Signalling Connection Control Part Interface (SCCPI) Specification

Signalling Connection Control Part Interface (SCCPI)

Specification

Version 1.1 Edition 7.20141001
Updated October 25, 2014
Distributed with Package openss7-1.1.7.20141001

Abstract:

This document is a Specification containing technical details concerning the implementation of the Signalling Connection Control Part Interface (SCCPI) for OpenSS7. It contains recommendations on software architecture as well as platform and system applicability of the Signalling Connection Control Part Interface (SCCPI). It provides abstraction of the Signalling Connection Control Part (SCCP) interface to these components as well as providing a basis for Signalling Connection Control Part control for other Signalling Connection Control Part protocols.

Brian Bidulock <bidulock@openss7.org> for
The OpenSS7 Project <http://www.openss7.org/>

Published by:

OpenSS7 Corporation
1469 Jefferys Crescent
Edmonton, Alberta T6L 6T1
Canada

Notice:

OpenSS7 Corporation disclaims all warranties with regard to this documentation including all implied warranties of merchantability, fitness for a particular purpose, non-infringement, or title; that the contents of the document are suitable for any purpose, or that the implementation of such contents will not infringe on any third party patents, copyrights, trademarks or other rights. In no event shall OpenSS7 Corporation be liable for any direct, indirect, special or consequential damages or any damages whatsoever resulting from loss of use, data or profits, whether in an action of contract, negligence or other tortious action, arising out of or in connection with any use of this document or the performance or implementation of the contents thereof.

Short Contents

Table of Contents

List of Figures

List of Tables

Preface

Notice

Abstract

This document is a Specification containing technical details concerning the implementation of the Signalling Connection Control Part Interface (SCCPI) for OpenSS7. It contains recommendations on software architecture as well as platform and system applicability of the Signalling Connection Control Part Interface (SCCPI).

This document specifies a Signalling Connection Control Part Interface (SCCPI) Specification in support of the OpenSS7 Signalling Connection Control Part (SCCP) protocol stacks. It provides abstraction of the Signalling Connection Control interface to these components as well as providing a basis for Signalling Connection Control control for other Signalling Connection Control protocols.

Purpose

The purpose of this document is to provide technical documentation of the Signalling Connection Control Part Interface (SCCPI). This document is intended to be included with the OpenSS7 STREAMS software package released by *OpenSS7 Corporation*. It is intended to assist software developers, maintainers and users of the Signalling Connection Control Part Interface (SCCPI) with understanding the software architecture and technical interfaces that are made available in the software package.

Intent

It is the intent of this document that it act as the primary source of information concerning the Signalling Connection Control Part Interface (SCCPI). This document is intended to provide information for writers of OpenSS7 Signalling Connection Control Part Interface (SCCPI) applications as well as writers of OpenSS7 Signalling Connection Control Part Interface (SCCPI) Users.

Audience

The audience for this document is software developers, maintainers and users and integrators of the Signalling Connection Control Part Interface (SCCPI). The target audience is developers and users of the OpenSS7 SS7 stack.

Revision History

Take care that you are working with a current version of this documentation: you will not be notified of updates. To ensure that you are working with a current version, check the OpenSS7 Project website for a current version.

A current version of this specification is normally distributed with the *OpenSS7* package, `openss7-1.1.7.20141001`.[1]

Version Control

Although the author has attempted to ensure that the information in this document is complete and correct, neither the Author nor OpenSS7 Corporation will take any responsibility in it. *OpenSS7 Corporation* is making this documentation available as a reference point for the industry. While *OpenSS7 Corporation* believes that these interfaces are well defined in this release of the document, minor changes may be made prior to products conforming to the interfaces being made available. *OpenSS7 Corporation* reserves the right to revise this software and documentation for any reason, including but not limited to, conformity with standards promulgated by various agencies, utilization of advances in the state of the technical arts, or the reflection of changes in the design of any techniques, or procedures embodied, described, or referred to herein. *OpenSS7 Corporation* is under no obligation to provide any feature listed herein.

```
$Log: sccpi.texi,v $
Revision 1.1.2.2  2011-02-07 02:21:42  brian
- updated manuals

Revision 1.1.2.1  2009-06-21 10:56:05  brian
- added files to new distro
```

ISO 9000 Compliance

Only the TEX, texinfo, or roff source for this maual is controlled. An opaque (printed, postscript or portable document format) version of this manual is a **UNCONTROLLED VERSION**.

Disclaimer

OpenSS7 Corporation **disclaims all warranties with regard to this documentation including all implied warranties of merchantability, fitness for a particular purpose, non-infrincement, or title; that the contents of the manual are suitable for any purpose, or that the implementation of such contents will not infringe on any third party patents, copyrights, trademarks or other rights. In no event shall** *OpenSS7 Corporation* **be liable for any direct, indirect, special or consequential damages or any damages whatsoever resulting from loss of use, data or profits, whether in an action or contract, negligence or other tortious action, arising out of or in connection with any use of this documentation or the performance or implementation of the contents thereof.**

U.S. Government Restricted Rights

If you are licensing this Software on behalf of the U.S. Government ("Government"), the following provisions apply to you. If the Software is supplied by the Department of Defense ("DoD"), it is classified as "Commercial Computer Software" under paragraph 252.227-7014 of the DoD Supplement to the Federal Aquisition Regulations ("DFARS") (or any successor regulations) and the Government is acquiring only the license rights granded herein (the license rights customarily provided to non-Government users). If the Software is supplied to any unit or agency of the Government other than DoD, it is classified as "Restricted Computer Software" and the Government's rights in the Software are defined in paragraph 52.227-19 of the Federal Acquisition Regulations ("FAR") (or any successor regulations) or, in the cases of NASA, in paragraph 18.52.227-86 of the NASA Supplerment to the FAR (or any successor regulations).

[1] http://www.openss7.org/repos/tarballs/openss7-1.1.7.20141001.tar.bz2

Acknowledgements

The OpenSS7 Project was funded in part by:

- Monavacon Limited
- OpenSS7 Corporation

Thanks to the subscribers to and sponsors of The OpenSS7 Project. Without their support, open software like this would not be possible.

As with most open source projects, this project would not have been possible without the valiant efforts and productive software of the Free Software Foundation, the Linux Kernel Community, and the open source software movement at large.

1 Introduction

This document specifies a STREAMS-based kernel-level instantiation of the ITU-T Signalling Connection Control Part Interface (SCCPI) definition. The Signalling Connection Control Part Interface (SCCPI) enables the user of a Signalling Connection Control service to access and use any of a variety of conforming Signalling Connection Control providers without specific knowledge of the provider's protocol. The service interface is designed to support any network Signalling Connection Control protocol and user Signalling Connection Control protocol. This interface only specifies access to Signalling Connection Control service providers, and does not address issues concerning Signalling Connection Control management, protocol performance, and performance analysis tools.

This specification assumes that the reader is familiar with ITU-T state machines and Signalling Connection Control interfaces (e.g. Q.711, T1.112), and STREAMS.

1.1 Related Documentation

— **ITU-T Recommendation Q.711 (White Book)**

— **ETSI EN 300 009-1**

— **ANSI T1.112/2002**

— **System V Interface Definition, Issue 2 — Volume 3**

1.1.1 Role

This document specifies an interface that supports the services provided by the *Signalling System No. 7 (SS7)* for ITU-T, ANSI and ETSI applications as described in ITU-T Recommendation Q.711, ANSI T1.112, ETSI EN 300 009-1. These specifications are targeted for use by developers and testers of protocol modules that require Signalling Connection Control service.

1.2 Definitions, Acronyms, Abbreviations

Originating SL User
A SL-User that initiates a Signalling Link.

Destination SL User
A SL-User with whom an originating SL user wishes to establish a Signalling Link.

ISO International Organization for Standardization

SL User Kernel level protocol or user level application that is accessing the services of the Signalling Link sub-layer.

SL Provider
Signalling Link sub-layer entity/entities that provide/s the services of the Signalling Link interface.

SLI Signalling Link Interface

TIDU Signalling Link Interface Data Unit

TSDU Signalling Link Service Data Unit

OSI Open Systems Interconnection

QOS Quality of Service

STREAMS A communication services development facility first available with UNIX System V Release 3.

2 The SCCP Layer

The SCCP Layer provides the means to manage the NC of SCCP-Users into connections. It is responsible for the routing and management of data to and from SS7 network connections between SCCP-user entities.

2.1 Model of the SCCPI

The SCCPI defines the services provided by the signalling link layer to the signalling link user at the boundary between the signalling link provider and the signalling link user entity. The interface consists of a set of primitives defined as STREAMS messages that provide access to the signalling link layer services, and are transferred between the SCCP user entity and the SCCP provider. These primitives are of two types; ones that originate from the SCCP user, and other that originate from the SCCP provider. The primitives that originate from the SCCP user make requests to the SCCP provider, or respond to an indication of an event of the SCCP provider. The primitives that originate from the SCCP provider are either confirmations of a request or are indications to the CCS user that an event has occurred. Figure 2.1 shows the model of the SCCPI.

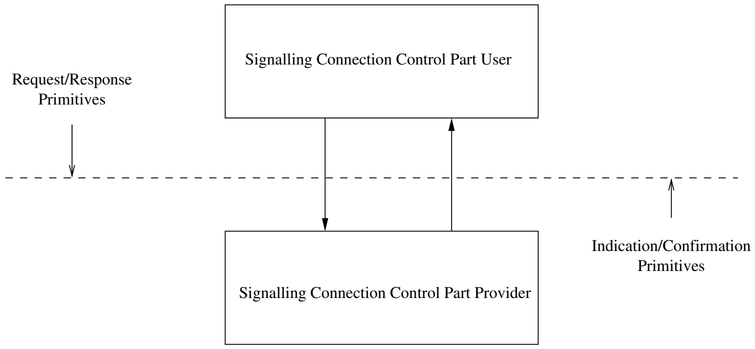
Signalling Connection Control Part User
Request/Response Primitives
Indication/Confirmation Primitives
Signalling Connection Control Part Provider

Figure 2.1: *Model of the SCCPI*

The SCCPI allows the SCCP provider to be configured with any SCCP user (such as TCAP) that also conforms to the SCCPI. A network layer user can also be a user program that conforms to the SCCPI and accesses the SCCP provider via `putmsg(2s)` and `getmsg(2s)` system calls.

2.2 SCCPI Services

The features of the SCCPI are defined in terms of the services provided by the SCCP, and the individual primitives that may flow between the SCCP-User and the SCCP.

The services supported by the SCCPI are based on two distinct modes of communication, connectionless (CLNS) and connection oriented (CONS). Within these modes, the SCCPI provides support for both sequenced and unsequenced message transfer. Also, the SCCPI supports services for local mangement.

2.2.1 CLNS

The main features of the connectionless mode of communication are:

1. it is datagram oriented;
2. it provides transfer of data in self contained units;
3. there is no logical relationship between these units of data.

Connectionless mode communication has no separate phases. Each unit of data is transmitted from source to destination independently, appropriate addressing information is included with each unit of data. Although the units of data are transmitted independently from source to destination, SCCP provides a high level of assurance of sequencing if sequenced service is requested. When unsequenced service is requested, there are no guarantees of proper sequence. Although SCCP services are inherently unreliable, SCCP provide a high level of assurance that messages are not lost.

The connectionless service of SCCP is suited to SCCP User protocols such as the Transaction Capabilities Application Part (TCAP).[1]

2.2.2 CONS

The main features of the SCCP connection oriented mode of communication are:

1. it is virtual circuit oriented;
2. it provides transfer of data via a pre-established path.

There are three phases to each instance of communication: Connection Establishment, Data Transfer; and Connection Termination. Units of data arrive at their destination in the same order as they departed their source when the sequenced delivery service is requested and the data is protected against duplication or loss of data within some specified quality of service.

The connection oriented service of SCCP is suited to SCCP User protocols such as the Integrated Services Digital Network User Part (ISUP), [Q.764] Telephone User Part (TUP), [Q.724] and Bearer Indexed Call Control (BICC).[2]

2.2.3 Local Management

The SCCPI specifications also defines a set of local management functions that apply to CONS and CLNS modes of communication. These services have local significance only.

2.2.4 Provider Management

The SCCPI specification also defines a set of provider management functions that apply to the SCCP service provider. These services have local and end-to-end significance.

2.3 SCCP Service Primitives

Table 2.1, Table 2.2, Table 2.3 and Table 2.4 summarize the SCCPI service primitives by their state and service

[1] ⟨undefined⟩ [Q.714], page ⟨undefined⟩.

[2] ISUP consists of *signalling relations* between two switches which also have digital facilities between them. In general an ISUP SCCP-User can communicate with many other SCCP-User peers, however, signalling between any given two enpoints only concerns the digital facilities which exist between the two endpoints. So, management of ISUP switches is best performed on a pairing of endpoints (*signalling relations*). Also, the CONS mode of operation is provided in support of DPC list Routing Keys for M3UA. [RFC 4666]

STATE	SERVICE	PRIMITIVES
Local Management	Information Reporting	N_INFO_REQ, N_INFO_ACK, N_ERROR_ACK
	Bind	N_BIND_REQ, N_BIND_ACK, N_UNBIND_REQ, N_OK_ACK, N_ERROR_ACK
	Options Management	N_OPTMGMT_REQ, N_OK_ACK, N_ERROR_ACK

Table 2.1: *SCCPI Service Primitives for Local Management*

STATE	SERVICE	PRIMITIVES
Data Transfer	Data Transfer	N_UNITDATA_REQ, N_UNITDATA_IND
	Error Management	N_UDERROR_IND, N_NOTICE_IND, N_STATE_IND, N_PCSTATE_IND

Table 2.2: *SCCPI Service Primitives for Connectionless Mode Data Transfer*

STATE	SERVICE	PRIMITIVES
Connection Establishment	Connection Establishment	N_CONN_REQ, N_CONN_IND, N_CONN_RES, N_CONN_CON, N_OK_ACK, N_ERROR_ACK
Data Transfer	Data Transfer	N_DATA_REQ, N_DATA_IND
	Error Management	N_NOTICE_IND, N_RESET_IND, N_DISCON_IND
Connection Release	Connection Release	N_DISCON_REQ, N_DISCON_IND, N_OK_ACK, N_ERROR_ACK

Table 2.3: *SCCPI Service Primitives for Connection Mode Data Transfer*

STATE	SERVICE	PRIMITIVES
Provider Management	Signalling Point Management	N_PCSTATE_IND, N_NOTICE_IND
	Mated Pair Management	N_COORD_REQ, N_COORD_IND, N_COORD_RES, N_COORD_CON, N_TRAFFIC_IND
	Layer Management	N_INFORM_REQ, N_INFORM_IND, N_STATE_REQ, N_STATE_IND

Table 2.4: *SCCPI Service Primitives for SCCP Management*

3 SCCPI Services Definition

This section describes the services of the SCCPI primitives. Time-sequence diagrams that illustrate the sequence of primitives are included.[1] The format of the primitives will be defined later in this document.

3.1 Local Management Services

The services defined in this section are outside the scope of international standards. These services apply to CONS and CLNS modes of communication. They are invoked for the initialization/de-initialization of a stream connected to the SCCP. They are also used to manage options supported by the SCCP and to report information on the supported parameter values.

3.1.1 Signalling Connection Control Part Information Reporting Service

This service provides information on the options supported by the SCCP provider.

N_INFO_REQ:
This primitive requests that the SCCP return the values of all the supported protocol parameters. This request may be invoked during any phase.

N_INFO_ACK:
This primitive is in response to the `N_INFO_REQ` primitive and returns the values of the supported protocol parameters to the SCCP-User.

The sequence of primitives for SCCP information management is shown in Figure 3.1.

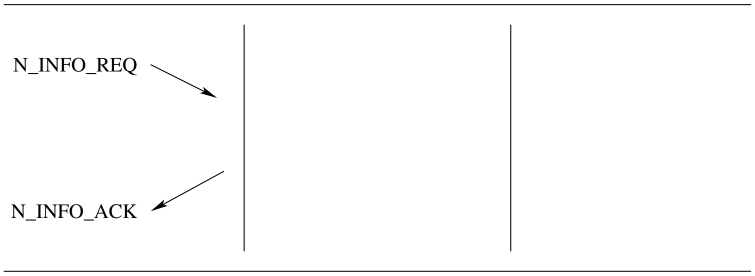

Figure 3.1: *Signalling Connection Control Part Information Reporting Service*

3.1.2 SCCP User Bind Service

This service allows an SCCP address (SCCP-SAPI) to be associated with a Stream.

