

(19) World Intellectual Property Organization  
International Bureau



(43) International Publication Date  
21 February 2008 (21.02.2008)

PCT

(10) International Publication Number  
**WO 2008/022315 A2**

(51) International Patent Classification:  
**H04M 3/00** (2006.01)

(74) Agent: **FITZGERALD, Thomas, R.**; Hiscock & Barclay, LLP, 2000 HSBC Plaza, 100 Chestnut Street, Rochester, NY 14604 (US).

(21) International Application Number:  
PCT/US2007/076213

(22) International Filing Date: 17 August 2007 (17.08.2007)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:  
60/838,208 17 August 2006 (17.08.2006) US

(81) Designated States (unless otherwise indicated, for every kind of national protection available): AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, SV, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(71) Applicant (for all designated States except US): **RED-COM LABORATORIES, INC.** [US/US]; One Redcom Center, Victor, NY 14564 (US).

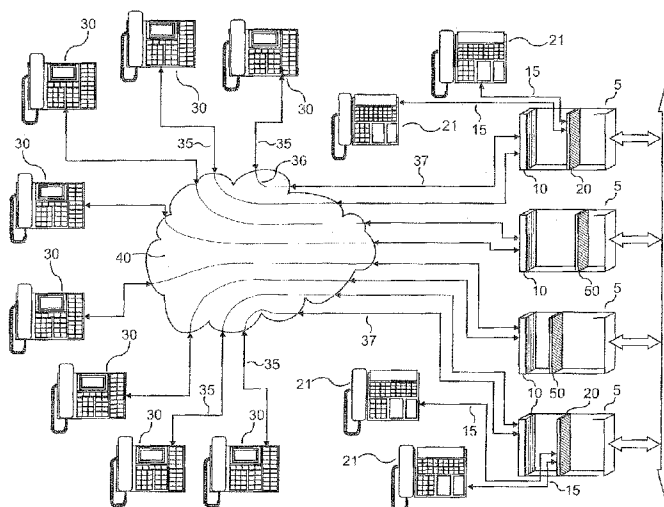
(84) Designated States (unless otherwise indicated, for every kind of regional protection available): ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IS, IT, LT, LU, LV, MC, MT, NL, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

(72) Inventors; and

(75) Inventors/Applicants (for US only): **DELMEGE, James, W.** [US/US]; 37 Berry Lane, North Chili, NY 14514 (US). **REYNOLDS, Paul, A.** [US/US]; 3235 Woodfield Drive, Walworth, NY 14568 (US).

[Continued on next page]

(54) Title: VOIP TELECOMMUNICATIONS SWITCH



(57) Abstract: An exploitation of the proper classification of conventional call-processing states into categories, each category corresponding to one of the call states of the SIP finite- state machine (FSM), yields simplified design, implementation, and operation of heterogeneous call processing. A separate FSM for the S I P- categorized call states, running in parallel with a conventional port event processing (PEP) FSM, implements a VoIP Local Call Controller by passing events between the SIP-type FSM and the PEP FSM only at the points where a transition between SIP categories occurs. The SIP FSM does not require notification of PEP FSM transitions not affecting the SIP call state. For this reason, none of the code associated with the transitions between PEP FSM states within the same major state category requires any modification. Consequently the integration of VoIP and PEP call processing can be done more simply, and at dramatically-reduced cost in software development, support, and maintenance.

WO 2008/022315 A2



**Published:**

- *without international search report and to be republished  
upon receipt of that report*

## **VoIP TELECOMMUNICATIONS SWITCH**

### **CROSS-REFERENCE TO RELATED APPLICATIONS**

Reference is made to and priority claimed from U.S. Provisional Application Ser. No. 60/838,208, filed August 17, 2006, and its entire contents is hereby incorporated by reference.

### **FIELD OF THE INVENTION**

The invention concerns the processing of telephone calls, and more specifically the processing of telephone calls over the Internet using the VoIP (Voice over Internet Protocol) technology and conventions.

### **BACKGROUND OF THE INVENTION**

Any telephone call progresses through a series of steps from beginning to end. Each step is identified with a state or status of the call, and the progress from one step to the next in a call is termed a transition. For example, a subscriber picks up a phone handset to make a call, and the call shifts from an idle state to a dial tone ('tone') state. When the subscriber begins to dial or press in a number, the call shifts from the tone state to a dialing ('dial') state. When the switching network locates the called line and begins sending ring signals to the receiving phone, the call state shifts from 'dial' to 'ringing'. This process continues until the call is ended and the state returns to 'idle'. These changes from state to state are the transitions of the state of the call.

In conventional call processing technologies, all the system intelligence for processing telephone calls has resided in the central switching element, for example, the High Density Exchange (HDX), a high-density version of the system originally covered in U.S. Patent No. 4,228,536, said patent incorporated herein by reference, and hereinafter referred to as "the '536 patent". The endpoint instruments in conventional call processing have been 'dumb', i.e., they do not contain any call processing elements.

Central switching elements in conventional call processing are not restricted to single physical components. For example, in systems using Modular Switching Peripheral (MSP) architecture, also known in the industry as Computer/Telephony Integration, or CTI, the MSP together with the host computer constitute the central switching element. Although ISDN (Integrated Services Digital Network) telephones have some intelligence, call control still resides almost exclusively with the central switching element.

VoIP transforms the conventional switching architecture. Simply by purchasing some VoIP telephones and connecting them to an existing Ethernet LAN, a provider can construct a basic VoIP telephone network. The provider can configure the phones either manually or through a simple Web-based interface. As a result, the provider can supply a private phone system without any special software or hardware anywhere else in the network, so that the phones can dial, ring, and talk to one another. In such an architecture, the phones contain all the intelligence needed to make the VoIP network operate.

