ABSTRACT

PETRI NET MODELING AND PERFORMANCE ANALYSIS OF Z39.50 SEARCH AND RETRIEVAL PROTOCOL

by
Roopali Paranjpe

Z39.50 is an American Standard for information retrieval. The standard depicts a client-server model of computing. With reference to the Open System Interconnection (OSI) model developed by International Standards Organization (ISO); Z39.50 is an *application layer* protocol. A Z39.50 client performs searching, querying and requesting functions and a Z39.50 server interfaces with the database in the remote system and responds to client requests by providing a set of records for the search query.

Z39.50 state tables describe a complex set of concurrent and serial events. A Petri net model is prepared for the entire system comprising the protocol. A Petri net methodology proves to be a very effective way of analyzing this client-server model. A Reachability Graph is applied to the Petri Net model to study the behavioral properties of the protocol. We have used a timed Petri net simulator to evaluate the performance of this protocol.

PETRI NET MODELING AND PERFORMANCE ANALYSIS OF Z39.50 SEARCH AND RETRIEVAL PROTOCOL

by
Roopali Paranjpe

A Thesis
Submitted to the Faculty of
New Jersey Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of
Master of Science in Electrical Engineering

Department of Electrical and Computer Engineering

May 1997

APPROVAL PAGE

PETRI NET MODELING AND PERFORMANCE ANALYSIS OF Z39.50 SEARCH AND RETRIEVAL PROTOCOL

Roopali Paranjpe

Dr. Mengchu Zhou, Thesis Advisor — Date
Associate Professor of Electrical and Computer Engineering, NJIT

Dr. Edwin Hou, Committee Member — Date
Associate Professor of Electrical and Computer Engineering, NJIT

Dr. John Carpinelli, Committee Member — Date
Associate Professor of Electrical and Computer Engineering,
Associate Professor of Computer and Information Sciences, NJIT

BIOGRAPHICAL SKETCH

Author: Roopali Paranjpe

Degree: Master of Science

Date: May 1997

Undergraduate and Graduate Education:

- Master of Science in Electrical Engineering,
New Jersey Institute of Technology, Newark, New Jersey, May 1997

- Bachelor of Engineering in Electronics and Telecommunications,
Government College of Engineering Pune, Pune, India, May 1992

Major: Electrical Engineering

To my family

ACKNOWLEDGMENT

I would like to express my deepest appreciation to Dr. Mengchu Zhou for his valuable guidance and encouragement throughout the thesis work. I would like to thank Dr. Edwin Hou and Dr. John Carpinelli for participating on my thesis committee.

Special thanks are extended to Mr. Paul Over of National Institute of Standards and Technology (NIST) for providing the ZPRISE client - server software. I wish to thank the Computer Science Institute at Brandenburg Technical University Cottbus for providing information on Petri net simulation tools.

Lastly, but not the least, I would also like to thank the Robert W. Van Houten Library staff at NJIT for maintaining an excellent facility.

TABLE OF CONTENTS

TABLE OF CONTENTS
(Continued)

TABLE OF CONTENTS
(Continued)

LIST OF TABLES

LIST OF TABLES
(Continued)

LIST OF TABLES
(Continued)

Table **Page**

LIST OF TABLES
(Continued)

Table **Page**

LIST OF FIGURES

LIST OF FIGURES
(Continued)

Figure **Page**

CHAPTER 1

INTRODUCTION

1.1 Objective

The objective of this thesis is to analyze the performance of ANSI/NISO Z39.50 Search and Retrieval protocol. In order to evaluate the high level protocol such as Z39.50, we have used the following modeling and analysis techniques:

We identified the different facilities provided by Z39.50 protocol standard. We found that there are eleven core services at the heart of this protocol and these services contribute to a very complex set of events and actions. To model these events and actions for further analysis, we have used the Petri net methodology. Using Petri net theory, we have developed a place-transition model for the entire system. To study the behavioral properties of the system, we have applied a Reachability Graph for the Petri net model. Assuming deterministic delays, time analysis is performed and completion times are computed for different successful events and operations. We have used a timed Petri net simulation tool for evaluation of Z39.50 services and operations.

1.2 Overview of Z39.50 Standard

Z39.50 is an American National Standard for information search and retrieval. It is a computer to computer communications standard which defines a set of rules and procedures for database searching and record retrieval.

1.2.1 Evolution of Z39.50

The Z39.50 protocol was originally proposed in 1984 for use with bibliographic information retrieval. National Information Standards Organization (NISO) approved the Z39.50 standard in 1988 (Z39.50-1988). Based on the enhancements proposed by Z39.50 Implementers Group, a second version of the standard was released in 1992 (Z39.50-1992). Z39.50-1992 replaced and superseded Z39.50-1988. A new set of services was introduced in Z39.50-1995 and a third version of the protocol was developed in 1995. Z39.50-1995 is a compatible superset of Z39.50-1992 and the international standard, ISO 10162/10163 [1, 2].

1.2.2 Motivation Behind Z39.50

Libraries are repositories of information. For effective exchange of information, it is only natural to think of a library network spanning the world. Such a network will prove to be an invaluable aid for students, professors and researchers. Z39.50 is one example of such a standard which serves the library, information and publishing communities.

The motivation behind the development of this standard was the numerous problems associated with the multiple database searching. The forms-based interface of the World Wide Web (WWW) along with the graphical clients or browsers is a relatively inexpensive solution for searching online catalog systems. However, there are some serious drawbacks of this system. Since no two information systems share the same interface characteristics, each system requires the user to master a new interface structure and a new set of custom-designed icons and symbols. The advantage of Z39.50 protocol is that it allows different information resources to look and act the same to the individual user.

Hyper Text Transfer Protocol (HTTP) which is at the core of the WWW, presents a stateless protocol model. The server does not have any knowledge of the previous search results. Z39.50 protocol depicts a state based model of information search and retrieval system. Although the existence of state based model for Z39.50 makes its implementation complicated, it is extremely useful for high-quality database searching.

Z39.50 protocol is not only restricted to bibliographic information retrieval but it also supports data, images and multimedia in a distributed network environment. It is based on the client-server architecture of computing and it operates over the Internet. Figure 1.1 shows the model of Z39.50 Information Retrieval (IR) system.

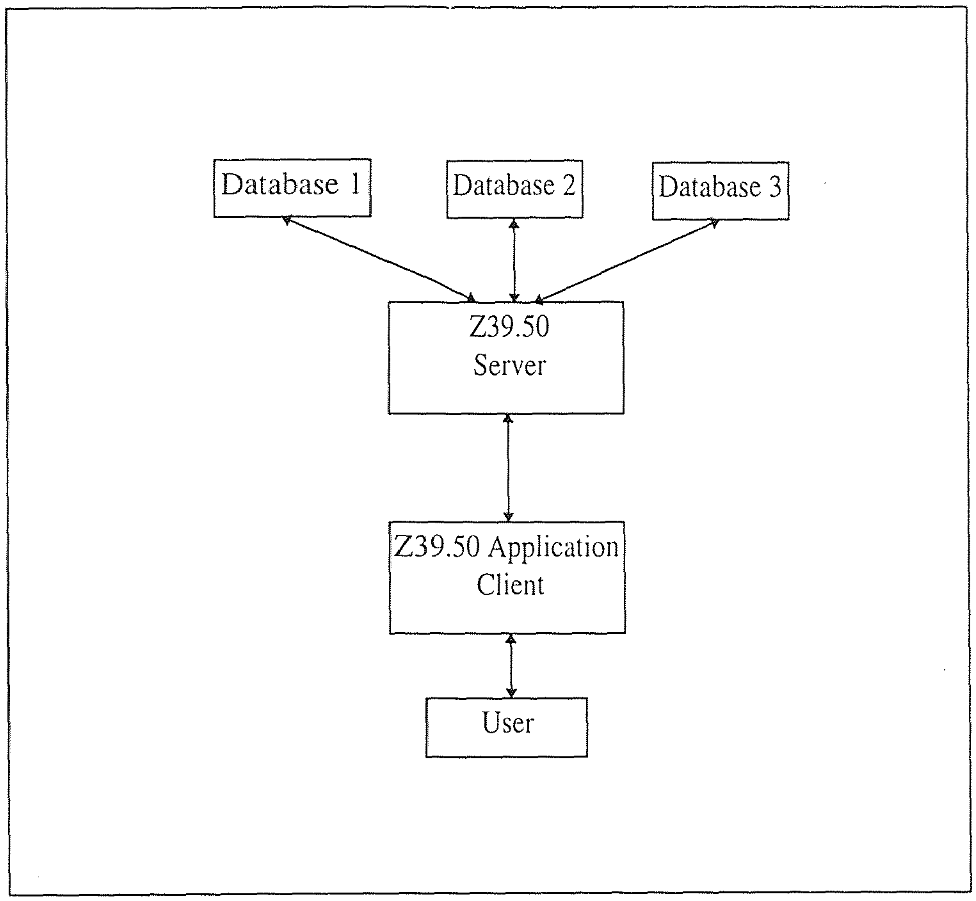


Figure 1.1 Model of Z39.50 Information Retrieval System

CHAPTER 2

Z39.50 - INFORMATION RETRIEVAL SYSTEM

2.1 Model and Characteristics

Z39.50 is an Information Retrieval (IR) system. In this IR system, two processes are involved: A initiating process called the client and a responding process, the server. The server is associated with one or more databases. Communication between the client and the server is carried out by the Z39.50 protocol. The portions of the client and the server that carry out the Z39.50 procedures are referred to respectively as the Z39.50 Origin and the Z39.50 Target.

Z39.50 is a *session oriented* protocol. The two important terminologies regarding the communication session for a Z39.50 protocol are: 1) Z-association and 2) A-association.

1. **Z-association:** The Origin communicates with the Target via a Z39.50 association which is also known as a Z-association. The Origin establishes a Z-association. A Z-association can be terminated by either the Origin or the Target. Within a Z-association, the role of the Origin and the Target is fixed. Once a Z-association is terminated, no status information is retained except information that is explicitly saved. There may be multiple consecutive as well as simultaneous operations within a Z-association.
2. **A-association:** A communication session between a database user and a database provider is known as Application association or A-association. Termination of an A

-association can terminate a Z-association. There may be multiple, consecutive Z-associations within an A-association.

2.1.1 Searching a Database

Z39.50 protocol enables a client on one computer to search and retrieve information from one or more server databases. A query is applied to a database, specifying values to be matched against the access points of the database. An access point is a unique or non-unique key that can be specified in the search of a record [1]. Following the process of search, a result set is made available by the Target to the Origin. This result set may be referenced in a subsequent query and manipulated to form a new result set.

2.1.2 Z39.50 Operations

The Origin can initiate an operation by generating an initiating request of a particular operation type. An operation is terminated upon receiving a respective terminating response from the Target. There are eight different types of operations: Init, Search, Present, Delete, Scan, Sort, Resource Report and Extended Services [1].

2.1.3 Z39.50 Services

Z39.50 services [1] are described as exchange of messages between the Origin and the Target. A message could be a request or a response. There are three types of services defined for the Z39.50 protocol: Confirmed, Non-Confirmed and Conditionally-Confirmed.

Confirmed Service: For a Confirmed service, a request (from the Origin or the Target) is always followed by a response (from the peer).

Non-Confirmed Service: A Non-Confirmed service is defined as a request from the Origin or the Target, with no corresponding response.

Conditionally-Confirmed Service: A Conditionally-Confirmed service may be invoked as either a Confirmed or a Non-Confirmed service. A response may or may not follow for a Conditionally-Confirmed service from the Origin or the Target.

2.2 Facilities of Information Retrieval Service

Information retrieval facilities consist of a single service or groups of services. There are eleven facilities identified for the Z39.50 standard [1].

1. **Initialization Facility (Init Service):** The Init service allows the Origin to establish a Z-association. In the Init request, the Origin proposes values for the initialization parameters. In the Init response, the Target responds with values for the initialization parameters. If the Target responds affirmatively, the Z-association is established.
2. **Search Facility (Search Service):** Search service enables the Origin to query databases at the Target system, and to retrieve information about the results of the query. The Target then creates the result set, which represents a set of records with the properties indicated by the query. The Target maintains this result set for subsequent retrieval requests.

3. **Retrieval Facility:** The retrieval facility consists of two services:

   - **Present Service:** In the Present operation, the Origin sends a Present request to get the response records according to a position within the result set maintained by the Target. The Origin requests a range of records: For example, N records beginning with M where,

     N = Number of records requested and

     M = Result set start position

     (Note: The upper bound on N is (Result count - M)+1)

   - **Segment Service:** This is often useful if the records requested by a Present request do not fit in a single segment.

4. **Result-Set-Delete Facility (Delete Service):** The Delete service enables the Origin to request the Target to delete a specified result set or all the result sets created during a Z-association.

5. **Browse Facility (Scan Service):** This service enables the Origin to initiate a Scan operation.

6. **Sort Facility (Sort Service):** The Sort service enables the Origin to initiate a Sort operation.

7. **Access Control Facility (Access Control Service):** The Access Control service allows the Target to challenge the Origin. An Access Control request issued by the Target could be a part of a specific operation or a Z-association. This mechanism is used to support password and authentication challenges. The details of Access Control service are discussed in a later chapter.

8. **Accounting/Resource Control Facility:** The Accounting/Resource control facility consists of three services:

   - **Resource Control Service:** The Resource Control service is initiated by the Target. It does not initiate an operation. A Resource Control request originated by the Target includes a resource report. A resource report informs the Origin about actual and predicted resource consumption.
   - **Trigger Resource Control Service:** This service is initiated by the Origin during an operation. The Trigger resource Control service permits the Origin to request the Target to initiate the Resource Control service.
   - **Resource-Report Service:** The Resource-Report service is initiated by the Origin to initiate a Resource-Report operation. The Resource Report service permits the Origin to request the target to send a resource report pertaining to a completed operation or to a Z-association.

9. **Explain Facility:** The Explain facility does not include any services. This facility is absent in version 2 (Z39.50-1992). It allows the Origin to obtain the details about the Target. The information about the databases available for searching can be found using the Explain facility.

10. **Extended Services Facility (Extended Services Service):** It enables the Origin to initiate an Extended Services operation.

11. **Termination Facility (Close Service):** It allows the Origin or the Target to abruptly terminate all active operations and to initiate the termination of a Z-association.

Table 2.1 explains the Z39.50 services, their types and whether they are initiated by the Origin or the Target or both.

Table 2.1 Z39.50 Services

Service Name	Service Type	Service Initiated By
Init	Confirmed	Origin
Search	Confirmed	Origin
Present	Confirmed	Origin
Segment	Non-Confirmed	Target
Delete	Confirmed	Origin
Scan	Confirmed	Origin
Sort	Confirmed	Origin
Access Control	Confirmed	Target
Resource Control	Conditionally Confirmed	Target
Trigger Resource Control	Non-Confirmed	Origin
Resource Report	Non-Confirmed	Origin
Extended services	Confirmed	Origin
Close	Confirmed	Origin/Target

CHAPTER 3

PROTOCOL MODEL AND SPECIFICATIONS

3.1 The OSI Reference Model

OSI is an abbreviation for Open Systems Interconnection. The OSI reference model divides the communication process between the two application programs into seven intermediate layers. Each layer provides a certain kind of service to the next higher layer. This service is provided by communicating with the peer entity in the same layer of the remote host using the service provided by the next lower layer. Figure 3.1 describes the seven layers of OSI model [3].

3.2 Service Provided by Presentation Layer

With reference to the OSI model, the Z39.50 protocol is an *application layer* protocol. It uses the presentation layer service to provide a connection between a Z39.50 Origin/Target pair. The communication service that supports this protocol is a *connection-oriented* service. The establishment of an A-association takes place in the following manner:

The life of an A-association has three distinct phases: Connection establishment, information transfer, and termination. Once a connection is established, Z39.50 Application Protocol Data Units (APDU) are transferred between the Origin and the Target. An APDU is a unit of information, transferred between the Origin and the Target.

It consists of application protocol information and application user data.

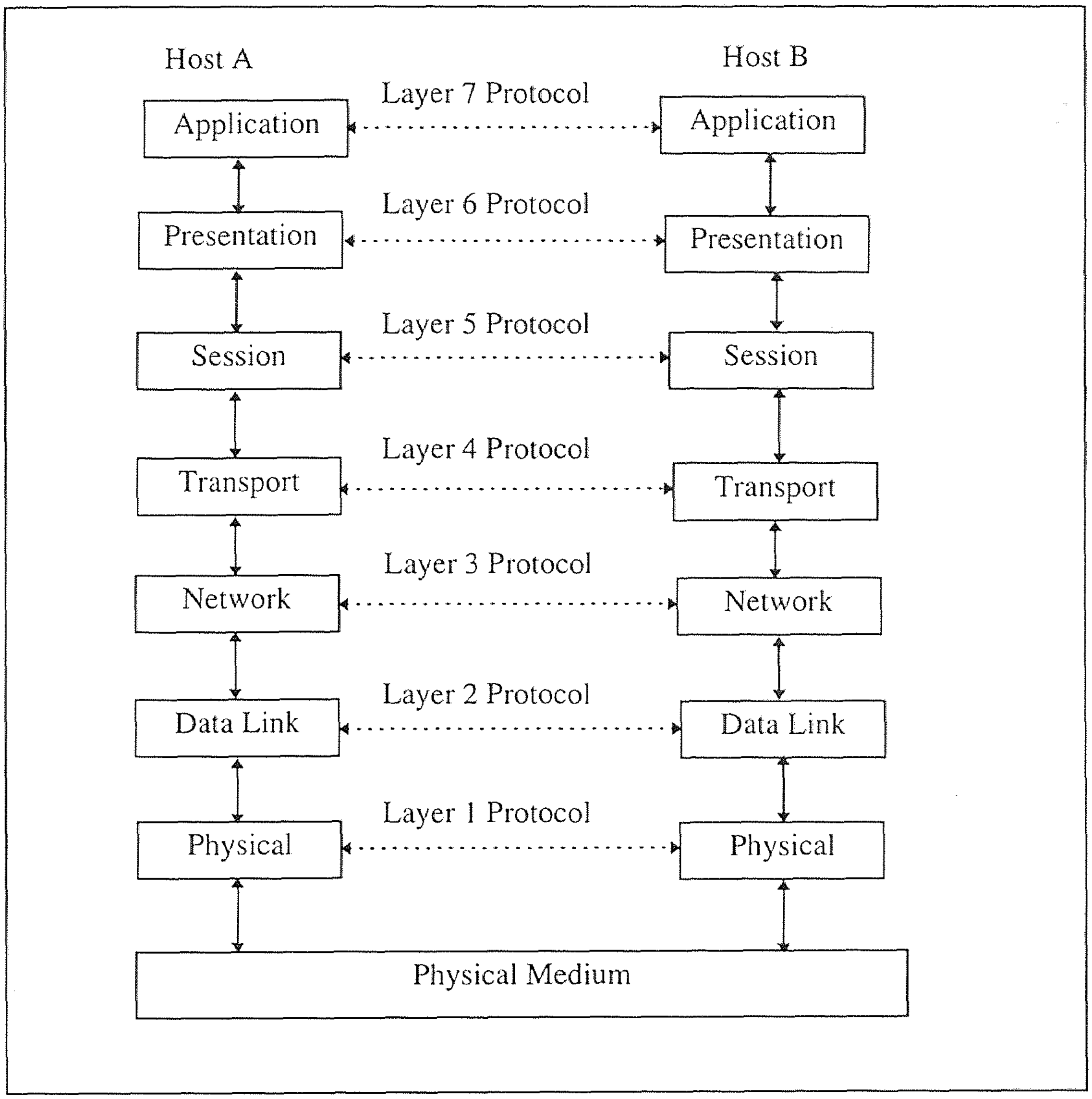


Figure 3.1 Seven Layer OSI Model [3]

3.3 Protocol Procedures

To understand the Z39.50 protocol procedure it is essential to understand an abstract concept of service user, service provider and service primitives [1].

The client is modeled as a service user together with the Origin and the server is modeled as a service user together with the Target. A service provider provides the communication between the two service users. A service primitive is an element of interaction between the service user and the service provider. There are four types of service primitives: Request, Indication, Response and Confirmation.

For a Confirmed service initiated by the Origin, (i.e. Z39.50: Init, Search, Present, Delete, Resource-Report, Sort, Scan, Extended Services) the service primitives are used in the following manner:

Request: The Origin service user sends a request to the Origin service provider in order to invoke some procedure.

Indication: The Target service provider then informs the Target service user about the request.

Response: The Target service user sends a response to the Target service provider at the completion of the procedure invoked by the Indication.

Confirmation: The Origin service provider sends confirmation to the Origin service user to complete the procedure invoked by the request.

For a Non-Confirmed service such as Segment, Trigger Resource Control and Close, only the Request and the Indication primitives are used.

3.4 Sequence of Interactions for a Confirmed Service

For a typical Confirmed service the following sequence of interactions takes place during the connection and transfer of messages.

1. The Origin service user sends a request to the Origin service provider.
2. A Protocol Data Unit (PDU) is transferred between the Origin service provider to the Target service provider.
3. The Target service provider sends an indication to the Target service user.
4. The Target service user sends a response to the Target service provider.
5. A PDU is transferred between the Target service provider and the Origin service provider.
6. The Origin service provider sends a confirmation to the Origin service user.

Figure 3.2 depicts the above mentioned sequence of interactions.

(Note: 1. For a Confirmed service initiated by the Target, the roles of Origin and Target are reversed.

2. For a Non-Confirmed service, only steps 1 through 3 apply.)

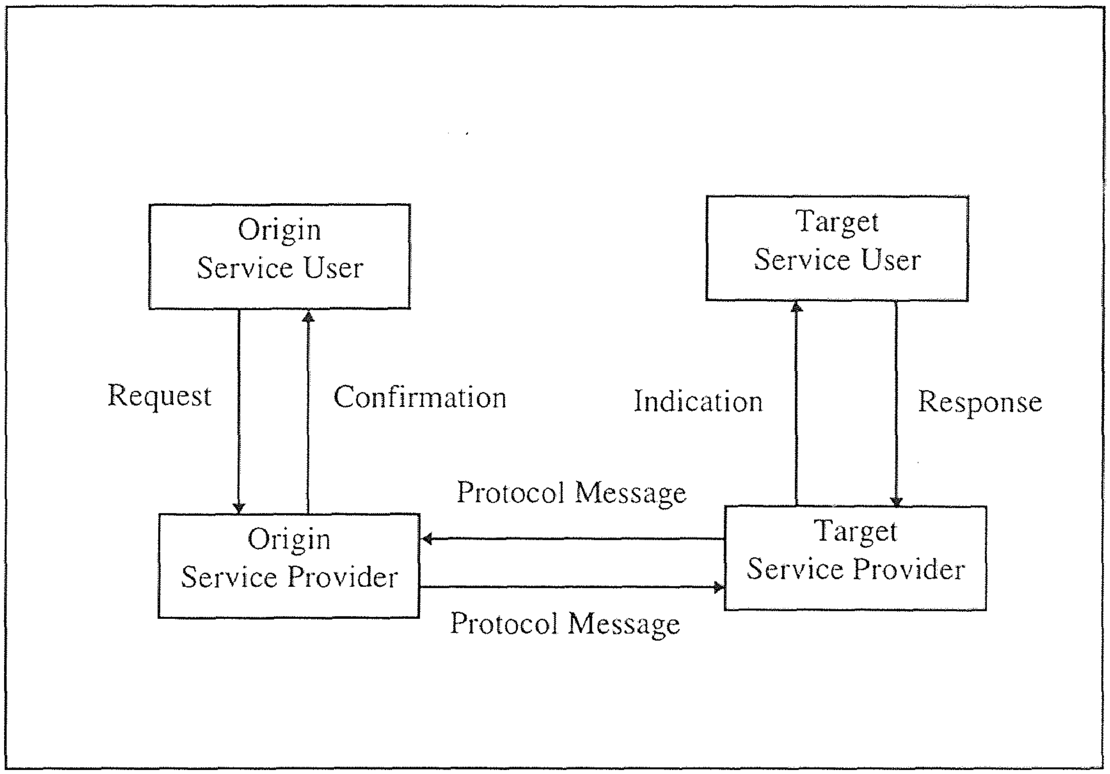


Figure 3.2 Z39.50 Sequence of Interactions

3.5 State Tables [1]

Z39.50 standard describes the state tables for the Origin as well as the Target. The state tables present the relationship between the state of an operation or a Z-association and the incoming events that occur in the protocol. The intersection of an incoming event (row) and a state (column) is called a state table cell. The state table shows the action to be taken by Origin or Target in a particular state upon receiving an incoming event. The state table cells also show the resulting state. One or more actions are separated by a semicolon (;). The last action in the sequence represents the resulting state and is always indicated in parenthesis.

There are some blank cells in the state tables. These blank cells indicate an invalid operation. A blank cell is a combination of an incoming event and a state which is not defined in the protocol model.

A question then arises as to what action should be taken in case of such an incorrect operation? Unfortunately, Z39.50 standard provides no information about the actions to be taken upon incorrect operations. The remedies for the incorrect operations are beyond the scope of this standard. However, implementers must consider the action to be taken in case of incorrect operations.

3.5.1 Predicates and Variables

Before proceeding with the discussion regarding the interpretation of state tables, it is essential to note the different predicates and variables [1] used in the state tables. The actions described in the state tables depend on certain conditions. These conditions are called predicates. In state table cells, the notation for these actions may occur in one of the following forms:

:[predicate] actions:

or

:[predicate] actions else actions:

Actions are either single action or multiple actions separated by semicolons. Table 3.1 describes predicates and their meaning.