It allows the SCCP-User to negotiate the number of connect indications that can remain unacknowledged for that SCCP-User (a connect indication is considered unacknowledged while it is awaiting a corresponding connect response or release request from the SCCP-User). This service also defines a mechanism that allows a Stream (bound to an SCCP address of the SCCP-User) to be reserved to handle incoming connections only. This Stream is referred to as the listener Stream.

[1] Conventions for the time-sequence diagrams are defined in ITU-T X.210. [X.210]

N_BIND_REQ:

This primitive requests that the SCCP-User be bound to a particular SCCP address (SCCP-SAPI), and negotiate the number of allowable outstanding connect indications for that address.

N_BIND_ACK:

This primitive is in response to the `N_BIND_REQ` primitive and indicates to the user that the specified SCCP-User has been bound to an SCCP address.

The sequence of primitives is shown in Figure 3.2.

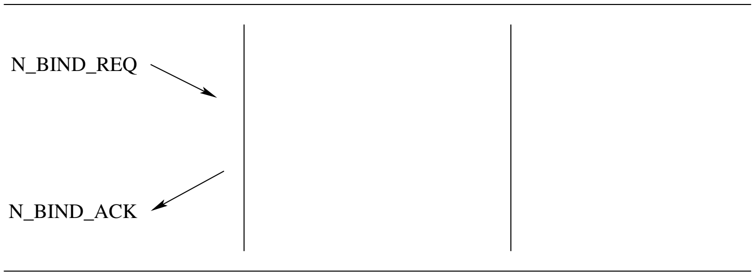

Figure 3.2: *Signalling Connection Control Part User Bind Service*

3.1.3 SCCP User Unbind Service

This service allows the SCCP-User to be unbound from an SCCP address.

N_UNBIND_REQ:

This primitive requests that the SCCP-User be unbound from the SCCP address that it had previously been bound to.

The sequence of primitives is shown in Figure 3.3.

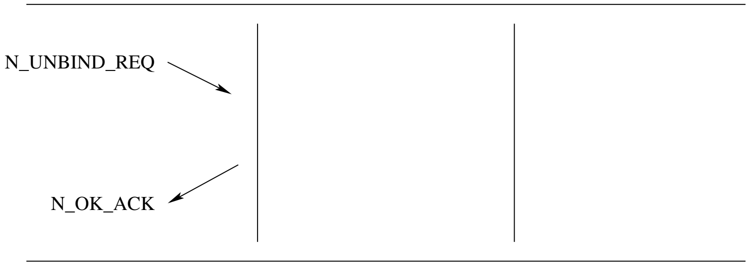

Figure 3.3: *Signalling Connection Control Part User Unbind Service*

3.1.4 Receipt Acknowledgement Service

N_OK_ACK:

This primitive indicates to the SCCP-User that the previous SCCP-User originated primitive was received successfully by the SCCP.

An example showing the sequence of primitives for successful receipt acknowledgement is depicted in Figure 3.4.

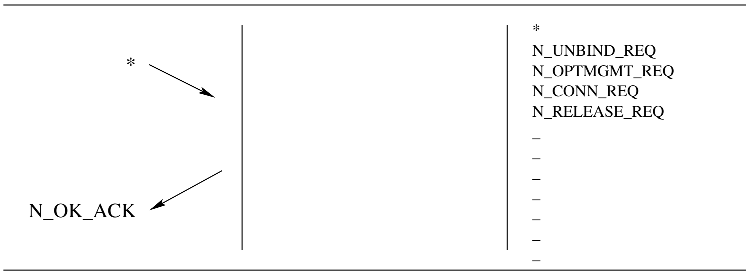

Figure 3.4: *Signalling Connection Control Part Receipt Ackknowledgement Service*

3.1.5 Options Management Service

This service allows the SCCP-User to manage options parameter values associated wtih the SCCP.

N_OPTMGMT_REQ:

This primitive allows the SCCP-User to select default values for options parameters within the range supported by the SCCP, and to indicate the default selection of receipt confirmation.

Figure 3.5 shows the sequence of primitives for SCCP options management.

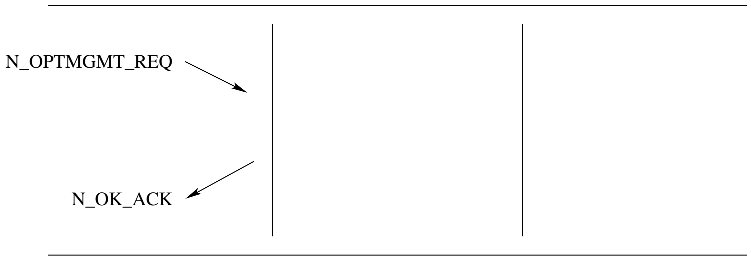

Figure 3.5: *Signalling Connection Control Part Options Management Service*

3.1.6 Error Acknowledgement Service

N_ERROR_ACK:

This primitive indicates to the SCCP-User that a non-fatal error has occured in the last SCCP-User originated request or response primitive (listed in Figure 3.6), on the stream.

Figure 3.6 shows the sequence or primitives for the error management primitive.

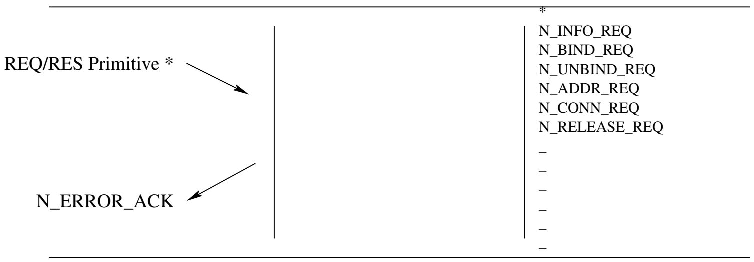

Figure 3.6: *Signalling Connection Control Part Error Acknowledgement Service*

3.2 Connectionless Services

The CLNS allows for the transfer of SCCP-User data in one or both directions simultaneously without establishing an NC between SCCP-User peers. A set of primitives are defined that carry user data and control information between the SCCP-User and SCCP entities. The primitives are modeled as requests initiated by the SCCP-User and indications initiated by the SCCP provider. Indications may be initiated by the SCCP independently from requests by the SCCP-User.

The connectionless SCCP service consists of one phase.

3.2.1 Data Transfer

3.2.1.1 User Primitives for Data Transfer

N_UNITDATA_REQ:
This primitive requests that the SCCP send the data unit to the specified destination with the specified sequence control.

3.2.1.2 Provider Primitives for Data Transfer

N_UNITDATA_IND:
This primitive indicates to the SCCP-User that a data unit has been received from the specified source address.

Figure 3.7 shows the sequence of primitives for the connectionless mode of data transfer.

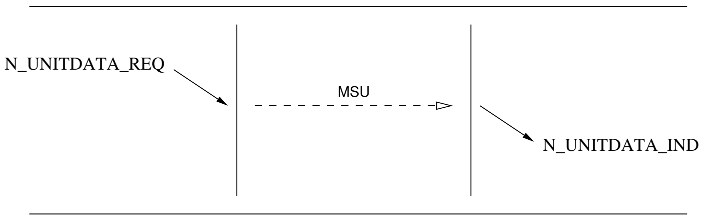

Figure 3.7: *Signalling Connection Control Part Data Transfer*

3.3 Connection Oriented Services

This section describes the required SCCP service primitives that define the CLNS interface.

The queue model for CLNS is discused in more detail in ITU-T Q.711. [Q.711] For Q.711 specific conformance considerations, see Addendum 1.

The queue model represents the operation of an SCCP connection in the abstract by a pair of queues linking the two SCCP addresses. There is one queue for each direction of signalling transfer. The ability of a user to add objects to a queue will be determined by the behavior of the user removing objects from that queue, and the state of the queue. The pair of queues is considered to be available for each potential NC. Objects that are entered or removed from the queue are either as a result of interactions at the two SCCP addresses, or as the result of SCCP initiatives.

- A queue is empty until a connect object has been entered and can be returned to this state, with loss of its contents, by the SCCP.
- Objects may be entered into a queue as a result of the action of the source SCCP-User, subject to control by the SCCP.
- Objects may also be entered into a queue by the SCCP.
- Objects are removed from the queue under the control of the receiving SCCP user.
- Objects are normally removed under the control of the SCCP-User in the same order as they were entered except:
- if the object is of a type defined to be able to advance ahead of the preceding object (however, no object is defined to be able to advance ahead of another object of the same type), or
- if the following object is defined to be destructive with respect to the preceding object on the queue. If necessary, the last object on the queue will be deleted to allow a destructive object to be entered — they will therefore always be added to the queue. For example, "reset" objects are defined to be destructive with respect to all other objects.

Table 3.1 shows the ordering relationship amoung the queue model objects.

Object X Object Y	CONNECT	DATA	MANAGEMENT	DISCONNECT
CONNECT	N/A	–	–	DES
DATA	N/A	–	AA	DES
MANAGEMENT	N/A	–	–	DES
DISCONNECT	N/A	N/A	N/A	–

Table 3.1: *Flow Control Relationships Between Queue Model Objects*

AA Indicates that Object X is defined to be able to advance ahead of preceding Object Y.

DES Indicates that Object X is defined to be destructive with respect to preceding Object Y.

– Indicates that Object X is neither destructive with respect to Object Y, nor able to advance ahead of Object Y.

N/A Indicates that Object X will not occur in a position succeeding Object Y in a valid state of a queue.

3.3.1 Connection Establishment Phase

A pair of queues is associated with an SCCP NC between two SCCP addresses when the SCCP receives an `N_CONN_REQ` primitive at one of the SCCP addresses resulting in a connect object being entered into the queue. The queues will remain associated with the SCCP NC until an `N_DISCON_REQ` primitive (resulting in a disconnect object) is either entered or removed from a queue. Similarly, in the queue from the remote SCCP-User, objects can be entered into the queue only after the connect object associated with an `N_CONN_REQ` has been entered into the queue.

The SCCP NC procedure will fail if the SCCP is unable to route to the remote SCCP-User.

3.3.1.1 User primitives for Successful SCCP Association Establishment

N_CONN_REQ:
This primitive requests that the SCCP establish an NC between the local SCCP-User and the specified destination.

3.3.1.2 Provider primitives for Successful SCCP Association Establishment

N_CONN_CON:
This primitive indicates to the SCCP-User that an NC request has been confirmed.

The sequence of primitives in a successful SCCP NC establishment is defined by the time sequence diagram as shown in Figure 3.8.

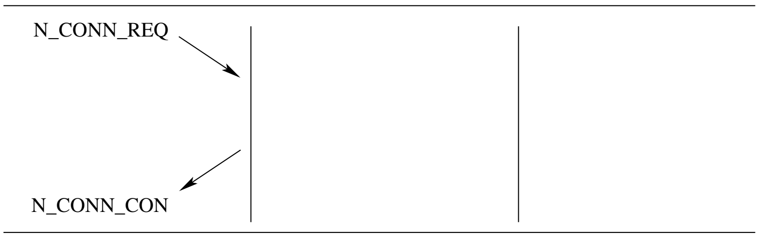

Figure 3.8: *Signalling Connection Control Part Association Service*

3.3.2 Data Transfer Phase

Flow control on the SCCP NC is done by management of queue capacity, by allowing objects of certain type to be inserted to the queues as shown in Table 3.1.

3.3.2.1 User primitives for SCCP Data Transfer

N_DATA_REQ:
This primitive requests that the SCCP transfer the specified data.

3.3.2.2 Provider primitives for SCCP Data Transfer

N_DATA_IND:
This primitive indicates to the SCCP-User that this message contains data.

Figure 3.9 shows the sequence of primitives for successful data transfer. The sequence of primitives may remain incomplete if an N_DISCON_REQ primitive occurs.

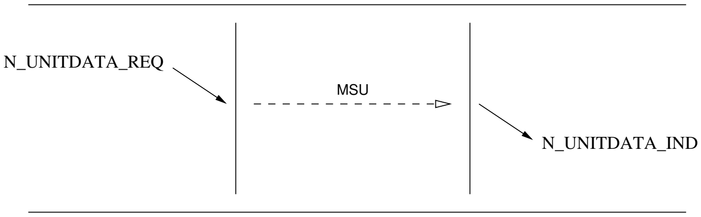

Figure 3.9: *Signalling Connection Control Part Data Transfer*

This sequence of primtives may remain incomplete if an N_RESET_IND or N_RESET_CON indication is received from the SCCP.

3.3.3 Error Management Primitives

The SCCP error management service is used by the SCCP to report detected loss of unrecoverable data.

3.3.3.1 Provider Primitives for Management

N_INFORM_REQ:
N_INFORM_IND:

Figure 3.10 shows the sequence of primitives for the connection mode error management primitives. The sequence of primitives may remain incomplete if an N_DISCON_REQ or N_DISCON_IND primitive occurs.

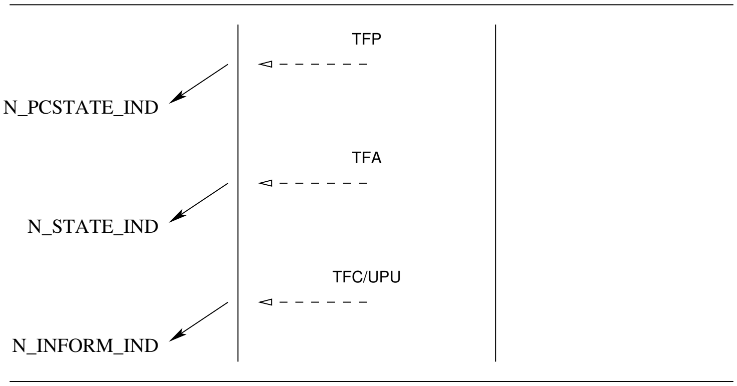

Figure 3.10: *Signalling Connection Control Part Error Management*

3.3.4 Connection Termination Phase

The SCCP NC release procedure is initialized by the insertion of a disconnect object (associated with an `N_DISCON_REQ`) into the queue. As shown if Figure 3.10, the disconnect procedure is destructive with respect to other objects in the queue, and eventually results in the emptying of queues and termination of the SCCP NC.

3.3.4.1 User Primitives for SCCP Association Termination

N_DISCON_REQ:
This primitive requests that the SCCP disconnect an existing SCCP NC.

The sequence of primitives are shown in the time sequence diagram in Figure 3.11.

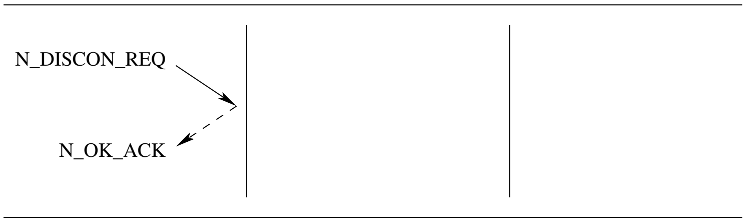

Figure 3.11: *Signalling Connection Control Part Connection Termination*

3.4 SCCP Provider Management Services

This section describes the required SCCP service primitives that define the SCCP Provider Management interface.

SCCP Provider Management allows for the coordination of SCCP management messages between SCCP Provider peers. A set of primitives are defined that invoke management actions that are communicated from SCCP to SCCP entities. The primitives are modeled as requires initiated by the SCCP management and indications initiated by the SCCP. Indications may be initiated by the SCCP independently from requests by the SCCP management.

The SCCP Provider Management service consists of one phase.

3.4.1 Subsystem Management

The SCCP subsystem management service allows SCCP management to allow or prohibit a subsystem or a duplicated subsystem.

3.4.1.1 User Primitives for Subsystem Management Service

N_COORD_REQ:
N_COORD_RES:

3.4.1.2 Provider Primitives for Subsystem Management Service

N_COORD_IND:
N_COORD_CON:

4 SCCPI Primitives

This section describes the format and parameters of the SCCPI primitives (Appendix A [Mapping SCCPI Primitives], page 89, shows the mapping of SCCPI primitives for the primitives defined in Q.711 [Q.711] and T1.112 [T1.112]).

Also, it discusses the states the primitive is valid in, the resulting state, and the acknowledgement that the primitive expects. (The state/event tables for these primitives are shown in Appendix B [State/Event Tables], page 93. The precedence tables for the SCCPI primitives are shown in Appendix C [Precedence Tables], page 95.) Rules for ITU-T conformance [Q.711] are described in addendum to this document in [Addendum for SCCP Conformance], page 85, rules for ANSI conformance [T1.112] are described in addendum [Addendum for SCCP Conformance], page 85, and rules for JITC conformance [JQ.711] are described in addendum [Addendum for SCCP Conformance], page 85.

