EXHIBIT 10 - FISCHER

U.S. Patent No. 5,412,717
"Computer System Security Method And Apparatus Having Program Authorization Information Data Structures"
Inventors: Addison M. Fischer
Filing Date: May 15, 1992
Date of Issue: May 2, 1995
("*Fischer*")

U.S. Patent No. 6,192,476 – Claim 1	***Fischer***
1. A method for providing security, the method comprising the steps of:	*Fischer* discloses a method for providing security. The Fischer disclosure employs "program authorization information," or "PAI," to limit a principal's access authority to other code and resources. "More particularly, the invention relates to a method and apparatus for providing enhanced computer system security while processing computer programs, particularly those of unknown origin, which are transmitted among users." *Fischer* at 1:20-25. "Thus, the present invention advantageously protects a user from any program to be executed. The present invention is particularly advantageous in light of current data processing practices where programs are obtained from a wide range of diverse, untrustworthy places such as computer bulletin boards or other users of unknown trustworthiness." *Fischer* at 2:49-55.
detecting when a request for an action is made by a principal; and	*Fischer* discloses detecting when a request for an action is made by a principal. *Fischer* inquires into the permissions of a given principal's PAI upon the principal's request for an action: "When the program is to perform a function or access a resource, the associated PAI is monitored to confirm that the operation is within the defined program limits. If the program attempts to do anything outside the authorized limits, then the program execution is halted." *Id.* at 2:43-48. Thus it inherent that *Fischer* detects a request of an action made by a principal. "Prior art operating systems are typically designed to protect data from computer users. In

U.S. Patent No. 6,192,476 – Claim 1	*Fischer*
	such systems, users are often assigned various authorities and are thereafter able to execute programs based on their associated authority. If a program is executing which exceeds the user's assigned authority, then such a system will halt execution of the program. Such prior art systems do not adequately protect computer users from computer viruses or the like. There are security systems which protect certain 'system' related files from being modified by a program. However, such systems do not typically protect a computer user from a program executing and modifying the user's own files. The present invention is directed to providing reliable security, even when operating with complex data structures, e.g., objects, containing their own program instructions, which are transmitted among users. The present invention also provides enhanced security when processing more conventional programs, even those of questionable origin, e.g., from a computer bulletin board, without exposing system programs or data to the potentially catastrophic consequences of computer viruses or of incompetent programming. The present method and apparatus utilizes a unique operating system design that includes a system monitor which limits the ability of a program about to be executed to the use of predefined resources (e.g., data files, disk writing capabilities etc.). The system monitor builds a data structure including a set of authorities defining that which a program is permitted to do and/or that which the program is precluded from doing. The set of authorities and/or restrictions assigned to a program to be executed are referred to herein as 'program authorization information' (or 'PAI'). Once defined, the program authorization information is thereafter associated with each program to be executed to thereby delineate the types of resources and functions that the program is allowed to utilize. The PAI associated with a particular program may be assigned by a computer system owner/user or by someone who the computer system owner/user implicitly trusts. The PAI defines the range of operations that a program may execute and/or defines those operations that a program cannot perform. The program is permitted to access what has been authorized and nothing else. In this fashion, the program may be regarded as being placed in

U.S. Patent No. 6,192,476 – Claim 1	*Fischer*
	a program capability limiting 'safety box'. This 'safety box' is thereafter associated with the program such that whenever the system monitor runs the program, the PAI for that program is likewise loaded and monitored. When the program is to perform a function or access a resource, the associated PAI is monitored to confirm that the operation is within the defined program limits. If the program attempts to do anything outside the authorized limits, then the program execution is halted. Thus, the present invention advantageously protects a user from any program to be executed. The present invention is particularly advantageous in light of current data processing practices where programs are obtained from a wide range of diverse, untrustworthy places such as computer bulletin boards or other users of unknown trustworthiness." *Fischer* at 1:60-2:55. "FIG. 4 is a flowchart which illustrates how a user may benefit from the use of program authorization information, particularly when executing a program of unknown trustworthiness. As indicated in block 121, a user may have a desire to execute a program of interest in which the user has no knowledge of the program's creator. Thus, the program has unknown trustworthiness and may, for example, have been accessed via an electronic bulletin board and may have arrived at the user's terminal via a telecommunications channel or diskette. Such a program, which might purport to be only a game, carries with it a significant risk that it may be infected by a virus. As indicated in block 122, the user may be protected by defining program authorization information which restricts the program to only unimportant or expendable files. If desired, the user may restrict such a program from modifying any files whatsoever. For example, the user may permit the program to only display images on the display screen and to perform game playing related functions. Alternatively, if the program is known to have a single work file, the PAI data may only permit use of such a single file. By limiting access only to a single work file, a program of unknown trustworthiness, cannot inject a virus into other user's programs or otherwise initiate system program malfunctions. Thus, in accordance with the present invention, a user, via a systems program, determines how much of the user's system will be put at risk by such a program so as to, for example, completely

U.S. Patent No. 6,192,476 – Claim 1	*Fischer*
	eliminate the ability of the program to use any privileged functions. The user then associates, for example, through an operator prompting, menu-driven system, a PAI with every program to be run on the system (or have such PAI or lack of PAI associated through predetermined default mechanisms). A system utility program is preferably employed to create the program authorization information in a manner which will be described in detail below in conjunction with FIGS. 6-9. After the PAI has been assigned, any time the system runs the associated program, the system software (in a manner to be described below) insures that the program safely runs in a manner consistent with the PAI. Thus, the program has been effectively placed in a 'safety box' (124). Turning back to FIG. 1, a program of unknown trust may be injected into the system via communications channel 12 or from a floppy disk loaded into terminal A. The program may be initially stored in, for example, the user's program disk memory 7. Thereafter, the user on keyboard 4 will, through interaction with the system's program identified above (with respect to block 122 of FIG. 4), associate the program authorization information with a program (in a manner such as shown in FIGS. 3A through 3D) such that the program may safely run on the user's system or perhaps, a PAI arrives with the program, in which case it is likely to be signed. FIG. 5 is an illustration of a program control block (PCB) data structure 140 in accordance with an exemplary embodiment of the present invention. The program control block 140 is the data structure utilized by the system monitor to control the execution of an associated program. The program control block 140 is loaded with program authorization information such that the PAI can be readily referenced as the associated program is executed so as to insure that the program performs functions and accesses resources in conformance with its assigned authorizations. The program control block associated with the program to be executed is located in a storage area which cannot be modified by the program." *Fischer* at 9:17-10:23.

U.S. Patent No. 6,192,476 – Claim 1	*Fischer*

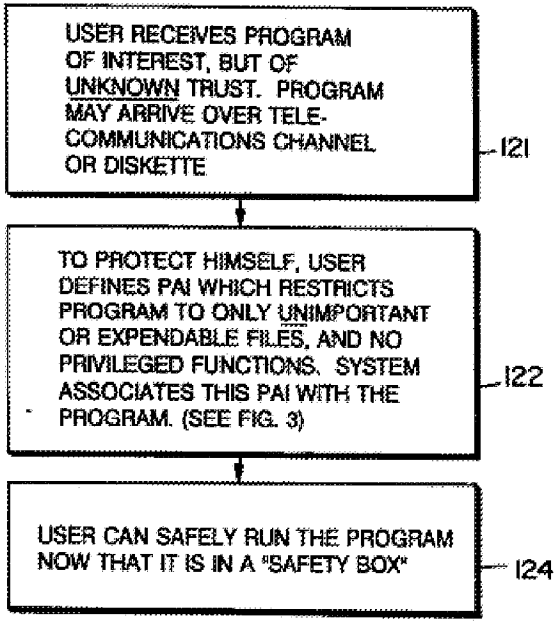
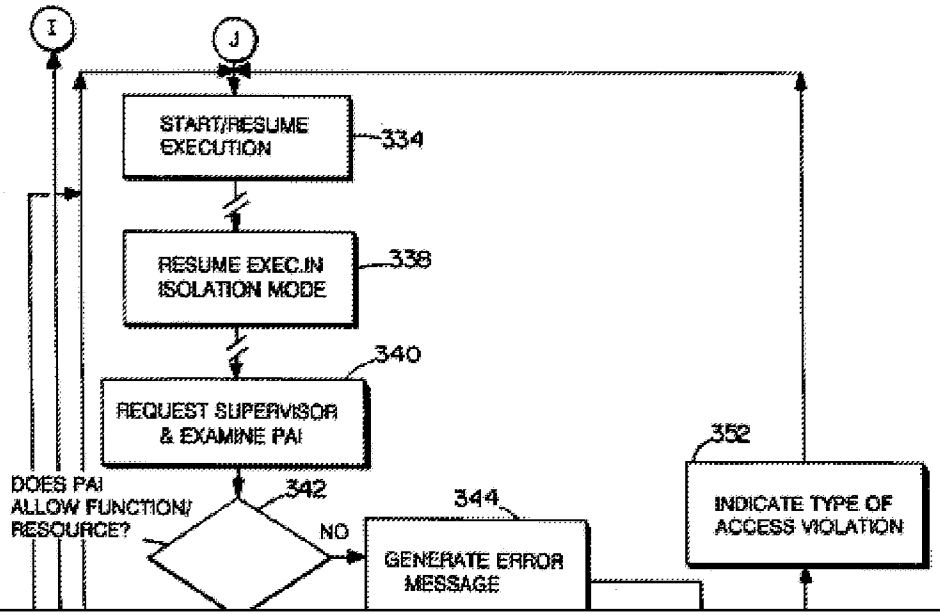


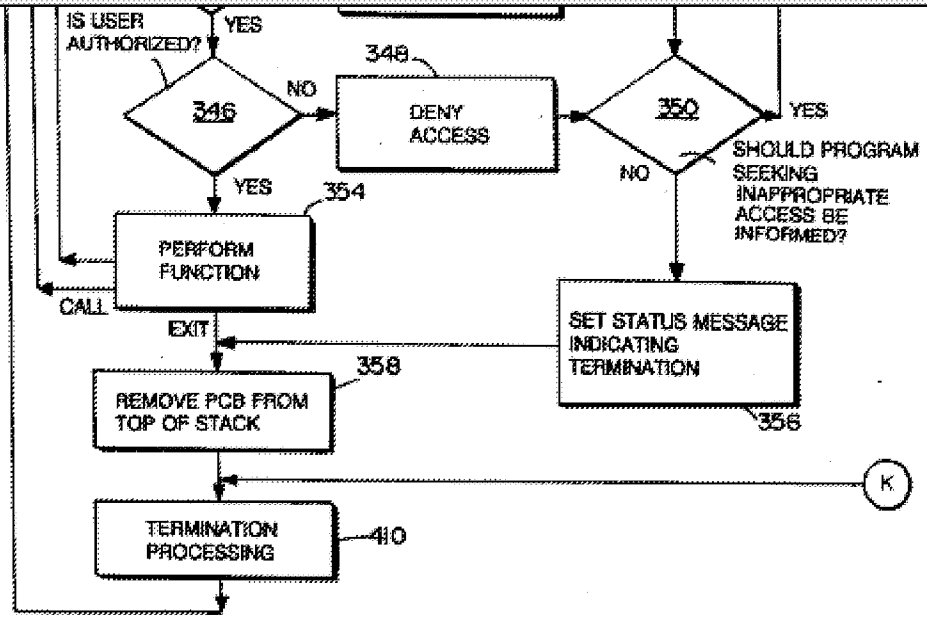
Fig. 4

Fischer at Fig. 4.

"FIGS. 10 and 11 illustrate the sequence of operations of a supervisor program for controlling the processing of a program being executed in accordance with its program authorization information. The processing of a program 'X' and its program authorization information illustrated in FIG. 10 is initiated while the computer is executing a supervisor routine. As shown in FIG. 10 at 300, a calling program calls program X for execution. Thereafter, a program control block is created for program X. The program control block created will not be added to the top of the execution stack until it is determined that the program is permitted to be invoked and verification is successful completed. Thus, if the program fails a security check, it will not be placed in the program execution chain. In addition to creating a 'tentative' program control block, the called program will be located through an appropriate program directory during the processing in block 302.

Thereafter, a check is made at block 304 to determine whether PAI has yet been associated

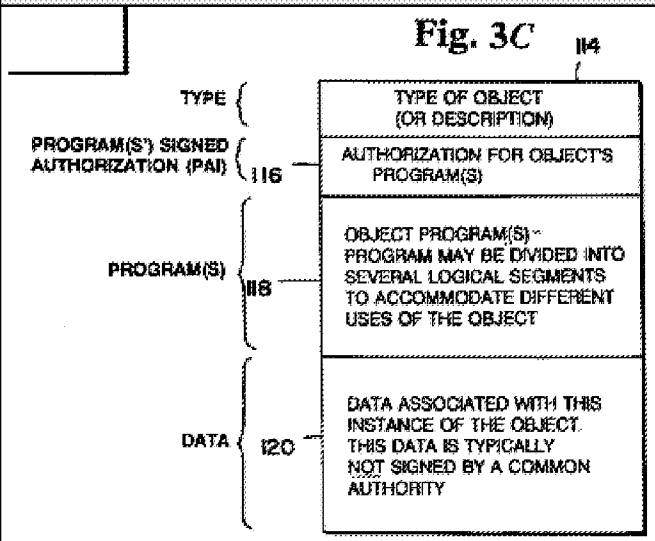
U.S. Patent No. 6,192,476 – Claim 1	*Fischer*
	with program X so as to place program X in the so-called 'safety box' described above. This PAI may or may not be signed depending upon its particular application as described above." *Fischer* at 15:56-16:11. 

U.S. Patent No. 6,192,476 – Claim 1	*Fischer*
	 **Fig. 11** *Fischer* at Fig. 11.
in response to detecting the request, determining whether said action is authorized based on permissions associated with a plurality of routines in a calling hierarchy associated with said principal, wherein said permissions are associated with said plurality of routines based on a first association between protection domains and permissions.	*Fischer* discloses in response to detecting the request, determining whether said action is authorized based on permissions associated with a plurality of routines in a calling hierarchy associated with said principal, wherein said permissions are associated with said plurality of routines based on a first association between protection domains and permissions. *Fischer* discloses "an originating program" that "calls a program" having a program control block, or PCB. *Fischer* at 10:24-26. "Each new PCB will include a field . . . that points to the 'previous' of calling program control block." *See id.* at 10:27-29. Then, "[w]hen a called program finishes executing, the system removes its associated PCB from the top of the executed stack, removes the associated program from storage, removes the associated authorizing information and accesses the program control block immediately below it in the stack." *See id.* at 10:31-36. *Fischer* also incorporates by reference two patents which explicitly disclose data hierarchies: *see id.* at 6:37 (incorporating by reference U.S. Patent

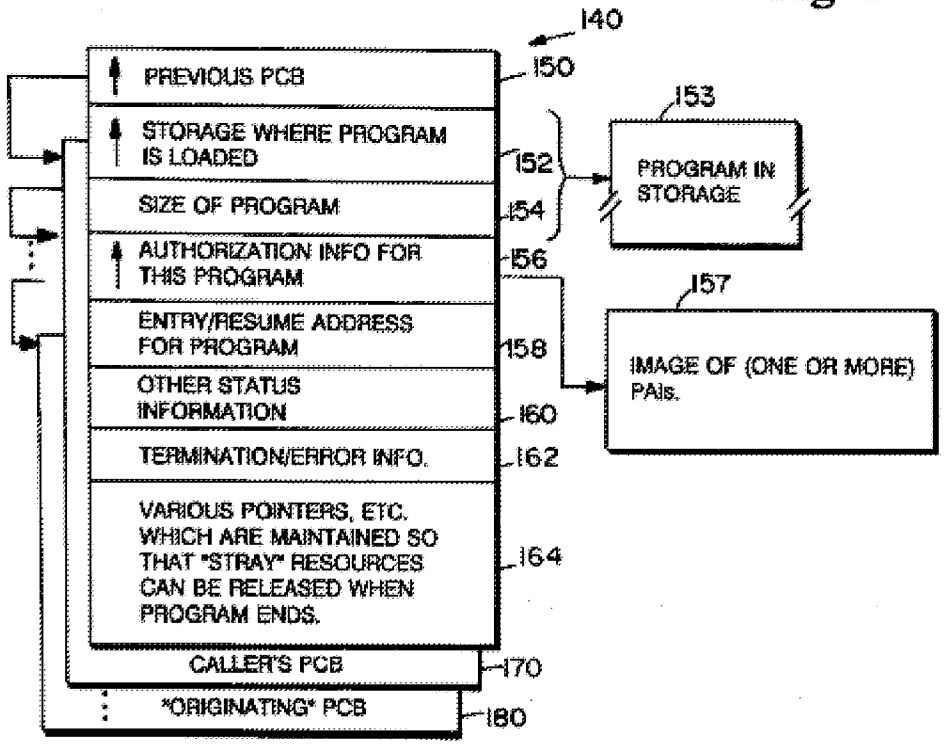
U.S. Patent No. 6,192,476 – Claim 1	*Fischer*
	Nos. 4,868,877 and 5,005,200 which disclose an "enhanced digital signature certification" which employs "[a] hierarchy of nested certifications and signatures." *See* U.S. Patent No. 5,005,200, Abstract. "The present method an apparatus utilizes a unique operation system design that includes a system monitor which limits the ability of a program about to be executed to the use of predefined resources (e.g., data files, disk writing capabilities etc.). The system monitor builds a data structure including a set of authorities defining that which a program is permitted to do and/or that which the program is precluded from doing. The set of authorities and/or restrictions assigned to a program to be executed are referred to herein as 'program authorization information' (or 'PAI'). Once defined, the program authorization information is thereafter associated with each program to be executed to thereby delineate the types of resources and functions that the program is allowed to utilize. The PAI associated with a particular program may be assigned by a computer system owner/user or by someone who the computer system owner/user implicitly trusts. The PAI defines the range of operations that a program may execute and/or defines those operations that a program cannot perform. The program is permitted to access what has been authorized and nothing else. In this fashion, the program may be regarded as being placed in a program capability limiting 'safety box.' This 'safety box' is thereafter associated with the program such that whenever the system monitor runs the program, the PAI for that program is likewise loaded and monitored. When the program is to perform a function or access a resource, the associated PAI is monitored to confirm that the operation is within the defined program limits. If the program attempts to do anything outside the authorized limits, then the program execution is halted." *Fischer* at 2:16-48. "Thereafter, the program X's program authorizing information is combined, as appropriate, with the PAI associated with the PCB of the calling program, if any. This combined PAI, which may include multiple PAI's, is then stored in an area of storage which cannot generally be modified by the program and the address of the PAI is stored in the process control block (PCB) as indicated in field 156 of FIG. 5. Thus, if program X is called by a

U.S. Patent No. 6,192,476 – Claim 1	*Fischer*
	calling program, it is subject to all its own constraints as well as being combined in some way with the constraints of the calling program, which aggregate constraints are embodied into program X's PAI. In this fashion, a calling program may not be permitted to exceed its assigned bounds by merely calling another program." *Fischer* at 19:40-54. "Even programs with no known trustworthiness can be used after program authorization information associates a wide range of restrictions to thereby allow potentially beneficial programs to be safely used--even if they do not have an official certification of trust. The present invention also allows an unlimited number of different resources and functions to be controlled. For example, some useful resources/functions which may be controlled include: the ability to limit a program to certain files or data sets; the ability to transmit data via electronic mail to someone outside the user's domain; the ability of a program to create or solicit digital signatures; the ability to limit access to a program of certain security classes, etc." *Fischer* at 3:48-61. "The present invention is directed to providing reliable security, even when operating with complex data structures, e.g., objects, containing their own program instructions, which are transmitted among users." *Fischer* at 2:6-9. "Through the use of the present invention, general object oriented data may be transferred from user to user without exposing users to the potential dangers of viruses or mischievous users." *Fischer* at 4:10-13. "In one contemplated embodiment of the present invention, programs may be part of data objects, which are written in a high-level control language and are executed by a standardized interpreter program which executes this high-level language. In this case, part of the interpreter's task is to verify that the functions encountered in the high level logic are, in fact, permissible. If such tasks are not permissible, the interpreter then suppresses the execution of the program not authorized to perform such tasks." *Fischer* at 3:11-20. "In accordance with the present invention, a PAI is associated with programs to be executed.

U.S. Patent No. 6,192,476 – Claim 1	*Fischer*
	FIGS. 3A through 3D depict four exemplary approaches for associating program authorization information with a program. . . . . . . FIG. 3C shows an important application in which a PAI data structure is associated with a program according to an embodiment of the present invention. FIG. 3C shows an illustrative data structure for a secure exchangeable 'object'. The data structure may be signed by a trusted authority. The signing of such a data structure allows the object to be securely transmitted from user to user. Although the data structure shown in FIG. 3 is set forth in a general format, it may be structured as set forth in the inventor's copending application filed on Apr. 6, 1992 and entitled 'Method and Apparatus for Creating, Supporting and Processing a Travelling Program' (U.S. Ser. No. 07/863,552.), which application is hereby expressly incorporated herein by reference. . . . The program authorization information is embedded in a segment 116 which specifies the authorization for the object's program or programs in a manner to be described more fully hereinafter." *Fischer* at 7:14-18, 7:49-8:2. "Thereafter, the PAI is stored using, for example, one of the approaches set forth in FIGS. 3A through 3D so that it is associated with its program 272 . . . ." *Fischer* at 15:24-26.

U.S. Patent No. 6,192,476 – Claim 1	*Fischer*
	 *Fischer* at Fig. 3C. "FIGS. 10 and 11 illustrate the sequence of operations of a supervisor program for controlling the processing of a program being executed in accordance with its program authorization information." *Fischer* at 15:56-59. "Depending on the processing in block 316, a decision is made in block 322 whether the signatures are valid, authorized and trusted. If the signatures are not determined to be valid, then the routing branches to block 324 where the execution in program X is suppressed." *Fischer* at 16:66-17:3. "If the processing in blocks 322 and 316 reveal that the signatures are valid, then the processing in block 326 is performed." *Fischer* at 17:31-33. "FIG. 5 is an illustration of a program control block (PCB) data structure 140 in accordance with an exemplary embodiment of the present invention. The program control block 140 is the data structure utilized by the system monitor to control the execution of an associated

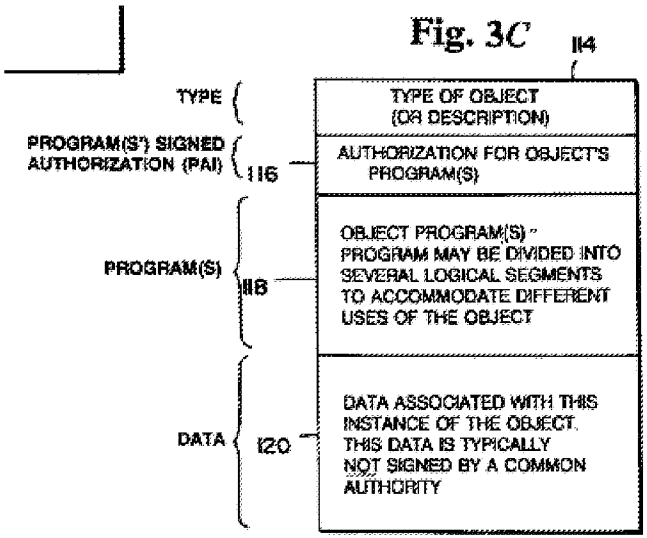
U.S. Patent No. 6,192,476 – Claim 1	*Fischer*
	program. The program control block 140 is loaded with program authorization information such that the PAI can be readily referenced as the associated program is executed so as to insure that the program performs functions and accesses resources in conformance with its assigned authorizations. The program control block associated with the program to be executed is located in a storage area which cannot be modified by the program. As shown in FIG. 5, an originating program (whose PCB is identified at 180) calls a program (having a PCB 170) which will, in turn, will call the program 140 is shown in detail in FIG. 5. Each new PCB will include a field such as 150 that points to the 'previous' or calling program control block. A field may also be utilized to identify the 'next' program control block file. When a called program finishes executing, the system removes its associated PCB from the top of the executed stack, removes the associated program from storage, removes the associated authorizing information and accesses the program control block immediately below it in the stack. When another program is called, the reverse process occurs such that a new PCB is created which is placed on top of the stack, which again points to the previous PCB as indicated in field 150." *Fischer* at 10:8-39.

U.S. Patent No. 6,192,476 – Claim 1	*Fischer*
	Fig. 5  *Fischer* at Fig. 5. "Thereafter, the program X's program authorizing information is combined, as appropriate, with the PAI associated with the PCB of the calling program, if any. This combined PAI, which may include multiple PAI's, is then stored in an area of storage which cannot generally be modified by the program and the address of the PAI is stored in the process control block (PCB) as indicated in field 156 of FIG. 5. Thus, if program X is called by a calling program, it is subject to all its own constraints as well as being combined in some way with the constraints of the calling program, which aggregate constraints are embodied into program X's PAI. In this fashion, a calling program may not be permitted to exceed its

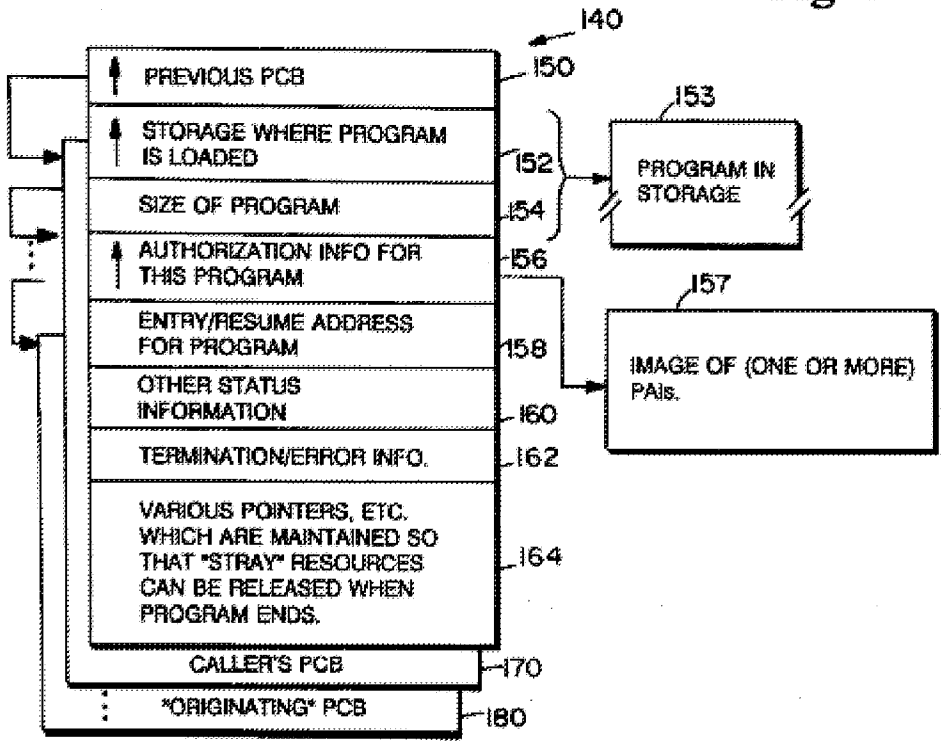
U.S. Patent No. 6,192,476 – Claim 1	*Fischer*
	assigned bounds by merely calling another program. There are many alternative ways that a program's PAI could be combined with the PAI of the program which invokes it--depending on the strategies which are applicable to the current environment, and the inherent nature of the programs themselves. It may even be likely that even the method of combination is itself one of the PAI authorities, or qualifiers, of either or both the invoking or invoked program. For example, it is reasonable to restrict a called program to the lesser of its 'normal' PAI authority and that of its calling program--to insure the calling program cannot mischievously misuse the called program's greater authority to circumvent its own limitations. On the other hand, for called programs which carefully verify their own actions, it could be possible to allow the called program greater inherent authority than the program which calls it--this way sensitive resources could be made available to wider use by mediating such use through trusted sub-programs. The possibilities for such combination must be carefully considered, not only by the designers of the underlying control system, but also by those who assign authority to each program. Thereafter, the program is loaded and the hash of the program is computed based on the algorithm specified in the program's PAI." *Fischer* at 17:40-18:10.

U.S. Patent No. 6,192,476 – Claim 2	*Fischer*
2. The method of claim 1, wherein: the step of detecting when a request for an action is made includes detecting when a request for an action is made by a thread; and	*Fischer* discloses the method of claim 1, wherein: the step of detecting when a request for an action is made includes detecting when a request for an action is made by a thread. *Fischer* inquires into the permissions of a given principal's PAI upon the principal's request for an action: "When the program is to perform a function or access a resource, the associated PAI is monitored to confirm that the operation is within the defined program limits. If the program attempts to do anything outside the authorized limits, then the program execution is halted." *Id.* at 2:43-48. Thus it inherent that *Fischer* detects a request of an action made by a principal. *Fischer* discloses that the invention can be directed to complex data structures, which would include a "thread." "The present invention is directed to providing reliable security, even when operating with complex data structures, e.g., objects, containing their own program instructions, which are transmitted among users. The present invention also provides enhanced security when

U.S. Patent No. 6,192,476 – Claim 2	*Fischer*
	processing more conventional programs, even those of questionable origin, e.g., from a computer bulletin board, without exposing system programs or data to the potentially catastrophic consequences of computer viruses or of incompetent programming. The present method and apparatus utilizes a unique operating system design that includes a system monitor which limits the ability of a program about to be executed to the use of predefined resources (e.g., data files, disk writing capabilities etc.). The system monitor builds a data structure including a set of authorities defining that which a program is permitted to do and/or that which the program is precluded from doing." *Fischer* at 2:6-23. "In accordance with the present invention, a PAI is associated with programs to be executed. FIGS. 3A through 3D depict four exemplary approaches for associating program authorization information with a program. . . . . . . FIG. 3C shows an important application in which a PAI data structure is associated with a program according to an embodiment of the present invention. FIG. 3C shows an illustrative data structure for a secure exchangeable 'object'. The data structure may be signed by a trusted authority. The signing of such a data structure allows the object to be securely transmitted from user to user. Although the data structure shown in FIG. 3 is set forth in a general format, it may be structured as set forth in the inventor's copending application filed on Apr. 6, 1992 and entitled 'Method and Apparatus for Creating, Supporting and Processing a Travelling Program' (U.S. Ser. No. 07/863,552.), which application is hereby expressly incorporated herein by reference. . . . The program authorization information is embedded in a segment 116 which specifies the authorization for the object's program or programs in a manner to be described more fully hereinafter." *Fischer* at 7:14-18, 7:49-8:2. "Thereafter, the PAI is stored using, for example, one of the approaches set forth in FIGS. 3A through 3D so that it is associated with its program 272 . . . ." *Fischer* at 15:24-26.

U.S. Patent No. 6,192,476 – Claim 2	*Fischer*
	 *Fischer* at Fig. 3C.
the step of determining whether said action is authorized includes determining whether said action is authorized based on an association between permissions and a plurality of routines in a calling hierarchy associated with said thread.	*Fischer* discloses the step of determining whether said action is authorized includes determining whether said action is authorized based on an association between permissions and a plurality of routines in a calling hierarchy associated with said thread. *Fischer* discloses "an originating program" that "calls a program" having a program control block, or PCB. *Fischer* at 10:24-26. "Each new PCB will include a field . . . that points to the 'previous' of calling program control block." *See id.* at 10:27-29. Then, "[w]hen a called program finishes executing, the system removes its associated PCB from the top of the executed stack, removes the associated program from storage, removes the associated authorizing information and accesses the program control block immediately below it in the stack." *See id.* at 10:31-36. *Fischer* also incorporates by reference two patents which explicitly disclose data hierarchies: *see id.* at 6:37 (incorporating by reference U.S. Patent Nos. 4,868,877 and 5,005,200 which disclose an "enhanced digital signature certification" which employs "[a] hierarchy of nested certifications and signatures." *See* U.S. Patent No. 5,005,200, Abstract.

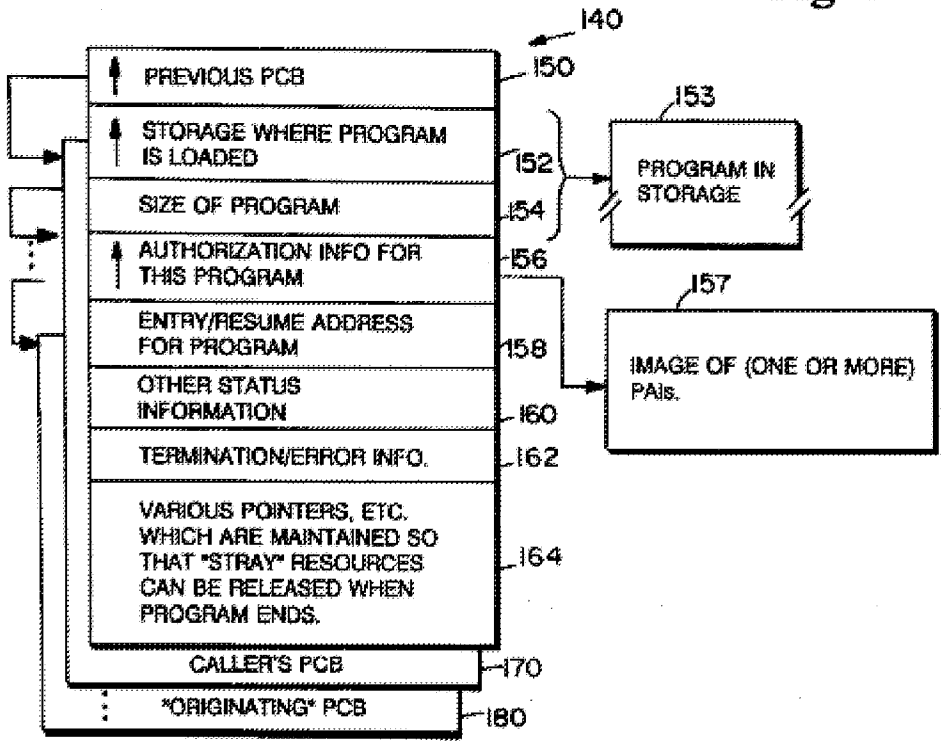
U.S. Patent No. 6,192,476 – Claim 2	*Fischer*
	"FIG. 5 is an illustration of a program control block (PCB) data structure 140 in accordance with an exemplary embodiment of the present invention. The program control block 140 is the data structure utilized by the system monitor to control the execution of an associated program. The program control block 140 is loaded with program authorization information such that the PAI can be readily referenced as the associated program is executed so as to insure that the program performs functions and accesses resources in conformance with its assigned authorizations. The program control block associated with the program to be executed is located in a storage area which cannot be modified by the program. As shown in FIG. 5, an originating program (whose PCB is identified at 180) calls a program (having a PCB 170) which will, in turn, will call the program 140 is shown in detail in FIG. 5. Each new PCB will include a field such as 150 that points to the 'previous' or calling program control block. A field may also be utilized to identify the 'next' program control block file. When a called program finishes executing, the system removes its associated PCB from the top of the executed stack, removes the associated program from storage, removes the associated authorizing information and accesses the program control block immediately below it in the stack. When another program is called, the reverse process occurs such that a new PCB is created which is placed on top of the stack, which again points to the previous PCB as indicated in field 150." *Fischer* at 10:8-39.

U.S. Patent No. 6,192,476 – Claim 2	*Fischer*
	Fig. 5  *Fischer* at Fig. 5. "Thereafter, the program X's program authorizing information is combined, as appropriate, with the PAI associated with the PCB of the calling program, if any. This combined PAI, which may include multiple PAI's, is then stored in an area of storage which cannot generally be modified by the program and the address of the PAI is stored in the process control block (PCB) as indicated in field 156 of FIG. 5. Thus, if program X is called by a calling program, it is subject to all its own constraints as well as being combined in some way with the constraints of the calling program, which aggregate constraints are embodied into program X's PAI. In this fashion, a calling program may not be permitted to exceed its

U.S. Patent No. 6,192,476 – Claim 2	*Fischer*
	assigned bounds by merely calling another program. There are many alternative ways that a program's PAI could be combined with the PAI of the program which invokes it--depending on the strategies which are applicable to the current environment, and the inherent nature of the programs themselves. It may even be likely that even the method of combination is itself one of the PAI authorities, or qualifiers, of either or both the invoking or invoked program. For example, it is reasonable to restrict a called program to the lesser of its 'normal' PAI authority and that of its calling program--to insure the calling program cannot mischievously misuse the called program's greater authority to circumvent its own limitations. On the other hand, for called programs which carefully verify their own actions, it could be possible to allow the called program greater inherent authority than the program which calls it--this way sensitive resources could be made available to wider use by mediating such use through trusted sub-programs. The possibilities for such combination must be carefully considered, not only by the designers of the underlying control system, but also by those who assign authority to each program. Thereafter, the program is loaded and the hash of the program is computed based on the algorithm specified in the program's PAI." *Fischer* at 17:40-18:10.

U.S. Patent No. 6,192,476 – Claim 3	*Fischer*
3. The method of claim 1, wherein: the calling hierarchy includes a first routine; and	*Fischer* discloses the method of claim 1, wherein: the calling hierarchy includes a first routine. For example, *Fischer* discloses calling a plurality of routines: "As shown in FIG. 5, an originating program (whose PCB is identified at 180) calls a program (having a PCB 170) which will, in turn, will call the program 140 is shown in detail in FIG. 5." *Fischer* at 10:24-27. "FIGS. 10 and 11 illustrate the sequence of operations of a supervisor program for controlling the processing of a program being executed in accordance with its program authorization information. The processing of a program 'X' and its program authorization information illustrated in FIG. 10 is initiated while the computer is executing a supervisor routine." *Fischer* at 15:56-62.

U.S. Patent No. 6,192,476 – Claim 3	*Fischer*
	"FIG. 5 is an illustration of a program control block (PCB) data structure 140 in accordance with an exemplary embodiment of the present invention. The program control block 140 is the data structure utilized by the system monitor to control the execution of an associated program. The program control block 140 is loaded with program authorization information such that the PAI can be readily referenced as the associated program is executed so as to insure that the program performs functions and accesses resources in conformance with its assigned authorizations. The program control block associated with the program to be executed is located in a storage area which cannot be modified by the program. As shown in FIG. 5, an originating program (whose PCB is identified at 180) calls a program (having a PCB 170) which will, in turn, will call the program 140 is shown in detail in FIG. 5. Each new PCB will include a field such as 150 that points to the 'previous' or calling program control block. A field may also be utilized to identify the 'next' program control block file. When a called program finishes executing, the system removes its associated PCB from the top of the executed stack, removes the associated program from storage, removes the associated authorizing information and accesses the program control block immediately below it in the stack. When another program is called, the reverse process occurs such that a new PCB is created which is placed on top of the stack, which again points to the previous PCB as indicated in field 150." *Fischer* at 10:8-39.

