INTEGRATED INFORMATION
SUPPORT SYSTEM (IISS)
Volume VI - Network Transaction Manager Subsystem
Part 2 - NTM Programmer's Guide

General Electric Company
Production Resources Consulting
One River Road
Schenectady, New York 12345

November 1985

Approved for public release; distribution is unlimited.

PREPARED FOR:
MATERIALS LABORATORY
AIR FORCE WRIGHT AERONAUTICAL LABORATORIES
AIR FORCE SYSTEMS COMMAND
WRIGHT-PATTERSON AFB, OH 45433-6533
NOTICE

When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely related Government procurement operation, the United States Government thereby incurs no responsibility nor any obligation whatsoever; and the fact that the government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication or otherwise as in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

This report has been reviewed by the Office of Public Affairs (ASD/PA) and is releasable to the National Technical Information Service (NTIS). At NTIS, it will be available to the general public, including foreign nations.

This technical report has been reviewed and is approved for publication.

David L. Judson, Project Manager
AFWAL/MLTC
WRIGHT PATTERSON AFB OH 45433

DATE 5 Aug 1986

FOR THE COMMANDER:

Gerald C. Shumaker, Branch Chief
AFWAL/MLTC
WRIGHT PATTERSON AFB OH 45433

DATE 7 Aug 86

"If your address has changed, if you wish to be removed from our mailing list, or if the addressee is no longer employed by your organization please notify AFWAL/MLTC, W-PAFB, OH 45433 to help us maintain a current mailing list."

Copies of this report should not be returned unless return is required by security considerations, contractual obligations, or notice on a specific document.
This technical manual describes the services provided by the Network Transaction Manager (NTH) for application processes (APs). Information on integrating new APs with IISS and writing new APs for IISS is provided. Each NTH service call is described in detail.
11. Title

Integrated Information Support System (IISS)
Vol VI - Network Transaction Manager Subsystem
Part 2 - NTM Programmer's Guide

<table>
<thead>
<tr>
<th>Accession For</th>
</tr>
</thead>
<tbody>
<tr>
<td>NTIS CDAEE</td>
</tr>
<tr>
<td>DTIC TAB</td>
</tr>
<tr>
<td>Unannounced</td>
</tr>
<tr>
<td>Justification</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Distribution/Availability Codes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dist</td>
</tr>
<tr>
<td>A-1</td>
</tr>
</tbody>
</table>
This programmer’s guide covers the work performed under Air Force Contract F33615-80-C-5155 (ICAM Project 6201). This contract is sponsored by the Materials Laboratory, Air Force Systems Command, Wright-Patterson Air Force Base, Ohio. It was administered under the technical direction of Mr. Gerald C. Shumaker, ICAM Program Manager, Manufacturing Technology Division, through Project Manager, Mr. David Judson. The Prime Contractor was Production Resources Consulting of the General Electric Company, Schenectady, New York, under the direction of Mr. Allan Rubenstein. The General Electric Project Manager was Mr. Myron Hurlbut of Industrial Automation Systems Department, Albany, New York.

Certain work aimed at improving Test Bed Technology has been performed by other contracts with Project 6201 performing integrating functions. This work consisted of enhancements to Test Bed software and establishment and operation of Test Bed hardware and communications for developers and other users. Documentation relating to the Test Bed from all of these contractors and projects have been integrated under Project 6201 for publication and treatment as an integrated set of documents. The particular contributors to each document are noted on the Report Documentation Page (DD1473). A listing and description of the entire project documentation system and how they are related is contained in document FTR620100001, Project Overview.

The subcontractors and their contributing activities were as follows:

**TASK 4.2**

<table>
<thead>
<tr>
<th>Subcontractors</th>
<th>Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boeing Military Aircraft Company (BMAC)</td>
<td>Reviewer</td>
</tr>
<tr>
<td>D. Appleton Company (DACOM)</td>
<td>Responsible for IDEF support, state-of-the-art literature search</td>
</tr>
<tr>
<td>General Dynamics/ Ft. Worth</td>
<td>Responsible for factory view function and information models</td>
</tr>
</tbody>
</table>
### Subcontractors Role

**Illinois Institute of Technology**
- Responsible for factory view function research (IITRI) and information models of small and medium-size business

**North American Rockwell**
- Reviewer

**Northrop Corporation**
- Responsible for factory view function and information models

**Pritsker and Associates**
- Responsible for IDEF2 support

**SofTech**
- Responsible for IDEF0 support

### TASKS 4.3 - 4.9 (TEST BED)

<table>
<thead>
<tr>
<th>Subcontractors</th>
<th>Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boeing Military Aircraft Company (BMAC)</td>
<td>Responsible for consultation on applications of the technology and on IBM computer technology.</td>
</tr>
<tr>
<td>Computer Technology Associates (CTA)</td>
<td>Assisted in the areas of communications systems, system design and integration methodology, and design of the Network Transaction Manager.</td>
</tr>
<tr>
<td>Control Data Corporation (CDC)</td>
<td>Responsible for the Common Data Model (CDM) implementation and part of the CDM design (shared with DACOM).</td>
</tr>
<tr>
<td>D. Appleton Company (DACOM)</td>
<td>Responsible for the overall CDM Subsystem design integration and test plan, as well as part of the design of the CDM (shared with CDC). DACOM also developed the integration Methodology and did the schema mappings for the Application Subsystems.</td>
</tr>
</tbody>
</table>
**Subcontractors**

<table>
<thead>
<tr>
<th>Subcontractors</th>
<th>Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>Digital Equipment Corporation (DEC)</td>
<td>Consulting and support of the performance testing and on DEC software and computer systems operation.</td>
</tr>
<tr>
<td>McDonnell Douglas Automation Company (McAuto)</td>
<td>Responsible for the support and enhancements to the Network Transaction Manager Subsystem during 1984/1985 period.</td>
</tr>
<tr>
<td>On-Line Software International (OSI)</td>
<td>Responsible for programming the Communications Subsystem on the IBM and for consulting on the IBM.</td>
</tr>
<tr>
<td>Rath and Strong Systems Products (RSSP) (In 1985 became McCormack &amp; Dodge)</td>
<td>Responsible for assistance in the implementation and use of the MRP II package (PIOS) that they supplied.</td>
</tr>
<tr>
<td>SofTech, Inc.</td>
<td>Responsible for the design and implementation of the Network Transaction Manager (NTM) in 1981/1984 period.</td>
</tr>
<tr>
<td>Software Performance Engineering (SPE)</td>
<td>Responsible for directing the work on performance evaluation and analysis.</td>
</tr>
<tr>
<td>Structural Dynamics Research Corporation (SDRC)</td>
<td>Responsible for the User Interface and Virtual Terminal Interface Subsystems.</td>
</tr>
</tbody>
</table>

Subcontractors and other prime contractors under other projects who have contributed to Test Bed Technology, their contributing activities and responsible projects are as follows:

<table>
<thead>
<tr>
<th>Subcontractors</th>
<th>Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>General Dynamics/ Ft. Worth</td>
<td>Responsible for factory view</td>
</tr>
<tr>
<td>Contractors</td>
<td>ICAM Project</td>
</tr>
<tr>
<td>-----------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>Boeing Military Aircraft Company</td>
<td>1701, 2201, 2202</td>
</tr>
<tr>
<td>Control Data Corporation (CDC)</td>
<td>1502, 1701</td>
</tr>
<tr>
<td>D. Appleton Company (DACOM)</td>
<td>1502</td>
</tr>
<tr>
<td>General Electric</td>
<td>1502</td>
</tr>
<tr>
<td>Hughes Aircraft Company (HAC)</td>
<td>1701</td>
</tr>
<tr>
<td>Structural Dynamics Research Corporation</td>
<td>1502, 1701, 1703</td>
</tr>
</tbody>
</table>
# TABLE OF CONTENTS

## SECTION 1.0 INTRODUCTION ..................................... 1-1

## SECTION 2.0 THE IISS ENVIRONMENT .......................... 2-1

## SECTION 3.0 INTEGRATING NEW USER APPLICATION PROCESSES (APs) INTO THE IISS ENVIRONMENT .................. 3-1

1. Application Process Interface for New Applications ........................................ 3-1
   1.1 Overview of the NTM Services ............................................... 3-1
   1.2 Initiation (INITIAL) .................................................. 3-6
   1.3 SEND Messages .......................................................... 3-6
   1.4 RECEIVE Messages ..................................................... 3-9
   1.5 Termination (TRMNAT) .................................................. 3-10
   1.6 AP-AP Communication - The Logical Channel Concept .................................. 3-10

2. Writing New Application Processes for the IISS Test Bed .................................. 3-15

## SECTION 4.0 INTEGRATING EXISTING APs ...................... 4-1

## SECTION 5.0 NTM SERVICE CALLS (Alphabetic by Entry Name) ................................. 5-1

1. Services Available to All Categories of IISS "User" .................................... 5-1
2. Services Available Only to IISS Components and System Operator ....................... 5-52
3. Services Which Will Be Available as Future Enhancements ................................ 5-62
4. Description of Parameter Used in NTM Service Calls .................................... 5-63

## APPENDIX A NTM MESSAGES OF CONCERN TO APs ................ A-1

## APPENDIX B AP CHARACTERISTICS ................................ B-1

## APPENDIX C INTEGRATING IISS COMPONENT APs .................. C-1

## APPENDIX D HELPFUL HINTS ........................................ D-1

## APPENDIX E REMOTE COMPILe AND LINK SPECIFICATION ............... E-1

1. RCL Overview ..................................................... E-1
   1.1 Expert Layer .................................................. E-2
   1.2 Novice Layer ................................................ E-4
TABLE OF CONTENTS (Continued)

E.1.3  RCLETEST ................................................. E-4

APPENDIX F  RCL PROGRAMMER’S GUIDE .......................... F-1

LIST OF ILLUSTRATIONS

<table>
<thead>
<tr>
<th>Figure</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2-1</td>
<td>IISS SYSTEM EXTERNAL INTERFACES</td>
<td>2-2</td>
</tr>
<tr>
<td>2-2</td>
<td>IISS ARCHITECTURE -</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CONCEPTUAL MODEL NTM ON VAX</td>
<td>2-3</td>
</tr>
<tr>
<td>3-1</td>
<td>NTM/AP INTERFACES</td>
<td>3-2</td>
</tr>
<tr>
<td>3-2</td>
<td>IISS AP-COBOL PROCEDURE DIVISION STRUCTURE</td>
<td>3-3</td>
</tr>
<tr>
<td>3-3</td>
<td>NTM SERVICES CALLS</td>
<td>3-5</td>
</tr>
<tr>
<td>3-4a</td>
<td>Simple Paired Message Handling - Logical Channels</td>
<td>3-12</td>
</tr>
<tr>
<td>3-4b</td>
<td>Multiple Pairs Between</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Two APs Using Logical Channels</td>
<td>3-12</td>
</tr>
<tr>
<td>3-4c</td>
<td>Multiple Pairs Between</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Many Destination APs</td>
<td>3-13</td>
</tr>
<tr>
<td>3-4d</td>
<td>AP Chaining - Logical Channels</td>
<td>3-14</td>
</tr>
<tr>
<td>3-4e</td>
<td>Maintain a Communication Path Using Logical Channels</td>
<td></td>
</tr>
<tr>
<td>C-1</td>
<td>NTM - UI Interface</td>
<td>C-1</td>
</tr>
<tr>
<td>C-2</td>
<td>NTM - COMM Interface</td>
<td>C-4</td>
</tr>
<tr>
<td>E-1</td>
<td>RCL Program Flow</td>
<td>E-2</td>
</tr>
<tr>
<td>E-2</td>
<td>RCL Software Architecture Diagram</td>
<td>E-3</td>
</tr>
</tbody>
</table>
## LIST OF SYMBOLS, ABBREVIATIONS, ACRONYMS

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AP</td>
<td>Application Process</td>
</tr>
<tr>
<td>APC</td>
<td>Application Process Cluster</td>
</tr>
<tr>
<td>API</td>
<td>Application Process Interface</td>
</tr>
<tr>
<td>CDMRP</td>
<td>Common Data Model Request Processor</td>
</tr>
<tr>
<td>COMM</td>
<td>Communications Handler</td>
</tr>
<tr>
<td>CPC</td>
<td>Computer Program Configuration Item</td>
</tr>
<tr>
<td>DBMS</td>
<td>Data Base Management System</td>
</tr>
<tr>
<td>DML</td>
<td>Data Manipulation Language</td>
</tr>
<tr>
<td>ICAM</td>
<td>Integrated Computer Aided Manufacturing</td>
</tr>
<tr>
<td>IDSS</td>
<td>Integrated Decision Support System</td>
</tr>
<tr>
<td>IISS</td>
<td>Integrated Information Support System</td>
</tr>
<tr>
<td>IPC</td>
<td>Inter Process Communication</td>
</tr>
<tr>
<td>LAN</td>
<td>Local Area Network</td>
</tr>
<tr>
<td>MCKM</td>
<td>Manufacturing Control Material Management</td>
</tr>
<tr>
<td>MDL</td>
<td>Message Definition Language</td>
</tr>
<tr>
<td>MM</td>
<td>Message Manager</td>
</tr>
<tr>
<td>MO</td>
<td>Maintain Operability</td>
</tr>
<tr>
<td>MPU</td>
<td>Message Processing Unit</td>
</tr>
<tr>
<td>MRP</td>
<td>Materials Requirements Planning</td>
</tr>
<tr>
<td>MSG</td>
<td>Message</td>
</tr>
<tr>
<td>NTM</td>
<td>Network Transaction Manager</td>
</tr>
<tr>
<td>OS</td>
<td>Operating System</td>
</tr>
<tr>
<td>PM</td>
<td>Process Manager</td>
</tr>
<tr>
<td>QA</td>
<td>Quality Assurance</td>
</tr>
<tr>
<td>SS</td>
<td>System Specification</td>
</tr>
<tr>
<td>UI</td>
<td>User Interface</td>
</tr>
<tr>
<td>VAX</td>
<td>Trademark of Digital Equipment Corporation: 32 bit minicomputer</td>
</tr>
<tr>
<td>VMS</td>
<td>Trademark of Digital Equipment Corporation: The VAX OS</td>
</tr>
</tbody>
</table>
SECTION 1
INTRODUCTION

This NTM Programmer’s Guide describes the services provided to IISS Programmers by the Network Transaction Manager (NTM). These services are used by IISS Application Programs to send messages to and receive messages from other programs in the IISS.

The Programmer’s Guide is being released as a series of interim documents to allow the IISS coalition members who are now building IISS component programs to use the currently available NTM service calls in their programs. The available services have been used both in the development of NTM APs and in the development of IISS component APs. The calls described for these services are in their final calling form.

Throughout the document there are notes on restrictions, helpful hints, and various “lessons learned.” For the user’s convenience, these are collected in Appendix D.
SECTION 2
THE IISS ENVIRONMENT

The Integrated Information Support System (IISS) is a test computing environment used to investigate and demonstrate techniques for integrating data resident on heterogeneous databases supported on heterogeneous computers. The environment will support users' and application processes' access to this integrated resident IISS data in a controlled distributed processing environment. Network Transaction Manager (NTM) components that reside on each host cooperate to perform the coordination, communication and housekeeping functions required to integrate the application processes into the IISS system and to allow the integrated APs the access to the data and to each other within a well-defined authorization structure. The NTM components form the distributed executive of the IISS.

The IISS architecture is described in Figures 2-1 and 2-2. The architecture is based on the concept of cooperating clusters of application processes (AP clusters). Each cluster is a collection of Application Processes (APs) that are uniquely addressable but may form a subsystem or application from the user's perspective. Each cluster has its own dedicated portion of the NTM to provide services directly to each AP residing on the cluster. Certain AP clusters support IISS system components. These include the Communications (COMM), the Common Data Model Request Processor (CDMRP), and the User Interface (UI) clusters. These system components in combination with the entire NTM are the IISS network operating system and utilities. They cooperate to provide transaction processing, communication, and data integration services to users at IISS terminals and to Application Processes.
Notes:

1. The IISS operator interface will be implemented as an IISS terminal or as a separate console interface. It is treated as a separate entity.

2. For the initial test bed, the User Interface (UI) will reside on only one host, the VAX. There will be one UI instance for each IISS terminal that is logged on.

3. The Communication components are named to indicate the link pair (i.e., COMM V-H indicates the component on the VAX that communicates with the Honeywell Level 6).

Figure 2-1. IISS System External Interfaces. An MTM will reside on each of the three hosts configured as illustrated to IISS components and other subsystems.

