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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
2 November 2006 (02.11.2006)

PCT

(10) International Publication Number
WO 2006/115641 A2

(51) International Patent Classification:
G06F 7/00 (2006.01)

(21) International Application Number:
PCT/US2006/010345

(22) International Filing Date: 22 March 2006 (22.03.2006)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
11/111,882 22 April 2005 (22.04.2005) US

(71) Applicant *(for all designated States except US)*: **MICROSOFT CORPORATION** [US/US]; One Microsoft Way, Redmond, Washington 98052-6399 (US).

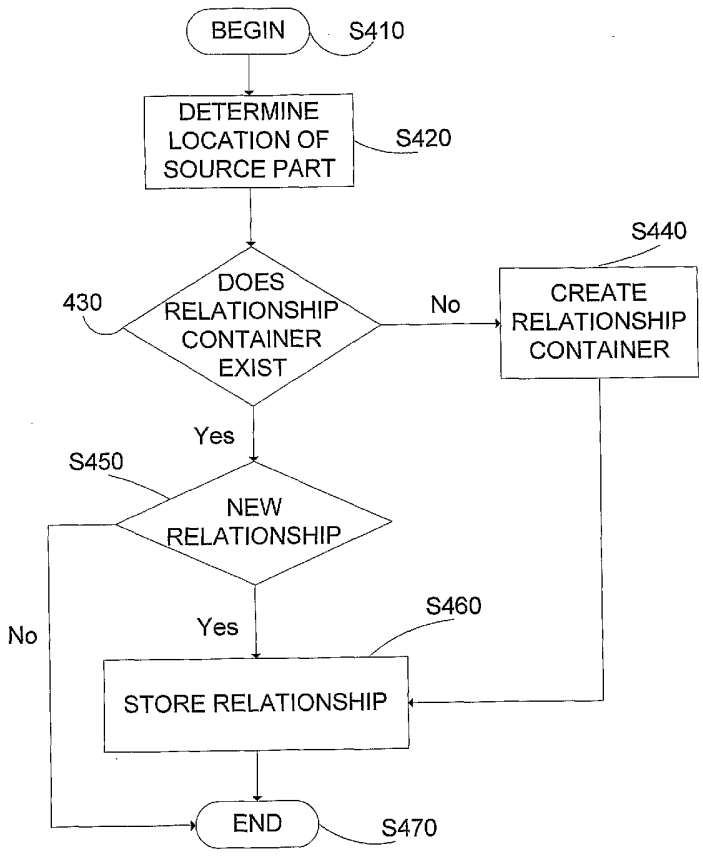
(72) Inventors: **SHUR, Andrey**; One Microsoft Way, Redmond, Washington 98052-6399 (US). **MACKENZIE, Bruce A.**; One Microsoft Way, Redmond, Washington 98052-6399 (US). **WALKER, Charles S.**; One Microsoft Way, Redmond, Washington 98052-6399 (US). **ORNSTEIN, David B.**; One Microsoft Way, Redmond, Washington 98052-6399 (US). **DUNIETZ, Jerry J.**; One Microsoft Way, Redmond, Washington 98052-6399 (US). **POLLOCK, Joshua M.**; One Microsoft Way, Redmond, Washington 98052-6399 (US). **SHETH, Sarjana B.**; One Microsoft Way, Redmond, Washington 98052-6399 (US). **NICHOLS, Isaac E.**; One Microsoft Way, Redmond, Washington 98052-6399 (US).

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

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM),

[Continued on next page]

(54) Title: EFFICIENTLY DESCRIBING RELATIONSHIPS BETWEEN RESOURCES



(57) Abstract: A relationship data structure associated with a source resource enables methods to discover and describe relationships between the source resource and a plurality of target resources. The relationships are stored in a format independent of the encoding of the source resource. Each relationship between the source resource and the plurality of target resources is stored in a content-neutral format, and the relationship data structure stores, a location of each target resource, a type of relationship with each target resource and an identifier to uniquely identify each relationship between the source resource and each target resource. Accordingly, the relationship data structure allows a decoder to directly discover the relationships between the source resource and the plurality of target resources without decoding the source resource or target resources.

WO 2006/115641 A2

European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IS, IT, LT, LU, LV, MC, NL, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Declarations under Rule 4.17:

— *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii))*

— *as to the applicant's entitlement to claim the priority of the earlier application (Rule 4.17(iii))*

Published:

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

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

EFFICIENTLY DESCRIBING RELATIONSHIPS BETWEEN RESOURCES

FIELD OF THE INVENTION

[0001] The present invention relates generally to relationship management. Specifically, a system and method are provided to discover relationships between resources by referring to a relationship data structure.

BACKGROUND OF THE INVENTION

[0002] Currently, documents such as, web pages, images and movies are generally encoded in two types of encoding including content-specific and application-specific encoding. A document may include relationship information that defines a relation between the document and another document. The relationship information is encoded in the content-specific or application-specific encoding of the document.

[0003] A decoder must know how the document is encoded to extract the relationship information. The decoder extracts the relationship information within a document by decoding the entire document and extracting the relationship information. The relationship information will not be extracted when the decoder does not understand the encoding of the document.

[0004] A problem is created when the decoder attempts to view or update the relationship information in a secure document that is encrypted or digitally signed. If the secure document is encrypted, the decoder will be unable to decode the secure document and extract the relationship information in the document because the decoder does not know how the secure document and the relationship information are encoded. Moreover, when the secure document is digitally signed the decoder will be able to decode the secure document and extract the relationship information. However, decoding the document and extracting the relationship information may invalidate the digital signature because decoding the document and extracting the relationship information may change a document attribute tracking a date the document was last accessed.

[0005] Therefore, for at least the foregoing reasons, a method to quickly discover internal and external relationships of a document without decoding the document is needed.

SUMMARY OF THE INVENTION

[0006] These and other problems, in the art, are solved by providing a relationship data structure that enables discovery of one or more relationships between a source resource and one or more target resources, without decoding the source or target resources.

