(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
7 September 2007 (07.09.2007)

PCT

(10) International Publication Number
WO 2007/100468 A2

(51) International Patent Classification:
G06F 15/16 (2006.01)

(21) International Application Number:
PCT/US2007/003500

(22) International Filing Date: 9 February 2007 (09.02.2007)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
11/352,563 13 February 2006 (13.02.2006) US

(71) Applicant *(for all designated States except US)*: **SIG-TEC** [US/US]; 6713 Lakeway Drive, Chanhassen, MN 55317 (US).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: **HOGHAUG, Robert, John** [US/US]; 16767 Blind Lake Trail S.E., Prior Lake, MN 55372 (US). **HOGHAUG, Thomas, Andrew** [US/US]; 6713 Lakeway Drive, Chanhassen, MN 55317 (US).

(74) Agents: **JAEGER, Hugh, D.** et al.; Hugh D. Jaeger, P.A., 150 Lake Street West, Suite 106, P.O. Box 672, Wayzata, MN 55391 (US).

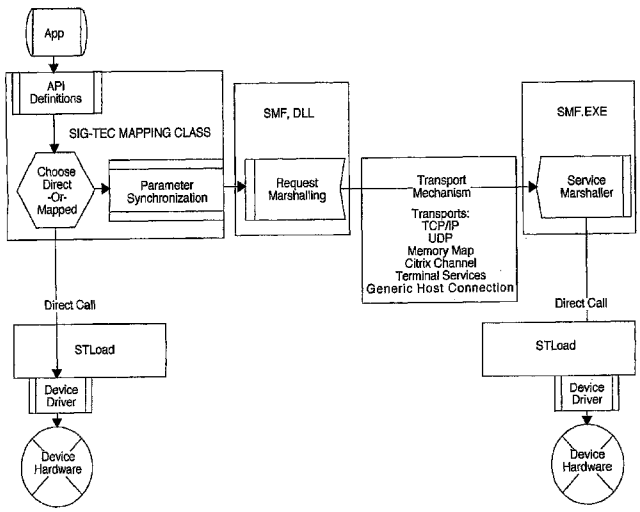
(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LV, LY, MA, MD, 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.

(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, 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).

Published:
— *without international search report and to be republished upon receipt of that report*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

(54) Title: SECURE MESSAGING FACILITY SYSTEM



(57) Abstract: A secure message facility transfers authentication data between various applications, operating systems, and authentication devices and software in the form of messages. These messages comprise a data structure with a standard header with fields that describe the class, length, and type of message, and routing information. This header information is used to route the message to the appropriate handler. The messages are transferred between applications via the messaging facility DLL and the messaging facility Service. The messaging facility DLL is intended to be loaded by an application. The messaging facility DLL forms the messages, directs them to the appropriate messaging facility service (local or remote) and interprets the responses. The messages sent between the messaging facility DLL and messaging facility Service are extremely flexible and can be used to send any type of data or content of messages.

WO 2007/100468 A2

SECURE MESSAGING FACILITY SYSTEM

CROSS REFERENCES TO RELATED APPLICATIONS

[0001] This application claims benefit from the earlier filed U.S. Provisional Application No. 60/653,250 filed February 15, 2005, entitled "Software Messaging Facility System", and is hereby incorporated into this application by reference as if fully set forth herein.

[0002] This patent application is also related to U.S. Provisional Application 60/643,029 filed January 11, 2005, entitled "Multiple User Desktop Graphical Identification and Authentication"; U.S. Provisional Application No. 60/653,249 filed February 15, 2005, entitled "Software Authentication Facility"; and U.S. utility application entitled "Secure Authentication Facility" (Attorney Docket P602), filed concurrently herewith, application to be assigned, a copy of which is attached and the disclosure of which is incorporated herein by reference.

BACKGROUND OF THE INVENTION

FIELD OF THE INVENTION

[0003] The present invention relates to software messaging for computer workstations, servers, networks, and the like, and more particularly, to a method, software, and system for passage of authentication data between computer resources, including hardware devices, operating systems, and graphical identification and authentication modules, and application software, whether locally, remotely, or in combination. The secure messaging facility is intended for use by individuals and businesses that need to limit access to secured resources.

DESCRIPTION OF THE PRIOR ART

[0004] Most authentication-based applications, other than certain proprietary authentication components, communicate with one another by directly loading each other into their respective address spaces. This method of interaction is known as tightly coupled, meaning that individual applications directly call functions of the other applications and load the hardware drivers directly. This method of interaction is very rigid and limiting, since a high degree of compatibility is needed for the applications to be able to properly call functions of the other applications. In order to obtain this high degree of compatibility, highly customized modifications to application and authentication software can be required. The ability to accommodate local and remote users and authentication devices, and provide secure access to local and remote resources, severely limits the flexibility of such an approach.

SUMMARY OF THE INVENTION

[0005] The general purpose of the present invention is to provide a system for transferring data between various applications to enable efficient and flexible secure authentication. Specifically, the present invention overcomes inadequacies of the prior art by creating a novel architecture for authentication-related data transfer and communication between the various users, authentication modules, authentication devices, operating systems, application software, and other resources, including local and remote hardware, software, and data. The present invention provides this utility by transferring all the required data through the messaging facility in the form of messages. Since the authentication-related data is transferred throughout the messaging facility layer in the form of messages, the secure messaging facility of the present invention provides for simpler and more flexible authentication among various local and remote resources. The messages use a request and response format meaning that for every message request submitted there is a response message generated.

[0006] The present secure messaging facility has particular utility for computers, workstations, and servers running Microsoft Windows NT based operating systems. The present invention also has utility for computers running other operating systems, and can be adapted for such.

[0007] The messaging facility is comprised of three main components; the first creates and processes messages and contains the messaging facility library which is an object code library such as those loaded by Windows applications. These libraries are normally referred to as

Dynamic Link Libraries or DLLs. The second component is the messaging facility Service which is a specialized version of the standard Windows application which is designed to be loaded when the operating system starts. The third component is a proprietary authentication server.

