

**IN THE UNITED STATES DISTRICT COURT
FOR THE WESTERN DISTRICT OF TEXAS
WACO DIVISION**

TOPIA TECHNOLOGY, INC.,

Plaintiff,

v.

DROPBOX, INC., SAILPOINT
TECHNOLOGIES HOLDINGS, INC., and
CLEAR CHANNEL OUTDOOR
HOLDINGS, INC.

Defendants.

Case No. 6:21-cv-01373

JURY TRIAL DEMANDED

COMPLAINT FOR PATENT INFRINGEMENT

Plaintiff Topia Technology, Inc., (“Topia” or “Plaintiff”) files this complaint for patent infringement against Defendant Dropbox, Inc., (“Dropbox”), SailPoint Technologies Holdings, Inc. (“SailPoint”) and Clear Channel Outdoor Holdings, Inc. (“Clear Channel”)(collectively “Defendants”), and alleges as follows:

NATURE OF ACTION

1. This is an action for patent infringement arising under 35 U.S.C. § 1 *et seq.*, including §§ 271, 283, 284, and 285.

THE PARTIES

2. Topia is a company organized and existing under the laws of the State of Washington with its principal place of business in Tacoma, Washington.

3. Upon information and belief, Dropbox, Inc. (“Dropbox” or “Defendant”), is a corporation organized and existing under the laws of the State of Delaware.

4. Dropbox has a regular and established place of business in this District, including an office in Austin Texas located at 501 Congress Ave, Austin, Texas 78701. Dropbox’s

registered agent for service of process is Corporation Service Company d/b/a CSC-Lawyers Incorporating Service Company, at 211 E. 7th Street, Suite 620, Austin TX 78701.

5. Upon information and belief, SailPoint is a corporation organized and existing under the laws of the State of Delaware.

6. SailPoint has a regular and established place of business in this District, including its headquarters in Austin, Texas located at 11120 Four Points Drive, Suite 100, Austin TX 78726. SailPoint's registered agent for service of process is C T Corporation System located at 1999 Bryan St., STE. 900, Dallas, TX 75201.

7. Upon information and belief, Clear Channel is a corporation organized and existing under the laws of the State of Delaware.

8. Clear Channel has a regular and established place of business in this District, including its headquarters in San Antonio, TX, located at 4830 North Loop, 1604 West, Suite 111, San Antonio, TX 78249. Clear Channel's registered agent for service of process is C T Corporation System located at 1999 Bryan St., STE. 900, Dallas, TX 75201.

JURISDICTION AND VENUE

9. This Court has original jurisdiction over the subject matter of this action pursuant to 28 U.S.C. §§ 1331, 1367, and 1338(a).

10. Upon information and belief, Dropbox is in the business of providing online document storage and synchronization products and services, including products and services that infringe Plaintiff's patents identified below, through Dropbox's online platforms and mobile applications to customers. Upon information and belief, Dropbox has about 700 million users across 180 countries, including users in the State of Texas and this District.

11. Upon information and belief, Dropbox is subject to personal jurisdiction of this Court because it has a regular and established place of business in Austin, Texas, and is a resident of this state in this judicial District.

12. Venue is proper in this Court under 28 U.S.C. §§ 1391 and 1400, because Dropbox has committed infringing acts in this District and has a regular and established place of business in this District.

13. Upon information and belief, SailPoint is in the business of using Dropbox products and services and providing valued added integration services and products for Dropbox's online document storage and synchronization products and services, including those products and services that infringe Plaintiff's patents identified below, through Dropbox's online platforms and mobile applications to customers.

14. Upon information and belief, SailPoint is subject to personal jurisdiction of this Court because it has a regular and established place of business in Austin, Texas, and is a resident of this state in this judicial District.

15. Venue is proper in this Court under 28 U.S.C. §§ 1391 and 1400, because SailPoint has committed infringing acts in this District and has a regular and established place of business in this District.

16. Upon information and belief, Clear Channel is a Dropbox customer and uses Dropbox's online document storage and synchronization products and services, including those products and services that infringe Plaintiff's patents identified below, through Dropbox's online platforms and mobile applications.

17. Upon information and belief, Clear Channel is subject to personal jurisdiction of this Court because it has a regular and established place of business in San Antonio, Texas, and is a resident of this state in this judicial District.

18. Venue is proper in this Court under 28 U.S.C. §§ 1391 and 1400, because Clear Channel has committed infringing acts in this District and has a regular and established place of business in this District.

19. The allegations provided below are exemplary and without prejudice to Plaintiff's infringement contentions that will be provided pursuant to the Court's scheduling order and local civil rules, including after discovery as provided under the Federal Rules of Civil Procedure. In providing these allegations, Plaintiff does not convey or imply any particular claim constructions or the precise scope of the claims of the asserted patents. Plaintiff's proposed claim constructions, if any, will be provided pursuant to the Court's scheduling order and local civil rules.

COUNT ONE

INFRINGEMENT OF U.S. PATENT NO. 9,143,561

20. Plaintiff incorporates paragraphs 1 through 19 as though fully set forth herein.

21. U.S. Patent No. 9,143,561 ("the '561 Patent"), entitled "Architecture For Management of Digital Files Across Distributed Network," issued on September 22, 2015. A copy of the '561 Patent is attached as Exhibit 1.

22. The '561 patent is generally directed to systems and methods for sharing electronic files between multiple devices, wherein when a user modifies an electronic file on a device, a copy of the modified electronic file is automatically transferred to at least one other device.

23. Plaintiff is the owner by assignment of all rights, title, and interest in and to the '561 Patent, including the right to assert all causes of action arising under the '561 Patent and the right to all remedies for the infringement of the '561 Patent.

24. Claim 1 of the '561 Patent recites:

1. A system, comprising:

a first electronic device configured to selectively execute a first application, the first electronic device being in communication with a second electronic device and a third electronic device, each associated with a user wherein the first electronic device is configured to:

receive from a second application executable on the second electronic device a copy of a first electronic file automatically transferred from the second application when the user modifies a content of the first electronic file; and

wherein the first electronic device is further configured to receive from a third application executable on the third electronic device a copy of a second electronic file automatically transferred from the third application when the user modifies a content of the second electronic file; and

wherein the first application is further configured to automatically transfer the modified first electronic file copy to the third electronic device to replace an older version of the first electronic file stored on the third electronic device with the modified first electronic file copy having the content modified by the user; and

automatically transfer the modified second electronic file copy to the second electronic device to replace an older version of the second electronic file stored on the second electronic device with the modified second electronic file copy having the content modified by the user;

wherein the second application automatically transfers the copy of the modified first electronic file to the first electronic device upon determining that a save operation has been performed on the modified first electronic file.

