DOKUZ EYLÜL UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

DESIGNING AND IMPLEMENTING A WORKFLOW ENGINE WITH USING FINITE STATE AUTOMATA

by

Oğuzhan KAYIŞ

October, 2019

İZMİR

DESIGNING AND IMPLEMENTING A WORKFLOW ENGINE WITH USING FINITE STATE AUTOMATA

A Thesis Submitted to the
Graduate School of Natural and Applied Sciences of Dokuz Eylül University
In Partial Fulfillment of the Requirements for the Degree of Master of Science
in Computer Engineering Program

by
Oğuzhan KAYIŞ

October, 2019
İZMİR

M.Sc THESIS EXAMINATION RESULT FORM

We have read the thesis entitled **"DESIGNING AND IMPLEMENTING A WORKFLOW ENGINE WITH USING FINITE STATE AUTOMATA"** completed by **OĞUZHAN KAYIŞ** under supervision of **ASSIST. PROF. DR. SEMİH UTKU** and we certify that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assist. Prof. Dr. Semih UTKU

Supervisor

Doç.Dr. Mehmet Nidal Özcanhan

(Jury Member)

Prof. Dr. Mehmet Ünlütürk

(Jury Member)

Prof.Dr. Kadriye ERTEKİN

Director

Graduate School of Natural and Applied Sciences

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to project advisor Assist. Prof. Dr. Semih UTKU for continuous guidance, generous. I am grateful for your guidance throughout my graduate education and my life.

In addition, I would like to thank my co-workers for their support, especially to Veli MUTLU and Ruhi YARDAM.

At last, I would like to thank my family for their trust, patience and encouragement and thanks to my dear dentist friend MK.

Oğuzhan KAYIŞ

DESIGNING AND IMPLEMENTING A WORKFLOW ENGINE WITH USING FINITE STATE AUTOMATA

ABSTRACT

This study discusses the design and implementation of a software component that is called workflow. Workflow is a technology which is used by generally in large scale companies in industry for organizing and tracking different objects. The challenge, which we undertake in the study, designing a workflow engine to manage complex processing steps dynamically. The main issue is creating a dynamic system that is flexible enough to adapt to different businesses and different processes within the enterprises. The idea behind the workflow is like some of the models in computer science; graphs, finite state automata and petrinets etc. Finite state automata (FSA) used to model workflow. States of the FSA is representing the states of workflow and a standard workflow has start and stop states like FSA.

Business process in industrial areas changes over time according to many factors. For this purpose, within the scope of the study, a rule engine integrated with the workflow engine was developed. The rule engine enables the workflow engine to adapt to changing business processes at runtime. The system was designed to automate the follow-up processes of the documents connected to the products in a technology products company. Documents are related to materials of the company and designed system integrated into the material management module of ERP system of the company. After go-live document request log data stored in custom tables. With using the data collected from document requests, were analysed using sequential pattern mining methods.

Keywords: Workflow engine, rule engine, document tracking system, sequential pattern mining

SONLU DURUM OTOMATI KULLANARAK İŞ AKIŞ MOTORU TASARIMI VE GERÇEKLEŞTİRİMİ

ÖZ

Bu çalışmada, iş akışı adı verilen bir yazılım bileşeninin tasarımını ve gerçekleştirimini incelenmektedir. İş akışı, genellikle endüstrideki büyük ölçekli şirketler tarafından farklı nesneleri takip etmek için kullanılan bir teknolojidir. Çalışmada üstlendiğimiz zorluk, karmaşık işlem adımlarını dinamik olarak yönetmek için bir iş akışı motoru tasarlamaktır. Temel amaç, farklı işletmelere ve işletmelerdeki farklı süreçlere uyum sağlayabilecek esneklikte bir dinamik sistem oluşturmaktır. İş akışının arkasındaki fikir, bilgisayar bilimindeki bazı modellere benzer; grafikler, sonlu durum otomatları ve petrinets vs. Çalışma kapsamında iş akışını modellemek sonlu durum otomatları (SDO) kullanılmıştır. SDO'nun durumları, iş akışı durumlarını temsil etmektedir ve standart bir iş akışında SDO gibi başlangıç ve bitiş durumları bulunur.

Endüstriyel alanlarda iş süreci zaman içerisinde birçok faktöre göre değişmektedir. Bu amaçla, çalışma kapsamında iş akışı motoruyla bütünleşmiş bir kural motoru geliştirilmiştir. Kural motoru, iş akışı motorunun çalışma zamanında değişen iş süreçlerine adapte olmasını sağlamaktadır. Sistem, bir teknoloji ürünleri şirketinde ürünlere bağlı belgelerin takip işlemlerini otomatikleştirmek için tasarlanmıştır. Belgeler firmanın malzemeleriyle ilgilidir ve firmanın ERP sisteminin malzeme yönetim modülüne entegre edilmiştir. Canlı sistem geçişi sonra belge talepleri ile ilgili bilgiler saklanmıştır. Bu bilgiler kullanılarak sıralı örüntü madenciliği yöntemi ile analizler gerçekleştirilmiştir.

Anahtar kelimeler: İş akış motoru, kural motoru, belge takip sistemi, sıralı örüntü madenciliği

CONTENTS

Page

LIST OF FIGURES

Page

LIST OF TABLES

Page

CHAPTER ONE
INTRODUCTION

Towards the end of the 20th century, firms began to grow by taking other firms or divide company into various subunits. Group of companies consisting of different units are divided into sub-modules such as sales, logistics and distribution, marketing, production, finance in their internal business. The communication and systematic necessity of the sub-units of the growing companies required the formation of information technology units and ERP solutions (Chung & Synder, 1999). The information and technology departments established within the developing companies produce software-based communication solutions. These solutions, which are fed by electronic mail infrastructure, have produced different workflow processes and workflow engines integrated with these modules have been developed to manage these processes.

The workflow engine is a system that manages business processes in an industrial sense with software and human resources (Rosemann & Brocke, 2015). Although companies use their workflow engines to track documents such as sales order, purchase order, invoice and delivery note, which are included in the process especially by the sales & distribution and accounting departments, they also use it for the sharing of product-related information between departments in the production area.

1.1 Overview

In the industrial sense, work usually consists of more than one activity. Each activity is performed by a specific processing unit. Examples of these units are working groups, software or database management systems (Rusinkiewicz & Sheth, 1995). Workflow systems are also grouped under two different groups. The first is an activity-based workflow management model. This model focuses on the activities required to complete the work.

Another type is an object-based application. In this model, contrary to the first one, the possible movements of the object are defined for completion of the work (Bergmann, 2007). This study is designed to develop an object-based workflow engine.

In the study, finite state machine structure is used in workflow engine modelling stage. In this aspect the study is also answers the question of how a finite state machine can be stored in a relational database. In addition, a rule engine that strengthens the workflow engine with flexibility is designed and realized.

The rule engine is a software component that allows individuals who are not direct developers to intervene in the flow of a business process management system. (McCoy & Sinur, 2006) It has become very popular in recent years because it is easy to read and maintainable (De Leusse, Kwolek, & Zielinski, 2012). A business rule is an expression for determining a flow and procedure. Like a basic software language, the user can compare the expressions with the developed rule engine and set up equations that can be true or false. It is possible to make assignments according to the result of these equations.

The solution was designed to virtualize the processes of tracking documents related to products in a technology manufacturing company. The process starts with the document requesting by the foreign trade department of the company. Then the request is forwarded to previously classficated departments. Process finally ends with any of approval, rejection, preparation of documents. Before being installed from this system, this process, which was managed manually in the company, was automated with this work. User transaction-based data is logged in the developed system.

Sequential pattern mining techniques were applied on the data obtained from the document requests entered as a result of the first six months after the project went live. The results were shared with the relevant business units and it was proposed that some requested documents be requested as a package.

While this work essentially involves the design and realization of a workflow engine, in the background it offers an additional rule engine design that will give the workflow engine flexibility. In addition, the data obtained from the sample document tracking system of the study and data mining methods were analysed. In this respect, the application of sequential pattern mining on a workflow engine has been realized in this study.

1.2 The aim of the thesis

Workflow engine is a software component used for tracking objects. Designing and implementing a workflow engine is the main aim of the thesis. The system is intended to adapt to the dynamic workflow processes. For achieving the goal, a rule engine designed and implemented.

Implementation applied on an ERP system of a company to tracking material related documents. The company uses SAP as ERP system and developments applied to material management module of the system. The aim of the implementing the workflow engine is the automation of manual document tracking processes.

1.3 Thesis organization

The thesis is divided into 8 chapters. The description of the workflow end rule engines and an overview is given in Chapter 1. In Chapter 2 related works given about workflow engine, rule engine and data mining applications. Chapter 3 explains workflow engine in a detailed view. Chapter 4 is related to classical patterns of the workflow engine. Chapter 5 explains the details of the implementation. Chapter 6 is the part of rule engine. There are explanations about sequential pattern mining application of Chapter 7. Conclusion is given in Chapter 8.

CHAPTER TWO
RELATED WORKS

In the literature, it is seen that the studies are processed in two different ways through the workflow and rule engine. Although these systems are generally developed independently from each other, there are cases where the ERP (Enterprise Resource Planning) systems and these two products are used in an integrated structure.

In an article published by a group of scientists in 1995, they addressed workflow management processes as a natural consequence and necessity of developing industrial life. In the study, different approaches in the process from modelling to implementation of workflow are classified and terms used in workflow management are explained (Georgakopoulos, Hornick, & Sheth, 1995).

In a book published in 2000, Van Der Aalst and others thoroughly examined workflow processes, management, and the relationship between this process and information technologies. In the model presented in this study, a modelling is described through petrinets. The areas in which the study can be applied are exemplified on different branches of industry such as a travel agency, bicycle factory, driving school. In this study, it is emphasized that the workflow should be designed in a sustainable and readable structure in order to increase the efficiency of the enterprise (Van Der Aalst, Van Hee, K. M, & van Hee K, 2004).

In another study published in 2010 by Chun Ouyang and others, they referred to an earlier study and discussed WF-nets as a subset of petroleum and examined the project YAWL under different headings. In this study, which is mentioned about general concepts such as how a workflow engine can define the process, like our study, data mining is also emphasized. It is mentioned that both system and transaction logs can contain meaningful data and a study on this data can be useful (Rosemann et al., 2015).

In the study published in 2018, they divided the structures used for workflow modelling into static and dynamic. In this article, it is emphasized that the needs of the

industry change dynamically in daily life, but the frequently used workflow engines are static and suggested for the dynamic workflow engine. In addition, the basic patterns that we explained under the title of workflow engine are also explained in this study (Khawla & Molnár, 2018).

