincisi, üretim aracı ile ilişkili durumları (sepetlerin varlığı ya da yokluğunu) saptamak amacıyla kullanılmıştır.

Diğer PİA ise tuş takımı ve gösterge için ayrılmıştır. Tuş takımı ve gösterge, yamak bilgisayara verilerin girilmesi, girilen verilerin ve usta bilgisayardan gelen mesajların gösterilmesi (kaç tane yaptı, ne kadar para kazandı v.s gibi) amacıyla kullanılmaktadır.



Şekil 3.5 : Yamak bilgisayarın sembolik gösterimi

3.2 Simülasyon Destek Sistemi

2. Bölümde tanıtıldığı gibi USDUS'nin temel yapısında, üretim ortamının sümülasyonu yer almaktadır. Üretim ortamının tüm özelliklerini taşıyan bir simülasyon desteği, uzman sistem yaklaşımına yol göstermektedir. Tez çalışmasında, gerçek üretim ortamının davranışlarını incelemek üzere, aynı simülasyon yazılımından yararlanılmıştır. Dolayısıyla, tezde simülasyon yazılımından iki türlü yararlanılmıştır:

1. Üretim ortamının simülasyonu amacıyla
2. Simülasyon desteği amacıyla

Simülasyona dayalı bir yöntemin üreteceği sonuçların doğruluğu, modellemenin doğruluğu ile orantılıdır. 2. Bölümde, üretim ortamında yer alan birimlerin neler olduğu ve bunların nasıl modelleneceği anlatılmıştır. Bu bölüm içinde, simülasyon yazılımının nasıl gerçeklendiği ve bu gerçekleme sırasında kullanılan yöntem ve tekniklere yer verilmiştir.

Her simülasyon denemesinde olduğu gibi, USDUS için geliştirilen simülasyon tekniğini deneyebilmek için, gerçek bir üretim ortamı ele alınmıştır. Daha önce belirtildiği gibi, bu tez çalışmasında ele alınan üretim ortamı modeli, partiler halinde yapılan aralıklı üretim biçimidir. Partiler halinde yapılan aralıklı üretim yöntemi için bir konfeksiyon fabrikası örnek olarak seçilmiştir. Bir konfeksiyon fabrikasının örnek olarak seçilmesinin ardında, gerçek bir fabrika üzerinde araştırma yapılmış ve aşağıdaki bilgiler elde edilmiştir:

3.2.1 Konfeksiyon Fabrikası için Temel Kavramlar

Örme alanında faaliyet gösteren fabrikada, üretim sürekli olmakla beraber, üretimin haftalık dönemler için planlandığı gözlenmiştir. Üretilen ürünlerin ana çeşidi, model olarak anılmaktadır. Her model, birden çok beden ve renk karışımı ile üretilmektedir. Fabrikadaki gözlemler sonunda ortaya çıkan temel kavramlar aşağıda sıralanmıştır:

Model

Üretimi yapılan her tasarıma, yani biribirinin aynı olan ürünler kümesine, bir model denilmektedir. Her model birden fazla renkte (birden fazla renk karışık olarak ta kullanılabilir) ve bedende üretilmektedir. Bir modelin üretilmesi sırasında, izlenecek olan süreç adımları, simülasyonda **"Süreç Adımlarının Tanımlanması"** programı yardımıyla tanımlanmıştır. Bir ürünün oluşmasında, ardarda gerçeklenen süreç adımları p_i olarak gösterilecektir. Üretim ortamında gerçeklenebilecek süreç adımlarının toplam sayısı m olarak belirlenmiştir. Ancak bir ürünün üretilmesi aşamasında, tüm üretim aşamalarından geçmesi gerekmemektedir. Bu durumda, herhangi bir ürünün (j. ürün) oluşması genel olarak şöyle yazılabilir:

$$S_j = p_{j1} + p_{j2} + \cdots + p_{ji} + \cdots \qquad (2\text{-}1)$$

Yukarıda değinildiği gibi bir ürün, çok sayıda işlemin peşpeşe gerçeklenmesi ile meydana gelmektedir. Üretim ortamında üretilecek olan ürünlerin toplam model sayısı nn ve bir model içindeki ürün sayısı dd_j olabilir. Ancak, bir anda, bir başka anlatımla, bir üretim diliminde, üretim ortamında bulunabilecek model sayısı n dir. Her bir modelden bir anda üretilecek olan miktar ise, d_j olarak sınırlanmış olabilir.

Bir modelin tasarımı aşamasında, hangi işlem adımlarından oluşacağı belirlenmektedir. İşlem adımları birer işlem koduyla tanımlanmaktadır. Süreç adımları tanımlama programında model kodu, açıklaması girildikten sonra süreç adımları sırasıyla tanımlanır. Model kodu, çalışana bağımlı ve bağımsız işlem süreleri ve üretim aracı tipi burada tanımlanmaktadır.

Yukarıda belirtilen hususların ışığında, ürünler, bilgisayar ortamında modellenmiştir.

Varyant

Bir modeldeki renk çeşitliliği varyant olarak adlandırılır. Tek renkli ürünlerde, renk sayısı varyant sayısına eşittir. Bir modeldeki varyant sayısı N_v ile gösterilmiştir. Üretim sırasında, bir modelin tüm varyant ve bedenleri aynı anda işleme alınmaktadır.

Bir model, genellikle üç ya da beş beden olarak üretilmektedir. Bir model için her bedenden üretilecek olan miktarın aynı olması koşulu yoktur. Bir bedenden ne kadar üretileceği B_b (b beden) ile gösterilmiştir. Üretim aşamalarında varyant ve beden etkili olmaktadır. Bu nedenle, simülasyon sırasında beden ve varyantlarla ilgili ayrıntı yer almaktadır.

İncelenen fabrikada, bir model ürünün bir varyant ve bir bedeninden, genellikle, en az 12 adet üretilmekte ve bunlar bir sepet içinde, üretim ortamında hareket etmektedirler. Bir modelin bir bedeninden üretilen miktar W_b (beden) ile gösterilmiştir. Genelde sabit ve 12 olan, bir sepet içindeki ürün sayısı $N_{sü}$ ile gösterilmiştir.

İncelenen konfeksiyon fabrikasında, her sepet üzerinde, sepet içindeki ürün ile ilgili olarak, model, varyant ve beden bilgileri ile üretim sırasında, üzerinde uygulanacak süreç adımları yer almaktadır.

Bir model ürünün bir bedeninin gerektirdiği sepet sayısı N_{bs} olarak gösterilmiştir. Yukarıdaki tanımlamalara açıklık getirmek amacıyla, aşağıdaki hesaplama verilmiştir:

Üç beden ve beş varyant olarak tasarlanan bir model için, birinci bedenden 240, ikinci bedenden 360 ve üçüncü bedenden 240 ve toplam 840 adet üretilmesi planlanmıştır. Buna göre; üretimde kullanılacak sepet sayıları Tablo 3.1 de verilmiştir.

Tablo 3.1
Üretim sepet sayıları

	1. varyant	2. varyant	3. varyant	4. varyant	5. varyant
1. beden	4	4	4	4	4
2. beden	6	6	6	6	6
3. beden	4	4	4	4	4

$dd_j = \sum_{b=1}^{bb} B_b$ örnek için

$dd_j = B_1 + B_2 + B_3$

$= N_v W_1 + N_v W_2 + N_v W_3$ örnek için sayılar yerleştirilirse

$= 5 * 48 + 5 * 72 + 5 * 48$ olarak bulunur. Sepet sayısı açısından denklemi düzenlersek

$= N_v N_{1s} + N_v N_{2s} + N_v N_{3s}$ yine sayısal değerler yerleştirilirse

$= 5 * 4 + 5 * 6 + 5 * 4$ sonucu bulunur.

Sonuç olarak ele alınan model için 70 sepet gerekecektir

3.2.2 Ekran Sembolleri

Simülasyon sisteminde, ürünler üretim araçları ve çalışanlar ile ilgili bilgiler görüntülenmektedir.

Ürün Sembolleri

Simülasyon ekranında, herbir üretim sepeti birer kutu ile gösterilmiştir. Bir modelin bir varyantı ve bir bedenine ilişkin sepetler aynı renkte gösterilmiştir. Ürün sepetleri,

- Ürün hazırlama deposunda (Ana depo)
- Üretim ortamı girişinde,
- Üretim araçlarının yedek iş konumunda,
- Üretim araçlarının tamamlanmış iş konumunda
- Bekleme kuyruklarında
- Üretim ortamı çıkışında

görünecektir.

Ürün sepetleri birer kutu ile gösterilmiştir. Her bir varyant ve beden farklı renkteki kutularla gösterilmiştir.

Yedek İş

Simülasyon ekranının sol kısmında gösterilir. Çalışacak ilk makinalarda kendileri için boş yer olmayan işlere yedek iş denir. Bu tipten işler çalışacakları ilk makinalardan herhangi biri boşaldığında hemen oraya gönderilir. Kendi renkleri için ayrılmış olan yerde ve dörtlü sepetler halinde beklerler.

Ana Depo

Simulasyonun haftalık olduğu daha önceden belirtilmişti. Üzerinde inceleme yapılan fabrikada haftada üretilen ürün sayısı 15000 adettir. Bu sayıyı **haftalık fabrika kapasitesi** olarak düşünmekte mümkündür. Buna göre haftada beş iş günü olduğu varsayımıyla fabrikanın günlük iş kapasitesi 3000 dir.

Bir sepette ortalama 12 adet ürün olduğu varsayılırsa, günlük sepet sayısı 250 olarak bulunur. Üretim ortamının girişinde 48 adet sepetin gösterimine yer ayrılmıştır. Dolayısıyla, geri kalan sepetlerin ana depoda beklediği varsayılacaktır.

Tamamlanmış İş

Ekranın sağ kısmındaki sepetleri simüle eden kareler tamamlanmış iş kutularıdır. Her işin kendine göre bir sepet grubu için kutusu vardır. Fabrika ortamında daha önceden tanımlanan adımlardan geçen ürünler tamamlanmış iş alanında toplanmaya başlarlar. Ortamdaki, tüm sepetlerdeki üretim sonucu biten işler, tamamlanmış iş katagorisinden çıkarak kaybolurlar. Gerçek hayatta bu ürünlerin gittiği yer bir depodur.

Kuyruk

USDUS'nin temel hedefi, üretim ortamında oluşan kuyruk ve iş beklemelerini azaltmaktır. Bu nedenle, oluşan kuyrukların görülmesi gereklidir. Kuyrukların hangi durumlarda oluşacağı 2. Bölümde anlatılmıştır. Kuyruklar sepetlerin rengindedir.

3.2.3 Üretim Araçları Sembolleri

Üretim ortamında bulunan üretim araçları, belli işleri gerçeklemek üzere kullanılmaktadır. Yani, işlem adımlarının üzerinde gerçekleştirildiği araçlara verilen addır. Belli işlemleri gerçeklemek üzere birden fazla üretim aracı kullanılmaktadır. Üretim için gerekli olan üretim araçlarının sayısının, üretim öncesinde hesaplandığı varsayılmaktadır. Bu basit hesaplamanın yöntemi 2. Bölümde verilmiştir. Üretim ortamında bulunan tüm üretim araçlarının sayısı qq ve belli bir anda kullanılanlarının sayısı ise q ile gösterilmiştir.

- Simülasyon ortamında bulunacak olan makinalar **Makina Bilgileri Tanımlama** programı yardımıyla sisteme tanıtılmaktadır.

Yerinde yapılan incelemelere uygun olarak, üretim ortamı simülasyonunda, 16 farklı tip makinaya (romeyöz, overlok, ilik, singer v.b.) yer verilmiştir.

Üretim ortamında her tip makinadan en fazla 20 adet olabileceği varsayılmıştır. Bu sayı da gerçek ortamı yansıtmaktadır. Zira, incelenen konfeksiyon fabrikasında da aynı cins üretim aracından en fazla 20 adet mevcuttur. Simülasyonda her tip makina farklı bir sembolle gösterilmiştir. Gerçek üretim ortamında da aynı tip makinaların peşpeşe birer küme oluşturduğu göz önünde bulundurularak simülasyonun gerçek ortamla olan uyumu sağlanmıştır.

Simülasyon ekranındaki semboller ile ilgili açıklamalar aşağıda verilmiştir: [24]

a. **Bekleyen Makina :** *(Boş Sembol : Beyaz çerçeve, içi siyah)* Bu durumdaki makinalar kullanıma hazır makinalardır. Üzerine herhangi bir iş yüklendiğinde çalışırlar. Makina başında çalışan hazır beklemektedir. Makinalar iş beklemektedir, dolayısıyla boştadır.

b. **Çalışan Makina :** *(Bir Renkle Doldurulmuş Sembol : Renkli çerçeve, içi aynı renk)* Daha önceden belirtildiği gibi her renk bir modelin bir varyant ve bir bedenini ifade etmekteydi. Makina o sırada hangi beden ve varyantla ilgili sepeti işlemekte ise o varyantın rengini alır. İşlem bitirildiğinde tekrar boş sembol haline dönüşür.

c. **Arızalı Makina :** *(Kırmızıyla Doldurmuş Sembolün İçinde Sarı Çember)* Bu üretim aracı arızalanmış demektir. Kullanılamaz. Üzerinde yarım kalan işlem başka bir makina tarafından tamamlanacaktır. Arızalanan makina onarım sürecine girmiştir.

d. **Boştaki Makina :** *(Maviyle Doldurulmuş Sembolün İçinde Beyaz Çember)* Üretim aracı işlevsel olarak çalışır durumdadır, fakat başında işi yapacak olan çalışan bulunmamaktadır. Bu duruma 3 şekilde ulaşılabilir:

    1. *c durumundan sonra*: Makina tamir edilmiş, yeniden üretime alınmıştır. Başına insan verilmesini bekler.
    2. *Simülasyon ilk başladığı anda*: Bilindiği gibi, maliyetler yüzünden üretim aracı sayısı, çalışanların sayısından daha fazladır. Bu yüzden daha simülasyon ilk başladığında bazı makinaların başında çalışan olmayacaktır.
    3. *Arada elemanların başka makinalara kaydırılması*: Uzman sistem çalışanların, daha verimli olarak çalışabileceği makinaları bildiğinden, kuyruklar oluştuğunda, eritmek için boştaki insanları başarılarını da değerlendirerek buralara kaydırır.

3.2.4 Olaylar ve Sürelerinin Belirlenmesi

Yazılımda kullanılan olaylar ve sürelerinin belirlenmesi aşağıda anlatılmıştır.

Bir Ürün Sepetinin İşlemde Kalma Süresinin Hesaplanması

Bir ürünün üretilmesi sırasında gerçeklenecek olan süreç adımlarının süreleri sabit değildir. Bu süre makina hızına ve bu makinada çalışan kişinin becerisine göre değişmektedir. Bu çalışma ile sunulan sistemde, üretim alanında çalışan her insanın, her tip makina için çalışma becerileri tanımlanmaktadır. Beceri katsayısının nasıl tanımlandığı 2. Bölümde anlatılmıştır.

Gerçeklenen yazılımda, her çalışanın her makina için beceri katsayısını görme olanağı bulunmaktadır. Ayrıca, makinaların hızları da bilindiğinden, sepetlerin işlemde kalma süreleri, 2. bölümde detayları verilen hesaplama yöntemleri uyarınca kolaylıkla hesaplanmaktadır. Genel bağıntı aşağıda gösterilmiştir.

$$t_i = t_y + t_h + t_o\,(70/b) + t_a \qquad (2\text{-}7)$$

Gerçek Zaman Saati Oluşturma

Sistemde simulasyon ile gerçek zaman arasındaki zaman bağlantısı şu şekilde hesaplanır. Gerçek Zaman Saati, bir başka deyişle simülasyonun ölçeğidir. Sistemde kurulu bulunan gerçek zaman kavramı için öncelikle kritik makina tanımı yapılmalıdır.

Kritik Makina

Aşağı yukarı her ürünün uğradığı ve işlem süresinin en kısa olduğu makinalar kritik makinadır. İşlem süresi kısa olduğu için bu makinalardan üretim ortamında fazlaca bulundurmak uygun değildir. Eğer gereğinden fazla bulundurulursa işletme açısından gereksiz yatırıma neden olur.

Kritik makina sayısı oldukça kısıtlıdır. İncelenen üretim ortamında kritik makina olarak tanımlayabileceğimiz makinalardan (örnek işletmede ilik dikme makinası) sadece iki adet bulunduğu gözlendi. Ürünlerin büyük bir çoğunluğu bu makinalara en az birer kez uğramakta ve işlem süreleri de çok kısa olmaktadır.

Daha önce ana depo tanımında bahsedildiği gibi işletmede, günde üretilen ürün sayısı 3000 adet civarındadır. Bir sepetteki ürün sayısı $N_{sü}$ dikkate alındığında 12 olduğu biliniyor. Günde üretim ortamında işlenen sepet sayısı, $3000/N_{sü} = 250$ 'dir. Üretim ortamında bulunan işçi sayısının 69 olduğu gözünde bulundurulursa; bir operatörün bir günde işlediği sepet sayısı da, yaklaşık 4 ($_{sepet/adam}$) bulunur.

Kritik makina sayısının iki olduğu bilindiğine göre ve günde işlenen sepet sayısı 250 olduğuna göre, her gün bir kritik makinadan en az 125 sepet geçmek durumundadır. Bir günlük çalışma süresinin 8 saat olduğu varsayımıyla, bir kritik makinada bir sepetin işlenip geçme süresi olarak :

$8x60/125 = 3.9 \cong 4$ (dakika/sepet) bulunur.

Sepet durumlarını görebilmek ve bilgi kaybetmemek için örnekleme zamanımız, bulduğumuz sayının en az yarısı olmalıdır. Buradan varılan sonuca göre sistemin Gerçek Zaman saati 2 dakikadır.

Simülasyon esnasında ekranın sağ üst köşesinde yer alan sayaç her ilerlediğinde gerçekte iki dakika geçmektedir.

Makinaların Bozulması

Simülasyonda makinaların bozulmaları yazılımla gerçekleştirilmektedir. 2.Bölümde, bir makinanın iki bozulması arasında geçen sürenin (MTBF : Mean Time Between Failure) Poisson dağılımına göre olduğu belirtilmişti. Bu bağıntı, yazılımla gerçeklenmiştir.

Sistemde makina tanımlama programı yardımıyla her makina tanımlanırken bu makinaya ait Ortalama Bozulma Süresi (OBS) de alınmaktadır. Alınan bu süreye göre ilk başlangıç aşamasında tüm makinalar arasında bir inceleme yapılarak makinalara Poisson dağılımına göre bir bozulma süresi tesbit edilmektedir. Gerçek zaman saati belirlenen değere geldiğinde ilgili makina simülasyon tarafından bozulmakta, belirlenen tamir süresi boyunca da bozuk kalmaktadır. Onarım süresi de tamamlanan makina tekrar devreye girdiğinde, yine Poisson dağılımına göre bir bozulma süresi tesbit edilmektedir. Bakınız, bölüm 4.2.2.1.2

3.3 Uzman Sistem Desteği

Üretim ortamındaki üretim araçları, hazır durumda değillerse üzerlerine iş kabul etmezler. Bunların, hazır durumda olabilmesi için fiziksel olarak çalışabiliyor olmaları yetmemektedir. Başlarında, kendilerini kullanacak bir insana ihtiyaç bulunmaktadır. Simülasyon esnasında ekranda bazı üretim araçlarının gözükmesine rağmen üzerlerine hiçbir iş yüklenmediği gözlemlenebilir. Yani, bazı üretim araçlarının başında bir çalışan olsa dahi, işin kendisine gelmemesi dolayısıyla boşta kalabilir. Böylesi bir durum karşısında o makinanın başındaki elemanın prim alamaması gibi istenmeyen bir durumla karşılaşılabilir.

USDUS'nin sayesinde, böylesi bir durumla karşılaşıldığında, ya o üretim aracına iş akışı sağlanır veya iş durumu gereğinden az ise, üretim aracının başındaki insan, ihtiyaç görülen bir başka makina grubuna kaydırılır. Diğer olası durumlarla ilgili bilgiler aşağıda anlatılmıştır.

- **Çalışan Makinalardan, Herhangi Birisinin Arızalanması Durumunda :** Arızalanan makina başındaki eleman, bu andan itibaren boşa çıkar. Böylesi bir durum karşısında, açığa çıkan çalışanın işsiz kalmaması için, bu bilgi simülasyon ortamına iletilir. Uzman sistem programı, bildirilen operatörün niteliğine göre bu elemanı, başka makinalara kaydırabilir. Uzman sistemin yer değiştirdiği operatörler simülasyona bir ara dosya yardımıyla belirtildikten sonra, bu operatör simülasyonda da yeni yerine taşınmış olur. Klasik yapıdaki bir fabrika ortamında ise bu operatörler ustabaşlarının göstermiş olduğu yeni makinaya geçeceklerdi.
- **Çalışan Elemanın İzin Alarak Makina Başından Ayrılması :** Bu ayrılıklar küçük süreli olabileceği gibi daha uzun süreli (bir kaç saat veya günboyu) olabilir. Fabrika ortamında işçinin ayrılış sebebi ve süresi yamak bilgisayar tarafından usta bilgisayara bildirilir. Yerine, yeni birini yerleştirip yerleştirmemek kararını usta bilgisayar, eğer ilerde kuyruk olma ihtimali görürse veya halihazırda kuyruk varsa (kendi algoritmasına göre) başka bir operatörü atayabilir veya yapacağı simülasyon sonucu gerek görmeyebilir.
- **Onarım Süresi Tamamlanan Makinanın Durumu :** Böyle bir durumda da makina başında operatör olmayacaktır. Çünkü makina ilk bozulduğunda uzman sistem programınca çalışanlar başka makinalara kaydırılmıştı. Bu operatörün tamir edilen makinaya dönmesi veya herhangi başka bir makinadan bu makinaya operatör aktarılması uzman sistem algoritmasına göre yazılmış olan simülasyon programının yapması gereken bir düzenlemedir.

- **Makina Başından Bir Çalışanın Alınması Durumu :** USDUS sistemi, üretim ortamında, bir makina grubunda iş yığılması yani, kuyruk oluştuğunu tespit ettiğinde, simülasyonu başlatır ve iş durumu müsait olan makina gruplarından bazı elemanları, yeni makina grubuna kaydırabilir.

Çalışanların yeni makinalara atanmaları işleminde, o üretim araçlarını kullanma başarıları göz önüne alınmaktadır.

Yukarıda verilen açıklamalardan anlaşılacağı gibi, kullanılan uzman sistem yöntemi; kurallara dayalı uzman sistem yöntemidir.

3.4 Yönetim Sistemi

Daha önce Şekil 2.4'te gösterildiği gibi bu tez çalışmasının amacı, en iyi biçimde düzenlenmiş olan bir üretim dizgesini, önce bilgisayar ile donatmak; bu sayede, üretim ortamını gözlemek ve ardından uzman sistem yöntemlerinden de yararlanarak yönetmektir.

Yönetim sistemi, üretim ortamının anlık durumlarını sürekli olarak gözleyerek (monitör ederek), eşik değerler aşılıncaya kadar beklemede kalır.

Tablo 3.2

Üretim ortamı durum kütüğü

	Kuyruk	Bekleyen	Arızalı
Durum Kütüğü	x	y	z
Eşik	X	Y	Z

Bu durum aşağıdaki gibi yazılır:

Eğer (x ⟩ X) + (y ⟩ Y) + (z ⟩ Z) ise
    Durum_kütüğünü_Oku
    Uzman_sistemi_çalıştır
    Çözüm_Önerisini_Öğren
    Öneriye_Uygun_Simülasyon_Yap
    Sonuca_Bak
        Eğer Sonuç iyileştirici ise
            aynı öneriyi kullanarak yeniden simülasyon yap
    değil ise
        bir önceki sonucu yeterli varsay
    son duruma göre üretim ortamını düzenle

Eşik değerlerin aşıldığı durumlarda, uzman sistemden yardım almaya başlar.

- Kuyruk
- Bekleyen
- Arızalı

Yukarıda belirtilen üç durum karşısında, yönetim sistemi aşağıda açıklandığı gibi davranır:

1. Uzman sistemin ilk bakacağı durum kuyruk oluşmuş mu ? Kuyruk oluşmamış ise üretim ortamında yapılacak hiç bir değişiklik gerekmemektedir.

2. Kuyruk oluşmuş ise: İş bekleyen bir veya birden fazla çalışan var mı? Varsa bunlar içinde, kuyruğun giderilmesinde en etkili olacak kişiyi seç. Varsa bu çalışanların, kuyruğun oluştuğu noktaya kaydırılması fikri oluşuyor. Bu yeni duruma göre, simülasyon başlatılır. Simülasyon sonucunda, tekrar kuyruklar incelenerek, kuyruğun eşik seviyesine inip, inmediği kontrol edilir. Şayet, eşik seviyesine inmemiş ise, bir başka boşta çalışan olup olmadığını kontrol eder. Varsa kaydırma işlemini yaparak tekrar simülasyonu başlatır. Kuyruğun, eşik seviyesine inmesine kadar bu işlem böyle devam eder.

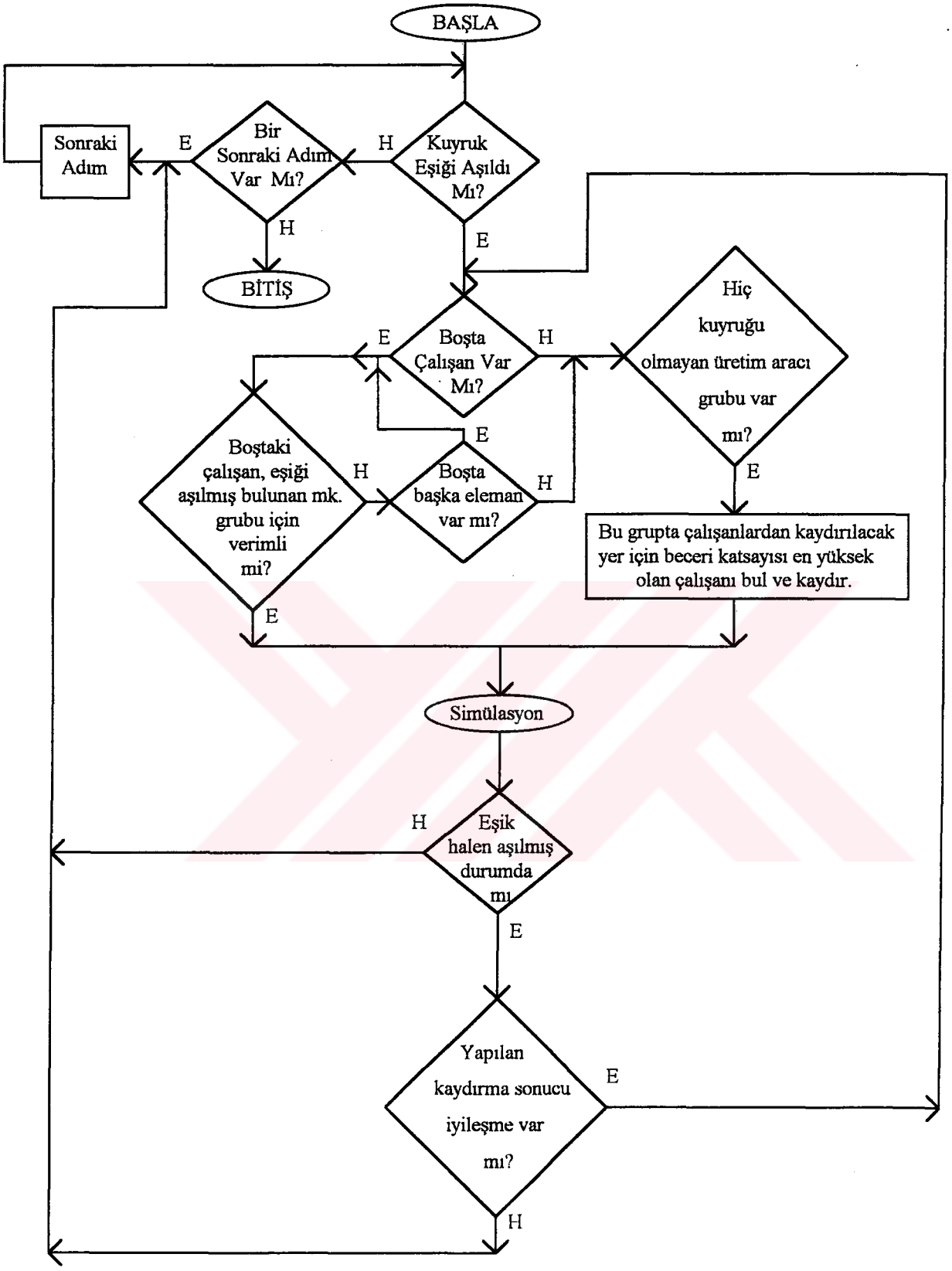
3. Bir sonraki adımda, kuyruk yok olacak, ancak iş bekleme durumuna gelinecektir. Bu anlaşıldığında, bir önceki sümülasyon adımının uygun olduğu sonucu ortaya çıkacaktır. Şimdi kimlerin yer değiştireceği bellidir ve bu talimat olarak üretim ortamına verilir.