DEFENDANT DROPBOX

25. Dropbox offers a suite of products and services, including Dropbox Professional, Dropbox Standard and Dropbox Advanced for Businesses and Dropbox Plus and Dropbox Family for individuals which practice each and every limitation of one or more claims of the '561 patent. Dropbox's products and services include systems and methods for sharing electronic files between multiple devices, wherein when a user modifies an electronic file on a device, a copy of the modified electronic file is automatically transferred to at least one other device.

26. Dropbox products and services involve systems that include various client and server devices and software. For example, Dropbox has one central hub for online file storage that is accessible through client applications on Windows, Mac, Linux, iOS, Android, and web browsers.

27. Dropbox's server infrastructure includes a first electronic device (*e.g.*, a server system) executing an application (*e.g.*, running Dropbox server software). The first electronic device (*e.g.*, the server system) is in communication with a second electronic device (*e.g.*, the first client device such as a laptop or a smart phone) and a third electronic device (*e.g.*, the second client device such as a laptop or a smart phone). Dropbox promotes its products and services as a system for providing secure access to all of its customers files from any device:

Dropbox gives you secure access to all your files. Collaborate with friends, family, and coworkers from any device.

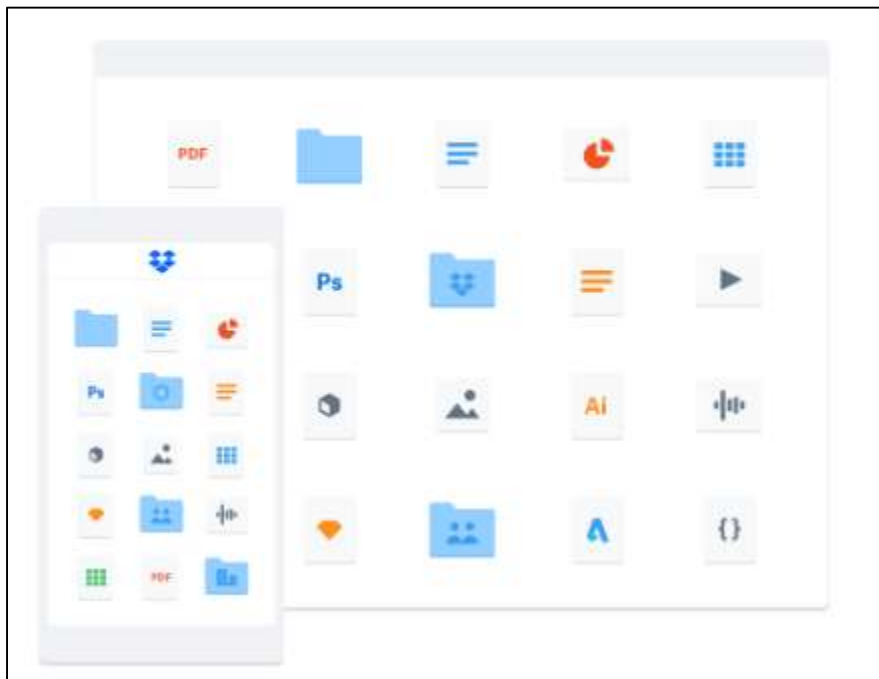
<https://www.dropbox.com> (last visited December 16, 2021)

Access your files from multiple devices

Dropbox offers one central hub for online file storage, file sharing, and syncing. Whether you're at work or on the road, your files are synced across your devices and accessible in real time. Access your Dropbox account with desktop apps on Windows and Mac, our mobile app for iOS or Android devices, and on the web through your browser.

How do I access cloud storage?

You can access your cloud storage with Dropbox on your phone using the Dropbox app as long as your phone is connected to wifi. You can also access cloud storage via a Windows, Mac, or Linux operating systems one of two ways: either through the web on [dropbox.com](https://www.dropbox.com) or with our desktop app. You just need to make sure your device is connected to the internet to upload and access your files.



<https://www.dropbox.com/features/cloud-storage> (last visited December 16, 2021)

Can I access Dropbox on my mobile device?

You can access your Dropbox account, and your Dropbox files, with the Dropbox mobile app for your phone or tablet (including Android, iPhone, and iPad). Alternatively, you can [sign into](#) dropbox.com on your mobile device in any mobile browser app.

The Dropbox mobile app is free and lets you:

- Access all your Dropbox files and folders
- Browse and preview files in Dropbox
- Use third-party apps to open and edit files
- Take photos and videos using your built-in camera and save them directly to Dropbox
- [Make important files available for offline access](#)
- Share your files with links

<https://help.dropbox.com/installs-integrations/mobile/access-dropbox> (last visited December 16, 2021)

28. Dropbox maintains a worldwide server infrastructure having multiple servers deployed in multiple data centers, including numerous data centers located in the United States. Dropbox provides the following overview of its architecture:

Data centers

Dropbox corporate and production systems are housed at third-party subservice organization data centers and managed service providers located in the United States. These third-party service providers are responsible for the physical, environmental, and operational security controls at the boundaries of Dropbox Infrastructure. Dropbox is responsible for the logical, network, and application security of our Infrastructure housed at third-party data centers.

