

Network Working Group  
Internet-Draft  
Expires: August 14, 2001

Bormann  
Kutscher  
Ott  
TZI, Universitaet Bremen  
Trossen  
Nokia Research Center  
February 13, 2001

Simple Conference Control Protocol -- Service Specification  
draft-ietf-mmusic-sccp-01.txt

Status of this Memo

This document is an Internet-Draft and is in full conformance with all provisions of [Section 10 of RFC2026](#).

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

To view the entire list of Internet-Draft Shadow Directories, see <http://www.ietf.org/shadow.html>.

This Internet-Draft will expire on August 14, 2001.

Copyright Notice

Copyright (C) The Internet Society (2001). All Rights Reserved.

Abstract

This document defines the services for a simple conference control protocol (SCCP) to be used for tightly coupled conferences. It is part of the Internet Multimedia Conferencing Architecture, proposed in [1].

The SCCP services provide functionality for management of the set of members, management of the set of application/media sessions, and for floor control to implement access control rules for distributed application resources.

Note that this document does not specify how to implement the proposed services. For that, different mappings are specified based

Internet-Draft

sccp-services

February 2001

on different transport layer techniques.

This document is a product of the Multiparty Multimedia Session Control (MMUSIC) working group of the Internet Engineering Task Force. Comments are solicited and should be addressed to the working group's mailing list at [confctrl@isi.edu](mailto:confctrl@isi.edu) and/or the authors.

## Table of Contents

<a href="#">1.</a>	Introduction . . . . .	<a href="#">3</a>
<a href="#">1.1</a>	Overview . . . . .	<a href="#">3</a>
<a href="#">1.2</a>	SCCP and Conference Setup . . . . .	<a href="#">3</a>
<a href="#">1.3</a>	SCCP and Capability Negotiation . . . . .	<a href="#">4</a>
<a href="#">2.</a>	Definition of Terms . . . . .	<a href="#">5</a>
<a href="#">3.</a>	Services of SCCP . . . . .	<a href="#">7</a>
<a href="#">3.1</a>	Conference Management . . . . .	<a href="#">7</a>
<a href="#">3.2</a>	Application Session Management . . . . .	<a href="#">7</a>
<a href="#">3.3</a>	Floor Control . . . . .	<a href="#">8</a>
<a href="#">4.</a>	Requirements for Mappings onto underlying Transports . . . . .	<a href="#">10</a>
<a href="#">5.</a>	A Model for Configuration and Capability Negotiation . . . . .	<a href="#">11</a>
<a href="#">6.</a>	SDP considerations . . . . .	<a href="#">14</a>
<a href="#">7.</a>	Security Considerations . . . . .	<a href="#">15</a>
	References . . . . .	<a href="#">16</a>
	Authors' Addresses . . . . .	<a href="#">16</a>
<a href="#">A.</a>	Message Formats . . . . .	<a href="#">18</a>
	Full Copyright Statement . . . . .	<a href="#">24</a>

Internet-Draft

sccp-services

February 2001

## [1.](#) Introduction

### [1.1](#) Overview

The Internet Multimedia Conferencing Architecture [[1](#)] currently comprises conference control elements only for loosely coupled conferences, i.e., "crowds that gather around an attraction". However, many conferences have more formal policies with respect to the underlying "social protocol" of the specific scenario. Also, it may be desirable to change parameters of the conference, e.g., set of media/applications and their parameters, in a coordinated way for all participants. Conferences that have facilities for this purpose shall be termed "tightly coupled conferences" in this document.

This document defines services for simple conference control of tightly coupled conferences. The services provided by this protocol were guided by the services offered by the ITU-T recommendations of the H.323 series [[7](#)], namely:

- o management of the set of members participating in the conference;
- o management of the set of media/application sessions that constitute the conference; and
- o floor control, which especially enables "conducted" conferences or implementation of arbitrary access control on distributed resources.

Note that this document specifies only the services to be provided but not the protocol mechanisms to be used for implementation. The latter are to be specified by different "mapping drafts" for specific transport layer techniques.