2-2
Figure 2-2. IISS Architecture - Conceptual Model NTM on VAX
A major function of the NTM is process management for all IISS APs. As part of this process management, an interface between each AP and the NTM is provided. It is called the AP Interface and is described in detail in Sections 3, 4, and 5. It provides a set of high level NTM calls for use by the AP, that are, in concept, similar to traditional operating system calls. The functionality represented by these calls is provided by using the message services of the NTM. The message services are transparent to the AP.
SECTION 3

INTEGRATING NEW USER APPLICATION PROCESSES (APs) INTO THE IISS ENVIRONMENT

3.1 Application Process Interface for New Applications

3.1.1 Overview of the NTM Services

The Application Process (AP) Interface is a group of subroutines that are linked to each new AP to provide the integration of the AP into the IISS Test Bed. Conceptually the AP, AP Interface, and NTM interfaces are described in Figure 3-1.

The number of input mailboxes for an AP is specified as part of the AP characteristics as 0, 1, or 2. If an AP will receive no messages (e.g., all of its information is in a shared database), it will have zero mailboxes. To receive normal messages from other APs, then a single mailbox is specified. This will be referred to as the “cold” or “normal” mailbox by the NTM. If an AP wants to receive high priority messages from the NTM, such as “shutdown pending,” then it must specify two mailboxes. This second mailbox will be referred to by the NTM as the “hot” or “priority” mailbox. (Note: The priority mailbox is not accessible for sending messages between applications, as there is no message priority logic supported by the NTM). A third mailbox type, the “Acknowledgment” mailbox, or “ACK” mailbox, is automatically provided by the system if one or two mailboxes are specified by the user. This is used strictly for the NTM MPU to send back acknowledgment messages to the NTM services to indicate that messages have been received and are acceptable. In this way the ACK messages can go straight back and do not become queued up with other messages in either the hot or cold mailboxes. Thus, in fact, there are 0, 2, or 3 mailboxes actually used, and even if 0 are specified, one is provided during startup to receive startup information, but this is not seen by the user.
The AP Interface provides an AP with the ability to send messages to other APs and NTM components in the IISS. These communicating APs may be associated with the same or different AP Clusters and may reside on different IISS host computers. The AP programmer can send messages to any AP that it is authorized to access by using the NTM high level "send" calls. The programmer need only know the IISS name of the AP to which it is sending the message, and the AP-AP message protocol (message types, etc.). The NTM provides the routing and message delivery processing for the APs.

The only restriction placed on an AP that is to be integrated into the IISS, and therefore use the message services of the NTM, is that the AP be written according to a format that includes NTM Initiation (INITAL) and termination (TRMNAT) calls at the beginning and the end of its Procedure Division (COBOL) (Figure 3-2). The CALL "INITAL" and CALL "TRMNAT," respectively, provide the IISS connection and termination service. Communication between IISS APs is accomplished by using the NTM calls that are described in Section 5. For example, the CALL "NSEND" USING ..., will cause a program's message to be delivered to the NTM for routing. The NTM routes the messages through the IISS to the destination AP specified in the "NSEND" call. The destination AP receives its messages by using the NTM "RCV" call.
null
Figure 3-3 lists the NTM Service calls. There are four functional categories of calls: connection services, communication services, NTM Requests, and privileged services. Most APs will use only the connection and communication services. The NTM Request Services provide status information that can help the AP optimize its performance. They also offer processing services to APs, such as the UI and CDMRP, that have special NTM handling requirements.

The basic connection services ("INITAL" and "TRNMAT") and communications services (sending messages and receiving messages) are described from a functional viewpoint in Sections 3.1.2 through 3.1.4. All of the calls, with their arguments and return codes, are described in Section 5.
Connection Services
- **INITIAL** Provide initiation services for an AP
- **TRMNAT** Signal AP termination status
- **ENDRCY** Signal end of recovery processing

Communication Services
- **MSEND** Send a message
- **GDSEND** Send a guaranteed delivery message
- **ISEND** Send an initiation request
- **QSEND** Send reply message (used by Queue-Server APs only)
- **CHKMSG** Check for any current messages (use RCV to retrieve messages)
- **SETDLY** Specify delay condition for next message
- **SIGERR** Notify the NTM and UI of an AP (non-fatal) error
- **GDACK** Acknowledge receipt of a guaranteed delivery message
- **MSGACK** Acknowledge receipt of a message
- **RCV** Receive a message
- **TSTMOD** Switch IISS message test mode on or off

NTM Requests
- **APSTAT** Get the status of a specified AP
- **HSTATS** Get the status of a specified host
- **WHHTST** Request the name of the current host
- **WHATAC** Request the name of the current AP Cluster
- **WKONCA** Request "wake-up" on specified AP Cluster availability
- **ACSTAT** Get the status of a specified AP Cluster
- **SIGABT** Signal to NTM to abort an AP
- **PRSTAT** Get the status of one or more paired messages
- **GDSTAT** Get the status of one or more guaranteed delivery messages
- **GETUSR** Get the user's name and Original Source APname

Privileged Services
- **INICOM** Provide initiation services for the COMM APs
- **INITEX** Provide initiation services for UI AP
- **LOGON** Send IISS user information to NTM
- **LOGOFF** Send IISS user Logoff information to NTM
- **CHGROL** Change the users's role during a session
- **TRMNAX** Signal COMM AP Termination Status

* Services currently available for use by IISS AP developers.

Figure 3-3. NTM Services Calls
3.1.2 **Initiation ("INITAL")**

INITAL sets up the NTH to provide message services to all APs except for COMM and the UI (COMM and UI special initiation requirements are described in Appendix C). Messages are routed and delivered by the NTH through ‘mailboxes’. INITAL provides the AP mailbox connection to the NTH, performs the initiation logic necessary to handle later NTH calls from the AP, and establishes IISS condition handling for the AP. The condition handling service** traps all machine and operating system exception conditions, informs the NTH of the event (who informs the original source AP of the event, if possible), and then aborts the AP. This service provides an additional level of integrity checking to the IISS.

The NTH initiation service also provides system state information to APs that perform special logic on certain events (i.e., IISS startup, IISS Recovery, or for AP startup events such as "First Run of AP"). The system state message also provides the AP Interface with certain AP characteristics such as the number of mailboxes the AP supports.

Section 5 contains a more complete description of INITAL and its arguments. It also includes an example of its use and guidelines for establishing buffer space for the APs’ messages.

3.1.3 **SEND Messages**

APs may send messages to other APs in the IISS by using the basic message delivery services of the NTH. Four different send calls are provided to APs. The first, "NSEND", is used for normal AP-AP communication. The others, "GDSEND" (guaranteed delivery)**, "ISEND" (specific initiation of a new AP instance), and "QSEND" (queue-server reply message), all request special NTH message services. Additionally, the user can specify delayed, conditional or test mode delivery with special calls to enable these services prior to actually sending the message. The following paragraphs contain a description of the basic ‘send’ functions followed by a more detailed description of the three “send” calls.

**This service is not implemented.
The AP Interface provides the following basic functions on all of the send calls:

a. The AP Interface receives a message from an AP that is to be delivered to another AP when the AP issues any one of the four send calls.

b. The Interface encapsulates the AP’s data (message) into NTM messages. The AP’s data will be packetized into NTM message units that have an NTM header which is used for NTM processing and routing. The NTM also provides message continuation logic, transparent to the AP, for data strings which are longer than the NTM maximum message size.

c. The AP Interface delivers these messages containing the user’s data to the local MPU’s mailbox.

d. The AP Interface returns the NTM’s accept-status of the AP’s message to the AP. A successful return indicates that the NTM has performed an integrity check on the message header, authorized the message, set up pairing information for messages that require a response, and has accepted the message for delivery. A nonsuccessful return will indicate where the send failed NTM processing. Failure may be due to invalid calling arguments, send service processing errors, or NTM table errors.

e. Some messages sent by an AP require responses from the receiving AP. These are called message pairs. The NTM provides message pairing support to the APs on the "NSEND" and "ISEND" calls. An AP can indicate that a particular message requires a response, and hence, pairing support from the NTM, by setting the timeout indicator argument of the "send" calls with an appropriate value. The NTM will provide the APs required timeout service where the elapsed time expires before a response has returned. The AP developer indicates the AP timeout handling requirements to the IISS CDM administrator before the AP becomes part of the IISS. (See Appendix B for AP characteristics and Handling Options.)
f. An AP may request conditional or time-triggered delivery of a message by using the "SETDLY" call just prior to a "send" call. "SETDLY" provides three triggered delivery modes for a message: 1) a specified absolute time (i.e., 12:58 p.m., June 20, 1984), 2) delivery after some elapsed relative time (i.e., four hours after the message is submitted), or 3) on some condition (i.e., after Event X).

g. An AP can establish a test mode of operation with the "TSTMOD" service. The "TSTMOD" call allows the Test Mode service to be either enabled or disabled. All messages that are sent while the sending AP is in test-mode will have the NTM header test-mode field set to "1". The destination AP will receive the message with a "RCV" call whose return code will be set to indicate that the message was sent in test mode. The receiving AP then must handle the test-mode message appropriately (i.e., perhaps inhibiting file updates). The TSTMOD call further enables the AP to receive messages signalling an AP error condition (SIGERR).

CALL "NSEND" USING ... is used to send data that requires no special NTM handling beyond message pairing and continuation logic. The caller specifies the destination AP name and may use a logical channel specifier (see Section 3.1.6 and Section 5) to manage communications between the sending and receiving APs.

CALL "GDSEND" USING ... is used to send a guaranteed delivery message to an AP. (A guaranteed delivery message is a "registered letter" message to the NTM that this message will be delivered to its destination. The NTM provides special logging for later delivery of the message if a required host or AP cluster is not available when the message is sent, and recovery of these messages in the event of a system "crash.") The "GDSEND" call returns the NTM's message serial number to the sending AP in the event that the sending AP wishes to later determine the status of the message (CALL "GDSTAT" USING MSG-SERIAL-NO). The destination AP, on receipt of a guaranteed delivery message, must acknowledge that receipt (CALL "GDACK") when the AP completes the processing of the guaranteed delivery message.

CALL "ISEND" USING ... is the "send" call that should be used when an AP is aware that its message will require the initiation of a new instance of the destination AP. It can also
be used by APs who require concurrent access to multiple instances of the same AP. This service allows those APs (primarily component APs) to specifically request the initiation of a new instance of an AP. If the AP writer is in doubt whether to use an "ISEND" or "NSEND", "NSEND" should be the one chosen. The NTM can normally determine whether or not an AP requires an initiation, and will handle these messages accordingly.

The "ISEND" call must be used in the situation where the destination AP has the characteristic of requiring a specific initiation message. This restriction is placed on the AP by the AP Developer when the AP is installed on the IISS. It serves to prevent the initiation of the AP upon receipt of unsolicited messages. This restriction applies only to initiation messages. Once the AP is running, it may receive any message from an authorized source AP using CALL "NSEND".

CALL "QSEND" USING... is the send call to use if you are a Queue-Server Application. Q-servers are a special type of application that is usually used by component APs to get information that is kept system- (or at least machine-) wide. By keeping up one copy of a program that can be called by any number of applications, this program is common to all APs. Every time it gets a message, it sends back the needed information as soon as possible. In this way it is really replying to the last message as if it was a request of some sort. The QSEND call guarantees that the reply is sent specifically to the sender of the last message received.

3.1.4 RECEIVE Messages

An AP receives messages from other APs in the IISS by using the CALL "RCV" USING ... service. The AP has two basic options available to it on the "RCV" call. These are a wait/no-wait option and a receive any message/receive specific message option. These options are described below.

WAIT/NO WAIT OPTION: If the AP uses a "RCV" with the wait argument set, its processing will be suspended until a message arrives (the message being either the message it was waiting for or an indicator of a time-out). On a "RCV" with "no wait", the AP will have control returned to it immediately. The return code on the "RCV" with "no wait" will either indicate that a message was received and can then be retrieved in the call's DATA argument or will
indicate that no message of the type indicated has arrived for the AP.

RECEIVE ANY/RECEIVE SPECIFIC MESSAGE OPTION: On a "RCV" call, an AP can use this option in several ways. An AP may request to receive the first message in its buffer regardless of the source, the first message from a specific AP; the first from a specific AP on a specified channel (Section 3.1.6); or the first message from any AP on a specified channel. This feature provides flexibility to the AP programmer and off loads some of the AP's message bookkeeping and buffering. For example, if an AP has issued multiple "sends" and needs a response from a particular AP before it can continue, it can issue a "RCV" with a wait and the source specified. The AP Interface will buffer all of the AP's incoming messages until the requested response arrives and then return control and the requested message to the AP.

A companion service to "RCV" is "CHKMSG". It checks the AP's mailbox to determine whether any messages have arrived for the AP. (The AP can also check to see if a message has arrived from a particular source or channel by specifying one or both of these in the call). The AP can later retrieve any messages indicated by "CHKMSG" with the call "RCV". The "CHKMSG" service should be particularly useful to AP programs that receive unsolicited messages. For example, an AP that can receive IISS shutdown messages could periodically call "CHKMSG" to determine if it has received a shutdown message.

3.1.5 Termination (TRMNAT)

"TRMNAT" disconnects the AP from the NTM, does any required cleanup, (such as mailbox deletion) and then terminates the AP. It is required as the last executable statement in a new IISS AP's program.

3.1.6 AP-AP Communication - the Logical Channel Concept

The NTM supports several modes of communication between IISS APs by providing a logical channel specifier capability. Logical channels are an optional argument in the send and receive calls that can be used by the APs to:

- Pair Request and Response Messages. The source AP supplies a logical channel specifier with the destination AP's name on a send data request. The
destination AP receives the logical channel specifier on the receive data call (CALL "RCV") and uses it when it returns the response with its CALL "NSEND." See Figure 3-4a. In this way, a message pair can be confirmed by matching the pair source AP and the logical channel. This is critical when the response comes from an AP that is not the destination specified in the original request message.

- Pair Multiple Outstanding Requests and Responses. The AP can use logical channel specifiers to manage multiple outstanding message pairs between one or several AP destinations. (See Figures 3-4b and 3-4c.) The requesting AP manages the pairs by using the destination AP and logical channel to identify a unique pair. For example, in Figure 3-4b, AP1 sends four messages to AP2, each on a different channel. AP2, whose protocol determines that it expects multiple messages from AP1, issues four consecutive CALL "RCV"s (Section 5). On each of its RCVs, it obtains AP1's name and the logical channel specifier for the specific message. AP2 pairs the requests from AP1 to its responses by using the relevant logical channel when it sends a response to AP1. This provides an easy and flexible pair management capability to the AP Programmer. Note that the AP must manage the assignment of the logical channel to the message.
API specifies logical channel 1 on its "CALL SEND" message.

AP2 receives a message from API on Channel 1 and uses Channel 1 when responding to API.

L.C. = Logical Channel

**Figure 3-4a. Simple Paired Message Handling - Logical Channels**

**Figure 3-4b. Multiple Pairs Between Two APs Using Logical Channels**
Figure 5-4c. Multiple Pairs Between One AP and Many Destination APs
There is a restriction on this use of logical channels. When an AP wishes to communicate with a second AP on multiple channels, it must specify the destination instance when a new channel ID is introduced. This requires that the source AP has received at least one message from the destination AP in order to obtain the destination AP's instance number. This restriction applies only when dealing with AP's requiring child chaining support.

Support AP Chaining. The AP can maintain a chain of communications between APs by using the Logical Channel specifier as the AP chain identifier (Figure 3-4d). Each AP in the chain will use the same logical channel when sending messages to or receiving messages from any other AP in the chain. The NTM supports the chaining by dynamically building a chain or child table as the chain is built. In this table, the child AP is assigned the channel IS specified in the message causing it’s initiation. This function allows any AP in the chain to get the name of the chain’s originating AP and the channel of this chain (CALL "GETUSR" ... , see Section 5). With this information, any AP in the chain can send a message directly to the chain’s originator.

Figure 3-4d. AP Chaining-Logical Channels
Maintain a Communication Path Between Two AP’s. An AP can maintain a "telephone like" conversation (Figure 3-4e) with another AP by reserving one channel for this function. The APs support this link by using the same logical channel on their "send" and "receive" calls. This is most useful to an AP that maintains simultaneous communication with multiple APs.

When a request message is sent, it is quite possible for the requesting AP to receive the response from an AP other than the destination specified in the original "send" call. (As an example: API requests data from AP2. AP2 does not have the data itself and in its turn spawns AP3 with instructions to obtain the data and send it directly to API. In this situation, the logical channel is the critical variable in matching the request and response message pair. Therefore, in order to support these message pairs it is necessary to place a restriction on the use of the channel numbers. This restriction requires that the requesting AP use unique channel numbers for each outstanding message pair. This should not impact an AP's functionality and allows the NTM to ensure the integrity of its pairing and chaining logic. Figure 3-4c shows the supported use of unique channel numbers for pairs.