[0007] A relationship data structure is stored on a computer-readable medium and describes one or more relationships between the source resource and the one or more target resources. Each relationship of the relationship data structure comprises identification information to uniquely identify a relationship between the source resource and a target resource of the one or more target resources, target information to specify a location of the target resource, and type information to define the semantics of the relationship between the source and the target resource.

[0008] Also, a relationship data structure is associated with a source resource to enable a method to describe a plurality of relationships between a source resource and a plurality of target resources, the plurality of relationships being encoded in a format independent of a source resource's or target resource's encoding. The method to describe the plurality of relationships includes selecting a namespace that defines a relationship schema, populating the relationship data structure based on the plurality of relationships and the relationship schema, and storing the plurality of relationships in the relationship data structure.

[0009] Moreover, a relationship data structure enables a method to discover one or more relationships between a source resource and one or more target resources. To discover the one or more relationships, a location of the source resource is determined. Subsequently, the location of the source resource is employed to ascertain whether a relationship data structure is associated with the source resource. If the relationship data structure is associated with the source resource, the one or more relationships are decoded in accordance with a relationship schema, and the one or more decoded relationships are displayed.

[0010] Accordingly, a relationship data structure provides access to relationships between a source resource and a plurality of target resources without

decoding the source or target resources. The relationship data structure facilitates faster communication with disparate systems because the relationships are faster to resolve and may be stored in a content-neutral format.

[0011] Additional advantages and novel features will be set forth in the description which follows and in part may become apparent to those skilled in the art upon examination of the following or may be learned by practice of the invention.

BRIEF DESCRIPTION OF THE DRAWINGS

[0012] The present invention is described in detail below with reference to the attached figures, wherein:

[0013] FIG. 1 is a block diagram that illustrates a computing environment adapted to implement the present invention;

[0014] FIG. 2 illustrates a package stored in a network environment, and a plurality of relationships between a source resource and a plurality of target resources;

[0015] FIG. 3 illustrates a relationship data structure utilized by the present invention;

[0016] FIG. 4 is a flow chart, of an embodiment of the present invention, which illustrates a method to create the relationship data structure of FIG. 3; and

[0017] FIG. 5 is a flow chart, of an embodiment of the present invention, which illustrates a method to discover one or more relationships by referring to the relationship data structure of FIG. 3.

DETAILED DESCRIPTION OF EMBODIMENTS

[0018] An embodiment of the present invention utilizes a relationship data structure to discover one or more relationships between a source resource and one or more target resources. The source and target resources may be packages addressable using a pack protocol as described in "Pack URI Scheme to Identify and Reference Parts of a Package." The relationship data structure is associated with the source resource and is stored in a content-neutral format.

[0019] FIG. 1 is a block diagram that illustrates a computing environment adapted to implement the present invention. The computing system environment

100 is only one example of a suitable computing environment and is not intended to suggest any limitation as to the scope of use or functionality. Neither should the computing environment 100 be interpreted as having any dependency or requirement relating to any one or combination of components illustrated in the exemplary operating environment 100.

[0020] The present invention is operational with numerous other general purpose or special purpose computing system environments or configurations. Examples of well known computing systems, environments, and/or configurations that may be suitable for use with the invention include, but are not limited to, personal computers, server computers, hand-held or laptop devices, multiprocessor systems, microprocessor-based systems, set top boxes, programmable consumer electronics, network PCs, minicomputers, mainframe computers, distributed computing environments that include any of the above systems or devices, and the like.

[0021] The present invention may be described in the general context of computer-executable instructions, such as program modules, being executed by a computer. Generally, program modules include routines, programs, objects, components, data structures, etc. that perform particular tasks or implement particular abstract data types. The present invention may also be practiced in distributed computing environments where tasks are performed by remote processing devices that are linked through a communications network. In a distributed computing environment, program modules may be located in both local and remote computer storage media including memory storage devices.

[0022] With reference to FIG. 1, an exemplary system for implementing the present invention includes a general purpose computing device in the form of a computer 110. Components of computer 110 may include, but are not limited to, a processing unit 120, a system memory 130, and a system bus 121 that couples various system components including the system memory to the processing unit 120. The system bus 121 may be any of several types of bus structures including a memory bus or memory controller, a peripheral bus, and a local bus using any of a variety of bus architectures. By way of example, and not limitation, such

architectures include Industry Standard Architecture (ISA) bus, Micro Channel Architecture (MCA) bus, Enhanced ISA (EISA) bus, Video Electronics Standards Association (VESA) local bus, Peripheral Component Interconnect Express (PCI Express) and Peripheral Component Interconnect (PCI) bus also known as Mezzanine bus.

[0023] Computer 110 typically includes a variety of computer readable media. Computer readable media can be any available media that can be accessed by computer 110 and includes both volatile and nonvolatile media, removable and non-removable media. By way of example, and not limitation, computer readable media may comprise computer storage media and communication media. Computer storage media includes volatile and nonvolatile, removable and non-removable media implemented in any method or technology for storage of information such as computer readable instructions, data structures, program modules or other data. Computer storage media includes, but is not limited to, RAM, ROM, EEPROM, flash memory or other memory technology, CD-ROM, digital versatile disks (DVD) or other optical disk storage, magnetic cassettes, magnetic tape, magnetic disk storage or other magnetic storage devices, or any other medium which can be used to store the desired information and which can accessed by computer 110. Communication media typically embodies computer readable instructions, data structures, program modules or other data in a modulated data signal such as a carrier wave or other transport mechanism and includes any information delivery media. The term "modulated data signal" means a signal that has one or more of its characteristics set or changed in such a manner as to encode information in the signal. By way of example, and not limitation, communication media includes wired media such as a wired network or direct-wired connection, and wireless media such as acoustic, RF, infrared and other wireless media. Combinations of the any of the above should also be included within the scope of computer readable media.