### [1.2](#) SCCP and Conference Setup

The Internet Multimedia Conferencing Architecture described in [1] provides a categorization of conference management concepts and corresponding technologies. The domain of conference management is divided into conference setup (including conference discovery) and conference course control.

While conference setup mechanisms, such as SIP [2], provide means to distribute a proper initial state to the involved end systems, the purpose of a conference course control protocol like SCCP is to manage a state during the lifetime of a conference.

However, in cases where conferences are set up with SIP, the state managed by SCCP would incorporate the initial conference state. This initial state usually includes configuration of media sessions, that

might result from negotiation processes. One element of the initial configuration of SCCP-enabled conference will be the configuration of the SCCP channel and transport mapping-specific information.

While we clearly distinguish between the roles of conference setup and conference course control there is a strong relation between these two forms of conference control and it is therefore desirable to define a way how to emerge an initial conference state (setup and distributed with SIP, SAP, or by other means) into an SCCP-state.

### [1.3](#) SCCP and Capability Negotiation

The configuration information of application sessions in the setup protocols SIP and SAP [3] is specified in a session description, in the moment this is often an SDP [4] description.

Because SDP addresses the description of conferences only, a new conference description framework is currently being defined ([6]). This work does not only address the issue of describing sessions but also the issue of capability negotiation.

In [Section 3.2](#) capability (re-)negotiation is listed as one of the functionalities provided by SCCP for application session management. [Section 5](#) presents a model for configuration and capability negotiation that is currently pursued by [6]. The refinement of the SCCP services in future versions of this document will consider this model.

## [2.](#) Definition of Terms

Participant: A human being that takes part in a conference.

Member: The system, including software and hardware, that takes part in a computer conference, representing a single participant.

End system: A host or a set of locally interconnected hosts [[1](#)] that is used as an interface to a teleconference by a single participant. The end system runs all the required conferencing software. Together with this software it constitutes a member.

SCCP entity: An instantiation of an SCCP implementation running on an end system for a single member. An SCCP entity (or entity for short) represents a specific participant in the conference using the SCCP protocol.

Conference controller: An application that interacts closely with an SCCP entity on one hand to implement the conference policy and with the participant on the other hand to realize her wishes.

UCI: A universal communication identifier of a person. Used as the E-mail address of an individual as well as the address that can be used to invite that individual via SIP [2].

Presence: A representation of the fact that an identified person is using a particular end system for the purpose of (potentially or actually) representing this person in one or more conferences. A presence corresponds to that person "being logged in" at an end system and (potentially or actually) being available for conferencing. There is a one-to-many relationship between presences and members: one presence may be member of many conferences. There is a one-to-one relationship between members and the cross-product of presences and the set of conferences this presence appears in (which cross-product contains the set of "appearances" of each presence).

Conference context: All session and membership information kept at each member of a conference which can be accessed by each SCCP entity through appropriate service requests.

Profile: An initial description of the conference, including assignment of roles to particular members, time limits for speakers, attributes of the conference (open/close, conducted/anarchic, etc).

Application session (AS), Session: The set of media agents/applications that act as peers to each other within a conference. For real-time data, this generally will be an RTP

session; for other application protocols, other horizontal protocols may define their own type of session concept. Possible synonyms are "application group" or "media agent group".

Floor context: State information about floors which can be accessed by each SCCP entity through appropriate service requests.

### [3.](#) Services of SCCP

This section describes the services provided by SCCP and realized by appropriate protocol mechanisms. Note that the latter is not within the scope of this document.

#### [3.1](#) Conference Management

SCCP is responsible for managing a conference context containing membership information of all current conference participants. The following operations are possible in the context of SCCP conference management:

invite other users: SCCP users may invite other users to participate in the current SCCP conference.

join the conference: SCCP provides to dynamically join a running conference. The conference context is updated appropriately.

leave the conference: SCCP also provides to dynamically leave a conference without disturbance. The conference context is updated appropriately.

terminate the conference: in the case of a "conducted" conference, a running conference may be terminated by the conductor resulting in a destruction of the entire conference.

obtain conference context: maintain the information stored in the conference context.

