US 20080320607A1

(19) United States

(12) Patent Application Publication

(10) Pub. No.: US 2008/0320607 A1

(43) Pub. Date: Dec. 25, 2008

Richardson

(54) **SYSTEM AND METHOD FOR AUDITING SOFTWARE USAGE**

(75) Inventor: **Ric B. Richardson**, Irvine, CA (US)

Correspondence Address:
CONNOLLY BOVE LODGE & HUTZ LLP
P.O. BOX 2207
WILMINGTON, DE 19899 (US)

(73) Assignee: **UNILOC USA**, Irvine, CA (US)

(21) Appl. No.: **12/140,917**

(22) Filed: **Jun. 17, 2008**

Related U.S. Application Data

(60) Provisional application No. 60/945,359, filed on Jun. 21, 2007.

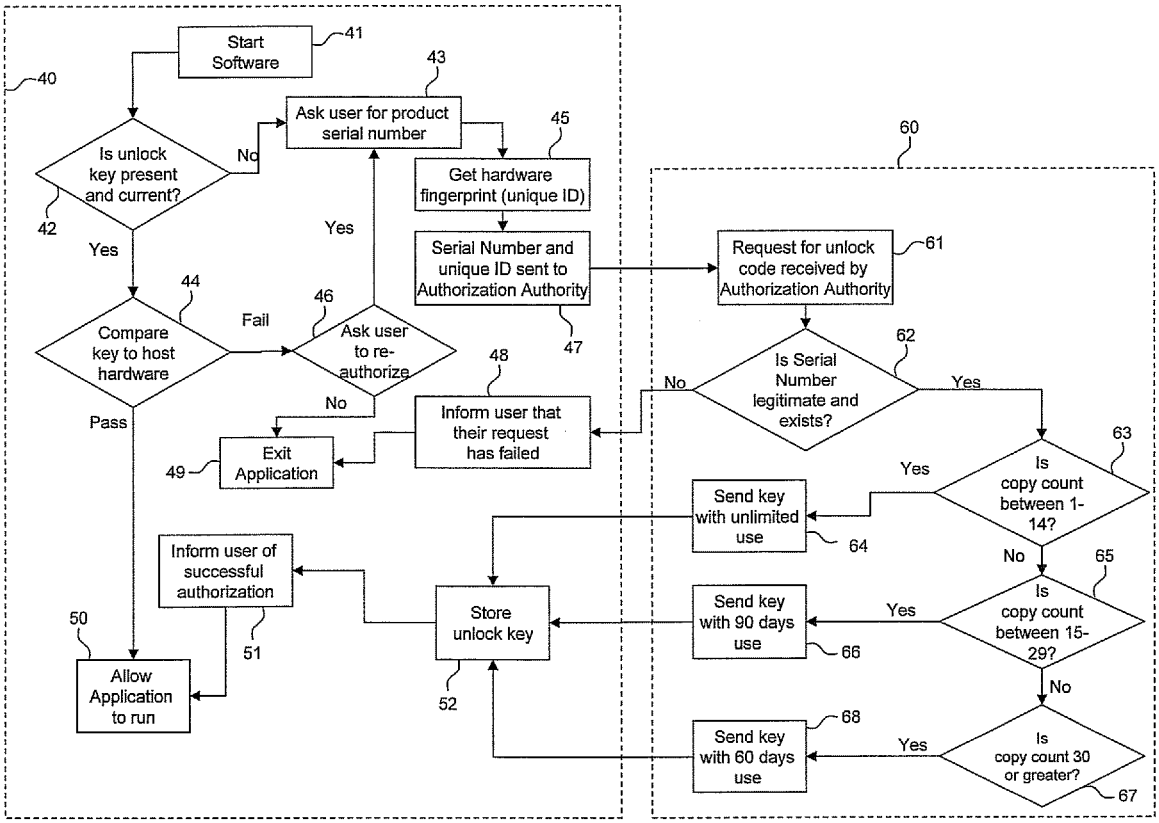
Publication Classification

(51) **Int. Cl.**
G06F 21/22 (2006.01)

(52) **U.S. Cl.** **726/33**

(57) **ABSTRACT**

Systems and methods are provided for auditing and selectively restricting software usage based on, for example, software copy counts or execution counts. In one embodiment, the method comprises verifying whether the serial number for a software installed on a computing device corresponds to one of recognized serial numbers, and calculating a copy count (or software execution count) for the serial number. In response to the copy count exceeding a defined upper limit, a limited unlock key may be sent to the device. The limited unlock key may allow the software to be executed on the device for a defined time period, a defined number of executions, and/or with at least one feature of the software disabled.