[0024] The system memory 130 includes computer storage media in the form of volatile and/or nonvolatile memory such as read only memory (ROM) 131 and random access memory (RAM) 132. A basic input/output system 133 (BIOS),

containing the basic routines that help to transfer information between elements within computer 110, such as during start-up, is typically stored in ROM 131. RAM 132 typically contains data and/or program modules that are immediately accessible to and/or presently being operated on by processing unit 120. By way of example, and not limitation, FIG. 1 illustrates operating system 134, application programs 135, other program modules 136, and program data 137.

[0025] The computer 110 may also include other removable/non-removable, volatile/nonvolatile computer storage media. By way of example only, FIG. 1 illustrates a hard disk drive 140 that reads from or writes to non-removable, nonvolatile magnetic media, a magnetic disk drive 151 that reads from or writes to a removable, nonvolatile magnetic disk 152, and an optical disk drive 155 that reads from or writes to a removable, nonvolatile optical disk 156 such as a CD ROM or other optical media. Other removable/non-removable, volatile/nonvolatile computer storage media that can be used in the exemplary operating environment include, but are not limited to, magnetic tape cassettes, flash memory cards, digital versatile disks, digital video tape, solid state RAM, solid state ROM, and the like. The hard disk drive 141 is typically connected to the system bus 121 through an non-removable memory interface such as interface 140, and magnetic disk drive 151 and optical disk drive 155 are typically connected to the system bus 121 by a removable memory interface, such as interface 150.

[0026] The drives and their associated computer storage media discussed above and illustrated in FIG. 1, provide storage of computer readable instructions, data structures, program modules and other data for the computer 110. In Figure 1, for example, hard disk drive 141 is illustrated as storing operating system 144, application programs 145, other program modules 146, and program data 147. Note that these components can either be the same as or different from operating system 134, application programs 135, other program modules 136, and program data 137. Operating system 144, application programs 145, other program modules 146, and program data 147 are given different numbers here to illustrate that, at a minimum, they are different copies. A user may enter commands and information into the processing unit 120 through input devices such as a keyboard 162 and

pointing device 161, commonly referred to as a mouse, trackball or touch pad. Other input devices (not shown) may include a microphone, joystick, game pad, satellite dish, scanner, or the like. These and other input devices are often connected to the processing unit 120 through a user input interface 160 that is coupled to the system bus, but may be connected by other interface and bus structures, such as a parallel port, game port or a universal serial bus (USB). A monitor 191 or other type of display device is also connected to the system bus 121 via an interface, such as a video interface 190. In addition to the monitor, computers may also include other peripheral output devices such as speakers 197 and printer 196, which may be connected through an output peripheral interface 190.

[0027] The computer 110 may operate in a networked environment using logical connections to one or more remote computers, such as a remote computer 180. The remote computer 180 may be a personal computer, a server, a router, a network PC, a peer device or other common network node, and typically includes many or all of the elements described above relative to the computer 110, although only a memory storage device 181 has been illustrated in FIG 1. The logical connections depicted in FIG. 1 include a local area network (LAN) 171 and a wide area network (WAN) 173, but may also include other networks. Such networking environments are commonplace in offices, enterprise-wide computer networks, intranets and the Internet.

[0028] When used in a LAN networking environment, the computer 110 is connected to the LAN 171 through a network interface or adapter 170. When used in a WAN networking environment, the computer 110 typically includes a modem 172 or other means for establishing communications over the WAN 173, such as the Internet. The modem 172, which may be internal or external, may be connected to the system bus 121 via the user input interface 160, or other appropriate mechanism. In a networked environment, program modules depicted relative to the computer 110, or portions thereof, may be stored in the remote memory storage device. By way of example, and not limitation, FIG. 1 illustrates remote application programs 185 as residing on memory device 181. It will be appreciated

that the network connections shown are exemplary and other means of establishing a communications link between the computers may be used.

[0029] FIG. 2 illustrates a package 210 stored in a network environment 200, and a plurality of relationships between a source resource 211 and a plurality of target resources 213-215 and 221.

[0030] With reference to FIGS. 1 and 2, the package 210 is stored on the computer 110. The computer 110 stores a plurality of other packages at addresses relative to the current location of package 210. The package 210 is a container that stores the source resource 211, the relationship resource 212, and the target resources 213-215. The package 210 may also contain a plurality of other source resources and target resources (not shown).

[0031] The source resource 211 may be a file such as, for example, an image, a web page, a document, a video or a package similar to package 210. The source resource 211 is related to the target resources 213-215 and 221. Although not shown, the source resource 211 may also be related to a target resource contained in a package that is relative to package 210. The source resource 211 is associated with the relationship resource 212. The source resource 211 is a resource from which a relationship originates.

[0032] The target resources 213-215 and 221 may be files such as, for example, image files, web pages, documents, videos or packages similar to package 210. The target resources 213-215 are local resources with respect to source resource 211, whereas the target resource 221 is an external resource stored on the remote computer 180, accessible via network 230. The target resource 221 is stored in a database 220 on the remote computer 180. The target resources 213-215 and 221 are resources where the relationship terminates. However, the target resources 213-215 and 221 may take on characteristics associated with the source resource when the target resource is associated with a relationship resource similar to the relationship resource 212.

[0033] The relationship resource 212 stores information indicating that the source resource 211 is related to target resources 213-215 and 221. This information is called relationship information and may be used to construct a

resource hierarchy or to provide metadata about the source resource 211. Further detail of the relationship resource 212 is disclosed below with reference to FIG. 3.

[0034] FIG. 3 illustrates a relationship data structure 300 utilized by the present invention. With reference to FIGS. 2 and 3, the relationship data structure 300 defines the relationship resource 212. A relationships tag 310 defines a plurality of relationships between a source resource and a plurality of target resources. The relationships tag 310 may include a namespace element 315 to define a schema and semantics for the plurality of relationships. The schema and semantics may define a content-neutral encoding for the relationship data structure.

