            Interworking SIP and Intelligent Network (IN) Applications

Status of This Memo

   This memo provides information for the Internet community. It does
   not specify an Internet standard of any kind. Distribution of this
   memo is unlimited.

IESG Note

   This RFC is not a candidate for any level of Internet Standard. The
   IETF disclaims any knowledge of the fitness of this RFC for any
   purpose, and in particular notes that the decision to publish is not
   based on IETF review for such things as security, congestion control,
   or inappropriate interaction with deployed protocols. The RFC Editor
   has chosen to publish this document at its discretion. Readers of
   this document should exercise caution in evaluating its value for
   implementation and deployment. See RFC 3932 for more information.

Abstract

   Public Switched Telephone Network (PSTN) services such as 800-number
   routing (freephone), time-and-day routing, credit-card calling, and
   virtual private network (mapping a private network number into a
   public number) are realized by the Intelligent Network (IN). This
   document addresses means to support existing IN services from Session
   Initiation Protocol (SIP) endpoints for an IP-host-to-phone call.
   The call request is originated on a SIP endpoint, but the services to
   the call are provided by the data and procedures resident in the
   PSTN/IN. To provide IN services in a transparent manner to SIP
   endpoints, this document describes the mechanism for interworking SIP
   and Intelligent Network Application Part (INAP).

Table of Contents

[1](#). Introduction

   PSTN services such as 800-number routing (freephone), time-and-day
   routing, credit-card calling, and virtual private network (mapping a
   private network number into a public number) are realized by the
   Intelligent Network. IN is an architectural concept for the real-
   time execution of network services and customer applications [[1](#)]. IN
   is, by design, de-coupled from the call processing component of the
   PSTN. In this document, we describe the means to leverage this
   decoupling to provide IN services from SIP-based entities.

   First, we will explain the basics of IN. Figure 1 shows a simplified
   IN architecture, in which telephone switches called Service Switching
   Points (SSPs) are connected via a packet network called Signaling
   System No. 7 (SS7) to Service Control Points (SCPs), which are
   general purpose computers. At certain points in a call, a switch can
   interrupt a call and request instructions from an SCP on how to
   proceed with the call. The points at which a call can be interrupted
   are standardized within the Basic Call State Model (BCSM) [[1](#), [2](#)].
   The BCSM models contain two processes, one each for the originating
   and terminating part of a call.

When the SCP receives a request for instructions, it can reply with a
single response, such as a simple number translation augmented by
criteria like time of day or day of week, or, in turn, initiate a
complex dialog with the switch. The situation is further complicated
by the necessity to engage other specialized devices that collect
digits, play recorded announcements, perform text-to-speech or
speech-to-text conversions, etc. (These devices are not discussed
here.) The related protocol, as well as the BCSM, is standardized by
the ITU-T and known as the Intelligent Network Application Part
protocol (INAP) [4]. Only the protocol, not an SCP API, has been
standardized.

```
                        +-----------+
                        |           |
                        |    SCP    |
                        |           |
                        +-----------+
                            ||
                            ||
                           /  \
                          /    \
                         / INAP \
                        /        \
                       /          \
            +--------+  ISUP    +--------+
            |  SSP   |********* |  SSP   |
            +--------+          +--------+
```
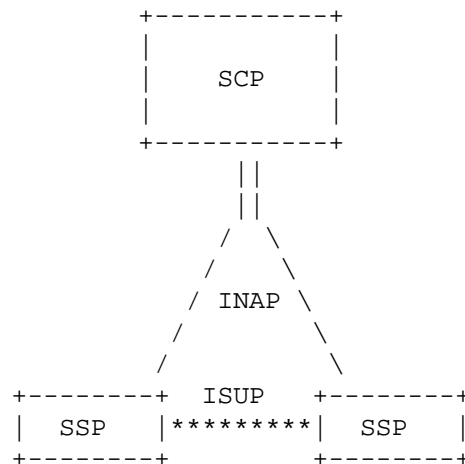
              Figure 1. Simplified IN Architecture

The overall objective is to ensure that IN control of Voice over IP
(VoIP) services in networks can be readily specified and implemented
by adapting standards and software used in the present networks.
This approach leads to services that function the same when a user
connects to present or future networks, simplifies service evolution
from present to future, and leads to more rapid implementation.

The rest of this document is organized as follows: Section 2 contains
the architectural model of an IN aware SIP entity. Section 3
provides some issues to be taken into account when performing SIP/IN
interworking (SIN). Section 4 discusses the IN service control based
on the SIN approach. The technique outlined in this document focuses
on the call models of IN and the SIP protocol state machine; Section
5 thus establishes a complete mapping between the two state machines
that allows access to IN services from SIP endpoints. Section 6
includes call flows of IN services executing on SIP endpoints. These
services are readily enabled by the technique described in this
document. Finally, Section 7 covers security aspects of SIN.

List of Acronyms

    B2BUA   Back-to-Back User Agent
    BCSM   Basic Call State Model
    CCF   Call Control Function
    DP   Detection Point
    DTMF   Dual Tone Multi-Frequency
    IN   Intelligent Network
    INAP   Intelligent Network Application Part
    IP   Internet Protocol
    ITU-T   International Telecommunications Union -
                 Telecommunications Standardization Sector
    O_BCSM   Originating Basic Call State Model
    PIC   Point in Call
    PSTN   Public Switched Telephone Network
    RTP   Real Time Protocol
    R-URI   Request URI
    SCF   Service Control Function
    SCP   Service Control Point
    SIGTRAN   Signal Transport Working Group in IETF
    SIN   SIP/IN Interworking
    SIP   Session Initiation Protocol
    SS7   Signaling System No. 7
    SSF   Service Switching Function
    SSP   Service Switching Point
    T_BCSM   Terminating Basic Call State Model
    UA   User Agent
    UAC   User Agent Client
    UAS   User Agent Server
    VoIP   Voice over IP
    VPN   Virtual Private Network

2. Access to IN-Services from a SIP Entity

   The intent of this document is to provide the means to support
   existing IN-based applications in a SIP [3] environment. One way to
   gain access to IN services transparently from SIP (e.g., through the
   same detection points (DPs) and point-in-call (PIC) used by
   traditional switches) is to map the SIP protocol state machine to the
   IN call models [1].

   From the viewpoint of IN elements such as the SCP, the request's
   origin from a SIP entity rather than a call processing function on a
   traditional switch is immaterial. Thus, it is important that the SIP
   entity be able to provide the same features as the traditional
   switch, including operating as an SSP for IN features. The SIP
   entity should also maintain call state and trigger queries to IN-
   based services, as do traditional switches.

This document does not intend to specify which SIP entity shall
operate as an SSP; however, for the sake of completeness, it should
be mentioned that this task should be performed by SIP entities at
(or near) the core of the network rather than at the SIP end points
themselves. To that extent, SIP entities such as proxy servers and
Back-to-Back user agents (B2BUAs) may be employed. Generally
speaking, proxy servers can be used for IN services that occur during
a call setup and teardown. For IN services requiring specialized
media handling (such as DTMF detection) or specialized call control
(such as placing parties on hold) B2BUAs will be required.