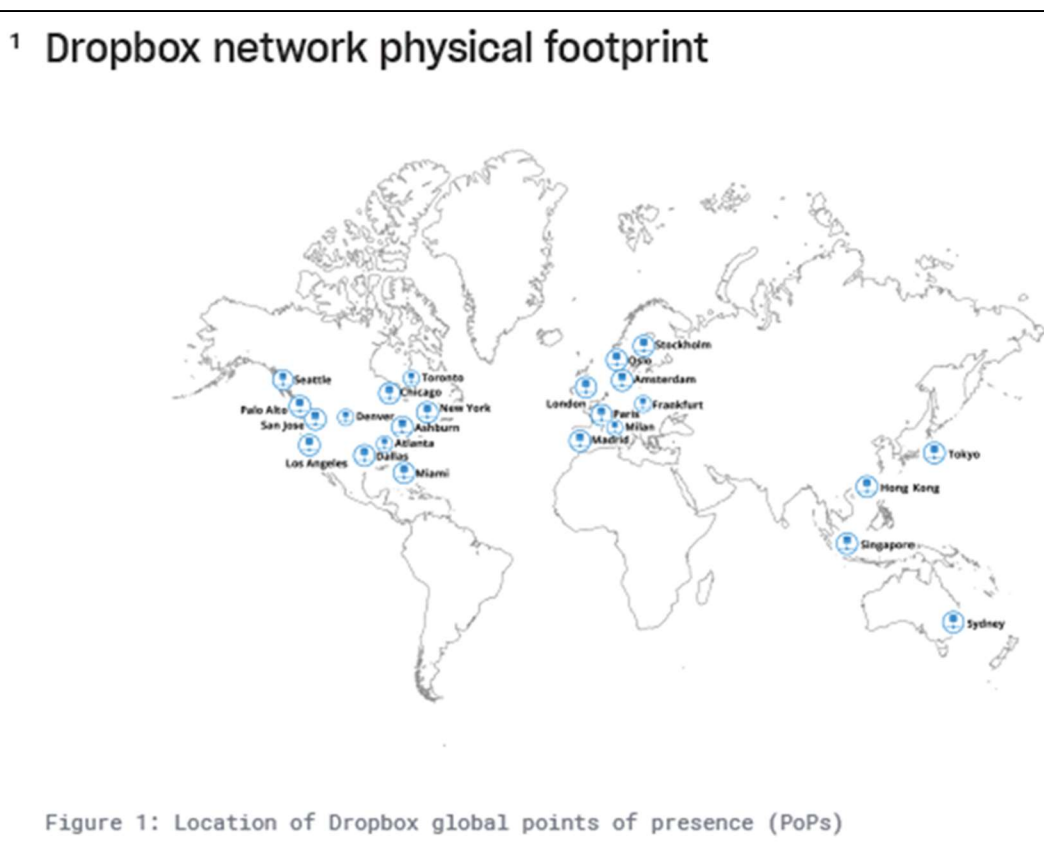
<https://www.dropbox.com/business/trust/security/architecture> (last visited December 16, 2021)

Where is my data stored?

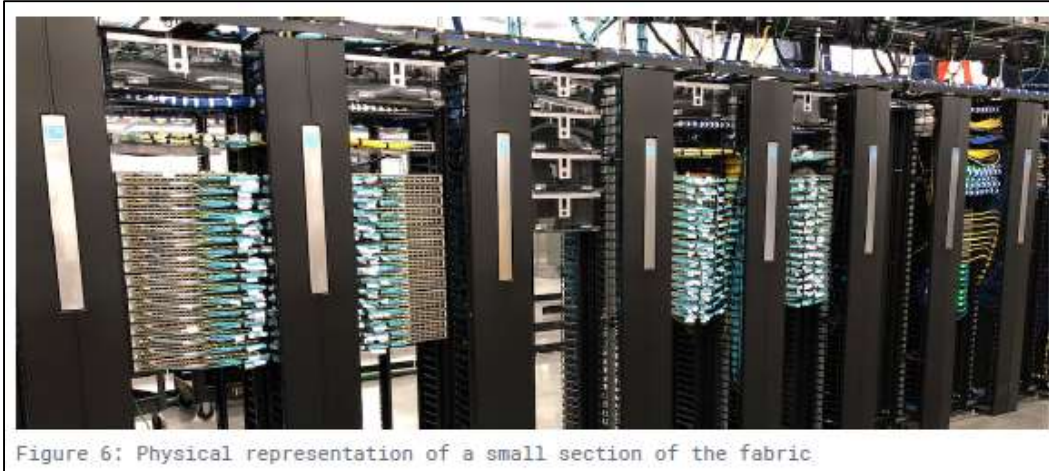
Once a file is added to your Dropbox, it's synced to our secure online servers. All files stored online by Dropbox are encrypted and kept in secure storage servers. Storage servers are located in data centers across the United States. Additionally, storage servers are available in Germany, Australia, and Japan for some Dropbox Business users.

<https://help.dropbox.com/accounts-billing/security/physical-location-data-storage> (last visited

December 16, 2021)



We currently have global network presence and multiple data centers in California, Texas and Virginia. From a redundancy perspective, the North American continent is carved into regions—East, Central, and West—thereby having a distributed data center approach and improving resiliency in events of failure.



<https://dropbox.tech/infrastructure/the-scalable-fabric-behind-our-growing-data-center-network> (last visited December 16, 2021)

29. Clients are served by proprietary point of presence Edge servers that are closest.

United States users are served by geographically proximate servers:

¹ **Dropbox scale**

Dropbox has more than half a billion registered users who trust us with over an [exabyte of data](#) and petabytes of corresponding metadata. For the Traffic team this means millions of HTTP requests and terabits of traffic. To support all of that we've built an extensive network of points of presence (PoPs) around the world that we call Edge.

³ Picking PoP Locations

As of today, Dropbox has 20 PoPs around the world:



We try to alternate new PoP placement between selecting the most advantageous PoP for the existing and potential Dropbox users.

A tiny script helps us brute-force the problem by:

A tiny script helps us brute-force the problem by:

1. Splitting the Earth into 7th level s2 regions
2. Placing all the existing PoPs
3. Computing the distance to the nearest PoP for all the regions weighted by "population"
4. Doing exhaustive search to find the "best" location for the new PoP
5. Adding it to the map
6. Looping back to step 3, etc.

By "population" one can use pretty much any metric we want to optimize, for example total number of people in the area, or number of existing/potential users. As for the loss function to determine the score of each placement one can use something standard like L1 or L2 loss. In our case we try to overcompensate for the effects of latency on the TCP throughput.

PoPs consist of network equipment and sets of Linux servers. An average PoP has good connectivity: backbone, multiple transits, public and private peering. By increasing our network connectivity, we decrease the time packets spend in the public internet and therefore heavily decrease packet loss and improve TCP throughput. Currently about half of our traffic comes from peering.