For sake of simplicity, SCCP does not provide more sophisticated features like merging, appending, or splitting entire conferences.

### [3.2](#) Application Session Management

SCCP provides functionality to manage a set of media/application sessions that constitute the conference, e.g., for real-time data this will be an RTP session [\[5\]](#).

Each media/application set is maintained in the conference context. Hence, its parameters can be obtained using appropriate conference context functions. However, it is also possible to create application sessions without registering them in the conference context due to scalability reasons.

Hence, SCCP offers the following application session management functions:

negotiating and changing the configuration of application sessions:



allows to find an appropriate configuration of one or more application sessions. Different policies for different types of conferences and different requirements for different media types have to be considered (symmetric vs. asymmetric configurations, equal rights for participants or not etc.) The negotiation and changing of the configuration can be applied on existing application sessions (re-negotiation) or on newly created application sessions. See [Section 5](#) for the description of a model for configuration and capability negotiation.

creating application sessions: defines a media/application set being identified by a unique identifier within the conference. The allowance to create application sessions depends on the conference policy, e.g., in a conducted conference only the conductor is allowed to create application sessions. The members of the application session are stored in the appropriate application session context entry.

terminating application sessions: an SCCP entity (as permitted in the conference profile) deletes an application session. The conference context is updated and the termination is signaled to the appropriate local application(s).

joining application sessions: an SCCP entity starts the application which then joins the SCCP application session. The conference context is updated appropriately by adding the application as a peer in the media/application set.

leaving application sessions: an SCCP entity terminates the local application and leave the SCCP application session. The conference context is updated appropriately.

inexact application sessions: For large conferences, it may make sense to mark an application session as "inexact", i.e., no join or leave messages are to be distributed. This may also be useful in case that application protocols are able to maintain membership information by themselves.

### [3.3](#) Floor Control

SCCP provides floor control facilities to support application state synchronization. Additionally, conductorship of conferences is also realized using the floor control functionality.

Hence, SCCP supports to map "social protocols", i.e., the rules to access application objects like audiovisual streams, onto distributed systems. However, the mapping of floors onto application semantics is not within the scope of SCCP.

Internet-Draft

sccp-services

February 2001

Each floor within SCCP is identified using a conference-unique name. The naming pattern is not within the scope of SCCP.

Hence, SCCP provides the following floor control services:

grab floor: allocates a floor for exclusive use by the requesting participant

inhibit floor: allocates a floor for non-exclusive use by several participants

release floor: releases a previously allocated floor; the state of the floor is changed accordingly

test floor: ask for the current state (F\_FREE, F\_GRABBED, F\_INHIBITED) of the floor

ask floor: ask the current floor holder to grant an exclusive floor to the requesting SCCP entity

give floor: grant an exclusive floor to another participant

holders of floor: ask for a list of current floor holders

It can be seen that the provided floor control service is very similar to the T.122 services [\[8\]](#) of the H.323 standard. However, requesting the current floor holder list is not supported by the T.122 standard.

Internet-Draft

sccp-services

February 2001

#### [4.](#) Requirements for Mappings onto underlying Transports

As previously mentioned in the introduction, this draft does not specify the mappings onto different transport layer mechanisms. However, some basic functionality is assumed by the underlying transport to provide.

The requirements for transport mappings are:

**Reliable message transport:** Transport layer services are assumed to provide reliable, consistent delivery of data units called "messages". Reliability is bounded by the fact that member end systems may find that they no longer can reliably interact with the other members, e.g., due to a network partitioning. When considering unicast transfer of messages, a "connection failure indication" is mandatory to be delivered to the SCCP layer.

**Globally ordered message delivery:** Messages are globally ordered. Each message is assigned a message number by the appropriate transport service, and messages are delivered to SCCP entities in monotonic message number order. In the rest of this document, the term "distribute" will be used to indicate that a member end system sends a message using the transport service.

## 5. A Model for Configuration and Capability Negotiation

This section provides a model for configuration and capability negotiation adopted from [\[6\]](#).

In this document, the features enabled and restricted by fixed hardware and software resources of end systems are termed "system capabilities". For example, the capability to process (or generate) audio data of a given codec format is one of the system capabilities of an audio conferencing system.