4. Boşta çalışan olmadığı durumlarda;

   - Kuyruğun en az olduğu bölümlerden, çalışan kaydırılması yapılarak üretim ortamının simülasyonu yapılır.
   - Bu durumda, öncelikle çalışanın alındığı bölümdeki kuyruk oluşumuna bakılır.
   - Kuyruk, eşik seviyesini aşmış ise, bu çalışanın alınması işleminden vazgeçilerek, bir başka bölümdeki çalışan kaydırılarak, tekrar simülasyon başlatılır.
   - İşlem böylece devam eder.
   - Kaydırılacak çalışan/çalışanların beceri katsayılarına bakılır.

5. Simülasyon, yarım günlük yapılır. Daha, uzun süreli yapıldığında gerçekçi olamaz, teorik olur. Simülasyon ekranının sağ üst köşesindeki sayaçın her adımı 2 dakika olarak tanımlanmıştır. Dolayısıyla, yarım günlük bir simülasyon, yarım günün 4 saat olduğu kabulünden hareketle, 4 * 60/2 = 4 * 30 = 120 adımdır. Yani, simülasyon programının 120 adım işletilmesi, üretim ortamının yarım günlük gözlenmesine karşılıktır.

'Bekleyen' ve *'Arıza'* durumlarında da, benzer çözümler üretilir.



Şekil 3.6 : Uzman sistem algoritması akış diyagramı

4. Bölüm

Simülasyon ve Sonuçlar

Bu bölümde, bu çalışma için geliştirilmiş bulunan yazılım paketinin, teknik yönünün tanıtımına ve ayrıca, üretim ortamının çeşitli simülasyonları sonucu, elde edilen grafikler (USDUS üretim ortamında var/yok) sayesinde, önerilen USDUS'nin üretim ortamına sağladığı üstünlükler, grafiksel olarak gösterilmeye çalışılacaktır.

4.1 Bir Günlük Üretim Tablosunun Hazırlanması

Bu bölümde, simülasyon için 10 çeşit örnek model üzerinde, gerçek zaman verileriyle, üretim ortamının birebir simülasyonu yapılmıştır. Bu modeller için, Tablo 2.2 ye dayalı olarak gerekli makine sayıları hesaplanmıştır ve yapılan bu hesaplamalar, Tablo 4.1 de gösterilmiştir. Buna göre, 10 çeşit model seçilmiş (Bunlar Tabloda S1,S2....S10 biçiminde gösterilmiştir) ve bu işlerin en fazla 16 iş istasyonuna uğrayabileceği kabulü yapılmıştır. Gerçek üretim ortamında da ürünlerin, en fazla 8-16 iş istasyonuna uğradıkları tespit edilmiştir. Dolayısıyla, simülasyon ortamının, gerçek üretim ortamıyla aynı özelliklere haiz olması sağlanmıştır. Ayrıca, bu hesaplamalarda, çalışanların "*ortalama beceri*" süreleri gözönüne alınmıştır.

Tablonun birinci sütununda, S1 den S10' a kadar seçilen model grupları gösterilmiş olup ikinci sütunda ise, her model grubuna ait birim model adetleri gösterilmiştir. Tablonun birinci satırında, g1 den g16'ya kadar makina grupları gösterilmiştir. Tabloda bir iş parçasının, uğrayacağı her bir iş istasyonunda kalacağı süreler dakika cinsinden gösterilmiştir. Bu tablodan yararlanarak, ele alınan on modelin belli sayıdaki üretimi için gerekli olan zamanlar Tablo 4.2 de gösterilmiştir.

Tablo 4.2 incelendiğinde, buradaki değerlerin, Tablo 4.1 de tek bir iş parçası için verilmiş olan değerler kullanılarak üretildiği görülmektedir. Bu işlemler sonucu oluşturulan tablonun son sütununda ise, aynı model iş grubunun, makina gruplarında geçirecekleri "*toplam işlem*" sürelerini vermektedir.

Sürenin 0 olması, ele alınan modelin iş istasyonuna uğramaması anlamını taşır.

Tablonun en alt satırında, toplam iş sayısı ve her bir iş istasyonunun toplam işlem süreleri verilmiştir. Seçilmiş model grubu ve belirlenmiş üretim adetleri için, gerekli olan makine sayılarının hesaplanması için, Tablo 4.2 nin en alt satırından yararlanılacaktır. Bizim için en önemli satır, makina gruplarında 10 model iş için sarfedilecek olan sürelerin toplamını gösteren, matrisin alt bölümündeki satırdır. Zira buradan elde edilen değerler, bir günlük çalışma saatine bölündüğünde : (8 saat veya 480 dak.) Çıkacak olan sonuçlar, bu işin gerçeklenmesi için gerekecek olan "*makina sayısını*" verir. Burada ortaya çıkan küsüratlı değerler, tam sayılara yuvarlanarak tablonun bir alt satırı oluşturulmuştur. Yukarıda yapılan hesaplamalar neticesinde, bulunan makina adetleri, iş gününde aynı anda kullanılacak olan minumum makina sayısıdır. Her hangi bir arızi durum karşısında iş akışının etkilenmemesi için, her makina grubunda hesaplanan daha fazla makina bulunmaktadır ve bunlara "*yedek makinalar*" adı verilmiştir.

Tablo-4.1

İşler ve üretim araçları arasındaki ilişkiler

			İŞ İSTASYONLARI VE ÜRETİM SÜRELERİ															
	Model	Adet	g1	g2	g3	g4	g5	g6	g7	g8	g9	g10	g11	g12	g13	g14	g15	g16	Toplam
	S1	360	3	0	2	1	0	0,5	0	5	1	2	1	0	0	0,5	0	0	16
	S2	288	2	2	3	1	1	0	0	5	1	2	1	0	0	0,5	0	0	18,5
	S3	360	2	2	1	1	0	0,5	0	6	0	3	0	3	1	1	0	1	21,5
	S4	288	3	1	2	1	1	1	1	7	0	4	1	2	1	1	1	0	27
	S5	360	2	2	1	1	0	0	1	3	1	1	1	2	1	1	0	1	18
	S6	144	1	1	1	0	0	0	1	4	1	4	1	2	0,5	1	0	0,5	18
	S7	288	4	0	2	1	1	1	0	3	1	2	1	0,5	1	0,5	1	0	19
	S8	288	2	2	2	0	0	0	3	4	1	3	1	1	1	0,5	0	0	20,5
	S9	360	1	1	2	1	0	0	1	4	0	3	1	1	0	1	0	0	16
	S10	288	2	2	1	1	0	0	1	5	1	3	1	1	0	0,5	0	0	18,5
Toplam Adet		3024																	
Toplam Süre			22	13	17	8	3	3	8	46	7	27	9	12,5	5,5	7,5	2	2,5	

Tablo-4.2

Toplam süre olarak işler ve üretim araçları arasındaki ilişkiler

		İŞ İSTASYONLARI VE ÜRETİM SÜRELERİ																
Model	Adet	g1	g2	g3	g4	g5	g6	g7	g8	g9	g10	g11	g12	g13	g14	g15	g16	
																		Toplam
S1	360	1080	0	720	360	0	180	0	1800	360	720	360	0	0	180	0	0	5760
S2	288	576	576	864	288	288	0	0	1440	288	576	288	0	0	144	0	0	5328
S3	360	720	720	360	360	0	180	0	2160	0	1080	0	1080	360	360	0	360	7740
S4	288	864	288	576	288	288	288	288	2016	0	1152	288	576	288	288	288	0	7776
S5	360	720	720	360	360	0	0	360	1080	360	360	360	720	360	360	0	360	6480
S6	144	144	144	144	0	0	0	144	576	144	576	144	288	72	144	0	72	2592
S7	288	1152	0	576	288	288	288	0	864	288	576	288	144	288	144	288	0	5472
S8	288	576	576	576	0	0	0	864	1152	288	864	288	288	288	144	0	0	5904
S9	360	360	360	720	360	0	0	360	1440	0	1080	360	360	0	360	0	0	5760
S10	288	576	576	288	288	0	0	288	1440	288	864	288	288	0	144	0	0	5328
Toplam Adet	3024																	58140
Toplam Süre		6768	3960	5184	2592	864	936	2304	13968	2016	7848	2664	3744	1656	2268	576	792	58140
Makina HESABI		14,1	8,25	10,8	5,4	1,8	1,95	4,8	29,1	4,2	16,35	5,55	7,8	3,45	4,725	1,2	1,65	121,125
Makina SAYISI		14	8	11	5	2	2	5	29	4	16	6	8	3	5	1	2	121

4.2 Simülasyon Yazılımı

Yukarıda belli bir model grubu için, belli üretim miktarlarına göre, gerekli olan makine adetleri, basit bir görüşle hesaplanmıştır. USDUS 'nin üstünlüklerini kanıtlamak üzere, iki aşamalı bir çalışma gerçeklenmiştir.

Birinci aşama : Basit hesaplama yöntemiyle, üretim araçlarının sayıları belirlenmiş ve uzman sistem desteği olmaksızın simülasyon gerçeklenmiştir.

İkinci aşama : Basit hesaplama yöntemiyle, üretim araçlarının sayıları belirlenmiş ve uzman sistem destekli üretim ortamı sümüle edilmiştir.

Kısaca ana hatları verilen simülasyon uygulamasının ayrıntıları aşağıda verilmiştir.

4.2.1 Simülasyon Programında Kullanılan Yaklaşımlar

Kısıtlamalar

Simülasyon programı, işlevsel olarak, üretim ortamıyla birebir eşdeğer olmasına karşın, ekran gösterimindeki kısıtlamalar sonucu, üretim ortamında bir günde kullanılan makinaların tamamı ekranda gösterilememiştir. Grafik ekrandaki görüntü sınırlaması haricinde, sistemin simülasyonu esnasında ve hesaplamalar yapılırken herhangi bir sınırlama yoktur. İstendiğinde, çok daha büyük üretim ortamlarının (yüzlerce çeşit makina için) simülasyonu yapılarak sonuçlar alınmaktadır. Çalışmanın, grafik ortamda gösterimi, olayın daha kolay anlaşılabilir olmasına katkıda bulunmak amacıyla gerçeklenmiştir.

Simülasyon ekranının getirdiği kısıtlamayı çözümlemek için, yukarıda örnek olarak ele alınan üretim uygulaması için hesaplanmış ve sonuçları Tablo 4.1 ve Tablo 4.2 de verilmiş değerler(üretim adedi hariç) yarıyarıya azaltılarak simülasyonda kullanılmıştır. Simülasyon için kullanılan bu son değerler Tablo 4.3 te verilmiştir. Görüldüğü gibi, 'g_4, g_6, g_7, g_8, g_9, g_{14}, g_{16}' iş istasyonları tablodan çıkartılmıştır. Herşey aynı oranda ölçeklendiği için, çıkacak olan sonuçlar da güvenilir ve gerçek hayatla oransal olarak birebir olacaktır. Bu üretim ortamıyla ilgili tablo aşağıda gösterilmiştir.

Tablo 4.3

Simülasyona esas makina ve model sürelerini gösteren tablo.

Model	Adet	İŞ İSTASYONLARI VE ÜRETİM SÜRELERİ									
		g1	g2	g3	g5	g10	g11	g12	g13	g15	Toplam
S1	360	1080	0	720	0	720	360	0	0	0	2880
S2	288	576	576	864	288	576	288	0	0	0	3168
S3	360	720	720	360	0	1080	0	1080	360	0	4320
S4	288	864	288	576	288	1152	288	576	288	288	4608
S5	360	720	720	360	0	360	360	720	360	0	3600
S6	144	144	144	144	0	576	144	288	72	0	1512
S7	288	1152	0	576	288	576	288	144	288	288	3600
S8	288	576	576	576	0	864	288	288	288	0	3456
S9	360	360	360	720	0	1080	360	360	0	0	3240
S10	288	576	576	288	0	864	288	288	0	0	2880
Toplam Adet	3024										33264
Toplam Süre		6768	3960	5184	864	7848	2664	3744	1656	576	33264
Makina HESABI	T/480	14,1	8,25	10,8	1,8	16,35	5,55	7,8	3,45	1,2	69,3
Makina SAYISI		14	8	11	2	16	6	8	3	1	69
Ekranda Gösterilen Makina SAYISI		18	10	15	3	20	8	10	5	2	91

4.2.2 Simülasyon Yazılımının Tanıtılması

Simülasyon yazılımı üç ana bölümden oluşmaktadır. Bu bölümler ve altındaki başlıkları şöyledir:

Veri Girişi : Üretim ortamında bulunan üretim araçlarının tanıtımı

Makinaların ortalama bozulma ve tamir sürelerinin belirlenmesi

Çalışanların üretim araçlarına atanması

Çalışan insanların tanıtılması (beceri tablosu)

Modellerin tanıtılması

Model içindeki çeşitlilik sayılarının belirtilmesi

Süreç adımlarının ve sürelerinin belirlenmesi

Parametreler : USDUS parametresi (var/yok)
Kuyruk eşik seviyesi
Öngörü simülasyon adım sayısının belirlenmesi
Durum tablolarının yazılacağı dosyaların belirlenmesi

Simülasyon : Kullanıcı arabirimi (grafik)
Veri girişi ve parametrelerin okunması
Durum tablolarının oluşturulması
Uzman sistem desteği
Gerçek zaman saati
Rastgele sayı üretimi
Normal dağılıma dönüştürme
Poisson dağılımına dönüştürme
Arıza oluşturma
Çalışanların hasta olması

Tezin bu bölümünde simülasyon bölümleri anlatılmıştır.

4.2.2.1 Veri Girişi

Simülasyon yazılımına, simüle edebilmesi için üretim ortamının tanıtılması gerekmektedir. Üretim ortamı temel olarak üretim araçları ve çalışanlardan oluşmaktadır. Modellerin de bu süreç adımlarıyla birlikte simülasyon sistemine tanıtılması gerekmektedir. Verilerin girişi için birbirinden farklı dört program bulunmaktadır:

İş Grubu Tanımlama Programı,
Üretim Araçları Tanımlama Programı,
Çalışan-Üretim Araçları Arasındaki İlişkileri Tanımlama Programı,
Üretim Araçlarına Çalışanları Atama Programı,

Bölümün ilerleyen kısımlarında bu programların tanıtımı yapılacaktır.

4.2.2.1.1 Üretim Ortamında Bulunan Üretim Araçlarının Tanıtımı

Üretim aracını tanımlamak için, üretim aracı tanımlama programı kullanılır.Programın ekran görüntüsü Şekil 4.1 deki gibidir. Bu program ile başında çalışan olup olmamasına bakılmaksızın üretim ortamında bulunan ve çalışabilir durumda bulunan tüm üretim araçları tanımlanmaktadır. Burada adı geçen **üretim aracı tipi**; üretim ortamındaki üretim aracı grubunu belirler. Aynı gruptan birden fazla üretim aracı olması durumunda **üretim aracı sayısı** alanına bu üretim araçlarının sayısı yazılmaktadır. **Bilinen Adı** alanına, üretim ortamında bu üretim aracı ne isimle anılıyorsa bu isim yazılır.

USDUS_P_02 UZMAN SİSTEME DAYALI ÜRETİM SİSTEMİ "USDUS" 15/01/1995

Üretim Araçları Tanımlama Programı

Ü.Aracı Tipi	Ü.Aracı Sayisi	O. Bozulma Süresi OBS	O. Onarım Süresi OOS	Üretim Aracının Bilinen Adı
1	18	10	60	OVERLOK MAKİNASI
2	10	10	120	KESİM MAKİNASI
3	15	10	25	SİNGER
4	3	10	128	İLİK AÇMA MAKİNASI
5	20	18	10	ROMEYÖZ
6	8	10	120	TEYEL MAKİNASI
7	10	10	10	REŞME MAKİNASI
8	5	10	10	PUNTARAJ MAKİNASI
9	2	100	10	DÜĞME DİKME MAKİNASI

F2 : KABUL F3 : SON F4 : İPTAL

Şekil 4-1. Üretim araçları tanımlama programı.

Girilen kayıtlar DATA dizinindeki OUT_ISMK.000 dosyasında tutulur. Simülasyon esnasında buradaki bilgiler okunarak üretim ortamının ana hatları ekranında belirtilir.

Şekil 4.1 de bulunan Üretim Araçları tanımlama programına Tablo 4-3 de tanımlanan değerlere göre olması gereken gerçek değerler girilmiştir.

4.2.2.1.2 Makinaların Ortalama Bozulma ve Tamir Sürelerinin Belirlenmesi

Simülasyon sürelerinin belirlenmesinde Ortalama Bozulma Süresi (**OBS**) ve Ortalama Onarım Süresi (**OOS**) önemli parametrelerdir. Bu parametreler üretim araçları grupları bazında belirlenir.

Ortalama Bozulma Süreleri ve Ortalama Onarım Sürelerinin simülasyon yazılımına tanıtımı da yine Şekil 4-1'de verilmiş olan Üretim Araçları Tanımlama programı tarafından yapılmaktadır. Üretim Araçları Tanımlama programında bulunan OBS ve OOS alanlarına girilen bilgiler tarafından yapılmaktadır. Buraya girilen Tanımlama süresi simülasyon programı tarafından saat olarak algılanır. Onarım süresine girilen bilgi ise dakika olarak algılanır. Gerçek hayatta da üretim araçlarının bozulma aralıkları günler mertebesinde, tamir süreleri ise saatler mertebesindedir.
Üretim araçlarıyla ilgili işlemler yapılırken poisson dağılımından faydalanılır. Simülasyon programının onarım ve tamir süreleriyle ilgili sürelerde poisson dağılımını ne şekilde kullandığına Bölüm 4.2.2.3.8'de değinilmiştir.

4.2.2.1.3 Çalışanların Üretim Araçlarına Atanması

Üretim ortamında üretim başladığı anda çalışanların yerleşim durumlarının simülasyon yazılımına bilgi olarak verilmesi gerekmektedir. Makina başına çalışanın var olup olmadığı bilgisinin verilmesi simülasyon sistemi açısından yeterli değildir. Çünkü her çalışan her makina grubunda aynı verimle çalışamamakta ya da bazı çalışanlar bazı üretim aracı gurubunda hiç çalışamamaktadır. Çalışanların üretim araçları grubunda ne gibi bir ilişkiyle çalıştıkları Tezin 2. bölümündeki Tablo 2.1'de tanımlanmıştır. Bu tablonun da gözönüne alınmasıyla birlikte, üretim aracı başında çalışanın olup olmaması yanında, hangi çalışanın hangi üretim aracının başında bulunduğu bilgisinin de simülasyon sistemine verilmesi gerekecektir.

Bu işlemler, çalışanların üretim araçlarına atanması başlığı altında toplanabilir. Şekil 4.2'de sözü edilen bu işlemin nasıl gerçekleştirildiği görülmektedir.

USDUS_P_04 UZMAN SİSTEME DAYALI ÜRETİM SİSTEMİ "USDUS" 15/01/1995

Üretim Araçlarına Çalışanları Atama

Ü.Aracı Tipi	Ü.Aracı Numarası	Çalışan Kodu
001	001	001
001	002	002
001	003	003
001	004	004
001	005	005
001	006	006
001	007	007
001	008	008
001	009	009
001	010	010
001	011	011
001	012	012
001	013	013

F2 : KABUL F3 : SON F4 : İPTAL

Şekil 4-2. Üretim araçlarına çalışanları atama programı.

4.2.2.1.4 Çalışan İnsanların Tanıtılması (Beceri Tablosu)

Tezin 2.bölümünde, çalışanlar alt başlığı altında anlatılmış ve Tablo 2.1 ile tanımlanmış bulunan çalışanlarla, üretim araçlarının arasındaki ilişkilerin simülasyon sistemine tanıtılması bu program ile gerçekleştirilir. Şekil 4.3'de Çalışan - Makina İlişkilendirme programının ekran görüntüsü görülmektedir. Şekildeki 00 değerleri, o makinayı o çalışanın kullanamayacağı anlamına gelmektedir.

USDUS_P_03 UZMAN SİSTEME DAYALI ÜRETİM SİSTEMİ "USDUS" 15/01/1995

Çalışanlar - Üretim Araçları Arasındaki İlişkileri Tanımlama

İnsan \ Makina	1	2	3	4	5	6	7	8	9
1	42	64	46	00	51	43	90	90	51
2	57	55	64	64	48	92	45	77	45
3	90	88	72	68	99	89	65	78	00
4	62	51	47	51	63	00	49	84	47
5	65	42	00	52	57	70	69	66	95
6	62	48	63	51	85	70	99	42	69
7	00	41	62	97	57	44	88	55	91
8	67	83	84	48	98	97	91	58	58
9	87	95	93	98	47	50	00	49	67
10	84	65	52	51	45	59	68	85	94
11	89	84	83	88	54	65	44	60	46
12	65	70	84	54	63	46	88	63	58
13	53	64	57	67	68	00	59	44	46

F2 : KABUL F3 : SON F4 : İPTAL

Şekil 4-3. Çalışan insanların tanıtımı programı.

Simülasyon yazılımı açısından beceri tablosunun önemi büyüktür. Uzman sistem algoritması çalıştığında, çalışanların yerlerini değiştirmeden önce, çalışanların kaydırılmak istendiği üretim aracı grubundaki becerisine bakmakta ve karar vermesi için bu değerin belirlenmiş ve hazır olması gerekmektedir.

Program girilen bilgileri DATA dizinindeki OUT_INMK.000 dosyasına yazmaktadır. Simülasyon yazılımı da buradan aldığı bilgilerle ekranı düzenlemektedir.

4.2.2.1.5 Modellerin Tanıtılması

Simülasyon yazılımı tarafından yapılacak işlemlerin belirtildiği programdır. Tezin 2. Bölümünde Tablo 2-2'de işler ve üretim araçları arasındaki ilişkilerin nasıl olabileceği belirtilmiştir. Modellerin simülasyona tanıtılması işlemi Şekil 4.4'de görülen ekran yardımıyla gerçekleştirilir.

Programda iş grubu adıyla adlandırılan model numaralarıdır. Simülasyon sisteminde model numarası için üç uzunlukta yer ayrılmıştır. Böylelikle sisteme 999'a kadar model tanıtmak mümkündür. İşlem adımları birbirini takip eder ve modelin bir sonraki adımda hangi üretim aracına uğrayacağını belirtmek için kullanılır. 2-7 de tanımlanan t_i işlem süresi sabit ve değişken olarak ikiye ayrılabilir. İşlem süresi bileşenlerini oluşturan ve insandan bağımsız olan işin, makinada çalışma süresi t_h, bu programda Çalışma Süresi (sabit) sütununa girilir. İşlem süresinin diğer bileşenleri, işin başındaki insana bağımlı olup Çalışma Süresi (değişken) sütunundan programa tanıtılır.

Son şekillerde anlatılan programların ortak özellikleri, aşağıya doğru ilerledikçe arka sayfaları bulunmasıdır. Arka sayfalara, PageDown tuşuyla ya da aşağı ok tuşuna basmak suretiyle geçmek mümkündür.

İş Bilgileri tanımlama programı, çıktı olarak DATA dizinindeki OUT_IS.999 dosyalarına yazılır. Buradaki 999 uzantısı model numarasını belirtmektedir.

USDUS_P_01 UZMAN SİSTEME DAYALI ÜRETİM SİSTEMİ "USDUS" 15/01/1995

İş Grubu Tanımlama Programı

İş Grubu No : 004 **Varyant Sayısı** : 12
İş Kodu : KB125 **İş Adı** : BISIKLET YAKA KAZAK

İşlem Adımları	Kul Prm	Çalışma Süresi (sabit)	Çalışma Süresi (değişken)	Üretim Aracı
I	k	th	to	g
1	1	1	36	1
2	1	1	12	2
3	1	1	24	3
4	1	1	12	4
5	1	1	48	5
6	1	1	12	6
7	1	1	24	7
8	1	1	12	8
9	1	1	12	9

F2 : KABUL F3 : SON F4 : İPTAL

Şekil 4-4. Modellerin tanıtımı programı.

4.2.2.1.6 Model İçindeki Çeşitlilik Sayılarının Belirtilmesi

Gerçekte modeller renk/varyant ve beden sayıları ile birlikte tanımlanır. Simülasyon işleminde, herbir modelin bir renk/varyant ve bir bedeni, bir iş olarak kabul edilmiştir. Bu nedenle bir model için (renk/varyant) * (beden sayısı) kadar iş (çeşitlilik) ortaya çıkmaktadır. Simülasyon programında, renk ve bedenler ayrı ayrı gösterilemediği için çeşitlilik sayısı olarak adlandırılabilecek ve model numarası ile model içerisindeki sıra numarasından oluşturulan bir numaralandırma sistemi takip edilmiştir.

Simülasyon ekranında, en solda sepetler yerleştirildikçe hesaplanan çeşitlilik numaraları sepet kutucuklarının sol kısmına yazılır.

4.2.2.1.7 Süreç Adımlarının ve Sürelerinin Belirlenmesi

4.2.2.1.5'de anlatıldığı biçimde bir model tanıtılırken, modelin uğrayacak olduğu üretim araçlarının sırayla belirlenmesi gerekmektedir. Tablo 2.2'de tanımlanmış olan ilişkide k kullanım parametresinin 1 olduğu her bir satırın süreç adımı olarak simülasyon yazılımına tanıtılması gerekmektedir. Tanıtım esnasında iki zaman parametresi bulunmaktadır. Bu parametrelerden ilki, sabit olandır ve sepetin ilgili üretim aracında geçirecek olduğu sürenin, çalışandan bağımsız olan ve sadece makinaya bağlı olan kısmının girildiği parametredir. İkinci zaman parametresi ise, sepetin üretim aracında bulunduğu sürenin, çalışana bağımlı olan kısmını oluşturur. Bu parametrenin diğerinden farkı, üretim aracı başında bulunan operatöre bağlı olarak, sürenin değişken olmasıdır. Daha önceden Bölüm 4.2.2.1.4'de sözü edilen beceri tablosu girişine göre burada bildirilen süreler simülasyon esnasında değişecektir.

4.2.2.2 Parametreler

Simülasyon yazılımının çalışmasını etkileyen bir takım parametreler programın dışında, DATA dizininde param1.txt dosyasında bulunur. Parametrelerin değerine göre, simülasyon yazılımının çalışması değişmektedir. Bu parametrelerin neler olduğu ve parametrelerin değerlerine göre simülasyonun nasıl bir davranış takip edeceği bu bölümde irdelenmiştir.

4.2.2.2.1 USDUS Parametresi (Var / Yok)

Parametre dosyanın ikinci satırı şu şekildedir.

- Uzman sistem yaklaşımı : Var

Simülasyon yazılımı, bu satırdan okuyacak olduğu, "Var" ya da "Yok" bilgisine göre, USDUS algoritmalarını çalıştırır veya ortamı kendi haline bırakır, müdahale etmez. Üretim ortamında, simülasyonun çalışma süresini belirleyen en önemli parametrelerden biridir. Tezin sonraki bölümlerinde bu parametre ile oynanmak suretiyle çıkan sonuçlar irdelenmiştir.

4.2.2.2.2 Kuyruk Eşik Seviyesi

Parametre dosyasının üçüncü satırı şu şekildedir:

- Kuyruk Eşiği : 5

Kuyruk Eşiği parametresi Uzman sistem yaklaşımı ile birlikte anlamlı olan bir parametredir. Uzman sistem yaklaşımı algoritması her adımda çalışmaz. Uzman sistem algoritmasının çalışması için simülasyon yazılımının sistemin durumunun kötüye gittiğini anlaması gerekmektedir. Bu da kuyruk eşiği parametresinin uzunluğu ile bağlantılıdır. Eşiği aşan kuyruklar olduğunda simülasyon yazılımı uzman sistem desteğine başvurur.

Bu yüzden sistemin kuyrukları hangi seviyede tutulmak isteniyorsa, bu seviyeyle ilgili eşik değerinin belirlenerek parametre dosyasının ilgili satırına girilmesi gerekmektedir. Kuyruk eşiği aşıldığında simülasyon programı ortasındaki mavi band kırmızı renge boyanır.

4.2.2.2.3 Öngörü Simülasyon Adım Sayısının Belirlenmesi

Parametre dosyasının dördüncü satırı şu şekildedir:

- SIS Adım Sayısı : 120

Bu parametre de sadece Uzman Sistem Yaklaşımı ile birlikte anlamlıdır.

Bir önceki parametre olan, kuyruk eşiği parametresine göre, uzman sistem algoritması çalıştırıldığında, bu parametreyle belirlenen öngörü adım sayısına göre, simülasyon kendi içerisinde, uzman sistem algoritmasının yapmış olduğu değişikliği simüle ederek, sonucun olumlu olup olmadığını test eder. Simülasyon içerisinde simülasyon olarak tanımlanabilen, bu ikinci simülasyonun derinliğini bu parametre belli eder. Bu parametrenin normalde, bir üretim ortamında yarım günde geçen süre kadar olması gerekir. Sebep olarak, yapılan bir değişikliğin neticesini görebilmek için, yapılan değişiklik ve bu değişikliğin sebep olduğu üretim akışı değişikliklerinin hepsini görebilmek için, yarım iş gününün geçmesi gerekmektedir.

