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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
19 April 2018 (19.04.2018)

WIPO | PCT

(10) International Publication Number
WO 2018/068080 A1

(51) International Patent Classification:
G06F 17/30 (2006.01) *G06F 21/00* (2013.01)

(21) International Application Number:
PCT/AU2017/000219

(22) International Filing Date:
13 October 2017 (13.10.2017)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
2016904177 14 October 2016 (14.10.2016) AU

(71) Applicant: **JANUSNET PTY LIMITED** [AU/AU]; Level 8, 6A Glen Street, Milsons Point NSW 2061 (AU).

(72) Inventors: **COLLA, Gregory Alan**; Level 8, 6A Glen Street, Milsons Point NSW 2061 (AU). **HUANG, Kien Sen**; Level 8, 6A Glen Street, Milsons Point NSW 2061 (AU).

(74) Agent: **WALLINGTON-DUMMER PATENT AND TRADE MARK ATTORNEYS**; PO Box 3888, Sydney NSW 2001 (AU).

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

(54) Title: ARCHITECTURE, METHOD AND APPARATUS FOR ENFORCING COLLECTION AND DISPLAY OF COMPUTER FILE METADATA

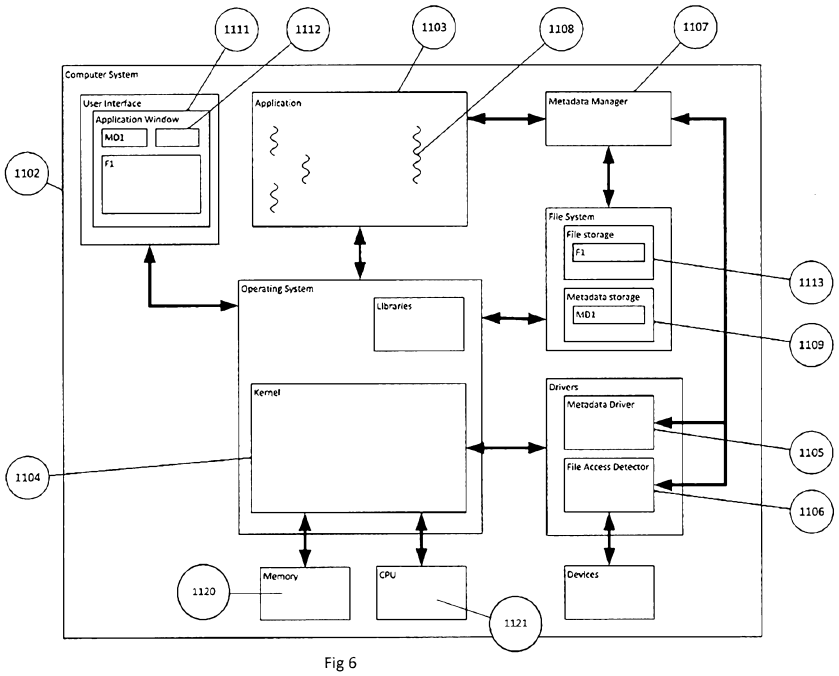


Fig 6

(57) Abstract: The present invention provides a system for displaying and capturing file metadata of an application data file stored on a computer; said computer including a processor and memory; said memory storing an operating system for the computer, a kernel of the operating system and Metadata Storage; said computer including a user interface and at least one installed application for interacting with a user which processes said application data file; said system comprising: a Metadata Filter Driver installed on said computer; at least one Metadata Application Thread, which the system injects into said application when a process for execution of said application is created; a Metadata Manager operating on said computer, whereby said Metadata Filter Driver signals said Metadata Manager when said at least one application data file is accessed, which then notifies said Thread appropriate to the at least one installed application which is accessing said application data file.

WO 2018/068080 A1

[Continued on next page]

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Published:

— *with international search report (Art. 21(3))*
— *with amended claims (Art. 19(1))*

Title: Architecture, Method and Apparatus for Enforcing Collection and Display of Computer File Metadata

[0001] **Field of invention**

[0002] The present invention relates to the field of enforcement of collection and display of computer file metadata and more particularly to a software architecture, method and apparatus for enabling the enforcement of collection and display of computer file metadata.

[0003] **Background**

[0004] File metadata information for enterprise is considered of vital importance for consistent identification and reference, attribution, resource integrity checking, classification, describing resource relationships, searching, sorting, generation of reports, online access, system automation and many other functions.

[0005] File metadata is not limited to simple file attributes such as its title, subject, author, creation date, but can contain much more information relevant to the business processes and systems of the enterprise, such as a document's revision number, its status, whether it is draft or final, who approved its release, company, copyright owner, number of pages, number of words, keywords, case identifier or document identifier. If the file is a photograph, the metadata can contain location details, when the photo was taken, the subject of the photo and details of the camera that took the photo, such as lens focal length and aperture settings. If the file is an audio recording, its metadata may contain recoding information, such as the artist, when it was recorded, equipment that was used for recording, and the name and location of the recording studio.

[0006] Files can be stored on many different media, such as file servers, Document Management Systems (DMS) (1), workstation hard drives, file servers, removable storage media (such as "thumb drives"), or in the cloud.

[0007] DMS systems are often configured to collect necessary metadata when files or documents are added to the system. This metadata may be generated, based on the content of the source file, or generated automatically, or the system may prompt the user to supply the metadata. In general, DMS systems store metadata in databases (catalogues), logically separate from the location of the file. For effective DMS system operation, an organisation must be prepared to commit resources to the purchasing, design, commissioning and operation of the system. Users must traverse a learning curve to use these systems effectively. Such overheads may be prohibitive for smaller organisations.

[0008] An alternative to storing file metadata in a database, is to store it in the file system itself, such as in Microsoft NTFS Alternate Data Streams (2) or Apple resource forks. Alternatively, file metadata can be stored in the file itself. For example, Microsoft Word has the capability to collect document properties, such as the document title, subject and author, which is stored in the file. Additional metadata can be collected and stored with the use of allow for the addition of Custom Document Properties (3). Such metadata can be used for automated file access control (4) and encryption (5). Only a minority of the applications in use by an organisation would have this capability. Furthermore, the entry of metadata is up to the operator, and can easily be forgotten.

[0009] Organisations, such as governments and enterprises, have security classification policies which define security classification systems and handling protocols, so that information generated by the organisation is handled correctly, both by humans and by computer systems, based on the information's sensitivity. For example, the Australian Government has published its security classification system (6), and its handling protocols (7). At a minimum, such systems and protocols normally define the appropriate range of classifications, and when a security classification should be applied, such as when a document is saved, or prior to printing, and where the security classification should be shown on a document.

[00010] Solutions exist to add security classification metadata to Microsoft Office documents when the associated software application is being used (8) (9). However,

these solutions are limited to applications where COM Application Programming Interfaces (APIs) are available in the application (10), and the application's API allows for the programmatic extension of the application's user interface. As such, these solutions do not provide for wider usage in other computer applications, such as Microsoft's Paintbrush, Notepad or applications which do not have an extension API, or where an application extension API is available, but does not provide the capability to dynamically modify the user interface. Furthermore, these solutions will not work in a non-Windows environment, such as Apple's OS X, BSD, Linux, Google Android, or other operating systems. As such, the scope of such solutions is quite limited, so security classification policies, or metadata collection policies cannot be fully enforced by these solutions.

[00011] Furthermore, the application of these solutions to a document can modify the document, say by adding metadata to the header or footer of a document, or otherwise within the text of the document. Such process can be contrary to an organisation's business practices. For example, when a document is sent by one organisation to a second organisation, the second organisation may want to keep the document intact in the form in which it was sent. Modifying the document's text in any way may diminish future evidentiary value of the document.

[00012] Solutions also exist to apply classification metadata to other file types (images, video, audio, PDF) programmatically (11). Such implementations, however, do not enforce user metadata policy, in that the display of file metadata when viewing the contents of a file may not be noticeable or conspicuous, nor do they enforce the mandatory collection of metadata.

[00013] Background art includes:

[00014] Taylor et al (12) describe systems and methods for providing user interface to receive metadata from a user for an application and associated generated file. This system is limited to applications where the method is a feature of the native application. Its limitation is that does not work for applications which are already compiled and deployed to the computer. It does not provide a mechanism to enforce

organisation policy of when the user must enter metadata.

[00015] Turco (8) describes a plug-in which extends Microsoft Office functionality to collect and manage file associated security classification metadata. This method is only appropriate for applications which have an extension application programming interface. Its limitation is that it will not work for applications which do not have an extension API.

[00016] Stahl et al (13) describe a method of securing access to a file in a document management application using security classification metadata. This system is limited in that it only operates on files which are stored in the document management application. It does not work on files stored outside of that system, such as on a workstation's file system nor on a network file share.

[00017] Trainer et al (14) describe method and system for a storage of files and associated file metadata. This system is limited in that the files are stored within the system's own persistent storage. It does not work with a workstation's own file storage, not with network file storage, such as a Storage Area Network (SAN). Furthermore, user interaction with the system is via the system's own user interface. It does not interact with the user when the user is using an application when processing a file.

[00018] In summary, there is a gap in the current field whereby organisations provide a range of computer applications for users to complete business tasks (Figure 7, shown as θ) and there is a subset of the organisation's applications (Figure 7, shown as ϕ) which provide a native function to enter and store custom metadata, there is a subset of the organisation's applications (Figure 7, shown as λ) which provide a function to enforce metadata collection and there is a subset of the organisation's applications (Figure 7, shown as ω) which have an extension API, whereby an extension can be used to enforce metadata policy. The organisation requires users to comply with metadata policy, such as the mandatory collection of appropriate metadata when files are saved, and/or users must be shown file metadata when operating on a file. Of all the applications used by the organisation, θ, only a small subset, shown shaded, can

provide the required functionality natively, or with the addition of an application extension. These prior art approaches are specific to the application being used to operate on the file, the operating system on which the application is executing, intentionally modifying the contents of the file, or not having the ability to collect metadata at appropriate times.

[00019] These problems may be addressed or ameliorated in at least some embodiments of the present invention, which in at least some preferred forms provide an architecture, method and apparatus to enforce metadata policy across a plethora of computer applications and operating systems.

[00020] In alternative forms these problems may be addressed or ameliorated in at least some embodiments of the present invention, which in at least some preferred forms provide an architecture, method and apparatus to enforce metadata policy across a plethora of computer applications and operating systems without modifying the content of the file.

[00021] **References**

1. **Wikipedia.** Document management system. [Online] [Cited: 7 Sep 2016.] https://en.wikipedia.org/wiki/Document_management_system.

2. **Microsoft Corporation.** File Classification Infrastructure Alternate Data Stream (ADS) File Format. [Online] 16 Dec 2011. https://msdn.microsoft.com/en-us/library/hh537062.aspx.

3. —. Understanding Custom Document Properties in Microsoft Office Word 2003. *Microsoft Developer Network.* [Online] January 2004. [Cited: 23 June 2016.] https://msdn.microsoft.com/en-us/library/aa537154.

4. —. Introducing Dynamic Access Control. *Technet.* [Online] 1 November 2012. [Cited: 28 June 2016.] http://social.technet.microsoft.com/wiki/contents/articles/14269.introducing-dynamic-access-control.aspx.

5. —. Using File Classification Infrastructure (FCI) and AD RMS to automatically protect sensitive information. [Online] 30 January 2010. https://blogs.technet.microsoft.com/filecab/2010/01/30/using-file-classification-infrastructure-fci-and-ad-rms-to-automatically-protect-sensitive-information/.

6. **Australian Government.** Australian Government security classification system. *Protective Security Policy Framework.* [Online] 6 May 2015. https://www.protectivesecurity.gov.au/informationsecurity/Pages/AustralianGovernmentSecurityClassificationSystem.aspx.

7. —. Protectively marking and handling sensitive and security classified information. *Protective Security Classification Framework.* [Online] 6 May 2015. https://www.protectivesecurity.gov.au/informationsecurity/Pages/ProtectivelyMarkingAndHandlingSensitiveAndSecurityClassifiedInformation.aspx.

8. **Turco, Anthony Jay.** *Electronic document classification and monitoring. 2003/0182583 A1* US, 25 Sep 2003.

9. **Pulfer et al.** *Document classification toolbar. 8,024,304* US, 26 October 2006.