As the VoIP system scales up in size, it obtains advantages from the reintroduction of a central switching element to facilitate system administration and provision of certain features. But in contrast to practice in conventional telephone networks, the reintroduced central switching element, often called a Call Manager or LCC (Local Call Controller), operates with the VoIP phones principally at a peer-to-peer level.

Integrating a VoIP system with a mature, complex, and feature-rich system utilizing centralized call processing, such as the HDX platform, presents unique challenges. Attempting to integrate existing centralized call processing logic with decentralized VoIP call processing makes control of a call an issue between the central switching element and the telephone itself.

Call processing is implemented using a finite state machine (FSM), a design structure well-known in the art, in which a call is always in one of a finite number of discrete states (e.g., Dialing, Ringing, Talking, and others), and an event occurring in the system can trigger an action to cause the call to progress from one state to another.

The HDX call-processing FSM, also called Port Event Processing (PEP), currently has hundreds of call states and thousands of software components used for processing the events which occur in each of these states. The integration of VoIP with the HDX would, under the application of ordinary skill in the art, require a system builder to rework most of the software components comprising the FSM (PEP).

The conventional way to implement a new feature within a call processing FSM is to begin by examining carefully each and every call state, event, and transition routine. "Transition routine" is the name for the actual code which executes when a given event is received for a port in a given state. A transition routine performs some action appropriate to the event and the current state, and its action may include a transition to a different state.

The next step in conventional feature implementation is the design and implementation of code for one or more existing transition routines, and possibly the addition of new states and events as well, together with new transition routines to process the new states and events. Every state transition in the FSM represents added exposure to potential software bugs whenever an incorrect modification is made in support of the new feature, or whenever a required change is erroneously overlooked. The large number of state transitions therefore makes feature implementation in the conventional way a highly error-prone task, raising its cost considerably. Avoidance of such extensive rework of software would be highly advantageous.

Furthermore, PEP/HDX integration, done properly, would offer providers a clear, smooth path between powerful, well-established, but less-flexible conventional call processing systems and the highly-flexible VoIP systems offering all the potential of the Internet. Opening such a path would grant telephony system providers clear advantages over either technology alone.

## SUMMARY

Since the Session Initiation Protocol (SIP) standard for voice-over-Internet Protocol (VoIP) only distinguishes major call states, its finite state machine (FSM) is

relatively simple. In contrast, the conventional call-processing FSM distinguishes up to hundreds of distinct states for any call. The present invention exploits the proper classification of conventional call-processing states into categories each corresponding to the major call states such as "idle", "dialing", "ringing" and "talking". These major call states are similar but not identical to the SIP FSM's call states. The invention provides a separate FSM for the SIP-categorized call states, running in parallel with a conventional port event processing (PEP) FSM and implements a VoIP LCC by passing events between the SIP-type FSM and the PEP FSM only at the points where a transition between major call state categories occurs.

Using the invention, the SIP FSM does not require notification of PEP FSM transitions not affecting the SIP call state. Similarly, the PEP FSM does not require notification of SIP FSM transitions not affecting the PEP FSM call state. Transitions between major states occur much less frequently than transitions between PEP FSM states within the same major call state category as classified by the invention. For this reason, none of the code associated with the transitions between PEP FSM states within the same major call state category requires any modification. Consequently the integration of VoIP and PEP call processing can be done using the present invention more simply, and at dramatically-reduced cost in software development, support, and maintenance.

### **BRIEF DESCRIPTION OF THE DRAWINGS**

Fig. 1 is an overall block diagram of the preferred embodiment showing a multi-shelf system and a number of connected analog and IP phones.

Fig. 2 is an expanded view of a single MSU showing the salient software elements.

Fig. 3 shows the grouping of port states into separate classes.

Fig. 4 is a flow chart of the invention's logic required for handling a port state transition.

Fig. 5 is an overall view of the PEP finite state machine.

Fig. 6 is an overall view of the SIP finite state machine.

## DETAILED DESCRIPTION OF THE INVENTION

Fig. 1 shows a plurality of Modular Switching Units (MSUs) 5. The present invention functions with one MSU in smaller configurations, and multiple MSUs in larger configurations. The MSU 5 functions generally as disclosed in the '536 patent, but as adapted for the present invention the port capacity and control processor (Controller) capabilities have been significantly improved. The Controller 10 in Fig. 1 corresponds to the MPU function in the '536 patent.

In Fig. 1, analog lines 15 are shown connected to time-division multiplexed (TDM) port interfaces 20. In the case of analog lines the TDM port interfaces 20 correspond to the line circuits of the '536 patent. Each analog line connection is a physical connection of the two conductors needed for an analog telephone circuit, commonly referred to as the 'tip' and 'ring' conductors.

Also in Fig. 1, Internet protocol (IP) phones 30 are shown connected 35, 36, 37 to the various Controllers 10. In the case of IP phones 30, there is no individual physical connection between the various IP phones 30 and the Controllers 10. Instead, there is a single physical connection extending from each phone 30 and from each Controller 10 to an IP routing infrastructure such as a local area network (LAN) 40. Each line 35, 36, 37 shown in Fig. 1 connecting an IP phone 30 with a controller 10 indicates a logical association.