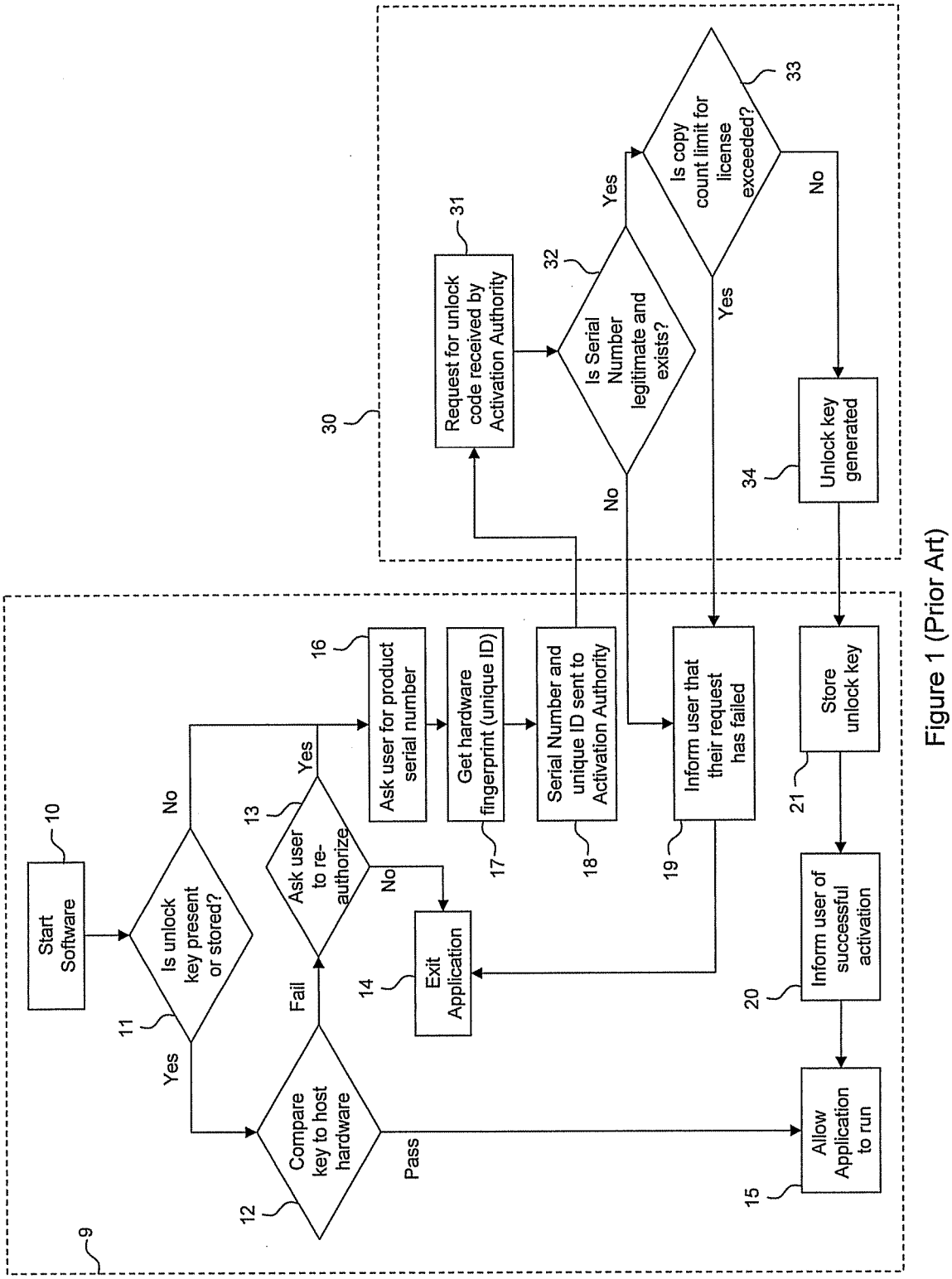


Figure 1 (Prior Art)