10. **Microsoft Corporation.** COM Add-ins Part I: Introducing an Office 2000 Solution for the Entire (Office) Family. *Microsoft Developer Network.* [Online] May 1999. [Cited: 23 June 2016.] https://msdn.microsoft.com/en-us/library/aa155767.

11. —. File Classification Infrastructure: Technical White Paper. *Microsoft Download Center.* [Online] May 2009. http://download.microsoft.com/download/D/D/3/DD3FB42C-D3C7-47C0-9431-40D80256FB0A/FCI_TDM_WP_May_09.pdf.

12. **Taylor et al.** *Methods and Systems for Providing a Customized User Interface for Viewing and Editing Meta-Data. 8,171,394 B2* US, 1 May 2012.

13. **Stahl, Noah Z, Bartlett, Wendy S and Nrooks, Randall S.** *Secure Document Management. 2010/0146593 A1* US, 10 Jun 2010.

14. **Trainor, James and Pike, James Thomas.** *Method and System for Managing Metadata Information. 2006/0235871 A1* US, 19 Oct 2006.

[00022] **Notes**

[00023] The term "comprising" (and grammatical variations thereof) is used in this specification in the inclusive sense of "having" or "including", and not in the exclusive sense of "consisting only of".

[00024] The above discussion of the prior art in the Background of the invention, is not an admission that any information discussed therein is citable prior art or part of the common general knowledge of persons skilled in the art in any country.

[00025] **Summary of Invention**

[00026] **Definitions**

[00027] **Application**: a set of instructions executable by a computer in order that the computer may carry out a function. An application may be a text editor program or it may be a spreadsheet program for example. It is commonly the case that applications read data from files in order to execute a programmed function and then output data to files as a result of execution of the programmed function.

[00028] **Persistent storage:** any storage device that retains data after power to that device is shut off. It is also sometimes referred to as non-volatile storage.

[00029] Hard disk drives and solid-state drives are common types of persistent storage. This can be in the form of file, block or object storage. On the other hand, RAM and cache are typically non-persistent, and data is erased when power is turned off. However, certain types such as non-volatile RAM and flash-based RAM are persistent. Persistence is beneficial so that in the event of a crash or reboot, data is not lost.

[00030] **Application Program Interface (API)**: a set of subroutine definitions, protocols, and tools for building application software. In general terms, it is a set of clearly defined methods of communication between various software components. A good API makes it easier to develop a computer program by providing all the building blocks, which are then put together by the programmer. An API may be for a web-based system, operating system, database system, computer hardware or software library. An API specification can take many forms, but often includes specifications for routines, data structures, object classes, variables or remote calls. POSIX, Microsoft Windows API, the C++ Standard Template Library and Java APIs are examples of different forms of APIs. Documentation for the API is usually provided to facilitate usage.

[00031] Just as a graphical user interface makes it easier for people to use programs, application programming interfaces make it easier for developers to use certain technologies in building applications. By abstracting the underlying implementation and only exposing objects or actions the developer needs, an API reduces the cognitive load on a programmer.

[00032] **Metadata**: broadly, metadata is data that provides information about data. A computer file will contain file data. In at least some instances there may be associated metadata with that file data. The metadata is logically distinct from the file data in most instances. The metadata may describe or summarise aspects of the file data and or it may specify rules for use of the file data.

[00033] **Thread**: a thread of execution is the smallest sequence of programmed instructions that can be managed independently by a scheduler, which is typically a part of the operating system. The implementation of threads and processes differs between operating systems, but in most cases a thread is a component of a process. Multiple threads can exist within one process, executing concurrently and sharing resources such as memory, while different processes do not share these resources. In particular, the threads of a process share its executable code and the values of its variables at any given time.

[00034] **Kernel**: the central part of an operating system. It manages the tasks of the computer and the hardware- most notably memory and CPU time. The kernel is the most fundamental part of an operating system. It can be thought of as the program which controls all other programs on the computer.

[00035] In most instances a computer user never interacts directly with the kernel. It runs behind the scenes.

[00036] **File system filter driver:** a file system filter driver is an optional driver that adds value to or modifies the behaviour of a file system. In the case of the Windows operating system a file system filter driver is a kernel-mode component that runs as part of the Windows executive;

[00037] A file system filter driver can filter I/O operations for one or more file systems or file system volumes. Depending on the nature of the driver, filter can mean log, observe, modify, or even prevent. Typical applications for file system filter drivers include antivirus utilities, encryption programs, and hierarchical storage management systems.

[00038] **Addin** (or plug-in, plugin, add-in, add-on, addon, or extension): is a software component that adds a specific feature to an existing computer program. When a program supports plug-ins, it enables customization. The common examples are the plug-ins used in web browsers to add new features such as search-engines, virus scanners, or the ability to use a new file type such as a new video format. Well-known browser plug-ins include the Adobe Flash Player, the QuickTime Player, and the Java plug-in, which can launch a user-activated Java applet on a web page to its execution on a local Java virtual machine.

[00039] Applications support plug-ins for many reasons. Some of the main reasons include:

- to enable third-party developers to create abilities which extend an application
- to support easily adding new features
- to reduce the size of an application

- to separate source code from an application because of incompatible software licenses.

[00040] At least some embodiments of the present invention provide an architecture, method and apparatus which provides additional functionality to applications on a computer system to give the applications the ability to display, capture and manage file metadata. In preferred forms, the architecture, method and apparatus operate without requiring modification of source code of the application. In preferred forms it also has no dependency on the application having an extension Application Programming Interface for reception of addin (plug-in) components.

[00041] Accordingly in one broad form of the invention there is provided a system for displaying and capturing file metadata of an application data file stored on a computer; said computer including a processor and memory; said memory storing an operating system for managing operations of the computer; said memory further storing a kernel of the operating system; said computer further including a user interface and at least one installed application installed on said computer for interacting with a user which, when executed by the processor under the control of the operating system, processes said application data file; said system comprising:

a. a File System Metadata Filter Driver installed on said computer, whereby said kernel of said operating system signals said File System Metadata Filter Driver when said installed application accesses said application data file;
b. at least one Metadata Application Thread, which the system injects into said installed application when an application process for execution of said installed application is created, whereby said Metadata Application Thread handles messages generated by said installed application and modifies the user interface of said installed application to display information related to installed application file metadata of said installed application;
c. Said memory including Metadata Storage which stores said application file metadata, from which said at least one Metadata Application Thread reads said file metadata, and to which said at least one Metadata Application Thread writes said installed application file metadata;

d. a Metadata Manager operating on said computer, whereby said File System Metadata Filter Driver signals said Metadata Manager when said at least one application data file is accessed, and which in turn notifies said Metadata Application Thread appropriate to that one of said at least one installed application which is accessing said application data file.

[00042] Accordingly in yet a further broad form of the invention there is provided a system for displaying and capturing file metadata of an application data file stored on a computer; said computer including a processor and memory; said memory storing an operating system for managing operations of the computer; said memory further storing a kernel of the operating system; said computer further including a user interface and at least one installed application installed on said computer for interacting with a user which, when executed by the processor under the control of the operating system, processes said application data file; and wherein said application has no function to capture custom metadata for said file; said system comprising:

a. a File System Metadata Filter Driver installed on said computer, whereby said kernel of said operating system triggers said File System Metadata Filter Driver when said installed application accesses said application data file;
b. a user interface component in the form of a Metadata Application Thread to display information related to said file metadata to the user of said computer; and captures file metadata information from said user;
c. metadata storage which stores said file metadata and wherein said metadata storage is persistent;
d. a Metadata Manager operating on said computer, whereby said File System Metadata Filter Driver signals said Metadata Manager when said application data files are accessed, and which in turn notifies said user interface component;

[00043] Accordingly in yet a further broad form of the invention there is provided a system for displaying and capturing file metadata of an application data file stored on a computer; said computer including a processor and memory; said memory storing an operating system for managing operations of the computer; said memory further storing a kernel of the operating system; said computer further including a user

interface and at least one installed application installed on said computer for interacting with a user which, when executed by the processor under the control of the operating system, processes said application data file; said system further comprising:

a. Metadata Storage which stores said file metadata, whereby said storage is persistent;
b. at least one Metadata Application Thread in each said installed application, whereby said Metadata Application Thread intercepts file operation calls from the installed application; handles said messages of said installed application; modifies the user interface of said installed application to display information related to said file metadata to the user of said computer; captures file metadata information from said user; and creates, updates, deletes and otherwise accesses said file metadata from said Metadata Storage;

[00044] In a further broad form of the invention there is provided a system to enforce metadata policy on use of data stored in a file without modifying the data in the file to which the meta data policy applies;
said system including a system for displaying and capturing file metadata of an application data file stored on a computer; said computer including a processor and memory; said memory storing an operating system for managing operations of the computer; said memory further storing a kernel of the operating system; said computer further including a user interface and at least one installed application installed on said computer for interacting with a user which, when executed by the processor under the control of the operating system, processes said application data file; said system comprising:

a. a File System Metadata Filter Driver installed on said computer, whereby said kernel of said operating system signals said File System Metadata Filter Driver when said installed application accesses said application data file;
b. at least one Metadata Application Thread, which the system injects into said installed application when an application process for execution of said installed application is created, whereby said Metadata Application Thread handles messages generated by said installed application and modifies the

user interface of said installed application to display information related to installed application file metadata of said installed application;

c. Said memory including Metadata Storage which stores said application file metadata, from which said at least one Metadata Application Thread reads said file metadata, and to which said at least one Metadata Application Thread writes said installed application file metadata;
d. a Metadata Manager operating on said computer, whereby said File System Metadata Filter Driver signals said Metadata Manager when said at least one application data file is accessed, and which in turn notifies said Metadata Application Thread appropriate to that one of said at least one installed application which is accessing said application data file.

[00045] In a further broad form of the invention there is provided a system of organization of programme structures and file structures in a computer comprising a processor in communication with a memory in order to enforce metadata policy on use of data stored in a file without modifying the data in the file to which the meta data policy applies;
said system including a system for displaying and capturing file metadata of an application data file stored on the computer; said computer including said processor and said memory; said memory storing an operating system for managing operations of the computer; said memory further storing a kernel of the operating system; said computer further including a user interface and at least one installed application installed on said computer for interacting with a user which, when executed by the processor under the control of the operating system, processes said application data file; said system comprising:

a. a File System Metadata Filter Driver installed on said computer, whereby said kernel of said operating system signals said File System Metadata Filter Driver when said installed application accesses said application data file;
b. at least one Metadata Application Thread, which the system injects into said installed application when an application process for execution of said installed application is created, whereby said Metadata Application Thread handles messages generated by said installed application and modifies the

user interface of said installed application to display information related to installed application file metadata of said installed application;

c. Said memory including Metadata Storage which stores said application file metadata, from which said at least one Metadata Application Thread reads said file metadata, and to which said at least one Metadata Application Thread writes said installed application file metadata;

d. a Metadata Manager operating on said computer, whereby said File System Metadata Filter Driver signals said Metadata Manager when said at least one application data file is accessed, and which in turn notifies said Metadata Application Thread appropriate to that one of said at least one installed application which is accessing said application data file.

[00046] In yet a further broad form of the invention there is provided a method for displaying and capturing file metadata of an application data file stored on a computer; said computer including a processor and memory; said memory storing an operating system for managing operations of the computer; said memory further storing a kernel of the operating system; said computer further including a user interface and at least one installed application installed on said computer for interacting with a user which, when executed by the processor under the control of the operating system, processes said application data file; said method comprising:

a. installing a File System Metadata Filter Driver on said computer, whereby said kernel of said operating system signals said File System Metadata Filter Driver when said installed application accesses said application data file;

b. injecting at least one Metadata Application Thread into said installed application when an application process for execution of said installed application is created, whereby said Metadata Application Thread handles messages generated by said installed application and modifies the user interface of said installed application to display information related to installed application file metadata of said installed application;

c. providing Metadata Storage in said memory which stores said application file metadata, from which said at least one Metadata Application Thread reads said file metadata, and to which said at least one Metadata Application Thread writes said installed application file metadata;

d. providing a Metadata Manager operating on said computer, whereby said File System Metadata Filter Driver signals said Metadata Manager when said at least one application data file is accessed, and which in turn notifies said Metadata Application Thread appropriate to that one of said at least one installed application which is accessing said application data file.