<https://dropbox.tech/infrastructure/dropbox-traffic-infrastructure-edge-network> (last visited December 16, 2021)

30. The first electronic device (*e.g.*, the server system) running Dropbox server software and the first and the second electronic client devices (*e.g.*, the first and second client devices such as laptop or a smart phone) running Dropbox software are associated with a user. Client devices are associated with a logged-in Dropbox user:

Create an account

To get started, go to dropbox.com and create an account. You can create a free Basic account with 2 GBs or [a paid account with additional storage space and features](#). Then, you can start uploading files to your account or creating new files in Dropbox.

Download and sign into the apps

Once you have a Dropbox account, you can [download the Dropbox desktop app](#) to your computer or [download the Dropbox mobile app](#) to your phone or tablet. After you download the Dropbox apps to your devices, [sign into your Dropbox account](#).

Sync your files

You can access all of the files you stored in Dropbox, no matter what device you're using or where you added the files. Any files you add or changes you make to your files are automatically updated, or "synced", everywhere you access them in Dropbox.

<https://help.dropbox.com/installs-integrations/sync-uploads/sync-overview> (last visited December 16, 2021)

31. The first electronic device (*e.g.*, the server system running Dropbox server software) is configured to receive from a second application executable (*e.g.*, Dropbox App) on the second electronic device (*e.g.*, the first client device such as a laptop or a smart phone) a copy of a first electronic file automatically transferred from the second application when the user modifies a content of the first electronic file.

32. Dropbox's server system receives, over a network, a copy of a first file from a first client device (*e.g.*, a laptop or a smart phone) associated with a user, the copy of the first file being automatically received from the first client device when the user modifies the content of the first file stored on the first client device, the copy of the first file being an updated version of the first file that is generated from the user modifying the content of the first file in the client device. When a user modifies a file that has been configured to be synced, the Dropbox software on the client device will upload the updated version of the file that the user modified to Dropbox's servers:

At its heart, sync is pretty straightforward. Every time you save a file on one device, it's uploaded to an online server. Since it now lives somewhere other than your hard drive — someplace that's always connected to the Internet — you can access the file from any other device. Plus, you don't even need to do anything to get the latest version. Each device repeatedly checks in with the server to see if there's anything new; if there is, they download it, automatically. There are a few riffs on this theme, but that's how [Dropbox for Business](#) sync works.

<https://blog.dropbox.com/topics/work-culture/what-is-file-sync> (last visited December 16, 2021)

33. Dropbox includes a file sync feature that works on client platforms:

Sync files across devices and platforms

It's easy to make your files accessible on your daily commute to work or on vacation. Save a file to the Dropbox folder on your computer, and it will synchronize automatically to your mobile device. Cloud file sync is available on multiple devices and platforms, from Windows and Mac to mobile devices like iPhone, iPad and Android via the Dropbox mobile app.

Newly saved or updated files are automatically synced everywhere, so you don't have to spend time emailing the newest versions to collaborators. And you can be reassured that all your important files are completely synced by looking for the green checkmark.

What is sync?

Sync is short for the word 'synchronize' which means an event that happens in more than one place simultaneously. In terms of technology, when you sync a device—such as a phone or tablet, with your computer, all of the data from your computer is automatically synchronized with that device. You can feel confident knowing that all your data—such as photos or work files—is available to you on different devices. This data syncing helps further protect you from data loss because it's saved in more than one place. With [Dropbox Smart Sync](#), you can also save space on your hard drive by removing old or 'stale' folders you don't use regularly and storing them to the cloud.

<https://www.dropbox.com/features/sync> (last visited December 16, 2021)

How does Dropbox sync my files?

This article is a basic introduction to how Dropbox syncs your files. It explains how you can store your files in Dropbox and sync them between your devices.

When you store files in Dropbox, your files are accessible from everywhere you use Dropbox. This includes:

- Dropbox.com in a web browser, on your computer or phone
- The Dropbox desktop app, on your computer
- The Dropbox mobile app, on your phone or tablet

Your files are also kept up to date everywhere you use Dropbox. This means that if you add or make changes to a file in one place, the file is automatically updated everywhere else.

How do I get started using Dropbox to sync my files?

You can start syncing all your files across all of your devices with a free Dropbox Basic plan. Dropbox Basic users have 2 GBs of storage space. There's no time limit or trial on a Basic plan, so you can try it as long as you like.

If you want more storage space or access to additional features, you can upgrade to a paid plan. [Learn more about the different Dropbox plan options.](#)

<https://help.dropbox.com/installs-integrations/sync-uploads/sync-overview> (last visited

December 16, 2021)

34. The first electronic device (*e.g.*, the server system running Dropbox server software) is configured to automatically transfer the modified first file copy to the third electronic device (*e.g.*, the second client device such as a laptop or a smart phone) to replace an older version of the first file stored on the third electronic device (*e.g.*, the second client device) with the modified first file copy having the content modified by the user. Dropbox allows multiple client devices to be linked to a user account, and modifying any file on any of the linked

client devices will cause the modified metadata and file contents to be uploaded from the device to Dropbox’s servers, and then downloaded by any other linked client device. Dropbox provides the following architecture for synchronizing the files across the devices:

Our file infrastructure is comprised of the following components:

Metadata servers

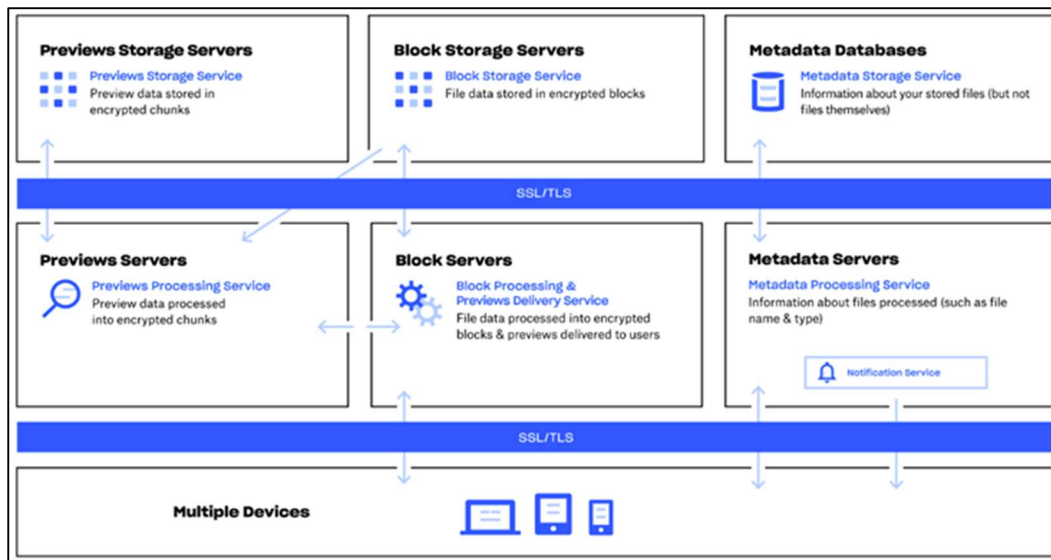
Certain basic information about user data, called metadata, is kept in its own discrete storage service and acts as an index for the data in users’ accounts. Metadata includes basic account and user information, like email address, name, and device names. Metadata also includes basic information about files, including file names and types, that helps support features like version history, recovery, and sync.