The most expeditious manner for providing existing IN services in the
IP domain is to use the deployed IN infrastructure as often as
possible. In SIP, the logical point to tap into for accessing
existing IN services is either the user agents or one of the proxies
physically closest to the user agent (and presumably in the same
administrative domain). However, SIP entities do not run an IN call
model; to access IN services transparently, the trick then is to
overlay the state machine of the SIP entity with an IN layer so that
call acceptance and routing is performed by the native state machine
and so that services are accessed through the IN layer by using an IN
call model. Such an IN-enabled SIP entity, operating in synchrony
with the events occurring at the SIP transaction level and
interacting with the IN elements (SCP), is depicted in Figure 2:
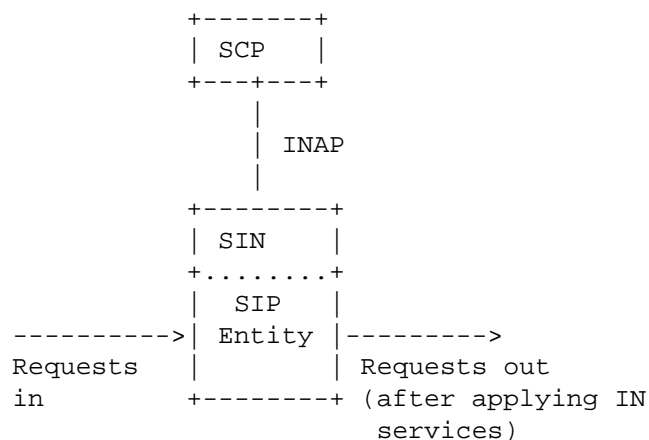
```
                     +-------+
                     | SCP   |
                     +---+---+
                         |
                         | INAP
                         |
                     +--------+
                     | SIN    |
                     +........+
                     |  SIP   |
          ---------->| Entity |--------->
          Requests   |        | Requests out
          in         +--------+ (after applying IN
                                    services)


          SIN: SIP/IN Interworking layer


          Figure 2.  SIP Entity Accessing IN Services
```

Section 5 proposes this mapping between the IN layer and the SIP
protocol state machine. Essentially, a SIP entity exhibiting this
mapping becomes a SIN-enabled SIP entity.

This document does not propose any extensions to SIP.

Figure 3 expands the SIP entity depicted in Figure 2 and further
details the architecture model involving IN and SIP interworking.
Events occurring at the SIP layer will be passed to the IN layer for
service application. More specifically, since IN services deal with
E.164 numbers, it is reasonable to assume that a SIN-enabled SIP
entity that seeks to provide services on such a number will consult
the IN layer for further processing, thus acting as a SIP-based SSP.
The IN layer will proceed through its BCSM states and, at appropriate
points in the call, will send queries to the SCP for call
disposition. Once the disposition of the call has been determined,
the SIP layer is informed and processes the transaction accordingly.

Note that the single SIP entity as modeled in this figure can in fact
represent several different physical instances in the network as, for
example, when one SIP entity is in charge of the terminal or access
network/domain, and another is in charge of the interface to the
Switched Circuit Network (SCN).

```
                  +-------+
                  |  SCP  |
                  +---o---+
                      |
                   +-----+
                         |
          *********|*********************************
          *  +-------|------------------+          *
          *  |+------o------+           |          *
          *  ||  SSF(IP)    |           |          *
          *  |+------------+            |          *
          *  ||  CCF(IP)    |           |          *
          *  |+------o------+           |          *
          *  +-------|------------------+          *
          *          |                 SIN-enabled *
          *  +-------o------------------+ SIP      *
          *  |      SIP Layer          | Entity    *
          *  +-------------------------+           *
          ********************************************
```
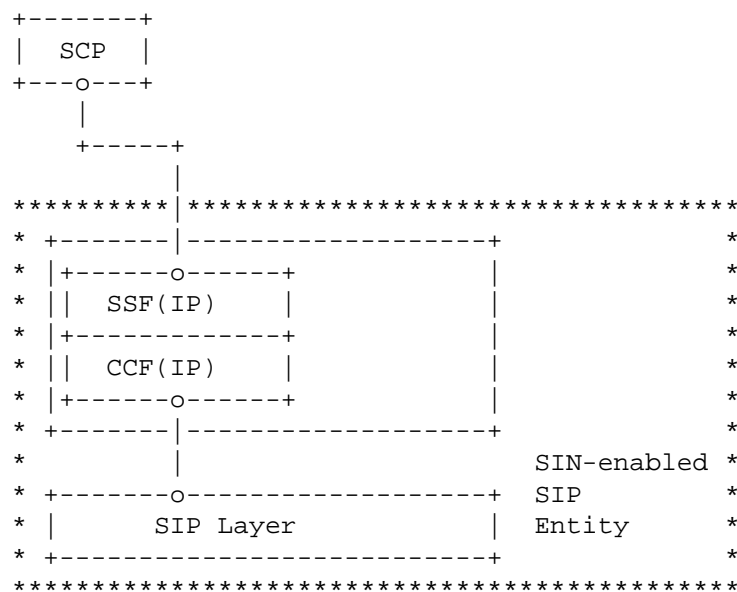
   Figure 3. Functional Architecture of a SIN-Enabled SIP Entity

The following architecture entities, used in Figure 3, are defined in
the Intelligent Network standards:

     Service Switching Function (SSF): IN functional entity that
     interacts with call control functions.

Call Control Function (CCF): IN functional entity that refers
to call and connection handling in the classical sense (i.e.,
that of an exchange).

3. Additional SIN Considerations

   In working between Internet Telephony and IN-PSTN networks, the main
   issue is to translate between the states produced by the Internet
   Telephony signaling and those used in traditional IN environments.
   Such a translation entails attention to the considerations listed
   below.

3.1. The Concept of State in SIP

   IN services occur within the context of a call, i.e., during call
   setup, call teardown, or in the middle of a call. SIP entities such
   as proxies, with which some of these services may be realized,
   typically run in transaction-stateful (or stateless) mode. In this
   mode, a SIP proxy that proxied the initial INVITE is not guaranteed
   to receive a subsequent request, such as a BYE. Fortunately, SIP has
   primitives to force proxies to run in a call-stateful mode; namely,
   the Record-Route header. This header forces the user agent client
   (UAC) and user agent server (UAS) to create a "route set" that
   consists of all intervening proxies through which subsequent requests
   must traverse. Thus SIP proxies must run in call-stateful mode in
   order to provide IN services on behalf of the UAs.

   A B2BUA is another SIP element in which IN services can be realized.
   As a B2BUA is a true SIP UA, it maintains complete call state and is
   thus capable of providing IN services.

3.2. Relationship between SCP and a SIN-Enabled SIP Entity

   In the architecture model proposed in this document, each SIN-enabled
   SIP entity is pre-configured to communicate with one logical SCP
   server, using whatever communication mechanism is appropriate.
   Different SIP servers (e.g., those in different administrative
   domains) may communicate with different SCP servers, so that there is
   no single SCP server responsible for all SIP servers.

   As Figures 1 and 2 depict, the IN-portion of the SIN-enabled SIP
   entity will communicate with the SCP. This interface between the IN
   call handling layer and the SCP is not specified by this document
   and, indeed, can be any one of the following, depending on the
   interfaces supported by the SCP: INAP over IP, INAP over SIGTRAN, or
   INAP over SS7.

   This document is only applicable when SIP-controlled Internet
   telephony devices seek to operate with PSTN devices. The SIP UAs
   using this interface would typically appear together with a media
   gateway. This document is *not* applicable in an all-IP network and
   is not needed in cases where PSTN media gateways (not speaking SIP)
   need to communicate with SCPs.