[0035] A relationship tag 320 defines a single relationship between the source resource and a target resource of the plurality of target resources. A plurality of relationship tags 320 may be nested within the relationships tag 310. The relationship tag 320 includes an identification element 325, a type element 326, a target element 327 and a targetBase element 328.

[0036] The identification element 325 uniquely identifies the relationship within the relationships tag 310. The identification element 325 cannot change and must be a valid identifier, such as an extensible Markup Language (XML) identifier. The validity of the identifier is verified according to the schema specified by the namespace element 315, such as, for example, a XML schema.

[0037] The type element 326 may be constrained to a range of values specified by the namespace element 315 or by an application program. The type element 326 further defines the semantics of the relationship. For example, the type element 326 may define whether a source resource requires or needs the target resource.

[0038] Similar to the type element 326, the targetBase element 328 further defines the semantics of the relationship between the source resource and the target resource. The targetBase element 328 may define whether the target resource is internal, local, relative or external.

[0039] The target element 327 defines the location of the target resource. The location may be an address identifier such as a Pack Uniform Resource Identifier (URI), an absolute Uniform Resource Locator (URL), or a relative URL.

A Pack URI is an addressing scheme utilized to reference a package. The Pack URI addressing scheme enables requests for an entire package or for a particular resource contained in the package. A detailed description of the Pack URI addressing scheme is disclosed in a co-pending application entitled "Pack URI Scheme to Identify and Reference Parts of a Package," which has been incorporated by reference.

[0040] Therefore, the relationship data structure 310 enables a user to discover and describe relationships to target resources that are local, within the package, external, target resources that a located at an absolute position, outside the package, on a different computer, and relative, target resource that are located at a position relative to the package. Moreover, the relationship data structure 310 provides a location to store source resource metadata. The metadata information includes annotations and descriptive data, such as print or display data.

[0041] FIG. 4 is a flow chart, of an embodiment of the present invention, which illustrates a method to create the relationship data structure 310 of FIG. 3. In step S410 an application or user generates a request to add a relationship to a source resource. Next, in step S420, the location of the source resource is determined. Upon determining the location of the source resource, in step 430, a check is initiated to verify the existence of the relationship data structure. If the relationship data structure exists, in step S450, a subsequent check is made to verify that the relationship is new. If the relationship is new or has been modified, it is stored in the relationship data structure, otherwise the relationship is a duplicate relationship and it is not stored. In an alternate embodiment the relationship is stored according to the schema and semantics specified by the namespace element 315.

[0042] If the relationship data structure does not exist, in step S440 a relationship data structure is instantiated and associated with the source resource. The relationship data structure may be associated with the source resource by creating a container, such as, a folder, at a location relative to the location of the source resource. Subsequently, the relationship data structure is stored in the container. In an embodiment of the present invention, if the source resource is

located at, for example, /content/spine.xml, the relationship data structure would be instantiated at /content/_rels/ and named "spine.xml.rels." To adhere to this naming convention the relationship data structure may be stored in a sub-container, referred to as "_rels," located at the location of the source resource, and the name of the relationship data structure would be the name of the source resource concatenated with ".rels." Using a similar naming convention to associate all relationship data structures with all source resources would allow efficient access to the relationship data structures based on the location of the source resources.

[0043] After the relationship data structure is instantiated, the relationship data structure is populated with relationship information, such as, for example, identification, target or type information. This relationship information is then stored, in step 460, to allow subsequent retrieval of the relationship information.

[0044] FIG. 5 is a flow chart, of another embodiment of the present invention, which illustrates a method to discover one or more relationships by referring to the relationship data structure of FIG. 3.

[0045] In response to a user or application relationship request to decode the relationship data structure, step S510, a check is made to ascertain whether the source resource is associated with the relationship data structure, in step S520. If the source resource is not associated with a relationship data structure, the relationship data structure is not decoded. One the other hand, if the relationship data structure exists the namespace for the data structure is determined, in step S530. Then, in step S540, a decoder or reader decodes the relationships. After decoding, the one or more relationships may be displayed in step S550. In an alternate embodiment the decoding of the relationship utilizes the schema and semantic specified by the namespace element 315.

[0046] Accordingly, a source resource related to a plurality of target resources will have a plurality of relationships that will be discovered according to the method described above. Additionally, embodiments of the present invention enable a decoder to decode the plurality of relationships associated with the source resource without decoding the source resource. The decoder can decode relationships when the content encoding of the plurality of relationships are in a

format different from the source or target resources encoding. Thus, a relationship between secure or encrypted source and target resources can be determined without breaching the security of the secure or encrypted source and target resources.

[0047] The foregoing descriptions of the invention are illustrative, and modifications in configuration and implementation will occur to persons skilled in the art. For instance, while the present invention has generally been described with relation to FIGS. 1-5, those descriptions are exemplary. Accordingly, the scope of the invention is to be limited only by the following claims.

CLAIMS

We claim:

1. A method to describe one or more relationships between a source resource and one or more target resources, the one or more relationships encoded in a format independent of an encoding of the source resource, the method comprising:

selecting a namespace that defines a relationship schema;

populating a relationship data structure based on the one or more relationships and the relationship schema;

storing the one or more relationships in the relationship data structure; and

associating the relationship data structure with the source resource.

2. The method according to claim 1, wherein the one or more target resources are resources selected from a group consisting of local, external and relative resources.

3. The method according to claim 1, wherein the source resource and the one or more target resources are packages.

4. The method according to claim 1, wherein populating the relationship data structure further comprises:

retrieving identification information that identifies the one or more relationships; and

validating the retrieved identification information by utilizing the relationship schema.

5. The method according to claim 1, wherein associating the relationship data structure with the source resource further comprises:

storing the relationship data structure at an address relative to a location of the source resource.

6. The method according to claim 5, wherein the address relative to the location of the source resource addresses a subfolder at the location of the source resource.

7. A computer-readable medium having computer-executable instructions to perform the method recited in claim 1.

8. A computer system having a processor, a memory, and an operating environment, the computer system operable to execute the method recited in claim 1.