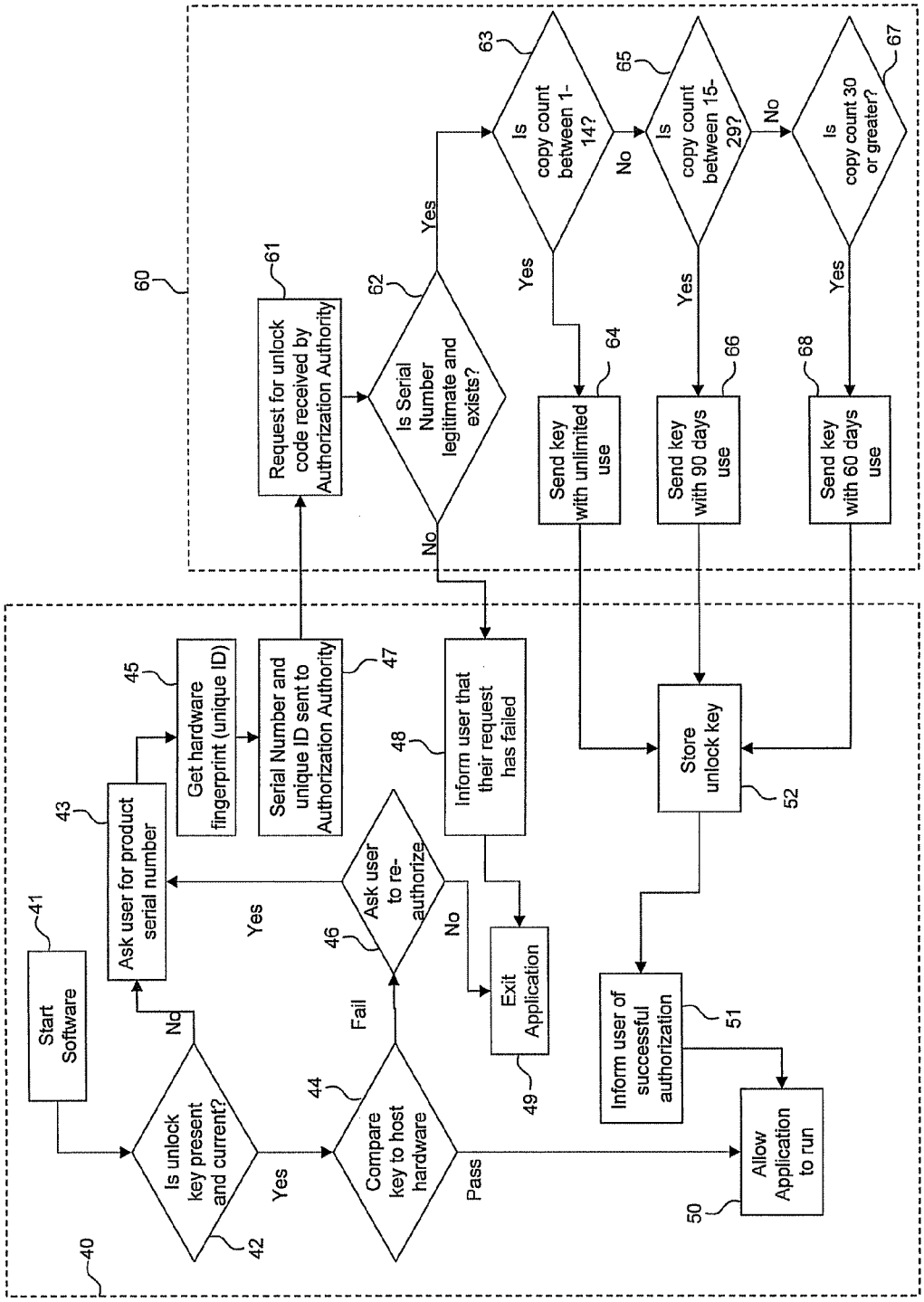


Figure 2

SYSTEM AND METHOD FOR AUDITING SOFTWARE USAGE

CROSS-REFERENCE TO RELATED APPLICATION(S)

[0001] This application claims priority pursuant to 35 U.S. C. §119(e) to U.S. Provisional Application No. 60/945,359, filed Jun. 21, 2007, which application is specifically incorporated herein, in its entirety, by reference.

BACKGROUND OF THE INVENTION

[0002] 1. Field of the Invention

[0003] The present invention is directed toward systems for auditing and restricting software usage, and related methods.