3.3. SIP REGISTER and IN Services

   SIP REGISTER provisions a SIP Proxy or SIP Registration server. The
   process is similar to the provisioning of an SCP/HLR in the switched
   circuit network. SCPs that provide VoIP based services can leverage
   this information directly. However, this document neither endorses
   nor prohibits such an architecture and, in fact, considers it an
   implementation decision.

3.4. Support of Announcements and Mid-Call Signaling

   Services in the IN such as credit-card calling typically play
   announcements and collect digits from the caller before a call is set
   up. Playing announcements and collecting digits require the
   manipulation of media streams. In SIP, proxies do not have access to
   the media data path. Thus, such services should be executed in a
   B2BUA.

   Although the SIP specification [3] allows for end points to be put on
   hold during a call or for a change of media streams to take place, it
   does not have any primitives to transport other than mid-call control
   information. This may include transporting DTMF digits, for example.
   Extensions to SIP, such as the INFO method [5] or the SIP event
   notification extension [6], can be considered for services requiring
   mid-call signaling. Alternatively, DTMF can be transported in RTP
   itself [7].

4. The SIN Architecture

4.1. Definitions

   The SIP architecture has the following functional elements defined in
   [3]:
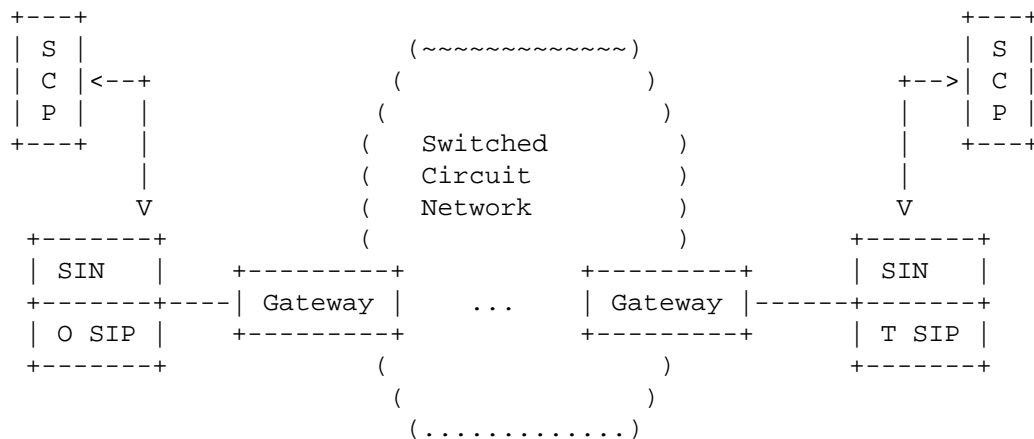
      - User agent client (UAC): The SIP functional entity that
         initiates a request.

      - User agent server (UAS): The SIP functional entity that
         terminates a request by sending 0 or more provisional SIP
         responses and one final SIP response.

      - Proxy server: An intermediary SIP entity that can act as both a
        UAS and a UAC. Acting as a UAS, it accepts requests from UACs,
        rewrites the Request-URI (R-URI), and, acting as a UAC, proxies
        the request to a downstream UAS. Proxies may retain
        significant call control state by inserting themselves in
        future SIP transactions beyond the initial INVITE.

      - Redirect server: An intermediary SIP entity that redirects
        callers to alternate locations, after possibly consulting a
        location server to determine the exact location of the callee
        (as specified in the R-URI).

      - Registrar: A SIP entity that accepts SIP REGISTER requests and
        maintains a binding from a high-level URL to the exact location
        for a user. This information is saved in some data-store that
        is also accessible to a SIP Proxy and a SIP Redirect server. A
        Registrar is usually co-located with a SIP Proxy or a SIP
        Redirect server.

      - Outbound proxy: A SIP proxy located near the originator of
        requests. It receives all outgoing requests from a particular
        UAC, including those requests whose R-URIs identify a host
        other than the outbound proxy. The outbound proxy sends these
        requests, after any local processing, to the address indicated
        in the R-URI.

      - Back-to-Back UA (B2BUA): A SIP entity that receives a request
        and processes it as a UAS. It also acts as a UAC and generates
        requests to determine how the incoming request is to be
        answered. A B2BUA maintains complete dialog state and must
        participate in all requests sent within the dialog.

4.2. IN Service Control Based on the SIN Approach

   Figure 4 depicts the possibility of IN service control based on the
   SIN approach. On both the originating and terminating ends, a SIN-
   capable SIP entity is assumed (it can be a proxy or a B2BUA). The "O
   SIP" entity is required for outgoing calls that require support for
   existing IN services. Likewise, on the callee's side (or terminating
   side), an equally configured entity ("T SIP") will be required to
   provide terminating side services. Note that the "O SIP" and "T SIP"
   entities correspond, respectively, to the IN O_BCSM and T_BCSM halves
   of the IN call model.

```
   +---+                                                      +---+
   | S |                    (~~~~~~~~~~~~~)                    | S |
   | C |<--+              (                 )            +-->| C |
   | P |   |             (                   )           |   | P |
   +---+   |            (    Switched         )          |   +---+
           |            (    Circuit          )          |
           V            (    Network          )          V
    +-------+           (                     )      +-------+
    | SIN   |    +---------+       +---------+       | SIN   |
    +-------+----| Gateway |  ...  | Gateway |------+-------+
    | O SIP |    +---------+       +---------+       | T SIP |
    +-------+           (                     )      +-------+
                        (                   )
                        (.............)
```

    O SIP: Originating SIP entity
    T SIP: Terminating SIP entity

    Figure 4. Overall SIN Architecture

5. Mapping of the SIP State Machine to the IN State Model

   This section establishes the mapping of the SIP protocol state
   machine to the IN generic basic call state model (BCSM) [2],
   independent of any capability sets [8, 9]. The BCSM is divided into
   two halves: an originating call model (O_BCSM) and a terminating call
   model (T_BCSM). There are a total of 19 PICs and 35 DPs between both
   the halves (11 PICs and 21 DPs for O_BCSM; 8 PICs and 14 DPs for
   T_BCSM) [1]. The SSPs, SCPs, and other IN elements track a call's
   progress in terms of the basic call model. The basic call model
   provides a common context for communication about a call.

   O_BCSM has 11 PICs:

   O_NULL: Starting state; call does not exist yet.
   AUTH_ORIG_ATTEMPT: Switch detects a call setup request.
   COLLECT_INFO: Switch collects the dial string from the calling party.
   ANALYZE_INFO: Complete dial string is translated into a routing
      address.
   SELECT_ROUTE: Physical route is selected, based on the routing
      address.
   AUTH_CALL_SETUP: Switch ensures the calling party is authorized to
      place the call.
   CALL_SENT: Control of call sent to terminating side.
   O_ALERTING: Switch waits for the called party to answer.
   O_ACTIVE: Connection established; communications ensue.
   O_DISCONNECT: Connection torn down.
   O_EXCEPTION: Switch detects an exceptional condition.