3.2 Writing New Application Processes for the IISS Test Bed

The following are guidelines for writing new applications for the IISS Test Bed.
1. For COBOL applications, the body of the Procedure Division must have the basic format illustrated in Figure 3-2.

2. If the AP provides logic for special startup conditions (IISS startup or recovery), then it must include logic to test the startup-status and perform the appropriate code indicated by the startup-status return. (See INITIAL example, Section 5.)

3. If the AP expects to receive unsolicited messages (time-outs, shutdown requests, etc.), then it will periodically request these messages using CALL "RCV"; or CALL "CHKMSG" followed by CALL "RCV".

4. If the AP characteristic record (Appendix B) indicates that the AP does special shutdown processing on IISS shutdown, then the AP must also service unsolicited messages in the manner described above in Item 3. On recognition of a shutdown message, it must perform its shutdown logic and then call "TRMNAT".

5. The AP must be linked or bound with the AP Interface routines for IISS execution.

6. The AP must be integrated into the IISS testbed through the CDM Administrator. The AP writer must indicate the AP's characteristics (see Appendix B) and execution requirements on a provided form.

7. The AP must supply the IISS AP name of the APs to which it sends messages on the NTM "send" calls. This name includes the directory prefix specifying the directory where the destination AP's executable module resides.

8. All AP Interface service return codes are described in the library file "SRVRET". Integrated APs can use this file by using the statement "COPY SRVRET OF IISSCLIB" in the Working-Storage Section of the data division. The legal values of the service's parameters are given in Section 5 of this document.

9. If the AP expects to receive high priority messages
PRM620142000
1 November 1985

(get user response, IISS shutdown pending, IISS Shutdown cancelled), it must support a "hot" mailbox.
SECTION 4

INTEGRATING EXISTING APs

Existing APs may be integrated into the IISS to use the NTM and other system services to the extent that the existing APs can be modified to use IISS services. The changes that may have to be made are specific to any application, so only general guidelines can be given here.

The minimum requirement is that the AP be modified to include the INITAL and TRMNAT services as explained in Section 3.1.1. With these, the AP can be called by the NTM and properly terminated and disconnected from the NTM. Beyond that, the AP can use any of the services as described in Section 3 if it is properly modified.

Since each application may be different, it is not possible to specify exactly how to integrate existing types of applications, but following are some guidelines of the types of things that should be looked for in the existing applications:

- All calls to the operating system must be reviewed for compatibility with IISS.
- Any input/output logic must be reviewed and probably revised. The application is not connected to a terminal except by way of the User Interface. Direct calls to a terminal, Cobol Display statement, for example, will go to the Operator's terminal.
- All user interaction through terminals must be converted to User Interface services (described in the User Interface manuals).
- Any use of "Event Flags" and "Mailboxes" (VAX system services) must be compatible with IISS.

The AP must, of course, be properly installed and its characteristics specified to the NTM like any other application.
SECTION 5

NTM SERVICES

5.1 Services Available to All Categories of IISS "User"

Services noted with an asterisk (*) are currently available. The service return parameter values are defined for each call. Examples of the uses of certain calls are also given.

The values of the service returns are defined in the include member "SRVRET".
ACSTAT

Get the status of a specified AP cluster.

Calling Sequence:

CALL "ACSTAT" USING AP-CLUSTER-NAME,
      RET-CODE.

Description:

ACSTAT returns to the caller a status code containing relevant
information about the AP cluster which was specified in the
request.

Inputs:

   AP-CLUSTER-NAME

Outputs:

   RET-CODE

RET-CODE Values:
(Values are defined in the include member SRVRET)

<table>
<thead>
<tr>
<th>Legal Value</th>
<th>Value Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACSTAT-AC-UP</td>
<td>The specified AP Cluster is active.</td>
</tr>
<tr>
<td>ACSTAT-AC-DOWN</td>
<td>The specified AP Cluster is not active.</td>
</tr>
</tbody>
</table>

5-2
APSTAT

Get the status of a specified AP spawned by the caller.

Calling Sequence:

CALL "APSTAT" USING AP-NAME,
     RET-CODE.

Description:

APSTAT returns to the caller a status code containing relevant information about the AP which was specified in the request. If the AP has initiated multiple instances of the specified AP, only information about the first AP found with that AP name will be returned.

Inputs:

AP-NAME

Outputs:

RET-CODE

RET-CODE Values:
(Values are defined in the include member SRVRET)

<table>
<thead>
<tr>
<th>Legal Value</th>
<th>Value Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>APSTAT-AP-EXECUTING</td>
<td>The specified AP is active</td>
</tr>
<tr>
<td>APSTAT-AP-INITIATED</td>
<td>The specified AP has been initiated but has not yet informed the NTM that it is active.</td>
</tr>
<tr>
<td>APSTAT-AP-INIT-PENDING</td>
<td>The specified AP received an Abort Command before its initiation message. When the initiation message arrives, the AP will be aborted.</td>
</tr>
</tbody>
</table>

5-3
<table>
<thead>
<tr>
<th>Legal Value</th>
<th>Value Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>APSTAT-AP-NOT-IN-AP-STATUS-TBL</td>
<td>There is no record of an active instance of the specified AP</td>
</tr>
<tr>
<td>APSTAT-AP-NOT-IN-IISS-DIR</td>
<td>There is no record of the existence (active or not) of the specified AP.</td>
</tr>
</tbody>
</table>
Check for the arrival of messages in the AP's mailbox.

Calling Sequence:

CALL "CHKMSG" USING LOGICAL-CHANNEL, SOURCE, RET-CODE.

Description:
CHKMSG can be used to determine whether any message, or a specific message, has arrived at the AP's Mailbox. The message can then be retrieved with a CALL "RCV" at a more convenient place in the program logic as CHKMSG does not deliver messages to the AP. This capability would be used, for example, in applications where a long calculation or database query is being performed, and the program must watch for Shutdown messages, but the program logic is such that a certain amount of processing is required, such as saving status information, before the current processing can be interrupted and a new message accepted and acted upon.

For Messages in the AP's cold mailbox:

- To check for any message, leave the logical channel and source arguments blank. If more than one message has arrived, the channel and source of the first message in the buffer will be returned.

- To check for a message from a specific source on any channel, specify the source argument and leave the channel argument blank. If more than one message has arrived from the specified source, the channel of the first message in the buffer from the specified source will be returned.

- To check for a message from a specific source on a specific channel, specify both of these arguments. If any messages have arrived from this source on this channel, the return CODE will indicate CHKMSG-MESSAGE-FOUND.

For Messages in the AP's hot mailbox:
To check for hot messages from the NTM (such as Shutdown Pending, Cancel Shutdown, or Shutdown), the message source must be specified as "NTMPU.....". The following blanks are required by the IISS naming conventions.
* CHKMSG (Continued)

Inputs or Outputs:

LOGICAL-CHANNEL
SOURCE

Outputs:

RET-CODE

RET-CODE Values: (Values are defined in the include member SRVRET)

<table>
<thead>
<tr>
<th>Legal Value</th>
<th>Value</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHKMSG-MESSAGE-FOUND</td>
<td>The specified (or any,</td>
<td>The specified (or any, depending on values in the call parameters) message was found.</td>
</tr>
<tr>
<td></td>
<td>depending on values in the</td>
<td>message was found.</td>
</tr>
<tr>
<td></td>
<td>call parameters) message</td>
<td></td>
</tr>
<tr>
<td></td>
<td>was found.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHKMSG-NO-MESSAGES</td>
<td>The specified message was</td>
<td>The specified message was not found - or - if any, no message was found.</td>
</tr>
<tr>
<td></td>
<td>not found - or - if any,</td>
<td></td>
</tr>
<tr>
<td></td>
<td>no message was found.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHKMSG-FATAL-ERROR</td>
<td>An error has occurred within</td>
<td>An error has occurred within the CHKMSG routine.</td>
</tr>
<tr>
<td></td>
<td>the CHKMSG routine.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHKMSG-BUFFER-FULL</td>
<td>There is no available space</td>
<td>There is no available space to hold the message.</td>
</tr>
<tr>
<td></td>
<td>to hold the message.</td>
<td></td>
</tr>
</tbody>
</table>

Examples:

1. To check for any cold message:

   MOVE SPACES TO MSG-SOURCE
   MOVE SPACES TO LOGICAL-CHANNEL
   CALL "CHKMSG" USING LOGICAL CHANNEL, MSG-SOURCE, RET-CODE
   IF CHKMSG-MESSAGE-FOUND
       PERFORM RECEIVE-MESSAGE
   ELSE
       NEXT SENTENCE

   5-7
2. To check for a "hot" message:

MOVE SPACES TO LOGICAL-CHANNEL.
MOVE "MTMPU...." to MSG-SOURCE.
CALL "CHKMSG" USING LOGICAL-CHANNEL,
      MSG-SOURCE,
      RET-CODE.

IF CHKMSG-MESSAGE-FOUND
   PERFORM SUSPEND-PROCESSING
   PERFORM RCV-UNSOL-MSG
   PERFORM SHUTDOWN-CHECK
   IF SD-PENDING
      PERFORM SD-PREP
   ELSE
      PERFORM UNSOL-MSG-ERR
   ELSE
      NEXT SENTENCE.

ENDRCY

Signal end of recovery processing to the NTM (for APs that do special processing in the IISS recovery mode).

Calling Sequence:

CALL "ENDRCY" USING ENDRCY-STATUS.

Description:

ENDRCY allows the calling AP to inform the NTM that the AP has completed its internal recovery processing. A relevant status code is passed to the NTM by the calling AP.

Inputs:

ENDRCY-STATUS

Outputs:

None

ENDRCY-STATUS Values:

<table>
<thead>
<tr>
<th>Legal Value</th>
<th>Value Representation</th>
<th>Value Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENDRCY-SUCCESSFUL</td>
<td>1</td>
<td>The AP has successfully completed its recovery procedures.</td>
</tr>
<tr>
<td>ENDRCY-NOT-SUCCESSFUL</td>
<td>0</td>
<td>The AP could not recover.</td>
</tr>
</tbody>
</table>
Signal receipt of a guaranteed delivery message.

Calling Sequence:

CALL "GDACK" USING MSG-SERIAL-NUMBER, RET-CODE.

Description:

GDACK is used by the calling AP to signal to the NTH that it has received and processed the specified guaranteed delivery message. It is a required response to guaranteed delivery messages and should be issued after the related processing is complete.

Inputs:

MSG-SERIAL-NUMBER (of received Guaranteed Delivery Message)

Outputs:

RET-CODE

RET-CODE Values:
(Values are defined in the include member SRVRET)

<table>
<thead>
<tr>
<th>Legal Value</th>
<th>Value Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>GDACK-SUCCESSFUL</td>
<td>The receiving AP has received and completed processing of the Guaranteed Delivery message.</td>
</tr>
<tr>
<td>GDACK-INVALID-SERIAL-NUMBER</td>
<td>There is no record of a guaranteed delivery message having the specified message serial number.</td>
</tr>
</tbody>
</table>
Send a guaranteed delivery message through the NTM.

Calling Sequence:

CALL "GDSEND" USING DESTINATION,
       LOGICAL-CHANNEL,
       DATA-TYPE,
       MESSAGE-TYPE,
       DATA-LENGTH,
       DATA,
       ACCEPT-STATUS,
       MSG-Serial-NUMBER.

Description:

GDSEND is used to send a guaranteed delivery message from an AP to any authorized destination via the services provided by the NTM and other subsystems of the IISS. The NTM guarantees the delivery of messages sent with this call.

Inputs:

- All except ACCEPT-STATUS and MSG-Serial-NUMBER.
- LOGICAL-CHANNEL is optional. (If blank, the AP Interface will supply a default value.)
- The Destination argument must include the directory prefix.

Outputs:

- ACCEPT-STATUS and MSG-Serial-NUMBER. (The returned MSG-Serial-NUMBER can be used in GDSTAT to find the status of the message sent with this GDSEND.)
ACCEPT-STATUS Values*:

SEND-MSG-ACCEPTED
SEND-MSG-NOT-AUTHORIZED
SEND-MSG-ILLEGAL-TYPE
SEND-INVALID-DESTINATION
SEND-INVALID-DATA-LENGTH
SEND-INVALID-BIN-NAT-FLAG
SEND-RESOURCES-NOT-AVAILABLE
SEND-INVALID-TIMEOUT-REQUEST
SEND-INVALID-SOURCE

*See NSEND for a full description of these values.
GDSTAT

Get the status of a specified guaranteed delivery message.

Calling Sequence:

CALL "GDSTAT" USING MSG-SERIAL-NUMBER,
    RET-CODE.

Description:

GDSTAT returns to the caller a status code containing relevant
information about the specified guaranteed delivery message
originated by the caller.

Inputs:

MSG-SERIAL-NUMBER

Outputs:

RET-CODE

RET-CODE Values:
(Values are defined in the include member SRVRET)

<table>
<thead>
<tr>
<th>Legal Value</th>
<th>Value Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>GDSTAT-MESSAGE-IN-SYSTEM</td>
<td>The specified guaranteed delivery message is in process.</td>
</tr>
<tr>
<td>GDSTAT-MESSAGE-NOT-FOUND</td>
<td>The specified message cannot be found.</td>
</tr>
</tbody>
</table>
Determine the AP Name and user name of the original node or source in an AP chain. The User Logon information is determined only where the original source AP is the User Interface.

Calling Sequence:

CALL "GETUSR" USING AP-NAME, LOGICAL-CHANNEL, USER-NAME, ROLE-NAME, TERMINAL-ID, RETURN-CODE.

Description:

GETUSR returns to the caller the current user name, AP-NAME, and logical channel associated with its original source. This call allows any AP in a chain to determine its originating source and chain communication channel. If the originating source is a user at a terminal, a user-name and the name of the associated AP, a UI, are returned.

If the originating source is an AP (not a terminal user), then the user Logon data will be blank on return. Only the AP-Name and Logical-Channel will have non-blank values.

Inputs:

None

Outputs:

AP-NAME
LOGICAL-CHANNEL
USER-NAME
ROLE-NAME
TERMINAL-ID
RETURN-CODE
* GETUSR (Continued)

RETURN-CODE Values:
(Values are defined in the include member SRVRET)

<table>
<thead>
<tr>
<th>Legal Value</th>
<th>Value Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>SRV-SUCCESSFUL</td>
<td>The GETUSR service has successfully obtained all of the requested data.</td>
</tr>
<tr>
<td>GETUSR-NOT-SUCCEED</td>
<td>The GETUSR service was not able to obtain the requested data.</td>
</tr>
<tr>
<td></td>
<td>This would be due to an error in accessing the LOGON Table.</td>
</tr>
</tbody>
</table>

Example:

Using Call "GETUSR" to obtain the values needed to send a message to the AP's original source.

CALL "GETUSR" USING AP-NAME,
  ORIG-CHANNEL,
  USER-NAME,
  ROLE-NAME,
  TERM-ID,
  RET-CODE.

IF SRV-SUCCESSFUL
  NEXT SENTENCE
ELSE
  MOVE "8" TO GOOF-CODE
  PERFORM GOOF-IN-PROGRAM
  MOVE "XTSAP5 HAS DONE ITS JOB." TO LAST-MSG.
  MOVE AP-NAME TO MSG-DESTINATION.
  MOVE ORIG-CHANNEL TO LOGICAL-CHANNEL.
  MOVE 25 TO DATA-LENGTH-SEND.
  MOVE "DM" TO MESSAGE-TYPE-SEND.
  MOVE ZEROS TO TIMEOUT-VALUE.
  MOVE LAST-MSG TO DATA-SEND.
  CALL "NSEND" USING MSG-DESTINATION,
    LOGICAL-CHANNEL,
    TIMEOUT-VALUE,
    BINARY-NATIVE-FLAG,
    MESSAGE-TYPE-SEND,
    DATA-LENGTH-SEND,
    DATA-SEND,
ACCEPT-STATUS.
IF SEND-MSG-ACCEPTED
PERFORM FINISH-PROGRAM
ELSE
MOVE "4" TO GOOF-CODE
PERFORM GOOF-IN-PROGRAM.
HSTATS

Get the status of a specified HOST.

Calling Sequence:

CALL "HSTATS" USING HOST-NAME, RET-CODE.

Description:

HSTATS returns to the caller a status code containing relevant information about the HOST which was specified in the request.