4.1 Local Management Primitives

These primitives apply to CLNS and CONS.

4.1.1 SCCP Information Request

N_INFO_REQ

This primitive requests the NS provider to return the values of all supported protocol parameters (see Section 4.1.2 [Network Information Acknowledgement], page 23), and also the current state of the NS provider (as defined in Appendix B [State/Event Tables], page 93). This primitive does not affect the state of the network provider and does not appear in the state tables.

Format

This primitive consits of one M_PCPROTO message block, structured as follows:

```
typedef struct {
        np_ulong PRIM_type;             /* always N_INFO_REQ */
} N_info_req_t;
```

Parameters

PRIM_type Specifies the primitive type: always N_INFO_REQ.

Valid States

This primitive in valid in any state where a local acknowledgement is not pending.

New State

The new state remains unchanged.

Acknowledgements

This primitive requires the NS provider to generate one of the following acknowledgements upon receipt of the primitive:

— *Successful:* Acknowledgement of the primitive via the N_INFO_ACK primitive.

— *Unsuccessful (Non-fatal errors):* There are no errors associated with the issuance of this primitive.

4.1.2 Network Information Acknowledgement

N_INFO_ACK

This primitive indicates to the NS user any relevant protocol-dependent parameters.[1] It should be initiated in response to the N_INFO_REQ primitive described above.

Format

This primitive consists of one M_PCPROTO message block, structured as follows:

```
typedef struct {
        np_ulong PRIM_type;             /* always N_INFO_ACK */
        np_ulong NSDU_size;             /* maximum NSDU size */
        np_ulong ENSDU_size;            /* maximum ENSDU size */
        np_ulong CDATA_size;            /* connect data size */
        np_ulong DDATA_size;            /* discon data size */
        np_ulong ADDR_size;             /* address size */
        np_ulong ADDR_length;           /* address length */
        np_ulong ADDR_offset;           /* address offset */
        np_ulong QOS_length;            /* QOS values length */
        np_ulong QOS_offset;            /* QOS values offset */
        np_ulong QOS_range_length;      /* length of QOS values' range */
        np_ulong QOS_range_offset;      /* offset of QOS values' range */
        np_ulong OPTIONS_flags;         /* bit masking for options supported */
        np_ulong NIDU_size;             /* network i/f data unit size */
        np_long SERV_type;              /* service type */
        np_ulong CURRENT_state;         /* current state */
        np_ulong PROVIDER_type;         /* type of NS provider */
        np_ulong NODU_size;             /* optimal NSDU size */
        np_ulong PROTOID_length;        /* length of bound protocol ids */
        np_ulong PROTOID_offset;        /* offset of bound protocol ids */
        np_ulong NPI_version;           /* version # of npi that is supported */
} N_info_ack_t;

/* Flags to indicate support of NS provider options */
#define REC_CONF_OPT    0x00000001L
#define EX_DATA_OPT     0x00000002L
#define DEFAULT_RC_SEL  0x00000004L

/* Service types supported by the NS provider */
#define N_CONS 1
#define N_CLNS 2

/* Valid provider types */
#define N_SNICFP 1
#define N_SUBNET 2
```

Parameters

The above fields have the following meaning:

PRIM_type Specifies the primitive type: always N_INFO_ACK.

NSDU_size Specifies the maximum size (in octets) of a *Network Service Data Unit (NSDU)* supported by the NS provider.

[1] In the future, this primitive will be modified such that it will allow the SCCPI to accept either sub-network point of attachment addresses or network addresses.

ENSDU_size Specifies the maximum size (in octets) of an *Expedited Network Service Data Unit (ENSDU)* supported by the NS provider.

CDATA_size Specifies the maximum number of octets of data that may be associated with connection establishment primitives.

DDATA_size Specifies the maximum number of octets of data that may be associated with the disconnect primitives.

ADDR_size Specifies the maximum size (in decimal digits) of a network address.

ADDR_length Specifies the length in bytes of the network address bound on the Stream on which the `N_INFO_REQ` primitive was issued (a network address is bound to a Stream with the `N_BIND_REQ` primitive).

ADDR_offset Specifies the offset of the bound network address from the beginning of the `M_PCPROTO` message block (this field should be ignored if the *ADDR_length* field is zero).

QOS_length In the connection-mode environment, when this primitive is invoked before the NC is established on the Stream, the values returned specify the default values supported by the NS provider. When this primitive is invoked after a NC has been established on the Stream, the values returned indicate the negotiated values for the QOS parameters. In the connection-less environment, these values represent the default or the selected QOS parameter values. In case a QOS parameter is not supported by NS Provider, a value of `QOS_UNKNOWN` will be returned. In the case where no QOS parameters are supported by the NS provider, this field will be zero.

QOS_offset Indicates the offset of the QOS parameters from the beginning of the `M_PCPROTO` message block.

QOS_range_length Indicates the length in bytes, of the available range of QOS parameters values supported by the NS provider. These ranges are used by the NS user to select QOS parameter values that are valid with the NS provider. QOS parameter values are selected, or the default values altered via the `N_OPTMGMT_REQ` primitive. In the connection-mode environment, the values for end-to-end QOS parameters may be specified with the `N_CONN_REQ` or `N_CONN_RES` primitives for negotiation. If the NS provider does not support a certain QOS parameter, its value will be set to `QOS_UNKNOWN`. In the case where no QOS parameters are supported by the NS provider, the length of this field will be zero.

QOS_range_offset Indicates the offset of the range of QOS parameter values from the beginning of the `M_PCPROTO` message block.

OPTIONS_flags Defines flags that indicate whether the options described below are supported by the NS provider. The possible options are receipt confirmation, expedited data and default selection for use of receipt confirmation.

NIDU_size This indicates the amount of user data that may be present in an `N_DATA_REQ` or `N_DATA_IND` primitive. The *NIDU_size* should not be larger than the *NSDU_size* specification.

SERV_type Indicates the service type supported by the NS provider. The possible values can be `N_CONS`, `N_CLNS`, (or both as indicated by using `N_CONS|N_CLNS`).

CURRENT_state Indicates the current state of the NS provider.

PROVIDER_type Indicates the type of NS provider. The possible values can be `N_SNICFP` or `N_SUBNET`. The value `N_SNICFP` indicates that the provider is the *Subnetwork Independent Convergence Function/Protocol* sub-layer of the network layer. The value `N_SUBNET` indicates that the provider is a subnetwork.

NODU_size Indicates the optimal NSDU size (in octets) of an NSDU given the current routing information.

PROTOID_length Indicates the length of the protocol identifiers that were bound using the `N_BIND_REQ`.

PROTOID_offset Indicates the offset of the protocol identifiers that were bound using the `N_BIND_REQ`, from the beginning of the `M_PCPROTO` message block.

NPI_version Indicates the current version of SCCPI that is supported. Always `N_VERSION_2` for this specificaiton.

Flags

`REC_CONF_OPT`

When set, it indicates that the NS provider supports receipt confirmation.

This flag is used only in the connection-mode environment. Also, this flag is not applicable to Signalling Connection Control Part, because SCCP does not support receipt confirmation.

`EX_DATA_OPT`

When set, it indicates that the NS provider supports expedited data transfer.

This flag is used only in the connection-mode environment. Also, this flag is not applicable to any Signalling Connection Control Part protocol class except Protocol Class 3.

`DEFAULT_RC_SEL`

When set, indicates that the default selection is for the use of receipt confirmation for every `N_DATA_REQ` primitive. This flag is only applicable when use of receipt confirmation is successfully negotiated via the `N_CONN_REQ` or `N_CONN_RES` primitives. *This flag is only used in the connection-mode environment. This flag is not applicable to Signalling Connection Control Part, because SCCP does not support receipt confirmation.*

Service Types

`N_CONS` When set, indicates that the NS provider supports connection-mode network services.

`N_CLNS` When set, indicates that the NS provider supports connection-less network services.

Valid States

This primitive is valid in any state in response to an `N_INFO_REQ` primitive.

New State

The state remains unchanged.

4.1.3 Bind Protocol Address Request

N_BIND_REQ

This primitive requests that the NS provider bind an NS user entity to a network address and negotiate the number of connect indications allowed to be outstanding by the NS provider for the specified NS user entity being bound.

Format

This primitive consists of one `M_PROTO` message block, structured as follows:

```
typedef struct {
        np_ulong PRIM_type;             /* always N_BIND_REQ */
        np_ulong ADDR_length;           /* length of address */
        np_ulong ADDR_offset;           /* offset of address */
        np_ulong CONIND_number;         /* req # of conn-indications to be queued */
        np_ulong BIND_flags;            /* flags associated with N_BIND_REQ */
        np_ulong PROTOID_length;        /* length of the protocol id */
        np_ulong PROTOID_offset;        /* offset of protocol id */
} N_bind_req_t;

/* Flags associated with N_BIND_REQ */
#define DEFAULT_LISTENER    0x00000001L
#define TOKEN_REQUEST       0x00000002L
#define DEFAULT_DEST        0x00000004L
```

Parameters

PRIM_type Specifies the primitive type: always `N_BIND_REQ`.

ADDR_length Specifies the length of the protocol address to bind.

ADDR_offset Specifies the offset of the protocol address to bind from the beginning of the `M_PROTO` message block.

CONIND_number Specifies the requested maximum number of outstanding connection indications to be issued.

This is the requested number of connection indications allowed to be outstanding by the NS provider for the specified protocol address. (If the number of outstanding connect indications equals *CONIND_number*, the NS provider need not discard further incoming connect indications, but may choose to queue them internally until the number of outstanding connect indications drops below the *CONIND_number*.) Only one Stream per network address is allowed to have a *CONIND_number* value greater than zero. This indicates to the network provider that this Stream is the listener Stream for the NS user. This Stream will be used by the NS provider for connect indications for that network address.

If a Stream is bound as a listener Stream, it will not be able to initiate connect requests. If the NS user attempts to send an `N_CONN_REQ` primitive down this Stream, an `N_ERROR_ACK` primitive will be sent to the NS user by the NS provider with an error value of `[NACCESS]`.

This field should be ignored in CLNS.

BIND_flags Specifies the bind option flags associated with the request.

PROTOID_length Specifies the length of protocol identifiers to bind.

PROTOID_offset Specifies the offset of protocol identifiers to bind from the beginning of the `M_PROTO` message block.

Flags

`DEFAULT_LISTENER`
When set, this flag indicates that this Stream is the *Default Listener Stream*. This Stream is used to pass connect indications for all incoming calls that contain protocol identifiers that are not bound to any other listener, or when a listener Stream with *CONIND_number* value of greater than zero is not found. Also, the default listener will receive all incoming call indications that contain no user data.

Only one *Default Listener Stream* is allowed per occurrence of NPI. An attempt to bind a *Default Listener Stream* when one is already bound should result in an error (of type `[NBOUND]`).

The `DEFAULT_LISTENER` *flag is ignored in CLNS.*

`TOKEN_REQUEST`
When set, this flag indicates to the NS provider that the NS user has requested that a *token* be assigned to the Stream (to be used in the NC response message), and the *token value* be returned to the NS user via the `N_BIND_ACK` primitive.

The *token* assigned by the NS provider can then be used by the NS user in a subsequent `N_CONN_RES` primitive to identify the Stream on which the NC is to be established.

The `TOKEN_REQUEST` *flag is ignored in CLNS.*

`DEFAULT_DEST`
When set, this flag indicates that this Stream is the *Default Destination Stream*. This Stream will receive all packets destined for the NSAP specified in the bind request. If no NSAP is indicated in the bind request, then this Stream should receive all packets destined to an NSAP that is bound to no other Stream.

Only one *Default Destination Stream* per NSAP is allowed per occurrence of NPI. An attempt to bind a *Default Destination Stream* to an NSAP when one is already bound should result in an error of type `[NBOUND]`.

The `DEFAULT_DEST` *flag is ignored in the CONS.*

Valid States

This primitive is valid in state `NS_UNBND` (see Appendix B [State/Event Tables], page 93).

New State

The new state is `NS_WACK_BREQ`.

Acknowledgements

The NS provider will generate one of the following acknowledgements upon receipt of the `N_BIND_REQ` primitive:

— *Successful:* Correct acknowledgement of the primitive is indicated using the `N_BIND_ACK` primitive.

— *Unsuccessful (Non-fatal errors):* These errors will be indicated using the `N_ERROR_ACK` primitive. The applicable non-fatal errors are as follows:

`[NBADADDR]` The network address was in an incorrect format or the address contained illegal information. It is not intended to indicate protocol errors.

`[NBOUND]` The NS user attempted to bind a second Stream to a network address with the *CONIND_number* set to a non-zero value, or attempted to bind a second Stream with the `DEFAULT_LISTENER` flag value set to non-zero.

`[NNOADDR]` The NS provider could not allocate an address.

`[NACCESS]` The NS user did not have proper permissions for the use of the requested address.

`[NOUTSTATE]` The primitive was issued from an invalid state.

`[NSYSERR]` A system error has occurred and the *UNIX*® system error is indicated in the primitive.

`[NNOPROTOID]` Protocol identifier could not be allocated.

4.1.4 Bind Protocol Address Acknowledgement

N_BIND_ACK

This primitive indicates to the NS user that the specified network user entity has been bound to the requested network address and that the specified number of connect indications are allowed to be queued by the NS provider for the specified network address.

Format

This primitives consists of one `M_PCPROTO` message block, structured as follows:

```
typedef struct {
    np_ulong PRIM_type;        /* always N_BIND_ACK */
    np_ulong ADDR_length;      /* address length */
    np_ulong ADDR_offset;      /* offset of address */
    np_ulong CONIND_number;    /* connection indications */
    np_ulong TOKEN_value;      /* NC response token value */
    np_ulong PROTOID_length;   /* length of protocol id */
    np_ulong PROTOID_offset;   /* offset from beg. of block */
} N_bind_ack_t;
```

Parameters

PRIM_type Indicates the primitive type: always `N_BIND_ACK`.

ADDR_length Indicates the length of the network address that was bound.

ADDR_offset Indicates the offset of the network address that was bound, from the beginning of the `M_PCPROTO` message block.

CONIND_number Indicates the accepted number of connection indications allowed to be outstanding by the NS provider for the specified network address. If its value is zero, this Stream cannot accept `N_CONN_IND` primitives. If its value is greater than zero, then the NS user can accept `N_CONN_IND` primitives up to the value specified in this parameter before having to respond with an `N_CONN_RES` or an `N_DISCON_REQ` primitive.
This field should be ignored for CLNS.

TOKEN_value Indicates the value of the *token* assigned to this Stream that can be used by the NS user in a `N_CONN_RES` primitive to accept an NC on this Stream. It is a non-zero value, and is unique to all Streams bound to the NS provider.
This field should be ignored for CLNS.

PROTOID_length Indicates the length of the protocol identifiers that were bound.

PROTOID_offset Indicates the offset of the protocol identifiers that were bound, from the beginning of the `M_PCPROTO` message block.

Note that the proper alignment of the address in the `M_PCPROTO` message block is not guaranteed.

Bind Rules:

The following rules apply to the binding of the specified network address to the Stream:

— If the *ADDR_length* field in the `N_BIND_REQ` primitive is zero, then the NS provider is to assign a network address to the user.

— The NS provider is to bind the network address as specified in the `N_BIND_REQ` primitive. If the NS provider cannot bind the specified address, it may assign another network address to the user. It is the network user's responsibility to check the network address returned in the `N_BIND_ACK` primitive to see if it is the same as the one requested.

The following rules apply to negotiating *CONIND_number* argument:

- — The *CONIND_number* in the `N_BIND_ACK` primitive must be less than or equal to the corresponding requested number as indicated in the `N_BIND_REQ` primitive.
- — Only one Stream that is bound to the indicated network address may have a negotiated accepted number of maximum connect requests greater than zero. If a `N_BIND_REQ` primitive specifies a value greater than zero, but another Stream has already bound itself to the given network address with a value greater than zero, the NS provider should assign another protocol address to the user.
- — If a Stream with *CONIND_number* greater than zero is used to accept a connection, the Stream will be found busy during the duration of that connection and no other Streams may be bound to that network address with a *CONIND_number* greater than zero. This will prevent more than one Stream bound to the identical network address from accepting connect indications.
- — A Stream requesting a *CONIND_number* of zero should always be legal. This indicates to the NS provider that the Stream is to be used to request connections only.
- — A Stream with a negotiated *CONIND_number* greater than zero may generate connect requests or accept connect indications.