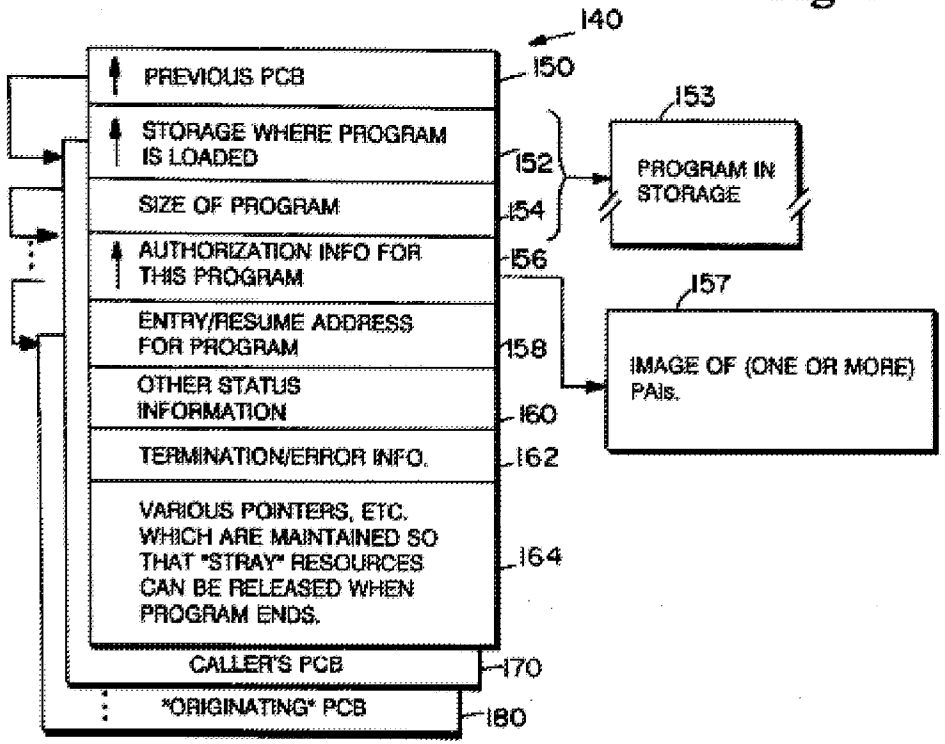
U.S. Patent No. 6,192,476 – Claim 3	*Fischer*
	Fig. 5  *Fischer* at Fig. 5. "FIG. 3C shows an important application in which a PAI data structure is associated with a program according to an embodiment of the present invention. FIG. 3C shows an illustrative data structure for a secure exchangeable 'object'. The data structure may be signed by a trusted authority. The signing of such a data structure allows the object to be securely transmitted from user to user. Although the data structure shown in FIG. 3 is set forth in a general format, it may be structured as set forth in the inventor's copending application filed on Apr. 6, 1992 and entitled 'Method and Apparatus for Creating, Supporting and Processing a Travelling Program' (U.S. Ser. No. 07/863,552.), which application is hereby

U.S. Patent No. 6,192,476 – Claim 3	*Fischer*
	expressly incorporated herein by reference. The data structure includes a header segment 114 which, by way of example only, may define the type of object that follows, e.g., a purchase order related object or any other type of electronic digital object. The program authorization information is embedded in a segment 116 which specifies the authorization for the object's program or programs in a manner to be described more fully hereinafter. The data structure includes an object program(s) segment 118, which for example, may control the manner in which an associated purchase order is displayed so as to leave blanks for variable fields which are interactively completed by the program user. The object program might store such data and send a copy of itself together with accompanying data in a manner which is described in detail in the applicant's above-identified copending application. As indicated in FIG. 3C, the program may be divided into several logical segments to accommodate different uses of the object. For example, the program may present a different display to the creator of a digital purchase order, than it displays to subsequent recipients. When the program is received by a recipient designated by the program, the recipient invokes a copy of the transmitted program to, for example, control the display of the purchase order tailored to the needs of the recipient." *Fischer* at 7:49-8:20.
the step of determining whether said action is authorized further includes determining whether a permission required to perform said action is encompassed by at least one permission associated with said first routine.	*Fischer* discloses the step of determining whether said action is authorized further includes determining whether a permission required to perform said action is encompassed by at least one permission associated with said first routine. *Fischer* utilizes the PAI data structure associated with a program to determine whether the requested action is authorized, as explained in further detail below. "FIG. 3C shows an important application in which a PAI data structure is associated with a program according to an embodiment of the present invention. FIG. 3C shows an illustrative data structure for a secure exchangeable 'object'. The data structure may be signed by a trusted authority. The signing of such a data structure allows the object to be securely transmitted from user to user. Although the data structure shown in FIG. 3 is set forth in a general format, it may be structured as set forth in the inventor's copending application filed

U.S. Patent No. 6,192,476 – Claim 3	*Fischer*
	on Apr. 6, 1992 and entitled 'Method and Apparatus for Creating, Supporting and Processing a Travelling Program' (U.S. Ser. No. 07/863,552.), which application is hereby expressly incorporated herein by reference. The data structure includes a header segment 114 which, by way of example only, may define the type of object that follows, e.g., a purchase order related object or any other type of electronic digital object. The program authorization information is embedded in a segment 116 which specifies the authorization for the object's program or programs in a manner to be described more fully hereinafter." *Fischer* at 7:49-8:2. "FIGS. 10 and 11 illustrate the sequence of operations of a supervisor program for controlling the processing of a program being executed in accordance with its program authorization information. The processing of a program 'X' and its program authorization information illustrated in FIG. 10 is initiated while the computer is executing a supervisor routine. As shown in FIG. 10 at 300, a calling program calls program X for execution. Thereafter, a program control block is created for program X. The program control block created will not be added to the top of the execution stack until it is determined that the program is permitted to be invoked and verification is successful completed. Thus, if the program fails a security check, it will not be placed in the program execution chain. In addition to creating a 'tentative' program control block, the called program will be located through an appropriate program directory during the processing in block 302. Thereafter, a check is made at block 304 to determine whether PAI has yet been associated with program X so as to place program X in the so-called 'safety box' described above. This PAI may or may not be signed depending upon its particular application as described above." *Fischer* at 15:56-16:11.

U.S. Patent No. 6,192,476 – Claim 4	*Fischer*
4. The method of claim 1, wherein the step of determining whether said action is authorized further includes determining whether a permission	*Fischer* discloses method of claim 1, wherein the step of determining whether said action is authorized further includes determining whether a permission required to perform said action is encompassed by at least one permission associated with each routine in said calling hierarchy. *Fischer* discloses an embodiment wherein PAI information is aggregated, and

U.S. Patent No. 6,192,476 – Claim 4	*Fischer*
required to perform said action is encompassed by at least one permission associated with each routine in said calling hierarchy.	access is determined based on this aggregated grouping. *Fischer* at Fig. 5. Access by a principal with an acceptable PAI within Fig. 5 of the *Fischer* disclosure inherently involves determining authorization based on at least one permission within a calling hierarchy. "As shown in FIG. 5, an originating program (whose PCB is identified at 180) calls a program (having a PCB 170) which will, in turn, will call the program 140 is shown in detail in FIG. 5. Each new PCB will include a field such as 150 that points to the 'previous' or calling program control block. A field may also be utilized to identify the 'next' program control block file. When a called program finishes executing, the system removes its associated PCB from the top of the executed stack, removes the associated program from storage, removes the associated authorizing information and accesses the program control block immediately below it in the stack. When another program is called, the reverse process occurs such that a new PCB is created which is placed on top of the stack, which again points to the previous PCB as indicated in field 150." *Fischer* at 10:23-39.

U.S. Patent No. 6,192,476 – Claim 4	*Fischer*
	Fig. 5  *Fischer* at Fig. 5. "Thereafter, the program X's program authorizing information is combined, as appropriate, with the PAI associated with the PCB of the calling program, if any. This combined PAI, which may include multiple PAI's, is then stored in an area of storage which cannot generally be modified by the program and the address of the PAI is stored in the process control block (PCB) as indicated in field 156 of FIG. 5. Thus, if program X is called by a calling program, it is subject to all its own constraints as well as being combined in some way with the constraints of the calling program, which aggregate constraints are embodied into program X's PAI. In this fashion, a calling program may not be permitted to exceed its

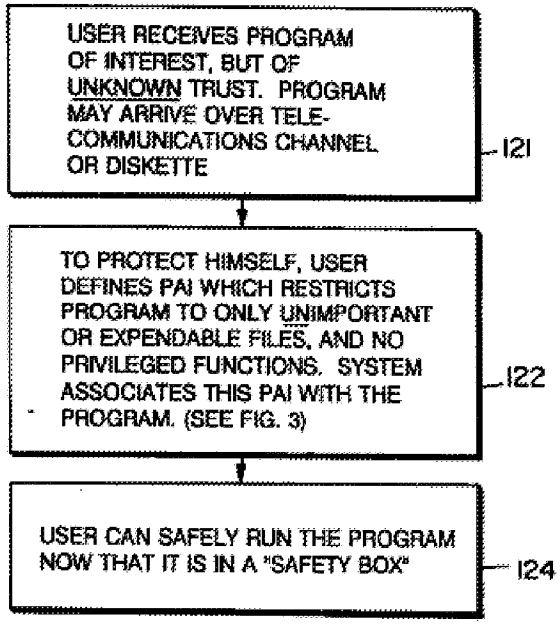
U.S. Patent No. 6,192,476 – Claim 4	*Fischer*
	assigned bounds by merely calling another program. There are many alternative ways that a program's PAI could be combined with the PAI of the program which invokes it--depending on the strategies which are applicable to the current environment, and the inherent nature of the programs themselves. It may even be likely that even the method of combination is itself one of the PAI authorities, or qualifiers, of either or both the invoking or invoked program. For example, it is reasonable to restrict a called program to the lesser of its 'normal' PAI authority and that of its calling program--to insure the calling program cannot mischievously misuse the called program's greater authority to circumvent its own limitations. On the other hand, for called programs which carefully verify their own actions, it could be possible to allow the called program greater inherent authority than the program which calls it--this way sensitive resources could be made available to wider use by mediating such use through trusted sub-programs. The possibilities for such combination must be carefully considered, not only by the designers of the underlying control system, but also by those who assign authority to each program. Thereafter, the program is loaded and the hash of the program is computed based on the algorithm specified in the program's PAI." *Fischer* at 17:40-18:10.

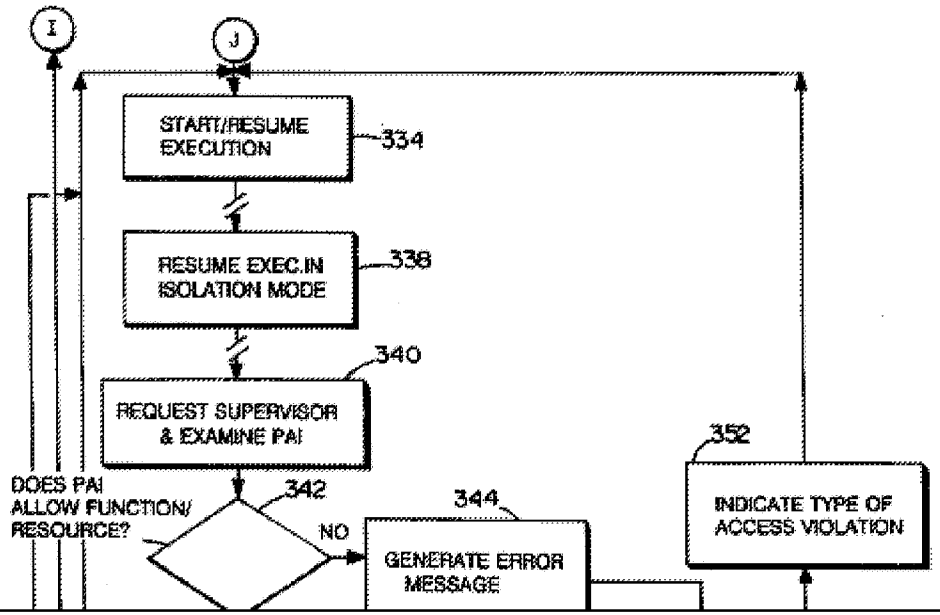
U.S. Patent No. 6,192,476 – Claim 5	*Fischer*
5. A method for providing security, the method comprising the steps of:	*Fischer* discloses a method for providing security. The Fischer disclosure employs "program authorization information," or "PAI," to limit a principal's access authority to other code and resources. "More particularly, the invention relates to a method and apparatus for providing enhanced computer system security while processing computer programs, particularly those of unknown origin, which are transmitted among users." *Fischer* at 1:20-25. "Thus, the present invention advantageously protects a user from any program to be executed. The present invention is particularly advantageous in light of current data processing practices where programs are obtained from a wide range of diverse, untrustworthy places such as computer bulletin boards or other users of unknown trustworthiness." *Fischer* at 2:49-55.

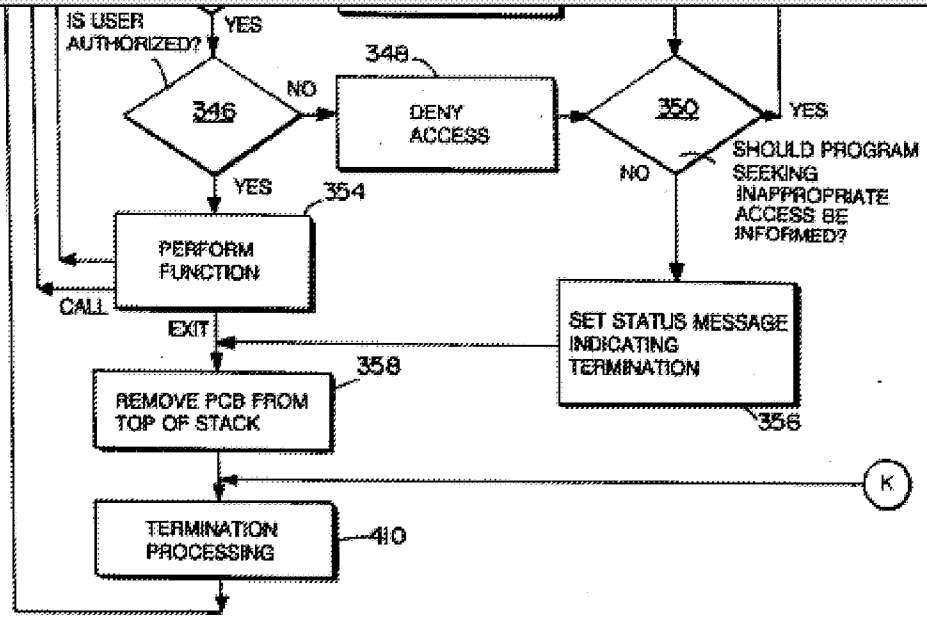
U.S. Patent No. 6,192,476 – Claim 5	*Fischer*
detecting when a request for an action is made by a principal,	*Fischer* discloses detecting when a request for an action is made by a principal. *Fischer* inquires into the permissions of a given principal's PAI upon the principal's request for an action: "When the program is to perform a function or access a resource, the associated PAI is monitored to confirm that the operation is within the defined program limits. If the program attempts to do anything outside the authorized limits, then the program execution is halted." *Id.* at 2:43-48. Thus it inherent that *Fischer* detects a request of an action made by a principal. "Prior art operating systems are typically designed to protect data from computer users. In such systems, users are often assigned various authorities and are thereafter able to execute programs based on their associated authority. If a program is executing which exceeds the user's assigned authority, then such a system will halt execution of the program. Such prior art systems do not adequately protect computer users from computer viruses or the like. There are security systems which protect certain 'system' related files from being modified by a program. However, such systems do not typically protect a computer user from a program executing and modifying the user's own files. The present invention is directed to providing reliable security, even when operating with complex data structures, e.g., objects, containing their own program instructions, which are transmitted among users. The present invention also provides enhanced security when processing more conventional programs, even those of questionable origin, e.g., from a computer bulletin board, without exposing system programs or data to the potentially catastrophic consequences of computer viruses or of incompetent programming. The present method and apparatus utilizes a unique operating system design that includes a system monitor which limits the ability of a program about to be executed to the use of predefined resources (e.g., data files, disk writing capabilities etc.). The system monitor builds a data structure including a set of authorities defining that which a program is permitted to do and/or that which the program is precluded from doing. The set of authorities and/or restrictions assigned to a program to be executed are referred to

U.S. Patent No. 6,192,476 – Claim 5	*Fischer*
	herein as 'program authorization information' (or 'PAI'). Once defined, the program authorization information is thereafter associated with each program to be executed to thereby delineate the types of resources and functions that the program is allowed to utilize. The PAI associated with a particular program may be assigned by a computer system owner/user or by someone who the computer system owner/user implicitly trusts. The PAI defines the range of operations that a program may execute and/or defines those operations that a program cannot perform. The program is permitted to access what has been authorized and nothing else. In this fashion, the program may be regarded as being placed in a program capability limiting 'safety box'. This 'safety box' is thereafter associated with the program such that whenever the system monitor runs the program, the PAI for that program is likewise loaded and monitored. When the program is to perform a function or access a resource, the associated PAI is monitored to confirm that the operation is within the defined program limits. If the program attempts to do anything outside the authorized limits, then the program execution is halted. Thus, the present invention advantageously protects a user from any program to be executed. The present invention is particularly advantageous in light of current data processing practices where programs are obtained from a wide range of diverse, untrustworthy places such as computer bulletin boards or other users of unknown trustworthiness." *Fischer* at 1:60-2:55. "FIG. 4 is a flowchart which illustrates how a user may benefit from the use of program authorization information, particularly when executing a program of unknown trustworthiness. As indicated in block 121, a user may have a desire to execute a program of interest in which the user has no knowledge of the program's creator. Thus, the program has unknown trustworthiness and may, for example, have been accessed via an electronic bulletin board and may have arrived at the user's terminal via a telecommunications channel or diskette. Such a program, which might purport to be only a game, carries with it a significant risk that it may be infected by a virus. As indicated in block 122, the user may be protected by defining program authorization

U.S. Patent No. 6,192,476 – Claim 5	*Fischer*
	information which restricts the program to only unimportant or expendable files. If desired, the user may restrict such a program from modifying any files whatsoever. For example, the user may permit the program to only display images on the display screen and to perform game playing related functions. Alternatively, if the program is known to have a single work file, the PAI data may only permit use of such a single file. By limiting access only to a single work file, a program of unknown trustworthiness, cannot inject a virus into other user's programs or otherwise initiate system program malfunctions. Thus, in accordance with the present invention, a user, via a systems program, determines how much of the user's system will be put at risk by such a program so as to, for example, completely eliminate the ability of the program to use any privileged functions. The user then associates, for example, through an operator prompting, menu-driven system, a PAI with every program to be run on the system (or have such PAI or lack of PAI associated through predetermined default mechanisms). A system utility program is preferably employed to create the program authorization information in a manner which will be described in detail below in conjunction with FIGS. 6-9. After the PAI has been assigned, any time the system runs the associated program, the system software (in a manner to be described below) insures that the program safely runs in a manner consistent with the PAI. Thus, the program has been effectively placed in a 'safety box' (124). Turning back to FIG. 1, a program of unknown trust may be injected into the system via communications channel 12 or from a floppy disk loaded into terminal A. The program may be initially stored in, for example, the user's program disk memory 7. Thereafter, the user on keyboard 4 will, through interaction with the system's program identified above (with respect to block 122 of FIG. 4), associate the program authorization information with a program (in a manner such as shown in FIGS. 3A through 3D) such that the program may safely run on the user's system or perhaps, a PAI arrives with the program, in which case it is likely to be signed. FIG. 5 is an illustration of a program control block (PCB) data structure 140 in accordance with an exemplary embodiment of the present invention. The program control block 140 is

U.S. Patent No. 6,192,476 – Claim 5	*Fischer*
	the data structure utilized by the system monitor to control the execution of an associated program. The program control block 140 is loaded with program authorization information such that the PAI can be readily referenced as the associated program is executed so as to insure that the program performs functions and accesses resources in conformance with its assigned authorizations. The program control block associated with the program to be executed is located in a storage area which cannot be modified by the program." *Fischer* at 9:17-10:23.  **Fig. 4** *Fischer* at Fig. 4. "FIGS. 10 and 11 illustrate the sequence of operations of a supervisor program for controlling the processing of a program being executed in accordance with its program authorization information. The processing of a program 'X' and its program authorization information illustrated in FIG. 10 is initiated while the computer is executing a supervisor routine. As shown in FIG. 10 at 300, a calling program calls program X for execution.

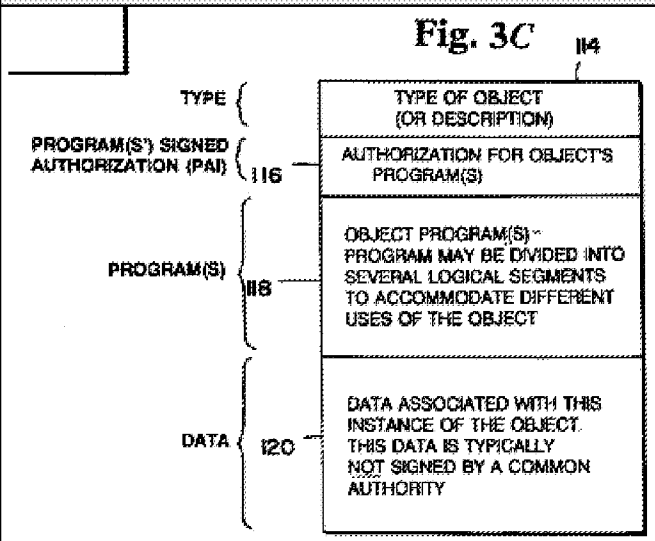
U.S. Patent No. 6,192,476 – Claim 5	*Fischer*
	Thereafter, a program control block is created for program X. The program control block created will not be added to the top of the execution stack until it is determined that the program is permitted to be invoked and verification is successful completed. Thus, if the program fails a security check, it will not be placed in the program execution chain. In addition to creating a 'tentative' program control block, the called program will be located through an appropriate program directory during the processing in block 302. Thereafter, a check is made at block 304 to determine whether PAI has yet been associated with program X so as to place program X in the so-called 'safety box' described above. This PAI may or may not be signed depending upon its particular application as described above." *Fischer* at 15:56-16:11. 

U.S. Patent No. 6,192,476 – Claim 5	*Fischer*
	 **Fig. 11** *Fischer* at Fig. 11.
determining whether said action is authorized based on an association between permissions and a plurality of routines in a calling hierarchy associated with said principal;	*Fischer* discloses determining whether said action is authorized based on an association between permissions and a plurality of routines in a calling hierarchy associated with said principal. *Fischer* discloses "an originating program" that "calls a program" having a program control block, or PCB. *Fischer* at 10:24-26. "Each new PCB will include a field . . . that points to the 'previous' of calling program control block." *See id.* at 10:27-29. Then, "[w]hen a called program finishes executing, the system removes its associated PCB from the top of the executed stack, removes the associated program from storage, removes the associated authorizing information and accesses the program control block immediately below it in the stack." *See id.* at 10:31-36. *Fischer* also incorporates by reference two patents which explicitly disclose data hierarchies: *see id.* at 6:37 (incorporating by reference U.S. Patent Nos. 4,868,877 and 5,005,200 which disclose an "enhanced digital signature certification" which employs "[a] hierarchy of nested

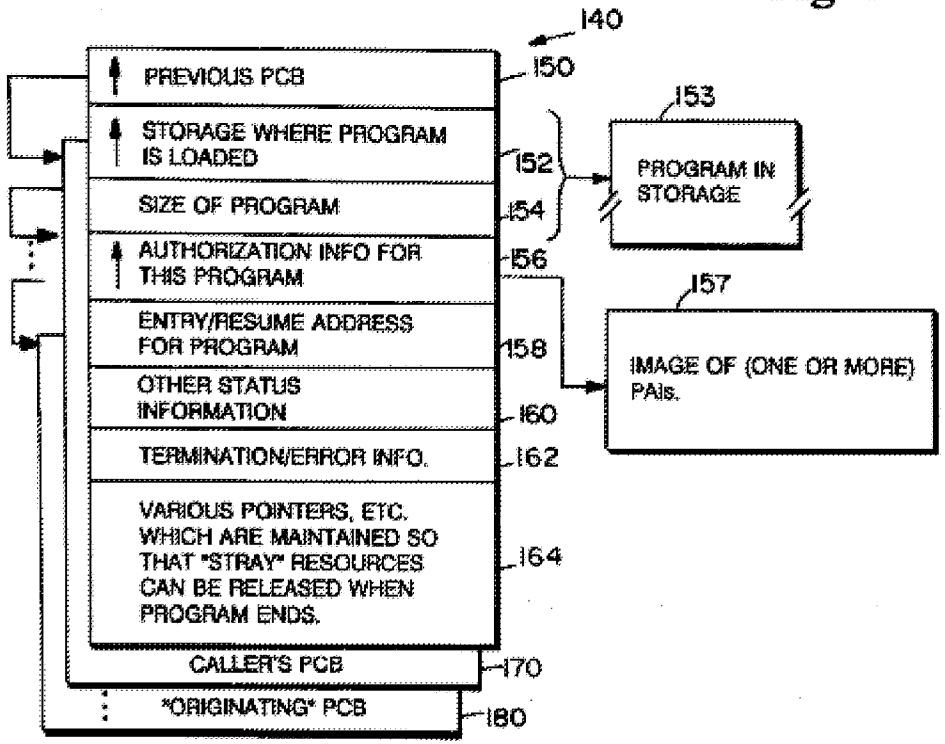
U.S. Patent No. 6,192,476 – Claim 5	*Fischer*
	certifications and signatures." *See* U.S. Patent No. 5,005,200, Abstract. "The present method an apparatus utilizes a unique operation system design that includes a system monitor which limits the ability of a program about to be executed to the use of predefined resources (e.g., data files, disk writing capabilities etc.). The system monitor builds a data structure including a set of authorities defining that which a program is permitted to do and/or that which the program is precluded from doing. The set of authorities and/or restrictions assigned to a program to be executed are referred to herein as 'program authorization information' (or 'PAI'). Once defined, the program authorization information is thereafter associated with each program to be executed to thereby delineate the types of resources and functions that the program is allowed to utilize. The PAI associated with a particular program may be assigned by a computer system owner/user or by someone who the computer system owner/user implicitly trusts. The PAI defines the range of operations that a program may execute and/or defines those operations that a program cannot perform. The program is permitted to access what has been authorized and nothing else. In this fashion, the program may be regarded as being placed in a program capability limiting 'safety box.' This 'safety box' is thereafter associated with the program such that whenever the system monitor runs the program, the PAI for that program is likewise loaded and monitored. When the program is to perform a function or access a resource, the associated PAI is monitored to confirm that the operation is within the defined program limits. If the program attempts to do anything outside the authorized limits, then the program execution is halted." *Fischer* at 2:16-48. "Thereafter, the program X's program authorizing information is combined, as appropriate, with the PAI associated with the PCB of the calling program, if any. This combined PAI, which may include multiple PAI's, is then stored in an area of storage which cannot generally be modified by the program and the address of the PAI is stored in the process control block (PCB) as indicated in field 156 of FIG. 5. Thus, if program X is called by a calling program, it is subject to all its own constraints as well as being combined in some way with the constraints of the calling program, which aggregate constraints are embodied

U.S. Patent No. 6,192,476 – Claim 5	*Fischer*
	into program X's PAI. In this fashion, a calling program may not be permitted to exceed its assigned bounds by merely calling another program." *Fischer* at 19:40-54. "Even programs with no known trustworthiness can be used after program authorization information associates a wide range of restrictions to thereby allow potentially beneficial programs to be safely used--even if they do not have an official certification of trust. The present invention also allows an unlimited number of different resources and functions to be controlled. For example, some useful resources/functions which may be controlled include: the ability to limit a program to certain files or data sets; the ability to transmit data via electronic mail to someone outside the user's domain; the ability of a program to create or solicit digital signatures; the ability to limit access to a program of certain security classes, etc." *Fischer* at 3:48-61. "The present invention is directed to providing reliable security, even when operating with complex data structures, e.g., objects, containing their own program instructions, which are transmitted among users." *Fischer* at 2:6-9. "Through the use of the present invention, general object oriented data may be transferred from user to user without exposing users to the potential dangers of viruses or mischievous users." *Fischer* at 4:10-13. "In one contemplated embodiment of the present invention, programs may be part of data objects, which are written in a high-level control language and are executed by a standardized interpreter program which executes this high-level language. In this case, part of the interpreter's task is to verify that the functions encountered in the high level logic are, in fact, permissible. If such tasks are not permissible, the interpreter then suppresses the execution of the program not authorized to perform such tasks." *Fischer* at 3:11-20. "In accordance with the present invention, a PAI is associated with programs to be executed. FIGS. 3A through 3D depict four exemplary approaches for associating program authorization information with a program. . . .

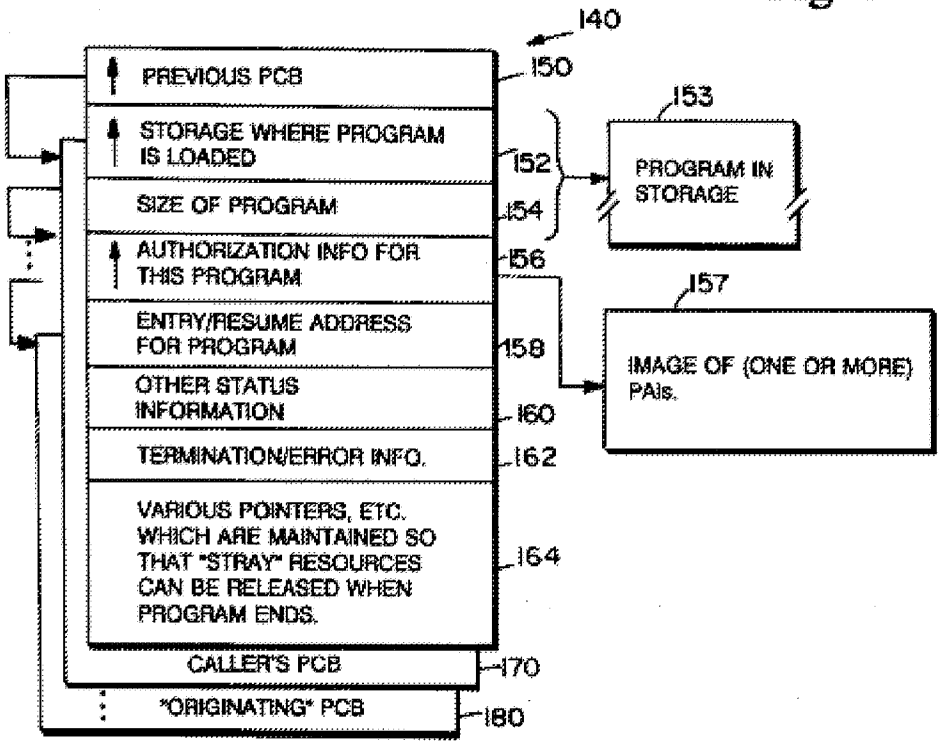
U.S. Patent No. 6,192,476 – Claim 5	*Fischer*
	. . . FIG. 3C shows an important application in which a PAI data structure is associated with a program according to an embodiment of the present invention. FIG. 3C shows an illustrative data structure for a secure exchangeable 'object'. The data structure may be signed by a trusted authority. The signing of such a data structure allows the object to be securely transmitted from user to user. Although the data structure shown in FIG. 3 is set forth in a general format, it may be structured as set forth in the inventor's copending application filed on Apr. 6, 1992 and entitled 'Method and Apparatus for Creating, Supporting and Processing a Travelling Program' (U.S. Ser. No. 07/863,552.), which application is hereby expressly incorporated herein by reference. . . . The program authorization information is embedded in a segment 116 which specifies the authorization for the object's program or programs in a manner to be described more fully hereinafter." *Fischer* at 7:14-18, 7:49-8:2. "Thereafter, the PAI is stored using, for example, one of the approaches set forth in FIGS. 3A through 3D so that it is associated with its program 272 . . . ." *Fischer* at 15:24-26.

U.S. Patent No. 6,192,476 – Claim 5	*Fischer*
	 *Fischer* at Fig. 3C. "FIGS. 10 and 11 illustrate the sequence of operations of a supervisor program for controlling the processing of a program being executed in accordance with its program authorization information." *Fischer* at 15:56-59. "Depending on the processing in block 316, a decision is made in block 322 whether the signatures are valid, authorized and trusted. If the signatures are not determined to be valid, then the routing branches to block 324 where the execution in program X is suppressed." *Fischer* at 16:66-17:3. "If the processing in blocks 322 and 316 reveal that the signatures are valid, then the processing in block 326 is performed." *Fischer* at 17:31-33. "FIG. 5 is an illustration of a program control block (PCB) data structure 140 in accordance with an exemplary embodiment of the present invention. The program control block 140 is the data structure utilized by the system monitor to control the execution of an associated

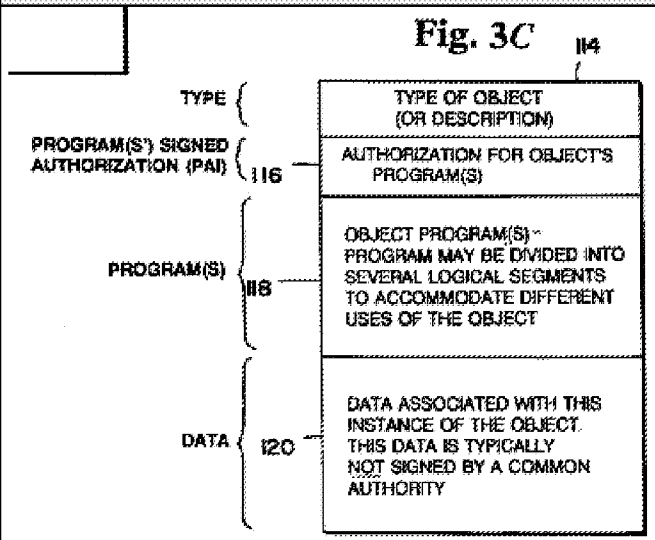
U.S. Patent No. 6,192,476 – Claim 5	*Fischer*
	program. The program control block 140 is loaded with program authorization information such that the PAI can be readily referenced as the associated program is executed so as to insure that the program performs functions and accesses resources in conformance with its assigned authorizations. The program control block associated with the program to be executed is located in a storage area which cannot be modified by the program. As shown in FIG. 5, an originating program (whose PCB is identified at 180) calls a program (having a PCB 170) which will, in turn, will call the program 140 is shown in detail in FIG. 5. Each new PCB will include a field such as 150 that points to the 'previous' or calling program control block. A field may also be utilized to identify the 'next' program control block file. When a called program finishes executing, the system removes its associated PCB from the top of the executed stack, removes the associated program from storage, removes the associated authorizing information and accesses the program control block immediately below it in the stack. When another program is called, the reverse process occurs such that a new PCB is created which is placed on top of the stack, which again points to the previous PCB as indicated in field 150." *Fischer* at 10:8-39.

U.S. Patent No. 6,192,476 – Claim 5	*Fischer*
	Fig. 5  *Fischer* at Fig. 5. "Thereafter, the program X's program authorizing information is combined, as appropriate, with the PAI associated with the PCB of the calling program, if any. This combined PAI, which may include multiple PAI's, is then stored in an area of storage which cannot generally be modified by the program and the address of the PAI is stored in the process control block (PCB) as indicated in field 156 of FIG. 5. Thus, if program X is called by a calling program, it is subject to all its own constraints as well as being combined in some way with the constraints of the calling program, which aggregate constraints are embodied into program X's PAI. In this fashion, a calling program may not be permitted to exceed its

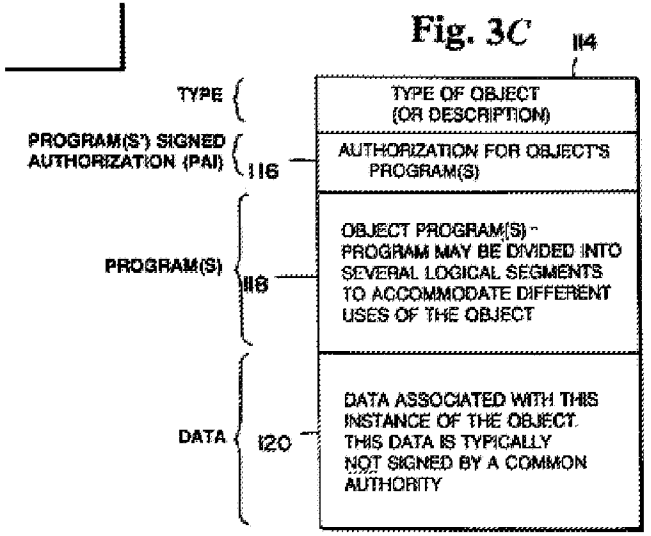
U.S. Patent No. 6,192,476 – Claim 5	*Fischer*
	assigned bounds by merely calling another program. There are many alternative ways that a program's PAI could be combined with the PAI of the program which invokes it--depending on the strategies which are applicable to the current environment, and the inherent nature of the programs themselves. It may even be likely that even the method of combination is itself one of the PAI authorities, or qualifiers, of either or both the invoking or invoked program. For example, it is reasonable to restrict a called program to the lesser of its 'normal' PAI authority and that of its calling program--to insure the calling program cannot mischievously misuse the called program's greater authority to circumvent its own limitations. On the other hand, for called programs which carefully verify their own actions, it could be possible to allow the called program greater inherent authority than the program which calls it--this way sensitive resources could be made available to wider use by mediating such use through trusted sub-programs. The possibilities for such combination must be carefully considered, not only by the designers of the underlying control system, but also by those who assign authority to each program. Thereafter, the program is loaded and the hash of the program is computed based on the algorithm specified in the program's PAI." *Fischer* at 17:40-18:10.
wherein each routine of said plurality of routines is associated with a class; and	*Fischer* discloses wherein each routine of said plurality of routines is associated with a class. For example, *Fischer* discloses that "[w]hen another program is called, the reverse process occurs such that a new PCB is created which is placed on top of the stack, which again points to the previous PCB as indicated in field 150." *Fischer* at 10:36-39. Such a program, which inherently implements code to be implemented on a stack, would also include a class. "As shown in FIG. 5, an originating program (whose PCB is identified at 180) calls a program (having a PCB 170) which will, in turn, will call the program 140 is shown in detail in FIG. 5. Each new PCB will include a field such as 150 that points to the 'previous' or calling program control block. A field may also be utilized to identify the 'next' program control block file. When a called program finishes executing, the system removes its associated PCB from the top of the executed stack, removes the associated program from storage, removes the

U.S. Patent No. 6,192,476 – Claim 5	*Fischer*
	associated authorizing information and accesses the program control block immediately below it in the stack. When another program is called, the reverse process occurs such that a new PCB is created which is placed on top of the stack, which again points to the previous PCB as indicated in field 150." *Fischer* at 10:23-39. Fig. 5  *Fischer* at Fig. 5. "In accordance with the present invention, a PAI is associated with programs to be executed. FIGS. 3A through 3D depict four exemplary approaches for associating program authorization information with a program. . . .