[00047] Preferably, the system includes configuration information which controls behaviour of said system.

[00048] Preferably, said system configuration information which controls behaviour of said system including nominating computer applications on which the system operates.

[00049] Preferably, said system modifies said installed application to capture said file metadata of said application data file from said user when said file is being saved within said installed application.

[00050] Preferably, said system modifies said user interface of said at least one installed application to allow said user to set metadata appropriate to said application data file.

[00051] Preferably, said system modifies a title bar displayed in said user interface of said installed application in order to display said installed application file metadata.

[00052] Preferably, said system modifies said application to display said file metadata in a client window of the user interface of said installed application.

[00053] Preferably, said system modifies said installed application to overlay said file metadata on a printout when said application data file is printed.

[00054] Preferably, said operating system is Microsoft Windows, or Apple OSX, a BSD derived operating system, such as FreeBSD or Open BSD, Linux or some other operating system which allows for additional software components which can be

signalled when files are accessed.

[00055] Preferably, said application is Microsoft's notepad.exe, or some other application which can open, display, edit and save text files.

[00056] Preferably, said application is Microsoft paintbrush.exe, or some other type of application which can open, display, edit and save image files.

[00057] Preferably, said application is Microsoft's Word.exe, OpenOffice, LibreOffice, Adobe PageMaker, WPS Office, or some other type of application which can open, display, edit and save document files.

[00058] Preferably, said installed application is Windows Media Player or some other type of application which plays back audio files, and wherein said system modifies said installed application to play audio cues associated with metadata of said audio files before, during and/or after the playing back of said audio file.

[00059] Preferably, said application plays back video files, such as Windows Media Player. Said system modifies said application to play video and/or audio cues associated with said file's metadata before, during and/or after the playing back of said file.

[00060] Preferably, said system modifies said application to display said metadata in the client region of said application during video playback.

[00061] Preferably, said application is a system application, such as Windows File Explorer, or some other system or third party application which provides file management functionality to said user, such as copying, moving, deleting, renaming, updating or creating files.

[00062] Preferably, said application is a web browser, such as Internet Explorer, Safari, Chrome, Firefox, or some other software application for retrieving, presenting, and traversing information from a computer network, such as the Internet or an

Intranet.

[00063] Preferably, said file metadata is security classification information.

[00064] Preferably, said File System Metadata filter Driver is a Windows mini filter driver, or a Max OS X listener, or some additional operating system kernel component which observes when the computer reads or writes files.

[00065] Preferably, said metadata storage is part of said computer's file system, such as Microsoft's Alternate Data Streams (ADS), or some form of extended file attribute, whereby the file system can associate a file with file metadata.

[00066] Preferably, said file metadata is stored as properties within said file, such as Microsoft Office's Custom Document Properties, or other file properties which are stored within the file.

[00067] Preferably, said file metadata is stored on said computer in a database, such as SQLite, BerkelyDB, or some other database installed on said computer, capable of persistent storage of said file metadata.

[00068] Preferably, said file metadata is stored on a computer connected to said computer by a computer network, such as database server, like Microsoft SQL Server or Oracle, or a cloud service, or some other network storage capable of persistent storage of said file metadata.

[00069] Preferably, said metadata is stored in a combination of the file system, a local database, a network accessible database, and in said file.

[00070] Preferably, said application is capable of transferring files, such as Skype, or Dropbox, or Filezilla, or a browser such as Internet Explorer, Google Chrome, Apple Safari, or Mozilla, or some other application which can copy or move a file from said computer to another computer over a computer network. Said system modifies said application to interrogate the metadata of said file prior to it being transferred, and can

reject the transfer of a file, based on rules defined in said configuration.

[00071] Preferably, one such rule which may be enforced by said system is that a security classified file can only be transmitted across network channels, and to destinations, with adequate protection.

[00072] Preferably, said computer is capable of performing screen capture.

[00073] Preferably, said system modifies said application to interrogate the metadata of said file being displayed, and if the display is subject to screen capture.

[00074] Preferably, the screen capture uses a screen sharing application, such as Skype, GoToMeeting or WebEx.

[00075] Preferably, the screen capture uses image or video capture applications, such as Window Snipping Tool, print screen or SnagIt.

[00076] Preferably, said system can block the display of said file during screen capture.

[00077] Preferably, said system can inherit metadata from said file being displayed and display in captured image.

[00078] Preferably, said system can inherit metadata from said file being displayed and store as metadata in captured image file.

[00079] Preferably, said manager tests validity of said file metadata by comparing said metadata with allowable ranges of metadata specified in said configuration and takes corrective action if invalid.

[00080] Preferably, said manager uses said configuration to control when said metadata is collected, such as each time said user saves a file through said application,

or only when said file metadata does not exist or is invalid.

[00081] Preferably, said additional thread provides a user interface to said user which allows the user to enter said metadata, whereby the user initiates the interaction via aid user interface.

[00082] Preferably, said user interface is a dialog, or a ribbon bar, or an application menu, or some other user interface control or collection of controls which allows said user to specify information to said system.

[00083] Preferably, said configuration controls the selection ranges presented to said user when collecting said metadata.

[00084] Preferably, said configuration allows for said file metadata which has one or more values which are hierarchical in nature, whereby the selection of one value, say a primary value, controls the valid ranges of further, or secondary values.

[00085] Preferably, said configuration allows for the selection of a minimum security classification, such as SECRET, before any caveat values, such as Releaseability Indicators, or Codewords can be selected.

[00086] Preferably, the selection of one caveat value excludes the selection of other caveats. For example, the selection of an EYES-ONLY caveat will invalidate the any RELEASABILITY caveats.

[00087] Preferably, said system modifies allowable ranges of metadata specified in said configuration, based on content of said file, and rules defined in said configuration.

[00088] Preferably, said additional thread scans said file's content for keywords or numeric sequences, such as credit card numbers, social security numbers, or bank account identifiers, birth dates, or images, or embedded objects, to identify

commercially sensitive or private, or otherwise sensitive information.

[00089] Preferably, on identifying said sensitive information, said system removes invalid options from metadata ranges, which would only allow said user to indicate that said file contains sensitive information.

[00090] Preferably, said configuration is stored as persistent data on said computer, such as in the registry, an XML file, a text file, or binary file, or some other storage which contains configuration data.

[00091] Preferably, said configuration is managed centrally, say using Group Policy, or a configuration server, or a database, or some other system by which settings are managed centrally or in the cloud, and deployed to one or more said computers connected using a computer network.

[00092] Preferably, when a said application writes a said file to said computer system, said File System Metadata Filter Driver scans the associated file stream to check if it contains valid metadata, and if not, notifies said manager component that metadata is missing.

[00093] Preferably, when a said application writes a said file to said computer system, said File System Metadata Filter Driver injects said collected metadata into the file stream.

[00094] Preferably, said files are stored on a hard drive on said computer, or on a removable drive, or on a computer network to which said computer is connected, or cloud storage, or some other system on which files can be stored and retrieved.

[00095] Preferably, said computer network is a local area network, an Intranet, Extranet or the Internet, or some other form of telecommunications network which allows computers to exchange data.

[00096] Preferably, the system further includes system configuration information which control behaviour of said system including nominating computer applications on which the system operates.

[00097] Preferably said user interface component uses a notification area of said operating system.

[00098] Preferably said notification area is the Windows Notification Area (System Tray), or the Status Menu of Apple OS X.

[00099] Preferably, there is provided a non-transitory computer-readable medium coded to implement the system described above.

[000100] Preferably, there is provided a non-transitory computer-readable medium coded to implement the method described above.

[000101] **Brief Description of the drawings**

[000102] Figure 1: System diagram showing major system components;

[000103] Figure 2: Sequence diagram showing how a Windows application can display the metadata of a file it has opened;

[000104] Figure 3: Sequence diagram showing the capture of file metadata when the file is saved;

[000105] Figure 4: Sequence diagram showing the capture of file metadata when activated by the user;

[000106] Figure 5: Modified Notepad.exe application showing file's classification and classification menu;

[000107] Figure 6: Block diagram of a system for enforcing collection and display of computer file metadata in accordance with a further embodiment of the present invention;

[000108] Figure 7 is a Venn Diagram display of types of applications available to an indicative organisation to complete computer assisted tasks.

[000109] **Detailed Description of Preferred Embodiments**

[000110] Referring to Figure 1, which shows system components used to enforce metadata policy on user applications on a desktop computer.

[000111] The user (101) interacts with the computer system (102), with the objective of completing various tasks. Applications (103) are installed on the computer to aid in the completion of those tasks. The user will use the computer system to launch one or more applications.

[000112] Applications will access file data using the file system. The kernel (104) contains various components to interact with the computer system's hardware and to manage operation. One such component is the Metadata Driver (105), which detects when processes are created or closed. Another kernel component is the File System Metadata Filter Driver (106) which detects when files are opened for reading, or saved by applications.

[000113] The Metadata Storage (109) stores file metadata for files on the computer system.

[000114] The Metadata Application Thread (108) is loaded into the Application. The Metadata Application Thread modifies the appearance of the Application on the computer system, to display metadata to the user, relevant to the file which the

Application accesses. It also presents an interactive control to the user, so that the user can specify metadata relevant to the file while using the application.

[000115] The Metadata Manager (107) co-ordinates operations between the components. It receives signals from the Metadata Driver and the File System Metadata Filter Driver, so it can respond to events. It creates the Metadata Application Thread and loads in into the Application. It accesses file metadata from the Metadata Storage. It signals the Metadata Application Thread when new metadata is available.

[000116] Referring to Figure 2, which is a sequence diagram showing how a Windows application can display the metadata of a file it has open.

[000117] The User (101) interacts with the Operating System (102) to launch the application (201). The operating system loads the Application (103, 202). The operating system kernel (104), which is part of the Operating System (102) signals the Metadata Driver (105) that a process has been created (203). It provides the process identifier and application path name as part of the signal information. The Metadata Driver signals the Metadata Manager (107) that the process has been created (204). The Metadata Manager checks that the newly created process should be monitored (205) by comparing the process' application path with known processes to monitor. The set of processes to monitor is part of the Metadata Policy (110).

[000118] The Metadata Manager creates a Metadata Application Thread (108)(206), which it injects into the Application's process (207). The Metadata Manager keeps a reference of the process identifier and the Metadata Application Thread (208).

[000119] At a later stage, the User (101) interacts with the application (103) to open a file (209). The Application makes internal call to open the file (210) which in turn makes a call to the system (102) to open the file (211). The system signals the File System Metadata Filter Driver (106) that a file is being opened (212). The file path and the Applications' process identifier are contained in the information in the signal. The File System Metadata Filter Driver signals the Metadata Manager (107) about the

file activity (213). The Metadata Manager retrieves the reference of the Metadata Application Thread, based on the process identifier from the signal (214). The Metadata Manager retrieves the file metadata (215, 216) from the Metadata Storage (109), based on the file's path that it received in the signal (213).

[000120] The Metadata Manager (107) signals to the Metadata Application Thread (108) that the file open is completed, with the file's metadata (217). The Metadata Application Thread converts the file metadata to text suitable for display to the user (219). It can use configuration from the Metadata Policy to decide which parts of the file metadata to display, how to translate it, and where to locate the display in the Application. The Metadata Application Thread updates the Application's window with the text (220).

[000121] Referring to Figure 3, which shows the flow of control undertaken to capture of file metadata when the file is saved.

[000122] The user (101) interacts with the Application (103) to save the file which is being displayed or edited (301). The Application retrieves the file path from memory (302). Alternatively, the user may specify a new path to which to save the file. The Application makes a call to the System (102) to save the file to the specified path (303). The System notifies the File System Metadata Filter Driver (106) that a file is to be saved, and passes parameters including the file path and the process identifier (304). The File System Metadata Filter Driver notifies the Metadata Manager (107) with the same parameters (305). Using the process identifier, the Metadata Manager retrieves (306) the Metadata Application Thread (108) that was created when the file was opened (206). The Metadata Manager (107) retrieves (307) the file's metadata from Metadata Storage (109), which is returned to the Metadata Manager (308). In some cases, the file may not have metadata (not shown) and no metadata will be returned.