If the above rules result in an error condition, then the NS provider must issue an `N_ERROR_ACK` *primitive to the NS user specifying the error as defined in the description of the* `N_BIND_REQ` *primitive, see Section 4.1.3 [Bind Protocol Address Request], page 26.*

Valid States

This primitive is valid in response to an `N_BIND_REQ` primitive and is valid in the state `NS_WACK_BREQ` (see Appendix B [State/Event Tables], page 93.)

New State

The new state is `NS_IDLE`.

4.1.5 Unbind Protocol Address Request

N_UNBIND_REQ

This primitive requests that the NS provider unbind the NS user entity that was previously bound to the network address.

Format

This primitives consists of one M_PROTO message block, structured as follows:

```
typedef struct {
        np_ulong PRIM_type;             /* always N_UNBIND_REQ */
} N_unbind_req_t;
```

Parameters

PRIM_type Specifies the primitive type: always N_UNBIND_REQ.

Valid States

This primitive is valid in the NS_IDLE state.

New State

The new state is NS_WACK_UREQ.

Acknowledgements

This primitive requires the NS provider to generate the following acknowledgements upon receipt of the primitive:

— *Successful:* Correct acknowledgement of the primitive is indicated via the N_OK_ACK primitive, see Section 4.1.8 [Successful Receipt Acknowledgement], page 36.

— *Unsuccessful (Non-fatal errors):* These errors will be indicated via the N_ERROR_ACK primitive. The applicable non-fatal errors are as follows:

[NOUTSTATE] The primitive was issued from an invalid state.

[NSYSERR] A system error has occurred and the *UNIX*® system error is indicated in the primitive.

4.1.6 SCCP Options Management Request

N_OPTMGMT_REQ

This primitive allows the NS user to manage QOS parameter values associated with the Stream.

Format

These primitives consists of one M_PROTO message block, structured as follows:

```
typedef struct {
    np_ulong PRIM_type;                         /* always N_OPTMGMT_REQ */
    np_ulong QOS_length;              /* length of QOS parameter values */
    np_ulong QOS_offset;              /* offset of QOS parameter values */
    np_ulong OPTMGMT_flags;           /* options management flags */
} N_optmgmt_req_t;
```

Parameters

PRIM_type Specifies the primitive type: always N_OPTMGMT_REQ.

QOS_length Specifies the length of the Quality of Service parameters. Specifies the length of the default values of the QOS parameters as selected by the NS user. In the connection-mode environment these values will be used in subsequent N_CONN_REQ primitives on the Stream that do not specify values for these QOS parameters. In the connection-less environment, these values represent the selected QOS values that would apply to each unit data transmission. If the NS user cannot determine the value of a QOS parameter, its value should be set to QOS_UNKNOWN. If the NS user does not specify any QOS parameter values, the length of this field should be set to zero.

QOS_offset Specifies the offset of the Quality of Service parameters, from the beginning of the M_PROTO message block.

OPTMGMT_flags Specifies the options management flags associated with the request. (See "Flags" below.)

Flags

DEFAULT_RC_SEL

When set, it indicates to the NS provider that the NS user's default selection is for the use of receipt confirmation with every N_DATA_REQ message (applicable only when its use is successfully negotiated via the N_CONN_REQ or N_CONN_RES primitives). This default indication is used only when the M_PROTO message block is not present in the N_DATA_REQ (i.e. the primitive only contains M_DATA message blocks).

This flag should be ignored in the connection-less environment.

Valid States

This primitive is valid in the NS_IDLE state.

New State

The new state is NS_WACK_OPTREQ.

Acknowledgements

The N_OPTMGMT_REQ primitive requires the NS provider to generate one of the following acknowledgements upon receipt of the primitive:

— *Successful:* Acknowledgement is via the N_OK_ACK primitive. At successful completion, the resulting state is NS_IDLE.

— *Unsuccessful (Non-fatal errors):* These errors are indicated in the N_ERROR_ACK primitive. The resulting state remains unchanged. The applicable non-fatal errors are defined as follows:

[NOUTSTATE] The primitive was issued from an invalid state.

[NBADQOSPARAM] The QOS parameter values specified are outside the range supported by the NS provider.

[NBADQOSTYPE] The QOS structure type is not supported by the NS provider.

[NSYSERR] A system error has occurred and the *UNIX*® system error is indicated in the primitive.

4.1.7 Error Acknowledgement

N_ERROR_ACK

This primitive indicates to the NS user that a non-fatal error has occurred in the last network-user-originated primitive. This may only be initiated as an acknowledgement for those primitives that require one. It also indicates to the user that no action was taken on the primitive that caused the error.

Format

This primitives consists of one `M_PCPROTO` message block, structured as follows:

```
typedef struct {
        np_ulong PRIM_type;             /* always N_ERROR_ACK */
        np_ulong ERROR_prim;            /* primitive in error */
        np_ulong NPI_error;             /* NPI error code */
        np_ulong UNIX_error;            /* UNIX error code */
} N_error_ack_t;
```

Parameters

PRIM_type Indicates the primitive type: always `N_ERROR_ACK`.

ERROR_prim Indicates the primitive type that caused the error.

NPI_error Indicates the Network Provider Interface error code.

UNIX_error Indicates the *UNIX*® system error code. This may only be non-zero when the *NPI_error* is equal to `[NSYSERR]`.

Error Primitives

One of the following error primitive types are allowed to be returned in the ERROR_prim field:

`N_BIND_REQ`	Bind Request.
`N_OPTMGMT_REQ`	Options Management Request.
`N_CONN_REQ`	Connect Request.
`N_CONN_RES`	Connect Response.
`N_RESET_REQ`	Reset Request.
`N_RESET_RES`	Reset Response.
`N_DISCON_REQ`	Disconnect Request.
`N_UNBIND_REQ`	Unbind Request.
`N_INFORM_REQ`	Inform Request.
`N_STATE_REQ`	State Request (SCCPI only).
`N_COORD_REQ`	Coordination Request (SCCPI only).
`N_COORD_RES`	Coordination Response (SCCPI only).

Also, any unrecognized primitive type may also be returned in conjunction with the `[NNOTSUPPORT]` *error code.*

Valid Error Codes

The following error codes are allowed to be returned in the NPI_error field:

`[NBADADDR]` The network address as specified in the primitive was in an incorrect format, or the address contained illegal information.

`[NBADOPT]` The options values as specified in the primitive were in an incorrect format, or they contained illegal information.

`[NBADQOSPARAM]` The QOS values specified are outside the range supported by the NS provider.

`[NBADQOSTYPE]` The QOS structure type is not supported by the NS provider.

`[NBADTOKEN]` Token used is not associated with an open Stream.

`[NNOADDR]` The NS provider could not allocate an address.

`[NACCESS]` The user did not have proper permissions.

`[NOUTSTATE]` The primitive was issued from an invalid state.

`[NBADSEQ]` The sequence number specified in the primitive was incorrect or illegal.

`[NBADFLAG]` The flags specified in the primitive were incorrect or illegal.

`[NBADDATA]` The amount of user data specified was outside the range supported by the NS provider.

`[NSYSERR]` A system error has occurred and the *UNIX*® system error is indicated in the primitive.

`[NNOTSUPPORT]` Specified primitive type is not known to the NS provider.

Valid States

This primitive is valid in all states that have a pending acknowledgement or confirmation.

New State

The new state is the same as the one from which the acknowledged request or response was issued.

4.1.8 Successful Receipt Acknowledgement

N_OK_ACK

This primitive indicates to the NS user that the previous network-user-originated primitive was received successfully by the network provider. It does not indicate to the NS user any network protocol action taken due to the issuance of the last primitive. The `N_OK_ACK` primitive may only be initiated as an acknowledgement for those user originated primitives that have no other means of confirmation.

Format

This primitives consists of one `M_PCPROTO` message block, structured as follows:

```
typedef struct {
        np_ulong PRIM_type;             /* always N_OK_ACK */
        np_ulong CORRECT_prim;          /* primitive being acknowledged */
} N_ok_ack_t;
```

Parameters

PRIM_type	Indicates the primitive type: always `N_OK_ACK`.
CORRECT_prim	Indicates the successfully received primitive type.

Correct Primitives

`N_OPTMGMT_REQ`	Options Management Request.
`N_CONN_RES`	Connection Response.
`N_RESET_RES`	Reset Response.
`N_DISCON_REQ`	Disconnect Request.
`N_UNBIND_REQ`	Unbind Request.
`N_COORD_RES`	Coordination Response (SCCPI only).
`N_INFORM_REQ`	Inform Request (SCCPI only).

Valid States

This primitive is issued in the following states:

`NS_WACK_UREQ`	Wait for acknowledgement of Unbind Request.
`NS_WACK_OPTREQ`	Wait for acknowledgement of Options Management Request.
`NS_WACK_RRES`	Wait for acknowledgement of Reset Response.
`NS_WACK_CRES`	Wait for acknowledgement of Connection Response.
`NS_WACK_DREQ6`	Wait for acknowledgement of Disconnect Request.
`NS_WACK_DREQ7`	Wait for acknowledgement of Disconnect Request.
`NS_WACK_DREQ9`	Wait for acknowledgement of Disconnect Request.
`NS_WACK_DREQ10`	Wait for acknowledgement of Disconnect Request.
`NS_WACK_DREQ11`	Wait for acknowledgement of Disconnect Request.

New State

The resulting state depends on the current state (see ⟨undefined⟩ [⟨undefined⟩], page ⟨undefined⟩, and ⟨undefined⟩ [⟨undefined⟩], page ⟨undefined⟩).

4.2 Connection Mode Primitives

This section describes the format of the CONS primitives and the rules associated with these primitives. The default values of the QOS parameters associated with an NC may be selected via the N_OPTMGMT_REQ primitive.

4.2.1 Connection Establishment Phase

The following SCCP service primitives pertain to the establishment of an NC between local and remote SCCP-SAPs, provided the SCCP users exist, and are known to the SCCP.

4.2.1.1 Signalling Connection Control Part Connection Request

N_CONN_REQ

This primitive requests that the SCCP form an NC to the specified destination.

Format

The format of the message is one M_PROTO message block followed by one or more M_DATA message blocks for the NS user data transfer. The specification of the NS user data is optional. The NS user can send any integral number of octets of data within the range supported by the NS provider (see Section 4.1.2 [Network Information Acknowledgement], page 23). If the user does not specify QOS parameter values, the default values (specified via N_OPTMGMT_REQ) are used by the NS provider.

The structure of the M_PROTO message block is as follows:

```
typedef struct {
        np_ulong PRIM_type;             /* always N_CONN_REQ */
        np_ulong DEST_length;           /* destination address length */
        np_ulong DEST_offset;           /* destination address offset */
        np_ulong CONN_flags;            /* bit masking for options flags */
        np_ulong QOS_length;            /* length of QOS parameter values */
        np_ulong QOS_offset;            /* offset of QOS parameter values */
} N_conn_req_t;

/* Flags to indicate if options are requested */
#define REC_CONF_OPT    0x00000001L
#define EX_DATA_OPT     0x00000002L
```

Parameters

PRIM_type Specifies the primitive type: always N_CONN_REQ.

DEST_length Specifies the length of the destination address to which to connect. Identifies the NS user to which the NC is to be established. This field will accommodate variable length addresses within a range supported by the NS provider.

DEST_offset Specifies the offset of the destination address to which to connect, from the beginning of the M_PROTO message block.

CONN_flags Specifies the connection options flags. (See "Flags" below.)

QOS_length Specifies the length of the Quality of Service parameters negotiated. Indicates the QOS parameter values that apply to the NC being requested. If the NS user cannot determine the value of a QOS parameter, its value should be set to QOS_UNKNOWN. If the NS user does not specify any QOS parameter values, the length of this field should be set to zero ('0').

QOS_offset Specifies the offset of the Quality of Service parameters negotiated, from the beginning of the M_PROTO message block.

Flags

REC_CONF_OPT

The receipt confirmation selection parameter indicates the use/availability of the receipt confirmation service on the NC. The receipt confirmation service must be supported by the NS provider to be used on the NC.

EX_DATA_OPT

Specifies the use of the expedited data transfer service on the NC. The expedited data transfer service must be provided by the NS provider for it to be used on the NC.

Valid States

This primitive is valid in state NS_IDLE.

New State

The new state is NS_WCON_CREQ.

Acknowledgements

The following acknowledgements are valid for this primitive:

— *Successful NC Establishment:* This is indicated using the N_CONN_CON primitive. This results in the NS_DATA_XFER state.

— *Unsuccessful NC Establishment:* This is indicated using the N_DISCON_IND primitive. For example, a connection may be rejected because either the called NS user cannot be reached, or the NS provider and/or the called NS user did not agree with the specified QOS. This results in the NS_IDLE state.

— *Unsuccessful (Non-fatal errors):* These are indicated using the N_ERROR_ACK primitive. The applicable non-fatal errors are defined as follows:

[NACCESS]	The user did not have proper permission for the user of the requested address or options.
[NBADQOSPARAM]	The QOS parameter values specified are outside the range supported by the NS provider.
[NBADQOSTYPE]	The QOS structure type is not supported by the NS provider.
[NBADADDR]	The network address was in an incorrect format or contained illegal information. It is not intended to indicate NC errors, such as an unreachable destination. These error types are included using the N_DISCON_IND primitive.
[NBADOPT]	The options were in an incorrect format, or they contain illegal information.
[NOUTSTATE]	The primitive was issued from an invalid state.
[NBADDATA]	The amount of user data specified was outside the range supported by the NS provider.
[NSYSERR]	A system error occurred and the *UNIX*® system error is indicated in the primitive.

4.2.1.2 Signalling Connection Control Part Connection Indication

N_CONN_IND

This primitive indicates to the destination NS user that a network connect request has been made by the user at the specified source address.

Format

The format of this message is one `M_PROTO` message block followed by one or more `M_DATA` message blocks for NS user data. The specification of NS user data is optional. The NS user can send any integral number of octets of data within the range supported by the NS provider. The NS user data will only be present if the corresponding `N_CONN_RES` had an NS user data parameter specified, and their data will be identical.

The structure of the `M_PROTO` message block is as follows:

```
typedef struct {
        np_ulong PRIM_type;             /* always N_CONN_IND */
        np_ulong DEST_length;           /* destination address length */
        np_ulong DEST_offset;           /* destination address offset */
        np_ulong SRC_length;            /* source address length */
        np_ulong SRC_offset;            /* source address offset */
        np_ulong SEQ_number;            /* sequence number */
        np_ulong CONN_flags;            /* bit masking for options flags */
        np_ulong QOS_length;            /* length of QOS parameter values */
        np_ulong QOS_offset;            /* offset of QOS parameter values */
} N_conn_ind_t;
```

Parameters

PRIM_type Indicates the primitive type: always `N_CONN_IND`.

DEST_length Indicates the length of the destination address. This is the an address identifying the NS user to which the NC is to be established.

DEST_offset Indicates the offset of the destination address, from the beginning of the `M_PROTO` message block.

SRC_length Indicates the length of the source address. The source address is the network address of the NS user from which the NC has been requested. The semantics of the value in the `N_CONN_IND` primitive is identical to the value associated with the Stream on which the `N_CONN_REQ` was issued.

SRC_offset Indicates the offset of the source address from the beginning of the `M_PROTO` message block.

SEQ_number Indicates the sequence number that can be used by the NS user to associate this message with the `N_CONN_RES` or `N_DISCON_REQ` primitive that is to follow. This value must be unique among the outstanding `N_CONN_IND` messages. The use of this field allows the NS user to issue the `N_CONN_RES` or the `N_DISCON_REQ` messages in any order.

CONN_flags Indicates the connection options flags associated with the indication.

QOS_length Indicates the length of the Quality of Service parameters. This is the QOS values that are negotiated during NC establishment. If the destination NS user does not agree to the range of QOS values specified by the source NS user in the N_CONN_REQ primitive, it will reject the NC establishment by invoking a N_DISCON_REQ primitive (the originator parameter in the N_DISCON_REQ primitive will indicate NS user initiated release). If the NS user does not support or cannot determine the value of a QOS parameter, its value will be set to QOS_UNKNOWN. If the NS user does not specify any QOS parameter values, the length of this field should be set to zero.

QOS_offset Indicates the offset of the Quality of Service parameters, from the beginning of the M_PROTO message block.

Flags

REC_CONF_OPT

The receipt confirmation selection parameter indicates the use/availability of the receipt confirmation service on the NC. The receipt confirmation service must be provided in the network service to be used on the NC.

EX_DATA_OPT

The expedited data selection parameter indicates the use/availability of the expedited data transfer service on the NC. The expedited data transfer service must be provided by the NS provider for it to be used on the NC.