Parametre değeri daha küçük tutulduğu durumda simülasyonun fazla ilerisi öngörülemez, fakat öngörüldüğü kadarıyla da yapılan değişiklik hakkında bir fikir elde edilebilir. Parametre büyütüldüğünde, ortaya çıkacak olan zaman kavramını da göz önünde tutarak, bu öngörü parametresinin optimumu tespit edilerek, uzman sistem yaklaşımı algoritması bunun etrafında çalıştırılmıştır.

4.2.2.2.4 Durum Tablolarının Yazılacağı Dosyaların Belirlenmesi

Parametre dosyasının beşinci satırı şu şekildedir:

- Kuyruk Uzunluklarının Yazılacak Olduğu Dosya : bit??.out

Burada belirtilmiş olan dosyalar, simülasyon esnasında DATA alt dizininde oluşturularak, simülasyon bitene kadar **bit??.out** dosyasına adım adım kuyruk değerleri ; **bib??.out** dosyasına ise, aynı sürelerdeki sistem bekleme süreleri yazılır. " **??** : burada iki basamaklı değişken bir sayıdır ".

Simülasyon sisteminde bulunan grafik programları ile oluşturulmuş olan bu dosyaların grafik olarak birbirleriyle ilişkilerinin incelenmesi olasıdır. Tezin bundan sonraki bölümlerinde çizilmiş olan grafikler, sözü edilen karşılaştırma programları kullanılmak suretiyle üretilmiştir.

4.2.2.3 Simülasyon

Simülasyon yazılımı veriler ve parametreler belirlenmek suretiyle, çalışabilir seviyeye gelmektedir. Parametre dosyalarından uzman sistemle ilgili parametreler alındıktan sonra, üretim araçları; dosyasından okunarak sayılarına göre ekrana yerleştirilir. Ardından okunan, çalışan-üretim araçları ilişki tablosuyla, çalışanlar makina başlarına yerleştirilir. Çalışanların yerleştirilmesiyle, boşta kalan üretim araçları belli olacağından bu makinalar da ekranda modellenir. Bundan sonraki adımda modeller okunmaya başlanır. Okunan model, sepetleriyle birlikte ekrana grafik olarak yerleştirilir. Bu işlem de bittiğinde simülasyon çalışabilir duruma gelmiştir. Başlamak için bir tuşa basılmasını bekler. Simülasyon başladığı andan itibaren, parametre dosyasında bildirilmiş olan dosyalara, kuyruk ve boşta bekleme değerlerini yazar. Simülasyon esnasında oluşan kuyruklar, uzman sistem yaklaşımının çalıştırılması, kuyrukların eşikleri aştığı anlar ekrandan görsel olarak izlenir. Simülasyon sonlanana kadar ekranda verilen bilgilerle adım süreleri, sistemde en son arızalanan üretim aracı numarası, en son tamir edilerek devreye alınan üretim aracı sayıları, biten varyans kodu bilgileri devamlı olarak izlenebilir.

Simülasyon yazılımının bu işlemleri yaparken izlediği yöntemler detaylı olarak bu şekilde incelenecektir.

4.2.2.3.1 Kullanıcı Arabirimi (Grafik)

Simülasyon yazılımının grafik arabirimi gerçekleştirilirken, gerçek üretim ortamına akış ve görünüm olarak benzetilmeye çalışılmıştır. Şekil 2.5'de üretim ortamının sembolik gösterimi verilmiştir.

Simülasyon grafik arabirimi esas olarak 4 ana bölümden oluşmaktadır. Bunlar :

- Giriş Kuyrukları
- Esas Üretim Ortamı
- Ara Kuyruklar
- Çıkış Kuyrukları

Giriş Kuyrukları :

Simülasyon yazılımında, üretim ortamında sepetler üzerinde işlem yapılmaktadır. Giriş kuyruklarında dört bölme vardır. Bir varyant için ortalama dört sepet olduğu gözönünde bulundurularak böyle bir uygulamaya gidilmiştir. Simülasyon yazılımı esas olarak ekranda gösterebileceği tüm sepetleri göstermektedir. Sepetler üretim ortamına sırayla çıkmaktadır. Giriş kuyruğunda bulunan dört sepet de üretim ortamına verildiğinde, artık bu giriş kuyruğu yeni sepet alma durumuna geçer. Simülasyon yazılımı bu durumda, ekranda yer olmadığı için henüz gösterilememiş olan sepet grubunu boşalan yere alır. Bu işlem, bu şekilde simülasyon için artık işlenecek sepet kalmayıncaya kadar devam etmektedir.

Kurulan bu algoritma nedeniyle, simülasyon sonu hariç, herhangi bir anda giriş kuyruklarına bakıldığında, her zaman için hepsinin dolu olduğu görülür.

Bir sepetin giriş kuyruğundan hareket ederek üretim ortamına girebilmesi için, sepetin çalışacak olduğu ilk üretim aracı grubundaki herhangi bir üretim aracının, bu sepeti işleyecek durumda olması gerekmektedir. Gerçek üretim ortamında da sepet dağıtımında böyle bir yöntem uygulanmaktadır. Bu kontrol uygulanmadığı takdirde bir anda tüm sepetler üretim ortamına dağılacak, sonuçta bir karışıklık meydana gelecektir.

Fiziksel ortamdaki bu durum ve çözümü, aynen simülasyon yazılımına da aktarılmıştır.

Sepetler giriş kuyruğuna yerleştirildiğinde, varyantta bulunan toplam sepet sayısı kadar yerleştirilir. Sepetler üretim ortamına ise teker teker verilir. Bunun sonucu olarak her bir giriş kuyruğu satırında yalnızca bir renkteki sepetler bulunmaktdır. Bir sepete, giriş kuyruğuna giriş sırasına göre renk verilir. Verilen bu renk, tüm üretim ortamı boyunca, yani sepet üzerindeki işlem bitinceye kadar değişmez.

Bu yöntem sepetlerin üretim ortamında kolay tekip edilebilir olması açısından uygulanmıştır.

Esas Üretim Ortamı :

Şekil 2.5'deki sembolik gösterimde giriş, çıkış ve ara kuyruklar arasındaki büyük alan simülasyon yazılımı tarafından üretim ortamı olarak kullanılır. Buradaki geometrik şekiller ise üretim ortamındaki üretim araçlarını temsil etmektedir. Aynı geometrik şekiller üretim aracı gruplarını temsil eder. Aynı grupta kaç tane üretim aracı bulunduğu, aynı tip geometrik şekilden kaç tane olduğundan anlaşılır.

Simülasyon yazılımı ilk çalıştırıldığında, daha önceden girilmiş olan üretim ortamının üretim aracı parkına göre, birer geometrik şekile atanarak gösterilir. Üretim araçlarının simülasyon yazılımına ne şekilde tanıtıldığı, bölüm 4.2.2.1.1'de daha önceden bahsedilmiştir. Üretim araçlarını tanımlayan programın ekran görüntüsü ise Şekil 4.1'de verilmiştir.

Üretim ortamında bulunacak olan üretim araçlarının tipleri, sayıları, özellikleri (bozulma ve onarım süreleri), simülasyon yazılımı çalıştırılmadan önce tanımlanmış olmalıdır. Tanımlanmış olan üretim araçlarına göre simülasyon yazılımı, kullanıcı grafik arabirimi üzerinde yerleştirmeyi kendiliğinden yapar. Simülasyon çalıştırılır çalıştırılmaz gelen grafik ekranda, üretim araçları yerine yerleştirilmiş şekliyle esas üretim ortamı bulunur.

Esas üretim ortamı 10 sütundan oluşur. Gösterimi kolaylaştırmak amacıyla iki bölmeye ayrılmıştır. Toplam olarak 20 hücre bulunmaktadır. Her hücreye 5 üretim aracı yerleştirilebilir. Buradan da anlaşılacağı üzere, simülasyon üretim ortamında 100 adet üretim aracının sığabileceği alan bulunmaktadır. Esas üretim alanında, bir grup üretim aracı için ayrılan en küçük alan, 5 üretim aracı alacak büyüklüktedir. Yani bir üretim aracı tipinden 2 tane üretim aracı olsa bile, esas üretim ortamı grafik alanından 5 üretim araçlık yer sarfedilir. Bunun yanında bir üretim aracı grubundan 5'den fazla üretim aracı olduğunda, bir sonraki 5'li gruba, daha fazla bulunuyorsa yandaki hücrelere doğru esas üretim alanı doldurulur.

Esas üretim ortamında tüm makinalar yerliştirildikten sonra, kullanıcı arabirimi tarafından kullanıcının bir tuşa basması beklenir.

Bir tuşa basıldıktan sonra simülasyon başlar. Simülasyon başlamadan önce, hangi dosyalardan ilk değerleri okuyacağı bundan sonraki bölüm olan 4.2.2.3.2'de anlatılacaktır.

Simülasyonun grafik ekranında üretimin akış yönü soldan sağa doğru olacaktır. Bir sepetin bu akış yönünde ilerlerken akış yönünün aksi yönde bir makinaya gitmesi olasıdır. Fiziksel üretim ortamında da bazı durumlarda bir sepet, bir makinaya, bir işlem için birkaç kez uğrayabilmektedir. Bu durumda da, üretim akış yönünün ters yönünde gidiş gelişler söz konusu olmaktadır. Simülasyon programında, bu tür durumlara karşı, çözüm bulunmaktadır.

Normalde üretim araçları, içleri boş birer geometrik şekille gösterilmektedir. Üzerlerine iş yüklenen üretim araçları, sepetlerin rengine göre renk değiştirmektedir. Simülasyon grafik arabirimine bakıldığında boşta duran ve çalışan üretim araçları kolaylıkla anlaşılabilmektedir.

Fiziksel ortamda, üretim araçlarının üzerlerine iş kabul edemediği durumlar bulunmaktadır. Bu durumlardan biri başlarında çalışan olmaması, diğer durum ise üretim araçlarının bozuk ya da tamirde olmasıdır. Başlarında çalışan olmayan üretim araçları, simülasyon ekranında, boş geometrik şekil ve içerisinde mavi nokta ile temsil edilir. Bozuk ya da tamirde olan üretim araçları da boş geometrik şekil ve içerisinde kırmızı nokta ile gösterilir.

Fiziksel üretim ortamında çalışanların yedek ve biten sepetleri bulunmaktadır (Ayrıntılı bilgi için bkz. Bölüm 2.4. Uzman Sisteme Dayalı Üretim Yönetimi). Simülasyonda da her bir üretim aracına ait yedek ve biten iş sepetleri bulunmaktadır. Yedek ve biten sepetler geometrik üretim aracı şekillerinin sağında ve solundaki kutularla temsil edilir. Soldaki yedek sepette, üretim aracında işlenen sepet bittikten sonra üzerinde çalışılacak iş bulunur. Yedek sepet kutusu, üzerinde bulunan yedek sepetin rengini alır. Üretim aracındaki sepetin işi bittiğinde, bu sepet sağdaki biten sepet kutusuna yerleştirilir. Sepet buradan alınıp bir sonraki üretim aracına sevkedilene kadar biten sepet kutusunda kalır.

Ara Kuyruklar :

Esas üretim ortamının üst kısmında oluşan ara kuyruklar Şekil 2.5'de gösterilmiştir. Biten iş kutularında bulunan sepetler bir sonraki işlem adımına gönderilmek üzere buralardan alınır. Sepetin bir sonraki adımda işlenecek olduğu üretim aracı grubunda boşta bulunan üretim aracı yoksa, bu sepet ilgili üretim aracı grubunun ara kuyruğuna atılır. Ara kuyruk oluşması ve sebepleri Bölüm 2.4'de incelenmiştir.

Simülasyonun grafik gösteriminde ara kuyruklar, yanyana en fazla 10 uzunluğa kadar büyüyebilecek 3 kolondan oluşmaktadır. Bu durumda bir üretim aracının başında 30 sepete kadar kuyruk oluşabilir. Bu sayıdan daha fazla kuyruk oluşması durumunda simülasyon yazılımı ara kuyruktaki sepetleri gösteremese de bunları hafizasında saklar. Simülasyon yazılımının grafik ara birimi, ekranda bulunan ara kuyruktaki sepetlerden eksilme olduğunda, hafizasında tutmuş olduğu sepetleri (en fazla 30 sepete kadar) göstermektedir.

Ara kuyruklar, kuyrukta bekleyen sepetlerin renklerini almaktadır. Bu sayede hangi sepetlerin ara kuyruklarda bekledikleri görsel olarak anlaşılmaktadır.

Ara kuyruklar hem ekranın üst kısmında, hem de alt kısmında oluşabilir. Ara kuyrukların nerede oluşacağını belirleyen, üretim araçlarının ekrana yerleştirilme biçimleridir. Daha önce anlatıldığı gibi, simülasyon yazılımı tanımlanan üretim araçlarını 5'li gruplar halinde simülasyon ekranına yerleştirmektedir. Bu durumda üretim aracı grubundaki ilk üretim aracı ekranın üst kısmından başlıyorsa, bu üretim aracı grubunun ara kuyruğu ekranın üst kısmında, ekranın alt kısmından başlıyorsa, bu üretim aracı grubunun ara kuyruğu ekranın alt kısmında oluşmaktadır.

Ara kuyruklarda bulunan sepetler, bir sonraki adım olarak üretim araçlarının yedek sepetlerine alınmaktadır.

Çıkış Kuyrukları :

Üretim ortamında tamamen işlenmiş bulunan sepet, simülasyon ekranının sağında bulunan çıkış kuyruklarına alınır. Çıkış kuyrukları giriş kuyruklarından farklı olarak her rengin toplandığı satırlardan oluşmaktadır. Bir kuyruğa dört sepet sığabilmektedir. Kuyruk sepetlerle tamamen dolduğunda boşaltılmakta, yeni gelecek olan sepetlere hazırlanmaktadır.

4.2.2.3.2 Veri Girişi ve Parametrelerin Okunması

Bölüm 4.2.2.1'de, iş grubu tanımlama, üretim araçları tanımlama, çalışan-üretim araçları arasındaki ilişkileri tanımlama, üretim araçlarına çalışanları atama programları ve bunların nasıl çalıştığı anlatılmıştı.

Bölüm 4.2.2.2'de ise simülasyon yazılımı tarafından kullanılan ve yazılıma yön veren parametrelerden bahsedilmiş ve bunların nereden okunacağı, hangi değerleri alabileceğinden bahsedilmişti.

Simülasyon yazılımı parametre dosyasını program ilk başladığı anda okumaktadır. Buradaki USDUS parametresi ile yazılım izleyeceği yöntemi belirlemekte, diğer parametrelerle de kuyruk oluşumları ve çözümleri hakkında bilgi edinmektedir.

4.2.2.3.3 Durum Tablolarının Oluşturulması

Durum tabloları; simülasyonun sonuçlarının yazıldığı dosyalara verilen genel addır. Simülasyon yazılımı sonradan incelenmek üzere bu değerleri üretmektedir. 4.3'de anlatılan simülasyon sonuçlarının değerlendirilmesi başlığı altında kullanılan değerler, durum tablolarından elde edilmektedir.

Durum tablolarının yazılacak olduğu dosyaların isimleri, birer parametre olarak parametre dosyasından alınmaktadır. Bölüm 4.2.2.2.4'de dosya isimleri hakkında ayrıntılı bilgi bulunmaktadır.

Simülasyon devam ederken her adım arasında, durum tablolarıyla ilgili kısma dallanılarak bu adımda sistemde bulunan toplam kuyruk uzunluğu ve toplam bekleyen sayısı hesaplanır. Hesaplanan bu değerler durum tablosu verisi olarak ilgili dosyalara yazılırlar. Simülasyon devam ettiği müddetçe, her adım için bu işlem yapılmaya devam edilir.

4.2.2.3.4 Uzman Sistem Desteği

Parametre dosyasından alınan kuyruk eşik seviyesi, simülasyon sırasında oluşan ara kuyrukların tehlike değerini belirler. Tehlikeli eşik seviyesini aşan durumlarda simülasyon programı uyarı verir. Esas üretim ortamı başlığı altında anlatılmış olan, iki parçalı grafik ekranın arasında, normalde mavi olan uyarı çubuğu, bir aşım durumuyla karşılaşıldığında kırmızı rengi alır. Aşım durumu devam ettiği müddetçe uyarı çubuğu kırmızı kalmaktadır. Aşım bittiğinde ise tekrar eski mavi halini almaktadır.

Uyarı çubuğu kırmızı olduğu durumlarda, simülasyon programı USDUS modunda çalıştırılıyorsa (Bölüm 4.2.2.2.1), simülasyonun uzman sistem kısmını çalıştırır. Simülasyonun uzman sistem algoritmasının akış diyagramı Şekil 3.6'da verildiği gibidir. Bu aşamada bir diğer parametre olan öngörü simülasyon adım sayısı (Bölüm 4.2.2.2.3) gözönüne alınarak yeni bir simülasyon yapılır. Bu simülasyon sonucu yeniden uzman sistem algoritmasının akış diyagramı (Şekil 3.6) gözönünde bulundurularak bir karar verilir ve bu karar uygulanır.

Temel olarak anlatılan uzman sistem desteği sonucunda, elde edilen sonuçlar karşılaştırmalı olarak ilerki bölümlerde anlatılmıştır. Uzman sistem algoritmasının C++ dilinde yazılmış olan kaynak kodları Ek A'da verilmiştir.

4.2.2.3.5 Gerçek Zaman Saati

Simülasyon yazılımı, işlem adımı için kendi içerisinde tuttuğu zaman sayacı ile ayarlamaktadır. Yazılım kendi içerisinde tanımlanmış bulunan fonksiyon tuşları yardımıyla hızlandırılıp yavaşlatılabilmektedir. Buna rağmen simülasyon sistemindeki her adımın gerçek zamanla ilişkisi bir parametre ile sağlanmaktadır. (Bölüm 2.7.1)

4.2.2.3.6 Rastgele Sayı Üretimi

Simülasyon boyunca rastgele sayılar üretilmesi gereken bölümler bulunmaktadır. Bu aşamalar :

- Arıza oluşturma aşaması
- Tamirin tamamlanma aşaması
- Çalışan hastalanma aşaması
- Çalışan izin alma aşaması

Bu aşamalarda gerek duyulan rastgele sayıları üretebilmek için program içerisinde bir rastgele sayı üreteci tanımlanmıştır. EK B.'de rastgele sayı üretimi yapan program parçası tanıtılmıştır.

4.2.2.3.7 Normal Dağılıma Dönüştürme

Bölüm 2.3.2'de Üretim Araçlarının Sağlamlığı başlığı altında anlatıldığı gibi bir üretim aracının onarım süresi normal dağılıma göre hesaplanmaktadır. Hesaplama işlemi için, C dilinin özellikleri kullanılmıştır. Normal dağılım değerlerinin hesaplanmasına baz olan değerler ise makina tanımlamaları yapılırken girilen OOS (Ortalama Onarım Süresi) 'lerdir.

Üretim araçları bozulduğunda OOS ve Normal dağılım göz önüne alınarak yapılan hesaplama neticesinde bulunan süre değeri, makina değişkenine yazılmakta; bu süre kadar süre geçene kadar makina onarımda kalmakta, bu süreye gelindiğinde ise üretim aracı onarılmış kabul edilerek üzerine tekrar iş kabul eder duruma geçirilmektedir.

4.2.2.3.8 Poisson Dağılımına Dönüştürme

Bölüm 2.3.2'de Üretim Araçlarının Sağlamlığı başlığı altında poisson dağılımı anlatılmış ve makinaların bozulma sürelerinin bu dağılıma göre olduğuna değinilmiştir. Poisson dağılımına baz olan ortalama değer ise, makina tanımlamaları yapılırken girilen, OBS (Ortalama Bozulma Süresi) 'dir.

Simülasyon yazılımında OBS ortalama değerine, poisson dağılımını uygulayarak sonucu elde eden bir algoritma bulunmaktadır. Bakınız EK B.

4.2.2.3.9 Arıza Oluşturma

Üretim araçlarının, simülasyon tarafından okunduğu başlangıç aşamasında, üretim aracıyla ilgili OBS (Ortalama Bozulma Süresi) de okunmaktadır. Bu değer ve poisson dağılımı algoritması gözönüne alınarak yapılan hesaplama neticesinde, bulunan süre değeri, üretim aracı değişkenine yazılmaktadır.

Üretim aracı değişkenlerine yazılan bu değerler neticesinde, üretim araçlarının bozulacakları adım sayısı ileriye doğru simülasyon yazılımı tarafından bilinmektedir. Bu süre gerçekleşince, simülasyon yazılımı üretim aracını bozmaktadır. Bu esnada üretim aracı üzerinde işlenen sepet varsa, sepet işlendiği kadar olan kısmı kaydedilerek tekrar kuyruğa atılmakta, üretim aracı başındaki yedek sepet doluysa, buradaki sepet de ara kuyruğa atılmaktadır. Bunun ardından, üretim aracı onarımıyla ilgili simülasyon kısımları çalıştırılmaktadır.

Üretim aracı onarım süreci bittiğinde de, bu üretim aracıyla ilgili OBS (Ortalama Bozulma Süresi) ve poisson dağılımı algoritması gözönünde bulundurularak, yeniden bir sayı üretilmektedir. Yeni üretilen bu sayı neticesinde, üretim aracının tekrar ne zaman bozulacağı tespit edilmektedir. Yukarıdaki bozulma algoritmasının tekrar çalışabilmesi için, yeni yazılan bu süre kadar simülasyonun ilerlemesi beklenecek ve süreye ulaşılınca makina tekrar bozulacaktır.

4.2.2.3.10 Çalışanların Hasta Olması

Üretim ortamında yapılan gözlemler neticesinde; günlük işe gelmeme oranı %5, mesai süresince çeşitli nedenlerden çalışmama oranı ise ortalama %2 olarak saptanmıştır. Bu değerler ışığında algoritma gereği, çalışanların içinde her an için belirtilen oranda hasta ve izinde çalışan olup olmadığı araştırılmakta, eğer bu orandan az çalışan, hasta ya da izinliyse, sağlam elemanlar hasta edilmektedir.

Hasta edilecek elemanların tesbitinde rastgele sayı üreteci kullanılır. Buna göre seçilen çalışan eleman üzerinde iş bulunma olasılığı da bulunmaktadır. Eğer rastgele seçilen çalışan üzerinde iş bulunmaktaysa, eleman üzerindeki iş, kalan süresi işlenmek üzere ara kuyruğa atılmaktadır. Ayrıca çalışanın kullandığı üretim aracına ait yedek sepet boş değilse, bu sepet de ara kuyruğa atılmaktadır.

Bundan sonra üretim aracı başında çalışan olmadığı, grafik olarak belirtilmektedir. Hasta edilen çalışanın kullandığı üretim aracı, üzerinde kuyruk oluşmuş bulunan, yada bu çalışanın hastalanması sonucunda yeni bir ara kuyruk oluşabilecek bir durumda olabilir. Bu sorun, daha önce anlatılmış olan Uzman Sistem yaklaşımı ile çözülecektir.

4.3 Simülasyon Sonuçları

Simülasyon boyunca oluşturulmuş olan durum tablolarının incelenmesi için bazı tanımların yapılması ve Bölüm 4.3'den itibaren verilecek sonuçların bu tanımlar uyarınca verilmesi gerekecektir. Bu amaçla bu bölümde, bazı tanımlar verilmiş ve bu tanımlar için hesaplama gerekiyorsa, gereken hesaplamanın nasıl yapılacağı tespitleri yapılmıştır.

4.3.1 Toplam İşlem Süresi (TİS)

"Toplam işlem süresi" (**TİS**) şu şekilde hesaplanabilir. Simülasyonun başlaması ile bitmesi arasındaki süreçte, simülasyonun kaç adımda bitmiş olduğu ekran çıktısı olarak bilinmektedir. Simülasyonun adım sayısı ile süre birebir kabul edilirse; bu sayı aynı zamanda toplam işlem süresini verir.

4.3.2 Toplam Sarfedilen İşgücü Süresi (TSİS)

Toplam işlem süresi, ortamda bulunan çalışan adediyle çarpıldığında, toplam olarak harcanan insan gücü bulunur. Bu şekilde hesaplanan süreye, "*Toplam Sarfedilen İşgücü Süresi*" (**TSİS**) adı verilmiştir.

TSİS = TİS * 69 olarak hesaplanır.

4.3.3 Toplam Bekleme Süresi (TBS)

Simülasyon programı, ayrıca her bir işlem adımında boşta bulunan, yani üzerinde iş olmayan çalışan sayısını da, ekran çıktısı olarak vermektedir. Buradan; toplam üretim süreci boyunca kaybedilen insan gücü elde edilir. Süre ile adım sayısı birebir kabul edilirse, toplam insan gücü "*Toplam Bekleme Süresi*" (**TBS**) ni vermektedir.

4.3.4 Kayıp İnsan Gücü (KİG)

"*Toplam Bekleme Süresi*" , "*Toplam Sarfedilen İşgücü Süresi*" ne oranlandığında, "*Kayıp İnsan Gücü*" (**KİG**)elde edilir.

KİG = TBS / TSİS olarak hesaplanır.

"*Kayıp İnsan Gücü*" % oransal değerinin, düşük olması arzu edilir. Gerçekten de, boşta bekleyen grafiklerinin incelenmesi sonucunda; USDUS kullanılmadan yapılan simülasyonlarda bu değerin, USDUS kullanılarak yapılan simülasyonlara göre daha büyük olduğu görülmektedir. Buradan, USDUS sisteminin üretim ortamında boşta iş bekleyenleri (kayıp insan gücünü) azalttığını ve işçilik maliyetini düşürdüğünü söyleyebiliriz

4.3.5 Kuyruk İntegrali (Kİ)

Simülasyon yazılımının ürettiği sonuçlardan bir diğeri ise, üretim ortamında oluşan kuyrukların hızıdır. Bu büyüklük şöyle hesaplanmaktadır: Her adım için, kuyruk boyları ölçülmekte ve tüm üretim süresince bu değerlerin toplamı (*kuyrukların integrali*) hesaplanmaktadır.

Bu değerler de, simülasyon sonucunda ekran çıktısı olarak görülmektedir.

4.3.6 Kuyruk Oluşum Hızı (KOH)

Kuyruk integrali (**Kİ**), üretim süresine oranlanarak, "*Kuyruk Oluşum Hızı*" (**KOH**) hesaplanmaktadır.

KOH = Kİ / TİS olarak hesaplanır.

"*Kuyruk Oluşum Hızı*" oransal değerinin düşük olması arzu edilir. Gerçekte, kuyruk oluşumu grafiklerinin incelenmesi sonucu; USDUS kullanılmadan yapılan simülasyonlarda bu değerin, USDUS kullanılarak yapılan simülasyonlara göre daha büyük olduğu görülmektedir. Bir üretim aracı grubunda oluşacak kuyruklar, başka bir üretim aracı grubunda iş bekleyen insanların ve makinaların oluşumuna neden olur. USDUS yaklaşımının, üretim ortamındaki işlerin beklemelerini azaltması nedeniyle, toplam üretim süresini kısaltacak ve üretim maliyetlerini olumlu yönde etkileyecektir.

4.4 Uzman Sistem Yaklaşımsız Simülasyon

Bu alt bölümde; daha önce Bölüm 4.2.2.2.1'de belirtildiği gibi, Uzman Sistem Parametresine **Yok** ifadesi yazılarak, uzman sistem yaklaşımsız simülasyon yapılmıştır. Buna göre elde edilen sonuçlar iki ana grafik altında toplanmıştır:

- Üretim Ortamında Zamana Göre Toplam Kuyruk Oluşumu Grafiği,
- Üretim Ortamında Zamana Göre Toplam Bekleyen Sayısı Grafiği.

Elde edilen sonuçların tesadüfi olmadığını göstermek için simülasyon peşpeşe 9 kez gerçeklenmiştir. Deneylerin tamamının aynı grafikte gösterilmesi durumunda, ortaya çıkacak olan karışıklığın giderilmesi için sonuçlar üçer üçer değerlendirilmiştir.

Her grafiğin altında ise Bölüm 4.3'de tanımlanmış bulunan parametrelerle ilgili sonuçlar verilmiştir.

Grafiklerin gösterimi tamamlandıktan sonra tüm deneylerin değerlerinin bir ortalaması alınmak suretiyle bir özet grafik üretilmiştir.