U.S. Patent No. 6,192,476 – Claim 5	*Fischer*
	. . . FIG. 3C shows an important application in which a PAI data structure is associated with a program according to an embodiment of the present invention. FIG. 3C shows an illustrative data structure for a secure exchangeable 'object'. The data structure may be signed by a trusted authority. The signing of such a data structure allows the object to be securely transmitted from user to user. Although the data structure shown in FIG. 3 is set forth in a general format, it may be structured as set forth in the inventor's copending application filed on Apr. 6, 1992 and entitled 'Method and Apparatus for Creating, Supporting and Processing a Travelling Program' (U.S. Ser. No. 07/863,552.), which application is hereby expressly incorporated herein by reference. . . . The program authorization information is embedded in a segment 116 which specifies the authorization for the object's program or programs in a manner to be described more fully hereinafter." *Fischer* at 7:14-18, 7:49-8:2. "Thereafter, the PAI is stored using, for example, one of the approaches set forth in FIGS. 3A through 3D so that it is associated with its program 272 . . . ." *Fischer* at 15:24-26.

U.S. Patent No. 6,192,476 – Claim 5	*Fischer*
	 *Fischer* at Fig. 3C.
wherein said association between permissions and said plurality of routines is based on a second association between classes and protection domains.	*Fischer* discloses wherein said association between permissions and said plurality of routines is based on a second association between classes and protection domains. *Fischer* discloses "an originating program" that "calls a program" having a program control block, or PCB. *Fischer* at 10:24-26. "Each new PCB will include a field . . . that points to the 'previous' of calling program control block." *See id.* at 10:27-29. Then, "[w]hen a called program finishes executing, the system removes its associated PCB from the top of the executed stack, removes the associated program from storage, removes the associated authorizing information and accesses the program control block immediately below it in the stack." *See id.* at 10:31-36. In this disclosure, the "program control block immediately below [the first control block] in the stack" is directly analogous to the second association between classes and protection domains. "Once defined, the program authorization information [(PAI)] is thereafter associated with each program to be executed to thereby delineate the types of resources and functions that the program is allowed to utilize. The PAI associated with a particular program may be assigned by a computer system owner/user or by someone who the computer system

U.S. Patent No. 6,192,476 – Claim 5	*Fischer*
	owner/user implicitly trusts." *Fischer* at 2:26-33. "The present invention is directed to providing reliable security, even when operating with complex data structures, e.g., objects, containing their own program instructions, which are transmitted among users." *Fischer* at 2:6-9. "Through the use of the present invention, general object oriented data may be transferred from user to user without exposing users to the potential dangers of viruses or mischievous users." *Fischer* at 4:10-13. "In one contemplated embodiment of the present invention, programs may be part of data objects, which are written in a high-level control language and are executed by a standardized interpreter program which executes this high-level language. In this case, part of the interpreter's task is to verify that the functions encountered in the high level logic are, in fact, permissible. If such tasks are not permissible, the interpreter then suppresses the execution of the program not authorized to perform such tasks." *Fischer* at 3:11-20. "In accordance with the present invention, a PAI is associated with programs to be executed. FIGS. 3A through 3D depict four exemplary approaches for associating program authorization information with a program. . . . . . . FIG. 3C shows an important application in which a PAI data structure is associated with a program according to an embodiment of the present invention. FIG. 3C shows an illustrative data structure for a secure exchangeable 'object'. The data structure may be signed by a trusted authority. The signing of such a data structure allows the object to be securely transmitted from user to user. Although the data structure shown in FIG. 3 is set forth in a general format, it may be structured as set forth in the inventor's copending application filed on Apr. 6, 1992 and entitled 'Method and Apparatus for Creating, Supporting and Processing a Travelling Program' (U.S. Ser. No. 07/863,552.), which application is hereby expressly incorporated herein by reference.

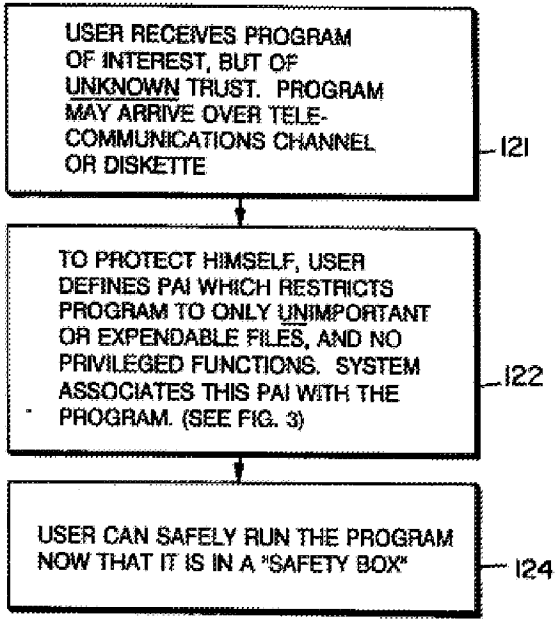
U.S. Patent No. 6,192,476 – Claim 5	*Fischer*
	. . . The program authorization information is embedded in a segment 116 which specifies the authorization for the object's program or programs in a manner to be described more fully hereinafter." *Fischer* at 7:14-18, 7:49-8:2. "Thereafter, the PAI is stored using, for example, one of the approaches set forth in FIGS. 3A through 3D so that it is associated with its program 272 . . . ." *Fischer* at 15:24-26. Fig. 3C  *Fischer* at Fig. 3C.

U.S. Patent No. 6,192,476 – Claim 6	*Fischer*
6. A method for providing security, the method comprising the steps of:	*Fischer* discloses a method for providing security. The Fischer disclosure employs "program authorization information," or "PAI," to limit a principal's access authority to other code and resources.

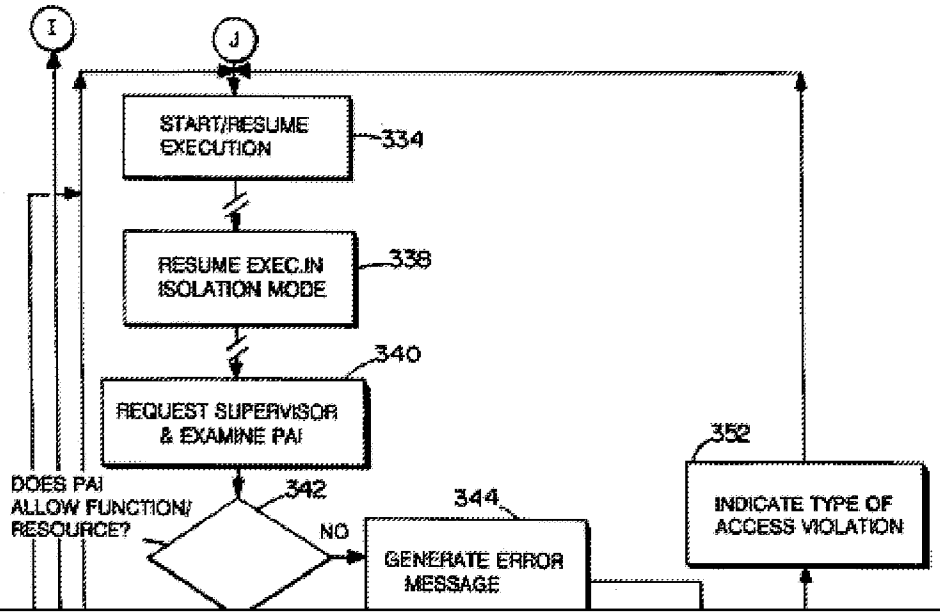
U.S. Patent No. 6,192,476 – Claim 6	*Fischer*
	"More particularly, the invention relates to a method and apparatus for providing enhanced computer system security while processing computer programs, particularly those of unknown origin, which are transmitted among users." *Fischer* at 1:20-25. "Thus, the present invention advantageously protects a user from any program to be executed. The present invention is particularly advantageous in light of current data processing practices where programs are obtained from a wide range of diverse, untrustworthy places such as computer bulletin boards or other users of unknown trustworthiness." *Fischer* at 2:49-55.
detecting when a request for an action is made by a principal; and	*Fischer* discloses detecting when a request for an action is made by a principal. *Fischer* inquires into the permissions of a given principal's PAI upon the principal's request for an action: "When the program is to perform a function or access a resource, the associated PAI is monitored to confirm that the operation is within the defined program limits. If the program attempts to do anything outside the authorized limits, then the program execution is halted." *Id.* at 2:43-48. Thus it inherent that *Fischer* detects a request of an action made by a principal. "Prior art operating systems are typically designed to protect data from computer users. In such systems, users are often assigned various authorities and are thereafter able to execute programs based on their associated authority. If a program is executing which exceeds the user's assigned authority, then such a system will halt execution of the program. Such prior art systems do not adequately protect computer users from computer viruses or the like. There are security systems which protect certain 'system' related files from being modified by a program. However, such systems do not typically protect a computer user from a program executing and modifying the user's own files. The present invention is directed to providing reliable security, even when operating with complex data structures, e.g., objects, containing their own program instructions, which are transmitted among users. The present invention also provides enhanced security when processing more conventional programs, even those of questionable origin, e.g., from a computer bulletin board, without exposing system programs or data to the potentially catastrophic consequences of computer viruses or of incompetent programming.

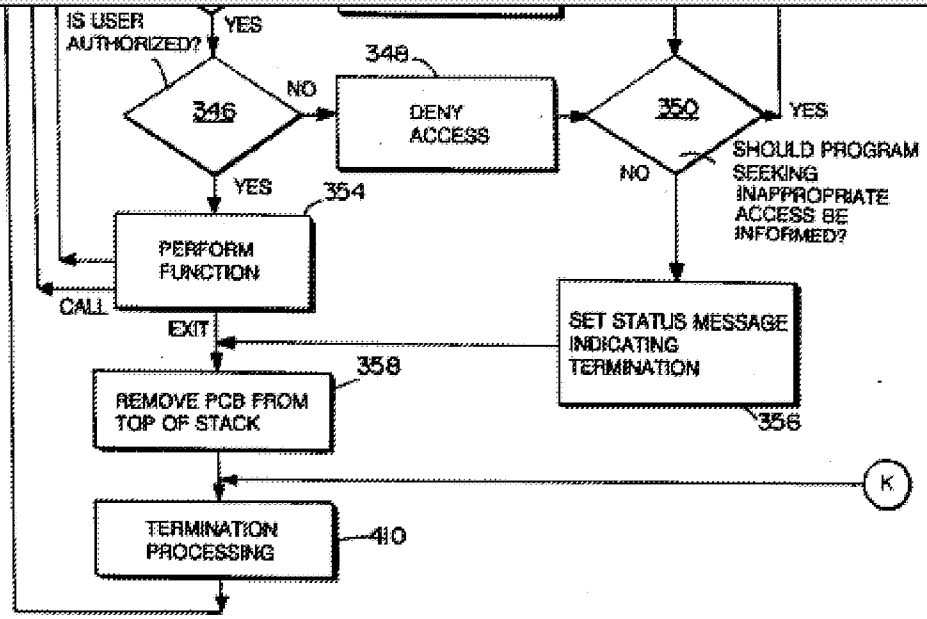
U.S. Patent No. 6,192,476 – Claim 6	*Fischer*
	The present method and apparatus utilizes a unique operating system design that includes a system monitor which limits the ability of a program about to be executed to the use of predefined resources (e.g., data files, disk writing capabilities etc.). The system monitor builds a data structure including a set of authorities defining that which a program is permitted to do and/or that which the program is precluded from doing. The set of authorities and/or restrictions assigned to a program to be executed are referred to herein as 'program authorization information' (or 'PAI'). Once defined, the program authorization information is thereafter associated with each program to be executed to thereby delineate the types of resources and functions that the program is allowed to utilize. The PAI associated with a particular program may be assigned by a computer system owner/user or by someone who the computer system owner/user implicitly trusts. The PAI defines the range of operations that a program may execute and/or defines those operations that a program cannot perform. The program is permitted to access what has been authorized and nothing else. In this fashion, the program may be regarded as being placed in a program capability limiting 'safety box'. This 'safety box' is thereafter associated with the program such that whenever the system monitor runs the program, the PAI for that program is likewise loaded and monitored. When the program is to perform a function or access a resource, the associated PAI is monitored to confirm that the operation is within the defined program limits. If the program attempts to do anything outside the authorized limits, then the program execution is halted. Thus, the present invention advantageously protects a user from any program to be executed. The present invention is particularly advantageous in light of current data processing practices where programs are obtained from a wide range of diverse, untrustworthy places such as computer bulletin boards or other users of unknown trustworthiness." *Fischer* at 1:60-2:55. "FIG. 4 is a flowchart which illustrates how a user may benefit from the use of program authorization information, particularly when executing a program of unknown trustworthiness. As indicated in block 121, a user may have a desire to execute a program of interest in which the user has no knowledge of the program's creator. Thus, the program has unknown trustworthiness and may, for example, have been accessed via an electronic bulletin board and may have arrived at the user's terminal via a telecommunications channel

U.S. Patent No. 6,192,476 – Claim 6	*Fischer*
	or diskette. Such a program, which might purport to be only a game, carries with it a significant risk that it may be infected by a virus. As indicated in block 122, the user may be protected by defining program authorization information which restricts the program to only unimportant or expendable files. If desired, the user may restrict such a program from modifying any files whatsoever. For example, the user may permit the program to only display images on the display screen and to perform game playing related functions. Alternatively, if the program is known to have a single work file, the PAI data may only permit use of such a single file. By limiting access only to a single work file, a program of unknown trustworthiness, cannot inject a virus into other user's programs or otherwise initiate system program malfunctions. Thus, in accordance with the present invention, a user, via a systems program, determines how much of the user's system will be put at risk by such a program so as to, for example, completely eliminate the ability of the program to use any privileged functions. The user then associates, for example, through an operator prompting, menu-driven system, a PAI with every program to be run on the system (or have such PAI or lack of PAI associated through predetermined default mechanisms). A system utility program is preferably employed to create the program authorization information in a manner which will be described in detail below in conjunction with FIGS. 6-9. After the PAI has been assigned, any time the system runs the associated program, the system software (in a manner to be described below) insures that the program safely runs in a manner consistent with the PAI. Thus, the program has been effectively placed in a 'safety box' (124). Turning back to FIG. 1, a program of unknown trust may be injected into the system via communications channel 12 or from a floppy disk loaded into terminal A. The program may be initially stored in, for example, the user's program disk memory 7. Thereafter, the user on keyboard 4 will, through interaction with the system's program identified above (with respect to block 122 of FIG. 4), associate the program authorization information with a program (in a manner such as shown in FIGS. 3A through 3D) such that the program may safely run on the user's system or perhaps, a PAI arrives with the program, in which case it is likely to be signed. FIG. 5 is an illustration of a program control block (PCB) data structure 140 in accordance with an exemplary embodiment of the present invention. The program control block 140 is

U.S. Patent No. 6,192,476 – Claim 6	*Fischer*
	the data structure utilized by the system monitor to control the execution of an associated program. The program control block 140 is loaded with program authorization information such that the PAI can be readily referenced as the associated program is executed so as to insure that the program performs functions and accesses resources in conformance with its assigned authorizations. The program control block associated with the program to be executed is located in a storage area which cannot be modified by the program." *Fischer* at 9:17-10:23.  **Fig. 4** *Fischer* at Fig. 4. "FIGS. 10 and 11 illustrate the sequence of operations of a supervisor program for controlling the processing of a program being executed in accordance with its program authorization information. The processing of a program 'X' and its program authorization information illustrated in FIG. 10 is initiated while the computer is executing a supervisor routine. As shown in FIG. 10 at 300, a calling program calls program X for execution. Thereafter, a program control block is created for program X. The program control block created will not be added to the top of the execution stack until it is determined that the

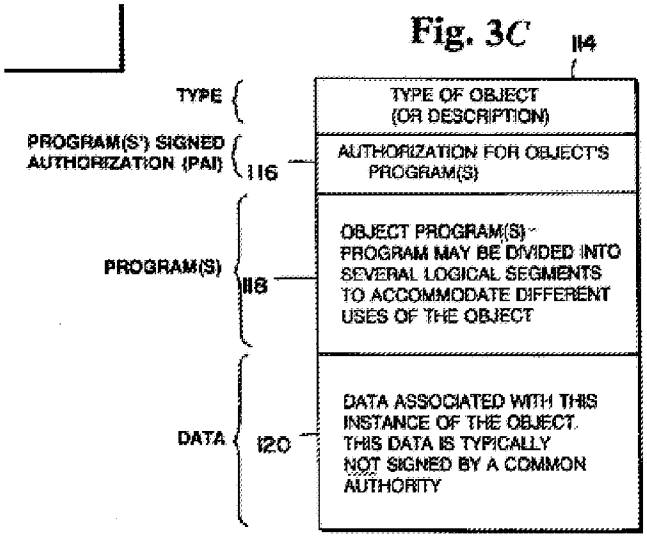
U.S. Patent No. 6,192,476 – Claim 6	*Fischer*
	program is permitted to be invoked and verification is successful completed. Thus, if the program fails a security check, it will not be placed in the program execution chain. In addition to creating a 'tentative' program control block, the called program will be located through an appropriate program directory during the processing in block 302. Thereafter, a check is made at block 304 to determine whether PAI has yet been associated with program X so as to place program X in the so-called 'safety box' described above. This PAI may or may not be signed depending upon its particular application as described above." *Fischer* at 15:56-16:11.



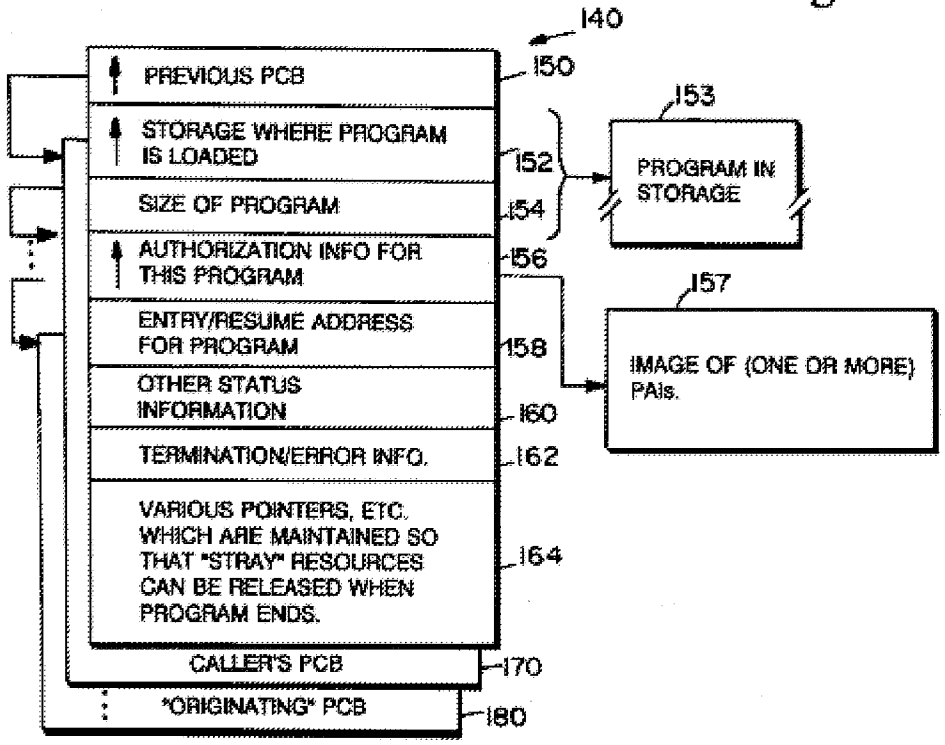
U.S. Patent No. 6,192,476 – Claim 6	*Fischer*
	 **Fig. 11** *Fischer* at Fig. 11.
in response to detecting the request, determining whether said action is authorized based on permissions associated with a plurality of routines in a calling hierarchy associated with said principal, wherein a first routine in said calling hierarchy is privileged; and	*Fischer* discloses in response to detecting the request, determining whether said action is authorized based on permissions associated with a plurality of routines in a calling hierarchy associated with said principal, wherein a first routine in said calling hierarchy is privileged. Access to a calling hierarchy is disclosed, as explained above with respect to claim 1. The limitation of "privileged" routines is also disclosed by *Fischer*; for example: "If the processing in blocks 322 and 316 reveal that the signatures are valid, then the processing in block 326 is performed." *Fischer* at 17:31-33. It is inherent that if the processing is not valid, the routine will not be performed, and is thus "privileged." "The present method an apparatus utilizes a unique operation system design that includes a system monitor which limits the ability of a program about to be executed to the use of predefined resources (e.g., data files, disk writing capabilities etc.). The system monitor

U.S. Patent No. 6,192,476 – Claim 6	*Fischer*
	builds a data structure including a set of authorities defining that which a program is permitted to do and/or that which the program is precluded from doing. The set of authorities and/or restrictions assigned to a program to be executed are referred to herein as 'program authorization information' (or 'PAI'). Once defined, the program authorization information is thereafter associated with each program to be executed to thereby delineate the types of resources and functions that the program is allowed to utilize. The PAI associated with a particular program may be assigned by a computer system owner/user or by someone who the computer system owner/user implicitly trusts. The PAI defines the range of operations that a program may execute and/or defines those operations that a program cannot perform. The program is permitted to access what has been authorized and nothing else. In this fashion, the program may be regarded as being placed in a program capability limiting 'safety box.' This 'safety box' is thereafter associated with the program such that whenever the system monitor runs the program, the PAI for that program is likewise loaded and monitored. When the program is to perform a function or access a resource, the associated PAI is monitored to confirm that the operation is within the defined program limits. If the program attempts to do anything outside the authorized limits, then the program execution is halted." *Fischer* at 2:16-48. "Thereafter, the program X's program authorizing information is combined, as appropriate, with the PAI associated with the PCB of the calling program, if any. This combined PAI, which may include multiple PAI's, is then stored in an area of storage which cannot generally be modified by the program and the address of the PAI is stored in the process control block (PCB) as indicated in field 156 of FIG. 5. Thus, if program X is called by a calling program, it is subject to all its own constraints as well as being combined in some way with the constraints of the calling program, which aggregate constraints are embodied into program X's PAI. In this fashion, a calling program may not be permitted to exceed its assigned bounds by merely calling another program." *Fischer* at 19:40-54. "Even programs with no known trustworthiness can be used after program authorization information associates a wide range of restrictions to thereby allow potentially beneficial

U.S. Patent No. 6,192,476 – Claim 6	*Fischer*
	programs to be safely used--even if they do not have an official certification of trust. The present invention also allows an unlimited number of different resources and functions to be controlled. For example, some useful resources/functions which may be controlled include: the ability to limit a program to certain files or data sets; the ability to transmit data via electronic mail to someone outside the user's domain; the ability of a program to create or solicit digital signatures; the ability to limit access to a program of certain security classes, etc." *Fischer* at 3:48-61. "The present invention is directed to providing reliable security, even when operating with complex data structures, e.g., objects, containing their own program instructions, which are transmitted among users." *Fischer* at 2:6-9. "Through the use of the present invention, general object oriented data may be transferred from user to user without exposing users to the potential dangers of viruses or mischievous users." *Fischer* at 4:10-13. "In one contemplated embodiment of the present invention, programs may be part of data objects, which are written in a high-level control language and are executed by a standardized interpreter program which executes this high-level language. In this case, part of the interpreter's task is to verify that the functions encountered in the high level logic are, in fact, permissible. If such tasks are not permissible, the interpreter then suppresses the execution of the program not authorized to perform such tasks." *Fischer* at 3:11-20. "In accordance with the present invention, a PAI is associated with programs to be executed. FIGS. 3A through 3D depict four exemplary approaches for associating program authorization information with a program. . . . . . . FIG. 3C shows an important application in which a PAI data structure is associated with a program according to an embodiment of the present invention. FIG. 3C shows an

U.S. Patent No. 6,192,476 – Claim 6	*Fischer*
	illustrative data structure for a secure exchangeable 'object'. The data structure may be signed by a trusted authority. The signing of such a data structure allows the object to be securely transmitted from user to user. Although the data structure shown in FIG. 3 is set forth in a general format, it may be structured as set forth in the inventor's copending application filed on Apr. 6, 1992 and entitled 'Method and Apparatus for Creating, Supporting and Processing a Travelling Program' (U.S. Ser. No. 07/863,552.), which application is hereby expressly incorporated herein by reference. . . . The program authorization information is embedded in a segment 116 which specifies the authorization for the object's program or programs in a manner to be described more fully hereinafter." *Fischer* at 7:14-18, 7:49-8:2. "Thereafter, the PAI is stored using, for example, one of the approaches set forth in FIGS. 3A through 3D so that it is associated with its program 272 . . . ." *Fischer* at 15:24-26. Fig. 3C  *Fischer* at Fig. 3C.

U.S. Patent No. 6,192,476 – Claim 6	*Fischer*
	"FIGS. 10 and 11 illustrate the sequence of operations of a supervisor program for controlling the processing of a program being executed in accordance with its program authorization information." *Fischer* at 15:56-59. "Depending on the processing in block 316, a decision is made in block 322 whether the signatures are valid, authorized and trusted. If the signatures are not determined to be valid, then the routing branches to block 324 where the execution in program X is suppressed." *Fischer* at 16:66-17:3. "If the processing in blocks 322 and 316 reveal that the signatures are valid, then the processing in block 326 is performed." *Fischer* at 17:31-33. "FIG. 5 is an illustration of a program control block (PCB) data structure 140 in accordance with an exemplary embodiment of the present invention. The program control block 140 is the data structure utilized by the system monitor to control the execution of an associated program. The program control block 140 is loaded with program authorization information such that the PAI can be readily referenced as the associated program is executed so as to insure that the program performs functions and accesses resources in conformance with its assigned authorizations. The program control block associated with the program to be executed is located in a storage area which cannot be modified by the program. As shown in FIG. 5, an originating program (whose PCB is identified at 180) calls a program (having a PCB 170) which will, in turn, will call the program 140 is shown in detail in FIG. 5. Each new PCB will include a field such as 150 that points to the 'previous' or calling program control block. A field may also be utilized to identify the 'next' program control block file. When a called program finishes executing, the system removes its associated PCB from the top of the executed stack, removes the associated program from storage, removes the

U.S. Patent No. 6,192,476 – Claim 6	*Fischer*
	associated authorizing information and accesses the program control block immediately below it in the stack. When another program is called, the reverse process occurs such that a new PCB is created which is placed on top of the stack, which again points to the previous PCB as indicated in field 150." *Fischer* at 10:8-39. **Fig. 5**  *Fischer* at Fig. 5. "Thereafter, the program X's program authorizing information is combined, as appropriate, with the PAI associated with the PCB of the calling program, if any. This combined PAI, which may include multiple PAI's, is then stored in an area of storage which cannot generally be modified by the program and the address of the PAI is stored in the process

U.S. Patent No. 6,192,476 – Claim 6	*Fischer*
	control block (PCB) as indicated in field 156 of FIG. 5. Thus, if program X is called by a calling program, it is subject to all its own constraints as well as being combined in some way with the constraints of the calling program, which aggregate constraints are embodied into program X's PAI. In this fashion, a calling program may not be permitted to exceed its assigned bounds by merely calling another program. There are many alternative ways that a program's PAI could be combined with the PAI of the program which invokes it--depending on the strategies which are applicable to the current environment, and the inherent nature of the programs themselves. It may even be likely that even the method of combination is itself one of the PAI authorities, or qualifiers, of either or both the invoking or invoked program. For example, it is reasonable to restrict a called program to the lesser of its 'normal' PAI authority and that of its calling program--to insure the calling program cannot mischievously misuse the called program's greater authority to circumvent its own limitations. On the other hand, for called programs which carefully verify their own actions, it could be possible to allow the called program greater inherent authority than the program which calls it--this way sensitive resources could be made available to wider use by mediating such use through trusted sub-programs. The possibilities for such combination must be carefully considered, not only by the designers of the underlying control system, but also by those who assign authority to each program. Thereafter, the program is loaded and the hash of the program is computed based on the algorithm specified in the program's PAI." *Fischer* at 17:40-18:10.
wherein the step of determining whether said action is authorized further includes determining whether a permission required to perform said action is encompassed by at least one permission associated with each routine in said calling hierarchy between and including said first routine and a second routine in said calling hierarchy, wherein said second routine is invoked after said first routine, wherein said second routine is a	*Fischer* discloses wherein the step of determining whether said action is authorized further includes determining whether a permission required to perform said action is encompassed by at least one permission associated with each routine in said calling hierarchy between and including said first routine and a second routine in said calling hierarchy, wherein said second routine is invoked after said first routine, wherein said second routine is a routine for performing said requested action. *Fischer* discloses "an originating program" that "calls a program" having a program control block, or PCB. *Fischer* at 10:24-26. "Each new PCB will include a field . . . that points to the 'previous' of calling program control block." *See id.* at 10:27-29. Then, "[w]hen a called program finishes executing, the system removes its associated PCB from the top of the executed stack, removes the associated program from storage, removes the associated authorizing information and accesses the

U.S. Patent No. 6,192,476 – Claim 6	*Fischer*
routine for performing said requested action.	program control block immediately below it in the stack." *See id.* at 10:31-36. In this disclosure, the "program control block immediately below [the first control block] in the stack" is directly analogous to the second routine in said calling hierarchy. It is also inherent that the control block below the first control block is "invoked after said first routine." *Fischer* also incorporates by reference two patents which explicitly disclose data hierarchies: *see id.* at 6:37 (incorporating by reference U.S. Patent Nos. 4,868,877 and 5,005,200 which disclose an "enhanced digital signature certification" which employs "[a] hierarchy of nested certifications and signatures." *See* U.S. Patent No. 5,005,200, Abstract. "Thereafter, the program X's program authorizing information is combined, as appropriate, with the PAI associated with the PCB of the calling program, if any. This combined PAI, which may include multiple PAI's, is then stored in an area of storage which cannot generally be modified by the program and the address of the PAI is stored in the process control block (PCB) as indicated in field 156 of FIG. 5. Thus, if program X is called by a calling program, it is subject to all its own constraints as well as being combined in some way with the constraints of the calling program, which aggregate constraints are embodied into program X's PAI. In this fashion, a calling program may not be permitted to exceed its assigned bounds by merely calling another program. There are many alternative ways that a program's PAI could be combined with the PAI of the program which invokes it--depending on the strategies which are applicable to the current environment, and the inherent nature of the programs themselves. It may even be likely that even the method of combination is itself one of the PAI authorities, or qualifiers, of either or both the invoking or invoked program. For example, it is reasonable to restrict a called program to the lesser of its 'normal' PAI authority and that of its calling program--to insure the calling program cannot mischievously misuse the called program's greater authority to circumvent its own limitations. On the other hand, for called programs which carefully verify their own actions, it could be possible to allow the called program greater inherent authority than the program which calls it--this way sensitive resources could be made available to wider use by mediating such use through trusted sub-programs. The possibilities for such combination must be carefully considered, not only by the designers of the underlying control system, but also by those who assign authority to each program. Thereafter, the program is loaded and the hash of the

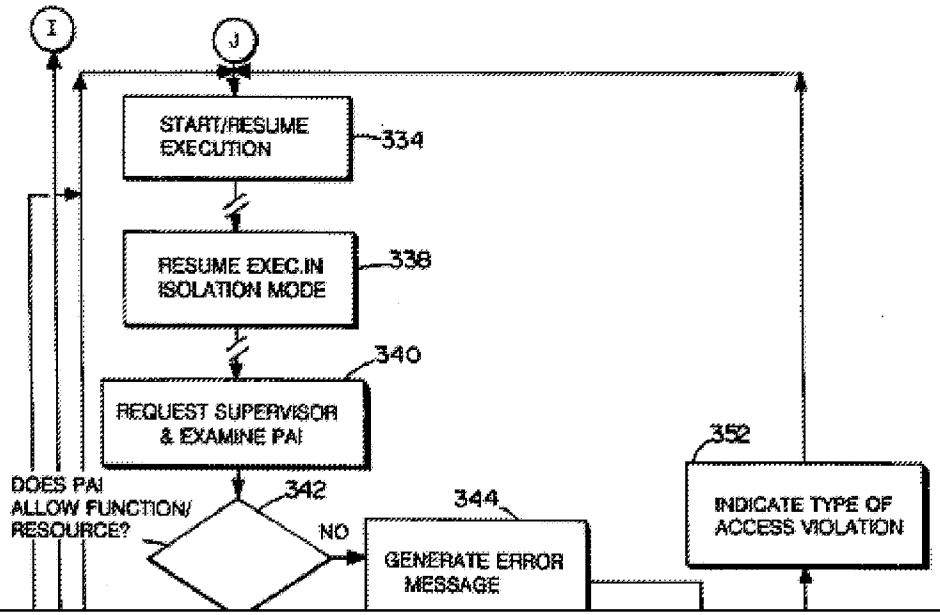
U.S. Patent No. 6,192,476 – Claim 6	*Fischer*
	program is computed based on the algorithm specified in the program's PAI." *Fischer* at 17:40-18:10. "The present invention, while it primarily focuses on defining functions which restrict the ability of a program to access resources normally allowed to users, could also, in an appropriate environment, be used to extend the capabilities beyond those normally allowed to a user. Thus, for example, programs whose PAI is signed by an authority recognized by the supervisor, could be allowed to perform extended functions." *Fischer* at 12:58-65.

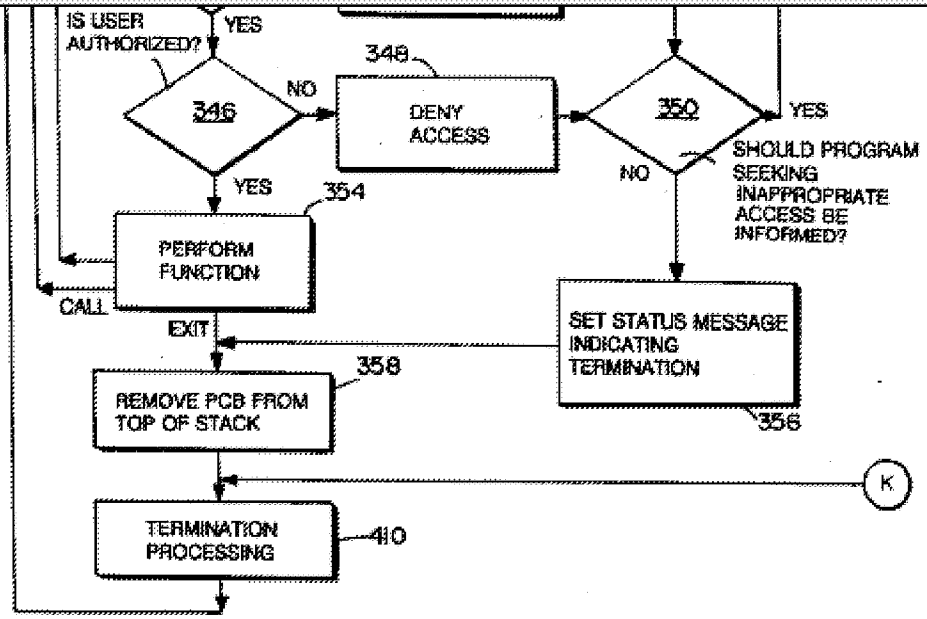
U.S. Patent No. 6,192,476 – Claim 7	*Fischer*
7. The method of claim 6, wherein the step of determining whether said permission required to perform said action is encompassed by at least one permission associated with each routine in said calling hierarchy between and including said first routine and said second routine further includes the steps of: determining whether said permission required is encompassed by at least one permission associated with said second routine; and	*Fischer* discloses the method of claim 6, wherein the step of determining whether said permission required to perform said action is encompassed by at least one permission associated with each routine in said calling hierarchy between and including said first routine and said second routine further includes the steps of: determining whether said permission required is encompassed by at least one permission associated with said second routine. *Fischer* discloses "an originating program" that "calls a program" having a program control block, or PCB. *Fischer* at 10:24-26. "Each new PCB will include a field . . . that points to the 'previous' of calling program control block." *See id.* at 10:27-29. Then, "[w]hen a called program finishes executing, the system removes its associated PCB from the top of the executed stack, removes the associated program from storage, removes the associated authorizing information and accesses the program control block immediately below it in the stack." *See id.* at 10:31-36. *Fischer* also incorporates by reference two patents which explicitly disclose data hierarchies: *see id.* at 6:37 (incorporating by reference U.S. Patent Nos. 4,868,877 and 5,005,200 which disclose an "enhanced digital signature certification" which employs "[a] hierarchy of nested certifications and signatures." *See* U.S. Patent No. 5,005,200, Abstract. "Thereafter, the program X's program authorizing information is combined, as appropriate, with the PAI associated with the PCB of the calling program, if any. This combined PAI, which may include multiple PAI's, is then stored in an area of storage which cannot generally be modified by the program and the address of the PAI is stored in the process control block (PCB) as indicated in field 156 of FIG. 5. Thus, if program X is called by a

U.S. Patent No. 6,192,476 – Claim 7	*Fischer*
	calling program, it is subject to all its own constraints as well as being combined in some way with the constraints of the calling program, which aggregate constraints are embodied into program X's PAI. In this fashion, a calling program may not be permitted to exceed its assigned bounds by merely calling another program. There are many alternative ways that a program's PAI could be combined with the PAI of the program which invokes it--depending on the strategies which are applicable to the current environment, and the inherent nature of the programs themselves. It may even be likely that even the method of combination is itself one of the PAI authorities, or qualifiers, of either or both the invoking or invoked program. For example, it is reasonable to restrict a called program to the lesser of its 'normal' PAI authority and that of its calling program--to insure the calling program cannot mischievously misuse the called program's greater authority to circumvent its own limitations. On the other hand, for called programs which carefully verify their own actions, it could be possible to allow the called program greater inherent authority than the program which calls it--this way sensitive resources could be made available to wider use by mediating such use through trusted sub-programs. The possibilities for such combination must be carefully considered, not only by the designers of the underlying control system, but also by those who assign authority to each program. Thereafter, the program is loaded and the hash of the program is computed based on the algorithm specified in the program's PAI." *Fischer* at 17:40-18:10.
in response to determining said permission required is encompassed by at least one permission associated with said second routine, then performing the steps of: A) selecting a next routine from said plurality of routines in said calling hierarchy,	*Fischer* discloses in response to determining said permission required is encompassed by at least one permission associated with said second routine, then performing the steps of: A) selecting a next routine from said plurality of routines in said calling hierarchy. For example, "Each new PCB will include a field such as 150 that points to the 'previous' or calling program control block. A field may also be utilized to identify the 'next' program control block file." *Fischer* at 10:23-30. "As shown in FIG. 5, an originating program (whose PCB is identified at 180) calls a program (having a PCB 170) which will, in turn, will call the program 140 is shown in detail in FIG. 5. Each new PCB will include a field such as 150 that points to the 'previous' or calling program control block. A field may also be utilized to identify the 'next' program control block file." *Fischer* at 10:23-30.