In the study of Stoilova and Stoilov, which describe the evolution of workflow engines; Under the heading of evolutionary development of workflow engines, both the historical process and the workflow engines are classified in three different ways. These headings; commercial and scientific software, tools according to the software language, classified according to the standards they support (Stoilova & Stoilov, 2006).

De Ley and Jacops have also developed a rule engine. Erwin and Jacops' articles about the Java-based rule engine have emphasized that although the rule engine is a software that can be used in many areas, it is not widely used (De Ley & Jacobs, 2011). The authors mentioned that the developed solution can work integrated with any Java application and emphasized that the rule engine can be considered as a notification tool.

In the study of Agrawal and others, the logs of the sample workflow diagrams examined over the graph model are emphasized. Both the real and the synthetic data generated from the samples, various analyses were performed and the starting and ending points of a workflow were sorted and data sets were formed. Finding the shortest path on the graph model or determining the noise data was applied to the data sets (Agrawal, Gunopulos, & Leymann, 1998).

It was first published in 2003 by Van der Aalst et al., In another application using workflow engine applications and data mining methods. In this study, the starting point of data mining on workflow engines is defined as log. In this study, examples of how meaningful data can be extracted from transactional log data are also given examples from transaction log data in ERP, CRM (Customer Relationship Management) applications (Van der Aalst et al., 2003).

In a study published in 2011, data mining methods over workflow engines were also emphasized. However, while other studies are about to extract meaningful data from logs, the aim of this study is; and to design a new model with petrinets (Singh & Aggarwal, 2011). Since the determination of the model in workflow management systems requires high responsibility and time for the work team, data mining has been proposed for this process.

CHAPTER THREE
WORKFLOW ENGINE

Workflow engine performed in the scope of another study covers the rules of automating all or part of a work process in an industrial sense (Allen, 2011). Since these rules are like the structure of a finite state machine, it is also emphasized how a finite state machine can be modelled in a relational database.

A basic flow includes the following structures;

- States: Any structure that can be found at a time for the tracking object is defined as a state. In addition to special cases such as start, end, cancellation, there may be numerous project-based specialized cases.
- Transaction: It is the process of switching from one state to another. Due to the business nature, it is not usually necessary to establish a link between all states. All transitions from the initial state to the other state and so on are called processes.
- Actions: The action required to perform the action is called an action. In a workflow engine, actions must be predetermined and deciding which action to complete and which action.
- Authorization: Transitions between states are triggered by actions. The triggering of these actions is necessary for the safety of the work process and for the separation of the owners of different jobs.

Figure 3.1 shows the flowchart of a document for the document tracking system where we apply the workflow engine.

In the workflow diagram shown in Figure 3.1, S1, S2, S3, S4, S5 represent the states. States given in the sample workflow diagram are categorized as below.

- S1: Start state,
- S2, S4: Interval states,
- S3: Cancel state,
- S4: Final state.

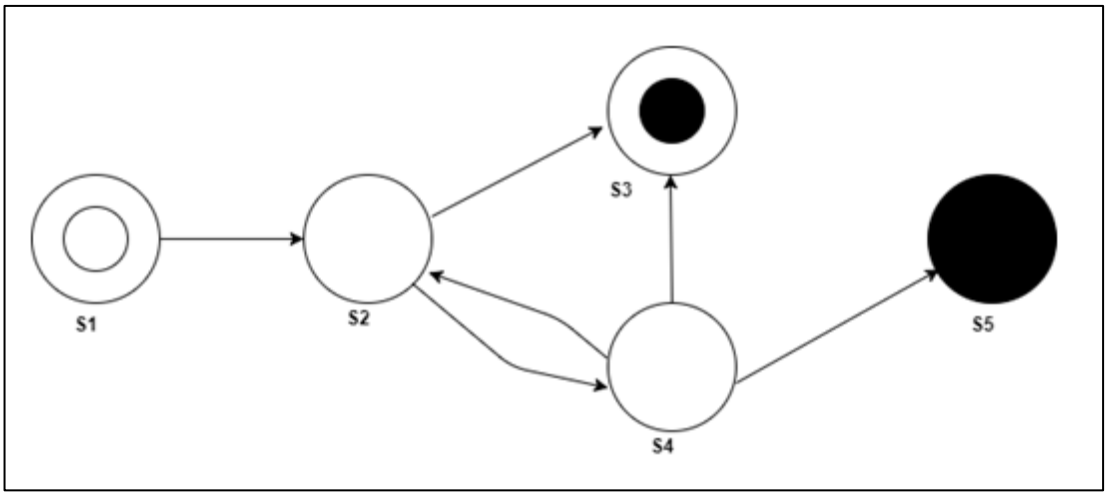


Figure 3.1 A sample workflow diagram

The object to be traced can be found in any of the above states at a time t. An object can only be found in one state at a time *t*, as the model is modelled with deterministic finite state machine (Lucas & Reynolds, 2005).

The arrows shown between the states represent processes. An ID is defined for each process in the system. These IDs will be used to create valid authorizations for transactions. Operations identified by a key in the background without repetition are also categorized into document tracking. When the buttons set on the front face are triggered, the corresponding type of operation is triggered.

List of the processes shown in Figure 3.1 is given below.

- S1-S2: Request creation,
- S2-S4 and S4-S2: Switching the next or previous states,
- S2-S3 and S4-S3: Cancelling operation,
- S4-S5: Switching to final state.

Actions is the necessary steps to handle these operations. In the developed structure, actions are also classified in order to trigger the front face. The following types of action are defined for the transition between the states listed. These types of action can be given in a group in the workflow or in separate groups as following; creating request, switching to another state, recall action, adding document to the request.

Ultimately, the necessary controls for these actions constitute the authority structure. In the developed system, authorization action and authorized group were used together.

It is possible to keep a conventional finite state machine in a matrix structure (Xu & Hong, 2012). However, a relational database was created because repetitive IDs are required for a structure that includes actions, states, and operations. The tables starting with the "_" mark in the database diagram (ER Diagram) in Figure 3.2 below are for the objects of the document tracking system which is the sample implementation of the workflow engine and the other tables belong to the general structure.

The contents of the tables shown in Figure 3.2 are explained below.

State_TP: It is used to store the state types. In basic terms it stores states like start state type, interval state type, final state type etc. This is used only storing master data. Therefore, it is not including any ID or tracked object classified data.

States: It is the data table of State_TP master table. It includes an ID as an attribute. The ID provide having multiple states in a process type of the flow. In document tracking system, document type is defined as a key. Therefore, state ID changes both the combination of state type and the process type.

Action_TP: It is used to represent action types, relate to table containing actions. This is used only for master data. There is not any ID information in this table.

Actions: It is the data table of Action_TP master table. In includes an ID attribute because same reason of ID attribute on States table. There may be defined more than one action type per a flow. Therefore, an ID attribute defined for the table.

_Doc_TP: Each non-repetitive flow within the document tracking system is identified as the document type key. It is the general key of the workflow engine

implementation. System tracks documents but all the rules are defined base on document types.

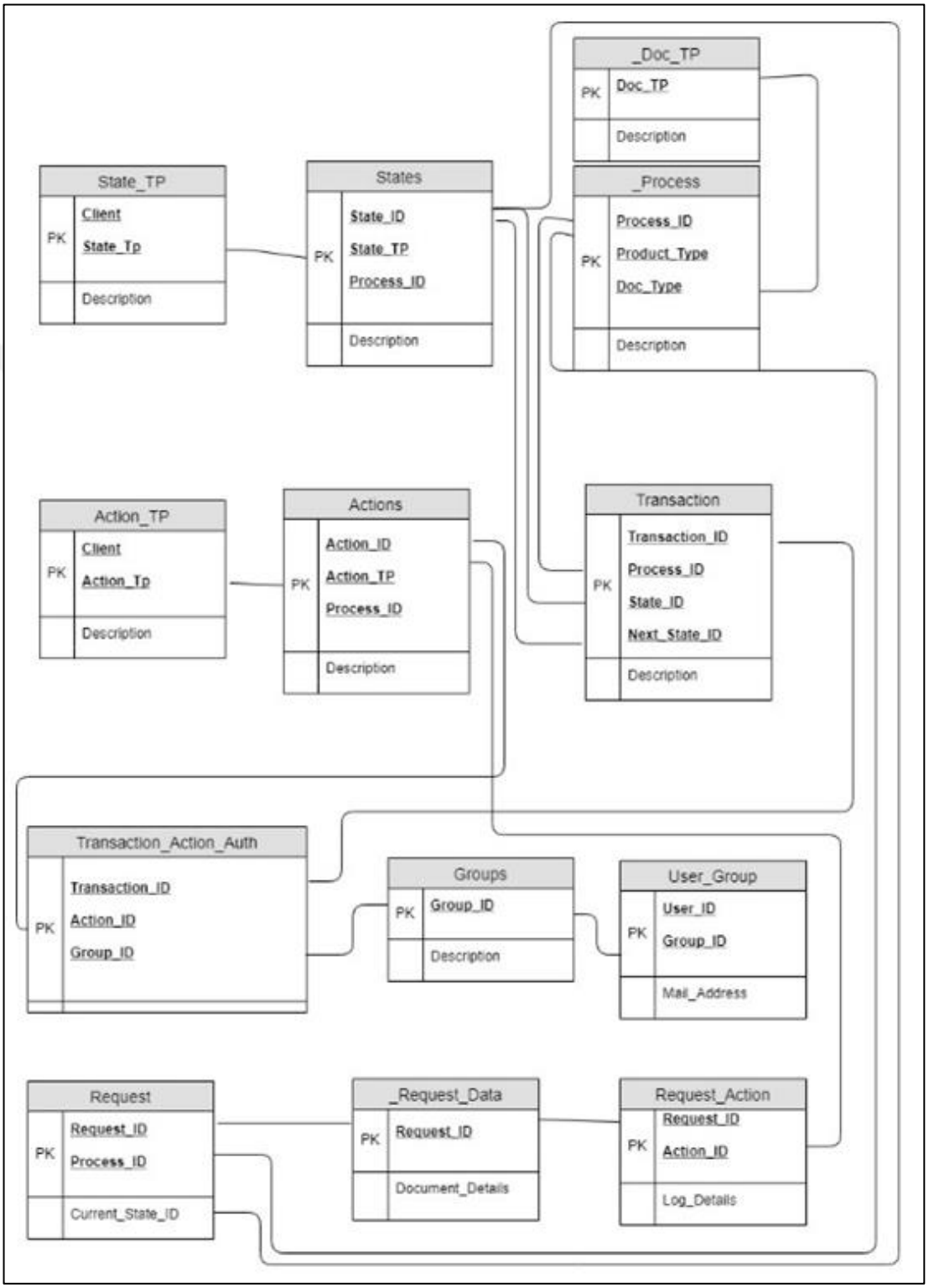


Figure 3.2 ER diagram

_Process: It represents each flow in the document tracking system. Each process is customized with the type of material to be requested and the type of document in this system. Process ID is used other tables as a foreign key attribute.