[0008] All messages originate in the messaging facility DLL (i.e., the Windows applications) and are sent to the messaging facility Service located on the same machine to which the application loaded the messaging facility DLL. The messages, however, need not be destined for the local messaging facility Service but may be destined instead for a messaging facility Service located on another machine, or for the proprietary authentication server. The final location of a particular message can be effected by various controls, including control by the local messaging facility Service, and including use of information stored in the message itself.

[0009] The messaging facility Service is started when the workstation or server boots, and stays running the entire time the operating system is on. The messaging facility Service is largely passive since it only responds to messages sent to it by the messaging facility dynamic link library. The primary purpose and function of the messaging facility DLL is to form the messages to be sent to the messaging facility Service where they can be processed or directed to the proprietary authentication Service or another messaging facility Service running on another system. The actual message sent between the messaging facility DLL and messaging facility Service is designed to be extremely flexible and can be used to send almost any type of information.

[0010] One significant aspect and feature of the present invention is secure messaging system for communication of authentication data, that is easy to integrate into any application software requiring high levels of security authentication, freeing the software developer from designing, implementing and testing their own version of an authentication system.

[0011] Another significant aspect and feature of the present invention is a secure authentication software development package for use with any combination or variety of biometric, token, proximity or password devices to securely authenticate the user.

[0012] A further significant aspect and feature of the present invention is the ability to maintain a detailed and secure audit trail of successful and unsuccessful authentications and logged events.

[0013] Yet another significant aspect and feature of the present invention is a communication and coordination system for secure authentication where software applications and authentication hardware are uncoupled, and can therefore be local or remote.

[0014] Still another significant aspect and feature of the present invention is a secure messaging facility which is easily adapted and independent of underlying hardware.

[0015] A further significant aspect and feature of the present invention is a secure messaging facility for flexibly implementing local and remote authentication, which is applicable to a variety of operating system environments, including Microsoft Windows NT/2000/XP related operating systems as well as non-Microsoft Windows NT/2000/XP related operating systems.

[0016] Having thus described embodiments and significant aspects and features of the present invention, it is the principal object of the present invention to provide a connection between software operating systems or user applications and authentication applications, authentication devices, or Services.

BRIEF DESCRIPTION OF THE DRAWINGS

[0017] Other objects of the present invention and many of the attendant advantages of the present invention will be readily appreciated as the same becomes better understood by reference to the following detailed description when considered in connection with the accompanying drawings, in which like reference numerals designate like parts throughout the figures thereof and wherein:

[0018] **FIG. 1** is a data flowchart of the present invention;

[0019] **FIG. 2** is a continuation of the data flowchart **FIG. 1**;

[0020] **FIG. 3** is a continuation of the data flowchart **FIGS. 1 and 2**;

[0021] **FIG. 4** is a continuation of the data flowchart from **FIGS. 1, 2 and 3**;

[0022] **FIG. 5** is a block diagram of the messaging facility layers;

[0023] **FIG. 6** is a block diagram of the message structure of the messaging facility and illustrating message header;

[0024] **FIG. 7** is a block diagram of the structure of the message data portion of a message of the message facility; and,

[0025] **FIG. 8** is a block diagram of the secure authentication facility.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

[0026] The present invention is a secure messaging facility which overcomes problems of the prior art methods and software. The messaging facility is comprised of three main components. The first component creates and processes messages and comprises the messaging facility library, which is an object code library loaded by software applications. These libraries are normally referred to as Dynamic Link Libraries or DLLs. The second component is the messaging facility Service which is a specialized version of the standard Windows application designed to be loaded when the operating system starts. The third component is a proprietary authentication server.

[0027] All messages originate in the messaging facility library DLL (i.e., the Windows applications) and are sent to the messaging facility Service located on the same machine to which the application loaded the messaging facility library. The messages, however, need not be destined for the local messaging facility Service but may be destined instead for an messaging facility Service located on another machine, or for the proprietary authentication server. The final location of a particular message can be effected by various controls, including control by the local messaging facility Service, and including use of information stored in the message itself. The use of the message itself to specify the destination of the message makes the present invention particularly adaptable.

[0028] The messaging facility Service is started when the workstation or server boots, and stays running the entire time the operating system is on. The messaging facility Service is largely passive since it only responds

to messages sent to it by the messaging facility dynamic link library. The primary purpose and function of the messaging facility DLL is to form the messages to be sent to the messaging facility Service where they can be processed or directed to the proprietary authentication Service or another messaging facility Service running on another system. The actual message sent between the messaging facility DLL and the messaging facility Service is designed to be extremely flexible and can be used to send almost any type of information.

[0029] As illustrated in **FIG. 5**, the secure messaging facility (SMF) comprises multiple components or layers which interact to enable proper function. One or multiple instances of the SMF DLL are utilized; some instances may be located on remote workstations, servers, or resources. A memory mapped file and mapping layer serves to coordinate the various instances. Transfer of SMF messages can be by TCP/IP as illustrated, or by other protocol. The SMF DLL(s) interact with each other via the memory mapped file / mapping layer facilitating auditing, logging management, and file management functions as depicted. In one embodiment, the secure messaging facility interacts with a secure authentication facility (an example of which is illustrated in **FIG. 8**) as indicated. The secure messaging facility communicates with authentication devices and any required drivers, and communicates with remote SMF via ports with local and/or remote logging as illustrated.

[0030] **FIG. 5** depicts the interaction of the various functional components of the present invention with each other, as well as other resources devices and software, including connections with local and remote software and hardware devices. Examples of authentication

devices which may be utilized with the secure messaging facility include, but are not limited to, passwords, tokens, SecurID, proximity devices, and various types of biometric authentication devices, such as fingerprint or other body feature scanner, sensor, or recorder, voice recognition, and other authentication devices as may become available, and with drivers required for their use. The secure messaging facility of the present invention facilitates utilization of such authentication device(s) even if the device(s) and the secure resource for which authentication is required are not located on the same workstation, server, terminal, kiosk, network access point, PDA, or other computing resource access device.