U.S. Patent No. 6,192,476 – Claim 7	*Fischer*
	"Thereafter, the program X's program authorizing information is combined, as appropriate, with the PAI associated with the PCB of the calling program, if any. This combined PAI, which may include multiple PAI's, is then stored in an area of storage which cannot generally be modified by the program and the address of the PAI is stored in the process control block (PCB) as indicated in field 156 of FIG. 5. Thus, if program X is called by a calling program, it is subject to all its own constraints as well as being combined in some way with the constraints of the calling program, which aggregate constraints are embodied into program X's PAI. In this fashion, a calling program may not be permitted to exceed its assigned bounds by merely calling another program. There are many alternative ways that a program's PAI could be combined with the PAI of the program which invokes it--depending on the strategies which are applicable to the current environment, and the inherent nature of the programs themselves. It may even be likely that even the method of combination is itself one of the PAI authorities, or qualifiers, of either or both the invoking or invoked program. For example, it is reasonable to restrict a called program to the lesser of its 'normal' PAI authority and that of its calling program--to insure the calling program cannot mischievously misuse the called program's greater authority to circumvent its own limitations. On the other hand, for called programs which carefully verify their own actions, it could be possible to allow the called program greater inherent authority than the program which calls it--this way sensitive resources could be made available to wider use by mediating such use through trusted sub-programs. The possibilities for such combination must be carefully considered, not only by the designers of the underlying control system, but also by those who assign authority to each program. Thereafter, the program is loaded and the hash of the program is computed based on the algorithm specified in the program's PAI." *Fischer* at 17:40-18:10. "There are many alternative ways that a program's PAI could be combined with the PAI of the program which invokes it--depending on the strategies which are applicable to the current environment, and the inherent nature of the programs themselves. It may even be likely that even the method of combination is itself one of the PAI authorities, or qualifiers, of either or both the invoking or invoked program.

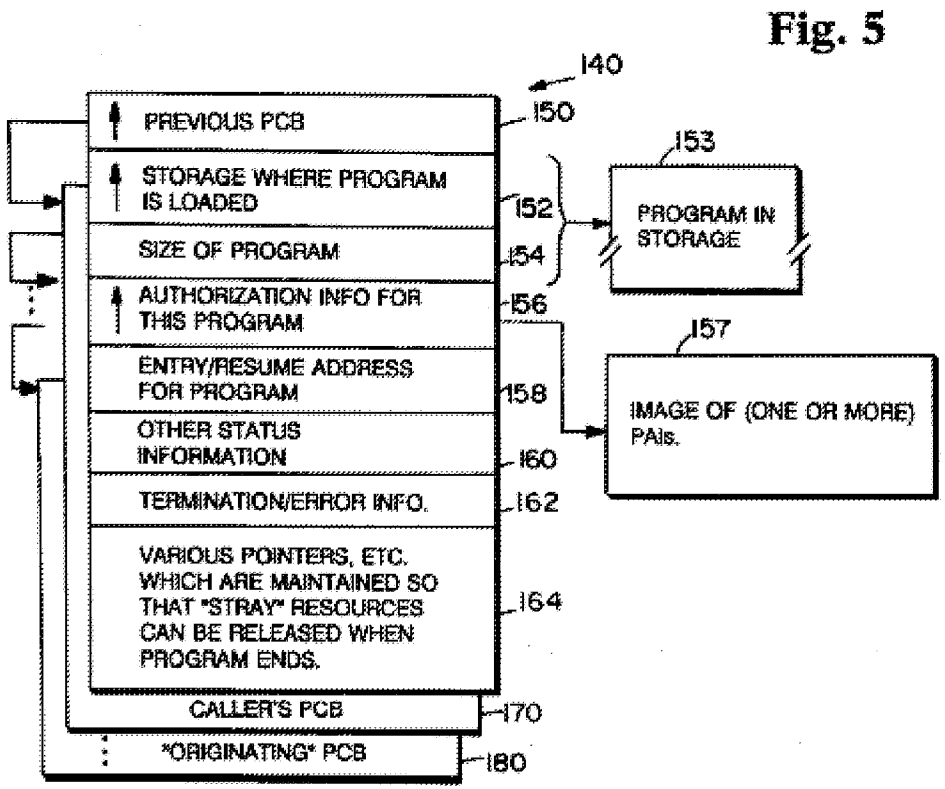
U.S. Patent No. 6,192,476 – Claim 7	*Fischer*
	For example, it is reasonable to restrict a called program to the lesser of its 'normal' PAI authority and that of its calling program--to insure the calling program cannot mischievously misuse the called program's greater authority to circumvent its own limitations. On the other hand, for called programs which carefully verify their own actions, it could be possible to allow the called program greater inherent authority than the program which calls it--this way sensitive resources could be made available to wider use by mediating such use through trusted sub-programs. The possibilities for such combination must be carefully considered, not only by the designers of the underlying control system, but also by those who assign authority to each program." *Fischer* at 17:54-18:8.
B) if said permission required is not encompassed by at least one permission associated with said next routine, then transmitting a message indicating that said permission required is not authorized, and	*Fischer* discloses if said permission required is not encompassed by at least one permission associated with said next routine, then transmitting a message, e.g., "an error message," indicating that said permission required is not authorized.

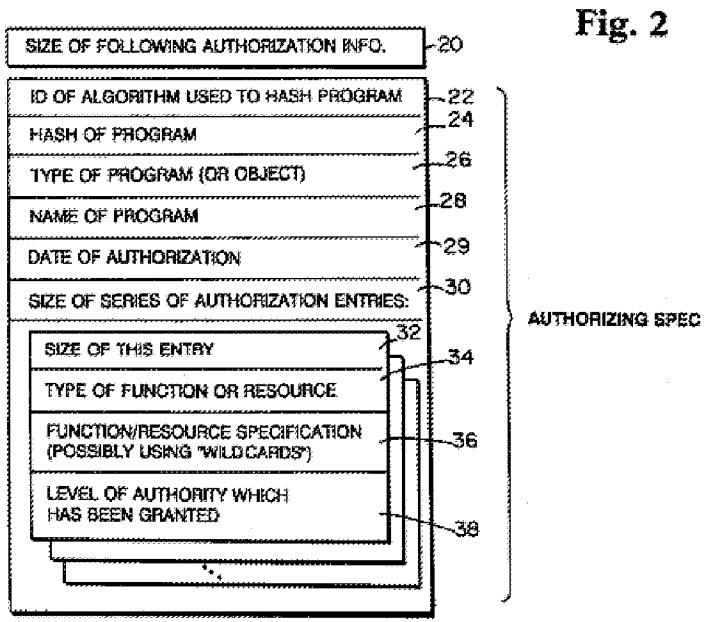
U.S. Patent No. 6,192,476 – Claim 7	*Fischer*
	

U.S. Patent No. 6,192,476 – Claim 7	*Fischer*
	 **Fig. 11** *Fischer* at Fig. 11. "If the check at 342 reveals that the PAI does not allow the attempted function or resource access, then a error message is generated in block 344 to indicate that the program is attempting to exceed its limits, access to the resource or function is denied and an appropriate error code or message is generated." *Fischer* at 19:27-33.
C) repeating steps A and B until: said permission required is not authorized by at least one permission associated with said next routine, there are no more routines to select from said plurality of routines in said calling hierarchy, or determining that said next routine is	*Fischer* discloses repeating steps A and B until: said permission required is not authorized by at least one permission associated with said next routine, there are no more routines to select from said plurality of routines in said calling hierarchy, or determining that said next routine is said first routine. Figure 11 of the *Fischer* patent discloses a recursive application of this principle to determine if one of a plurality of routines has the appropriate access levels. *See also*:

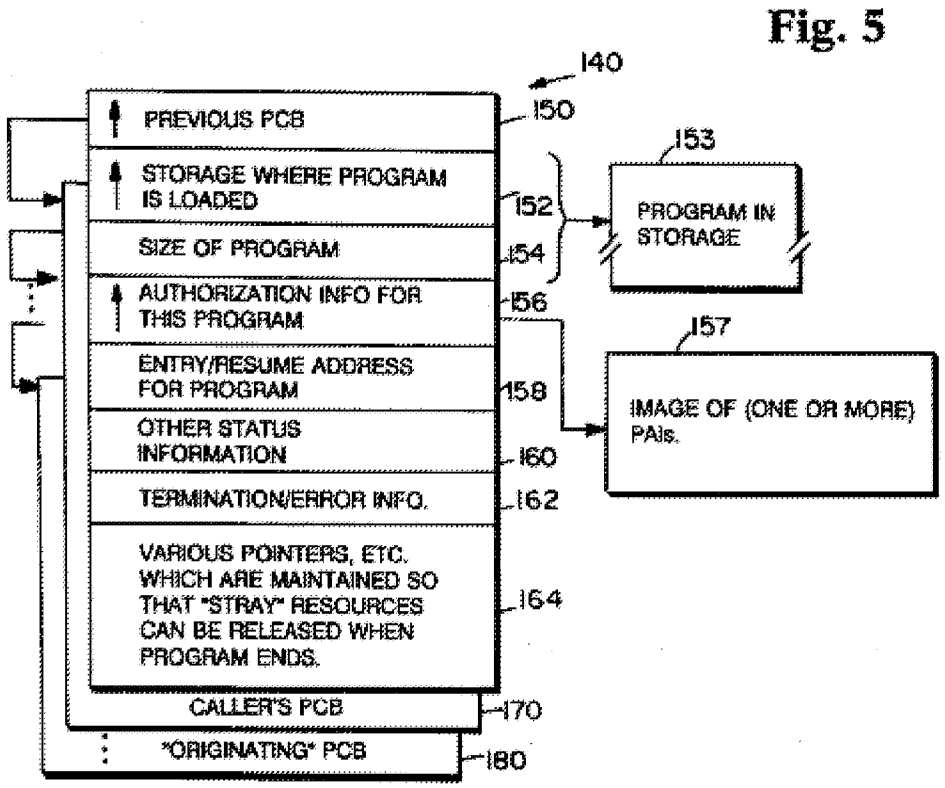
U.S. Patent No. 6,192,476 – Claim 7	*Fischer*
said first routine.	"There are many alternative ways that a program's PAI could be combined with the PAI of the program which invokes it--depending on the strategies which are applicable to the current environment, and the inherent nature of the programs themselves. It may even be likely that even the method of combination is itself one of the PAI authorities, or qualifiers, of either or both the invoking or invoked program. For example, it is reasonable to restrict a called program to the lesser of its 'normal' PAI authority and that of its calling program--to insure the calling program cannot mischievously misuse the called program's greater authority to circumvent its own limitations. On the other hand, for called programs which carefully verify their own actions, it could be possible to allow the called program greater inherent authority than the program which calls it--this way sensitive resources could be made available to wider use by mediating such use through trusted sub-programs. The possibilities for such combination must be carefully considered, not only by the designers of the underlying control system, but also by those who assign authority to each program." *Fischer* at 17:54-18:8.

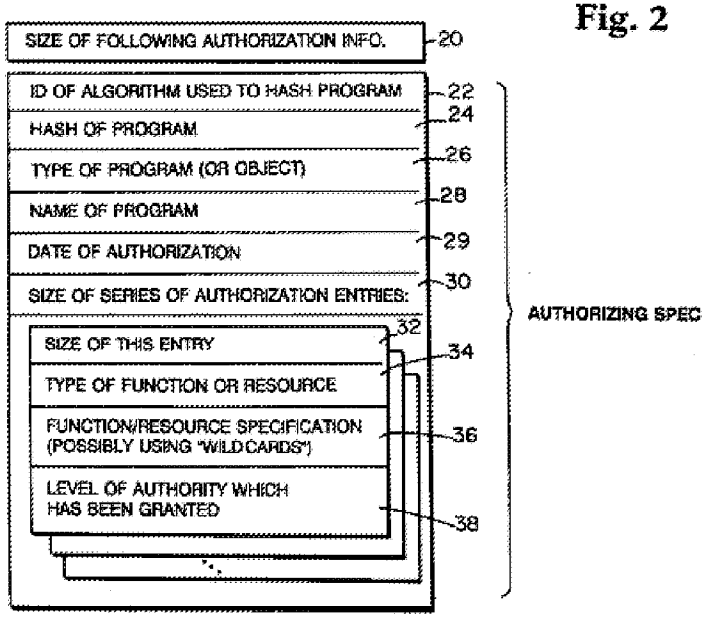
U.S. Patent No. 6,192,476 – Claim 8	*Fischer*
8. The method of claim 7, wherein: the method further includes the step of setting a flag associated with said first routine to indicate that said first routine is privileged; and	*Fischer* discloses the method of claim 7, wherein: the method further includes the step of setting a flag associated with said first routine to indicate that said first routine is privileged. "A field 156 of the program control block identifies the location in storage (157) of one or more PAI's." *Fischer* at 10:47-49. The field 156 of *Fischer* functions as would the "flag" of the instant limitation.

U.S. Patent No. 6,192,476 – Claim 8	*Fischer*
	 *Fischer* at Fig. 5. "A field 156 of the program control block identifies the location in storage (157) of one or more PAI's (which are located in an area of storage which cannot be altered by associated programs). The PAI's pointed to by field 156 are preferably structured in the manner indicated in FIG. 2 described above." *Fischer* at 10:47-52.

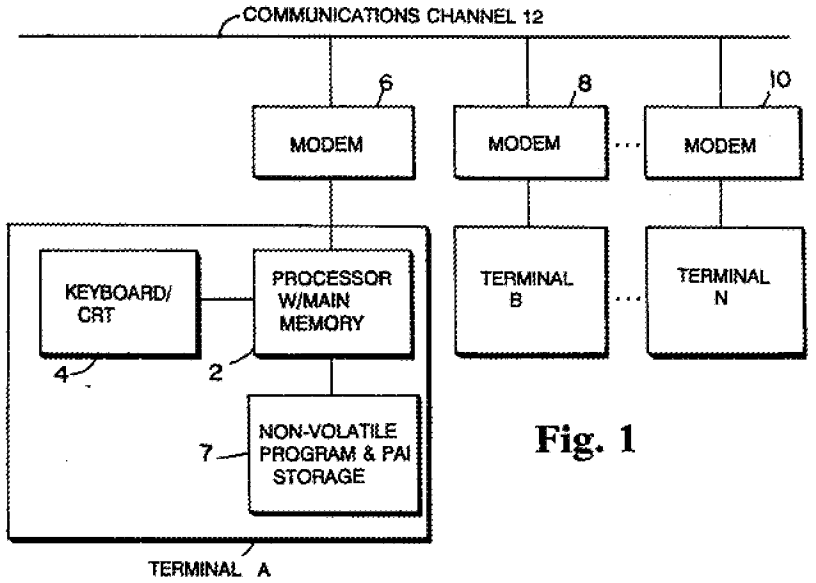
U.S. Patent No. 6,192,476 – Claim 8	*Fischer*
	 *Fischer* at Fig. 2.
the step of determining that said next routine is said first routine includes determining that a flag associated with said next routine indicates said next routine is privileged.	*Fischer* discloses the step of determining that said next routine is said first routine includes determining that a flag associated with said next routine indicates said next routine is privileged. *Fischer* broadly discloses a recursive approach and PAI associations that act as flags. Furthermore, *Fischer* discloses that slight variations on the PAI functionality are inherent in the programs themselves: "There are many alternative ways that a program's PAI could be combined with the PAI of the program which invokes it--depending on the strategies which are applicable to the current environment, and the inherent nature of the programs themselves." *Fischer* at 17:54-58.

U.S. Patent No. 6,192,476 – Claim 9	*Fischer*
9. The method of claim 8, wherein the	*Fischer* discloses the method of claim 8, wherein the step of setting said flag associated with

U.S. Patent No. 6,192,476 – Claim 9	*Fischer*
step of setting said flag associated with said first routine includes setting a flag in a frame in said calling hierarchy associated with said thread.	said first routine includes setting a flag in a frame in said calling hierarchy associated with said thread. "A field 156 of the program control block identifies the location in storage (157) of one or more PAI's." *Fischer* at 10:47-49. The field 156 of *Fischer* functions as would the "flag" of the instant limitation.  *Fischer* at Fig. 5. "A field 156 of the program control block identifies the location in storage (157) of one or more PAI's (which are located in an area of storage which cannot be altered by associated programs). The PAI's pointed to by field 156 are preferably structured in the manner

U.S. Patent No. 6,192,476 – Claim 9	*Fischer*
	indicated in FIG. 2 described above." *Fischer* at 10:47-52. **Fig. 2**  *Fischer* at Fig. 2.

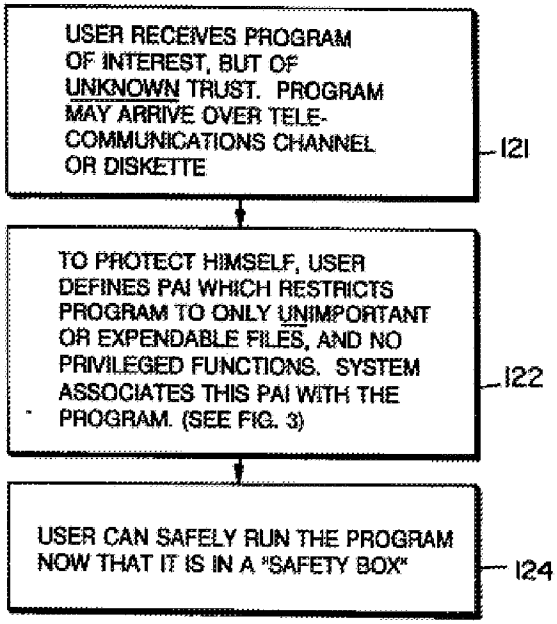
U.S. Patent No. 6,192,476 – Claim 10	*Fischer*
10. A computer-readable medium carrying one or more sequences of one or more instructions, the one or more sequences of the one or more instructions including instructions which, when executed by one or more processors, causes the one or more processors to perform the steps of:	*Fischer* discloses computer-readable medium, e.g, "main memory" or "a disk memory device," carrying one or more sequences of one or more instructions, the one or more sequences of the one or more instructions including instructions which, when executed by one or more processors, e.g. "IBM PC's having a processor," causes the one or more processors to perform steps. "FIG. 1 shows in block diagram form an exemplary communications system which may be used in conjunction with the present invention. . . . Terminals, A, B . . . N may, by way of example only, be IBM PC's having a processor (with main memory) 2 which is coupled to a conventional keyboard/CRT display 4. Additionally, each processor is preferably coupled

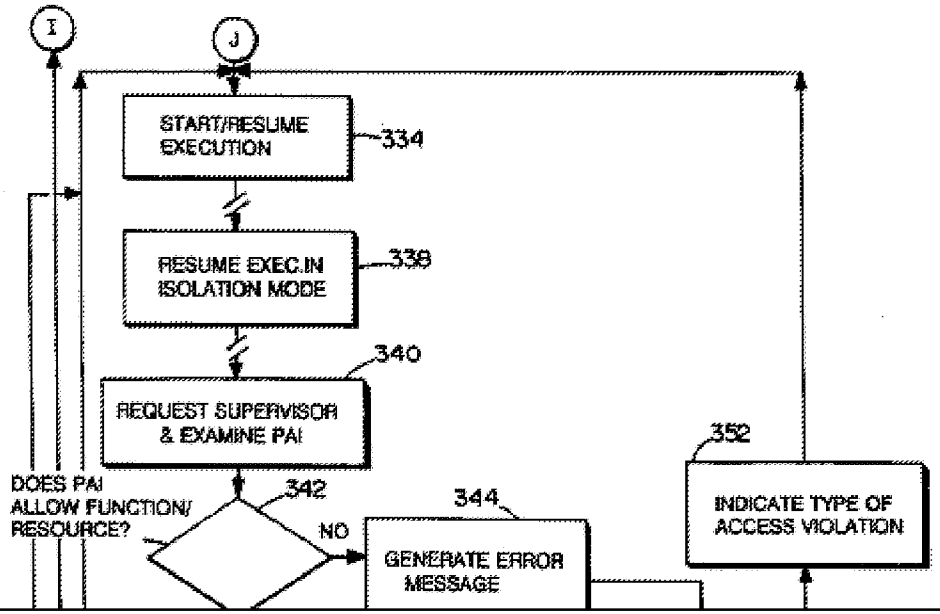
U.S. Patent No. 6,192,476 – Claim 10	*Fischer*
	to a non-volatile program and program authorization information (PAI) storage 7 which may be a disk memory device."  **Fig. 1** *Fischer* at 4:45-58 & Fig. 1. "Turning back to FIG. 1, a program of unknown trust may be injected into the system via communications channel 12 or from a floppy disk loaded into terminal A. The program may be initially stored in, for example, the user's program disk memory 7." *Fischer* at 9:64-68.
detecting when a request for an action is made by a principal; and	*Fischer* discloses detecting when a request for an action is made by a principal. *Fischer* inquires into the permissions of a given principal's PAI upon the principal's request for an action: "When the program is to perform a function or access a resource, the associated PAI is monitored to confirm that the operation is within the defined program limits. If the program attempts to do anything outside the authorized limits, then the program execution is halted." *Id.* at 2:43-48. Thus it inherent that *Fischer* detects a request of an action made by a principal.

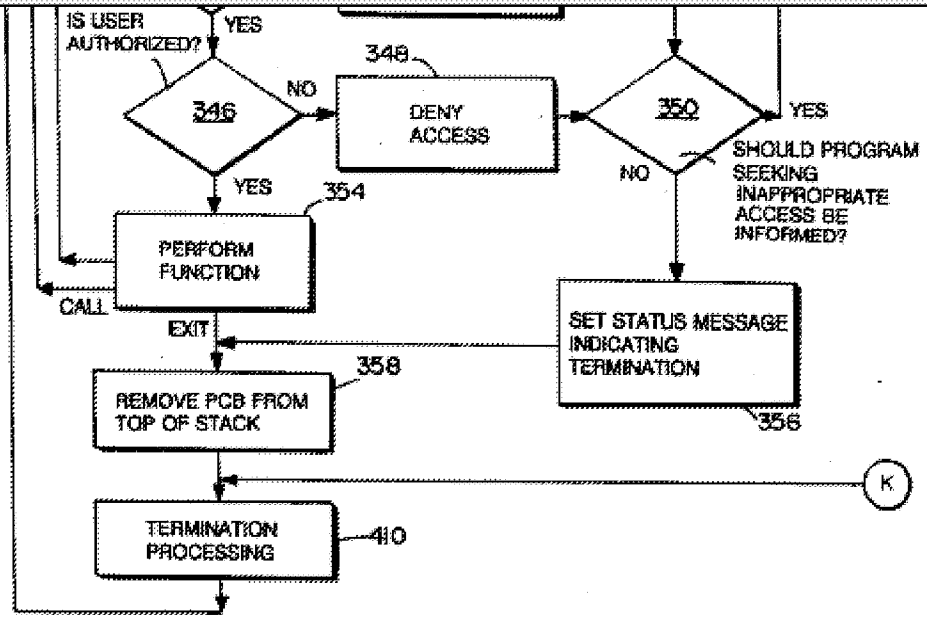
U.S. Patent No. 6,192,476 – Claim 10	*Fischer*
	"Prior art operating systems are typically designed to protect data from computer users. In such systems, users are often assigned various authorities and are thereafter able to execute programs based on their associated authority. If a program is executing which exceeds the user's assigned authority, then such a system will halt execution of the program. Such prior art systems do not adequately protect computer users from computer viruses or the like. There are security systems which protect certain 'system' related files from being modified by a program. However, such systems do not typically protect a computer user from a program executing and modifying the user's own files. The present invention is directed to providing reliable security, even when operating with complex data structures, e.g., objects, containing their own program instructions, which are transmitted among users. The present invention also provides enhanced security when processing more conventional programs, even those of questionable origin, e.g., from a computer bulletin board, without exposing system programs or data to the potentially catastrophic consequences of computer viruses or of incompetent programming. The present method and apparatus utilizes a unique operating system design that includes a system monitor which limits the ability of a program about to be executed to the use of predefined resources (e.g., data files, disk writing capabilities etc.). The system monitor builds a data structure including a set of authorities defining that which a program is permitted to do and/or that which the program is precluded from doing. The set of authorities and/or restrictions assigned to a program to be executed are referred to herein as 'program authorization information' (or 'PAI'). Once defined, the program authorization information is thereafter associated with each program to be executed to thereby delineate the types of resources and functions that the program is allowed to utilize. The PAI associated with a particular program may be assigned by a computer system owner/user or by someone who the computer system owner/user implicitly trusts. The PAI defines the range of operations that a program may execute and/or defines those

U.S. Patent No. 6,192,476 – Claim 10	*Fischer*
	operations that a program cannot perform. The program is permitted to access what has been authorized and nothing else. In this fashion, the program may be regarded as being placed in a program capability limiting 'safety box'. This 'safety box' is thereafter associated with the program such that whenever the system monitor runs the program, the PAI for that program is likewise loaded and monitored. When the program is to perform a function or access a resource, the associated PAI is monitored to confirm that the operation is within the defined program limits. If the program attempts to do anything outside the authorized limits, then the program execution is halted. Thus, the present invention advantageously protects a user from any program to be executed. The present invention is particularly advantageous in light of current data processing practices where programs are obtained from a wide range of diverse, untrustworthy places such as computer bulletin boards or other users of unknown trustworthiness." *Fischer* at 1:60-2:55. "FIG. 4 is a flowchart which illustrates how a user may benefit from the use of program authorization information, particularly when executing a program of unknown trustworthiness. As indicated in block 121, a user may have a desire to execute a program of interest in which the user has no knowledge of the program's creator. Thus, the program has unknown trustworthiness and may, for example, have been accessed via an electronic bulletin board and may have arrived at the user's terminal via a telecommunications channel or diskette. Such a program, which might purport to be only a game, carries with it a significant risk that it may be infected by a virus. As indicated in block 122, the user may be protected by defining program authorization information which restricts the program to only unimportant or expendable files. If desired, the user may restrict such a program from modifying any files whatsoever. For example, the user may permit the program to only display images on the display screen and to perform game playing related functions. Alternatively, if the program is known to have a single work file, the PAI data may only permit use of such a single file. By limiting access only to a single work file, a program of unknown trustworthiness, cannot inject a virus into other user's programs or otherwise initiate system program malfunctions. Thus, in accordance

U.S. Patent No. 6,192,476 – Claim 10	*Fischer*
	with the present invention, a user, via a systems program, determines how much of the user's system will be put at risk by such a program so as to, for example, completely eliminate the ability of the program to use any privileged functions. The user then associates, for example, through an operator prompting, menu-driven system, a PAI with every program to be run on the system (or have such PAI or lack of PAI associated through predetermined default mechanisms). A system utility program is preferably employed to create the program authorization information in a manner which will be described in detail below in conjunction with FIGS. 6-9. After the PAI has been assigned, any time the system runs the associated program, the system software (in a manner to be described below) insures that the program safely runs in a manner consistent with the PAI. Thus, the program has been effectively placed in a 'safety box' (124). Turning back to FIG. 1, a program of unknown trust may be injected into the system via communications channel 12 or from a floppy disk loaded into terminal A. The program may be initially stored in, for example, the user's program disk memory 7. Thereafter, the user on keyboard 4 will, through interaction with the system's program identified above (with respect to block 122 of FIG. 4), associate the program authorization information with a program (in a manner such as shown in FIGS. 3A through 3D) such that the program may safely run on the user's system or perhaps, a PAI arrives with the program, in which case it is likely to be signed. FIG. 5 is an illustration of a program control block (PCB) data structure 140 in accordance with an exemplary embodiment of the present invention. The program control block 140 is the data structure utilized by the system monitor to control the execution of an associated program. The program control block 140 is loaded with program authorization information such that the PAI can be readily referenced as the associated program is executed so as to insure that the program performs functions and accesses resources in conformance with its assigned authorizations. The program control block associated with the program to be executed is

U.S. Patent No. 6,192,476 – Claim 10	*Fischer*
	located in a storage area which cannot be modified by the program." *Fischer* at 9:17-10:23.  **Fig. 4** *Fischer* at Fig. 4. "FIGS. 10 and 11 illustrate the sequence of operations of a supervisor program for controlling the processing of a program being executed in accordance with its program authorization information. The processing of a program 'X' and its program authorization information illustrated in FIG. 10 is initiated while the computer is executing a supervisor routine. As shown in FIG. 10 at 300, a calling program calls program X for execution. Thereafter, a program control block is created for program X. The program control block created will not be added to the top of the execution stack until it is determined that the program is permitted to be invoked and verification is successful completed. Thus, if the program fails a security check, it will not be placed in the program execution chain. In addition to creating a 'tentative' program control block, the called program will be located through an appropriate program directory during the processing in block 302.

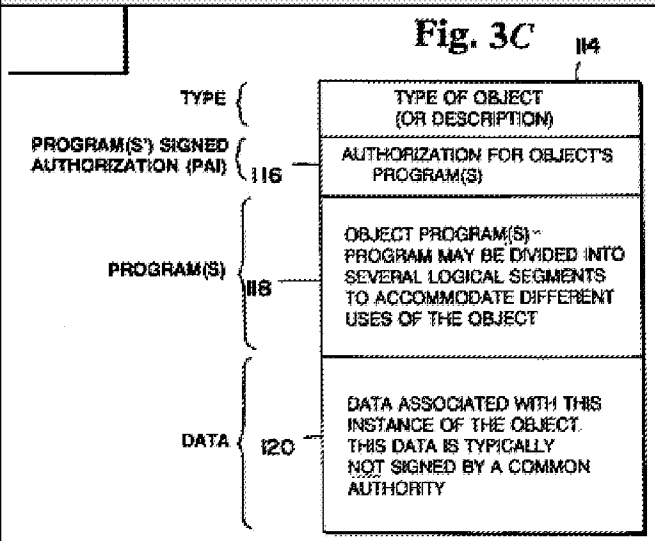
U.S. Patent No. 6,192,476 – Claim 10	*Fischer*
	Thereafter, a check is made at block 304 to determine whether PAI has yet been associated with program X so as to place program X in the so-called 'safety box' described above. This PAI may or may not be signed depending upon its particular application as described above." *Fischer* at 15:56-16:11. 

U.S. Patent No. 6,192,476 – Claim 10	*Fischer*
	 **Fig. 11** *Fischer* at Fig. 11.
in response to detecting the request, determining whether said action is authorized based on permissions associated with a plurality of routines in a calling hierarchy associated with said principal, wherein said permissions are associated with said plurality of routines based on a first association between protection domains and permissions.	*Fischer* discloses in response to detecting the request, determining whether said action is authorized based on permissions associated with a plurality of routines in a calling hierarchy associated with said principal, wherein said permissions are associated with said plurality of routines based on a first association between protection domains and permissions. *Fischer* discloses "an originating program" that "calls a program" having a program control block, or PCB. *Fischer* at 10:24-26. "Each new PCB will include a field . . . that points to the 'previous' of calling program control block." *See id.* at 10:27-29. Then, "[w]hen a called program finishes executing, the system removes its associated PCB from the top of the executed stack, removes the associated program from storage, removes the associated authorizing information and accesses the program control block immediately below it in the stack." *See id.* at 10:31-36. *Fischer* also incorporates by reference two patents which explicitly disclose data hierarchies: *see id.* at 6:37 (incorporating by reference U.S. Patent

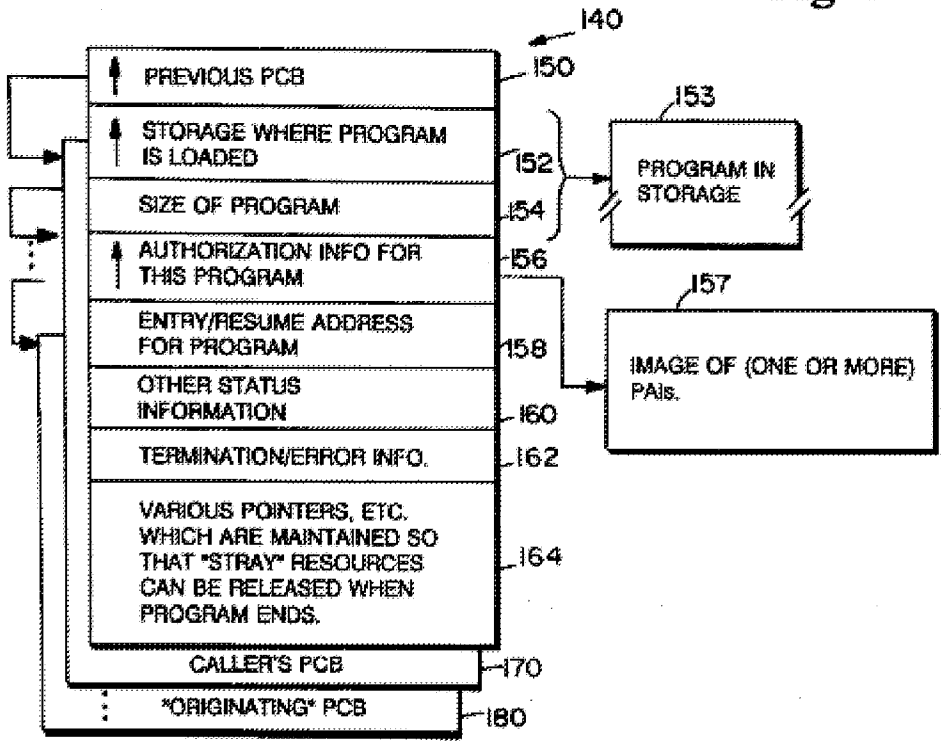
U.S. Patent No. 6,192,476 – Claim 10	*Fischer*
	Nos. 4,868,877 and 5,005,200 which disclose an "enhanced digital signature certification" which employs "[a] hierarchy of nested certifications and signatures." *See* U.S. Patent No. 5,005,200, Abstract. "The present method an apparatus utilizes a unique operation system design that includes a system monitor which limits the ability of a program about to be executed to the use of predefined resources (e.g., data files, disk writing capabilities etc.). The system monitor builds a data structure including a set of authorities defining that which a program is permitted to do and/or that which the program is precluded from doing. The set of authorities and/or restrictions assigned to a program to be executed are referred to herein as 'program authorization information' (or 'PAI'). Once defined, the program authorization information is thereafter associated with each program to be executed to thereby delineate the types of resources and functions that the program is allowed to utilize. The PAI associated with a particular program may be assigned by a computer system owner/user or by someone who the computer system owner/user implicitly trusts. The PAI defines the range of operations that a program may execute and/or defines those operations that a program cannot perform. The program is permitted to access what has been authorized and nothing else. In this fashion, the program may be regarded as being placed in a program capability limiting 'safety box.' This 'safety box' is thereafter associated with the program such that whenever the system monitor runs the program, the PAI for that program is likewise loaded and monitored. When the program is to perform a function or access a resource, the associated PAI is monitored to confirm that the operation is within the defined program limits. If the program attempts to do anything outside the authorized limits, then the program execution is halted." *Fischer* at 2:16-48. "Thereafter, the program X's program authorizing information is combined, as appropriate, with the PAI associated with the PCB of the calling program, if any. This combined PAI, which may include multiple PAI's, is then stored in an area of storage which cannot generally be modified by the program and the address of the PAI is stored in the process control block (PCB) as indicated in field 156 of FIG. 5. Thus, if program X is called by a

U.S. Patent No. 6,192,476 – Claim 10	*Fischer*
	calling program, it is subject to all its own constraints as well as being combined in some way with the constraints of the calling program, which aggregate constraints are embodied into program X's PAI. In this fashion, a calling program may not be permitted to exceed its assigned bounds by merely calling another program." *Fischer* at 19:40-54. "Even programs with no known trustworthiness can be used after program authorization information associates a wide range of restrictions to thereby allow potentially beneficial programs to be safely used--even if they do not have an official certification of trust. The present invention also allows an unlimited number of different resources and functions to be controlled. For example, some useful resources/functions which may be controlled include: the ability to limit a program to certain files or data sets; the ability to transmit data via electronic mail to someone outside the user's domain; the ability of a program to create or solicit digital signatures; the ability to limit access to a program of certain security classes, etc." *Fischer* at 3:48-61. "The present invention is directed to providing reliable security, even when operating with complex data structures, e.g., objects, containing their own program instructions, which are transmitted among users." *Fischer* at 2:6-9. "Through the use of the present invention, general object oriented data may be transferred from user to user without exposing users to the potential dangers of viruses or mischievous users." *Fischer* at 4:10-13. "In one contemplated embodiment of the present invention, programs may be part of data objects, which are written in a high-level control language and are executed by a standardized interpreter program which executes this high-level language. In this case, part of the interpreter's task is to verify that the functions encountered in the high level logic are, in fact, permissible. If such tasks are not permissible, the interpreter then suppresses the execution of the program not authorized to perform such tasks." *Fischer* at 3:11-20. "In accordance with the present invention, a PAI is associated with programs to be executed.

U.S. Patent No. 6,192,476 – Claim 10	*Fischer*
	FIGS. 3A through 3D depict four exemplary approaches for associating program authorization information with a program. . . . . . . FIG. 3C shows an important application in which a PAI data structure is associated with a program according to an embodiment of the present invention. FIG. 3C shows an illustrative data structure for a secure exchangeable 'object'. The data structure may be signed by a trusted authority. The signing of such a data structure allows the object to be securely transmitted from user to user. Although the data structure shown in FIG. 3 is set forth in a general format, it may be structured as set forth in the inventor's copending application filed on Apr. 6, 1992 and entitled 'Method and Apparatus for Creating, Supporting and Processing a Travelling Program' (U.S. Ser. No. 07/863,552.), which application is hereby expressly incorporated herein by reference. . . . The program authorization information is embedded in a segment 116 which specifies the authorization for the object's program or programs in a manner to be described more fully hereinafter." *Fischer* at 7:14-18, 7:49-8:2. "Thereafter, the PAI is stored using, for example, one of the approaches set forth in FIGS. 3A through 3D so that it is associated with its program 272 . . . ." *Fischer* at 15:24-26.