In multiparty multimedia conferences, participants employ different "components" in conducting the conference.

Example: In lecture multicast conferences one component might be the voice transmission for the lecturer, another the transmission of video pictures showing the lecturer and the third the transmission of presentation material.

Depending on system capabilities, user preferences and other technical and political constraints, different configurations can be chosen to accomplish the "deployment" of these components.

Each component can be characterized at least by (a) its intended use (i.e., the function it shall provide) and (b) a one or more possible ways to realize this function. Each way of realizing a particular function is referred to as a "configuration".

Example: A conference component's intended use may be to make

transparencies of a presentation visible to the audience on the Mbone. This can be achieved either by a video camera capturing the image and transmitting a video stream via some video tool or by loading a copy of the slides into a distributed electronic whiteboard. For each of these cases, additional parameters may exist, variations of which lead to additional configurations (see below).

Two configurations are considered different regardless of whether they employ entirely different mechanisms and protocols (as in the previous example) or they choose the same and differ only in a single parameter.

Example: In case of video transmission, a JPEG-based still image protocol may be used, H.261 encoded CIF images could be sent as could H.261 encoded QCIF images. All three cases constitute different configurations. Of course there are many more detailed protocol parameters.

In this system model, we distinguish two types of configurations:

- o potential configurations  
(a set of any number of configurations per component) indicating a system's functional capabilities as constrained by the intended use of the various components;
- o actual configurations  
(exactly one per instance of a component) reflecting the mode of operation of this component's particular instantiation.

Example: The potential configuration of the aforementioned video component may indicate support for JPEG, H.261/CIF, and H.261/QCIF. A particular instantiation for a video conference may use the actual configuration of H.261/CIF for exchanging video streams.

In summary, the key terms of this model are:

- o A multimedia session (streaming or conference) consists of one or more conference components for multimedia "interaction".
- o A component describes a particular type of interaction (e.g., audio conversation, slide presentation) that can be realized by

means of different applications (possibly using different protocols).

- o A configuration is a set of parameters that are required to implement a certain variation (realization) of a certain component. There are actual and potential configurations.
  - \* Potential configurations describe possible configurations that are supported by an end system.
  - \* An actual configuration is an "instantiation" of one of the potential configurations, i.e. a decision how to realize a certain component.

In less abstract words, potential configurations describe what a system can do ("capabilities") and actual configurations describe how a system is configured to operate at a certain point in time (media stream spec).

To decide on a certain actual configuration, a negotiation process needs to take place between the involved peers:

1. to determine which potential configuration(s) they have in common, and
2. to select one of this shared set of common potential configurations to be used for information exchange (e.g., based upon preferences, external constraints, etc.).

In SAP [3]-based session announcements on the Mbone, for which SDP was originally developed, the negotiation procedure is non-existent. Instead, the announcement contains the media stream description sent out (i.e., the actual configurations) which implicitly describe what a receiver must understand to participate.

In point-to-point scenarios, the negotiation procedure is typically carried out implicitly: each party informs the other about what it can receive and the respective sender chooses from this set a configuration that it can transmit.

Capability negotiation must not only work for 2-party conferences but is also required for multi-party conferences. Especially for the latter case it is required that the process of determining the

subset of allowable potential configurations is deterministic to reduce the number of required round trips before a session can be established.

## [6.](#) SDP considerations

This section defines how to describe conferences that make use of SCCP with SDP. Note the transport mappings for SCCP will require additional parameters to be configured and thus provide extensions to the SDP conventions presented here.

Usage of SCCP MUST be described in separate media description, where the media type of an SCCP media description is "control", i.e. the first sub-field of the m= line of the SCCP description has the value "control".

As specified in [4] the second sub-field is the transport port to which the media stream will be sent. In this case it is RECOMMENDED that for transport mappings where the concept of a transport port number is applicable the value of this field is interpreted as a port number. Even if this is not the case the second sub-field MUST contain a decimal number in the range 1024 to 65535 inclusive. If the transport mapping requires a range of port numbers to be specified the first port number MUST be followed by a "/" and the number of ports as a decimal number (as specified in [4]).