Inputs:

HOST-NAME

Outputs:

RET-CODE

RET-CODE Values:
(Values are defined in the include member SRVRET)

<table>
<thead>
<tr>
<th>Value</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>HSTATS-HOST-UP</td>
<td>The specified host is active.</td>
</tr>
<tr>
<td>HSTATS-HOST-DOWN</td>
<td>The specified host is not active.</td>
</tr>
<tr>
<td>HSTATS-HOST-NOT-IISS</td>
<td>The specified host is not part of the IISS configuration.</td>
</tr>
</tbody>
</table>
* INICOM

Provide initiation service for COMM AP.

Calling Sequence:

CALL "INICOM" USING COMM-RCV-EVENT-BLOCK,
   INPUT-MBX-NAME,
   APC-HOT-MBX-NAME,
   APC-COLD-MBX-NAME,
   RET-STATUS.

Description:

INICOM is a routine used by the COMM APs. It creates COMM's input mailbox, sends the COMM's "I'm Alive" message to the local MPU, and returns the mailbox names and initiation status to the COMM AP.

Inputs:

COMM-RCV-EVENT-BLOCK

Outputs:

INPUT-MBX-NAME
APC-HOT-MBX-NAME
APC-COLD-MBX-NAME
RET-STATUS
* INITIAL

Provide initiation services for an AP.

Calling Sequence:

CALL "INITIAL" USING BUFFER,
    BUFFER-SIZE,
    SYSTEM-STATE,
    RET-CODE.

Description:

INITIAL is the routine called by an AP to request that the AP interface perform the necessary initialization to allow the AP to execute and communicate with the IISS.

Inputs:

    BUFFER
    BUFFER-SIZE

Outputs:

    SYSTEM-STATE
    RET-CODE

RET-CODE Values:
(Values are defined in the include member SRVRET)

<table>
<thead>
<tr>
<th>Value</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>INITIAL-SUCCESSFUL</td>
<td>The AP has successfully connected with the NTM.</td>
</tr>
<tr>
<td>INITIAL-NOT-SUCCESSFUL</td>
<td>The AP did not connect with the NTM.</td>
</tr>
</tbody>
</table>
SYSTEM-STATE Values:
(Values are defined in the include member SRVRET)

<table>
<thead>
<tr>
<th>Legal Value</th>
<th>Value Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>INITIAL-RECOVERY</td>
<td>The IISS is currently running in recovery mode.</td>
</tr>
<tr>
<td>INITIAL-IISS-START</td>
<td>The IISS is currently running in start-up mode.</td>
</tr>
<tr>
<td>INITIAL-NORMAL</td>
<td>The IISS is operating normally.</td>
</tr>
<tr>
<td>INITIAL-FIRST-RUN</td>
<td>Identifies the first run of the AP after an event that is significant to the AP.</td>
</tr>
</tbody>
</table>
(Example: An AP that does no recovery, but does special first-run processing.)

DATA DIVISION.

WORKING-STORAGE SECTION.

COPY SRVRET OF IISSCLIB.

01 BUFFER PIC X(4096).
01 BUFFER-SIZE PIC 9(4) VALUE 4096.
01 SYSTEM-STATE PIC(X).
01 RET-CODE PIC X(3).

PROCEDURE DIVISION.

START PROGRAM.

   CALL "INITIAL" USING BUFFER,
      BUFFER-SIZE,
      SYSTEM-STATE,
      RET-CODE.

   IF INITIAL-SUCCESSFUL
      IF INITIAL-IISS-START OR
         INITIAL-FIRST-RUN
         PERFORM INITIAL-CODE
         PERFORM RUN-CODE
      ELSE
         PERFORM RUN-CODE
   ELSE
      PERFORM TERMINATION-CODE.

   INITIAL-CODE.

   RUN-CODE.

   TERMINATION-CODE.
**ISEND**

Send a message that requests the specific initiation of a new instance of an AP. Data for the new destination AP instance may be included in the message.

**Calling Sequence:**

```
CALL "ISEND" USING DESTINATION,
      LOGICAL-CHANNEL,
      TIMEOUT-VALUE,
      DATA-TYPE,
      MESSAGE-TYPE,
      DATA-LENGTH,
      DATA,
      ACCEPT-STATUS.
```

**Description:**

ISEND is used to specifically request the initiation of a new instance of a destination AP. Except for this feature, the services provided to an AP by this call are the same as NSEND. This call is intended to support complex APs that must manage communications between multiple instances of the same AP. The ISEND user must specify different channel indicators to manage the communication between multiple instances of the same AP. The ISEND call may be used with or without data.

**Inputs:**

All except ACCEPT-STATUS
LOGICAL-CHANNEL is optional. (Field should be blank if no channel is required.)
TIMEOUT-VALUE should be set to zero if pairing support is not required. A non-zero timeout value will activate the message pairing processing. The Destination argument must include the directory prefix.

**Outputs:**

ACCEPT-STATUS

5-22
* ISEND (Continued)

**ACCEPT-STATUS** Values:

SEND-MSG-ACCEPTED
SEND-MSG-NOT-AUTHORIZED
SEND-MSG-ILLEGAL-TYPE
SEND-INVALID-DESTINATION
SEND-INVALID-DATA-LENGTH
SEND-INVALID-DATA-TYPE
SEND-RESOURCES-NOT-AVAILABLE
SEND-INVALID-TIMEOUT-REQUEST
BUFFER-OVERFLOW
SEND-FATAL-ERROR
SEND-INVALID-SOURCE

*See NSEND for the description of these values.

**NOTES:**

The ISEND call must be used when starting APs having the characteristic of requiring a specific initiation. For APs having no restriction on initiation, the use of call ISEND serves to guarantee the initiation of a new instance.

**Example:**

MOVE "NTTSAP2MPU" TO MSG-DESTINATION.
MOVE "002" TO LOGICAL-CHANNEL.
MOVE "IR" TO MESSAGE-TYPE-SEND.
MOVE 25 TO DATA-LENGTH-SEND.
MOVE "SEND PART INVENTORY DATA." TO DATA-SEND.
MOVE ZEROS TO TIMEOUT-VALUE.
CALL "ISEND" USING MSG-DESTINATION,
LOGICAL-CHANNEL,
TIMEOUT-VALUE,
BINARY-NATIVE-FLAG,
MESSAGE-TYPE-SEND,
DATA-LENGTH-SEND,
DATA-SEND,
ACCEPT-STATUS.

IF SEND-MSG-ACCEPTED
DISPLAY "MESSAGE IS ON ITS WAY."
ELSE

5-23
MOVE "05" TO GOOF-CODE
PERFORM SERVICE-ERROR.
MSGACK

Acknowledge receipt of a message.

Calling Sequence:

CALL "MSGACK" USING ACCEPT-INDICATOR,
MSG-SERIAL-NUMBER,
RET-CODE.

Description:

MSGACK is used by a calling AP to notify the NTM that the AP has received a specified message. The NTM then formulates and delivers an acknowledgement message to the AP that requested the acknowledgement. It is part of the AP-AP protocol to determine when this simple acknowledgement to a message that requires a response can be used. The original message sender must send the message that requires this MSGACK as a paired message.

Inputs:

ACCEPT-INDICATOR
   Value: "0" = Msg-Not-Accepted
          "1" = Msg-Accepted

MSG-SERIAL-NUMBER

Outputs:

RET-CODE

RET-CODE Values:
(Values are defined in the include member SRVRET)

<table>
<thead>
<tr>
<th>Legal Value</th>
<th>Value</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSGACK-SUCCESSFUL</td>
<td></td>
<td>The specified message has been received.</td>
</tr>
<tr>
<td>MSGACK-INVALID- SERIAL- NUMBER</td>
<td></td>
<td>The specified message serial number is invalid.</td>
</tr>
</tbody>
</table>

5-25
Send a message through the NTM.

Calling Sequence:

CALL "NSEND" USING DESTINATION,
   LOGICAL-CHANNEL,
   TIMEOUT-VALUE,
   BINARY-NATIVE-FLAG,
   MESSAGE-TYPE,
   DATA-LENGTH,
   DATA,
   ACCEPT-STATUS.

Description:

NSEND is used to send a message that does not require special NTM handling from an AP to any authorized destination via the services provided by the NTM and other subsystems of IISS.

Inputs:

All except ACCEPT-STATUS
LOGICAL-CHANNEL can be blank if a specific channel is not required. TIMEOUT-VALUE must be zero if no pairing support is required. If a non-zero timeout value is specified, the AP must specify a non-blank LOGICAL-CHANNEL in order to pair the responses or timeout message. In Release 2.0 any non-zero timeout value will invoke the pairing support on a system timer (i.e., a system specified time is used instead of the value specified in the cell. In Release 2.0 the system timeout value was set to 30 seconds). The Destination argument must specify the directory prefix.

Outputs:

ACCEPT-STATUS
### NSEND (Continued)

**ACCEPT-STATUS Values:**
(Values are defined in the include member SRVRET)

<table>
<thead>
<tr>
<th>Legal Value</th>
<th>Value Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEND-MSG-ACCEPTED</td>
<td>The message sent under any of the SEND calls has been accepted by the NTM.</td>
</tr>
<tr>
<td>SEND-MSG-NOT-AUTHORIZED</td>
<td>The message failed the authorized check performed by the NTM. The source is not authorized to send a message of the given type to the destination.</td>
</tr>
<tr>
<td>SEND-MSG-ILLEGAL-TYPE</td>
<td>The given message type is not valid for the given destination or the Message Type argument is blank.</td>
</tr>
<tr>
<td>SEND-INVALID-DESTINATION</td>
<td>The destination AP name as in the calling argument was not found in the NTM tables.</td>
</tr>
<tr>
<td>SEND-INVALID-DATA-LENGTH</td>
<td>The data length argument is blank.</td>
</tr>
<tr>
<td>SEND-INVALID-BIN-NAT-FLAG</td>
<td>The given binary-native flag is not valid.</td>
</tr>
<tr>
<td>SEND-RESOURCES-NOT-AVAILABLE</td>
<td>The resources needed by the NTM to process the message are not available.</td>
</tr>
<tr>
<td>SEND-INVALID-TIMEOUT-REQUEST</td>
<td>The given timeout request is not valid.</td>
</tr>
<tr>
<td>BUFFER-OVERFLOW</td>
<td>The buffer used to hold messages for processing is full. There is nothing wrong with the message, it simply</td>
</tr>
</tbody>
</table>
cannot be processed at the time.

**SEND-FATAL-ERROR**

The MPU has rejected the message for reasons beyond the control of the source AP.

**SEND-INVALID-SOURCE**

The source AP name cannot be found in the NTH tables.
Examples:

1. To send a paired message. Note: The value of Binary-Native-Flag has been set in the DATA DIVISION.

   MOVE "NTTSAP3MPU" TO MSG-DESTINATION.
   MOVE "003" TO LOGICAL-CHANNEL.
   MOVE "ID" TO MESSAGE-TYPE-SEND.
   MOVE "DETERMINE THE LOCATION OF PART X AND MOVE ALL TO POINT B." TO DATA-SEND.
   MOVE 57 TO DATA-LENGTH-SEND.
   MOVE "0000000000000001" TO TIMEOUT-VALUE.
   CALL "NSEND" USING MSG-DESTINATION,
         LOGICAL-CHANNEL,
         TIMEOUT-VALUE,
         BINARY-NATIVE-FLAG,
         MESSAGE-TYPE-SEND,
         DATA-LENGTH-SEND,
         DATA-SEND,
         ACCEPT-STATUS.
   IF SEND-MSG-ACCEPTED
      DISPLAY "MESSAGE IS ON ITS WAY."
   ELSE
      MOVE "06" TO GOOF-CODE
      PERFORM SERVICE-ERROR.
   END-IF
Examples (Continued):

2. To send an unpaired message:

MOVE "NTTSAP8MPU" TO MSG-DESTINATION.
MOVE "005" TO LOGICAL-CHANNEL.
MOVE "ID" TO MESSAGE-TYPE-SEND.
MOVE "HELLO" TO DATA-SEND.
MOVE 5 TO DATA-LENGTH-SEND.
MOVE ZEROS TO TIMEOUT-VALUE.
CALL "NSEND" USING MSG-DESTINATION,
  LOGICAL-CHANNEL,
  TIMEOUT-VALUE,
  BINARY-NATIVE-FLAG,
  MESSAGE-TYPE-SEND,
  DATA-SEND,
  ACCEPT-STATUS.

IF SEND-MSG-ACCEPTED
  DISPLAY "MESSAGE IS ON ITS WAY."
ELSE
  MOVE "06" TO GOOF-CODE
  PERFORM SERVICE-ERROR.
PRSTAT

Get the status of a specified paired message.

Calling Sequence:

CALL "PRSTAT" USING DESTINATION,
    LOGICAL-CHANNEL,
    RET-CODE.

Description:

PRSTAT returns to the caller a status code containing relevant information about the specified paired message request originated by the caller.

Inputs:

DESTINATION
LOGICAL-CHANNEL

Outputs:

RET-CODE

RET-CODE Values:
(Values are defined in the include member SRVRET)

<table>
<thead>
<tr>
<th>Legal Value</th>
<th>Value Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRSTAT-MESSAGE-IN-SYSTEM</td>
<td>The specified message is being processed.</td>
</tr>
<tr>
<td>PRSTAT-MESSAGE-NOT-FOUND</td>
<td>The specified message cannot be found.</td>
</tr>
</tbody>
</table>

5-31
Send a reply message through the NTM from a queue-server AP.

Calling Sequence:

CALL "QSEND" USING DESTINATION, LOGICAL-CHANNEL, TIMEOUT-VALUE, BINARY-NATIVE-FLAG, MESSAGE-TYPE, DATA-LENGTH, DATA, ACCEPT-STATUS.

Description:

QSEND is used to send a reply message that uses the exact name and instance of the previous message received by this AP from the specified destination via the services provided by the NTM and other subsystems of IISS.

Inputs:

All except ACCEPT-STATUS
LOGICAL-CHANNEL can be blank if the specific channel is not known. The message will be sent out on the same channel upon which it was received. TIMEOUT-VALUE must be zero since no pairing support is available. The Destination argument must specify the directory prefix.

Outputs:

ACCEPT-STATUS (values same as those for NSEND)
Example:

1. To send a reply message. Note: The value of Binary-Native-Flag has been set in the DATA DIVISION. RCV-SOURCE was set upon return from the RCV call. LOGICAL-CHANNEL was set upon return from the RCV call.

MOVE RCV-SOURCE TO MSG-DESTINATION.
MOVE "ID" TO MESSAGE-TYPE-SEND.
MOVE "YOU ARE THE TENTH PROGRAM TO TALK TO ME TODAY" TO DATA-SEND.
MOVE 45 TO DATA-LENGTH-SEND.
MOVE ZERO TO TIMEOUT-VALUE.
CALL "QSEND" USING MSG-DESTINATION, LOGICAL-CHANNEL, TIMEOUT-VALUE, BINARY-NATIVE-FLAG, MESSAGE-TYPE-SEND, DATA-LENGTH-SEND, DATA-SEND, ACCEPT-STATUS.

IF SEND-MSG-ACCEPTED
    DISPLAY "REPLY MESSAGE IS ON ITS WAY."
ELSE
    MOVE "06" TO GOOF-CODE
    PERFORM SERVICE-ERROR.
Receive a message through the NTM.

Calling Sequence:

CALL "RCV" USING LOGICAL-CHANNEL, WAIT-FLAG, SOURCE, MESSAGE-TYPE, DATA-LENGTH, DATA, ACCEPT-STATUS, MSG-SERIAL-NUMBER.

Description:

RCV is used to receive a message, of any type, from any authorized source, including the NTM itself, via the services provided by the NTM and other subsystems of IISS.

Inputs:

WAIT-FLAG

Values:  "0" = No-Wait
         "1" = Wait

Outputs:

MSG-SERIAL-NUMBER, MESSAGE-TYPE, DATA-LENGTH, DATA, ACCEPT-STATUS

ACCEPT-STATUS Values:  
(Values are defined in the include member SRVRET)

Inputs/Outputs:

LOGICAL-CHANNEL,
SOURCE.
NOTES:

1. SOURCE and/or LOGICAL-CHANNEL can be specified if messages from a specific source, or specific source and logical-channel are required. If source and logical-channel are blank, the first message in the buffer will be returned with its source and logical-channel. On a source match, the directory prefix must be specified.

2. If the AP wishes to specify the NTM for a source match, the source parameter must be entered as "NTMPU......" on the call. This will retrieve any message that may be in the AP's hot mailbox. If the AP supports a "hot" mailbox, it will be routinely checked on a RCV where source and Logical channel are not specified.

3. If the RCV service is called after a call CHKMSG, the source parameter should be the message source returned on the CHKMSG call.