Metadata Databases

File metadata is stored in a MySQL-backed database service, and is sharded and replicated as needed to meet performance and high availability requirements.

Notification service

This is a separate service dedicated to monitoring if changes have been made to Dropbox accounts. No file data or metadata is stored or transferred here. Each client establishes a long poll connection to the notification service and waits. When a change to any file in Dropbox takes place, the notification service signals a change to the relevant client(s) by closing the long poll connection. Closing the connection signals that the client must connect to the Metadata Servers securely to synchronize any changes.



<https://www.dropbox.com/business/trust/security/architecture> (last visited December 16,

2021)

35. Using the Dropbox API, a client device can be configured to automatically make an API call to perform a push request for a modified file:

/upload

VERSION

DESCRIPTION Create a new file with the contents provided in the request. Do not use this to upload a file larger than 150 MB. Instead, create an upload session with [upload_session/start](#). Calls to this endpoint will count as data transport calls for any Dropbox Business teams with a limit on the number of data transport calls allowed per month. For more information, see the [Data transport limit page](#).

PARAMETERS

```
{
  "path": "/Homework/math/Matrices.txt",
  "mode": "add",
  "autorename": true,
  "mute": false,
  "strict_conflict": false
}
```

<https://www.dropbox.com/developers/documentation/http/documentation#files-upload> (last visited December 16, 2021)

36. Using the Dropbox API's Webhooks feature, file metadata will be automatically transferred to a client device when a file changes on another client device. Metadata including a change notification is sent first, after which third party apps may request additional change details metadata and then file contents:

Using Webhooks

Webhooks are a way for web apps to get real-time notifications when users' files change in Dropbox.

Once you register a URI to receive webhooks, Dropbox will send an HTTP request to that URI every time there's a change for any of your app's registered users.

Receiving notifications

Once your webhook URI is added, your app will start receiving "notification requests" every time a user's files change. A notification request is an HTTP POST request with a JSON body. The JSON has the following format:

```
{
  "list_folder": {
    "accounts": [
      "dbid:AAH4f99T0taONIb-OurWxbNQ6ywGRopQngc",
      ...
    ]
  },
  "delta": {
    "users": [
      12345678,
      23456789,
      ...
    ]
  }
}
```

Note that the payload of the notification request does not include the actual file changes. It only informs your app of which users have changes. You will typically want to call [/files/list_folder/continue](#) to get the latest changes for each user in the notification, keeping track of the latest cursor for each user as you go.

Typically, the code you run in response to a notification will make a call to [/files/list_folder/continue](#) to get the latest changes for a user. In our sample Markdown-to-HTML converter, we're keeping track of each user's OAuth access token and their latest cursor in [Redis](#) (a key-value store). This is the implementation of `process_user` for our Markdown-to-HTML converter sample app:

Best practices

Always respond to webhooks quickly

Your app only has ten seconds to respond to webhook requests. For the verification request, this is never really an issue, since your app doesn't need to do any real work to respond. For notification requests, however, your app will usually do something that takes time in response to the request. For example, an app processing file changes will call [/files/list_folder/continue](#) and then process the changed files. (In our Markdown example, we needed to download each Markdown file, convert it to HTML, and then upload the result.) It's important to keep in mind that `list_folder` payloads can sometimes be *very* large and require multiple round-trips to the Dropbox API, particularly when a new user first links your app and has a lot of files.

To make sure you can always respond within ten seconds, you should always do your work on a separate thread (as in the simple example above) or asynchronously using a queue.

[https://www.dropbox.com/developers/reference/webhooks?_tk=guides_lp&_ad=deepdive7&](https://www.dropbox.com/developers/reference/webhooks?_tk=guides_lp&_ad=deepdive7&_camp=webhooks)

[_camp=webhooks](#) (last visited December 16, 2021)

With webhooks configured, Dropbox sends an HTTP POST with the user IDs when changes occur. By saving the `cursor` for users, your application can then call `/files/list_folder_continue` to read the associated change when it receives the hook.

https://www.dropbox.com/developers/reference/content-access-guide?_tk=guides_lp&_ad=deepdive6&_camp=content_access (last visited December 16, 2021)

37. The second application (*e.g.*, Dropbox software application running on the first client device) automatically transfers the copy of the modified first file to the first electronic device (*e.g.*, the server system) upon determining that a save operation has been performed on the modified first electronic file. Saving the file to the client device causes the modified file to be automatically transferred to Dropbox's servers and synced to other devices:

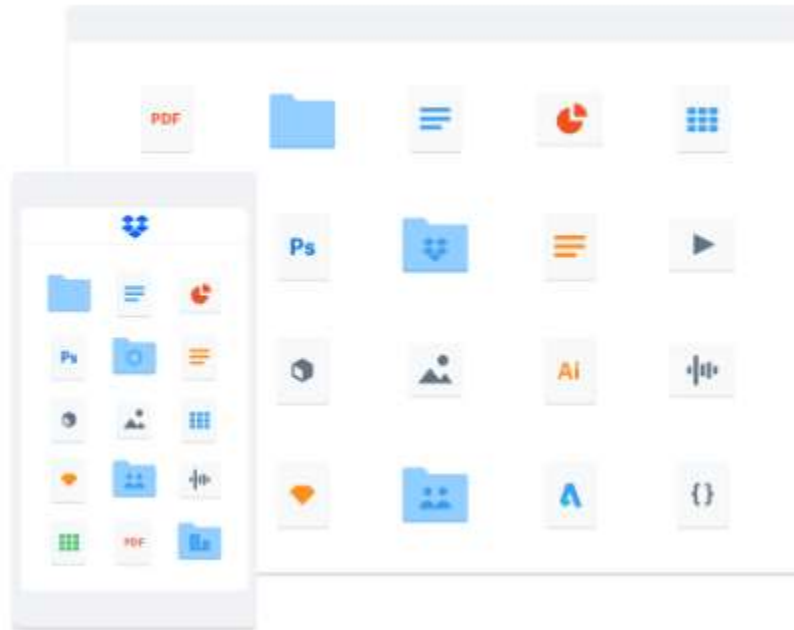
Sync files across devices and platforms

It's easy to make your files accessible on your daily commute to work or on vacation. Save a file to the Dropbox folder on your computer, and it will synchronize automatically to your mobile device. Cloud file sync is available on multiple devices and platforms, from Windows and Mac to mobile devices like iPhone, iPad and Android via the Dropbox mobile app.

Newly saved or updated files are automatically synced everywhere, so you don't have to spend time emailing the newest versions to collaborators. And you can be reassured that all your important files are completely synced by looking for the green checkmark.

What is sync?

Sync is short for the word 'synchronize' which means an event that happens in more than one place simultaneously. In terms of technology, when you sync a device—such as a phone or tablet, with your computer, all of the data from your computer is automatically synchronized with that device. You can feel confident knowing that all your data—such as photos or work files—is available to you on different devices. This data syncing helps further protect you from data loss because it's saved in more than one place. With [Dropbox Smart Sync](#), you can also save space on your hard drive by removing old or 'stale' folders you don't use regularly and storing them to the cloud.



<https://www.dropbox.com/features/sync> (last visited December 16, 2021)

38. The first electronic device (*e.g.*, the server system) is further configured to receive from a third application (*e.g.*, a Dropbox App) executable on the third electronic device (*e.g.*, the second client device) a copy of a second file automatically transferred from the third application when the user modifies a content of the second electronic file, and automatically transfer the modified second file copy to the second electronic device (*e.g.*, the first client device) to replace an older version of the second file stored on the second electronic device with the modified second electronic file copy having the content modified by the user.

39. On information and belief, Dropbox has directly infringed and continues to infringe, literally and/or under the doctrine of equivalents, one or more claims of the '561 Patent under 35 U.S.C. § 271(a) by developing, distributing, operating, using, selling, and/or offering to sell Dropbox products and services in the United States without authority. Included in these acts of infringement are situations where user devices and customers of Dropbox products and services interact with Dropbox servers to perform file sync operations.

40. On information and belief, Dropbox is liable for infringement, literally and/or under the doctrine of equivalents, of one or more claims of the '561 Patent under 35 U.S.C. § 271(b) based on its active marketing and promotion of its Dropbox products and services in the United States to its customers and prospective customers. On information and belief Dropbox has, and will continue to, intentionally encourage acts of direct infringement with knowledge of the '561 Patent and knowledge that its acts are encouraging infringement.

41. On information and belief, Dropbox is liable for infringement, literally and/or under the doctrine of equivalents, of one or more claims of the '561 Patent under 35 U.S.C. § 271(c), because Dropbox has had, and continues to have, knowledge that its Dropbox products are especially developed or adapted for a use that infringes the '561 Patent and constitute a material part of the claimed systems and methods. Dropbox has had, and continues to have, knowledge that there are no substantial non-infringing uses for these Products. Dropbox has infringed and continues to infringe the '561 Patent directly and/or indirectly in violation of 35 U.S.C. § 271(c).

DEFENDANT SAILPOINT

42. SailPoint makes and sells valued added integration services and products for Dropbox's online document storage and synchronization products and services, including those products and services that infringe Plaintiff's patents identified above, through Dropbox's online platforms and mobile applications to customers, which practice each and every limitation of one or more claims of the '561 Patent.

43. More specifically, SailPoint's Identity Governance Platform provides identity governance solutions that are integrated with Dropbox products and services, enabling users of Dropbox products and services to, among other things, control user access, identify sensitive data, and monitor for malicious behavior.

How does SailPoint integrate with Dropbox?

SailPoint collects and analyzes the data and file permissions in Dropbox to identify sensitive data and determine who has access to it. We can also alert you about inappropriate access to help you maintain security.

Examples

Data discovery and classification ▶

Permission analysis ▶


Access monitoring and alerting ▶

Identity for Dropbox

Could some of your files in Dropbox be a compliance risk?

SailPoint's integration with Dropbox helps you identify sensitive information, manage data ownership and alert you to any suspicious access activity. As more people collaborate and share information in the cloud, protecting access to your Dropbox files has never been more important.

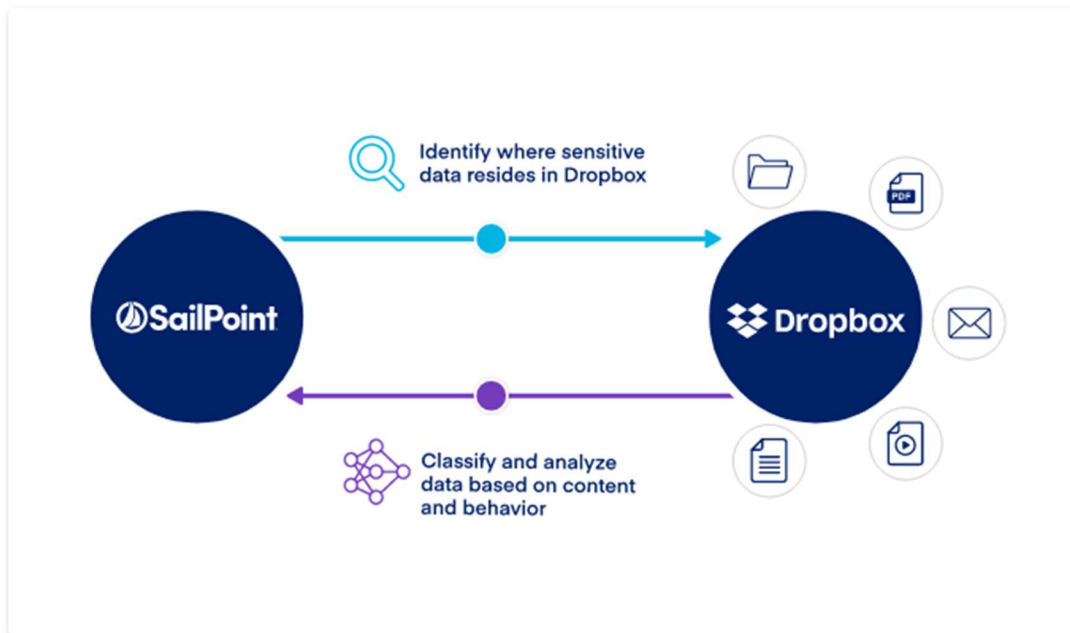
- Find and address your compliance gaps
- Ensure consistent governance across all data types
- Be alerted if inappropriate access is detected
- Minimize exposure to data leakage