U.S. Patent No. 6,192,476 – Claim 10	*Fischer*
	 *Fischer* at Fig. 3C. "FIGS. 10 and 11 illustrate the sequence of operations of a supervisor program for controlling the processing of a program being executed in accordance with its program authorization information." *Fischer* at 15:56-59. "Depending on the processing in block 316, a decision is made in block 322 whether the signatures are valid, authorized and trusted. If the signatures are not determined to be valid, then the routing branches to block 324 where the execution in program X is suppressed." *Fischer* at 16:66-17:3. "If the processing in blocks 322 and 316 reveal that the signatures are valid, then the processing in block 326 is performed." *Fischer* at 17:31-33. "FIG. 5 is an illustration of a program control block (PCB) data structure 140 in accordance with an exemplary embodiment of the present invention. The program control block 140 is the data structure utilized by the system monitor to control the execution of an associated

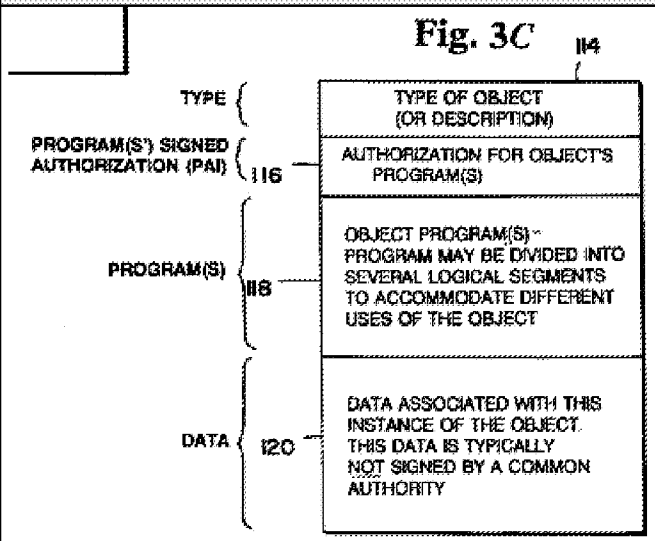
U.S. Patent No. 6,192,476 – Claim 10	*Fischer*
	program. The program control block 140 is loaded with program authorization information such that the PAI can be readily referenced as the associated program is executed so as to insure that the program performs functions and accesses resources in conformance with its assigned authorizations. The program control block associated with the program to be executed is located in a storage area which cannot be modified by the program. As shown in FIG. 5, an originating program (whose PCB is identified at 180) calls a program (having a PCB 170) which will, in turn, will call the program 140 is shown in detail in FIG. 5. Each new PCB will include a field such as 150 that points to the 'previous' or calling program control block. A field may also be utilized to identify the 'next' program control block file. When a called program finishes executing, the system removes its associated PCB from the top of the executed stack, removes the associated program from storage, removes the associated authorizing information and accesses the program control block immediately below it in the stack. When another program is called, the reverse process occurs such that a new PCB is created which is placed on top of the stack, which again points to the previous PCB as indicated in field 150." *Fischer* at 10:8-39.

U.S. Patent No. 6,192,476 – Claim 10	*Fischer*
	Fig. 5  *Fischer* at Fig. 5. "Thereafter, the program X's program authorizing information is combined, as appropriate, with the PAI associated with the PCB of the calling program, if any. This combined PAI, which may include multiple PAI's, is then stored in an area of storage which cannot generally be modified by the program and the address of the PAI is stored in the process control block (PCB) as indicated in field 156 of FIG. 5. Thus, if program X is called by a calling program, it is subject to all its own constraints as well as being combined in some way with the constraints of the calling program, which aggregate constraints are embodied into program X's PAI. In this fashion, a calling program may not be permitted to exceed its

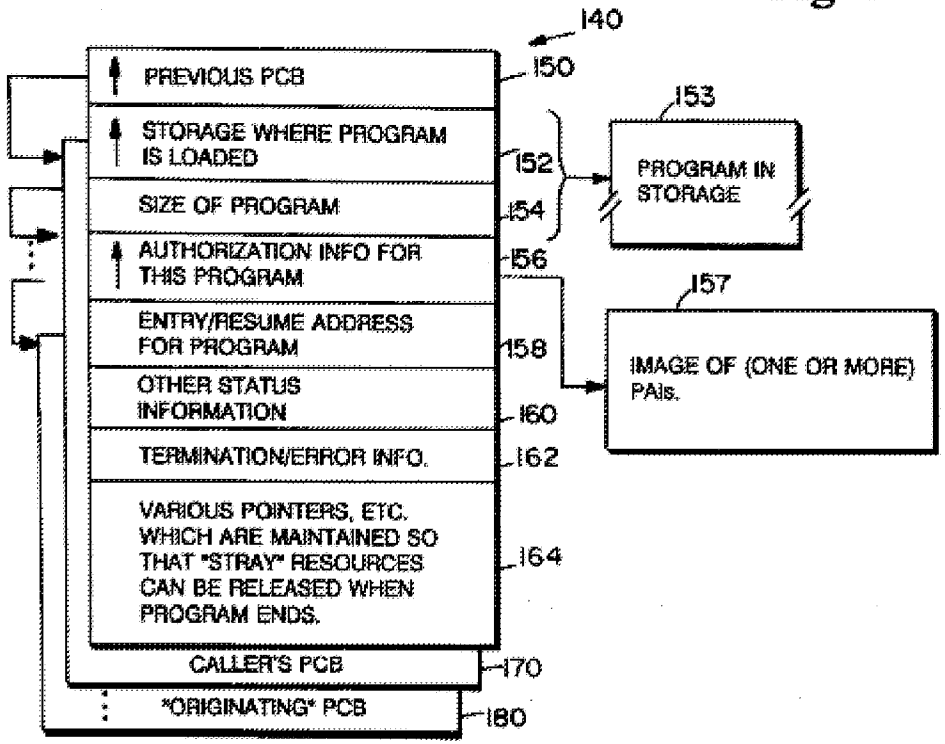
U.S. Patent No. 6,192,476 – Claim 10	*Fischer*
	assigned bounds by merely calling another program. There are many alternative ways that a program's PAI could be combined with the PAI of the program which invokes it--depending on the strategies which are applicable to the current environment, and the inherent nature of the programs themselves. It may even be likely that even the method of combination is itself one of the PAI authorities, or qualifiers, of either or both the invoking or invoked program. For example, it is reasonable to restrict a called program to the lesser of its 'normal' PAI authority and that of its calling program--to insure the calling program cannot mischievously misuse the called program's greater authority to circumvent its own limitations. On the other hand, for called programs which carefully verify their own actions, it could be possible to allow the called program greater inherent authority than the program which calls it--this way sensitive resources could be made available to wider use by mediating such use through trusted sub-programs. The possibilities for such combination must be carefully considered, not only by the designers of the underlying control system, but also by those who assign authority to each program. Thereafter, the program is loaded and the hash of the program is computed based on the algorithm specified in the program's PAI." *Fischer* at 17:40-18:10.

U.S. Patent No. 6,192,476 – Claim 11	*Fischer*
11. The computer-readable medium of claim 10, wherein: the step of detecting when a request for an action is made includes detecting when a request for an action is made by a thread; and	*Fischer* discloses the computer-readable medium of claim 10, wherein: the step of detecting when a request for an action is made includes detecting when a request for an action is made by a thread. *Fischer* discloses that the invention can be directed to complex data structures, which would include a "thread," as illustrated below. "The present invention is directed to providing reliable security, even when operating with complex data structures, e.g., objects, containing their own program instructions, which are transmitted among users. The present invention also provides enhanced security when processing more conventional programs, even those of questionable origin, e.g., from a computer bulletin board, without exposing system programs or data to the potentially catastrophic consequences of computer viruses or of incompetent programming. The present method and apparatus utilizes a unique operating system design that includes a

U.S. Patent No. 6,192,476 – Claim 11	*Fischer*
	system monitor which limits the ability of a program about to be executed to the use of predefined resources (e.g., data files, disk writing capabilities etc.). The system monitor builds a data structure including a set of authorities defining that which a program is permitted to do and/or that which the program is precluded from doing." *Fischer* at 2:6-23. "In accordance with the present invention, a PAI is associated with programs to be executed. FIGS. 3A through 3D depict four exemplary approaches for associating program authorization information with a program. . . . . . . FIG. 3C shows an important application in which a PAI data structure is associated with a program according to an embodiment of the present invention. FIG. 3C shows an illustrative data structure for a secure exchangeable 'object'. The data structure may be signed by a trusted authority. The signing of such a data structure allows the object to be securely transmitted from user to user. Although the data structure shown in FIG. 3 is set forth in a general format, it may be structured as set forth in the inventor's copending application filed on Apr. 6, 1992 and entitled 'Method and Apparatus for Creating, Supporting and Processing a Travelling Program' (U.S. Ser. No. 07/863,552.), which application is hereby expressly incorporated herein by reference. . . . The program authorization information is embedded in a segment 116 which specifies the authorization for the object's program or programs in a manner to be described more fully hereinafter." *Fischer* at 7:14-18, 7:49-8:2. "Thereafter, the PAI is stored using, for example, one of the approaches set forth in FIGS. 3A through 3D so that it is associated with its program 272 . . . ." *Fischer* at 15:24-26.

U.S. Patent No. 6,192,476 – Claim 11	*Fischer*
	 *Fischer* at Fig. 3C.
the step of determining whether said action is authorized includes determining whether said action is authorized based on an association between permissions and a plurality of routines in a calling hierarchy associated with said thread.	*Fischer* discloses the step of determining whether said action is authorized includes determining whether said action is authorized based on an association between permissions and a plurality of routines in a calling hierarchy associated with said thread. *Fischer* discloses "an originating program" that "calls a program" having a program control block, or PCB. *Fischer* at 10:24-26. "Each new PCB will include a field . . . that points to the 'previous' of calling program control block." *See id.* at 10:27-29. Then, "[w]hen a called program finishes executing, the system removes its associated PCB from the top of the executed stack, removes the associated program from storage, removes the associated authorizing information and accesses the program control block immediately below it in the stack." *See id.* at 10:31-36. *Fischer* also incorporates by reference two patents which explicitly disclose data hierarchies: *see id.* at 6:37 (incorporating by reference U.S. Patent Nos. 4,868,877 and 5,005,200 which disclose an "enhanced digital signature certification" which employs "[a] hierarchy of nested certifications and signatures." *See* U.S. Patent No. 5,005,200, Abstract. "FIG. 5 is an illustration of a program control block (PCB) data structure 140 in accordance

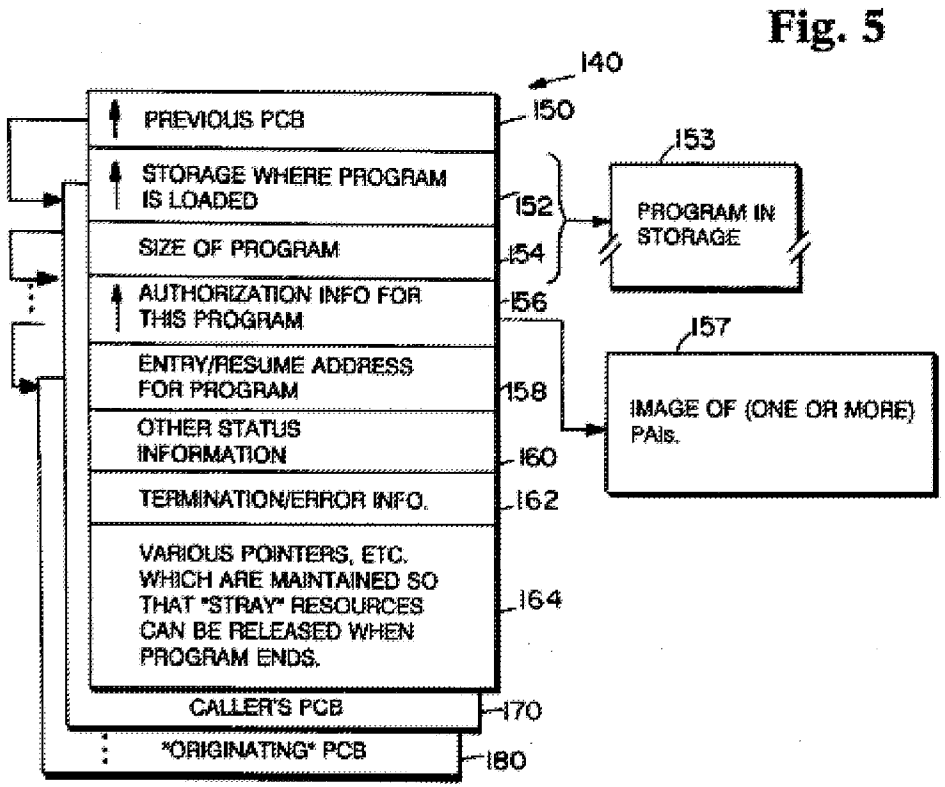
U.S. Patent No. 6,192,476 – Claim 11	*Fischer*
	with an exemplary embodiment of the present invention. The program control block 140 is the data structure utilized by the system monitor to control the execution of an associated program. The program control block 140 is loaded with program authorization information such that the PAI can be readily referenced as the associated program is executed so as to insure that the program performs functions and accesses resources in conformance with its assigned authorizations. The program control block associated with the program to be executed is located in a storage area which cannot be modified by the program. As shown in FIG. 5, an originating program (whose PCB is identified at 180) calls a program (having a PCB 170) which will, in turn, will call the program 140 is shown in detail in FIG. 5. Each new PCB will include a field such as 150 that points to the 'previous' or calling program control block. A field may also be utilized to identify the 'next' program control block file. When a called program finishes executing, the system removes its associated PCB from the top of the executed stack, removes the associated program from storage, removes the associated authorizing information and accesses the program control block immediately below it in the stack. When another program is called, the reverse process occurs such that a new PCB is created which is placed on top of the stack, which again points to the previous PCB as indicated in field 150." *Fischer* at 10:8-39.

U.S. Patent No. 6,192,476 – Claim 11	*Fischer*
	Fig. 5  *Fischer* at Fig. 5. "Thereafter, the program X's program authorizing information is combined, as appropriate, with the PAI associated with the PCB of the calling program, if any. This combined PAI, which may include multiple PAI's, is then stored in an area of storage which cannot generally be modified by the program and the address of the PAI is stored in the process control block (PCB) as indicated in field 156 of FIG. 5. Thus, if program X is called by a calling program, it is subject to all its own constraints as well as being combined in some way with the constraints of the calling program, which aggregate constraints are embodied into program X's PAI. In this fashion, a calling program may not be permitted to exceed its

U.S. Patent No. 6,192,476 – Claim 11	*Fischer*
	assigned bounds by merely calling another program. There are many alternative ways that a program's PAI could be combined with the PAI of the program which invokes it--depending on the strategies which are applicable to the current environment, and the inherent nature of the programs themselves. It may even be likely that even the method of combination is itself one of the PAI authorities, or qualifiers, of either or both the invoking or invoked program. For example, it is reasonable to restrict a called program to the lesser of its 'normal' PAI authority and that of its calling program--to insure the calling program cannot mischievously misuse the called program's greater authority to circumvent its own limitations. On the other hand, for called programs which carefully verify their own actions, it could be possible to allow the called program greater inherent authority than the program which calls it--this way sensitive resources could be made available to wider use by mediating such use through trusted sub-programs. The possibilities for such combination must be carefully considered, not only by the designers of the underlying control system, but also by those who assign authority to each program. Thereafter, the program is loaded and the hash of the program is computed based on the algorithm specified in the program's PAI." *Fischer* at 17:40-18:10.

U.S. Patent No. 6,192,476 – Claim 12	*Fischer*
12. The computer readable medium of claim 10, wherein: the calling hierarchy includes a first routine; and	*Fischer* discloses the computer readable medium, e.g., a "main memory," of claim 10, wherein: the calling hierarchy includes a first routine. For example, *Fischer* discloses calling a plurality of routines: "As shown in FIG. 5, an originating program (whose PCB is identified at 180) calls a program (having a PCB 170) which will, in turn, will call the program 140 is shown in detail in FIG. 5." *Fischer* at 10:24-27. "FIGS. 10 and 11 illustrate the sequence of operations of a supervisor program for controlling the processing of a program being executed in accordance with its program authorization information. The processing of a program 'X' and its program authorization information illustrated in FIG. 10 is initiated while the computer is executing a supervisor routine." *Fischer* at 15:56-62.

U.S. Patent No. 6,192,476 – Claim 12	*Fischer*
	"FIG. 5 is an illustration of a program control block (PCB) data structure 140 in accordance with an exemplary embodiment of the present invention. The program control block 140 is the data structure utilized by the system monitor to control the execution of an associated program. The program control block 140 is loaded with program authorization information such that the PAI can be readily referenced as the associated program is executed so as to insure that the program performs functions and accesses resources in conformance with its assigned authorizations. The program control block associated with the program to be executed is located in a storage area which cannot be modified by the program. As shown in FIG. 5, an originating program (whose PCB is identified at 180) calls a program (having a PCB 170) which will, in turn, will call the program 140 is shown in detail in FIG. 5. Each new PCB will include a field such as 150 that points to the 'previous' or calling program control block. A field may also be utilized to identify the 'next' program control block file. When a called program finishes executing, the system removes its associated PCB from the top of the executed stack, removes the associated program from storage, removes the associated authorizing information and accesses the program control block immediately below it in the stack. When another program is called, the reverse process occurs such that a new PCB is created which is placed on top of the stack, which again points to the previous PCB as indicated in field 150." *Fischer* at 10:8-39.

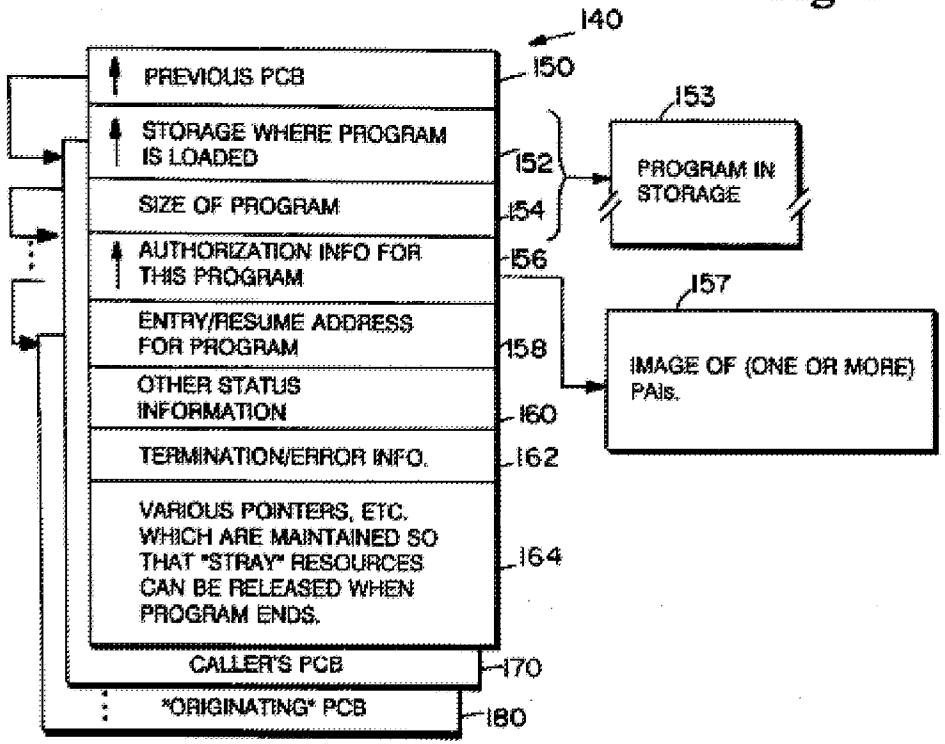
U.S. Patent No. 6,192,476 – Claim 12	*Fischer*
	 *Fischer* at Fig. 5. "FIG. 3C shows an important application in which a PAI data structure is associated with a program according to an embodiment of the present invention. FIG. 3C shows an illustrative data structure for a secure exchangeable 'object'. The data structure may be signed by a trusted authority. The signing of such a data structure allows the object to be securely transmitted from user to user. Although the data structure shown in FIG. 3 is set forth in a general format, it may be structured as set forth in the inventor's copending application filed on Apr. 6, 1992 and entitled 'Method and Apparatus for Creating, Supporting and Processing a Travelling Program' (U.S. Ser. No. 07/863,552.), which application is hereby

U.S. Patent No. 6,192,476 – Claim 12	*Fischer*
	expressly incorporated herein by reference. The data structure includes a header segment 114 which, by way of example only, may define the type of object that follows, e.g., a purchase order related object or any other type of electronic digital object. The program authorization information is embedded in a segment 116 which specifies the authorization for the object's program or programs in a manner to be described more fully hereinafter. The data structure includes an object program(s) segment 118, which for example, may control the manner in which an associated purchase order is displayed so as to leave blanks for variable fields which are interactively completed by the program user. The object program might store such data and send a copy of itself together with accompanying data in a manner which is described in detail in the applicant's above-identified copending application. As indicated in FIG. 3C, the program may be divided into several logical segments to accommodate different uses of the object. For example, the program may present a different display to the creator of a digital purchase order, than it displays to subsequent recipients. When the program is received by a recipient designated by the program, the recipient invokes a copy of the transmitted program to, for example, control the display of the purchase order tailored to the needs of the recipient." *Fischer* at 7:49-8:20.
the step of determining whether said action is authorized further includes determining whether a permission required to perform said action is encompassed by at least one permission associated with said first routine.	*Fischer* discloses the step of determining whether said action is authorized further includes determining whether a permission, e.g., "a PAI data structure," required to perform said action is encompassed by at least one permission associated with said first routine. "FIG. 3C shows an important application in which a PAI data structure is associated with a program according to an embodiment of the present invention. FIG. 3C shows an illustrative data structure for a secure exchangeable 'object'. The data structure may be signed by a trusted authority. The signing of such a data structure allows the object to be securely transmitted from user to user. Although the data structure shown in FIG. 3 is set forth in a general format, it may be structured as set forth in the inventor's copending application filed on Apr. 6, 1992 and entitled 'Method and Apparatus for Creating, Supporting and Processing a Travelling Program' (U.S. Ser. No. 07/863,552.), which application is hereby

U.S. Patent No. 6,192,476 – Claim 12	*Fischer*
	expressly incorporated herein by reference. The data structure includes a header segment 114 which, by way of example only, may define the type of object that follows, e.g., a purchase order related object or any other type of electronic digital object. The program authorization information is embedded in a segment 116 which specifies the authorization for the object's program or programs in a manner to be described more fully hereinafter." *Fischer* at 7:49-8:2. "FIGS. 10 and 11 illustrate the sequence of operations of a supervisor program for controlling the processing of a program being executed in accordance with its program authorization information. The processing of a program 'X' and its program authorization information illustrated in FIG. 10 is initiated while the computer is executing a supervisor routine. As shown in FIG. 10 at 300, a calling program calls program X for execution. Thereafter, a program control block is created for program X. The program control block created will not be added to the top of the execution stack until it is determined that the program is permitted to be invoked and verification is successful completed. Thus, if the program fails a security check, it will not be placed in the program execution chain. In addition to creating a 'tentative' program control block, the called program will be located through an appropriate program directory during the processing in block 302. Thereafter, a check is made at block 304 to determine whether PAI has yet been associated with program X so as to place program X in the so-called 'safety box' described above. This PAI may or may not be signed depending upon its particular application as described above." *Fischer* at 15:56-16:11.

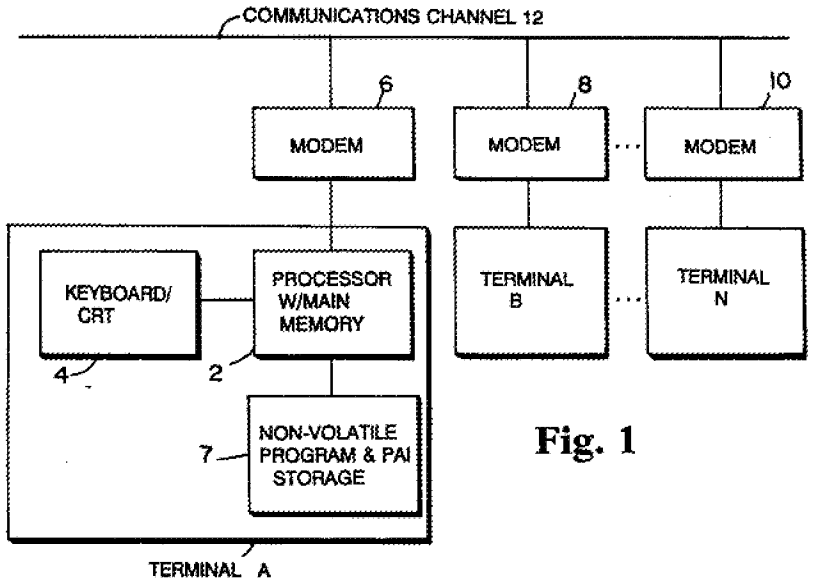
U.S. Patent No. 6,192,476 – Claim 13	*Fischer*
13. The computer readable medium of claim 10, wherein the step of determining whether said action is authorized further includes determining whether a permission required to perform said action is encompassed by	*Fischer* discloses the computer readable medium of claim 10, wherein the step of determining whether said action is authorized further includes determining whether a permission required to perform said action is encompassed by at least one permission associated with each routine in said calling hierarchy. *Fischer* discloses "an originating program" that "calls a program" having a program control block, or PCB. *Fischer* at 10:24-26. "Each new PCB will include a field . . . that points to the 'previous' of calling

U.S. Patent No. 6,192,476 – Claim 13	*Fischer*
at least one permission associated with each routine in said calling hierarchy.	program control block." *See id.* at 10:27-29. Then, "[w]hen a called program finishes executing, the system removes its associated PCB from the top of the executed stack, removes the associated program from storage, removes the associated authorizing information and accesses the program control block immediately below it in the stack." *See id.* at 10:31-36. *Fischer* also incorporates by reference two patents which explicitly disclose data hierarchies: *see id.* at 6:37 (incorporating by reference U.S. Patent Nos. 4,868,877 and 5,005,200 which disclose an "enhanced digital signature certification" which employs "[a] hierarchy of nested certifications and signatures." *See* U.S. Patent No. 5,005,200, Abstract. "As shown in FIG. 5, an originating program (whose PCB is identified at 180) calls a program (having a PCB 170) which will, in turn, will call the program 140 is shown in detail in FIG. 5. Each new PCB will include a field such as 150 that points to the 'previous' or calling program control block. A field may also be utilized to identify the 'next' program control block file. When a called program finishes executing, the system removes its associated PCB from the top of the executed stack, removes the associated program from storage, removes the associated authorizing information and accesses the program control block immediately below it in the stack. When another program is called, the reverse process occurs such that a new PCB is created which is placed on top of the stack, which again points to the previous PCB as indicated in field 150." *Fischer* at 10:23-39.

U.S. Patent No. 6,192,476 – Claim 13	*Fischer*
	Fig. 5  *Fischer* at Fig. 5. "Thereafter, the program X's program authorizing information is combined, as appropriate, with the PAI associated with the PCB of the calling program, if any. This combined PAI, which may include multiple PAI's, is then stored in an area of storage which cannot generally be modified by the program and the address of the PAI is stored in the process control block (PCB) as indicated in field 156 of FIG. 5. Thus, if program X is called by a calling program, it is subject to all its own constraints as well as being combined in some way with the constraints of the calling program, which aggregate constraints are embodied into program X's PAI. In this fashion, a calling program may not be permitted to exceed its

U.S. Patent No. 6,192,476 – Claim 13	*Fischer*
	assigned bounds by merely calling another program. There are many alternative ways that a program's PAI could be combined with the PAI of the program which invokes it--depending on the strategies which are applicable to the current environment, and the inherent nature of the programs themselves. It may even be likely that even the method of combination is itself one of the PAI authorities, or qualifiers, of either or both the invoking or invoked program. For example, it is reasonable to restrict a called program to the lesser of its 'normal' PAI authority and that of its calling program--to insure the calling program cannot mischievously misuse the called program's greater authority to circumvent its own limitations. On the other hand, for called programs which carefully verify their own actions, it could be possible to allow the called program greater inherent authority than the program which calls it--this way sensitive resources could be made available to wider use by mediating such use through trusted sub-programs. The possibilities for such combination must be carefully considered, not only by the designers of the underlying control system, but also by those who assign authority to each program. Thereafter, the program is loaded and the hash of the program is computed based on the algorithm specified in the program's PAI." *Fischer* at 17:40-18:10.

U.S. Patent No. 6,192,476 – Claim 14	*Fischer*
14. A computer-readable medium bearing instructions for providing security, the instructions including instructions for performing the steps of:	*Fischer* discloses a computer-readable medium, e.g., "a disk memory device," bearing instructions for providing security, the instructions including instructions for performing steps. "More particularly, the invention relates to a method and apparatus for providing enhanced computer system security while processing computer programs, particularly those of unknown origin, which are transmitted among users." *Fischer* at 1:20-25. "Thus, the present invention advantageously protects a user from any program to be executed. The present invention is particularly advantageous in light of current data processing practices where programs are obtained from a wide range of diverse, untrustworthy places such as computer bulletin boards or other users of unknown trustworthiness." *Fischer* at 2:49-55.

U.S. Patent No. 6,192,476 – Claim 14	*Fischer*
	"FIG. 1 shows in block diagram form an exemplary communications system which may be used in conjunction with the present invention. . . . Terminals, A, B . . . N may, by way of example only, be IBM PC's having a processor (with main memory) 2 which is coupled to a conventional keyboard/CRT display 4. Additionally, each processor is preferably coupled to a non-volatile program and program authorization information (PAI) storage 7 which may be a disk memory device."  **Fig. 1** *Fischer* at 4:45-58 & Fig. 1. "Turning back to FIG. 1, a program of unknown trust may be injected into the system via communications channel 12 or from a floppy disk loaded into terminal A. The program may be initially stored in, for example, the user's program disk memory 7." *Fischer* at 9:64-68.
detecting when a request for an action is made by a principal;	*Fischer* discloses detecting when a request for an action is made by a principal. *Fischer* inquires into the permissions of a given principal's PAI upon the principal's request for an

U.S. Patent No. 6,192,476 – Claim 14	*Fischer*
	action: "When the program is to perform a function or access a resource, the associated PAI is monitored to confirm that the operation is within the defined program limits. If the program attempts to do anything outside the authorized limits, then the program execution is halted." *Id.* at 2:43-48. Thus it inherent that *Fischer* detects a request of an action made by a principal. "Prior art operating systems are typically designed to protect data from computer users. In such systems, users are often assigned various authorities and are thereafter able to execute programs based on their associated authority. If a program is executing which exceeds the user's assigned authority, then such a system will halt execution of the program. Such prior art systems do not adequately protect computer users from computer viruses or the like. There are security systems which protect certain 'system' related files from being modified by a program. However, such systems do not typically protect a computer user from a program executing and modifying the user's own files. The present invention is directed to providing reliable security, even when operating with complex data structures, e.g., objects, containing their own program instructions, which are transmitted among users. The present invention also provides enhanced security when processing more conventional programs, even those of questionable origin, e.g., from a computer bulletin board, without exposing system programs or data to the potentially catastrophic consequences of computer viruses or of incompetent programming. The present method and apparatus utilizes a unique operating system design that includes a system monitor which limits the ability of a program about to be executed to the use of predefined resources (e.g., data files, disk writing capabilities etc.). The system monitor builds a data structure including a set of authorities defining that which a program is permitted to do and/or that which the program is precluded from doing. The set of authorities and/or restrictions assigned to a program to be executed are referred to herein as 'program authorization information' (or 'PAI'). Once defined, the program authorization information is thereafter associated with each program to be executed to

U.S. Patent No. 6,192,476 – Claim 14	*Fischer*
	thereby delineate the types of resources and functions that the program is allowed to utilize. The PAI associated with a particular program may be assigned by a computer system owner/user or by someone who the computer system owner/user implicitly trusts. The PAI defines the range of operations that a program may execute and/or defines those operations that a program cannot perform. The program is permitted to access what has been authorized and nothing else. In this fashion, the program may be regarded as being placed in a program capability limiting 'safety box'. This 'safety box' is thereafter associated with the program such that whenever the system monitor runs the program, the PAI for that program is likewise loaded and monitored. When the program is to perform a function or access a resource, the associated PAI is monitored to confirm that the operation is within the defined program limits. If the program attempts to do anything outside the authorized limits, then the program execution is halted. Thus, the present invention advantageously protects a user from any program to be executed. The present invention is particularly advantageous in light of current data processing practices where programs are obtained from a wide range of diverse, untrustworthy places such as computer bulletin boards or other users of unknown trustworthiness." *Fischer* at 1:60-2:55. "FIG. 4 is a flowchart which illustrates how a user may benefit from the use of program authorization information, particularly when executing a program of unknown trustworthiness. As indicated in block 121, a user may have a desire to execute a program of interest in which the user has no knowledge of the program's creator. Thus, the program has unknown trustworthiness and may, for example, have been accessed via an electronic bulletin board and may have arrived at the user's terminal via a telecommunications channel or diskette. Such a program, which might purport to be only a game, carries with it a significant risk that it may be infected by a virus. As indicated in block 122, the user may be protected by defining program authorization information which restricts the program to only unimportant or expendable files. If desired, the user may restrict such a program from modifying any files whatsoever. For example, the

U.S. Patent No. 6,192,476 – Claim 14	*Fischer*
	user may permit the program to only display images on the display screen and to perform game playing related functions. Alternatively, if the program is known to have a single work file, the PAI data may only permit use of such a single file. By limiting access only to a single work file, a program of unknown trustworthiness, cannot inject a virus into other user's programs or otherwise initiate system program malfunctions. Thus, in accordance with the present invention, a user, via a systems program, determines how much of the user's system will be put at risk by such a program so as to, for example, completely eliminate the ability of the program to use any privileged functions. The user then associates, for example, through an operator prompting, menu-driven system, a PAI with every program to be run on the system (or have such PAI or lack of PAI associated through predetermined default mechanisms). A system utility program is preferably employed to create the program authorization information in a manner which will be described in detail below in conjunction with FIGS. 6-9. After the PAI has been assigned, any time the system runs the associated program, the system software (in a manner to be described below) insures that the program safely runs in a manner consistent with the PAI. Thus, the program has been effectively placed in a 'safety box' (124). Turning back to FIG. 1, a program of unknown trust may be injected into the system via communications channel 12 or from a floppy disk loaded into terminal A. The program may be initially stored in, for example, the user's program disk memory 7. Thereafter, the user on keyboard 4 will, through interaction with the system's program identified above (with respect to block 122 of FIG. 4), associate the program authorization information with a program (in a manner such as shown in FIGS. 3A through 3D) such that the program may safely run on the user's system or perhaps, a PAI arrives with the program, in which case it is likely to be signed. FIG. 5 is an illustration of a program control block (PCB) data structure 140 in accordance with an exemplary embodiment of the present invention. The program control block 140 is the data structure utilized by the system monitor to control the execution of an associated program.

U.S. Patent No. 6,192,476 – Claim 14	*Fischer*
	The program control block 140 is loaded with program authorization information such that the PAI can be readily referenced as the associated program is executed so as to insure that the program performs functions and accesses resources in conformance with its assigned authorizations. The program control block associated with the program to be executed is located in a storage area which cannot be modified by the program." *Fischer* at 9:17-10:23.

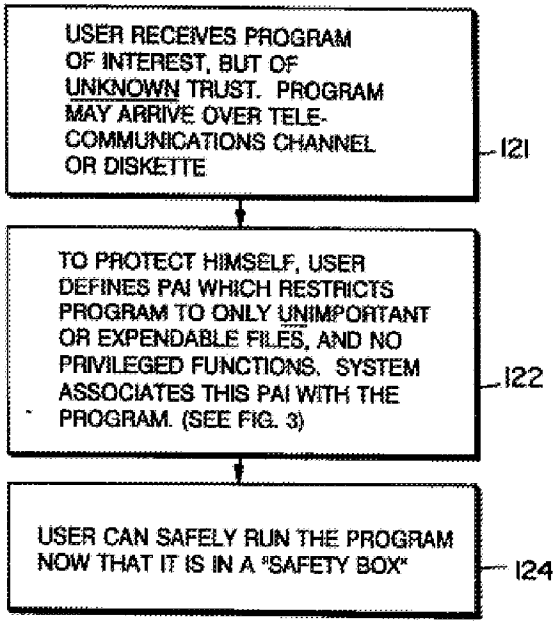
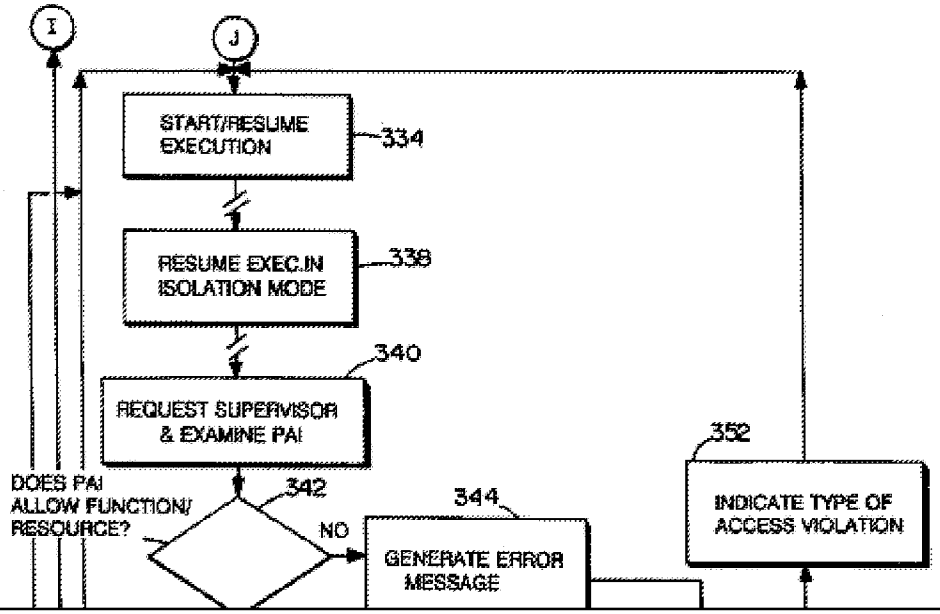
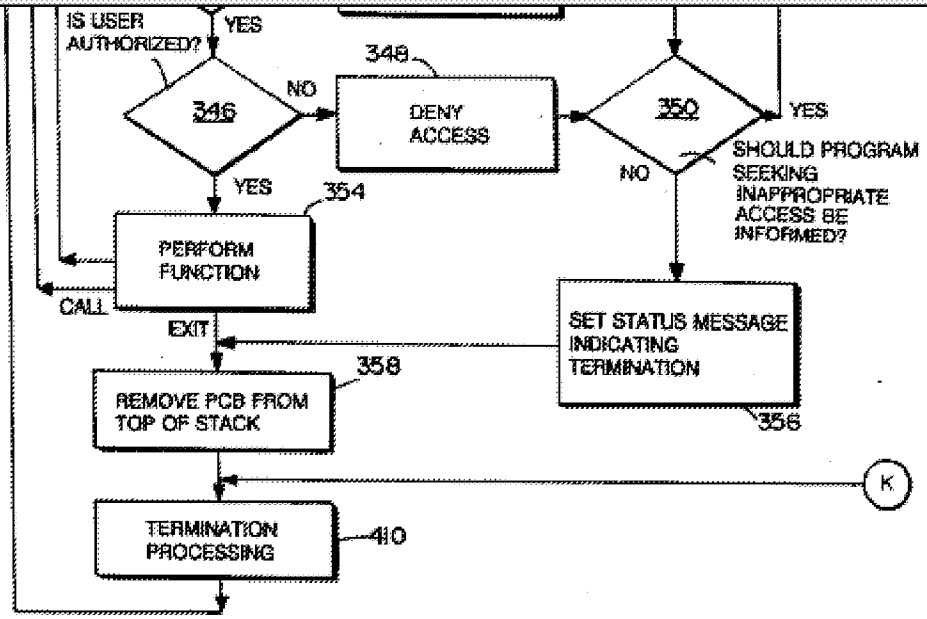


Fig. 4

Fischer at Fig. 4.