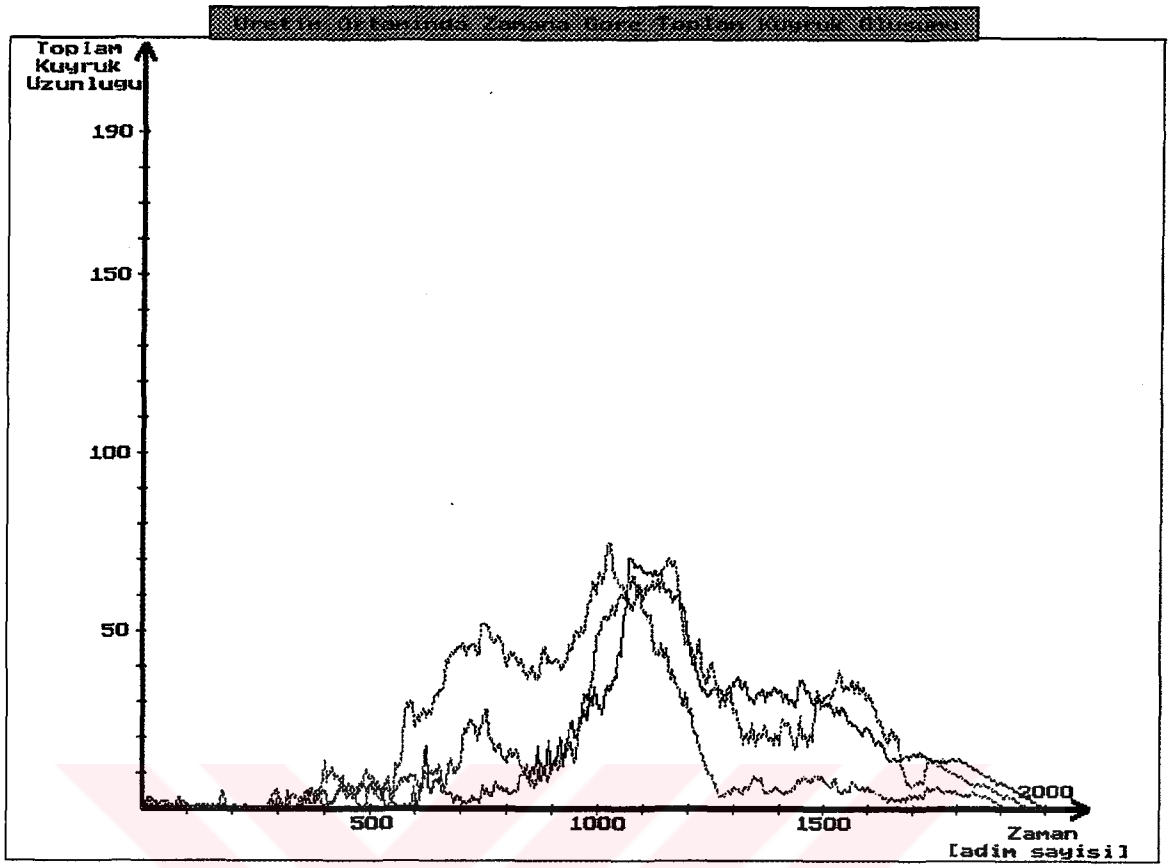
Simülasyon sisteminin stokastik olarak çalışması sonucunda hiçbir grafik birbirinin aynı değildir. Bunun için, ortalama alınır. Ancak, sistemin karakteristiklerini tüm grafiklerde izlemek olasıdır.

Şekil 4.5-7-9 ve Tablo 4.4-6-8 : USDUS kullanılmadan yapılan simülasyonlar sonucu oluşan kuyruk oluşumu grafiklerini ve kuyruk oluşumu değerlerini,

Şekil 4.6-8-10 ve Tablo 4.5-7-9 : USDUS kullanılmadan yapılan simülasyonlar sonucu oluşan boşta bekleyen grafiklerini ve boşta bekleyen değerlerini,

Şekil 4.11 ve Tablo 4.10 : USDUS kullanılmadan yapılan simülasyonların ortalaması sonucu oluşan toplam kuyruk oluşumu grafiklerini ve kuyruk oluşumu değerlerini,

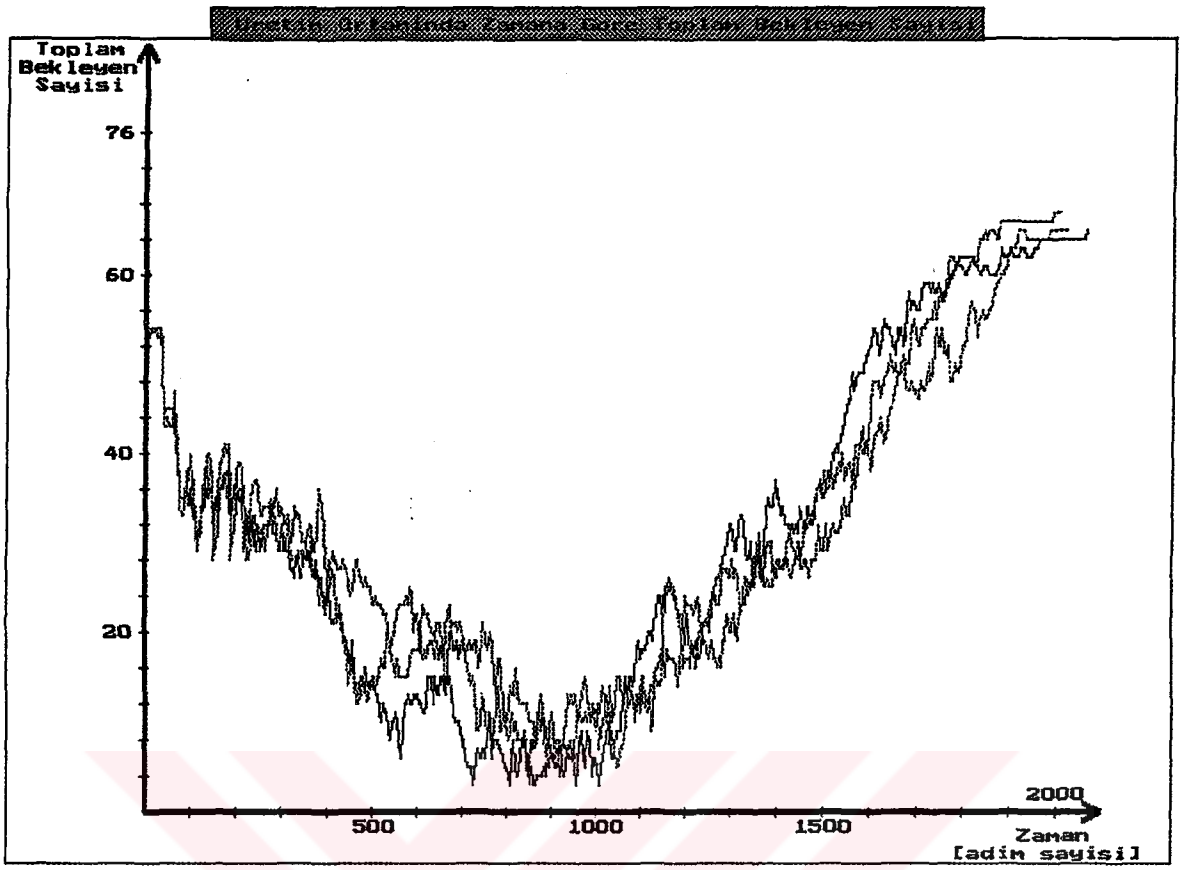
Şekil 4.12 ve Tablo 4.11 : USDUS kullanılmadan yapılan simülasyonların ortalaması sonucu oluşan toplam boşta bekleyen grafiklerini ve boşta bekleyen değerlerini, göstermektedir.



Şekil 4.5: USDUS kullanılmadan yapılan 1-2-3. simülasyonlar sonucu oluşan kuyruk oluşumu grafikleri

Tablo 4.4
USDUS kullanılmadan yapılan 1-2-3. simülasyonlar sonucu elde edilen kuyruk oluşumu değerleri

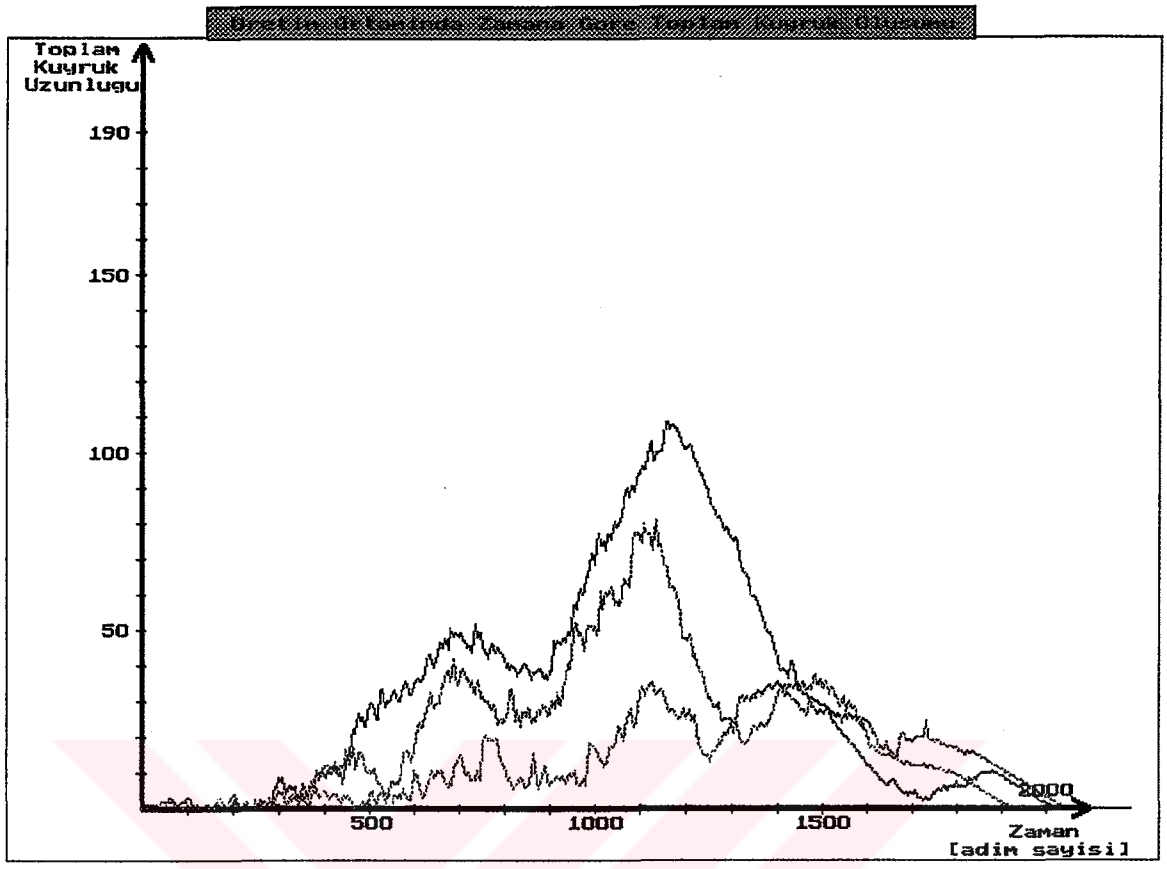
	1. Deneme	2. Deneme	3. Deneme
T.İ.S.	2071	2014	2029
T.S.İ.S.	142899	138966	140001
K.İ.	33385	22201	47441
K.O.H.	16.12	11.02	23.38



Şekil 4.6 : USDUS kullanılmadan yapılan 1-2-3. simülasyonlar sonucu oluşan boşta bekleyen grafikleri

Tablo 4.5
USDUS kullanılmadan yapılan 1-2-3. simülasyonlar sonucu elde edilen boşta bekleyen değerleri

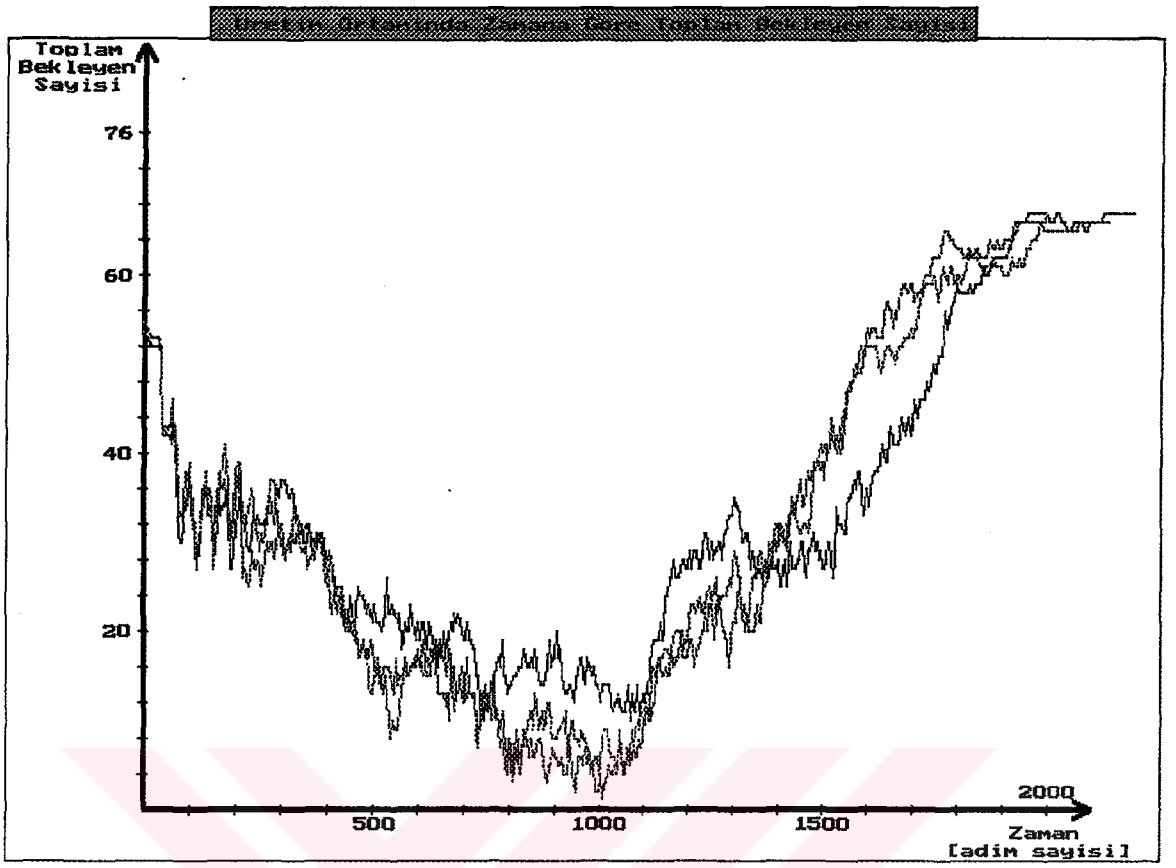
	1. Deneme	2. Deneme	3. Deneme
T.İ.S.	2071	2014	2029
T.S.İ.S.	142899	138966	140001
T.B.S.	65071	63558	61570
K.İ.G.	%45.54	%45.74	%43.98



Şekil 4.7: USDUS kullanılmadan yapılan 4-5-6. simülasyonlar sonucu oluşan kuyruk oluşumu grafikleri

Tablo 4.6
USDUS kullanılmadan yapılan 4-5-6. simülasyonlar sonucu elde edilen kuyruk oluşumu değerleri

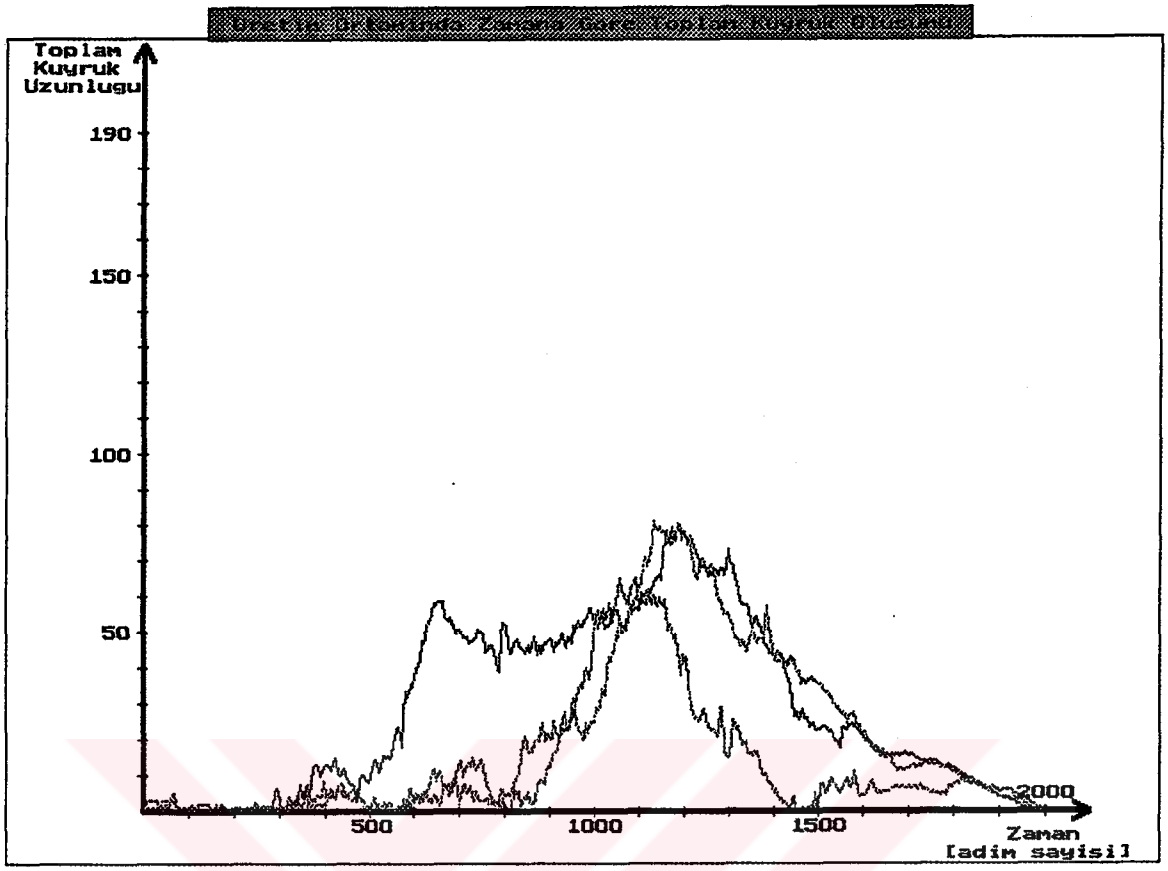
	1. Deneme	2. Deneme	3. Deneme
T.İ.S.	2199	2141	2000
T.S.İ.S.	151731	147729	138000
K.İ.	65065	47132	23964
K.O.H.	29.59	22.01	11.98



Şekil 4.8 : USDUS kullanılmadan yapılan 4-5-6. simülasyonlar sonucu oluşan boşta bekleyen grafikleri

Tablo 4.7
USDUS kullanılmadan yapılan 4-5-6. simülasyonlar sonucu elde edilen boşta bekleyen değerleri

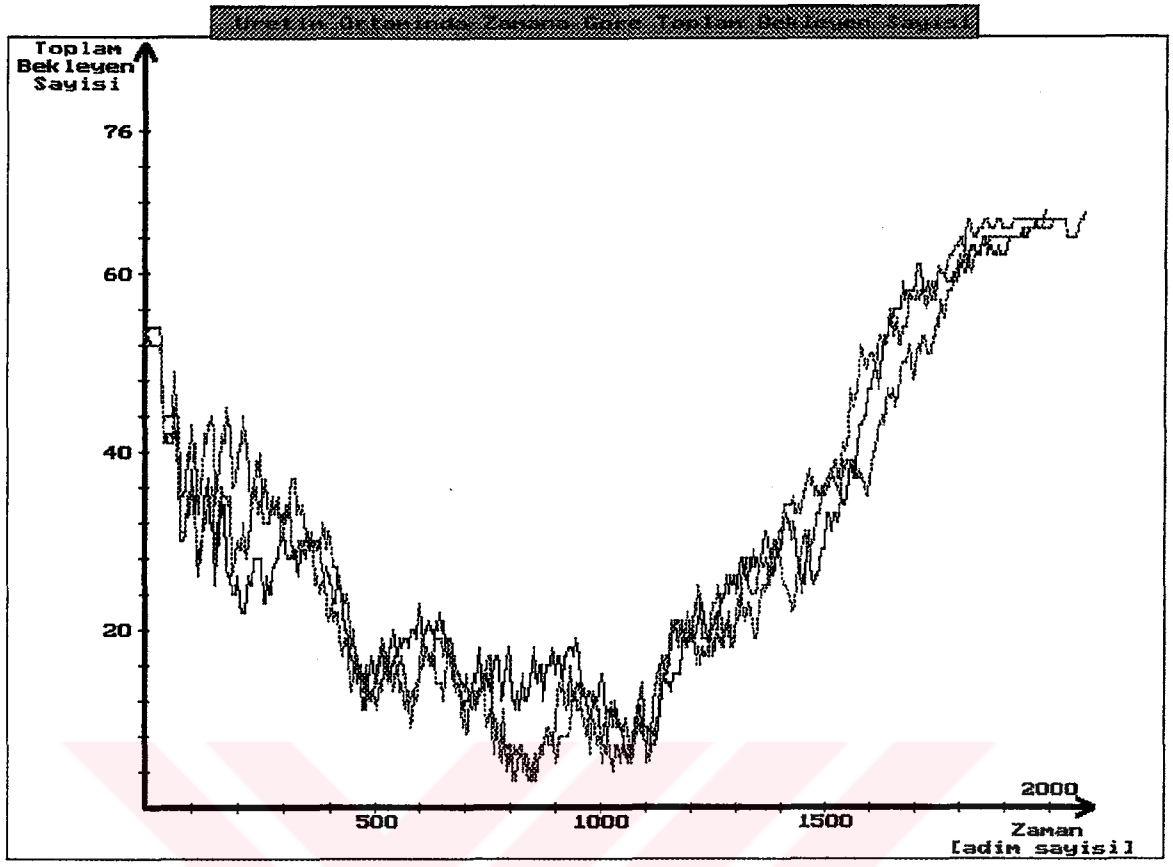
	1. Deneme	2. Deneme	3. Deneme
T.İ.S.	2199	2141	2000
T.S.İ.S.	151731	147729	138000
T.B.S.	75922	70337	60877
K.İ.G.	%50.04	%47.61	%44.11



Şekil 4.9: USDUS kullanılmadan yapılan 7-8-9. simülasyonlar sonucu oluşan kuyruk oluşumu grafikleri

Tablo 4.8
USDUS kullanılmadan yapılan 7-8-9. simülasyonlar sonucu elde edilen kuyruk oluşumu değerleri

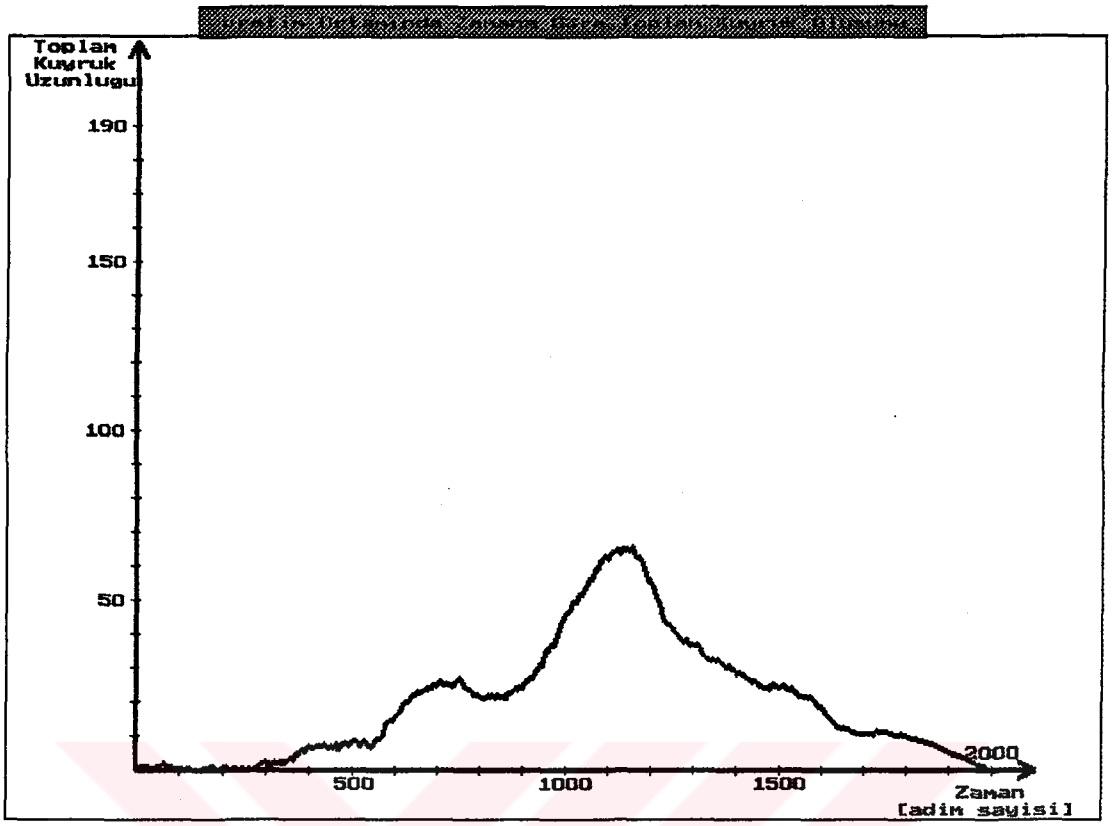
	1. Deneme	2. Deneme	3. Deneme
T.İ.S.	1992	2080	2008
T.S.İ.S.	137448	143520	138552
K.İ.	57668	24673	40008
K.O.H.	28.95	11.86	19.92



Şekil 4.10 : USDUS kullanılmadan yapılan 7-8-9. simülasyonlar sonucu oluşan boşta bekleyen grafikleri

Tablo 4.9
USDUS kullanılmadan yapılan 7-8-9. simülasyonlar sonucu elde edilen boşta bekleyen değerleri

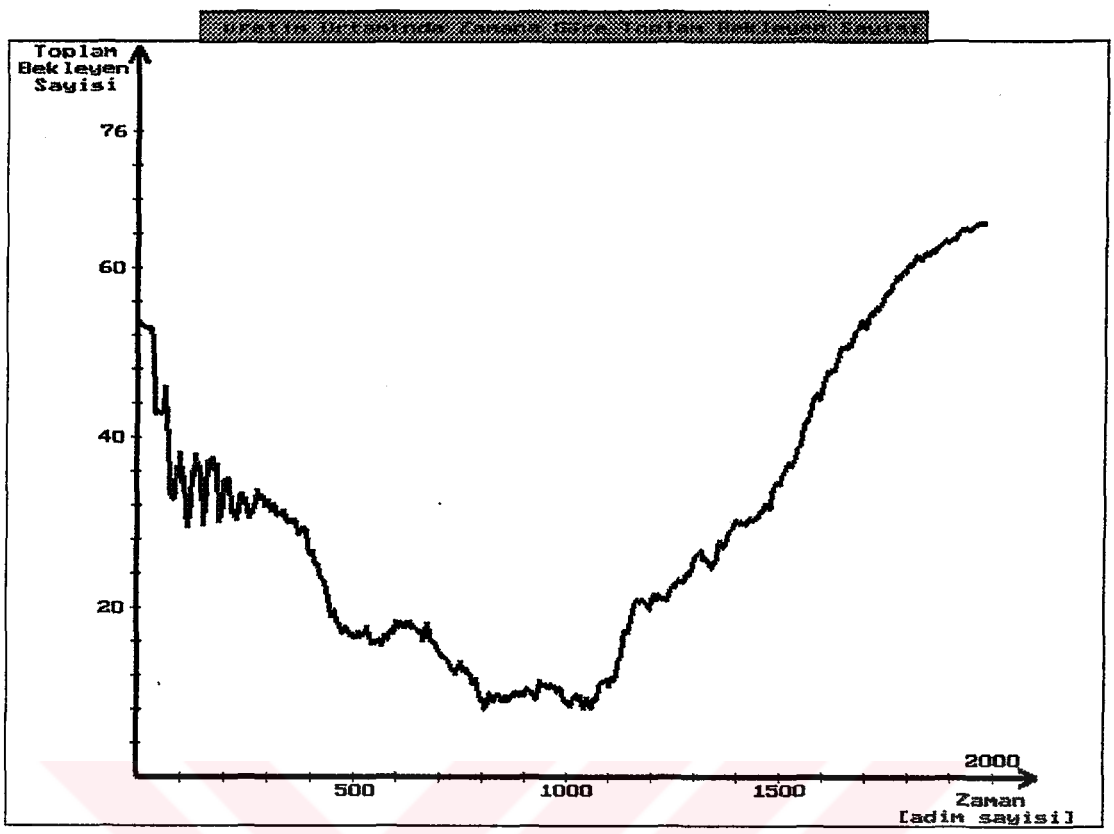
	1. Deneme	2. Deneme	3. Deneme
T.İ.S.	1992	2080	2008
T.S.İ.S.	137448	143520	138552
T.B.S.	60519	66156	60184
K.İ.G.	%44.03	%46.09	%43.44



Şekil 4.11 : USDUS kullanılmadan yapılan simülasyonların ortalaması sonucu oluşan toplam kuyruk oluşumu grafiği

Tablo 4.10
USDUS kullanılmadan yapılan simülasyonlar sonucu elde edilen ortalama kuyruk oluşumu değerleri

	Ortalama Değerler
T.İ.S.	2059
T.S.İ.S.	142094
K.İ.	40171
K.O.H.	19.43



Şekil 4.12 : USDUS kullanılmadan yapılan simülasyonların ortalaması sonucu oluşan toplam boşta bekleyen grafiği

Tablo 4.11
USDUS kullanılmadan yapılan simülasyonlar sonucu elde edilen ortalama boşta bekleyen değerleri

	Ortalama Değerler
T.İ.S.	2059
T.S.İ.S.	142094
T.B.S.	64910
K.İ.G.	%45.62

4.5 Uzman Sistem Yaklaşımlı Simülasyon

Bu alt bölümde; daha önce Bölüm 4.2.2.2.1'de belirtildiği gibi, Uzman Sistem Parametresine **Var** ifadesi yazılarak, uzman sistem yaklaşımlı simülasyon yapılmıştır. Her grafiğin altında ise Bölüm 4.3'de tanımlanmış bulunan parametrelerle ilgili sonuçlar verilmiştir.

