DOCUMENT INTERCHANGE ARCHITECTURE: TRANSACTION PROGRAMMER'S GUIDE

SC23-0763-0

DOCUMENT INTERCHANGE ARCHITECTURE: TRANSACTION PROGRAMMER'S GUIDE

SC23-0763-0

First Edition (June 1983)

Changes are made periodically to the information herein; any such changes will be reported in subsequent revisions or Technical Newsletters.

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM program product in this publication is not intended to state or imply that only IBM's program product may be used. Any functionally equivalent program may be used instead.

Publications are not stocked at the address given below. Requests for IBM publications should be made to your IBM representative or to the IBM branch office serving your locality.

A form for readers' comments is provided at the back of this publication. If the form has been removed, comments may be addressed to IBM Corporation, Department D31, 11400 Burnet Road, Austin, Texas 78758. IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you.

PREFACE

The Document Interchange Architecture (DIA) provides document interchange capabilities across a broad spectrum of IBM office systems and is the communication architecture between DIA application programs. Specifically, DIA defines the protocols and data structures that enable programs to communicate processing intentions and interchange data. DIA logically divides into several parts: an information interchange base and various DIA application services. The strategic architecture for physically transporting data between products is the System Network Architecture (SNA) LU 6.2. This manual describes the use of the SNA LU 6.2 protocol boundary (LU 6.2_PB) by DIA application programs.

This manual is intended for data processing managers, system analysts, designers, system programmers, application programmers, as well as systems engineers and product support representatives.

PREREQUISITE PUBLICATIONS

- Document Interchange Architecture: Concepts and Structures, SC23-0759
- Systems Network Architecture: Transaction Programmer's Reference Manual for LU 6.2, GC30-3084-0.

RELATED PUBLICATIONS

- Office Information Architectures: Concepts, GC23-0765-0
- Document Interchange Architecture: Document Distribution Services Reference, SC23-0762
- Document Interchange Architecture: Document Library Services Reference, SC23-0760
- Document Interchange Architecture: Application Processing Services Reference, SC23-0761
- Document Interchange Architecture: Interchange Document Profile, SC23-0764
- Document Content Architecture: Revisable-Form-Text Reference, SC23-0758
- Document Content Architecture: Final-Form-Text Reference, SC23-0757.

CONTENTS

CHAPTER 1. INTRODUCTION

This document is intended for use by programmers responsible for designing DIA programs which use LU 6.2 for communications with other DIA programs in a SNA network.

In a SNA network, a DIA program is one of several programs called SNA Service Transaction Programs. The DIA program in LU 6.2 is known as the Document Interchange Model with reserved Transaction Program Name (TPN) of X'20F0'. The phrase DIA program or DIA transaction program is used in this document to be consistent with related documents and historical use. DIA program always means Document Interchange Model in LU 6.2.

This document describes the mapping of DIA sessions and transaction level command/reply logic to use of specific verbs (or verb sequences) at the LU 6.2 protocol boundary. It also defines the detailed rules which must be followed in all designs to insure a consistent use of the LU 6.2 protocol boundary across all DIA implementations.

WARNING: Failure to use these rules may result in logical lockout of communications or DIA program failures and are done at the risk of the implementer.

The information contained in this document is organized as follows:

- "Chapter 2. DIA Transaction Program Concepts" on page 3 presents an overview of DIA/LU 6.2 program relationships and highlights key concepts in DIA application program design.

- "Chapter 3. Definition of DIA Processing States" on page 9 provides a description of the DIA transaction program states and relates them to the conversation states of the LU 6.2.

- "Chapter 4. DIA Rules for Using LU 6.2" on page 13 specifies the rules which must be followed by all DIA/LU 6.2 designs. These rules are described in terms of program states and the use of specific verbs or verb sequences at the protocol boundary.

  A table is also provided in this section summarizing the allowable use of verbs and specific parameter values by DIA programs.

- "Chapter 5. DIA/LU 6.2 Protocol Machines" on page 21 provides a finite state machine description for enforcing the DIA/LU 6.2 relationship rules specified in this document.

- "Chapter 6. DIA/LU 6.2 Data Flows" on page 29 provides detailed scenarios for each of the commands defined by the DIA architecture. Each scenario is preceded by a state diagram showing the flow of control and data for that command.

CHAPTER 2. DIA TRANSACTION PROGRAM CONCEPTS

DIA is the architecture used to interchange requests and replies between two processes. It is the set of conventions which allows office system functions to be distributed across a variety of products. The specific functions of document distribution, library services, and application support are described in related documents. DIA is the orderly exchange of request and reply information that causes the transaction programs to perform these functions in a coordinated way. The request and reply information is exchanged in data stream constructs defined by DIA. These constructs, called Document Interchange Units (DIU), are defined independently from any product-specific application program interfaces (API) used by processes to send and receive DIU entities, and independently from specific communication facilities used to provide connectivity between the two DIA session partners.

DIA USE OF LU 6.2 SERVICES

LU 6.2 is the strategic IBM architecture for SNA process communication. It has been designed to satisfy the needs of a wide range of products from host or system-level products to device-level products. In its simplest terms, LU 6.2 is designed to facilitate synchronous transaction-program-to-transaction-program attachment, resource allocation, and conversational flows for as short or as long a period of time as desired by the programs. The protocols provide for information integrity across a multiresource scope and provide recovery design in the event of communication failure. These characteristics of a communication support system are required by DIA implementations and were assumed in the design of DIA.

LU 6.2 Transaction Programming Environment

Figure 1 on page 4 depicts the overall relationship of the DIA architecture to the architecture provided by LU 6.2, as seen from the DIA transaction programmer's point of view.

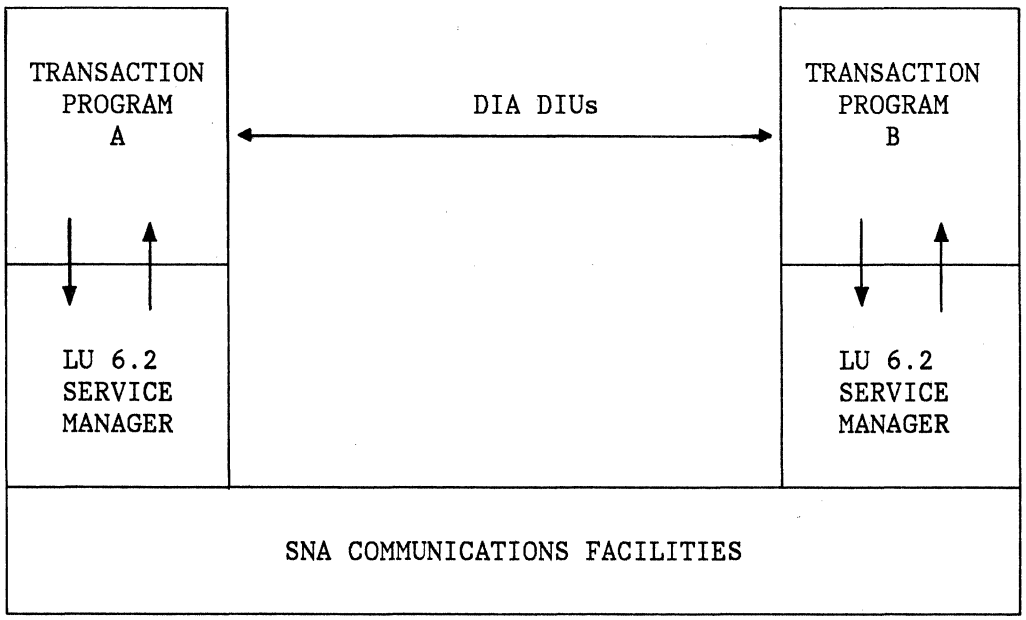


Figure 1. DIA/LU 6.2 System Relationships

Within a particular system, the DIA transaction program exploits system resources to perform functions. Some of these resources are local and outside the scope of LU 6.2. Other resources, particularly the LU 6.2 conversation with the remote DIA transaction program, are allocated by verbs and protocol of the LU 6.2 protocol boundary.

A particular instance of a DIA transaction program does not ordinarily have more than one LU 6.2 conversation. This conversation may contain only one request and reply or a long series of requests and replies. The resources required to perform these functions, such as library space and subordinate programs, are invoked by local interfaces.

The initial DIA transaction is stimulated through some interface to perform functions. These functions may require a conversation with a remote DIA transaction. The initial DIA transaction is the requestor of the LU 6.2 conversation or the session resource. When all information exchange is completed normally, it is the same initial DIA program which terminates use of the session resource. The concepts are illustrated in Figure 2 on page 5.

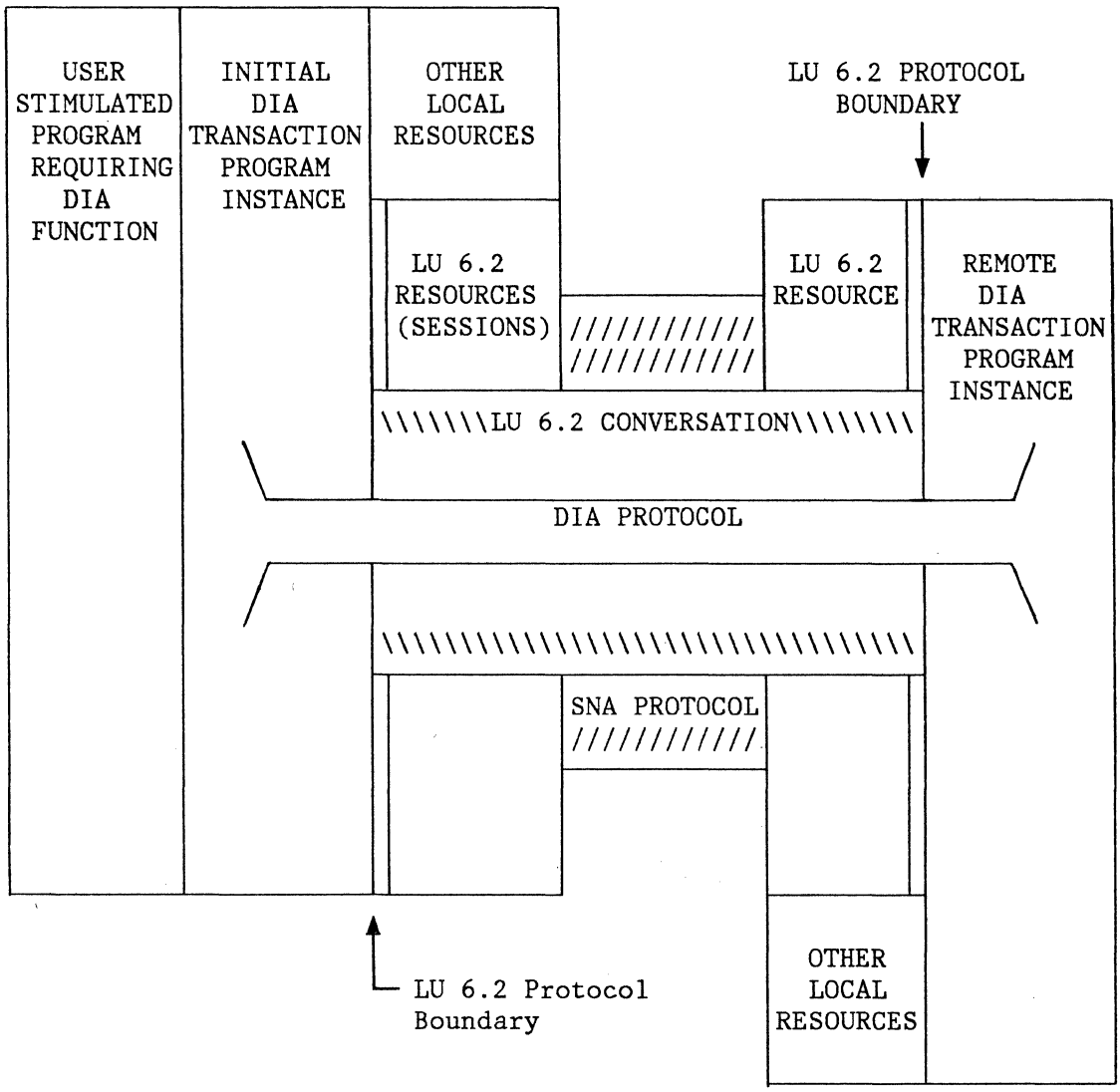


Figure 2. DIA/LU 6.2/Resource Relationships

LU 6.2 Transaction Programming Interface

The transaction programmer's interface to the LU 6.2 services is described in terms of verbs issued at the LU 6.2 protocol boundary. The verbs provide a semantic definition of the way that programs interact with SNA Services and are documented in Transaction Programmer's Reference Manual for LU 6.2. Products implementing LU 6.2 do not have to provide an interface which exactly matches the LU 6.2 protocol boundary. However, program communication interfaces which do exist in LU 6.2 products must be able to be mapped into the LU 6.2 protocol boundary.

In this document, the DIA transaction program rules for using the LU 6.2 protocol boundary are described in terms of the syntax and semantics defined in Transaction Programmer's Reference Manual for LU 6.2. All references in Transaction Programmer's Reference Manual for LU 6.2 regarding the treatment of

logical records may be equated to the handling of the DIU data stream components described below. This document assumes reader knowledge of the LU 6.2 conversation states and semantics defined in Transaction Programmer's Reference Manual for LU 6.2.

DIU STRUCTURE

The overall DIA Document Interchange Unit structure is shown in the figure below. A valid DIU consists of the following logical components: a DIU prefix, one or more DIA commands in a command sequence, data units, document units, and a DIU suffix. Each DIU component (or subcomponent) is a self-defining, variable length structured field that has been assigned a registered 2-byte identifier (ID). (See DIA: Concepts and Structures). The syntax and semantics of structured fields used in DIA are owned, maintained, and enforced for cross product use by IBM.

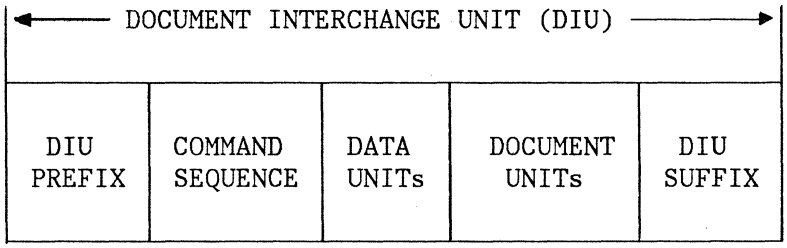


Figure 3. DIA Document Interchange Unit

DIU Component Definitions/Usage

The DIU prefix is used as a data stream delimiter which begins and identifies the DIU. The DIU-ID field of the prefix is used as a correlation value which permits a DIU sent in reply to be correlated to this DIU.

The command sequence portion of the DIU contains one or more DIA commands which identify the functions to be performed. Each command directs the process to which the DIU is sent to perform some function or to act upon the data in a specific manner. Execution of the commands in the command sequence is the responsibility of the receiver of the DIU. The operations requested by the commands must be performed in the order in which the commands appear. The meaning of each command and its required replies, if any, are specified by DIA.

Data units contain information that is referenced by one or more commands in the current command sequence. Data units may be thought of as indirect or factored command operand values.

Document units carry the information which is the object for one or more command operations specified in the command sequence. The ID fields of this data stream component indicate that this is a document unit construct and denote the type of document unit.

The DIU suffix delimits the end of the DIU data stream. The type of suffix used determines the disposition of the data stream as follows:

- DIU suffix Type 1 indicates normal termination of the DIU data stream.

- DIU suffix Type 2 indicates that the sending application of the DIU encountered an unrecoverable DIA exception condition during the sending of the DIU. The DIU data is thus invalid and should be purged by the receiver of the DIU.

  NOTE: It should be remembered that the cause, scope, and recovery procedures for DIA exceptions are independent of LU 6.2 ABEND conditions or return codes. These two distinct levels of error conditions should not be confused.

The reader should refer to the DIA: Concepts and Structures for a detailed description of DIA data stream component semantics and syntax.

DIA COMMAND/REPLY PROTOCOLS

The essence of DIA is the set of commands which define the functions to be performed when they are exchanged between DIA applications. Architecturally, the DIU may contain one or more commands contained within a command sequence. Some implementation versions constrain the command sequence to only one command.

There are two classes of commands for which distinct reply protocols are defined for DIA.

No-Reply-Required Command Class (NRR)

This command class is defined to let one DIA process convey information to another DIA process without requiring a reply to the command.

Synchronous-Reply-Required Command Class (SRR)

This command class is defined to let one application request that another perform a function with that application necessarily replying to the request before any other interchange of requests is permitted.

Asynchronous-Reply-Required Command Class (ARR)

This command class is defined to let one application request that another perform a function when the reply to the request is not required before any other interchange of requests is permitted.

A replying command is a command that contains the CORRELATION operand. This operand indicates that the command is a reply to a previous SRR command and provides the data necessary to correlate to that reply required command.

DIA EXCEPTION HANDLING

The objective of DIA is for two application processes to exchange information in a reliable manner. The processes must be capable of recognizing when a Document Interchange Unit (DIU) is received with errors that are detected at a lower level such as the SNA communication facility. DIA is dependent on the fact that

the layers of support below the application level will notify the application process when a permanent error exists. This lower level includes the SNA layers for link level control and network services. SNA defines the alternatives for handling error conditions at that level. DIA assumes that all error procedures have been attempted by the component layers below the application programs and that unrecoverable errors are presented to the process application for final disposition.

The damage assessment process for a DIU with a permanent error is to interpret and evaluate the DIU entities to determine if the request can be successfully executed. If exceptions are detected and all of the recovery techniques have been unsuccessful, then the DIU cannot be reliably processed and the receiving process must terminate DIU processing.

The DIU analysis process consists of algorithms that evaluate the DIU syntax to determine which functions of the DIA repertoire are to be performed. This evaluation may discover syntax and semantic exceptions. The syntax exceptions are deviations from the defined structure formats, missing parameters, parameter values outside permitted ranges, and DIU entities incorrectly encoded. Semantic exceptions are usually detected when a requested operation is activated and some violation prevents successful completion. There is also a distinction between errors and exceptions. Errors are those failures that occur at the program or systems level. Exceptions are violations of the architecture that are detected in a DIU without, necessarily, an indication of a failure by a system or application component.

The syntax and semantic exceptions that prevent a requested DIA application process from being normally completed must be reported to the sender of the DIU. The ACKNOWLEDGE NRR command with an EXCEPTION CODE operand is used to report exception conditions detected by the receiver of a DIU. If there is an exception condition detected in the DIU containing the acknowledgement of the sender's request, then the DIU processing is terminated, the detected exception is sent by ACKNOWLEDGE, and a SIGN-OFF command is sent. Exceptions that are detected by a sender that prevent successful transportation of a complete DIU are specified in a form of the DIU suffix.

If an exception condition which prevents execution of the DIU function occurs while a DIU is being received (local LU 6.2 is in receive state), the ACKNOWLEDGE with exception code may be sent immediately, using the LU 6.2 SEND-ERROR verb to indicate the error condition, the LU 6.2 SEND-DATA to send the ACKNOWLEDGE data, and a RECEIVE_AND_WAIT to be ready for the next recovery step.

CHAPTER 3. DEFINITION OF DIA PROCESSING STATES

The dialogue that takes place between two DIA transaction programs may be described in terms of processing states. These states are updated during the processing of DIU command sequences and apply to all DIA function sets.

A DIA session is a DIA unit of work which lasts from DIA SIGN-ON to DIA SIGN-OFF. A DIA session exists on a LU 6.2 conversation. An LU 6.2 conversation is an instance of use of a LU 6.2 session. Thus, the LU 6.2 session is a resource used to support the DIA session.

The DIA processing states may be divided into two major categories:

- Session states which relate to establishing and terminating the DIA session.
- Command states which deal with managing the DIA request/reply command flows during an active DIA session. The states in each of these categories are described below.

DIA SESSION STATES

The following convention is used in these definitions. The symbols <name> specify the state name.

RESET:
<RESET>

A conversation between two DIA processes is not allocated and no DIA session exists. This is the initial program state of a DIA process.

NOT_ACTIVE:
<NOTACT>

A conversation between two DIA processes is allocated, but no DIA session exists or is pending. The DIA partner which allocated the conversation may now send SIGN-ON.

PENDING:
<PEND>

A conversation between two DIA processes is allocated and DIA SIGN-ON SRR has been sent or received. The DIA session is pending.

SONR_SENT:
<SONS>

A conversation between two DIA processes is allocated. A SIGN-ON NRR reply has been sent to indicate acceptance of a SIGN-ON request or a SIGN-ON NRR request (allowed only for migration by existing DIA implementation to effect mid-session change of

process roles) has been sent to change DIA session parameters. The process is in RECEIVE state waiting for a DIU to start the DIA session flow.

This state applies only to the process receiving an initial SIGN-ON request, or the process sending a mid-session SIGN-ON NRR request to change DIA session options.

IDLE_ACTIVE:
<IDLE>

A conversation between two DIA processes is allocated and the DIA session is established. The processes at either end may send or receive commands, depending upon the roles defined in the SIGN-ON command.

QUIET:
<QUIET>

The DIA session parameters are preserved outside of the DIA transaction program and a DEALLOCATE verb has been issued. No DIA activity is possible until an ALLOCATE verb from one DIA program generates a LU 6.2 conversation with a DIA program equivalent to its former remote partner. This is possible in two special cases. Case 1: Connection to an equivalent instance is possible when there is only one possible DIA transaction program instance in the remote LU. Case 2: Connection to an equivalent instance is possible if there is only one possible LU 6.2 session from the local to the remote LU; and when that session is used for conversation between DIA programs, the equivalent DIA programs are always evoked. This implies the recipient of the ALLOCATE generated LU 6.2 protocol must make the original DIA session parameters available to the newly allocated DIA program and the program uses these to put itself in the IDLE-ACTIVE state. Case 2 is known as the "single session assumption".

DIA REQUEST/REPLY COMMAND STATUS

SRR_SENT:
<SRRS>

The process has sent an SRR request and is awaiting a reply.

SRR_SENT & :
SRR-RECEIVED
<SRRSR>

The process has sent an SRR request and has received an SRR replying command. The process must now send a reply to the SRR reply received.

SRR_RECEIVED:
<SRRR>

The process has received an SRR request and must provide a reply as the next command sent.

SRR_RECEIVED:
&SRR_SENT
<SRRRS>

The process has received an SRR request and has returned an SRR replying command. The process is awaiting a reply to the SRR reply sent.

CHAPTER 4. DIA RULES FOR USING LU 6.2

This section details the rules which must be followed by DIA implementations using the LU 6.2 protocol boundary. The rules are described in terms of DIA states, DIA commands and command classes, and the specific verbs or verb sequences issued at the LU 6.2 protocol boundary.

A table is also provided summarizing the allowable use of verbs and parameter values which may be used by DIA applications, along with the ground rules for using optional verbs, if any, provided by the local LU 6.2 Service Manager.

RULES FOR USING ALLOCATE/DEALLOCATE

DIA uses the ALLOCATE/DEALLOCATE verbs to establish and terminate conversations with remote DIA transaction programs. The points within the DIA session where this may occur are defined in terms of DIA states.

A conversation must be allocated prior to sending a SIGN-ON command requesting a DIA session and may be deallocated when the DIA session is concluded. The conversation may also be deallocated within the DIA session when one of the two session partners wishes to enter a state of inactivity. This state is called the QUIET state in DIA.

The QUIET state is used by implementations for a variety of purposes which are independent of the semantics of the DIA architecture. For example, a work station product may choose to enter the QUIET state to accommodate a swapping of END USER interface programs prior to resuming DIA activity. A host DIA program may wish to enter the QUIET state to allow swapping or rollout of the program to occur for sharing of the core resources with other programs.

When the DIA session is in QUIET state, the conversation must be reallocated before the DIA session can be resumed. This is accomplished by issuing ALLOCATE to establish the conversation. As explained above in the definition of the QUIET state, it is the responsibility of the DIA session partners to correlate this conversation to an already existing DIA session.

The following rules apply to DIA's use of ALLOCATE/DEALLOCATE:

- ALLOCATE is issued in the DIA RESET state to establish a conversation for a new DIA session. The TPN value specified must be X'20F0F0F0', (DIA), and the TYPE specification must be CONVERSATION.

- The DEALLOCATE verb is issued in the DIA NOT_ACTIVE state to release the conversation at the conclusion of the DIA session. The TYPE specified when DEALLOCATE is issued in this state should be SYNC_LEVEL.

- DEALLOCATE is issued in the DIA IDLE_ACTIVE state when the program wishes to enter the DIA QUIET state. The TYPE parameter value used when DEALLOCATE is issued in this state should be SYNC_LEVEL.

- ALLOCATE is issued in the DIA QUIET state prior to resuming DIA activity. The LU_NAME, MODE_NAME will be used as a way of correlating conversations for implementations which make the single session assumption.

- DEALLOCATE is also issued by DIA exception handling processes when unrecoverable process errors are detected. (See "Rules for Handling Exception Conditions" on page 15.)

FLOW CONTROL RULES

The rules for controlling the DIA command flows in LU 6.2 implementations are summarized below.

1. The process which allocates the conversation is the DIA process which sends the SIGN-ON request. The SIGN-ON request must be sent in the SRR command class using SEND_DATA and must be followed by issuing RECEIVE_AND_WAIT.

2. If the SIGN-ON request is to be rejected, the receiver of the command must issue a SEND_ERROR verb followed by a SEND_DATA verb which sends an ACKNOWLEDGE reply with exception codes which describe the reason for rejection. There are two cases for the SIGN-ON sender to consider:

   a. If the ACKNOWLEDGE reply is sent followed by a DEALLOCATE TYPE (SYNC_LEVEL), the sender of SIGN-ON must deallocate the conversation and enter the DIA RESET state prior to initiating any new requests for session.

   b. If the ACKNOWLEDGE reply is sent followed by RECEIVE_AND_WAIT, the sender of SIGN-ON may either resend SIGN-ON with new DIA Session options, or deallocate the conversation and enter the DIA RESET state.