"FIGS. 10 and 11 illustrate the sequence of operations of a supervisor program for controlling the processing of a program being executed in accordance with its program authorization information. The processing of a program 'X' and its program authorization information illustrated in FIG. 10 is initiated while the computer is executing a supervisor routine. As shown in FIG. 10 at 300, a calling program calls program X for execution. Thereafter, a program control block is created for program X. The program control block created will not be added to the top of the execution stack until it is determined that the

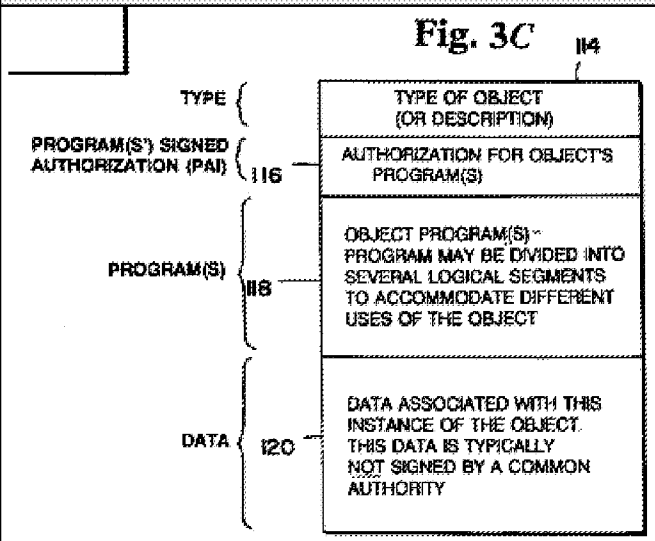
U.S. Patent No. 6,192,476 – Claim 14	*Fischer*
	program is permitted to be invoked and verification is successful completed. Thus, if the program fails a security check, it will not be placed in the program execution chain. In addition to creating a 'tentative' program control block, the called program will be located through an appropriate program directory during the processing in block 302. Thereafter, a check is made at block 304 to determine whether PAI has yet been associated with program X so as to place program X in the so-called 'safety box' described above. This PAI may or may not be signed depending upon its particular application as described above." *Fischer* at 15:56-16:11. 

U.S. Patent No. 6,192,476 – Claim 14	*Fischer*
	 **Fig. 11** *Fischer* at Fig. 11.
determining whether said action is authorized based on an association between permissions and a plurality of routines in a calling hierarchy associated with said principal;	*Fischer* discloses determining whether said action is authorized based on an association between permissions and a plurality of routines in a calling hierarchy associated with said principal. *Fischer* discloses "an originating program" that "calls a program" having a program control block, or PCB. *Fischer* at 10:24-26. "Each new PCB will include a field . . . that points to the 'previous' of calling program control block." *See id.* at 10:27-29. Then, "[w]hen a called program finishes executing, the system removes its associated PCB from the top of the executed stack, removes the associated program from storage, removes the associated authorizing information and accesses the program control block immediately below it in the stack." *See id.* at 10:31-36. *Fischer* also incorporates by reference two patents which explicitly disclose data hierarchies: *see id.* at 6:37 (incorporating by reference U.S. Patent Nos. 4,868,877 and 5,005,200 which disclose an "enhanced digital signature certification" which employs "[a] hierarchy of nested

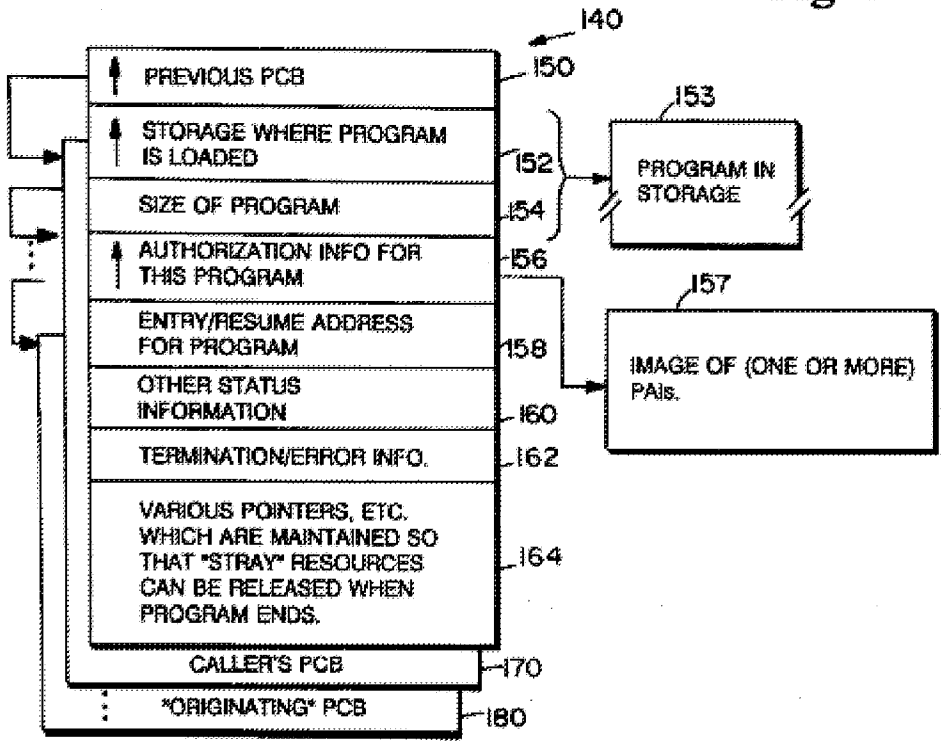
U.S. Patent No. 6,192,476 – Claim 14	*Fischer*
	certifications and signatures." *See* U.S. Patent No. 5,005,200, Abstract. "The present method an apparatus utilizes a unique operation system design that includes a system monitor which limits the ability of a program about to be executed to the use of predefined resources (e.g., data files, disk writing capabilities etc.). The system monitor builds a data structure including a set of authorities defining that which a program is permitted to do and/or that which the program is precluded from doing. The set of authorities and/or restrictions assigned to a program to be executed are referred to herein as 'program authorization information' (or 'PAI'). Once defined, the program authorization information is thereafter associated with each program to be executed to thereby delineate the types of resources and functions that the program is allowed to utilize. The PAI associated with a particular program may be assigned by a computer system owner/user or by someone who the computer system owner/user implicitly trusts. The PAI defines the range of operations that a program may execute and/or defines those operations that a program cannot perform. The program is permitted to access what has been authorized and nothing else. In this fashion, the program may be regarded as being placed in a program capability limiting 'safety box.' This 'safety box' is thereafter associated with the program such that whenever the system monitor runs the program, the PAI for that program is likewise loaded and monitored. When the program is to perform a function or access a resource, the associated PAI is monitored to confirm that the operation is within the defined program limits. If the program attempts to do anything outside the authorized limits, then the program execution is halted." *Fischer* at 2:16-48. "Thereafter, the program X's program authorizing information is combined, as appropriate, with the PAI associated with the PCB of the calling program, if any. This combined PAI, which may include multiple PAI's, is then stored in an area of storage which cannot generally be modified by the program and the address of the PAI is stored in the process control block (PCB) as indicated in field 156 of FIG. 5. Thus, if program X is called by a calling program, it is subject to all its own constraints as well as being combined in some way with the constraints of the calling program, which aggregate constraints are embodied

U.S. Patent No. 6,192,476 – Claim 14	*Fischer*
	into program X's PAI. In this fashion, a calling program may not be permitted to exceed its assigned bounds by merely calling another program." *Fischer* at 19:40-54. "Even programs with no known trustworthiness can be used after program authorization information associates a wide range of restrictions to thereby allow potentially beneficial programs to be safely used--even if they do not have an official certification of trust. The present invention also allows an unlimited number of different resources and functions to be controlled. For example, some useful resources/functions which may be controlled include: the ability to limit a program to certain files or data sets; the ability to transmit data via electronic mail to someone outside the user's domain; the ability of a program to create or solicit digital signatures; the ability to limit access to a program of certain security classes, etc." *Fischer* at 3:48-61. "The present invention is directed to providing reliable security, even when operating with complex data structures, e.g., objects, containing their own program instructions, which are transmitted among users." *Fischer* at 2:6-9. "Through the use of the present invention, general object oriented data may be transferred from user to user without exposing users to the potential dangers of viruses or mischievous users." *Fischer* at 4:10-13. "In one contemplated embodiment of the present invention, programs may be part of data objects, which are written in a high-level control language and are executed by a standardized interpreter program which executes this high-level language. In this case, part of the interpreter's task is to verify that the functions encountered in the high level logic are, in fact, permissible. If such tasks are not permissible, the interpreter then suppresses the execution of the program not authorized to perform such tasks." *Fischer* at 3:11-20. "In accordance with the present invention, a PAI is associated with programs to be executed. FIGS. 3A through 3D depict four exemplary approaches for associating program authorization information with a program. . . .

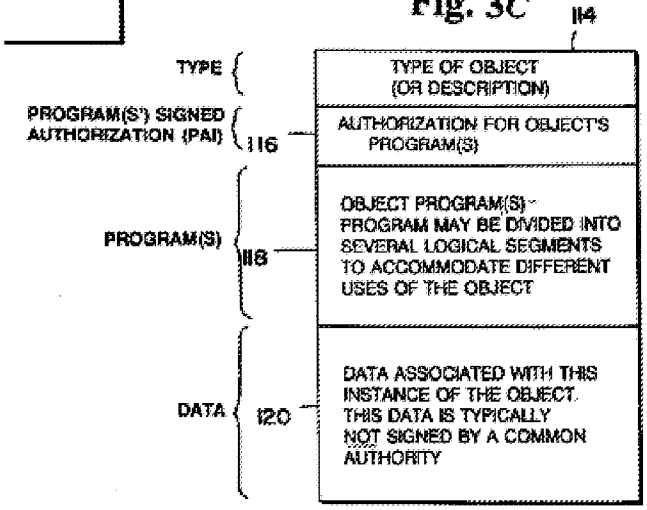
U.S. Patent No. 6,192,476 – Claim 14	*Fischer*
	. . . FIG. 3C shows an important application in which a PAI data structure is associated with a program according to an embodiment of the present invention. FIG. 3C shows an illustrative data structure for a secure exchangeable 'object'. The data structure may be signed by a trusted authority. The signing of such a data structure allows the object to be securely transmitted from user to user. Although the data structure shown in FIG. 3 is set forth in a general format, it may be structured as set forth in the inventor's copending application filed on Apr. 6, 1992 and entitled 'Method and Apparatus for Creating, Supporting and Processing a Travelling Program' (U.S. Ser. No. 07/863,552.), which application is hereby expressly incorporated herein by reference. . . . The program authorization information is embedded in a segment 116 which specifies the authorization for the object's program or programs in a manner to be described more fully hereinafter." *Fischer* at 7:14-18, 7:49-8:2. "Thereafter, the PAI is stored using, for example, one of the approaches set forth in FIGS. 3A through 3D so that it is associated with its program 272 . . . ." *Fischer* at 15:24-26.

U.S. Patent No. 6,192,476 – Claim 14	*Fischer*
	 *Fischer* at Fig. 3C. "FIGS. 10 and 11 illustrate the sequence of operations of a supervisor program for controlling the processing of a program being executed in accordance with its program authorization information." *Fischer* at 15:56-59. "Depending on the processing in block 316, a decision is made in block 322 whether the signatures are valid, authorized and trusted. If the signatures are not determined to be valid, then the routing branches to block 324 where the execution in program X is suppressed." *Fischer* at 16:66-17:3. "If the processing in blocks 322 and 316 reveal that the signatures are valid, then the processing in block 326 is performed." *Fischer* at 17:31-33. "FIG. 5 is an illustration of a program control block (PCB) data structure 140 in accordance with an exemplary embodiment of the present invention. The program control block 140 is the data structure utilized by the system monitor to control the execution of an associated

U.S. Patent No. 6,192,476 – Claim 14	*Fischer*
	program. The program control block 140 is loaded with program authorization information such that the PAI can be readily referenced as the associated program is executed so as to insure that the program performs functions and accesses resources in conformance with its assigned authorizations. The program control block associated with the program to be executed is located in a storage area which cannot be modified by the program. As shown in FIG. 5, an originating program (whose PCB is identified at 180) calls a program (having a PCB 170) which will, in turn, will call the program 140 is shown in detail in FIG. 5. Each new PCB will include a field such as 150 that points to the 'previous' or calling program control block. A field may also be utilized to identify the 'next' program control block file. When a called program finishes executing, the system removes its associated PCB from the top of the executed stack, removes the associated program from storage, removes the associated authorizing information and accesses the program control block immediately below it in the stack. When another program is called, the reverse process occurs such that a new PCB is created which is placed on top of the stack, which again points to the previous PCB as indicated in field 150." *Fischer* at 10:8-39.

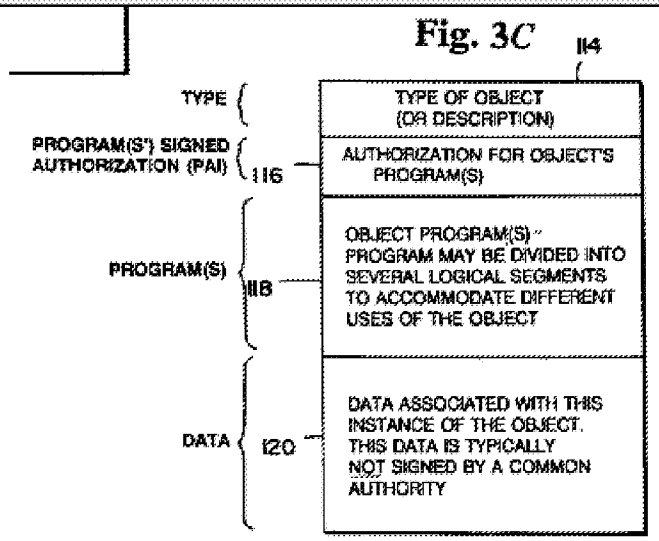
U.S. Patent No. 6,192,476 – Claim 14	*Fischer*
	Fig. 5  *Fischer* at Fig. 5. "Thereafter, the program X's program authorizing information is combined, as appropriate, with the PAI associated with the PCB of the calling program, if any. This combined PAI, which may include multiple PAI's, is then stored in an area of storage which cannot generally be modified by the program and the address of the PAI is stored in the process control block (PCB) as indicated in field 156 of FIG. 5. Thus, if program X is called by a calling program, it is subject to all its own constraints as well as being combined in some way with the constraints of the calling program, which aggregate constraints are embodied into program X's PAI. In this fashion, a calling program may not be permitted to exceed its

U.S. Patent No. 6,192,476 – Claim 14	*Fischer*
	assigned bounds by merely calling another program. There are many alternative ways that a program's PAI could be combined with the PAI of the program which invokes it--depending on the strategies which are applicable to the current environment, and the inherent nature of the programs themselves. It may even be likely that even the method of combination is itself one of the PAI authorities, or qualifiers, of either or both the invoking or invoked program. For example, it is reasonable to restrict a called program to the lesser of its 'normal' PAI authority and that of its calling program--to insure the calling program cannot mischievously misuse the called program's greater authority to circumvent its own limitations. On the other hand, for called programs which carefully verify their own actions, it could be possible to allow the called program greater inherent authority than the program which calls it--this way sensitive resources could be made available to wider use by mediating such use through trusted sub-programs. The possibilities for such combination must be carefully considered, not only by the designers of the underlying control system, but also by those who assign authority to each program. Thereafter, the program is loaded and the hash of the program is computed based on the algorithm specified in the program's PAI." *Fischer* at 17:40-18:10.
wherein each routine of said plurality of routines is associated with a class; and	*Fischer* discloses wherein each routine of said plurality of routines is associated with a class. For example, *Fischer* discloses that "[w]hen another program is called, the reverse process occurs such that a new PCB is created which is placed on top of the stack, which again points to the previous PCB as indicated in field 150." *Fischer* at 10:36-39. Such a program, which inherently implements code to be implemented on a stack, would also include a class. "In accordance with the present invention, a PAI is associated with programs to be executed. FIGS. 3A through 3D depict four exemplary approaches for associating program authorization information with a program. . . . . . . FIG. 3C shows an important application in which a PAI data structure is associated with a program according to an embodiment of the present invention. FIG. 3C shows an

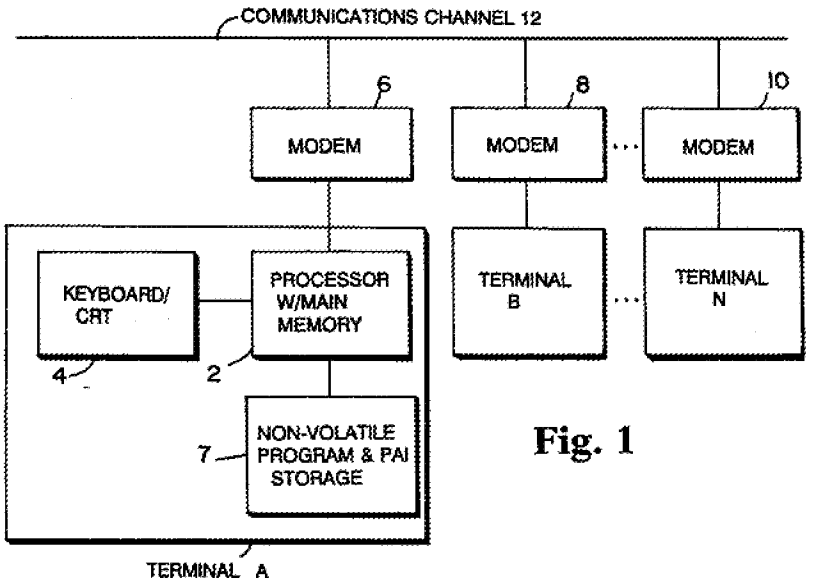
U.S. Patent No. 6,192,476 – Claim 14	*Fischer*
	illustrative data structure for a secure exchangeable 'object'. The data structure may be signed by a trusted authority. The signing of such a data structure allows the object to be securely transmitted from user to user. Although the data structure shown in FIG. 3 is set forth in a general format, it may be structured as set forth in the inventor's copending application filed on Apr. 6, 1992 and entitled 'Method and Apparatus for Creating, Supporting and Processing a Travelling Program' (U.S. Ser. No. 07/863,552.), which application is hereby expressly incorporated herein by reference. . . . The program authorization information is embedded in a segment 116 which specifies the authorization for the object's program or programs in a manner to be described more fully hereinafter." *Fischer* at 7:14-18, 7:49-8:2. "Thereafter, the PAI is stored using, for example, one of the approaches set forth in FIGS. 3A through 3D so that it is associated with its program 272 . . . ." *Fischer* at 15:24-26. Fig. 3C  *Fischer* at Fig. 3C.

U.S. Patent No. 6,192,476 – Claim 14	*Fischer*
wherein said association between permissions and said plurality of routines is based on a second association between classes and protection domains.	*Fischer* discloses said association between permissions and said plurality of routines is based on a second association between classes and protection domains. *Fischer* discloses "an originating program" that "calls a program" having a program control block, or PCB. *Fischer* at 10:24-26. "Each new PCB will include a field . . . that points to the 'previous' of calling program control block." *See id.* at 10:27-29. Then, "[w]hen a called program finishes executing, the system removes its associated PCB from the top of the executed stack, removes the associated program from storage, removes the associated authorizing information and accesses the program control block immediately below it in the stack." *See id.* at 10:31-36. In this disclosure, the "program control block immediately below [the first control block] in the stack" is directly analogous to the second association between classes and protection domains. "Once defined, the program authorization information [(PAI)] is thereafter associated with each program to be executed to thereby delineate the types of resources and functions that the program is allowed to utilize. The PAI associated with a particular program may be assigned by a computer system owner/user or by someone who the computer system owner/user implicitly trusts." *Fischer* at 2:26-33. "The present invention is directed to providing reliable security, even when operating with complex data structures, e.g., objects, containing their own program instructions, which are transmitted among users." *Fischer* at 2:6-9. "Through the use of the present invention, general object oriented data may be transferred from user to user without exposing users to the potential dangers of viruses or mischievous users." *Fischer* at 4:10-13. "In one contemplated embodiment of the present invention, programs may be part of data objects, which are written in a high-level control language and are executed by a standardized interpreter program which executes this high-level language. In this case, part of the interpreter's task is to verify that the functions encountered in the high level logic are, in fact, permissible. If such tasks are not permissible, the interpreter then suppresses the execution of the program not authorized to perform such tasks." *Fischer* at 3:11-20.

U.S. Patent No. 6,192,476 – Claim 14	*Fischer*
	"In accordance with the present invention, a PAI is associated with programs to be executed. FIGS. 3A through 3D depict four exemplary approaches for associating program authorization information with a program. . . . . . . FIG. 3C shows an important application in which a PAI data structure is associated with a program according to an embodiment of the present invention. FIG. 3C shows an illustrative data structure for a secure exchangeable 'object'. The data structure may be signed by a trusted authority. The signing of such a data structure allows the object to be securely transmitted from user to user. Although the data structure shown in FIG. 3 is set forth in a general format, it may be structured as set forth in the inventor's copending application filed on Apr. 6, 1992 and entitled 'Method and Apparatus for Creating, Supporting and Processing a Travelling Program' (U.S. Ser. No. 07/863,552.), which application is hereby expressly incorporated herein by reference. . . . The program authorization information is embedded in a segment 116 which specifies the authorization for the object's program or programs in a manner to be described more fully hereinafter." *Fischer* at 7:14-18, 7:49-8:2. "Thereafter, the PAI is stored using, for example, one of the approaches set forth in FIGS. 3A through 3D so that it is associated with its program 272 . . . ." *Fischer* at 15:24-26.

U.S. Patent No. 6,192,476 – Claim 14	*Fischer*
	 *Fischer* at Fig. 3C.

U.S. Patent No. 6,192,476 – Claim 15	*Fischer*
15. A computer-readable medium carrying one or more sequences of one or more instructions, the one or more sequences of the one or more instructions including instructions which, when executed by one or more processors, causes the one or more processors to perform the steps of:	*Fischer* discloses computer-readable medium, e.g., "a disk memory device," carrying one or more sequences of one or more instructions, the one or more sequences of the one or more instructions including instructions which, when executed by one or more processors, e.g., IBM PC's having a processor," causes the one or more processors to perform steps. "FIG. 1 shows in block diagram form an exemplary communications system which may be used in conjunction with the present invention. . . . Terminals, A, B . . . N may, by way of example only, be IBM PC's having a processor (with main memory) 2 which is coupled to a conventional keyboard/CRT display 4. Additionally, each processor is preferably coupled to a non-volatile program and program authorization information (PAI) storage 7 which may be a disk memory device."

U.S. Patent No. 6,192,476 – Claim 15	*Fischer*
	 **Fig. 1** *Fischer* at 4:45-58 & Fig. 1. "Turning back to FIG. 1, a program of unknown trust may be injected into the system via communications channel 12 or from a floppy disk loaded into terminal A. The program may be initially stored in, for example, the user's program disk memory 7." *Fischer* at 9:64-68.
detecting when a request for an action is made by a principal; and	*Fischer* discloses detecting when a request for an action is made by a principal. *Fischer* inquires into the permissions of a given principal's PAI upon the principal's request for an action: "When the program is to perform a function or access a resource, the associated PAI is monitored to confirm that the operation is within the defined program limits. If the program attempts to do anything outside the authorized limits, then the program execution is halted." *Id.* at 2:43-48. Thus it inherent that *Fischer* detects a request of an action made by a principal. "Prior art operating systems are typically designed to protect data from computer users. In

U.S. Patent No. 6,192,476 – Claim 15	*Fischer*
	such systems, users are often assigned various authorities and are thereafter able to execute programs based on their associated authority. If a program is executing which exceeds the user's assigned authority, then such a system will halt execution of the program. Such prior art systems do not adequately protect computer users from computer viruses or the like. There are security systems which protect certain 'system' related files from being modified by a program. However, such systems do not typically protect a computer user from a program executing and modifying the user's own files. The present invention is directed to providing reliable security, even when operating with complex data structures, e.g., objects, containing their own program instructions, which are transmitted among users. The present invention also provides enhanced security when processing more conventional programs, even those of questionable origin, e.g., from a computer bulletin board, without exposing system programs or data to the potentially catastrophic consequences of computer viruses or of incompetent programming. The present method and apparatus utilizes a unique operating system design that includes a system monitor which limits the ability of a program about to be executed to the use of predefined resources (e.g., data files, disk writing capabilities etc.). The system monitor builds a data structure including a set of authorities defining that which a program is permitted to do and/or that which the program is precluded from doing. The set of authorities and/or restrictions assigned to a program to be executed are referred to herein as 'program authorization information' (or 'PAI'). Once defined, the program authorization information is thereafter associated with each program to be executed to thereby delineate the types of resources and functions that the program is allowed to utilize. The PAI associated with a particular program may be assigned by a computer system owner/user or by someone who the computer system owner/user implicitly trusts. The PAI defines the range of operations that a program may execute and/or defines those operations that a program cannot perform. The program is permitted to access what has been authorized and nothing else. In this fashion, the program may be regarded as being placed in

U.S. Patent No. 6,192,476 – Claim 15	*Fischer*
	a program capability limiting 'safety box'. This 'safety box' is thereafter associated with the program such that whenever the system monitor runs the program, the PAI for that program is likewise loaded and monitored. When the program is to perform a function or access a resource, the associated PAI is monitored to confirm that the operation is within the defined program limits. If the program attempts to do anything outside the authorized limits, then the program execution is halted. Thus, the present invention advantageously protects a user from any program to be executed. The present invention is particularly advantageous in light of current data processing practices where programs are obtained from a wide range of diverse, untrustworthy places such as computer bulletin boards or other users of unknown trustworthiness." *Fischer* at 1:60-2:55. "FIG. 4 is a flowchart which illustrates how a user may benefit from the use of program authorization information, particularly when executing a program of unknown trustworthiness. As indicated in block 121, a user may have a desire to execute a program of interest in which the user has no knowledge of the program's creator. Thus, the program has unknown trustworthiness and may, for example, have been accessed via an electronic bulletin board and may have arrived at the user's terminal via a telecommunications channel or diskette. Such a program, which might purport to be only a game, carries with it a significant risk that it may be infected by a virus. As indicated in block 122, the user may be protected by defining program authorization information which restricts the program to only unimportant or expendable files. If desired, the user may restrict such a program from modifying any files whatsoever. For example, the user may permit the program to only display images on the display screen and to perform game playing related functions. Alternatively, if the program is known to have a single work file, the PAI data may only permit use of such a single file. By limiting access only to a single work file, a program of unknown trustworthiness, cannot inject a virus into other user's programs or otherwise initiate system program malfunctions. Thus, in accordance with the present invention, a user, via a systems program, determines how much of the user's system will be put at risk by such a program so as to, for example, completely

U.S. Patent No. 6,192,476 – Claim 15	*Fischer*
	eliminate the ability of the program to use any privileged functions. The user then associates, for example, through an operator prompting, menu-driven system, a PAI with every program to be run on the system (or have such PAI or lack of PAI associated through predetermined default mechanisms). A system utility program is preferably employed to create the program authorization information in a manner which will be described in detail below in conjunction with FIGS. 6-9. After the PAI has been assigned, any time the system runs the associated program, the system software (in a manner to be described below) insures that the program safely runs in a manner consistent with the PAI. Thus, the program has been effectively placed in a 'safety box' (124). Turning back to FIG. 1, a program of unknown trust may be injected into the system via communications channel 12 or from a floppy disk loaded into terminal A. The program may be initially stored in, for example, the user's program disk memory 7. Thereafter, the user on keyboard 4 will, through interaction with the system's program identified above (with respect to block 122 of FIG. 4), associate the program authorization information with a program (in a manner such as shown in FIGS. 3A through 3D) such that the program may safely run on the user's system or perhaps, a PAI arrives with the program, in which case it is likely to be signed. FIG. 5 is an illustration of a program control block (PCB) data structure 140 in accordance with an exemplary embodiment of the present invention. The program control block 140 is the data structure utilized by the system monitor to control the execution of an associated program. The program control block 140 is loaded with program authorization information such that the PAI can be readily referenced as the associated program is executed so as to insure that the program performs functions and accesses resources in conformance with its assigned authorizations. The program control block associated with the program to be executed is located in a storage area which cannot be modified by the program." *Fischer* at 9:17-10:23.

U.S. Patent No. 6,192,476 – Claim 15	*Fischer*

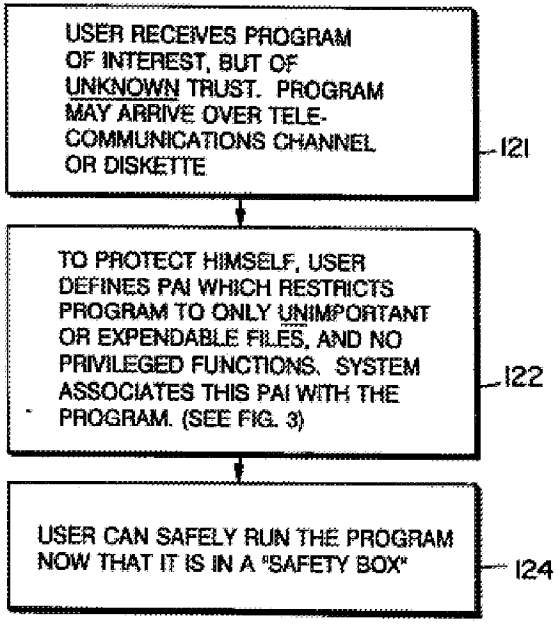
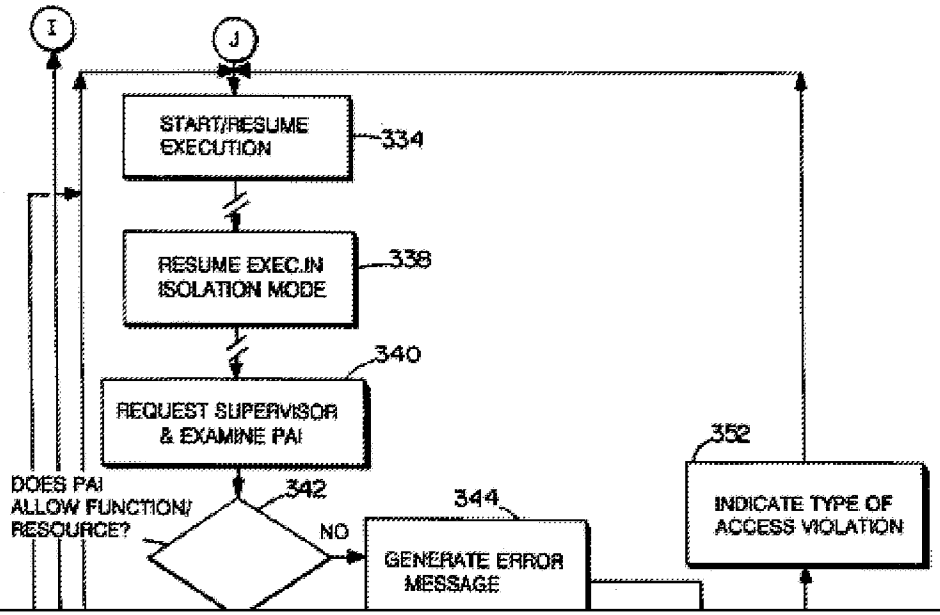


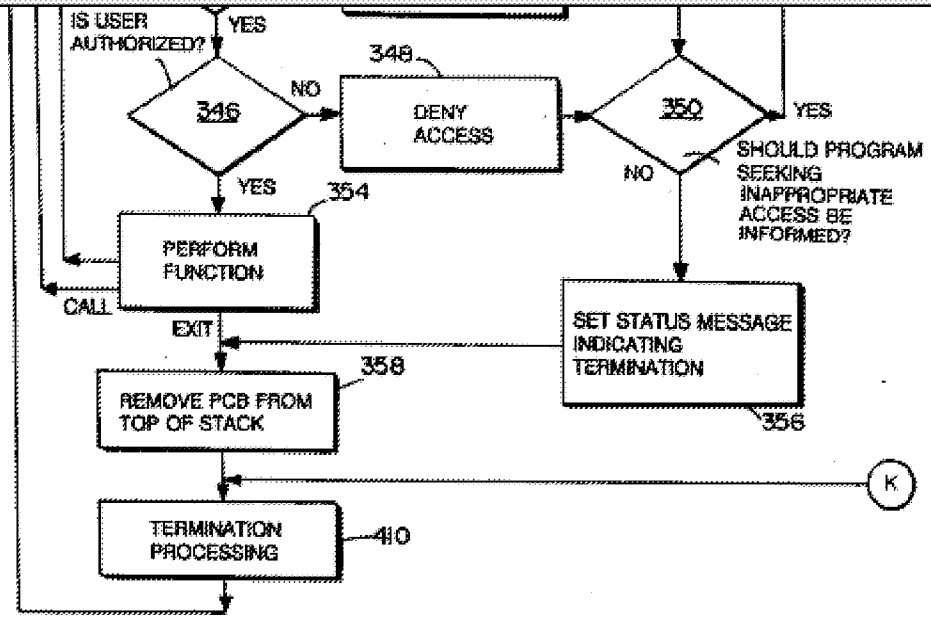
Fig. 4

Fischer at Fig. 4.

"FIGS. 10 and 11 illustrate the sequence of operations of a supervisor program for controlling the processing of a program being executed in accordance with its program authorization information. The processing of a program 'X' and its program authorization information illustrated in FIG. 10 is initiated while the computer is executing a supervisor routine. As shown in FIG. 10 at 300, a calling program calls program X for execution. Thereafter, a program control block is created for program X. The program control block created will not be added to the top of the execution stack until it is determined that the program is permitted to be invoked and verification is successful completed. Thus, if the program fails a security check, it will not be placed in the program execution chain. In addition to creating a 'tentative' program control block, the called program will be located through an appropriate program directory during the processing in block 302.

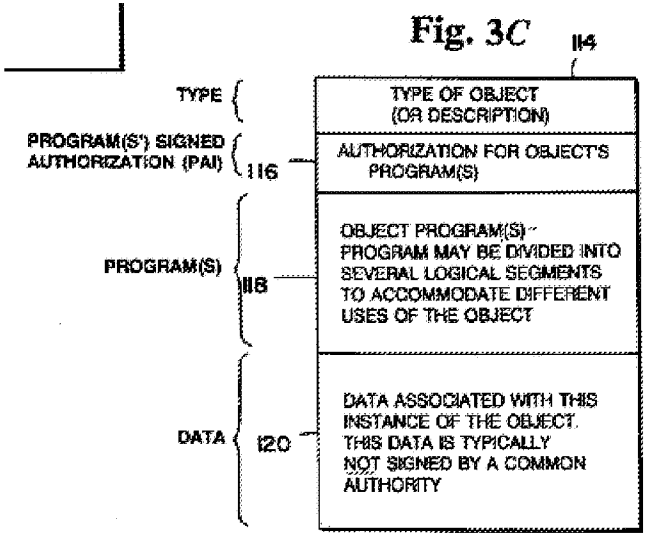
Thereafter, a check is made at block 304 to determine whether PAI has yet been associated

U.S. Patent No. 6,192,476 – Claim 15	*Fischer*
	with program X so as to place program X in the so-called 'safety box' described above. This PAI may or may not be signed depending upon its particular application as described above." *Fischer* at 15:56-16:11. 