T_BCSM has 8 PICS:

T_NULL: Starting state; call does not exist yet.
AUTH_TERM_ATT: Switch verifies whether the call can be sent to
   terminating party.
SELECT_FACILITY: Switch picks a terminating resource to send the call
   on.
PRESENT_CALL: Call is being presented to the called party.
T_ALERTING: Switch alerts the called party, e.g., by ringing the
   line.
T_ACTIVE: Connection established; communications ensue.
T_DISCONNECT: Connection torn down.
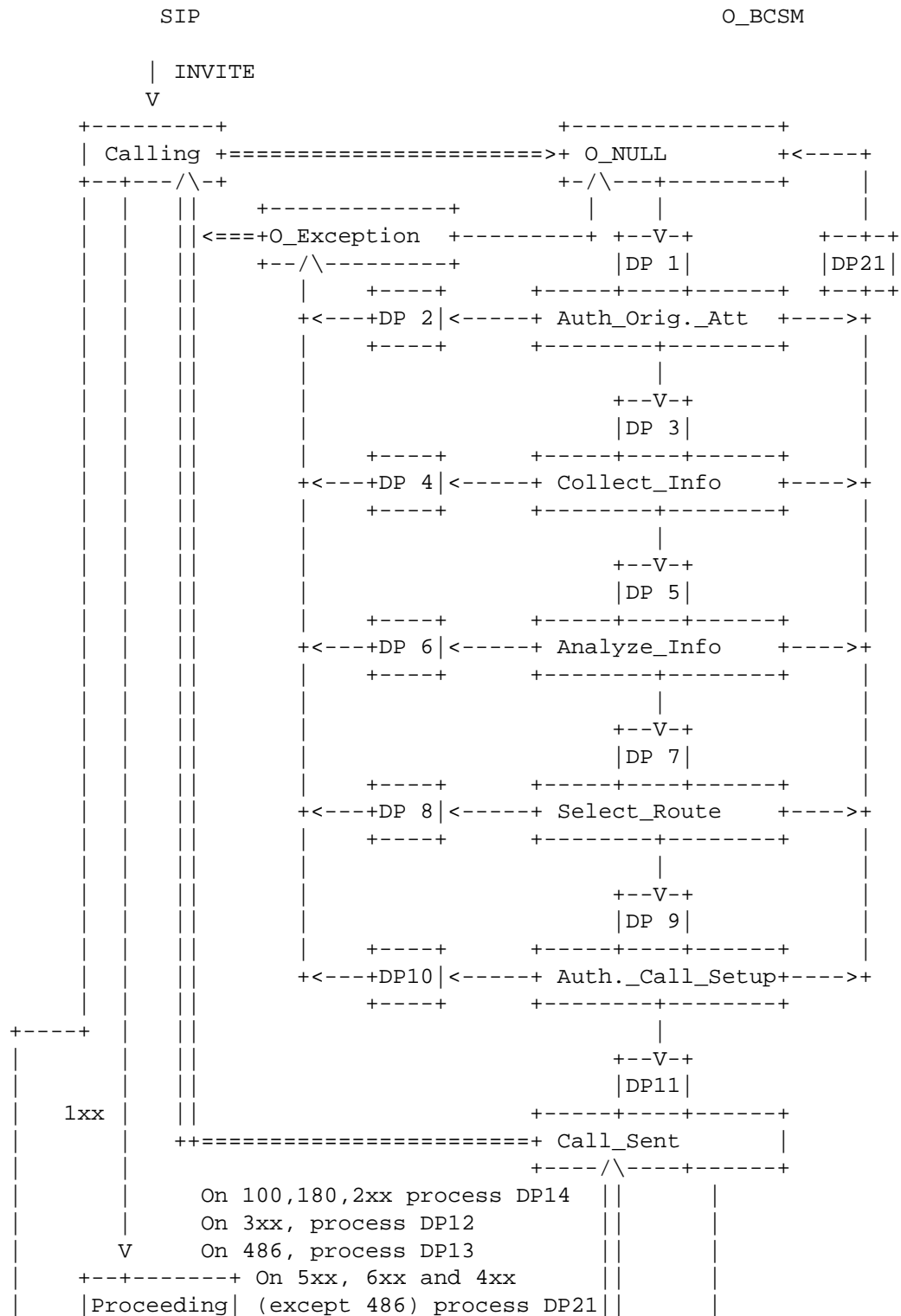T_EXCEPTION: Switch detects an exceptional condition.

The state machine for O_BCSM and T_BCSM is provided in [1] on pages
98 and 103, respectively. This state machine will be used for
subsequent discussion when the IN call states are mapped into SIP.

The next two sections contain the mapping of the SIP protocol state
machine to the IN BCSMs. Explaining all PICs and DPs in an IN call
model is beyond the scope of this document. It is assumed that the
reader has some familiarity with the PICs and DPs of the IN call
model. More information can be found in [1]. For a quick reference,
Appendix A contains a mapping of the DPs to the SIP response codes as
discussed in the next two sections.

5.1. Mapping SIP Protocol State Machine to O_BCSM

The 11 PICs of O_BCSM come into play when a call request (SIP INVITE
message) arrives from an upstream SIP client to an originating SIN-
enabled SIP entity running the IN call model. This entity will
create an O_BCSM object and initialize it in the O_NULL PIC. The
next seven IN PICs -- O_NULL, AUTH_ORIG_ATT, COLLECT_INFO,
ANALYZE_INFO, SELECT_ROUTE, AUTH_CALL_SETUP, and CALL_SENT -- can all
be mapped to the SIP "Calling" state.