Each IP phone 30 has a priori the IP address of a specific Controller 10 to which it periodically directs a SIP REGISTER message. By means of this REGISTER message the controller 10 is able to identify, authenticate, and communicate with the IP phone 30, so that the logical connection amounts to a physical connection as far as communication of SIP messages is concerned. The SIP protocol is a standard issued by the Internet Engineering Task Force (IETF) and defined in the IETF's Request for Comment (RFC) 3261, incorporated herein by reference. The IP routing infrastructure is well-known in the art, and comprises the network hardware and software, including routers, hubs, and

gateways and their supporting programs, needed for communication using Internet protocols. In Fig. 1, the IP routing infrastructure comprises a local area network that uses the Ethernet protocol.

The present invention retains support of calls originating at an IP phone and terminating at an analog phone, or originating at an analog phone and terminating at an IP phone. Such call support is provided via the Media Servers 50 shown in Fig. 1. Media Servers 50 are controlled by the MSU Controller 10 in the same manner as any other port circuit is controlled.

Media Servers (also called Media gateways) actually look like another kind of port interface, and are directly connected to the controller unit computer system, just like the other port interfaces. Logical connections 35, 36, 37 exist between VoIP phones 30 and controllers 10. When a call is up that does NOT require conversion from IP to TDM, no port interface is needed. But when a call is up that involves both a VoIP phone 30 and an analog or digital phone 21 connected to a port interface 20, then an additional logical connection will exist from the VoIP phone 30 through the IP network 40 to one of the Media Servers or gateways 50. This second logical connection handles the Real-Time-Protocol (the actual voice data) for the call, and the Media Server acts as a port interface in this case.

Although not shown in Fig. 1, other circuits of the general types disclosed in the '536 patent may also occur in the network shown.

The association or assignment of an IP phone to a specific Controller is arbitrary and in no way limits the capabilities of the system. The main objective is to distribute the IP phones as uniformly as possible among the Controllers in order to reduce the impact of a service outage resulting from the loss of an MSU.

Fig. 2 depicts the salient software elements of the Controller. The System Scheduler 12 directs the controller's processor to execute the SIP FSM process (SIPFSM) 14 when required, and the PEP FSM process (PEPFSM) 24 when required. The occurrence of an event that alters one or more conditions requiring processing by a specific FSM causes that FSM to be scheduled for execution.

The SIPFSM 14 receives call events from the UDP and TCP layers 15, 16 of the Internet protocol. Said events represent primarily SIP messages received from IP phones 30 associated with the Controller with which the IP phone has registered. The SIPFSM 14 also receives events from a second source, the PEPFSM 24, via the connection 34 shown in Fig. 2. Finally, the SIPFSM 14 receives events from a third source, the timers within the SIPFSM itself, marking the ends of time intervals, or delays.

Similarly, the PEPFSM 24 receives events from the Port Driver layer 25. These represent primarily port-related messages received from the various port interfaces associated with the Controller 10 in a particular MSU 5. The PEPFSM 24 also receives events from a second source, the SIPFSM 14 via the connection 44, as shown in Fig. 2. In the same manner as for the SIPFSM, the PEPFSM receives events from a third source, the timers within the PEPFSM itself, marking the ends of time intervals, or delays.

In single and multiple MSU systems, the PEPFSM sends and receives inter-process messages (IPMs). In multiple MSU systems, the IPMs are carried on Inter-MSU link 26 as shown in Fig. 2. IPMs signal events between separate instances of the PEPFSM in different MSUs.

Typical instances of port state classes, as determined by events occurring during a call, are: IDLE, TALKING, RINGING, TONE, and HOLD. The actual state classes existing in a given system may vary depending on the particular applications for which the system is designed. A list of port state classes is presented here as typical – within each port state class, there may be numerous distinct port states.

UNKNOWN

OFF\_LINE

IDLE

IN\_USE

TRANSITION

SEIZE

RELEASE

TONE  
TALK  
RINGING  
DIAL  
HOLD  
EQU\_BUSY  
RING\_BACK  
BUSY\_TONE  
AWAIT\_ATND  
MAGNETO\_CRANK

Fig. 3 shows a pair of PEPFSM state groupings, each state grouping comprising a port state class 70, 71. Each port state class 70, 71 contains one or more port states, Port state class 70 contains port states 72, 72a. Port state class 71 contains port states 73, 73a distinct from all port states 72, 72a of port state class 70. A system's FSM has exactly one port state, and therefore exactly one port state class, at a time. A particular call event occurring when the FSM is in a first port state 72 brings about a transition 720 from the first port state 72 to a second port state 72. Some events occurring when the FSM is in a first port state 72a bring about transitions 740 from a first port state class 70 of the first port state 72a to a second port state class 71 and a second port state 73a.

The first port state class 70 of Fig. 3 is shown as "A", which might correspond to the TALKING state, and the second port state class 71 of Fig. 3 is shown as "B", which might correspond to the RINGING state, but other port state classes may be defined and used, each port state class containing a plurality of port states, so that the depiction of Fig. 3 should not be taken in any limiting sense.

Fig. 4 shows the method of the present invention. The PEPFSM is shown as waiting (101) for an event to process. When an event is available, the System Scheduler 12 (Fig. 2) causes the PEPFSM to resume execution at the step "Get next PEP event" 103. At step 105 the invention sets a variable "begin\_class" associated with the current

port to the port class state of the current port's state (variable "current\_port\_state"). The variable "begin\_class" is associated with the current port, and is located in the call's port record 27 of Fig. 5. The invention's retention of the current port class state provides for detection of transitions between port class states whenever they occur.

The event is then dispatched (107) for transition. Dispatching means that the invention uses the combination of the type of event occurring and the current port state to invoke a specific transition routine (109). The processing taking place within the transition routine (109) may change the current port's state (variable "current\_port\_state"). When the transition routine returns after completion (111), the invention sets (113) a variable "end\_class" associated with the current port to the port class state of the current port's state (variable "current\_port\_state"). The variable "end\_class" is associated with the current port, and is located in the call's port record 27 of Fig. 5.