4. If the AP expects to receive paired messages it must use the condition "IF RCV-REPLY-REQUIRED-MESSAGE" to test the RCV call return.

5. If "RCV" is called with no wait set, the AP must be able to deal with the possibility of a "RCV-NO-MESSAGE" return.

<table>
<thead>
<tr>
<th>Legal Value</th>
<th>Value Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>RCV-NORMAL-MESSAGE</td>
<td>A message having a category of E, F, G, or H has been retrieved from the AP's mailbox.</td>
</tr>
<tr>
<td>RCV-REPLY-REQUIRED-MESSAGE</td>
<td>A message having a category of &quot;B&quot; or &quot;J&quot; has been retrieved from the AP's mailbox.</td>
</tr>
<tr>
<td>RCV-ACK-MESSAGE</td>
<td>Message Received is an ACK from a destination AP using the &quot;MSGACK&quot; service.</td>
</tr>
<tr>
<td>Legal Value</td>
<td>Value Definition</td>
</tr>
<tr>
<td>---------------------</td>
<td>---------------------------------------------------------------------</td>
</tr>
<tr>
<td>RCV-MACK-MESSAGE</td>
<td>Message Received is a NACK from a destination AP using the &quot;MSGACK&quot; service.</td>
</tr>
<tr>
<td>RCV-GD-MESSAGE</td>
<td>A message having a category of &quot;A&quot; has been retrieved from the AP's mailbox.</td>
</tr>
<tr>
<td>RCV-NO-MESSAGE</td>
<td>No message was found in the AP's mailbox.</td>
</tr>
<tr>
<td>RCV-TIME-OUT</td>
<td>A message time-out notification has arrived from the local MPU.</td>
</tr>
<tr>
<td>RCV-MSG-TEST-MODE</td>
<td>The message retrieved from the AP's mailbox is from an AP currently operating in test mode.</td>
</tr>
<tr>
<td>RCV-FATAL-ERROR</td>
<td>The NTM cannot access the Inter-Process Communication facilities.</td>
</tr>
<tr>
<td>RCV BUFFER-FULL</td>
<td>The resources needed to process incoming messages are not available.</td>
</tr>
</tbody>
</table>
Examples:

1. RCV with no wait – to retrieve any message.

   MOVE SPACES TO MSG-SOURCE.
   MOVE SPACES TO WAIT-FLAG.
   MOVE SPACES TO LOGICAL-CHANNEL.
   MOVE SPACES TO DATA-RCV.
   MOVE "0" TO WAIT-FLAG.
   CALL "RCV" USING LOGICAL-CHANNEL,
       WAIT-FLAG,
       MSG-SOURCE,
       MESSAGE-TYPE-RCV,
       DATA-LENGTH-RCV,
       DATA-RCV,
       ACCEPT-STATUS,
       MESSAGE-SERIAL-NUMBER.
   IF RCV-NORMAL-MESSAGE
       ADD 1 TO RCV-COUNT
   ELSE IF RCV-NO-MESSAGE
       DISPLAY "NO MESSAGE YET - HANG IN THERE!"
   ELSE
       MOVE "07" TO GOOF-CODE
       PERFORM SERVICE-ERROR.

2. RCV with a wait – The AP expects to receive paired messages.

   MOVE "1" TO WAIT-FLAG.
   MOVE SPACES TO LOGICAL-CHANNEL.
   CALL "RCV" USING LOGICAL-CHANNEL.
   CALL "RCV" USING LOGICAL-CHANNEL,
       WAIT-FLAG,
       MSG-SOURCE,
       MESSAGE-TYPE-RCV,
       DATA-LENGTH-RCV,
       DATA-RCV,
       ACCEPT-STATUS,
       MESSAGE-SERIAL-NUMBER.
   IF RCV-NORMAL-MESSAGE
       OR RCV-REPLY-REQUIRED MESSAGE
       NEXT SENTENCE
   ELSE
       MOVE "1" TO GOOF-CODE
       PERFORM GOOF-IN-PROGRAM.
3. RCV with a wait - The AP is looking for a specific message.

MOVE "1" TO WAIT-FLAG.
MOVE "MTPSAP1MPU" TO MSG-SOURCE.
MOVE "001" TO LOGICAL-CHANNEL.
CALL "RCV" USING LOGICAL-CHANNEL,
    WAIT-FLAG,
    MSG-SOURCE,
    MESSAGE-TYPE-RCV,
    DATA-LENGTH-RCV,
    ACCEPT-STATUS,
    MESSAGE-SERIAL-NUMBER.
IF RCV-NORMAL-MESSAGE OR RCV-REPLY-REQUIRED-MESSAGE
NEXT SENTENCE
ELSE
    MOVE "1" TO GOOF-CODE
    PERFORM GOOF-IN-PROGRAM.
SETDLY

Set the delay parameters for the next "NSEND," "ISEND," or "GDSEND" call.

Calling Sequence:

CALL "SETDLY" USING DELAY-TRIG-INDICATOR,
DELAY-TRIG-TIME-VALUE,
DELAY-TRIG-CONDITION-SPECIFIER,
RET-CODE.

Description:

SETDLY is used to specify the delay trigger parameters which are to be used in the next "NSEND," "ISEND," or "GDSEND" call. These are parameters that are put in the header of messages, but due to their infrequent use, this special call is provided instead of requiring that they be specified in the calling sequence for all "send" services. This logic has not been designed in Release 2.0, but the intent is to provide a mechanism to allow messages to be delivered, and hence programs started, after specified time delays, at a specified time of day, or following certain conditions, such as when the program next runs.

Inputs:

DELAY-TRIG-INDICATOR
DELAY-TRIG-TIME-VALUE
DELAY-TRIG-CONDITION-SPECIFIER

Outputs:

RET-CODE

RET-CODE Values:
(Values are defined in the include member SRVRET)

<table>
<thead>
<tr>
<th>Value</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>SETDLY-SUCCESSFUL</td>
<td>The delay condition specified</td>
</tr>
</tbody>
</table>

5-39
in the call has been set successfully.

**SETDLY-INVALID-TIME**

The time value given in the call is not valid -- the condition has not been set.

**SETDLY-INVALID-TRIG-COND**

The trigger condition given in the call is not valid -- the condition has not been set.
SIGABT

Signal to the NTM that an AP is to be aborted.

Calling Sequence:

CALL "SIGABT" USING AP-NAME,
  LOGICAL-CHANNEL,
  RET-CODE.

Description:

SIGABT allows the calling AP to indicate to the NTM that the AP wishes to abort another AP which it (the caller) has spawned, directly or indirectly.

Inputs:

- AP-NAME
- LOGICAL-CHANNEL

Outputs:

- RET-CODE

RET-CODE Values:
(Values are defined in the include member SRVRET)

<table>
<thead>
<tr>
<th>Value</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>SIGABT-SUCCESSFUL</td>
<td>The message to abort a child AP has been accepted for processing by the NTM.</td>
</tr>
<tr>
<td>SIGABT-NOT-SUCCESSFUL</td>
<td>The message to abort an AP has not been accepted by the NTM.</td>
</tr>
</tbody>
</table>

Note: These code values only signify the acceptance of the abort message by the NTM. If the AP requires a specific acknowledgement of the actual abort, it must indicate this...
requirement when it defines its characteristics to the CDM Administrator (see Appendix B).
* SIGERR

Allows an AP to signal the NTM and it's original source when a non-fatal error occurs.

Calling Sequence:

CALL "SIGERR" USING ERROR-CODE,
SEVERITY-LEVEL,
ERROR-DESCRIPTION,
RET-CODE.

Description:

SIGERR accepts error data from the calling AP and forwards that data to both the calling AP's original source and the Monitor AP.

Inputs:

ERROR-CODE
SEVERITY-LEVEL
ERROR-DESCRIPTION

Outputs:

RET-CODE

RET-CODE Values:
(Values are defined in the include member SRVRET)

<table>
<thead>
<tr>
<th>Legal Value</th>
<th>Value Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUCCESSFUL</td>
<td>The SIGERR message has been sent successfully.</td>
</tr>
<tr>
<td>SIGERR-UNSUCCESSFUL</td>
<td>The SIGERR message could not be sent.</td>
</tr>
</tbody>
</table>
* TRMNAT

Signal AP termination status to the NTM.

Calling Sequence:

CALL "TRMNAT" USING TERMINATION-STATUS.

Description:

TRMNAT allows an executing AP, which is terminating, to signal the NTM that it is terminating and pass to the NTM a status code specifying the termination conditions.

Inputs:

TERMINATION-STATUS

Outputs:

None

TERMINATION-STATUS Values:

<table>
<thead>
<tr>
<th>Legal Value</th>
<th>Value Representation</th>
<th>Value Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRMNAT-NORMAL-TERMINATION</td>
<td>&quot;1&quot;</td>
<td>The AP has terminated normally.</td>
</tr>
<tr>
<td>TRMNAT-SHUTDOWN-COMPLETE</td>
<td>&quot;2&quot;</td>
<td>The AP has completed its shutdown processing.</td>
</tr>
<tr>
<td>TRMNAT-ABORTED</td>
<td>&quot;3&quot;</td>
<td>The AP has terminated as the result of a soft abort command.</td>
</tr>
</tbody>
</table>
**TRMNAT**

**TERMINATION-STATUS Values (Continued):**

<table>
<thead>
<tr>
<th>Legal Value</th>
<th>Value Representation</th>
<th>Value Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRMNAT-EXCEPTION</td>
<td>&quot;4&quot;</td>
<td>The AP has terminated as the result of an internal (to the AP) exception condition.</td>
</tr>
<tr>
<td>TRMNAT-ON-BAD-INIT</td>
<td>&quot;5&quot;</td>
<td>The AP has terminated due to an error incurred when trying to connect to the NTM.</td>
</tr>
</tbody>
</table>

**NOTES:**

1. The Termination Status values defined above must be used in this call. These values are used by the NTM to direct the clean-up procedures executed upon the AP's termination.

2. The call "TRMNAT" with TERMINATION-STATUS set to TRMNAT-ON-BAD-INIT must be used if the AP did not successfully connect to the NTM in call "INITAL."

**Example:**

```plaintext
IF INITAL-NOT-SUCCESSFUL
   MOVE TRMNAT-ON-BAD-INIT TO TERMINATION-STATUS
ELSE
   MOVE TRMNAT-NORMAL-TERMINATION TO TERMINATION-STATUS.
   CALL "TRMNAT" USING TERMINATION-STATUS.
```

5-45
TSTMOD (Partially implemented)

Switch IISS message test mode on or off. When test mode is on, the calling AP will be able to receive error messages. When test mode is off, these error messages will be discarded.

Calling Sequence:

CALL "TSTMOD" USING TEST-STATUS,
    RET-CODE.

Description:

TSTMOD is used by a calling AP to indicate to the IISS system whether or not it wishes to receive error messages which are generated by its children (i.e., programs called by it). This requires special consideration in the program because these error messages can arrive at any time.

Inputs:

    TEST-STATUS
    Values:  "0" = Test-Mode off
             "1" = Test-Mode on
             "2" = Fatal Error Messages only

Outputs:

    RET-CODE

RET-CODE Values:
(Values are defined in the include member SRVRET)

<table>
<thead>
<tr>
<th>Legal Value</th>
<th>Value Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>TSTMOD-TEST-MODE-ON</td>
<td>The test mode value for subsequent messages has been set to &quot;on.&quot;</td>
</tr>
<tr>
<td>TSTMOD-TEST-MODE-OFF</td>
<td>The test mode value for subsequent messages has been set to &quot;off.&quot;</td>
</tr>
</tbody>
</table>
TSTMOD-FATAL-ONLY

The test mode value for subsequent messages has been set for fatal messages only.

TSTMOD-INVALID-REQUEST

The test status value given in the calling argument is not recognized by the service.

(Note: This description does not fully represent the intended test mode capability, nor the logic that was only partially designed and implemented in Release 2.0.)
WHATAC

Request the name of the AP cluster on which a given AP resides.

Calling Sequence:

CALL "WHATAC" USING AP-NAME,
       AP-CLUSTER-NAME,
       RET-CODE.

Description:

WHATAC will provide the caller with the name of the workstation that the given AP currently resides on. The user may specify a particular AP-NAME as input, or leave the AP-NAME field blank to obtain the name of the current AP cluster.

Inputs:

AP-NAME - (If blank, the NTM will return the name of the cluster on which the calling AP resides)

Outputs:

AP-CLUSTER-NAME
RET-CODE

RET-CODE Values:
(Values are defined in the include member SRVRET)

<table>
<thead>
<tr>
<th>Value</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>WHATAC-SUCCESSFUL</td>
<td>The APC name for the given AP has been found.</td>
</tr>
<tr>
<td>WHATAC-AP-NOT-FOUND</td>
<td>The given AP name was not found in the tables.</td>
</tr>
</tbody>
</table>
Request the name of the Host on which a given AP resides.

Calling Sequence:

CALL "WHTHST" USING AP-NAME,
       HOST-NAME,
       RET-CODE.

Description:

WHTHST will provide the caller with the name of the Host machine
that the specified AP currently resides on.

Inputs:

   AP-NAME - (If blank, the NTM will return the name of the
             host on which the calling AP resides)

Outputs:

   HOST-NAME
   RET-CODE

RET-CODE Values:
(Values are defined in the include member SRVRET)

<table>
<thead>
<tr>
<th>Legal Value</th>
<th>Value Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>WHTHST-SUCCESSFUL</td>
<td>The host name for the given AP has been found.</td>
</tr>
<tr>
<td>WHTHST-API-NOT-FOUND</td>
<td>The given AP name was not found in the NTM's AP Information table.</td>
</tr>
<tr>
<td>WHTHST-APC-NOT-FOUND</td>
<td>The APC name given in the AP Information table was not found in the AP Cluster Status Table.</td>
</tr>
</tbody>
</table>
WHST-NOT-SUCCESSFUL

The message requesting the AP's Host Name could not be sent.
WKONCA

Request "wake-up" on specified AP cluster availability.

Calling Sequence:
CALL "WKONCA" USING AP-CLUSTER-NAME,
        RET-CODE.

Description:
WKONCA will provide an AP with the ability to "hibernate" itself
until another specified AP cluster is available.

Inputs:

AP-CLUSTER-NAME

Outputs:

RET-CODE

RET-CODE Values:
(Values are defined in the include member SRVRET)

<table>
<thead>
<tr>
<th>Legal Value</th>
<th>Value Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>WKONCA-SUCCESSFUL</td>
<td>The &quot;wake-up call&quot; has been accepted.</td>
</tr>
<tr>
<td>WKONCA-AP-CLUSTER-NOT-FOUND</td>
<td>The specified AP Cluster was not found in the tables.</td>
</tr>
</tbody>
</table>
5.2 Services Available Only to IISS Components and System Operator

*CHGROL

Changes the user’s role name during a logon session. This service assumes that the UI has performed an authorization check on the new role prior to making the change.

Calling Sequence:

Call "CHGROL" USING TERMINAL-ID,
USER-NAME,
NEW-ROLE-NAME,
RET-CODE.

Description:

The new role name is given to the NTM where it replaces the previous role name in the logon table. NOTE: The UI must use the new role name in its call "LOGOFF" as this service does not maintain records of this change.

Inputs:

TERMINAL-ID
USER-NAME
NEW-ROLE-NAME

Outputs:

RET-CODE

RET-CODE Values:
(Values are defined in the include member SRVRET)

<table>
<thead>
<tr>
<th>Legal Value</th>
<th>Value Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHGROL-SUCCESSFUL</td>
<td>The message informing the Monitor AP of the new role name has been sent successfully.</td>
</tr>
<tr>
<td>CHGROL-NOT-SUCCESSFUL</td>
<td>The message to the Monitor AP</td>
</tr>
</tbody>
</table>
was not sent. The new role has not been entered in the Logon Table.
* INICOM

Provides initiation service for COMM AP.

Calling sequence:

Call "INICOM" USING COMM-RCV-EVENT-BLOCK,
    INPUT-MBX-NAME,
    APC-HOT-MBX-NAME,
    APC-COLD-MBX-NAME,
    RET-STATUS.

Description:

INICOM is a routine used by the COMM APs. It creates COMM's input mailbox, sends the COMM's "I'm Alive" message to the local MPU, and returns the mailbox names and initiation status to the COMM AP.

Inputs:

COMM-RCV-EVENT-BLOCK

Outputs:

INPUT-MBX-NAME
APC-HOT-MBX-NAME
APC-COLD-MBX-NAME
RET-STATUS
* INITEX

Provide initiation services for UI AP.

Calling Sequence:

CALL "INITEX" USING BUFFER,
      BUFFER-SIZE,
      RET-CODE.