   In the latter case, the remote program is notified of a normal deallocation through the return code of RECEIVE_AND_WAIT. The remote program should then issue DEALLOCATE TYPE (LOCAL) and enter the DIA RESET state.

3. If the DIA SIGN-ON request is accepted, the receiver of the command must send a SIGN-ON NRR reply. This should be followed by RECEIVE_AND_WAIT in order to provide the sender of SIGN-ON request with the first opportunity to send commands within the session.

   Upon receiving the SIGN-ON NRR reply, the SIGN-ON sender may relinquish first send control to his session partner, if desired, by issuing RECEIVE_AND_WAIT in the DIA IDLE_ACTIVE state. This invites the remote DIA Session partner to send the first command in the session. The DIA SIGN-ON request sender is given first opportunity to send commands on the assumption that the SIGN-ON request sender has the best idea of what work needs to be done on this session.

4. The receiver of DIA commands may request to be the command sender by issuing the REQUEST_TO_SEND verb. This verb may only be issued in the DIA IDLE_ACTIVE state.

Upon receiving notification of the REQUEST_TO_SEND, the local program may grant control to the remote program by issuing RECEIVE_AND_WAIT in the DIA IDLE_ACTIVE state. The local program, however, may elect to retain control and continue to be the command sender.

5. DIA command sequences, other than those allowed in the DIA/LU 6.2 Protocol Machines, are in violation of mapping rules. The process detecting an invalid command sequence will issue SEND-ERROR followed by DEALLOCATE TYPE (ABEND). See the section entitled DIA/LU 6.2 Data Flows.

6. Either process may enter the DIA QUIET state by issuing DEALLOCATE in the DIA IDLE_ACTIVE state.

7. Either process, with something to send, may cause a session which is in the QUIET state to be resumed by issuing the ALLOCATE verb. The DIA session is resumed with the process issuing ALLOCATE in the DIA IDLE_ACTIVE state and the other partner in the DIA IDLE_ACTIVE state.

RULES FOR HANDLING EXCEPTION CONDITIONS

Exception conditions and corresponding recovery actions are defined by the DIA architecture. The following rules apply to how these exception conditions are reported using the verbs of the protocol boundary. There are three general cases to consider:

- Exception conditions detected by the DIU receiver
- Exception conditions detected by the DIU sender
- Process or program exception conditions detected by either program partner.

The actions required in each of these cases are defined below.

- Exception conditions detected by the DIU receiver are reported to the sender of the DIU by issuing a SEND_ERROR verb followed by a SEND_DATA verb which sends an ACKNOWLEDGE replying command with exception condition codes describing the exception conditions.

  The sender is notified that an exception condition has been detected through the RETURN_CODE parameter of the LU 6.2 verbs. The program must then issue RECEIVE_AND_WAIT to obtain the information about the exception condition. The Correlation operand on ACKNOWLEDGE is used to correlate the exception condition data to a specific DIU sent.

- Exception conditions detected by the DIU sender during the sending of DIU information are reported to the receiver by terminating the data stream being sent with a DIU SUFFIX type 2. The SUFFIX carries information concerning the type of error detected.

  If a DIA logical record which contains an exception condition has been partially sent, then action must be taken to insure that the remote program is able to scan the data stream. This may be done in two ways. DIU receivers will support both methods.

The first method is to complete the sending of the DIA entity in error prior to sending the DIU SUFFIX logical record. The second method is to issue a SEND_ERROR verb, followed by a SEND_DATA verb which sends the DIU SUFFIX. In the second method, the truncation handling protocols of the LU 6.2 verbs are used to insure the scan of the data stream by the DIU receiver.

- Catastrophic process or program errors may be detected during the execution of either program partner. These errors usually indicate errors in program logic and are not recoverable. When a DIA program detects a catastrophic error of this type, it attempts to gracefully terminate the conversation by sending a SEND_ERROR prior to issuing DEALLOCATE TYPE (ABEND).

RULES FOR USING SYNCHRONIZATION PROTOCOLS

DIA/LU 6.2 implementations are required to support the synchronization protocols described in this section. In particular, programs may issue DEALLOCATE (SYNC_LEVEL) to synchronize the ending of a DIA/LU 6.2 conversation; programs will respond properly to CONFIRM requests presented at the protocol boundary via the WHAT_RECEIVED parameter returned on RECEIVE_AND_WAIT.

DIA applications are also allowed, but not required, to issue the CONFIRM verb at the conclusion of each request/reply command sequence if so desired.

The following rules apply to DIA's use of SYNC_LEVEL protocols:

- DEALLOCATE TYPE (SYNC_LEVEL) should be issued by a DIA process prior to entering the QUIET state. This verb may only be issued when the process is in the DIA IDLE_ACTIVE_SEND state.

  The purpose of this rule is to insure that the last command or reply sent has been received and processed correctly and that the application states of both programs are consistent before the conversation is allowed to terminate.

  A SEND_ERROR, SEND_DATA of an ACKNOWLEDGE command data flow will be returned if the last DIA command or reply sent was found to be in error.

- The SYNC_LEVEL operand must be specified when the DEALLOCATE verb is issued following the sending of the DIA SIGN_OFF command.

  The purpose of this rule is to insure that the remote program has received and processed the SIGN_OFF and has reset to DIA Not_Active state conditions prior to the termination of the conversation.

  A return code of ok indicates that the remote program has been reset to the DIA Not_Active state and has issued CONFIRMED. A SEND_ERROR, SEND_DATA of an ACKNOWLEDGE command data flow will be returned if a catastrophic exception condition has been detected in the processing of the SIGNOFF DIU.

- CONFIRM may only be issued when the DIA program is in IDLE_ACTIVE state. The smallest logical unit of work which can be synchronized within a DIA session is a request/reply command sequence operation. A request/reply

command sequence operation includes the processing of the command request and return of any required synchronous replies.

- Programs receiving a CONFIRM request are required to complete the processing of all DIU data received. If exception conditions exist in the data, they are detected and reported by SEND_ERROR, SEND_DATA of an ACKNOWLEDGE (EC) data flows as part of the normal DIA application processing of DIU's.

  If there are no outstanding synchronous replies required after all DIU data received has been processed, then the program responds to the CONFIRM request by issuing CONFIRMED.

  If there are outstanding synchronous replies required to conclude a command request, then the command operation cannot be confirmed. The program will respond by issuing SEND_ERROR followed by DEALLOCATE TYPE (ABEND).

DIA USE OF VERBS AND PARAMETERS

The tables in this section show the verbs and parameter values which DIA programs may use in LU 6.2 implementations. The Transaction Programmer's Reference Manual for LU 6.2 contains the detailed description of the LU 6.2 verbs and parameters. The value specified in each parameter is the DIA value prescribed for support on the office system base.

Optional Support Requirements

DIA implementations are permitted to take advantage of performance verbs offered by the local LU 6.2 Service Manager only if the resulting LU 6.2 internal flows to the remote program are identically equivalent in every way to the usage of verbs on the LU 6.2 base.

DIA implementations are also permitted to use other locally available LU 6.2 Service Manager options which do not have a session visible result, that is, are totally transparent to the remote program partner.

The use of local LU 6.2 Service options, as described above, is a product design decision.

Verb and Parameter	Comments
ALLOCATE RETURN_CODE (variable name) RESOURCE (variable name) LU NAME (OTHER,*) TPN (X'20F0') MODE_NAME (other than SNAVCMG) RETURN_CONTROL (WHEN_SESSION_ALLOCATED) SYNC_LEVEL (CONFIRM)	The value placed in the variable name supplied in RESOURCE is needed in all other commands which refer to this RESOURCE. The same variable name should therefore be provided. DIA TPN
GET_ATTRIBUTES RESOURCE (variable name) PARTNER_FULLY_QUALIFIED_LU_NAME (variable name) MODE_NAME (variable name) SYNC_LEVEL (variable name)	Values filled as a result of execution.
DEALLOCATE RETURN_CODE (variable name) RESOURCE (variable name) TYPE (SYNC_LEVEL) TYPE (LOCAL) TYPE (ABEND)	
CONFIRM RETURN_CODE (variable name) RESOURCE (variable name) REQUEST_TO_SEND_RECEIVED (variable name)	
CONFIRMED RESOURCE (variable name)	Issued in response to CONFIRM.

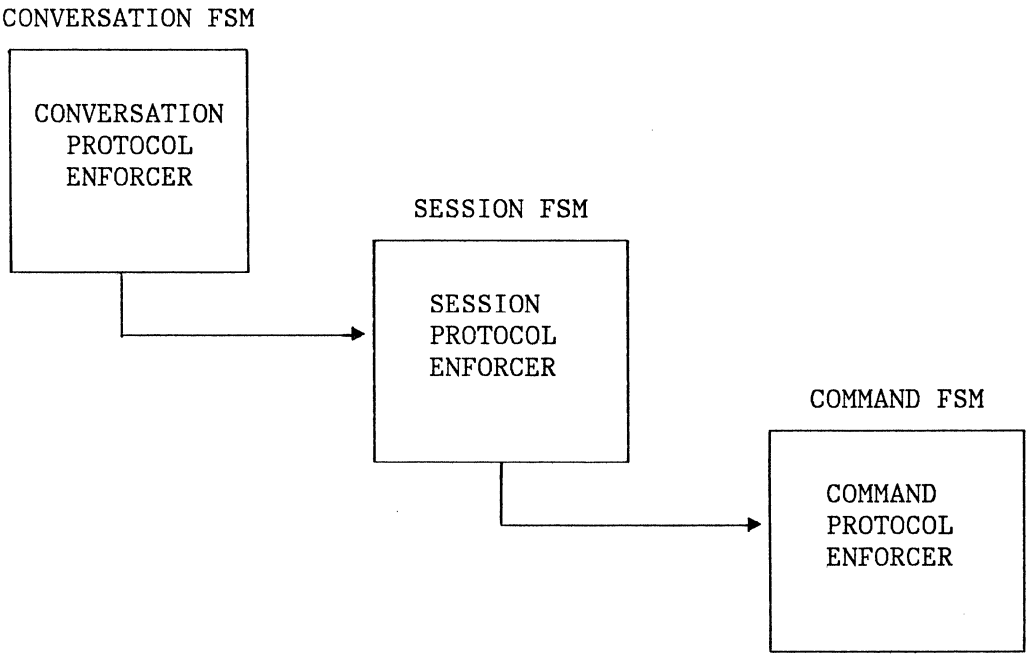
Figure 4. DIA Use of LU 6.2 Verbs (part 1 of 2)

Verb and Parameter	Comments
SEND_DATA RETURN_CODE (variable name) * REQUEST_TO_SEND_RECEIVED (variable name) * RESOURCE (variable name) LENGTH (variable name) DATA (variable name)	* Values are placed in these variables as a result of execution.
RECEIVE_AND_WAIT RETURN_CODE (variable name) * REQUEST_TO_SEND_RECEIVED (variable) * RESOURCE (variable name) WHAT_RECEIVED (variable name) * LENGTH (variable name) DATA (variable name) * FILL (BUFFER) FILL (LL)	Max length converted to actual length as a result of execution. Specify receipt independent of logical record structure.
REQUEST_TO_SEND RESOURCE (variable name)	
SEND_ERROR RETURN_CODE (variable name) REQUEST_TO_SEND_RECEIVED (variable name) RESOURCE (variable name) TYPE (PROG) LOG_DATA (variable name)	Data in variable is to be placed in error logs of system supporting the LU.

Figure 5. DIA Use of LU 6.2 Verbs (Part 2 of 2)

CHAPTER 5. DIA/LU 6.2 PROTOCOL MACHINES

The enforcement of the rules specified in this document may be defined by three finite state protocol machines which have the following hierarchical relationship:



The first protocol machine, Conversation FSM, enforces the DIA use of LU 6.2 conversational states. It checks the DIA operation to be performed against the LU 6.2 state of the DIA process. If the process is in a valid conversational state for the requested operation, then a state transition is made, if necessary, and the next level of protocol checking in Command FSM occurs. If the process is not in a valid conversational state for the requested operation, then a protocol error exists in the DIA conversation.

The second protocol machine, Session FSM, is entered to enforce the DIA session protocols. Session FSM determines if the input operation is valid for the current DIA session state and invokes the appropriate DIA/LU 6.2 sequence if the requested operation affects the setting of DIA session states. If the input operation is valid, but only affects the setting of active DIA session states, then the next level of protocol checking in the Command FSM takes place.

The final protocol machine, Command FSM, enforces the flow control protocols for commands which are exchanged within an active DIA session. The appropriate DIA/LU 6.2 sequences are indicated for each active command request/reply state of the DIA for each valid input operation.

The details of these protocol machines are provided in the sections which follow.

The action sequences which are invoked by these machines are defined in the section entitled "DIA/LU 6.2 Data Flows", which immediately follows the finite state machine descriptions.

In the following tables each entry is coded as:

X(Y), /, or -.

where:

X - Specifies the new state (as a numbered column). If blank (_), there is no state change.

Y - S1, S2 ... S9 are labels for command sequence which are resultant activity. These lists are in the second half table. If Y = FSM2*, then further checking in Command FSM is to be done.

/ or - means error or no action respectively.

DIA CONVERSATION FSM

DIA Conversation State Enforcer: CONVERSATION FSM

Function

Verbs Used by DIA Transaction Processes	LU 6.2 Conversation States						
	Reset	Send	Defer	Receive	Confirm	Sync* point	Deallo-cate
ALLOCATE	yes	no	no	no	no	no	no
CONFIRM	no	yes	yes	no	no	no	no
CONFIRMED	no	no	no	no	yes	no	no
DEALLOCATE with TYPE(SYNC_LEVEL)	no	yes	no	no	no	no	no
DEALLOCATE with TYPE(ABEND)	no	yes	yes	yes	yes	yes	yes
DEALLOCATE with TYPE(LOCAL)	no	no	no	no	no	no	yes
GET_ATTRIBUTES	no	yes	yes	yes	yes	yes	yes
RECEIVE_AND_WAIT	no	yes	no	yes	no	no	no
REQUEST_TO_SEND	no	no	no	yes	yes	yes	no
SEND_DATA	no	yes	no	no	no	no	no
SEND_ERROR	no	yes	no	yes	yes	yes	no

* LU 6.2 CONVERSATIONS USED BY DIA PROGRAMS DO NOT ENTER THIS STATE

Entries in this table are resultant activities defined as follows:

OUTPUT CODE	FUNCTION
yes	Signal Session FSM;
no	Indicates that the input operation should not occur in this state. If it does occur, then a process error exists and the program should issue DEALLOCATE (ABEND) if the cause of the error cannot be locally detected and corrected.

DIA SESSION PROTOCOL MACHINE

DIA Session State Protocol Enforcer: Session FSM

Inputs to the Session FSM are DIA commands.

- The DIU data sent or received in the Conversation FSM is further classified in Session FSM as to the type of command operation which is being requested.
- Session FSM deals directly with only those commands which change the session state of the DIA process. All other commands are passed through to Command FSM as indicated by entry lines labeled OTHER.
- Prior to entering Session FSM, all semantic and syntactic checks have been made on the DIU data. Exception conditions, if any, will appear in this machine and Command FSM as inputs to send ACKNOWLEDGE replies with EC codes.

	RESET 1	NOTACT 2	PEND 3	SONS 4	IDLE 5	QUIET 10
ALLOCATE-DIA-CONV	2(S1)	/	/	/	/	5(S1)
DEALLOCATE-DIA-CONV	/	1(S2)	/	/	10(S2)	/
S SIGN-ON SRR REQUEST	/	3(S4)	/	/	3(S4)	/
S SIGN-ON NRR REQUEST	/	/	/	/	4(S4)	/
S SIGN-ON NRR REPLY	/	/	4(S4)	/	/	/
S ACK (EC,SON)	/	/	1(S6)	/	/	/
R SIGN-ON SRR REQUEST	/	3	/	3	3	/
R SIGN-ON NRR REQUEST	/	/	/	5	_	/
R SIGN-ON NRR REPLY	/	/	5	/	/	/
R ACK (EC,SON)	/	/	2	2	/	/
S SIGN-OFF	/	/	/	/	2(S3)	/
R SIGN-OFF	/	/	/	2	2	/
S OTHER COMMAND	/	/	/	/	FSM2*	/
R OTHER COMMAND	/	/	/	5(FSM2*	FSM2*	/
SEND-DIA-CONFIRM	/	/	/	/	_(S9)	/
REC-DIA-CONFIRM	/	/	/	/	_	/
SEND-DIA-SEND	/	/	/	/	_(S7)	/
REC-DIA-SEND	/	/	/	/	_	/

NOTE: (*) Calls to Command FSM may result in transitions to other active DIA session states (for example, states 5, 6, 7, 8).

OUTPUT CODE	FUNCTION
FSM2	Signal Command FSM;
(/)	Indicates that the input operation should not occur in this state. If it does occur, then a process error exists and the program should issue DEALLOCATE TYPE (ABEND) if the cause of the error cannot be locally detected and corrected.
(-)	Indicates that the input operation does not result in a state change
S1	ALLOCATE (TPN = DIA) LU 6.2 CONVERSATION STATE = SEND
S2	DEALLOCATE (SYNCPT_LEVEL) LU 6.2 CONVERSATION STATE = RESET
S3	SEND_DATA (DIU) LU 6.2CONVERSATION STATE = SEND
S4	SEND_DATA (DIU) RECEIVE_AND_WAIT LU 6.2 CONVERSATION STATE = RECEIVE
S6	SEND_ERROR SEND_DATA (ACK (EC) DIU) DEALLOCATE (SYNCPT_LEVEL) LU 6.2 CONVERSATION STATE = RESET
S7	RECEIVE_AND_WAIT LU 6.2 CONVERSATION STATE = RECEIVE
S9	CONFIRMED LU 6.2 CONVERSATION STATE = RECEIVE

DIA COMMAND PROTOCOL MACHINE

DIA Command State Protocol Enforcer: Command FSM

Function

- FSM2 enforces the command request/reply protocols for command flow within an active DIA session.
- The LAST, NOT LAST input values to FSM2 correspond to the Last, Not-Last reply indicator value carried in the CORRELATION operand of all DIA replying commands.

	IDLE 5	SRRS 6	SRRSR 7	SRRR 8	SRRRS 9
S CMD REQUEST SRR	6(S4)	/	/	/	/
S CMD REPLY SRR, NOT LAST	/	/	/	9(S4)	/
S CMD REPLY NRR, LAST	/	/	6(S4)	5(S3)	/
S ACK (EC,CMD)	_(S8)	/	6(S5)	5(S5)	/
R CMD REQUEST SRR	8	/	/	/	/
R CMD REPLY SRR, NOT LAST	/	7	/	/	/
R CMD REPLY NRR, LAST	/	5	/	/	8
R ACK (EC,CMD)	_	5	/	/	8
S CMD REQUEST NRR	_(S3)	/	/	/	/
R CMD REQUEST NRR	_	/	/	/	/

OUTPUT CODE	FUNCTION
(/)	Indicates that the input operation should not occur in this state. If it does occur, then a process error exists and the program should issue DEALLOCATE TYPE (ABEND) if the cause of the error cannot be locally detected and corrected.
(-)	Indicates that the input operation does not result in a state change
S3	SEND_DATA (DIU) LU 6.2 CONVERSATION STATE = SEND
S4	SEND_DATA (DIU) RECEIVE_AND_WAIT LU 6.2 CONVERSATION STATE = RECEIVE
S5	SEND_ERROR SEND_DATA (ACK (EC) DIU) RECEIVE_AND_WAIT LU 6.2 CONVERSATION STATE = RECEIVE
S8	SEND_ERROR SEND_DATA (ACK (EC) DIU) LU 6.2 CONVERSATION STATE = SEND

CHAPTER 6. DIA/LU 6.2 DATA FLOWS

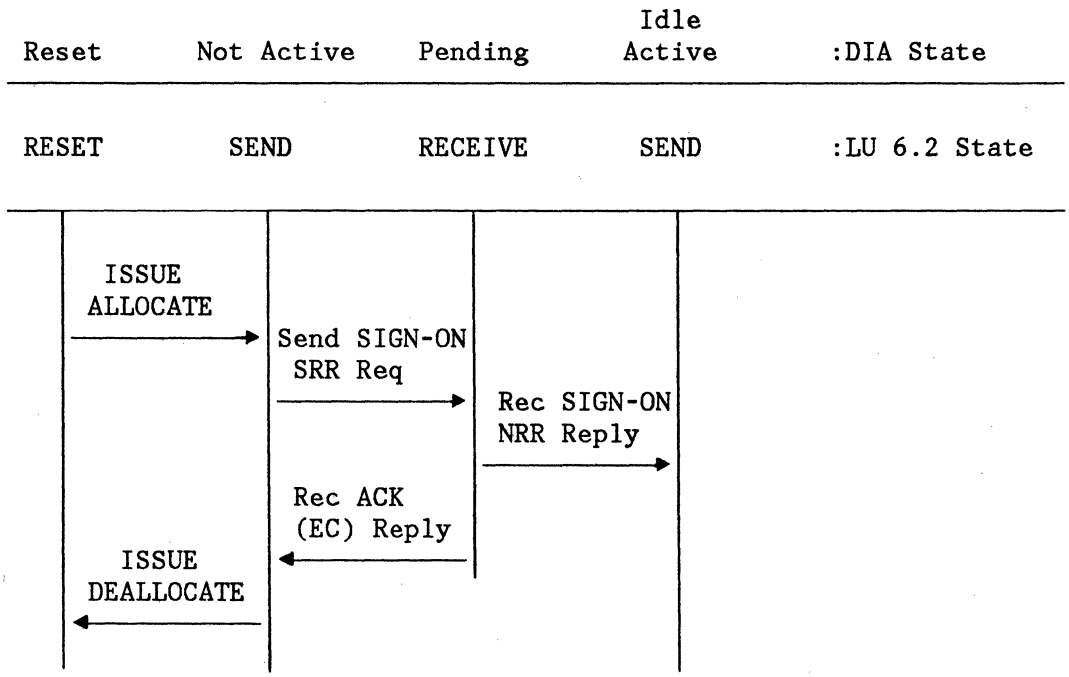
As an aid to transaction programmers, detailed explanatory scenarios are provided in this section for each of the commands defined by the DIA. Each scenario is preceded by a state transition diagram showing the flow of control and data for that command operation.