Figure 5 provides a visual map from the SIP protocol state machine to
the originating half of the IN call model. Note that control of the
call shuttles between the SIP protocol machine and the IN O_BCSM call
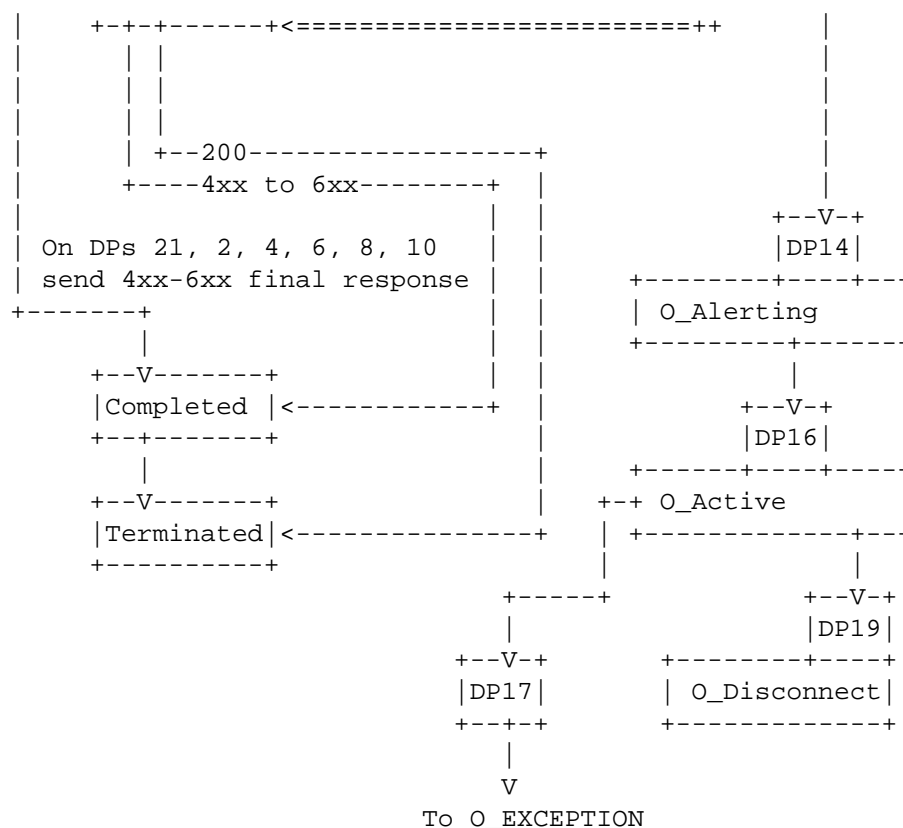model while it is being serviced.

```
          SIP                                    O_BCSM

          | INVITE
          V
     +---------+                     +--------------+
     | Calling +=====================>+ O_NULL      +<----+
     +--+---/\-+                     +-/\---+-------+    |
     |  |   ||    +------------+      |     |            |
     |  |   ||<===+O_Exception  +---------+ +--V-+       +--+-+
     |  |   ||    +--/\---------+         |DP 1|       |DP21|
     |  |   ||    |     +----+      +----+-----+-----+  +--+-+
     |  |   ||    +<---+DP 2|<----- + Auth_Orig._Att  +---->+
     |  |   ||    |    +----+      +--------+-------+    |
     |  |   ||    |                        |            |
     |  |   ||    |                     +--V-+          |
     |  |   ||    |                     |DP 3|          |
     |  |   ||    |    +----+      +-----+----+-----+    |
     |  |   ||    +<---+DP 4|<----- + Collect_Info    +---->+
     |  |   ||    |    +----+      +--------+-------+    |
     |  |   ||    |                        |            |
     |  |   ||    |                     +--V-+          |
     |  |   ||    |                     |DP 5|          |
     |  |   ||    |    +----+      +-----+----+-----+    |
     |  |   ||    +<---+DP 6|<----- + Analyze_Info    +---->+
     |  |   ||    |    +----+      +--------+-------+    |
     |  |   ||    |                        |            |
     |  |   ||    |                     +--V-+          |
     |  |   ||    |                     |DP 7|          |
     |  |   ||    |    +----+      +-----+----+-----+    |
     |  |   ||    +<---+DP 8|<----- + Select_Route    +---->+
     |  |   ||    |    +----+      +--------+-------+    |
     |  |   ||    |                        |            |
     |  |   ||    |                     +--V-+          |
     |  |   ||    |                     |DP 9|          |
     |  |   ||    |    +----+      +-----+----+-----+    |
     |  |   ||    +<---+DP10|<----- + Auth._Call_Setup+---->+
     |  |   ||    |    +----+      +--------+-------+
  +----+ |   ||    |                        |
  |    | |   ||    |                     +--V-+
  |    | |   ||    |                     |DP11|
  |    | |   ||    |                 +-----+----+------+
  |  1xx | |   ||    |                 +-----+----+------+
  |    | |   ++=====================+ Call_Sent      |
  |    | |        |                 +----/\----+------+
  |    | |    On 100,180,2xx process DP14 ||     |
  |    | |    On 3xx, process DP12         ||     |
  |    V    On 486, process DP13         ||     |
  | +--+-------+ On 5xx, 6xx and 4xx       ||     |
  | |Proceeding| (except 486) process DP21||     |
```

```
  |     +-+-+------+<=======================++       |
  |     | | |                               |        |
  |     | | |                               |        |
  |     | | |                               |        |
  |     | | +--200-----------------+        |        |
  |     +----4xx to 6xx--------+    |        |        |
  |                            |    |        +--V-+   |
  | On DPs 21, 2, 4, 6, 8, 10  |    |        |DP14|   |
  | send 4xx-6xx final response|    |    +-------+----+--+
 +-------+                     |    |    | O_Alerting   |
  |                            |    |    +---------+-----+
  +--V-------+                 |    |              |
  |Completed |<------------+   |    |          +--V-+
  +--+-------+             |   |    |          |DP16|
     |                     |   |    |    +------+----+----+
  +--V-------+             |   |    +--+ O_Active       |
  |Terminated|<--------------+   |  | +------------+--+
  +----------+             |      |  |              |
                          +-----+  |          +--V-+
                          |        |DP19|
                       +--V-+    +-------+----+
                       |DP17|    | O_Disconnect|
                       +--+-+    +-------------+
                          |
                          V
                     To O_EXCEPTION
```

Legend:

| Communication between
| states in the same
V protocol

======> Communication between IN Layer and SIP Protocol
        State machine to transfer call state

         Figure 5. Mapping from SIP to O_BCSM

   The SIP "Calling" protocol state has enough functionality to absorb
   the seven PICs as described below:

      O_NULL: This PIC is basically a fall through state to the next
      PIC, AUTHORIZE_ORIGINATION_ATTEMPT.

      AUTHORIZE_ORIGINATION_ATTEMPT: In this PIC, the IN layer has
      detected that someone wishes to make a call. Under some
      circumstances (e.g., if the user is not allowed to make calls
      during certain hours), such a call cannot be placed. SIP can
      authorize the calling party by using a set of policy directives

configured by the SIP administrator. If the called party is
authorized to place the call, the IN layer is instructed to enter
the next PIC, COLLECT_INFO through DP 3
(Origination_Attempt_Authorized). If for some reason the call
cannot be authorized, DP 2 (Origination_Denied) is processed, and
control transfers to the SIP state machine. The SIP state machine
must format and send a non-2xx final response (possibly 403) to
the upstream entity.

COLLECT_INFO: This PIC is responsible for collecting a dial string
from the calling party and verifying the format of the string. If
overlap dialing is being used, this PIC can invoke DP 4
(Collect_Timeout) and transfer control to the SIP state machine,
which will format and send a non-2xx final response (possibly a
484). If the dial string is valid, DP 5 (Collected_Info) is
processed, and the IN layer is instructed to enter the next PIC,
ANALYZE_INFO.

ANALYZE_INFO: This PIC is responsible for translating the dial
string to a routing number. Many IN services, such as freephone,
LNP (Local Number Portability), and OCS (Originating Call
Screening) occur during this PIC. The IN layer can use the R-URI
of the SIP INVITE request for analysis. If the analysis succeeds,
the IN layer is instructed to enter the next PIC, SELECT_ROUTE.
If the analysis fails, DP 6 (Invalid_Info) is processed, and the
control transfers to the SIP state machine, which will generate a
non-2xx final response (possibly 400, 401, 403, 404, 405, 406,
410, 414, 415, 416, 485, or 488) and send it to the upstream
entity.

SELECT_ROUTE: In the circuit-switched network, the actual physical
route has to be selected at this point. The SIP analogue would be
to determine the next hop SIP server. This could be chosen by a
variety of means. For instance, if the Request URI in the
incoming INVITE request is an E.164 number, the SIP entity can use
a protocol like TRIP [10] to find the best gateway to egress the
request onto the PSTN. If a successful route is selected, the IN
call model moves to PIC AUTH_CALL_SETUP via DP 9 (Route_Selected).
Otherwise, the control transfers to the SIP state machine via DP 8
(Route_Select_Failure), which will generate a non-2xx final
response (possibly 488) and send it to the upstream entity.

AUTH_CALL_SETUP: Certain service features restrict the type of
call that may originate on a given line or trunk. This PIC is the
point at which relevant restrictions are examined. If no such
restrictions are encountered, the IN call model moves to PIC
CALL_SENT via DP 11 (Origination_Authorized). If a restriction is
encountered that prohibits further processing of the call, DP 10

(Authorization_Failure) is processed, and control is transferred
to the SIP state machine, which will generate a non-2xx final
response (possibly 404, 488, or 502). Otherwise, DP 11
(Origination_Authorized) is processed, and the IN layer is
instructed to enter the next PIC, CALL_SENT.