Learn more

- How identity governance helps ensure GDPR compliance
- Governing unstructured data and data access
- Watch our Dropbox integration demo

Data discovery and classification



<https://www.sailpoint.com/integrations/dropbox/> (last visited December 16, 2021)

SailPoint is the leading provider of enterprise identity governance solutions. SailPoint's open identity platform enables organizations to manage the entire identity lifecycle of users accessing Dropbox Business, more efficiently control user access, securely prepare your data for migration to Dropbox Business, identify sensitive data, and monitor for malicious behavior.

- **Oversee and streamline who accesses Dropbox Business.** SailPoint identity governance solutions allow you to manage and govern users, groups, and entitlements for DropBox and automate the requesting and provisioning of user access. Improve security and audit performance by instantly reviewing and remediating access.
- **Monitor inappropriate access.** Centralize your visibility to users and their access across Dropbox and other cloud and on-premises applications. Admins can easily audit and ensure access is within corporate policy.
- **Gain greater data visibility.** Identify and classify data prior to migration to gain clear visibility across your data assets and optimize what data should be moved to Dropbox.
- **Clean up permissions.** Collect and analyze permissions to files for deeper insight into who has access and remediate inappropriate access issues to mitigate security risk to your content.
- **Establish data ownership.** Empower data owners who have the most intelligence about the data to take on a key role in managing access and ensure the right users have access to the right data.
- **Support hybrid environments.** Whether your data resides within your datacenter or in Dropbox Business, govern it with a centralized set of controls and policies.

<https://www.dropbox.com/app-integrations/sailpoint> (last visited December 16, 2021)

44. On information and belief, SailPoint, in order to develop, support, and provide its identity governance solutions, has directly infringed and continues to infringe, literally and/or under the doctrine of equivalents, one or more claims of the '561 Patent under 35 U.S.C. § 271(a) by using Dropbox products and services in the United States and abroad without

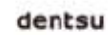
authority. Included in these acts of infringement are situations where devices associated with SailPoint interact with Dropbox servers to perform file sync operations.

DEFENDANT CLEAR CHANNEL

45. Clear Channel uses Dropbox’s online document storage and synchronization products and services, including those products and services that infringe Plaintiff’s patents identified above, through Dropbox’s online platforms and mobile applications to customers, which practice each and every limitation of one or more claims of the ’561 patent.

46. On its website, Dropbox identifies Clear Channel as a user of Dropbox Business.

**Businesses across the world of media trust
Dropbox**



<https://www.dropbox.com/business/solutions/media> (last visited December 16, 2021)

47. In 2017, Clear Channel along with Dropbox released a study disclosing Clear Channel’s purchase of Dropbox Business licenses for its employees and summarizing the commercial benefits from the use of Dropbox Business. A copy of Clear Channel’s study is attached as Exhibit 2.

Dropbox Business

Clear Channel Outdoor

In January 2017, Clear Channel purchased 1,200 enterprise Dropbox licenses to provide an enhanced security and collaboration experience for end users. Through three business cases, this study outlines the total return on investment achieved by Clear Channel as a result of its deployment. The study's findings are summarized below.

\$677k
total financial benefit

541%
return on investment

<https://www.insightsforprofessionals.com/management/leadership/dropbox-businness-and-clear-channel-outdoor/download> (last visited December 16, 2021)

48. On information and belief, Clear Channel has directly infringed and continues to infringe, literally and/or under the doctrine of equivalents, one or more claims of the '561 Patent under 35 U.S.C. § 271(a) by using Dropbox products and services in the United States and abroad without authority. Included in these acts of infringement are situations where devices associated with Clear Channel interact with Dropbox servers to perform file sync operations.

COUNT TWO

INFRINGEMENT OF U.S. PATENT NO. 10,006,942

49. Plaintiff incorporates paragraphs 1 through 48 as though fully set forth herein.

50. Plaintiff is the owner by assignment of U.S. Patent No. 10,067,942 (the "'942 Patent"), entitled "Architecture For Management of Digital Files Across Distributed Network," issued on September 4, 2018. A copy of the '942 Patent is attached as Exhibit 3.

51. The '942 Patent is generally directed to systems and methods for sharing electronic files between multiple devices, wherein when a user modifies an electronic file on a device, a copy of the modified electronic file is automatically transferred to other devices based on a communication state of the other devices.

52. Plaintiff is the owner by assignment of all rights, title, and interest in and to the '942 Patent, including the right to assert all causes of action arising under the '942 Patent and the right to all remedies for the infringement of the '942 Patent.