Transaction: It shows all steps in a process. This also shows the other situation in which one can be passed. Since a separate ID is shown for each state, the transition from one state to more than one state can also be defined.

Groups: It is the table that defines the units to be used in the authorizations. It does not contain any system usernames. This table used as a master table for storing department information.

User_Group: It is used to map departments of company with system usernames. Authorization is related to group information. And each user has a username which related to user group.

Transaction_Action_Auth: It is used to define which action applies to each action and which group is authorized for that action. It is a composed key table which contains both action and authorized group information. System uses this table that manages authorization. As an addition there is a superuser role defined in front-end. Superuser has authorization to both for data and operation related processes. Data related authorization defined related to process.

Request: It represents a request object that can be performed for each process. It is defined as a workflow engine table. Details of the request object is also stored another table.

_Request_Data: It represents each request generated for document tracking. In this table, detailed information such as material number for document tracking, e-mail address of the requestor is also stored. The table is the detail table of the Request table.

Request_Action: It is used for storing operational log information. When a user creates a request or a state is changed, system creates a record. Stores date, time, and user information and transaction information. The information in this table is mainly used for sequential pattern mining application.

CHAPTER FOUR
WORKFLOW PATTERNS

In a workflow engine, the basic control flow patterns that we cannot model due to the structure of the processes we use and the processes we use are listed below. These flow patterns generally refer to the characteristics of workflow engines.

1. Sequence
2. Parallel Split
3. Synchronization AND-Join
4. Exclusive Choice XOR-Split
5. Simple Merge XOR-Join
6. Discriminator
7. Arbitrary Cycles
8. Implicit Termination
9. Cancel Case.

4.1 Sequence

The sequence workflow model is the most basic pattern representing the transition from one state to another state linearly, regardless of any condition. The graphical representation is shown in Figure 4.1. In the document tracking system switching from one state to another state is used as sequence model.

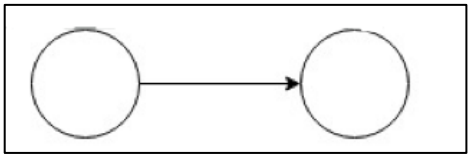


Figure 4.1 Sequence model

4.2 Parallel Split

The parallel division workflow pattern divides a flow. In this structure, it represents the condition in which a flow is unconditionally divided into multiple states in parallel. Graphical representation is given in Figure 4.2.

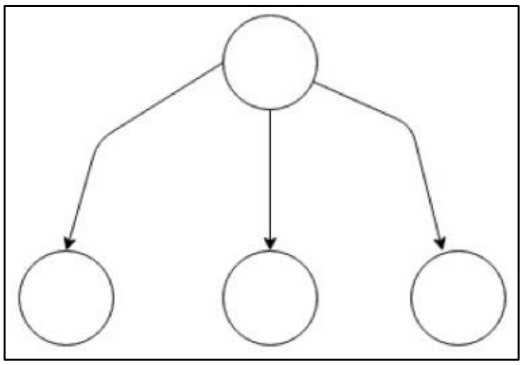


Figure 4.2 Parallel split model

4.3 Synchronization AND-Join

The synchronization workflow model refers to the synchronization of multiple parallel states to a single state. In this model, parallel steps are combined in exactly one case. Graphical representation is shown in Figure 4.3.

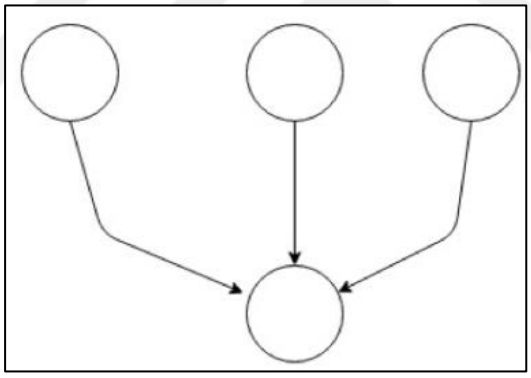


Figure 4.3 Synchronization model

4.4 Exclusive Choice XOR-Split

An exclusive selection workflow pattern defines multiple paths. Within this model, an object can exist in more than one state. The finite-state machine used in this study does not support this structure and was not developed in accordance with this pattern. Graphical representation is given in Figure 4.4.

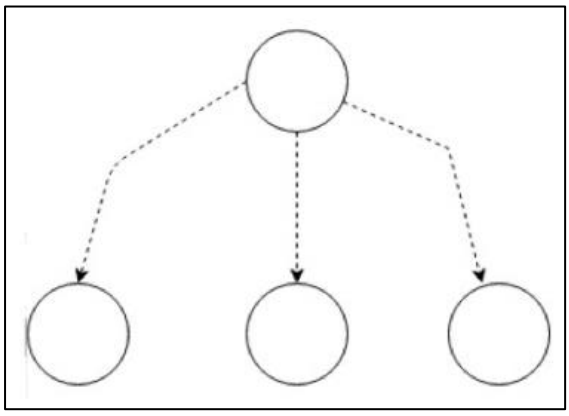


Figure 4.4 Exclusive choose model

4.5 Simple Merge XOR-Join

The simple merge model is used to merge defined paths. Exact defining is defined for a case. Like the special selection model, this structure was not developed in this study. Graphical representation is shown in Figure 4.5.

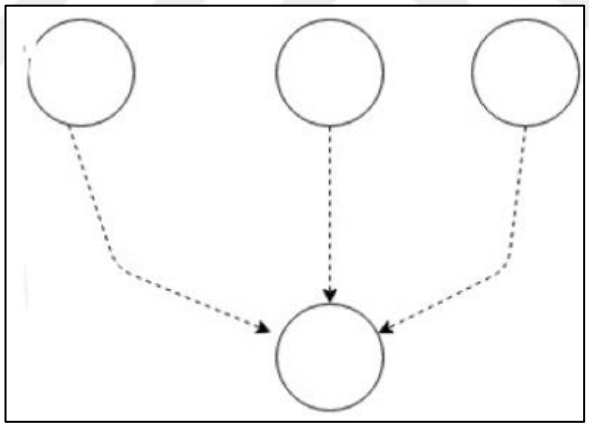


Figure 4.5 Simple merge model

4.6 Discriminator

This model is developed for assumptions jobs. It is a situation where parallel division is used jointly with special selection. It is the case where multiple states are linked to one state, but all incoming states must be completed before the object can move to the next state.

Although this model has not been developed in the scope of the document tracking system, the proposed structure has been designed to be suitable for the implementation of this model.

4.7 Arbitrary Cycle

A general structure of workflow models is that they do not allow loops within them. Repeating one or more events in workflow engines is not supported. Although no additional study was conducted to prevent loops, loops were not performed in the models applied.

4.8 Implicit Termination

Termination of the workflow is generally defined as the state where no new activity is allowed in the last case the object arrives. However, in some models it is also possible to terminate the process directly. Within the scope of this study, an infrastructure has been created in which users who submit a document request can assign their requests to cancellation status regardless of their status.

4.9 Cancel Case

It is a model developed to cancel a flow model. In the scope of this study, the cancellation type case was designed.

The comparison of the described models with other workflow systems is given in Table 4.1 (Bergmann, 2007).

Table 4.1 Comparation of workflow engine

	ZETA Components	(Van der aalst, & Ter Hofstede, 2005)	(Pirt, 2004)	Galaxy comm.	RADICORE	Visual workflow	Verve workflow	Staffware	IBM workflow	(Ran, Brebner, & Gorton, 2001)	I-flow	SAP R / 3	Thesis
4.1	√	√	√	√	√	√	√	√	√	√	√	√	√
4.2	√	√	√	√	√	√	√	√	√	√	√	√	√
4.3	√	√		√	√	√	√	√	√	√	√	√	
4.4	√	√		√	√	√	√	√	√	√	√	√	
4.5	√	√					√			√		√	√
4.6		√				√	√	√	√	√	√	√	
4.7	√	√			√		√	√		√		√	√
4.8	√	√			√		√	√		√		√	√
4.9	√	√				√	√			√		√	√
Rules													√

CHAPTER FIVE
WORKFLOW ENGINE IMPLEMENTATION

Designed system implemented on SAP system of a technology manufacturing company. System designed to manage material related documents. It works integrated with material management model of SAP.

Material Management (MM) is one the major modules of SAP system. It includes plenty of screens and tables. Most of the tables of module includes a common key which is material number. This material number field also became a key field in our implementation. For creating a full flow some maintenance operation is necessary. After examined system requirements, we decided to design system to allow creating different flows related with document type, material type, and material revision number in same cases.

Below is an example of the EWF system through a document workflow and flow charts. However, the document outside the system may be adapted to track the flow of any object. Figure 5.1 shows the SAP menu created for the flow. In this menu, which is divided into two basic sections as adaptations (UYARLAMALAR) and reports (RAPORLAR), the maintenance screens required for a flow definition are listed. From these menu elements, Document Types (DOKUMAN TIPLERI) and Flow Definitions (AKIS TANIMLARI) document flow are defined specifically and flow document type and document request via system.

In order to define the flow of an object, it is first necessary to determine what conditions the flow on the system will change. The flow parameters determined by analysing on an institution basis are then added to the system's user profiles explicitly. In the example application, the flow type of the document and the type of the product are determined on a basis. Before the documents are user defined entries, users need to log in to the screen which determines which documents should be followed through the system. Figure 5.2 shows the maintenance screen of document type part. Document

type table includes some business rule related specification such as automatic creation of the document etc.

Figure 5.1 SAP menu of the document tracking system

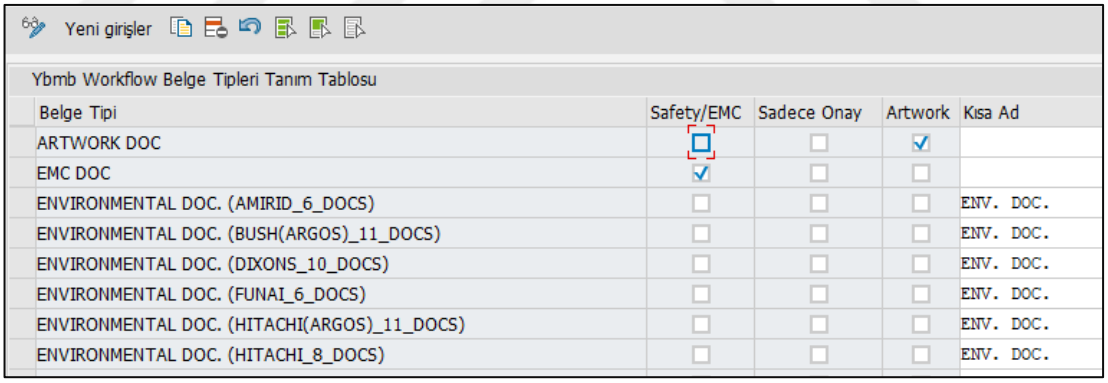


Figure 5.2 Document type maintenance screen

The part of this screen which is connected to the system is the document type section. This field is used as a reference in flow descriptors, but user requests on document type basis can also be added on this screen. In this example application, the document name that the user wishes to save is specified as a short name, and the documents with the automatic attribute are entered in the maintenance screen. In the system, document requests are answered by different groups, according to the type of the document which has been requested.