Conventions used in these scenarios are:

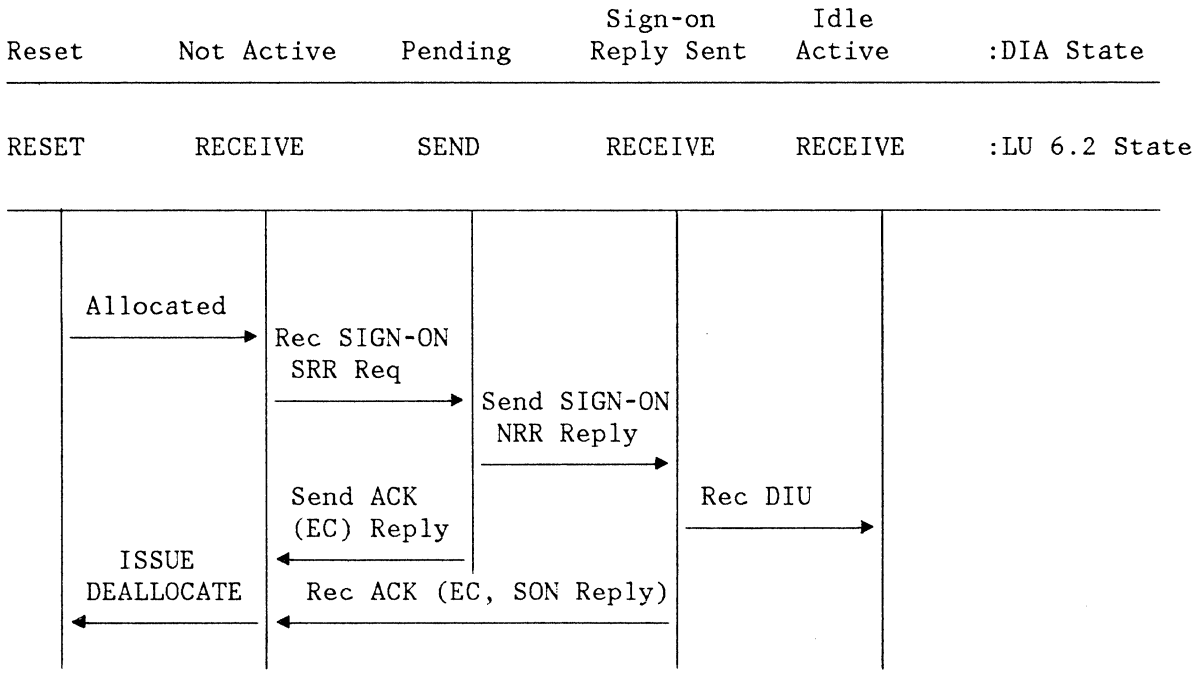
<XXX>	XXX = DIA program state
|YYY|	YYY = DIA architected processing
ZZZ	ZZZ = paraphrased LU 6.2 processing or verb usage
—WWW→ or ←WWW—	= paraphrased DIA architected message
——→ or ←——	= LU 6.2 program notification of issuance of SEND_ERROR verb
* VVV *	VVV = illustrative notes

SIGN-ON COMMAND

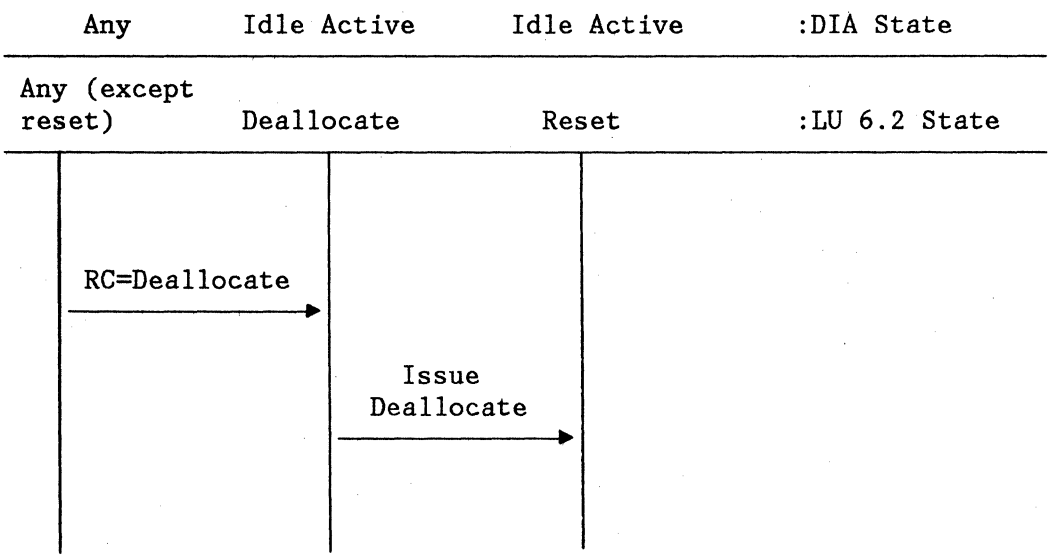
SENDER OF SIGN-ON



RECEIVER OF SIGN-ON

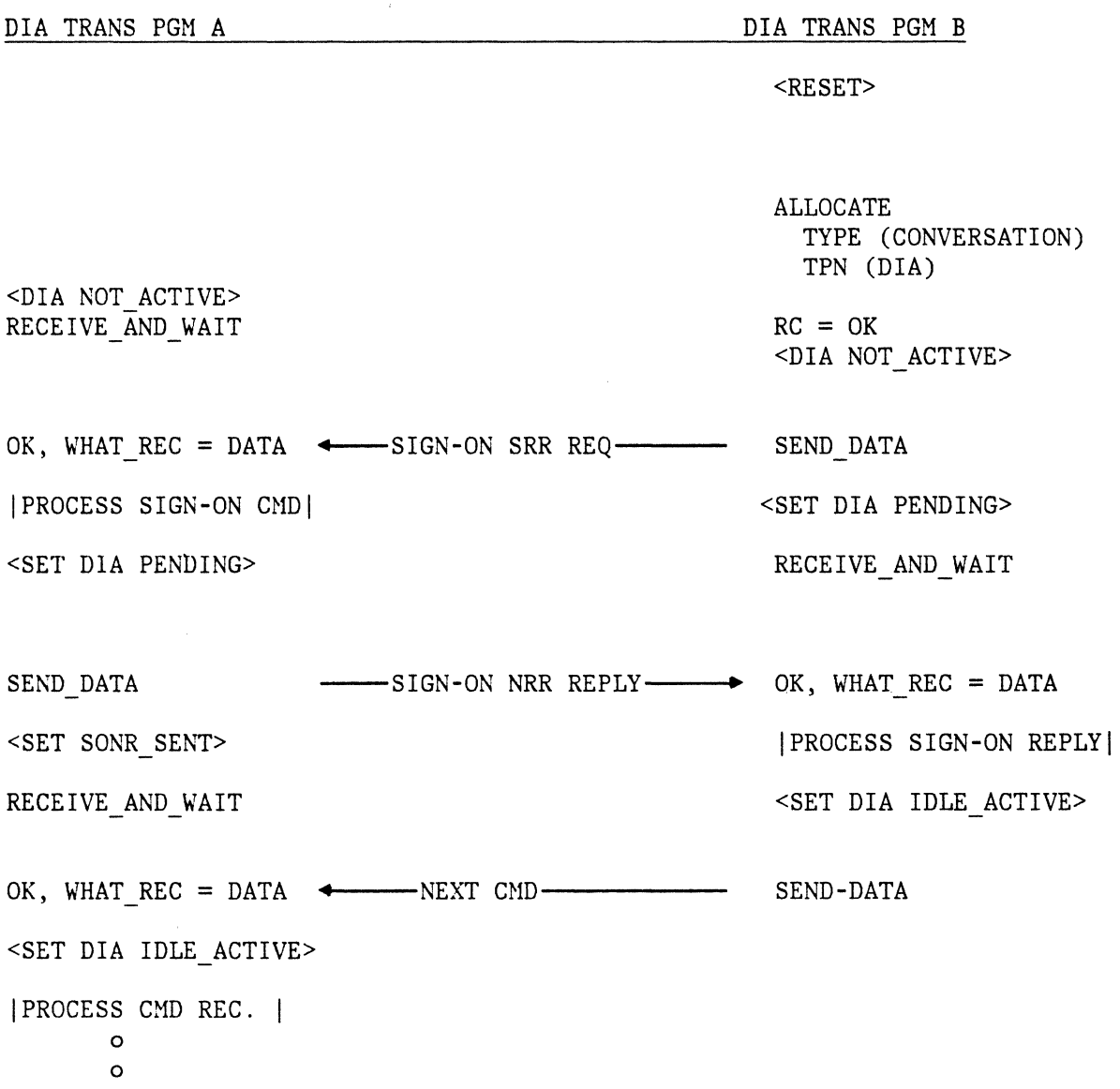


The following FSM applies to this and all other scenarios below:



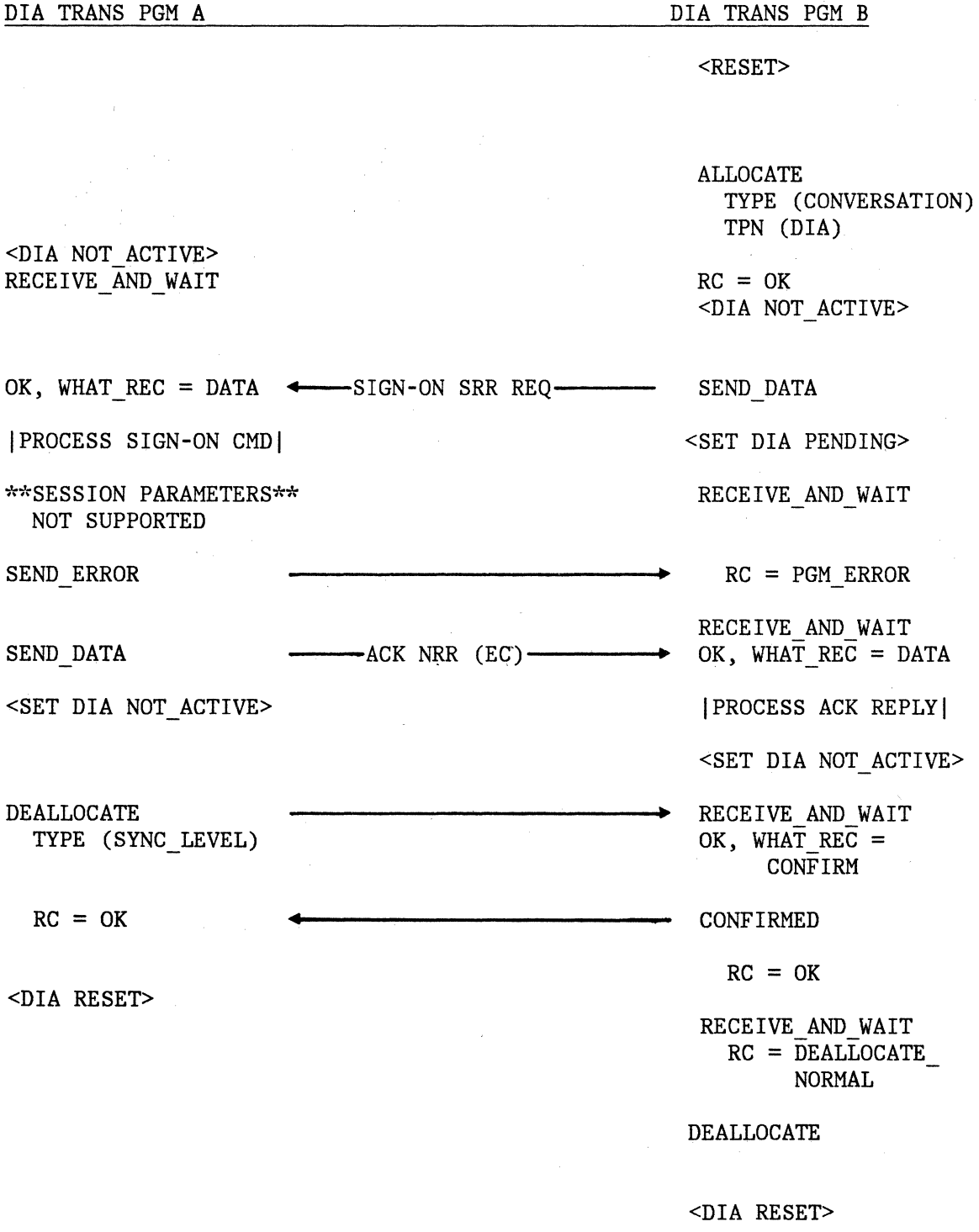
DIA SIGN-ON SCENARIO

CASE1: RECEIVER ACCEPTS SIGN-ON



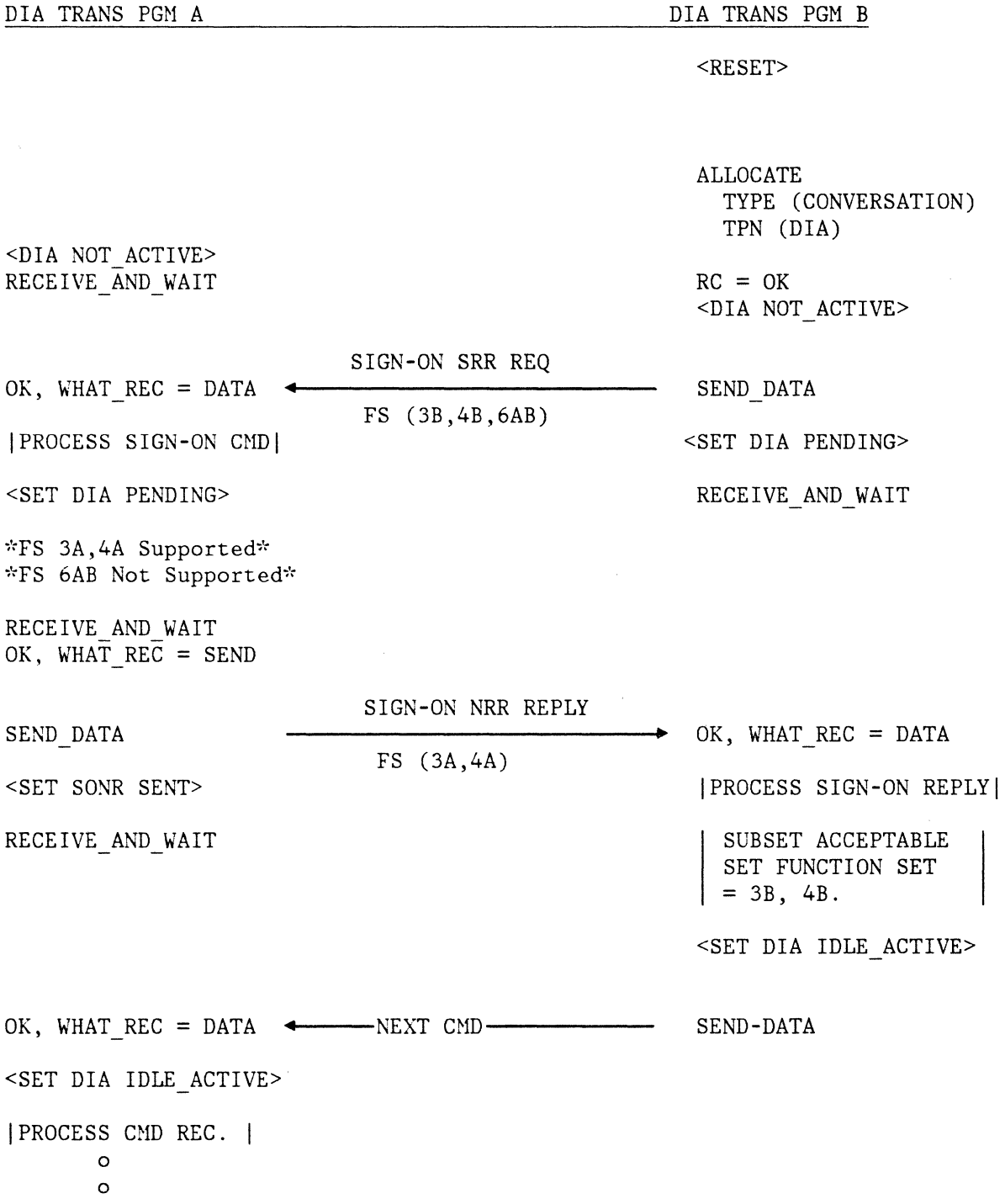
DIA SIGN-ON SCENARIO

CASE2: RECEIVER REJECTS SIGN-ON



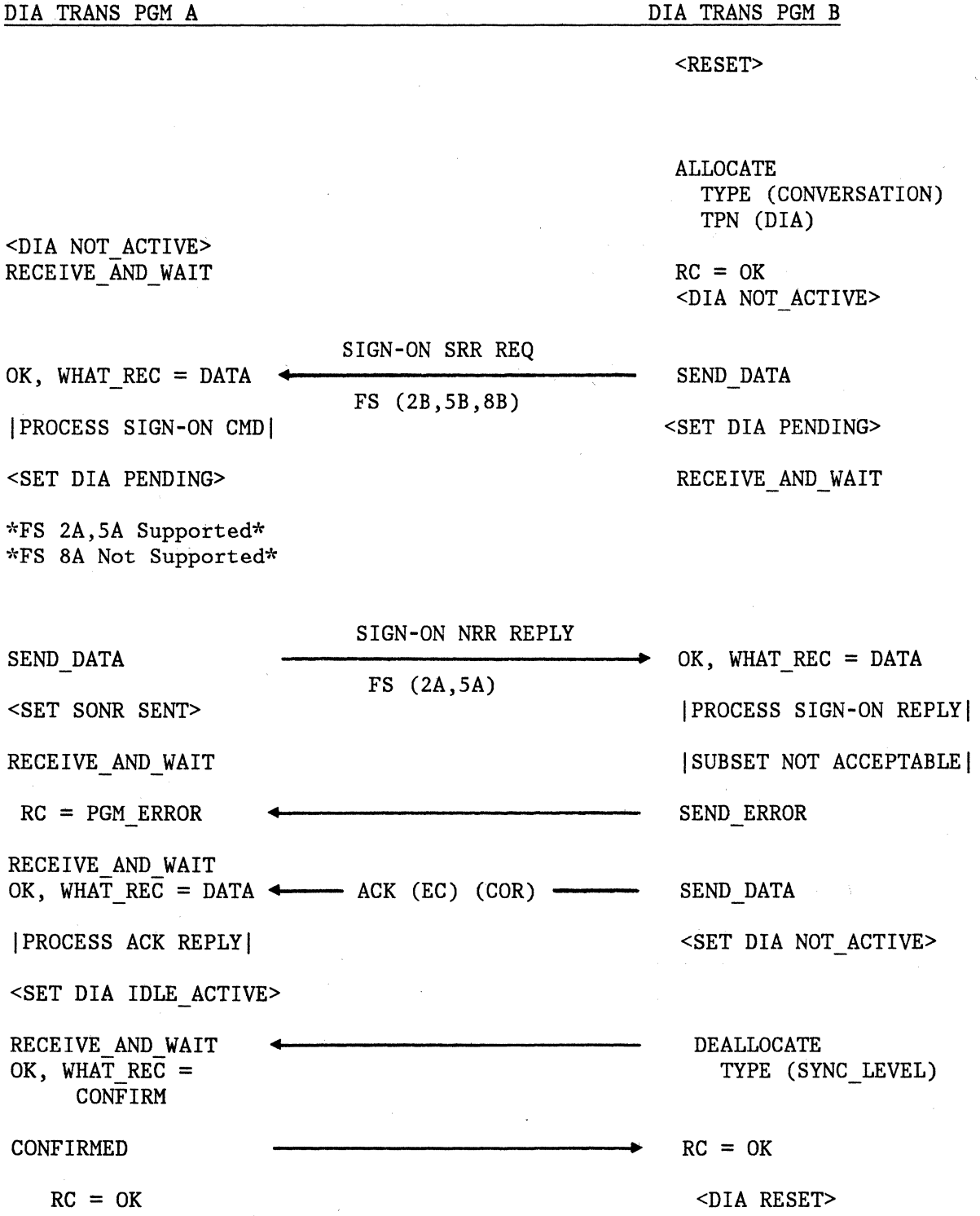
DIA SIGN-ON SCENARIO

CASE3: SENDER ACCEPTS NEGOTIATED SIGN-ON REPLY



DIA SIGN-ON SCENARIO

CASE4: SENDER REJECTS NEGOTIATED SIGN-ON REPLY



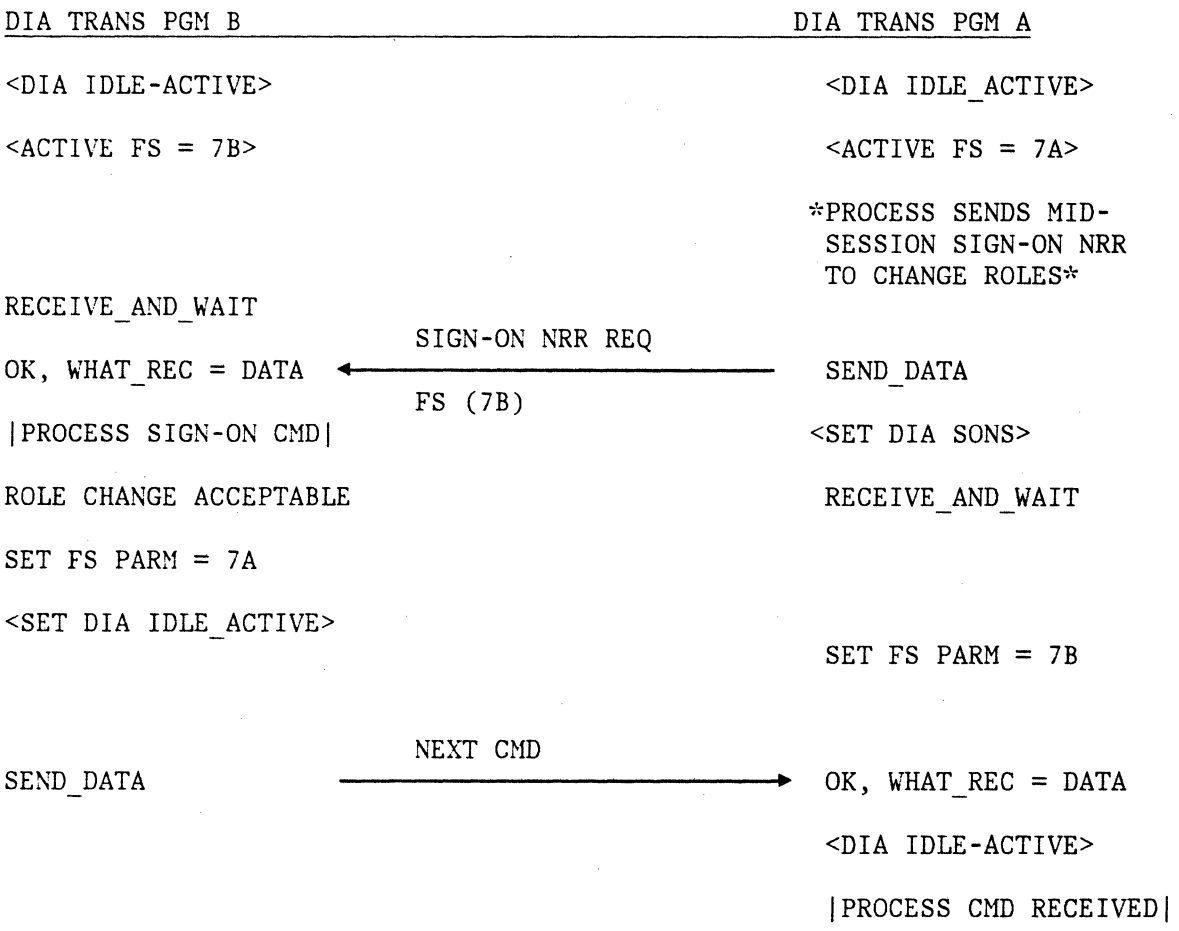
```
RECEIVE_AND_WAIT
  RC = DEALLOCATE_NORMAL

DEALLOCATE
  RC = OK

<DIA RESET>
```

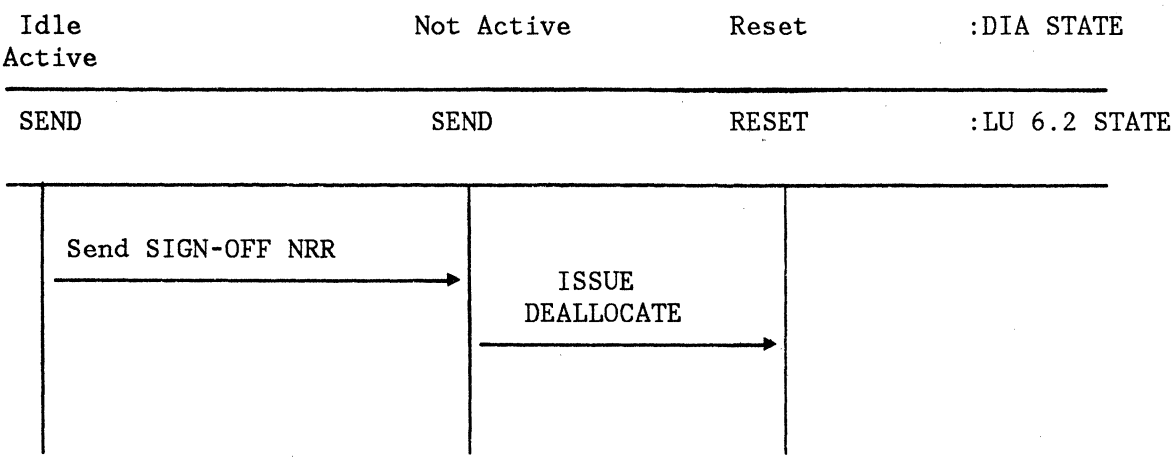
DIA SIGN-ON SCENARIO

CASE5: RECEIVER ACCEPTS MID-SESSION SIGN_ON NRR REQUEST
(This case is allowed only with Function Set 7 for migration)