Table 3.1 State Table Predicates and their Meaning

Predicate	**Meaning**
resp	*Response Required* on a Resource Control PDU
noResp	*No Response Required* on a Resource Control PDU
conc	*Concurrent Operations* in effect
noOps	*No Active Operations*

There are certain variables used in the state tables which describe the type of operation, number of operations and the final state of the Origin or the Target after the transactions. Table 3.2 represents all such state table variables and their meaning.

Table 3.2 State Table Variables and their Meaning

Variable	**Meaning**
<op>	Operation type other than Init (Search, Present, Delete, Scan, Sort, Resource-Report, Extended Services)
opCnt	Number of active operations
retSt	Return state. Go to the state whose value is retSt

3.5.2 State Table Interpretation

Appendix A describes the state definitions. Every event in the state table transforms the Origin and the Target to a new state and appropriate messages are sent to the network. In order to help understand the effect of these events on the Origin and the Target states, we have discussed Table 3.3 in detail.

Table 3.3 shows the various states of the Origin during a Z-association initialization phase. It also describes the different protocol messages transferred to the network as a result of the state table events. In the state table, every state is associated with a number between 0 and 11. For example, 0 represents *Closed* state of the Origin. The interpretation of each row of Table 3.3 is as follows:

Row 1: It can be seen from Table 2.1 that the Init service is initiated by the Origin and it is a Confirmed service. The intersection of the *Init req* sent by the Origin user and the Origin *Closed* state, is the state table cell (InitPDU;1). This cell shows that the *Init PDU* protocol message is sent to the network and the Origin state changes from *Closed* (0) to *Init sent* (1). Once a request is sent, the Origin in the *Init sent* state awaiting a response from the Target.

(Note: State tables show the effect of different events on the Origin and the Target states but they do not describe the sequence in which different events may occur. For example, while the Origin is in the Init sent state, an Access Control request protocol message (Acc PDU) or a Resource Control request protocol message can occur before Initialization response (Init resp PDU+ or Init resp PDU-) from the Target.)

Row 2: If there is no *Acc PDU* or *Rsc PDU*, then depending upon the positive or negative response from the Target the Origin state changes. Row 2 of Table 3.3 shows that upon receiving the *Init resp PDU+* message from the network (i.e. a positive response from the Target), a confirmation message *(Init conf+)* is sent to the Origin user and a counter for active operations is initialized. At this time, if the Origin and the Target decide upon the Concurrent operations mode, then the Origin state changes from *Init sent* (1) to *Concurrent Idle* (5) otherwise it changes to *Serial Idle* (4).

Row 3: If the Target sends a negative response (*Init resp PDU-*) then the Origin returns to the original (*Closed*) state and *Init conf-* message informs the Origin user about the initialization failure.

Row 4: If there is an *Acc PDU* event, *Acc ind* protocol message is sent to the Origin user and the Origin state changes from *Init sent* (1) to *Acc recvd* (2).

Row 5: In the *Acc recvd* state, the Origin awaits for the response from the Origin user. Upon receiving a response (*Acc resp*), a response PDU message (*Acc resp PDU*) is sent to the network and the Origin returns to the *Init sent* (1) state.

Row 6: If the Origin receives a *Rsc PDU* in the *Init sent* state, then a *Rsc ind* protocol message is sent to the Origin user. As mentioned earlier, the Resource Control service is a Conditionally Confirmed service. If the Target indicates in the *Rsc PDU* that a response is required from the Origin then the Origin state changes to *Rsc recvd* (3). If the Target does not want a response then the Origin remains in the *Init sent* (1) state.

Row 7: In the *Rsc recvd* state, the Origin receives a response (*Rsc resp*) from the Origin user. A message (*Rsc resp PDU*) is sent to the network and the Origin returns to *Init sent* (1) state.

Tables 3.4 - 3.5 describe the processing and termination phases for the Origin Z-association respectively and can be interpreted in a similar manner. Table 3.6 represents the Present operation for the Origin and Table 3.7 shows all the operations other than Present (i.e. Search, Delete, Resource-Report, Scan and Sort). The Present operation consists of a Present request followed by one or more Segmentation requests. Hence a separate state table is described for the Present operation in order to distinguish it from other operations. A similar set of states and events is described in the Target state tables.

Tables 3.8 - 3.10 are for the initialization, processing and termination phases of the Target Z-association respectively. Table 3.11 represents the Present operation whereas Table 3.12 describes all operations other than Present for the Target.

Table 3.3 State Table for Origin Z39.50 Association: Initialization Phase

State Event	Closed 0	Init sent 1	Acc recvd 2	Rsc recvd 3
Init req	Init PDU;(1)			
Init resp PDU+		Init conf+;set opCnt=0;:[conc] (5) else (4)		
Init resp PDU-		Init conf-;(0)		
Acc PDU		Acc ind;(2)		
Acc resp			Acc resp PDU; (1)	
Rsc PDU		Rsc ind;:[resp](3) else (1)		
Rsc resp				Rsc resp PDU;(1)

Table 3.4 State Table for Origin Z39.50 Association: Processing Phase

State / Event	Serial Idle 4	Concurrent Idle 5	Serial Active 6	Concurrent Active 7	Z-Acc recvd 8	Z-Rsc Recvd 9
<op> req	Initiate <op> operation; (6)	Initiate <op> operation; (7)		Initiate <op> operation; (6)	Initiate <op> operation; set RetSt =7; (8)	Initiate <op> operation; set RetSt =7; (9)
EndOp ind			(4)	Decr; :[noOps](5) else (7);	Decr; :[noOps] set Retst = 5; (8)	Decr; :[noOps] set Retst = 5; (9)
Z-Acc PDU		Acc ind;set RetSt = 5; (8)		Acc ind;set RetSt = 7; (8)		
Z-Acc resp					Acc resp PDU; (RetSt)	
Z-Rsc PDU		Rsc ind; :[resp] set RetSt = 5; (9) else (5)		Rsc ind; :[resp] set RetSt = 7; (9) else (7)		
Z-Rsc resp						Rsc Resp PDU; (RetSt)
Close req	Close PDU; (10)	Close PDU; (10)	Close PDU; KillOps; (10)	Close PDU; KillOps; (10)	Close PDU; KillOps; (10)	Close PDU; KillOps; (10)
Close PDU	Close ind; (11)	Close ind; (11)	Close ind; KillOps; (11)	Close ind; KillOps; (11)	Close ind; KillOps; (11)	Close ind; KillOps; (11)

Table 3.5 State Table for Origin Z39.50 Association: Termination Phase

State Event	Closed sent 10	Closed Recvd 11
AnyOp PDU	(10)	
Z-Rsc PDU	:[noResp] Rsc ind; (10)	
Z-Acc PDU	(10)	
Close resp		Close PDU; (0)
Close PDU	Close conf; (0)	

Table 3.6 State Table for Origin Present Operation

State Event	Present sent 1	Rsc recvd 2	Acc recvd 3
Rsc PDU	Rsc ind;:[resp](2) else (1)		
Rsc resp		Rsc resp PDU; (1)	
Acc PDU	Acc ind; (3)		
Acc resp			Acc resp PDU; (1)
Trigrc req	Trigrc PDU; (1)		
Seg PDU	Seg ind; (1)		
Prsnt resp PDU	Prsnt conf; EndOp ind; exit		

Table 3.7 State Table for Origin Operation other than Present

State Event	**<op> sent** **1**	**Rsc recvd** **2**	**Acc recvd** **3**
Rsc PDU	Rsc ind;:[resp](2) else (1)		
Rsc resp		Rsc resp PDU; (1)	
Acc PDU	Acc ind; (3)		
Acc resp			Acc resp PDU; (1)
Trigrc req	Trigrc PDU; (1)		
<op> resp PDU	<op> conf; EndOp ind; exit		

Table 3.8 State Table for Target Z39.50 Association: Initialization Phase

State Event	**Closed** **0**	**Init Recvd** **1**	**Acc Sent** **2**	**Rsc Sent** **3**
Init PDU	Init ind;(1)			
Init resp+		Init resp PDU+; set opCnt = 0;:[conc] 5 else (4)		
Init resp-		Init resp PDU-;(0)		
Acc req		Acc PDU;(2)		
Acc resp PDU			Acc conf;(1)	
Rsc req		Rsc PDU;:[resp](3) else (1)		
Rsc resp PDU				Rsc conf;(1)

Table 3.9 State Table for Target Z39.50 Association: Processing Phase

State Event	Serial Idle 4	Concurrent Idle 5	Serial Active 6	Concurrent Active 7	Z-Acc sent 8	Z-Rsc sent 9
<op> PDU	Initiate <op> operation; (6)	Initiate <op> operation; (7)		Initiate <op> operation; (6)	Initiate <op> operation; set RetSt =7; (8)	Initiate <op> operation; set RetSt =7; (9)
EndOp ind			(4)	Decr; :[noOps](5) else (7);	Decr; :[noOps] set Retst = 5; (8)	Decr; :[noOps] set Retst = 5; (9)
Z-Acc req		Acc PDU; set RetSt = 5; (8)		Acc PDU; set RetSt = 7; (8)		
Z-Acc PDU					Acc conf; (RetSt)	
Z-Rsc req		Rsc PDU; :[resp] set RetSt = 5; (9) else (5)		Rsc PDU; :[resp] set RetSt = 7; (9) else (7)		
Z-Rsc resp PDU						Rsc conf (RetSt)
Close req	Close PDU; (10)	Close PDU; (10)	Close PDU; KillOps; (10)	Close PDU; KillOps; (10)	Close PDU; KillOps; (10)	Close PDU; KillOps; (10)
Close PDU	Close ind; (11)	Close ind; (11)	Close ind; KillOps; (11)	Close ind; KillOps; (11)	Close ind; KillOps; (11)	Close ind; KillOps; (11)

Table 3.10 State Table for Target Z39.50 Association: Termination Phase

State Event	Closed sent 10	Closed Recvd 11
AnyOp PDU	(10)	
Z-Rsc req noResp		Rsc PDU; (10)
Close resp		Close PDU; (0)
Close PDU	Close conf; (0)	

Table 3.11 State Table for Target Present Operation

State Event	Present recvd 1	Rsc sent 2	Acc sent 3
Rsc req	Rsc PDU;:[resp](2) else (1)		
Rsc resp PDU		Rsc conf; (1)	
Acc req	Acc PDU; (3)		
Acc resp PDU			Acc conf; (1)
Trigrc PDU	Trigrc ind; (1)		
Seg req	Seg PDU; (1)		
Prsnt resp	Prsnt resp PDU; EndOp ind; exit		

Table 3.12 State Table for Target Operation other than Present

State Event	**Present recvd** **1**	**Rsc sent** **2**	**Acc sent** **3**
Rsc req	Rsc PDU;:[resp](2) else (1)		
Rsc resp PDU		Rsc conf; (1)	
Acc req	Acc PDU; (3)		
Acc resp PDU			Acc conf; (1)
Trigrc PDU	Trigrc ind; (1)		
<op> resp	<op> resp PDU; EndOp ind; exit		

3.6 State Table Analysis

From the state tables, it is possible to identify the different occasions when the Target issues an Access Control request or a Resource Control request. It is also possible to determine the effect of a Close request issued by the Origin or the Target during a Z-association and operation. The state tables also illustrate the set of actions to be taken in case of Serial operations as well as Concurrent operations. In the following sections we will review how the Origin generates Serial as well as Concurrent operation requests. We will also discuss in detail the two outstanding services offered by this protocol: Access Control service and Resource Control service.

3.6.1 Serial and Concurrent Operations

A typical operation consists of the initiating request, the terminating response, along with any intermediate Access Control and Resource Control requests and responses, Trigger Resource Control requests and Segment requests. The Origin assigns a reference identification number (reference-id) to every operation and it may initiate multiple Concurrent operations, each identified by a different reference-id. For example, the Origin may initiate a Search operation with some reference-id, and it subsequently initiates another Search operation (with different reference-id) before receiving response for the first operation. In this case, there are two Concurrent operations in effect. When the Origin and the Target are in the Concurrent operations mode, there cannot be two operations with the same reference-ids.

Serial operations are consecutive operations. In case of Serial operations, the Origin initiates the second operation only after receiving response for the first one. When the Origin and the Target are in the Serial operations mode, the consecutive operations can have the same reference-id.

3.6.2 Access Control Service

The Target issues an Access Control request about a Z-association or a particular operation. In Concurrent operations mode, the Target issues an Access Control request after initialization. There may or may not be any active operation at this point. This Access Control request issued by the Target is pertaining to a Z-association. At this time, the Target might suspend processing some or all the operations till it receives a positive response from the Origin. If the response is acceptable to the Target, the suspended operations will proceed in a normal way. If the Origin fails to respond correctly, the Target might terminate one or more operations and the Target may close the Z-association.

If the Target sends the Access Control request during an operation, then it suspends processing that operation till it receives a positive response from the Origin. Upon receiving an affirmative response, the Target resumes processing the operation in the usual fashion. If the response from the Origin is negative then the Target terminates that operation.

3.6.3 Resource Control Service

The Target issues a Resource Control request pertaining to a specific operation or a Z-association.

In Concurrent operations mode, a reference-id in the Resource Control request indicates that the request is pertaining to an operation. The reference-id identifies an active operation. If a reference-id is not present in the Resource Control request then that request is pertaining to a Z-association.

In Serial operations mode, the Resource Control request must include the reference-id and the Resource Control request is pertaining to an operation. The Resource Control request indicates whether a response is required or not. If so, the Origin must issue a Resource Control response. Upon receiving the response, the Target issues a terminating response. After processing the terminating response, an operation is concluded. If a response is not required, then the Origin must not issue a Resource Control response. The Target subsequently issues the terminating response and operation is concluded.

CHAPTER 4

PETRI NET MODEL FOR Z39.50 PROTOCOL

4.1 Petri Net Theory

Petri net theory was originally developed by Carl Adam Petri and presented in his doctoral dissertation in 1962. Petri nets (PNs) are a graphical tool for the formal description of systems whose dynamics are characterized by concurrency, synchronization, mutual exclusion and conflict, which are typical features of a distributed environment. PNs can be fruitfully applied in a variety of fields, ranging from distributed computing to flexible manufacturing, communication protocols, control systems, transportation, banking, and organization of work [4].

4.1.1 Components of a Petri Net Model

A PN consists of Places (P) and Transitions (T) which are linked to each other by Arcs (A) [5].

Places: Places are used to describe possible local system state. Graphically, places are represented by circles as shown in Table 4.1.

Transitions: Rectangular boxes or bars represent transitions. Transitions are the active system components. They are used to describe events that may modify the system state.

For each transition, there is set of input arcs and a set of output arcs. A PN is referred to as a four-tuple (P, T, I, O). where

I: $(P \times T) \rightarrow N$ is an input function that defines directed arcs from places to transitions where N is a set of non-negative integers.

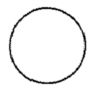
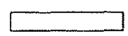
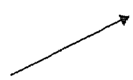
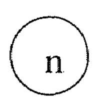
O : $(P \times T) \rightarrow N$ is an output function which defines directed arcs from transitions to places [6].

Arcs : Arcs specify the relation between the local states and events in two ways: They indicate the local state in which the event can occur, and the local state transformations induced by the event.

Tokens : Tokens reside in places, and are used to specify the PN state (also called the PN marking). According to PN theory, a transition is enabled if there is at least one token in each of its input place.

Table 4.1 illustrates the graphical representation of the above mentioned Petri net components.

Table 4.1 Graphical Representation of Petri Net Components

Petri Net Component	Description
	Place
	Immediate Transition (no firing time)
	Timed Transition
	Token
	Arc Illustrating Input and Output Mapping
	n = Number of Tokens Associated With A Place

4.1.2 Types of Transitions and Arcs

A transition can be immediate or timed in terms of their delays and sequential, concurrent, or in conflict with respect to others. When a token is located in a place, the place is said to be enabled. When a token flows from one place to another, the process is referred to as 'firing'.

Immediate transitions fire as soon as they are enabled. Immediate transitions are represented by bars. In a timed transition, there is a delay between enabling and firing. Timed transitions are represented by rectangular boxes and can be either random (stochastic) or deterministic.

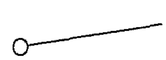
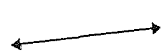
In case of a conflict, the enabled transitions may not fire simultaneously since they share the same input place. When two or more transitions fire simultaneously, they are called concurrent transitions.

The relations between the places 'P' and the transitions 'T' are represented by arcs. Some extensions of arcs include inhibitor arc and self-loop arc.

Inhibitor Arc: A transition associated with the inhibitor arc will not fire if the place to which it links contains a token. Graphically, an inhibitor arc ends with a small circle at a transition. They are generally used as a safety feature to prevent execution of a particular task or action.

Self-Loop Arc: A self-loop arc puts the token back to the same place after firing. Table 4.2 illustrates the graphical representation of inhibitor and self-loop arc.

Table 4.2 PN Arcs Types and Description

Arc Type	**Description**
	Inhibitor Arc
	Self-Loop Arc

4.1.3 Timed Petri Nets

An ordinary Petri net model does not include the time factor, hence the transitions fire instantaneously. This is not the situation in reality. Operations take a finite amount of time to complete. When time is added to the firing of the transitions, it becomes possible to calculate important properties of a system such as cycle times, delay and throughput. Two types of delays can be associated with the PN model: 1) Stochastic delay and 2) Deterministic delay [6].

Stochastic Delay: Stochastic PNs associate an exponentially distributed random variable to the firing time of each transition. More general stochastic PNs allow generally distributed random variables.

Deterministic Delay: A Deterministic PN has a fixed firing time associated with each transition. Time may be associated with places. A token deposited in a place becomes available after a certain time delay. Time can also be associated with the transition. The tokens are removed from the input places immediately when the transition is enabled, but are held for a fixed period of time before they are added to the output places.

4.2 Modeling Z39.50 with Petri Nets

To evaluate the complex nature of Z39.50 protocol, it is essential to model different services and operations for the Origin and the Target. A sequence of events can be understood in a better fashion from the graphical representation of the model. Using the state tables [1], we developed Petri nets for initialization phase, processing phase and termination phase for Z39.50 Origin as well as Z39.50 Target. We have also modeled various operations during a typical Z-association for the Origin and the Target.

The states such as *Close*, *Init Sent* and events such as *Init req* from the state table form the places of the Petri net model. When a certain event takes place, the Origin or the Target changes its state to a new state and corresponding signals are sent to the Origin user or Target user or to the network. Following sections describe PN model for the entire system.

4.2.1 Z-association: Initialization Phase

Figure 4.1 shows the behavior of the Origin and the Target during Z-association initialization phase. Initially both the Origin and the Target are in a *Closed* state represented by places p2 and p14 respectively. An initialization request (*Init req*) is indicated by a token in place p1. The firing of transition t1 indicates the incoming initialization request. Tokens in p2 and p3 fire transition t2 and tokens are deposited in the output places p4 (*Init sent*) and p13 (*Init req PDU)*. At this point transition t7 fires and the Target state changes to *Init recvd* (a tokn in p15) and *Init ind* signal is sent to the Target user. A self-loop arc connecting the place p11 and transition t9 indicates that a token is always present in place p11 (i.e., a response from the Target). Depending upon the nature of the Target response, transition t10 or t11 fires and the system changes its state to *Closed* or *Serial* Idle(token in p8 and p19) or *Concurrent Idle* (token in p9 and p20). This completes the Origin and the Target Initialization process. Tables 4.3 - 4.6 present detail description of different states of Origin and Target and various signals sent during the process. The different requests that might be issued by the Target during the initialization process are discussed in the later sections.

4.2.2 Z-association: Processing Phase

Once a *session* is established, the Origin is ready to receive different operation requests and the Target is ready to send the corresponding responses. The Origin might receive different requests regarding authentication and resource consumption. The Origin can receive a Close request at any time after the initialization process is over. We have modeled the Access Control, Resource Control and Close requests separately. At any time the Origin or the Target can close Z-association by sending a Close request (*Close req*). The Target may close the connection if there are any authentication problems during the processing phase. Generally, the Origin terminates the *session* upon completion of all the operations.

The Origin and the Target decide upon the mode of operation during the initialization process. Depending upon the mode of operation, two different scenarios can be observed. 1) Behavior of Origin and Target during Concurrent operations and 2) Behavior of Origin and Target during Serial operations. In Figure 4.1, places p8 and p19 represent the Origin and the Target respectively in the Concurrent operations mode and the places p9 and p20 show the Serial operations behavior.

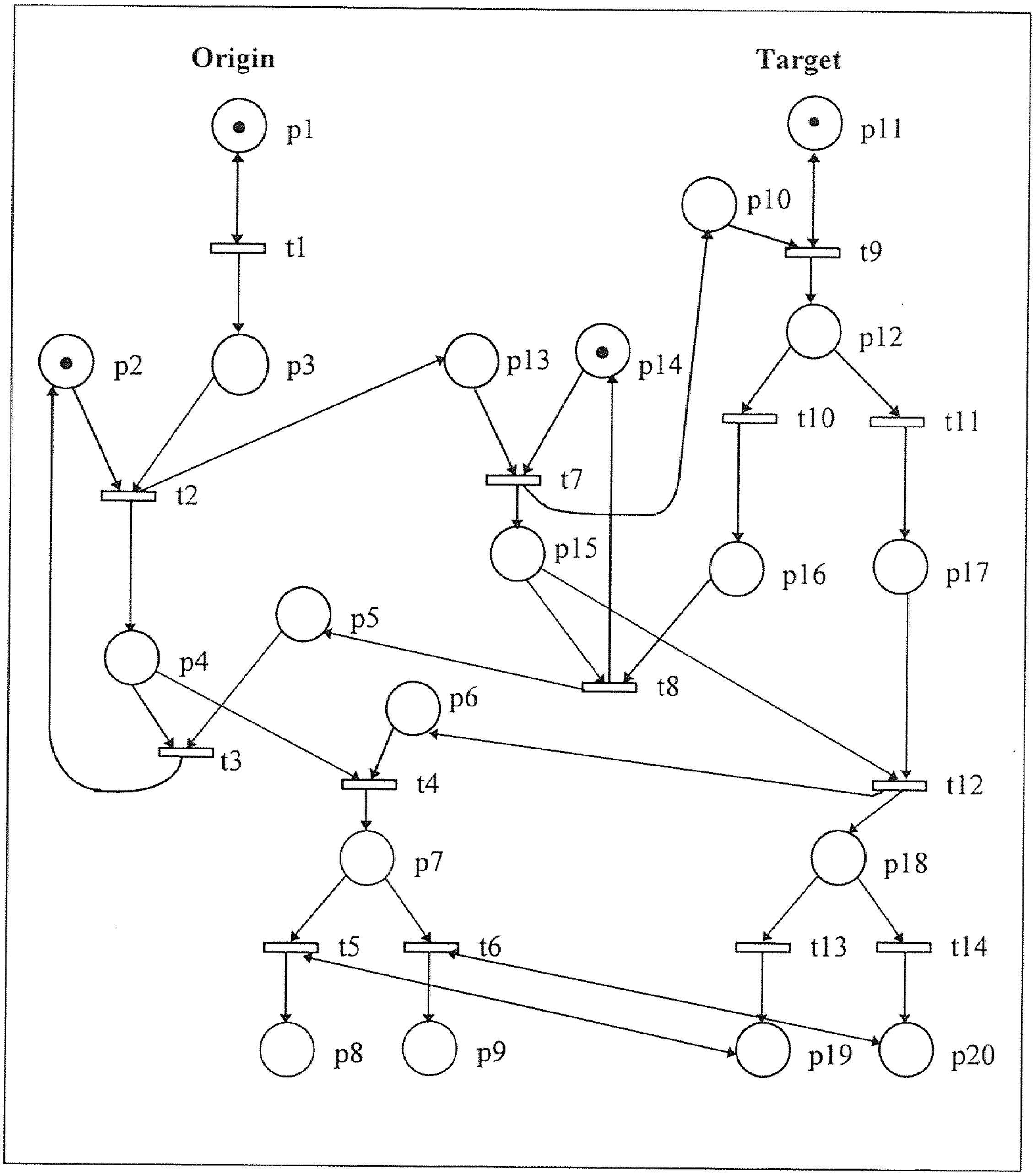


Figure 4.1 Initialization Phase

Table 4.3 Initialization Phase - Origin Place Description

Place	Description
p1	Init Req Ready
p2	Closed (Origin)
p3	Init Req
p4	Init Sent
p5	Init Resp PDU-
p6	Init Resp PDU+
p7	Concurrent/Serial Decision
p8	Serial Idle
p9	Concurrent Idle

Table 4.4 Initialization Phase - Target Place Description

Place	Description
p10	Init ind
p11	Init Resp Ready
p12	Init Resp
p13	Init PDU
p14	Closed (Target)
p15	Init Recvd
p16	Init Resp-
p17	Init Resp+
p18	Concurrent/Serial Decision
p19	Serial Idle
p20	Concurrent Idle

Table 4.5 Initialization Phase - Origin Transition Description

Transition	Description
t1	Init Req sent
t2	Init PDU sent
t3	Init Conf- sent
t4	Init Conf+ sent
t5	Serial Idle sent
t6	Concurrent Idle sent
t7	Init Ind sent

Table 4.6 Initialization Phase - Target Transition Description

Transition	Description
t8	Init Resp PDU- sent
t9	Init Resp sent
t10	Init Resp- sent
t11	Init Resp+ sent
t12	Init Resp PDU+ sent
t13	Serial Idle sent
t14	Concurrent Idle sent

Concurrent Operations Mode: Figure 4.2 shows the PN model for processing phase when the Origin and the Target are in Concurrent operations mode. In the Concurrent operations mode, the Origin and the Target go to *Concurrent Idle* state after initialization (places p9 and p20 respectively). After receiving an operation request, transitions t15 and t16 fire, the Origin becomes *Concurrent Active* (p23) and an operation is initiated. A protocol message regarding the operation (*Op PDU)* indicated by place p25 is sent to the Target. At this point the transition t23 fires and the Target indicates the Target user about the operation and becomes *Concurrent Active* (p27). In the operation response (*Op resp*) the Target also sends a signal indicating end of that operation (*EndOp ind*). The *EndOp ind* (for the Target) and *Op resp* are represented by p31 and p32 respectively.