The group definitions are effective in the flow screens, the mail notifications of the workflow and the reports of the system. They are entered via the maintenance screen shown in Figure 5.3 and Figure 5.4 respectively.

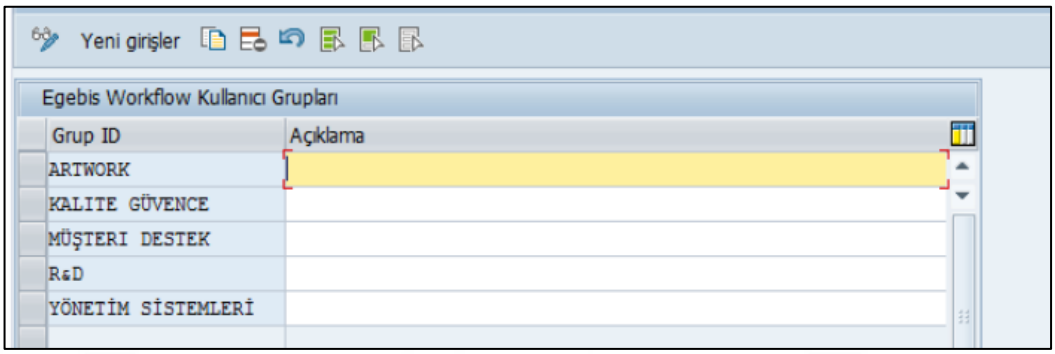


Figure 5.3 User group definition

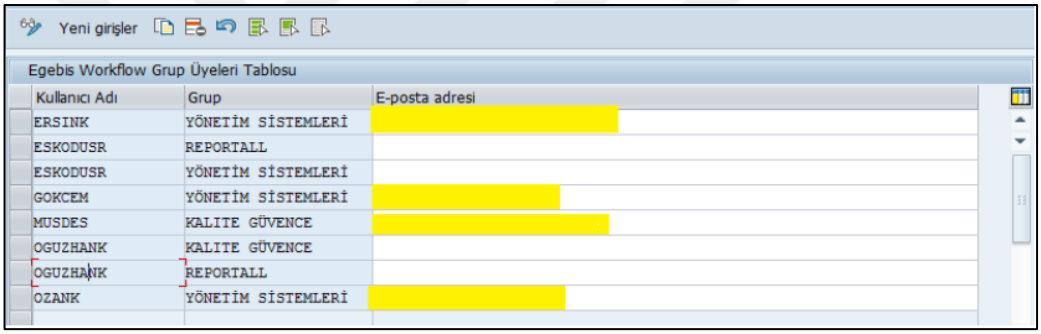


Figure 5.4 Users and user groups relations

The system checks the authorization transactions through the SAP usernames. The screen shown in Figure 5.4 is used to associate the username and group fields, and if an additional email address is to be used on the system's mail address, it is also added via this screen.

The entire flow of the document over the system is managed through the column that is identified as Process ID. In Figure 5.5, it is examined the detail of the record on the red frame. We will use this column information when we need to determine what steps are there for this record with key C01, how these steps are linked to each other, and authorization should be done separately.

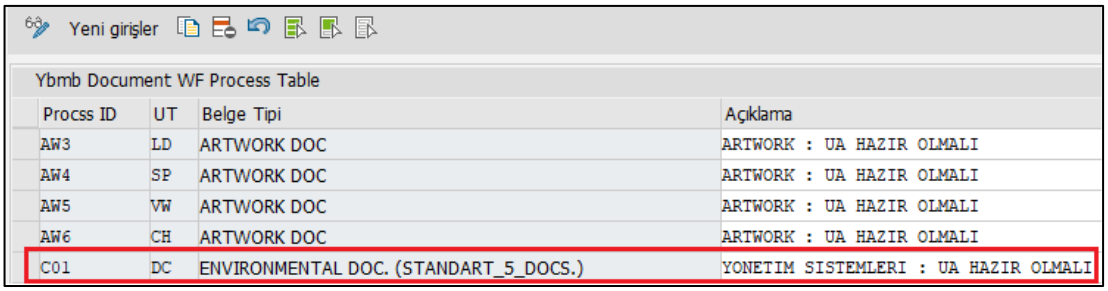


Figure 5.5 Flow definition

Flow is the combining of states of the objects. So as soon as flow is labelled like shown in Figure 5.5, the status of the states must be determined. According to business rules of the system there are six main status types. Even if this can be change related with the business rule of applied company, start and complete case must exist for providing functionality of finite state machine supported workflow engine. The six-status type is given below.

- Start state
- Normal (interval) state
- Waiting BOM (specific for document flow)
- Waiting production (specific for document flow)
- Complete state
- Cancel state.

The first step for the C01 flow as exemplified in Figure 5.5 is the identification of conditions that will be used in the flow. The flow is defined as contains each state type ones, but it may take any type of the state any number of pieces. States of the flow C01 is given in Figure 5.6.

At any time during the document workflow, the user may move the current flow to another state by performing one of the actions described in Figure 5.7 and Figure 5.8. Defined actions are listed below, can be edited on demand, and new action types can be added. Users can do the following operations.

- Approving a request
- Cancelling a request
- Creating a request

- Adding document path to the response a request.

Egebis Workflow State

State ID	State Tp	Procss ID	Tanım	Açıklama
570	ST	C01		
571	NR	C01		
572	CN	C01		
573	CM	C01		
574	UA	C01		

Figure 5.6 States

Egebis Workflow Action Types

Tip	Açıklama
CMF	COMFIRM
CNC	CANCEL
CRT	OLUŞTURMA
URL	LINK EKLEME

Figure 5.7 Actions

The actions defined in the flow C01 given in the example of Figure 5.5 are defined as shown in Figure 5.7. In this flow, although one action object is created for request creation, rejection, acknowledgement and link insertion actions, *N* pieces of action types can be found.

Egebis Workflow Action Table

Action ID	Action Type	Process ID	Tanım	Açıklama
403	CMF	C01		
404	CNC	C01		
405	CRT	C01		
406	URL	C01		

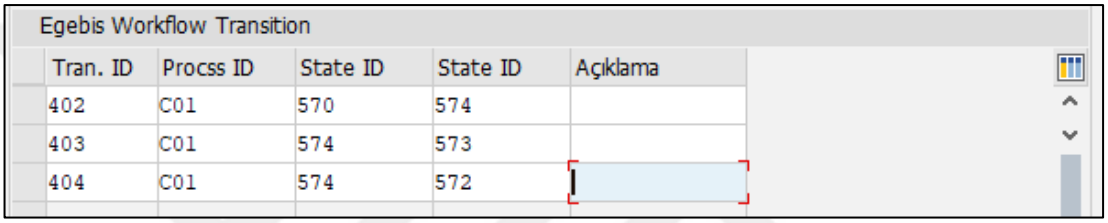
Figure 5.8 Actions of flow

The connection between states that we added in Figure 5.6 shown in Figure 5.9. According to the definition of the process, there are three possible transitions defined at flow C01.

First of them is the transition 402. It is shown at first row of the table that shown in Figure 10. It connects the state 570 to the state 574. State 570 represents a start state and the state 574 represent a BOM waiting state. (Single way direction)

Second of them is the transition 403. It connects the state 574 to the state 573. It is a transition from BOM waiting state to complete state.

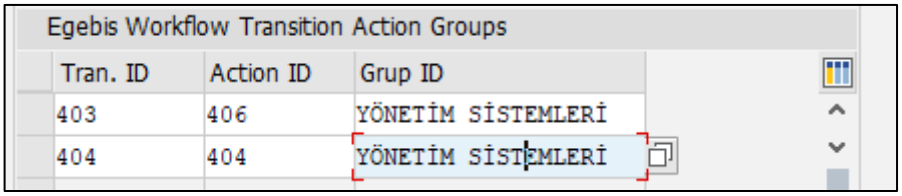
Last of them is the transition 404. It connects the state 574 to the state 572. It is a transition from BOM waiting state to cancel state.


Egebis Workflow Transition

Tran. ID	Procss ID	State ID	State ID	Açıklama
402	C01	570	574	
403	C01	574	573	
404	C01	574	572	

Figure 5.9 Flow definition

As the final step of the flow adjustment by which the action of flow indicated above can be achieved by defining who are authorized adaptations are input as in Figure 5.10.


Egebis Workflow Transition Action Groups

Tran. ID	Action ID	Grup ID
403	406	YÖNETİM SİSTEMLERİ
404	404	YÖNETİM SİSTEMLERİ

Figure 5.10 Authorization setting

The first row creates a mapping between transition 403 action 406 and the user group (YONETIM SISTEMLERI), there is an authorization for the group to act transition 403. This record told us 'Any member of the group (YONETIM SISTEMLERI) may complete the process by adding a link to the document when flow on BOM Waiting state. The visual representation of this C01 flow shown in Figure 5.11.

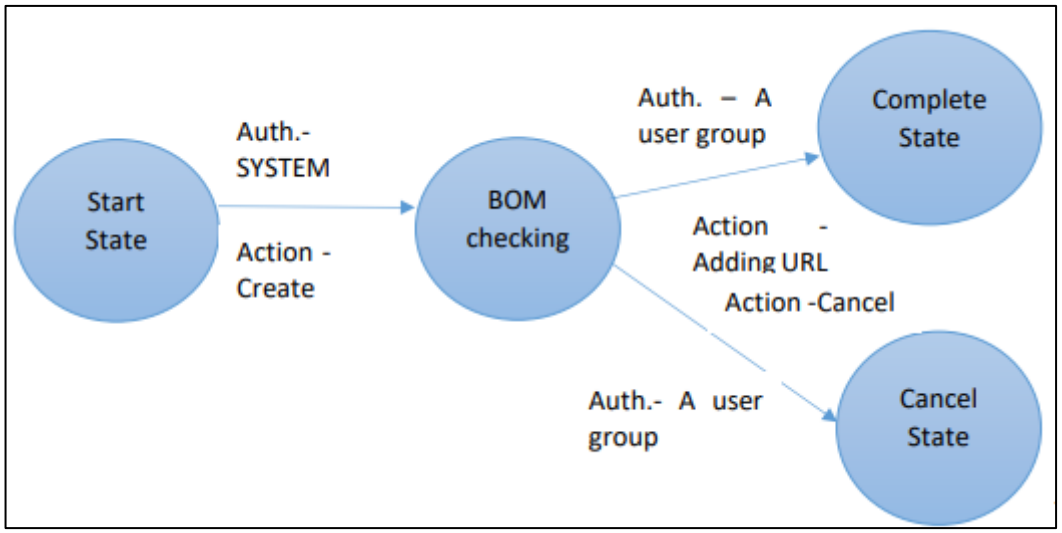


Figure 5.11 Flow diagram

Users perform document requests on a screen that lists possible documents that may be requested. This screen also contains an additional screen informing the user about the general capabilities and possible operations of the system and the icons on this screen. Request screen shown in Figure 5.12.