[000123] The Metadata Manager notifies the Metadata Application Thread (108) that the file is being saved (309), and provides the metadata, if it exists. The Metadata Application Thread prompts the user to confirm the metadata (310). The user can

modify the metadata as necessary. If no metadata exists for the file, the user selects valid metadata. The Metadata Application Thread returns the metadata to the Metadata Manager (107). The Metadata Manager writes the metadata to the Metadata Storage (109), indexed by the file path (312).

[000124] Referring to Figure 4, which shows the capture of file metadata when activated by the user.

[000125] The user (101) interacts with the Metadata Application Thread (108) to set the metadata of the file which the Application (103) is displaying (401). The Metadata Application Thread has extended the user interface of the Application with additional controls (see Figure 5). The Metadata Application Thread retrieves from memory the path of the file being displayed (402). The Metadata Application Thread gets from memory the existing metadata for the file (403). The Metadata Application Thread prompts the user with the existing metadata, which the user may modify (404). The Metadata Application Thread notifies the Metadata Manager (107) of changes, and provides the path and the updated metadata (405). The Metadata Manager writes the metadata to the Metadata Storage (109), indexed by the file path (406).

[000126] Referring to Figure 5, which shows the Microsoft Windows Notepad.exe application (text editor), as extended by the invention, operating on Microsoft Windows 10.

[000127] The Application, Notepad.exe, is shown, contained by the Window Border (501),

[000128] The title bar includes the Control Menu box (502) and shows the file name of the file being edited (504), and the application name (505). The title bar has been extended by the addition of a classification control (503).

[000129] The Menu bar (508) shows the options available to the user when using the application.

[000130] The Status bar (509) shows the Application's status. In this example it displays the cursor position.

[000131] The client area of the application window shows the content of the file (507).

[000132] The image also shows how the application window is further extended by the System. The file metadata is displayed (506). In this case the security classification of the file is shown between the title bar and the menu bar.

Further embodiment

[000133] Figure 6 describes a further embodiment of the present invention wherein like components are numbered as for components described in earlier embodiments except in the 1000 series.

[000134] It describes in block diagram form apparatus and methodology to enforce the data policy on use of data stored in a file without modifying the data in the file to which the metadata policy applies.

[000135] Referring to Figure 6, there is shown system components used to enforce metadata policy on user applications on a desktop computer in accordance with a further embodiment of the present invention.

[000136] The user (not shown) interacts with the computer system (1102), with the objective of completing various tasks. Applications (1103) are installed on the computer to aid in the completion of those tasks. The user will use the computer system to launch one or more applications.

[000137] Applications will access file data using the file system. The kernel (1104) contains various components to interact with the computer system's hardware and to manage operation. One such component is the Metadata Driver (1105), which detects when processes are created or closed. Another kernel component is the File System Metadata Filter Driver (1106) which observes when files are opened for reading, or

saved by applications.

[000138] The Metadata Storage (1109) contains file metadata for files including the metadata MD-1 for file F-1 on the computer system. In a preferred form the files including file F-1 are stored in separate file data storage (1110). More preferably the separate file data storage 1110 is logically separate storage from the file metadata storage 1109.

[000139] The Metadata Application Thread (1108) which is executable as an additional thread in the application 1103 is loaded into the Application 1103. The Metadata Application Thread 1108 modifies the appearance, e.g. Application Window, of the user interface (1111) generated by the Application 1103 on the computer system 1102, in order to display metadata including meta data MD-1 to the user, relevant to the file F-1 which the Application accesses. It also presents an interactive control (1112) on the Application's user interface 1111 to the user, so that the user can specify metadata including metadata MD-1 relevant to the file F-1 while using the application 1103.

[000140] The Metadata Manager (1107) co-ordinates operation between the components. It receives signals from the Metadata Driver 1105 and the file access detector in this instance in the form of File System Metadata Filter Driver 1106, so it can respond to events. It creates the Metadata Application Thread 1108 and loads it into the Application 1103. It accesses file metadata including metadata MD-1 from the Metadata Storage 1109. It signals the Metadata Application Thread executing as thread 1108 when new metadata including metadata MD-1 is available.

[000141] In one particular form code to implement the system and method of any of the previously described embodiments is stored on a non-transitory medium such as a memory device. The code can then be loaded into memory 1120 from the non-transitory medium for execution by CPU 1121.

[000142] **Industrial applicability**

[000143] Embodiments of the invention provide data structures and program structures for the handling and the display of metadata including, in particular metadata associated with layered security systems.

CLAIMS

1. A system for displaying and capturing file metadata of an application data file stored on a computer; said computer including a processor and memory; said memory storing an operating system for managing operations of the computer; said memory further storing a kernel of the operating system; said computer further including a user interface and at least one installed application installed on said computer for interacting with a user which, when executed by the processor under the control of the operating system, processes said application data file; said system comprising:
    a. a File System Metadata Filter Driver installed on said computer, whereby said kernel of said operating system signals said File System Metadata Filter Driver when said installed application accesses said application data file;
    b. at least one Metadata Application Thread, which the system injects into said installed application when an application process for execution of said installed application is created, whereby said Metadata Application Thread handles messages generated by said installed application and modifies the user interface of said installed application to display information related to installed application file metadata of said installed application;
    c. Said memory including Metadata Storage which stores said application file metadata, from which said at least one Metadata Application Thread reads said file metadata, and to which said at least one Metadata Application Thread writes said installed application file metadata;
    d. a Metadata Manager operating on said computer, whereby said File System Metadata Filter Driver signals said Metadata Manager when said at least one application data file is accessed, and which in turn notifies said Metadata Application Thread appropriate to that one of said at least one installed application which is accessing said application data file.

2. The system of claim 1 including configuration information which controls behaviour of said system.

3. The system of claim 2 further including said system configuration information which controls behaviour of said system including nominating computer applications on

which the system operates.

4. The system of Claim 1, wherein said system modifies said installed application to capture said file metadata of said application data file from said user when said file is being saved within said installed application.

5. The system of any previous claim wherein said system modifies said user interface of said at least one installed application to allow said user to set metadata appropriate to said application data file.

6. The system of any previous claim wherein said system modifies a title bar displayed in said user interface of said installed application in order to display said installed application file metadata.

7. The system of any previous claim wherein said system modifies said application to display said file metadata in a client window of the user interface of said installed application.

8. The system of any previous claim wherein said system modifies said installed application to overlay said file metadata on a printout when said application data file is printed.

9. The system of any previous claim wherein said operating system is Microsoft Windows, or Apple OSX, a BSD derived operating system, such as FreeBSD or Open BSD, Linux or some other operating system which allows for additional software components which can be signalled when files are accessed.

10. The system of any previous claim wherein said application is Microsoft's notepad.exe, or some other application which can open, display, edit and save text files.

11. The system of any previous claim wherein said application is Microsoft paintbrush.exe, or some other type of application which can open, display, edit and

save image files.

12. The system of any previous claim wherein said application is Microsoft's Word.exe, OpenOffice, LibreOffice, Adobe PageMaker, WPS Office, or some other type of application which can open, display, edit and save document files.

13. The system of any previous claim wherein said installed application is Windows Media Player or some other type of application which plays back audio files, and wherein said system modifies said installed application to play audio cues associated with metadata of said audio files before, during and/or after the playing back of said audio file.

14. The system of any previous claim wherein said application plays back video files, such as Windows Media Player. Said system modifies said application to play video and/or audio cues associated with said file's metadata before, during and/or after the playing back of said file.

15. The system of any previous claim wherein said system modifies said application to display said metadata in the client region of said application during video playback.

16. The system of any previous claim wherein said application is a system application, such as Windows File Explorer, or some other system or third party application which provides file management functionality to said user, such as copying, moving, deleting, renaming, updating or creating files.

17. The system of any previous claim wherein said application is a web browser, such as Internet Explorer, Safari, Chrome, Firefox, or some other software application for retrieving, presenting, and traversing information from a computer network, such as the Internet or an Intranet.

18. The system of any previous claim wherein said file metadata is security classification information.

19. The system of any previous claim wherein said File System Metadata filter Driver is a Windows mini filter driver, or a Max OS X listener, or some additional operating system kernel component which observes when the computer reads or writes files.

20. The system of any previous claim wherein said metadata storage is part of said computer's file system, such as Microsoft's Alternate Data Streams (ADS), or some form of extended file attribute, whereby the file system can associate a file with file metadata.

21. The system of any previous claim wherein said file metadata is stored as properties within said file, such as Microsoft Office's Custom Document Properties, or other file properties which are stored within the file.

22. The system of any previous claim wherein said file metadata is stored on said computer in a database, such as SQLite, BerkelyDB, or some other database installed on said computer, capable of persistent storage of said file metadata.

23. The system of any previous claim wherein said file metadata is stored on a computer connected to said computer by a computer network, such as database server, like Microsoft SQL Server or Oracle, or a cloud service, or some other network storage capable of persistent storage of said file metadata.

24. The system of any previous claim wherein said metadata is stored in a combination of the file system, a local database, a network accessible database, and in said file.

25. The system of any previous claim wherein said application is capable of transferring files, such as Skype, or Dropbox, or Filezilla, or a browser such as Internet Explorer, Google Chrome, Apple Safari, or Mozilla, or some other application which can copy or move a file from said computer to another computer over a computer network. Said system modifies said application to interrogate the metadata of said file prior to it being transferred, and can reject the transfer of a file, based on rules defined in said configuration.

26. The system of any previous claim wherein one such rule which may be enforced by said system is that a security classified file can only be transmitted across network channels, and to destinations, with adequate protection.

27. The system of any previous claim wherein said computer is capable of performing screen capture.

28. The system of any previous claim wherein said system modifies said application to interrogate the metadata of said file being displayed, and if the display is subject to screen capture.

29. The system of claim 27 or 28 wherein the screen capture uses a screen sharing application, such as Skype, GoToMeeting or WebEx.

30. The system of claim 27 or 28 wherein the screen capture uses image or video capture applications, such as Window Snipping Tool, print screen or SnagIt.

31. The system of any one of claims 27 to 30 wherein said system can block the display of said file during screen capture.

32. The system of any one of claims 27 to 30 wherein said system can inherit metadata from said file being displayed and display in captured image.

33. The system of any one of claims 27 to 30 wherein said system can inherit metadata from said file being displayed and store as metadata in captured image file.

34. The system of any previous claim wherein said manager tests validity of said file metadata by comparing said metadata with allowable ranges of metadata specified in said configuration and takes corrective action if invalid.

35. The system of any previous claim wherein said manager uses said configuration to control when said metadata is collected, such as each time said user saves a file

through said application, or only when said file metadata does not exist or is invalid.

36. The system of any previous claim wherein said additional thread provides a user interface to said user which allows the user to enter said metadata, whereby the user initiates the interaction via aid user interface.

37. The system of any previous claim wherein said user interface is a dialog, or a ribbon bar, or an application menu, or some other user interface control or collection of controls which allows said user to specify information to said system.

38. The system of any previous claim wherein said configuration controls the selection ranges presented to said user when collecting said metadata.

39. The system of any previous claim wherein said configuration allows for said file metadata which has one or more values which are hierarchical in nature, whereby the selection of one value, say a primary value, controls the valid ranges of further, or secondary values.

40. The system of any previous claim wherein said configuration allows for the selection of a minimum security classification, such as SECRET, before any caveat values, such as Releaseability Indicators, or Codewords can be selected.

41. The system of any previous claim wherein the selection of one caveat value excludes the selection of other caveats. For example, the selection of an EYES-ONLY caveat will invalidate the any RELEASABILITY caveats.

42. The system of any previous claim wherein said system modifies allowable ranges of metadata specified in said configuration, based on content of said file, and rules defined in said configuration.

43. The system of any previous claim wherein said additional thread scans said file's content for keywords or numeric sequences, such as credit card numbers, social security numbers, or bank account identifiers, birth dates, or images, or embedded

objects, to identify commercially sensitive or private, or otherwise sensitive information.

44. The system of the previous claim 43 wherein on identifying said sensitive information, said system removes invalid options from metadata ranges, which would only allow said user to indicate that said file contains sensitive information.

45. The system of any previous claim wherein said configuration is stored as persistent data on said computer, such as in the registry, an XML file, a text file, or binary file, or some other storage which contains configuration data.