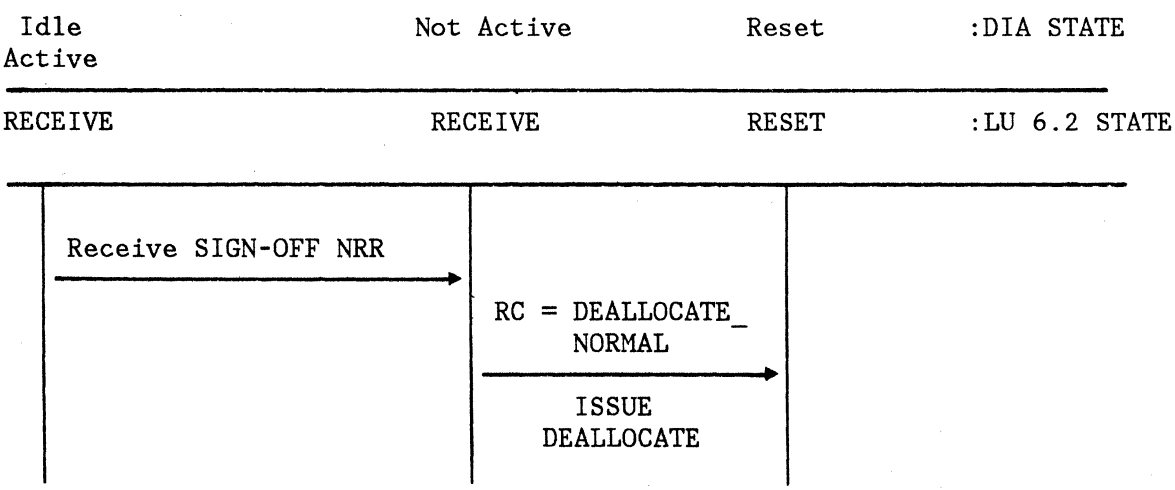


SIGN-OFF COMMAND

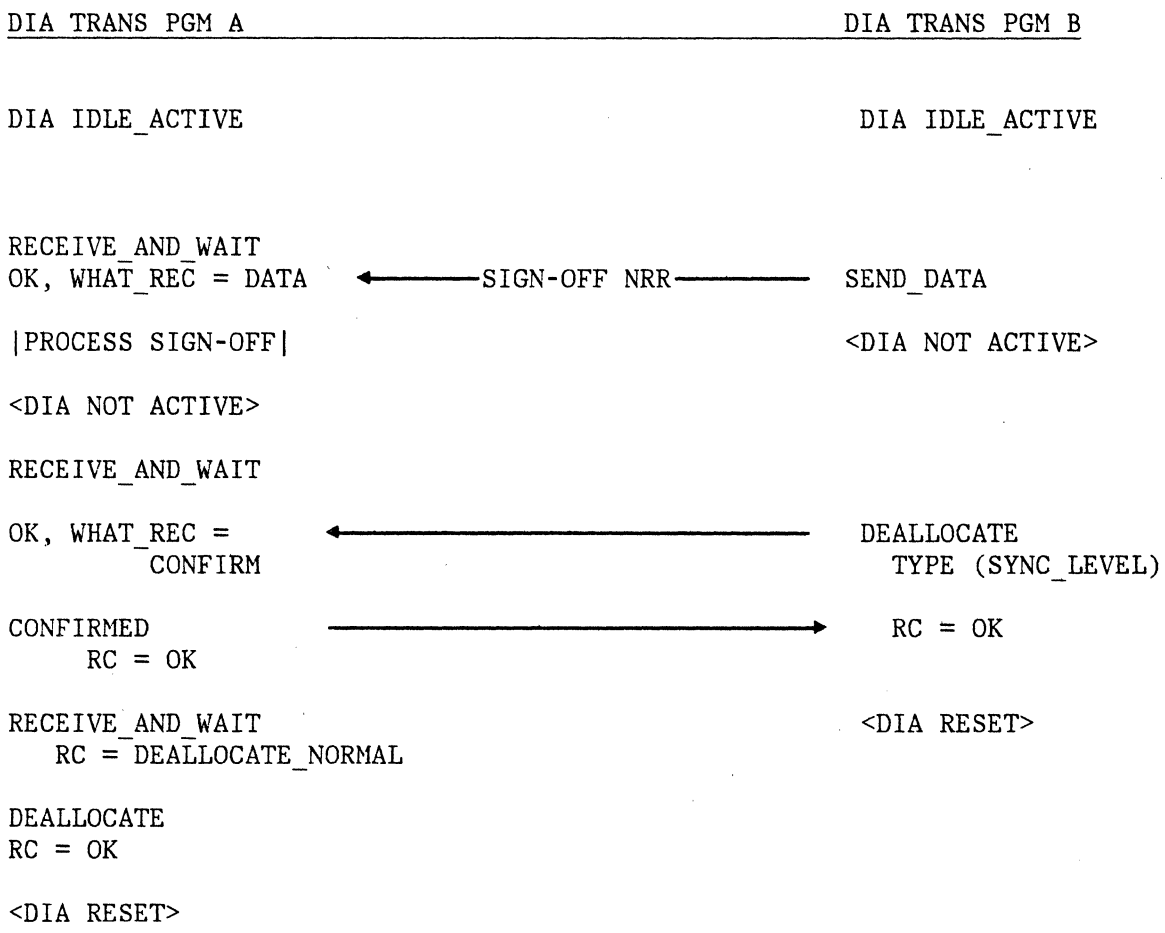
SENDER OF SIGN-OFF



RECEIVER OF SIGN-OFF

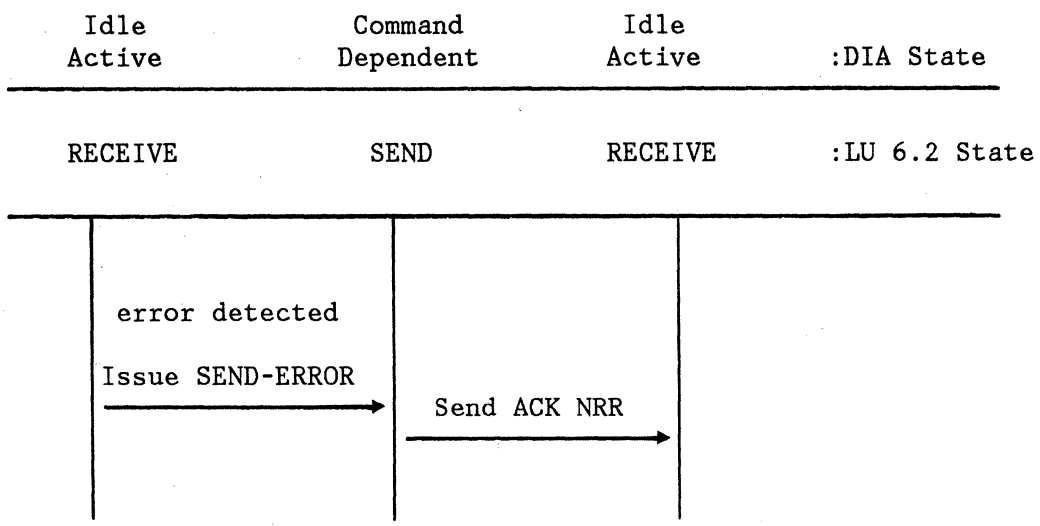


DIA SIGN-OFF SCENARIO

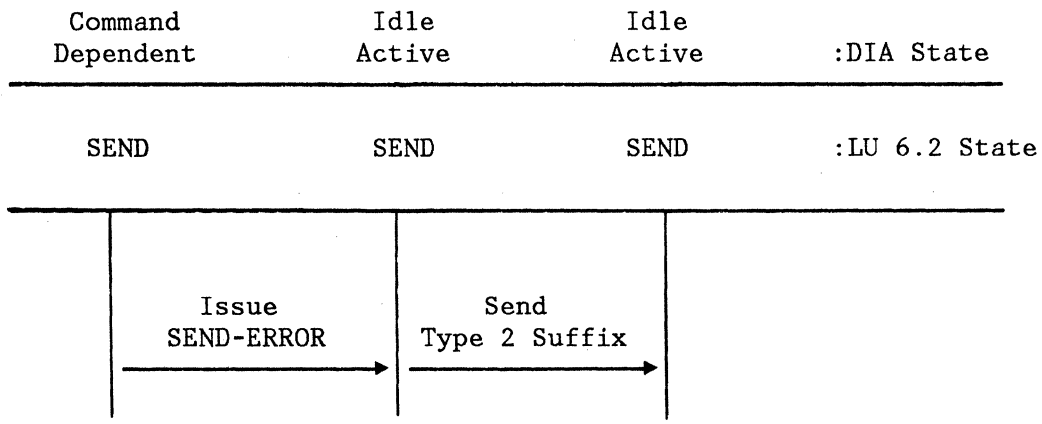


SEND-ERROR SEQUENCES

DIU RECEIVER FSM

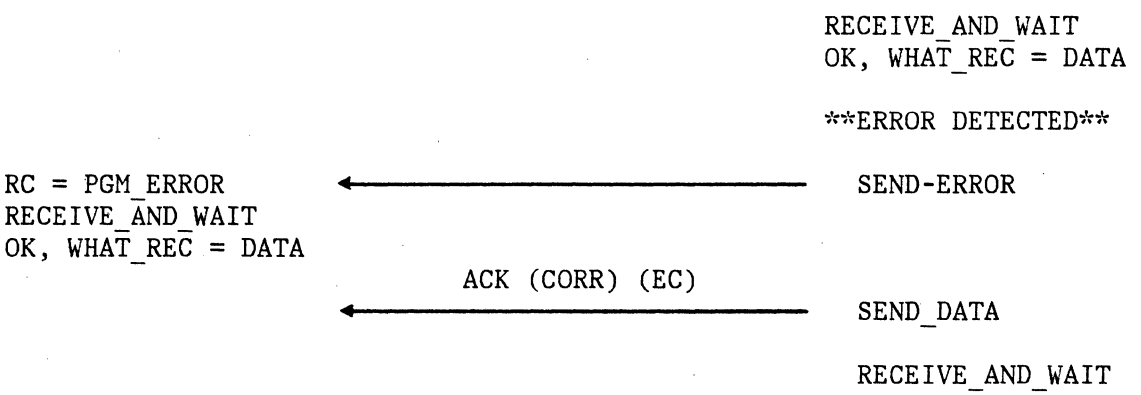


DIU SENDER FSM

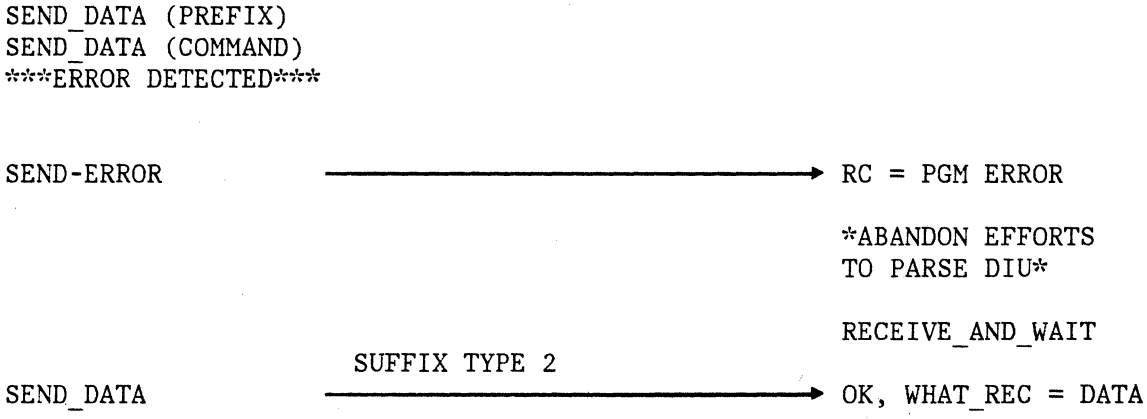


SEND-ERROR SCENARIOS

EXCEPTIONS DETECTED BY DIU RECEIVER

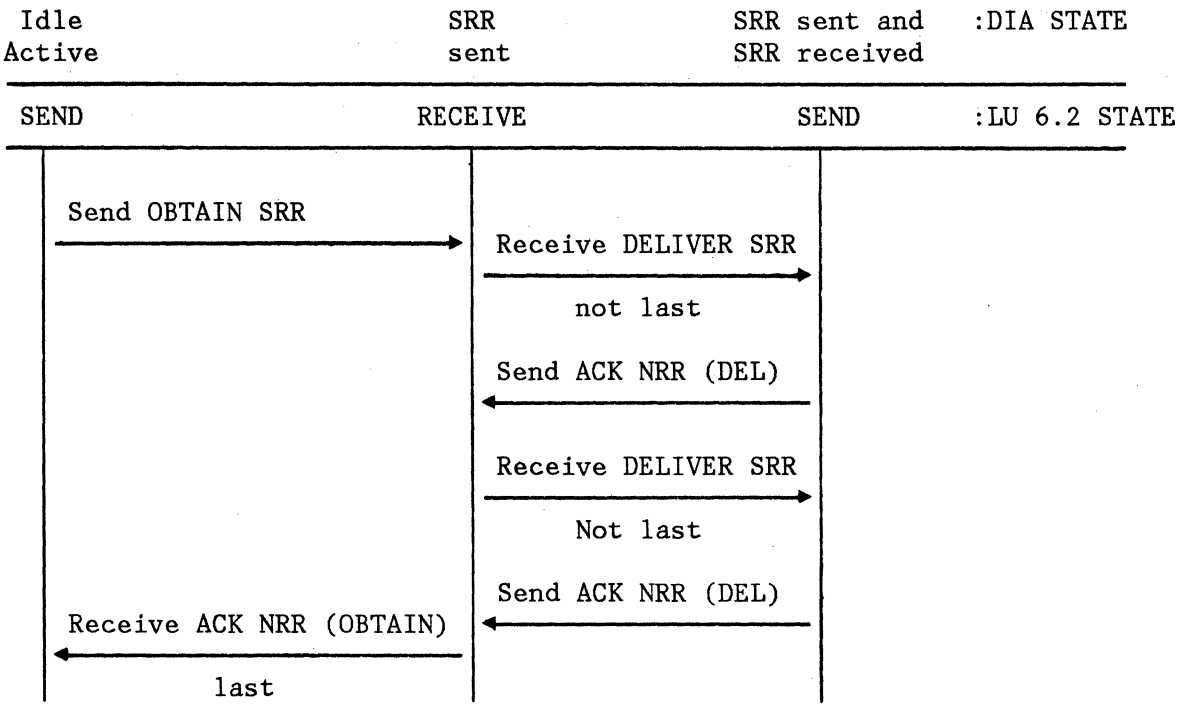


EXCEPTION CONDITIONS DETECTED BY THE DIU SENDER

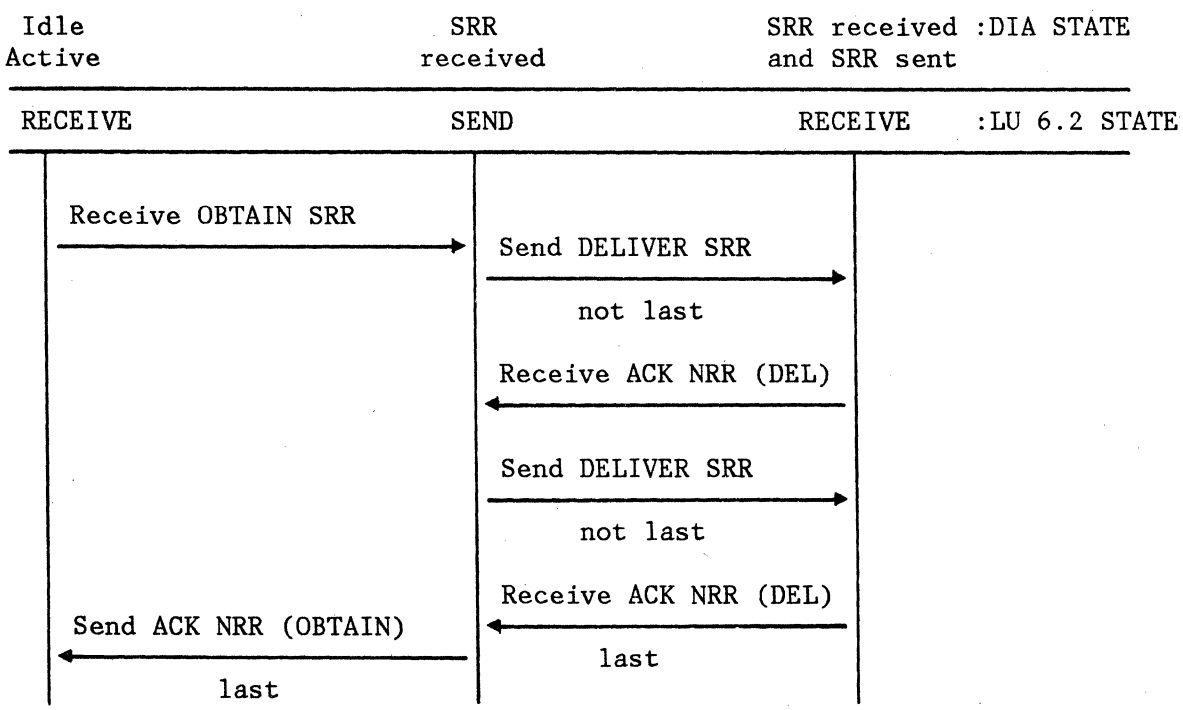


OBTAIN COMMAND

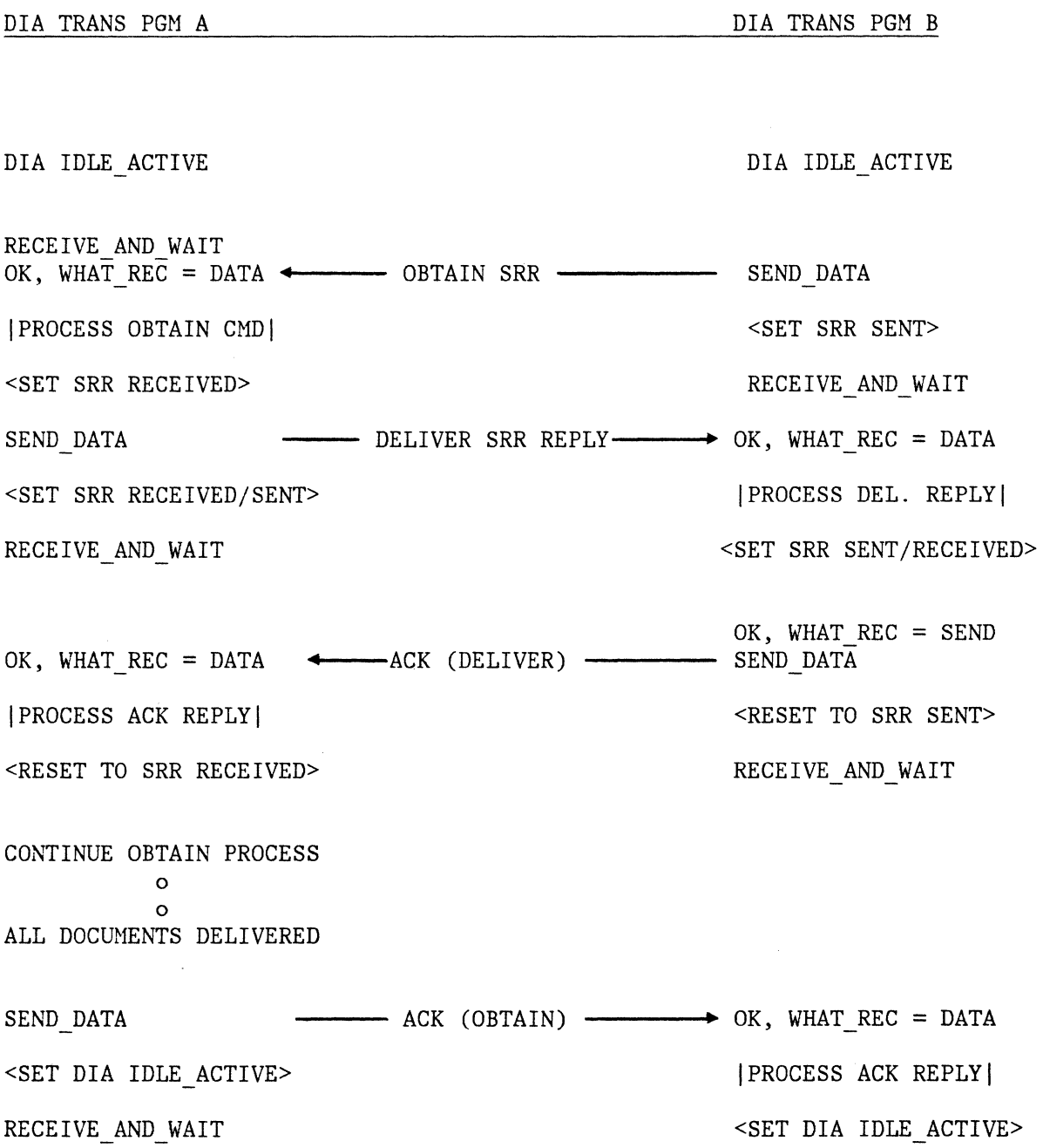
SENDER OF OBTAIN



RECEIVER OF OBTAIN

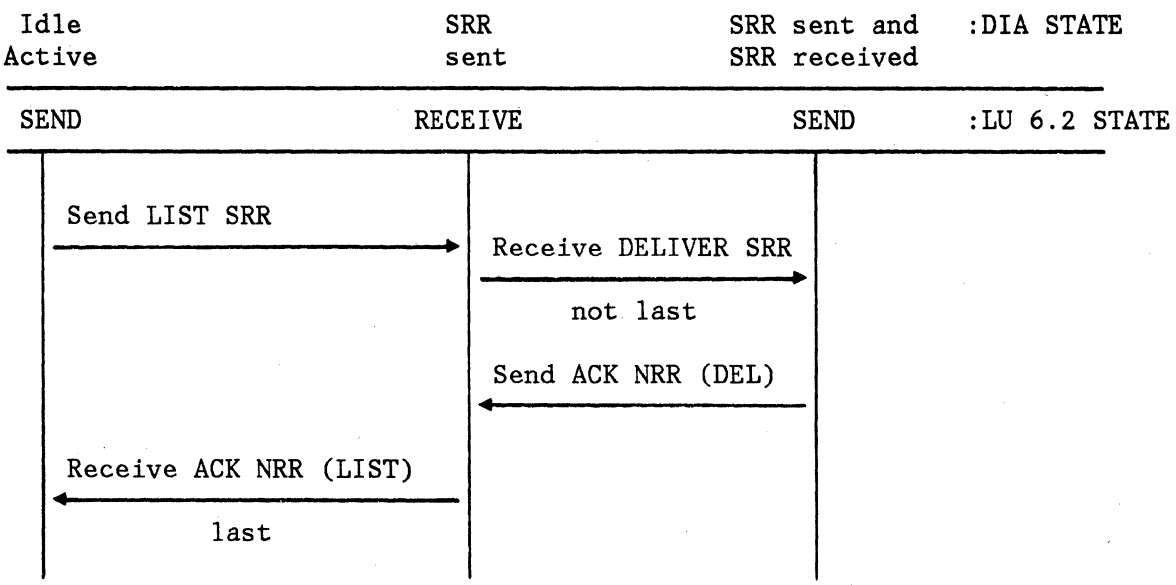


DIA OBTAIN SCENARIO

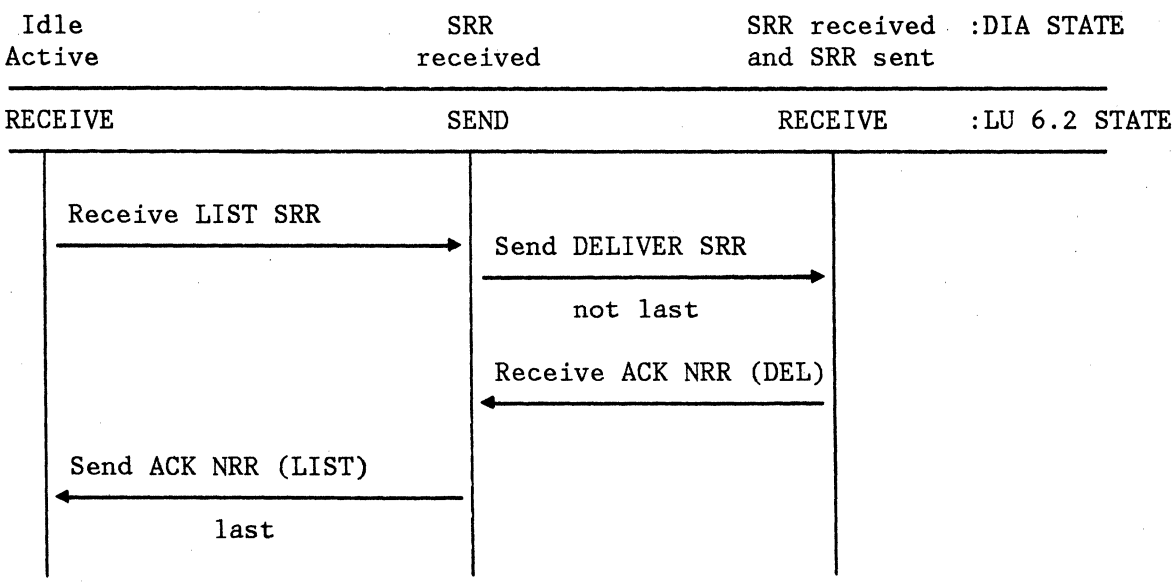


LIST COMMAND

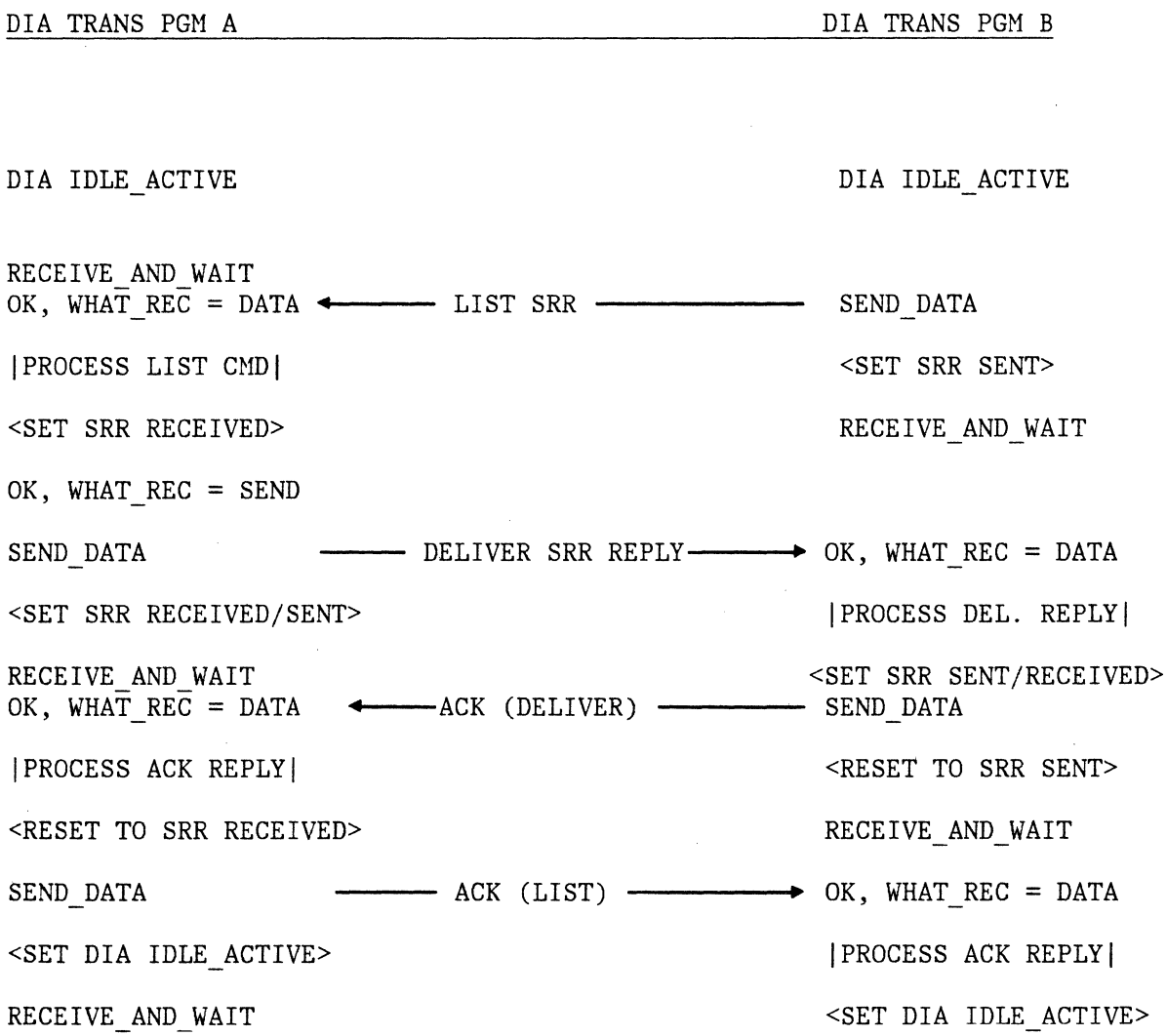
SENDER OF LIST



RECEIVER OF LIST

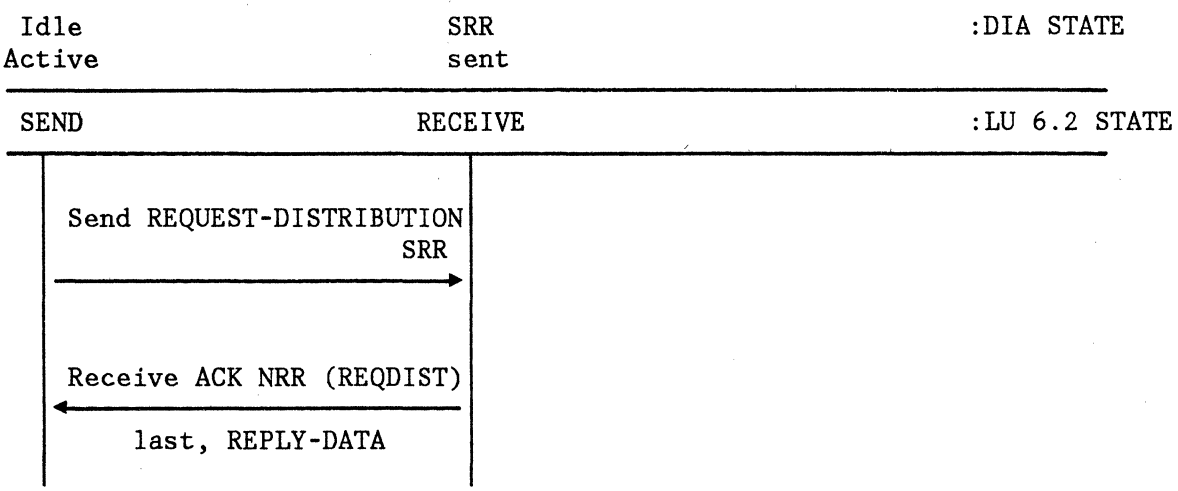


DIA LIST SCENARIO