Bölüm 4.2.2.2'de belirtilen tüm parametreler simülasyon programının USDUS'lu yaklaşım kısmında kullanıldığı için daha önceden belirlenmiştir. Yapılan deneylerde kuyruk eşiği olarak 5 seçilmiştir.

Deneyler birden fazla tekrarlanarak, stokastik olaylar sonucu olabilmesi muhtemel değişik durumlara karşı emin olunmak amaçlanmıştır.

Bir önceki bölümde verilen grafiklerle aynı yaklaşım takip edilmiştir. Buna göre USDUS yaklaşımlı 9 deney, üçlü gruplar halinde grafiklere aktarılmış, her bir deney için kuyruk oluşum ve boşta bekleyen grafikleri ayrı ayrı verilmiştir.

Grafiklerin gösterimi tamamlandıktan sonra tüm deneylerin değerlerinin bir ortalaması alınmak suretiyle bir özet grafik üretilmiştir.

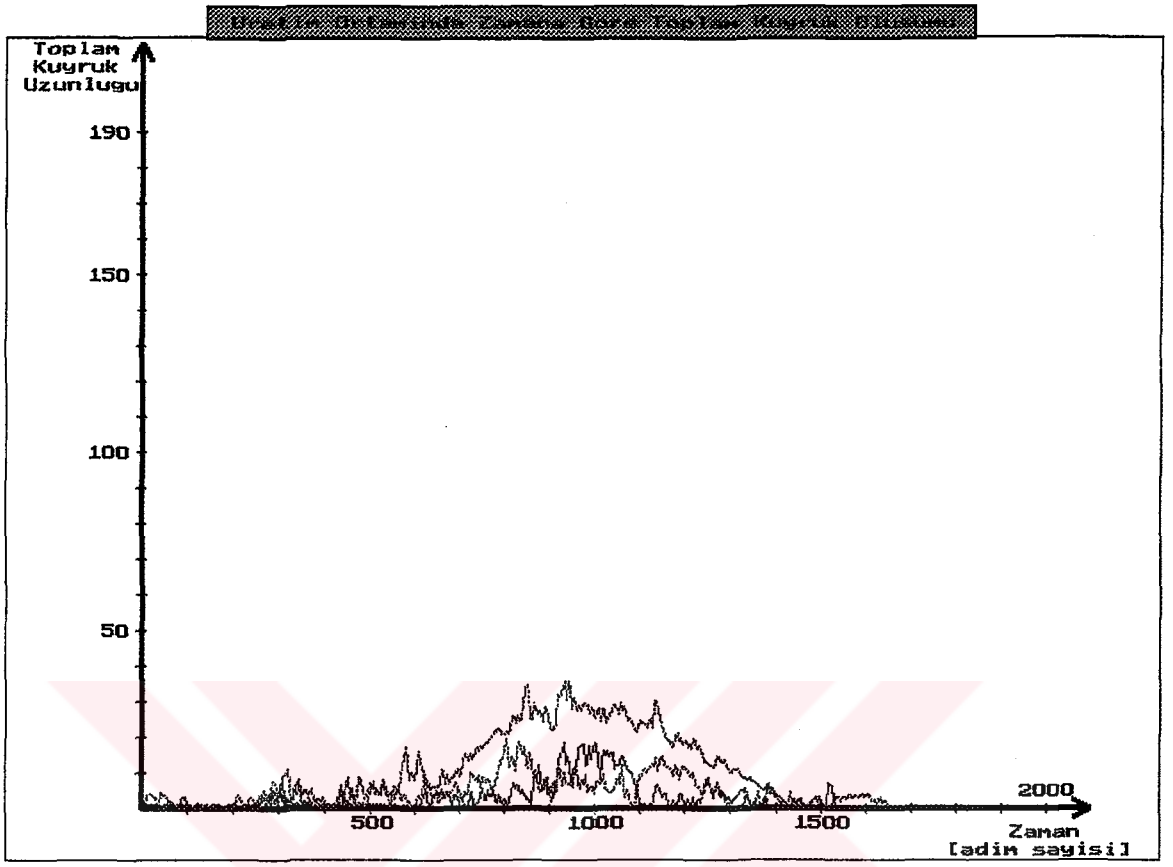
Simülasyon sisteminin stokastik olarak çalışması sonucunda hiçbir grafik birbirinin aynı değildir. Buna rağmen sistemin karakteristiklerini tüm grafiklerde izlemek olasıdır.

Şekil 4.13-15-17 ve Tablo 4.12-14-16 : USDUS kullanılarak yapılan simülasyonlar sonucu oluşan kuyruk oluşumu grafiklerini ve kuyruk oluşumu değerlerini,

Şekil 4.14-16-18 ve Tablo 4.13-15-17 : USDUS kullanılarak yapılan simülasyonlar sonucu oluşan boşta bekleyen grafiklerini ve boşta bekleyen değerlerini,

Şekil 4.19 ve Tablo 4.18 : USDUS kullanılarak yapılan simülasyonların ortalaması sonucu oluşan toplam kuyruk oluşumu grafiklerini ve kuyruk oluşumu değerlerini,

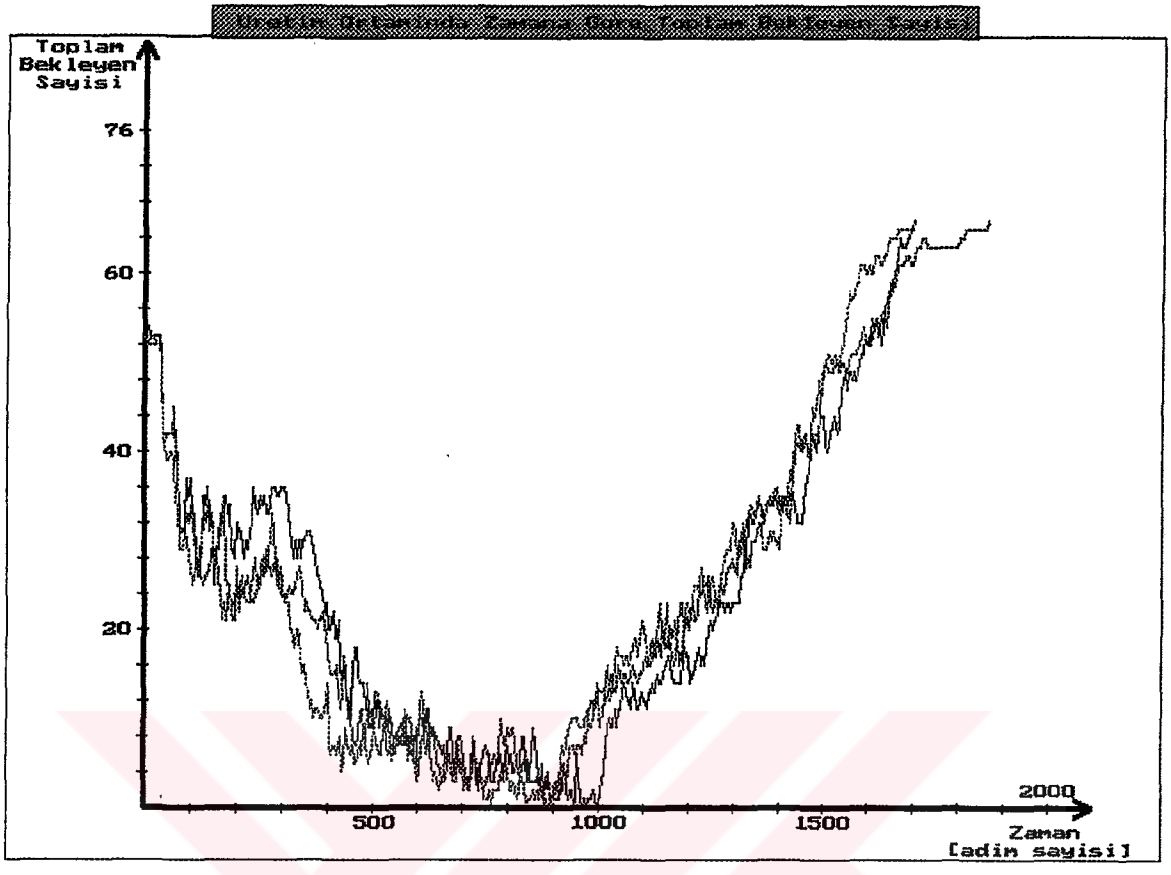
Şekil 4.20 ve Tablo 4.19 : USDUS kullanılarak yapılan simülasyonların ortalaması sonucu oluşan toplam boşta bekleyen grafiklerini ve boşta bekleyen değerlerini, göstermektedir.



Şekil 4.13: USDUS kullanılarak yapılan 1-2-3. simülasyonlar sonucu oluşan kuyruk oluşumu grafikleri

Tablo 4.12
USDUS kullanılarak yapılan 1-2-3. simülasyonlar sonucu elde edilen kuyruk oluşumu değerleri

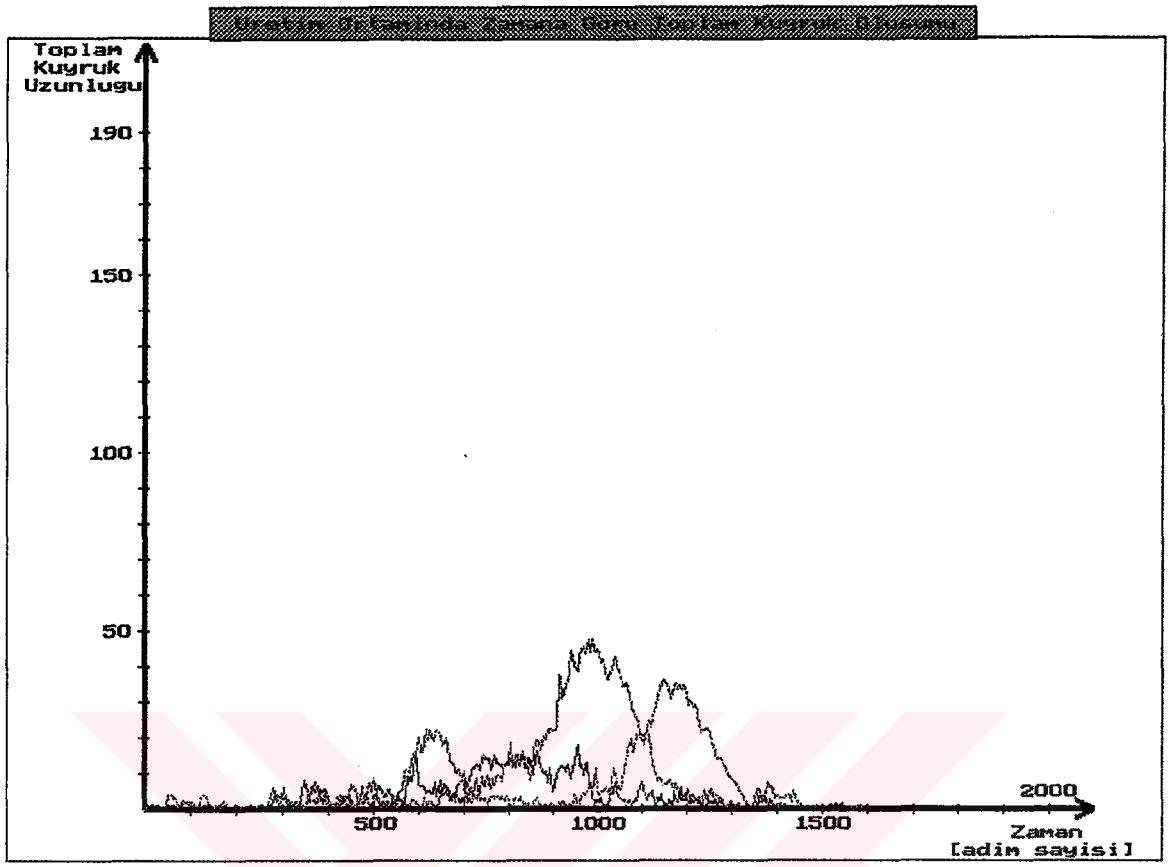
	1. Deneme	2. Deneme	3. Deneme
T.İ.S.	1704	1868	1690
T.S.İ.S.	117576	128892	116610
K.İ.	5282	17085	7322
K.O.H.	3.10	9.15	4.33



Şekil 4.14 : USDUS kullanılarak yapılan 1-2-3. simülasyonlar sonucu oluşan boşta bekleyen grafikleri

Tablo 4.13
USDUS kullanılarak yapılan 1-2-3. simülasyonlar sonucu elde edilen boşta bekleyen değerleri

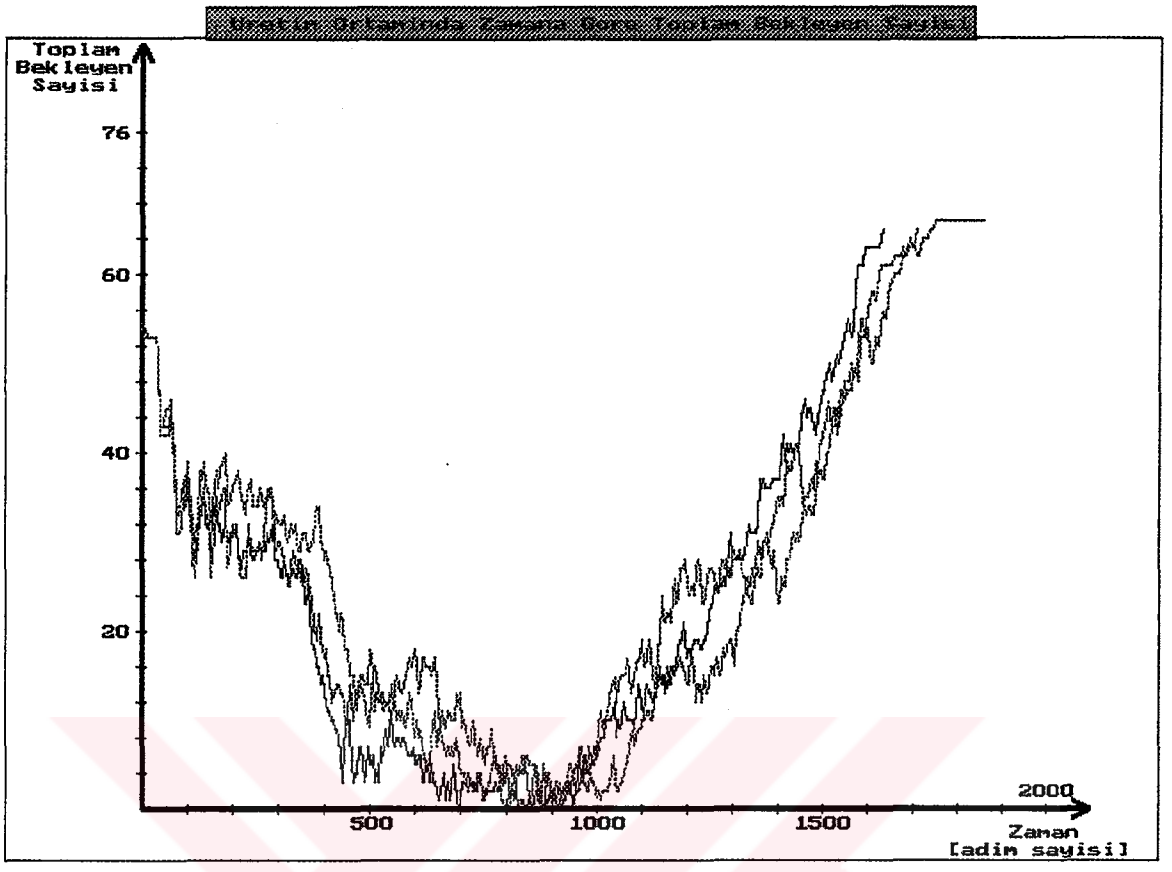
	1. Deneme	2. Deneme	3. Deneme
T.İ.S.	1704	1868	1690
T.S.İ.S.	117576	128892	116610
T.B.S.	40586	50944	39196
K.İ.G.	%34.52	%39.52	%33.61



Şekil 4.15: USDUS kullanılarak yapılan 4-5-6. simülasyonlar sonucu oluşan kuyruk oluşumu grafikleri

Tablo 4.14
USDUS kullanılarak yapılan 4-5-6. simülasyonlar sonucu elde edilen kuyruk oluşumu değerleri

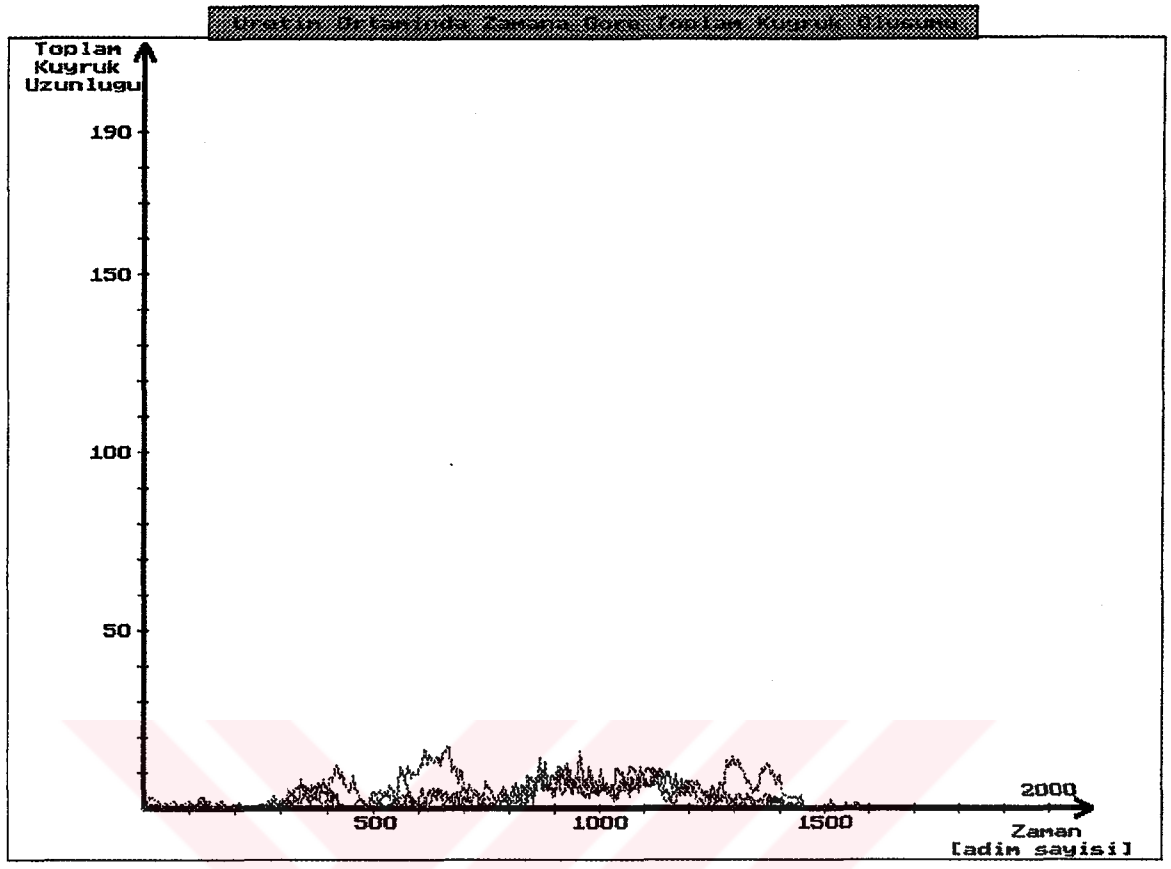
	1. Deneme	2. Deneme	3. Deneme
T.İ.S.	1637	1713	1863
T.S.İ.S.	112953	118197	128547
K.İ.	5853	14748	8451
K.O.H.	3.58	8.61	4.54



Şekil 4.16 : USDUS kullanılarak yapılan 4-5-6. simülasyonlar sonucu oluşan boşta bekleyen grafikleri

Tablo 4.15
USDUS kullanılarak yapılan 4-5-6. simülasyonlar sonucu elde edilen boşta bekleyen değerleri

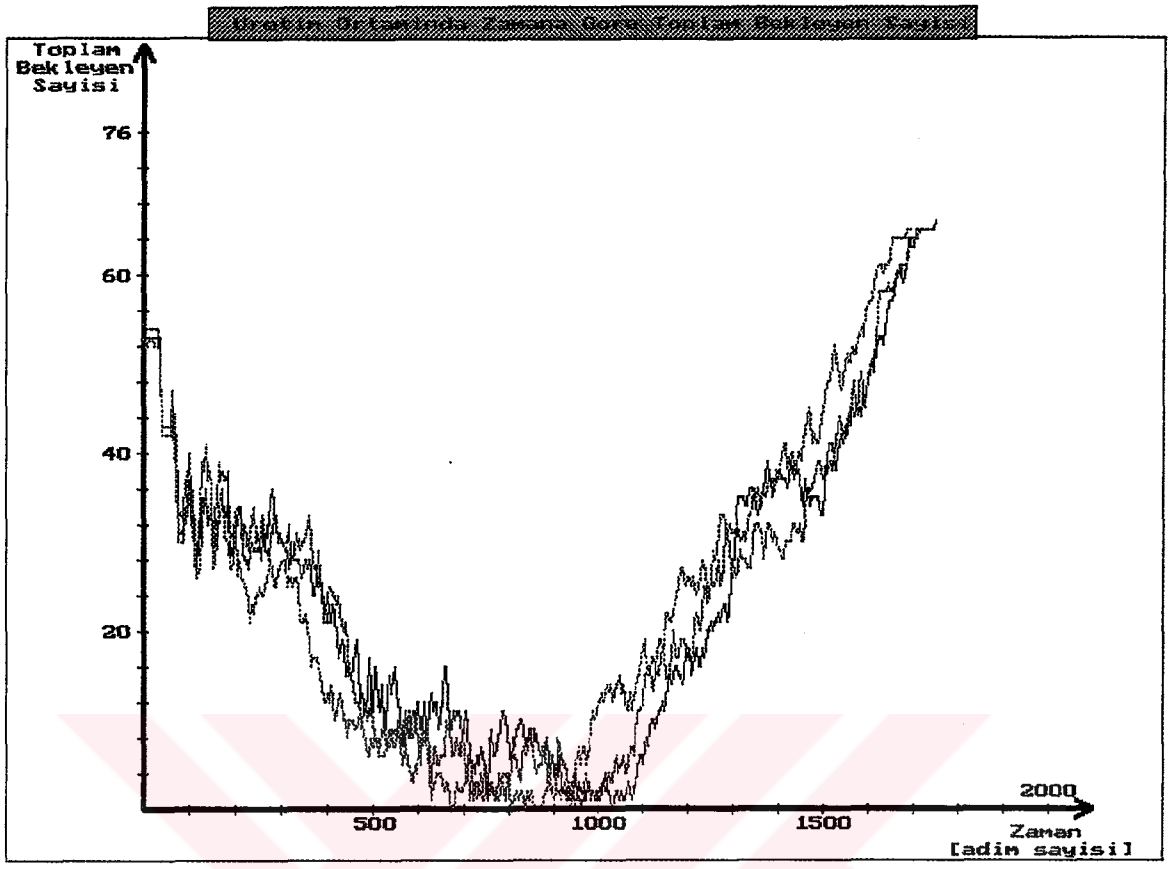
	1. Deneme	2. Deneme	3. Deneme
T.İ.S.	1637	1713	1863
T.S.İ.S.	112953	118197	128547
T.B.S.	35698	41191	52244
K.İ.G.	%31.60	%34.85	%40.64



Şekil 4.17: USDUS kullanılarak yapılan 7-8-9. simülasyonlar sonucu oluşan kuyruk oluşumu grafikleri

Tablo 4.16
USDUS kullanılarak yapılan 7-8-9. simülasyonlar sonucu elde edilen kuyruk oluşumu değerleri

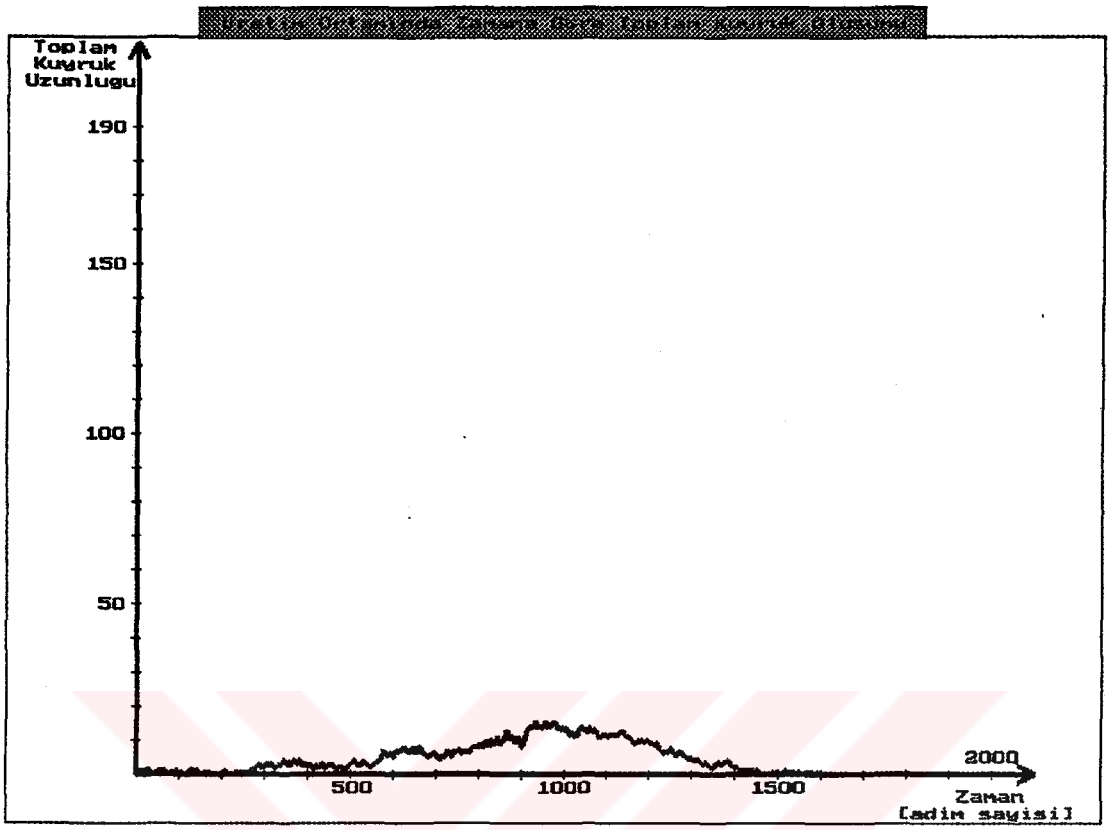
	1. Deneme	2. Deneme	3. Deneme
T.İ.S.	1752	1747	1699
T.S.İ.S.	120888	120543	117231
K.İ.	4510	6584	6035
K.O.H.	2.57	3.77	3.55



Şekil 4.18 : USDUS kullanılarak yapılan 7-8-9. simülasyonlar sonucu oluşan boşta bekleyen grafikleri

Tablo 4.17
USDUS kullanılarak yapılan 7-8-9. simülasyonlar sonucu elde edilen boşta bekleyen değerleri