In order to keep track of the number of operation requests, a counter is kept in the PN model. This counter is called an operation counter (place p24 for the Origin operation counter and place p28 for the Target operation counter). Once the Origin and Target are in *Concurrent Active* state, every new operation request increments the operation counter. After receiving the *EndOp ind* signal, this counter is decremented. If no more operations are in progress, then the Origin and the Target return to initial states, i.e. *Concurrent Idle*. If there are one or more operations in progress, then the Origin and the Target return to *Concurrent Active* state. Tables 4.7 - 4.10 show various places and transitions and their role in a Concurrent operation mode. To indicate the multiple operation requests, three tokens are shown in place p23.

Serial Operations Mode: When there are Serial operations in progress, the Origin and the Target are in *Serial Idle* (p8 and p19 respectively) state. In this situation, requests are processed in a consecutive fashion. The Origin becomes *Serial Active* (p34) after receiving the operation request. The Target also changes its state after sending the operation indication (*op ind*). After receiving the operation response, an *EndOp ind* signal is sent and the Origin and Target go back to the original state (p8 and p19 respectively) and are ready to receive subsequent operation requests. Figure 4.3 describes Serial operations during a Z-association processing phase. Tables 4.11 - 4.14 present the detail description of the places and transitions for the Origin and the Target.

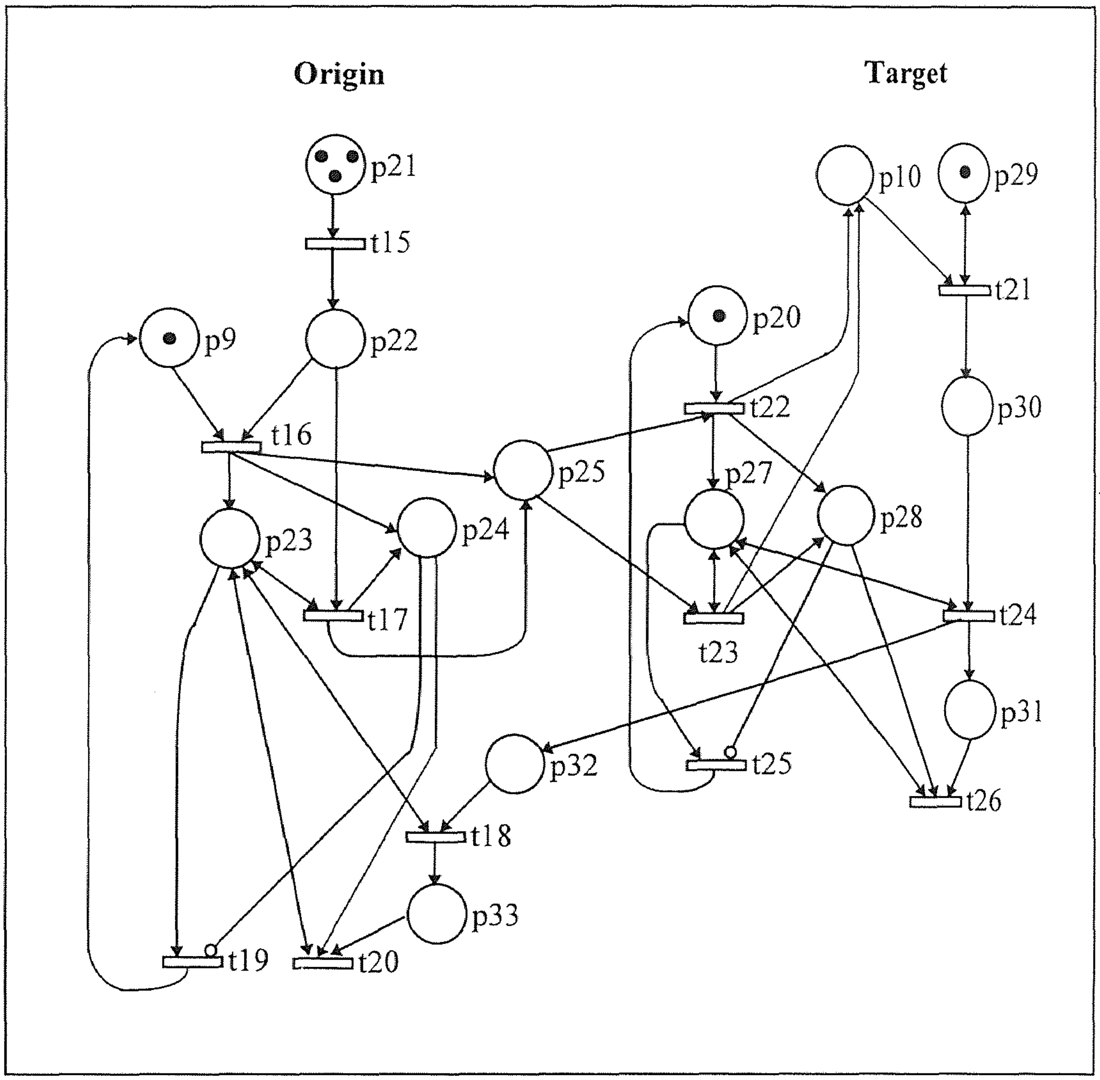


Figure 4.2 Processing Phase - Concurrent Operations

Table 4.7 Concurrent Operations - Origin Place Description

Place	Description
p9	Concurrent Idle
p21	Op Request Ready
p22	Op Req
p23	Concurrent Active
p24	Operation Counter
p25	Op PDU
p33	EndOp ind

Table 4.8 Concurrent Operations - Target Place Description

Place	Description
p20	Concurrent Idle
p26	Concurrent Active
p27	Operation Counter
p28	Op ind
p29	Op Response PDU
p30	Op resp
p31	EndOp ind
p32	Op Resp PDU

Table 4.9 Concurrent Operations - Origin Transition Description

Transition	Description
t15	Op Req sent
t16	Op PDU sent
t17	Op PDU sent
t18	EndOp ind sent
t19	No Operations (go to Concurrent Idle)
t20	Active Operations (go to Concurrent Active)

Table 4.10 Concurrent Operations - Target Transition Description

Transition	Description
t21	Op Resp sent
t22	Op ind sent
t23	Op ind sent
t24	Op Resp PDU sent
t25	No Operations (Go To Concurrent Idle)
t26	Active Operations (Go To Concurrent Active)

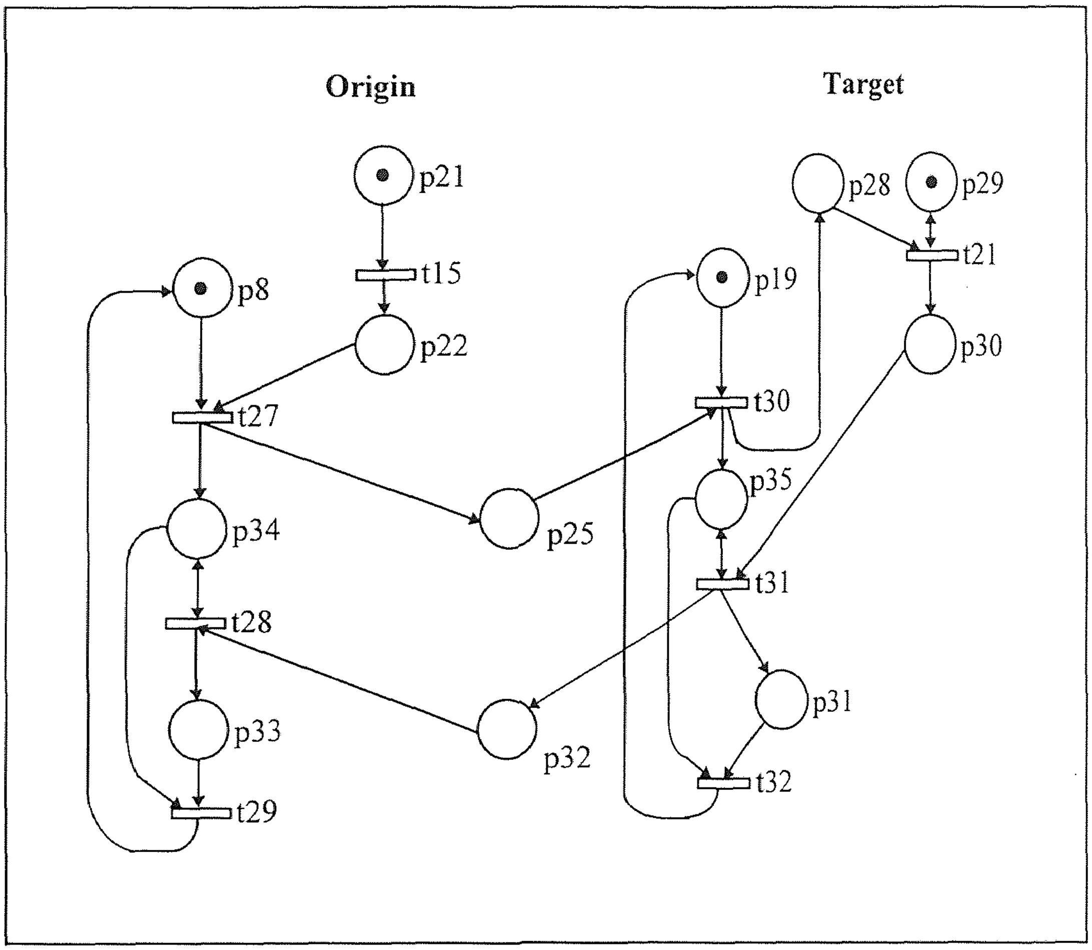


Figure 4.3 Processing Phase - Serial Operations

Table 4.11 Serial Operations - Origin Place Description

Place	Description
p8	Serial Idle
p21	Op Req Ready
p22	Op Req
p34	Serial Active
p33	EndOp ind
p25	Op PDU

Table 4.12 Serial Operations - Target Place Description

Place	Description
p19	Serial Idle
p28	Op ind
p29	Op Resp Ready
p30	Op Resp
p35	Serial Active
p31	EndOp ind
p32	Op Resp PDU

Table 4.13 Serial Operations - Origin Transition Description

Transition	Description
t15	Op Req sent
t27	Op PDU sent
t28	EndOp ind sent
t29	Go To Serial Idle state

Table 4.14 Serial Operations - Target Transition Description

Transition	Description
t21	Op Resp sent
t30	Op ind sent
t31	Op Response PDU sent
t32	Go To Serial Idle

4.2.3 Z-association: Termination Phase

A Close request from the Origin and the Target are shows in the figure 4.4 and figure 4.5 respectively. When Origin user sends a Close request (*Close req*), a *Close PDU* is sent to the Target. The Target informs the Target user by sending Close indication (*Close ind*). Target user sends a Close response (*Close resp*). Upon receiving this *Close PDU*, a Close confirmation (*Close conf*) is sent to the Origin user and termination process completes. A Close request from the Origin and the Target are represented as follows:

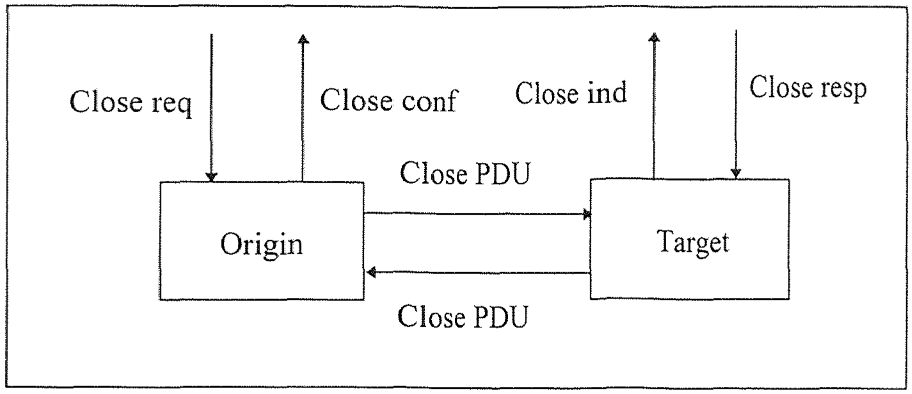


Figure 4.4 Close Request from Origin User

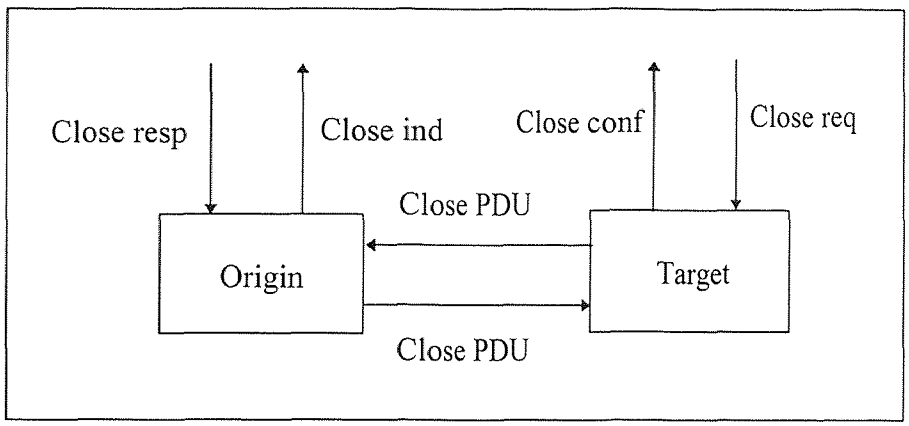


Figure 4.5 Close Request from Target User

Figure 4.6 is a PN model for the Termination process initiated by the Origin and Tables 4.15 - 4.18 describe the places and transitions in this PN model. A Close request can come at any time after the initialization process. Here we have shown a Close request encountered when the Origin and the Target are in a *Serial Idle* mode (places p8 and p19 respectively). Place p36 shows the Close request initiated by the Origin and place p42 shows the Close response from the Target. After completion of the Termination process, the Origin and the Target return to *Closed* state (places p2 and p14 respectively).

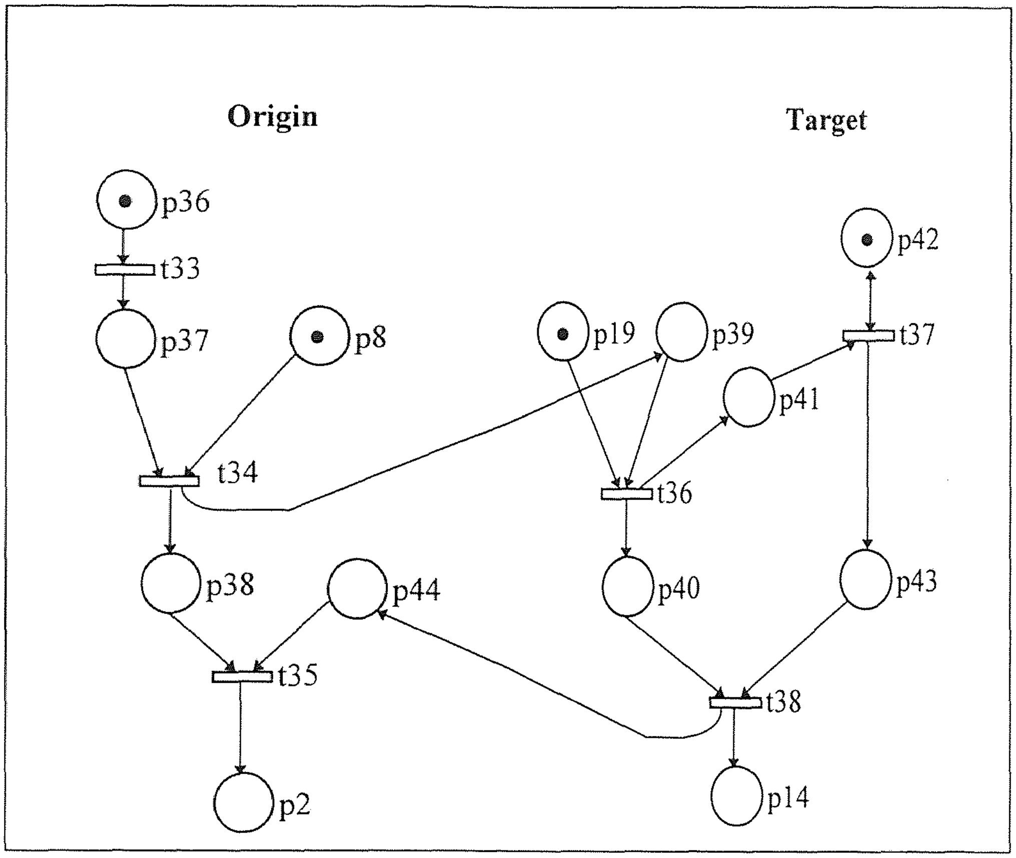


Figure 4.6 Termination Phase - Close Request from Origin

Table 4.15 Close Request from Origin - Origin Place Description

Place	Description
p36	Close Request Ready
p37	Close Req
p8	Serial Idle
p38	Close Sent
p44	Close Resp PDU
p2	Closed (Origin)

Table 4.16 Close Request from Origin - Target Transition Description

Place	Description
p19	Serial Idle
p39	Close PDU
p40	Close Recvd
p41	Close Ind
p42	Close Resp Ready
p43	Close Resp
p14	Closed (Target)

Table 4.17 Close Request from Origin - Origin Transition Description

Transition	Description
t33	Close Req sent
t34	Close PDU sent
t35	Close Conf sent

Table 4.18 Close Request from Origin - Target Transition Description

Transition	Description
t36	Close Ind sent
t37	Close Resp sent
t38	Close Resp PDU sent

The Target can also send a Close request. In this case, the roles of the Origin and the Target are reversed. Figure 4.7 shows the PN model for the termination process where the Target initiates the request (p36). In this case, a response comes from the Origin (p42). At the end of the termination process, the Origin and the Target return to their initial states indicated by p2 and p14 respectively. Tables 4.19 - 4.22 explain the places and transitions in the PN model.

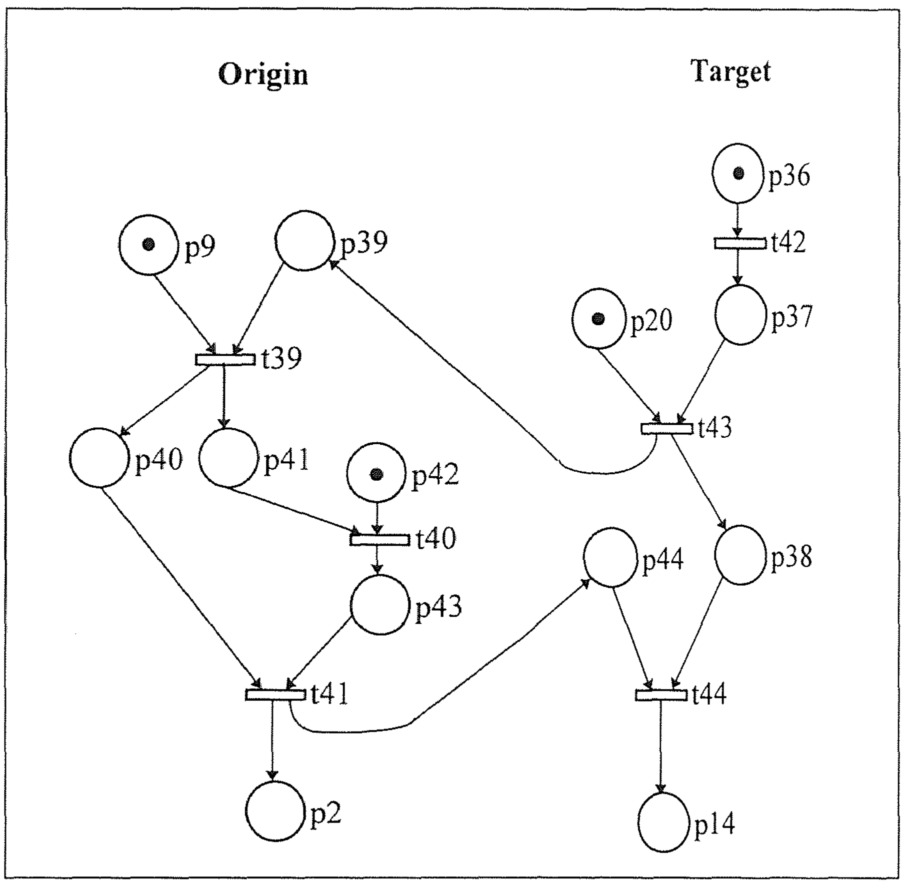


Figure 4.7 Termination Process - Close Request from Target

Table 4.19 Close Request From Target - Origin Place Description

Place	**Description**
p9	Serial Idle
p39	Close PDU
p40	Close Recvd
p41	Close Ind
p42	Close Resp Ready
p43	Close Resp
p2	Closed (Origin)

Table 4.20 Close Request from Target - Target Transition Description

Place	**Description**
p20	Serial Idle
p36	Close Req Ready
p37	Close Req
p38	Close Sent
p44	Close Resp PDU
p14	Closed (Target)

Table 4.21 Close Request from Target - Origin Transition Description

Transition	**Description**
t39	Close Ind sent
t40	Close Resp sent
t41	Close Resp PDU

Table 4.22 Close Request from Target - Target Transition Description

Transition	Description
t42	Close Req sent
t43	Close PDU sent
t44	Close Conf sent

4.2.4 Access Control Request: During Initialization Phase

Access Control events and actions and states are distinguished according to an operation or Z-association [1]. Target issues an Access Control request during an initialization phase or processing phase of a Z-association or operation.

During the initialization phase, the Origin can receive an Access Control request when it is in the *Init sent* (p4) state. The Target user sends a request and changes its state to *Acc sent* (p53). Upon receiving an Access Control request PDU (*Acc PDU*), the Origin state becomes *Acc recvd* (p46) and an indication is sent to the Origin user. Since Access Control service is a Confirmed service, the Origin user must provide a response (*Acc resp*). After sending the *Acc resp* signal, the Origin returns to the *Init sent* state (p4) and after sending the confirmation signal (*Acc conf*), the Target returns to *Init Recvd* (p15) state. Figure 4.8 shows the PN model for an Access Control request during initialization process and Tables 4.23 - 4.26 present a detailed description of PN model components.

4.2.5 Access Control Request: During Processing Phase

In a Z-association processing phase, all the abbreviations for states, events and actions for Access Control operations begin with "Z". (e.g. "*Z-Acc PDU*") [1]. The Origin receives an Access Control request when it is in either *Concurrent Idle* (p9) or *Concurrent Active* (p23) state. The Target changes its state to *Z-Acc sent* (p53) upon receiving the Access Control request (*Z-Acc req*). At this point the *Z-Acc PDU* (p45) is sent to the network and the Origin state changes to *Z-Acc recvd* (p46). There can be multiple operation requests when the Target and Origin are in Z-Acc sent and Z-Acc recvd states respectively.

Figure 4.9 shows how the operation counters for the Origin and the Target are updated after each operation request and response. These operation counters are indicated by the places p24 and p27 respectively. As long as there are active operations, the Origin and the Target always return to *Z-Acc recvd* and *Z-Acc sent* stages respectively. As long as there are active operations, the Origin and the Target are in the *Concurrent Active* state. When there are no more active operations, the Origin and the Target become *Concurrent Idle* (p9 and p20 respectively). Tables 4.27 - 4.30 describe the places and transitions involved in this process.