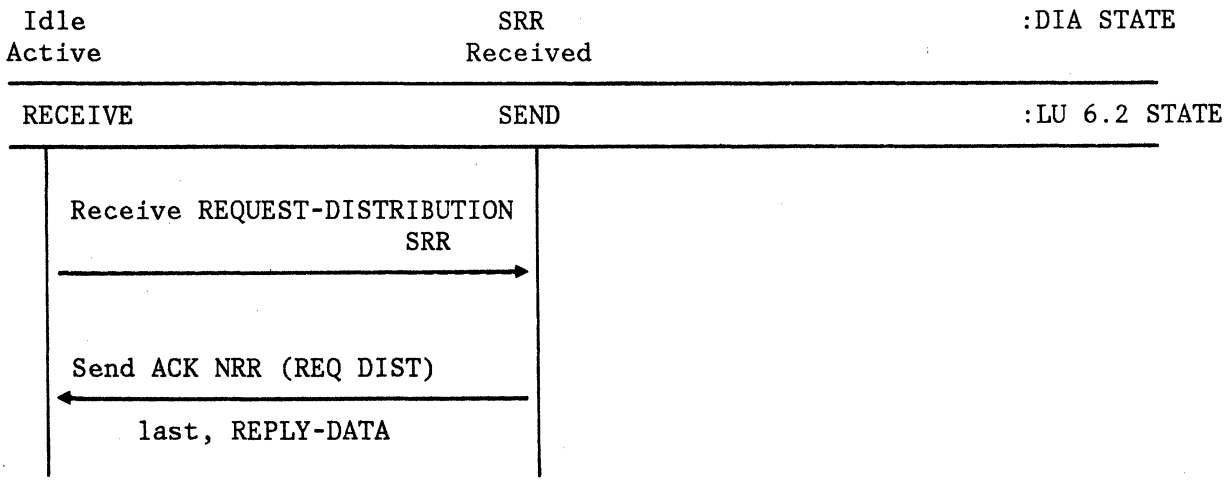


REQUEST-DISTRIBUTION COMMAND

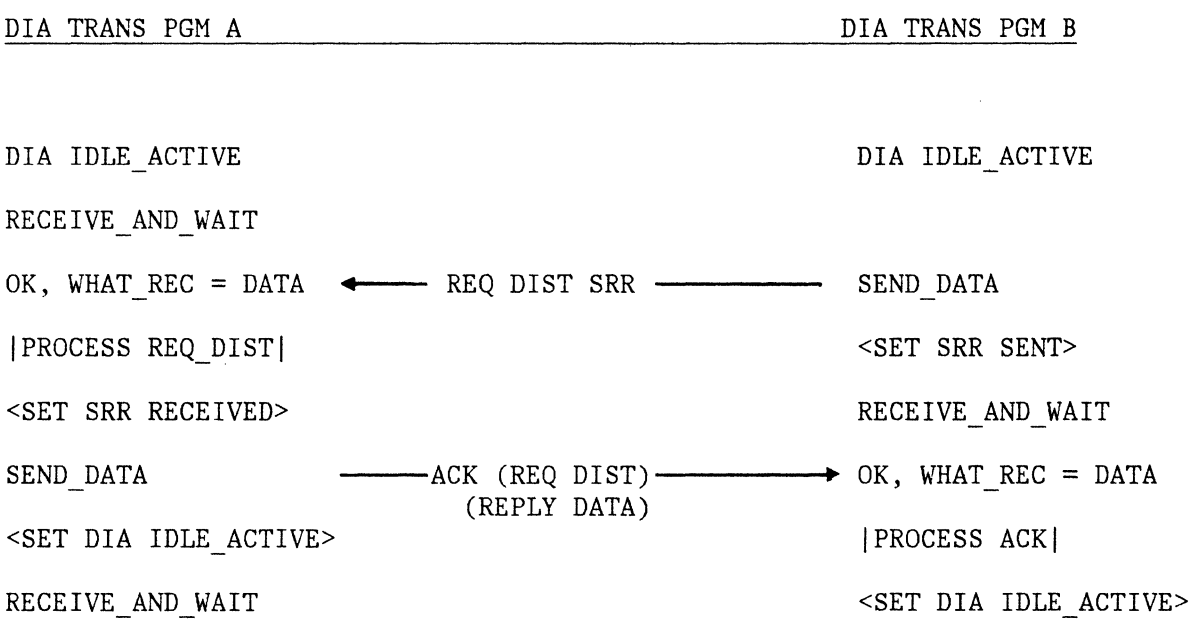
SENDER OF REQUEST-DISTRIBUTION



RECEIVER OF REQUEST-DISTRIBUTION

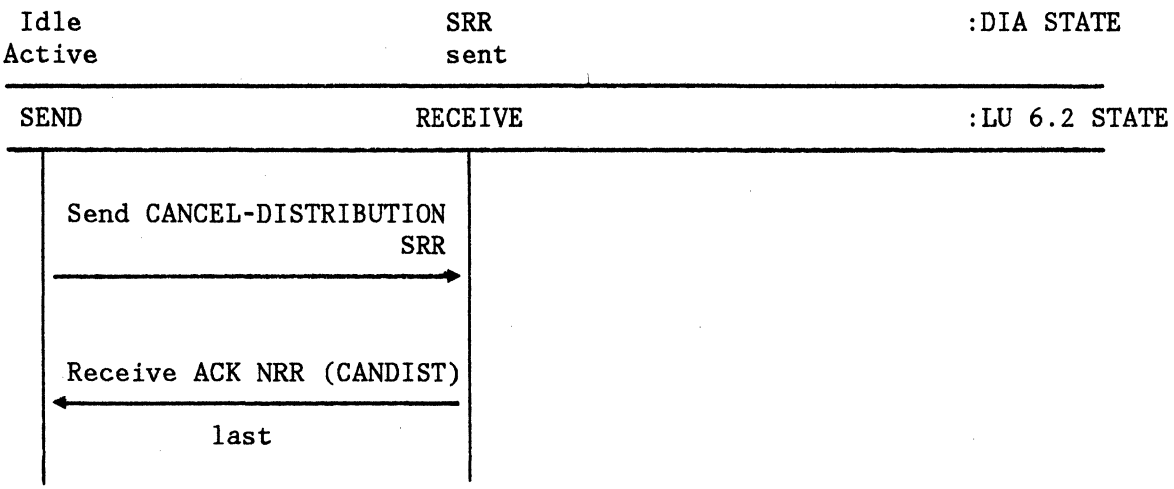


DIA REQUEST-DISTRIBUTION SCENARIO

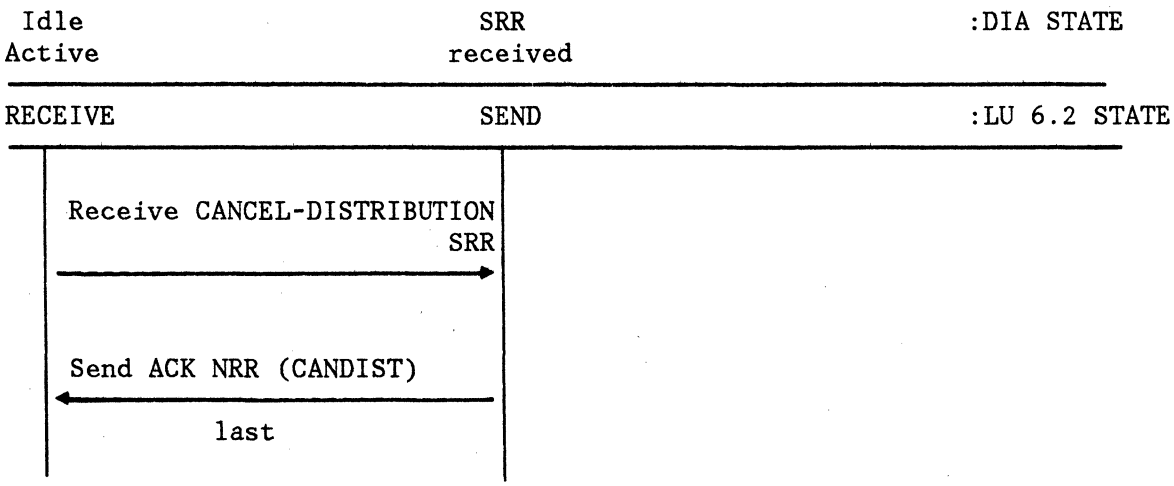


CANCEL-DISTRIBUTION COMMAND

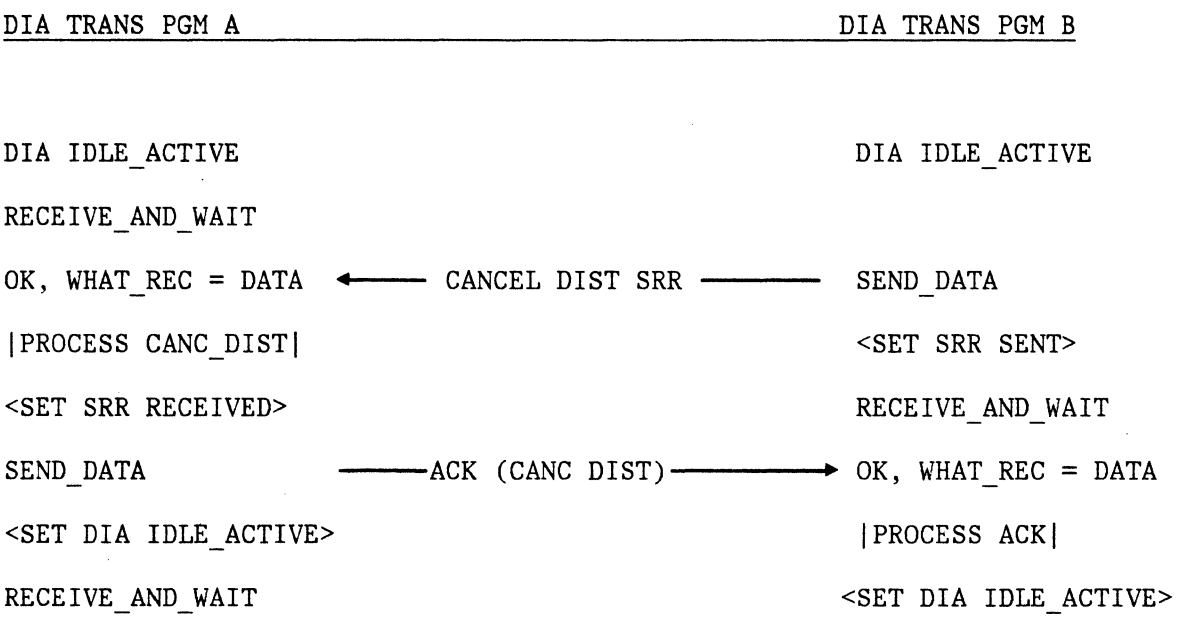
SENDER OF CANCEL-DISTRIBUTION



RECEIVER OF CANCEL-DISTRIBUTION

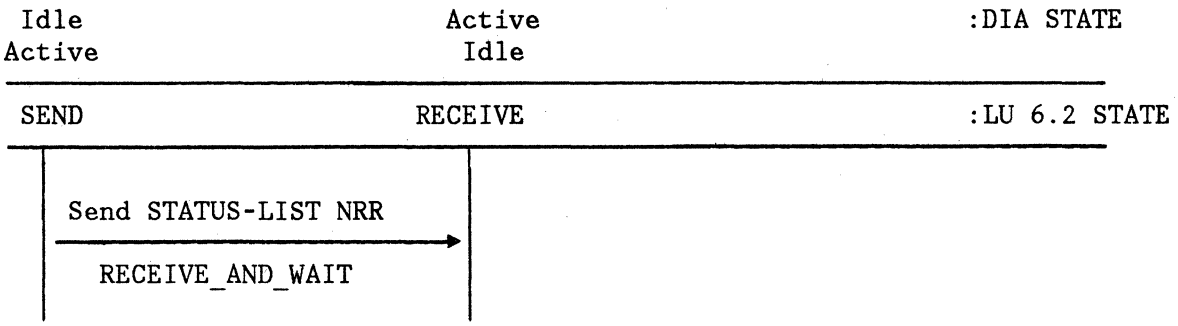


DIA CANCEL-DISTRIBUTION SCENARIO

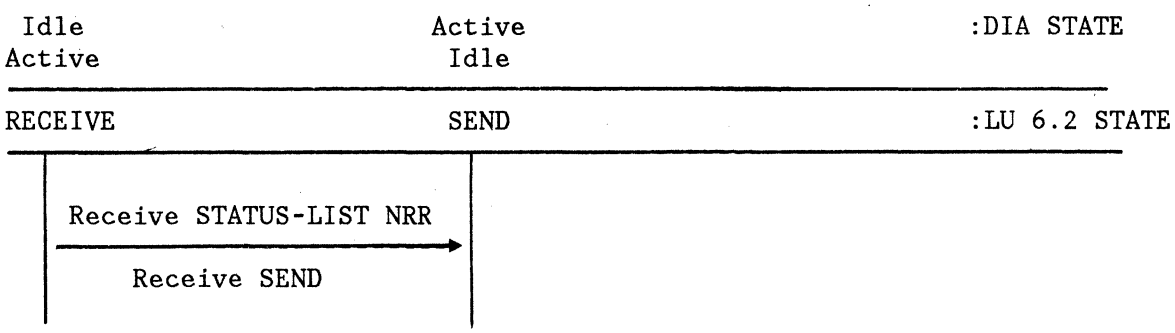


STATUS-LIST COMMAND

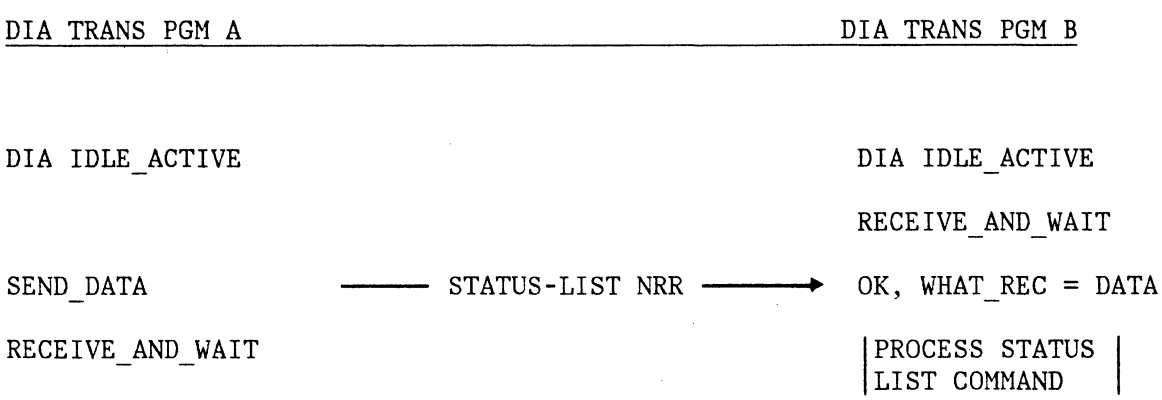
SENDER OF STATUS-LIST



RECEIVER OF STATUS-LIST

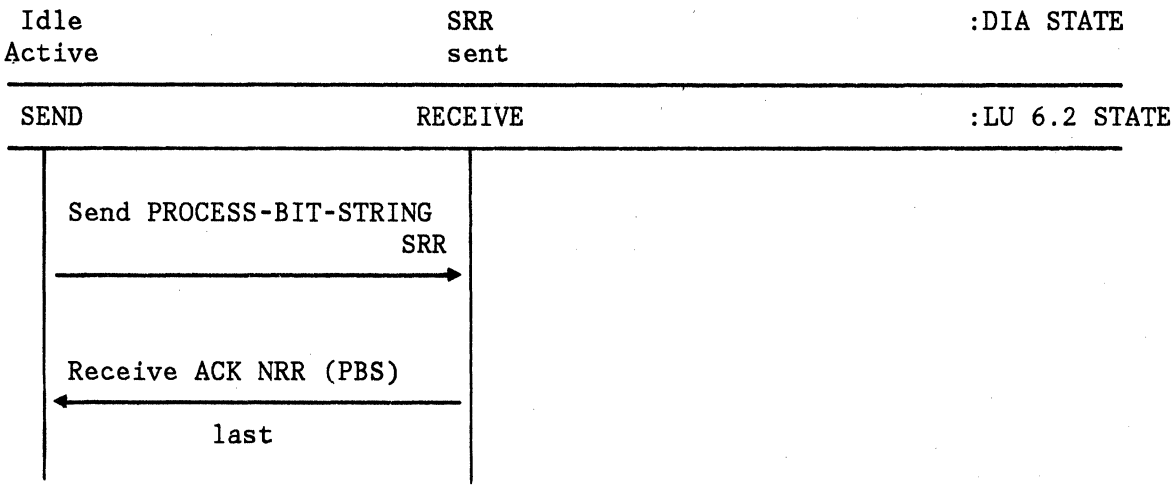


DIA STATUS-LIST SCENARIO

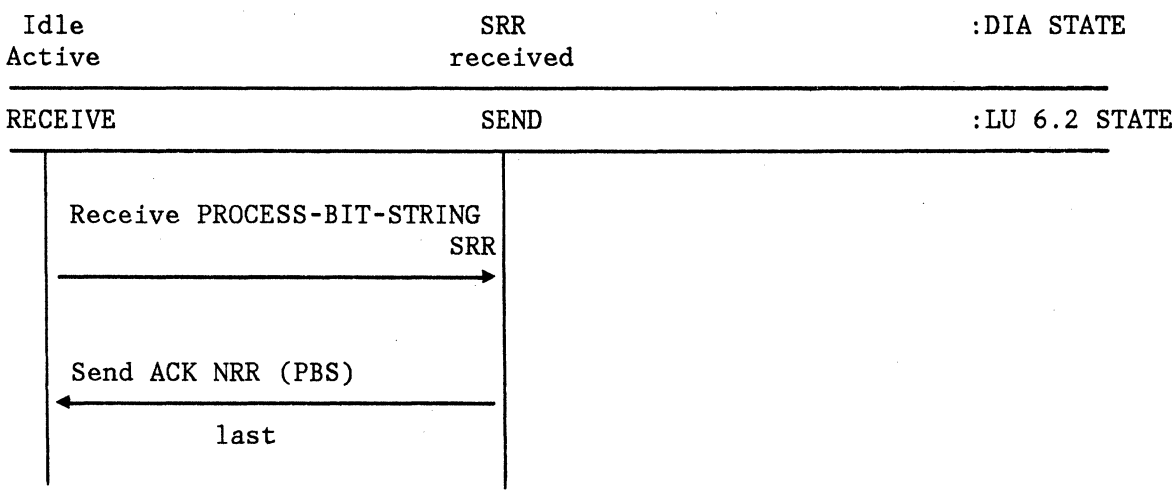


PROCESS-BIT-STRING COMMAND

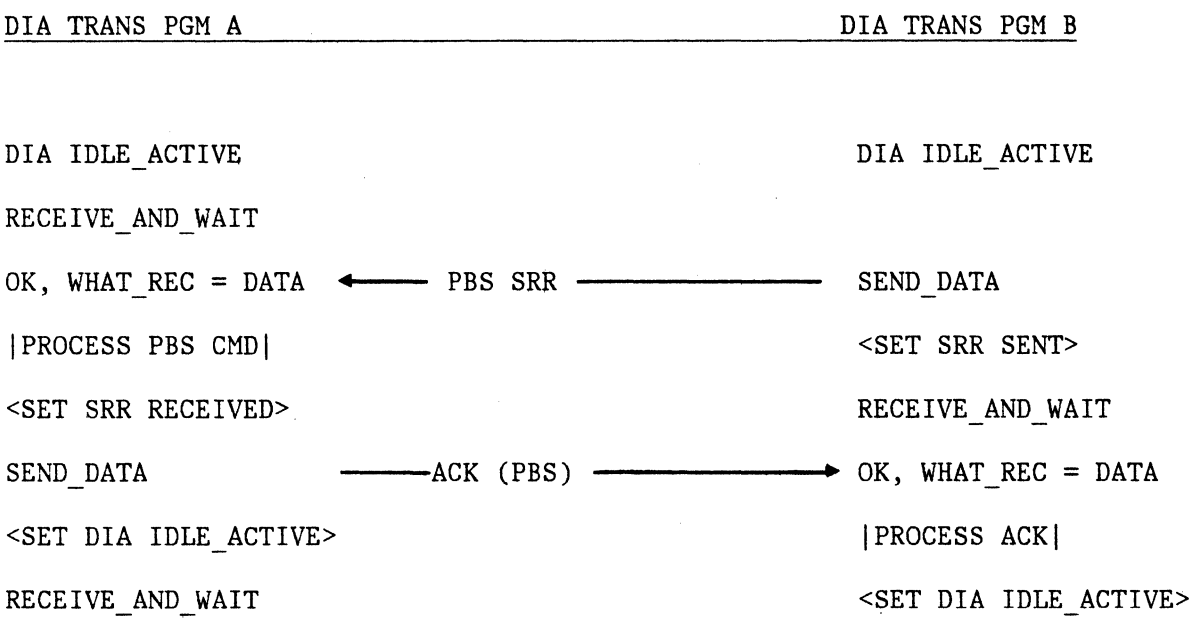
SENDER OF PROCESS-BIT-STRING



RECEIVER OF PROCESS-BIT-STRING

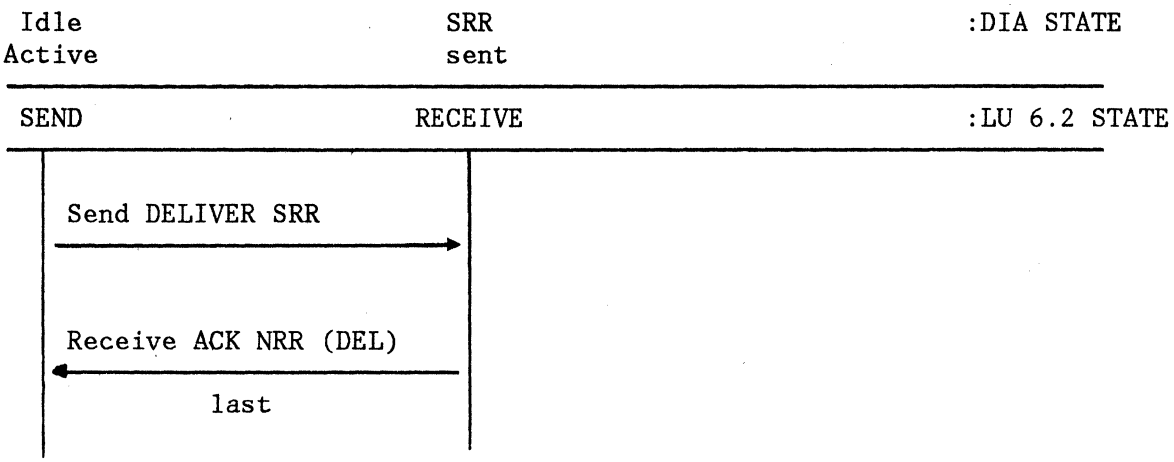