The third sub-field specifies the transport protocol. The first portion of the value MUST be set to "SCCP" followed by "/" and an identifier for the SCCP transport mechanism (to be defined in corresponding transport mapping specifications).

In SDP, the fourth sub-field is used to specify media formats as RTP payload types. For SCCP, the value "0" MUST be used.

Additional attributes that might be defined in "a=" lines are yet to be defined (or specified by transport mapping specifications.)

Example of an SCCP description in SDP:

```
m=control 30000 SCCP/XY 0
```



## 7. Security Considerations

The authentication and encryption model for SCCP will be defined in a future version of this document.

Internet-Draft

sccp-services

February 2001

## References

- [1] Handley, M., Crowcroft, J., Bormann, C. and J. Ott, "The Internet Multimedia Conferencing Architecture", Internet Draft [draft-ietf-mmusic-confarch-03.txt](#), July 2000.
- [2] Handley, , Schulzrinne, H., Schooler, and Rosenberg, "SIP: Session Initiation Protocol", Internet Draft [draft-ietf-sip-rfc2543bis-02.txt](#), November 2000.
- [3] Handley, M., Perkins, C. and E. Whelan, "Session Announcement Protocol", [RFC 2974](#), October 2000.
- [4] Handley, M. and V. Jacobsen, "SDP: Session Description Protocol", [RFC 2327](#), April 1998.
- [5] Schulzrinne, H., Casner, S., Frederick, R. and V. Jacobsen, "RTP: A Transport Protocol for Real-Time Applications", [RFC 1889](#), January 1996.
- [6] Kutscher, , Ott, and Bormann, "Requirements for Session Description and Capability Negotiation", Internet Draft [draft-kutscher-mmusic-sdpng-req-01.txt](#), November 2000.
- [7] ITU-T, "Visual Telephone Systems and Equipment for Local Area Networks with Non-Guaranteed Quality of Service", ITU-T Recommendation H.323, 2000.
- [8] ITU-T, "Multipoint communication service - Service definition", ITU-T Recommendation T.122, February 1998.

## Authors' Addresses

Carsten Bormann  
TZI, Universitaet Bremen  
Bibliothekstr. 1  
Bremen 28359  
Germany

Phone: +49.421.218-7024  
Fax: +49.421.218-7000  
EMail: [cabo@tzi.org](mailto:cabo@tzi.org)

Internet-Draft

sccp-services

February 2001

Dirk Kutscher  
TZI, Universitaet Bremen  
Bibliothekstr. 1  
Bremen 28359  
Germany

Phone: +49.421.218-7595  
Fax: +49.421.218-7000  
EMail: dku@tzi.org

Joerg Ott  
TZI, Universitaet Bremen  
Bibliothekstr. 1  
Bremen 28359  
Germany

Phone: +49.421.201-7028  
Fax: +49.421.218-7000  
EMail: jo@tzi.org

Dirk Trossen  
Nokia Research Center  
5 Wayside Road  
Burlington, MA 01803  
USA

Phone: +1.781.993-3605  
Fax: +1.781.993-1907  
EMail: dirk.trossen@nokia.com

Internet-Draft

sccp-services

February 2001

## [Appendix A](#). Message Formats

Note that this XDR-like specification does not imply any particular type of encoding of the PDUs. The encoding may be defined independently, or [RFC 1832](#) encoding may be used.

```
/* basic type definitions */
typedef string      SCCP_Name<>;
typedef opaque      SCCP_Value<>;
typedef SCCP_Name SCCP_Namelist<>;
typedef integer      SCCP_Sync;

/* Status of a floor to be given back with most requests */
enum SCCP_Status {
    F_FREE,
    F_GRABBED,
    F_INHIBITED,
    F_GIVING
};
/* Flag of an application session (see also section 3.2) */
enum SCCP_AS_Flag {
    AS_EXACT,
    AS_INEXACT
};
/* Error for requests, currently only for floor control */
enum SCCP_Error {
    SCCP_E_GRABBED,          /* floor grabbed */
    SCCP_E_INHIBITED,        /* floor inhibited */
    SCCP_E_FREE              /* floor free */
};
```

```

};

/* SCCP_AS is the application session in the conference context
* Name:      identifies the application session
* flag:      exact or inexact session description
* value:     reflects upper layer semantic
* namelist:  list of media/application sets
*/
struct SCCP_AS {
    SCCP_Name      name;
    SCCP_AS_Flag   flag;
    SCCP_Value     value;           /* upper layer semantics */
    SCCP_NameList  namelist;
};

/* SCCP_Member is the member entry in the conference context
* presence:  presence information of the member
*/
struct SCCP_Member {
    SCCP_Name      presence;