CALL_SENT: At this point, the request needs to be sent to the
downstream entity. The IN layer waits for a signal confirming
either that the call has been presented to the called party or
that a called party cannot be reached for a particular reason.
The control is transferred to the SIP state machine. The SIP
state machine should now send the call to the next downstream
server determined in PIC SELECT_ROUTE. The IN call model now
blocks until unblocked by the SIP state machine.

If the above seven PICs have been successfully negotiated, the
SIN-enabled SIP entity now sends the SIP INVITE message to the
next hop server. Further processing now depends on the
provisional responses (if any) and the final response received by
the SIP protocol state machine. The core SIP specification does
not guarantee the delivery of 1xx responses; thus special
processing is needed at the IN layer to transition to the next PIC
(O_ALERTING) from the CALL_SENT PIC. The special processing
needed for responses while the SIP state machine is in the
"Proceeding" state and the IN layer is in the "CALL_SENT" state is
described next.

   A 100 response received at the SIP state machine elicits no
   special behavior in the IN layer.

   A 180 response received at the SIP entity enables the
   processing of DP 14 (O_Term_Seized), however, a state
   transition to O_ALERTING is not undertaken yet. Instead, the
   IN layer is instructed to remain in the CALL_SENT PIC until a
   final response is received.

   A 2xx response received at the SIP entity enables the
   processing of DP 14 (O_Term_Seized), and the immediate
   transition to the next state, O_ALERTING (processing in
   O_ALERTING is described later).

   A 3xx response received at the SIP entity enables the
   processing of DP 12 (Route_Failure). The IN call model from
   this point goes back to the SELECT_ROUTE PIC to select a new
   route for the contacts in the 3xx final response (not shown in
   Figure 5 for brevity).

A 486 (Busy Here) response received at the SIP entity enables
the processing of DP 13 (O_Called_Party_Busy) and resources for
the call are released at the IN call model.

If the SIN-enabled SIP entity gets a 4xx (except 486), 5xx, or
6xx final response, DP 21 (O_Calling_Party_Disconnect &
O_Abandon) is processed and control passes to the SIP state
machine. Since a call was not successfully established, both
the IN layer and the SIP state machine can release resources
for the call.

O_ALERTING - This PIC will be entered as a result of receiving a
200-class response. Since a 200-class response to an INVITE
indicates acceptance, this PIC is mostly a fall through to the
next PIC, O_ACTIVE via DP 16 (O_Answer).

O_ACTIVE - At this point, the call is active. Once in this state,
the call may get disconnected only when one of the following three
events occur: (1) the network connection fails, (2) the called
party disconnects the call, or (3) the calling party disconnects
the call. If event (1) occurs, DP 17 (O_Connection_Failure) is
processed and call control is transferred to the SIP protocol
state machine. Since the network failed, there is not much sense
in attempting to send a BYE request; thus, both the SIP protocol
state machine and the IN call layer should release all resources
associated with the call and initialize themselves to the null
state. Event (2) results in the processing of DP 19
(O_DISCONNECT) and a move to the last PIC, O_DISCONNECT. Event
(3) occurs if the calling party deliberately terminated the call.
In this case, DP 21 (O_Abandon & O_Calling_Party_Disconnect) will
be processed, and control will be passed to the SIP protocol state
machine. The SIP protocol state machine must send a BYE request
and wait for a final response. The IN layer releases all of its
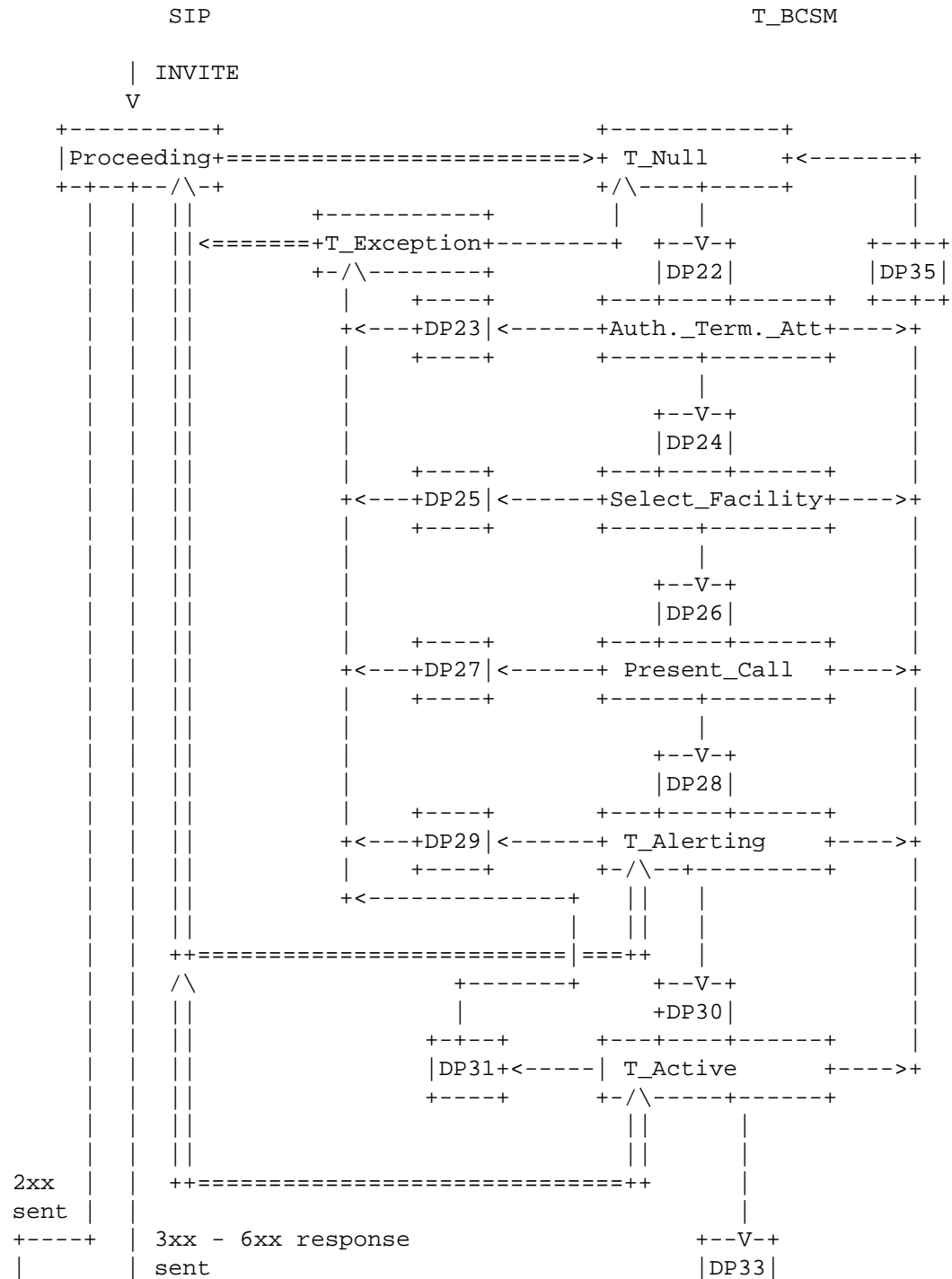resources and initializes itself to the null state.