Valid States

This primitive is valid in the states NS_IDLE and NS_WRES_CIND.

New State

In both cases the resulting state is NS_WRES_CIND (the number of connect indications waiting for user response is incremented by one).

4.2.1.3 Signalling Connection Control Part Connection Response

N_CONN_RES

This primitive allows the destination NS user to request that the network provider accept a previous connect request.

Format

The format of this primitive is one `M_PROTO` message block followed by one or more `M_DATA` message blocks (for NS user data). The specification of the NS user data is optional.

The NS user can send any integral number of octets of data within the range supported by the NS provider.

The structure of the `M_PROTO` block is as follows:

```
typedef struct {
        np_ulong PRIM_type;             /* always N_CONN_RES */
        np_ulong TOKEN_value;           /* NC response token value */
        np_ulong RES_length;            /* responding address length */
        np_ulong RES_offset;            /* responding address offset */
        np_ulong SEQ_number;            /* sequence number */
        np_ulong CONN_flags;            /* bit masking for options flags */
        np_ulong QOS_length;            /* length of QOS parameter values */
        np_ulong QOS_offset;            /* offset of QOS parameter values */
} N_conn_res_t;
```

Parameters

PRIM_type Specifies the primitive type: always `N_CONN_RES`.

TOKEN_value Specifies the response token value of the Stream upon which the connection is to be accepted, or zero, if the connection is to be accepted on the issuing Stream. This value is used to identify the Stream that the NS user want to establish the NC on. (Its value is determined by the NS user by issuing a `N_BIND_REQ` primitive with the `TOKEN_REQUEST` flag set. The token value is returned in the `N_BIND_ACK`). The value of this field should be non-zero when the NS user wants to establish the NC on a Stream other than the Stream on which the `N_CONN_IND` arrived. If the NS user wants to establish a NC on the same Stream that the `N_CONN_IND` arrived on, then the value of this field should be zero (‘`0`’).

RES_length Specifies the length of the responding address. This field conveys the network address of the NS user to which the NC has been established. Under certain circumstances, such as call redirection, generic addressing, etc., the value of this parameter may be different from the destination address parameter specified in the corresponding `N_CONN_REQ`.

RES_offset Specifies the offset of the responding address from the beginning of the `M_PROTO` message block.

SEQ_number Specifies the sequence number of the corresponding connection indication to which this primitive is responding. This is the sequence number of the `N_CONN_RES` primitive. It is used by the NS provider to associate the `N_CONN_RES` message with an outstanding `N_CONN_IND` message. An invalid sequence number should result in an `N_ERROR_ACK` primitive with the error type `[NBADSEQ]`.

CONN_flags Specifies the connection options flags associated with the connection response. (See “Flags” below.)

QOS_length Specifies the length of the Quality of Service parameters. This is the QOS parameter values that are negotiated during NC establishment by invoking a N_DISCON_REQ primitive (the originator parameter in the *N_DISCON_REQ* primitive will indicate NS user invoked release). If the NS user cannot determine the value of a QOS parameter, its value should be set to QOS_UNKNOWN. If the NS user does not specify any QOS parameter values, the length of this field should be set to zero (‘0’).

QOS_offset Specifies the offset of the Quality of Service parameters from the beginning of the M_PROTO message block.

Flags

REC_CONF_OPT

The receipt confirmation selection parameter indicates the use/availability of the receipt confirmation service on the NC. The receipt confirmation service must be provided in the network service to be used on the NC. *SCCP does not provide receipt confirmation so this flag will not be indicated and will be ignored when specified. Alternately, when specified the NS provider may retun an* N_ERROR_ACK *primitive with error type* [NBADOPT]*.*

EX_DATA_OPT

The expedited data selection parameter indicates the use/availability of the expedited data transfer service on the NC. The expedited data transfer service must be provided by the NS provider for it to be used on the NC. *SCCP only provides for expedited data on Protocol Class 3 connections. This flag will only be indicated and can only be accepted when Protocol Class 3 operation is supported.*

Valid States

This primitive is valid in state NS_WRES_CIND.

New State

The new state is NS_WACK_CRES.

Acknowledgements

The NS provider should generate one of the following acknowledgements upon receipt of this primitive:

- *Successful:* Successful completion is indicated via the N_OK_ACK primitive. The final state will be NS_DATA_XFER for the accepting Stream and NS_IDLE or NS_WRES_CIND for the listening Stream when the listening Stream is different than the accepting Stream and depending upon whether there are additional outstanding connection indications.
- *Unsuccessful (Non-fatal errors):* Errors are indicated with the N_ERROR_ACK primitive. The applicable non-fatal errors are defined as follows:

  [NBADOPT] The options were in an incorrect format, or they contained illegal information.

  [NBADQOSPARAM] The QOS parameter values specified are outside the range supported by the NS provider.

[NBADQOSTYPE] The QOS structure type is not supported by the NS provider.

[NBADTOKEN] The token specified is not associated with an open Stream.

[NACCESS] The user did not have proper permissions for the use of the options or the token or response identifier.

[NOUTSTATE] The primitive was issued from an invalid state.

[NBADDATA] The amount of user data specified was outside the range supported by the NS provider.

[NBADSEQ] The sequence number specified in the primitive was incorrect or illegal.

[NSYSERR] A system error has occurred and the *UNIX*® system error is indicated in the primitive.

4.2.1.4 Signalling Connection Control Part Connection Confirmation

N_CONN_CON

This primitive indicates to the source NS user that the network connect request has been confirmed on the specified responding address.

Format

The format of the N_CONN_CON primitive is one M_PROTO message block followed by one or more M_DATA message blocks (for NS user data). The specification of the NS user data is optional.

The NS user can send any integral number of octets of NS user data within a range supported by the NS provider (see Section 4.1.2 [Network Information Acknowledgement], page 23). The NS user data will only be present if the corresponding N_CONN_RES had NS user data specified with it, and their data will always be identical.

The structure of the M_PROTO message block is as follows:

```
typedef struct {
        np_ulong PRIM_type;            /* always N_CONN_CON */
        np_ulong RES_length;           /* responding address length */
        np_ulong RES_offset;           /* responding address offset */
        np_ulong CONN_flags;           /* bit masking for options flags */
        np_ulong QOS_length;           /* length of QOS parameter values */
        np_ulong QOS_offset;           /* offset of QOS parameter values */
} N_conn_con_t;
```

Parameters

PRIM_type Indicates the primitive type: always N_CONN_CON.

RES_length Indicates the length of the responding address. This field conveys the network address of the NS user entity to which the NC has been established. The semantics of the values in the N_CONN_CON is identical to the values in N_CONN_RES. Under certain circumstances, such as call redirection, generic addressing, etc., the value of this parameter may be different from the destination address parameter specification in the corresponding N_CONN_REQ.

RES_offset Indicates the offset of the responding address from the beginning of the M_PROTO message block.

CONN_flags Indicates the connect options flags associated with the connect confirmation. (See “Flags” below.)

QOS_length Indicates the length of the Quality of Service parameters. This field conveys the QOS parameter values selected by the responding NS user. If the NS provider does not support or cannot determine the selected value of the QOS parameter, its value will be set to QOS_UNKNOWN. If the NS provider does not specify any QOS parameter values, the length of this field should be set to zero (‘0’).

QOS_offset Indicates the offset of the Quality of Service parameters from the beginning of the M_PROTO message block.

Flags

REC_CONF_OPT

The receipt confirmation selection parameter indicates the use/availability of the receipt confirmation service on the NC. The receipt confirmation service must be provided in the network service to be used on the NC.

EX_DATA_OPT

The expedited data selection parameter indicates the use/availability of the expedited data transfer service on the NC. The expedited data transfer service must be provided by the NS provider for it to be used on the NC.

Valid States

This primitive is valid in state NS_WCON_CREQ.

New State

The new state is NS_DATA_XFER.

4.2.2 Normal Data Transfer Phase

The data transfer service primitives provide for an exchange of NS user data known as NSDUs, in either direction or in both directions simultaneously on an NC. The network service preserves both the sequence and the boundaries of the NSDUs (when the NS provider supports NSDUs).

4.2.2.1 Normal Data Transfer Request

N_DATA_REQ

This user-originated primitive specifies to the NS provider that this message contains NS user data. It allows the transfer of NS user data between NS users without modification by the NS provider. The NS user must send any integral number of octets of data greater than zero. In a case where the size of the NSDU exceeds the NIDU (as specified by the size of the *NIDU_size* parameter of the N_INFO_ACK primitive), the NSDU may be broken up into more than one NIDU. When an NSDU is broken up into more than one NIDU, the N_MORE_DATA_FLAG will be set on each NIDU except the last one. The N_RC_FLAG may only be set on the last NIDU.

Format

The format of the message is one or more M_DATA message blocks. Use of a M_PROTO message block is optional. The M_PROTO message block is used for two reasons:

1. to indicate that the NSDU is broken into more than one NIDUs, and that the data carried in the following M_DATA message block constitutes one NIDU;
2. to indicate whether receipt confirmation is desired for the NSDU.

Guidelines for use of M_PROTO:

The following guidelines must be followed with respect to the use of the M_PROTO message block:

1. The M_PROTO message block need not be present when the NSDU size is less than or equal to the NIDU size and one of the following is true:
   — receipt confirmation has been negotiated for non-user (with the N_CONN_REQ and N_CONN_RES primitives); or
   — receipt confirmation has been successfully negotiated for use or non-use and the default selection as specified with the N_OPTMGMT_REQ primitive is to be used.
2. The M_PROTO message block must be present when:
   — the NSDU size is greater than the NIDU size;
   — receipt confirmation has been successfully negotiated for use and the default selection as specified with the N_OPTMGMT_REQ primitive needs to be overridden.

The structure of the M_PROTO message block, if present, is as follows:

```
typedef struct {
        np_ulong PRIM_type;             /* always N_DATA_REQ */
        np_ulong DATA_xfer_flags;       /* data transfer flags */
} N_data_req_t;

/* Data Transfer Flags */
#define N_MORE_DATA_FLAG    0x00000001L
#define N_RC_FLAG           0x00000002L
```

Parameters

PRIM_type	Specifies the primitive type: always N_DATA_REQ.
DATA_xfer_flags	Specifies the data transfer flags associated with the data. (See “Flags” below.)

Flags

N_MORE_DATA_FLAG
: When set, this flag indicates that the next N_DATA_REQ primitive (NISDU) is also part of this NSDU.

N_RC_FLAG
: By setting this flag on the N_DATA_REQ, the originating NS user can request confirmation of receipt of the N_DATA_REQ primitive. The receipt is provided by the N_DATACK_IND primitive. The parameter may only be present if use of Receipt Confirmation was agreed by both NS users and the NS provider during NC establishment.

Valid States

This primitive is valid in the NS_DATA_XFER state.

New State

The resulting state remains the same (NS_DATA_XFER).

Acknowledgements

This primitive does not require any acknowledgements, although it may generate a fatal error. This is indicated to the NS user with a M_ERROR STREAMS message type (specifying an error number value of [EPROTO]) that results in the failure of all system calls on that Stream. The applicable errors are defined as follows:

[EPROTO]
: This indicates one of the following unrecoverable protocol conditions:

— The network interface was found to be in an incorrect state.

— The amount of NS user data associated with the primitive is outside the range supported by the NS provider (as specified by the *NIDU_size* parameter of the N_INFO_ACK primitive).

— The options requested are either not supported by the NS provider or its use not specified with the N_CONN_REQ primitive.

— The M_PROTO message block was not followed by one or more M_DATA message blocks.

— The amount of NS user data associated with the current NSDU is outside the range supported by the NS provider (as specified by the *NSDU_size* parameter in the N_INFO_ACK primitive.)

— The N_RC_FLAG and N_MORE_DATA_FLAG were both set in the primitive, or the flags field contained an unknown value.

NOTE: If the interface is in the NS_IDLE or NS_WRES_RIND states when the provider receives the N_DATA_REQ primitive, then the NS provider should discard the request without generating a fatal error.

4.2.2.2 Normal Data Transfer Indication

N_DATA_IND

This network-provider-originated primitive indicates to the NS user that this message contains NS user data. As in the N_DATA_REQ primitive, the NSDU can be segmented into more than one NIDUs. The NIDUs are associated with the NSDU by using the N_MORE_DATA_FLAG. The N_RC_FLAG is allowed to be set only on the last NIDU.

Format

The format of the message is one or more M_DATA message blocks. The value of the NS user data field is always the same as that supplied in the corresponding N_DATA_REQ primitive at the peer service access point. Use of M_PROTO message blocks is optional (see guidelines under see Section 4.2.2.1 [Normal Data Transfer Request], page 46).

The structure of the M_PROTO message block, if present, is as follows:

```
typedef struct {
        np_ulong PRIM_type;             /* always N_DATA_IND */
        np_ulong DATA_xfer_flags;       /* data transfer flags */
} N_data_ind_t;

/* Data Transfer Flags */
#define N_MORE_DATA_FLAG    0x00000001L
#define N_RC_FLAG           0x00000002L
```

Parameters

PRIM_type Indicates the primitive type: always N_DATA_IND.

DATA_xfer_flags Indicates the data transfer flags associated with the data. (See “Flags” below.)

Flags

N_MORE_DATA_FLAG
When set, indicates that the next N_DATA_IND message (NIDU) is part of this NSDU.

N_RC_FLAG The value of the parameter may indicate either that confirmation is requested or that it is not requested. The parameter is allowed to be set only if use of Receipt Confirmation was agreed to between both the NS users and the NS provider during NC establishment. The value of this parameter is always identical to that supplied in the corresponding N_DATA_REQ primitive.

Valid States

This primitive is valid in state NS_DATA_XFER.

New State

The resulting state remains the same (NS_DATA_XFER).

4.2.3 Receipt Confirmation Service Primitives

The receipt confirmation service is requested by the confirmation request parameter on the N_DATA_REQ primitive. For each and every NSDU with the confirmation request parameter set, the receiving NS user should return an N_DATACK_REQ primitive. Such acknowledgements should be issued in the same sequence as the corresponding N_DATA_IND primitives are received, and are to be conveyed by the NS provider in such a way so as to preserve them distinct from any previous or subsequent acknowledgements. The NS user may thus correlate them with the original requests by counting. When an NSDU has been segmented into more than one NIDUs, only the last NIDU is allowed to request receipt confirmation. N_DATACK_REQ primitive will not be subject to the flow control affecting N_DATA_REQ primitives at the same NC endpoint. N_DATACK_IND primitives will not be subject to the flow control affecting N_DATA_IND primitives at the same NC endpoint.

The use of the receipt confirmation service must be agreed to by the two NS users of the NC and the NS provider during the NC establishment by using the DEFAULT_RC parameter on the N_CONN_REQ or N_CONN_RES primitive.

4.2.3.1 Data Acknowledgement Request

N_DATACK_REQ

This is a user-originated primitive that requests that the network provider acknowledge the N_DATA_IND that had previously been received with the receipt confirmation parameter set.

Format

The format of the primitive is one M_PROTO message block, structured as follows:

```
typedef struct {
        np_ulong PRIM_type;             /* always N_DATACK_REQ */
} N_datack_req_t;
```

Parameters

PRIM_type Indicates the primitive type: always N_DATACK_REQ.

Valid States

This primitive is valid in state NS_DATA_XFER.

New State

The resulting state remains the same (NS_DATA_XFER).

Acknowledgements

This primitive does not require any acknowledgements, although it may generate a fatal (unrecoverable) error. This is indicated via an M_ERROR STREAMS message type (issued to the NS user specifying the error number value of [EPROTO]), which results in the failure of all system calls on that Stream. The allowable errors are as follows:

[EPROTO] This indicates the following unrecoverable protocol condition:

— The network interface was found to be in an incorrect state.

NOTE: If the interface is in the NS_IDLE state when the provider receives the N_DATACK_REQ primitive, then the NS provider should discard the request without generating a fatal error. If the NS provider had no knowledge of a previous `N_DATA_IND` with the receipt confirmation flag set, then the NS provider should just ignore the request without generating a fatal error.

4.2.3.2 Data Acknowledgement Indication

N_DATACK_IND

This is a NS provider originated primitive that indicates to the network service user that the remote network service user has acknowledged the data that had previously been sent with the receipt confirmation set.

Format

The format of the primitive is one M_PROTO message block, structured as follows:

```
typedef struct {
        np_ulong PRIM_type;             /* always N_DATACK_IND */
} N_datack_ind_t;
```

Parameters

PRIM_type Indicates the primitive type: always N_DATACK_IND.

Valid States

This primitive is valid in state NS_DATA_XFER.