[0004] 2. Description of the Related Art

[0005] Many systems for the protection of software products against piracy and abuse of copyright exist today. Popular approaches described in U.S. Pat. No. 5,490,216 and U.S. Pat. No. 6,243,468 link the license of the user of the software to a specific hardware platform by devising a unique identifier from the measurable characteristics, settings and identifiers already present within the computing hardware and its attached peripherals. With the above described method, the protected software communicates with an authorization authority, usually an Internet based online server, controlled by the software publisher. This communication is needed to ensure that the licensed party does not exceed the usage rights of the license that has been granted by the publisher.

[0006] The systems used in the art keep a record of how many devices have been authorized to run against each license, and when a certain predefined limit has been exceeded, the authorization server denies the software users request to run additional copies of the software on additional devices. For example, a publisher might allow five copies to be made and used of their copyrighted software for each user license sold. The first five requests made to the authorization authority may be allowed however the sixth and subsequent requests would be denied. Possibly the licensee would be encouraged to acquire an additional license.

[0007] This system has drawn criticism from software buyers since many users expect to be able to use software they have purchased on as many devices as they want as long as they own and use the devices. Additionally software buyers are changing, upgrading and replacing their computing devices on a more regular basis as people use computers more and more. This in turn requires additional flexibility on the part of the authorization authority to compensate for reasonable fluctuations in the usage circumstances of users protected under laws such as the Fair Use Act.

[0008] Software buyers may be dissatisfied with current authorization systems due to their inability to determine the difference between legitimate users, that may reasonably require a large number of copies of software for use on their own computing devices, and illegal copies made by pirates and or others who willfully abuse license terms by making indiscriminate copies for other users such as sharing over peer to peer software distribution networks.

to allow/disallow the running of licensed software over the complete life and usage of the software product.

SUMMARY OF THE INVENTION

[0010] In accordance with one aspect of the embodiments described herein, there is provided a method for auditing software usage, comprising: (a) receiving a serial number for a software installed on a computing device and a device identifier for the device; (b) determining whether the received serial number corresponds to one of recognized serial numbers; (c) in response to determination that the received serial number corresponds to one of the recognized serial numbers, calculating a copy count (or software execution count) for the received serial number; (d) in response to the copy count not exceeding a first upper limit of software copies (or software executions), sending an unlimited unlock key to the device; and (e) in response to the copy count being greater than the first preset number but not exceeding a second upper limit, sending a first limited unlock key to the device.

[0011] In accordance with another aspect of the embodiments described herein, there is provided a method for auditing software usage, comprising: (a) sending a serial number for a software installed on a computing device and a device identifier for the device to an authorization authority, the authorization authority calculating a copy count for the serial number (or software execution count); (b) in response to the copy count not exceeding a first upper limit of software copies (or software executions), receiving from the authorization authority an unlimited unlock key for the software; and (c) in response to the copy count being greater than the first upper limit but not exceeding a second upper limit, receiving from the authorization authority a first limited unlock key for the software.

BRIEF DESCRIPTION OF THE DRAWINGS

[0012] FIG. **1** illustrates a known authorization system (prior art).

[0013] FIG. **2** illustrates an embodiment of an authorization system.

DETAILED DESCRIPTION

[0014] The present invention addresses the need for a system and method for auditing and selectively controlling/limiting software usage. In the exemplary embodiments described herein, there are presented systems and methods for that allow for the rental of software where software is allowed to run for a pre-specified period of time in return for the purchase of a time limited license. It is noted that publishers sometimes use a time limited free trial period in which it is hoped that the user will want to continue the use of the software after the expiry of the free use period thereby requiring a license purchase and subsequent connection to the authorization authority. It is further noted that limited time licenses are may be used rather than new installs of software to trigger a requirement for the protected software to communicate with the authorization authority.

[0015] FIG. **1** illustrates a system and method for software copy control that is believed to be known in the art. The system generally comprises client-side system (e.g., client **9**) and an authorization system (e.g., authorization authority **30**). The system of FIG. **1** implements a client-side process on

a computing device, often called a client **9** in the art. The protected software then checks to see if a license is granted for this software to run. A common way of doing this is to store an unlock key on the computing device which the software uses to verify the license. On this basis the software checks to see if the unlock key is present (step **11**). If an unlock key is not present, an authorization process is started (step **16**). If the unlock key is present then the key is compared to the unique hardware configuration of the computing device (step **12**). Information about the components, peripherals and settings of the computing device are compiled into a unique identifier that can be used to verify the identity of the device for purposes of identification.