[0031] **FIG. 1** is a flowchart of the data flow of the present invention. The flowchart depicts the data flow based on the calls and requests of the software. In this example and embodiment, a particular software application (App) requires communication and access to device hardware. The application via API definitions interacts with a proprietary function, Sig-Tec Mapping Class (CST Map) which determines whether a direct or mapped configuration can be utilized. In the direct configuration, a direct call is made, with STLoad and access to the device hardware via any required device driver. In the mapped configuration, with parameter synchronization the secure messaging facility and marshaling request utilizes any of a variety of transport mechanisms (including, but not limited to, TCP/IP, UDP, Memory Map, Citrix, terminal services, or generic host connection) communicating with the secure messaging facility executable file and service marshaller to perform a direct call, with STLoad and access to the device hardware via any required device driver. Thus, direct

call by the application, and call via the secure messaging facility, can coexist depending on the particular requirements and any customization of the application. **FIG. 1** also details the interaction between the DLL entry and the executable program. The varying data will return different values based on response or validity.

[0032] In another embodiment of the present invention, **FIG. 2** is a flowchart which depicts data flow of the present invention's validation functions and validation values returned. Data flows via the CST Map to verification of the validity of the map instance, to decision whether a direct call or an indirect call is to be performed. In the case of an indirect call, a channel request is performed, parameters validated, input parameters copied, and a channel call is performed with validation and copying of the return values.

[0033] In a further extended and detailed flowchart, **FIGS. 3 and 4** depicts logic and data flow for and embodiment of the present invention, with validation of data. As in **FIGS. 1 and 2**, there is a path for direct call and a path for mapped call via the secure messaging facility. With various initialization and logical tests, the direct mode results in a direct call and the mapped or indirect mode utilizes the secure messaging facility, with memory map and validation of input / output parameters, creation of a message, transport of the message, call of SMF function, and a resulting true or false from the function process, to return the result to the CST Map. Item A of **FIG. 3** is a return failure path which is entered when any of several logical tests return false. Items B and C from **FIG. 3** continue to items B and C of **FIG. 4**.

[0034] **FIG. 6** is a block diagram of the message structure of a message of the secure messaging facility of

the present invention. The message contains multiple elements in a header, and message data TLV which will be described in **FIG. 7.** The header can be configured in a variety of ways, depending on the particular requirements of the system. In this example, the header information comprises structure size, version, message type, packet flags, routing information, and response code fields. Additional fields may be incorporated as well, if desired, in keeping with the present invention. In any case, some type of header will typically be used, although the particular fields can be structured differently from those depicted. Request and response messages consist of data structures starting with a static header containing fields for message routing information, the class that a message belongs to, the message type within the specific class, the length of the message data and the information or value associated with the message. All message data takes the form of a TLV (Type, Length, and Value). A message may have more the one TLV associated with it. Header information is used to route, classify, and determine the appropriate local handler and whether it should be sent on to another system for further processing.

[0035] **FIG. 7** is a block diagram of the structure of the message data portion of the message structure of **FIG. 6.** The Message Data TLV block of **FIG. 6** is detailed in **FIG. 7.** The Message structure has the form TLV, that is, type, length, and value. The message may comprise multiple TLV blocks, although only one is depicted in **FIGS. 6 and 7.** Type is a numerical classification depending on the particular function of the message; length defines how long the message is (number of bytes or bits or characters), and value is the particular data being transferred or returned via the message.

[0036] An example of a secure authentication facility is illustrated in **FIG. 8**, which connects to **FIG. 7** via items A, B and C. In this example, a graphical identification and authentication (GINA) or third party application interacts with and accesses a secure authentication facility, API functions, and a memory mapped file via the secure messaging facility DLLs.

MODE OF OPERATION

[0037] The secure messaging facility has a very flexible messaging architecture. In order for an application to use the messaging facility components, a programmer will link and load the messaging facility library (DLL). This can be accomplished using a standard Win32API call or by using the newer COM and Windows. NET methods for accessing Windows DLLs. Once the messaging facility library is loaded into an application, it must be initialized. This initialization sets up a shared memory region that allows contact and information exchange with the messaging facility Service located on the same system as the messaging facility DLL. The messaging facility DLL may not need to access any of the handlers in the messaging facility Service but utilizes a connection to the local messaging facility so that it may contact the appropriate messaging facility Service running on a remote system.

[0038] The messaging facility Service is passive and does not normally distinguish between local messages retrieved from the shared memory region (an application running the local messaging facility DLL) and messages retrieved from the TCP/IP connection or other type of communication protocol, information sent to the messaging facility Service from a messaging facility DLL located on another system. This simplifies the coding of the messaging facility Service as all messages to the handlers are acted upon in the same manner and only the response messages are different in the information contained in the header. All messages contain routing information and the messaging facility Services do not need to keep any state information as to where or how the message was received. The messaging facility Service is passive in nature, so it

never initiates communication with the messaging facility DLL.

[0039] The messaging facility Service is designed to receive messages from the messaging facility, route them to the correct message handler and then formulate an appropriate response. The messaging facility Service handlers are not dependent upon message origination, as this information is stored in the message header and is only used by the send message routine to route the message back to the correct messaging facility library. The messaging facility Service is intended to handle multiple connections from multiple messaging facility libraries either located locally or remotely. The routing information for each message is stored in the message header and is the responsibility of the messaging facility DLL. This simplifies the messaging facility Service logic and the design of the messages themselves. There is a hierarchy to how message are created and how the handlers for these messages are determined, and all messages are placed in a specific class that describes in general what type of message it is. Some examples are the audit message class and the credential message class. Within a message class there can be multiple message categories; the messages created by the messaging facility are generally derived from the application programming interfaces (API).

[0040] In some embodiments, the secure messaging facility is adapted for 32-bit Windows NT-based operating systems, including, but not limited to, Windows NT 4.0 Server/Workstation, Windows 2000 Server/Workstation, and Windows XP systems.

[0041] In another embodiment, the secure messaging facility is adapted for a 64-bit operating system environment.

[0042] In yet another embodiment, the secure authentication facility is configured to run on operating systems other than Windows NT based operating systems.

[0043] Some embodiments of the present invention comprise software. Additional embodiments of the present invention comprise methods of authentication data transfer. One such method provides for transfer of user authentication data accommodating remote authentication device(s). Another such method provides for transfer of user authentication data accommodating remote applications and resources.