O_DISCONNECT: When the SIP entity receives a BYE request, the IN
layer is instructed to move to the last PIC, O_DISCONNECT via DP
19. A final response for the BYE is generated and transmitted by
the SIP entity, and the call resources are freed by both the SIP
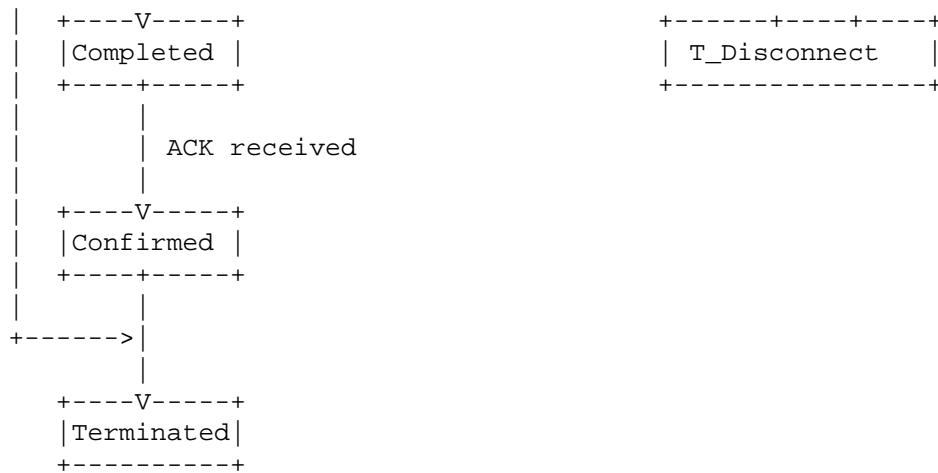protocol state machine and the IN layer.

5.2. Mapping SIP Protocol State Machine to T_BCSM

   The T_BCSM object is created when a SIP INVITE message makes its way
   to the terminating SIN-enabled SIP entity. This entity creates the
   T_BCSM object and initializes it to the T_NULL PIC.

Figure 6 provides a visual map from the SIP protocol state machine to
the terminating half of the IN call model:

```
          SIP                                       T_BCSM

         | INVITE
         V
    +----------+                         +-----------+
    |Proceeding+========================>+ T_Null    +<-------+
    +-+--+--/\-+                         +/\----+-----+        |
     | |  ||        +-----------+         |     |              |
     | |  ||<=======+T_Exception+--------+   +--V-+        +--+-+
     | |  ||        +-/\--------+        |DP22|        |DP35|
     | |  ||         |   +----+          +---+----+-----+  +--+-+
     | |  ||        +<---+DP23|<------+Auth._Term._Att+---->+
     | |  ||         |   +----+          +------+-------+     |
     | |  ||         |                          |            |
     | |  ||         |                       +--V-+          |
     | |  ||         |                       |DP24|          |
     | |  ||         |   +----+          +---+----+-----+     |
     | |  ||        +<---+DP25|<------+Select_Facility+---->+
     | |  ||         |   +----+          +------+-------+     |
     | |  ||         |                          |            |
     | |  ||         |                       +--V-+          |
     | |  ||         |                       |DP26|          |
     | |  ||         |   +----+          +---+----+------+    |
     | |  ||        +<---+DP27|<------+ Present_Call  +---->+
     | |  ||         |   +----+          +------+-------+     |
     | |  ||         |                          |            |
     | |  ||         |                       +--V-+          |
     | |  ||         |                       |DP28|          |
     | |  ||         |   +----+          +---+----+------+    |
     | |  ||        +<---+DP29|<------+ T_Alerting    +---->+
     | |  ||         |   +----+          +-/\--+---------+     |
     | |  ||        +<-------------+     ||   |            |
     | |  ||                       |     ||   |            |
     | |  ++========================|===++   |            |
     | |  /\                    +-------+    +--V-+         |
     | |  ||                    |           +DP30|         |
     | |  ||                  +-+--+    +---+----+-----+    |
     | |  ||                  |DP31+<-----| T_Active    +---->+
     | |  ||                  +----+    +-/\-----+------+
     | |  ||                           ||    |
     | |  ||                           ||    |
2xx  |  |  ++============================++    |
sent |  |                                     |
+----+  | 3xx - 6xx response              +--V-+
|       | sent                            |DP33|
```

```
|  +----V-----+                          +------+----+----+
|  |Completed |                          | T_Disconnect   |
|  +----+-----+                          +---------------+
|       |
|       | ACK received
|       |
|  +----V-----+
|  |Confirmed |
|  +----+-----+
|       |
+------>|
        |
   +----V-----+
   |Terminated|
   +----------+
```

     Legend:

      | Communication between
      | states in the same
      V protocol
      ======> Communication between IN call model and SIP
              protocol state machine to transfer call state

        Figure 6. Mapping from SIP to T_BCSM

   The SIP "Proceeding" state has enough functionality to absorb the
   first five PICS -- T_Null, Authorize_Termination_Attempt,
   Select_Facility, Present_Call, T_Alerting -- as described below:

      T_NULL: At this PIC, the terminating end creates the call at the
      IN layer. The incoming call results in the processing of DP 22,
      Termination_Attempt, and a transition to the next PIC,
      AUTHORIZE_TERMINATION_ATTEMPT, takes place.

      AUTHORIZE_TERMINATION_ATTEMPT: At this PIC, it is ascertained that
      the called party wishes to receive the call and that the
      facilities of the called party are compatible with those of the
      calling party. If any of these conditions is not met, DP 23
      (Termination_Denied) is invoked, and the call control is
      transferred to the SIP protocol state machine. The SIP protocol
      state machine can format and send a non-2xx final response
      (possibly 403, 405, 415, or 480). If the conditions of the PIC
      are met, processing of DP 24 (Termination_Authorized) is invoked,
      and a transition to the next PIC, SELECT_FACILITY, takes place.

SELECT_FACILITY: In circuit switched networks, this PIC is
intended to select a line or trunk to reach the called party. As
lines or trunks are not applicable in an IP network, a SIN-enabled
SIP entity can use this PIC to interface with a PSTN gateway and
select a line/trunk to route the call. If the called party is
busy, or if a line/trunk cannot be seized, the processing of DP 25
(T_Called_Party_Busy) is invoked, and the call goes to the SIP
protocol state machine. The SIP protocol state machine must
format and send a non-2xx final response (possibly 486 or 600).
If a line/trunk was successfully seized, the processing of DP 26
(Terminating_Resource_Available) is invoked, and a transition to
the next PIC, PRESENT_CALL, takes place.

PRESENT_CALL: At this point, the call is being presented (via the
ISUP ACM message, or Q.931 Alerting message, or simply by ringing
a POTS phone). If there was an error presenting the call, the
processing of DP 27 (Presentation_Failure) is invoked, and the
call control is transferred to the SIP protocol state machine,
which must format and send a non-2xx final response (possibly
480). If the call was successfully presented, the processing of
DP 28 (T_Term_Seized) is invoked, and a transition to the next
PIC, T_ALERTING, takes place.

T_ALERTING: At this point, the called party is being "alerted".
Control now passes momentarily to the SIP protocol state machine
so that it can generate and send a "180 Ringing" response to its
peer. Furthermore, since network resources have been allocated
for the call, timers are set to prevent indefinite holding of such
resources. The expiration of the relevant timers results in the
processing of DP 29 (T_No_Answer), and the call control is
transferred to the SIP protocol state machine, which must format
and send a non-2xx final response (possibly 408). If the called
party answers, then DP 30 (T_Answer) is processed, followed by a
transition to the next PIC, T_ACTIVE.