46. The system of any previous claim wherein said configuration is managed centrally, say using Group Policy, or a configuration server, or a database, or some other system by which settings are managed centrally or in the cloud, and deployed to one or more said computers connected using a computer network.

47. The system of any previous claim wherein when a said application writes a said file to said computer system, said File System Metadata Filter Driver scans the associated file stream to check if it contains valid metadata, and if not, notifies said manager component that metadata is missing.

48. The system of any previous claim wherein when a said application writes a said file to said computer system, said File System Metadata Filter Driver injects said collected metadata into the file stream.

49. The system of any previous claim wherein said files are stored on a hard drive on said computer, or on a removable drive, or on a computer network to which said computer is connected, or cloud storage, or some other system on which files can be stored and retrieved.

50. The system of any previous claim wherein said computer network is a local area network, an Intranet, Extranet or the Internet, or some other form of telecommunications network which allows computers to exchange data.

51. The system of any previous claim further including system configuration information which control behaviour of said system including nominating computer applications on which the system operates.

52. A system for displaying and capturing file metadata of an application data file stored on a computer; said computer including a processor and memory; said memory storing an operating system for managing operations of the computer; said memory further storing a kernel of the operating system; said computer further including a user interface and at least one installed application installed on said computer for interacting with a user which, when executed by the processor under the control of the operating system, processes said application data file; and wherein said application has no function to capture custom metadata for said file; said system comprising:
    a. a File System Metadata Filter Driver installed on said computer, whereby said kernel of said operating system triggers said File System Metadata Filter Driver when said installed application accesses said application data file;
    b. a user interface component in the form of a Metadata Application Thread to display information related to said file metadata to the user of said computer; and captures file metadata information from said user;
    c. metadata storage which stores said file metadata and wherein said metadata storage is persistent;
    d. a Metadata Manager operating on said computer, whereby said File System Metadata Filter Driver signals said Metadata Manager when said application data files are accessed, and which in turn notifies said user interface component;

53. The system of claim 52 further including configuration information which controls behaviour of said system.

54. The system of claim 51, wherein said user interface component uses a notification area of said operating system.

55. The system of claim 54 wherein said notification area is the Windows Notification Area (System Tray), or the Status Menu of Apple OS X.

56. A system for displaying and capturing file metadata of an application data file stored on a computer; said computer including a processor and memory; said memory storing an operating system for managing operations of the computer; said memory further storing a kernel of the operating system; said computer further including a user interface and at least one installed application installed on said computer for interacting with a user which, when executed by the processor under the control of the operating system, processes said application data file; said system further comprising:
    a. Metadata Storage which stores said file metadata, whereby said storage is persistent;
    b. at least one Metadata Application Thread in each said installed application, whereby said Metadata Application Thread intercepts file operation calls from the installed application; handles said messages of said installed application; modifies the user interface of said installed application to display information related to said file metadata to the user of said computer; captures file metadata information from said user; and creates, updates, deletes and otherwise accesses said file metadata from said Metadata Storage;

57. The system of claim 56 further including configuration information which controls behaviour of said system.

58. A system to enforce metadata policy on use of data stored in a file without modifying the data in the file to which the meta data policy applies;
said system including a system for displaying and capturing file metadata of an application data file stored on a computer; said computer including a processor and memory; said memory storing an operating system for managing operations of the computer; said memory further storing a kernel of the operating system; said computer further including a user interface and at least one installed application installed on said computer for interacting with a user which, when executed by the processor under the control of the operating system, processes said application data file; said system comprising:

a. a File System Metadata Filter Driver installed on said computer, whereby said kernel of said operating system signals said File System Metadata Filter Driver when said installed application accesses said application data file;
b. at least one Metadata Application Thread, which the system injects into said installed application when an application process for execution of said installed application is created, whereby said Metadata Application Thread handles messages generated by said installed application and modifies the user interface of said installed application to display information related to installed application file metadata of said installed application;
c. Said memory including Metadata Storage which stores said application file metadata, from which said at least one Metadata Application Thread reads said file metadata, and to which said at least one Metadata Application Thread writes said installed application file metadata;
d. a Metadata Manager operating on said computer, whereby said File System Metadata Filter Driver signals said Metadata Manager when said at least one application data file is accessed, and which in turn notifies said Metadata Application Thread appropriate to that one of said at least one installed application which is accessing said application data file.

59. A system of organization of programme structures and file structures in a computer comprising a processor in communication with a memory in order to enforce metadata policy on use of data stored in a file without modifying the data in the file to which the meta data policy applies;
said system including a system for displaying and capturing file metadata of an application data file stored on the computer; said computer including said processor and said memory; said memory storing an operating system for managing operations of the computer; said memory further storing a kernel of the operating system; said computer further including a user interface and at least one installed application installed on said computer for interacting with a user which, when executed by the processor under the control of the operating system, processes said application data file; said system comprising:

a. a File System Metadata Filter Driver installed on said computer, whereby said kernel of said operating system signals said File System Metadata Filter Driver when said installed application accesses said application data file;
b. at least one Metadata Application Thread, which the system injects into said installed application when an application process for execution of said installed application is created, whereby said Metadata Application Thread handles messages generated by said installed application and modifies the user interface of said installed application to display information related to installed application file metadata of said installed application;
c. Said memory including Metadata Storage which stores said application file metadata, from which said at least one Metadata Application Thread reads said file metadata, and to which said at least one Metadata Application Thread writes said installed application file metadata;
d. a Metadata Manager operating on said computer, whereby said File System Metadata Filter Driver signals said Metadata Manager when said at least one application data file is accessed, and which in turn notifies said Metadata Application Thread appropriate to that one of said at least one installed application which is accessing said application data file.

60. A method for displaying and capturing file metadata of an application data file stored on a computer; said computer including a processor and memory; said memory storing an operating system for managing operations of the computer; said memory further storing a kernel of the operating system; said computer further including a user interface and at least one installed application installed on said computer for interacting with a user which, when executed by the processor under the control of the operating system, processes said application data file; said method comprising:
a. installing a File System Metadata Filter Driver on said computer, whereby said kernel of said operating system signals said File System Metadata Filter Driver when said installed application accesses said application data file;
b. injecting at least one Metadata Application Thread into said installed application when an application process for execution of said installed application is created, whereby said Metadata Application Thread handles messages generated by said installed application and modifies the user

interface of said installed application to display information related to installed application file metadata of said installed application;

c. providing Metadata Storage in said memory which stores said application file metadata, from which said at least one Metadata Application Thread reads said file metadata, and to which said at least one Metadata Application Thread writes said installed application file metadata;

d. providing a Metadata Manager operating on said computer, whereby said File System Metadata Filter Driver signals said Metadata Manager when said at least one application data file is accessed, and which in turn notifies said Metadata Application Thread appropriate to that one of said at least one installed application which is accessing said application data file.

61. A non-transitory computer-readable medium coded to implement the system of any one of claims 1 to 59.

62. A non-transitory computer-readable medium coded to implement the method of claim 60.

AMENDED CLAIMS
received by the International Bureau on 13 March 2018 (13.03.2018)

CLAIMS

1. A system for displaying and capturing file metadata of an application data file stored on a computer; said computer including a processor and memory; said memory storing an operating system for managing operations of the computer; said memory further storing a kernel of the operating system; said computer further including a user interface and at least one installed application installed on said computer for interacting with a user which, when executed by the processor under the control of the operating system, processes said application data file; said system comprising:
    a. a File System Metadata Filter Driver installed on said computer, whereby said kernel of said operating system signals said File System Metadata Filter Driver when said installed application accesses said application data file;
    b. at least one Metadata Application Thread, which the system injects into said installed application when an application process for execution of said installed application is created, whereby said Metadata Application Thread handles messages generated by said installed application and modifies the user interface of said installed application to display information related to installed application file metadata of said installed application;
    c. Said memory including Metadata Storage which stores said application file metadata, from which said at least one Metadata Application Thread reads said file metadata, and to which said at least one Metadata Application Thread writes said installed application file metadata;
    d. a Metadata Manager operating on said computer, whereby said File System Metadata Filter Driver signals said Metadata Manager when said at least one application data file is accessed, and which in turn notifies said Metadata Application Thread appropriate to that one of said at least one installed application which is accessing said application data file.

2. The system of claim 1 including configuration information which controls behaviour of said system.

3. The system of claim 2 further including said system configuration information which controls behaviour of said system including nominating computer applications on which the system operates.

4. The system of Claim 1, wherein said system modifies said installed application to capture said file metadata of said application data file from said user when said file is being saved within said installed application.

5. The system of any previous claim wherein said system modifies said user interface of said at least one installed application to allow said user to set metadata appropriate to said application data file.

6. The system of any previous claim wherein said system modifies a title bar displayed in said user interface of said installed application in order to display said installed application file metadata.

7. The system of any previous claim wherein said system modifies said application to display said file metadata in a client window of the user interface of said installed application.

8. The system of any previous claim wherein said system modifies said installed application to overlay said file metadata on a printout when said application data file is printed.

9. The system of any previous claim wherein said operating system is Microsoft Windows, or Apple OSX, a BSD derived operating system, such as FreeBSD or Open BSD, Linux or some other operating system which allows for additional software components which can be signalled when files are accessed.

10. The system of any previous claim wherein said application is Microsoft's notepad.exe, or some other application which can open, display, edit and save text files.

11. The system of any previous claim wherein said application is Microsoft paintbrush.exe, or some other type of application which can open, display, edit and save image files.

12. The system of any previous claim wherein said application is Microsoft's Word.exe, OpenOffice, LibreOffice, Adobe PageMaker, WPS Office, or some other type of application which can open, display, edit and save document files.

13. The system of any previous claim wherein said installed application is Windows Media Player or some other type of application which plays back audio files, and wherein said system modifies said installed application to play audio cues associated with metadata of said audio files before, during and/or after the playing back of said audio file.

14. The system of any previous claim wherein said application plays back video files, such as Windows Media Player. Said system modifies said application to play video and/or audio cues associated with said file's metadata before, during and/or after the playing back of said file.

15. The system of any previous claim wherein said system modifies said application to display said metadata in the client region of said application during video playback.

16. The system of any previous claim wherein said application is a system application, such as Windows File Explorer, or some other system or third party application which provides file management functionality to said user, such as copying, moving, deleting, renaming, updating or creating files.

17. The system of any previous claim wherein said application is a web browser, such as Internet Explorer, Safari, Chrome, Firefox, or some other software application for retrieving, presenting, and traversing information from a computer network, such as the Internet or an Intranet.

18. The system of any previous claim wherein said file metadata is security classification information.

19. The system of any previous claim wherein said File System Metadata filter Driver is a Windows mini filter driver, or a Max OS X listener, or some additional operating system kernel component which observes when the computer reads or writes files.

20. The system of any previous claim wherein said metadata storage is part of said computer's file system, such as Microsoft's Alternate Data Streams (ADS), or some form of extended file attribute, whereby the file system can associate a file with file metadata.

21. The system of any previous claim wherein said file metadata is stored as properties within said file, such as Microsoft Office's Custom Document Properties, or other file properties which are stored within the file.

22. The system of any previous claim wherein said file metadata is stored on said computer in a database, such as SQLite, BerkelyDB, or some other database installed on said computer, capable of persistent storage of said file metadata.

23. The system of any previous claim wherein said file metadata is stored on a computer connected to said computer by a computer network, such as database server, like Microsoft SQL Server or Oracle, or a cloud service, or some other network storage capable of persistent storage of said file metadata.

24. The system of any previous claim wherein said metadata is stored in a combination of the file system, a local database, a network accessible database, and in said file.

25. The system of any previous claim wherein said application is capable of transferring files, such as Skype, or Dropbox, or Filezilla, or a browser such as Internet Explorer, Google Chrome, Apple Safari, or Mozilla, or some other application which can copy or move a file from said computer to another computer over a computer network. Said system modifies said application to interrogate the metadata of said file prior to it

being transferred, and can reject the transfer of a file, based on rules defined in said configuration.

26. The system of any previous claim wherein one such rule which may be enforced by said system is that a security classified file can only be transmitted across network channels, and to destinations, with adequate protection.