[0044] Various modifications can be made to the present invention without departing from the apparent scope hereof. This description will suggest many variations and alternatives to one of ordinary skill in this art. The various elements described may be combined or modified for combination as desired. All these alternatives and variations are intended to be included within the scope of the claims. Further, the particular features presented in the dependent claims can be combined with each other in other manners within the scope of the invention.

IT IS CLAIMED:

1. Software for transfer of user authentication data, comprising:

a. a DLL which communicates with application software;

b. said DLL creates a message which includes the authentication data and destination information; and,

c. said message communicates the authentication data from an origin and delivers the authentication data to a destination.

2. The software of claim 1, wherein said origin and said destination are each selected from a list consisting of local application software, remote application software, operating system, network software, local driver, remote driver, local authentication device and remote authentication device.

3. The software of claim 1, wherein at least one of said origin and said destination is an authentication device, and said authentication device is selected from the list consisting of passwords, tokens, SecurID, proximity devices, biometric authentication devices, fingerprint scanner, body feature scanner, body feature sensor, sound recorder, and voice recognition device.

4. The software of claim 1, wherein said DLL is compatible with Microsoft Windows NT/2000/2003/XP based 32 bit and 64 bit operating systems.

5. The software of claim 1, wherein said DLL is compatible with non-Microsoft Windows NT/2000/2003/XP based operating systems.

6. The software of claim 1, wherein said DLL functions when the user is attached to a domain.

7. The software of claim 1, wherein said DLL functions when the user is not attached to a domain.

8. The software of claim 1, further comprising a software developer's kit with an application programming interface to said software.

9. The software of claim 1, wherein at least one of said origin and said destination is located remotely, and remote communication is accomplished under TCP/IP.

10. A method of transferring user authentication data comprising the steps of:

a. providing a DLL which creates a message which includes the authentication data;

b. transferring the message from an origin; and,

c. receiving the message by a destination and thereby receiving the authentication data.

12. The method of claim 10, wherein at least one of origin and the destination are located remotely.

13. The method of claim 12, wherein the transferring of the message utilizes TCP/IP.

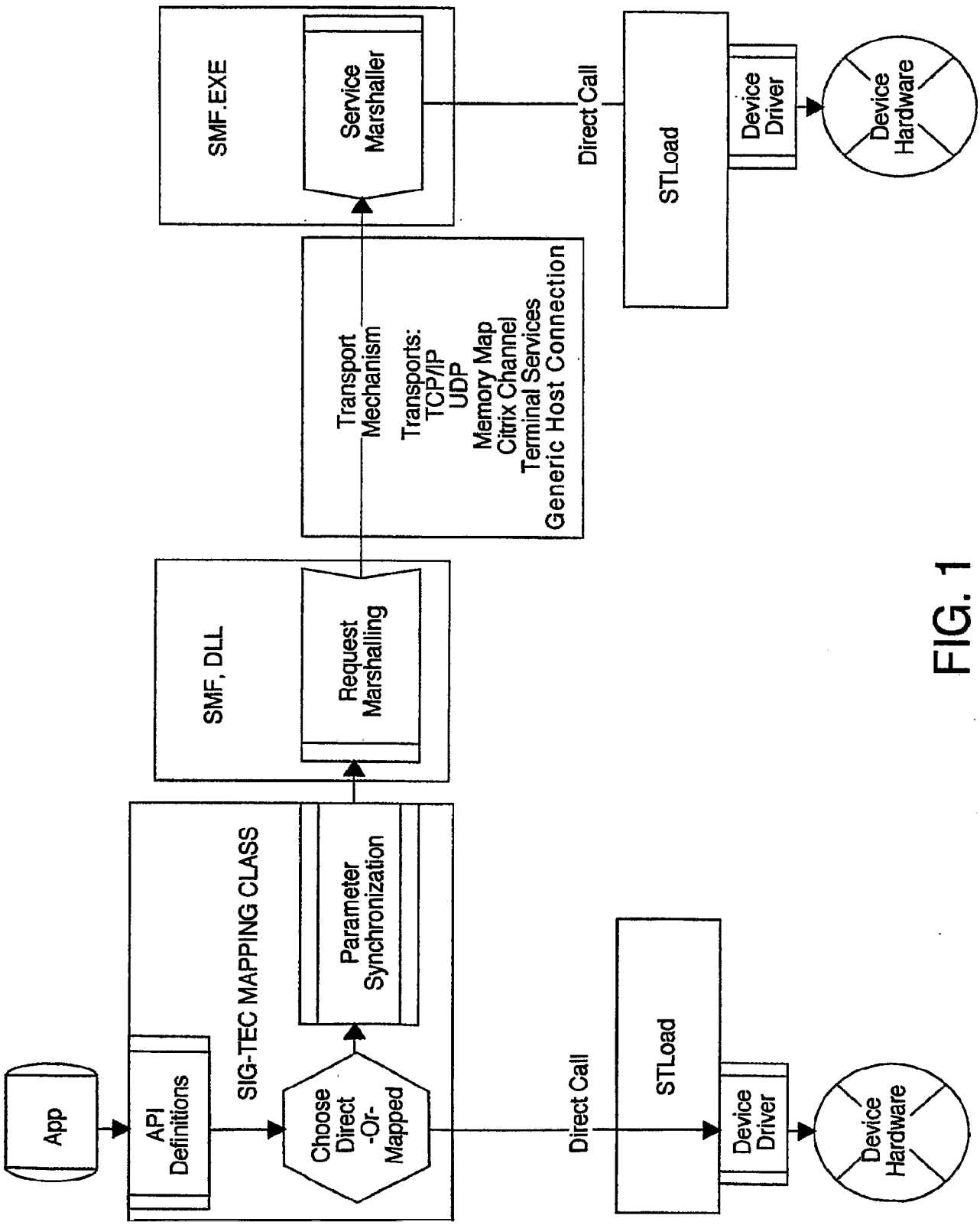

App
API Definitions
SIG-TEC MAPPING CLASS
Choose Direct -Or- Mapped
Parameter Synchronization
SMF, DLL
Request Marshalling
Transport Mechanism
Transports:
TCP/IP
UDP
Memory Map
Citrix Channel
Terminal Services
Generic Host Connection
SMF.EXE
Service Marshaller
Direct Call
STLoad
Device Driver
Device Hardware
Direct Call
STLoad
Device Driver
Device Hardware