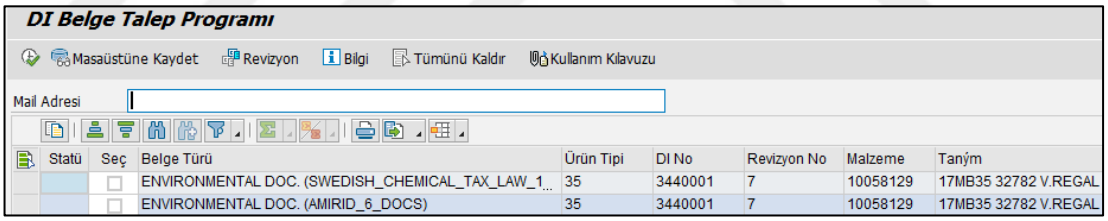
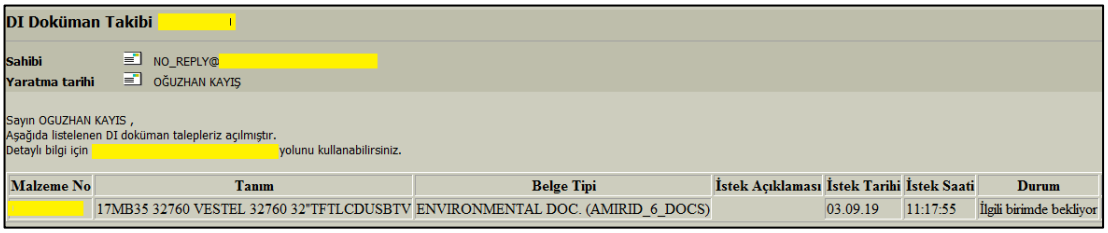


Figure 5.12 Request screen

Using the selection cells in each row, the request is created with the document types are marked after pressing the standard execute button. As soon as the request creation, the request details sent to the related group members and the user who just opened the request, by e-mail. In addition, the requesting individual is informed about the status of the queue on the group basis. The example of the mail form is shown in Figure 5.13 and Figure 5.14.

To avoid unnecessary workload, if there is an open request for a document, additional requests for the same document cannot be created through this screen until

the request is completed. Requests cannot be created for completed documents. However, the revision request can be created with the revision button similarly.



DI Doküman Takibi

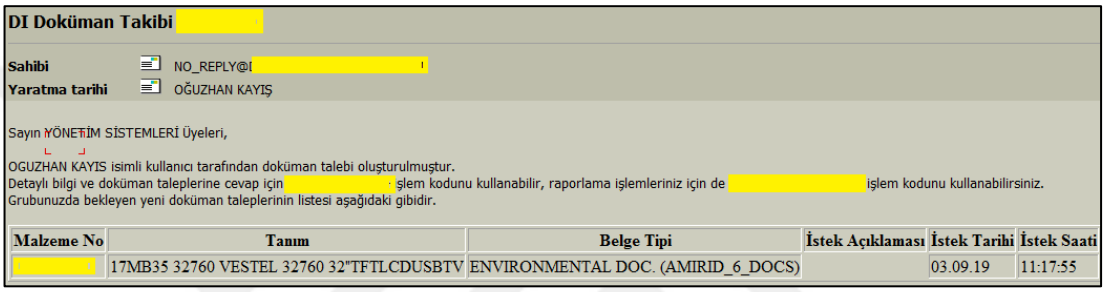
Sahibi NO_REPLY@

Yaratma tarihi OĞUZHAN KAYIŞ

Sayın OGUZHAN KAYIS ,
Aşağıda listelenen DI doküman talepleriz açılmıştır.
Detaylı bilgi için yolunu kullanabilirsiniz.

Malzeme No	Tanım	Belge Tipi	İstek Açıklaması	İstek Tarihi	İstek Saati	Durum
	17MB35 32760 VESTEL 32760 32"TFTLCDUSBTV	ENVIRONMENTAL DOC. (AMIRID_6_DOCS)		03.09.19	11:17:55	İlgili birimde bekliyor

Figure 5.13 Example mail form for the user that created request



DI Doküman Takibi

Sahibi NO_REPLY@

Yaratma tarihi OĞUZHAN KAYIŞ

Sayın YÖNETİM SİSTEMLERİ Üyeleri,

OGUZHAN KAYIS isimli kullanıcı tarafından doküman talebi oluşturulmuştur.
Detaylı bilgi ve doküman taleplerine cevap için işlem kodunu kullanabilir, raporlama işlemleriniz için de işlem kodunu kullanabilirsiniz.
Grubunuzda bekleyen yeni doküman taleplerinin listesi aşağıdaki gibidir.

Malzeme No	Tanım	Belge Tipi	İstek Açıklaması	İstek Tarihi	İstek Saati
	17MB35 32760 VESTEL 32760 32"TFTLCDUSBTV	ENVIRONMENTAL DOC. (AMIRID_6_DOCS)		03.09.19	11:17:55

Figure 5.14 Example mail form for the relevant group members

Additionally, with the "Save to Desktop" option on the same screen, if the requested document is have already prepared, it can save in the system created directory path.

After the request creation, the flow goes to the next step according to the states that it is related to the requested document. The request response screen is shown in Figure 5.15, which may be specialized according to the which actions possible on the current state.

The data on this screen varies from group to group. According to the group that the user who is entering the program is in, the requests listed within the authorization check. For a row to be selected on the sample screen, there are two possible different operations can be performed, which shown in the flow diagram of Figure 5.11.

If the user wishes to add a path the screen from which the user can select the document path is opened. After path operation, the request may be moved to the completed state or it may be rejected.

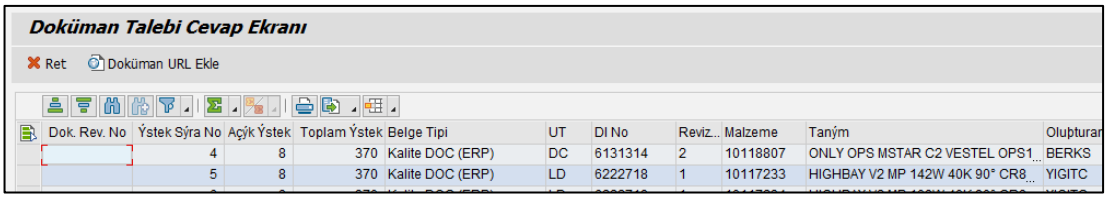


Figure 5.15 Document request response screen

If the request is rejected, a popup is displayed, which is the reason for the user rejection, after which the flow is completed by cancelling. After this process, both the user who opened the request and the user who executes the operation are sent mail for notification.

There is also a report in the workflow system for using of request owner. It can display the status of their own requests, the waiting duration time, related group that responsible for each request, related group members etc. As shown in Figure 5.16, the report has a selection screen with the filter of all status defined on the system.

This screen is also customized and used as a report where groups can query all document operations belonging to their group.

The report can be varied, detailed and changeable according to the institution's needs. In order to meet the needs of the existing system, all information related to the request is available. However, the current situation of the request, the waiting period after the request is opened and the related group for each document exist on the report screen. In addition, the address information on the SAP system of these group members can be accessed on the screen which is opened by double clicking on the related group information.

The reporting screen also includes a 'SUPERUSER' feature. The user with the 'SUPERUSER' authority can report all the flows on the system while the requesting users can display only their own requests, group members can observe requests only in their own group. This allows a group-based performance evaluation if necessary. Any number of ‘SUPERUSER' can be defined in the system, anyone with this authority can be a member of any group at the same time.

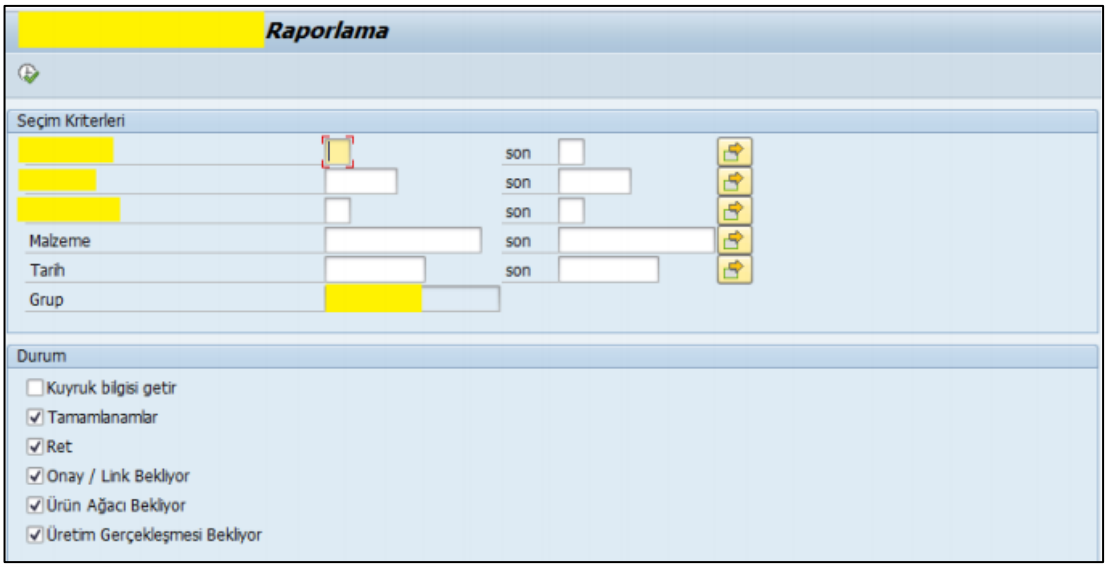


Figure 5.16 Selection screen of user reports

In order to create a flow within the system, it is basically necessary to enter data into the five main maintenance screens. In a flow system such as those that define similar flows for the document type, there is a program that can create flow with reference flow ID, instead of making these definitions from the maintenance screens.

If there are similar documents flow with shown in Figure 5.11, it is possible to create any number of new document flow like that, with given the flow ID C01.