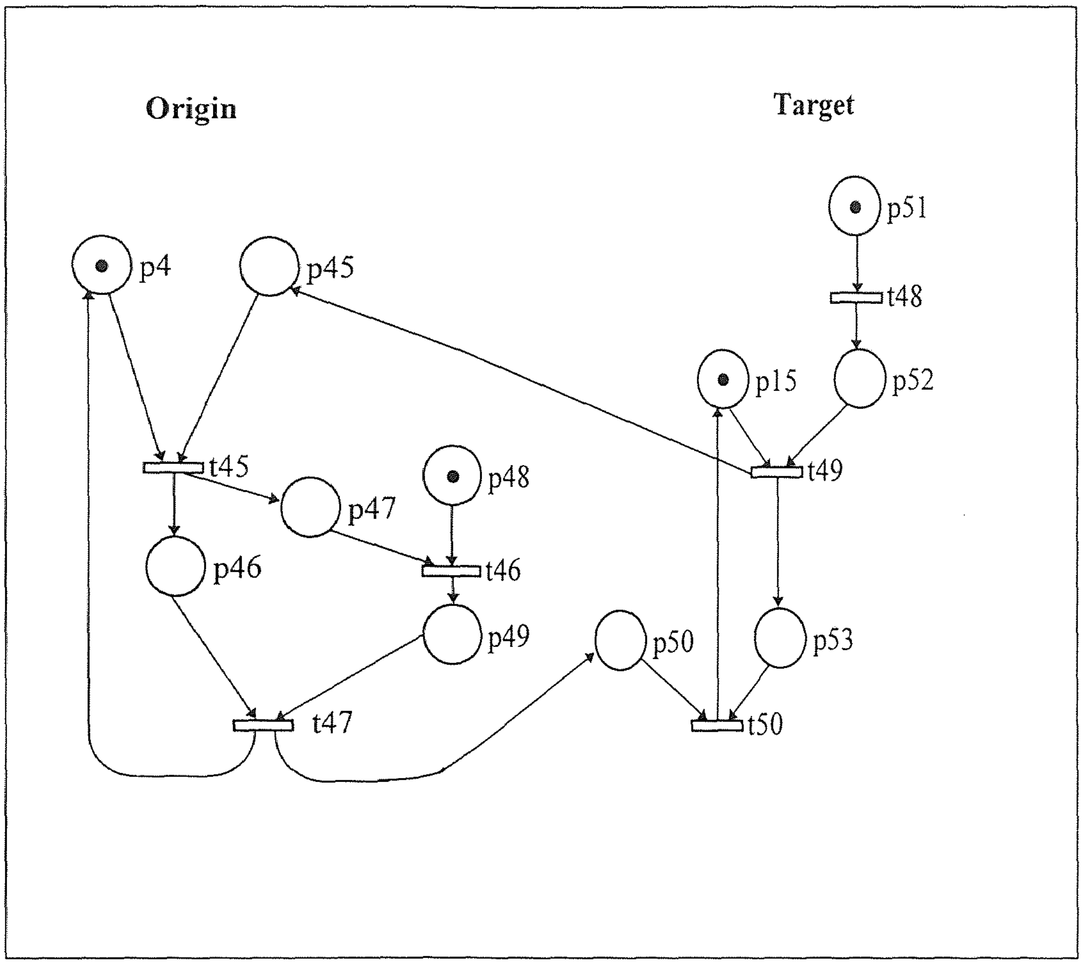


Figure 4.8 Access Control Request During Initialization Phase

Table 4.23 Access Control During Initialization - Origin Place Description

Place	Description
p4	Init Sent
p45	Acc PDU
p46	Acc Recvd
p47	Acc ind
p48	Acc Resp Ready
p49	Acc Resp

Table 4.24 Access Control During Initialization - Target Place Description

Place	Description
p15	Init Recvd
p50	Acc Resp PDU
p51	Acc Req Ready
p52	Acc Req
p53	Acc Sent

Table 4.25 Access Control During Initialization - Origin Transition Description

Transition	Description
t45	Acc Ind sent
t46	Acc Resp sent
t47	Acc Resp PDU sent

Table 4.26 Access Control During Initialization - Target Transition Description

Transition	Description
t48	Acc Req sent
t49	Acc PDU sent
t50	Acc Conf sent

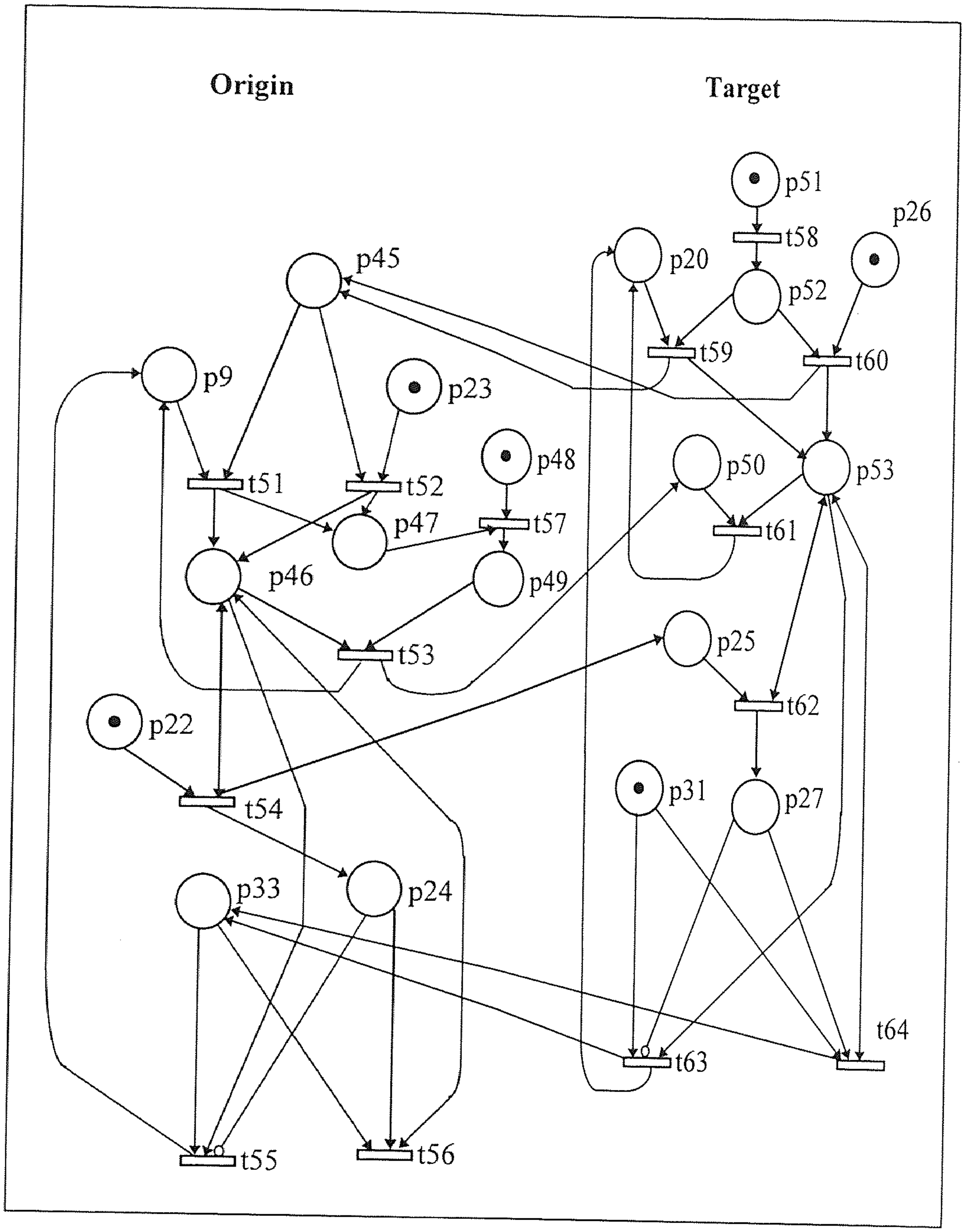


Figure 4.9 Access Control Request During Processing Phase

Table 4.27 Access Control During Processing - Origin Place Description

Place	Description
p9	Concurrent Idle
p45	Z-Acc PDU
p46	Z-Acc Recvd
p23	Concurrent Active
p22	Op Req
p24	Operation Counter
p33	EndOp Ind
p47	Z-Acc Ind
p48	Z-Acc Response Ready
p49	Z-Acc Response

Table 4.28 Access Control During Processing - Target Place Description

Place	Description
p20	Concurrent Idle
p51	Acc Request Ready
p52	Acc Req
p26	Concurrent Active
p50	Z-Acc Response PDU
p53	Z-Acc Sent
p25	Op PDU
p31	EndOp Ind
p27	Operation Counter

Table 4.29 Access Control During Processing - Origin Transition Description

Transition	Description
t51	Acc Ind sent
t52	Acc Ind sent
t53	Z-Acc Resp PDU sent
t54	Op PDU sent
t55	No Operation (Concurrent Idle)
t56	Active Operations (Z-Acc Recvd)
t57	Z-Acc Resp sent

Table 4.30 Access Control During Processing - Target Transition Description

Transition	Description
t58	Z-Acc Request sent
t59	Z-Acc PDU sent
t60	Z-Acc PDU sent
t61	Acc Conf sent
t62	Op Conf sent
t63	No Operation (Concurrent Idle)
t64	Active Operation (A-Acc Sent)

4.2.6 Access Control Request: During Operation

The Access Control request for all the operation types follows a similar set of activities. After receiving an operation request, the Origin is in *Op sent* (p23) state and the Target is in *Op recvd* (p27*)* state. After completing the series of Access Control requests and responses, the Origin and the Target return to *Op sent* and *Op recvd* states respectively. Figure 4.10 describes the Access Control request during operation. *Op sent* and *Op recvd* states represent all the operation types. Tables 4.31 - 4.34 show the place and transition description.

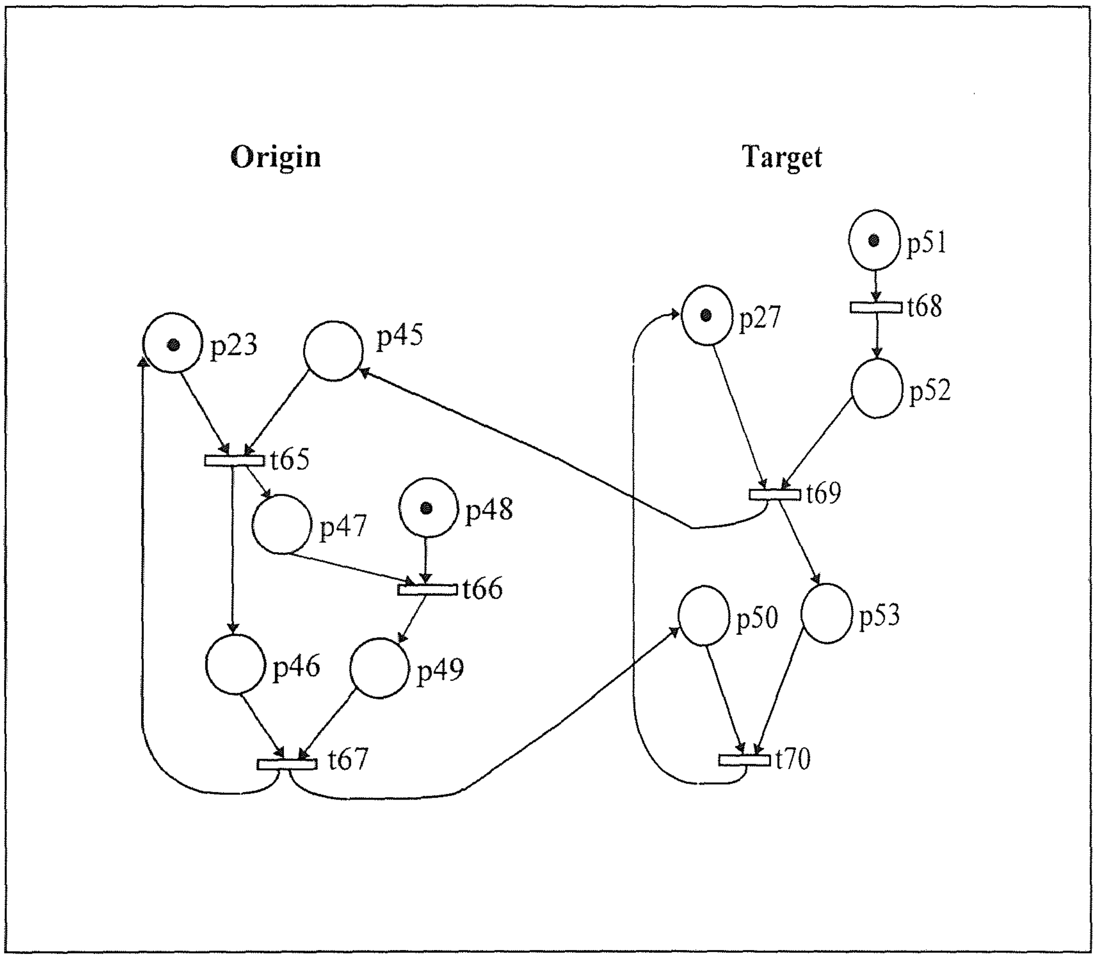


Figure 4.10 Access Control Request During Operation

Table 4.31 Access Control During Operation - Origin Place Description

Place	Description
p45	Acc PDU
p46	Acc Recvd
p47	Acc Ind
p48	Acc Response Ready
p49	Acc Resp
p54	Op Sent

Table 4.32 Access Control During Operation - Target Place Description

Place	Description
p50	Acc Resp PDU
p51	Acc Request Ready
p52	Acc Req
p53	Acc Sent
p55	Op Recvd

Table 4.33 Access Control During Operation - Origin Transition Description

Transition	Description
t65	Acc Ind sent
t66	Acc Resp sent
t67	Acc Resp PDU sent

Table 4.34 Access Control During Operation - Target Transition Description

Transition	Description
t68	Acc Request sent
t69	Acc PDU sent
t70	Acc Conf sent

4.2.7 Resource Control Request: During Initialization

Target can issue a Resource Control request during an initialization, processing or termination phases. The Resource Control request can also appear during an operation. The behavior of a Resource Control request is exactly identical to that of Access Control request except for one difference: The Resource Control service is a Conditionally Confirmed service. A response may or may not follow for a Resource Control request.

Figure 4.11 describes a Resource Control request during initialization. If the Target indicates the "response required" predicate in the Resource Control request, only then the Origin sends a Resource Control response. In the initialization process, the Resource Control request appears when the Origin and the Target are in the *Init sent* and *Init recvd* states respectively. Upon receiving the Resource Control request, the *Rsc PDU* (p56) is sent to the network. After responding to the Resource Control request, *Rsc resp PDU* (p65) is sent to the Target by the Origin.

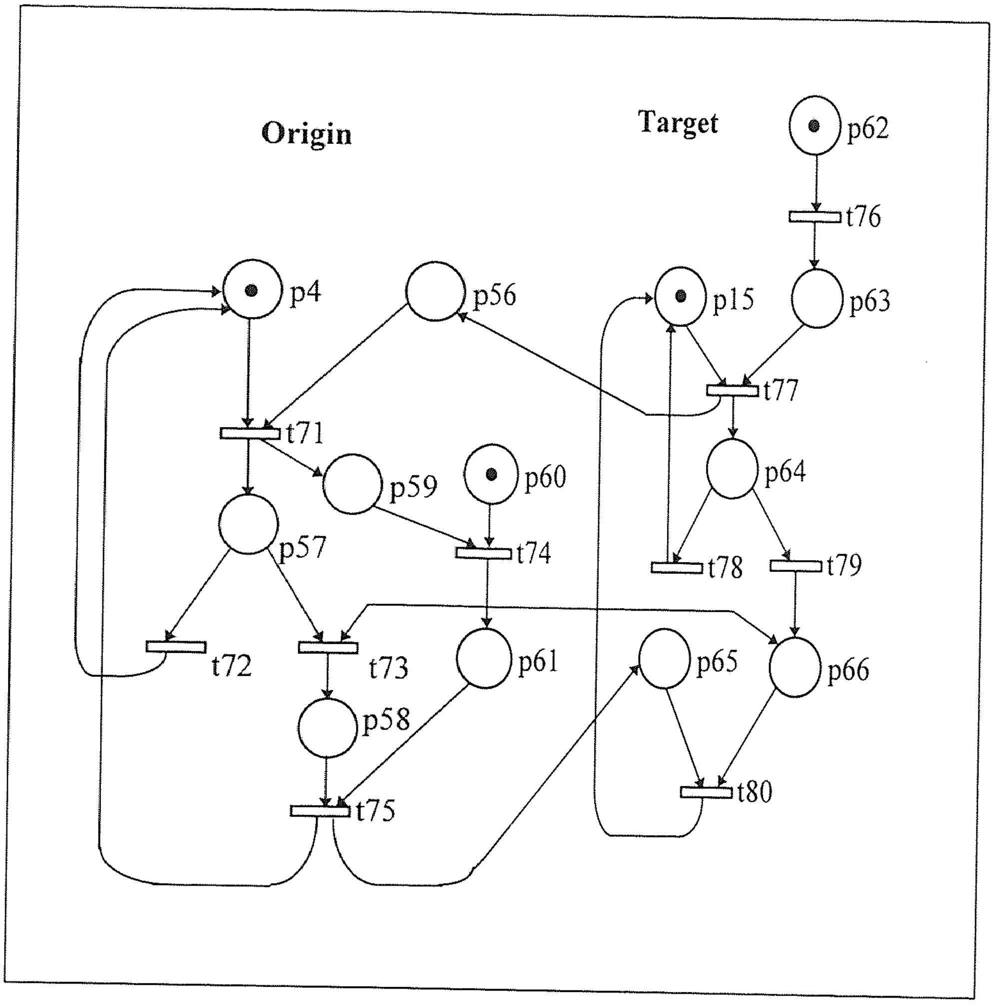


Figure 4.11 Resource Control Request During Initialization

It can be seen from figure 4.11 that the Origin and the Target are in *Init sent* (p4) and *Init recvd* (p15) state when the Resource Control request (*Rsc req*) appears in p62. If the response is expected from the Origin, then the target and the Origin go to *Rsc sent* (p66) and *Rsc recvd* (p58) states respectively. After receiving the Resource control response (*Rsc resp*) (p60) from the Origin, the Target and then the Origin return to their original states (p15 and p4 respectively). In this model, transitions t79 and t73 fire if the response is required from the origin and the transitions t78 and t72 fire if the response is not required from the Origin. Tables 4.35 - 4.38 describe the places and transitions in this PN model.

Table 4.35 Resource Control During Initialization - Origin Place Description

Place	Description
p4	Init Sent
p56	Rsc PDU
p57	Response/No Response Decision
p58	Rsc Recvd
p59	Rsc Ind
p60	Rsc Response Ready
p61	Rsc Response

Table 4.36 Resource Control During Initialization - Target Place Description

Place	Description
p15	Init Recvd
p62	Rsc Request Ready
p63	Rsc Req
p64	Response/No Response Decision
p65	Rsc Sent
p66	Rsc Response PDU

Table 4.37 Resource Control During Initialization - Origin Transition Description

Transition	Description
t71	Rsc Ind Sent
t72	No Response (Go To Init Recvd State)
t73	Response (Go To Rsc Recvd State)
t74	Rsc Response sent
t75	Rsc Response PDU sent

Table 4.38 Resource Control During Initialization - Target Transition Description

Transition	Description
t76	Rsc Request sent
t77	Rsc PDU sent
t78	No Response (Go To Init Recvd State)
t79	Response (Go To Rsc Sent)
t80	Rsc Conf sent

4.2.8 Resource Control Request: During Processing Phase

In Figure 4.12, places p9 and p26 represent the *Concurrent Idle* states for the Origin and the Target respectively. Place p62 shows the Resource Control request and place p60 shows the Resource Control response. All the abbreviations for the states and the events begin with "Z" which indicates that the Resource Control request is pertaining to a Z-association. A Resource Control request PDU *Z-Rsc PDU* (p56) is sent to the Origin after receiving the Resource Control request. The Origin responds to the Resource Control request if the Target indicates that the response is required. If the response is not required then the Origin and the Target return to their original state indicated by the places p9 and p26. The operation counter for the Origin and the Target indicated by the places p and p27 is maintained to keep track of the total number of active operations. This counter is decremented upon the termination of the operation.

An operation request (p22) can occur before Resource Control response when the Origin and the Target are in *Z-Rsc recvd* (p58) and *Z-Rsc sent* (p53) states respectively. An end operation signal *EndOp ind* (p31) from the Target concludes the operation and the Origin and the Target return to p58 and p53 respectively. Upon receiving a Resource control response (p60), the Origin and the Target go to their initial states p9 and p26 respectively. Tables 4.39 - 4.42 show the description of this PN model.

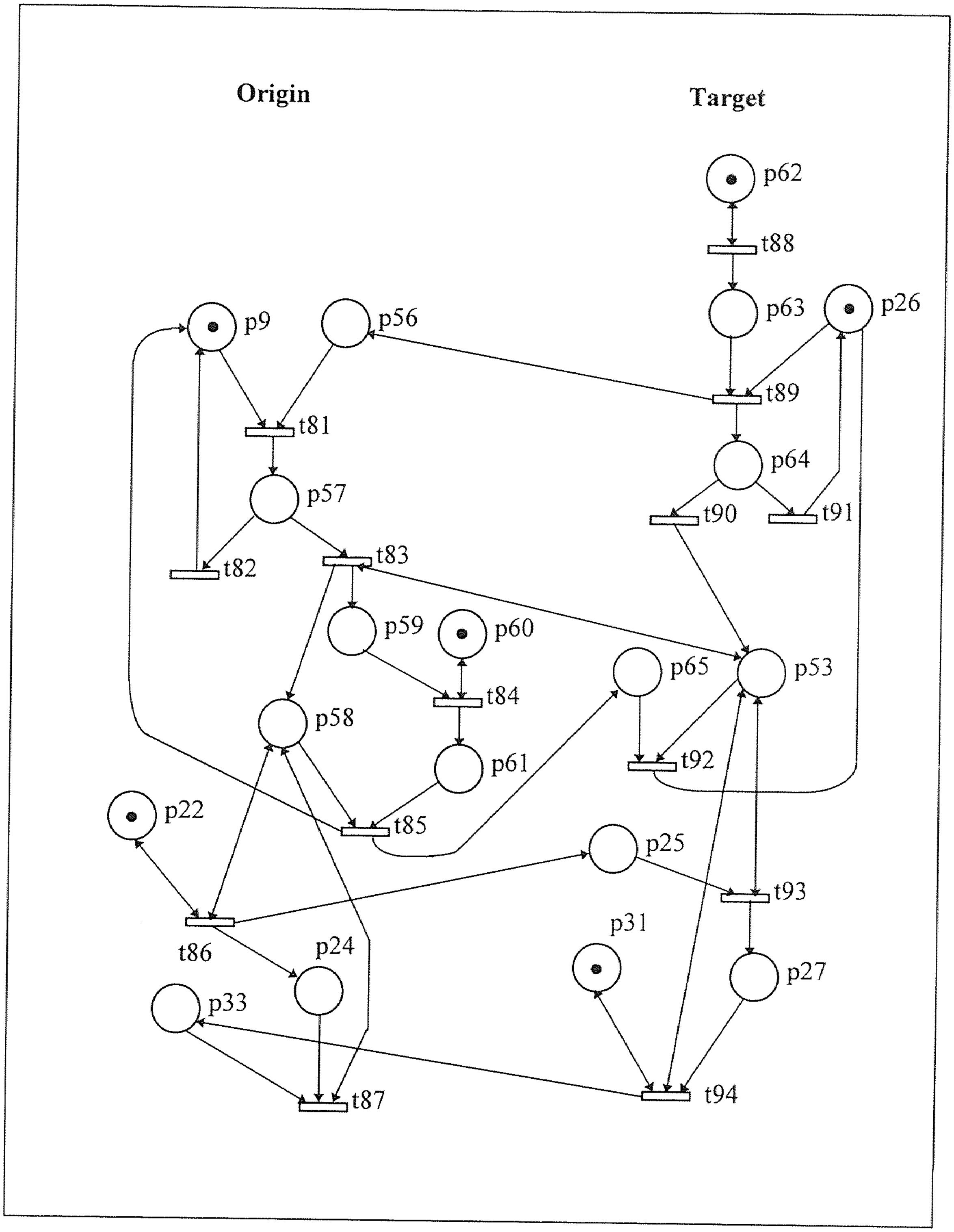


Figure 4.12 Resource Control Request During Processing

Table 4.39 Resource Control During Operation - Origin Place Description

Place	Description
p9	Concurrent Idle
p56	Z-Rsc PDU
p57	Response/No Response Decision
p58	Z-Rsc recvd
p59	Rsc ind
p60	Z-Rsc Response Ready
p61	Z-Rsc resp
p22	Op req
p23	Concurrent Active
p24	Operation Counter
p33	EndOp ind

Table 4.40 Resource Control Request - Target Place Description

Place	Description
p20	Concurrent Idle
p25	Op PDU
p26	Concurrent Active
p27	Operation Counter
p31	EndOp ind
p53	Z-Rsc sent
p62	Z-Rsc Request Ready
p63	Z-Rsc req
p64	Response/No Response Decision
p65	Z-Rsc resp PDU

Table 4.41 Resource Control During Processing - Origin Transition Description

Transition	Description
t81	No Response (Go To Concurrent Idle)
t82	Z-Rsc ind sent
t83	Response (Go To Z-Rsc recvd)
t84	Z-Rsc ind sent
t85	Z-Rsc Response sent
t86	Z-Rsc resp PDU sent
t87	Op PDU sent
t88	No Operations(Go To Concurrent Idle)
t89	Active Operations(Go To Z-Rsc recvd)

Table 4.42 Resource Control During Processing - Target Transition Description

Transition	Description
t90	Z-Rsc Request Ready
t91	Z-Rsc PDU sent
t92	Z-Rsc PDU sent
t93	No Response(Go To Concurrent Idle)
t94	Response(Go To Z-Rsc sent)
t95	Z-Rsc conf sent
t96	Op conf sent
t97	No Operations(Go To Concurrent Idle)
t98	Active Operations(Go To Z-Rsc sent)

4.2.9 Resource Control Request: During Operation

A Resource Control request can occur when the Origin and the target are in the *Op sent* and *Op recvd* state respectively. These system goes into these states during an operation request. The *Op sent* and *Op recvd* states are nothing but the *Concurrent Active* or *Serial Active* states of the Origin and the Target. Here they are represented as the *Concurrent Active* states (p23 for Origin and p26 for the Target). The Target indicates in the Resource Control request whether a response from the Origin is required. Transitions t97 and t102 fire if the response from the Origin is desired. The Origin and the Target go to *Rsc recvd* (p58) and *Rsc sent* (p53) states respectively. Upon receiving a Resource Control response (p60), the Origin and the Target return to their initial states (p23 and p26 respectively). If the response is not required then the transitions t102 and t96 fire. Firing of theses transitions will take the system back to the initial state *Op sent* and *Op recvd* of the Origin and Target respectively. Tables 4.43 - 4.46 explain the places and transitions in the PN model.