The invention then compares (115) the current port's "begin\_class" value with the current port's "end\_class" value. If the two values are unequal, the invention sends (117) the event to the SIPFSM to initiate the SIPFSM processing of the change of port state. If the two values are equal, the invention bypasses all SIPFSM processing and awaits the next PEP event.

SIPFSM and PEPFSM processes are quite similar in design and structure. Fig. 5 illustrates the use of port records (27) to store call states for the PEPFSM, and Fig. 6 shows the use of session records (17) to store call state classes for the SIPFSM.

The invention's port class comparison, a single test performed in one place for all port state transitions, eliminates the design, implementation, testing, and operation of like tests within each PEPFSM transition routine. In this way a myriad of PEPFSM transition routines are relieved of the task of communicating with the SIPFSM.

The invention's loose coupling of the PEPFSM and the SIPFSM thereby allows both state machines to function more coherently, and simplifies software maintenance due to the reduced likelihood of introducing bugs in the SIP protocol by making non-SIP-related changes to PEP transition routines within any port state class.

While the invention has been described with reference to preferred embodiments, it will be understood by those skilled in the art that various changes may be made and equivalents may be substituted for elements thereof to adapt to particular situations without departing from the scope of the invention. Therefore, it is intended that the invention not be limited to the particular embodiments disclosed as the best mode contemplated for carrying out this invention, but that the invention will include all embodiments falling within the scope and spirit of the appended claims.

**CLAIMS**

1. An apparatus for managing calls in a telephone system having switch-accessed phones and voice over internet (VoIP) phones comprising:
  - a controller with a plurality of ports, said ports coupled to the switch-accessed phones and coupled to a network of VoIP phones,
  - said controller having first and second finite state machines,
  - the first finite state machine for processing all calls and having a set of classes , each class having one or more states for each major state,
  - the second finite state machine coupled to the VoIP ports and having the same classes as the first finite state machine;
  - scheduling software for communicating a change in a class of a phone coupled to one machine to the finite state machine coupled to the other phone.
2. The apparatus of claim 1 wherein the controller has one or more port interface units, each port interface unit controlled by the controller and connected to conventional digital or analog subscriber lines for controlling phone calls to or from a conventional subscriber line.
3. The apparatus of claim 1 wherein the controller has one or more Internet protocol ports for connecting to an Internet infrastructure network.
4. The apparatus of claim 3 wherein the controller registers one or more IP phones on the IP infrastructure for receiving periodic messages from the IP phones to thereby establish logical communication connections between phones connected to or registered with the controller and the IP phones via the IP infrastructure by identifying and authenticating the registered IP phones.
5. The apparatus of claim 1 wherein the controller executes class transitions for all phone and all state transitions for the conventional phones.

6. A modular switching unit for interconnection to other like modular switching units for providing telephone service using voice-over-Internet protocol (IP) and conventional digital and analog telephony protocols, comprising:

- a controller comprising a processor, memory, operating system and one or more application programs including state machine programs;

- one or more port interface units, each port interface unit controlled by the controller and connected to conventional digital or analog subscriber lines for controlling phone calls to or from a conventional subscriber line;

- one or more IP ports for connecting to an Internet protocol infrastructure network;

- one or more media gateways for connecting the port interface units to the controller;

- said controller further comprising

- a first finite-state software program for processing class and state transitions to and from conventional subscriber lines connected to the port interfaces and for executing class transitions to and from telephones logically coupled to the IP ports,

- a second finite-state software program for tracking class transitions on IP phones connected to the IP ports, and

- a scheduling software program for scheduling and coordinating class transitions between conventional phones and IP phones.

7. An apparatus for processing calls between voice-over-Internet protocol (IP) phones and conventional switched-access phones:

- means for connecting one or more conventional switched-access phones to a modular switching unit with a controller for controlling communications to and from the conventional phones;

- means for connecting the controller to an IP infrastructure;

- means for registering one or more IP phones on the IP infrastructure with the controller;

- means for receiving periodic message from the IP phones at the controller for establishing logical communication connections between phones connected to or

registered with the controller and the IP phones via the IP infrastructure by identifying and authenticating the registered IP phones;

said controller establishing a logical communication connection between a conventional phone and an IP phone;

said controller having a first finite state machine to process all calls, said first finite state machine having classes for all phones and one or more states for conventional phones;

said controller having a second finite state machine for tracking class changes of IP phones;

said controller having a scheduler program for sending a message from one finite state machine to the other finite state machine when either the IP phone or the conventional phone changes class; and

said controller processing the call between to the two phones by operating the finite state machines in accordance with transitions among classes of all phones.

8. The apparatus of claim 7 wherein the controller executes transitions between states of said conventional phones.

9. A switching telecommunications system for providing telephone service using both voice-over-Internet protocol and conventional digital and analog telephony protocols, comprising:

one or more modular switching units for interconnecting telephone subscriber lines;

one or more port interface units, each one for connecting one or more conventional digital or analog subscriber lines each to a modular switching unit;

one or more controller unit computer systems for connecting one or more voice-over-Internet lines each to a modular switching unit;

one or more media gateways for connecting the port interface units to the controller unit computer systems;

one or more digital telecommunications networks, each using an Internet set of protocols, connected to the controller unit computer;

one or more Internet-protocol telephones, each connected to one or more of the digital telecommunications networks;

one or more conventional digital or analog telephones, each connected to a conventional telephone network;

a first finite-state software program for processing call class transitions in telephone calls involving conventional digital or analog telephones or Internet Protocol telephones and processing call state transitions in conventional digital or analog telephones;

a second finite-state software program for processing call class transitions in the Internet Protocol associated with said Internet-protocol telephones;

a scheduling software program for scheduling and coordinating the first and second finite-state software programs based on the call state transitions in each telephone call;

an instance of the first finite-state software program operating in each controller unit computer system for processing all class and state transitions;

an instance of the second finite-state software program operating in each controller unit computer system; and

an instance of the scheduling software program operating in each controller unit computer system.