CHAPTER SIX
RULE ENGINE

The rule engine is a structure that can work independently of the workflow engine and allows non-software-based individuals to create conditional rules and modify the process at run time. It contains an interpreter and a semantic control component. This structure, which allows the predetermined variables to change according to the entered conditions, has been developed in a semantic manner like the third level programming languages. It does not include data type definitions because it uses variables that are ready in the background.

Rule engine designed and implemented on document tracking system to improve flexibility. Some of the defined operations may be change by validations. In some cases, a step of the process should be passed, or a document type may be blocked for a time. Chancing the live system is not preferred and it takes time. Instead of using regularly modification, a system which allow modify system with parameters used.

The main feature of a rule engine is that it is designed in an infrastructure that can adapt to changing business processes at runtime.

The rule engine developed with this work allows it to interfere with the tracked object in working time when it is needed by working integrated with the workflow engine.

It consists of expressions within the rule engine and assignments running in connection with expressions. Figure 6.1 illustrates how a statement solves a sample process in a basic notation.

An expression refers to a semantic structure with operands on the right and left of the operant along with an operant.

```
IF ( (
. . ( current_state EQ 'W' ) AND
. .( ( doc_type EQ 'ENVIROMENTAL_DOC1 ) OR
. . . ( doc_type EQ 'ENVIROMENTAL_DOC4 )
. . )
) AND
( user_group EQ 'SAFETY' )
{
// atama işlemi
}
```

Figure 6.1 A sample formula

One operand has been used for correct or incorrect expression of expression by performing various comparison operations depending on the two operators on the right and left. The result of a conditional expression can only result in true or false. If the result of the expression is correct, the assignment operation following the expression is performed.

Operators represent pre-defined variables in system internal or logical true or false expressions that are the result of a nested query. As the operant, the constants whose meanings are given in Table 6.1 are defined.

There are also some definitions to navigate input stream into expression and assignment parts. These special character sets defined as the keywords of the interpreter. Keywords chosen from the most popular programming languages. These keywords and their explanation are given in Table 6.2.

The formula expression given as an example in Figure 6.1 for an exemplary implementation of the rule engine on the document tracking system. "*current_state*", "*user_group*" and "*doc_type*" represent pre-defined variables in the program. Values shown in single quotes (') are constants used in expressions.

Table 6.1 Operand list

Operand	Name	Description
EQ =	Equal	If operant1 and operant2 are equal, it will be true.
NE != <>	Not Equal	Reverse of EQ operation.
GT >	Greater	If operant1 is greater than operant2, it will be true.
GE	Greater or equal	If operant1 is greater or equal to operant2, it will be true.
LT <	Less than	If operant1 is less than operant2 it will be true.
LE	Less or equal	If operant1 is less or equal to operant2, it will be true.
CO	Contains	If operant1 contains characters of operant2, it will be true.
CN	Not contains	Reverse of CO operation.
CA	Contains any	If operant2 includes at least character of operant1, it will be true.
NA	Not contains	Reverse of CA operation.
CS	Contains character	If content of operant2 includes operant1. It will be true.
NS	Not contains character	Reverse of CS operation.
CP	Contain pattern	If pattern includes in characters string, it will be true.
CN	Not contain patterns	Reverse of CP operation.

The formula expression given as an example in Figure 6.1 for an exemplary implementation of the rule engine on the document tracking system. "*current_state*", "*user_group*" and "*doc_type*" represent pre-defined variables in the program. Values shown in single quotes (') are constants used in expressions.

The example formula given as input to the system is converted into a virtual tree structure within the system. In this system, where the depth levels of the brackets in the input determine the depth of the tree, the representation of the above formula in the tree structure is shown in Figure 6.2.

The so-called codes of the algorithm used for the processing of the rules are given below. In the "*rule_process*" function, the input is first divided into conditions

according to keywords. Each condition is processed after the first character set is partitioned, with input to the function "*expression_process*".

Table 6.2 Keyword list

Keywords	Name	Description
AND	And	If both operands are true, result will be true.
OR	Or	If any of the operant is true, result will be true.
//	Comment line	Comment line starts characters.
/	Escape character	It used as an escape character for special cases.
(	Left parenthesis	Starts and expression.
)	Right parenthesis	Ends of expression.
;	Semicolon	End of line character.
IF, ELSEIF	If, elseif	Starts an expression.
ELSE	Else	Starts the last expression.
	Space	Defined as separator between operands and operator.

The "*rule_process*" function takes the condition and condition-dependent assignment as input and sends the character set associated with the condition to the "*expression_solve*" function. If the condition is true, the assignment-related section is sent to the "*assignment_process*" function for assignment. Flowchart of the "*expression_process*" and "*rule_process*" functions is given in Figure 6.3.

The function "*expression_solve*" is the function where the entered condition expression results in true or false. The function that works in recursive format continues to shrink with the resolution of the expression given as input in each iteration. The exit condition is triggered when the input can be set to true or false.

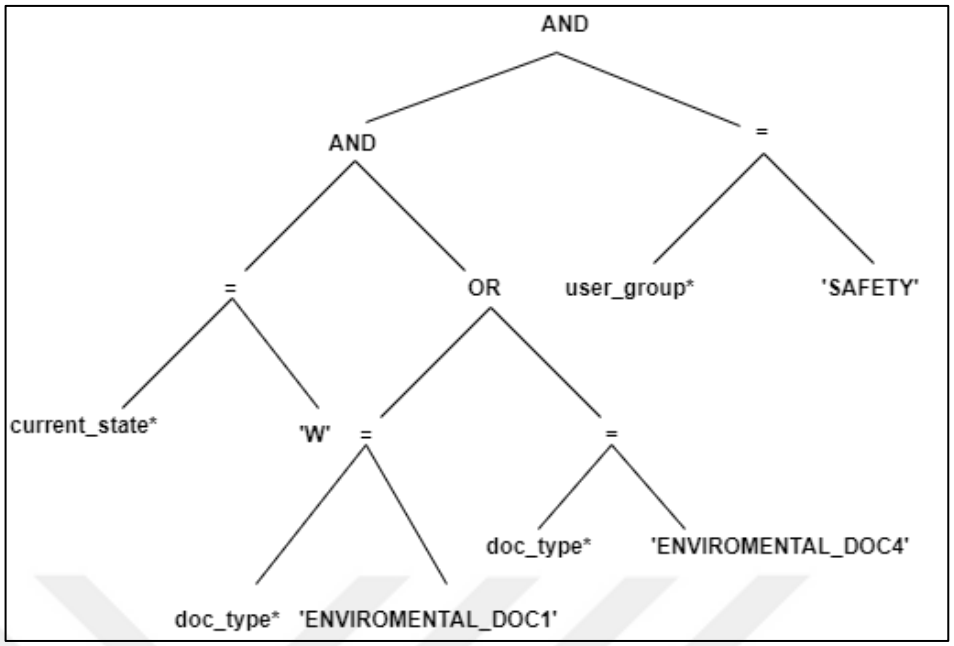


Figure 6.2 A sample tree structure

The function starts by identifying the most nested parentheses from nested condition expressions. In the leaf sections of the tree structure operands of the most innermost condition expressions are placed.

Indicates the atomic deepest expression in the sequence of characters following an open parenthesis if the next parenthesis is an open parenthesis. In the loop, the characters of the input expression are circulated to find atomic expressions. Since the indexes of all atomic expressions are determined in a single loop, the character string between the indexes in that loop is sent to the function "*expression_verify*". This function returns true or false by analysing the expression with the operators given in Table 6.1.

The true or false character returned from the function is replaced by the character set between the specified indexes in the basic input. After this step, the expressions at the lowest level of the virtual tree structure created are solved. Flowchart of the "*expression_solve*" and the "*expression_verify*" is given in Figure 3.27.

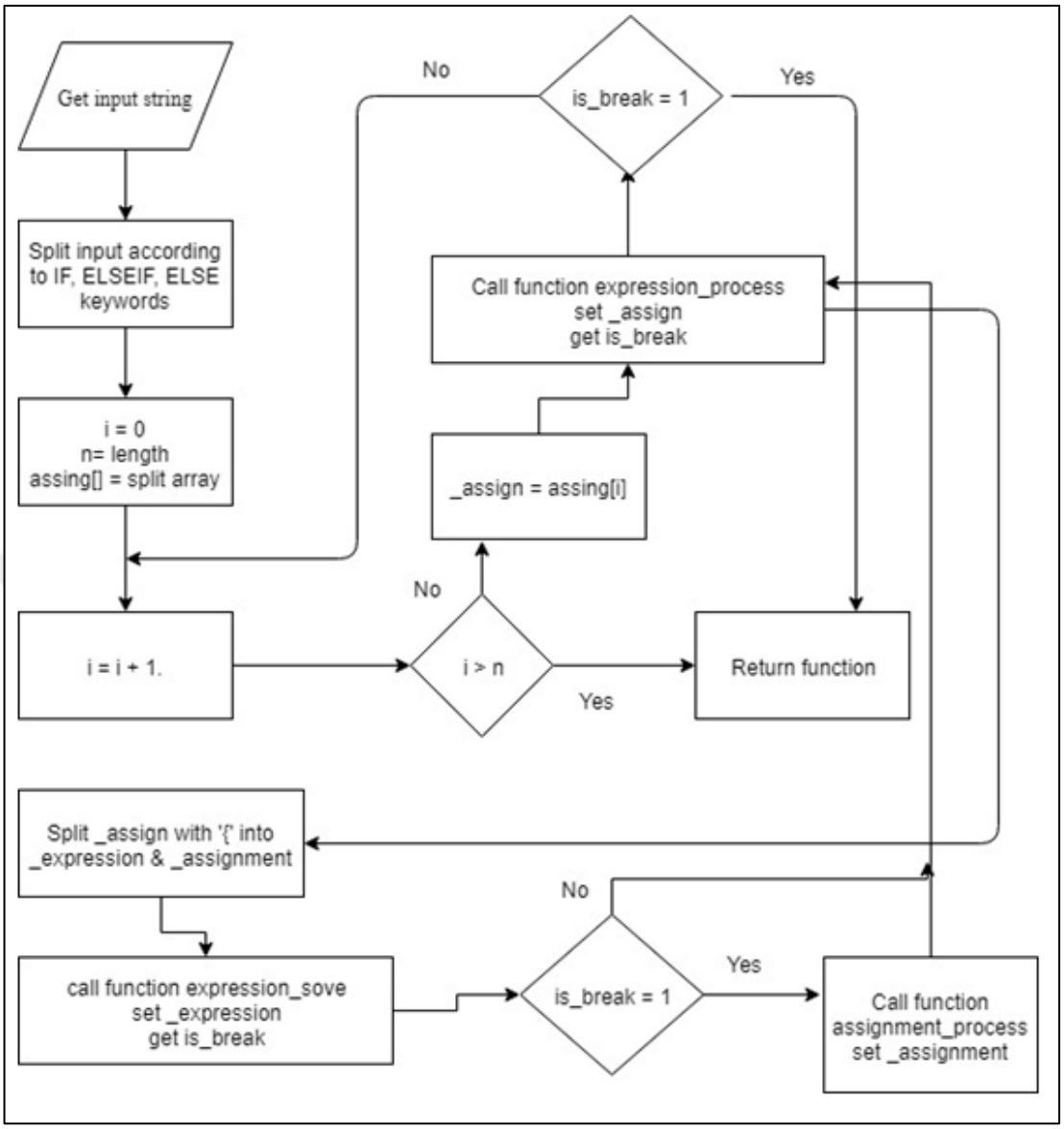


Figure 6.3 Flowchart of rule process algorithm

A screen developed for entering rules. The screen shown in figure 3.28 includes a text area. There are some standard SAP screen elements like copy past & search buttons. Also, there is a syntax checking button. After button click, a function triggers that same logic of expression solving. If expression result cannot calculate system return a syntax error. Comments may be created via screen button or a text may be uncommented.