This concludes the discussion of PN modeling for different services and operations. We now focus on performance analysis based on these models.

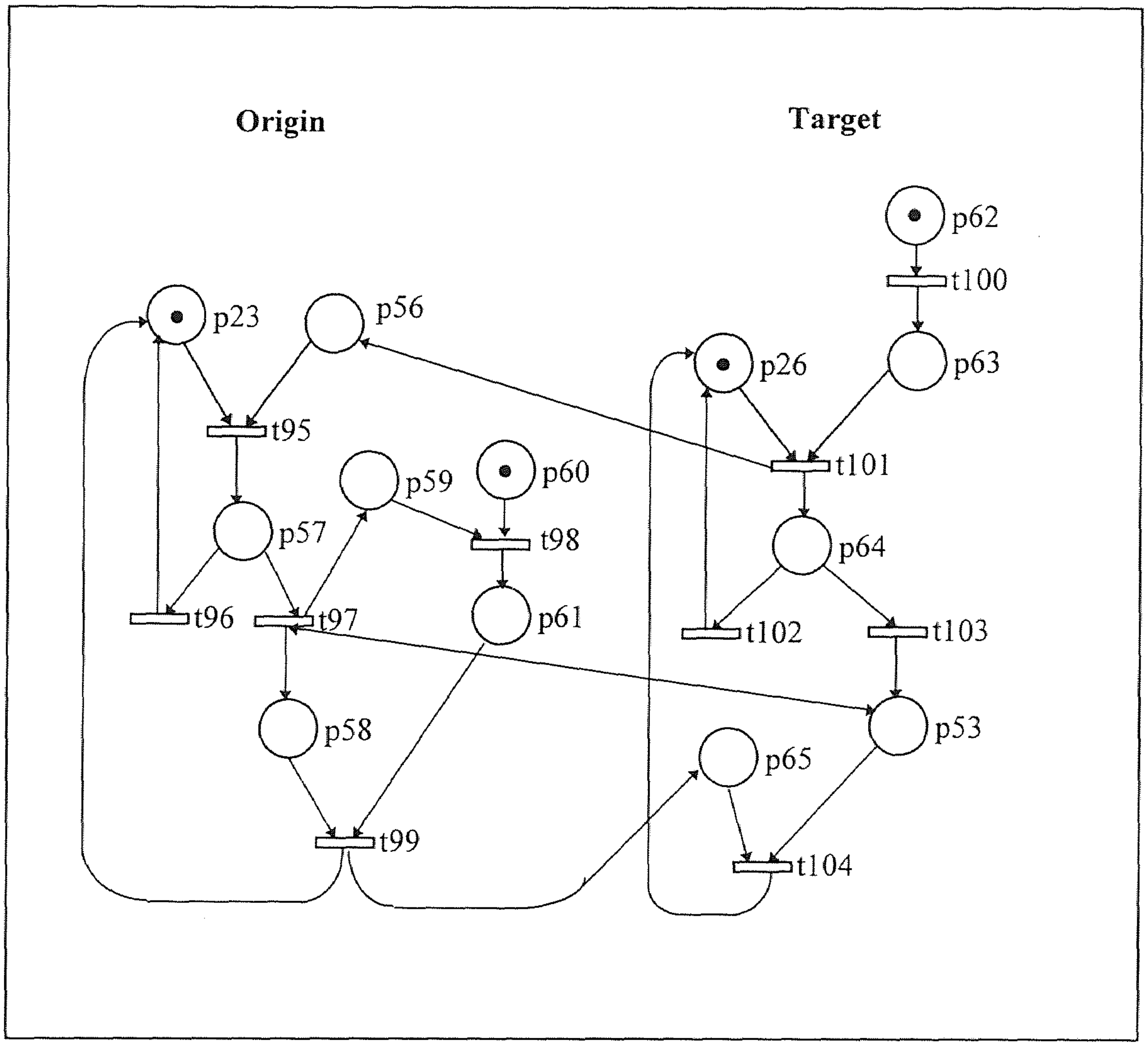


Figure 4.13 Resource Control Request During Operation

Table 4.43 Resource Control During Operation - Origin PlaceDescription

Place	Description
p23	Concurrent Active
p56	Rsc PDU
p57	Response/No Response Decision
p58	Rsc recvd
p59	Rsc ind
p60	Rsc Response Ready
p61	Rsc resp

Table 4.44 Resource Control During Operation - Target Place Description

Place	Description
p26	Concurrent Active
p53	Rsc Sent
p62	Rsc Request Ready
p63	Rsc Req
p64	Response/No Response Decision
p65	Rsc Response PDU

Table 4.45 Resource Control Request - Origin Transition Description

Transition	Description
t95	Response /No Response Decision sent
t96	No Response (Go To Concurrent Active)
t97	Rsc Ind sent
t98	Rsc Response sent
t99	Rsc Response PDU sent

Table 4.46 Resource Control Request - Target Transition Description

Transition	Description
t100	Rsc Request sent
t101	Rsc PDU sent
t102	No Response (Go To Concurrent Active)
t103	Response (Go TO Rsc Sent)
t104	Rsc Conf sent

CHAPTER 5

PERFORMANCE ANALYSIS OF Z39.50

5.1 Petri Net Properties

There are several different properties of a Petri net model. These properties are mainly classified as the behavioral and structural properties [6]. The behavioral properties depend on the initial state or marking of a Petri net. The structural properties on the other hand depend on the net structure of a Petri net. Reachability, boundedness, conservativeness and liveness [4,5,6] are some examples of the behavioral properties. In this section we shall describe the Petri net reachability property in detail. A Reachability Graph is prepared for the entire Z39.50 system. Using a timed Petri net simulation tool (TiPNet) [7], we have evaluated the system performance.

5.2 Petri Nets and Reachability

A reachability property is extremely useful in finding out whether a system can reach a specific state. It also describes the sequence of transitions that may fire in order to transform a system from one state to another.

Marking: The marking of a Petri net at any instance in time depicts the position of the tokens with respect to the places of the Petri net. An initial marking (M0) describes the initial state of the system. The sequence of firing of transitions moves the Petri net into a new marking (Mi) where Mi represents the new state that is reached as a result of a specific sequence of transition firings.

Reachability Set and Reachability Graph: A marking Mi is said to be reachable from a marking M0 if there exists a sequence of transition firings that transforms a marking from M0 to Mi. A set of all reachable markings from M0 is called a Reachability Set. A Reachability Set contains no information about the transition sequence to be fired to reach each marking. This information is present in the Reachability Graph. We have drawn a Reachability Graph for Z39.50 in several steps. Figure 5.1 presents M0 that is nothing but the initial state of the system. In a Reachability Graph, each place is associated with a natural number and this number represents the number of tokens in that place at any given time. For example, if there are four places p1, p2, p3 and p4 then the marking Mi (1 0 0 2) represents one token in place p1, none tokens in places p2 and p3 and two tokens in place p4. The Petri net model for Z39.50 system has approximately 64 places and 104 transitions. It is difficult to draw a Reachability Graph for a system whose initial marking contains 64 natural numbers. To simplify this situation, in our case we have indicated only those places in the marking that has tokens. For example, initial marking M0 (p1 p2 p4 p11) indicates that there is one token in places p1, p2, p4 and p11 and all other places have zero tokens. Another advantage of this notation is that one can easily identify the places that are responsible for the firing of the transition and subsequently changing the state of the system. In Figure 5.1 the arc ti from M0 to Mi shows that Mi is directly reachable from M0.

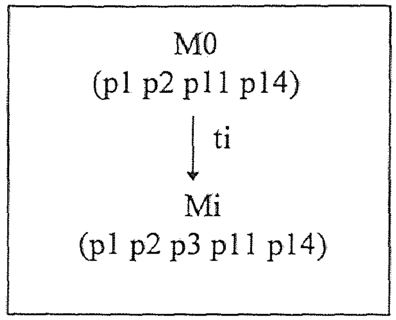


Figure 5.1 Example of Marking and Firing Transition

If a place p1 has two or more tokens then it can be represented as jp1 where j is a positive integer (e.g. 2 tokens in p1 can be represented as 2p1 in the Reachability Graph).

In Figure 5.2, transition t1 is enabled in marking M0 (p1 p2 p11 p14) and its firing transforms the system state to M1 (p1 p2 p3 p11 p14). In M1, transition t2 is enabled whose firing will give rise to a new marking M2 and so on. There can be more than one transition enabled in a given state. A Reachability Graph represents the behavior of a system. Depending upon the system response, the transition t10 or t11 can fire changing the system state to either M5 or M5' respectively. From M5, the system goes back to the original state (that is, initial marking) and from M5' the subsequent transitions are fired leading the system to either Serial or Concurrent operations state. Figure 5.3 - 5.4 show the transition firing sequence when the system is in the Concurrent and Serial operations state respectively. Figures 5.5 - 5.12 show changes in the system state upon receiving different requests such as Close request, Access Control request and Resource Control request. After receiving a Close request, the system returns to the original state indicating that the initial marking is reached.

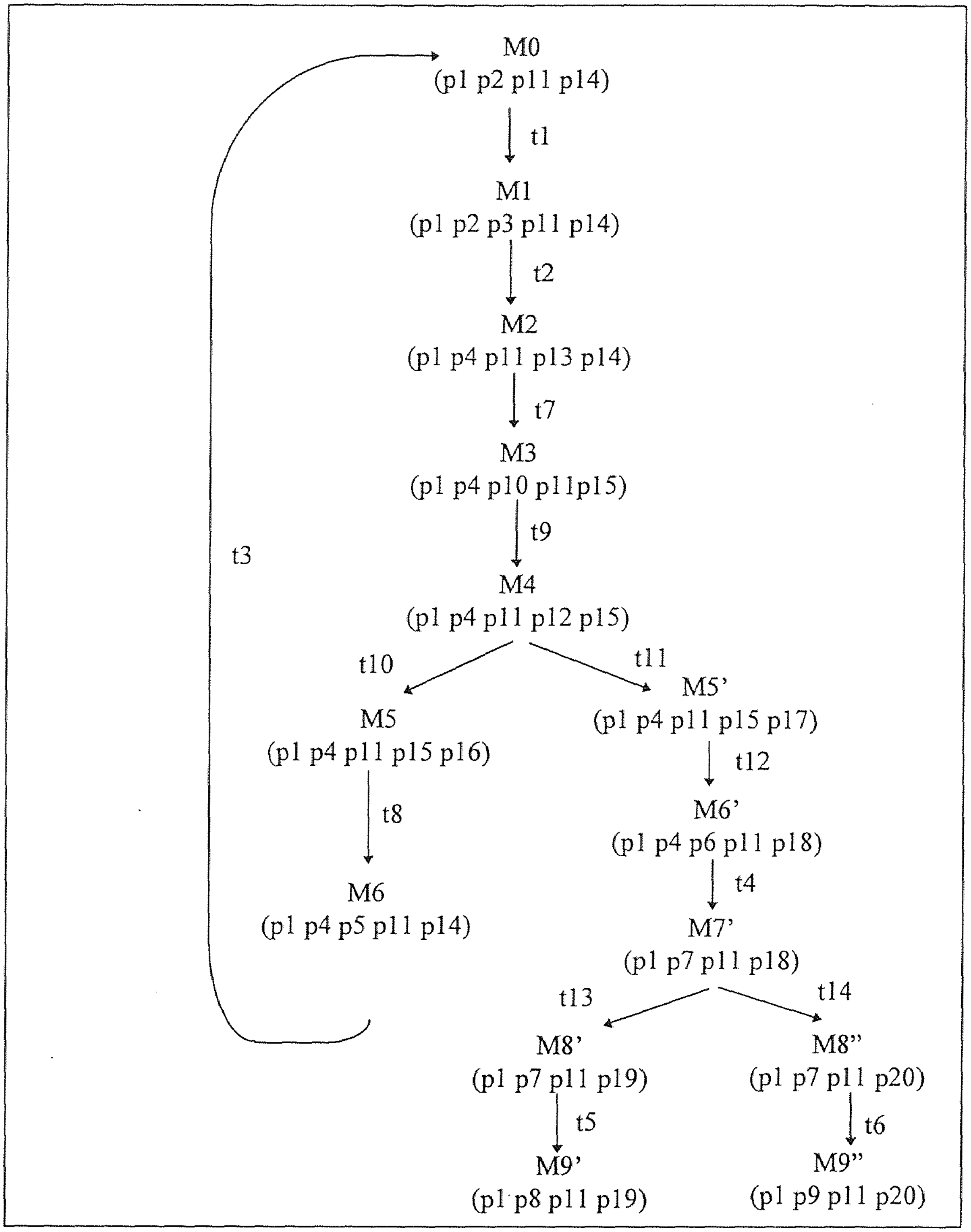


Figure 5.2 Transition Sequence - Initialization Phase (Refer Fig. 4.1)

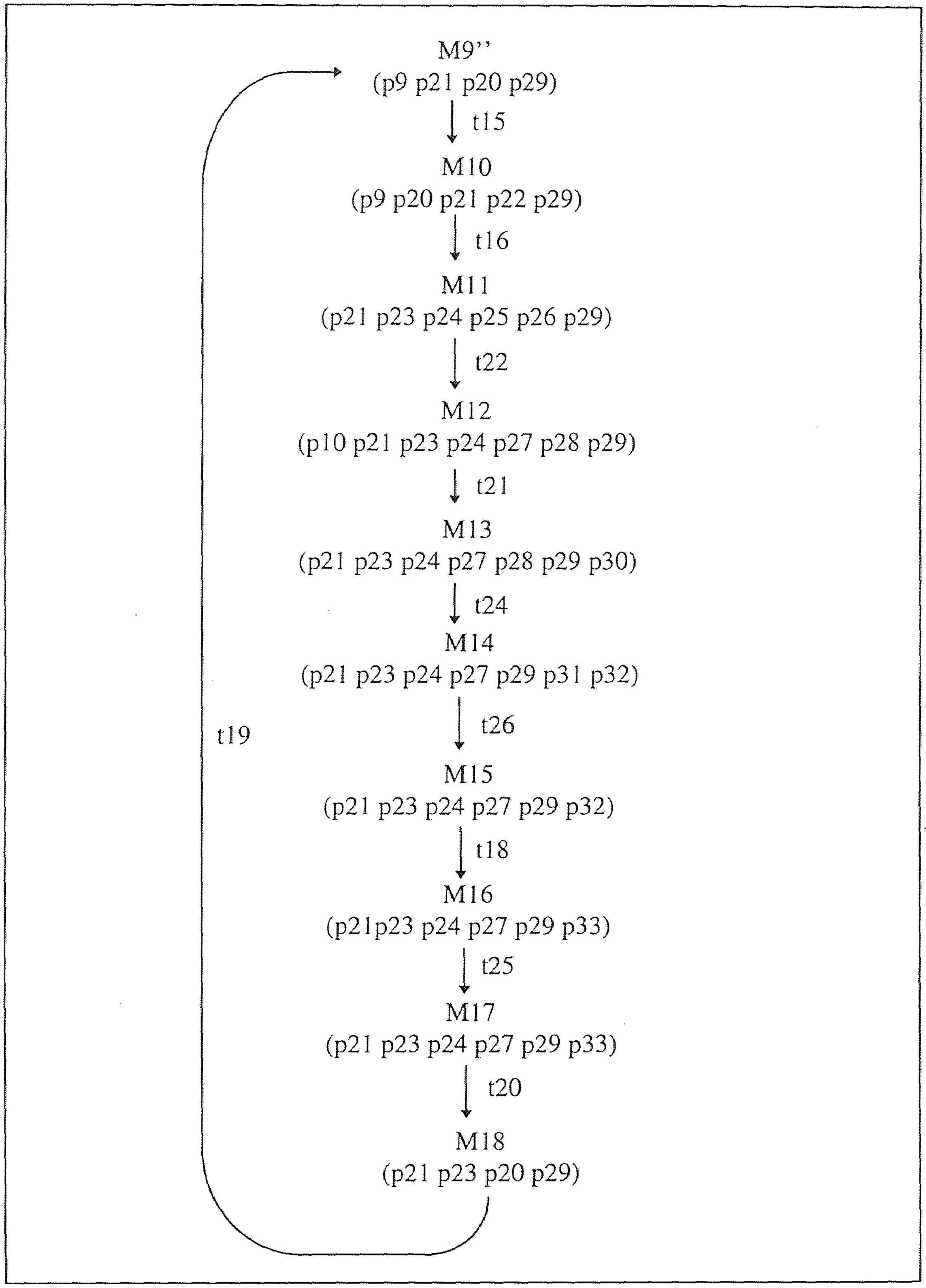


Figure 5.3 Transition Sequence - Concurrent Operations (Refer Fig. 4.2)

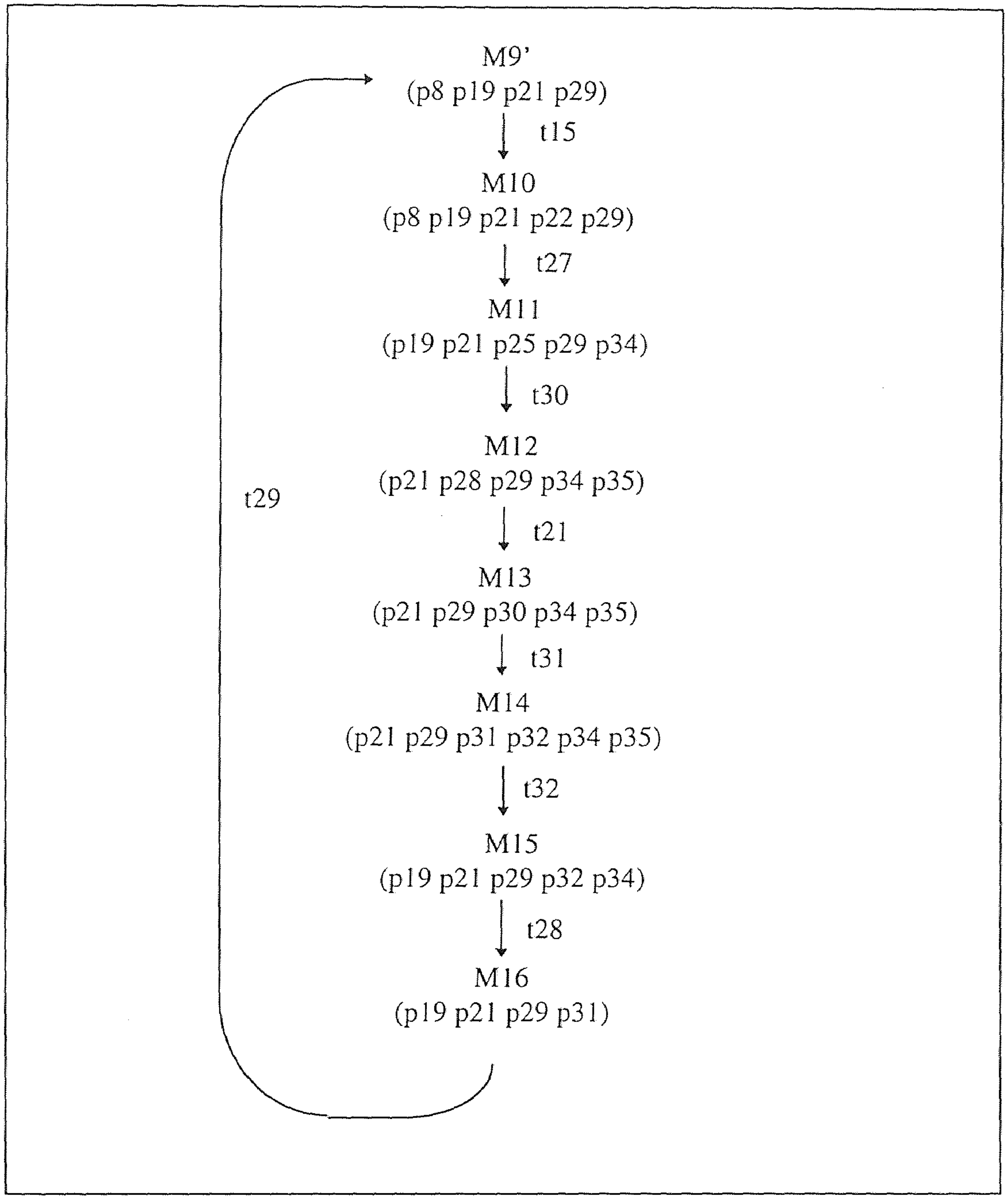


Figure 5.4 Transition Sequence - Serial Operations (Refer Fig. 4.3)

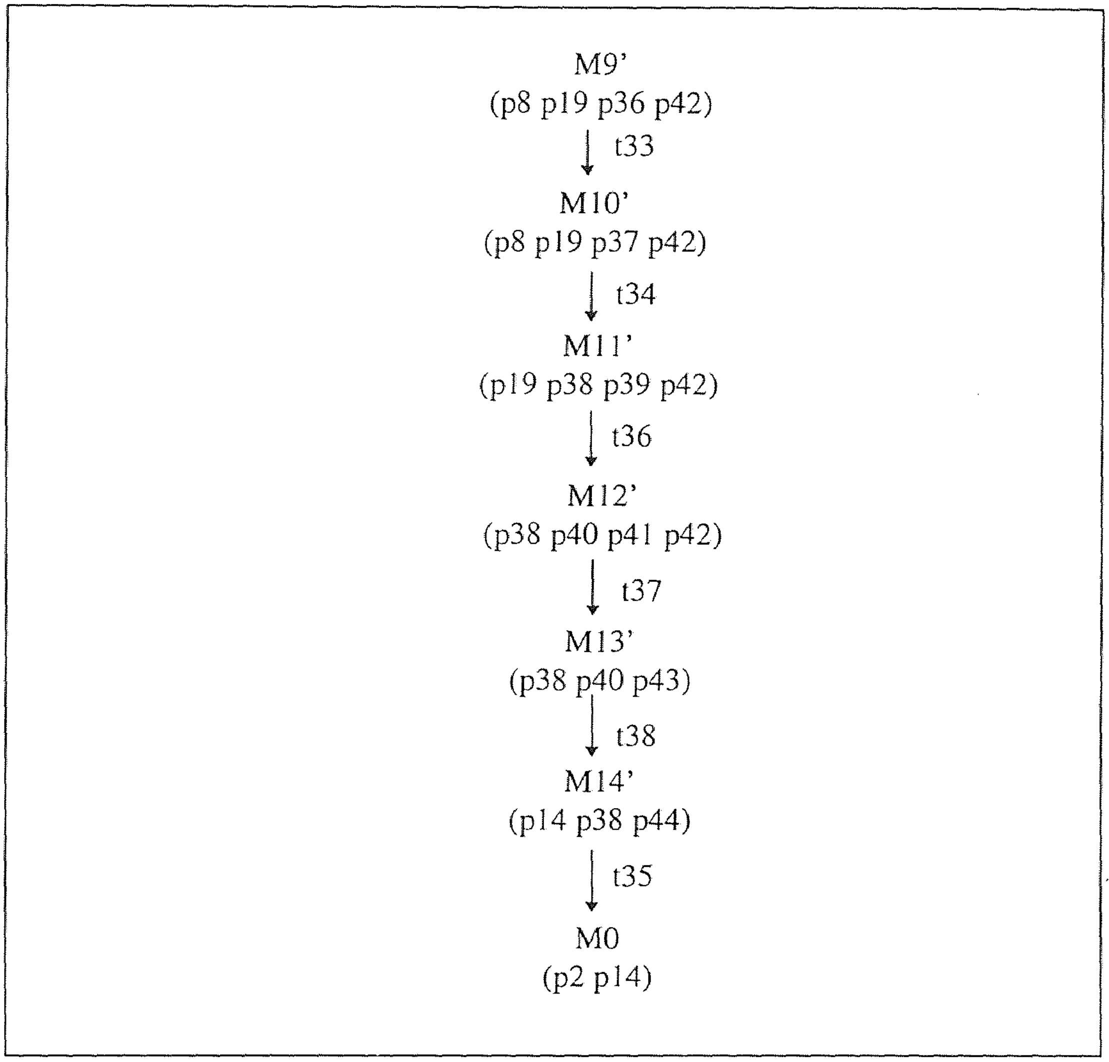


Figure 5.5 Transition Sequence - Close Request from Origin (Refer Fig 4.6)