DIA PROCESS-BIT-STRING SCENARIO

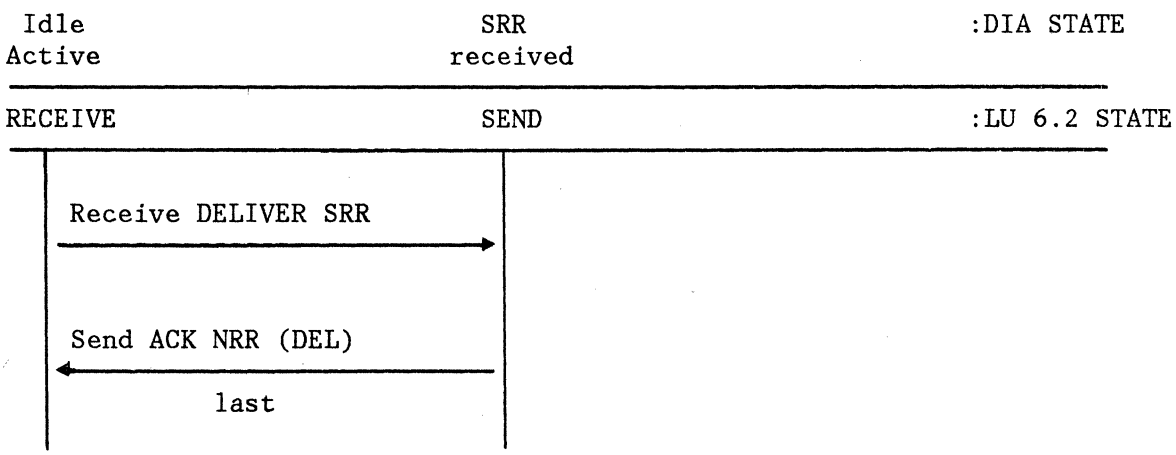


DELIVER COMMAND

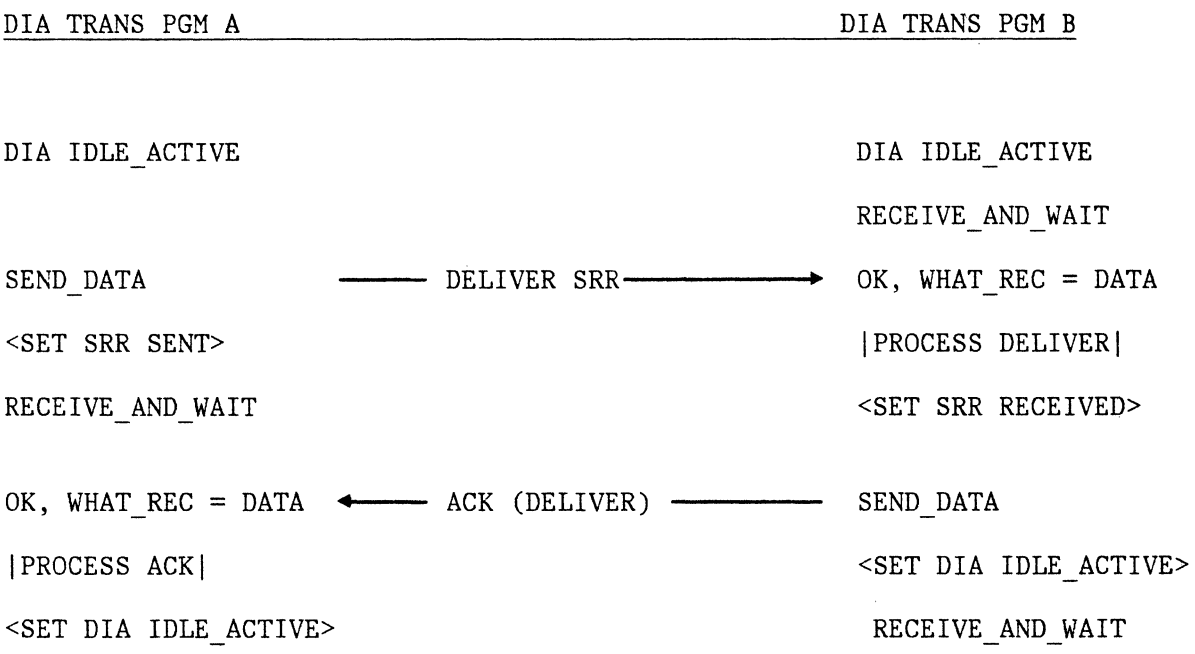
SENDER OF DELIVER



RECEIVER OF DELIVER

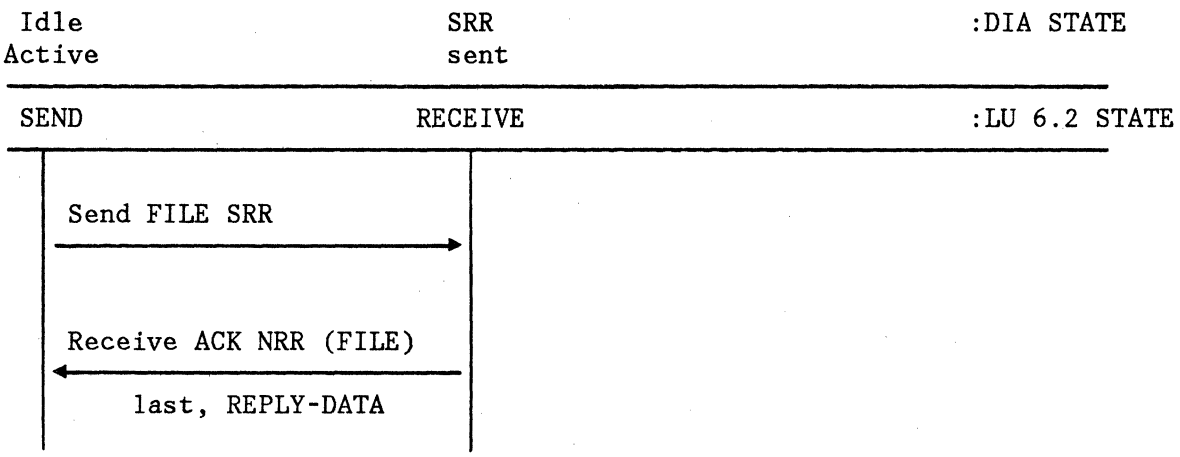


DIA DELIVER SCENARIO

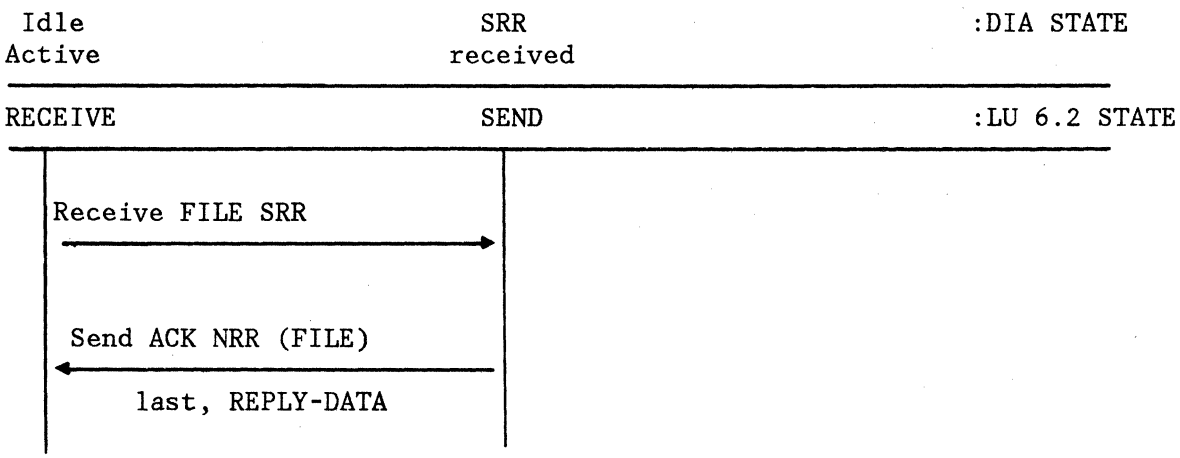


FILE COMMAND

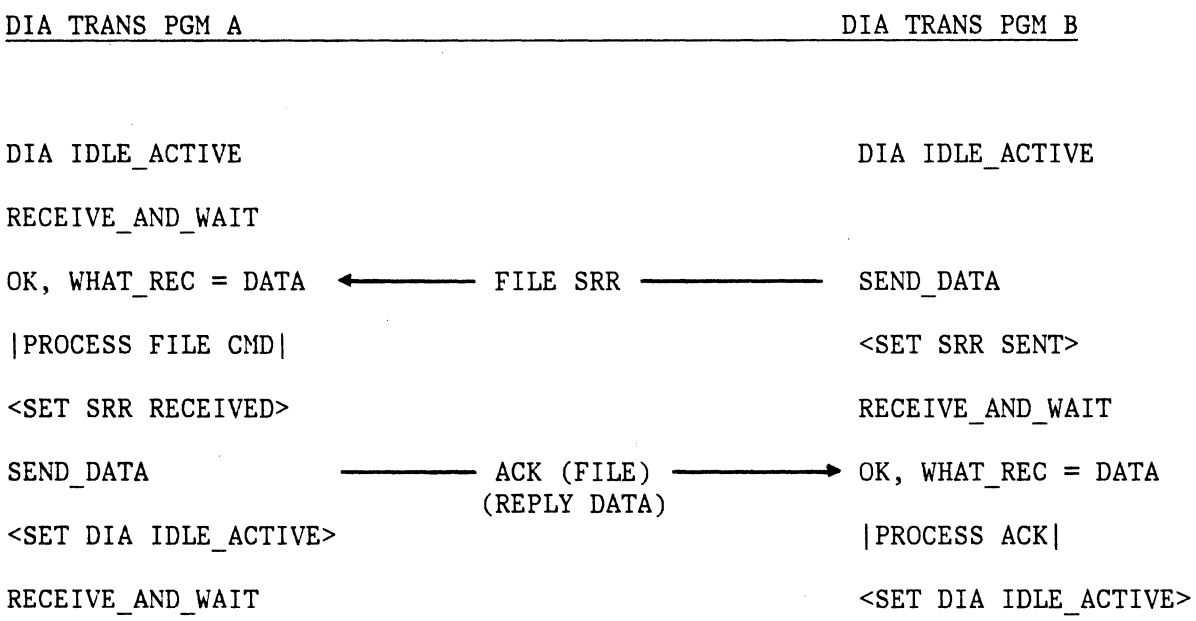
SENDER OF FILE



RECEIVER OF FILE

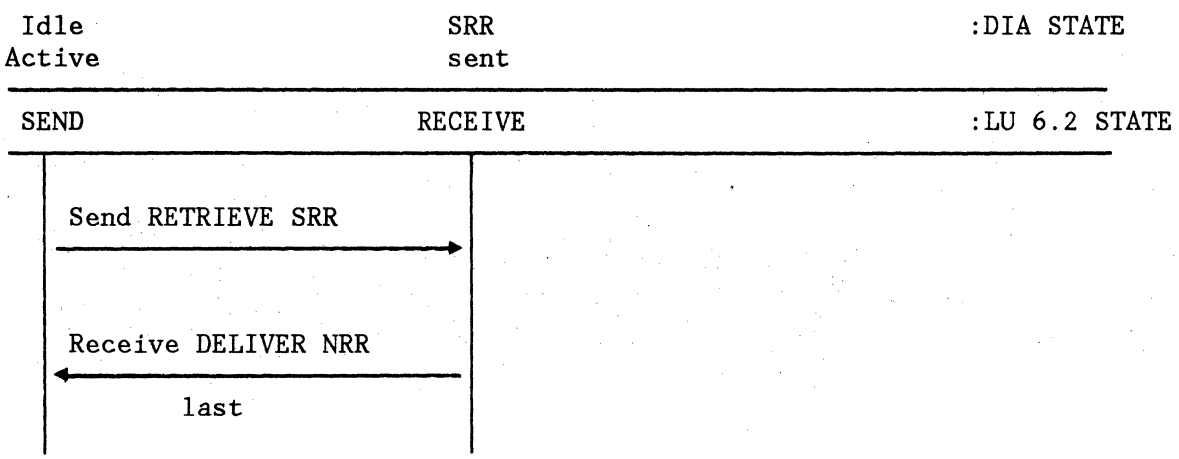


DIA FILE SCENARIO

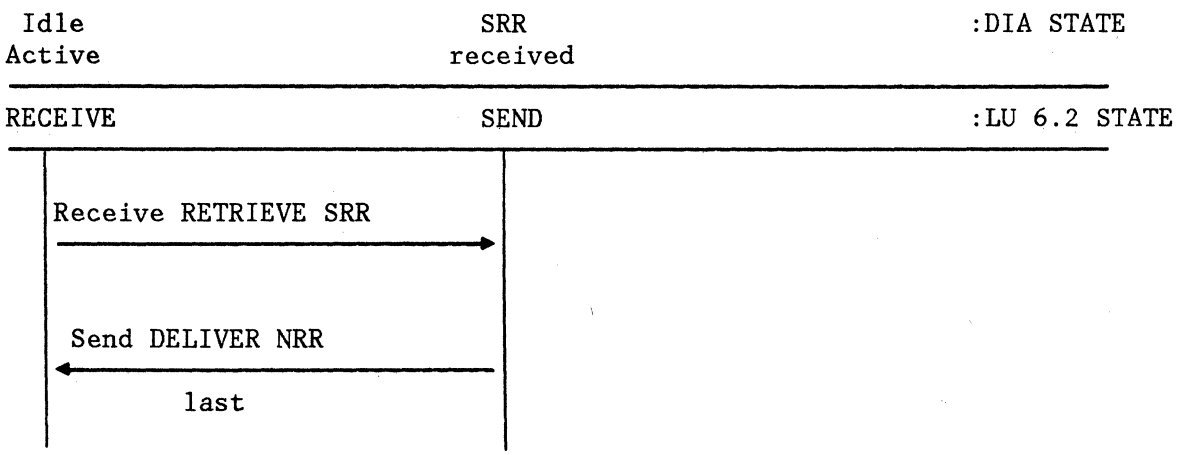


RETRIEVE COMMAND

SENDER OF RETRIEVE

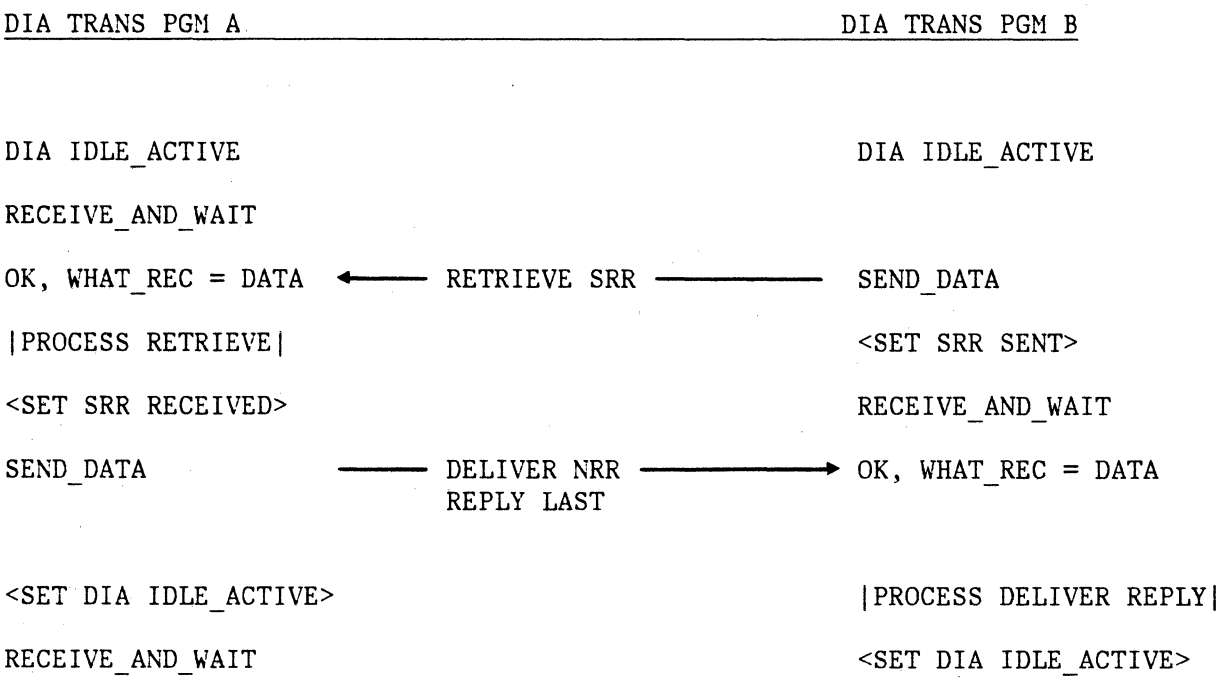


RECEIVER OF RETRIEVE



DIA RETRIEVE SCENARIO 1

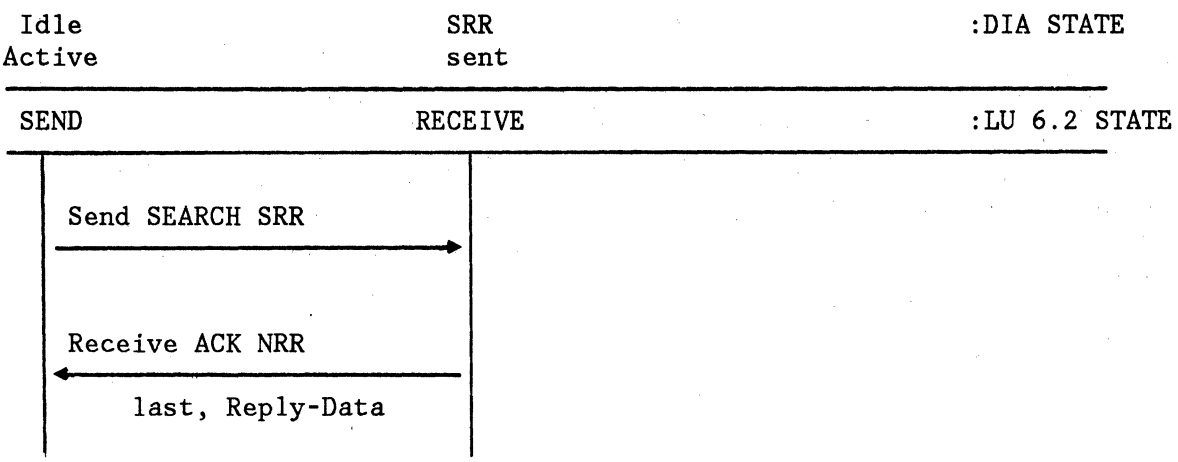
(DOCUMENT DELIVERED NRR)



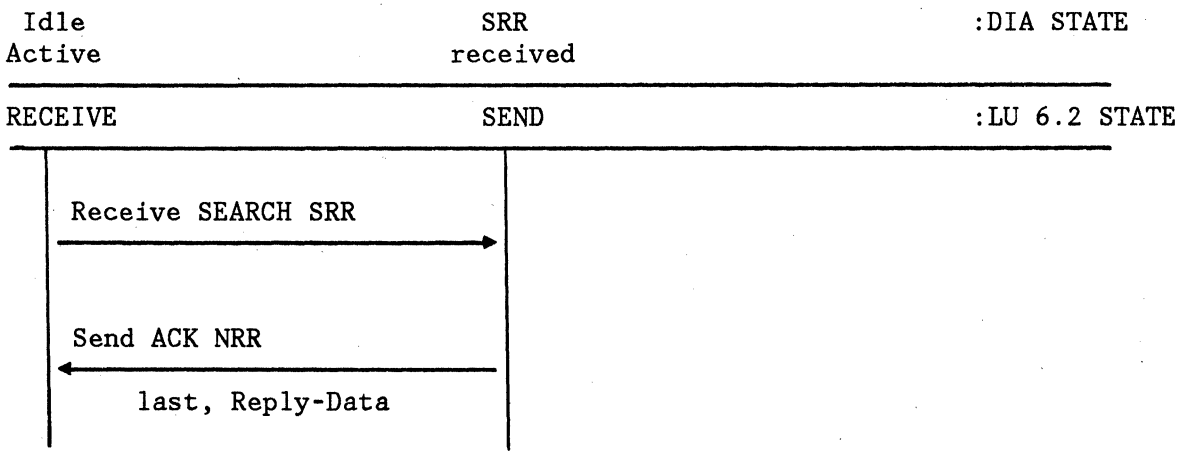
SEARCH COMMAND

SENDER OF SEARCH

(NO DOCUMENTS FOUND CASE)

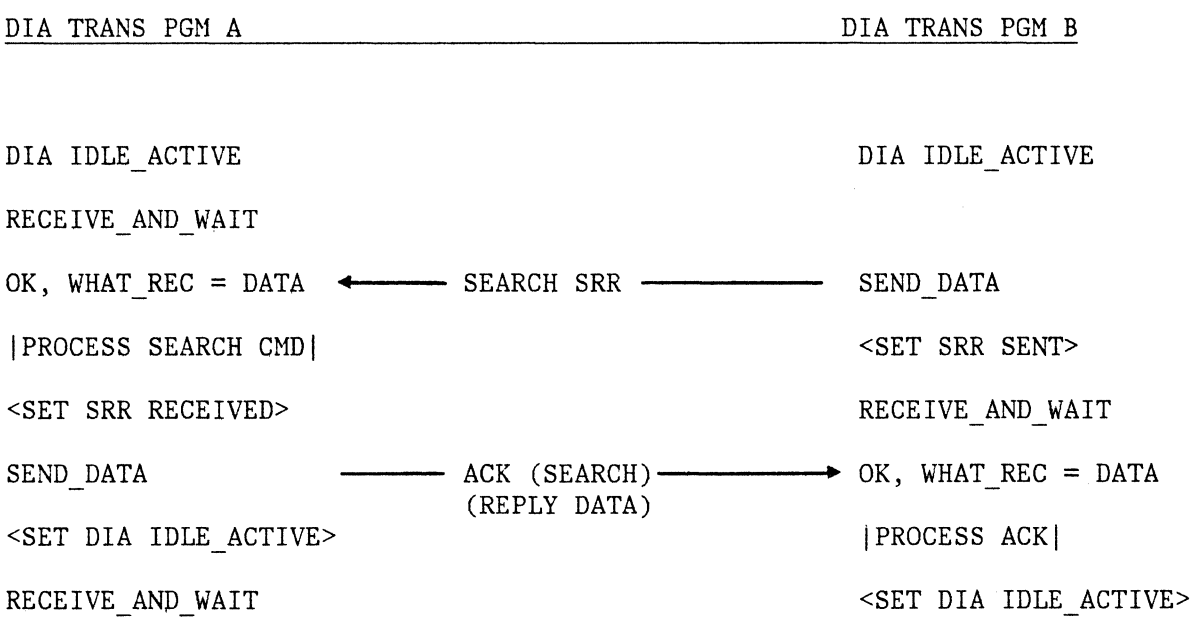


RECEIVER OF SEARCH



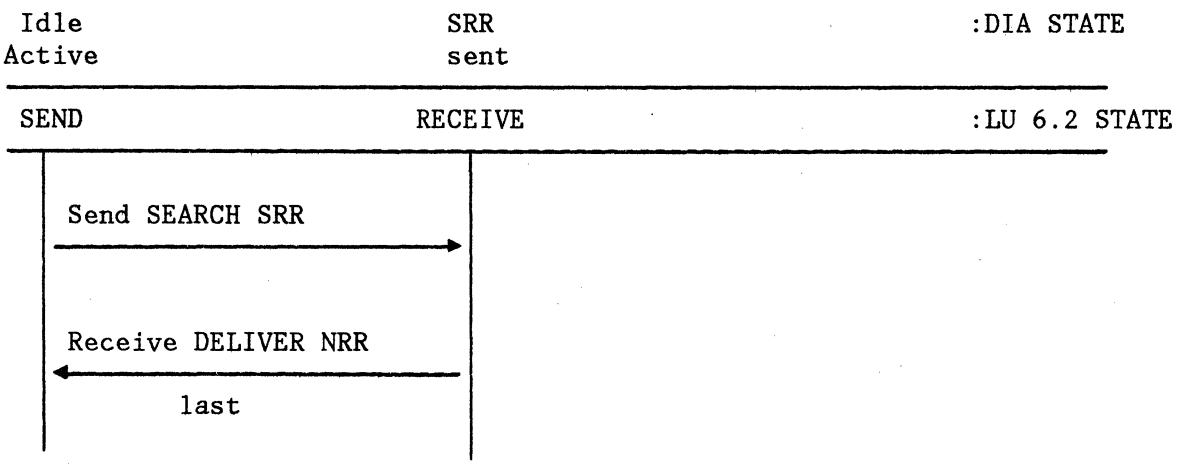
DIA SEARCH SCENARIO 1

(NO DOCUMENTS FOUND CASE)