27. The system of any previous claim wherein said computer is capable of performing screen capture.

28. The system of any previous claim wherein said system modifies said application to interrogate the metadata of said file being displayed, and if the display is subject to screen capture.

29. The system of claim 27 or 28 wherein the screen capture uses a screen sharing application, such as Skype, GoToMeeting or WebEx.

30. The system of claim 27 or 28 wherein the screen capture uses image or video capture applications, such as Window Snipping Tool, print screen or SnagIt.

31. The system of any one of claims 27 to 30 wherein said system can block the display of said file during screen capture.

32. The system of any one of claims 27 to 30 wherein said system can inherit metadata from said file being displayed and display in captured image.

33. The system of any one of claims 27 to 30 wherein said system can inherit metadata from said file being displayed and store as metadata in captured image file.

34. The system of any previous claim wherein said manager tests validity of said file metadata by comparing said metadata with allowable ranges of metadata specified in said configuration and takes corrective action if invalid.

35. The system of any previous claim wherein said manager uses said configuration to control when said metadata is collected, such as each time said user saves a file through said application, or only when said file metadata does not exist or is invalid.

36. The system of any previous claim wherein said additional thread provides a user interface to said user which allows the user to enter said metadata, whereby the user initiates the interaction via aid user interface.

37. The system of any previous claim wherein said user interface is a dialog, or a ribbon bar, or an application menu, or some other user interface control or collection of controls which allows said user to specify information to said system.

38. The system of any previous claim wherein said configuration controls the selection ranges presented to said user when collecting said metadata.

39. The system of any previous claim wherein said configuration allows for said file metadata which has one or more values which are hierarchical in nature, whereby the selection of one value, say a primary value, controls the valid ranges of further, or secondary values.

40. The system of any previous claim wherein said configuration allows for the selection of a minimum security classification, such as SECRET, before any caveat values, such as Releaseability Indicators, or Codewords can be selected.

41. The system of any previous claim wherein the selection of one caveat value excludes the selection of other caveats. For example, the selection of an EYES-ONLY caveat will invalidate the any RELEASABILITY caveats.

42. The system of any previous claim wherein said system modifies allowable ranges of metadata specified in said configuration, based on content of said file, and rules defined in said configuration.

43. The system of any previous claim wherein said additional thread scans said file's content for keywords or numeric sequences, such as credit card numbers, social security numbers, or bank account identifiers, birth dates, or images, or embedded objects, to identify commercially sensitive or private, or otherwise sensitive information.

44. The system of the previous claim 43 wherein on identifying said sensitive information, said system removes invalid options from metadata ranges, which would only allow said user to indicate that said file contains sensitive information.

45. The system of any previous claim wherein said configuration is stored as persistent data on said computer, such as in the registry, an XML file, a text file, or binary file, or some other storage which contains configuration data.

46. The system of any previous claim wherein said configuration is managed centrally, say using Group Policy, or a configuration server, or a database, or some other system by which settings are managed centrally or in the cloud, and deployed to one or more said computers connected using a computer network.

47. The system of any previous claim wherein when a said application writes a said file to said computer system, said File System Metadata Filter Driver scans the associated file stream to check if it contains valid metadata, and if not, notifies said manager component that metadata is missing.

48. The system of any previous claim wherein when a said application writes a said file to said computer system, said File System Metadata Filter Driver injects said collected metadata into the file stream.

49. The system of any previous claim wherein said files are stored on a hard drive on said computer, or on a removable drive, or on a computer network to which said computer is connected, or cloud storage, or some other system on which files can be stored and retrieved.

50. The system of any previous claim wherein said computer network is a local area network, an Intranet, Extranet or the Internet, or some other form of telecommunications network which allows computers to exchange data.
51. The system of any previous claim further including system configuration information which control behaviour of said system including nominating computer applications on which the system operates.

52. A system for displaying and capturing file metadata of an application data file stored on a computer; said computer including a processor and memory; said memory storing an operating system for managing operations of the computer; said memory further storing a kernel of the operating system; said computer further including a user interface and at least one installed application installed on said computer for interacting with a user which, when executed by the processor under the control of the operating system, processes said application data file; and wherein said application has no function to capture custom metadata for said file; said system comprising:
    a. a File System Metadata Filter Driver installed on said computer, whereby said kernel of said operating system triggers said File System Metadata Filter Driver when said installed application accesses said application data file;
    b. a user interface component in the form of a Metadata Application Thread to display information related to said file metadata to the user of said computer; and captures file metadata information from said user;
    c. metadata storage which stores said file metadata and wherein said metadata storage is persistent;
    d. a Metadata Manager operating on said computer, whereby said File System Metadata Filter Driver signals said Metadata Manager when said application data files are accessed, and which in turn notifies said user interface component;

53. The system of claim 52 further including configuration information which controls behaviour of said system.

54. The system of claim 51, wherein said user interface component uses a notification area of said operating system.

55. The system of claim 54 wherein said notification area is the Windows Notification Area (System Tray), or the Status Menu of Apple OS X.

56. A system for displaying and capturing file metadata of an application data file stored on a computer; said computer including a processor and memory; said memory storing an operating system for managing operations of the computer; said memory further storing a kernel of the operating system; said computer further including a user interface and at least one installed application installed on said computer for interacting with a user which, when executed by the processor under the control of the operating system, processes said application data file; said system further comprising:
    a. Metadata Storage which stores said file metadata, whereby said storage is persistent;
    b. at least one Metadata Application Thread in each said installed application, whereby said Metadata Application Thread intercepts file operation calls from the installed application; handles said messages of said installed application; modifies the user interface of said installed application to display information related to said file metadata to the user of said computer; captures file metadata information from said user; and creates, updates, deletes and otherwise accesses said file metadata from said Metadata Storage;

57. The system of claim 56 further including configuration information which controls behaviour of said system.

58. A system to enforce metadata policy on use of data stored in a file without modifying the data in the file to which the meta data policy applies;
said system including a system for displaying and capturing file metadata of an application data file stored on a computer; said computer including a processor and memory; said memory storing an operating system for managing operations of the computer; said memory further storing a kernel of the operating system; said computer further including a user interface and at least one installed application installed on said computer for interacting with a user which, when executed by the

processor under the control of the operating system, processes said application data file; said system comprising:

a. a File System Metadata Filter Driver installed on said computer, whereby said kernel of said operating system signals said File System Metadata Filter Driver when said installed application accesses said application data file;
b. at least one Metadata Application Thread, which the system injects into said installed application when an application process for execution of said installed application is created, whereby said Metadata Application Thread handles messages generated by said installed application and modifies the user interface of said installed application to display information related to installed application file metadata of said installed application;
c. Said memory including Metadata Storage which stores said application file metadata, from which said at least one Metadata Application Thread reads said file metadata, and to which said at least one Metadata Application Thread writes said installed application file metadata;
d. a Metadata Manager operating on said computer, whereby said File System Metadata Filter Driver signals said Metadata Manager when said at least one application data file is accessed, and which in turn notifies said Metadata Application Thread appropriate to that one of said at least one installed application which is accessing said application data file.

59. A system of organization of programme structures and file structures in a computer comprising a processor in communication with a memory in order to enforce metadata policy on use of data stored in a file without modifying the data in the file to which the meta data policy applies;
said system including a system for displaying and capturing file metadata of an application data file stored on the computer; said computer including said processor and said memory; said memory storing an operating system for managing operations of the computer; said memory further storing a kernel of the operating system; said computer further including a user interface and at least one installed application installed on said computer for interacting with a user which, when executed by the processor under the control of the operating system, processes said application data file; said system comprising:

a. a File System Metadata Filter Driver installed on said computer, whereby said kernel of said operating system signals said File System Metadata Filter Driver when said installed application accesses said application data file;
b. at least one Metadata Application Thread, which the system injects into said installed application when an application process for execution of said installed application is created, whereby said Metadata Application Thread handles messages generated by said installed application and modifies the user interface of said installed application to display information related to installed application file metadata of said installed application;
c. Said memory including Metadata Storage which stores said application file metadata, from which said at least one Metadata Application Thread reads said file metadata, and to which said at least one Metadata Application Thread writes said installed application file metadata;
d. a Metadata Manager operating on said computer, whereby said File System Metadata Filter Driver signals said Metadata Manager when said at least one application data file is accessed, and which in turn notifies said Metadata Application Thread appropriate to that one of said at least one installed application which is accessing said application data file.

60. The system of any previous claim wherein said File System Metadata filter Driver is a Windows mini filter driver, or a Max OS X listener, or some additional operating system component which observes when the computer reads or writes files.

61. The system of any previous claim wherein said File System Metadata filter Driver is an additional operating system component which observes when the computer reads or writes files.

62. A method for displaying and capturing file metadata of an application data file stored on a computer; said computer including a processor and memory; said memory storing an operating system for managing operations of the computer; said memory further storing a kernel of the operating system; said computer further including a user interface and at least one installed application installed on said computer for

interacting with a user which, when executed by the processor under the control of the operating system, processes said application data file; said method comprising:

a. installing a File System Metadata Filter Driver on said computer, whereby said kernel of said operating system signals said File System Metadata Filter Driver when said installed application accesses said application data file;
b. injecting at least one Metadata Application Thread into said installed application when an application process for execution of said installed application is created, whereby said Metadata Application Thread handles messages generated by said installed application and modifies the user interface of said installed application to display information related to installed application file metadata of said installed application;
c. providing Metadata Storage in said memory which stores said application file metadata, from which said at least one Metadata Application Thread reads said file metadata, and to which said at least one Metadata Application Thread writes said installed application file metadata;
d. providing a Metadata Manager operating on said computer, whereby said File System Metadata Filter Driver signals said Metadata Manager when said at least one application data file is accessed, and which in turn notifies said Metadata Application Thread appropriate to that one of said at least one installed application which is accessing said application data file.

63. A non-transitory computer-readable medium coded to implement the system of any one of claims 1 to 61.

64. A non-transitory computer-readable medium coded to implement the method of claim62.

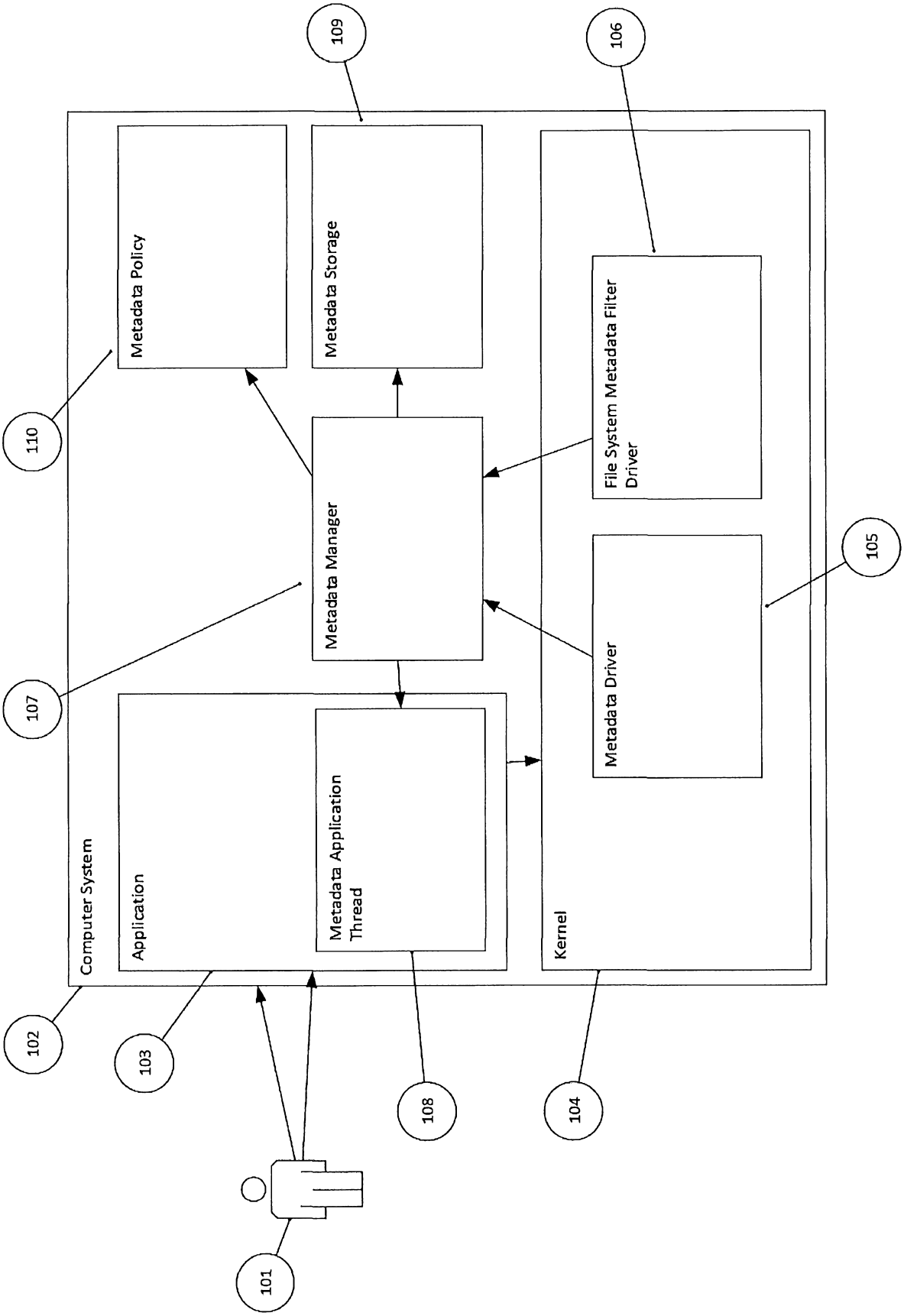

109
106
Metadata Policy
Metadata Storage
110
File System Metadata Filter Driver
Metadata Manager
105
107
Metadata Driver
Application
Metadata Application Thread
Kernel
Computer System
102
103
108
104
101