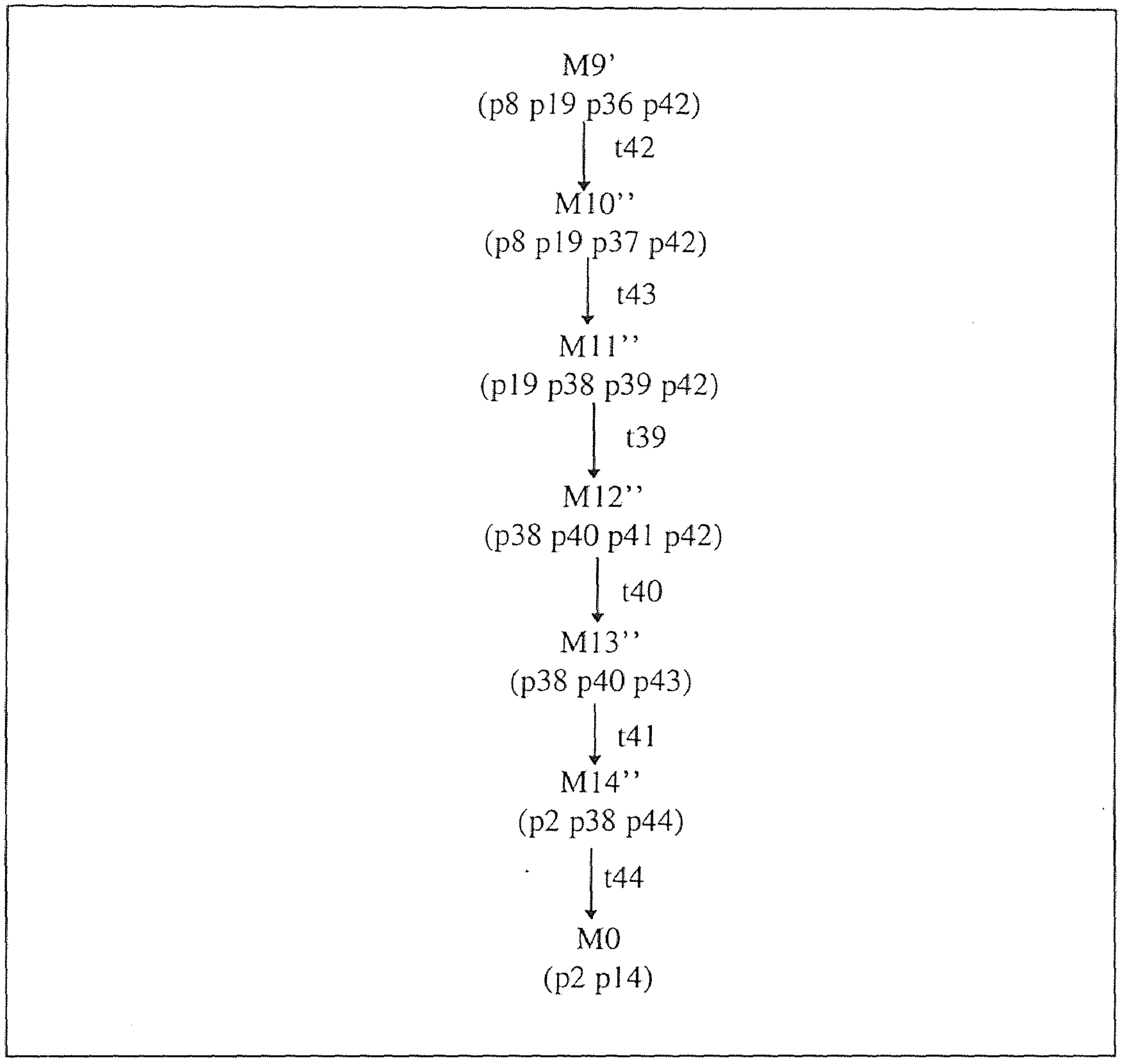


Figure 5.6 Transition Sequence - Close Request from Target (Refer Fig. 4.7)

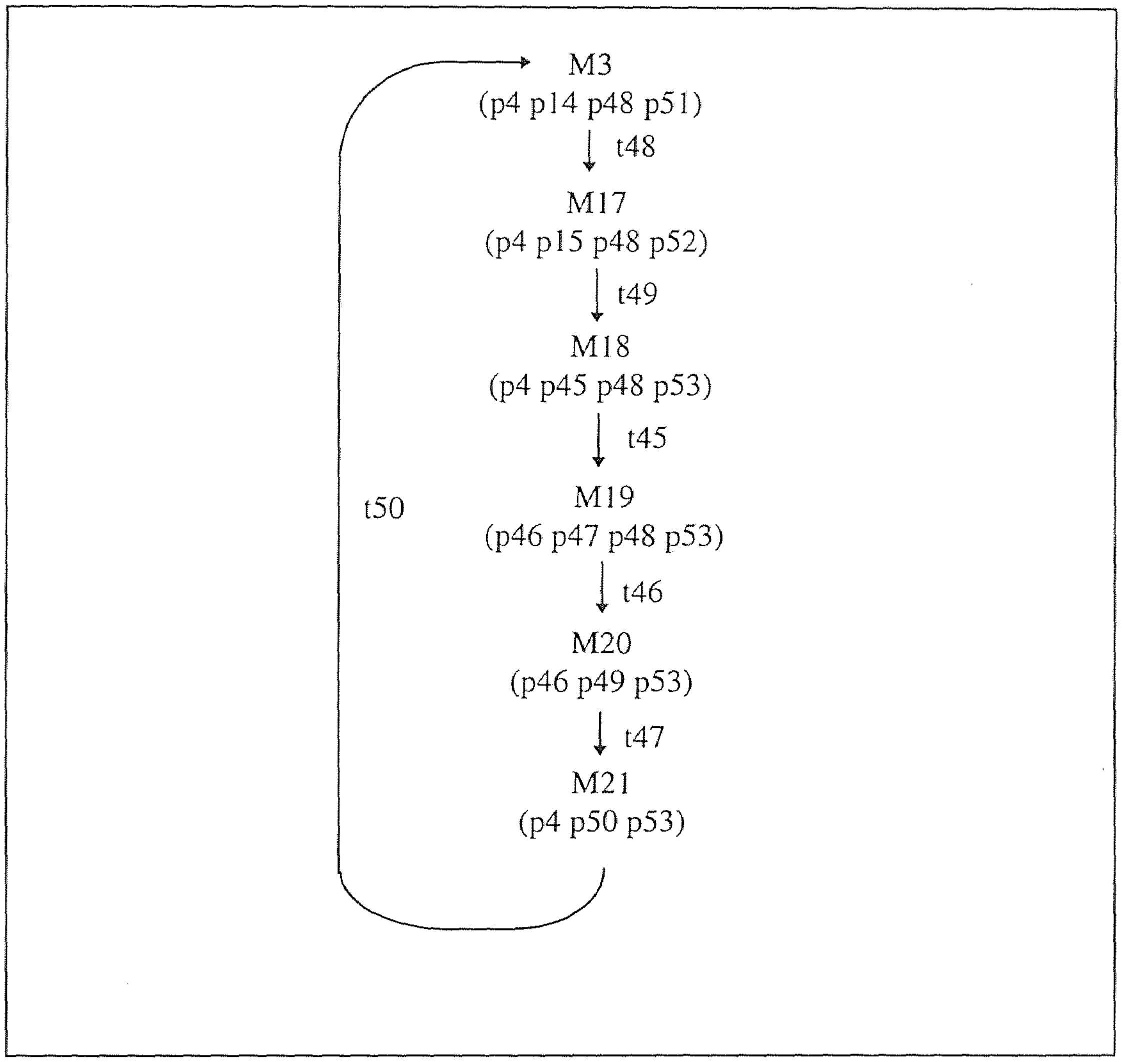


Figure 5.7 Transition Sequence - Access Control During Initialization (Refer Fig. 4.8)

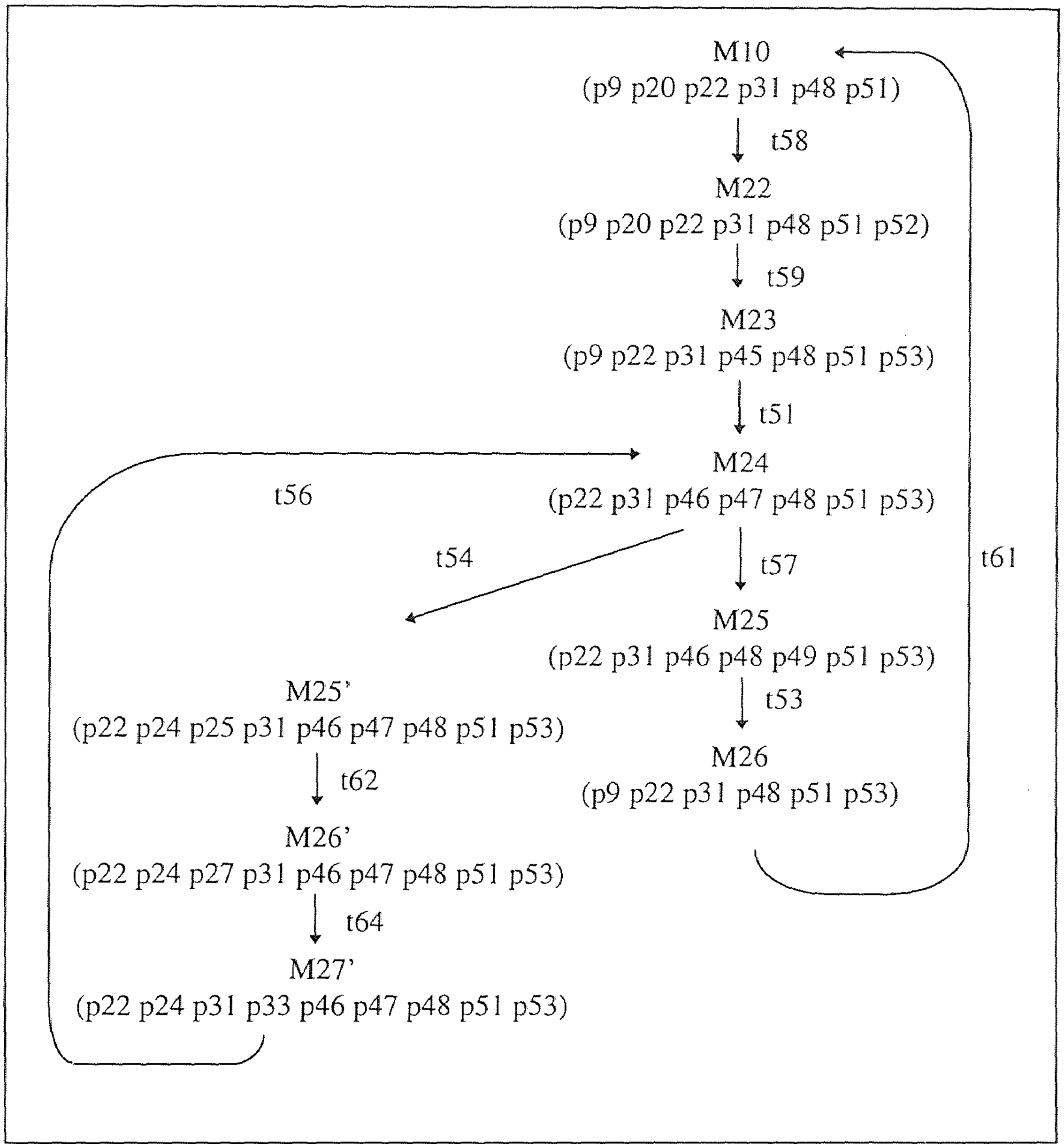


Figure 5.8 Transition Sequence - Access Control During Concurrent Operations (Refer Fig. 4.9)

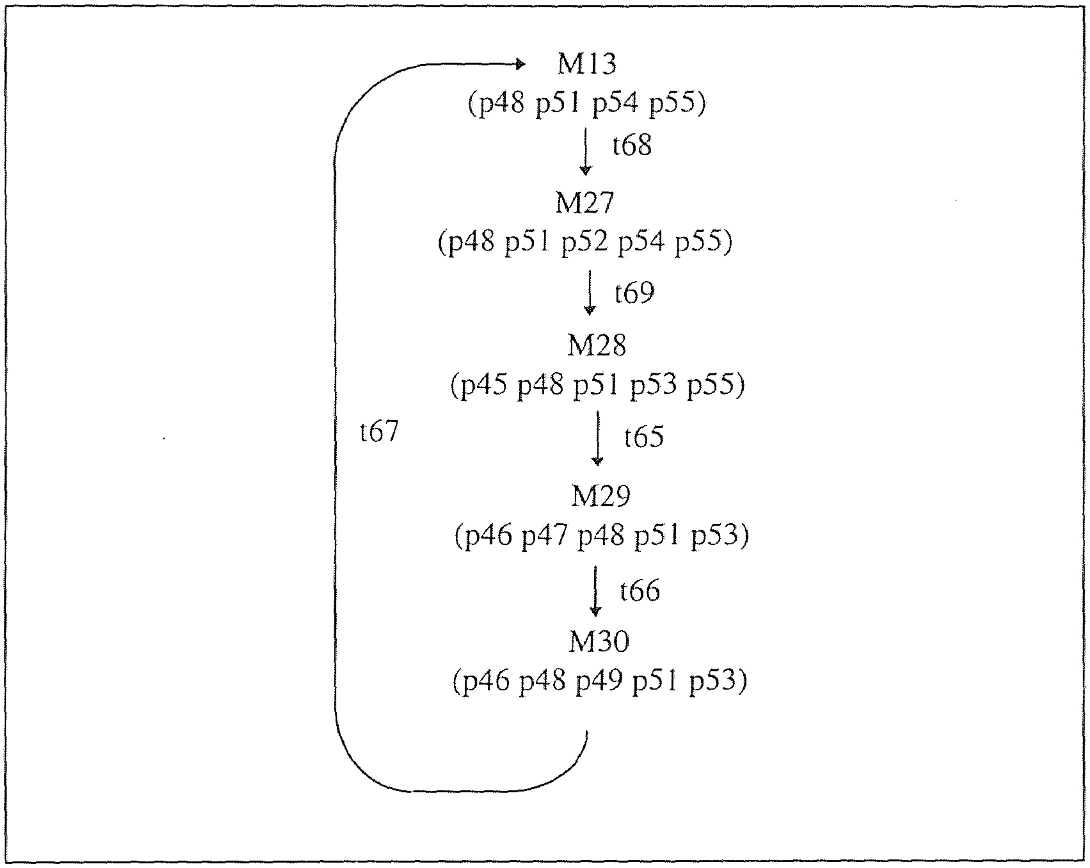


Figure 5.9 Transition Sequence - Access Control During Operation (Refer Fig. 4.10)

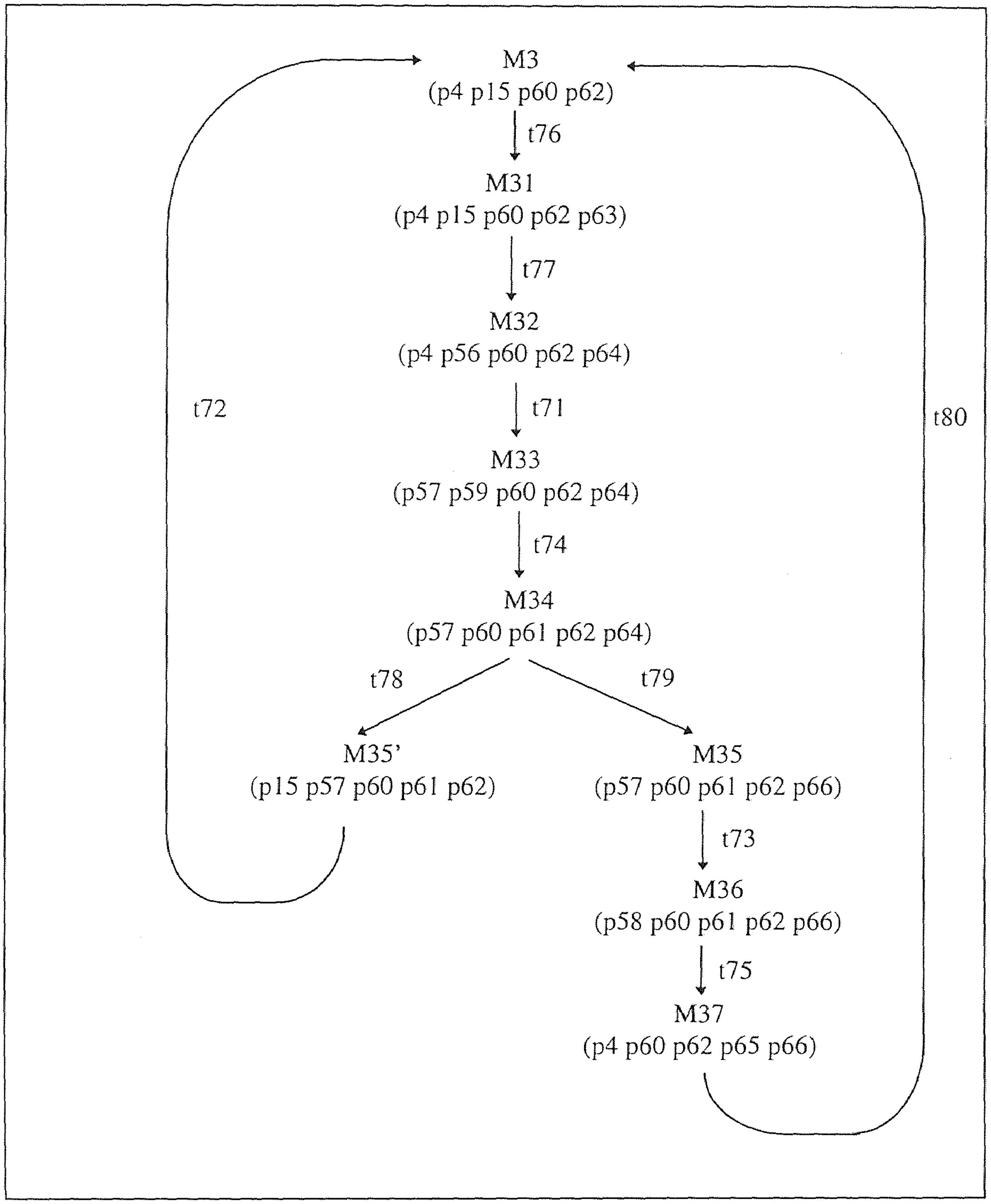


Figure 5.10 Transition Sequence - Resource Control During Initialization (Refer Fig. 4.11)

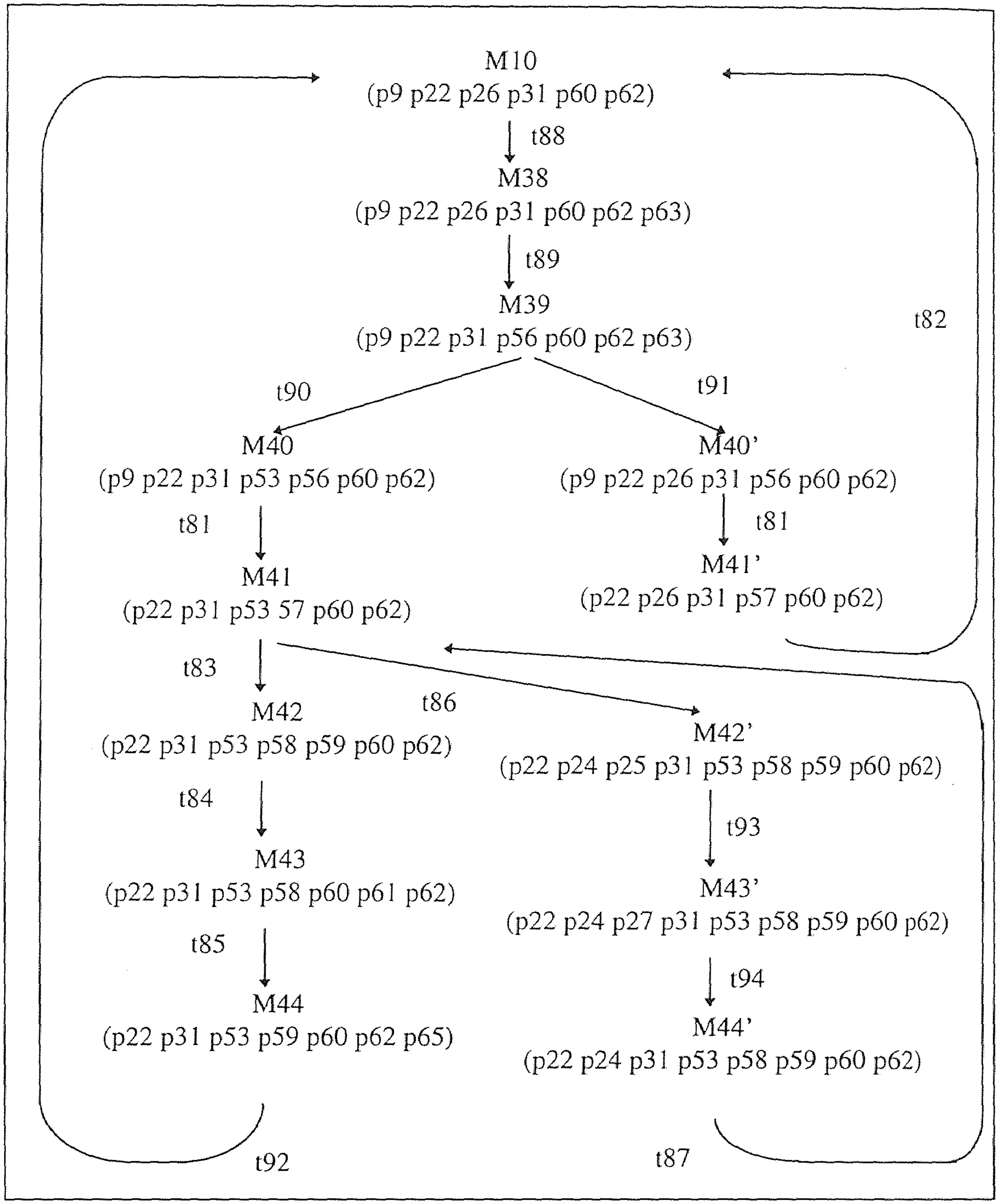


Figure 5.11 Transition Sequence - Resource Control During Processing (Refer Fig. 4.12)

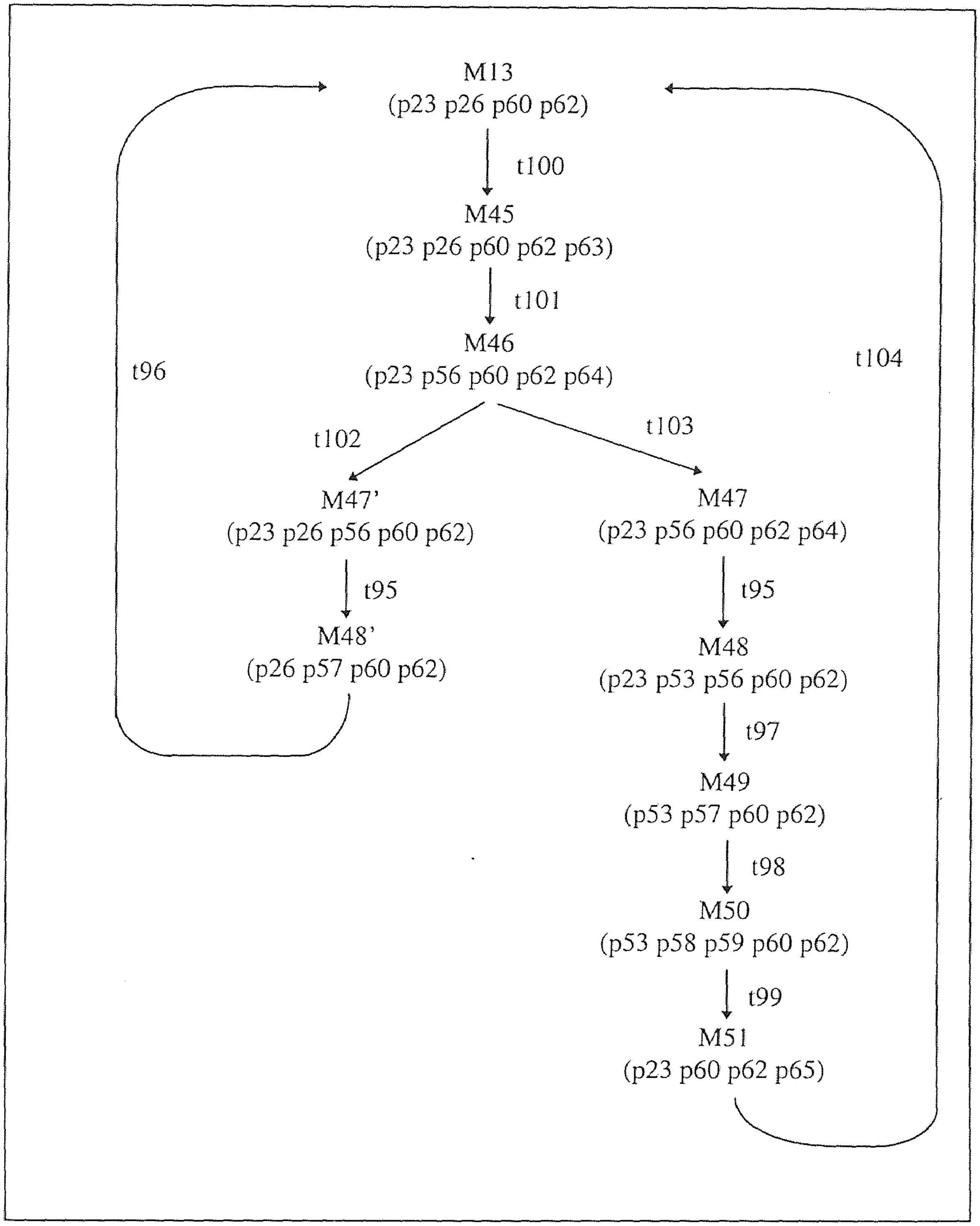


Figure 5.12 Transition Sequence - Resource Control During Operation (Refer Fig. 4.13)

5.3 Reachability Graph for Z39.50 System

It is essential to integrate the different Z39.50 services along with the initialization, processing and termination phases of Z39.50 system in order to compute the time required to complete these processes. Considering the transition firing sequences in Figure 5.3 - 5.12, we have prepared a Reachability Graph for the entire Z39.50 system.