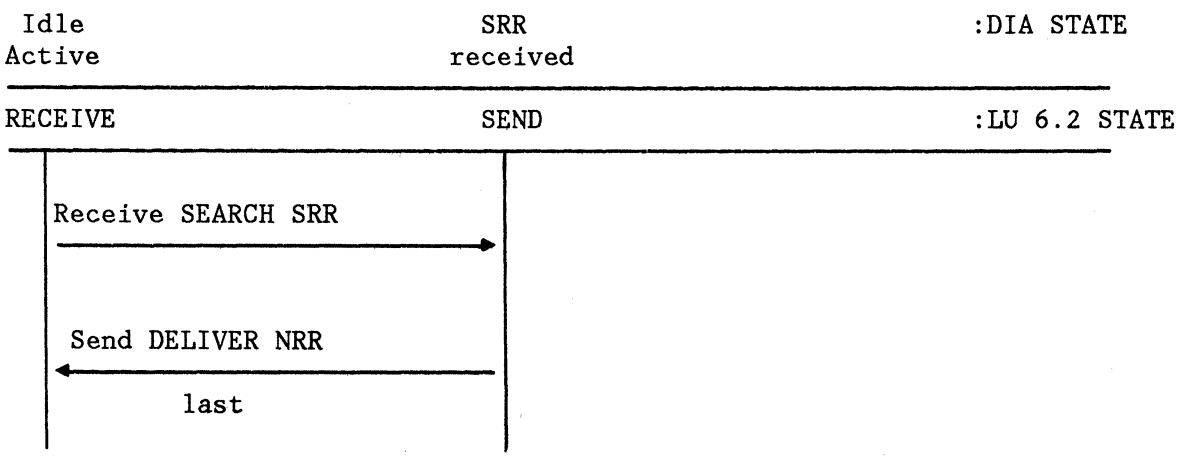


SENDER OF SEARCH

(DOCUMENTS FOUND CASE)

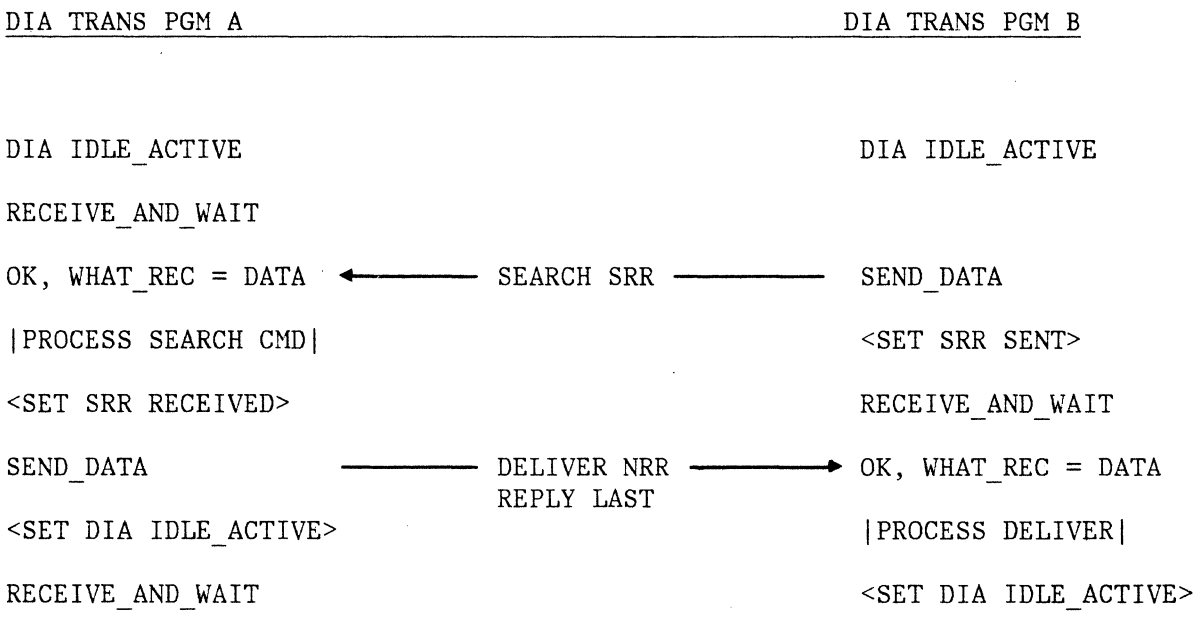


RECEIVER OF SEARCH



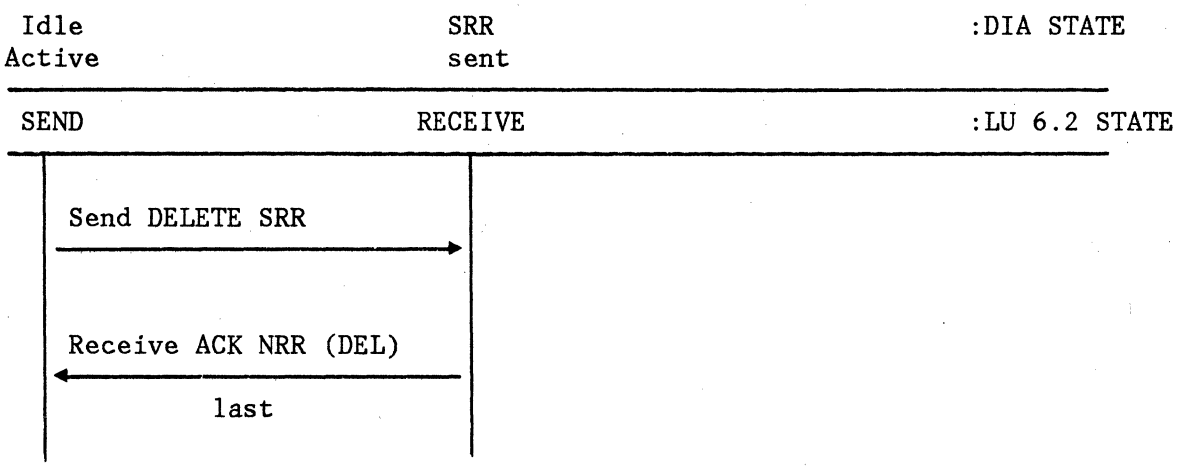
DIA SEARCH SCENARIO 2

(DOCUMENTS FOUND CASE)

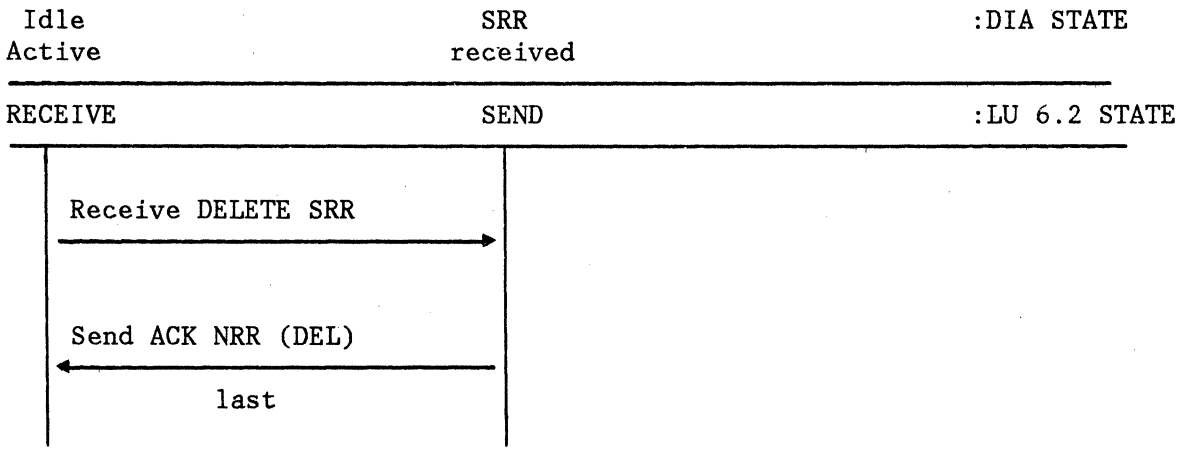


DELETE COMMAND

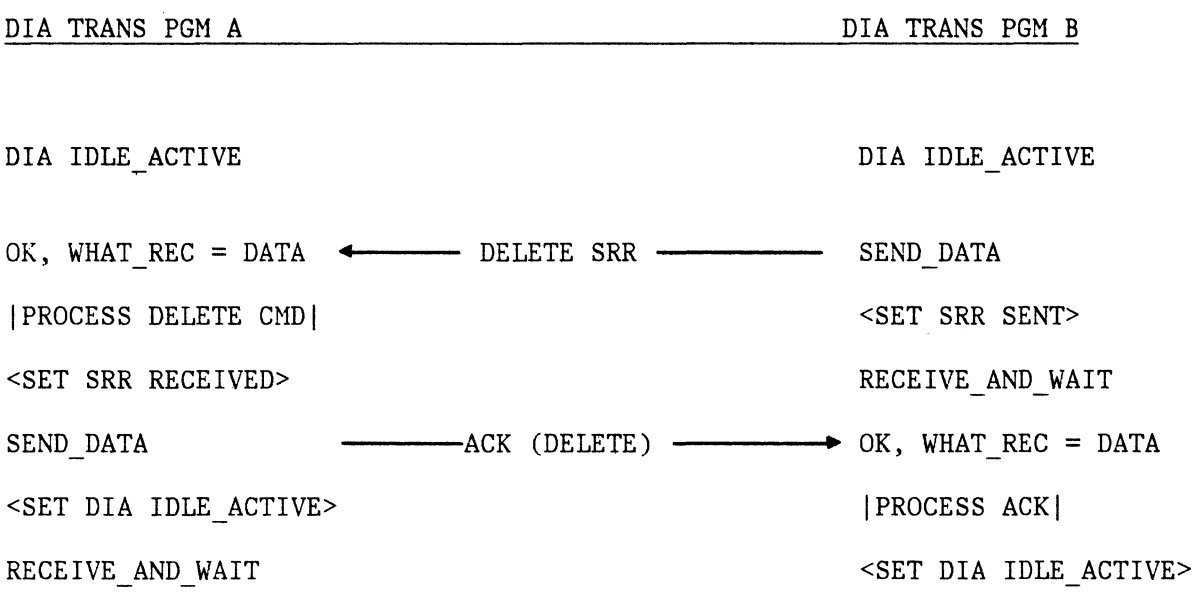
SENDER OF DELETE



RECEIVER OF DELETE



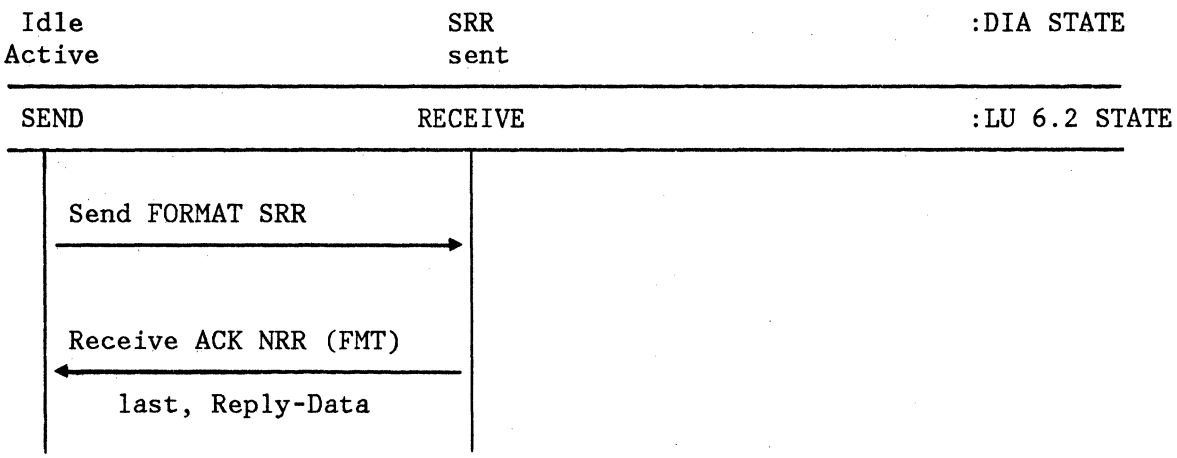
DIA DELETE SCENARIO



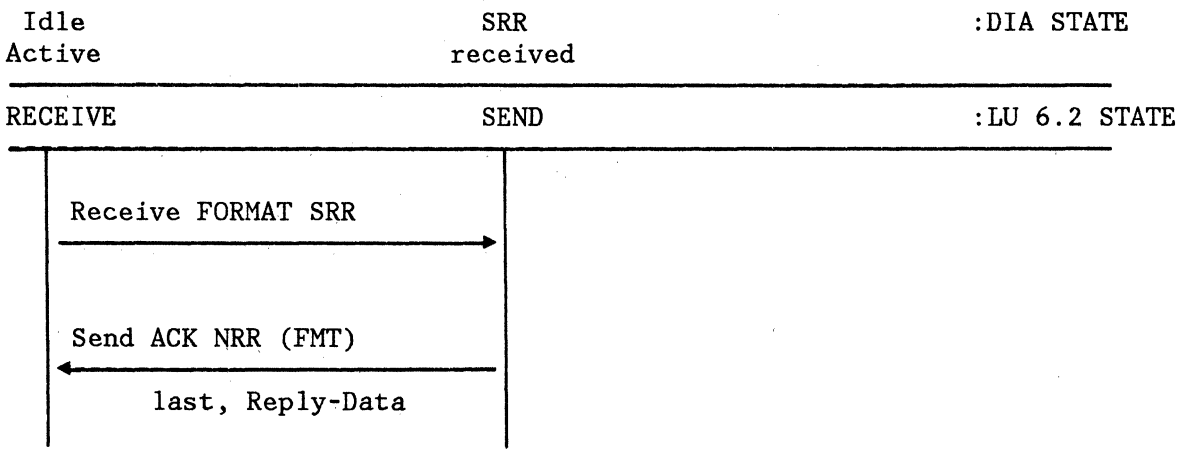
FORMAT COMMAND

SENDER OF FORMAT

(FORMAT AND FILE CASE)

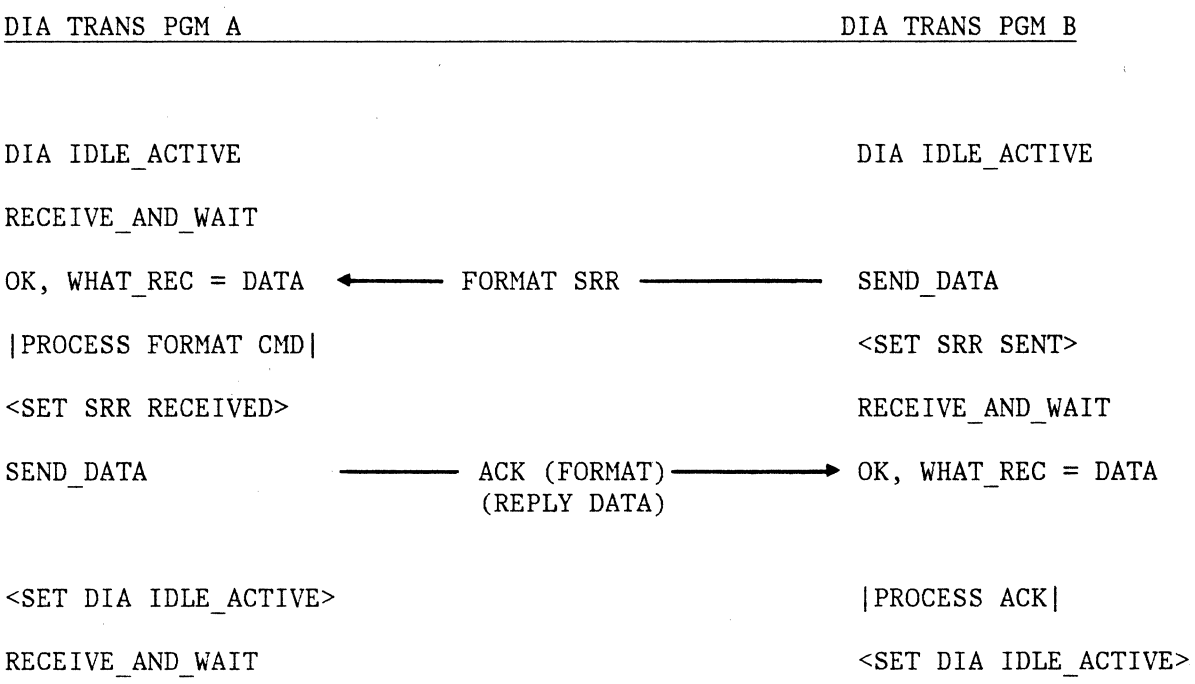


RECEIVER OF FORMAT



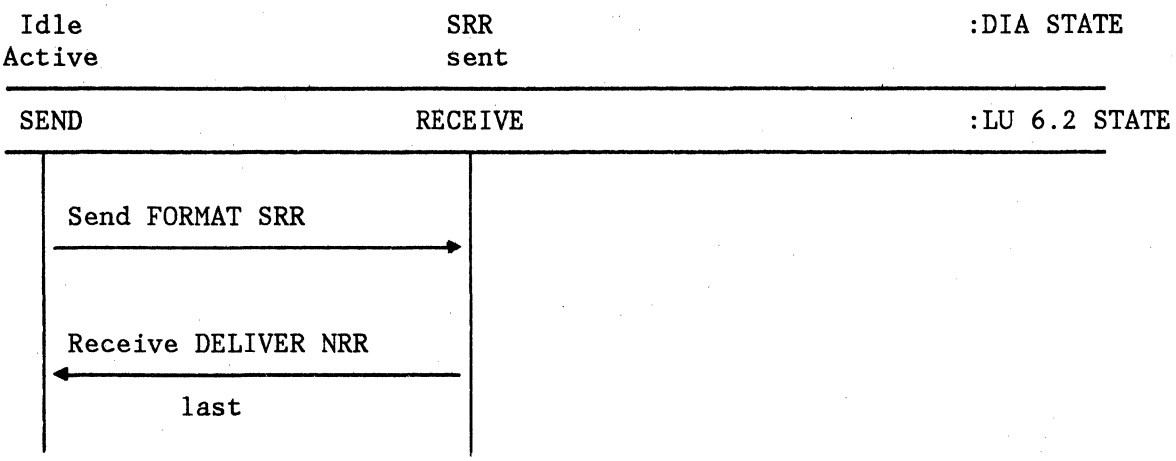
DIA FORMAT SCENARIO 1

(FORMAT AND FILE)

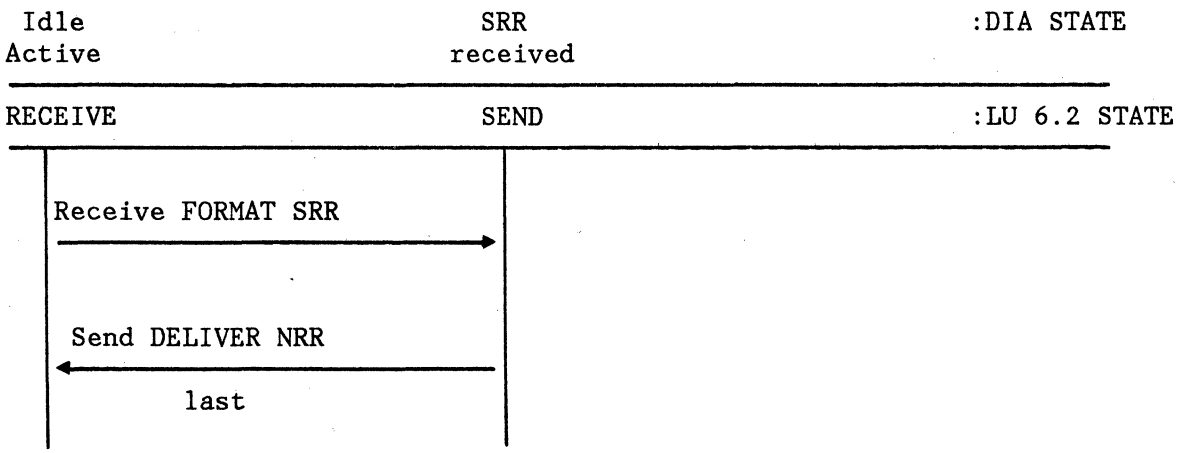


SENDER OF FORMAT

(FORMAT AND DELIVER CASE)

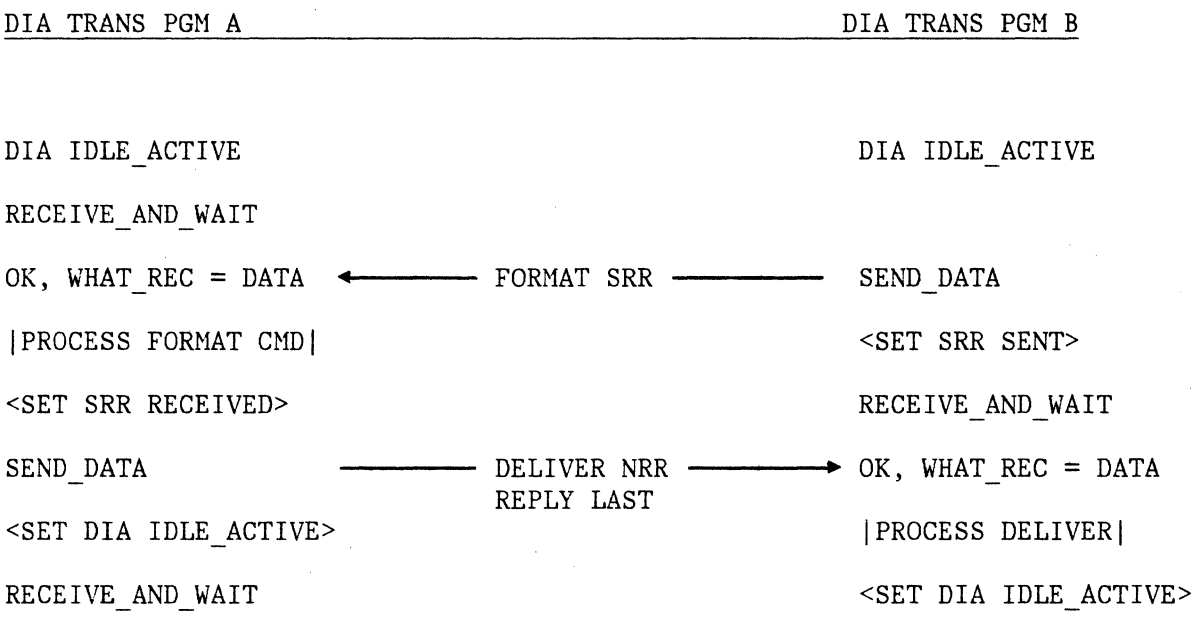


RECEIVER OF FORMAT



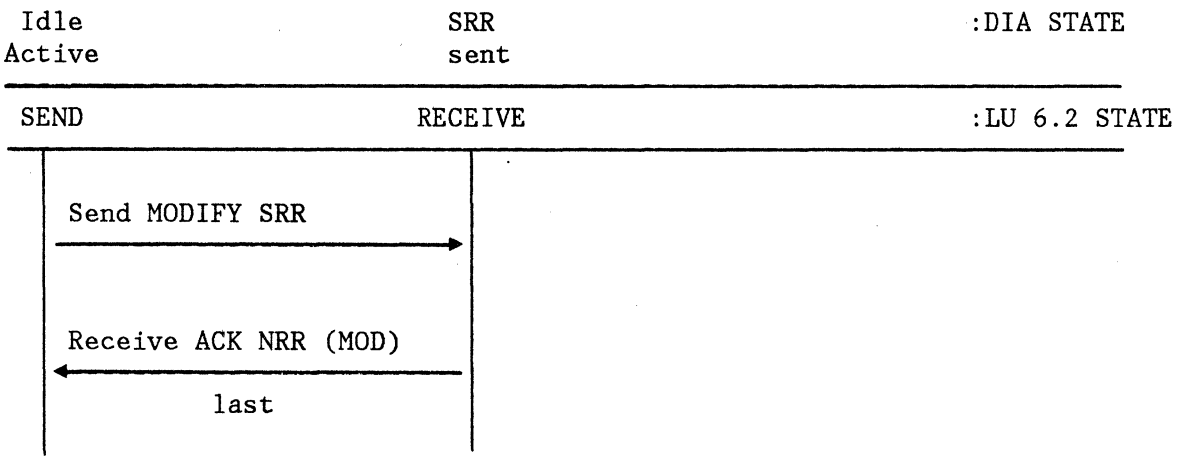
DIA FORMAT SCENARIO 2

(FORMAT AND DELIVER NRR)