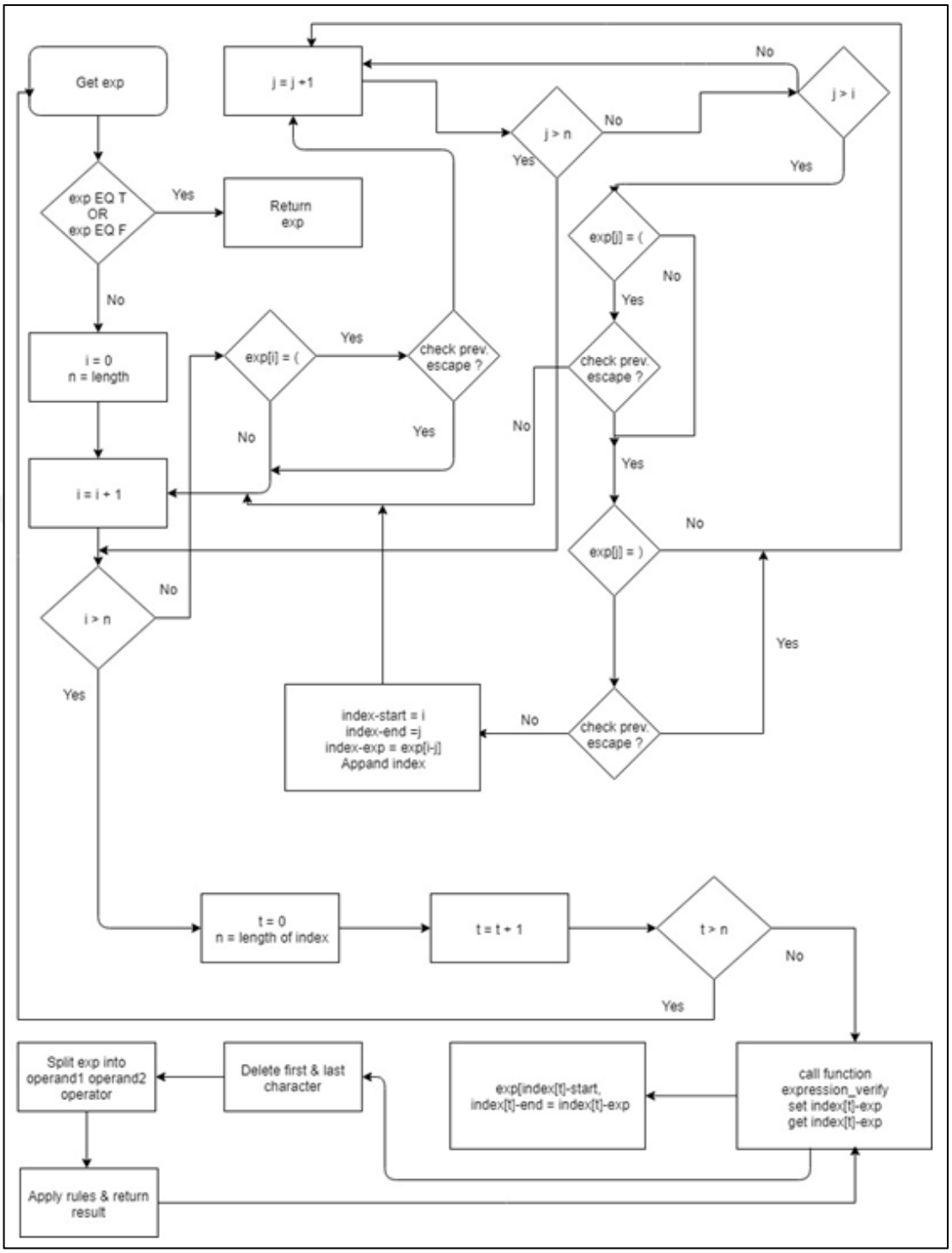


Figure 6.4 Flowchart of expression solving algorithm

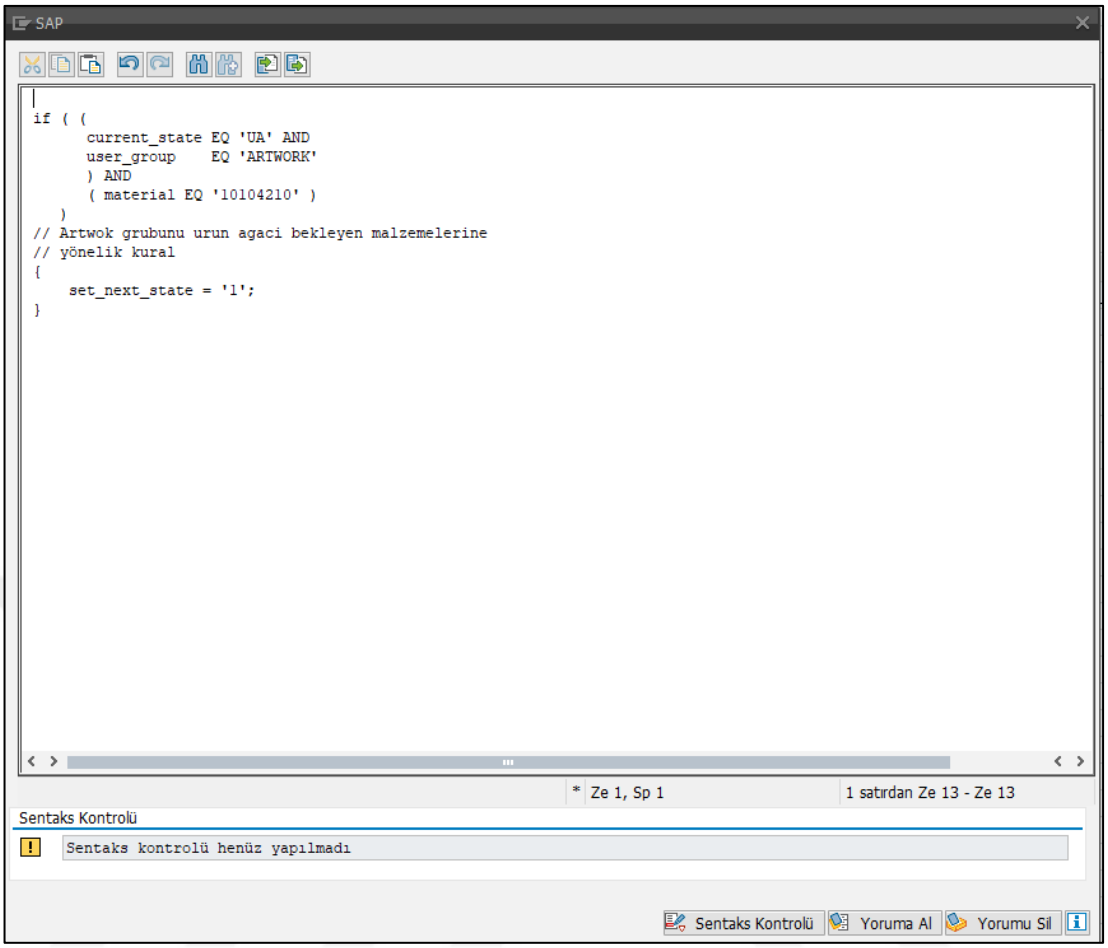


Figure 6.5 Rule maintenance screen

CHAPTER SEVEN
RESULTS AND CONCLUSION

After the study was completed as an application, it was tested on real data in a company. In this way, the process in the company that operates manually, physically by paper or e-mail; has turned into a software system. Although it is seen that flows can be created dynamically in this process, it has been noticed that flow structure and dependencies can affect system performance. Therefore, in order to avoid repeated transaction costs, a sequential pattern mining study was conducted on the data collected in the step of document request processing.

In this study, generalized pattern mining algorithm (GSP) like Apriori method was used. In general, GSP algorithm is not very different from Apriori algorithm; it differs from each other in that it performs processing for the array sequence, not just one element. (Desai & Ganatra, 2014)

Therefore, the data preparation process varies. In the process of providing data ready to be processed from raw data; There are generally two different methods for the use of invalid data in the process of general relational data analysis. In this process, invalid data is never involved, ignored, or invalid data is used as the average of valid data. However, in sequential pattern mining, these two methods were not preferred because sequential frequency patterns were examined, and all unnecessary data were assumed to be meaningful uniform (Zhang, S., Zhang, C., & Yang, 2003).

In this study, WEKA was used as a data mining tool and the data was exported in .txt format with a program with three customized tables. The structure of the tables used is given in Figure 7.1.

In order to create the data set, the tables taken from the live system were downloaded to the local area in txt format and then converted to a format that can be used as input to the WEKA tool on a program capable of file operations. The program is used as input from the data in the tables detailed below.

Request: Stores general information about the requested object in the workflow engine.

Request_Data: It is a request table that differs in the implementation of the workflow engine for each different purpose. The form used for document tracking contains additional fields for the requested material.

Requset_Action: It stores the actions that take place up to a moment on the object being requested. This table is also considered as a log structure.