9. A method to discover one or more relationships between a source resource and one or more target resources, the one or more relationships encoded in a format independent of an encoding of the source resource or the one or more target resources, the method comprising:

locating the source resource;

determining whether a relationship data structure is associated with the source resource;

decoding the one or more relationships without decoding the source resource; and

displaying the one or more relationships.

10. The method according to claim 9, wherein the source resource or the one or more target resources are secure resources.

11. The method according to claim 9, wherein the source and the one or more target resources are addressed using a Pack URI.

12. The method according to claim 9, wherein decoding the one or more relationships without decoding the source resource further comprises:

prior to decoding the one or more relationships, determining a schema of the relationship data structure; and

utilizing the schema to decode the one or more relationships stored in the relationship data structure.

13. The method according to claim 9, wherein determining whether the relationship data structure is associated with the source resource further comprises:

checking for an existence of a subfolder at the location of the source resource.

14. The method according to claim 13, wherein the subfolder is located at an address relative to the location of the source resource.

15. A computer-readable medium having computer-executable instructions to perform the method recited in claim 9.

16. A computer system having a processor, a memory, and an operating environment, the computer system operable to execute the method recited in claim 9.

17. A relationship data structure, stored on a computer-readable medium, providing data corresponding to one or more relationships between a source resource and one or more target resources, the relationship data structure comprises:

identification information to uniquely identify a relationship of the one or more relationships;

target information to specify a location of a target resource of the one or more target resources; and

type information to define the relationship between the source resource and the target resource.

18. The relationship data structure of claim 17, further comprising:

semantic information to assist when decoding the relationship.

19. The relationship data structure of claim 17, wherein the one or more relationships stored in the relationship data structure are nested.

20. The relationship data structure of claim 17, wherein the relationship data structure includes metadata to describe the source resource.

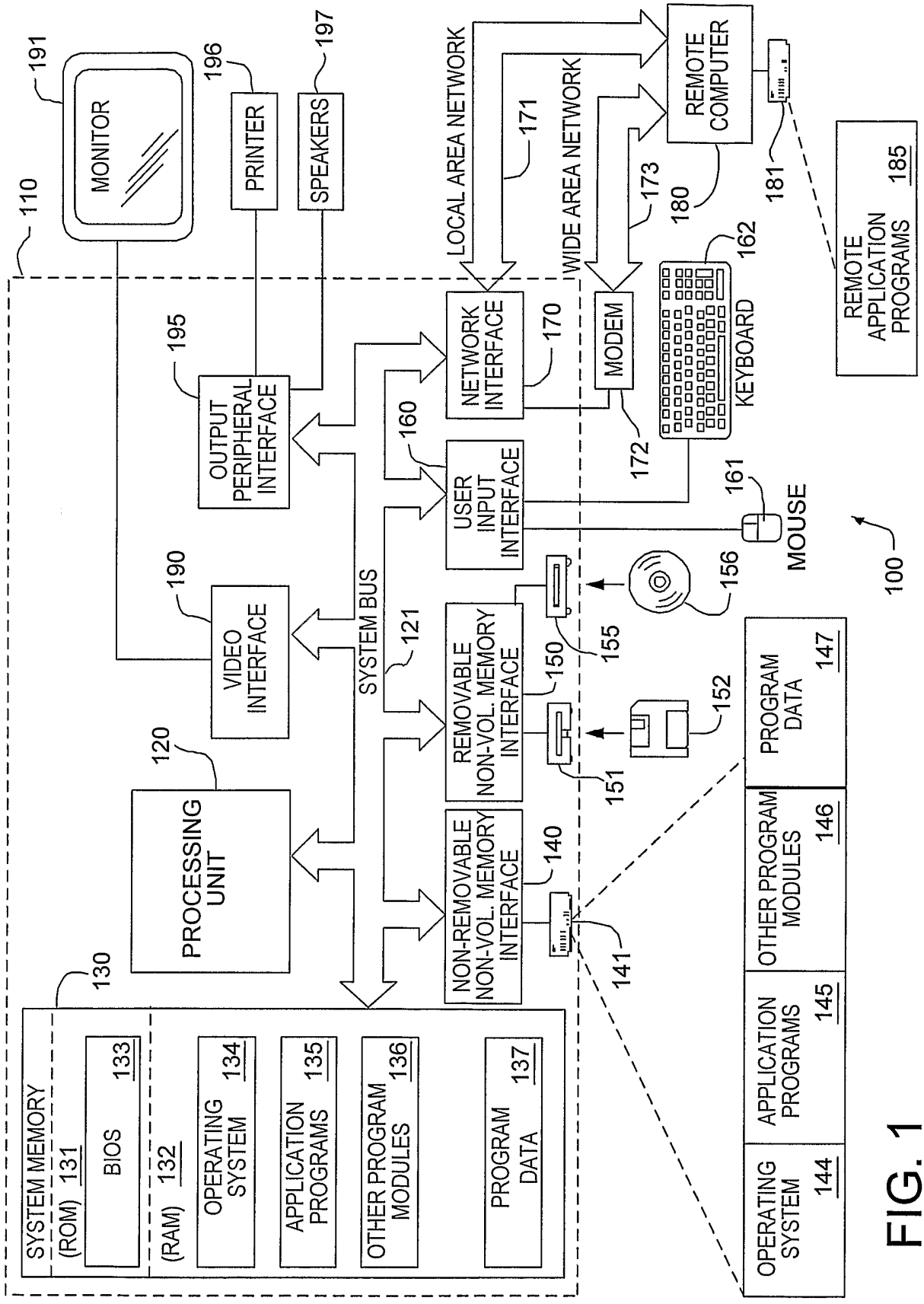

100
110
120
121
130
131
132
133
134
135
136
137
140
141
144
145
146
147
150
151
152
155
156
160
161
162
170
171
172
173
180
181
185
190
191
195
196
197
BIOS
BOOT ROM
COMPUTER
DISPLAY
HARD DISK
KEYBOARD
LOCAL AREA NETWORK
MODEM
MONITOR
MOUSE
NETWORK INTERFACE
NON-REMOVABLE NON-VOL. MEMORY INTERFACE
NON-VOL. MEMORY INTERFACE
OPERATING SYSTEM
OTHER PROGRAM MODULES
OUTPUT PERIPHERAL INTERFACE
PARALLEL PORT
PC
PDA
PHONE
PRINTER
PROCESSING UNIT
PROGRAM DATA
REMOTE APPLICATION PROGRAMS
REMOTE COMPUTER
REMOVABLE NON-VOL. MEMORY INTERFACE
SATELLITE DISH
SCANNER
SCSI
SPEAKERS
SYSTEM BUS
SYSTEM MEMORY
(ROM)
(RAM)
USB
USER INPUT INTERFACE
VIDEO INTERFACE
WIDE AREA NETWORK