New State

The resulting state remains the same (NS_DATA_XFER).

4.2.4 Expedited Data Transfer Service

The expedited data transfer service provides a further means of information exchange on an NC in both directions simultaneously. The transfer of expedited network service data unit (ENSDU) is subject to separate flow control from that applying to NS user data. (However, a separate STREAMS message type for expedited data is not available with *UNIX® System V Release 3.1*. Until a new STREAMS message type is provided, expedited data will be implemented via queue manipulation). The NS provider should guarantee that an expedited-NSDU will not be delivered after any subsequently issued NSDU or expedited-NSDU on that NC. The relationship between normal and expedited data is shown in Table 2.2. Expedited data can still be delivered when the receiving NS user is not accepting normal data (however this cannot be guaranteed if there are blockages occurring in the lower layers). The expedited data transfer service is a NS provider option, and its use must be agreed by the two NS users of the NC and the NS provider during NC establishment by using the `EX_DATA_OPT` parameter on the `N_CONN_REQ` and `N_CONN_RES` primitives.

4.2.4.1 Expedited Data Transfer Request

N_EXDATA_REQ

This is an NS user originated primitive and is used to indicate to the network provider that the message block contains an ENSDU.

Format

The format of the message is one `M_PROTO` message block, followed by one or more `M_DATA` message blocks. The NS user must send an integral number of octets of data within the range supported by the NS provider (see Section 4.1.2 [Network Information Acknowledgement], page 23).

The structure of the `M_PROTO` message block is as follows:

```
typedef struct {
        np_ulong PRIM_type;             /* always N_EXDATA_REQ */
} N_exdata_req_t;
```

Parameters

PRIM_type Specifies the primitive type: always `N_EXDATA_REQ`.

Valid States

This primitive is valid in state `NS_DATA_XFER`.

New State

The resulting state remains the same (`NS_DATA_XFER`).

Acknowledgements

This primitive does not require any acknowledgements, although it may generate a fatal (unrecoverable) error. This is indicated with an `M_ERROR` STREAMS message type (issued to the NS user with the error number value of `[EPROTO]`), which results in the failure of all system calls on that Stream. The applicable errors are as follows:

`[EPROTO]` This indicates one of the following unrecoverable protocol conditions:

- — The network interface was found to be in an incorrect state.
- — The amount of NS user data associated with the primitive defines an expedited network service data unit of a size that is outside the range supported by the NS provider.
- — Expedited data transfer is either not supported by the NS provider or not requested with the `N_CONN_REQ` primitive.

NOTE: If the interface is in the `NS_IDLE` or `NS_WRES_RIND` states when the provider receives the `N_EXDATA_REQ` primitive, then the NS provider should discard the request without generating a fatal error.

4.2.4.2 Expedited Data Transfer Indication

N_EXDATA_IND

This is a NS provider originated primitive and is used to indicate to the NS user that this message contains an ENSDU.

Format

The format of the message is one M_PROTO message block, followed by one or more M_DATA message blocks. The value of the data in the M_DATA message blocks is identical to that supplied with the corresponding N_EXDATA_REQ primitive.

The structure of the M_PROTO message block is as follows:

```
typedef struct {
        np_ulong PRIM_type;             /* always N_EXDATA_IND */
} N_exdata_ind_t;
```

Parameters

PRIM_type Indicates the primitive type: always N_EXDATA_IND.

Valid States

This primitive is valid in the state NS_DATA_XFER.

New State

The resulting state remains the same (NS_DATA_XFER).

4.2.5 Reset Service

The reset service can be used by the NS user to resynchronize the use of the NC; or by the NS provider to report detected loss of data unrecoverable within the network service.

All loss of data that does not involve loss of the NC is reported in this way. Invocation of the reset service will unblock the flow of NSDUs and ENSDUs in case of congestion of the NC; it will cause the NS provider to discard NSDUs, ENSDUs, or confirmations of receipt associated with the NC (see Table 2.1), and to notify any NS user or users that did not invoke reset that a reset has occurred. The service will be completed in finite time irrespective of the acceptance of the NSDUs, ENSDUs, and confirmations of receipt by the NS users.

4.2.5.1 Reset Request

N_RESET_REQ

This user-originated primitive requests that the NS provider reset the network connection.

Format

The format of this primitive is one `M_PROTO` message block, structured as follows:

```
typedef struct {
        np_ulong PRIM_type;            /* always N_RESET_REQ */
        np_ulong RESET_reason;         /* reason for reset */
} N_reset_req_t;
```

Parameters

PRIM_type Specifies the primitive type: always `N_RESET_REQ`.
RESET_reason Specifies the reason for the reset. (See “Reasons” below.)

Valid States

This primitive is valid in the `NS_DATA_XFER` state.

New State

The resulting state is `NS_WACK_RREQ`.

Acknowledgements

The NS provider should generate one of the following acknowledgements upon receipt of this primitive:

— *Successful:* This primitive does not require an immediate acknowledgement, although when the resynchronization completes successfully, an `N_RESET_CON` primitive is issued to the NS user that issued the `N_RESET_REQ`.

— *Unsuccessful (Non-fatal errors):* A non-fatal error is acknowledged with the `N_ERROR_ACK` primitive. In this case the resulting state remains unchanged. The following non-fatal error codes are valid:

[NOUTSTATE] The primitive was issued from an invalid state.
[NSYSERR] A system error has occurred and the *UNIX*® system error is indicated with the `N_ERROR_ACK` primitive.

NOTE: If the interface is in the `NS_IDLE` state when the provider receives the `N_RESET_REQ` primitive, then the NS provider should discard the message without generating an error.

4.2.5.2 Reset Indication

N_RESET_IND

This network-provider-originated primitive indicates to the NS user that the network connection has been reset.

Format

The format of the message is one `M_PROTO` message block, structured as follows:

```
typedef struct {
        np_ulong PRIM_type;             /* always N_RESET_IND */
        np_ulong RESET_orig;            /* reset originator */
        np_ulong RESET_reason;          /* reason for reset */
} N_reset_ind_t;
```

Parameters

PRIM_type Indicates the primitive type: always `N_RESET_IND`.

RESET_orig Indicates the source of the reset. (See "Reasons" below.)

RESET_reason Indicates the reason fro the reset. (See "Reasons" below.)

Valid States

This primitive is valid in the `NS_DATA_XFER` state.

New State

The new state is `NS_WRES_RIND`.

4.2.5.3 Reset Response

N_RESET_RES

This user-originated primitive indicates that the NS user has accepted a reset request.

Format

The format of the primitive is one `M_PROTO` message block and is structured as follows:

```
typedef struct {
        np_ulong PRIM_type;             /* always N_RESET_RES */
} N_reset_res_t;
```

Parameters

PRIM_type Specifies the primitive type: always `N_RESET_RES`.

Valid States

This primitive is valid in state `NS_WRES_RIND`.

New State

The new state is `NS_WACK_RRES`.

Acknowledgements

The NS provider should generate one of the following acknowledgements upon receipt of this primitive:

— *Successful:* The successful completion of this primitive is indicated with the `N_OK_ACK` primitive. This results in the data transfer state (`NS_DATA_XFER`).

— *Unsuccessful (Non-fatal errors):* An unsuccessful completion of this primitive is indicated with the `N_ERROR_ACK` primitive. The resulting state remains the same. The following non-fatal error codes are valid:

[NOUTSTATE] The primitive was issued from an invalid state.

[NSYSERR] A system error has occurred and the *UNIX*® system error is indicated in the `N_ERROR_ACK` primitive.

NOTE: If the interface is in the `NS_IDLE` state when the provider receives the `N_RESET_RES` primitive, then the NS provider should discard the message without generating an error.

4.2.5.4 Reset Confirmation

N_RESET_CON

This NS provider-originated primitive indicates to the network user that initiated the reset, that the reset request has been confirmed. The NS providers is allowed to issue the N_RESET_CON primitive to the NS user that initiated the reset even before receiving a N_RESET_RES.

Format

The format of the primitive is one M_PROTO message block, structured as follows:

```
typedef struct {
        np_ulong PRIM_type;            /* always N_RESET_CON */
} N_reset_con_t;
```

Parameters

PRIM_type	Indicates the primitive type: always N_RESET_CON.

Valid States

This primitive is valid in state NS_WCON_RREQ.

New State

The resulting state is NS_DATA_XFER.

4.2.6 Inform Service

4.2.6.1 SCCP Inform Request

N_INFORM_REQ

4.2.6.2 Format

```
#define N_INFORM_REQ    33
typedef struct {
        np_ulong PRIM_type;             /* always N_INFORM_REQ */
        np_ulong QOS_length;            /* qos parameters */
        np_ulong QOS_offset;
        np_ulong REASON;                /* inform reason */
} N_inform_req_t;

/* Inform reasons for use with N_INFORM_REQ */
#define N_SCCP_INFR_NSU_FAILURE         4
#define N_SCCP_INFR_NSU_CONGESTION      5
#define N_SCCP_INFR_NSU_QOS_CHANGE      6

/* Inform reasons for use with N_INFORM */
#define N_SCCP_INFR_UNSPECIFIED         7
```

Parameters

PRIM_type Specifies the primitive type: always `N_INFORM_REQ`.

QOS_length Specifies the length of the Quality of Service parameters.

QOS_offset Specifies the offset of the Quality of Service parameters from the beginning of the `M_PROTO` message block.

REASON Specifies the reason for informing the NS provider. (See "Reason" below.)

Reason

Valid States

This primitive is valid in the `NS_DATA_XFER` state.

New State

The new state remains unchanged.

Acknowledgements

4.2.6.3 SCCP Inform Indication

N_INFORM_IND

4.2.6.4 Format

```
#define N_INFORM_IND    34
typedef struct {
        np_ulong PRIM_type;             /* always N_INFORM_IND */
        np_ulong QOS_length;            /* qos parameters */
        np_ulong QOS_offset;
        np_ulong REASON;                /* inform reason */
} N_inform_ind_t;

/* Inform reasons for use with N_INFORM_IND */
#define N_SCCP_INFR_NSP_FAILURE         1
#define N_SCCP_INFR_NSP_CONGESTION      2
#define N_SCCP_INFR_NSP_QOS_CHANGE      3

/* Inform reasons for use with N_INFORM */
#define N_SCCP_INFR_UNSPECIFIED         7
```

Parameters

PRIM_type	Indicates the primitive type: always `N_INFORM_IND`.
QOS_length	Indicates the length of the Quality of Service parameters.
QOS_offset	Indicates the offset of the Quality of Service parameters from the beginning of the `M_PROTO` message block.
REASON	Indicates the reason for informing the NS user. (See “Reason” below.)

Reason

Valid States

This primitive is valid in the `NS_DATA_XFER` state.

New State

The new state remains unchanged.

4.2.7 Network Connection Release Phase

The NC release service primitives are used to release a NC. The release may be performed by:

- either or both of the NS users to release an established NC;
- the NS provider to release an established NC (all failures to maintain an NC are indicated in this manner);
- the destination NS user to reject an `N_CONN_IND`;
- by the NS provider to indicate its inability to establish a requested NC.

An NC release is permitted at any time regardless of the current phase of the NC. Once an NC release procedure has been invoked, the NC will be released; a request for release cannot be rejected. The network service does not guarantee delivery of any data once the NC release phase is entered (see Table 2.1).

4.2.7.1 Disconnect Request

N_DISCON_REQ

This user-originated primitive requests that the NS provider deny a request for a network connection, or disconnect an existing connection.

Format

The format of the primitive is one `M_PROTO` message block, followed by one or more `M_DATA` message blocks (for NS user data). The NS user data may be lost if the NS provider initiates release before the `N_DISCON_IND` is delivered. Therefore, the NS user data parameter is present only if the originator parameters (see Section 4.2.7.2 [Disconnect Indication], page 63) indicates that the release was originated by an NS user. The NS user may send any integral number of octets of data within a range supported by the NS provider (see Section 4.1.2 [Network Information Acknowledgement], page 23).

The structure of the `M_PROTO` message block is as follows:

```
typedef struct {
        np_ulong PRIM_type;            /* always N_DISCON_REQ */
        np_ulong DISCON_reason;        /* reason */
        np_ulong RES_length;           /* responding address length */
        np_ulong RES_offset;           /* responding address offset */
        np_ulong SEQ_number;           /* sequence number */
} N_discon_req_t;
```

Parameters

PRIM_type Specifies the primitive type: always `N_DISCON_REQ`.

DISCON_reason Specifies the disconnect reason. (See “Reason” below.)

RES_length Specifies the length of the responding address. The responding address parameter is an optional parameter, and is present in the primitive only in the case where the primitive is used to indicate rejection of an NC establishment attempt by an NS user. The responding address parameter conveys the network address of the NS user entity from which the `N_DISCON_REQ` was issued and under certain circumstances (e.g. call redirection, generic addressing, etc.) may be different from the *Destination Address* in the corresponding `N_CONN_REQ` primitive.

RES_offset Specifies the offset of the responding address from the beginning of the M_PROTO message block.

SEQ_number Specifies the connection indication being disconnected. When non-zero, it identifies the sequence number of the N_CONN_IND message being rejected. This number is used by the NS provider to associate the N_DISCON_REQ with an unacknowledged N_CONN_IND that is to be rejected. If the N_DISCON_REQ is rejecting a NC that is already established (or rejecting a N_CONN_REQ that the NS user had previously sent and has not yet been confirmed), then this field should have a value of zero ('0').

Valid States

This primitive is valid in states NS_WCON_CREQ, NS_WRES_CIND, NS_DATA_XFER, NS_WCON_RREQ and NS_WRES_RIND.

New State

The new state depends on the original state (see ⟨undefined⟩ [⟨undefined⟩], page ⟨undefined⟩).

Acknowledgements

The NS provider should generate one of the following acknowledgements upon receipt of this primitive:

- *Successful:* Successful completion is indicated with the N_OK_ACK primitive.
- *Unsuccessful (Non-fatal errors):* Errors are indicated with the N_ERROR_ACK primitive. The applicable non-fatal errors are as follows:

  [NOUTSTATE] The primitive was issued from an invalid state.

  [NBADDATA] The amount of user data specified was outside the range supported by the NS provider.

  [NSYSERR] A system error has occurred and the *UNIX*® system error is indicated in the primitive.

  [NBADSEQ] The specified sequence number referred to an invalid N_CONN_IND message, or the N_DISCON_REQ is rejecting an NC that is already established (or rejecting an N_CONN_REQ that the NS user had previously sent and has not yet been confirmed) and the value of the sequence number is not '0'.

4.2.7.2 Disconnect Indication

N_DISCON_IND

This network-provider originated primitive indicates to the NS user that either a request for connection has been denied or an existing connection has been disconnected.

Format

The format of the message is one M_PROTO message block, followed by one or more M_DATA blocks. The value of the NS user data parameter is identical to the value in the corresponding N_DISCON_REQ primitive. The NS user data parameter is present only if the originator parameter indicates that the release was initiated by the NS user.

The structure of the M_PROTO message block is as follows:

```
typedef struct {
        np_ulong PRIM_type;            /* always N_DISCON_IND */
        np_ulong DISCON_orig;          /* originator */
        np_ulong DISCON_reason;        /* reason */
        np_ulong RES_length;           /* address length */
        np_ulong RES_offset;           /* address offset */
        np_ulong SEQ_number;           /* sequence number */
} N_discon_ind_t;
```

Parameters

PRIM_type Indicates the primitive type: always N_DISCON_REQ.

DISCON_orig Indicates the disconnect source. (See "Reason" below.)

DISCON_reason Indicates the disconnect reason. (See "Reason" below.)

RES_length Indicates the length of the responding address. The responding address parameter is an optional parameter, and is present in the primitive only in the case where the primitive is used to indicate rejection of an NC establishment attempt by an NS user. When not present, the value of this parameter is zero. When present, the value of the disconnect address parameter is identical to that supplied with the corresponding N_DISCON_REQ primitive.

RES_offset Indicates the offset of the responding address from the beginning of the M_PROTO message block.

SEQ_number Indicates the connection indication being disconnected. When its value is non-zero, it identifies the sequence number associated with the N_CONN_IND primitive that is being aborted.

The value of this parameter must be zero when:

a. indicating the rejection of a previously issued N_CONN_REQ primitive; or

b. indicating the release of a NC that is already successfully established.

When this field is non-zero and its value is the same as the sequence number assigned to an unacknowledged N_CONN_IND, it indicates that the NS provider is canceling the unacknowledged N_CONN_IND.

Valid States

The valid states are as follows:

NS_WCON_CREQ Waiting confirmation of connection request.