Description:

INITEX is a routine called by APs that requires special NTM connection service (i.e., UI). It requests that the AP interface perform the necessary initialization to allow the special AP to connect to and communicate with the IISS.

Inputs:

BUFFER
BUFFER-SIZE

Outputs:

RET-CODE

RET-CODE Values:
(Values are defined in the include member SRVRET)

<table>
<thead>
<tr>
<th>Legal Value</th>
<th>Value Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>INITEX-SUCCESSFUL</td>
<td>The UI has successfully connected to the NTM.</td>
</tr>
<tr>
<td>INITEX-NOT-SUCCESSFUL</td>
<td>The UI could not connect to the NTM.</td>
</tr>
<tr>
<td>INITEX-RES-NOT-AVAIL</td>
<td>The UI's AP status table entry could not be made due to a table full condition. The connection was not made to the NTM. As this is a temporary condition, INITEX may be called again.</td>
</tr>
</tbody>
</table>
Example:

CALL "INITEX" USING BUFFER,
       BUFFER-SIZE,
       RET-CODE.

IF INITEX-NOT-SUCCESSFUL
   MOVE "01" TO GOOF-CODE
   PERFORM SERVICE-ERROR
ELSE NEXT SENTENCE.
"LOGOFF"

Allows a UI to inform the NTM of a user logoff.

Calling Sequence:

CALL "LOGOFF" USING TERMINAL-ID,
    USER-NAME,
    ROLE-NAME,
    RET-CODE.

Description:

LOGOFF is used by the UI to inform the NTM of a user logoff.

Inputs:

    TERMINAL-ID
    USER-NAME
    ROLE-NAME

Outputs:

    RET-CODE

RET-CODE Values:
(Values are defined in the include member SRVRET)

<table>
<thead>
<tr>
<th>Legal Value</th>
<th>Value Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOGOFF-SUCCESSFUL</td>
<td>The Monitor AP has been sent a message informing it that the User has logged off.</td>
</tr>
<tr>
<td>LOGOFF-NOT-SUCCESSFUL</td>
<td>The message to the Monitor AP was not sent.</td>
</tr>
</tbody>
</table>
NOTE:

Where the role-name is changed during a given session (via Call "CHGROL"), the value of the role name parameter must be the current role the user is operating under.

Example:

CALL "LOGOFF" USING TERMINAL-ID, USER-NAME, ROLE-NAME, RET-CODE.

IF LOGOFF-SUCCESSFUL
   NEXT SENTENCE
ELSE
   DISPLAY "WE'RE DONE BUT WE HAVE A BUG IN LOGOFF".
* LOGON

Allows a UI to pass logon information to the NTM.

Calling Sequence:

CALL "LOGON" USING TERMINAL-ID, USER-NAME, ROLE-NAME, SESSION-START-TIME, CHANNEL-RANGE-START, CHANNEL-RANGE-END, RET-CODE.

Description:

LOGON is used by a UI to provide the NTM with the logon "user" information which the NTM needs to build its logon table. The NTM assumes that the UI has already performed the required authority checks on the user.

Inputs:

TERMINAL-ID
USER-NAME
ROLE-NAME
SESSION-START-TIME
CHANNEL-RANGE-START
CHANNEL-RANGE-END

Outputs:

RET-CODE

RET-CODE Values: (Values are defined in the include member SRVRET)

<table>
<thead>
<tr>
<th>Legal Value</th>
<th>Value Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOGON-SUCCESSFUL</td>
<td>The Monitor AP has successfully entered the Logon data in the Logon Table.</td>
</tr>
<tr>
<td>LOGON-NOT-SUCCESSFUL</td>
<td>The Monitor AP could not write the new entry to the Logon table.</td>
</tr>
</tbody>
</table>
Example:

*OBTAIN THE USER NAME AND ROLE NAME
*GET THE SESSION START TIME

CALL "GET-TIME" USING STR-TIME
GIVING SS-STATUS.
IF SS-STATUS NOT = SS-NORMAL
   DISPLAY "BAD CALL"
ELSE
   CALL "ASCII-TIME" USING
   BY REFERENCE TIME-LENGTH
   BY DESCRIPTOR SESSION-START-TIME
   BY REFERENCE STR-TIME
   BY VALUE 0,
   GIVING SS-STATUS.
   IF SS-STATUS = SS-NORMAL
       DISPLAY SESSION-START-TIME
   ELSE
       DISPLAY "BAD CALL".

*INFORM THE NTM OF A SUCCESSFUL LOGON

CALL "LOGON" USING TERMINAL-ID,
USER-NAME,
ROLE-NAME,
SESSION-START-TIME,
CHANNEL-RANGE-START,
CHANNEL-RANGE-END,
RET-CODE.
IF LOGON-NOT-SUCCESSFUL
   MOVE "02" TO GOOF-CODE
   PERFORM SERVICE-ERROR
ELSE NEXT SENTENCE.

NOTES:

1. CHANNEL-RANGE-START and CHANNEL-RANGE-END specify a contiguous set of logical channel specifiers assigned to a user by the UI. They provide the flexibility to support a multiple terminal UI.

2. The SESSION-START-TIME is currently configured in VAX ASCII. It may be obtained by the UI on the VAX by the call "SYS$ASCTIM".
3. On the VAX the GET-TIME and ASCII-TIME functions can be implemented using the VMS system services SYS$GETTIM and SYS$ASCTIM directly.
5.3 Services Which Will Be Available as Future Enhancements

**CKRSRC**

Check the availability of all resources needed for a specified task.

**Calling Sequence:**

CALL "CKRSRC" USING TASK-CODE,
RET-CODE.

**Description:**

CKRSRC returns to the caller a status flag which indicates whether or not all resources needed to perform a specified task exist.

**Inputs:**

TASK-CODE

**Outputs:**

RET-CODE

**RET-CODE Values:**

<table>
<thead>
<tr>
<th>Legal Value</th>
<th>Value Representation</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>TBD</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
5.4 Description of Parameters Used in NTM Service Calls

ACCEPT-INDICATOR

ACCEPT-STATUS

The parameter ACCEPT-STATUS contains a code which indicates whether or not the current message was successfully processed by the NTM. Its description in COBOL is

```
01 ACCEPT-STATUS PIC X(5).
```

AP-CLUSTER-NAME

The parameter AP-CLUSTER-NAME contains a 3-character alphanumeric that is used to identify a specific work station. Its description in COBOL is

```
01 AP-CLUSTER-NAME PIC X(3).
```

AP-NAME

The parameter AP-NAME contains a 10-character alphanumeric that is used to specify the AP for whom the caller wishes to obtain status information. The general format of AP names is

```
DPSYAPNAME
```

where "DP" is the directory ID for the directory where the AP's executable module resides, "SY" is the subsystem identifier, "APNAME" is the unique AP name on the specified subsystem. Its description in COBOL is

```
01 AP-NAME PIC X(10).
```

BINARY-NATIVE-FLAG

The parameter BINARY-NATIVE-FLAG contains a code which specifies the generic type of data contained in the data portion of a message. Binary indicates that the data is in the host
machine's internal representation form whereas native indicates that the data is character data represented by the host machine's character code (ASCII, EBCDIC, etc.). Its description in COBOL is

01 BINARY-NATIVE-FLAG PIC X.

BUFFER

The parameter BUFFER is the name of the AP area that is passed to the AP Interface on the INITIAL and INITEX calls. The AP Interface uses this area to buffer incoming AP messages. It allows the AP to size this space according to its communication requirements.

01 BUFFER PIC X (BUFFER-SIZE)

BUFFER-SIZE

The parameter BUFFER-SIZE is the size (in bytes) of the buffer area, BUFFER, that is used by the AP Interface to hold incoming AP messages. Guidelines for determining this value are presented in the description of BUFFER. Its description in COBOL is

01 BUFFER-SIZE PIC 9(4).

Any value 0001-9999 is supported.

CHANNEL-RANGE-START

The lowest Logical Channel ID in the range of ID's allocated to a user at Logon. This item is represented as a Logical Channel ID. Its value is dynamic. The item is used as a field in the Logon Table.

01 CHANNEL-RANGE-START PIC X(3).

CHANNEL-RANGE-END
The highest Logical Channel ID in the range of ID's allocated to a user at Logon. This item is represented as a Logical Channel ID. Its value is dynamic. It is used as a field in the Logon Table.

01 CHANNEL-RANGE-END PIC X(3).

DATA

The parameter DATA contains an alphanumeric which represents the data portion of the message currently being processed. Its description in COBOL is

01 DATA PIC X(DATA-LENGTH).

While the maximum length of one data package is 1908 bytes, the continuation logic of the "SEND" calls allows this value to be set at any number up to 9999.

DATA-LENGTH

The parameter DATA-LENGTH contains a 4 digit numeric value which specifies the length of the data portion of the message currently being processed. Its description in COBOL is

01 DATA-LENGTH PIC 9(5)COMP.

DELAY-TRIG-CONDITION-SPECIFIER

The parameter DELAY-TRIG-CONDITION-SPECIFIER contains a code which specifies the condition which is used in conjunction with a conditional delay trigger request. Its description in COBOL is

01 DELAY-TRIG-CONDITION-SPECIFIER PIC X.

Values are TBD.

DELAY-TRIG-INDICATOR

The parameter DELAY-TRIG-INDICATOR contains a code which
specifies the type of delay trigger, if any, to apply to the message currently being processed. Its description in COBOL is

```cobol
01 DELAY-TRIG-INDICATOR PIC X.
```

Values are TBD.

### DELAY-TRIG-TIME-VALUE

The parameter DELAY-TRIG-TIME-VALUE contains a 15-character numeric value which represents a time value in 100 ns increments which is to be used in conjunction with a specified delay trigger option. Its description in COBOL is

```cobol
01 DELAY-TRIG-TIME-VALUE PIC X(15).
```

### DESTINATION

The parameter DESTINATION contains a 10-character alphanumeric that is used to specify the destination AP for the message currently being processed. See the description of AP-NAME for more detail. Its description in COBOL is

```cobol
01 DESTINATION PIC X(10).
```

### ENDCRY-STATUS

The parameter ENDCRY-STATUS contains a code which indicates whether or not the AP which is called "ENDCRY" successfully completed recovery processing. Its description in COBOL is

```cobol
01 ENDCRY-STATUS PIC X.
```

### ERROR-CODE

The parameter ERROR-CODE contains the value associated with the error that triggers a call to "SIGERR". Its description in COBOL is

```cobol
01 ERROR-CODE PIC X(5).
```
The parameter **ERROR-DESCRIPTION** contains information about the error triggering a call to "SIGERR". The content of this parameter is defined by the calling AP. The parameter's description in COBOL is:

```
01 ERROR-DESCRIPTION           PIC X(72).
```

The parameter **HOST-NAME** contains a 3-character alphanumeric that is used to specify the "physical" name of the host computer system that the caller is on. Its description in COBOL is:

```
01 HOST-NAME                   PIC X(3).
```

The parameter **LOGICAL-CHANNEL** contains a value which is used by the APs and the NTM to manage communication paths between APs (see Section 3.1.6). A value must be supplied on a SEND call when the AP wants to pair the send with a response. Its description in COBOL is:

```
01 LOGICAL-CHANNEL             PIC X(3).
```

The parameter **MESSAGE-TYPE** contains a code which specifies the message type of the message currently being processed. Its description in COBOL is:

```
01 MESSAGE-TYPE                PIC X(2).
```

The parameter **MSG-SERIAL-NUMBER** contains a 7-character
alphanumeric that is used to identify a message. Its description in COBOL is

```
01 MSG-SERIAL-NUMBER PIC X(7).
```

**RET-CODE**

The parameter RET-CODE contains a value which indicates the return status of a specific request. Its description in COBOL is

```
01 RET-CODE PIC X(5).
```

**ROLE-NAME**

The parameter ROLE-NAME contains a 10-character alphanumeric which identifies the role under which a user is logged on. Its description in COBOL is

```
01 ROLE-NAME PIC X(10).
```

**SESSION-START-TIME**

The parameter SESSION-START-TIME contains a 23-character alphanumeric which specifies the system clock time when a specified user logged on. Its description in COBOL is

```
01 SESSION-START-TIME PIC X(23).
```

**SEVERITY-LEVEL**

The parameter contains a code specifying the level of the error that triggered a call to "SIGERR". Its description in COBOL is

```
01 SEVERITY-LEVEL PIC X.
```

**SOURCE**

5-68
The parameter SOURCE contains a 10-alphanumeric that is used to specify the source AP for a call to RCV with a source match. See the description of AP-NAME for more detail. Its description in COBOL is

```
01 SOURCE PIC X(10).
```

SYSTEM-STATE

The parameter SYSTEM-STATE contains a code which indicates the system state of the IISS at the time "INITIAL" is called. Its description in COBOL is

```
01 SYSTEM-STATE PIC X.
```

TASK-CODE

The parameter TASK-CODE contains a 4-character alphanumeric which represents the identifier for a specific generic task which can be performed on the IISS Test Bed. It is used on conjunction with a "CKRSRC" request. Its description in COBOL is

```
01 TASK-CODE PIC X(4).
```

TERMINAL-ID

The parameter TERMINAL-ID identifies the terminal where a given user is logged on. Its description in COBOL is

```
01 TERMINAL-ID PIC X(2).
```

TERMINATION-STATUS

The parameter TERMINATION-STATUS contains a code which indicates the specific condition under which an AP is terminating. Its description in COBOL is

```
01 TERMINATION-STATUS PIC X.
```
TEST-STATUS

The parameter TEST-STATUS contains a code which indicates whether the AP wishes to receive any asynchronous error message, fatal errors only, or not at all. Its description in COBOL is

01 TEST-STATUS PIC X.

TIMEOUT-VALUE

The parameter TIMEOUT-VALUE contains a 15-character numeric value which represents a time value in 100 ns increments which is used in conjunction with a paired message request. Its description in COBOL is

01 TIMEOUT-VALUE PIC X(15).

USER-NAME

The parameter USER-NAME contains an 8-character alphanumeric value which identifies a specific user. Its description in COBOL is

01 USER-NAME PIC X(8).

WAIT-FLAG

The parameter WAIT-FLAG contains a code which indicates whether or not the message currently being processed should have a wait associated with it. Its description in COBOL is

01 WAIT-FLAG PIC X.
APPENDIX A

MESSAGE FORMATS

An AP may receive messages from the NTM if it has designated by its AP Characteristics (see Appendix B) that it wants to receive NTM messages, and which ones it wants to receive. These messages will provide either status information or child APs or system commands.

Each message is described below as to purpose, priority, source, type, data, and action to be taken by the receiving AP.

Messages exchanged between APs may use the message type for internal protocols. The only restrictions on the use of message types are:

1. There must always be a message type.
2. The type "XI" is reserved for the message associated with the "MSGACK" service only.
3. The type "SE" is reserved for messages associated with the "SIGERR" service.
Message Type: AP Ending

Type ID: AE

Message Priority: Low (retrieved from Cold Mailbox)*

Message Purpose:

This message is sent to APs requiring messages on child events. The message informs the parent AP that a child AP has terminated processing.

Message Source: Local MPU

Data Carried in Data Portion:

Child-AP-Process-Name
AP-Name PIC X(10)
Instance PIC X(2)
Child-AP-Channel-ID PIC X(3)
Child-AP-Status PIC X
(at Termination)

Action by Receiving AP:

The AP may continue processing. If a new instance of the child AP is needed, it may be started using call "ISEND".

*On the CALL "RCV" for this message the AP may leave the source parameter blank or specify "NTMPU....." as the source.
Message Type: Cancel Shutdown

Type ID: CS

Message Priority: High (retrieved from Hot Mailbox)

Message Purpose:

Overrides a shutdown pending message.

Message Source: Local MPU

Data Carried in Data Portion: None.

Action by Receiving AP: (UI only)

Display a message to the user that shutdown has been cancelled. The session can then continue.
Message Type: Shutdown AP

Type ID: DA

Message Priority: High (retrieved from Hot Mailbox)

Message Purpose:

Command to the AP to commence its internal shutdown procedures.

Message Source: Local MPU

Data Carried in Data Portion: None.

Action by Receiving AP:

Begin shutdown procedures. Inform the NTM when shutdown is complete via call "TRMNAT" using the TRMNAT-SHUTDOWN-COMPLETE status value.
Message Type: Shutdown Pending

Message Priority: High (retrieved from Hot Mailbox)

Message Purpose:

Notification that the IISS will be shutting down in X minutes. This message is sent once a minute until shutdown processing begins.