10. The system of claim 9 wherein the scheduling software program further comprises:

a software component for detecting all call class transitions;

a software component for detecting call state transitions requiring the use of the first-state software program;

a software component for invoking the first finite-state software program; and

a software component for invoking the second finite-state software program.

11. A method for processing telephone calls involving voice-over-Internet telephones and conventional telephones, comprising the steps of:

classifying call state transitions for conventional telephone calls into state transitions which alter the call state class of a telephone call and state transitions which do not alter the call state class of a telephone call;

implementing a first finite-state software program for class transitions for all calls and for implementing state transitions for conventional calls;

operating the first finite-state software program to control and process all calls including processing all class and state changes in conventional phones and class changes in voice-over-Internet phones;

implementing a second finite-state software program to tracking class changes of calls connecting a conventional subscriber line with a voice-over-Internet telephone;

exchanging messages between the two finite-state software programs when either phone transitions from one class to another class.

12. The method of claim 11 wherein the step of processing any telephone call involving both voice-over-Internet telephones and conventional telephones using the first and second finite-state software programs further comprises the steps of:

receiving a call event;  
storing the current state of the call's port as a beginning state class;  
processing the call event using the first finite-state software program;  
storing the current state of the call's port as an ending state class;  
comparing the beginning state class to the ending state class; and  
only when the beginning state class is not the same as the ending state class,  
posting an event to the second finite-state software program.

13. A method for processing calls between voice-over-Internet protocol (IP) phones and conventional switched-access phones:

connecting one or more conventional switched-access phones to a modular switching unit with a controller for controlling communications to and from the conventional phones;

connecting the controller to an IP infrastructure;

registering one or more IP phones on the IP infrastructure with the controller;

receiving periodic messages from the IP phones at the controller for establishing logical communication connections between phones connected to or registered with the controller and the IP phones via the IP infrastructure by identifying and authenticating the registered IP phones;

running a first finite state machine in the controller to process all calls, said first finite state machine having a group of classes for all calls and group of states which do not alter the class of conventional phones;

running a second finite state machine in the controller for tracking class transitions of IP phones corresponding to the class transitions of the conventional phones;

sending a message from one finite state machine to the other finite state machine when either phone changes a class; and

executing transitions among said classes and states for said conventional phones and executing transitions between classes when a conventional phone is logically connected to an IP phone.