NS_WRES_CIND Waiting response of connection indication.

NS_DATA_XFER Waiting response of data transfer.

`NS_WCON_RREQ`	Waiting confirmation of reset request.
`NS_WRES_RIND`	Waiting response to reset indication.

New State

The new state is `NS_IDLE` (except when number of outstanding connect indications is greater than 1, in which case the resulting state is `NS_WRES_CIND`).

4.3 Connectionless Mode Primitives

This section describes the format of the CLNS primitives and the rules associated with these primitives. The values of the QOS parameters associated with each unit data transmission are selected with the N_OPTMGMT_REQ primitive.

4.3.1 Unit Data Transfer

4.3.1.1 Unit Data Request

N_UNITDATA_REQ

This primitive requests that the NS provider send the specified datagram to the specified destination.

Format

The format of the primitive is one M_PROTO message block followed by one or more M_DATA message blocks. The M_PROTO message block is structured as followed:

```
typedef struct {
        np_ulong PRIM_type;             /* always N_UNITDATA_REQ */
        np_ulong DEST_length;           /* destination address length */
        np_ulong DEST_offset;           /* destination address offset */
        np_ulong RESERVED_field[2];     /* reserved field for DLPI compatibility */
} N_unitdata_req_t;
```

Parameters

PRIM_type Specifies the primitive type: always N_UNITDATA_REQ.

DEST_length Specifies the length of the destination address.

DEST_offset Specifies the offset of the destination address from the beginning of the M_PROTO message block.

RESERVED_field[0] Specified the length of the Quality of Service parameters.

RESERVED_field[1] Specified the offset of the Quality of Service parameters from the beginning of the M_PROTO message block.

Valid States

This primitive is valid in state NS_IDLE.

New State

The resulting state remains unchanged.

Acknowledgements

The NS provider should generate one of the following acknowledgements upon receipt of this primitive:

— *Successful:* There is no acknowledgement for the successful completion of this primitive.

— *Unsuccessful (Non-fatal errors):* If a non-fatal error occurs, it is the responsibility of the NS provider to report it with the N_UDERROR_IND primitive. The following non-fatal error codes are allowed:

[NBADADDR] The network address as specified in the primitive was in an incorrect format, or the address contained illegal information.

`[NBADDATA]` The amount of user data specified was outside the range supported by the NS provider.

`[NOUTSTATE]` The primitive was issued from an invalid state.

— *Fatal Error:* Fatal errors are indicated with an `M_ERROR` STREAMS message type (issued to the NS user with the error number value of `[EPROTO]`), that results in the failure of all *UNIX*® system calls on the Stream. The fatal errors are as follows:

`[EPROTO]` This indicates one of the following unrecoverable protocol conditions:

— The network service interface was found to be in an incorrect state.

— The amount of NS user data associated with the primitive defines a network service data unit larger than that allowed by the NS provider.

4.3.1.2 Unit Data Indication

N_UNITDATA_IND

This primitive indicates to the NS user that a datagram has been received from the specified source address.

Format

The format of the message is one M_PROTO message block followed by one or more M_DATA message blocks containing at least one byte of data. The format of the M_PROTO is as follows:

```
typedef struct {
        np_ulong PRIM_type;             /* always N_UNITDATA_IND */
        np_ulong SRC_length;            /* source address length */
        np_ulong SRC_offset;            /* source address offset */
        np_ulong DEST_length;           /* source address length */
        np_ulong DEST_offset;           /* source address offset */
        np_ulong ERROR_type;            /* reserved field for DLPI compatibility */
} N_unitdata_ind_t;
```

Parameters

PRIM_type Indicates the primitive type: always N_UNITDATA_IND.

SRC_length Indicates the length of the source network address. This address is the same as the value associated with the Stream on which the N_UNITDATA_REQ was issued.

SRC_offset Indicates the offset of the source address from the beginning of the M_PROTO message block.

DEST_length Indicates the length of the destination address. The address is the same as in the corresponding N_UNITDATA_REQ primitive.

DEST_offset Indicates the offset of the destination address from the beginning of the M_PROTO message block.

ERROR_type Specifies the reason for the error. The possible values are:

N_UD_CONGESTION
This packet experienced congestion during its delivery.

Valid States

This primitive is valid in state NS_IDLE.

New State

The resulting state remains unchanged.

4.3.2 Unit Data Error

4.3.2.1 Unit Data Error Indication

N_UDERROR_IND

This primitive indicates to the NS user that a datagram with the specified destination address and QOS parameters has resulted in an error condition.

Format

The format of the primitive is one `M_PROTO` message block, structured as follows:

```
typedef struct {
        np_ulong PRIM_type;             /* always N_UDERROR_IND */
        np_ulong DEST_length;           /* destination address length */
        np_ulong DEST_offset;           /* destination address offset */
        np_ulong RESERVED_field;        /* reserved field for DLPI compatibility */
        np_ulong ERROR_type;            /* error type */
} N_uderror_ind_t;
```

Parameters

PRIM_type Indicates the primitive type: always `N_UDERROR_IND`.

DEST_length Indicates the length of the destination address. The address is the same as in the corresponding `N_UNITDATA_REQ` primitive.

DEST_offset Indicates the offset of the destination address from the beginning of the `M_PROTO` message block.

RESERVED_field This field is reserved whose value must be set to zero.

ERROR_type Indicates the reason for the error. (See “Errors” below.)

Valid States

This primitive is valid in state `NS_IDLE`.

New State

The resulting state remains unchanged.

4.3.2.2 Notice Indication

N_NOTICE_IND

This primitive indicates to the NS user that a datagram with the specifid destination address, source address and Quality of Service parameters has been returned due to an error.

Format

The format of the mssage is one `M_PROTO` message block followed by one or more `M_DATA` message blocks (containing the originalling sent NS user data).

```
#define N_NOTICE_IND    32
typedef struct {
        np_ulong PRIM_type;             /* always N_NOTICE_IND */
        np_ulong DEST_length;           /* address data was sent to */
        np_ulong DEST_offset;
        np_ulong SRC_length;            /* address data was sent from */
        np_ulong SRC_offset;
        np_ulong QOS_length;            /* QOS parameters data was sent with */
        np_ulong QOS_offset;
        np_ulong RETURN_cause;          /* reason for return of data */
} N_notice_ind_t;
```

Parameters

PRIM_type	Indicates the primitive type: always `N_NOTICE_IND`.
DEST_length	Indicates the length of the destination address.
DEST_offset	Indicates the offset of the destination address from the beginning of the `M_PROTO` message block.
SRC_length	Indicates the length of the source address.
SRC_offset	Indicates the offset of the source address from the beginning of the `M_PROTO` message block.
QOS_length	Indicates the length of the Quality of Service parameters.
QOS_offset	Indicates the length of the Quality of Service parameters from the beginning of the `M_PROTO` message block.
RETURN_cause	Indicates the cause for the return of the datagram. (See “Cause” below.)

Cause

`SCCP_RETC_NO_ADDRESS_TYPE_TRANSLATION`
No address type translation.

`SCCP_RETC_NO_ADDRESS_TRANSLATION`
No address translation.

`SCCP_RETC_SUBSYSTEM_CONGESTION`
Subsystem congestion.

`SCCP_RETC_SUBSYSTEM_FAILURE`
Subsystem failure.

`SCCP_RETC_UNEQUIPPED_USER`
Unequipped user.

`SCCP_RETC_MTP_FAILURE`
: MTP failure.

`SCCP_RETC_NETWORK_CONGESTION`
: Network congestion.

`SCCP_RETC_UNQUALIFIED`
: Unqualified.

`SCCP_RETC_MESSAGE_TRANSPORT_ERROR`
: Message transport error.

`SCCP_RETC_LOCAL_PROCESSING_ERROR`
: Local processing error.

`SCCP_RETC_NO_REASSEMBLY_AT_DESTINATION`
: No reassembly at destination.

`SCCP_RETC_SCCP_FAILURE`
: SCCP failure.

`SCCP_RETC_SCCP_HOP_COUNTER_VIOLATION`
: SCCP hop counter violation.

`SCCP_RETC_SEGMENTATION_NOT_SUPPORTED`
: Segmentation not supported.

`SCCP_RETC_SEGMENTATION_FAILURE`
: Segmenetation failure.

`SCCP_RETC_MESSAGE_CHANGE_FAILURE`
: Message change failure.

`SCCP_RETC_INVALID_INS_ROUTING_REQUEST`
: Invalid INS routing request.

`SCCP_RETC_INVALID_INSI_ROUTING_REQUEST`
: Invalid INSI routing request.

`SCCP_RETC_UNAUTHORIZED_MESSAGE`
: Unauthorized message.

`SCCP_RETC_MESSAGE_INCOMPATIBILITY`
: Message incompatibility.

`SCCP_RETC_CANNOT_PERFORM_ISNI_CONSTRAINED_ROUTING`
: Cannot perform ISNI constrained routing.

`SCCP_RETC_REDUNDANT_ISNI_CONSTRAINED_ROUTING_INFO`
: Redundant ISNI constrained routing information.

`SCCP_RETC_UNABLE_TO_PERFORM_ISNI_IDENTIFICATION`
: Unable to perform ISNI identification.

Valid States

This primitive is valid in state `NS_IDLE`.

New State

The resulting state remains unchanged.

4.4 SCCP Provider Management Primitives

4.4.1 SCCP Status Service

4.4.1.1 State Request

N_STATE_REQ

Format

```
#define N_STATE_REQ     39
typedef struct {
        np_ulong PRIM_type;             /* always N_STATE_REQ */
        np_ulong ADDR_length;           /* affected subsystem */
        np_ulong ADDR_offset;
        np_ulong STATUS;                /* user status */
} N_state_req_t;
```

Parameters

PRIM_type	Specifies the primitive type: always `N_STATE_REQ`.
ADDR_length	Specifies the length of the affected address (point code and subsystem).
ADDR_offset	Specifies the offset of the affected address (point code and subsystem) from the beginning of the `M_PROTO` message block.
STATUS	Specifies the user status. (See "Status" below.)

Status

Valid States

New State

Acknowledgements

The NS provider should generate one of the following acknowledgements upon receipt of this primitive:

— *Succcessful:*

— *Unsuccessful (Non-fatal errors):*

4.4.1.2 State Indication

N_STATE_IND

Format

The format of the message is one `M_PCPROTO` message block. The structure of the `M_PCPROTO` message block is as follows:

```
#define N_STATE_IND     40
typedef struct {
        np_ulong PRIM_type;             /* always N_STATE_IND */
        np_ulong ADDR_length;           /* affected subsystem */
        np_ulong ADDR_offset;
        np_ulong STATUS;                /* user status */
        np_ulong SMI;                   /* subsystem multiplicity indicator */
} N_state_ind_t;
```

Parameters

PRIM_type Indicates the primitive type: always `N_STATE_IND`.

ADDR_length Indicates the length of the affected address (point code and subsystem).

ADDR_offset Indicates the offset of the affected address (point code and subsystem) from the beginning of the `M_PROTO` message block.

STATUS Indicates the user status. (See “Status” below.)

SMI Indicates the subsystem multiplicity indicator.

Type and Status

Valid States

New State

4.4.2 SCCP PC Status Service

4.4.2.1 PC State Indication

N_PCSTATE_IND

This primitive indicates to the SCCP-User that the indicated remote SCCP-entity (signalling point) is temporarily inaccessible. This implies the inaccessibility of remote SCCP-User at the affected signalling point.

Format

The format of the message is one `M_PROTO` message block. The structure of the `M_PROTO` message block is as follows:

```
#define N_PCSTATE_IND   41
typedef struct {
        np_ulong PRIM_type;             /* always N_PCSTATE_IND */
        np_ulong ADDR_length;           /* affected point code */
        np_ulong ADDR_offset;
        np_ulong STATUS;                /* status */
} N_pcstate_ind_t;
```

Parameters

PRIM_type	Indicates the primitive type: always `N_PCSTATE_IND`.
ADDR_length	Indicates the length of the affected address (point code and subsystem).
ADDR_offset	Indicates the offset of the affected address (point code and subsystem) from the beginning of the `M_PROTO` message block.
STATUS	Indicates the user status. (See “Status” below.)

Valid States

New State

4.4.3 SCCP Coordination Service

4.4.3.1 Coordination Request

N_COORD_REQ

Format

```
#define N_COORD_REQ     35
typedef struct {
        np_ulong PRIM_type;             /* alwyas N_COORD_REQ */
        np_ulong ADDR_length;           /* affected subsystem */
        np_ulong ADDR_offset;
} N_coord_req_t;
```

Parameters

Valid States

New State

Acknowledgements

The NS provider should generate one of the following acknowledgements upon receipt of this primitive:

— *Succcessful:*

— *Unsuccessful (Non-fatal errors):*

4.4.3.2 Coordination Indication

N_COORD_IND

Format

```
#define N_COORD_IND     37
typedef struct {
        np_ulong PRIM_type;             /* alwyas N_COORD_IND */
        np_ulong ADDR_length;           /* affected subsystem */
        np_ulong ADDR_offset;
        np_ulong SMI;                   /* subsystem multiplicity indicator */
} N_coord_ind_t;
```

Parameters

PRIM_type	Indicates the primitive type: always `N_COORD_IND`.
ADDR_length	Indicates the length of the affected address (point code and subsystem).
ADDR_offset	Indicates the offset of the affected address (point code and subsystem) from the beginning of the `M_PROTO` message block.
SMI	Indicates the subsystem multiplicity indicator.

Valid States

New State

4.4.3.3 Coordination Response

N_COORD_RES

Format

```
#define N_COORD_RES     36
typedef struct {
        np_ulong PRIM_type;             /* always N_COORD_RES */
        np_ulong ADDR_length;           /* affected subsystem */
        np_ulong ADDR_offset;
} N_coord_res_t;
```

Parameters

PRIM_type Specifies the primitive type: always N_COORD_RES.

ADDR_length Specifies the length of the affected address (point code and subsystem).

ADDR_offset Specifies the offset of the affected address (point code and subsystem) from the beginning of the M_PROTO message block.

Valid States

New State

Acknowledgements

The NS provider should generate one of the following acknowledgements upon receipt of this primitive:

— *Succcessful:*

— *Unsuccessful (Non-fatal errors):*

4.4.3.4 Coordination Confirmation

N_COORD_CON

Format

```
#define N_COORD_CON     38
typedef struct {
        np_ulong PRIM_type;             /* always N_COORD_CON */
        np_ulong ADDR_length;           /* affected subsystem */
        np_ulong ADDR_offset;
        np_ulong SMI;                   /* subsystem multiplicity indicator */
} N_coord_con_t;
```

Parameters

PRIM_type Indicates the primitive type: always `N_PCSTATE_IND`.

ADDR_length Indicates the length of the affected address (point code and subsystem).

ADDR_offset Indicates the offset of the affected address (point code and subsystem) from the beginning of the `M_PROTO` message block.

STATUS Indicates the user status. (See “Status” below.)

Valid States

New State

5 Diagnostics Requirements

Two error handling facilities should be provided to the SCCP user: one to handle non-fatal errors, and the other to handle fatal errors.

5.1 Non-Fatal Error Handling Facility

These are errors that do not change the state of the SCCP service interface as seen by the SCCP service user, and provide the user the option of reissuing the SCCP service primitive with the corrected options specification. The non-fatal error handling is provided only to those primitive that require acknowledgements, and uses the `N_ERROR_ACK` primitive to report these errors. These errors retain the state of hte SCCP service interface the same as it was before the SCCP service provider received the primitive that was in error. Syntax errros and rule violations are reported via the non-fatal error handling facility.

5.2 Fatal Error Handling Facility

These errors are issued by the SCCP provider when it detects errors that are not correctable by the SCCP user, or if it is unable to report a correctible error to the SCCP user. Fatal errors are indicated via the STREAMS message type `M_ERROR` with the UNIX system error `[EPROTO]`. The `M_ERROR` STREAMS message type will result in the failure of all the UNIX system calls on the stream. The SCCP user can recover from a fatal error by having all the processes close the files associated with the stream, and then reopening them for processing.

These errors are issued by the SCCP when it detects errors that are not correctable by the SCCP service user, or if it is unable to report a correctable error to the SCCP service user. Fatal errors are indicated via the STREAMS message type `M_ERROR` with the UNIX system error `[EPROTO]`. The `M_ERROR` STREAMS message type will result in the failure of all the UNIX system calls on the stream. The SCCP service user can recover from a fatal error by having all the processes close the files associated with the stream, and then reopening them for processing.