FIG. 1

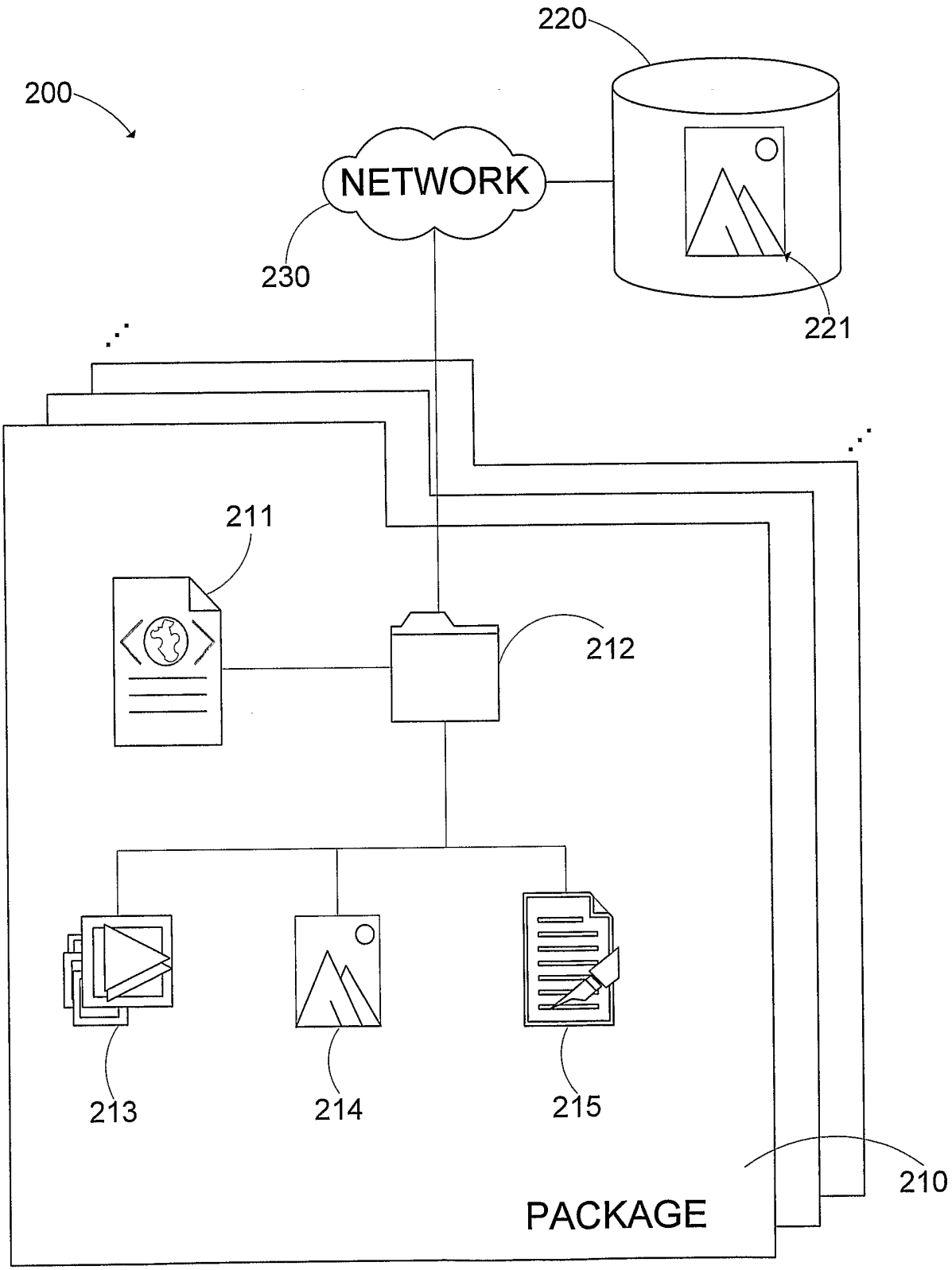


FIG. 2

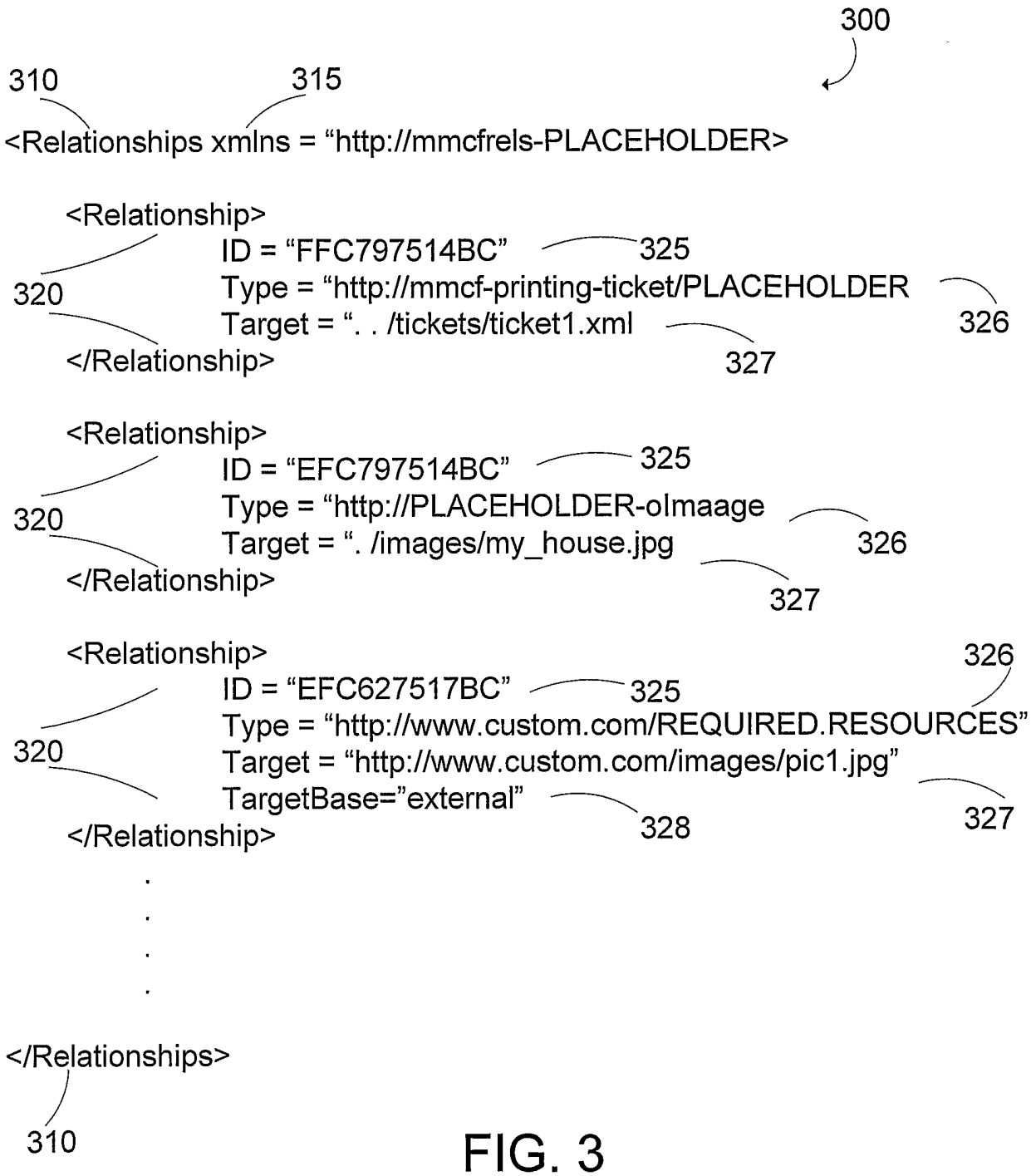

300
310
315
<Relationships xmlns = "http://mmcfrels-PLACEHOLDER>
<Relationship>
ID = "FFC797514BC"
325
320
Type = "http://mmcf-printing-ticket/PLACEHOLDER
326
Target = ". . /tickets/ticket1.xml
327
</Relationship>
<Relationship>
ID = "EFC797514BC"
325
320
Type = "http://PLACEHOLDER-olmaage
326
Target = ". /images/my_house.jpg
327
</Relationship>
<Relationship>
326
ID = "EFC627517BC"
325
Type = "http://www.custom.com/REQUIRED.RESOURCES"
320
Target = "http://www.custom.com/images/pic1.jpg"
TargetBase="external"
328
327
</Relationship>
</Relationships>
310