FIG. 1

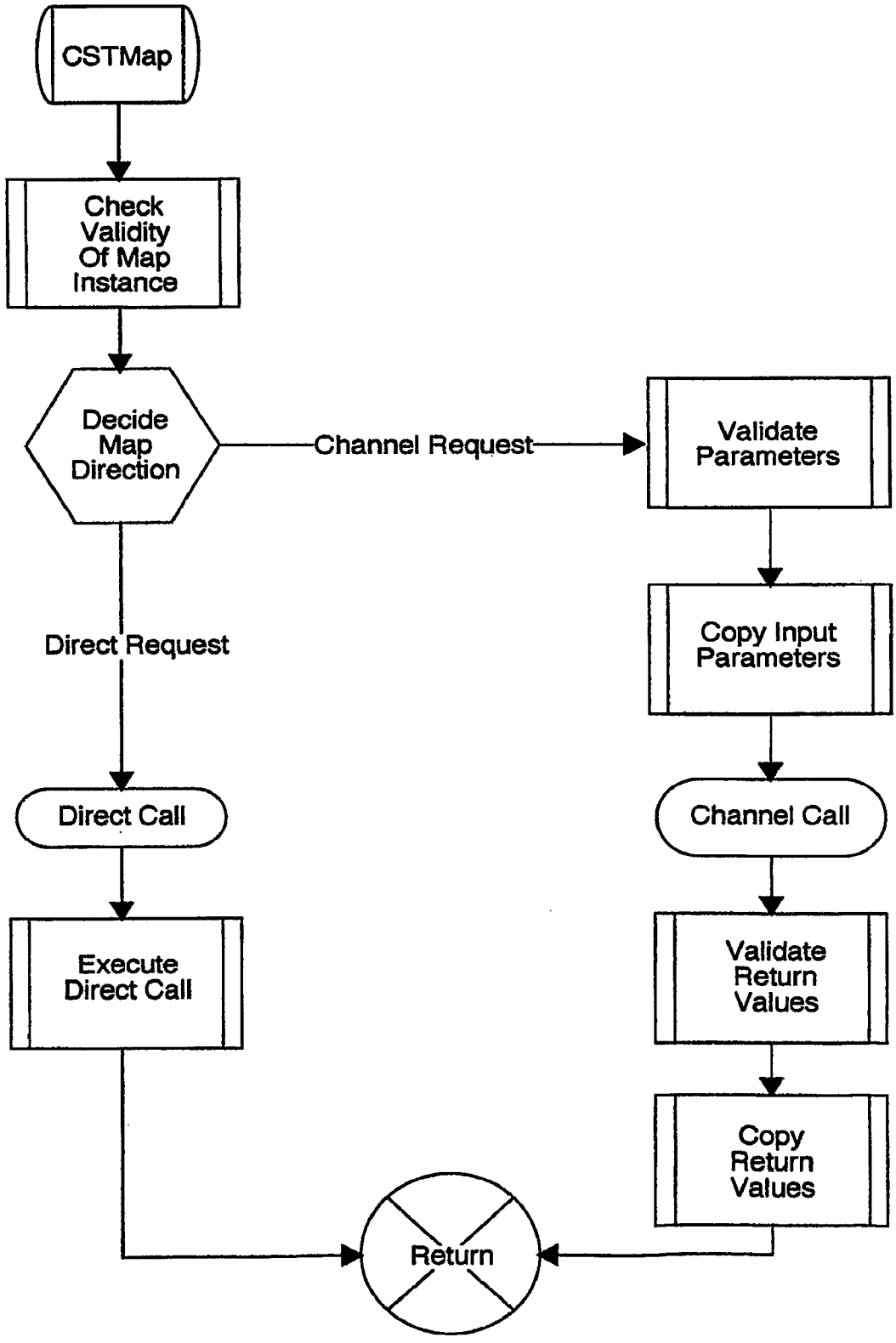

CSTMap
Check Validity Of Map Instance
Decide Map Direction
Channel Request
Validate Parameters
Copy Input Parameters
Channel Call
Validate Return Values
Copy Return Values
Direct Request
Direct Call
Execute Direct Call
Return