This Reachability Graph has two components Mi and ti where Mi is the marking and ti the transition that will be fired. We have presented this graph in three parts (Figures 5.12-5.14). In part 1 (Figure 5.13), initialization process along with the possible Access Control and Resource Control requests is shown. If the initialization fails, the system returns to the original state and if it succeeds the system goes to Serial or Concurrent state. Part 2 (Figure 5.14) and Part 3 (Figure 5.15) show the Concurrent and Serial operations along with the Close, Access Control and Resource control service requests. *(Note: A Close request can come any time(either from the Origin or Target) once a Z-association is established. Here for simplicity, we have shown a Close request when the system is in Concurrent or Serial Active mode.)*

5.4 Z39.50 Process Time Computations

From the Reachability Graph, it is possible to compute the time required to complete different Z39.50 processes. Various requests that may occur during these processes are indicated by Loop numbers. For example, Loop 2 shows Access Control request during initialization. Table 5.1 shows the interpretation of different Loops and delays associated with these Loops.

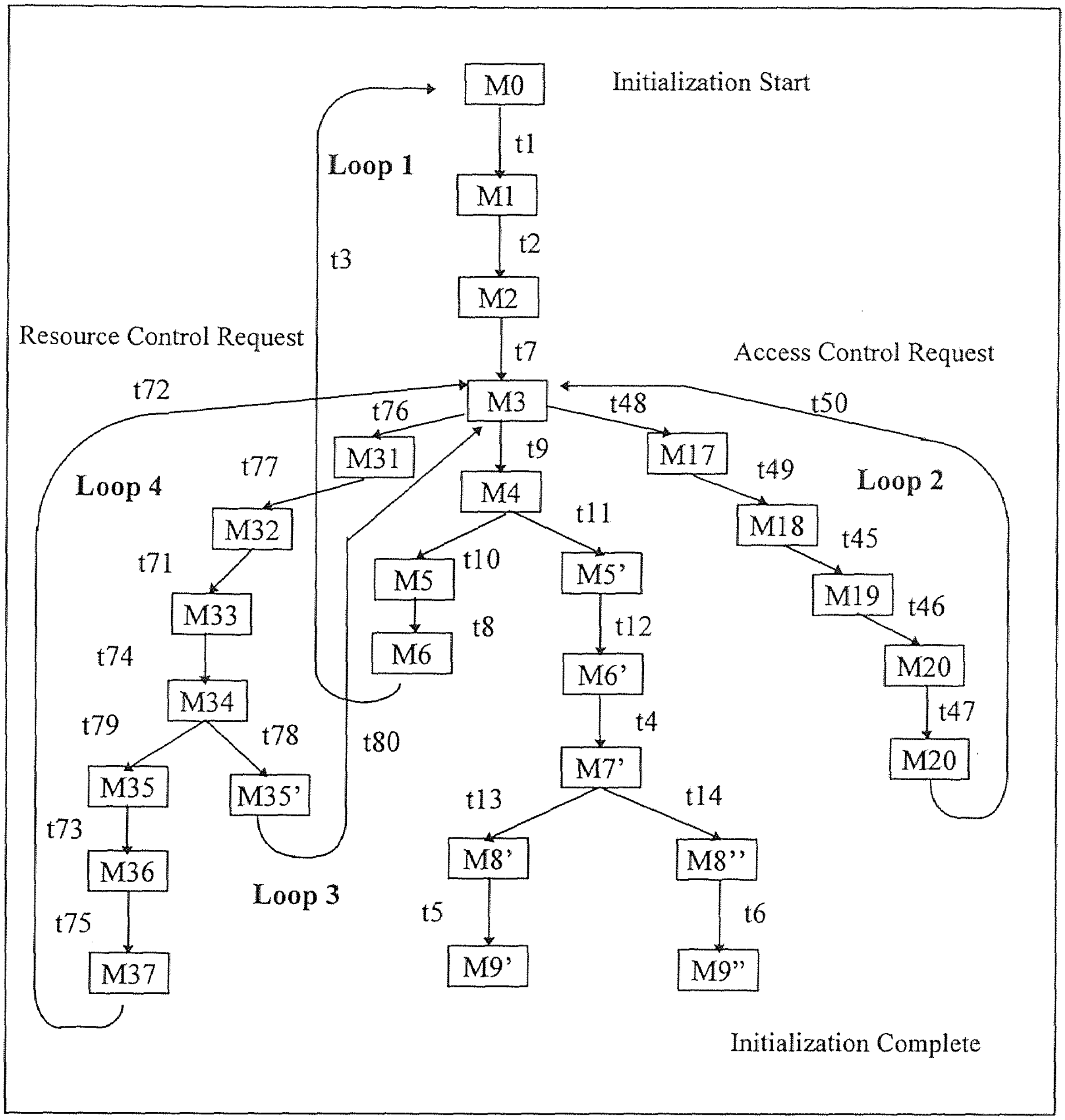


Figure 5.13 Reachability Graph for Z39.50 System Part 1

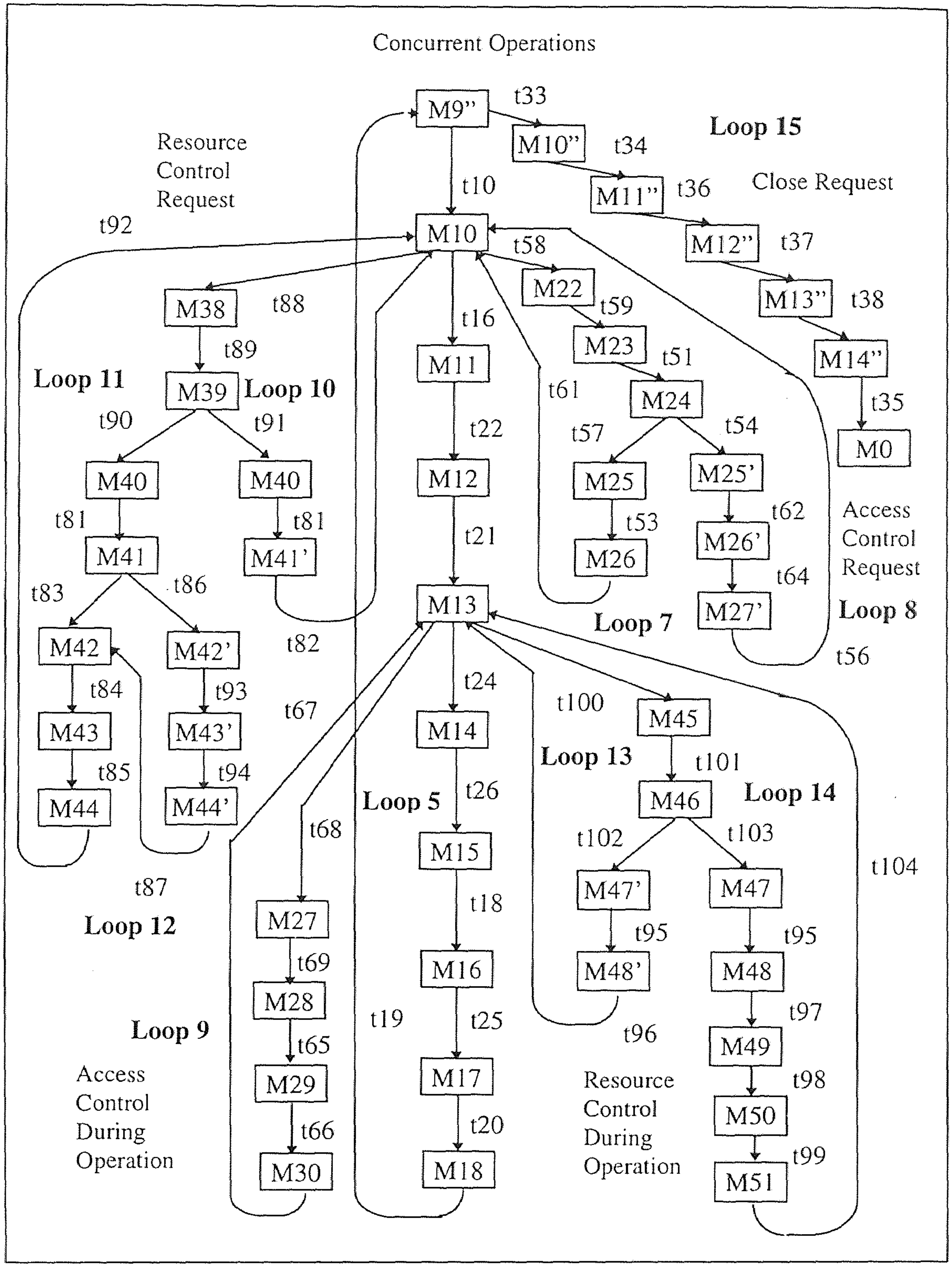


Figure 5.14 Reachability Graph for Z39.50 System Part 2

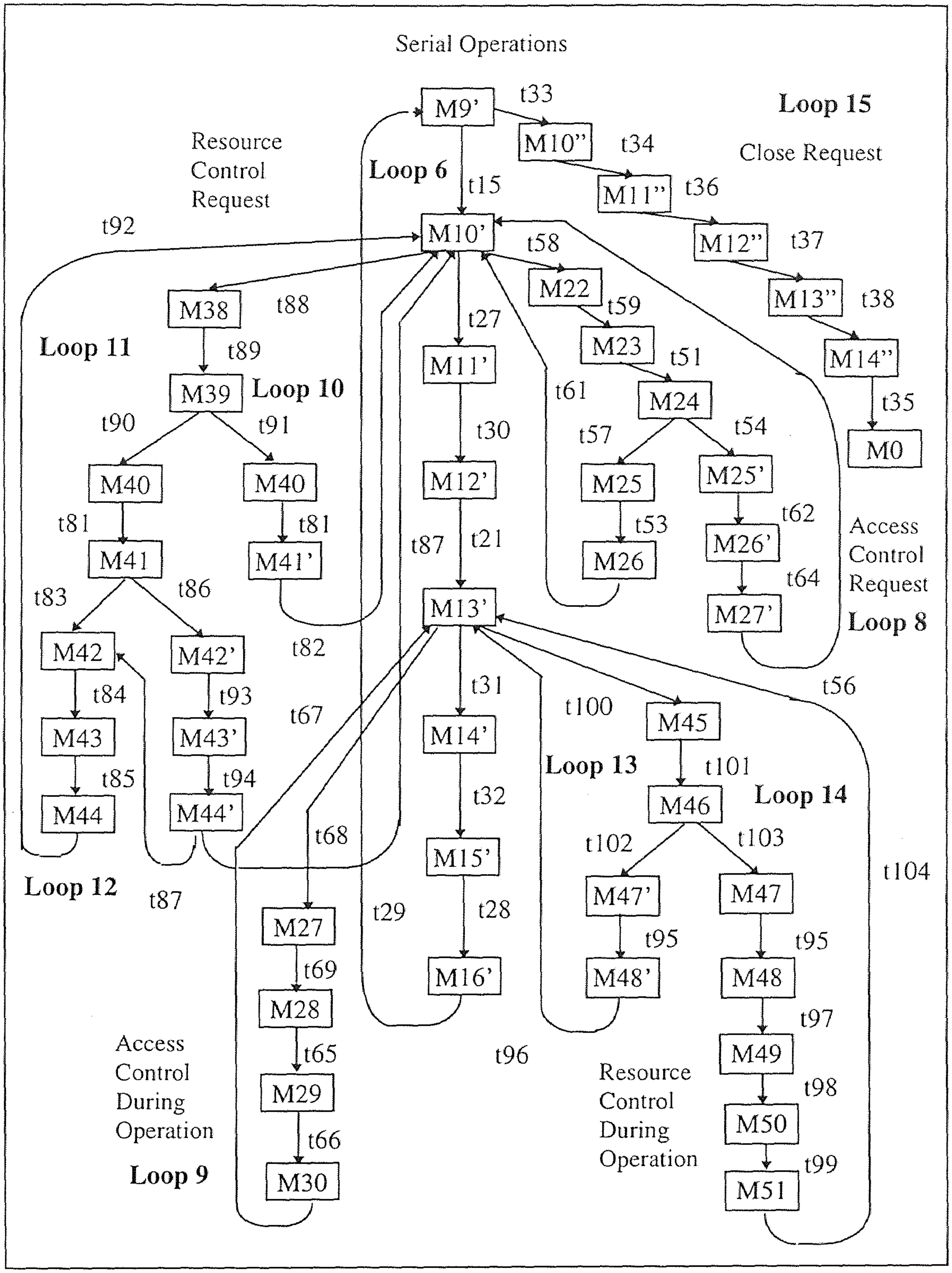


Figure 5.15 Reachability Graph for Z39.50 System Part 3

Table 5.1 Z39.50 Process Time Computations

Loop No.	Z39.50 Process	Time Required To Complete The Process
—	Initialization (Success)	t1 + t2 + t7 + t9+ t11 + t12 + t4 + t13(t14) + t5(t6)
Loop 1	Initialization (Failure)	t1 + t2 + t7 + t9 + t10 + t8 + t3
Loop 2	Access Control Request During Initialization	t48 + t49 + t45 + t46 + t47
Loop 3	Resource Control Request During Initialization (No Response Required)	t76 + t77 + t71 + t74 + t80
Loop 4	Resource Control Request During Initialization (Response Required)	t76 + t77 + t71 + t74 + 79 + t73 + t75 + t72
Loop 5	Concurrent Operation	t10 + t16 + t22 + t21 + t24 + t26 + t18 + t25 + t20 + t19
Loop 6	Serial Operation	t15 + t27 + t30 + t21 + t31 + t32 + t28 + t29
Loop 7	Access Control Request During Processing	t58 + t59 + t51 + t57 + t53 + t61
Loop 8	Access Control Request + Operation Request + End Operation Indication	t58 + t59 + t51 + t6 + t54 + t62 + t64 + t56
Loop 9	Access Control Request During Operation	t68 + t69 + t65 + t66 + t67
Loop 10	Resource Control Request During Processing (No Response Required)	t88 + t89 + t91 + t81 + t82
Loop 11	Resource Control Request During Processing (Response Required)	t88 + t89 + t90 + t81 + t83 + t84 + t85 + t92
Loop 12	Resource Control Request + Operation Request + End Operation Indication	t88 + t89 + t90 + t81 + t86 + t93 + t94 + t84 + t85 + t92
Loop 13	Resource Control Request During Operation (No Response Required)	t100 + t101 + t102 + t95 + t96
Loop 14	Resource Control Request During Operation (Response Required)	t100 + t101 + t103 + t95 + t97 + t98 + t99 + t104
Loop 15	Close Request	t33 + t34 + t36 + t37 + t38 + t35

5.5 Z39.50 Process Analysis

Initially, all the process analysis has been carried out by assuming the delay associated with each transition in Figure 5.13 - 5.15 to be one unit time. From Table 5.1 it is observed that the time required for successful initialization is 9 units

$$(t1 + t2 + t7 + t9 + t11 + t12 + t4 + t13(t14) + t5(t6))$$

If an Access Control or Resource Control request appears during the initialization phase, then it takes additional 5 to 8 units to complete the initialization process.

$$(t48 + t49 + t45 + t46 + t47) \quad \text{Access Control Request}$$

$$(t76 + t77 + t71 + t74 + t79 + t73 + t75 + t72) \quad \text{Resource Control Request}$$

Figure 5.16 graphically represents the effect of different services on initialization process. It is observed that with the Access Control and Resource Control services the initialization process time increases by around 50% to 80%.

When Concurrent operations are in effect, the time required to successfully complete the operation is 10 units. Depending upon the type of request, it can take 5 to 10 units time to complete that service as shown in the Figure 5.17. Hence the total time for the concurrent operation is increased by 50% to 100%.

When Serial operations are in effect, the time required for an operation is 8 units. The overhead due to Access Control, Resource Control and Close requests is 5 to 10 units time. Again, the overall time increases by 60% to 120%. Figure 5.18 shows the effect of Access Control, Resource Control and Close request on Serial operation.

5.6 Timed Petri Net Simulator

In the Reachability Graph, Loop 5 represents the cycle time for Concurrent operations. Here for simplicity, we have assumed only one operation request and calculated the time required to complete that operation. In reality, there can be multiple concurrent operation requests and responses at any given time. The Reachability Graph of a system becomes too complex to represent and analyze if multiple operation requests are present. Hence we have used a timed Petri net simulator TiPNet to compute the time required for successful completion of multiple concurrent operation requests.

TiPNet, developed at NJIT [8, 9, 10, 11, 12, 13] is a highly interactive graphical tool for drawing and simulating systems with stochastic as well as deterministic delays. With the help of a very user friendly menu, a Petri net can be drawn using TiPNet. A net can be verified for its semantic and syntactic correctness using the *verify* option. Delays can be specified for each transition and terminating condition for the net can also be specified. The net can be simulated using *step* or *run* simulation modes.

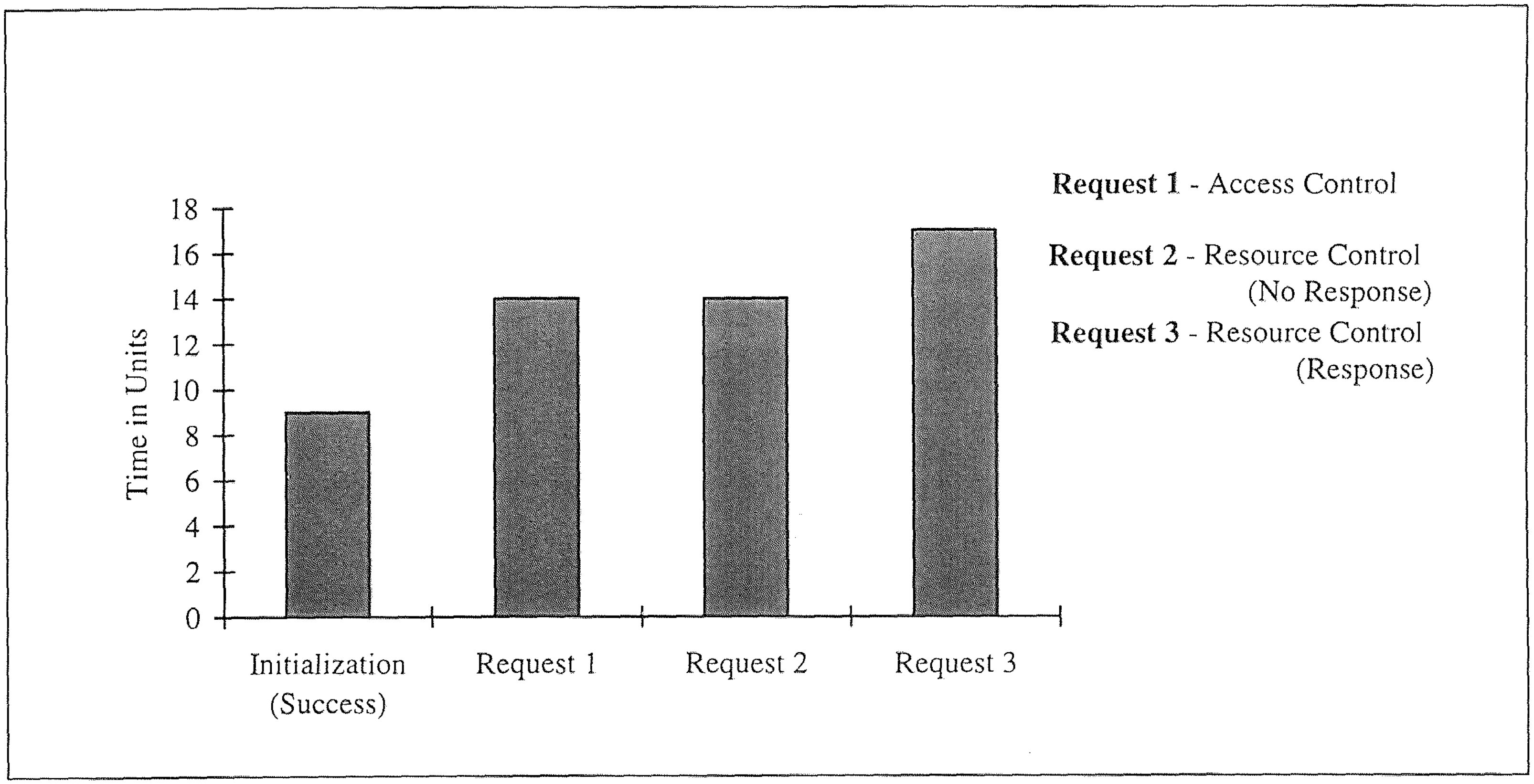


Figure 5.16 Effect of Z39.50 Services on Initialization

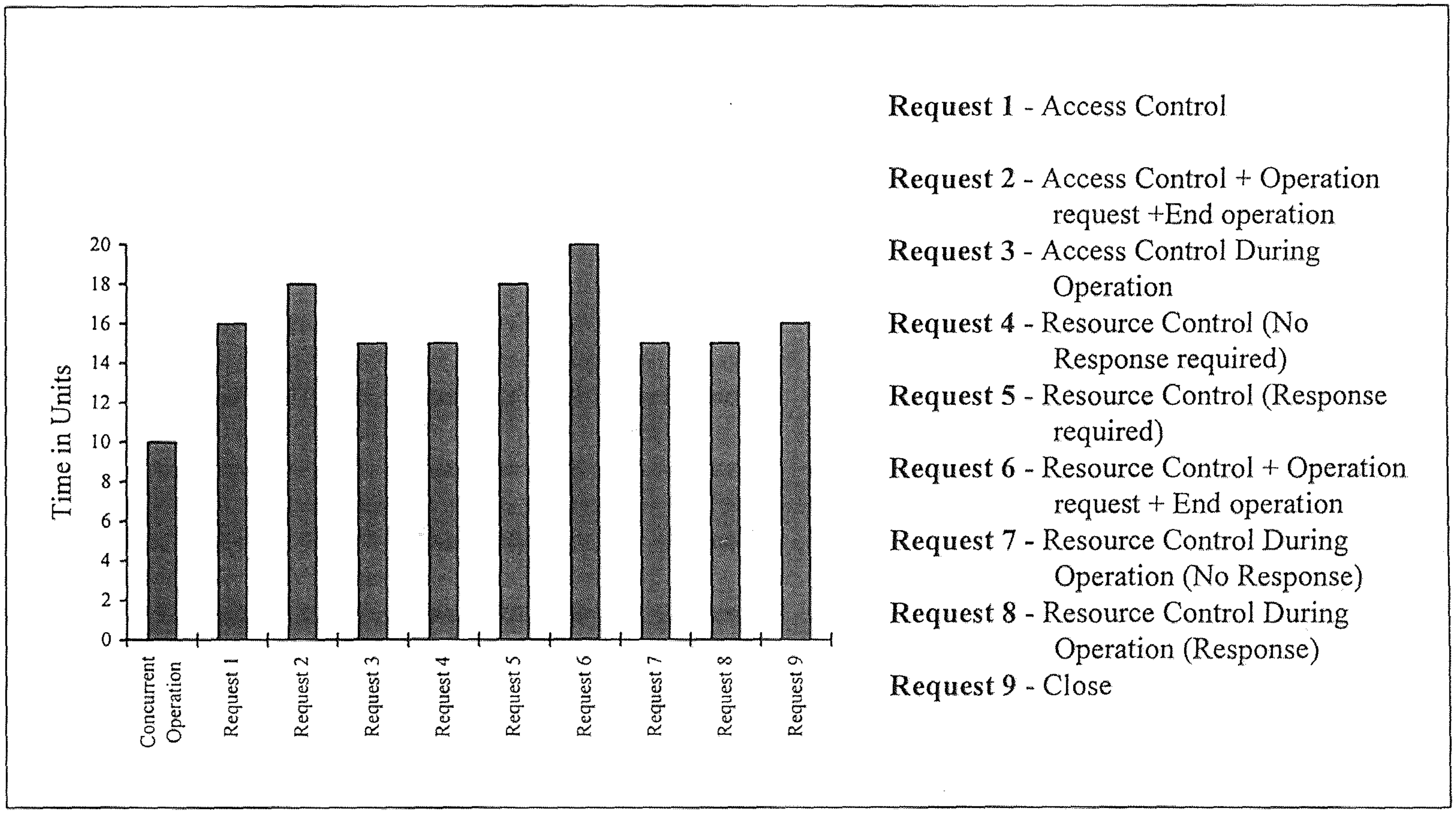
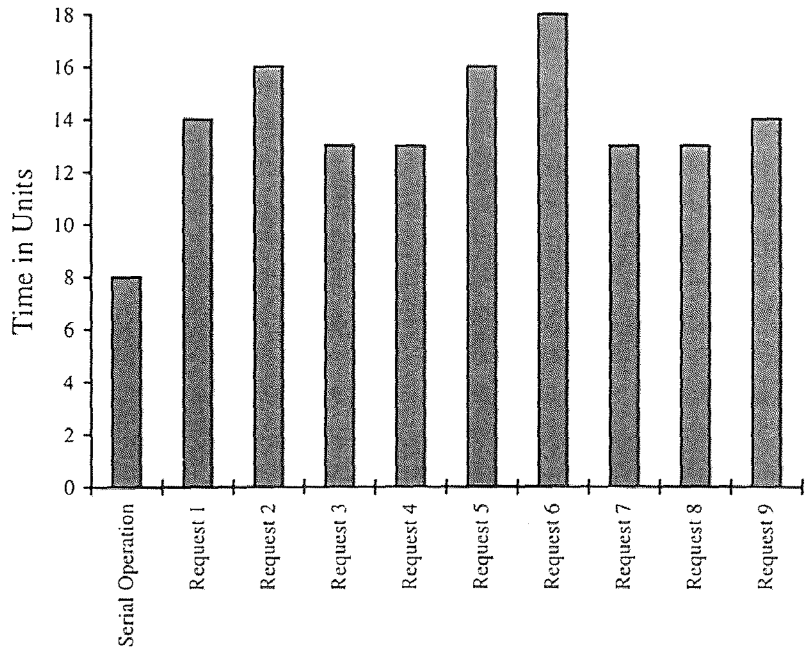


Figure 5.17 Effect of Z39.50 Services on Concurrent Operations



Request 1 - Access Control

Request 2 - Access Control + Operation request +End operation

Request 3 - Access Control During Operation

Request 4 - Resource Control (No Response required)

Request 5 - Resource Control (Response required)

Request 6 - Resource Control + Operation request + End operation

Request 7 - Resource Control During Operation (No Response)

Request 8 - Resource Control During Operation (Response)

Request 9 - Close

Figure 5.18 Effect of Z39.50 Services on Serial Operations

5.7 Concurrent Operations Simulation

TiPNet is used to simulate the behavior of Concurrent operations. One objective of the simulation is to examine the utilization of the timed transition. In TiPNet the percentage utilization is defined as:

$$\frac{\text{Total transition delay} \times 100}{\text{Total simulation time}}$$

The *log* file shows every step of simulation. It can be used to understand the system behavior. However for large number of steps, the *utilization report* file is more significant. The *report* file indicates the percentage utilization of timed transitions and percentage utilization of all the places [8].