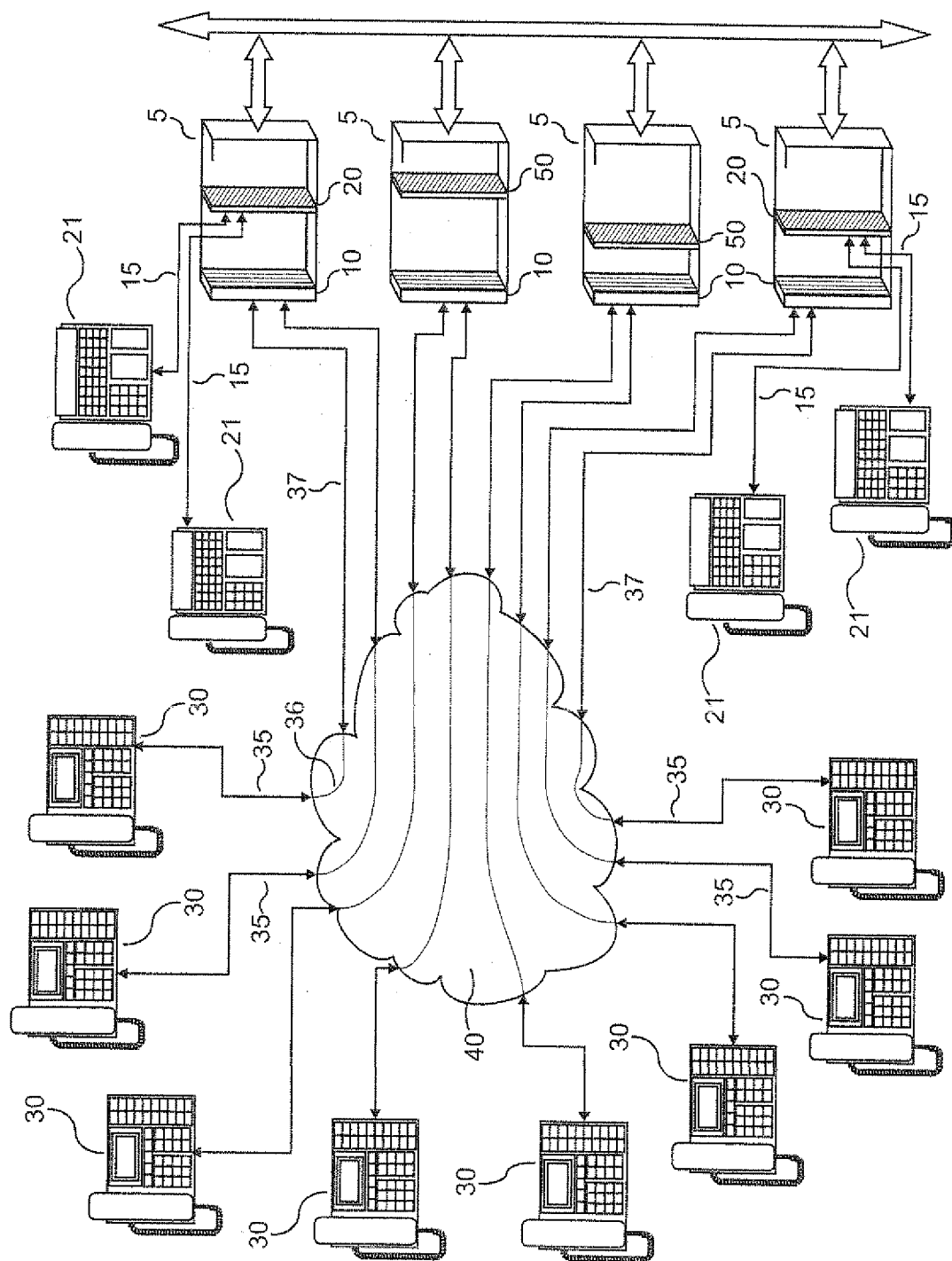


FIG. 1

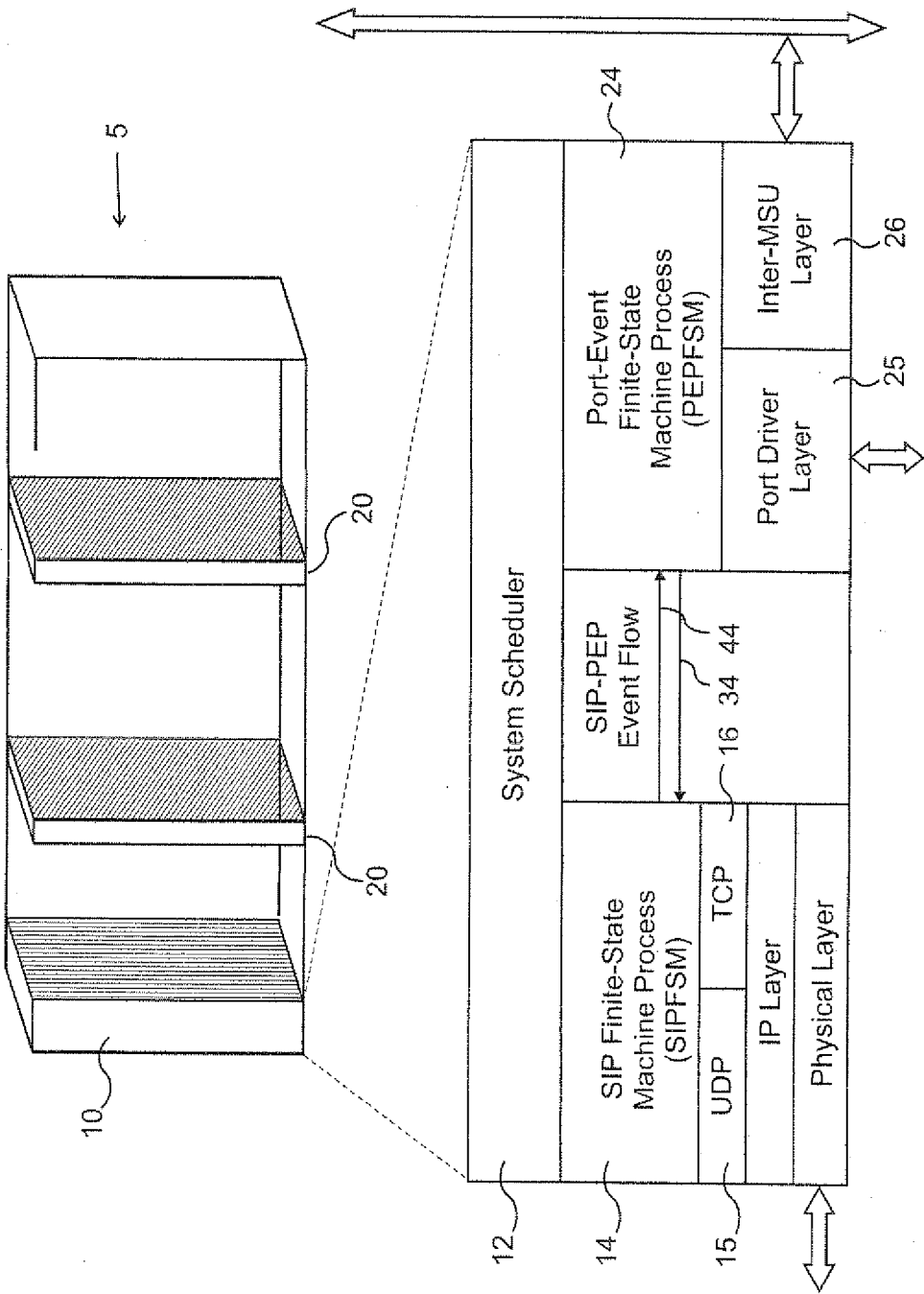