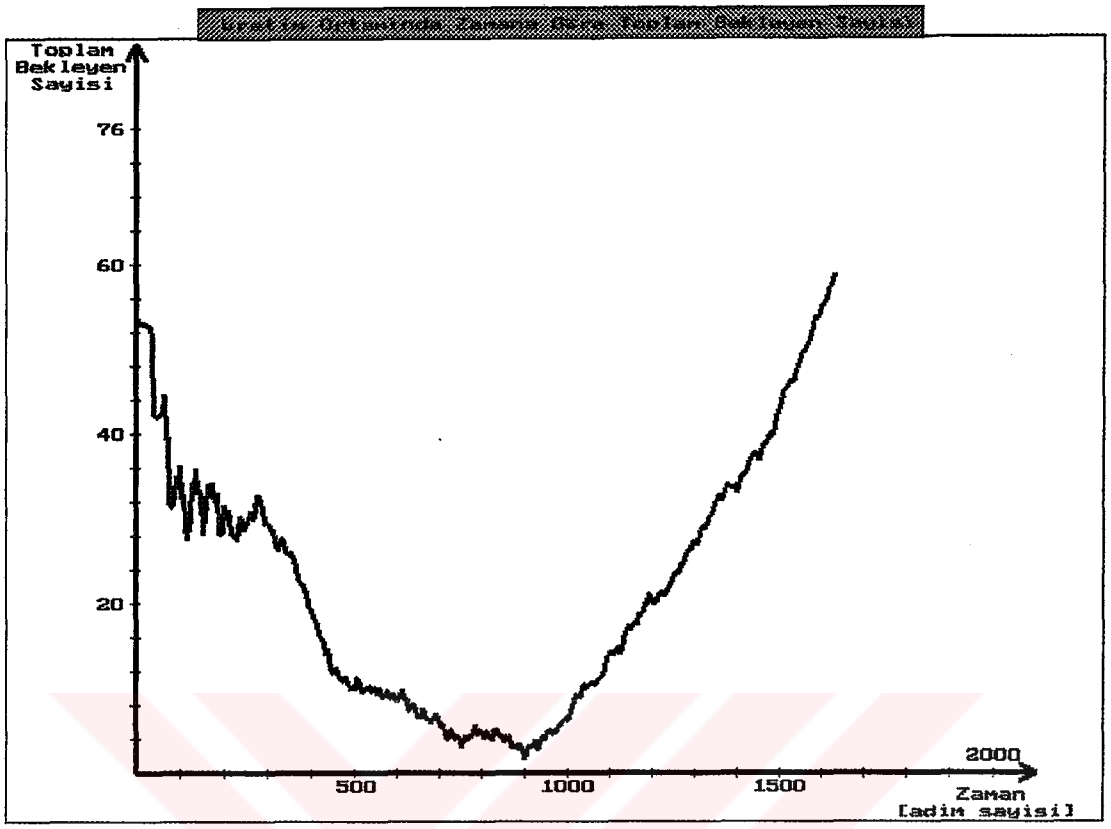
	1. Deneme	2. Deneme	3. Deneme
T.İ.S.	1752	1747	1699
T.S.İ.S.	120888	120543	117231
T.B.S.	43288	42958	40194
K.İ.G.	%35.81	%35.64	%34.29



Şekil 4.19 : USDUS kullanılarak yapılan simülasyonların ortalaması sonucu oluşan toplam kuyruk oluşumu grafiği

Tablo 4.18
USDUS kullanılarak yapılan simülasyonlar sonucu elde edilen ortalama kuyruk oluşumu değerleri

	Ortalama Değerler
T.İ.S.	1741
T.S.İ.S.	120160
K.İ.	8430
K.O.H.	4.80

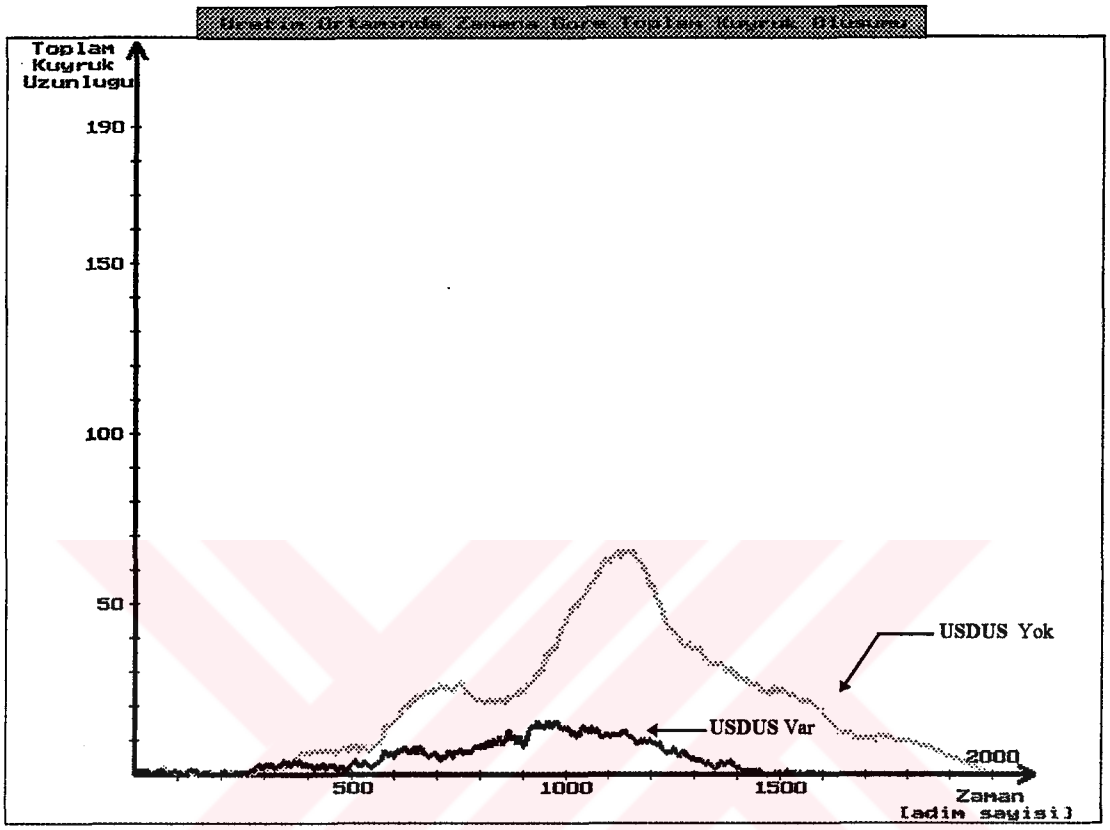


Şekil 4.20 : USDUS kullanılarak yapılan simülasyonların ortalaması sonucu oluşan toplam boşta bekleyen grafiği

Tablo 4.19
USDUS kullanılarak yapılan simülasyonlar sonucu elde edilen ortalama boşta bekleyen değerleri

	Ortalama Değerler
T.İ.S.	1741
T.S.İ.S.	120160
T.B.S.	42922
K.İ.G.	%35.61

4.6 Sonuçların Karşılaştırılması

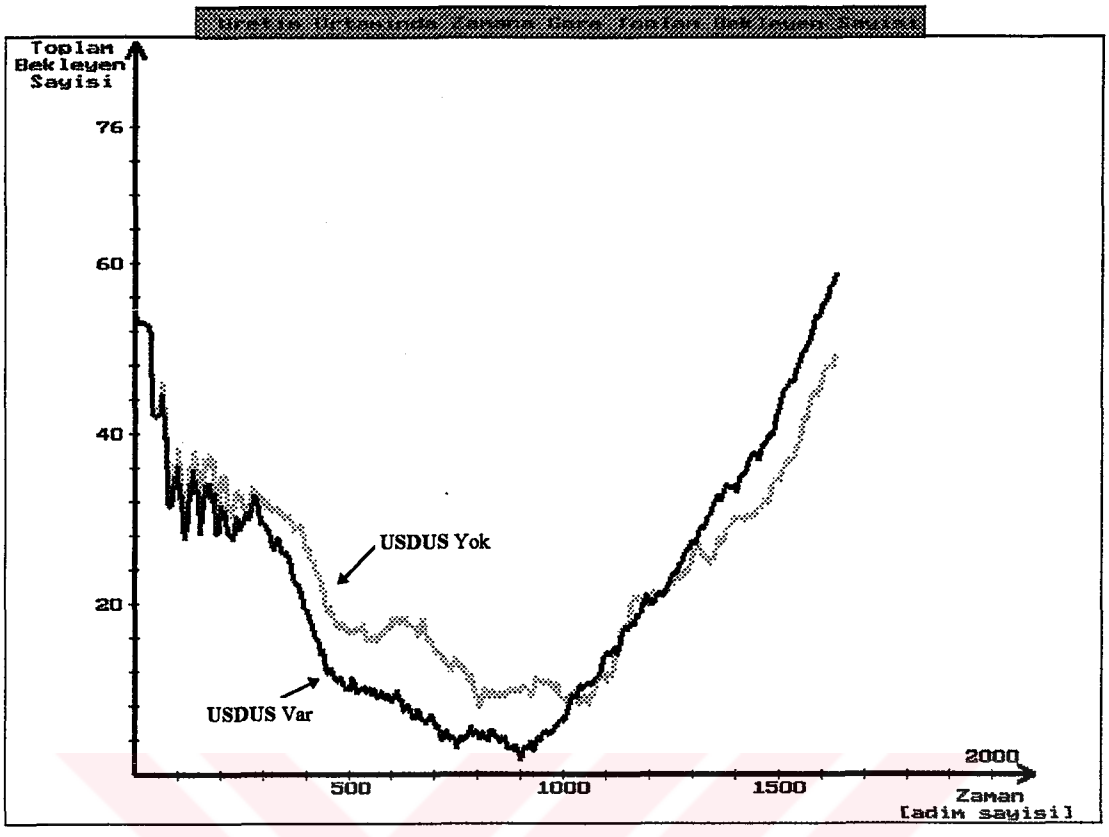


Şekil 4.21 : USDUS (var/yok) toplam kuyruk oluşumu karşılaştırma grafiği

Şekil 4.21'de; daha önce Şekil 4.11'de verilen USDUS kullanılmadan yapılan denemelerin toplam grafiği ile, Şekil 4.19'da verilen USDUS kullanılarak yapılan denemelerin toplam grafiği, karşılaştırılmak amacıyla üst üste gösterilmiştir.

Kuyruk oluşum grafikleri altındaki alan, simülasyon esnasında oluşan toplam kuyruklar hakkında bilgi vermektedir. Şekil 4.21'deki grafikte, USDUS kullanılan denemelerde toplam kuyruk sayısının (grafik altındaki alan) azaldığı gözlemlenebilir.

Tablo 4.10 ile Tablo 4.18 ise sözü edilen durumları sayısal olarak ifade etmektedir.



Şekil 4.22 : USDUS (var/yok) toplam boşta bekleyenler karşılaştırma grafiği

Şekil 4.22'de; daha önce Şekil 4.12'de verilen USDUS kullanılmadan yapılan denemelerin toplam grafiği ile, Şekil 4.20'de verilen USDUS kullanılarak yapılan denemelerin toplam grafiği, karşılaştırılmak amacıyla üst üste gösterilmiştir.

Grafikler incelendiğinde, ilk 100 adımda her iki grafiğin de yaklaşık olarak aynı gittiği izlenebilir. Bunun sebebi; simülasyon ilk başladığında, ürünler henüz yükleme aşamasında olup, çalışanların çoğunun boşta beklemesi ve kuyruklar oluşmadığından dolayı da, USDUS algoritmasının devreye girmemesinden kaynaklanmaktadır.

Grafiğe 100 ila 1250 adımları arasında bakıldığında, yaklaşık olarak her adım için, USDUS'lu grafiğin altındaki alanın diğerine göre daha az olduğu gözlenebilir. Bu istenen bir durumdur ve sayısal olarak Tablo 4.11 ile Tablo 4.19 da ifade edilmiştir.

1250. Adımdan sonra USDUS'lu grafikte, boşta bekleyenlerin daha çok olduğu izlenmektedir. Bu durumdan, USDUS yönetimi sonucu çalışanların boşta beklediği anlamını çıkarmak yanlış olur. Buradaki kötüleşmenin sebebi, USDUS'lu yaklaşımın simülasyonu daha erken bitirdiği için, üretim araçlarının başlarındaki çalışanların, boşta kalmasıdır. Üretim hiç durmadan devam etseydi, grafiğin bu durumuyla karşılaşılmayacaktı.

Sonuç olarak: Şekil 4.21 ve 4.22 değerlendirildiğinde; USDUS'lu yaklaşımın, daha iyi sonuç verdiği gözlenmektedir.

USDUS'un, üretim sistemine kazandırdığı üstünlükler, aşağıda gösterilmiştir.

1. Kuyruk sürelerini kısaltmaktadır,
2. Bekleyen sayısını azaltmaktadır,
3. Toplam üretim süresini azaltmaktadır,

SONUÇLAR

1. Kaynak planlanması ve iş planı, stokastik olarak en iyi hesaplanmış bir üretim ortamının, gerçek uygulamada planlandığı ve hesaplandığı biçimde gerçekleşmeyeceği varsayılmıştır.

2. Kaynak planlanması ve iş planı, statik olarak hesaplanmış bir üretim ortamında, zaman içinde iş bekleyen makinalar ve makina bekleyen işlerin (kuyrukların) oluşacağı kabul edilmiştir.

3. Beklemeler ve kuyruklar, üretim ortamının verimliliğini azaltacak ve üretim maliyetini artıracaktır.

4. Bu tez çalışmasında, kuyrukların kısaltılması ve beklemelerin azaltılması için, uzman sistem tekniğinin kullanılması ve bu kullanımı sağlamak için, gerekli olan donanım ve yazılım önerilmiştir.

5. Önerilen USDUS sisteminin uygulaması sonucunda, beklemelerin azaldığı ve kuyrukların kısaldığı gösterilmiştir.

6. Böylece, tezin başlangıcında ortaya konulan hedeflere ulaşılmıştır.

'Uzman sisteme dayalı üretim sistemi' kısaca USDUS' nin üretim ortamına katkıları küçümsenmeyecek ölçülerdedir. Beklemeler ve kuyrukların azaltılması, verimliliğin arttırılması ve üretim maliyetlerinin azaltılmasına önemli ölçüde katkı sağlamaktadır. Ancak, daha başka yöntemlerle daha da iyi çözümler üretilebilir. Yapılan çalışma bu konuyla ilgilenen kişilere fikir verme veya yol gösterme açısından yararlı olacaktır.

KAYNAKLAR

[1] KUSIAK, A. and CHEN, M., A Knowledge-Based System for Scheduling in Automated Manufacturing, pp. 192-196
'3rd International Conference on CAD/CAM, Robotics and Factories of the Future (CARS and FOF'88) Proceedings/Springer Verlag' vol.2

[2] YUNG, J.P. and WANG, H.P.(BEN), Automated Process Planning for Mechanical Assembly Operations, pp. 131-133
'3rd International Conference on CAD/CAM, Robotics and Factories of the Future (CARS and FOF'88) Proceedings/Springer Verlag' vol.2

[3] MEHDIAN, M. and SAGAR, R., A Prototype Flexible Manufacturing System, pp. 309-315,
'3rd International Conference on CAD/CAM, Robotics and Factories of the Future (CARS and FOF'88) Proceedings/Springer Verlag' vol.1

[4] O'CONNOR, D.E., Using Expert Systems to Manage Change and Complexity in Manufacturing, pp. 149-157,
[OCO84], In W. Reitman (ed.) Artificial İntelligence Applications for Business, Noorwood, N.J., Ablex, 1984 (XCON) (IMACS) (ISA)

[5] McDERMOTT, J., Building Expert Systems, pp. 11-22
[MCD84], In W. Reitman (ed.) Artificial İntelligence Applications for Business, Noorwood, N.J., Ablex, 1984 (XCON) (PTRANS) (XSEL)

[6] FOX, M.S. and SMITH, S.F., A Knowledge-Based System for Factory Scheduling, pp. 25-49
[FOX84b], Expert Systems, vol.1, no.1, 1984, (ISIS)

[7] TATSIOPOULOS, I.P. and PAPPAS, I.A., A Knowledge-Based Approach for Designing Industry-Specific Production Management Software, pp. 337-348
'Computer Integrated Manufacturing, Proceedings of the Sixth CIM-Europe Annual Conference, May 1990 Lisbon, Springen Verlag'

[8] BILLINGTON, A.J. and BOUCHER, A.R., On-Line Hierarchical Model Simulation for Process Fault Detection and Localisation, An Object Oriented Approach, pp. 310-320
'Computer Integrated Manufacturing, Proceedings of the Sixth CIM-Europe Annual Conference, May 1990 Lisbon, Springen Verlag'

[9] RIBEIRO, M.I., BISPO, C., SENTIEIRO, J., FERREIRA, C.P., CARVALHO, L., ALMEIDA, R., FERREIRA, J.N., 'A CIM Approach to Scheduling and Material Handling', pp. 140-151
'Computer Integrated Manufacturing, Proceedings of the Sixth CIM-Europe Annual Conference, May 1990 Lisbon, Springen Verlag'

[10] NYHUIS, F. and DUMKE, W.H., Computer-Aided Analysis of the Manufacturing Process of Job-Shop Production by a New Input-Output Method, pp. 287-296
'3rd International Conference on CAD/CAM, Robotics and Factories of the Future (CARS and FOF'88) Proceedings/Springer Verlag' vol.1

[11] LeVEAUX, J.M., FRAILE, M., MAZZOCCHI, G., Integrated Monitoring and Diagnostics in Modern Automated Manufacturing, pp. 173-182
'Computer Integrated Manufacturing, Proceedings of the Sixth CIM-Europe Annual Conference, May 1990 Lisbon, Springen Verlag'

[12] BERA, H., An Unorthodox Approach to Job-Scheduling, pp. 136-141
'3rd International Conference on CAD/CAM, Robotics and Factories of the Future (CARS and FOF'88) Proceedings/Springer Verlag' vol.2

[13] RONG, Y., Dynamic Approach of Computer Automatic Process Planning, pp. 430-433
'3rd International Conference on CAD/CAM, Robotics and Factories of the Future (CARS and FOF'88) Proceedings/Springer Verlag' vol.1

[14] ULFSBY, S., Dynamic Production Scheduling, pp. 153-169
'Computer Integrated Manufacturing, Proceedings of the Sixth CIM-Europe Annual Conference, May 1990 Lisbon, Springen Verlag'

[15] GROOVER, M.P., ZIMMERS, E.W., CAD/CAM: Computer-Aided Design and Manufacturing, Printice-Hall, 1984

[16] WALPOLE, R.E., Introduction to Statistics, Macmillan, 1974

[17] PAPOULIS, A., Probability, Random Variables, and Stochastic Processes, McGraw-Hill, 1965

[18] EVANS, J.R. and LINDSAY, W.M., The Management and Control of Quality, West, 1993

[19] JONHSON P.E., What Kind of Expert Should A System be ?, The Journal of Medicine and Philosophy, vol.8, pp. 77-97, 1983

[20] WATERMAN, D.A., A Guide to Expert Systems, Addison-Wesley, 1986

[21] CASHMORE, C., LYALL, W.R., Business Information, Printice-Hall, 1991

[22] SHANNON, R.E., Communication in the Presence of Noise, Proc. IRE, V 37, pp. 10-21, Jan 1949

[23] ADALI, E., Dağıtılmış Bilgisayarlarla Denetim, Doçentlik Tezi, İ.T.Ü.,Mart1980

[24] MUŞLU, B., Bilgisayar Destekli Üretim Ortamının Simülasyonu, Master Tezi İ.T.Ü., Haziran 1994

[25] YÜCEL, M.N., Monte-Karlo Metodu, İ.T.Ü., 1973

[26] MANO, M.M., Computer System Architecture, Prentice-Hall, 1982

[27] GROVER, M.P., Automation, Production Systems and Computer Integrated Manufacturing, Prentice-Hall, 1987

[28] FRENCH, S., Sequencing and Scheduling, Ellis Horwood Series 1990

[29] SAGE, A.P., System Design for Human Interaction, IEEE-Press, 1987

[30] CHARNIAK, E. and McDERMOTT, D., Introduction to Artificial Intelligence, Addison-Wesley, 1985

[31] BODEN, M.A., The Philosophy of Artificial Intelligence, Oxford, 1992

[32] KELLER, R., Expert System Technology, Yourdon-Press, 1987

[33] LIEBOWITZ, J., DeSALVO, D.A., Structuring Expert Systems, Yourdon-Press, 1989

[34] MURRAY, J.T. and MURRAY, M.J., Expert Systems in Data Processing, McGraw-Hill, 1988

[35] HILLIER, F.S. and LIEBERMAN, G. J., Introduction to Operations Research, McGraw-Hill, 1990

[36] TAHA, H.A., Operations Research, Macmillan, 1989

[37] BRONSON, R., Operations Research, Schaum's Outline Series, 1983

[38] MONKS, J.G., Operations Management, McGraw-Hill, 1987

[39] GAITHER, N., Production and Operations Management, Dryden-Press, 1992

[40] WEISS, H.J. and GERSHON, M.E., Production and Operations Management, Allyn and Bacon, 1989