Fig 1

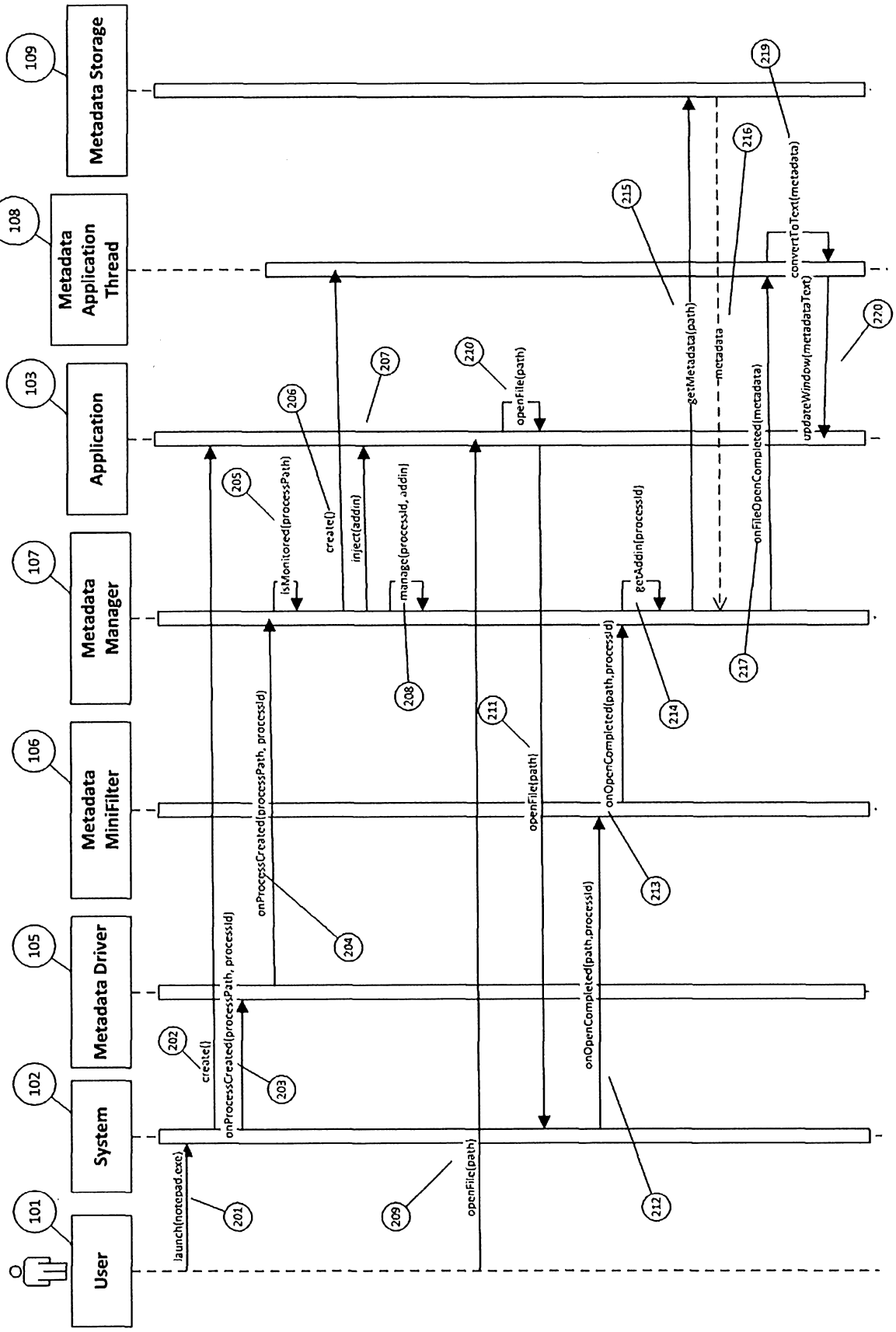

Metadata Storage
Metadata Application Thread
Application
Metadata Manager
Metadata MiniFilter
Metadata Driver
System
User
launch(notepad.exe)
create()
onProcessCreated(processPath, processId)
onProcessCreated(processPath, processId)
isMonitored(processPath)
create()
inject(addin)
manage(processId, addin)
openFile(path)
openFile(path)
getAddin(processId)
onOpenCompleted(path,processId)
onOpenCompleted(path,processId)
getMetadata(path)
metadata
onFileOpenCompleted(metadata)
convertToText(metadata)
updateWindow(metadataText)
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
219
220

Fig 2

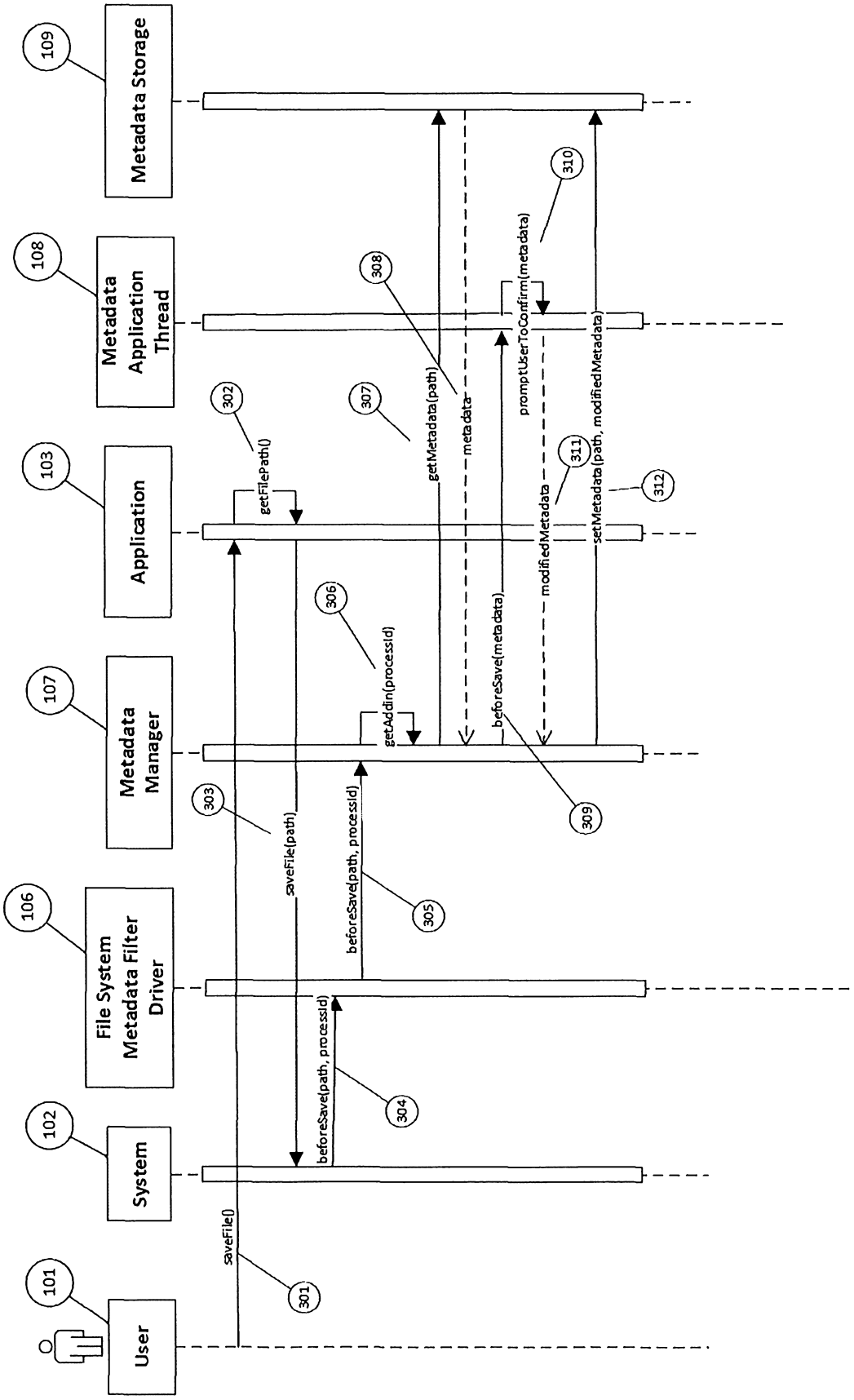

109
Metadata Storage
108
Metadata Application Thread
103
Application
107
Metadata Manager
106
File System Metadata Filter Driver
102
System
101
User
saveFile()
301
getFilePath()
302
saveFile(path)
303
beforeSave(path, processId)
304
beforeSave(path, processId)
305
getAddin(processId)
306
getMetadata(path)
307
metadata
308
beforeSave(metadata)
309
promptUserToConfirm(metadata)
310
modifiedMetadata
311
setMetadata(path, modifiedMetadata)
312