FIG. 2

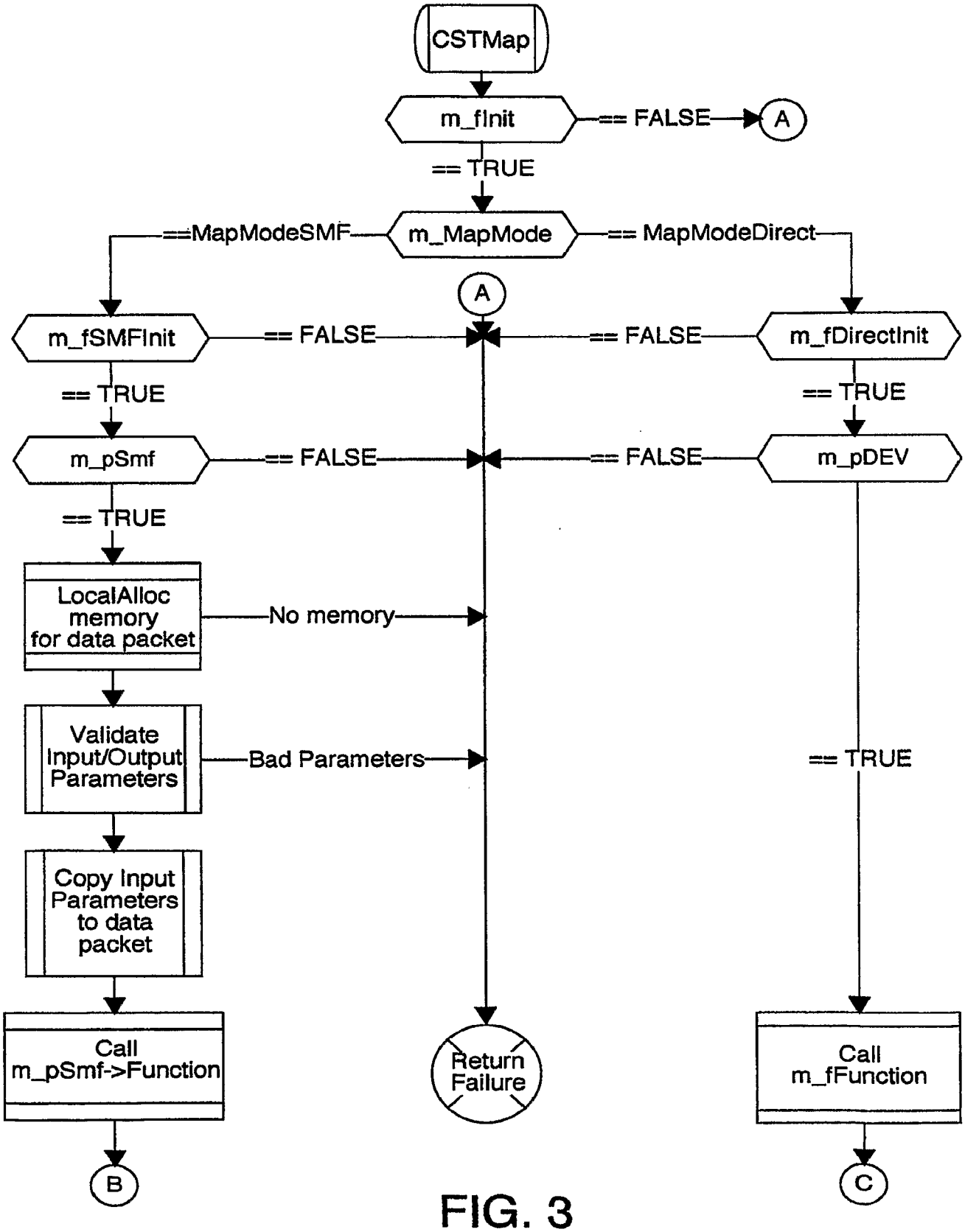

CSTMap
m_fInit
== FALSE
A
== TRUE
m_MapMode
==MapModeSMF
== MapModeDirect
A
m_fSMFInit
== FALSE
== FALSE
m_fDirectInit
== TRUE
== TRUE
m_pSmf
== FALSE
== FALSE
m_pDEV
== TRUE
LocalAlloc memory for data packet
No memory
Validate Input/Output Parameters
Bad Parameters
== TRUE
Copy Input Parameters to data packet
Call m_pSmf->Function
Return Failure
Call m_fFunction
B
C