[41] KARAYALÇIN, İ.İ., Harekat Araştırması, İ.T.Ü., 1979

EK A. UYGULAMA PROGRAMLARI

```
/*************************************************************************
                        Uzman.c : USDUS
                Doktora Tezi, Uzman Sisteme Dayalı Üretim Sistemi
                Fonksiyonlarının Bulunduğu Program Parçası
                        Y. Müh. H. Engin DEMİRAY
*************************************************************************/

int UsdusYaklasimi(int MkGrubu)
{
int OpNo, i, MkNo;
int Cikis, BaskaKuyrukOlusanMakinaVar;
int SimuleEt, TekKuyruk;

 if ( !USDUS_VAR )
   return(True);

 BaskaKuyrukOlusanMakinaVar = False;
 for ( i = 0 ; i < MAX_MAKINA; i++ ) {
  if ( i != MkGrubu ) {
    if ( KuyrukUzunlugu[i] >= TEHLIKELI_KUYRUK_SEVIYESI)
        BaskaKuyrukOlusanMakinaVar = True;
    }
  }

 Cikis = False;
 SimuleEt = False;
 TekKuyruk = KuyrukUzunlugu[MkGrubu];
 do {
  if ( BasindaAdamOlmayanMakinaVar(MkGrubu, &MkNo ) ) {
    if ( MakinasiTamirdeOlanOperatorVar( &OpNo ) ) {
        makina[ MkNo ]. soperator = OpNo;
        operator[OpNo]. imakina = MkNo;
        makina[ MkNo ]. mdurum = 2;
        SimuleEt = True;
        }
    else if ( BostaAdamVar( MkGrubu, &OpNo ) ) {
        makina [ operator[ OpNo ].imakina ]. soperator = -1;
        makina [ operator[ OpNo ].imakina ]. mdurum = 4;
        makina[ MkNo ]. soperator = OpNo;
        operator[OpNo]. imakina = MkNo;
        makina[ MkNo ]. mdurum = 2;
        SimuleEt = True;
        }
```

```
      else {
          // Bosta Adam Olmadigina Gore, Hic Kuyrugu Olmayan
          // Biryerden Adam Alinacak....
          DigerGruplardanUygunElemanBul( &OpNo, MkGrubu );
          makina [ operator[ OpNo ].imakina ]. soperator = -1;
          makina [ operator[ OpNo ].imakina ]. mdurum = 4;
          makina[ MkNo ]. soperator = OpNo;
          operator[OpNo]. imakina = MkNo;
          makina[ MkNo ]. mdurum = 2;
          SimuleEt = True;
          }
      }
      else {
      // Basinda Adam Olmayan Makina Yoksa Birsey Yapamayacak ....
      Cikis = True;
      SimuleEt = False;
      }
    if ( SimuleEt ) {
      HepsiniDosyayaYaz();
      SimulasyonIcindeSimulasyon();
      if ( KuyrukUzunlugu[MkGrubu] < TekKuyruk ) {
          if ( KuyrukUzunlugu[MkGrubu] < TEHLIKELI_KUYRUK_SEVIYESI ) {
            Cikis = True;
            // Basariya Ulasildi ..
            }
          else {
            Cikis = False;
            TekKuyruk = KuyrukUzunlugu[MkGrubu];
            // Basariya Ulasildi, ama yeterli degil ...
            }
          }
      else {
          // Kuyruk Iyilesmedi.. Biraz Daha Adam Kaydirmak Gerekli
          Cikis = False;
          TekKuyruk = KuyrukUzunlugu[MkGrubu];
          }
      HepsiniDosyadanOku();
      }

   } while( !Cikis);

 return(True);
}
```

```
/*****************************************************************
***** Fonksiyon Adı : MakinasıTamirdeOlanOperatorVar()      *********
***** Parametreler  : OperatorNo -> Değeri önemsiz.         *********
***** Dönen Değer   : Kendisi -> Makinası boşta olan opeatör *********
*****                 varsa True, yoksa False               *********
*****                 Operetor -> Şartlara uyan çalışan varsa *********
*****                 bu parametre anlamlı ve içeriğinde o çalı- *********
*****                 şanın numarası var.                   *********
*****************************************************************/


int MakinasiTamirdeOlanOperatorVar( int *Operator )
{
 int i;
 int AdamVar = False;

  *Operator = -1;
  for ( i = 0; i < MAX_OPERATOR; i++) {
    if ( operator[i].idurum == 2 ) {
      // Operator Bosta ....
      if ( makina [ operator[i].imakina ]. mdurum == 5 ) {
          if (InsanMakinaOraniOku(i, makina[operator[i].imakina].tip) > 0 ) {
            *Operator = i;
            AdamVar = True;
            }
          }
      }
    }

 if ( AdamVar ) {
   makina [ operator[ *Operator ].imakina ]. soperator = -1;
   }

return(AdamVar);
}
```

```
/*************************************************************
***** Fonksiyon Adı : BostaAdamVar()                  *********
***** Parametreler  : MakinaGrubu -> Hangi numaralı   *********
*****                 makina grubu için inceleme yapılacak.*****
*****                 Operator -> Başlangıçta önemsiz *********
***** Dönen Değer   : Kendisi -> Makinası boşta olan opeatör *****
*****                 varsa True, yoksa False         *********
*****                 Operetor -> Şartlara uyan çalışan varsa *****
*****                 bu parametre anlamlı ve içeriğinde o çalı- *****
*****                 şanın numarası var.             *********
***** Açıklama      : Çalışanları tarayarak boşta çalışan olup *****
*****                 olmadığını bulur.               *********
*************************************************************/


int BostaAdamVar( int MakinaGrubu, int *Operator )
{
 int i;
 int AdamVar = False;
 int EskiOran, YeniOran;

  *Operator = -1;
  EskiOran = 0;

  for ( i = 0; i < MAX_OPERATOR; i++) {
   if ( operator[i].idurum == 2 ) {
     // Operator Bosta ....
     if ( makina [ operator[i].imakina ]. mdurum == 2 ) {
          YeniOran = InsanMakinaOraniOku(i, makina[operator[i].imakina].tip);
          if ( YeniOran > EskiOran ) {
            if ( makina [ operator[i].imakina ]. tip != MakinaGrubu ) {
              // Tekrar simule ederken ayni degerleri bulasin diye ...
              *Operator = i;
              AdamVar = True;
              EskiOran = YeniOran;
              }
            }
          }
     else {
          hata(737);
          }
     }
   }
return(AdamVar);
}
```

```
/*************************************************************
***** Fonksiyon Adı : BasindaAdamOlmayanMakinaVar()          *********
***** Parametreler  : MkTip -> Hangi numaralı                *********
*****                 makina grubu için inceleme yapılacak.  *********
*****                 MkNo -> Başlangıçta önemsiz            *********
***** Dönen Değer   : Kendisi -> Başında adam olmayan makina *********
*****                 varsa True, yoksa False                *********
*****                 MkNo -> Başında adam olmayan makina    *********
*****                 varsa parametre anlamlı ve içeriğinde o ça- *********
*****                 lışanın numarası var.                  *********
***** Açıklama      : Uzman sistemin karar verebilmesi için önce-*********
*****                 likle gerekli bilgileri toplaması gerekir. *********
*****                 Bu fonksiyonla da başında kuyruk oluşan  *********
*****                 makina tipinin USDUS'a uygunluğu         *********
*****                 araştırılır.                             *********
*************************************************************/

int BasindaAdamOlmayanMakinaVar( int MkTip, int *MkNo )
{
int i, MkVar, Cikis;

  MkVar = False;
  Cikis = False;
  *MkNo = -1;
  i = 0;

  do {
    if ( i >= ( MAX_MAKINA * 10) ) {
      Cikis = True;
      }
    else if ( makina[i].tip == MkTip && makina[i].mdurum == 4 ) {
      Cikis = True;
      *MkNo = i;
      MkVar = True;
      }
    else {
      i++;
      }
    } while ( !Cikis );
return(MkVar);
}
```

```
/*************************************************************
***** Fonksiyon Adı : AdamsizMakina()                  *********
***** Parametreler  : MkNo -> Başlangıçta önemsiz.     *********
***** Dönen Değer   : Kendisi -> Adamsız makina varsa  *********
*****                 True, yoksa False                *********
*****                 MkNo -> Şartlara uyan makina varsa *********
*****                 bu parametre anlamlı ve içeriğinde o maki- *********
*****                 nanın numarası var.              *********
*************************************************************/

int AdamsizMakina( int *MkNo )
{
int i, MkVar, Cikis;

  MkVar = False;
  Cikis = False;
  *MkNo = -1;
  i = 0;

  do {
    if ( i >= ( MAX_MAKINA * 10) ) {
      Cikis = True;
      hata(813);
      }
    else if ( makina[i].soperator == -1 && makina[i].mdurum == 4) {
      Cikis = True;
      *MkNo = i;
      MkVar = True;
      }
    else {
      i++;
      }
    } while ( !Cikis );

return(MkVar);
}
```

```
/*************************************************************
***** Fonksiyon Adı : DiğerGruplardanUygunElemanBul()      *********
***** Parametreler  : GrupNo -> Hangi numaralı             *********
*****                 makina grubu için inceleme yapılacak.*********
*****                 Operator -> Başlangıçta önemsiz      *********
***** Dönen Değer   : Operetor -> Şartlara uyan çalışan varsa *******
*****                 bu parametre anlamlı ve içeriğinde o çalı- *****
*****                 şanın numarası var.                  *********
***** Açıklama      : Çalışanları tarayarak, farklı bir grupta *******
*****                 çalışan ve kuyruk olan gruba en uygun *********
*****                 çalışanı bulur.                      *********
*************************************************************/


int DigerGruplardanUygunElemanBul( int *Operator, int GrupNo )
{
 int i;
 int AdamVar = False;

  *Operator = -1;
  for ( i = 0; i < MAX_OPERATOR; i++) {
    if ( operator[i].idurum == 1 && makina[operator[i].makina].tip != GrupNo ) {
      if ( makina [ operator[i].imakina ]. mdurum == 5 ) {
              if (InsanMakinaOraniOku(i, makina[operator[i].imakina].tip) > 0 ) {
                *Operator = i;
                AdamVar = True;
                }
              }
      }
    }

 if ( AdamVar ) {
   makina [ operator[ *Operator ].imakina ]. soperator = -1;
   }

return(AdamVar);
}
```

```
/*****************************************************************
***** Fonksiyon Adı : SimulasyonIcindeSimulasyon                 *********
***** Dönen Değer   : Kendisi -> Yapılan İç simülasyon sonucu    *********
*****                 durumda iyileşme varsa True, yoksa False  *********
*****                 döner.                                    *********
***** Açıklama      : Simülasyon algritma gereği durumda iyileş- *********
*****                 me olup-olmadığını anlayabilmek için kendi*********
*****                 içerisinde tekrar bir simülasyon yapar.   *********
*****************************************************************/


int SimulasyonIcindeSimulasyon()
{

int CevrimI, sayac;
int Gec1, Bulundu, Is_Pr_Ptr;
int Cikis, MakinaVar, MkTip;
int PrNum, IsNo;

int PrNo, KIMkNum;
int IsNumarasi, IsProsesNumarasi, Oran, ilkno, MakinaTipi;
int i, CevrimSayisi;


 CevrimSayisi = SIS_ADIM_SAYISI;

 //########################## I S    C E V R I M I ######################
 for ( CevrimI = 0; CevrimI < CevrimSayisi; CevrimI++ ) {

  // Is Cevrimi .....
   for ( IsNo = 0; IsNo < MAX_IS*MAX_PROSES; IsNo++ ) {
    if ( is[IsNo].sdurum == 2 ) {
        BaskaProsesVar(IsNo, &Is_Pr_Ptr);
        Bulundu = -1;
        i = 0;
        do {
         if ( makina[i].tip == is_proses[Is_Pr_Ptr].makinat && makina[i].mdurum == 2 )
          Bulundu++;
         if ( makina[i].tip == is_proses[Is_Pr_Ptr].makinat && makina[i].mdurum == 3
&&
          makina[i].yedeksepet == 32100 )
          Bulundu++;
         i++;
        } while( ( i < MAX_MAKINA*10 ) );
```

```
if ( Bulundu <= 0 ) {    // Bosta Makina Yoksa Geri Don .....
 }
else {  // elseNo : 3
 Cikis = False;
 i = 0;
 do {
  if ( makina_kuyruk[i].MakinaNum == is_proses[Is_Pr_Ptr].makinat ) {
   Cikis = True;
   MakinaVar = True;
   }

  if ( i >= MAX_PROGRAM_KUYRUK )
   Cikis = True;
  else
   i++;
  } while ( !Cikis );

 if ( !MakinaVar ) {
   // ########## Prosesi Calistir Basladi ###################
   Bulundu = False;

   i = 0;

   do {
       if ( proses[i].proses == 0 )
         Bulundu = True;
       else
         i++;
       }  while( ( i < MAX_PROSES*10 ) && !Bulundu );

   if ( i == MAX_PROSES*10 )
       hata(1003);

   PrNum = i;

   is[IsNo].prprptr = PrNum;
   is[IsNo].isprptr = Is_Pr_Ptr;

   proses[PrNum].proses = UniqueProsesNumber;
   UniqueProsesNumber++;
   proses[PrNum].tip = is_proses[Is_Pr_Ptr].tip;
   proses[PrNum].isofpr = IsNo;
   proses[PrNum].makinat = is_proses[Is_Pr_Ptr].makinat;
   is_proses[Is_Pr_Ptr].sure = is_proses[Is_Pr_Ptr].sureth +
                               is_proses[Is_Pr_Ptr].sureto;
   proses[PrNum].scalisma = is_proses[Is_Pr_Ptr].sure;
```

```
            proses[PrNum].szaman = proses[PrNum].skuyruk = 0.0;

            proses[PrNum].pdurum = 2; /* Proses Kuyrukta. */

            // ######### Makina Kuyruguna Ekle Basladi ##################
            MkTip = is_proses[Is_Pr_Ptr].makinat;
            KuyrukUzunlugu[MkTip]++;
                // ######### BosKuyrukNo Basladi ##################
                i = 0;
                Cikis = False;
                 do {
                   if ( makina_kuyruk[i].MakinaNum == 0 )
                     Cikis = True;
                   else if ( i >= MAX_PROGRAM_KUYRUK )
                     Cikis = True;
                   else
                     i++;
                   } while (Cikis == False);

                if (i >= MAX_PROGRAM_KUYRUK )
                  hata(1102);
                // ######### BosKuyrukNo Bitti ##################

            makina_kuyruk[i].MakinaNum = MkTip;
            makina_kuyruk[i].MProses = PrNum;
            // ######### Makina Kuyruguna Ekle Bitti ##################


            // ######### Prosesi Calistir Bitti ##################

           is[IsNo].sdurum = 1;  /* Calisiyor Yap */
           } // ProsesiCalistir / Makina Var
         } // elseNo : 3
    } // Isdurum == 2..

else {
 /* Calisan Bir Prosese Yapilacak Islemler */
 Is_Pr_Ptr = is[IsNo].isprptr;
 i = is[ IsNo ].prprptr;
    /* Bu Isin Su Anda Calisan Prosesinin
          Numarasi */
 if ( proses[i].pdurum != 4 ) {
   }
 else {
   /* Proses Bitmis. */
   proses[i].proses = 0;
```

```
if (BaskaProsesVar(IsNo, &Is_Pr_Ptr) ) {

        // ######### Prosesi Calistir Basladi ##################
        Bulundu = False;

        i = 0;

        do {
            if ( proses[i].proses == 0 )
              Bulundu = True;
            else
              i++;
            }  while( ( i < MAX_PROSES*10 ) && !Bulundu );

        if ( i == MAX_PROSES*10 )
            hata(1003);

        PrNum = i;

        is[IsNo].prprptr = PrNum;
        is[IsNo].isprptr = Is_Pr_Ptr;

        proses[PrNum].proses = UniqueProsesNumber;
        UniqueProsesNumber++;
        proses[PrNum].tip = is_proses[Is_Pr_Ptr].tip;
        proses[PrNum].isofpr = IsNo;
        proses[PrNum].makinat = is_proses[Is_Pr_Ptr].makinat;
        proses[PrNum].scalisma = is_proses[Is_Pr_Ptr].sureth +
                                 is_proses[Is_Pr_Ptr].sureto;
        proses[PrNum].szaman = proses[PrNum].skuyruk = 0.0;
        proses[PrNum].pdurum = 2; /* Proses Kuyrukta. */

        // ######### Makina Kuyruguna Ekle Basladi ##################
        MkTip = is_proses[Is_Pr_Ptr].makinat;
        KuyrukUzunlugu[MkTip]++;
            // ######### BosKuyrukNo Basladi ##################
            i = 0;
            Cikis = False;
             do {
              if ( makina_kuyruk[i].MakinaNum == 0 )
                Cikis = True;
              else if ( i >= MAX_PROGRAM_KUYRUK )
                Cikis = True;
              else
                i++;
              } while (Cikis == False);
```

```
                if (i >= MAX_PROGRAM_KUYRUK )
                  hata(1102);
                // ######### BosKuyrukNo Bitti ##################

              makina_kuyruk[i].MakinaNum = MkTip;
              makina_kuyruk[i].MProses = PrNum;
              // ######### Makina Kuyruguna Ekle Bitti ##################


              // ######### Prosesi Calistir Bitti ##################
             } // Baska Proses Var ..
     else {
        /* Bu Is Tamamen Bitmis Durumda */
        is[IsNo].sdurum = 0; /* Is Bitti */
        is[IsNo].isprptr = 0;
        is[IsNo].prprptr = 0;
        }
    } // Proses Bitmemis ..
  } // Is Durum != 2
 } // for Iscevrimi
//########################## P R   C E V R I M I ########################
for ( PrNo = 0; PrNo < MAX_PROSES*10; PrNo++ ) {

 if ( PrNo == MAX_PROSES*10 ) {
   }
 else if ( proses[PrNo].proses == 0 ) {
  }
 else if ( proses[PrNo].pdurum == 4 ) {
  }
 else if ( proses[PrNo].pdurum == 1 ) {
  }
 else if ( proses[PrNo].pdurum == 2 ) {
// ############### Kuyruk Islemi Baslangici ##################
  // ############ Bos Makina Var Baslangici ##############
    Bulundu = False;
    i = 0;
    do {
        if ( makina[i].tip == proses[PrNo].makinat &&
            makina[i].mdurum == 2 &&
            makina[i].yedeksepet == 32100 &&
            operator[ makina[i].soperator ].idurum == 2 )
                Bulundu = True;
        else
         i++;
        }  while( ( i < MAX_MAKINA*10 ) && !Bulundu );
```

```
 // ############ Bos Makina Var Bitisi ##############

if ( Bulundu ) {
 /* Bosta Makina Varsa Prosesi Calistir. */
 KIMkNum = i;
 proses[PrNo].makinak =KIMkNum;
 proses[PrNo].pdurum = 3; /* Proses Calistirildi. */
 IsNumarasi = proses[PrNo].isofpr;
 IsProsesNumarasi = is[IsNumarasi].isprptr;
 if ( proses[PrNo].scalisma == 0 ) {
  Oran = MakinaAtananInsanOku( proses[PrNo].makinat,
                               makina[KIMkNum].kod );
  proses[PrNo].scalisma = is_proses[IsProsesNumarasi].sureth +
                          is_proses[IsProsesNumarasi].sureto*Oran/100;
  }

 makina[KIMkNum].mdurum = 3; /* Makina Calistirildi. */
 makina[KIMkNum].proses = PrNo;
 operator[makina[KIMkNum].soperator].idurum = 1;

 //############ MakinaKuyrugundanSil Baslangic ################

 MkTip = proses[PrNo].makinat;

if ( KuyrukUzunlugu[MkTip] == 0 ) {
 hata(1012);    // Kuyruk Zaten Sifir.
 }
else {
 KuyrukUzunlugu[MkTip]--;

 ilkno = 32000;
 Cikis = False;
 i = 0;
 do {
  if ( i >= MAX_PROGRAM_KUYRUK )
      Cikis = True;
  if ( makina_kuyruk[i].MakinaNum == MkTip &&
       makina_kuyruk[i].MProses == PrNo ) {
      Cikis = True;
      ilkno = i;
      }
  else {
      i++;
      }
  } while ( !Cikis );
```

```
    if ( ilkno == 32000 )
        hata(1124);

 if ( ilkno > MAX_PROGRAM_KUYRUK ) {
  hata(1734);
  }

 makina_kuyruk[ilkno].MakinaNum = 0;
 makina_kuyruk[ilkno].MProses = 0;

  } // Else KuyrukUzunlugu...

  //############ MakinaKuyrugundanSil Bitti ##################
  } // if Bosta Makina Var ...
 else {

Bulundu = False;
MakinaTipi = proses[PrNo].makinat;
i = 0;
do {
  if (  makina[i].tip == MakinaTipi &&
    ( ( makina[i].mdurum == 2 && operator[makina[i].soperator].idurum==2 ) ||
    ( makina[i].mdurum == 3 && makina[i].yedeksepet == 32100 ) ) )
    Bulundu = True;
  else
    i++;
  } while( ( i < MAX_MAKINA*10 ) && !Bulundu );

KIMkNum = i;
if ( Bulundu ) {
  /* Bosta Makinanin Yedek Sepeti Varsa Prosesi Calistir. */
  proses[PrNo].makinak =KIMkNum;
  proses[PrNo].pdurum = 5; /* Proses Yedekte Calistirildi. */
  IsNumarasi = proses[PrNo].isofpr;
  IsProsesNumarasi = is[IsNumarasi].isprptr;
  if ( proses[PrNo].scalisma == 0 ) {
    /* Proses Daha Onceden Calismis. Tekrar Sure
        Hesaplanirsa, hesaplanan bu surenin bir
        once hesaplanadan daha az olma ihtimali
        bulunuyor.. */
    Oran = MakinaAtananInsanOku( proses[PrNo].makinat,
                                 makina[KIMkNum].kod );
    proses[PrNo].scalisma = is_proses[IsProsesNumarasi].sureth +
                            is_proses[IsProsesNumarasi].sureto*Oran/100;
    }
```

```
 makina[KIMkNum].yedeksepet = PrNo;

 //############ MakinaKuyrugundanSil Baslangic #################

 MkTip = proses[PrNo].makinat;

if ( KuyrukUzunlugu[MkTip] == 0 ) {
 hata(1012);    // Kuyruk Zaten Sifir.
 }
else {
 KuyrukUzunlugu[MkTip]--;

 ilkno = 32000;
 Cikis = False;
 i = 0;
 do {
  if ( i >= MAX_PROGRAM_KUYRUK )
      Cikis = True;
  if ( makina_kuyruk[i].MakinaNum == MkTip &&
       makina_kuyruk[i].MProses == PrNo ) {
      Cikis = True;
      ilkno = i;
      }
  else {
      i++;
      }
  } while ( !Cikis );

  if ( ilkno == 32000 )
      hata(1124);

if ( ilkno > MAX_PROGRAM_KUYRUK ) {
 hata(1734);
 }

makina_kuyruk[ilkno].MakinaNum = 0;
makina_kuyruk[ilkno].MProses = 0;

 } // Else KuyrukUzunlugu...
 //############ MakinaKuyrugundanSil Bitti #################
  } //  if Bulundu
 } // else
}  // else if pdurum == 2
```

```
else if ( proses[PrNo].pdurum == 3 ) {
  proses[PrNo].szaman++;
  if (proses[PrNo].scalisma == proses[PrNo].szaman) {
    /* Proses Bitti. Ama Hazir Sepetine Alinmali.. */

    proses[PrNo].pdurum = 6;  /* Proses Biten Sepette */

    /* Makinayi Bosalt */
    makina[proses[PrNo].makinak].mdurum = 2;
    makina[proses[PrNo].makinak].proses = 0;
    makina[proses[PrNo].makinak].bitensepet = PrNo;
    operator[makina[ proses[PrNo].makinak ].soperator ].idurum = 2;

    }
  }
else if ( proses[PrNo].pdurum == 5 ) {

  if ( makina [ proses[PrNo].makinak ] .mdurum == 2 ) {
    // Yedekte Bekledigi Makina Bosalmis..
    makina [ proses[PrNo].makinak ] .mdurum = 3;
    makina [ proses[PrNo].makinak ] .proses = PrNo;
    makina [ proses[PrNo].makinak ] .yedeksepet = 32100;
    operator [ makina[ proses[PrNo].makinak ].soperator ].idurum = 1;
    proses[PrNo].pdurum = 3;
    }
  if ( makina [ proses[PrNo].makinak ] .mdurum == 5 ||
        makina [ proses[PrNo].makinak ] .mdurum == 4 ) {
    // Yedekte Bekledigi Makina Bozuk ya da basinda adam yok ..
    // Aslinda boyle bir durum olmamasi gerekiyor.
    proses[PrNo]. pdurum = 2;
    makina[ proses[PrNo].makinak ] .yedeksepet = 32100;

    // ############### Makina Kuyruguna Ekle Baslangici ##################
    MkTip = makina [proses[PrNo].makinak].tip;
    if ( KuyrukUzunlugu[MkTip] > MAX_EKRAN_KUYRUK )
        hata(1011);    // Kuyruk Ekrana Sigmiyor.
    else {
        KuyrukUzunlugu[MkTip]++;
```

```
                // ######### BosKuyrukNo Basladi ###################
                Cikis = False;
                i = 0;
                 do {
                   if ( makina_kuyruk[i].MakinaNum == 0 )
                     Cikis = True;
                   else if ( i >= MAX_PROGRAM_KUYRUK )
                     Cikis = True;
                   else
                     i++;
                   } while (Cikis == False);

                if (i >= MAX_PROGRAM_KUYRUK )
                  hata(1102);
                // ######### BosKuyrukNo Bitti ###################


          makina_kuyruk[i].MakinaNum = MkTip;
          makina_kuyruk[i].MProses = PrNo;
          } // Else
      } // if mdurum == 4 || mdurum == 5
    } // else if pdurum = 5

  else if ( proses[PrNo].pdurum == 6 ) {
    proses[PrNo].proses = 0;
    proses[PrNo].pdurum = 4;  /* Proses Bitti */

    /* Makinayi Bosalt */
    makina[proses[PrNo].makinak].bitensepet = 0;
    }

 } // for pr_cevrimi


 } // For Cevrim
return(True);
}  // Simulasyon Simulasyonu Fonksiyonu ...
```

```
/*************************************************************************
```

Gr_Uzm_K.c
Doktora Tezi, Kuyrukların Grafik Olarak
Gösterimlerinin Yapıldığı Program Parçası
Y. Müh. H. Engin DEMİRAY

NOT : Bu program kullanılarak elde edilen şekillerin listesi :