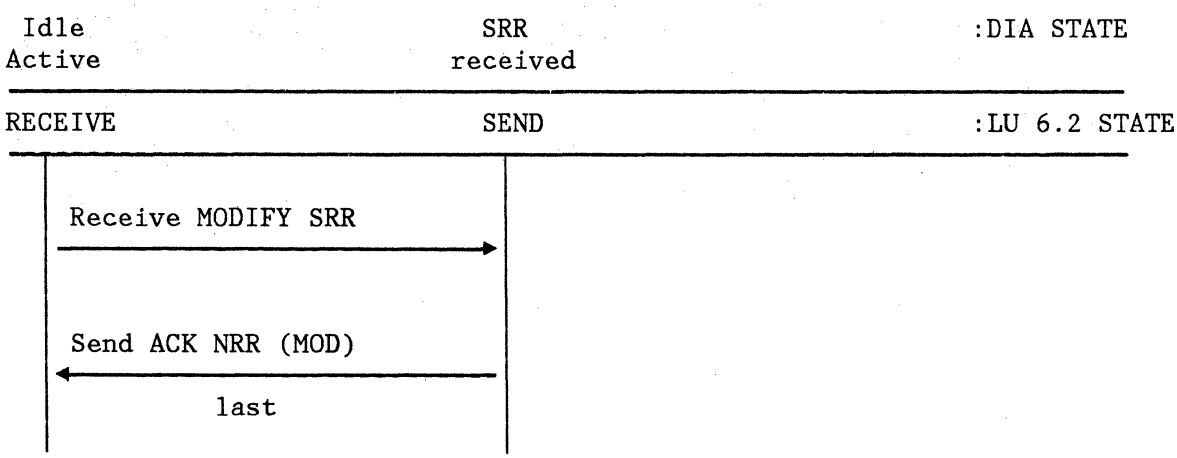


MODIFY COMMAND

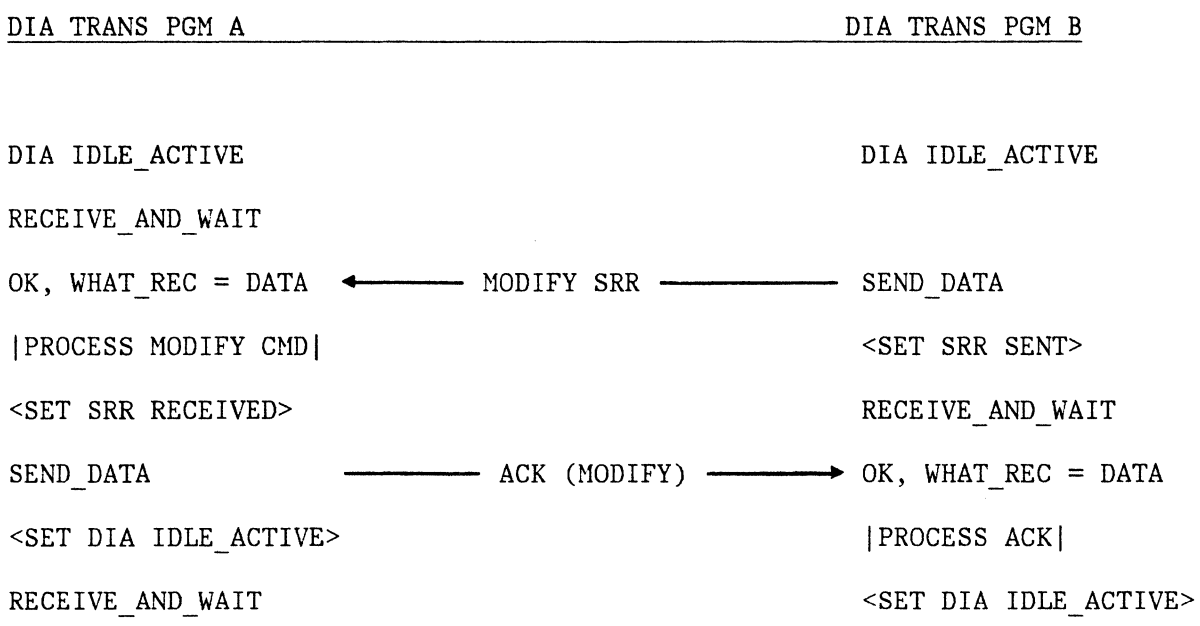
SENDER OF MODIFY



RECEIVER OF MODIFY

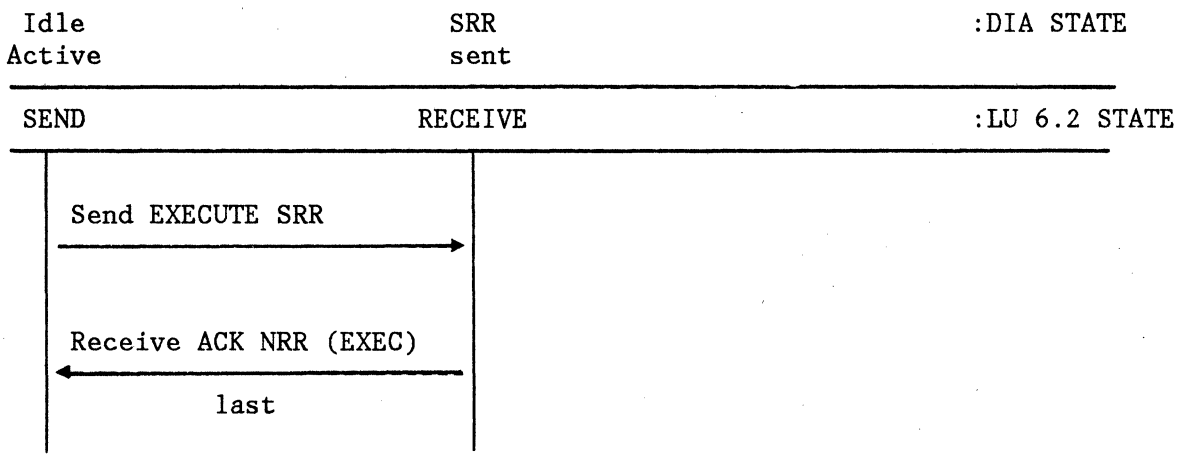


DIA MODIFY SCENARIO

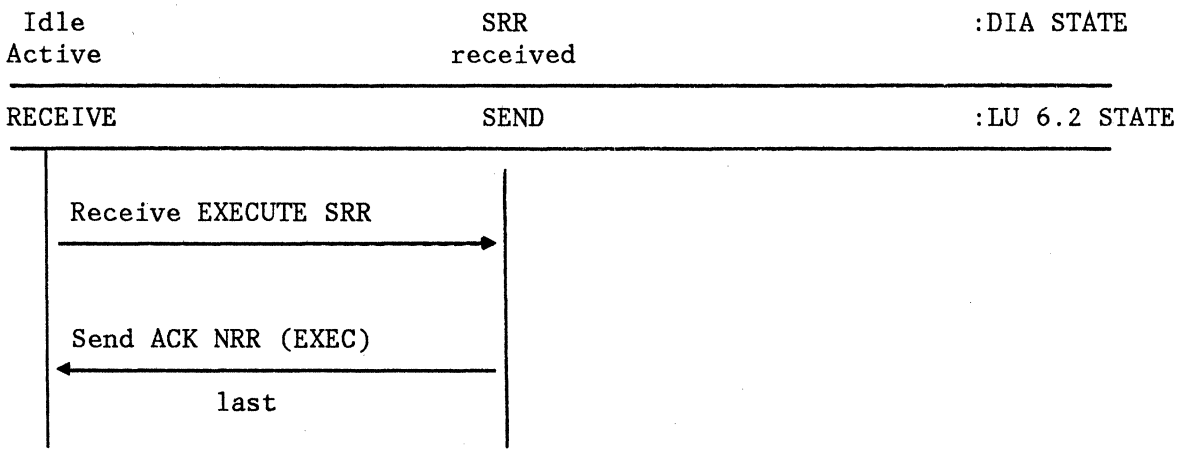


EXECUTE COMMAND

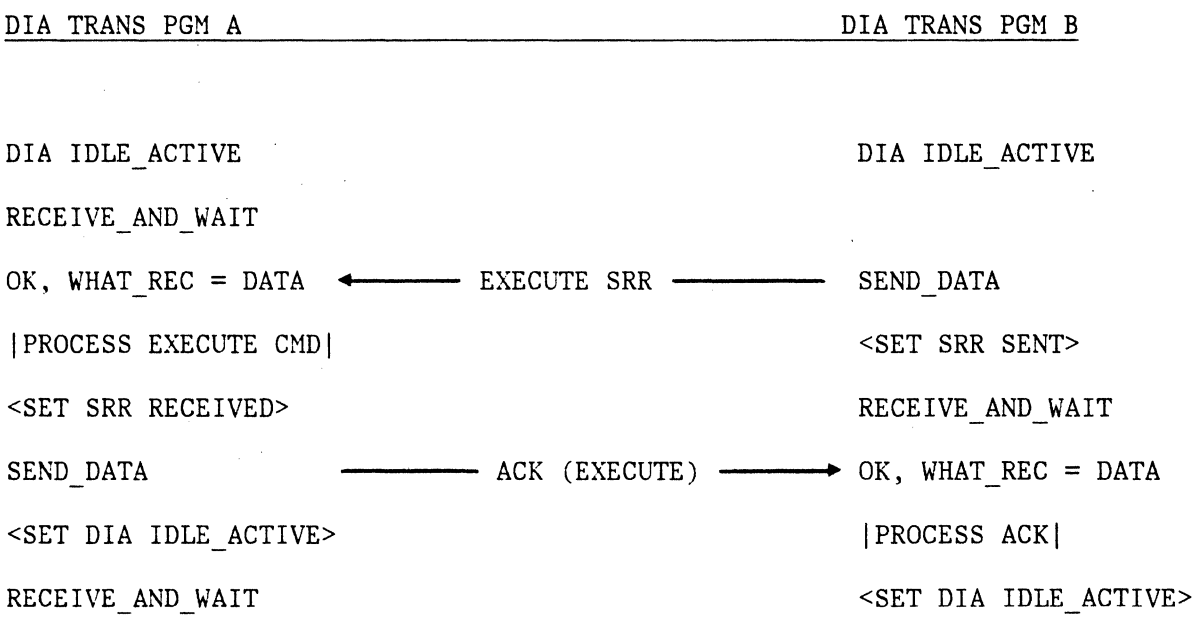
SENDER OF EXECUTE



RECEIVER OF EXECUTE

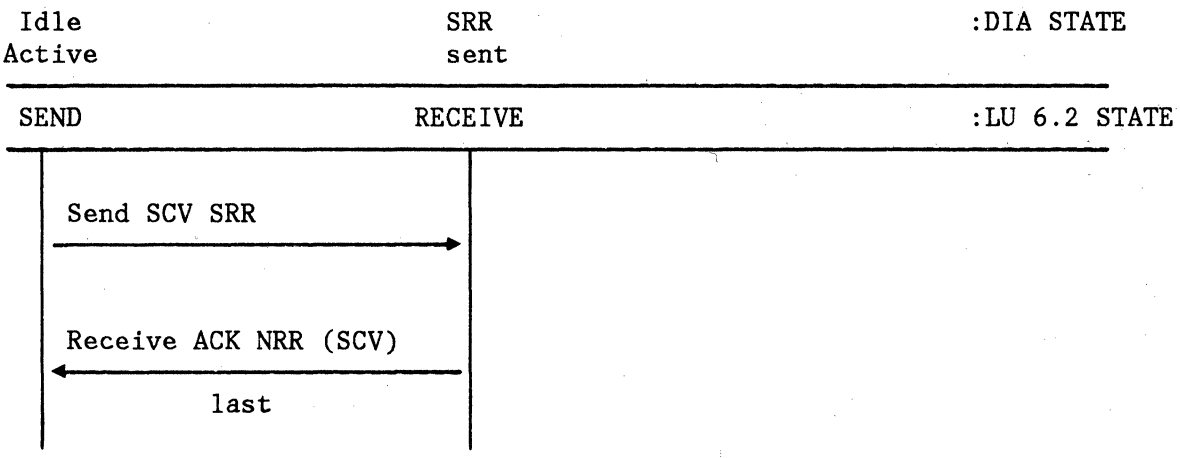


DIA EXECUTE SCENARIO

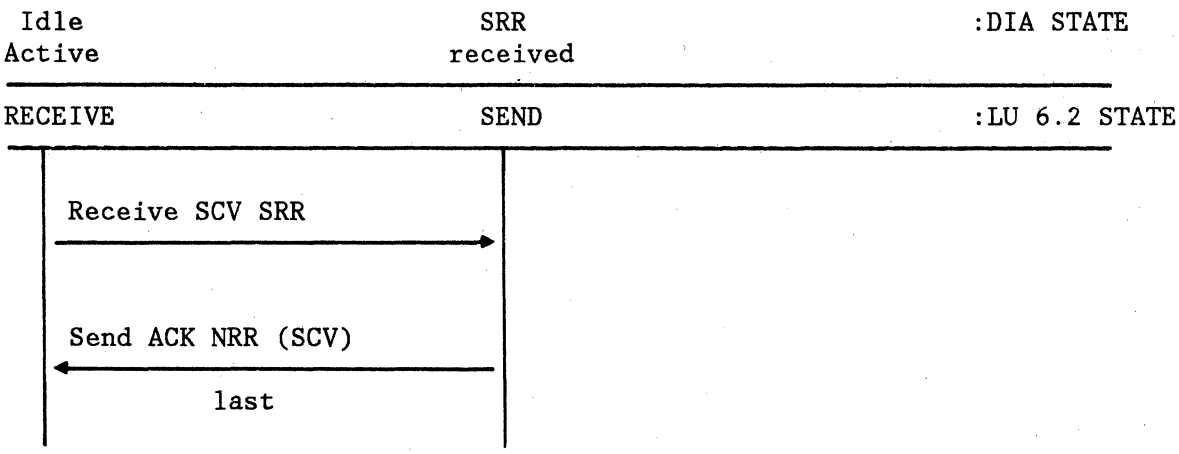


SET-CONTROL-VALUE COMMAND

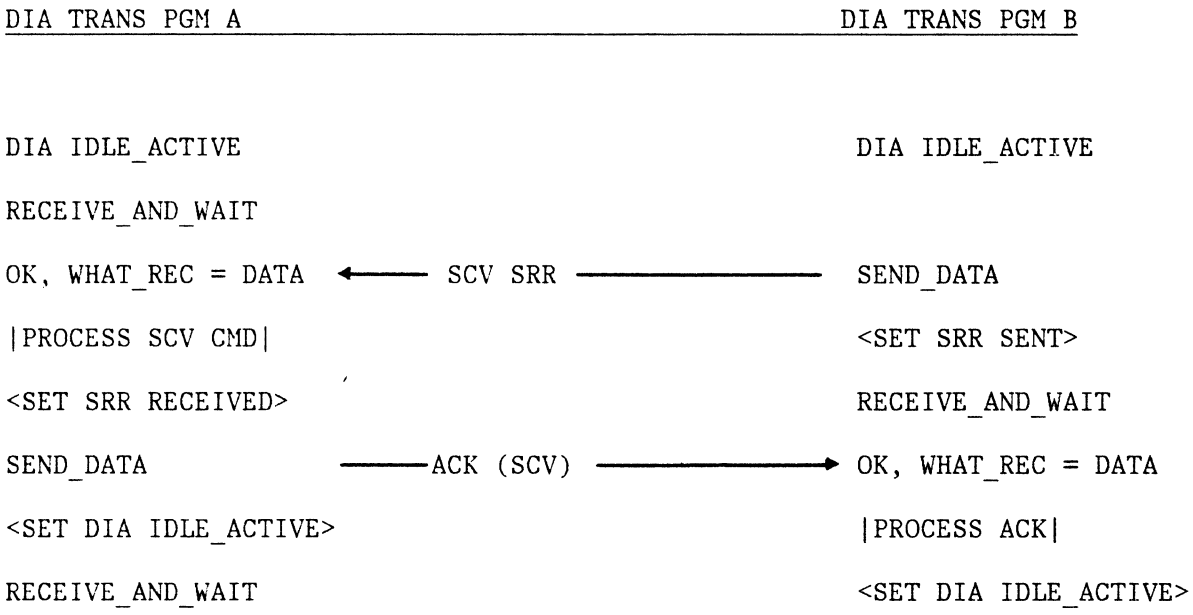
SENDER OF SET-CONTROL-VALUE



RECEIVER OF SET-CONTROL-VALUE



DIA SET-CONTROL-VALUE SCENARIO



GLOSSARY

access code. A 4-byte decimal value, assigned to a document by the primary owner, that determines the set of users allowed to access the document.

address. (1) A character or group of characters that identifies a register, a particular part of storage, or some other data source or destination. (2) In DIA, a 1- to 8-byte character string that identifies the logical components of an office system network. These logical components are: source nodes, recipient nodes, and office system nodes.

affinity. A defined relationship that permits the DIA resources of a source or recipient to be accessed on his behalf by another user.

application processing services. The set of services that provide DIA functions enabling users to access processing capabilities of a remote node.

ARR. Asynchronous reply required.

asynchronous reply required (ARR). A command class that requests asynchronous processing and reply of a DIA function.

COD. Confirmation-of-delivery.

command. The function to be performed by the receiving DIA process.

command sequence. A DIU data stream component containing a set of one or more commands.

condition code. Defines the specific exception condition detected by the receiver of a DIU.

confirmation-of-delivery (COD). An asynchronous message returned to the source node of a distribution request that indicates the information distributed has been delivered to the recipient node.

control variable. A DIA entity maintained by a DIA process for the purpose of verification and authorization.

correlation value. Information used to uniquely identify and correlate the request to the reply.

data unit. A DIU data stream component that contains information referenced by operands of a command in the DIU.

data variable. A variable length collection of information contained in a structured field.

destination node. The office system node that provides services for attached source and recipient nodes.

DIA. Document interchange architecture.

DIA session. A logical connection between two DIA processes that is used to exchange information.

distribution. In general, the function provided by DIA of transporting information from a

source node to one or more recipient nodes.

distribution document name. A unique identifier assigned to each distribution request.

distribution library. The collection of distribution queues and data storage provided by an office system node for the purpose of document distribution.

distribution queue. A queue of distribution and status information to be delivered to source or recipient nodes.

distribution system. The collection of office system nodes, source nodes, and recipient nodes that are interconnected to form an office system network.

DIU. Document interchange unit.

DIU component. A self-defining, variable length structured field. The DIU components are: prefix, command sequence, data unit, document unit, and suffix.

DIU subcomponent. A self-defining, variable length structured field contained within a DIU component.

document. (1) (ISO) A data medium and the data recorded on it, that generally has permanence and that can be read by man or machine. (2) A unified collection of information pertaining to a specific subject or related subjects.

document content introducer. The DIU data stream subcomponent that identifies the beginning of the document content.

document descriptor. A set of profile parameters describing a document that satisfied a document library search request.

document descriptor document. A collection of one or more document descriptors.

document distribution services. The set of services that provide DIA functions enabling users to distribute information in a distribution system.

Document Interchange Architecture (DIA). The specification of rules and data streams necessary to interchange information in a consistent, predictable manner.

document interchange unit (DIU). The basic unit of information exchanged between DIA processes.

document library. A repository on which documents and document related information is stored.

document library services. The set of services that provide DIA functions enabling users to manage the contents of a document library.

document type. A classification that identifies the structure and format of a document.

document unit. A DIU data stream component that contains the document and related document information.

document unit identifier. The DIU data stream subcomponent that contains the document type and system code identifier of the document.

end user. (1) The ultimate source or destination of information flowing through a

system. (2) In DIA, a program, device, person, or system that uses DIA for the purpose of information interchange.

exception condition class. The type of exception condition detected by the receiver of a DIU. The exception classes are: session, syntax, semantic, process, and sender.

exception condition data. A field containing the DIU data stream component or subcomponent that caused the exception condition.

exception condition object. An identifier of the DIU component or subcomponent that caused the exception condition.

format byte. That part of the structure field introducer that defines the format and content of the structured field data variable.

function set. The set of commands that identify the scope of work. Function sets have been defined so that each set contains all commands required for a well-defined, usable, and complete set of functions for a given category of services.

GCID. Graphic character set ID.

graphic character set ID (GCID). The registry for graphic character sets and code pages.

ID. That part of the structured field introducer that defines the class and type of the structured field.

IDP. Interchange document profile.

Interchange Document Profile (IDP). A set of descriptors that identify and describe a document.

introducer. A 5-byte structured field identifier. The introducer contains a 2-byte length field, a 2-byte ID, and a format byte.

introducer extension. An optional extension to the structured field introducer used for segmentation of the structured field.

ISS. Introducer extension.

LADN. Library assigned document name.

library assigned document name (LADN). A unique name assigned to documents filed in the document library.

message. A collection of information transmitted from one point to another.

No reply required (NRR). A command class used when the function requested does not require a reply.

NRR. No reply required.

office system node. The DIA process that provides the services for attached source or recipient nodes.

operand. (1) (ISO) An entity to which an operation is applied. (2) A data stream subcomponent that controls the execution of the command.

originating node. The office system node that provides services for attached source nodes.

OSN. Office system node.

owner-delegate. A user that is designated as secondary owner by the primary owner of the document in the document library.

password. A character string used for validation and authorization to gain access to a resource.

personal. A distribution class of service that requires the recipient to supply a password to receive the distributed information.

prefix. The DIU data stream component that introduces and identifies the DIU.

primary owner. The user who files the document in the document library.

priority. A distribution class of service that prioritizes the distributions so information of higher priority is delivered before information of lower priority.

process. (1) A systematic sequence of operations to produce a specified result. (2) In DIA, a program that uses the DIA rules and data structures to interchange information.

profile parameter. A field of a subprofile that identifies and describes the document.

recipient. An end user that receives information in an office system network.

recipient node. A DIA logical component that provides services on behalf of recipients.

recovery action. The procedure recommended by the process that detected an exception condition.

reply. A command that is used to respond to a previously received request.

request. A command that specifies a function to be performed.

search argument. A search selection criterion that contains the profile parameter identifier, the search data value, and the search comparison operator.

search data parameter set. A collection of one or more search data parameters and the logical operators used to relate them.

search result list. A user named object that contains references to documents selected by the SEARCH command process.

segmentation. The division of a DIU data stream component into two or more segments.

source. An end user that requests services in an office system network.

source node. A DIA logical component that provides services on behalf of sources.

SRR. Synchronous reply required.

structured field. A self-defining, variable length field comprised of an introducer, an optional introducer extension, and a data variable.

subprofile. A set of profile parameters that describe the characteristics and attributes of a document.

suffix. The DIU data stream component that terminates the DIU.

synchronous reply required (SRR). A command class that requests synchronous processing and reply of a DIA function.

system code. An identifier associated with the originator of the document that is contained in a DIU document unit.

user. See end user.

INDEX

Document Interchange Architecture:
Transaction Programmer's Guide
Order No. SC23-0763-0

READER'S COMMENT FORM

You may use this form to communicate your comments about this publication, its organization, or subject matter, with the understanding that IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you. Your comments will be sent to the author's department for whatever review and action, if any, are deemed appropriate.

Note: *Copies of IBM publications are not stocked at the location to which this form is addressed. Please direct any requests for copies of publications, or for assistance in using your IBM system, to your IBM representative or to the IBM branch office serving your locality.*

Possible topics for comment are:
Clarity Accuracy Completeness Organization Coding Retrieval Legibility

If you wish a reply, give your name, company, mailing address, and date:

_____

_____

_____

_____

Note: Staples can cause problems with automated mail sorting equipment. Please use pressure sensitive or other gummed tape to seal this form.

What is your occupation? _____

Number of latest Newsletter associated with this publication: _____

Thank you for your cooperation. No postage stamp necessary if mailed in the U.S.A. (Elsewhere, an IBM office or representative will be happy to forward your comments or you may mail directly to the address in the Edition Notice on the back of the title page.)

SC23-0763-0

Reader's Comment Form

Fold and tape | **Please Do Not Staple** | Fold and tape

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

International Business Machines Corporation
Department D31, Building 803
11400 Burnet Rd.
Austin, Texas 78758

Fold and tape | **Please Do Not Staple** | Fold and tape

IBM

Cut or Fold Along Line

Document Interchange Architecture: Transaction Programmer's Guide Printed in U.S.A. SC23-0763-0

Document Interchange Architecture:
Transaction Programmer's Guide
Order No. SC23-0763-0

READER'S COMMENT FORM

You may use this form to communicate your comments about this publication, its organization, or subject matter, with the understanding that IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you. Your comments will be sent to the author's department for whatever review and action, if any, are deemed appropriate.

Note: *Copies of IBM publications are not stocked at the location to which this form is addressed. Please direct any requests for copies of publications, or for assistance in using your IBM system, to your IBM representative or to the IBM branch office serving your locality.*

Possible topics for comment are:
Clarity Accuracy Completeness Organization Coding Retrieval Legibility

If you wish a reply, give your name, company, mailing address, and date:

_____

_____

_____

_____

Note: Staples can cause problems with automated mail sorting equipment.
Please use pressure sensitive or other gummed tape to seal this form.

What is your occupation? _____

Number of latest Newsletter associated with this publication: _____

Thank you for your cooperation. No postage stamp necessary if mailed in the U.S.A. (Elsewhere, an IBM office or representative will be happy to forward your comments or you may mail directly to the address in the Edition Notice on the back of the title page.)

SC23-0763-0

Reader's Comment Form

Fold and tape **Please Do Not Staple** Fold and tape

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

International Business Machines Corporation
Department D31, Building 803
11400 Burnet Rd.
Austin, Texas 78758

Fold and tape **Please Do Not Staple** Fold and tape

IBM

Cut or Fold Along Line

Document Interchange Architecture: Transaction Programmer's Guide Printed in U.S.A. SC23-0763-0

Document Interchange Architecture:
Transaction Programmer's Guide
Order No. SC23-0763-0

READER'S COMMENT FORM

You may use this form to communicate your comments about this publication, its organization, or subject matter, with the understanding that IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you. Your comments will be sent to the author's department for whatever review and action, if any, are deemed appropriate.

Note: *Copies of IBM publications are not stocked at the location to which this form is addressed. Please direct any requests for copies of publications, or for assistance in using your IBM system, to your IBM representative or to the IBM branch office serving your locality.*

Possible topics for comment are:
Clarity Accuracy Completeness Organization Coding Retrieval Legibility

If you wish a reply, give your name, company, mailing address, and date:

_____

_____

_____

_____

Note: Staples can cause problems with automated mail sorting equipment.
Please use pressure sensitive or other gummed tape to seal this form.

What is your occupation? _____

Number of latest Newsletter associated with this publication: _____

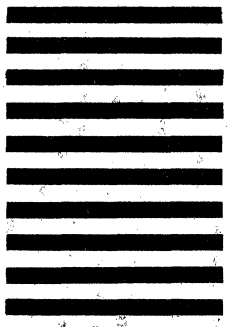
Thank you for your cooperation. No postage stamp necessary if mailed in the U.S.A. (Elsewhere, an IBM office or representative will be happy to forward your comments or you may mail directly to the address in the Edition Notice on the back of the title page.)

SC23-0763-0

Reader's Comment Form

Fold and tape | **Please Do Not Staple** | Fold and tape

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES



BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

International Business Machines Corporation
Department D31, Building 803
11400 Burnet Rd.
Austin, Texas 78758

Fold and tape | **Please Do Not Staple** | Fold and tape

IBM

Cut or Fold Along Line

Document Interchange Architecture: Transaction Programmer's Guide Printed in U.S.A. SC23-0763-0

Document Interchange Architecture: Transaction Programmer's Guide Printed in U.S.A. SC23-0763-0

SC23-0763-0