A Petri net model for Concurrent operations is drawn and the net is verified for connectivity. We have associated the deterministic (constant firing time) delays with each transition

5.7.1 Simulation Results of Case 1

The objective of this simulation experiment is to find out the time required to process an individual operation request when the Origin and the Target are in the Concurrent operations mode. To analyze the Concurrent operations behavior the Petri net in Figure 4.2 (Processing Phase-Concurrent Operations) is simulated with initial marking (p9 5p21 p26 p29). The number of tokens in p21 is 5. The deterministic delay time of each transition is fixed at 1. The cycle time i.e. the time required to process an individual operation request is computed as:

$$\frac{\text{Total Simulation Time}}{\text{Total Number of Operation Requests}}$$

For fixed transition delays, the net is simulated for different operation requests and corresponding cycle times are computed. Table 5.2 shows the simulation results for various operation requests.

Table 5.2 indicates that the cycle time for the Concurrent operations increases the number of Concurrent operations request increase. Figure 5.19 is the graphical representation of the effect of increasing number of requests on the cycle time when the Origin and the Target are in the Concurrent operations mode.

Table 5.2 Operation Requests and Cycle Time

Operation Requests (Tokens in p21)	Cycle Time in Units
5	3.40
10	3.20
15	3.17
20	3.10
30	3.07
35	3.06
50	3.07
100	3.07
200	3.01

5.7.2 Simulation Results of Case 2

It can be seen from Figure 4.2 that transitions t18 and t23 play an important role in the overall performance of the net. Transition t23 is fired when an operation request is sent to the network and transition t18 is fired when an operation response is received from the network. All other transitions involved in the Concurrent operations process are local to either the Origin or the Target. The firing delays associated with the transitions t23 and t18 are mainly dependent on the underlying network parameters. Network capacity, network traffic and the data size to be transferred are some of the deciding factors for the delays of transitions t23 and t18.

In reality, the delays with t23 and t18 could range from nanoseconds to milliseconds. In order to bring our theoretical Petri net model closer to the real time situations, we have performed the following simulation experiment.

1. We have identified t23 and t18 as the 'critical' transitions. Here, the word 'critical' implies that the performance of these components need to be monitored.
2. The delay time with t23 is varied from 1 to 10 with all other transition delays fixed at 1. The number of operation requests (i.e. tokens at p23) is 5.
3. Step 2 is repeated for t18 = 5 and t18 = 10.
4. Steps 2 and 3 are repeated for tokens at p23 equal to 50.

Tables 5.3 and 5.4 show the observations for the above experiment.

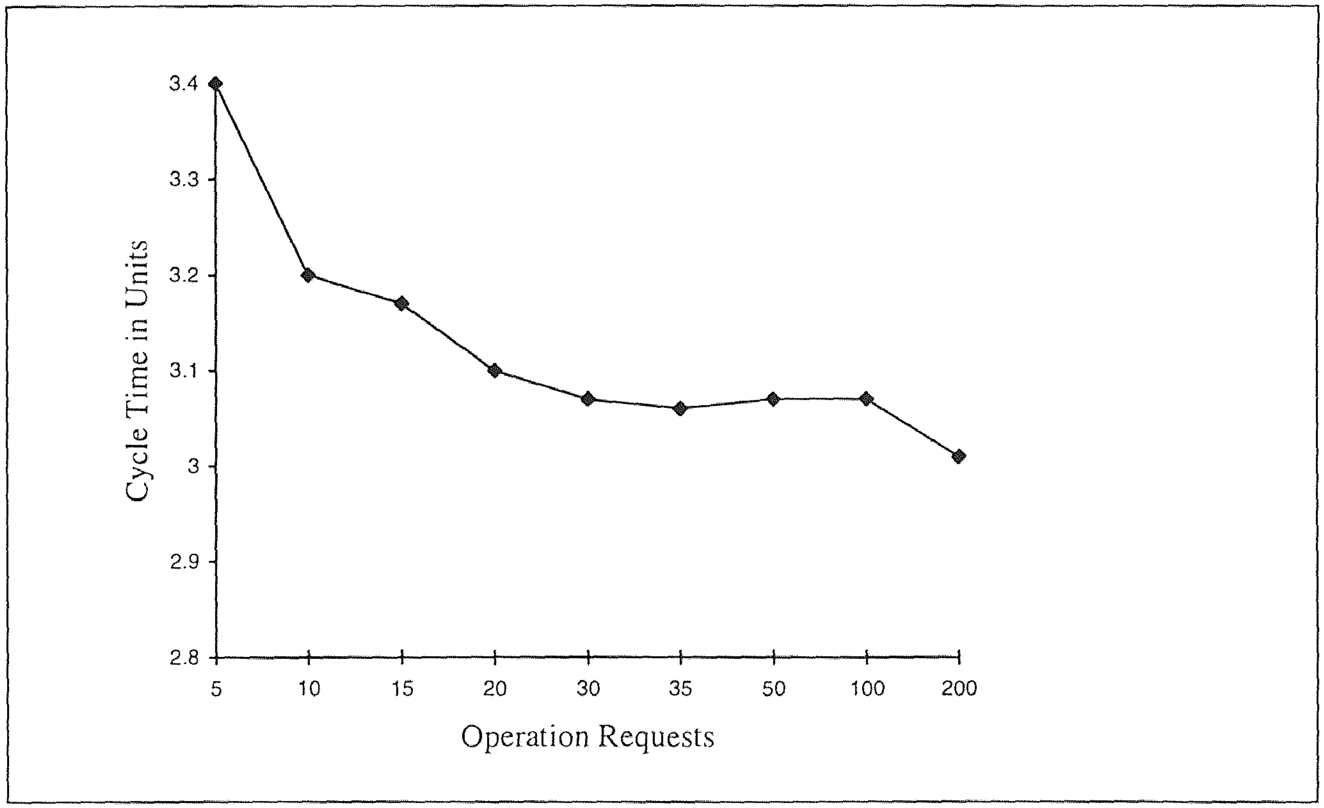


Figure 5.19 Cycle Time Versus Operation Requests (Transition Delay = 1 Unit)

Table 5.3 Effect of t23 Delay on Cycle Time (Operation Requests = 5)

Delay With T23 (in Units)	**Cycle Time (for t18 = 1 Unit)**	**Cycle Time (for t18 = 5 Units)**	**Cycle Time (for t18 = 10Units)**
1	3.4	7.4	12.4
2	3.6	7.4	12.4
3	4.6	7.4	12.4
4	5.0	7.4	12.4
5	5.8	7.4	12.4
6	6.4	7.4	12.4
7	7.6	8.6	12.4
8	8.4	9.2	12.4
9	9.0	10.0	12.4
10	9.8	10.8	12.4

Table 5.4 Effect of t23 Delay on Cycle Time (Operation Requests = 50)

Delay With T23 (In Units)	**Cycle Time (For T18 = 1 Unit)**	**Cycle Time (For T18 = 5 Units)**	**Cycle Time (For T18 = 10Units)**
1	3.04	7.04	12.04
2	3.04	7.04	12.04
3	4.00	7.04	12.04
4	5.00	7.04	12.04
5	5.98	7.04	12.04
6	6.80	7.04	12.04
7	7.94	8.04	12.04
8	8.80	8.80	12.04
9	9.90	9.62	12.04
10	10.40	10.64	12.04

Figures 5.20 and 5.21 represent the effect of t23 delay on cycle time of concurrent operations for five and fifty operation requests respectively. From these two figures it is observed that for higher delay associated with t18 (e.g. t18 = 10), the cycle time for an individual concurrent operation request remains constant for any variation of delay with transition t23 and any number of concurrent operation requests.

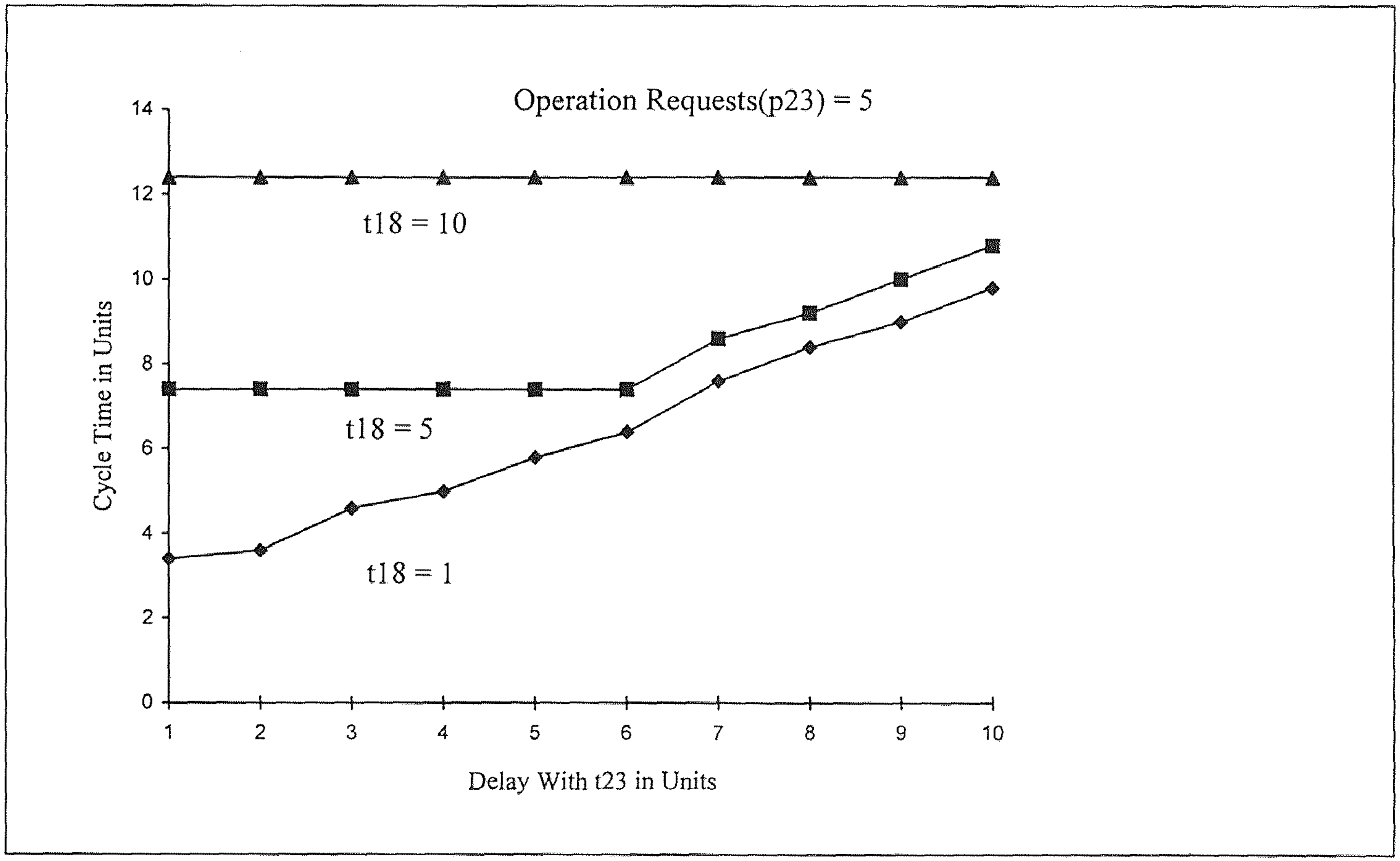


Figure 5.20 Effect of t23 Delay on Cycle Time (p23 = 5)

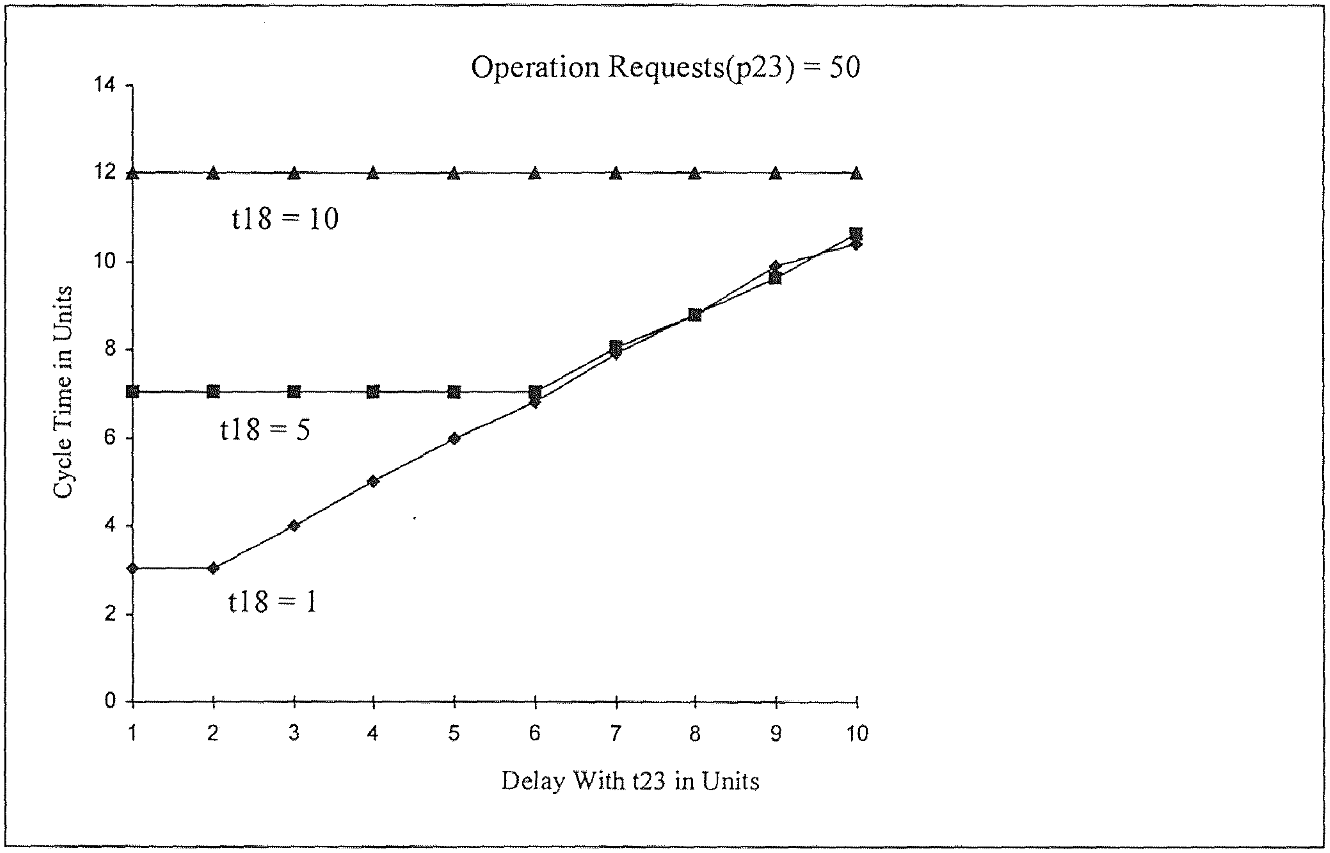


Figure 5.21 Effect of t23 Delay on Cycle Time (p23 = 50)

CHAPTER 6

CONCLUSION

6.1 Summary

The complexity of the Z39.50 state table was simplified using Petri net modeling methodology. We prepared a Petri net model for each and every sub-section of the system and thus a model for the entire system was developed. This model gives a clear understanding of the various services provided by this protocol. Using Reachability Graph, we have done a behavioral analysis of the Petri net model. The temporal analysis is done using timed Petri net simulator (TiPNet) developed at NJIT. The following observations were made based on our study and experiments:

1. Z39.50 protocol is capable of processing various operation requests simultaneously. When there are multiple operation in progress, we have found that the time required to process an individual request decreases as number of requests increases. We have observed a significant reduction in processing time when the number of operation requests vary from 5 to 50. As we further increase the number of requests, the net is saturated and there is no significant drop in the cycle time.
2. The transitions responsible for transferring the operation request to the network receiving the operation response from the net are identified as the critical components of the model. The cycle time gradually increases as the request time (delay associated

3. with t23) increases. The response time (delay associated with t18) plays a significant role in the operation cycle time. For a significantly high response time (high delay with t18), the variation in the request time (delay associated with t23) has almost no effect on the cycle time. This phenomenon is observed for small as well as large number of simultaneous operation requests.
4. The Access Control service and Resource Control service are the two distinguishable features provided by this protocol. We have found that these services contribute significantly towards the performance of Z39.50 processes. The overall process time can increase by 50% to 100% due to these services.
5. From the Reachability Graph for Concurrent operations, it can be stated theoretically that the time required to process an operation request is 9 units time (4.5 units for 2 requests). However, if the Access Control or Resource Control Request occur during the process of Concurrent Operations, the system has to pause the active operations and process these requests affecting the cycle time.

6.2 Future Work

We have installed ZPRISE (a Z39.50 client-server software) developed at National Institute of Standards and Technology (NIST). ZPRISE supports Z39.50 version 3. The Zclient installed on the local Sun operating system establishes a connection with the remote Zserver at NIST. One can get an adequate feel of Z39.50 services and operation using ZPRISE.

A theoretical model for Z39.50 protocol has been prepared and by performing additional experiments we have tried to simulate the behavior of the Z39.50 protocol. Using network monitoring packets such as etherman [13] or tcpdump [14], the actual time delays can be measured and these delays can be substituted in our model to find out the system performance.

6.3 Current Implementations

The Library of Congress (LOC) is the official maintenance agency for the Z39.50 standard. Z39.50-1995 (Version 3) has been successfully implemented over TCP/IP. Many major library software vendors like Ameritech have implemented the standard while others are in the process of producing Z39.50 compliant products. Implementers also include Government organizations such as LOC, National Library of Canada and The British Library. Academic institutions such as University of California, Berkeley, Carnegie Mellon University have also implemented Z39.50 protocols.

APPENDIX A

STATE TABLE DEFINITIONS

Origin States for Z-association:

Closed (0): The Origin is awaiting an initialization request (*Init req*) from the service user.

Init sent (1): The Origin is awaiting an initialization response protocol data unit (*Init resp PDU*) from the Target.

Acc recvd (2): During initialization the Origin has received an Access Control PDU (*Acc PDU*) and is awaiting an Access Control response (*Acc resp*) from the service user.

Rsc recvd (3): During initialization the Origin has received a Resource Control PDU (*Rsc PDU*) and is awaiting a Resource Control response (*Rsc resp*) from the service user.

Serial Idle (4): The Z-association is established, there are no active operations, and the Origin and the Target are in Serial operations mode.

Concurrent Idle (5): The Z-association is established, there are no active operations, and the Origin and the Target are in the Concurrent operations mode.

Serial Active (6): There is an active operation and the Serial operations mode is in effect.
Concurrent Active (7): There is at least one active operation, and Concurrent operations mode is in effect.

Z-Acc recvd (8): The Origin has received an *Acc PDU* pertaining to the Z-association and is awaiting an Access Control response from the service user.

Z-Rsc recvd (9): The Origin has received a *RscPDU* pertaining to the Z-association and is awaiting a Resource Control response from the service user.

Close sent (10): The Origin is awaiting a *Close PDU* from the Target.

Close recvd (11): The Origin is awaiting a close response (*Close resp*) from the service user.

Origin States for Operation:

Present/Op sent (1): The Origin is awaiting a Present response PDU (*Prsnt resp PDU*) from the Target. For operations other than Present, the Origin is awaiting an operation response PDU (*<op>resp PDU*) from the Target.

Rsc recvd (2): The Origin has received a *Rsc PDU* pertaining to an operation and is awaiting a Resource Control response form the service user.

Acc recvd (3): The Origin has received an Acc PDU and is awaiting an Access Control response from the service user.

Target States for Z-association:

Closed (0): The Target is awaiting an *Init PDU* from the Origin.

Init recvd (1): The Target is awaiting an *Init Response* from the service user.

Acc sent (2): During initialization the Target has sent an Access Control PDU (*Acc PDU*) and is awaiting an Access Control response PDU from the Origin.

Rsc sent (3): During initialization the Target has sent a Resource Control PDU (*Rsc PDU*) and is awaiting a Resource Control response (*Rsc resp*) from the Origin.
Serial Idle (4): The Z-association is established, there are no active operations, and the Origin and the Target are in Serial operations mode.

Concurrent Idle (5): The Z-association is established, there are no active operations, and the Origin and the Target are in the Concurrent operations mode.

Serial Active (6): There is an active operation and the Serial operations mode is in effect.

Concurrent Active (7): There is at least one active operation, and Concurrent operations mode is in effect.

Z-Acc sent (8): The Target has sent an *Acc PDU* pertaining to the Z-association and is awaiting an Access Control response PDU from the origin.

Z-Rsc sent (9): The Target has sent a Resource Control PDU (*Rsc PDU*) pertaining to the Z-association and is awaiting a Resource Control response PDU from the Origin.

Close sent (10): The Origin is awaiting a *Close PDU* from the Origin.

Close recvd (11): The Target is awaiting a Close response (*Close resp*) from the service user.

Target States for Operation:

Present/Op sent (1): The Target is awaiting a Present response from the service user. For operations other than Present, the Target is awaiting an operation response PDU (*<op>resp PDU*) from the service user.

Rsc sent (2): The Target has sent a *Rsc PDU* pertaining to an operation and is awaiting a Resource Control response form the Origin.

Acc recvd (3): The Target has sent an Acc PDU and is awaiting an Access Control response from the Origin.

REFERENCES

1. ANSI/ NISO Z39.50 - 1995, "Information Retrieval (Z39.50): Application Service Definition and Protocol Specification", NIST, MD 1995.

2. Michael and M. Hinnebusch, *From A to Z39.50 A Networking Primer,* Mecklermedia, Westport, CT, 1995.

3. A. Tanenbaum, *Computer Networks,* Prentice Hall, Englewood Cliffs, NJ, second ed., 1990.

4. M. Marsan, G. Balbo, G. Conte, S. Donatelli and G. Franceschinis, *Modeling With Generalized Stochastic Petri Nets*, John Wiley & Sons, New York, NY, 1995.

5. J. Peterson, *Petri Net Theory and the Modeling of Systems*, Prentice Hall, Englewood Cliffs, NJ, 1981.

6. M. Zhou, *Petri Nets in Flexible and Agile Automation*, Kluwer Academic Publishers, Norwell, MA 1995.

7. B. Liu, A. Robbi, "TiPNet: a Graphical Tool for Timed Petri Nets" *in Proceedings of the Sixth International Workshop on Petri Nets and Performance Models*, Durham, NC, pp. 212 - 213, October 1995.

8. B. Liu, "A Graphical Tool for the Simulation of Timed Petri Nets", Master's Thesis, Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ, 1994.

9. A. Gilani, "A Graphical Editor for Petri Nets", Master's Thesis, Electrical and Computer Engineering Department, New Jersey Institute of Technology, Newark, NJ, 1989.

10. A. Shukla, "A Petri Net Simulation Tool", Master's Thesis, Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ, 1990.

11. J. Siddiqi, "A Graphical Tool for the Simulation of Timed Petri Nets", Master's Thesis, Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ, 1991.

12. H. Juneja, "Object Oriented Design of Petri Net Simulator", Master's Thesis, Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ, 1993.

13. Williamson, *Packetman v1.2 Users Manual*, Curtin University of Technology, Perth, Australia, 1995.

14. W. Richard Stevens, *TCP/IP Illustrated Volume 1 The Protocols*, Addison-Wesley Professional Computing Series, Reading, MA, 1994.