[0016] If the hardware identity has not changed, the software is allowed to continue to run (step **15**) such that the user can use the software. If the hardware identity is not the same or has changed then the user is asked for permission to re-authorize or re-activate the software (step **13**). If the user chooses not to re-activate the software the protected program/software is terminated (step **14**). If the user chooses to continue, the software starts the re-authorization process by asking the user to input the product's serial number (step **16**). The serial number may be used to represent a license number that may be issued to the user as part of a software purchase transaction. Next, the unique device identifier is compiled from the computers unique hardware configuration (step **17**). Then both the serial number and the unique device identifier are sent or communicated to the authorization authority **30** for license verification.

[0017] The authorization authority **30** checks to see if a valid license exists for the user that is requesting authorization, and if the user has not exceeded the limit set for the number of copies allowed under the license terms. The authorization authority **30** receives the serial number and the unique device identifier (step **31**) and then checks to see if the serial number exists and represents a legitimate license (step **32**). If the serial number does not exist in a database of the Authorization Authorities **30**, then a message is communicated back to the client system **9** and displayed to the user (step **19**) before the software is terminated **14**. If the serial number does exist (step **32**) then the count of previous successful authorizations is calculated and a decision/determination is made by the authorization authority as to whether or not the copy count limit has been exceeded for this particular serial number (step **33**).

[0018] If the copy count for the serial number has been exceeded, as determined at step **33**, then a message is sent to the client system and the user is informed that their request for authorization has failed (step **19**). Subsequently the client software is terminated (step **14**). If the copy count for the serial number has not been exceeded, as determined at step **33**, then an unlock key is generated (step **34**) for the specific serial number and unique device identifier, and communicated to the client system **9**. Upon receipt of the unlock key by the client system **9** the unlock key is stored (step **21**) for future reference by the license checking system and the user is informed that their request for authorization of their software was successful (step **20**). The software is then allowed to run (step **15**).

[0019] With reference to FIG. **2**, there is shown an exemplary audit system that allows copy control after the initial authorization of the licensed software. The copy controlled

approaches described herein are applicable to computing devices in general, including but not limited to, desktops, laptops, tablet computers, PDAs, mobile devices, mobile phones, vehicle onboard computers, or any network device capable of communication with a computer network.

[0020] The protected software checks to see if a license is granted for this software to run. An exemplary way of doing this is to store an unlock key or code on the computing device which the software uses to verify the license. On this basis the software checks to see if the unlock key is present (step **42**). If an unlock key is not present, an authorization process is started (step **43**). If the unlock key is present, then the key is compared to the hardware configuration of the computing device (step **44**). Information about the components, peripherals and settings of the computing device are compiled into a unique identifier that can be used to verify the identity of the device for purposes of identification.

[0021] The identification information or device identifier generally comprises information that is expected to be unique for the computing device. The device identifier is preferably generated from non-user-configurable machine parameters of the computing device, such as, for example, hard disk serial number, MAC ID, RAM manufacturing date, etc. It is noted that each data storage device of the computing device may gave a large variety of damage and unusable data sectors that are nearly unique to each physical unit. Accordingly, the process for generating a device identifier may include measuring physical, non-user-configurable characteristics of disk drives and solid state memory devices.

[0022] The machine parameters may relate to the platform on which a web browser or another application runs, such as, for example, CPU number, or unique parameters associated with the firmware in use. The machine parameters may also include system configuration information, such as amount of memory, type of processor, software or operating system serial number, etc. The device identifier generated from the machine parameters may include the computing device's IP address and/or other geo-location code to add another layer of specificity to the computing device's unique identifier. In the alternative, or in addition, the device identifier may comprise a randomly generated and assigned number that is unique for and stored on the computing device.

[0023] If the hardware identity has not changed, the software is allowed to continue to run (step **50**) such that the user can use the software. If the hardware identity is not the same or has changed, then the user is asked for permission to re-authorize or re-activate the software (step **46**). If the user chooses not to re-activate the software, the protected program is terminated (step **49**). If the user chooses to continue, the software starts the re-authorization process by asking the user to input the product serial number (step **43**). The serial number may represent a license number that is usually issued to the user as part of a software purchase transaction. Next, the unique device identifier is compiled from the hardware configuration of the computing device (step **45**). Then both the serial number and the unique device identifier are sent or communicated (step **47**) to the authorization authority or system **60** for license verification.

[0024] The authorization authority **60** checks to see if a valid license exists for the user that is requesting authorization, and if the user has not exceeded the limit set for the number of copies allowed under the license terms. The autho-

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS
Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS
Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS
Sync your system to PACER to automate legal marketing.