FIG. 2

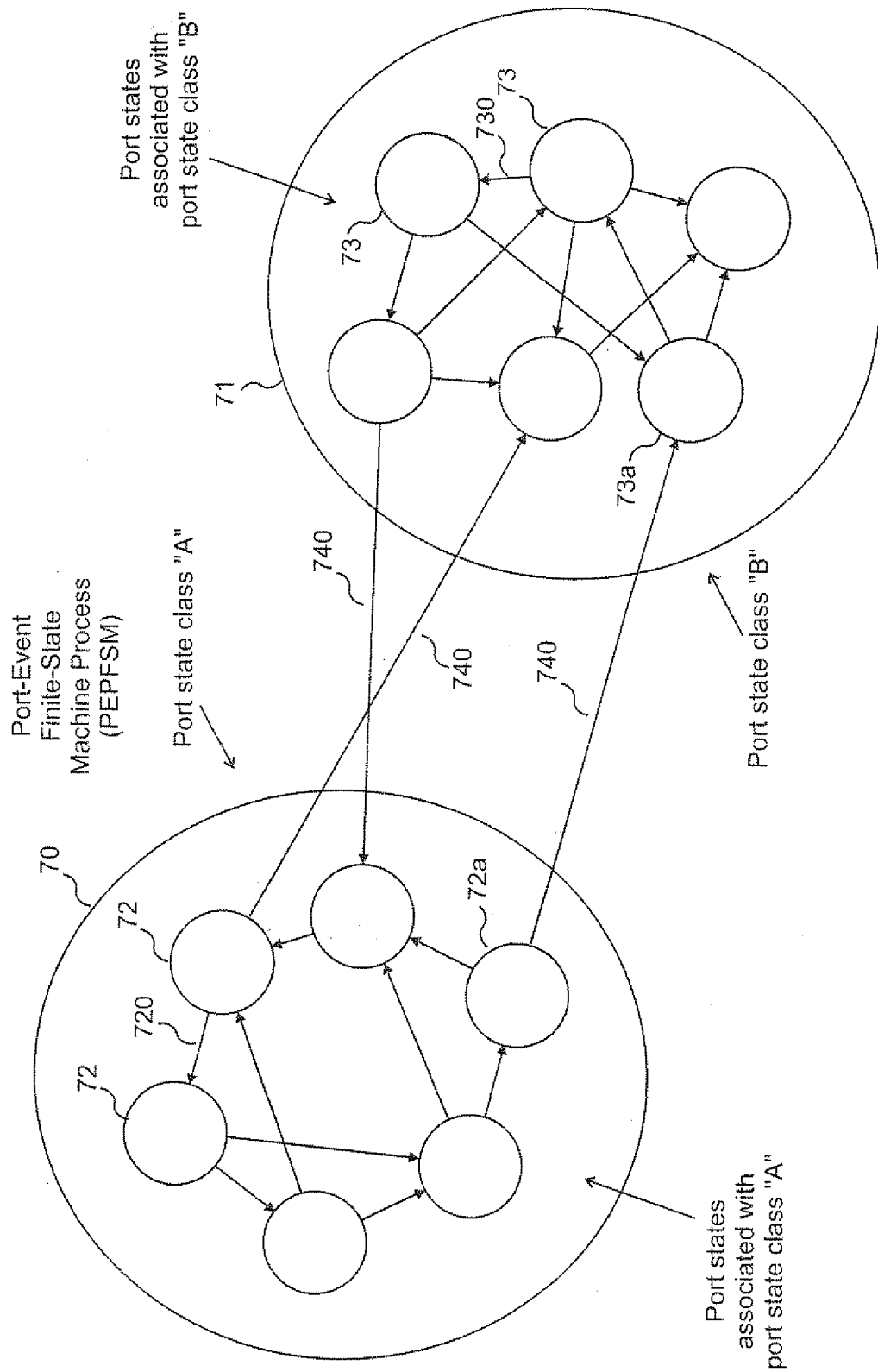
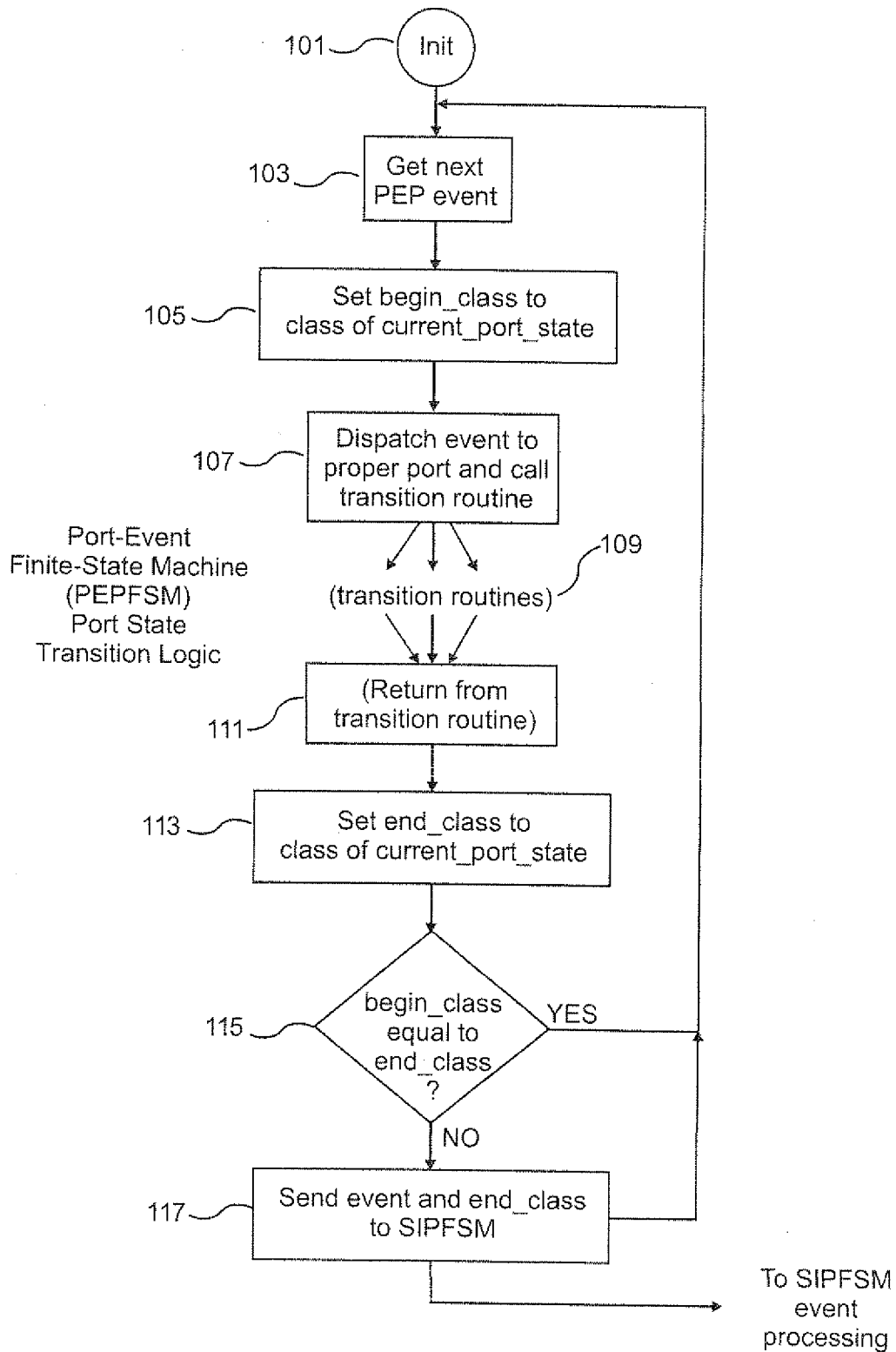