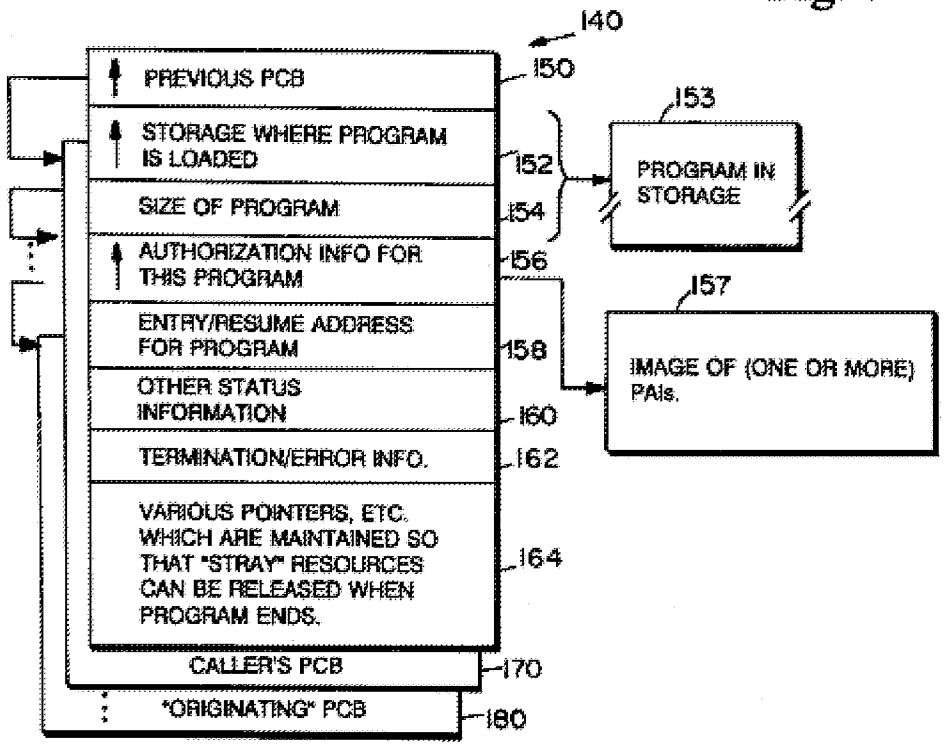
U.S. Patent No. 6,192,476 – Claim 15	*Fischer*
	 **Fig. 11** *Fischer* at Fig. 11.
in response to detecting the request, determining whether said action is authorized based on permissions associated with a plurality of routines in a calling hierarchy associated with said principal, wherein a first routine in said calling hierarchy is privileged; and	*Fischer* discloses in response to detecting the request, determining whether said action is authorized based on permissions associated with a plurality of routines in a calling hierarchy associated with said principal, wherein a first routine in said calling hierarchy is privileged. The limitation of "privileged" routines is also disclosed by *Fischer*; for example: "If the processing in blocks 322 and 316 reveal that the signatures are valid, then the processing in block 326 is performed." *Fischer* at 17:31-33. It is inherent that if the processing is not valid, the routine will not be performed, and is thus "privileged." "The present method an apparatus utilizes a unique operation system design that includes a system monitor which limits the ability of a program about to be executed to the use of predefined resources (e.g., data files, disk writing capabilities etc.). The system monitor builds a data structure including a set of authorities defining that which a program is

U.S. Patent No. 6,192,476 – Claim 15	*Fischer*
	permitted to do and/or that which the program is precluded from doing. The set of authorities and/or restrictions assigned to a program to be executed are referred to herein as 'program authorization information' (or 'PAI'). Once defined, the program authorization information is thereafter associated with each program to be executed to thereby delineate the types of resources and functions that the program is allowed to utilize. The PAI associated with a particular program may be assigned by a computer system owner/user or by someone who the computer system owner/user implicitly trusts. The PAI defines the range of operations that a program may execute and/or defines those operations that a program cannot perform. The program is permitted to access what has been authorized and nothing else. In this fashion, the program may be regarded as being placed in a program capability limiting 'safety box.' This 'safety box' is thereafter associated with the program such that whenever the system monitor runs the program, the PAI for that program is likewise loaded and monitored. When the program is to perform a function or access a resource, the associated PAI is monitored to confirm that the operation is within the defined program limits. If the program attempts to do anything outside the authorized limits, then the program execution is halted." *Fischer* at 2:16-48. "Thereafter, the program X's program authorizing information is combined, as appropriate, with the PAI associated with the PCB of the calling program, if any. This combined PAI, which may include multiple PAI's, is then stored in an area of storage which cannot generally be modified by the program and the address of the PAI is stored in the process control block (PCB) as indicated in field 156 of FIG. 5. Thus, if program X is called by a calling program, it is subject to all its own constraints as well as being combined in some way with the constraints of the calling program, which aggregate constraints are embodied into program X's PAI. In this fashion, a calling program may not be permitted to exceed its assigned bounds by merely calling another program." *Fischer* at 19:40-54. "Even programs with no known trustworthiness can be used after program authorization information associates a wide range of restrictions to thereby allow potentially beneficial programs to be safely used--even if they do not have an official certification of trust.

U.S. Patent No. 6,192,476 – Claim 15	*Fischer*
	The present invention also allows an unlimited number of different resources and functions to be controlled. For example, some useful resources/functions which may be controlled include: the ability to limit a program to certain files or data sets; the ability to transmit data via electronic mail to someone outside the user's domain; the ability of a program to create or solicit digital signatures; the ability to limit access to a program of certain security classes, etc." *Fischer* at 3:48-61. "The present invention is directed to providing reliable security, even when operating with complex data structures, e.g., objects, containing their own program instructions, which are transmitted among users." *Fischer* at 2:6-9. "Through the use of the present invention, general object oriented data may be transferred from user to user without exposing users to the potential dangers of viruses or mischievous users." *Fischer* at 4:10-13. "In one contemplated embodiment of the present invention, programs may be part of data objects, which are written in a high-level control language and are executed by a standardized interpreter program which executes this high-level language. In this case, part of the interpreter's task is to verify that the functions encountered in the high level logic are, in fact, permissible. If such tasks are not permissible, the interpreter then suppresses the execution of the program not authorized to perform such tasks." *Fischer* at 3:11-20. "In accordance with the present invention, a PAI is associated with programs to be executed. FIGS. 3A through 3D depict four exemplary approaches for associating program authorization information with a program. . . . . . . FIG. 3C shows an important application in which a PAI data structure is associated with a program according to an embodiment of the present invention. FIG. 3C shows an illustrative data structure for a secure exchangeable 'object'. The data structure may be

U.S. Patent No. 6,192,476 – Claim 15	*Fischer*
	signed by a trusted authority. The signing of such a data structure allows the object to be securely transmitted from user to user. Although the data structure shown in FIG. 3 is set forth in a general format, it may be structured as set forth in the inventor's copending application filed on Apr. 6, 1992 and entitled 'Method and Apparatus for Creating, Supporting and Processing a Travelling Program' (U.S. Ser. No. 07/863,552.), which application is hereby expressly incorporated herein by reference. . . . The program authorization information is embedded in a segment 116 which specifies the authorization for the object's program or programs in a manner to be described more fully hereinafter." *Fischer* at 7:14-18, 7:49-8:2. "Thereafter, the PAI is stored using, for example, one of the approaches set forth in FIGS. 3A through 3D so that it is associated with its program 272 . . . ." *Fischer* at 15:24-26. Fig. 3C  *Fischer* at Fig. 3C.

U.S. Patent No. 6,192,476 – Claim 15	*Fischer*
	"FIGS. 10 and 11 illustrate the sequence of operations of a supervisor program for controlling the processing of a program being executed in accordance with its program authorization information." *Fischer* at 15:56-59. "Depending on the processing in block 316, a decision is made in block 322 whether the signatures are valid, authorized and trusted. If the signatures are not determined to be valid, then the routing branches to block 324 where the execution in program X is suppressed." *Fischer* at 16:66-17:3. "If the processing in blocks 322 and 316 reveal that the signatures are valid, then the processing in block 326 is performed." *Fischer* at 17:31-33. "FIG. 5 is an illustration of a program control block (PCB) data structure 140 in accordance with an exemplary embodiment of the present invention. The program control block 140 is the data structure utilized by the system monitor to control the execution of an associated program. The program control block 140 is loaded with program authorization information such that the PAI can be readily referenced as the associated program is executed so as to insure that the program performs functions and accesses resources in conformance with its assigned authorizations. The program control block associated with the program to be executed is located in a storage area which cannot be modified by the program. As shown in FIG. 5, an originating program (whose PCB is identified at 180) calls a program (having a PCB 170) which will, in turn, will call the program 140 is shown in detail in FIG. 5. Each new PCB will include a field such as 150 that points to the 'previous' or calling program control block. A field may also be utilized to identify the 'next' program control block file. When a called program finishes executing, the system removes its associated PCB from the top of the executed stack, removes the associated program from storage, removes the associated authorizing information and accesses the program control block immediately

U.S. Patent No. 6,192,476 – Claim 15	*Fischer*
	below it in the stack. When another program is called, the reverse process occurs such that a new PCB is created which is placed on top of the stack, which again points to the previous PCB as indicated in field 150." *Fischer* at 10:8-39. **Fig. 5**  *Fischer* at Fig. 5. "Thereafter, the program X's program authorizing information is combined, as appropriate, with the PAI associated with the PCB of the calling program, if any. This combined PAI, which may include multiple PAI's, is then stored in an area of storage which cannot generally be modified by the program and the address of the PAI is stored in the process control block (PCB) as indicated in field 156 of FIG. 5. Thus, if program X is called by a

U.S. Patent No. 6,192,476 – Claim 15	*Fischer*
	calling program, it is subject to all its own constraints as well as being combined in some way with the constraints of the calling program, which aggregate constraints are embodied into program X's PAI. In this fashion, a calling program may not be permitted to exceed its assigned bounds by merely calling another program. There are many alternative ways that a program's PAI could be combined with the PAI of the program which invokes it--depending on the strategies which are applicable to the current environment, and the inherent nature of the programs themselves. It may even be likely that even the method of combination is itself one of the PAI authorities, or qualifiers, of either or both the invoking or invoked program. For example, it is reasonable to restrict a called program to the lesser of its 'normal' PAI authority and that of its calling program--to insure the calling program cannot mischievously misuse the called program's greater authority to circumvent its own limitations. On the other hand, for called programs which carefully verify their own actions, it could be possible to allow the called program greater inherent authority than the program which calls it--this way sensitive resources could be made available to wider use by mediating such use through trusted sub-programs. The possibilities for such combination must be carefully considered, not only by the designers of the underlying control system, but also by those who assign authority to each program. Thereafter, the program is loaded and the hash of the program is computed based on the algorithm specified in the program's PAI." *Fischer* at 17:40-18:10.
wherein the step of determining whether said action is authorized further includes determining whether a permission required to perform said action is encompassed by at least one permission associated with each routine in said calling hierarchy between and including said first routine and a second routine in said calling hierarchy, wherein said second routine is invoked after said first routine, wherein said second routine is a routine for performing said requested	*Fischer* discloses wherein the step of determining whether said action is authorized further includes determining whether a permission required to perform said action is encompassed by at least one permission associated with each routine in said calling hierarchy between and including said first routine and a second routine in said calling hierarchy, wherein said second routine is invoked after said first routine, wherein said second routine is a routine for performing said requested action. For example, *Fischer* discloses calling a plurality of routines: "As shown in FIG. 5, an originating program (whose PCB is identified at 180) calls a program (having a PCB 170) which will, in turn, will call the program 140 is shown in detail in FIG. 5." *Fischer* at 10:24-27. This inherently implicates a first and a second routine. *Fischer* also discloses a calling hierarchy as described in claim 1. "Thereafter, the program X's program authorizing information is combined, as appropriate,

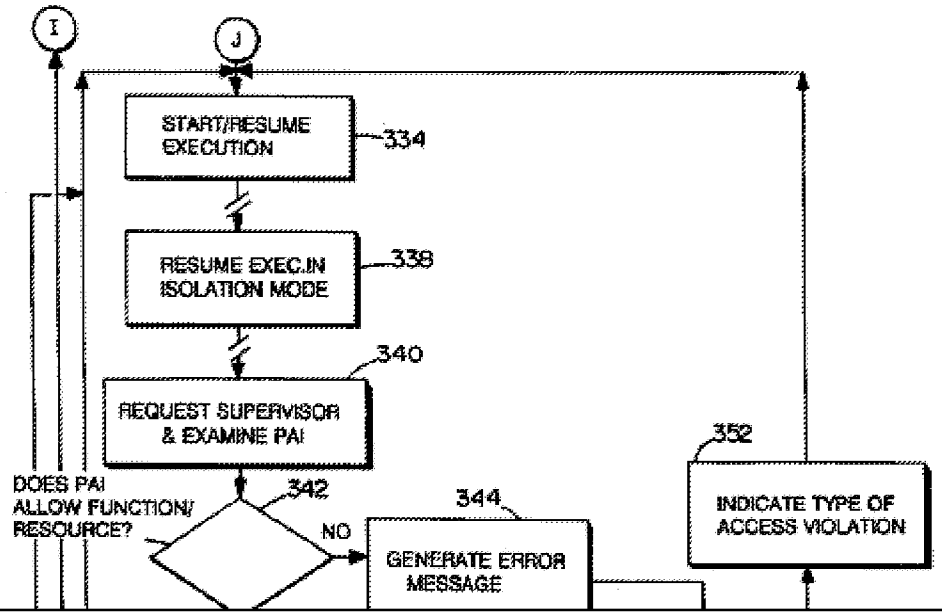
U.S. Patent No. 6,192,476 – Claim 15	*Fischer*
action.	with the PAI associated with the PCB of the calling program, if any. This combined PAI, which may include multiple PAI's, is then stored in an area of storage which cannot generally be modified by the program and the address of the PAI is stored in the process control block (PCB) as indicated in field 156 of FIG. 5. Thus, if program X is called by a calling program, it is subject to all its own constraints as well as being combined in some way with the constraints of the calling program, which aggregate constraints are embodied into program X's PAI. In this fashion, a calling program may not be permitted to exceed its assigned bounds by merely calling another program. There are many alternative ways that a program's PAI could be combined with the PAI of the program which invokes it--depending on the strategies which are applicable to the current environment, and the inherent nature of the programs themselves. It may even be likely that even the method of combination is itself one of the PAI authorities, or qualifiers, of either or both the invoking or invoked program. For example, it is reasonable to restrict a called program to the lesser of its 'normal' PAI authority and that of its calling program--to insure the calling program cannot mischievously misuse the called program's greater authority to circumvent its own limitations. On the other hand, for called programs which carefully verify their own actions, it could be possible to allow the called program greater inherent authority than the program which calls it--this way sensitive resources could be made available to wider use by mediating such use through trusted sub-programs. The possibilities for such combination must be carefully considered, not only by the designers of the underlying control system, but also by those who assign authority to each program. Thereafter, the program is loaded and the hash of the program is computed based on the algorithm specified in the program's PAI." *Fischer* at 17:40-18:10. "The present invention, while it primarily focuses on defining functions which restrict the ability of a program to access resources normally allowed to users, could also, in an appropriate environment, be used to extend the capabilities beyond those normally allowed to a user. Thus, for example, programs whose PAI is signed by an authority recognized by the supervisor, could be allowed to perform extended functions." *Fischer* at 12:58-65.

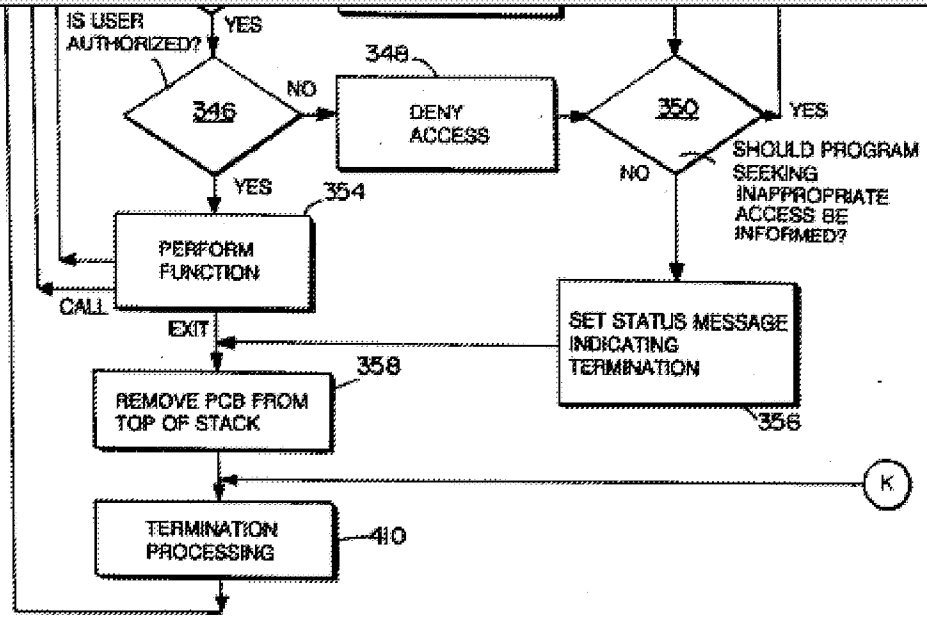
U.S. Patent No. 6,192,476 – Claim 16	*Fischer*

U.S. Patent No. 6,192,476 – Claim 16	*Fischer*
16. The computer readable medium of claim 15, wherein the step of determining whether said permission required to perform said action is encompassed by at least one permission associated with each routine in said calling hierarchy between and including said first routine and said second routine further includes the steps of: determining whether said permission required is encompassed by at least one permission associated with said second routine; and	*Fischer* discloses the computer readable medium of claim 15, wherein the step of determining whether said permission required to perform said action is encompassed by at least one permission associated with each routine in said calling hierarchy between and including said first routine and said second routine further includes the steps of: determining whether said permission required is encompassed by at least one permission associated with said second routine. *Fischer* discloses "an originating program" that "calls a program" having a program control block, or PCB. *Fischer* at 10:24-26. "Each new PCB will include a field . . . that points to the 'previous' of calling program control block." *See id.* at 10:27-29. Then, "[w]hen a called program finishes executing, the system removes its associated PCB from the top of the executed stack, removes the associated program from storage, removes the associated authorizing information and accesses the program control block immediately below it in the stack." *See id.* at 10:31-36. In this disclosure, the "program control block immediately below [the first control block] in the stack" is directly analogous to a second routine. And it is the goal of *Fischer* to limit the execution of tasks depending on the permission or "PAI" of the routine or data structure in question. "Thereafter, the program X's program authorizing information is combined, as appropriate, with the PAI associated with the PCB of the calling program, if any. This combined PAI, which may include multiple PAI's, is then stored in an area of storage which cannot generally be modified by the program and the address of the PAI is stored in the process control block (PCB) as indicated in field 156 of FIG. 5. Thus, if program X is called by a calling program, it is subject to all its own constraints as well as being combined in some way with the constraints of the calling program, which aggregate constraints are embodied into program X's PAI. In this fashion, a calling program may not be permitted to exceed its assigned bounds by merely calling another program. There are many alternative ways that a program's PAI could be combined with the PAI of the program which invokes it--depending on the strategies which are applicable to the current environment, and the inherent nature of the programs themselves. It may even be likely that even the method of combination is itself one of the PAI authorities, or qualifiers, of either or both the invoking or invoked program. For example, it is reasonable to restrict a called program to the lesser of its 'normal' PAI authority and that of its calling program--to insure the calling program cannot mischievously misuse the called program's greater authority to circumvent its own limitations.

U.S. Patent No. 6,192,476 – Claim 16	*Fischer*
	On the other hand, for called programs which carefully verify their own actions, it could be possible to allow the called program greater inherent authority than the program which calls it--this way sensitive resources could be made available to wider use by mediating such use through trusted sub-programs. The possibilities for such combination must be carefully considered, not only by the designers of the underlying control system, but also by those who assign authority to each program. Thereafter, the program is loaded and the hash of the program is computed based on the algorithm specified in the program's PAI." *Fischer* at 17:40-18:10.
in response to determining said permission required is encompassed by at least one permission associated with said second routine, then performing the steps of: A) selecting a next routine from said plurality of routines in said calling hierarchy,	*Fischer* discloses in response to determining said permission required is encompassed by at least one permission associated with said second routine, then performing the steps of: A) selecting a next routine from said plurality of routines in said calling hierarchy. *Fischer* discloses "an originating program" that "calls a program" having a program control block, or PCB. *Fischer* at 10:24-26. "Each new PCB will include a field . . . that points to the 'previous' of calling program control block." *See id.* at 10:27-29. Then, "[w]hen a called program finishes executing, the system removes its associated PCB from the top of the executed stack, removes the associated program from storage, removes the associated authorizing information and accesses the program control block immediately below it in the stack." *See id.* at 10:31-36. *Fischer* also incorporates by reference two patents which explicitly disclose data hierarchies: *see id.* at 6:37 (incorporating by reference U.S. Patent Nos. 4,868,877 and 5,005,200 which disclose an "enhanced digital signature certification" which employs "[a] hierarchy of nested certifications and signatures." *See* U.S. Patent No. 5,005,200, Abstract. "As shown in FIG. 5, an originating program (whose PCB is identified at 180) calls a program (having a PCB 170) which will, in turn, will call the program 140 is shown in detail in FIG. 5. Each new PCB will include a field such as 150 that points to the 'previous' or calling program control block. A field may also be utilized to identify the 'next' program control block file." *Fischer* at 10:23-30. "Thereafter, the program X's program authorizing information is combined, as appropriate, with the PAI associated with the PCB of the calling program, if any. This combined PAI,

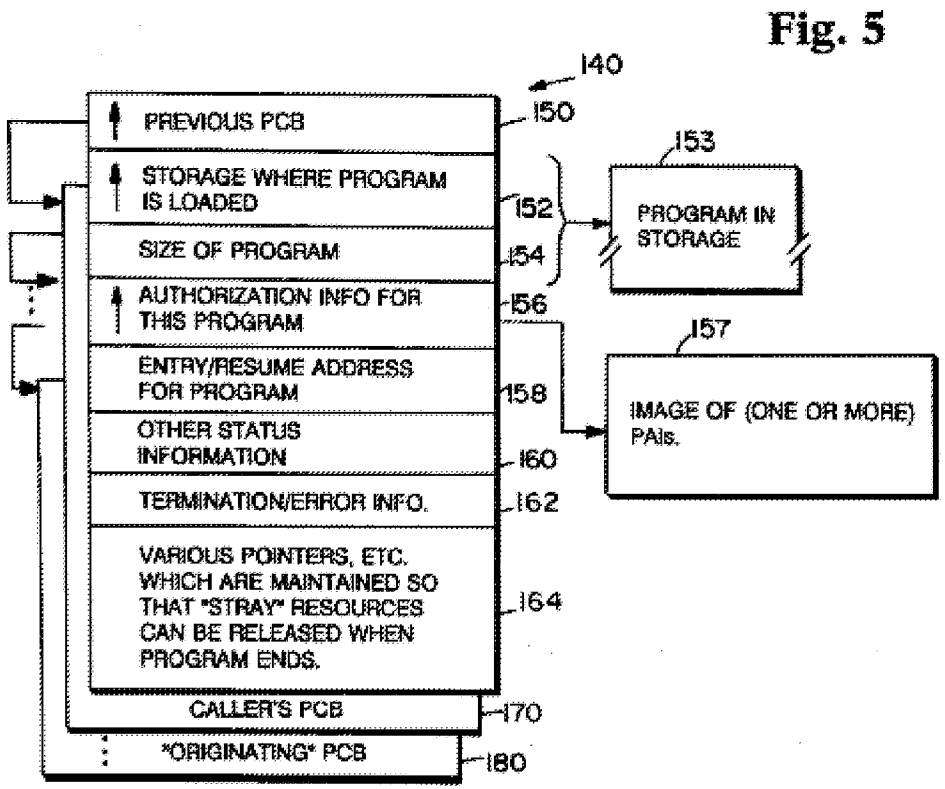
U.S. Patent No. 6,192,476 – Claim 16	*Fischer*
	which may include multiple PAI's, is then stored in an area of storage which cannot generally be modified by the program and the address of the PAI is stored in the process control block (PCB) as indicated in field 156 of FIG. 5. Thus, if program X is called by a calling program, it is subject to all its own constraints as well as being combined in some way with the constraints of the calling program, which aggregate constraints are embodied into program X's PAI. In this fashion, a calling program may not be permitted to exceed its assigned bounds by merely calling another program. There are many alternative ways that a program's PAI could be combined with the PAI of the program which invokes it--depending on the strategies which are applicable to the current environment, and the inherent nature of the programs themselves. It may even be likely that even the method of combination is itself one of the PAI authorities, or qualifiers, of either or both the invoking or invoked program. For example, it is reasonable to restrict a called program to the lesser of its 'normal' PAI authority and that of its calling program--to insure the calling program cannot mischievously misuse the called program's greater authority to circumvent its own limitations. On the other hand, for called programs which carefully verify their own actions, it could be possible to allow the called program greater inherent authority than the program which calls it--this way sensitive resources could be made available to wider use by mediating such use through trusted sub-programs. The possibilities for such combination must be carefully considered, not only by the designers of the underlying control system, but also by those who assign authority to each program. Thereafter, the program is loaded and the hash of the program is computed based on the algorithm specified in the program's PAI." *Fischer* at 17:40-18:10. "There are many alternative ways that a program's PAI could be combined with the PAI of the program which invokes it--depending on the strategies which are applicable to the current environment, and the inherent nature of the programs themselves. It may even be likely that even the method of combination is itself one of the PAI authorities, or qualifiers, of either or both the invoking or invoked program. For example, it is reasonable to restrict a called program to the lesser of its 'normal' PAI authority and that of its calling program--to insure the calling program cannot mischievously

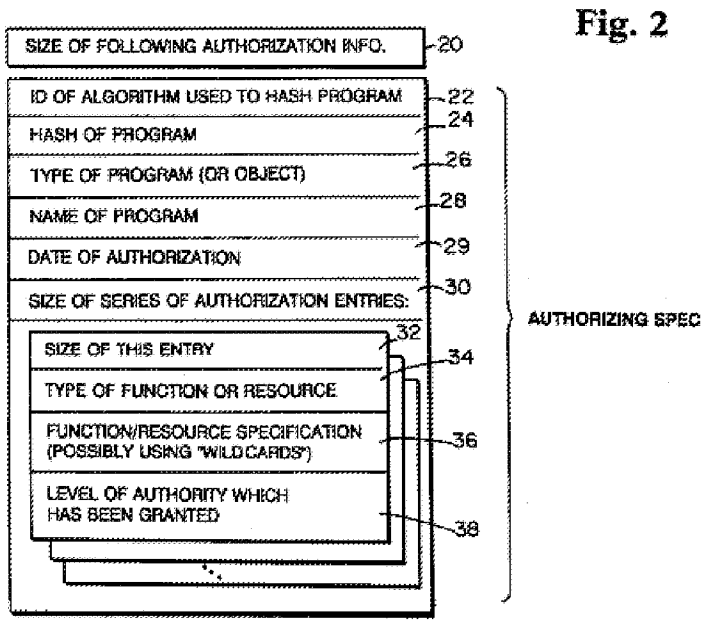
U.S. Patent No. 6,192,476 – Claim 16	*Fischer*
	misuse the called program's greater authority to circumvent its own limitations. On the other hand, for called programs which carefully verify their own actions, it could be possible to allow the called program greater inherent authority than the program which calls it--this way sensitive resources could be made available to wider use by mediating such use through trusted sub-programs. The possibilities for such combination must be carefully considered, not only by the designers of the underlying control system, but also by those who assign authority to each program." *Fischer* at 17:54-18:8.
B) if said permission required is not encompassed by at least one permission associated with said next routine, then transmitting a message indicating that said permission required is not authorized, and	*Fischer* discloses if said permission required is not encompassed by at least one permission associated with said next routine, then transmitting a message, e.g, "an appropriate error code or message," indicating that said permission required is not authorized. 

U.S. Patent No. 6,192,476 – Claim 16	*Fischer*
	 **Fig. 11** *Fischer* at Fig. 11. "If the check at 342 reveals that the PAI does not allow the attempted function or resource access, then a error message is generated in block 344 to indicate that the program is attempting to exceed its limits, access to the resource or function is denied and an appropriate error code or message is generated." *Fischer* at 19:27-33.
C) repeating steps A and B until: said permission required is not authorized by at least one permission associated with said next routine, there are no more routines to select from said plurality of routines in said calling hierarchy, or determining that said next routine is	*Fischer* discloses repeating steps A and B until: said permission required is not authorized by at least one permission associated with said next routine, there are no more routines to select from said plurality of routines in said calling hierarchy, or determining that said next routine is said first routine. *Fischer* discloses a recursive inquiry into the layered PAI associated with a given data structure or structures. It is inherent that this recursive method would determine when all structures had been analyzed.

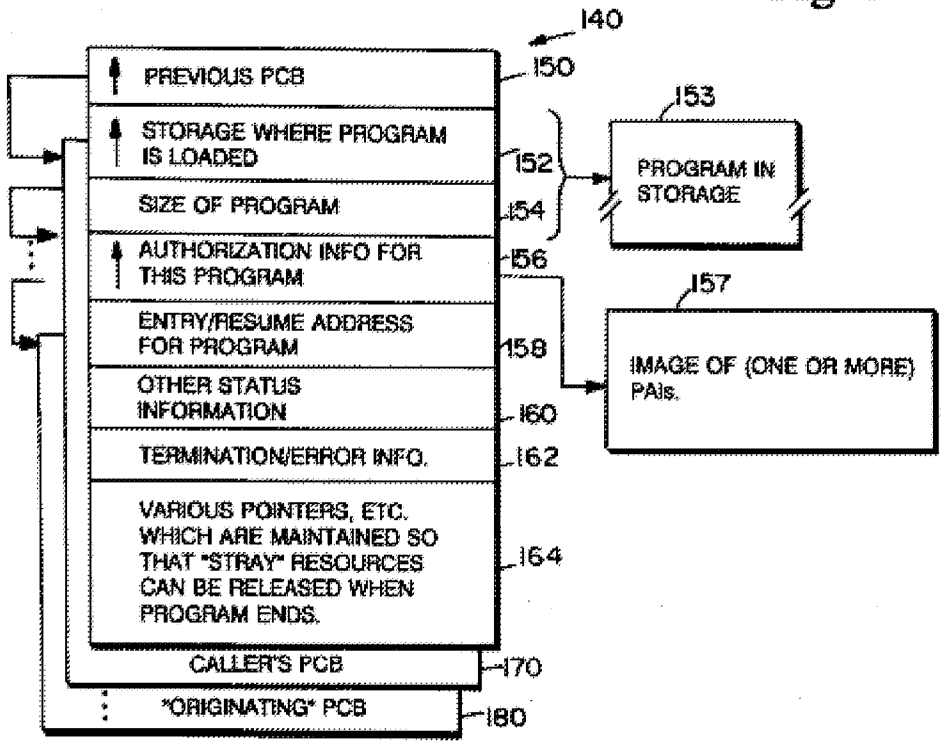
U.S. Patent No. 6,192,476 – Claim 16	*Fischer*
said first routine.	"There are many alternative ways that a program's PAI could be combined with the PAI of the program which invokes it--depending on the strategies which are applicable to the current environment, and the inherent nature of the programs themselves. It may even be likely that even the method of combination is itself one of the PAI authorities, or qualifiers, of either or both the invoking or invoked program. For example, it is reasonable to restrict a called program to the lesser of its 'normal' PAI authority and that of its calling program--to insure the calling program cannot mischievously misuse the called program's greater authority to circumvent its own limitations. On the other hand, for called programs which carefully verify their own actions, it could be possible to allow the called program greater inherent authority than the program which calls it--this way sensitive resources could be made available to wider use by mediating such use through trusted sub-programs. The possibilities for such combination must be carefully considered, not only by the designers of the underlying control system, but also by those who assign authority to each program." *Fischer* at 17:54-18:8.

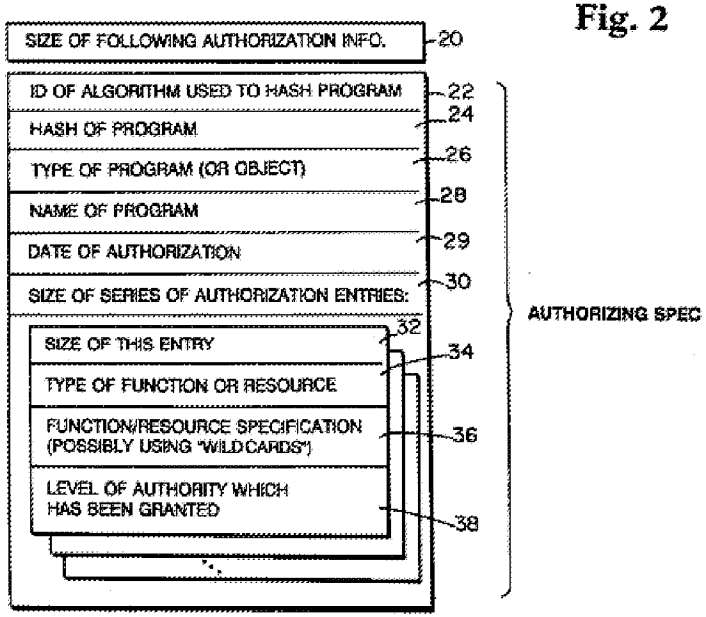
U.S. Patent No. 6,192,476 – Claim 17	*Fischer*
17. The computer readable medium of claim 16, wherein: the computer readable medium further comprises one or more instructions for performing the step of setting a flag associated with said first routine to indicate that said first routine is privileged; and	*Fischer* discloses the computer readable medium of claim 16, wherein: the computer readable medium further comprises one or more instructions for performing the step of setting a flag associated with said first routine to indicate that said first routine is privileged. "A field 156 of the program control block identifies the location in storage (157) of one or more PAI's." *Fischer* at 10:47-49. The field 156 of *Fischer* functions as would the "flag" of the instant limitation.

U.S. Patent No. 6,192,476 – Claim 17	*Fischer*
	 *Fischer* at Fig. 5. "A field 156 of the program control block identifies the location in storage (157) of one or more PAI's (which are located in an area of storage which cannot be altered by associated programs). The PAI's pointed to by field 156 are preferably structured in the manner indicated in FIG. 2 described above." *Fischer* at 10:47-52.

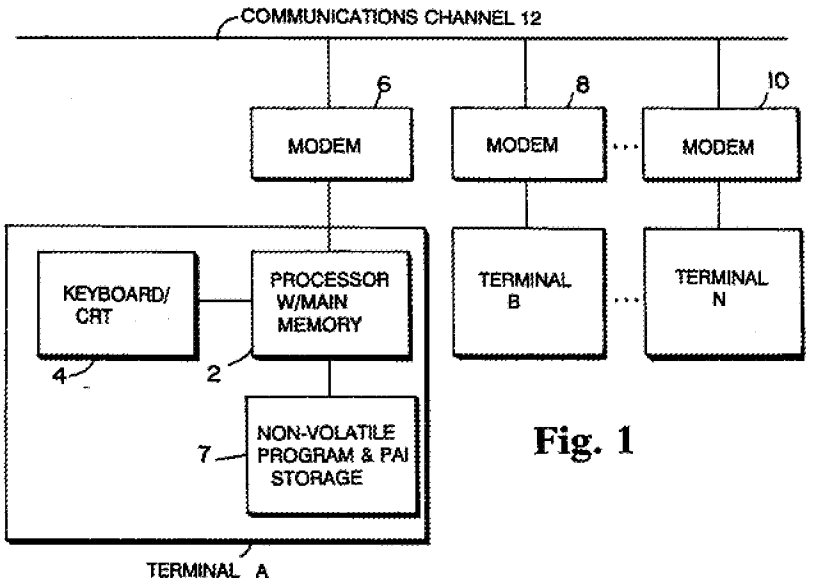
U.S. Patent No. 6,192,476 – Claim 17	*Fischer*
	 *Fischer* at Fig. 2.
the step of determining that said next routine is said first routine includes determining that a flag associated with said next routine indicates said next routine is privileged.	*Fischer* discloses the step of determining that said next routine is said first routine includes determining that a flag associated with said next routine indicates said next routine is privileged. *Fischer* broadly discloses a recursive approach and PAI associations that act as flags. Furthermore, *Fischer* discloses that slight variations on the PAI functionality are inherent in the programs themselves: "There are many alternative ways that a program's PAI could be combined with the PAI of the program which invokes it--depending on the strategies which are applicable to the current environment, and the inherent nature of the programs themselves." *Fischer* at 17:54-58.

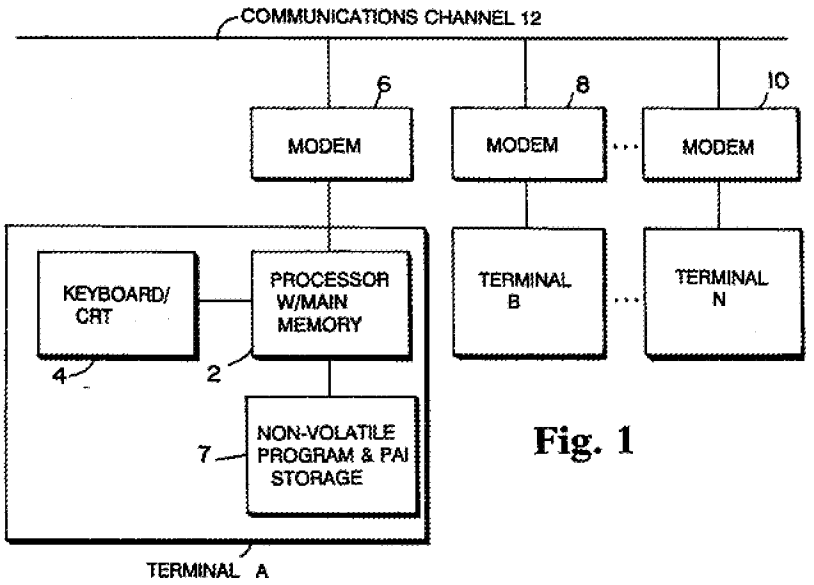
U.S. Patent No. 6,192,476 – Claim 18	
18. The computer readable medium of	*Fischer* discloses computer readable medium of claim 17, wherein the step of setting said

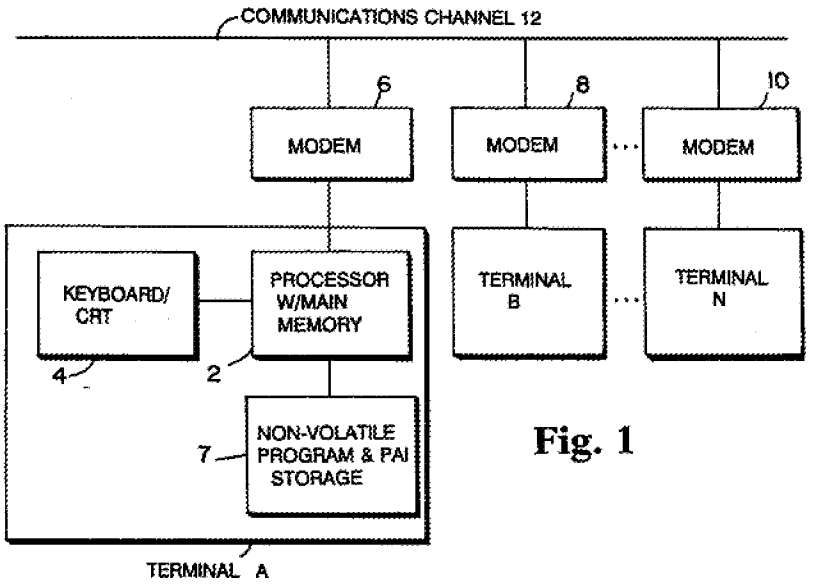
U.S. Patent No. 6,192,476 – Claim 18	
claim 17, wherein the step of setting said flag associated with said first routine includes setting a flag in a frame in said calling hierarchy associated with said thread.	flag associated with said first routine includes setting a flag in a frame in said calling hierarchy associated with said thread. "A field 156 of the program control block identifies the location in storage (157) of one or more PAI's." *Fischer* at 10:47-49. The field 156 of *Fischer* functions as would the "flag" of the instant limitation. Fig. 5  *Fischer* at Fig. 5. "A field 156 of the program control block identifies the location in storage (157) of one or more PAI's (which are located in an area of storage which cannot be altered by associated programs). The PAI's pointed to by field 156 are preferably structured in the manner

U.S. Patent No. 6,192,476 – Claim 18	
	indicated in FIG. 2 described above." *Fischer* at 10:47-52. **Fig. 2**  *Fischer* at Fig. 2.

U.S. Patent No. 6,192,476 – Claim 19	*Fischer*
19. A computer system comprising:	*Fischer* discloses a computer system, e.g., an "IBM PC[] having a processor." "FIG. 1 shows in block diagram form an exemplary communications system which may be used in conjunction with the present invention. . . . Terminals, A, B . . . N may, by way of example only, be IBM PC's having a processor (with main memory) 2 which is coupled to a conventional keyboard/CRT display 4. Additionally, each processor is preferably coupled to a non-volatile program and program authorization information (PAI) storage 7 which may be a disk memory device."