6 SCCPI Input-Output Controls

7 SCCPI Management Information Base

Addendum for SCCP Conformance

Addendum for ITU-T Q.711 Conformance

This addendum describes the formats and rules that are specific to ETSI EN 300 009-1 V3.2.2. The addendum must be used along with the generic SCCPI as defined in the main document, and the EN 300 009-1 conformance defined in Addendum 2, when implementing an SCCP that will be configured with the EN 300 008-1 Signalling Connection Control Part.

Primitives and Rules for ETSI EN 300 009-1 V3.2.2 Conformance

The following are the additional rules that apply to the SCCPI primitives for ETSI EN 300 009-1 V3.2.2 compatibility.

Local Management Primitives

Parameters

Flags

Rules

Connection Mode Primitives

Parameters

Flags

Rules

Connectionless Primitives

Parameters

Flags

Rules

Addendum for ANSI T1.112 Conformance

Addendum for ETSI EN 300 009-1 Conformance

Appendix A Mapping SCCPI Primitives

A.1 Mapping SCCPI Primitives to Q.711

Table A-1 shows the mapping of the SCCPI primitives to the SCCP definition primitives listed in ITU-T Recommendation Q.711.

The mapping of SCCPI primitives to Q.711 primitives is shown in ⟨undefined⟩ [⟨undefined⟩], page ⟨undefined⟩. For the most part, this mapping is a one to one mapping of service primitives, with the exception of *Connect Request* and *Disconnect Request*.

In Q.711 there is no concept of an *NC* between SCCP-entities. In OpenSS7 SCCPI, the `N_CONN_REQ` and `N_DISCON_REQ` primitives are used to establish and release an NC between SCCP-entities.

A.2 Mapping SCCPI Primitives to ANSI T1.112

The mapping of SCCPI primitives to T1.112 primitives is shown in ⟨undefined⟩ [⟨undefined⟩], page ⟨undefined⟩. For the most part, this mapping is a one to one mapping of service primitives, with the exception of *Connect Request* and *Disconnect Request*.

In T1.112 there is no concept of an *NC* between SCCP-entities. In OpenSS7 SCCPI, the `N_CONN_REQ` and `N_DISCON_REQ` primitives are used to establish and release an NC between SCCP-entities.

A.3 Mapping SCCPI Primitives to ETSI EN 300 009-1

The mapping of SCCPI primitives to EN 300 009-1 primitives is shown in ⟨undefined⟩ [⟨undefined⟩], page ⟨undefined⟩. For the most part, this mapping is a one to one mapping of service primitives, with the exception of *Connect Request* and *Disconnect Request*.

In EN 300 009-1 there is no concept of an *NC* between SCCP-entities. In OpenSS7 SCCPI, the `N_CONN_REQ` and `N_DISCON_REQ` primitives are used to establish and release an NC between SCCP-entities.

Appendix B State/Event Tables

Appendix C Precedence Tables

Appendix D SCCPI Header Files

Appendix E SCCPI Library

Appendix F SCCPI Drivers and Modules

Appendix G SCCPI Utilities

Appendix H SCCPI File Formats

Appendix I SCCPI Compatibility and Porting

Glossary

Signalling Data Link Service Data Unit
A grouping of SDL user data whose boundaries are preserved from one end of the signalling data link connection to the other.

Data transfer
The phase in connection and connectionless modes that supports the transfer of data between to signalling data link users.

SDL provider
The signalling data link layer protocol that provides the services of the signalling data link interface.

SDL user
The user-level application or user-level or kernel-level protocol that accesses the services of the signalling data link layer.

Local management
The phase in connection and connectionless modes in which a SDL user initializes a Stream and attaches a PPA address to the Stream. Primitives in this phase generate local operations only.

PPA
The point at which a system attaches itself to a physical communications medium.

PPA identifier
An identifier of a particular physical medium over which communication transpires.

Acronyms

ANSI	American National Standards Institute
CCITT	The International Telegraph and Telephone Consutative Committee, old name for ITU-T
CONS	Connection-Oriented Network Service
CUD	Call User Data
DCE	Data Circuit-terminating Equipment
DDN	Defence Data Network
DLPI	Data Link Provider Interface
DLSAP	Destination Link Service Access Point
DNIC	Data Network Identification Code
DSAP	Destination Service Access Point
DTE	Data Terminal Equipment
ENSDU	Expedited Network Service Data Unit
ETSI	European Telecommunications Standards Institute
HDLC	High-Level Data Link Control
IEEE	Institute of Electrical and Electronics Engineers
IP	Internet Protocol
ISDNI	ISDN Interface
ISDN	Integrated Services Digital Network
ISO	International Organization for Standardization
ISUPI	ISUP Interface
ISUP	ISDN User Part
ITU	International Telecommunications Union
ITU-T	ITU Telecom Sector
LAN	Local Area Network
LAPB	Link Access Procedure (Balanced), ISO/IEC 7776
LAPD	Link Access Procedure D-Channel, Q.921
LAPF	Link Access Procedure Frame Mode, Q.922
LAP	Link Access Procedure
LCI	Logical Channel Identifier
LLC1	Logical Link Control Type 1
LLC2	Logical Link Control Type 2
LLC3	Logical Link Control Type 3
LLC	Logical Link Control
LLI	Logical Link Inteface
LSAP	Link Service Access Point
MAC	Media Access Control
MTPI	Message Transfer Part Interface
MTP	Message Transfer Part
NLI	Network Layer Interface
NPDU	Network Protocol Data Unit
NPI	Network Provider Interface
NPI	Numbering Plan Indicator
NSAP	Network Service Access Point
NSDU	Network Service Data Unit
NSP	Network Service Provider
NS	Network Service

NSU	Network Service User
NUI	Network User Information
PAD	Packet Assembler/Disassembler
PDN	Public Data Network
PDU	Protocol Data Unit
PLP	Packet Layer Protocol
PPA	Physical Point of Attachment
PSDN	Public Switched Data Network
PSTN	Public Switch Telephone Network
PVC	Permanent Virtual Circuit
QOS	Quality of Service
RPOA	Recognized Private Operating Agency
SAP	Service Access Point
SCCPI	Signalling Connection Control Part Interface
SCCP	Signalling Connection Control Part
SDLI	Signalling Data Link Interface
SDL	Signalling Data Link
SDTI	Signalling Data Terminal Interface
SDT	Signalling Data Terminal
SDU	Service Data Unit
SLI	Signalling Link Interface
SLSAP	Source Link Service Access Point
SL	Signalling Link
SNPA	Subnetwork Point of Attachment
SSAP	Source Service Access Point
SVC	Switched Virtual Circuit
TCAP	Transaction Capabilities Application Part
TCI	Transaction Component Interface
TC	Component Handling Sub-Layer
TLI	Transport Layer Interface
TOA/NPI	Type of Address/Numbering Plan Indicator
TOA	Type of Address
TPI	Transport Provider Interface
TRI	Transaction Interface
TR	Transaction Handling Sub-Layer
VC	Virtual Circuit
WAN	Wide Area Network
X.121	ITU-T Recommendation X.121
X.25	ITU-T Recommendation X.25
X.28	ITU-T Recommendation X.28
X.3	ITU-T Recommendation X.3
X.75	ITU-T Recommendation X.75
XX25	X.25 Programming Inteface using XTI
XXX	X.3, X.28, X.29

References

[1] ITU-T Recommendation Q.700, *Introduction to CCITT Signalling System No. 7*, March 1993, (Geneva), ITU, ITU-T Telecommunication Standardization Sector of ITU, (Previously "CCITT Recommendation").

[2] ITU-T Recommendation Q.701, *Functional Description of the Message Transfer Part (MTP) of Signalling System No. 7*, March 1993, (Geneva), ITU, ITU-T Telecommunication Standardization Sector of ITU, (Previously "CCITT Recommendation").

[3] ITU-T Recommendation Q.702, *Signalling System No. 7—Signalling Data Link*, March 1993, (Geneva), ITU, ITU-T Telecommunication Standardization Sector of ITU, (Previously "CCITT Recommendation").

[4] ITU-T Recommendation Q.703, *Signalling System No. 7—Signalling Link*, March 1993, (Geneva), ITU, ITU-T Telecommunication Standardization Sector of ITU, (Previously "CCITT Recommendation").

[5] ITU-T Recommendation Q.704, *Message Transfer Part—Signalling Network Functions and Messages*, March 1993, (Geneva), ITU, ITU-T Telecommunication Standardization Sector of ITU, (Previously "CCITT Recommendation").

[6] Geoffrey Gerrietts; Dave Grothe, Mikel Matthews, Dave Healy, *CDI—Application Program Interface Guide*, March 1999, (Savoy, IL), GCOM, Inc.

[7] ITU-T Recommendation Q.771, *Signalling System No. 7—Functional Description of Transaction Capabilities*, March 1993, (Geneva), ITU, ITU-T Telecommunication Standardization Sector of ITU, (Previously "CCITT Recommendation").

Licenses

All code presented in this manual is licensed under the [GNU Affero General Public License], page 115. The text of this manual is licensed under the [GNU Free Documentation License], page 125, with no invariant sections, no front-cover texts and no back-cover texts. Please note, however, that it is just plain wrong to modify statements of, or attribute statements to, the Author or *OpenSS7 Corporation*.

GNU Affero General Public License

Preamble

The GNU Affero General Public License is a free, copyleft license for software and other kinds of works, specifically designed to ensure cooperation with the community in the case of network server software.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, our General Public Licenses are intended to guarantee your freedom to share and change all versions of a program–to make sure it remains free software for all its users.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

Developers that use our General Public Licenses protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License which gives you legal permission to copy, distribute and/or modify the software.

A secondary benefit of defending all users' freedom is that improvements made in alternate versions of the program, if they receive widespread use, become available for other developers to incorporate. Many developers of free software are heartened and encouraged by the resulting cooperation. However, in the case of software used on network servers, this result may fail to come about. The GNU General Public License permits making a modified version and letting the public access it on a server without ever releasing its source code to the public.

The GNU Affero General Public License is designed specifically to ensure that, in such cases, the modified source code becomes available to the community. It requires the operator of a network server to provide the source code of the modified version running there to the users of that server. Therefore, public use of a modified version, on a publicly accessible server, gives the public access to the source code of the modified version.

An older license, called the Affero General Public License and published by Affero, was designed to accomplish similar goals. This is a different license, not a version of the Affero GPL, but Affero has released a new version of the Affero GPL which permits relicensing under this license.

The precise terms and conditions for copying, distribution and modification follow.

Terms and Conditions

0. Definitions.

   “This License” refers to version 3 of the GNU Affero General Public License.

   “Copyright” also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

   “The Program” refers to any copyrightable work licensed under this License. Each licensee is addressed as “you”. “Licensees” and “recipients” may be individuals or organizations.

   To “modify” a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a “modified version” of the earlier work or a work “based on” the earlier work.

   A “covered work” means either the unmodified Program or a work based on the Program.

   To “propagate” a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

   To “convey” a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

   An interactive user interface displays “Appropriate Legal Notices” to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

   The “source code” for a work means the preferred form of the work for making modifications to it. “Object code” means any non-source form of a work.

   A “Standard Interface” means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

   The “System Libraries” of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A “Major Component”, in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

   The “Corresponding Source” for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work’s System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

   All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

   You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

   Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

   No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

   When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

   You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

   You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

   You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

   a. The work must carry prominent notices stating that you modified it, and giving a relevant date.

b. The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".

c. You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.

d. If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

a. Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.

b. Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

c. Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d. Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e. Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

   "Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

   When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

a. Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
b. Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
c. Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or
d. Limiting the use for publicity purposes of names of licensors or authors of the material; or
e. Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
f. Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

   You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

    Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

    An “entity transaction” is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party’s predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

    You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

    A “contributor” is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor’s “contributor version”.

    A contributor’s “essential patent claims” are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, “control” includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

    Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor’s essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

    In the following three paragraphs, a “patent license” is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To “grant” such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

    If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. “Knowingly relying” means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient’s use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is “discriminatory” if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others’ Freedom.

    If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Remote Network Interaction; Use with the GNU General Public License.

    Notwithstanding any other provision of this License, if you modify the Program, your modified version must prominently offer all users interacting with it remotely through a network (if your version supports such interaction) an opportunity to receive the Corresponding Source of your version by providing access to the Corresponding Source from a network server at no charge, through some standard or customary means of facilitating copying of software. This Corresponding Source shall include the Corresponding Source for any work covered by version 3 of the GNU General Public License that is incorporated pursuant to the following paragraph.

    Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the work with which it is combined will remain governed by version 3 of the GNU General Public License.

14. Revised Versions of this License.

    The Free Software Foundation may publish revised and/or new versions of the GNU Affero General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU Affero General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU Affero General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU Affero General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

    THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

    IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

    If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
one line to give the program's name and a brief idea of what it does.
Copyright (C) year name of author

This program is free software: you can redistribute it and/or modify
it under the terms of the GNU Affero General Public License as published by
the Free Software Foundation, either version 3 of the License, or (at
your option) any later version.

This program is distributed in the hope that it will be useful, but
WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
Affero General Public License for more details.

You should have received a copy of the GNU Affero General Public License
along with this program.  If not, see http://www.gnu.org/licenses/.
```

Also add information on how to contact you by electronic and paper mail.

If your software can interact with users remotely through a network, you should also make sure that it provides a way for users to get its source. For example, if your program is a web application, its interface could display a "Source" link that leads users to an archive of the code. There are many ways you could offer source, and different solutions will be better for different programs; see section 13 for the specific requirements.

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if necessary. For more information on this, and how to apply and follow the GNU AGPL, see `http://www.gnu.org/licenses/`.

GNU Free Documentation License

GNU FREE DOCUMENTATION LICENSE
Version 1.3, 3 November 2008

0. PREAMBLE

   The purpose of this License is to make a manual, textbook, or other functional and useful document *free* in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

   This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

   We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

   This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

   A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

   A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

   The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The “Cover Texts” are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A “Transparent” copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not “Transparent” is called “Opaque”.

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTEX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The “Title Page” means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, “Title Page” means the text near the most prominent appearance of the work’s title, preceding the beginning of the body of the text.

The “publisher” means any person or entity that distributes copies of the Document to the public.

A section “Entitled XYZ” means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as “Acknowledgements”, “Dedications”, “Endorsements”, or “History”.) To “Preserve the Title” of such a section when you modify the Document means that it remains a section “Entitled XYZ” according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

   You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

   You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

   If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

   If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

   If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

   It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

   You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

   A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

   B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

   C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

   D. Preserve all the copyright notices of the Document.

   E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

   F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.

O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

   You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements."

6. COLLECTIONS OF DOCUMENTS

   You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

   You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

   A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

   If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

   Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

   If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

   You may not copy, modify, sublicense, or distribute the Document except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, or distribute it is void, and will automatically terminate your rights under this License.

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, receipt of a copy of some or all of the same material does not give you any rights to use it.

10. FUTURE REVISIONS OF THIS LICENSE

    The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

    Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation. If the Document specifies that a proxy can decide which future versions of this License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Document.

11. RELICENSING

    "Massive Multiauthor Collaboration Site" (or "MMC Site") means any World Wide Web server that publishes copyrightable works and also provides prominent facilities for anybody to edit those works. A public wiki that anybody can edit is an example of such a server. A "Massive Multiauthor Collaboration" (or "MMC") contained in the site means any set of copyrightable works thus published on the MMC site.

    "CC-BY-SA" means the Creative Commons Attribution-Share Alike 3.0 license published by Creative Commons Corporation, a not-for-profit corporation with a principal place of business in San Francisco, California, as well as future copyleft versions of that license published by that same organization.

    "Incorporate" means to publish or republish a Document, in whole or in part, as part of another Document.

    An MMC is "eligible for relicensing" if it is licensed under this License, and if all works that were first published under this License somewhere other than this MMC, and subsequently incorporated in whole or in part into the MMC, (1) had no cover texts or invariant sections, and (2) were thus incorporated prior to November 1, 2008.

    The operator of an MMC Site may republish an MMC contained in the site under CC-BY-SA on the same site at any time before August 1, 2009, provided the MMC is eligible for relicensing.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

```
  Copyright (C)  year  your name.
  Permission is granted to copy, distribute and/or modify this document
  under the terms of the GNU Free Documentation License, Version 1.3
  or any later version published by the Free Software Foundation;
  with no Invariant Sections, no Front-Cover Texts, and no Back-Cover
  Texts.  A copy of the license is included in the section entitled ``GNU
  Free Documentation License''.
```

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with. . . Texts." line with this:

```
    with the Invariant Sections being list their titles, with
    the Front-Cover Texts being list, and with the Back-Cover Texts
    being list.
```

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

Index