FIG. 3

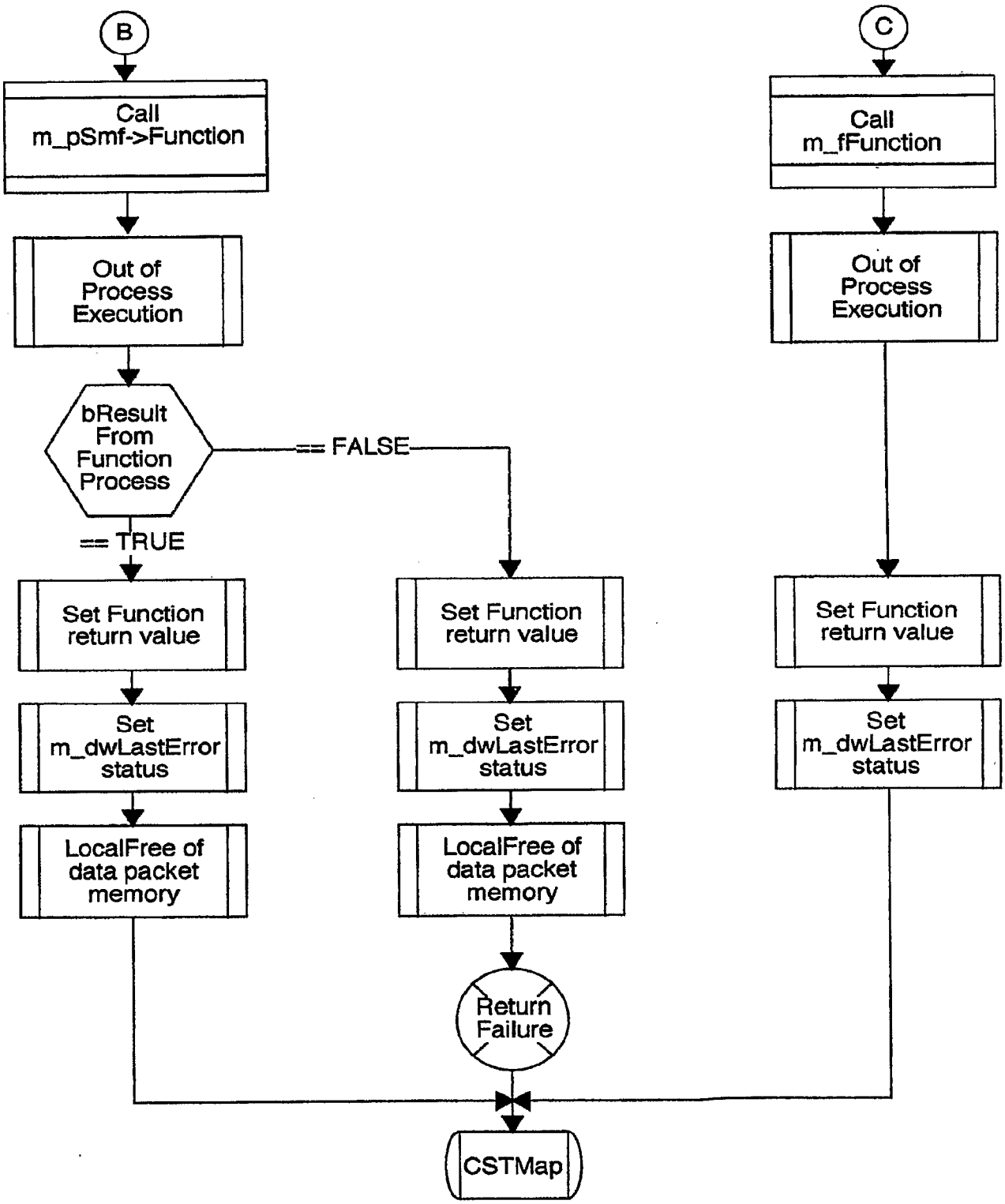

B
Call m_pSmf->Function
Out of Process Execution
bResult From Function Process
== FALSE
== TRUE
Set Function return value
Set m_dwLastError status
LocalFree of data packet memory
Set Function return value
Set m_dwLastError status
LocalFree of data packet memory
Return Failure
C
Call m_fFunction
Out of Process Execution
Set Function return value
Set m_dwLastError status
CSTMap

FIG. 4

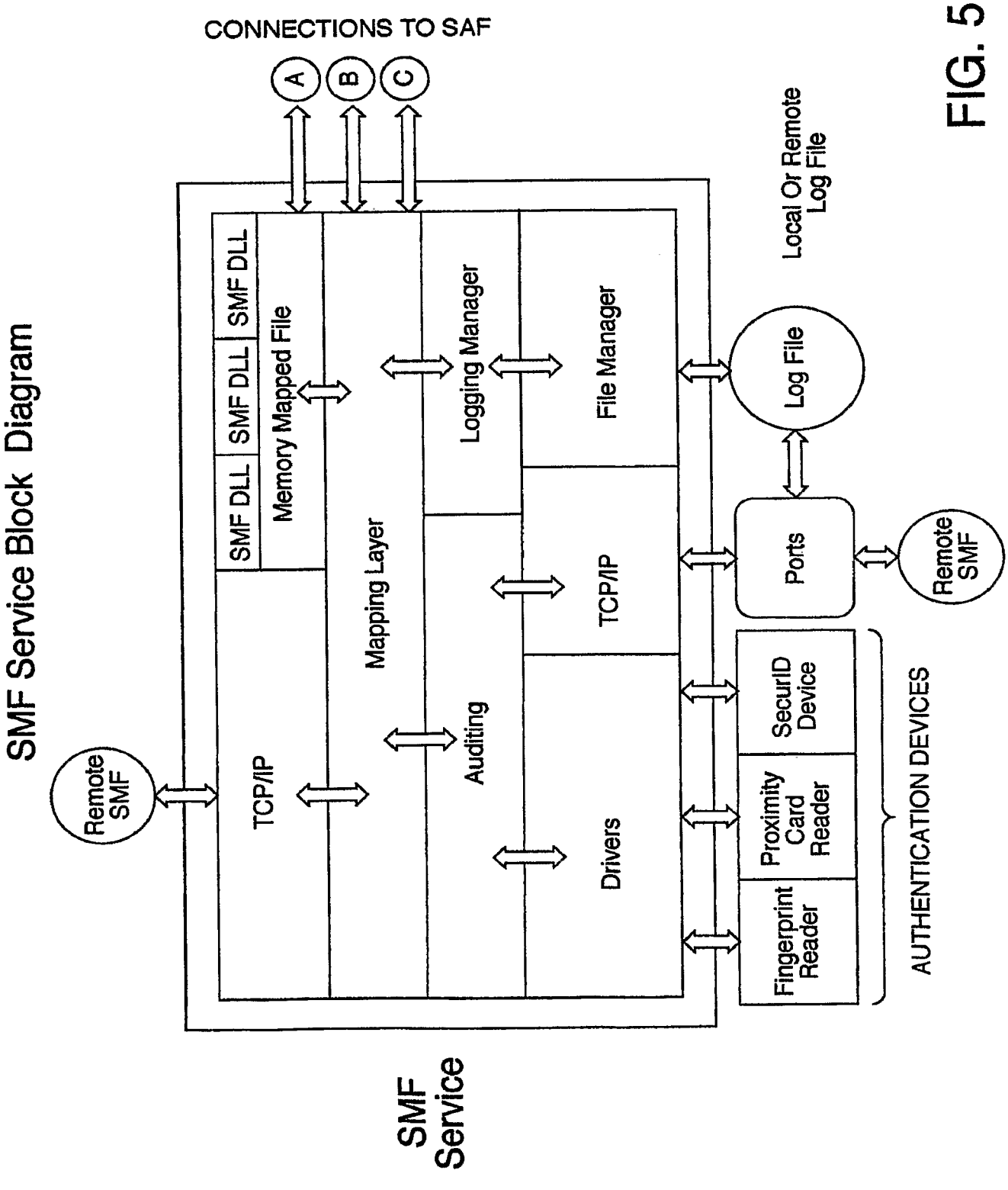

SMF Service Block Diagram
CONNECTIONS TO SAF
A
B
C
SMF DLL
SMF DLL
SMF DLL
Memory Mapped File
Logging Manager
File Manager
Local Or Remote Log File
Log File
Mapping Layer
TCP/IP
Auditing
TCP/IP
Drivers
Ports
Remote SMF
Remote SMF
SecurID Device
Proximity Card Reader
Fingerprint Reader
AUTHENTICATION DEVICES
SMF Service