U.S. Patent No. 6,192,476 – Claim 19	*Fischer*
	 **Fig. 1** *Fischer* at 4:45-58 & Fig. 1.
a processor;	*Fischer* discloses a processor, i.e., an "IBM PC[] having a processor." "FIG. 1 shows in block diagram form an exemplary communications system which may be used in conjunction with the present invention. . . . Terminals, A, B . . . N may, by way of example only, be IBM PC's having a processor (with main memory) 2 which is coupled to a conventional keyboard/CRT display 4. Additionally, each processor is preferably coupled to a non-volatile program and program authorization information (PAI) storage 7 which may be a disk memory device."

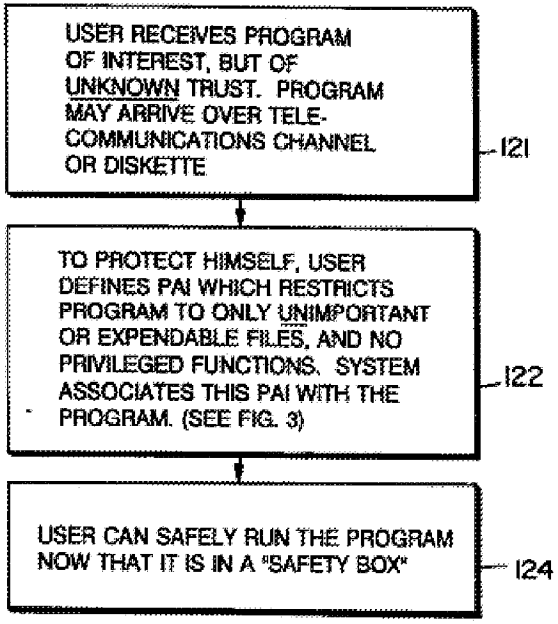
U.S. Patent No. 6,192,476 – Claim 19	*Fischer*
	 **Fig. 1** *Fischer* at 4:45-58 & Fig. 1.
a memory coupled to said processor;	*Fischer* discloses a memory, e.g., "a disk memory device," coupled to said processor. "FIG. 1 shows in block diagram form an exemplary communications system which may be used in conjunction with the present invention. . . . Terminals, A, B . . . N may, by way of example only, be IBM PC's having a processor (with main memory) 2 which is coupled to a conventional keyboard/CRT display 4. Additionally, each processor is preferably coupled to a non-volatile program and program authorization information (PAI) storage 7 which may be a disk memory device."

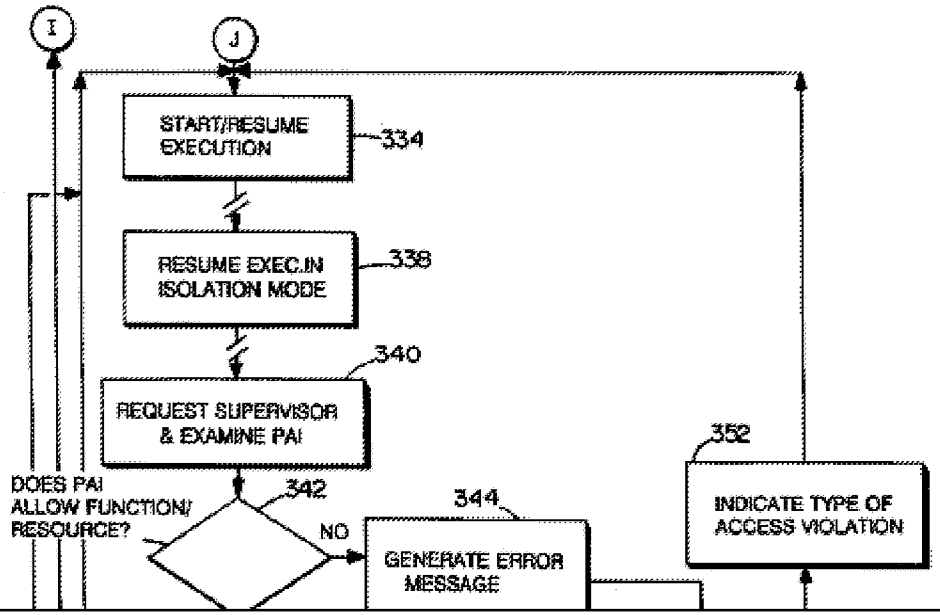
U.S. Patent No. 6,192,476 – Claim 19	*Fischer*
	 **Fig. 1** *Fischer* at 4:45-58 & Fig. 1.
said processor being configured to detect when a request for an action is made by a principal; and	*Fischer* discloses said processor being configured to detect when a request for an action is made by a principal. *Fischer* inquires into the permissions of a given principal's PAI upon the principal's request for an action: "When the program is to perform a function or access a resource, the associated PAI is monitored to confirm that the operation is within the defined program limits. If the program attempts to do anything outside the authorized limits, then the program execution is halted." *Id.* at 2:43-48. Thus it inherent that *Fischer* detects a request of an action made by a principal. "Prior art operating systems are typically designed to protect data from computer users. In such systems, users are often assigned various authorities and are thereafter able to execute programs based on their associated authority. If a program is executing which exceeds the user's assigned authority, then such a system will halt execution of the program. Such prior art systems do not adequately protect computer users from computer viruses or the like.

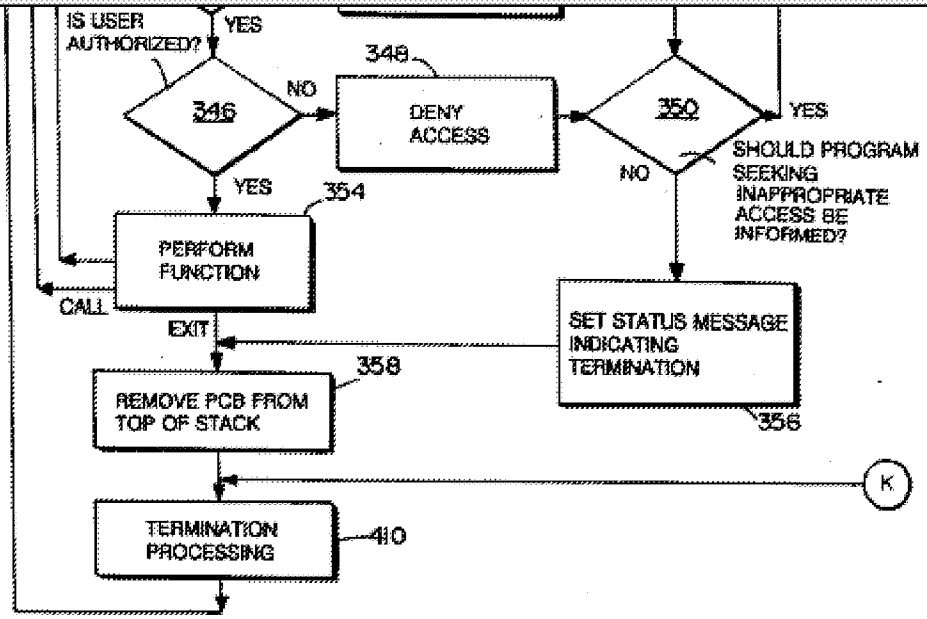
U.S. Patent No. 6,192,476 – Claim 19	*Fischer*
	There are security systems which protect certain 'system' related files from being modified by a program. However, such systems do not typically protect a computer user from a program executing and modifying the user's own files. The present invention is directed to providing reliable security, even when operating with complex data structures, e.g., objects, containing their own program instructions, which are transmitted among users. The present invention also provides enhanced security when processing more conventional programs, even those of questionable origin, e.g., from a computer bulletin board, without exposing system programs or data to the potentially catastrophic consequences of computer viruses or of incompetent programming. The present method and apparatus utilizes a unique operating system design that includes a system monitor which limits the ability of a program about to be executed to the use of predefined resources (e.g., data files, disk writing capabilities etc.). The system monitor builds a data structure including a set of authorities defining that which a program is permitted to do and/or that which the program is precluded from doing. The set of authorities and/or restrictions assigned to a program to be executed are referred to herein as 'program authorization information' (or 'PAI'). Once defined, the program authorization information is thereafter associated with each program to be executed to thereby delineate the types of resources and functions that the program is allowed to utilize. The PAI associated with a particular program may be assigned by a computer system owner/user or by someone who the computer system owner/user implicitly trusts. The PAI defines the range of operations that a program may execute and/or defines those operations that a program cannot perform. The program is permitted to access what has been authorized and nothing else. In this fashion, the program may be regarded as being placed in a program capability limiting 'safety box'. This 'safety box' is thereafter associated with the program such that whenever the system monitor runs the program, the PAI for that program is likewise loaded and monitored. When the program is to perform a function or access a resource, the associated PAI is monitored to confirm that the operation is within the defined program limits. If the program attempts to do anything outside the authorized limits, then the

U.S. Patent No. 6,192,476 – Claim 19	*Fischer*
	program execution is halted. Thus, the present invention advantageously protects a user from any program to be executed. The present invention is particularly advantageous in light of current data processing practices where programs are obtained from a wide range of diverse, untrustworthy places such as computer bulletin boards or other users of unknown trustworthiness." *Fischer* at 1:60-2:55. "FIG. 4 is a flowchart which illustrates how a user may benefit from the use of program authorization information, particularly when executing a program of unknown trustworthiness. As indicated in block 121, a user may have a desire to execute a program of interest in which the user has no knowledge of the program's creator. Thus, the program has unknown trustworthiness and may, for example, have been accessed via an electronic bulletin board and may have arrived at the user's terminal via a telecommunications channel or diskette. Such a program, which might purport to be only a game, carries with it a significant risk that it may be infected by a virus. As indicated in block 122, the user may be protected by defining program authorization information which restricts the program to only unimportant or expendable files. If desired, the user may restrict such a program from modifying any files whatsoever. For example, the user may permit the program to only display images on the display screen and to perform game playing related functions. Alternatively, if the program is known to have a single work file, the PAI data may only permit use of such a single file. By limiting access only to a single work file, a program of unknown trustworthiness, cannot inject a virus into other user's programs or otherwise initiate system program malfunctions. Thus, in accordance with the present invention, a user, via a systems program, determines how much of the user's system will be put at risk by such a program so as to, for example, completely eliminate the ability of the program to use any privileged functions. The user then associates, for example, through an operator prompting, menu-driven system, a PAI with every program to be run on the system (or have such PAI or lack of PAI associated through predetermined default mechanisms). A system utility program is preferably employed to create the program authorization information in a manner which will be described in detail below in conjunction

U.S. Patent No. 6,192,476 – Claim 19	*Fischer*
	with FIGS. 6-9. After the PAI has been assigned, any time the system runs the associated program, the system software (in a manner to be described below) insures that the program safely runs in a manner consistent with the PAI. Thus, the program has been effectively placed in a 'safety box' (124). Turning back to FIG. 1, a program of unknown trust may be injected into the system via communications channel 12 or from a floppy disk loaded into terminal A. The program may be initially stored in, for example, the user's program disk memory 7. Thereafter, the user on keyboard 4 will, through interaction with the system's program identified above (with respect to block 122 of FIG. 4), associate the program authorization information with a program (in a manner such as shown in FIGS. 3A through 3D) such that the program may safely run on the user's system or perhaps, a PAI arrives with the program, in which case it is likely to be signed. FIG. 5 is an illustration of a program control block (PCB) data structure 140 in accordance with an exemplary embodiment of the present invention. The program control block 140 is the data structure utilized by the system monitor to control the execution of an associated program. The program control block 140 is loaded with program authorization information such that the PAI can be readily referenced as the associated program is executed so as to insure that the program performs functions and accesses resources in conformance with its assigned authorizations. The program control block associated with the program to be executed is located in a storage area which cannot be modified by the program." *Fischer* at 9:17-10:23.

U.S. Patent No. 6,192,476 – Claim 19	*Fischer*
	 **Fig. 4** *Fischer* at Fig. 4. "FIGS. 10 and 11 illustrate the sequence of operations of a supervisor program for controlling the processing of a program being executed in accordance with its program authorization information. The processing of a program 'X' and its program authorization information illustrated in FIG. 10 is initiated while the computer is executing a supervisor routine. As shown in FIG. 10 at 300, a calling program calls program X for execution. Thereafter, a program control block is created for program X. The program control block created will not be added to the top of the execution stack until it is determined that the program is permitted to be invoked and verification is successful completed. Thus, if the program fails a security check, it will not be placed in the program execution chain. In addition to creating a 'tentative' program control block, the called program will be located through an appropriate program directory during the processing in block 302. Thereafter, a check is made at block 304 to determine whether PAI has yet been associated

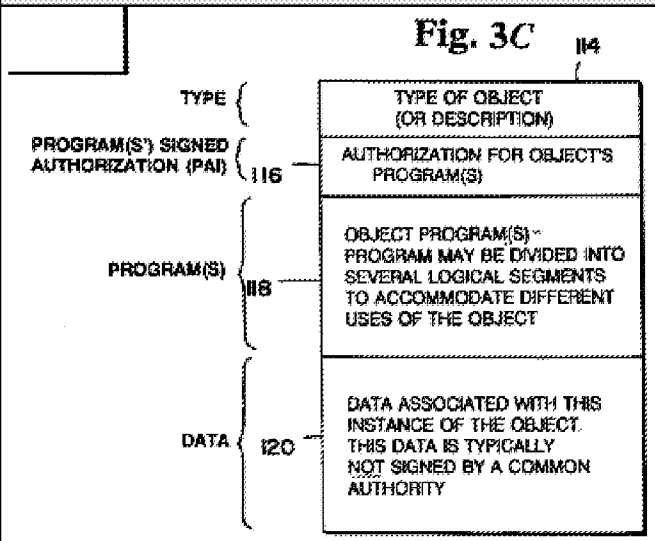
U.S. Patent No. 6,192,476 – Claim 19	*Fischer*
	with program X so as to place program X in the so-called 'safety box' described above. This PAI may or may not be signed depending upon its particular application as described above." *Fischer* at 15:56-16:11. 

U.S. Patent No. 6,192,476 – Claim 19	*Fischer*
	 **Fig. 11** *Fischer* at Fig. 11.
said processor being configured to respond to detecting the request by determining whether said action is authorized based on permissions associated with a plurality of routines in a calling hierarchy associated with said principal, wherein said permissions are associated with said plurality of routines based on a first association between protection domains and permissions.	*Fischer* discloses said processor being configured to respond to detecting the request by determining whether said action is authorized based on permissions associated with a plurality of routines in a calling hierarchy associated with said principal, wherein said permissions are associated with said plurality of routines based on a first association between protection domains and permissions. *Fischer* discloses "an originating program" that "calls a program" having a program control block, or PCB. *Fischer* at 10:24-26. "Each new PCB will include a field . . . that points to the 'previous' of calling program control block." *See id.* at 10:27-29. Then, "[w]hen a called program finishes executing, the system removes its associated PCB from the top of the executed stack, removes the associated program from storage, removes the associated authorizing information and accesses the program control block immediately below it in the stack." *See id.* at 10:31-36. *Fischer* also incorporates by reference two patents which explicitly disclose data hierarchies:

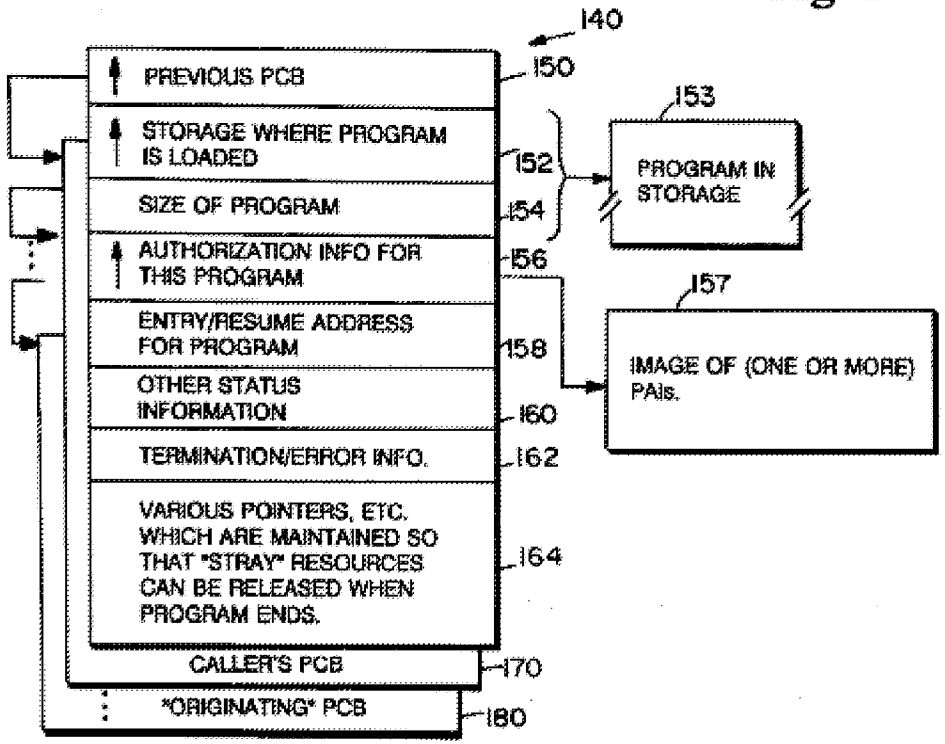
U.S. Patent No. 6,192,476 – Claim 19	*Fischer*
	see id. at 6:37 (incorporating by reference U.S. Patent Nos. 4,868,877 and 5,005,200 which disclose an "enhanced digital signature certification" which employs "[a] hierarchy of nested certifications and signatures." *See* U.S. Patent No. 5,005,200, Abstract. "The present method an apparatus utilizes a unique operation system design that includes a system monitor which limits the ability of a program about to be executed to the use of predefined resources (e.g., data files, disk writing capabilities etc.). The system monitor builds a data structure including a set of authorities defining that which a program is permitted to do and/or that which the program is precluded from doing. The set of authorities and/or restrictions assigned to a program to be executed are referred to herein as 'program authorization information' (or 'PAI'). Once defined, the program authorization information is thereafter associated with each program to be executed to thereby delineate the types of resources and functions that the program is allowed to utilize. The PAI associated with a particular program may be assigned by a computer system owner/user or by someone who the computer system owner/user implicitly trusts. The PAI defines the range of operations that a program may execute and/or defines those operations that a program cannot perform. The program is permitted to access what has been authorized and nothing else. In this fashion, the program may be regarded as being placed in a program capability limiting 'safety box.' This 'safety box' is thereafter associated with the program such that whenever the system monitor runs the program, the PAI for that program is likewise loaded and monitored. When the program is to perform a function or access a resource, the associated PAI is monitored to confirm that the operation is within the defined program limits. If the program attempts to do anything outside the authorized limits, then the program execution is halted." *Fischer* at 2:16-48. "Thereafter, the program X's program authorizing information is combined, as appropriate, with the PAI associated with the PCB of the calling program, if any. This combined PAI, which may include multiple PAI's, is then stored in an area of storage which cannot generally be modified by the program and the address of the PAI is stored in the process control block (PCB) as indicated in field 156 of FIG. 5. Thus, if program X is called by a

U.S. Patent No. 6,192,476 – Claim 19	*Fischer*
	calling program, it is subject to all its own constraints as well as being combined in some way with the constraints of the calling program, which aggregate constraints are embodied into program X's PAI. In this fashion, a calling program may not be permitted to exceed its assigned bounds by merely calling another program." *Fischer* at 19:40-54. "Even programs with no known trustworthiness can be used after program authorization information associates a wide range of restrictions to thereby allow potentially beneficial programs to be safely used--even if they do not have an official certification of trust. The present invention also allows an unlimited number of different resources and functions to be controlled. For example, some useful resources/functions which may be controlled include: the ability to limit a program to certain files or data sets; the ability to transmit data via electronic mail to someone outside the user's domain; the ability of a program to create or solicit digital signatures; the ability to limit access to a program of certain security classes, etc." *Fischer* at 3:48-61. "The present invention is directed to providing reliable security, even when operating with complex data structures, e.g., objects, containing their own program instructions, which are transmitted among users." *Fischer* at 2:6-9. "Through the use of the present invention, general object oriented data may be transferred from user to user without exposing users to the potential dangers of viruses or mischievous users." *Fischer* at 4:10-13. "In one contemplated embodiment of the present invention, programs may be part of data objects, which are written in a high-level control language and are executed by a standardized interpreter program which executes this high-level language. In this case, part of the interpreter's task is to verify that the functions encountered in the high level logic are, in fact, permissible. If such tasks are not permissible, the interpreter then suppresses the execution of the program not authorized to perform such tasks." *Fischer* at 3:11-20. "In accordance with the present invention, a PAI is associated with programs to be executed.

U.S. Patent No. 6,192,476 – Claim 19	*Fischer*
	FIGS. 3A through 3D depict four exemplary approaches for associating program authorization information with a program. . . . . . . FIG. 3C shows an important application in which a PAI data structure is associated with a program according to an embodiment of the present invention. FIG. 3C shows an illustrative data structure for a secure exchangeable 'object'. The data structure may be signed by a trusted authority. The signing of such a data structure allows the object to be securely transmitted from user to user. Although the data structure shown in FIG. 3 is set forth in a general format, it may be structured as set forth in the inventor's copending application filed on Apr. 6, 1992 and entitled 'Method and Apparatus for Creating, Supporting and Processing a Travelling Program' (U.S. Ser. No. 07/863,552.), which application is hereby expressly incorporated herein by reference. . . . The program authorization information is embedded in a segment 116 which specifies the authorization for the object's program or programs in a manner to be described more fully hereinafter." *Fischer* at 7:14-18, 7:49-8:2. "Thereafter, the PAI is stored using, for example, one of the approaches set forth in FIGS. 3A through 3D so that it is associated with its program 272 . . . ." *Fischer* at 15:24-26.

U.S. Patent No. 6,192,476 – Claim 19	*Fischer*
	 *Fischer* at Fig. 3C. "FIGS. 10 and 11 illustrate the sequence of operations of a supervisor program for controlling the processing of a program being executed in accordance with its program authorization information." *Fischer* at 15:56-59. "Depending on the processing in block 316, a decision is made in block 322 whether the signatures are valid, authorized and trusted. If the signatures are not determined to be valid, then the routing branches to block 324 where the execution in program X is suppressed." *Fischer* at 16:66-17:3. "If the processing in blocks 322 and 316 reveal that the signatures are valid, then the processing in block 326 is performed." *Fischer* at 17:31-33. "FIG. 5 is an illustration of a program control block (PCB) data structure 140 in accordance with an exemplary embodiment of the present invention. The program control block 140 is the data structure utilized by the system monitor to control the execution of an associated

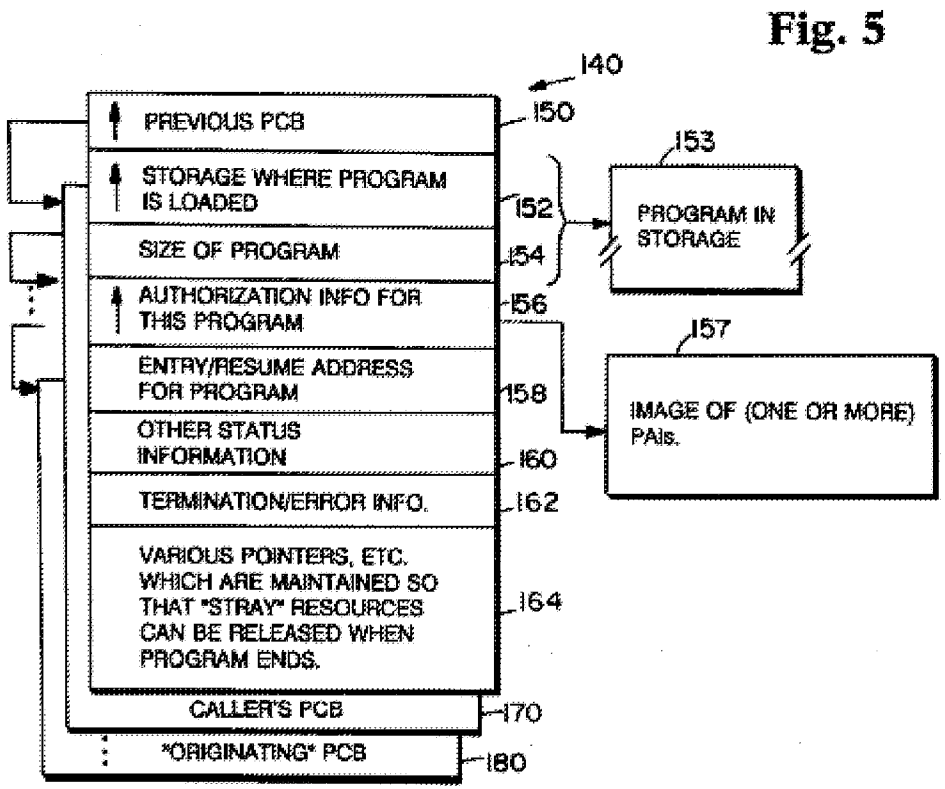
U.S. Patent No. 6,192,476 – Claim 19	*Fischer*
	program. The program control block 140 is loaded with program authorization information such that the PAI can be readily referenced as the associated program is executed so as to insure that the program performs functions and accesses resources in conformance with its assigned authorizations. The program control block associated with the program to be executed is located in a storage area which cannot be modified by the program. As shown in FIG. 5, an originating program (whose PCB is identified at 180) calls a program (having a PCB 170) which will, in turn, will call the program 140 is shown in detail in FIG. 5. Each new PCB will include a field such as 150 that points to the 'previous' or calling program control block. A field may also be utilized to identify the 'next' program control block file. When a called program finishes executing, the system removes its associated PCB from the top of the executed stack, removes the associated program from storage, removes the associated authorizing information and accesses the program control block immediately below it in the stack. When another program is called, the reverse process occurs such that a new PCB is created which is placed on top of the stack, which again points to the previous PCB as indicated in field 150." *Fischer* at 10:8-39.

U.S. Patent No. 6,192,476 – Claim 19	*Fischer*
	Fig. 5  *Fischer* at Fig. 5. "Thereafter, the program X's program authorizing information is combined, as appropriate, with the PAI associated with the PCB of the calling program, if any. This combined PAI, which may include multiple PAI's, is then stored in an area of storage which cannot generally be modified by the program and the address of the PAI is stored in the process control block (PCB) as indicated in field 156 of FIG. 5. Thus, if program X is called by a calling program, it is subject to all its own constraints as well as being combined in some way with the constraints of the calling program, which aggregate constraints are embodied into program X's PAI. In this fashion, a calling program may not be permitted to exceed its

U.S. Patent No. 6,192,476 – Claim 19	*Fischer*
	assigned bounds by merely calling another program. There are many alternative ways that a program's PAI could be combined with the PAI of the program which invokes it--depending on the strategies which are applicable to the current environment, and the inherent nature of the programs themselves. It may even be likely that even the method of combination is itself one of the PAI authorities, or qualifiers, of either or both the invoking or invoked program. For example, it is reasonable to restrict a called program to the lesser of its 'normal' PAI authority and that of its calling program--to insure the calling program cannot mischievously misuse the called program's greater authority to circumvent its own limitations. On the other hand, for called programs which carefully verify their own actions, it could be possible to allow the called program greater inherent authority than the program which calls it--this way sensitive resources could be made available to wider use by mediating such use through trusted sub-programs. The possibilities for such combination must be carefully considered, not only by the designers of the underlying control system, but also by those who assign authority to each program. Thereafter, the program is loaded and the hash of the program is computed based on the algorithm specified in the program's PAI." *Fischer* at 17:40-18:10.

U.S. Patent No. 6,192,476 – Claim 20	*Fischer*
20. The computer system of claim 19, wherein: the calling hierarchy includes a first routine; and	*Fischer* discloses the computer system of claim 19, wherein: the calling hierarchy includes a first routine. For example, *Fischer* discloses calling a plurality of routines: "As shown in FIG. 5, an originating program (whose PCB is identified at 180) calls a program (having a PCB 170) which will, in turn, will call the program 140 is shown in detail in FIG. 5." *Fischer* at 10:24-27. "FIGS. 10 and 11 illustrate the sequence of operations of a supervisor program for controlling the processing of a program being executed in accordance with its program authorization information. The processing of a program 'X' and its program authorization information illustrated in FIG. 10 is initiated while the computer is executing a supervisor routine." *Fischer* at 15:56-62. "FIG. 5 is an illustration of a program control block (PCB) data structure 140 in accordance

U.S. Patent No. 6,192,476 – Claim 20	*Fischer*
	with an exemplary embodiment of the present invention. The program control block 140 is the data structure utilized by the system monitor to control the execution of an associated program. The program control block 140 is loaded with program authorization information such that the PAI can be readily referenced as the associated program is executed so as to insure that the program performs functions and accesses resources in conformance with its assigned authorizations. The program control block associated with the program to be executed is located in a storage area which cannot be modified by the program. As shown in FIG. 5, an originating program (whose PCB is identified at 180) calls a program (having a PCB 170) which will, in turn, will call the program 140 is shown in detail in FIG. 5. Each new PCB will include a field such as 150 that points to the 'previous' or calling program control block. A field may also be utilized to identify the 'next' program control block file. When a called program finishes executing, the system removes its associated PCB from the top of the executed stack, removes the associated program from storage, removes the associated authorizing information and accesses the program control block immediately below it in the stack. When another program is called, the reverse process occurs such that a new PCB is created which is placed on top of the stack, which again points to the previous PCB as indicated in field 150." *Fischer* at 10:8-39.

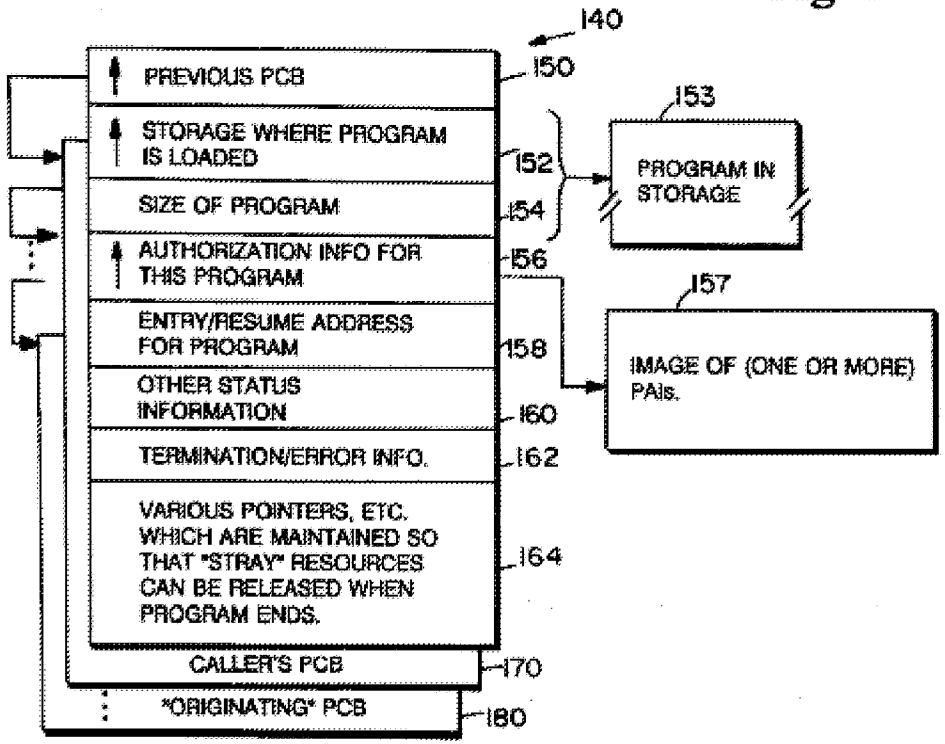
U.S. Patent No. 6,192,476 – Claim 20	*Fischer*
	 *Fischer* at Fig. 5. "FIG. 3C shows an important application in which a PAI data structure is associated with a program according to an embodiment of the present invention. FIG. 3C shows an illustrative data structure for a secure exchangeable 'object'. The data structure may be signed by a trusted authority. The signing of such a data structure allows the object to be securely transmitted from user to user. Although the data structure shown in FIG. 3 is set forth in a general format, it may be structured as set forth in the inventor's copending application filed on Apr. 6, 1992 and entitled 'Method and Apparatus for Creating, Supporting and Processing a Travelling Program' (U.S. Ser. No. 07/863,552.), which application is hereby

U.S. Patent No. 6,192,476 – Claim 20	*Fischer*
	expressly incorporated herein by reference. The data structure includes a header segment 114 which, by way of example only, may define the type of object that follows, e.g., a purchase order related object or any other type of electronic digital object. The program authorization information is embedded in a segment 116 which specifies the authorization for the object's program or programs in a manner to be described more fully hereinafter. The data structure includes an object program(s) segment 118, which for example, may control the manner in which an associated purchase order is displayed so as to leave blanks for variable fields which are interactively completed by the program user. The object program might store such data and send a copy of itself together with accompanying data in a manner which is described in detail in the applicant's above-identified copending application. As indicated in FIG. 3C, the program may be divided into several logical segments to accommodate different uses of the object. For example, the program may present a different display to the creator of a digital purchase order, than it displays to subsequent recipients. When the program is received by a recipient designated by the program, the recipient invokes a copy of the transmitted program to, for example, control the display of the purchase order tailored to the needs of the recipient." *Fischer* at 7:49-8:20.
said processor is configured to determine whether said action is authorized by determining whether a permission required to perform said action is encompassed by at least one permission associated with said first routine.	*Fischer* discloses said processor is configured to determine whether said action is authorized by determining whether a permission required to perform said action is encompassed by at least one permission associated with said first routine. This is the function of the PAI as disclosed by *Fischer*. For example: "Thereafter, a check is made at block 304 to determine whether PAI has yet been associated with program X so as to place program X in the so-called 'safety box' described above. This PAI may or may not be signed depending upon its particular application as described above." *Fischer* at 15:56-16:11. "FIG. 3C shows an important application in which a PAI data structure is associated with a program according to an embodiment of the present invention. FIG. 3C shows an illustrative data structure for a secure exchangeable 'object'. The data structure may be signed by a trusted authority. The signing of such a data structure allows the object to be securely

U.S. Patent No. 6,192,476 – Claim 20	*Fischer*
	transmitted from user to user. Although the data structure shown in FIG. 3 is set forth in a general format, it may be structured as set forth in the inventor's copending application filed on Apr. 6, 1992 and entitled 'Method and Apparatus for Creating, Supporting and Processing a Travelling Program' (U.S. Ser. No. 07/863,552.), which application is hereby expressly incorporated herein by reference. The data structure includes a header segment 114 which, by way of example only, may define the type of object that follows, e.g., a purchase order related object or any other type of electronic digital object. The program authorization information is embedded in a segment 116 which specifies the authorization for the object's program or programs in a manner to be described more fully hereinafter." *Fischer* at 7:49-8:2. "FIGS. 10 and 11 illustrate the sequence of operations of a supervisor program for controlling the processing of a program being executed in accordance with its program authorization information. The processing of a program 'X' and its program authorization information illustrated in FIG. 10 is initiated while the computer is executing a supervisor routine. As shown in FIG. 10 at 300, a calling program calls program X for execution. Thereafter, a program control block is created for program X. The program control block created will not be added to the top of the execution stack until it is determined that the program is permitted to be invoked and verification is successful completed. Thus, if the program fails a security check, it will not be placed in the program execution chain. In addition to creating a 'tentative' program control block, the called program will be located through an appropriate program directory during the processing in block 302.

U.S. Patent No. 6,192,476 – Claim 21	*Fischer*
21. The computer system of claim 19, wherein said processor is configured to determine whether said action is authorized by determining whether a permission required to perform said action is encompassed by at least one permission associated with each routine	*Fischer* discloses the computer system of claim 19, wherein said processor is configured to determine whether said action is authorized by determining whether a permission required to perform said action is encompassed by at least one permission associated with each routine in said calling hierarchy. For example, *Fischer* discloses "an originating program" that "calls a program" having a program control block, or PCB. *Fischer* at 10:24-26. "Each new PCB will include a field . . . that points to the 'previous' of calling program control block." *See id.* at 10:27-29. Then, "[w]hen a called program finishes executing, the system

U.S. Patent No. 6,192,476 – Claim 21	*Fischer*
in said calling hierarchy.	removes its associated PCB from the top of the executed stack, removes the associated program from storage, removes the associated authorizing information and accesses the program control block immediately below it in the stack." *See id.* at 10:31-36. In this disclosure, the "program control block" sets the permission level for a routine in a calling hierarchy. "As shown in FIG. 5, an originating program (whose PCB is identified at 180) calls a program (having a PCB 170) which will, in turn, will call the program 140 is shown in detail in FIG. 5. Each new PCB will include a field such as 150 that points to the 'previous' or calling program control block. A field may also be utilized to identify the 'next' program control block file. When a called program finishes executing, the system removes its associated PCB from the top of the executed stack, removes the associated program from storage, removes the associated authorizing information and accesses the program control block immediately below it in the stack. When another program is called, the reverse process occurs such that a new PCB is created which is placed on top of the stack, which again points to the previous PCB as indicated in field 150." *Fischer* at 10:23-39.

U.S. Patent No. 6,192,476 – Claim 21	*Fischer*
	Fig. 5  *Fischer* at Fig. 5. "Thereafter, the program X's program authorizing information is combined, as appropriate, with the PAI associated with the PCB of the calling program, if any. This combined PAI, which may include multiple PAI's, is then stored in an area of storage which cannot generally be modified by the program and the address of the PAI is stored in the process control block (PCB) as indicated in field 156 of FIG. 5. Thus, if program X is called by a calling program, it is subject to all its own constraints as well as being combined in some way with the constraints of the calling program, which aggregate constraints are embodied into program X's PAI. In this fashion, a calling program may not be permitted to exceed its

U.S. Patent No. 6,192,476 – Claim 21	*Fischer*
	assigned bounds by merely calling another program. There are many alternative ways that a program's PAI could be combined with the PAI of the program which invokes it--depending on the strategies which are applicable to the current environment, and the inherent nature of the programs themselves. It may even be likely that even the method of combination is itself one of the PAI authorities, or qualifiers, of either or both the invoking or invoked program. For example, it is reasonable to restrict a called program to the lesser of its 'normal' PAI authority and that of its calling program--to insure the calling program cannot mischievously misuse the called program's greater authority to circumvent its own limitations. On the other hand, for called programs which carefully verify their own actions, it could be possible to allow the called program greater inherent authority than the program which calls it--this way sensitive resources could be made available to wider use by mediating such use through trusted sub-programs. The possibilities for such combination must be carefully considered, not only by the designers of the underlying control system, but also by those who assign authority to each program. Thereafter, the program is loaded and the hash of the program is computed based on the algorithm specified in the program's PAI." *Fischer* at 17:40-18:10.