- Şekil 4.5
- Şekil 4.7
- Şekil 4.9
- Şekil 4.13
- Şekil 4.15
- Şekil 4.17

```
*************************************************************************/

#include <stdio.h>
#include <graphics.h>
#include "tanim.h"
#include "tool.h"


#define True 1
#define False 0


int  NoktaUzunlugu;
long int KuyrukIntegrali;
FILE *DosyaX;
char UVarYok[2], ChGrupNo[2];
```

```
/*******************************************************
**** Fonksiyon Adı   : main()                    **********
**** Açıklama        : Programın ana kısmı       **********
******************************************************* */


main()
{
int i, j, k, k1, k2, k3, KayitSayisi, MaxKayitSayisi;
int Renk, EskiX, EskiY;
char YazilacakRenk[15];
int  GrupNo;


 if ( EkranOku() == -100 )
   return(0);

 yaz(1,13,"",9);

 GrupNo = atoi( ChGrupNo );

 KayitSayisi = MaxKayitSayisi = 0;

   i = (GrupNo - 1 ) * 3 + 1;
   KuyrukIntegrali = 0;
   KayitSayisi = DosyaVar(i, True);
   if ( KayitSayisi > 0 ) {
     RenkBul(i, YazilacakRenk);
     printf("File Sira No : %2d, Kayit Sayisi :  %4d, KI : %5ld, Renk : %s \n",
                              i, KayitSayisi, KuyrukIntegrali, YazilacakRenk );
     if ( KayitSayisi > MaxKayitSayisi)
            MaxKayitSayisi = KayitSayisi;
     }

   i++;
   KuyrukIntegrali = 0;
   KayitSayisi = DosyaVar(i, True);
   if ( KayitSayisi > 0 ) {
     RenkBul(i, YazilacakRenk);
     printf("File Sira No : %2d, Kayit Sayisi :  %4d, KI : %5ld, Renk : %s \n",
                              i, KayitSayisi, KuyrukIntegrali, YazilacakRenk );
     if ( KayitSayisi > MaxKayitSayisi)
            MaxKayitSayisi = KayitSayisi;
     }
```

```
  i++;
  KuyrukIntegrali = 0;
  KayitSayisi = DosyaVar(i, True);
  if ( KayitSayisi > 0 ) {
    RenkBul(i, YazilacakRenk);
    printf("File Sira No : %2d, Kayit Sayisi :  %4d, KI : %5ld, Renk : %s \n",
                              i, KayitSayisi, KuyrukIntegrali, YazilacakRenk );
    if ( KayitSayisi > MaxKayitSayisi)
          MaxKayitSayisi = KayitSayisi;
    }

printf("SONUC : %4d\n", MaxKayitSayisi);
NoktaUzunlugu = 1;
if ( MaxKayitSayisi > 500 ) {
  NoktaUzunlugu = MaxKayitSayisi/500 + 1;
  }

Renk = 1;
EskiX = 25;
EskiY = 400;

i = getch();

GrafikAc();
for ( i = (GrupNo - 1 ) * 3 + 1; i < (GrupNo - 1 ) * 3 + 4; i++ ) {
  KayitSayisi = DosyaVar(i, True);
  Renk = i;
  if ( KayitSayisi > 0 ) {
    setcolor(Renk);
    EskiX = 75;
    EskiY = 450;
    k = k1 = k2 = k3 = 0;
    DosyaVar(i, False);  // Dosyayi Acmak Icin ...
    for ( j = 0; j < KayitSayisi; j++ ) {
          k = DosyadanOku();
           if ( j % 4  == 3 ) {
             k = ( k1 + k2 + k3 + k ) / 2;
             line( EskiX, EskiY, 75+( j/4 ), 450-k );
             EskiX = 75+( j/4 );
             EskiY = 450 - k;
             }
           else if ( j % 4 == 2 ) {
             k3 = k;
             }
```

```
            else if ( j % 4 == 1 ) {
              k2 = k;
              }
            else {
              k1 = k;
              }
          }
    Renk++;
    } //if
   } //for


 i = getch();
 GrafikKapat();

return(0);

}


/*****************************************************
***** Fonksiyon Adı : DosyaVar()                 *********
***** Parametreler  : DosyaNo -> Hangi numaralı  *********
*****                 dosyanın aranacak olduğu   *********
*****                 opt -> False ise sadece dosyayı *********
*****                 açar True ise hem açar hem de *********
*****                 dosyadaki kayıt sayısını bulur. *********
***** Dönen Değer   : Kendisi -> İstenen numaralı *********
*****                 dosyadaki satır sayısını (kayıt *********
*****                 sayısı) döner.             *********
***** Açıklama      : Numarası verilen dosyanın var *********
*****                 olup olmadığını bulur.     *********
*****************************************************/

int DosyaVar(int DosyaNo, int opt)
{
 int  Cikis;
 int  DosyadakiSatirSayisi;
 char Str3[3], Str20[20];


 strcpy(Str3,"00");
 sprintf(Str3, "%2d", DosyaNo);
 if (Str3[0] == ' ')
   Str3[0] = '0';
```

```
if ( UVarYok[0] == 'V' )
  strcpy(Str20, "data\\uv\\bit");
else if ( UVarYok[0] == 'Y' )
  strcpy(Str20, "data\\uy\\bit");

strcat(Str20, Str3);
strcat(Str20, ".out");

DosyadakiSatirSayisi = 0;

if ( (DosyaX = fopen(Str20, "rt")) == NULL ) {
  DosyadakiSatirSayisi = 0;
  }
else {
  Cikis = False;
  if ( opt == False )
    return(0);
  do {
    if ( fgets(Str20, 10, DosyaX) ) {
      Str20[6] = NULL;
      KuyrukIntegrali += atoi(Str20);
      DosyadakiSatirSayisi++;
      }
    else
      Cikis = True;
    } while ( !Cikis );
  fclose(DosyaX);
  }

return(DosyadakiSatirSayisi);
}
```

```
/*************************************************
**** Fonksiyon Adı   : DosyadanOku()            ******
**** Parametreler    : Yok                      ******
**** Dönen Değer     : Kendisi -> Okunan Değer  ******
**** Açıklama        : Açık durumda bulunan dosyada******
****                   işaretçinin bulunduğu alandaki ******
****                   kaydı okur.              ******
*************************************************/

int DosyadanOku()
{
char Str20[20];

 fgets(Str20, 10, DosyaX);

 return( atoi(Str20) );
}

/*************************************************
**** Fonksiyon Adı   : GrafikAc()               ******
**** Parametreler    : Yok                      ******
**** Dönen Değer     : Kendisi -> Yok           ******
**** Açıklama        : Grafik Ekranın açılmasını sağlar.*****
****                   Ekrana grafik için koordinatlar ******
****                   çizilir.                 ******
*************************************************/

int GrafikAc(void)
{
  int gdriver = DETECT, gmode, errorcode;
  int gectip, gecsayi, i;
  unsigned int size;

  initgraph(&gdriver, &gmode, "e:\\tc\\bgi");
  errorcode = graphresult();
  if (errorcode != grOk)
  {
    printf("Graphics error: %s\n", grapherrormsg(errorcode));
    printf("Press any key to halt:");
    getch();
    exit(1);          /* return with error code */
  }

  setbkcolor(15);
  cleardevice();
```

```
   // En Distaki Cercevenin Cizilmesi ..
   setcolor(1);
   setlinestyle(SOLID_LINE,1,1);
   line(0,17, 110, 17);
   line(534, 17, getmaxx(), 17 );
   rectangle(110, 0, 534, 18);
   setfillstyle(SOLID_FILL, 10);
   floodfill(111, 1, 1);
   gyaz(124, 6, "Uretim Ortaminda Zamana Gore Toplam Kuyruk Olusumu",9);

   line(0, getmaxy(), getmaxx(), getmaxy() );
   line(0,17, 0, getmaxy() );
   line(getmaxx(), 17, getmaxx(), getmaxy() );
   setlinestyle(0, 0, 3);
   line(75, 450, 600, 450);
   line(75, 20,  75,  450);
   line(600,450,590,445);
   line(600,450,590,455);
   line(75,20,80,30);
   line(75,20,70,30);
   setlinestyle(0, 0, 0);
   gyaz(555, 460, "Zaman", 9);
   gyaz(520, 470, "[adim sayisi]",9);
   gyaz(15, 20, "Toplam", 9);
   gyaz(15, 30, "Kuyruk",9);
   gyaz(10, 40, "Uzunlugu", 9);

   for ( i = 0; i < 20; i++ ) {
    line( 75 + ( i+1 ) * 25, 448, 75 + ( i+1 ) * 25, 453);
    }
   gyaz(75+ 5*25-10, 455,"500",1);
   gyaz(75+10*25-15, 455,"1000",1);
   gyaz(75+15*25-15, 455,"1500",1);
   gyaz(75+20*25-15, 437,"2000",1);

   for ( i = 0; i < 19; i++ ) {
    line( 72 , 450 - ( ( i+1 ) * 20 ), 77, 450 - ( ( i+1 ) * 20 ) );
    }

   gyaz(45, 447 - 5*20," 50",1);
   gyaz(45, 447 -10*20,"100",1);
   gyaz(45, 447 -15*20,"150",1);
   gyaz(45, 447 -19*20,"190",1);

 return 0;
}
```

```
/*****************************************************
**** Fonksiyon Adi   : GrafikKapat()                 *****
**** Parametreler    : Yok                           *****
**** Dönen Değer     : Kendisi -> Yok                *****
**** Açıklama        : Grafik Ekranın kapatılmasını sağlar. *****
*****************************************************/

int GrafikKapat(void)
{
 closegraph();
 return 0;
}

/*****************************************************
**** Fonksiyon Adı   : RenkBul()                     ******
**** Parametreler    : Gelen -> Renk numarası        ******
**** Dönen Değer     : RENK -> İlgili renk numarasının ******
****                   ekrana yazdırılması için açıklaması. ******
**** Açıklama        : Parametre olarak gelen renk   ******
****                   numarasına göre açıklamasını  ******
****                   bularak gönderir.             ******
*****************************************************/

int RenkBul(int Gelen, char  RENK[15])
{
 switch ( Gelen ) {
  case 1 : strcpy(RENK, "BLUE");
           break;
  case 2 : strcpy(RENK, "GREEN");
           break;
  case 3 : strcpy(RENK, "CYAN");
           break;
  case 4 : strcpy(RENK, "RED");
           break;
  case 5 : strcpy(RENK, "MAGENTA");
           break;
  case 6 : strcpy(RENK, "BROWN");
           break;
  case 7 : strcpy(RENK, "LIGHT GRAY");
           break;
  case 8 : strcpy(RENK, "DARK GRAY");
           break;
  case 9 : strcpy(RENK, "LIGHT BLUE");
           break;
  case 10: strcpy(RENK, "LIGHT GREEN");
           break;
```

```
   case 11: strcpy(RENK, "LIGHT CYAN");
            break;
   case 12: strcpy(RENK, "LIGHT RED");
            break;
   case 13: strcpy(RENK, "LIGHT MAGENTA");
            break;
   case 14: strcpy(RENK, "YELLOW");
            break;
   case 15: strcpy(RENK, "WHITE");
            break;
   }
 return;
}

/*****************************************************
**** Fonksiyon Adı   : gyaz()                    ******
**** Parametreler    : x -> Ekrandaki sütun parametresi ******
****                   y -> Ekrandaki satır parametresi ******
****                   yazı -> Ekranda yazılacak olan yazı ******
****                   _renk -> Yazının ekrana hangi ******
****                   renkte yazılacağı.        ******
****                   Dönen Değer  : Kendisi -> yok ******
**** Açıklama        : Ekranda verilen satır ve sütun değe-******
****                   rine göre grafik olarak yazı yazar. ******
*****************************************************/

int gyaz(int x, int y, char *yazi, int _renk)
{
 int EskiRenk;

 EskiRenk = getcolor();
 setcolor(_renk);
 outtextxy(x,y,yazi);
 setcolor(EskiRenk);
 return;

}

int EkranOku()
{
int sayac, Cikis, sonuc;

 GenelInit();
 clrscr();

 MFileOpen();
```

```
  strcpy(UVarYok, "V");
  strcpy(ChGrupNo,"1");
  yaz(43,7,UVarYok,9);
  yaz(43,9,ChGrupNo,9);

  sonuc = 0;
  sayac = 1;
  Cikis = False;
  do {
  switch(sayac)  {
     case 1 : oku(43,7,2,UVarYok,0,9,'y',&sonuc);
              break;
     case 2 : oku(43,9,2,ChGrupNo,1,9,'y',&sonuc);
              break;
     }

  if (sonuc == -100 || sonuc == 200)
    Cikis = True;
  else if (sonuc == 60 || sonuc == 0) {
    if (sayac == 2)
      sayac = 1;
    else
      sayac++;
    }
  else if (sonuc == 10) {
    if (sayac == 1)
      sayac = 2;
    else
      sayac--;
    }
  } while (!Cikis);

 return(sonuc);
}
```

```
MFileOpen()
{
int Yer;

  yaz(17,1,kisa_ad,2);
  yaz(68,1,Date,4);

  yaz(1,1,"USDUS_P_06",9);
  yaz(20,2,"Kuyruklarin Grafik Olarak Gösterimi",9);

  yaz(20, 7,"USDUS Var / Yok    : ",5);
  yaz(46, 7,"(V/Y)",7);

  yaz(20, 9,"3'lu Grup Numarasi : ",5);

  yaz(5 ,22,"F2 : KABUL",7);
  yaz(17,22,"F3 : SON  ",7);

return;
}
```

```
/*********************************************************************
                          Gr_Uzm_B.c
              Doktora Tezi, Bekleyen Çalışanların Grafik Olarak
              Gösterimlerinin Yapıldığı Program Parçası
                      Y. Müh. H. Engin DEMİRAY

NOT : Bu program kullanılarak elde edilen şekillerin listesi :

    • Şekil 4.6
    • Şekil 4.8
    • Şekil 4.10
    • Şekil 4.14
    • Şekil 4.16
    • Şekil 4.18

*********************************************************************/

#include <stdio.h>
#include <graphics.h>

#include "tanim.h"
#include "tool.h"

#define True 1
#define False 0


int  NoktaUzunlugu;
long int ToplamBekleme;

FILE *DosyaX;
char UVarYok[2], ChGrupNo[2];
```

```
/*******************************************************
**** Fonksiyon Adı   : main()                    **********
**** Açıklama        : Programın ana kısmı       **********
******************************************************* */

main()
{
int i, j, k, k1, k2, k3, KayitSayisi, MaxKayitSayisi;
int Renk, EskiX, EskiY, Ilk;
char YazilacakRenk[15];
int  GrupNo;


 if ( EkranOku() == -100 )
   return(0);

 yaz(1,13,"",9);

 GrupNo = atoi( ChGrupNo );

 KayitSayisi = MaxKayitSayisi = 0;

  i = (GrupNo - 1 ) * 3 + 1;

KayitSayisi = MaxKayitSayisi = 0;
for ( i = (GrupNo - 1 ) * 3 + 1; i <  (GrupNo - 1 ) * 3 + 4; i++ ) {
  ToplamBekleme = 0;
  KayitSayisi = DosyaVar(i, True);
  if ( KayitSayisi > 0 ) {
    RenkBul(i, YazilacakRenk);
    printf("File Sira No : %2d, Kayit Sayisi :  %4d, TBS : %5ld, Renk : %s \n",
                              i, KayitSayisi, ToplamBekleme, YazilacakRenk );
    }
  if ( KayitSayisi > MaxKayitSayisi)
    MaxKayitSayisi = KayitSayisi;
  }

printf("SONUC : %4d\n", MaxKayitSayisi);
NoktaUzunlugu = 1;
if ( MaxKayitSayisi > 500 ) {
  NoktaUzunlugu = MaxKayitSayisi/500 + 1;
  }

Renk = 1;
EskiX = 75;
EskiY = 450;
```

```
 i = getch();
 GrafikAc();

 for ( i = (GrupNo - 1 ) * 3 + 1; i <  (GrupNo - 1 ) * 3 + 4; i++ ) {
  KayitSayisi = DosyaVar(i, True);
  if ( KayitSayisi > 0 ) {
    Renk = i;
    setcolor(Renk);
    Ilk = True;
    k = k1 = k2 = k3 = 0;
    DosyaVar(i, False);  // Dosyayi Acmak Icin ...
    for ( j = 0; j < KayitSayisi; j++ ) {
           k = DosyadanOku();
            if ( j % 4  == 3 ) {
              k = ( k1 + k2 + k3 + k ) / 4;
              if ( Ilk ) {
                EskiX = 75+( j/4 );
                EskiY = 450-k*5;
                Ilk = False;
                }
              line( EskiX, EskiY, 75+( j/4 ), 450-k*5 );
              EskiX = 75+( j/4 );
              EskiY = 450-k*5;
              }
            else if ( j % 4 == 2 ) {
              k3 = k;
              }
            else if ( j % 4 == 1 ) {
              k2 = k;
              }
            else {
              k1 = k;
              }
           }
    Renk++;
    } //if
  } //for


 i = getch();
 GrafikKapat();

return(0);

}
```

```
/*****************************************************
***** Fonksiyon Adı : DosyaVar()                  *********
***** Parametreler  : DosyaNo -> Hangi numaralı   *********
*****                dosyanın aranacak olduğu     *********
*****                opt -> False ise sadece dosyayı *********
*****                açar True ise hem açar hem de *********
*****                dosyadaki kayıt sayısını bulur. *********
***** Dönen Değer  : Kendisi -> İstenen numaralı  *********
*****                dosyadaki satır sayısını (kayıt *********
*****                sayısı) döner.               *********
***** Açıklama     : Numarası verilen dosyanın var *********
*****                olup olmadığını bulur.       *********
*****************************************************/

int DosyaVar(int DosyaNo, int opt)
{
 int  Cikis;
 int  DosyadakiSatirSayisi;
 char Str3[3], Str20[20];


 strcpy(Str3,"00");
 sprintf(Str3, "%2d", DosyaNo);
 if (Str3[0] == ' ')
   Str3[0] = '0';

  if ( UVarYok[0] == 'V' )
   strcpy(Str20, "data\\uv\\bib");
 else if ( UVarYok[0] == 'Y' )
   strcpy(Str20, "data\\uy\\bib");

 strcat(Str20, Str3);
 strcat(Str20, ".out");

 DosyadakiSatirSayisi = 0;

 if ( (DosyaX = fopen(Str20, "rt")) == NULL ) {
   DosyadakiSatirSayisi = 0;
   }
 else {
   Cikis = False;
   if ( opt == False )
     return(0);
   do {
     if ( fgets(Str20, 10, DosyaX) ) {
       Str20[6] = NULL;
```

```
      ToplamBekleme += atoi(Str20);
      DosyadakiSatirSayisi++;
      }
     else
      Cikis = True;
    } while ( !Cikis );
  fclose(DosyaX);
  }

 return(DosyadakiSatirSayisi);
}

/*************************************************
**** Fonksiyon Adı   : DosyadanOku()          ******
**** Parametreler    : Yok                    ******
**** Dönen Değer     : Kendisi -> Okunan Değer******
**** Açıklama        : Açık durumda bulunan dosyada******
****                   işaretçinin bulunduğu alandaki ******
****                   kaydı okur.           ******
*************************************************/

int DosyadanOku()
{
char Str20[20];

  fgets(Str20, 10, DosyaX);

 return( atoi(Str20) );
}
```

```
/*************************************************
**** Fonksiyon Adı   : GrafikAc()                ******
**** Parametreler    : Yok                       ******
**** Dönen Değer     : Kendisi -> Yok            ******
**** Açıklama        : Grafik Ekranın açılmasını sağlar.*****
****                  Ekrana grafik için koordinatlar ******
****                  çizilir.                    ******
*************************************************/

int GrafikAc(void)
{
  int gdriver = DETECT, gmode, errorcode;
  int gectip, gecsayi, i;
  unsigned int size;

  initgraph(&gdriver, &gmode, "e:\\tc\\bgi");
  errorcode = graphresult();
  if (errorcode != grOk)
  {
    printf("Graphics error: %s\n", grapherrormsg(errorcode));
    printf("Press any key to halt:");
    getch();
    exit(1);            /* return with error code */
  }

  // En Distaki Cercevenin Cizilmesi ..
  setbkcolor(15);
  cleardevice();

  // En Distaki Cercevenin Cizilmesi ..
  setcolor(1);
  setlinestyle(SOLID_LINE,1,1);
  line(0,17, 110, 17);
  line(534, 17, getmaxx(), 17 );
  rectangle(110, 0, 534, 18);
  setfillstyle(SOLID_FILL, 10);
  floodfill(111, 1, 1);
  gyaz(124, 6, "Uretim Ortaminda Zamana Gore Toplam Bekleyen Sayisi",9);

  line(0, getmaxy(), getmaxx(), getmaxy() );

  line(0,17, 0, getmaxy() );
  line(getmaxx(), 17, getmaxx(), getmaxy() );

  setlinestyle(0, 0, 3);
  line(75, 450, 600, 450);
```

```
  line(75, 20,  75,  450);
  line(600,450,590,445);
  line(600,450,590,455);
  line(75,20,80,30);
  line(75,20,70,30);
  setlinestyle(0, 0, 0);
  gyaz(555, 460, "Zaman", 9);
  gyaz(520, 470, "[adim sayisi]",9);
  gyaz(15, 20, "Toplam", 9);
  gyaz(6, 30, "Bekleyen",9);
  gyaz(15, 40, "Sayisi", 9);
  for ( i = 0; i < 20; i++ ) {
   line( 75 + ( i+1 ) * 25, 448, 75 + ( i+1 ) * 25, 453);
   }
  gyaz(75+ 5*25-10, 455,"500",1);
  gyaz(75+10*25-15, 455,"1000",1);
  gyaz(75+15*25-15, 455,"1500",1);
  gyaz(75+20*25-15, 437,"2000",1);
  for ( i = 0; i < 19; i++ ) {
   line( 72 , 450 - ( ( i+1 ) * 20 ), 77, 450 - ( ( i+1 ) * 20 ) );
   }
  gyaz(45, 447 - 5*20," 20",1);
  gyaz(45, 447 -10*20," 40",1);
  gyaz(45, 447 -15*20," 60",1);
  gyaz(45, 447 -19*20," 76",1);
 return 0;
}

int EkranOku()
{
int sayac, Cikis, sonuc;

 GenelInit();
 clrscr();

 MFileOpen();

 strcpy(UVarYok, "V");
 strcpy(ChGrupNo,"1");
 yaz(43,7,UVarYok,9);
 yaz(43,9,ChGrupNo,9);

 sonuc = 0;
 sayac = 1;
 Cikis = False;
```

```
  do {
  switch(sayac)  {
     case 1 : oku(43,7,2,UVarYok,0,9,'y',&sonuc);
              break;
     case 2 : oku(43,9,2,ChGrupNo,1,9,'y',&sonuc);
              break;
     }

  if (sonuc == -100 || sonuc == 200)
    Cikis = True;
  else if (sonuc == 60 || sonuc == 0) {
    if (sayac == 2)
      sayac = 1;
    else
      sayac++;
    }
  else if (sonuc == 10) {
    if (sayac == 1)
      sayac = 2;
    else
      sayac--;
    }
  } while (!Cikis);

 return(sonuc);
}

MFileOpen()
{
int Yer;

  yaz(17,1,kisa_ad,2);
  yaz(68,1,Date,4);

  yaz(1,1,"USDUS_P_07",9);
  yaz(20,2,"Bekleyenlerin Grafik Olarak G”sterimi",9);

  yaz(20, 7,"USDUS Var / Yok    : ",5);
  yaz(46, 7,"(V/Y)",7);

  yaz(20, 9,"3'lu Grup Numaras□ : ",5);

  yaz(5 ,22,"F2 : KABUL",7);
  yaz(17,22,"F3 : SON  ",7);
return;
}
```

```
/*************************************************************************
```

Gr_Uz_OK.c

Doktora Tezi, Kuyrukların, Ortalamalarının Grafik Olarak
Gösterimlerinin Yapıldığı Program Parçası
Y. Müh. H. Engin DEMİRAY

NOT : Bu program kullanılarak elde edilen şekillerin listesi :

- Şekil 4.11
- Şekil 4.19

```
*************************************************************************/

#include <stdio.h>
#include <graphics.h>
#include "tanim.h"
#include "tool.h"

#define True 1
#define False 0

int  NoktaUzunlugu;
FILE *DosyaX;
int  dizi[2500];
int  IncelenenDosya, YeniDeneme, BulunanKayit;
char UVarYok[2];

main()
{
int i, j, k, k1, k2, k3, KayitSayisi, MaxKayitSayisi;
int Renk, EskiX, EskiY;
char YazilacakRenk[15];
int Ortalama=0;

 KayitSayisi = MaxKayitSayisi = 0;
 BulunanKayit = 0;
 ParametreDosyasiOku();
 DiziInit();

 if ( EkranOku() == -100 )
   return(0);

 yaz(1,9,"",9);
```

```
  for ( i = 0; i < 10; i++ ) {
    KayitSayisi = DosyaVar(i, True);
    if ( KayitSayisi > 0 ) {
      BulunanKayit++;
      RenkBul(i, YazilacakRenk);
      printf("File Sira No : %2d, Kayit Sayisi :  %4d,  Renk : %s \n",
                                  i, KayitSayisi, YazilacakRenk );
      }
    if ( KayitSayisi > MaxKayitSayisi)
      MaxKayitSayisi = KayitSayisi;

    Ortalama += KayitSayisi;
    }

  printf("SONUC : %4d          ORTALAMA :  %4d \n",
          MaxKayitSayisi, Ortalama/BulunanKayit);
  NoktaUzunlugu = 1;
  if ( MaxKayitSayisi > 500 ) {
    NoktaUzunlugu = MaxKayitSayisi/500 + 1;
    }

  Renk = 1;
  EskiX = 75;
  EskiY = 450;

  i = getch();
  GrafikAc();

  setcolor(Renk);
  setlinestyle(0,0,3);

  for (i = 1; i < 498; i++ ) {
    k1 = 75 + i;

    k2 = 450 - ( ( dizi[i*4]+dizi[i*4+1]+dizi[i*4+2]+dizi[i*4+3] ) /
                    ( BulunanKayit*2 ) );
    line( EskiX, EskiY, k1, k2);
    EskiX = k1;
    EskiY = k2;
    }
  i = getch();
  GrafikKapat();
 return(0);

}
```

```
/*************************************************************************
```

Gr_Uz_OB.c
Doktora Tezi, Bekleyenlerin Ortalama Olarak
Gösterimlerinin Yapıldığı Program Parçası
Y. Müh. H. Engin DEMİRAY

NOT : Bu program kullanılarak elde edilen şekillerin listesi :

- Şekil 4.12
- Şekil 4.20

```
*************************************************************************/

#include <stdio.h>
#include <graphics.h>
#include "tanim.h"
#include "tool.h"

#define True 1
#define False 0

int  NoktaUzunlugu;
FILE *DosyaX;
int  dizi[2500];
int  IncelenenDosya, YeniDeneme, BulunanKayit;
char UVarYok[2];


main()
{
int i, j, k, k1, k2, k3, KayitSayisi, MaxKayitSayisi;
int Renk, EskiX, EskiY, Ilk;
char YazilacakRenk[15];
int Ortalama=0;
int SonUzunluk;

 KayitSayisi = MaxKayitSayisi = 0;
 ParametreDosyasiOku();
 DiziInit();

 if ( EkranOku() == -100 )
   return(0);

 yaz(1,9,"",9);
```

```
KayitSayisi = MaxKayitSayisi = 0;
BulunanKayit = 0;
DiziInit();
for ( i = 0; i < 10; i++ ) {
  KayitSayisi = DosyaVar(i, True);
  if ( KayitSayisi > 0 ) {
    BulunanKayit++;
    RenkBul(i, YazilacakRenk);
    printf("File Sira No : %2d, Kayit Sayisi :  %4d,  Renk : %s \n",
                              i, KayitSayisi, YazilacakRenk );
    }
  if ( KayitSayisi > MaxKayitSayisi)
    MaxKayitSayisi = KayitSayisi;

  Ortalama += KayitSayisi;
  }

printf("SONUC : %4d          ORTALAMA :  %4d \n",
        MaxKayitSayisi, Ortalama/BulunanKayit);

NoktaUzunlugu = 1;
if ( MaxKayitSayisi > 500 ) {
  NoktaUzunlugu = MaxKayitSayisi/500 + 1;
  }

Renk = 1;
EskiX = 75;
EskiY = 450;

i = getch();
GrafikAc();

setcolor(Renk);
setlinestyle(0,0,3);

if ( UVarYok[0] == 'V' )
  SonUzunluk = 409;
else if ( UVarYok[0] == 'Y' )
  SonUzunluk = 495;

for (i = 0; i < SonUzunluk; i++ ) {
  k1 = 75 + i;
  k2 = 450 - ( ( dizi[i*4]+dizi[i*4+1]+dizi[i*4+2]+dizi[i*4+3] )*5 /
                ( BulunanKayit*4 ) );

  line( EskiX, EskiY, k1, k2);
```

```
    EskiX = k1;
    EskiY = k2;
    }

  i = getch();
  GrafikKapat();

 return(0);

}


ParametreDosyasiOku()
{
char Parametre[100];
char KUYRUK_DOSYA[13];
int i;
 FILE *__file_1;
 if ((__file_1 = fopen("data\\param1.txt", "rt")) == NULL) {
   fprintf(stderr, "Cannot open file. (param1.txt) \n");
   exit(0);
   }

 memtospace(Parametre);        /* Kisa Ad Icin Oku */
 fgets(Parametre,64,__file_1);
 memtospace(Parametre);        /* Uzman Sistem Yaklasimi Oku */
 fgets(Parametre,64,__file_1);
 memtospace(Parametre);        /* Kuyruk Esigini Oku */
 fgets(Parametre,64,__file_1);
 memtospace(Parametre);        /* Simulasyon Icinde Simulayon Adimi */
 fgets(Parametre,64,__file_1);
 memtospace(Parametre);        /* Kuyrugun Yazilacagi Dosya Adi */
 fgets(Parametre,64,__file_1);
 memtospace(Parametre);        /* Yeni Denemeler / Eski Datalar */
 fgets(Parametre,64,__file_1);
  /*
     Yeni Denemeler / Eski Datalar Modu     : Y
     012345678901234567890123456789012345678901234567890123456789012345 */
 if ( Parametre[41] == 'Y' )
   YeniDeneme = True;
 else
   YeniDeneme = False;
 fclose(__file_1);

return;
}
```

```
/*********************************************************************
```

Gr_Uz_TK.c

Doktora Tezi, Kuyrukta Bekleyenlerin Karşılaştırılmalı ve Grafik Olarak Gösterimlerinin Yapıldığı Program Parçası

Y. Müh. H. Engin DEMİRAY

NOT : Bu program kullanılarak elde edilen şekillerin listesi :

- Şekil 4.21

```
*********************************************************************/

#include <stdio.h>
#include <graphics.h>
#include "tanim.h"
#include "tool.h"

#define True 1
#define False 0

int  NoktaUzunlugu;
FILE *DosyaX;
int  dizi[2500];
int  IncelenenDosya, YeniDeneme, BulunanKayit;

main()
{
int i, j, k, k1, k2, k3, KayitSayisi, MaxKayitSayisi;
int EskiX, EskiY, Ilk;
char YazilacakRenk[15];

 KayitSayisi = MaxKayitSayisi = 0;
 BulunanKayit = 0;
 ParametreDosyasiOku();
 DiziInit();

 for ( i = 0; i < 10; i++ ) {
  KayitSayisi = DosyaVar(i, True);
  if ( KayitSayisi > 0 ) {
   RenkBul(i, YazilacakRenk);
   BulunanKayit++;
   }
  if ( KayitSayisi > MaxKayitSayisi)
   MaxKayitSayisi = KayitSayisi;
  }
```

```
NoktaUzunlugu = 1;
if ( MaxKayitSayisi > 500 ) {
  NoktaUzunlugu = MaxKayitSayisi/500 + 1;
  }

EskiX = 75;
EskiY = 450;

GrafikAc();

setcolor(1);
setlinestyle(0,0,3);
Ilk = True;

for (i = 1; i < 498; i++ ) {
  k1 = 75 + i;
  k2 = 450 - ( ( dizi[i*4]+dizi[i*4+1]+dizi[i*4+2]+dizi[i*4+3] ) /
               ( BulunanKayit*2 ) );


  line( EskiX, EskiY, k1, k2);

  if (Ilk) {
    setcolor(14);
    Ilk = False;
    }


  EskiX = k1;
  EskiY = k2;
  }

//#########################################################################

KayitSayisi = MaxKayitSayisi = 0;
BulunanKayit = 0;
DiziInit();

for ( i = 0; i < 10; i++ ) {
  KayitSayisi = DosyaVar(i, False);
  if ( KayitSayisi > 0 ) {
    RenkBul(i, YazilacakRenk);
    BulunanKayit++;
    }
  if ( KayitSayisi > MaxKayitSayisi)
    MaxKayitSayisi = KayitSayisi;
```

```
  }

 NoktaUzunlugu = 1;
 if ( MaxKayitSayisi > 500 ) {
  NoktaUzunlugu = MaxKayitSayisi/500 + 1;
  }

 EskiX = 75;
 EskiY = 450;

 setcolor(1);

 setlinestyle(0,0,3);

 for (i = 1; i < 498; i++ ) {
  k1 = 75 + i;
  k2 = 450 - ( ( dizi[i*4]+dizi[i*4+1]+dizi[i*4+2]+dizi[i*4+3] ) /
              ( BulunanKayit*2 ) );


  line( EskiX, EskiY, k1, k2);

  EskiX = k1;
  EskiY = k2;
  }


 i = getch();
 GrafikKapat();

return(0);

}
```

```
/*************************************************************************
```

Gr_Uz_TB.c

Doktora Tezi, Bekleyen Çalışanların Karşılaştırılmalı ve Toplam Olarak Gösterimlerinin Yapıldığı Program Parçası

Y. Müh. H. Engin DEMİRAY

NOT : Bu program kullanılarak elde edilen şekillerin listesi :

- Şekil 4.22

```
*************************************************************************/
#include <stdio.h>
#include <graphics.h>
#include "tanim.h"
#include "tool.h"

#define True 1
#define False 0

int NoktaUzunlugu;
FILE *DosyaX;
int dizi[2500];
int IncelenenDosya, YeniDeneme, BulunanKayit;

main()
{
int i, j, k, k1, k2, k3, KayitSayisi, MaxKayitSayisi;
int EskiX, EskiY, Ilk;
char YazilacakRenk[15];


 KayitSayisi = MaxKayitSayisi = 0;
 BulunanKayit = 0;
 ParametreDosyasiOku();
 DiziInit();

 for ( i = 0; i < 10; i++ ) {
  KayitSayisi = DosyaVar(i, True);
  if ( KayitSayisi > 0 ) {
   RenkBul(i, YazilacakRenk);
   BulunanKayit++;
   }
  if ( KayitSayisi > MaxKayitSayisi)
   MaxKayitSayisi = KayitSayisi;
  }
```

```
NoktaUzunlugu = 1;
if ( MaxKayitSayisi > 500 ) {
  NoktaUzunlugu = MaxKayitSayisi/500 + 1;
  }

EskiX = 75;
EskiY = 450;

GrafikAc();

setcolor(1);
setlinestyle(0,0,3);
Ilk = True;

for (i = 0; i < 409; i++ ) {
  k1 = 75 + i;
  k2 = 450 - ( ( dizi[i*4]+dizi[i*4+1]+dizi[i*4+2]+dizi[i*4+3] )*5 /
                ( BulunanKayit*4 ) );

  line( EskiX, EskiY, k1, k2);

  if (Ilk) {
    setcolor(14);
    Ilk = False;
    }

  EskiX = k1;
  EskiY = k2;
  }
//#########################################################################

KayitSayisi = MaxKayitSayisi = 0;
BulunanKayit = 0;
DiziInit();
for ( i = 0; i < 10; i++ ) {
  KayitSayisi = DosyaVar(i, False);
  if ( KayitSayisi > 0 ) {
    RenkBul(i, YazilacakRenk);
    BulunanKayit++;
    }
  if ( KayitSayisi > MaxKayitSayisi)
    MaxKayitSayisi = KayitSayisi;
  }

NoktaUzunlugu = 1;
```

```
  if ( MaxKayitSayisi > 500 ) {
    NoktaUzunlugu = MaxKayitSayisi/500 + 1;
    }

  EskiX = 75;
  EskiY = 450;

  Ilk = True;

  setlinestyle(0,0,3);
  setcolor(1);

  for (i = 0; i < 409; i++ ) {
    k1 = 75 + i;
    k2 = 450 - ( ( dizi[i*4]+dizi[i*4+1]+dizi[i*4+2]+dizi[i*4+3] )*5 /
                 ( BulunanKayit*4 ) );

    line( EskiX, EskiY, k1, k2);



    EskiX = k1;
    EskiY = k2;
    }



  i = getch();
  GrafikKapat();

 return(0);

}
```

EK B. POISSON UYGULAMASI

```
/*************************************************************
**** Fonksiyon Adı   : PoissonForMakina()                 **********
**** Açıklama        : Gelen parametre uyarınca           **********
****                   poisson bağıntısı uygulanarak sonuç **********
****                   döndürülür.                       **********
*************************************************************/


int PoissonForMakina(int Seviye)
{
 double x, lamda, t, Katsayi;
 FILE *output;
 int i1, Netice;

 lamda = 0.5;
 x = 0.0;
 Katsayi = Seviye / lamda;

 Rastgele(&x);
 t = ( -1.0 ) * ( 1/lamda ) * log(1-x);
 Netice = (int)(t * Katsayi);

 return(Netice);
}


Rastgele(dbDon)
double *dbDon;
{
 int gec;
 gec = rand() % 10000;
 *dbDon = gec / 10000.0;
 return;
}
```

ÖZGEÇMİŞ

Hasan Engin DEMİRAY, 1951 yılında İzmit'te doğdu. Öğrenimini sırasıyla, Paşabahçe ilkokulunda, Beykoz ve Maltepe ortaokullarında sürdürerek, 1968 yılında Pendik Lisesinden mezun oldu. Aynı yıl, İstanbul Üniversitesi Fen Fakültesi *'Matematik-Fizik Bölümü'* ne kaydoldu. 1969 yılında bu okuldan ayrılarak İstanbul Teknik Üniversitesi, Elektrik Fakültesi *'Elektronik ve Haberleşme Bölümü'* nde öğrenimine devam etmeye başladı, 1976 yılında mezun oldu.

Öğrenim yılları esnasında çalışmaya başladığı EL-SA Elektrik ve Taahhüt firmasında ortak ve yönetici olarak çalıştı. 1976 yılında buradan ayrılarak 6 ay kadar PTT Telefon Başmüdürlüğünde mühendis olarak görev yaptı. Daha sonra, askerlik görevi için gittiği Yassıada Deniz Eğitim Komutanlığında temel askerlik eğitimini tamamladıktan sonra, Çanakkale Boğaz Komutanlığına bağlı birliklerde (Alıcı İstasyon komutanlığı, Telekomünikasyon Şebekesi inşası esnasında Kontrol Mühendisliği gibi) çeşitli görevlerde bulundu (52'nci dönem Dz.Yd.Sby.Öğr. 1977-1979).

1979 yılında, terhisten hemen sonra; Türk Philips Tic.A.Ş.'de servis koordinatörlüğü görevine başladı. Aynı dönemde, İstanbul Teknik Üniversitesi Elektrik Fakültesi *'Elektronik ve Haberleşme Bölümü'* nde Master eğitimine başladı ve buradan 1981 yılında mezun oldu. 1983 yılında, Philips'teki görevinden ayrılarak, Cihan Elektronik A.Ş. de Servis Müdürü olarak göreve başladı. Daha sonra, müdürü olduğu bölümü şirket haline dönüştürerek, Cihan Elektronik Servis Hizmetleri A.Ş.'yi kurdu. Şirketin kurucu Genel Müdürlüğü göreviyle birlikte, Pazarlama Grup Başkanlığı görevinide, yoğun tez çalışmalarıyla birlikte, Aralık 1994'e kadar sürdürmüştür.