Message Source: Local MPU

Data Carried in Data Portion:

- Shutdown-Data PIC X(2) Value "SP"
- Time-Until-Shutdown PIC X(2) (in Minutes)

Action by Receiving AP: (UI only)

Inform the user that shutdown will take place in X (Time-Until-Shutdown) minutes. Continue to check the hot mailbox for either the next shutdown pending message or a cancel shutdown message.
Message Type: Signal Error

Type ID: SE

Message Priority: Low

Message Purpose:
Asynchronous message informing an AP of an existing error condition.

Message Source: Child AP

Data Carried in Data Portion:

- ERROR-CODE PIC X(5)
- SEVERITY-LEVEL PIC X
- ERROR-SOURCE-AP PIC X(10)
- ERROR-DESCRIPTION PIC X(72)

Action by Receiving AP:
Defined by internal AP protocol.
Message Type: Unsuccessful Initiation  

Type ID: NI

Message Priority: Low (retrieved from Cold Mailbox)

Message Purpose:

Sent when a child AP fails initiation.

Message Source: Child MPU

Data Carried in Data Portion:

<table>
<thead>
<tr>
<th>Child-AP-Name</th>
<th>PIC X(10)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parent-AP-Process-Name</td>
<td>PIC X(12)</td>
</tr>
</tbody>
</table>

Action by Receiving AP:

Terminate any processing requiring the child AP.
APPENDIX B

AP CHARACTERISTICS

In order to integrate new APs into the IISS, the AP Programmer will have to define the AP's message processing features to the CDM administrator. The NTM will use this information to build an AP characteristic record that the NTM will use to determine the correct message handling procedures for the AP. The AP characteristics that have been defined for this AP record for the NTM are the following.

a. AP Participates in IISS Shutdown (Yes/No). The AP programmer must indicate whether the AP performs special cleanup or processing on IISS shutdown. APs that do participate in shutdown must regularly check for "shutdown" messages from the NTM. (Using "NTMPU....." as the source).

b. AP Participates in IISS Recovery (Yes/No). APs that participate in IISS recovery provide recovery logic when they receive a "recover" indicator in the SYSTEM-STATE return of the "INITIAL" call. Upon completion of recovery processing, the AP must call "ENDRCY". The NTM will initiate all APs that do recovery processing during this IISS state.

c. AP's I/O Characteristics. The AP programmer must characterize the AP's I/O characteristics as one of the following.

1. Does not send to or receive any messages from other IISS APs. (i.e., supports no mailboxes).

2. Sends and receives messages; but sends no messages that require responses. (These APs will receive an error condition if they issue any PVC with a non-zero time-out value.)

3. Sends and Receives messages - that require responses as well as ones that may not. This indicates to the NTM that this AP may use NTM message pairing support.
4. Is a "Queue Server" - Can receive messages from multiple APs (maximum number of APs from which it can receive messages during one program run must also be indicated).

d. AP Handling on Message Time-Outs. The services initially identified are:

1. AP receives time-out messages and decides whether to resubmit, terminate, or requests a clock reset by the NTM on the outstanding message (the clock reset service is an enhancement feature). This allows the program to decide whether it wants to wait a longer period of time for a response.

2. Is terminated by the NTM on a time-out.

e. AP Handling on "Child" or "Spawned Task" Termination

1. On normal child termination -
   - AP wants a termination status message (Yes/No)

2. On abnormal child termination or abort
   - AP wants a termination status message (Yes/No)

3. Require an abort of AP on "child" termination (Yes/No)

f. AP Chaining Support. AP's of the type Queue Server are not amenable to child chaining support. These AP's may, however, receive message chaining support from the NTM. All AP's must be identified with respect to the type of chaining support that is required, Child, Messages, or none.

The NTM record will also contain information that will be established by the CDM Administrator with the assistance of the AP Programmer. This includes the:

a. maximum number of concurrent instances of the AP allowed, and
b. the maximum number of queued messages allowed for the AP.
APPENDIX C

INTEGRATING IISS COMPONENT APs

C.1 User Interface

The User Interface (UI) is the application process that interfaces to the IISS user terminals. Initially, there will be one UI per IISS terminal with a number of UIs associated with one AP Cluster. Conceptually, the NTM-UI Interface is depicted in Figure C-1.

Figure C-1. NTM-UI Interface

1. UI represents the UI code.
   UI (AP) Interface is the special AP Interface for the UI.
   The UI code is bound with the UI (AP) Interface.
2. The UI-1 Mailbox is the Input mailbox for the first instance of the UI.
3. These represent the AP Cluster high and low priority mailboxes.
4. The MPU is the Message Processing Unit of the NTM.

The connection protocol of these terminals causes this AP to be handled in a slightly different manner than other AP's. However, this difference is transparent to the UI APs. The UI (AP) Interface and the NTM protocols handle this special requirement by providing the necessary NTM connection logic on the UI's initiation call. CALL "INITEX" (see Section 5 for a description of the call and its arguments). The services of INITEX are described below. The formats of UI-NTM messages are
in Appendix A. This implementation is specific for the VAX under VMS.

C.1.1 UI-NTM Initiation Service (INITEX)

The UI requires an "external" initiation connection service that is supplied by the routine "INITEX." This allows a user to logon to an IISS the NTM, and initiate a UI process. The UI must now connect to the NTM, rather than the NTM initiating the UI as in the normal IISS process initiation procedure. External connection is required in light of the way that the UI will manage the terminals. INITEX performs the following initiation functions for the UI.

1. Sends an "I'm alive" message that contains the UI's operating system given process name to the UI's NTM.

2. Creates the UI's input mailboxes.

3. Establishes the IISS condition handler for the UI.

4. Saves the UI's buffer address and buffer size for later message services.

5. Returns to the UI with the status of the initiation.

C.1.2 The UI and Logical-Channels

A UI will manage communications between the NTM and any terminals connected to it (initially, one). The logical channel specifier provides a mechanism for the UI to map messages to terminals, or to multiple screens on a given terminal. The UI can manage the mapping between messages and screens or terminals by maintaining a table that carries the current channel assignments for a terminal or screen and using the channel numbers as suggested in Section 3.

The requirement for an AP to send an unsolicited form to a terminal can be supported by the use of a specified channel for UI unsolicited messages (channel 000 is being reserved for these unsolicited messages for single-terminal Uls).

Multiterminal Uls can also be supported by the NTM with a slight modification to the INITEX routine. The UI will assign
blocks of channel specifiers to a terminal when the terminal logs on. The NTM will need to know the channel numbers associated with a logon, and will provide a message format for this data when multiterminal UIs are developed.

C.1.3 UI-NTM Interface Programming Conventions

The UI must use the following guidelines to communicate successfully in the IISS Test Bed:

1. It must be bound with the UI (AP) Interface supplied by the NTM.

2. It must initiate communications with the NTM with a 'CALL "INITEX" USING'.

3. It must support asynchronously received (unsolicited) messages using CALL "RCV" or CALL "CHKMSG" and CALL "RCV" at regular intervals.

4. It must handle "shutdown" messages.

5. If the IISS is in a recovery or down state, the UIs will get a "connection-failure-reason" status return on the CALL "INITEX." It should inform the user at the terminal of the state of the IISS and provide a logoff or local mode capability to the user.

6. It must use the NTM Service Calls (Section 5) to communicate with other IISS APs.

7. It should terminate with the NTM call, "TRMNAT".
C.2 COMM

The Communication or COMM AP cluster will support a COMM application process for each host connection. The COMM AP cluster on the VAX host is conceptually represented in Figure C-2.

![Diagram of NTM-COMM Interface](image)

**Figure C-2. NTM-COMM Interface**

Because the COMM APs perform many of the AP Interface functions directly, they will have only a selected set of AP Interface routines bound to them. The exact configuration of routines has not been determined. However, from the initialization logic described in Section 3 and the NTM Services in Section 5, a tailored interface can be designed for the COMM APs.

C.3 The CDMRP

The Common Data Model Request Processor (CDMRP) is treated as a new AP by the NTM and will use the conventions of Section 3 to develop its IISS communication capability. It should handle asynchronously received messages, the IISS recovery mode shutdown, and time-outs.
APPENDIX D
HELPFUL HINTS

1. Paired Messages must have a non-zero timeout value on the CALL "SEND". The receiving AP must expect a return of "REV-REPLY-REQUIRED-MESSAGE" on its CALL "RCV".

2. AP's having the characteristic of requiring a specific initiation message must be started with a CALL "ISEND". An initiation message sent with a CALL "NSEND" will be rejected for these AP's.

3. CALL "CHKMSG" checks the mailbox once. If the AP needs to poll the mailboxes (for NTM messages, for example) it must invoke the call periodically. A timer or a counter may be used. As a guideline, when IISS shutdown is pending, a message to that effect is sent every minute until shutdown procedures actually begin.

4. "TRNMAT" is the last executable statement in a program. This service ends the AP's execution. The AP should not stop it's own run.

5. The logical channel is a critical variable in pairing messages. It must be specified whenever the sending AP wants to ensure that it will receive the correct response from the correct instance of the responding AP. See Section 3.1.6 for details.

6. On any "SEND" call, the sending AP must specify the destination message type, and data length. There are no default values for these data items.

7. When CALL "CHKMSG" returns the information that a message has been found, the AP can immediately invoke the CALL "RCV" to retrieve the message.

8. AP's invoking any service must expect to receive any of the returns.

9. When the AP wants to receive a message from its hot mailbox, it must specify "NTMPU......" as the source in the calling parameters. This holds for both CALL "RCV" and CALL "CHKMSG".
10. Message Types may be defined by the AP programmer. The type "X1" and "SE" are the only type values currently reserved for the NTM.

11. If the AP supports a "hot" mailbox, a call "RCV" with any match or channel match will cause the service to poll both the hot and cold mailboxes for a message.

12. If an AP wishes to receive asynchronous error messages, a call to TSTMOD is required to set test mode to on.

13. The directory prefix of the destination AP must be specified in any "SEND" call. If the NTM cannot specify the location of the executable module, initiation cannot take place.
APPENDIX E

REMOTE COMPILE AND LINK

E.1 RCL Overview

The Function of RCL is to provide IISS programmers and users with the ability to have primitive operations performed on a specified node. This is beneficial to the AP programmer who wishes to have system commands executed from within the AP. For the IISS user RCL provides the expertise to have system commands built and executed to perform a desired task.

The RCL is divided into three layers, as shown in Figure E-1. The "expert" layer actually gets the commands executed. This layer can be used by AP programmers who have specific operations to perform and wish to build their own system commands. It is also used by the next layer up. The "novice" layer can be used by AP programmers who wish to have operations performed without having to build the necessary system command. The novice layer is also used by the third layer. The "user" layer provides the interface between the external IISS user and RCL services.

The RCL is a component of the NTM services. Its relationship to the rest of the NTM is shown in the RCL Software Architecture Diagram, Figure E-2. The "expert" layer has been implemented on the VAX node. It still remains to be implemented on the other IISS nodes. The "novice" layer services have been designed but not implemented. The "user" layer is still in the development stage.

E.1.1 Expert Layer

The function of "expert" layer, or RCLE, is to execute system control cards which it receives in a message at initiation. The RCLE AP performs this task by building a system command file which it then executes. This command file includes the control cards found in the message area with some additional command file instructions. The RCLE AP is started by using the SNDRCLE services.

When RCLE has completed its tasks, it will send a message to its parent to return the status and results of its execution. This message will contain a code to indicate the success or failure of the operation. and, if successful, the message will
NOTE: RCL PROGRAMS INTERFACE WITH MPU

Figure E-1. RCL Program Flow
contain the name of a file which contains the information generated by the command file which was created but not executed. The file that is returned should be deleted by the caller when it is finished processing the information.

E.1.2 Novice Layer

The "novice" layer consists of services to provide five operations. These operations are compiles, links, directory/catalog listings, deletion of files, and insertion of VAX files into libraries. The services are named RCLCOM, RCLINK, RCLCAT, RCLDEL, and RCLSTO. Each service will call on the RCLN AP on the proper node to have the control card built for the operation.

The RCLN AP is responsible for building the actual control card, calling on RCLE ot have the operation performed, deleting the output file after it is finished checking the result of the operation, and returning the success or failure status to the caller of the "novice" service.

E.1.3 RCLETEST

A test program, RCLETEST, has been written to demonstrate the "expert" layer of RCL on the VAX. The test runs as an external AP which calls on the RCLE AP. The test provides two methods for having control cards executed by RCLE.

The RCLEDEMO option has RCLE execute the TSRCL command file. This command file contains the system commands to compile the COBOL source file "TSRCL.COB," link the object code which is created, and then run the executable image. The TSRCL.COB is a simple COBOL program which inputs its own source code and prints it to the output file. After running the program, the command file will also provide cleanup by deleting the object and image files created. When this test option is run successfully, the returned output file will contain a copy of the TSRCL.COB source.

The second test option allows the tester to submit his own commands to be executed by RCLE. These may be any valid system commands including one to start a tester written command file. Any command files submitted will be run in the batch mode. The tester then can check the output file to determine if the expected results were returned.
E.2 Program Specification for the RCLE AP

E.2.1 Function of RCLE

1. The function of the RCLE AP is to execute system control cards which it receives in a message at initiation. The AP performs this task by building a system command file which it then executes. This command file includes the control cards found in the message area with some additional command file instructions.

2. When RCLE has completed its tasks, it will send a message to its parent to return the status and results of its execution. This message will contain a code to indicate the success or failure of the operation, and, if successful, the message will include the name of a file which contains the information generated by the command file. If RCLE was unsuccessful, the message area will contain the name of the command file which was created but not executed.

3. RCLE uses the T1V message processing unit (MPU). The AP has VAX specifics, but the logic was designed for conversion to another node. This AP requires the privilege to create a detached process.

E.2.2 Implementation of RCLE

1. The RCLE AP is implemented in five major steps. Two of the steps, the first and the last, are the standard initiation and termination of a child AP. The remaining three are building the command file, executing the command file, and sending the message to the parent.

2. The command file is built with the help of three subroutines and the filenamer AP accessed through an interface. The subroutine RCLFILE opens, closes, and writes to the command file built by RCLE. These control cards turn off command file verification, direct command file program flow in the event of an error, and delete the command file before exiting. The filenamer AP is used to generate a unique filename for the command file. The RCLE builds the command file in the following steps:

A. Call on the filenamer to get a unique name for the command file.

B. Call RCLFILE to open the command file.
C. Call PRECOM to write system-specific control cards needed before the actual control cards.

D. Write the control cards received in the message area to the command file.

E. Call POSTCOM to write system-specific control cards needed after the actual control cards.

F. Call RCLFILE to close the command file.

3. The command file is executed by starting it as a detached process. This is accomplished by using the VAX SYSTEM service $CREPRC. The filenamer AP is also used again to generate a unique name for the output file parameter in $CREPRC. This output file will receive all information generated by the execution of the command file. After the command file is started, RCLE periodically attempts to open the command file. If it is unsuccessful in doing so, it knows that the command file has deleted itself and completed execution.

4. A message is sent by RCLE to its parent through the normal NTH NSEND service call. The message contains two things. A status code indicates the success or failure of RCLE (an unsuccessful call to system service $CREPRC or failure of the filenamer AP would result in a failure code). The second part of the message would contain the name of a file. If RCLE is successful, this file will be the output file from the execution of the command file. If RCLE is unsuccessful, it will be the name of the command file which was built but not executed.
E.3 Program Specification for RCLETEST

The RCLETEST AP demonstrates the capabilities of the RCL "expert" layer. This test will go through the logon procedure, initialize to the NTM, give the test person the demonstration options, and go through the logoff procedure when the tester is finished. The test options are to run the TSRCL command or to allow the test person to submit his own commands or file for execution.

The TSRCL command file contains the system commands to compile the COBOL source file "TSRCL.COB," link the object code which is created, and then run the executable image. The TSRCL.COB is a simple COBOL program which inputs its own source code and prints it to the output file. After running the program, the command file will also perform clean-up by deleting the created object and executable files. When this test is run successfully, the test person will be returned the name of an output file which contains the generated information including the output from the COBOL program.

If the tester selects the second test option, he will be prompted to submit his own system commands for execution. These may be any valid system commands including one to run a tester-written command file. The tester must realize, though, that if a command file is submitted to RCL for execution, it will be run in batch mode and not interactively. The tester ends the input of commands by entering a blank line.

Both test options are then followed by a request for the tester to enter the node to be used in the operation. The RCLE only exists on the VAX node at this time, but other nodes may be input to determine what results will occur if an inactive or invalid node is entered. If a blank line is entered, the test will default to the VAX node.

When the node has been entered, the tester will be returned to status of the SNDRCLE service call. If the node was valid and service to initiate RCLE returned a good status code, the tester will receive the message that the RCLE AP has been initiated. The test program will now be waiting on a response from RCLE. The test person can check the status of the test program by entering a blank line. (Entering CANCEL will cause SIGABT to be called to terminate RCLE.)