```

```

};

/* SCCP_Floor is the core entry in the floor context
* Name:      identifies the floor
* status:    status of floor
* Holders:   list of floor holders for inhibited floor
* mapping_data: implementation-dependent data to be stored for
               administration purpose
*/
struct SCCP_Floor {
    SCCP_Name      name;
    SCCP_Status    status;
    SCCP_NameList  Holders;
    SCCP_Value     mapping_data;
};

/* SCCP_Conference_Context contains list of
* - members
* - sessions
*/
struct SCCP_Conference_Context {
    SCCP_Member  members<>;

```

```

        SCCP_AS      sessions<>;
};

/* SCCP_Floor_Context contains list of floors to be maintained
   depending on the chosen mapping for realization
*/
struct SCCP_Floor_Context {
        SCCP_Floor floors<>;
};

enum SCCP_Type {
        SCCP_T_JOIN,          /* join conference */
        SCCP_T_LEAVE,         /* leave conference */
        SCCP_T_ACCEPT,        /* accept joining member */
        SCCP_T_CONTEXT,       /* context */

        SCCP_T_ASCREATE,      /* create application session */
        SCCP_T_ASDELETE,      /* delete application session */
        SCCP_T_ASJOIN,        /* join application session */
        SCCP_T_ASLEAVE,       /* leave application session */

        SCCP_T_FLOOR_GRAB,    /* grab floor */
        SCCP_T_FLOOR_INHIBIT, /* inhibit floor */
        SCCP_T_FLOOR_RELEASE, /* release floor */
        SCCP_T_FLOOR_TEST,    /* test floor */
        SCCP_T_FLOOR_ASK,     /* ask for floor */
        SCCP_T_FLOOR_GIVE,    /* give floor to other user */

```

```

        SCCP_T_FLOOR_GIVEN,    /* indication to originator */
        SCCP_T_FLOOR HOLDER,   /* ask for floor holder list */
        SCCP_T_FLOOR HOLDER_ASK, /* collect floor holder list */
        SCCP_T_FLOOR HOLDER_LIST, /* indicate floor holder list */
        SCCP_T_FLOOR_STATUS,    /* indicate floor status */
        SCCP_T_FLOOR_ERROR,     /* floor control error */

        SCCP_T_INVALID         /* last sccp pdu type */
};

struct SCCP_Join {
        SCCP_Name      presence; /* joining member */
        SCCP_Flags     flags;    /* precept etc. */
        SCCP_Value     value;    /* user data */
};

```

```

struct SCCP_Accept {
    SCCP_Name      presence;          /* joining member */
};

struct SCCP_Leave {
    SCCP_Name      presence;          /* joining member */
};

struct SCCP_Context_Msg {
    SCCP_Conference_Context  context;
    SCCP_Sync                sync;
};

struct SCCP_AS_Create {
    SCCP_Name      name;
    SCCP_Value     value;
    SCCP_Namelist  names;
};

struct SCCP_AS_Delete {
    SCCP_Name      name;
};

struct SCCP_AS_Join {
    SCCP_Name      name;
    SCCP_Name      entry;
};

struct SCCP_AS_Leave {
    SCCP_Name      name;
    SCCP_Name      entry;
};

```

```

struct SCCP_Floor_Grab {
    SCCP_Name  presence;
    SCCP_Name  floor;
};

struct SCCP_Floor_Inhibit {
    SCCP_Name  presence;
    SCCP_Name  floor;
};