FIG. 3

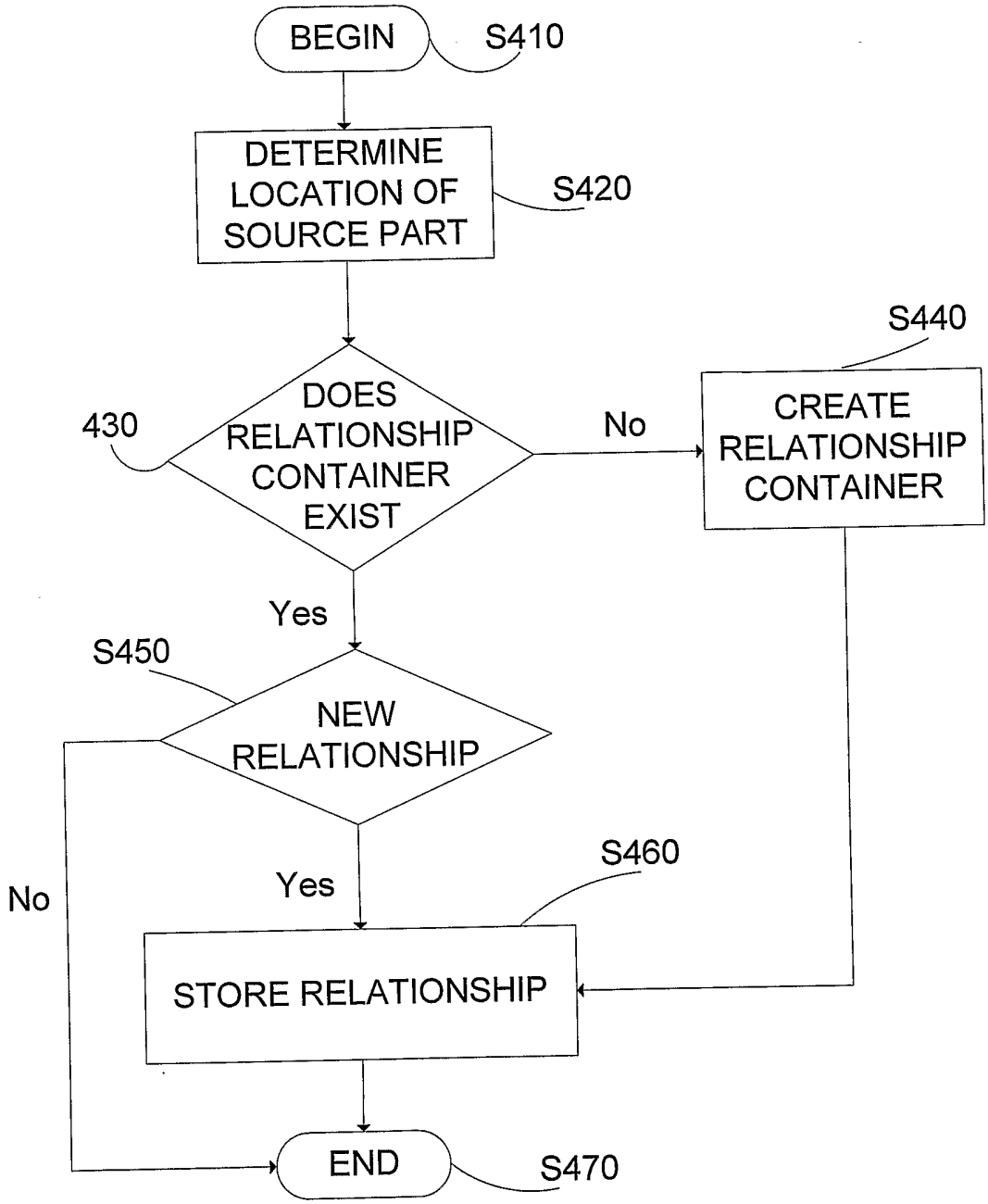

BEGIN
S410
DETERMINE LOCATION OF SOURCE PART
S420
S440
DOES RELATIONSHIP CONTAINER EXIST
430
No
CREATE RELATIONSHIP CONTAINER
Yes
S450
NEW RELATIONSHIP
No
Yes
S460
STORE RELATIONSHIP
END
S470

FIG. 4

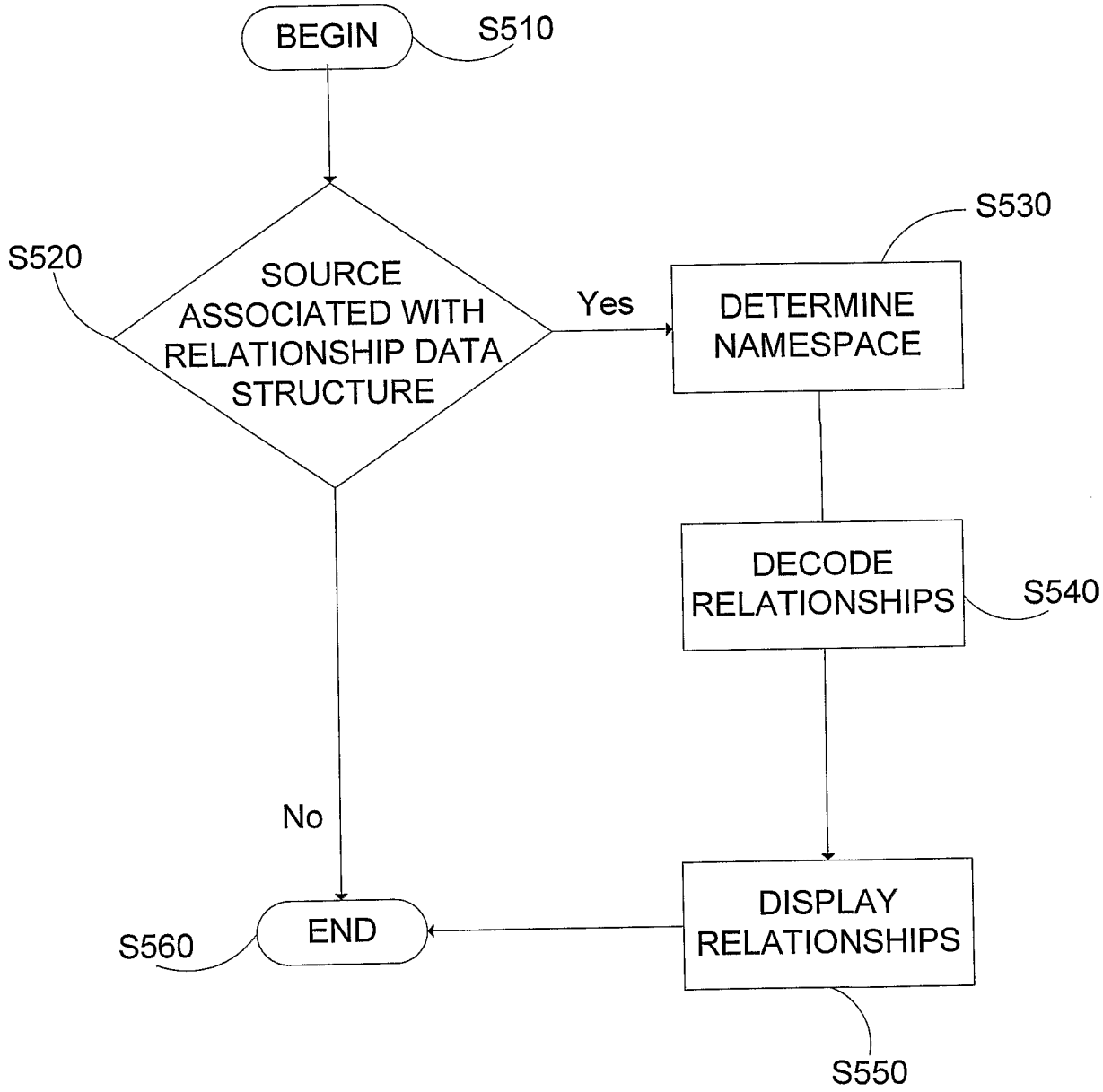

BEGIN
S510
S520
SOURCE ASSOCIATED WITH RELATIONSHIP DATA STRUCTURE
Yes
S530
DETERMINE NAMESPACE
DECODE RELATIONSHIPS
S540
No
END
S560
DISPLAY RELATIONSHIPS
S550

FIG. 5