After the above five PICs have been negotiated, the rest are mapped
as follows:

T_ACTIVE: The call is now active. Once this state is reached, the
call may become inactive under one of the following three
conditions: (1) The network fails the connection, (2) the called
party disconnects the call, or (3) the calling party disconnects
the call. Event (1) results in the processing of DP 31
(T_Connection_Failure), and call control is transferred to the SIP
protocol state machine. Since the network failed, there is little
sense in attempting to send a BYE request; thus, both the SIP
protocol state machine and the IN call layer should release all
resources associated with the call and initialize themselves to

the null state. Event (2) results in the processing of DP 33
(T_Disconnect) and a transition to the next PIC, T_DISCONNECT.
Event (3) occurs at the receipt of a BYE request at the SIP
protocol state machine (not shown in Figure 6). Resources for the
call should be deallocated, and the SIP protocol state machine
must send a 200 OK for the BYE request (not shown in Figure 6).

T_DISCONNECT: In this PIC, the disconnect treatment associated
with the called party's having disconnected the call is performed
at the IN layer. The SIP protocol state machine sends out a BYE
and awaits a final response for the BYE (not shown in Figure 6).

6. Examples of Call Flows

Two examples are provided here to show how SIP protocol state machine
and the IN call model work synchronously with each other.

In the first example, a SIP UAC originates a call request destined to
an 800 freephone number:

    INVITE sip:18005551212@example.com SIP/2.0
    From: sip:16305551212@example.net;tag=991-7as-66ff
    To: sip:18005551212@example.com
    Via: SIP/2.0/UDP stn1.example.net
    Call-ID: 67188121@example.net
    CSeq: 1 INVITE

The request makes its way to the originating SIP network server
running an IN call model. The SIP network server hands, at the very
least, the To: field and the From: field to the IN layer for
freephone number translation. The IN layer proceeds through its PICs
and at the ANALYSE_INFO PIC consults the SCP for freephone
translation. The translated number is returned to the SIP network
server, which forwards the message to the next hop SIP proxy, with
the freephone number replaced by the translated number:

    INVITE sip:18475551212@example.com SIP/2.0
    From: sip:16305551212@example.net;tag=991-7as-66ff
    To: sip:18005551212@example.com
    Via: SIP/2.0/UDP ext-stn2.example.net
    Via: SIP/2.0/UDP stn1.example.net
    Call-ID: 67188121@example.net
    CSeq: 1 INVITE

In the next example, a SIP UAC originates a call request destined to
a 900 number:

```
INVITE sip:19005551212@example.com SIP/2.0
From: sip:16305551212@example.net;tag=991-7as-66dd
To: sip:19005551212@example.com
Via: SIP/2.0/UDP stn1.example.net
Call-ID: 88112@example.net
CSeq: 1 INVITE
```

The request makes its way to the originating SIP network server
running an IN call model. The SIP network server hands, at the very
least, the To: field and the From: field to the IN layer for 900
number translation. The IN layer proceeds through its PICs and at
the ANALYSE_INFO PIC consults the SCP for the translation. During
the translation, the SCP detects that the originating party is not
allowed to make 900 calls. It passes this information to the
originating SIP network server, which informs the SIP UAC by using a
SIP "403 Forbidden" response status code:

```
SIP/2.0 403 Forbidden
From: sip:16305551212@example.net;tag=991-7as-66dd
To: sip:19005551212@example.com;tag=78K-909II
Via: SIP/2.0/UDP stn1.example.net
Call-ID: 88112@example.net
CSeq: 1 INVITE
```

7. Security Considerations

   Security considerations for SIN services cover both networks being
   used, namely, the PSTN and the Internet. SIN uses the security
   measures in place for both the networks. With reference to Figure 2,
   the INAP messages between the SCP and the SIN-enabled SIP entity must
   be secured by the signaling transport used between the SCP and the
   SIN-enabled entity. Likewise, the requests coming into the SIN-
   enabled SIP entity must first be authenticated and, if need be,
   encrypted as well, using the means and procedures defined in [3] for
   SIP requests.

8. References

8.1. Normative References

   [1]   I. Faynberg, L. Gabuzda, M. Kaplan, and N.Shah, "The
         Intelligent Network Standards: Their Application to Services,"
         McGraw-Hill, 1997.

   [2]   ITU-T Q.1204 1993: Recommendation Q.1204, "Intelligent Network
         Distributed Functional Plane Architecture," International
         Telecommunications Union Standardization Section, Geneva.

   [3]   Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A.,
         Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP:
         Session Initiation Protocol", RFC 3261, June 2002.

8.2. Informative References

   [4]   ITU-T Q.1208: "General aspects of the Intelligent Network
         Application protocol"

   [5]   Donovan, S., "The SIP INFO Method", RFC 2976, October 2000.

   [6]   Roach, A.B., "Session Initiation Protocol (SIP)-Specific Event
         Notification", RFC 3265, June 2002.

   [7]   Schulzrinne, H. and S. Petrack, "RTP Payload for DTMF Digits,
         Telephony Tones and Telephony Signals", RFC 2833, May 2000.

   [8]   ITU-T Q.1218: "Interface Recommendation for Intelligent Network
         Capability Set 1".

   [9]   ITU-T Q.1228: "Interface Recommendation for Intelligent Network
         Capability Set 2".

   [10] Rosenberg, J., Salama, H., and M. Squire, "Telephony Routing
         over IP (TRIP)", RFC 3219, January 2002.

Appendix A: Mapping of 4xx-6xx Responses in SIP to IN Detections Points

    The mapping of error codes 4xx-6xx responses in SIP to the possible
    Detection Points in PIC Originating and Terminating Call Handling is
    indicated in the table below. The reason phrase in the 4xx-6xx
    response is reproduced from [3].

            SIP response code   DP mapping to IN
            -----------------   ---------------------
            200 OK   DP 14
            3xx   DP 12
            403 Forbidden   DP 2, DP 21
            484 Address Incomplete   DP 4, DP 21
            400 Bad Request   DP 6, DP 21
            401 Unauthorized   DP 6, DP 21
            403 Forbidden   DP 6, DP 21, DP 23
            404 Not Found   DP 6, DP 21
            405 Method Not Allowed   DP 6, DP 21, DP 23
            406 Not Acceptable   DP 6, DP 21
            408 Request Timeout   DP 29
            410 Gone   DP 6, DP 21
            414 Request-URI Too Long   DP 6, DP 21
            415 Unsupported Media Type   DP 6, DP 21, DP 23
            416 Unsupported URI Scheme   DP 6, DP 21
            480 Temporarily Unavailable   DP 23, DP 27
            485 Ambiguous   DP 6, DP 21
            486 Busy Here   DP 13, DP 21, DP 25
            488 Not Acceptable Here   DP 6, DP 21

Acknowledgments

Author's Addresses

   Vijay K. Gurbani
   Lucent Technologies, Inc.
   2000 Lucent Lane, Rm 6G-440
   Naperville, Illinois 60566
   USA
   Phone: +1 630 224 0216
   EMail: vkg@lucent.com

   Frans Haerens
   Alcatel Bell
   Francis Welles Plein,1
   Belgium
   Phone: +32 3 240 9034
   EMail: frans.haerens@alcatel.be

   Vidhi Rastogi
   Wipro Technologies
   Plot No.72, Keonics Electronics City,
   Hosur Main Road,
   Bangalore 226 560 100
   Phone: +91 80 51381869
   EMail: vidhi.rastogi@wipro.com

Full Copyright Statement

Intellectual Property

Acknowledgement