```

```

};

struct SCCP_Floor_Release {
    SCCP_Name presence;
    SCCP_Name floor;
};

struct SCCP_Floor_Test {
    SCCP_Name presence;
    SCCP_Name floor;
};

struct SCCP_Floor_Ask {
    SCCP_Name presence;
    SCCP_Name floor;
};

struct SCCP_Floor_Give {
    SCCP_Name presence_given;
    SCCP_Name presence_giving;
    SCCP_Name floor;
};

struct SCCP_Floor_Given {
    SCCP_Name presence_given;
    SCCP_Name presence_giving;
    SCCP_Name floor;
};

struct SCCP_Floor_Holder {
    SCCP_Name presence;
    SCCP_Name floor;
};

struct SCCP_Floor_Holder_Ask {
    SCCP_Name presence;
    SCCP_Name floor;
    SCCP_NameList floor_holders;
};

struct SCCP_Floor_Holder_List {

```



```

        SCCP_Name floor;
        SCCP_NameList floor_holders;
};

struct SCCP_Floor_Status {
        SCCP_Object floor;
};

struct SCCP_Floor_Error {
        SCCP_Name presence;
        SCCP_Name floor;
        SCCP_Error error;
};

union SCCP_Union switch (SCCP_Type type) {

        case SCCP_T_JOIN:
                SCCP_Join                JOIN;
        case SCCP_T_ACCEPT:
                SCCP_Accept                ACCEPT;
        case SCCP_T_LEAVE:
                SCCP_Leave                  LEAVE;
        case SCCP_T_CONTEXT:
                SCCP_Context_Msg           CONTEXT;

        case SCCP_T_AS_CREATE:
                SCCP_AS_Create             AS_CREATE;
        case SCCP_T_AS_DELETE:
                SCCP_AS_Delete             AS_DELETE;
        case SCCP_T_AS_JOIN:
                SCCP_AS_Join               AS_JOIN; /* member, session */
        case SCCP_T_AS_LEAVE:
                SCCP_AS_Leave               AS_LEAVE; /* member, session */

        case SCCP_T_FLOOR_GRAB:
                SCCP_Floor_Grab            FLOOR_GRAB;
        case SCCP_T_FLOOR_INHIBIT:
                SCCP_Floor_Inhibit         FLOOR_INHIBIT;
        case SCCP_T_FLOOR_RELEASE:
                SCCP_Floor_Release         FLOOR_RELEASE;
        case SCCP_T_FLOOR_TEST:
                SCCP_Floor_Test            FLOOR_TEST;
        case SCCP_T_FLOOR_ASK:
                SCCP_Floor_Ask             FLOOR_ASK;
        case SCCP_T_FLOOR_GIVE:
                SCCP_Floor_Give            FLOOR_GIVE;
        case SCCP_T_FLOOR_GIVEN:
                SCCP_Floor_Given           FLOOR_GIVEN;
};

```

```
case SCCP_T_FLOOR_HOLDER:
    SCCP_Floor_Holder    FLOOR_HOLDER;
case SCCP_T_FLOOR_HOLDER_ASK:
    SCCP_Floor_Holder_Ask  FLOOR_HOLDER_ASK;
case SCCP_T_FLOOR_HOLDER_LIST:
    SCCP_Floor_Holder_List FLOOR_HOLDER_LIST;
case SCCP_T_FLOOR_STATUS:
    SCCP_Floor_Status     FLOOR_STATUS;
case SCCP_T_FLOOR_ERROR:
    SCCP_Floor_Error      FLOOR_ERROR;
};

struct SCCP_Message {
    SCCP_Header      hdr;
    SCCP_Union       sccp_un<>;
};
```

---

Internet-Draft

sccp-services

February 2001

## Full Copyright Statement

Copyright (C) The Internet Society (2001). All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the Internet Society or other Internet organizations, except as needed for the purpose of developing Internet standards in which case the procedures for copyrights defined in the Internet Standards process must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by the Internet Society or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

## Acknowledgement

Funding for the RFC editor function is currently provided by the Internet Society.

Bormann, et. al.

Expires August 14, 2001

[Page 24]