FIG. 3



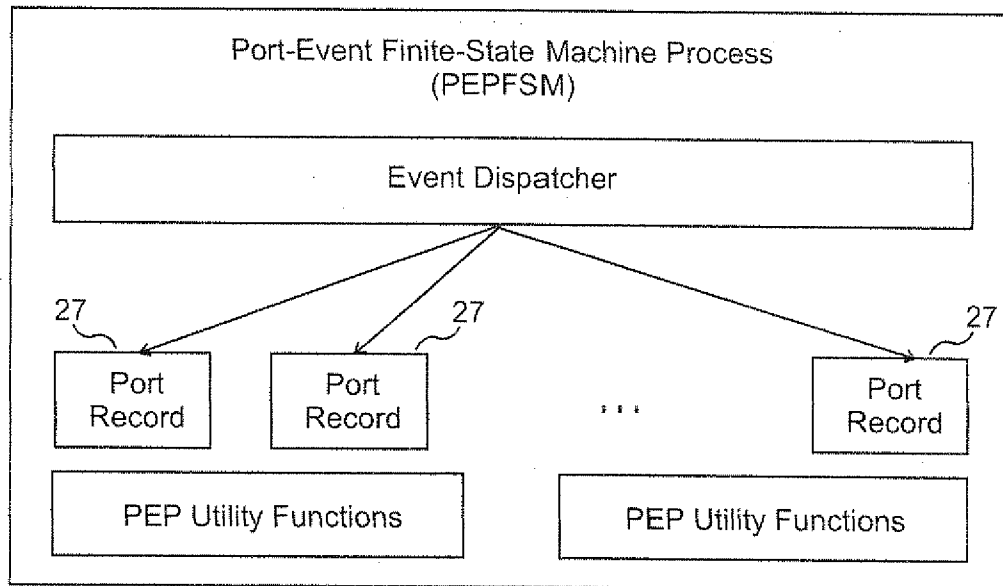


FIG. 5

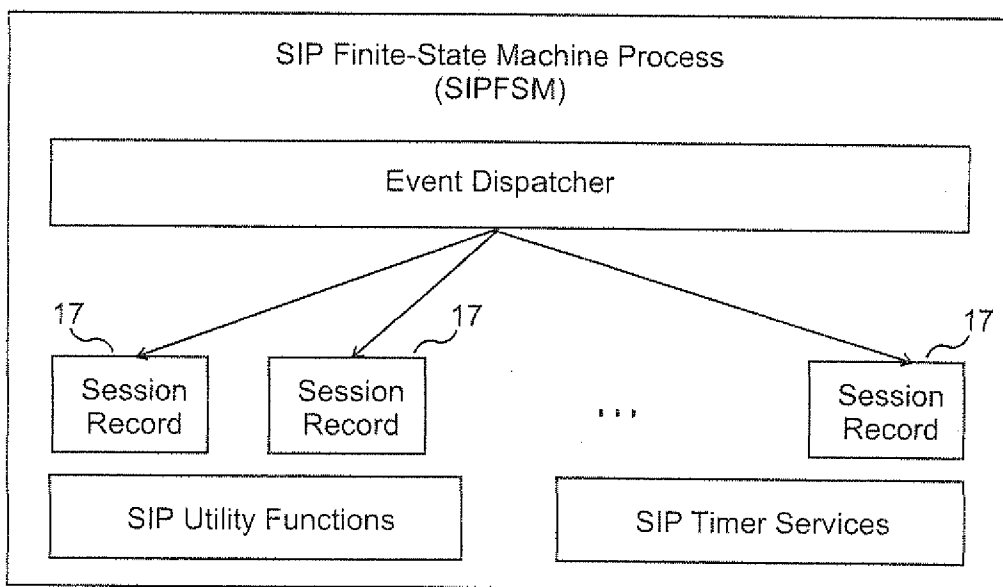


FIG. 6