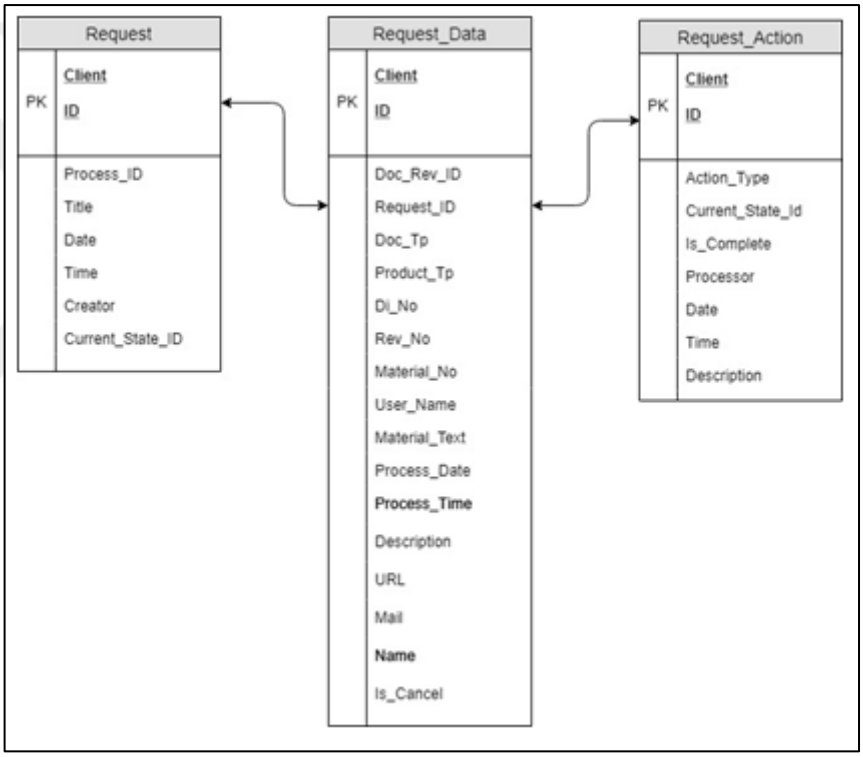


Figure 7.1 Tables used when creating dataset

Labels for document types were first created because long text expressions adversely affect performance in structures where sequential patterns are searched. At this point, it is seen that there is some data that is not valid and not real. It has been assessed that these unrealistic situations have occurred / arose as a result of system installation or some future end user errors. However, ignoring these data, which is a conventional method, was not found suitable for this study. Because a deleted document request was not excluded because it would disrupt a sequential relationship

and possibly cause the longest possible sequential pattern to disappear This process uses a different method to process invalid data A data set was created that can be processed without subtracting all data using a new label for queues that are not valid.

The program that takes the file as input generates a file in the format that can be input to WEKA tool as output. The dataset was created to contain only document labels, date and time.

The date and time are combined and converted into a non-repeated sequence of numbers. In the last case, the data set contains only numbers indicating the sequence and labels describing document names. An example data set is shown in Figure 7.2.

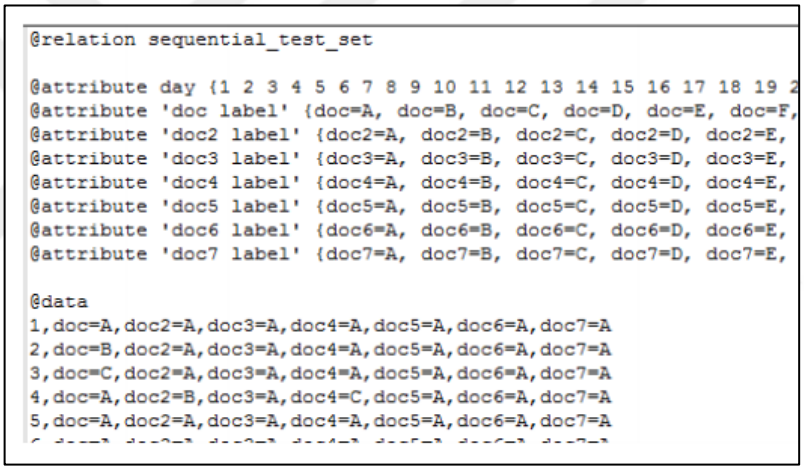


Figure 7.2 Sample view of dataset

Number of requests per document types are given in Table 7.1. Document name field is designed as a long character set. Therefore, before applying the GSP algorithm, a single character label given to each of document names. Totally 1579 requests have been created via system for 861 unique materials. When some of the requests includes only one document type, some of them includes multiple document types. Each request per material defined as a sequence. Sequence length is the number of unique document type in a sequence. Longest sequence length is 7.

Some of the document flow definitions deleted after one month of the go live. These deleted documents are not important for pattern recognition.

Therefore, these deleted document types are defined as noise data. According to ignoring method, another unique label “X” set to the deleted document types.

Table 7.1 Number of requests per document types

Document name	Label	Count
Artwork Doc	A	356
EMC doc	B	388
Environmental doc. (Amirid_6_docs)	C	66
Environmental doc. (Bush (Argos)_11_docs)	D	127
Environmental doc. (Dixons_10_docs)	E	211
Environmental doc. (Funai_6_docs)	F	119
Environmental doc. (Hitachi (Argos)_11_docs)	G	393
Others	X	19
Total		1579

Table 7.2 shows the rates of the sequence length distribution in the sequence set. Although most of the sequences consist of only 1 document type, there are some long sequences.

Table 7.2 includes rows for each length of the sequences. It includes 7 records as item. Total number of document request is 1579. But the total number of the sequence is 861. Because sequence is the collected version of document requests with key of material number.

Most common sequence length is 1. It means generally only 1 document requested by users. Although these tables give us statistical information, it is not enough to determine a pattern of these document types. Therefore, a sequential pattern mining algorithm, GSP, applied to this dataset with giving different minimum support value as parameters.

Table 7.2 Rates of sequences length distribution

Sequence length	Count	Rate
1: 1 unique document	438	50.9
2: 2 unique documents	354	41.1
3: 3 unique documents	38	4.5
4: 4 unique documents	25	2.9
5: 5 unique documents	2	0.2
6: 6 unique documents	3	0.3
7: 7 unique documents	1	0.1
Total	861	100

In the test results where the minimum support value was determined as 0.6, two document label pairs were calculated and in the test results where the minimum support value was determined as 0.9, one document label pair was calculated above the limit. Table 7.3 show the relationship between sequences and minimum support values.

Table 7.3 Sequences with minimum support value

Sequences	Minimum support value
<D, F>	0.9
<D, F>	0.8
<D, F>	0.7
<D, F>, <A, B>	0.6

In this thesis, an activity-based workflow engine is designed with modelling of deterministic finite state automata. This thesis compares the designed system and some popular workflow engine designs by comparing similarities and differences according

to ability of implementing workflow engine patterns. Designed system also shows the advantages of storing finite state automata models in a relational database rather than a matrix.

The biggest contribution of designed system is the developed rule engine. The rule engine enables workflow engine to adapt variable business request in real time. Rule engine includes expression and assignment solve algorithms by using virtual tree structures.

Designed workflow engine implemented into an ERP system of a company for tracking material related document request. In this respect, the study also includes the design and implementation of a document tracking system. With the document tracking system, the process where the users start with the document request and the relevant people prepare the documents and return to the requestors has become traceable.

In addition, a study was conducted to determine the document types that are repeated in a sequence per material. Using the transaction log information generated by the system, a dataset was created, and sequential pattern mining was performed over this dataset. Sequentially repeated document requests were determined and shared with the related business units.

REFERENCES

Agrawal, R., Gunopulos, D., & Leymann, F. (1998). Mining process models from workflow logs. In *International Conference on Extending Database Technology,* 467-483.

Allen, R. (2001). *Workflow: An introduction. workflow handbook.* Florida: Future Strategies.

Bergmann, S. (2007). *Design and implementation of a workflow engine*. M.Sc Thesis, Rheinische Friedrich-Wilhelms University, Bonn.

Chung, S., & Synder, C. (1999). ERP initiation-a historical perspective. *AMCIS 1999 Proceedings,* 76.

De Leusse, P., Kwolek, B., & Zielinski, K. (2012). A common interface for multi-rule-engine distributed systems. *arXiv preprint arXiv:1203.0435*.

De Ley, E., & Jacobs, D. (2011). Rules-based analysis with JBoss Drools: adding intelligence to automation. *Proceedings of ICALEPCS 2011,* 790-793.

Desai, N., & Ganatra, A. (2014). Draw Attention to Potential Customer with the Help of Subjective Measures in Sequential Pattern Mining (SPM) Approach. In *Proceeding of International Conference on Recent Trends in Information, Telecommunication and Computing, ITC.*

Georgakopoulos, D., Hornick, M., & Sheth, A. (1995). An overview of workflow management: From process modeling to workflow automation infrastructure. *Distributed and Parallel Databases, 3*(2), 119-153.

Khawla, B., & Molnár, B. (2018). Dynamic business process: comparative models and workflow patterns. In *The 11th conference of Phd students in computer science*, 41.

Lucas, S. M., & Reynolds, T. J. (2005). Learning deterministic finite automata with a smart state labeling evolutionary algorithm. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (7) 1063-1074.

McCoy, D. W., & Sinur, J. (2006). Achieving Agility: The Agile Power of Business Rules. *Gartner, Special report on Driving Enterprise Agility, 20*.

Molnár, B., & Máriás, Z. (2015). Design and implementation of a workflow oriented ERP system. In *2015 12th International Joint Conference on e-Business and Telecommunications (ICETE) 2*, 160-167. IEEE.

Pirt, B. (2004). *Learning eZ publish 3: Building Content Management Solutions*. Birmingham: Packt Publishing Ltd.

Ran, S., Brebner, P., & Gorton, I. (2001). The rigorous evaluation of Enterprise Java Bean technology. In *Proceedings 15th International Conference on Information Networking*, 93-100. IEEE.

Rosemann, M., & Brocke, J. V. (2015). *Handbook on business process management 1: introduction, methods, and information systems*. Berlin: Springer.

Rusinkiewicz, M., & Sheth, A. (1995). Specification and execution of transactional workflows. In *Modern database systems*, 592-620.

Singh, G. N., & Aggarwal, S. (2011). A Process Model for Workflow Mining. *International Journal of Information Technology and Knowledge Management, 4*(2), 719-722.

Stoilova, K. P., & Stoilov, T. A. (2006). Evolution of the workflow management systems. In *Scientific Conference on Information, Communication and Energy Systems and Technologies-ICEST* , 225-228.

Van Der Aalst, W. M., & Ter Hofstede, A. H. (2005). YAWL: yet another workflow language. *Information Systems, 30*(4), 245-275.

Van der Aalst, W. M., van Dongen, B. F., Herbst, J., Maruster, L., Schimm, G., & Weijters, A. J. (2003). Workflow mining: A survey of issues and approaches. *Data & Knowledge Engineering, 47*(2), 237-267.

Van Der Aalst, W., Van Hee, K. M., & van Hee, K. (2004). *Workflow management: models, methods, and systems*. Cambridge: MIT press.

Xu, X., & Hong, Y. (2012). Matrix expression and reachability analysis of finite automata. *Journal of Control Theory and Applications, 10*(2), 210-215.

Zhang, S., Zhang, C., & Yang, Q. (2003). Data preparation for data mining. *Applied Artificial Intelligence, 17*(5-6), 375-381.