FIG. 5

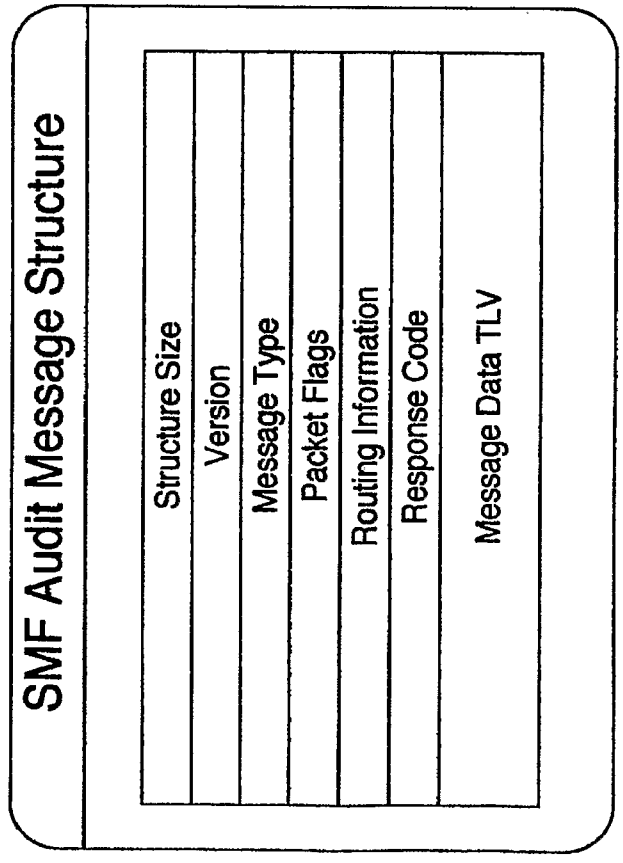

SMF Audit Message Structure
Structure Size
Version
Message Type
Packet Flags
Routing Information
Response Code
Message Data TLV

FIG. 6

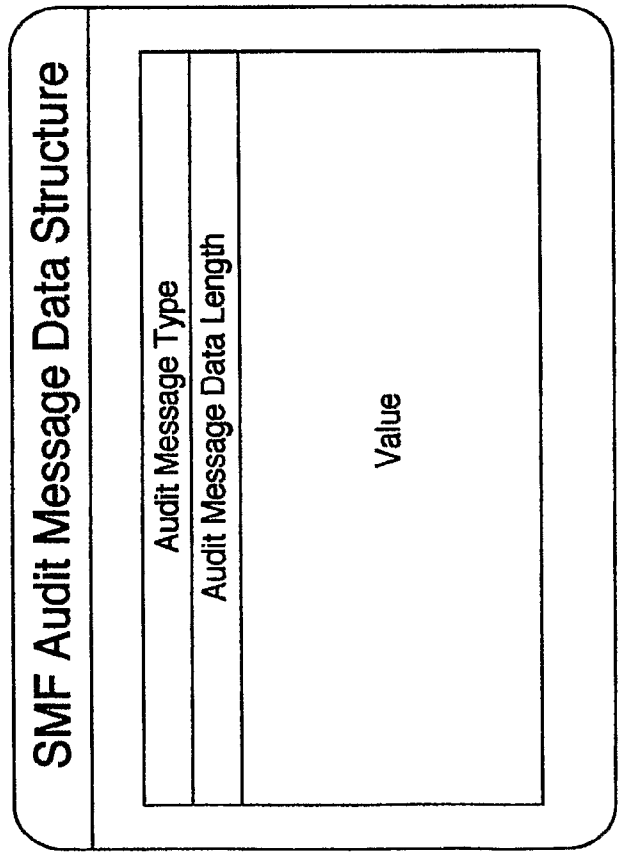

SMF Audit Message Data Structure
Audit Message Type
Audit Message Data Length
Value

FIG. 7

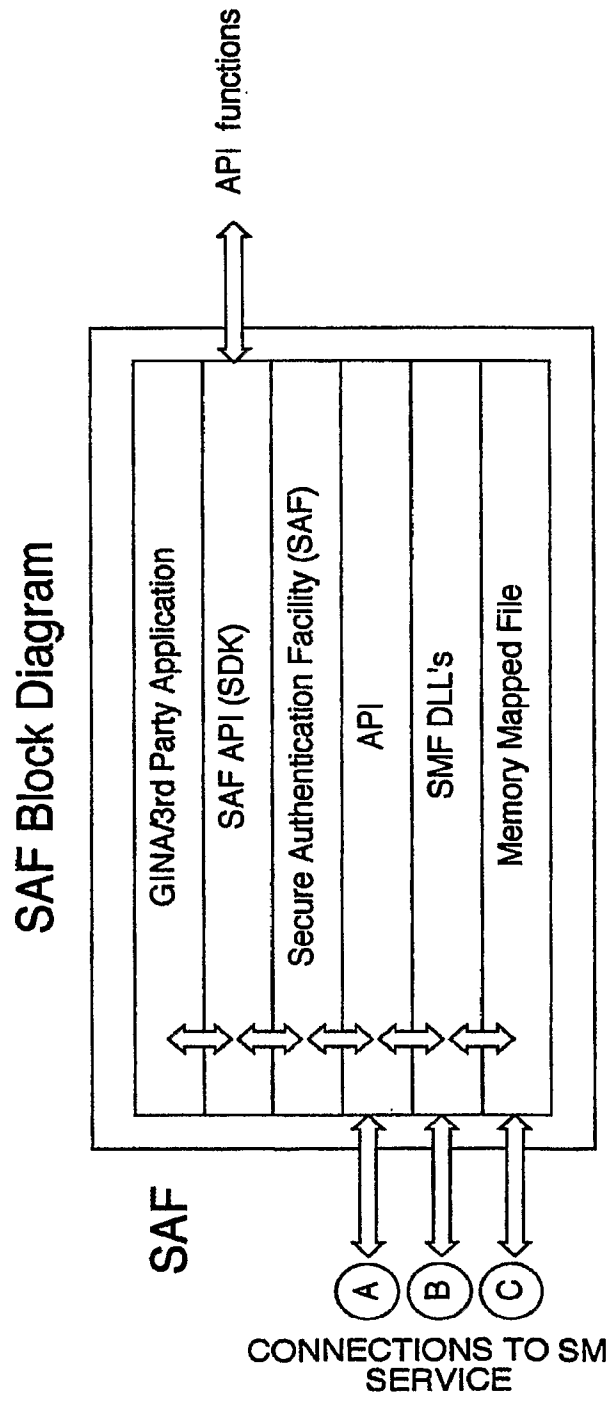

SAF Block Diagram
API functions
GINA/3rd Party Application
SAF API (SDK)
Secure Authentication Facility (SAF)
API
SMF DLL's
Memory Mapped File
SAF
A
B
C
CONNECTIONS TO SMF SERVICE

FIG. 8