Fig 3

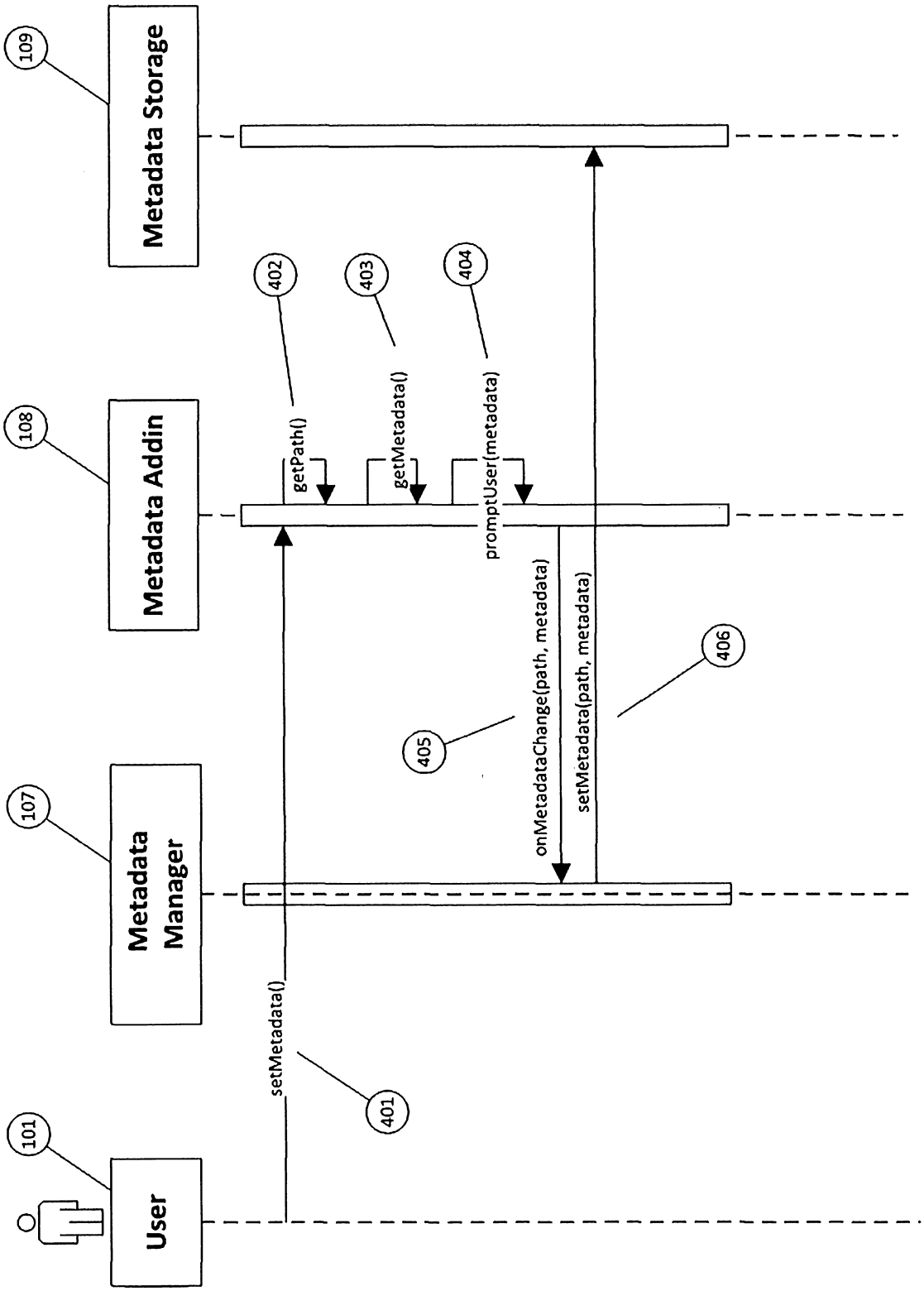

109
Metadata Storage
108
Metadata Addin
107
Metadata Manager
101
User
402
403
404
getPath()
getMetadata()
promptUser(metadata)
onMetadataChange(path, metadata)
setMetadata(path, metadata)
406
405
setMetadata()
401

Fig 4

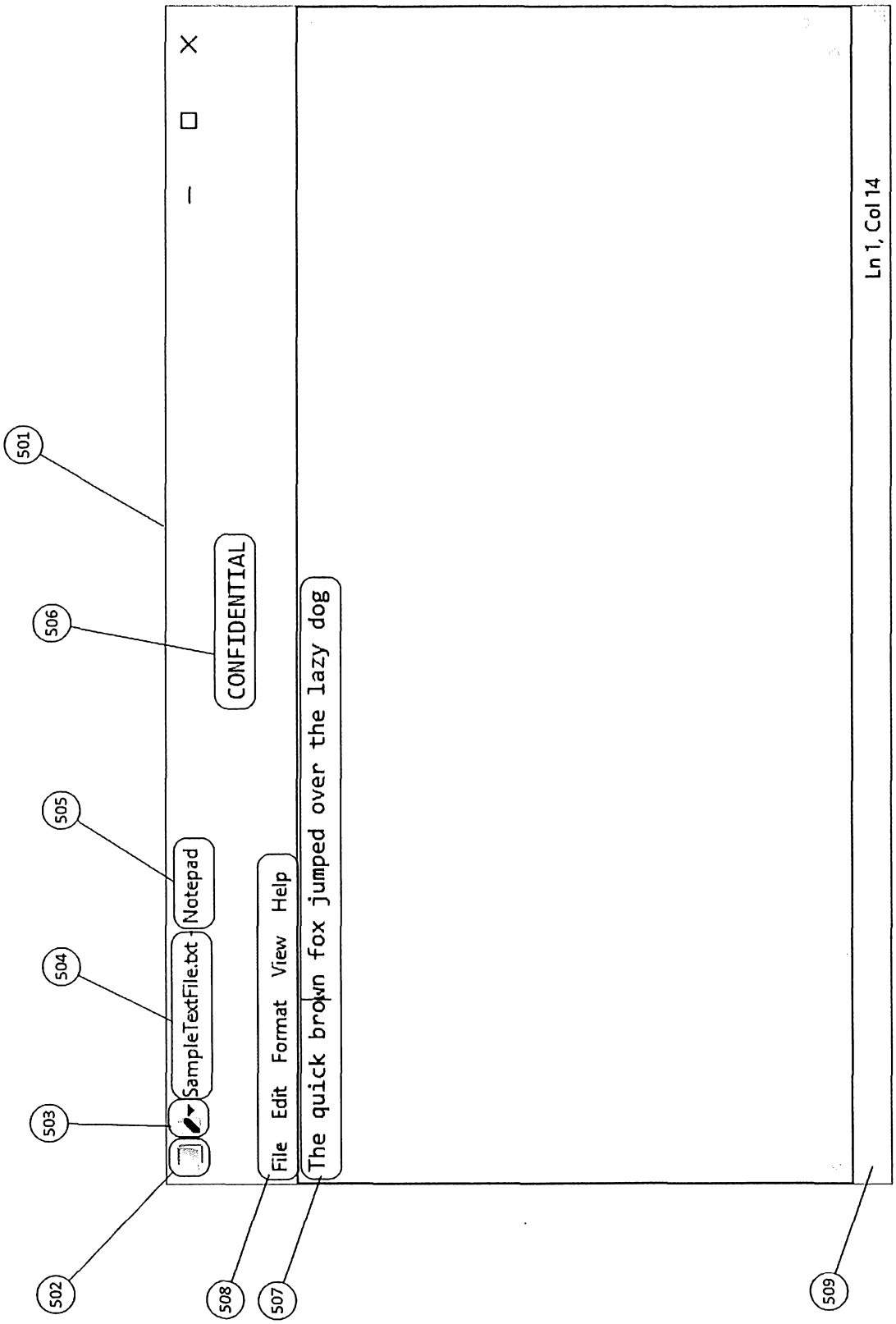

501
CONFIDENTIAL
506
505
Notepad
504
SampleTextFile.txt
503
502
File Edit Format View Help
The quick brown fox jumped over the lazy dog
508
507
Ln 1, Col 14
509

Fig 5

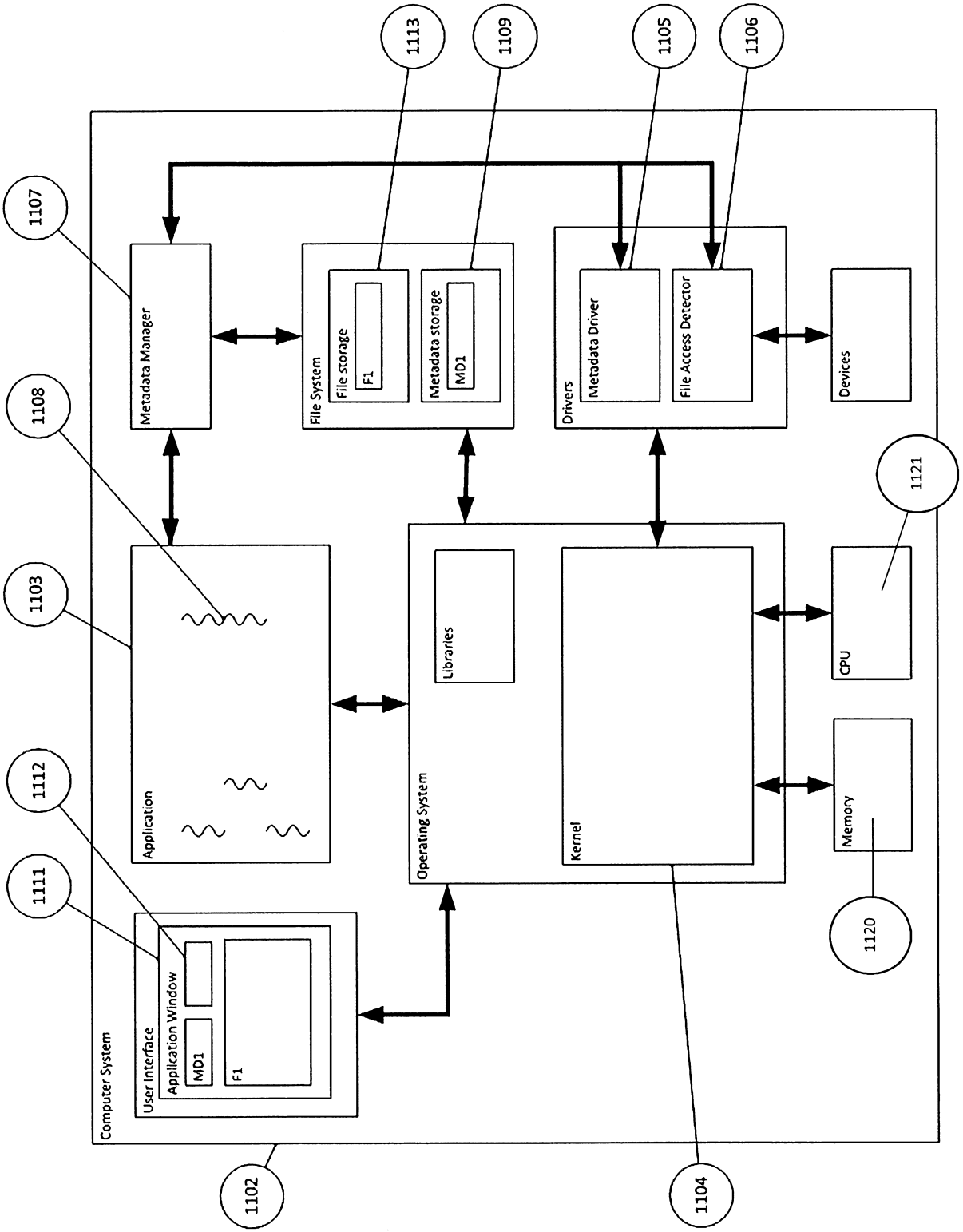

1113
1109
1105
1106
1107
1108
1103
1112
1111
1102
1104
1121
1120
Metadata Manager
File System
File storage
F1
Metadata storage
MD1
Drivers
Metadata Driver
File Access Detector
Devices
Application
Libraries
Operating System
Kernel
CPU
Memory
Computer System
User Interface
Application Window
MD1
F1

Fig 6

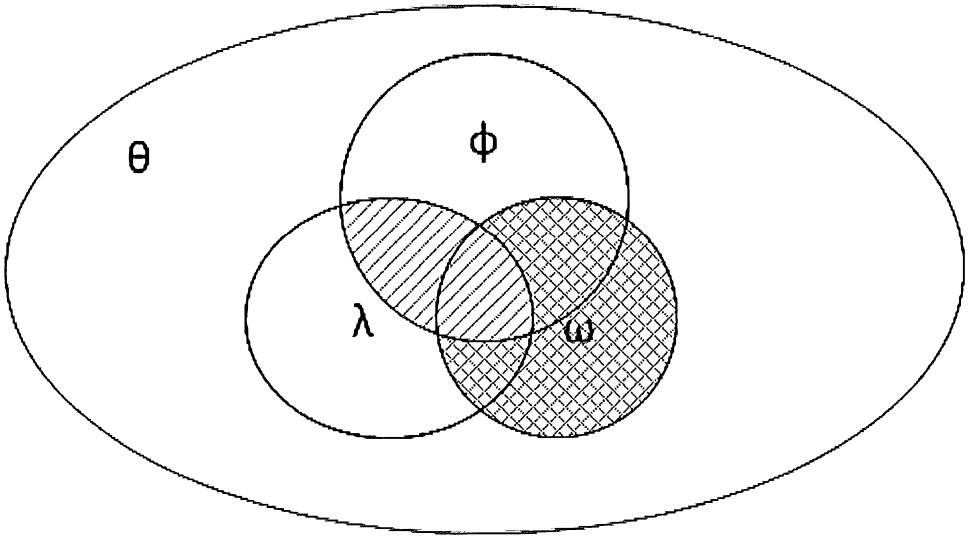


Fig 7