When RCLE completes execution, it will send a message to the test AP. The next status request will then contain the
results of RCLE. If RCLE completed successfully, the test person will be given the name of the output file which contains the results of the operation. If RCLE completed unsuccessfully, the tester will be given the name of the command file which was built but not executed.

At the completion of a test, the test person will be given the test options again. If he is finished, the END option should be chosen. This causes the test AP to terminate from the NTM. At termination RCLETEST will also give the test person the name of a command file to be run to delete all files his tests have generated.
F.1 Introduction

This RCL Programmer's Guide describes the services provided to IISS Programmers by the RCL component of the Network Transaction Manager (NTM). These services are used by IISS Application Programs to perform specific system tests.

The RCL guide is being released as a reference for component AP programmers who desire to use the service currently available and, as a development, report on the RCL services yet to be implemented. The available service will be fully documented on its functionality and usage. The services yet to be implemented will be described along with considerations still to be made in their development.

F.2 RCL Overview

The RCL provides the AP programmer with the ability to perform specific system tasks from within his AP. It consists of two APs on each node and six services to call the appropriate AP. The services and APs of the RCL are separated into two groups.

The "expert" layer allows the programmer to call the expert service to get the appropriate AP initiated to perform the tasks specified by the control cards he has built. The expert service will validate the node and that the control card area has the proper end-of-lines and a termination sequence but leaves the details of creating valid system control cards up to the calling AP. Once the service determines the validity of the call, it will return a status to the caller. If a valid call was made, the RCLE AP on the requested node will be initiated to perform the tasks and send the results back to the caller.

The "novice" layer takes the burden of building valid control cards away from the programmer and puts it into the hands of the novice services and APs. The services provide for performing five different system tasks. Each service will validate the node of the requested operation and then call the novice AP on the appropriate node, if valid. The novice AP will actually build the card and then call the expert layer to have the service performed. The RCLN will receive the generated
information, determine if the operation was a success or failure, and send it to the caller of the novice service.

F.3  RCL Services

F.3.1  RCL Service Availability

The RCL Expert Service is currently available. The service return parameter values are defined and an example of the service usage is given.

The values of the service returns are defined in the include member "SRVRET."
Send control card(s) to RCLE AP for execution.

Calling Sequence:

CALL "SNDRCLE" USING NODE.
CHANNEL.
CTL-CARD.
RET-CODE.

Description:

SNDRCLE will check that the caller has requested the initiation of a RCLE AP on a valid node with a valid control card area format. (See page F-19 for a description of the control card area.) If these parameters are valid, SNDRCLE will initiate an instance of the proper AP. The service returns a status to the caller. If a valid status is returned, the caller will want to issue a RCV to receive the information from the RCLE AP. The information will be in a file generated by RCLE. The caller is responsible for deleting this file after extracting the desired information.

The received message will have the following format:

<table>
<thead>
<tr>
<th>BYTE</th>
<th>CONTENTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-3</td>
<td>IDENTIFIER ROLE</td>
</tr>
<tr>
<td>4</td>
<td>blank</td>
</tr>
<tr>
<td>5</td>
<td>STATUS CODE</td>
</tr>
<tr>
<td>6</td>
<td>blank</td>
</tr>
<tr>
<td>3-36</td>
<td>Filename containing information</td>
</tr>
</tbody>
</table>

If the message is identified as a RCLE message, the caller will want to check the status code for success or failure. An 'S' code indicates successful completion of RCLE. An 'F' code indicates failure.

Inputs:

NODE
CHANNEL
CTL-CARD
Outputs:

RET-CODE

RET-CODE Values:
(Values are defined in the include member SRVRET)

<table>
<thead>
<tr>
<th>Legal Value</th>
<th>Value Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>RCL-INITIATED</td>
<td>Call valid / RCLE started.</td>
</tr>
<tr>
<td>RCLE-NODE INVALID</td>
<td>Invalid node for IISS.</td>
</tr>
<tr>
<td>RCLE-LINE-INVALID</td>
<td>Control card exceeded 73 characters in length.</td>
</tr>
<tr>
<td>RCLE-CARD-INVALID</td>
<td>Control card area did not end with termination sequence.</td>
</tr>
<tr>
<td>RCLE-INITIATION FAILED</td>
<td>The RCLE AP could not be initiated.</td>
</tr>
</tbody>
</table>
Example:

1. To compile the COBOL source "EXAMPLE.COB" on VAX host.

   01 CTL-CARD.
   05 CC-COMMAND PIC X(10) VALUE SPACES.
   05 CC-FILENAM PIC X(60) VALUE SPACES.
   05 TERMINATOR PIC XX VALUE ' '.

* THE TERMINATOR IS THE TWO END OF LINE CHARACTERS (1Eh)
* REQUIRED TO END A VALID CTL-CARD AREA.

   01 DATA-RCV PIC X(4096).
   01 RCLE-RCV REDEFINES DATA-RCV.
   05 IDENTIFIER PIC X(4).
   05 FILLER PIC X.
   05 STATUS PIC X.
       88 RCLE-SUCCESS VALUE 'S'.
       88 RCLE-FAILED VALUE 'F'.
   05 FILLER PIC X.
   05 INFO FILE PIC X(30).
   05 FILLER PIC X(4059).

   MOVE 'VAX' TO NODE.
   MOVE '001' TO CHANNEL.
   MOVE 'COB/ANSI' TO CC-COMMAND.
   MOVE 'EXAMPLE' TO CC-FILENAM.
   CALL "SNDRCLE" USING NODE.
       CHANNEL.
       CTL-CARD.
       RET-CODE.

IF ROLE-INITIATED
   PERFORM GET-RCLE-MESSAGE
ELSE
   PERFORM ERROR-PROCEDURE.
GET-RCLE-MESSAGE.
MOVE SPACES TO LOGICAL-CHANNEL.
MOVE SPACES TO MSG-SOURCE.
MOVE SPACES TO DATA-RCV.
PERFORM CHKMSG-PARAGRAPH
UNTIL CHKMSG-MESSAGE-FOUND.
CALL "RCV" USING LOGICAL-CHANNEL.
   WAIT-FLAG.
   MSG-SOURCE.
   MSG-TYPE-RCV.
   DATA-LENGTH-RCV.
   DATA-RCV.
   ACCEPT-STATUS.
   MESSAGE-SERIAL-NUMBER.
IF IDENTIFIER = RCLE
   PERFORM RCV-LAST-MESSAGE
IF RCLE-SUCCESS
   NEXT SENTENCE
ELSE
   PERFORM ERROR-PROCEDURE
ELSE
   PERFORM UNSOLICITED-MESSAGE.
E.3.2 Novice Services Which Remain To Be Implemented

RCLCAT

Obtain a Directory / Catalog listing.

Calling Sequence:

CALL RCLCAT USING NODE.
   CHANNEL.
   CATALOG.
   LISTFILE.
   RET-CODE.

Description:

RCLCAT will put a directory / catalog listing specified by CATALOG into the file LISTFILE. The service will initially check that NODE is a valid IISS node and then initiate RCLN on that node to build the proper control card. A status return will be returned to the caller and, if valid, the caller should issue a RCV to receive the completion status of the operation.

Inputs:

   NODE
   CHANNEL
   CATALOG
   LISTFILE

Outputs:

   RET-CODE
RCLCAT (Continued)

RET-CODE Values:
(Values are defined in the include member SRVRET)

<table>
<thead>
<tr>
<th>Legal Value</th>
<th>Value Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>RCLN-INITIATED</td>
<td>Call valid / RCLN started.</td>
</tr>
<tr>
<td>RCLCAT-NODE-INVALID</td>
<td>Invalid node for IISS.</td>
</tr>
<tr>
<td>RCLN-INITIATION-FAILED</td>
<td>The RCLN AP could not be initiated.</td>
</tr>
</tbody>
</table>
RCLOCOM

Compile (COBOL, Fortran, C, FLAN) source code.

Calling Sequence:

CALL RCLOCOM USING NODE.
    CHANNEL.
    FUNCTION.
    SOURCE.
    OBJECT.
    LIST.
    ERROR.
    RET-CODE.

Description:

RCLOCOM will do the compile specified in FUNCTION on the source code specified by SOURCE. The service will initially check that NODE is a valid IISS node and then initiate RCLN on that node to build the proper control card. A status will be returned to the caller and, if valid, the caller should issue a RCV to receive the completion status of the operation. Development questions still remain on what parameters should be included on the control card and if the service should give the caller the option to have certain parameters included.

Inputs:

    FUNCTION
    SOURCE
    OBJECT
    LIST
    ERROR

Outputs:

    RET-CODE
### RET-CODE Values
(Values are defined in the include member SRVRET)

<table>
<thead>
<tr>
<th>Legal Value</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>RCLN-INITIATED</td>
<td>Call valid / RCLN started.</td>
</tr>
<tr>
<td>RCLCOM-NODE-INVALID</td>
<td>Invalid node for IISS.</td>
</tr>
<tr>
<td>RCLN-INITIATION-FAILED</td>
<td>The RCLN AP could not be initiated.</td>
</tr>
</tbody>
</table>
RCLDEL

Delete a file / dataset.

Calling Sequence:

CALL RCLDEL USING NODE.
   CHANNEL.
   DEL-FILE
   RET-CODE.

Description:

RCLDEL will delete the file specified in DEL-FILE. The service will initially check that NODE is a valid IISS node and then initiate RCLN on that node to build the proper control card. A status return will be returned to the caller and, if valid, the caller should issue a RCV to receive the completion status of the operation.

Inputs:

   NODE
   CHANNEL
   DEL-FILE

Outputs:

   RET-CODE
### RCLDEL (Continued)

**RET-CODE Values:**
(Values are defined in the include member SRVRET)

<table>
<thead>
<tr>
<th>Legal Value</th>
<th>Value Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>RCLN-INITIATED</td>
<td>Call valid / RCLN started.</td>
</tr>
<tr>
<td>RCLDEL-NODE-INVALID</td>
<td>Invalid node for IISS.</td>
</tr>
<tr>
<td>RCLN-INITIATION-FAILED</td>
<td>The RCLN AP could not be started.</td>
</tr>
</tbody>
</table>
RCLINK

Link object files and libraries.

Calling Sequence:

CALL RCLINK USING NODE.
    CHANNEL.
    EXEC.
    P1.
    P2.
    RET-CODE.

Description:

RCLINK will link the object files and libraries specified in P1 and P2. P1 is an array to contain up to eight object files to be linked. At least one file must be specified. P2 is an array to contain up to eight optional object libraries to be used in the link. The created executable will be in the file specified in EXEC. The service will initially check that NODE is a valid IISS node and then initiate RCLN on that node to build the proper control card. A status will be returned to the caller and, if valid, the caller should issue a RCV to receive the completion status of the operation. Development questions remain on what parameters should be included on the control card and if the service should give the caller the option to have certain parameters included.

Inputs:

    NODE
    CHANNEL
    EXEC
    F1
    F2

Outputs:

    RET-CODE
<table>
<thead>
<tr>
<th>Legal Value</th>
<th>Value Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>RCLN-INITIATED</td>
<td>Call valid / RCLN started.</td>
</tr>
<tr>
<td>RCLINK-NODE-INVALID</td>
<td>Invalid node for IISS.</td>
</tr>
<tr>
<td>RCLN-INITIATION-FAILED</td>
<td>The RCLN AP could not be initiated.</td>
</tr>
</tbody>
</table>
RCLSTO

Store file into object library (VAX only).

Calling Sequence:

CALL RCLSTO USING NODE.
    CHANNEL.
    STORE-FILE.
    LIBRARY.
    NAME.
    RET-CODE.

Description:

RCLSTO is a VAX-specific service to store the file specified in STORE-FILE in the library specified in LIBRARY as NAME. The service will initially check that NODE is valid for this operation and then initiate RCLN to build the proper control card. A status will be returned to the caller and, if valid, the caller should issue a RCV to receive the completion status of the operation.

Inputs:

    NODE
    CHANNEL
    FILE
    LIBRARY
    NAME

Outputs:

    RET-CODE
RCLSTO (Continued)

RET-CODE Values
(Values are defined in the include member SRVRET)

<table>
<thead>
<tr>
<th>Legal Value</th>
<th>Value Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>RCLN-INITIATED</td>
<td>Call valid / RCLN started.</td>
</tr>
<tr>
<td>RCLSTO-NODE-INVALID</td>
<td>Invalid node for IISS.</td>
</tr>
<tr>
<td>RCLN-INITIATION-FAILED</td>
<td>The RCLN AP could not be started.</td>
</tr>
</tbody>
</table>
3.3 Description of Parameters Used in RCL Service Calls

CATALOG

The parameter CATALOG contains the fully qualified name of a directory or catalog to be listed by RCLCAT.

01 CATALOG PIC X(100).

CHANNEL

The parameter CHANNEL contains a value which is used by APs and the RCL services to send messages via the NTM. Its description in COBOL is

01 CHANNEL PIC X(3).

CTL-CARD

The parameter CTL-CARD is an area to contain system control cards to be processed by SNDRCLE. Individual cards in this area cannot exceed 73 characters including an end-of-line (1Eh) indicator. The CTL-CARD area has a size of 1096 bytes, which allows for a maximum of 15 control cards of the maximum length of 73 with one additional end-of-line indicator at the end to terminate all control cards. Amounts and size of control cards less than these are allowable with service performance being improved by efficient use of the end of lines. If any line exceeds 73 characters, or the CTL-CARD area does not contain the final end-of-line terminator, an error code will be returned by SNDRCLE. The parameter’s description in COBOL is

01 CTL-CARD PIC X(1096).

DEL-FILE

The parameter DEL-FILE contains the fully qualified name of a file to be deleted by RCLDEL. Its description in COBOL is

01 DEL-FILE PIC X(100).

ERROR

F-17
The parameter ERROR contains the fully qualified name of a file to receive error results from a compile started by RCLCOM. Its description in COBOL is

```
01 ERROR PIC X(100).
```

EXEC

The parameter EXEC contains the fully qualified name of a file to receive the executable code generated by RCLINK. Its description in COBOL is

```
01 EXEC PIC X(100).
```

FUNCTION

The parameter FUNCTION contains the type of Compile to be started by RCLCOM. Options are 'COB', 'FOR', 'C', AND 'FLAN'. The parameter's description in COBOL is

```
01 FUNCTION PIC X(4).
```

LIBRARY

The parameter LIBRARY contains the fully qualified name of an object library for a file to be stored in by RCLSTO. Its description in COBOL is

```
01 LIBRARY PIC X(100).
```

LIST

The parameter LIST contains the fully qualified name of a file to receive the listing from a compile started by RCLCOM. Its description in COBOL is

```
01 LIST PIC X(100).
```

LISTFILE

The parameter LISTFILE contains the fully qualified name of a file to receive the directory / catalog listing from RCLCAT.
Its description in COBOL is

01 LISTFILE          PIC X(100).

NAME

The parameter NAME contains the name for a VAX object to be stored in an object library by RCLSTO. Its description in COBOL is

01 NAME             PIC X(100).

NODE

The parameter NODE contains the IISS node for a requested operation. Its description in COBOL is

01 NODE             PIC X(3).

OBJECT

The parameter OBJECT contains the fully qualified name of a file for the object to be stored by a compile started by RCLCOM. Its description in COBOL is

01 OBJECT           PIX X(100).
The parameters P1 and P2 are arrays to contain names of object files or libraries to be used in the linking started RCLINK. P1 contains up to eight object files, and P2 contains up to eight object libraries. The parameter descriptions in COBOL are

```cobol
01 P1 PIC X(800).
01 P1TBL REDEFINES P1.
   05 OBJFILES OCCURS 8 TIMES
       INDEXED BY P1-INDEX.
       10 FILENAM PIC X(100).
01 P2 PIC X(800).
01 P2TBL REDEFINES P2.
   05 LIBRARIES OCCURS 8 TIMES
       INDEXED BY P2-INDEX.
       10 LIBNAM PIC X(100).
```

RET-CODE

The parameter RET-CODE contains a value which indicated the return status of a specific request. Its description in COBOL is

```cobol
01 RET-CODE PIC X(5).
```

SOURCE

The parameter SOURCE contains the fully qualified name of a file to be compiled through RCLCOM. Its description in COBOL is

```cobol
01 SOURCE PIC X(100).
```

STORE-FILE

The parameter STORE-FILE contains the fully qualified name of a file to be stored in an object library by RCLSTO. Its description in COBOL is

```cobol
01 STORE-FILE PIC X(100).
```
END
8-81
P71C