53. For example, claim 1 of the '942 Patent states:

1. A system, comprising:

a first electronic device, associated with a user, configured to:

receive, via a first application at the first electronic device, a copy of a modified

first electronic file from a second application at a second electronic device

associated with the user, wherein the modified first electronic file copy is

automatically received from the second application responsive to the user

modifying a content of the first electronic file;

determine whether the first electronic device is in communication with a third

electronic device;

automatically send, via the first application, the modified first electronic file copy

to a third application at the third electronic device responsive to the

determination that the first electronic device is in communication with the

third electronic device and responsive to receiving the modified first

electronic file copy from the second electronic device;

receive, via the first application, a copy of a modified second electronic file from

the third application at the third electronic device associated with the user,

wherein the modified second electronic file copy is automatically received from the third application responsive to the user modifying a content of the second electronic file;

determine whether the first electronic device is in communication with the second electronic device; and

automatically send, via the first application, the modified second electronic file copy to the second application at the second electronic device responsive to the determination that the first electronic device is in communication with the second electronic device and responsive to receiving the modified second electronic file copy from the third electronic device,


wherein, responsive to sending the modified first electronic file copy to the third electronic device, an older version of the first electronic file stored on the third electronic device is automatically caused to be replaced with the modified first electronic file copy such that the modified first electronic file copy is stored on the third electronic device in lieu of the older version of the first electronic file, and

wherein, responsive to sending the modified second electronic file copy to the second electronic device, an older version of the second electronic file stored on the second electronic device is automatically caused to be replaced with the modified second electronic file copy such that the modified second electronic file copy is stored on the second electronic device in lieu of the older version of the second electronic file.

DEFENDANT DROPBOX

54. In addition to the description in paragraphs 25-41, Dropbox's products and services include systems and methods for sharing electronic files between multiple client devices, wherein when a user modifies an electronic file on a device, a copy of the modified electronic file is automatically transferred to other devices based on a communication state of the other devices.

55. The first electronic device (*e.g.*, the server system) is configured to determine whether the first electronic device is in communication with a third electronic device (*e.g.*, the second client device such as a laptop or a smart phone). Dropbox's server system and the client devices determine that the server is in contact with a respective client device after a period of being offline:



Not connected to internet

A gray Dropbox icon means that the Dropbox desktop app isn't connected to the internet. This means that changes you make to the Dropbox files and folders on your computer won't update everywhere you access your files in Dropbox until you're connected to internet again. [Learn why Dropbox might not be connecting and how to solve it.](#)

<https://help.dropbox.com/installs-integrations/sync-uploads/sync-icons> (last visited December 16, 2021)

No Wi-fi? No problem.

With offline sync and access, you can easily choose to make files in your Dropbox available when you're away from Wi-Fi. No signal on the commuter rail, airplane, or while working remote at the coffee shop? No problem—simply make the files you need available offline to access where and when you need them.

What does working remotely or working offline mean? ^

Remote working or working offline are two states of activity that can sometimes be related. Working offline means you're not connected to the Internet but are still able to access your files and folders. You can work on an offline file and any changes will sync when you're back online. Working remotely means you're away from your normal working environment; this could mean you're offline without access to Wi-Fi or have limited Internet connection. Some team members may be full time remote workers or fully remote employees that never travel to a central office.

Can I work offline with Dropbox?
Yes, you can work offline with Dropbox files and folders from your mobile device. To make files available offline, simply turn on the "Available offline" option for individual files you'd like to work with. Dropbox Professional, Dropbox Business, or Dropbox Enterprise customers can also make folders available offline by switching on the same option in each folder.

<https://www.dropbox.com/features/sync/work-remotely-offline> (last visited December 16, 2021)

56. The first electronic device (*e.g.*, the server system) is configured to automatically send, via the first application (*e.g.*, application running on the server system), the modified first electronic file copy to a third application at the third electronic device (*e.g.*, the second client device) responsive to the determination that the first electronic device is in communication with the third electronic device and responsive to receiving the modified first electronic file copy from the second electronic device (*e.g.*, the first client device). The Dropbox server system will automatically send the modified file copy to the client device responsive to the determination that the client device is connected to the Dropbox server system and responsive to changes to the file in another client device:

Get offline access on all your devices
With the desktop app, locally synchronized folders and files are available even when you're away from an Internet connection. Once you get back online, Dropbox will automatically synchronize your folders and files with all the latest changes. You can also select files to access offline on your Android or iPhone smartphone, and even your iPad.

<https://www.dropbox.com/features/sync> (last visited December 16, 2021)

57. Upon information and belief, Dropbox has infringed and continues to infringe, literally or under the doctrine of equivalents, one or more claims of the '942 Patent by developing, distributing, operating, using, selling, and/or offering to sell Dropbox products and services in the United States without authority. Included in these acts of infringement are situations where user devices and customers of Dropbox products and services interact with Dropbox servers to perform file sync operations.

58. On information and belief, Dropbox is liable for infringement, literally and/or under the doctrine of equivalents, of one or more claims of the '942 Patent under 35 U.S.C. § 271(b) based on its active marketing and promotion of its Dropbox products and services in the United States to its customers and prospective customers. On information and belief Dropbox has, and will continue to, intentionally encourage acts of direct infringement with knowledge of the '942 Patent and knowledge that its acts are encouraging infringement.

59. On information and belief, Dropbox is liable for infringement, literally and/or under the doctrine of equivalents, of one or more claims of the '942 Patent under 35 U.S.C. § 271(c), because Dropbox has had, and continues to have, knowledge that its Dropbox products are especially developed or adapted for a use that infringes the '942 Patent and constitute a material part of the claimed systems and methods. Dropbox has had, and continues to have, knowledge that there are no substantial noninfringing uses for these Products. Dropbox has infringed and continues to infringe the '942 Patent directly or indirectly in violation of 35 U.S.C. § 271(c).

DEFENDANT SAILPOINT

60. SailPoint makes and sells valued added integration services and products for Dropbox's online document storage and synchronization products and services, including those products and services that infringe Plaintiff's patents identified above, through Dropbox's online platforms and mobile applications to customers, which practice each and every limitation of one or more claims of the '942 patent.

61. More specifically, SailPoint's Identity Governance Platform provides identity governance solutions that are integrated with Dropbox products and services, enabling users of Dropbox products and services to, among other things, control user access, identify sensitive data, and monitor for malicious behavior.

How does SailPoint integrate with Dropbox?

SailPoint collects and analyzes the data and file permissions in Dropbox to identify sensitive data and determine who has access to it. We can also alert you about inappropriate access to help you maintain security.

Examples

Data discovery and classification ▶

Permission analysis ▶


Access monitoring and alerting ▶

Identity for Dropbox

Could some of your files in Dropbox be a compliance risk?

SailPoint's integration with Dropbox helps you identify sensitive information, manage data ownership and alert you to any suspicious access activity. As more people collaborate and share information in the cloud, protecting access to your Dropbox files has never been more important.

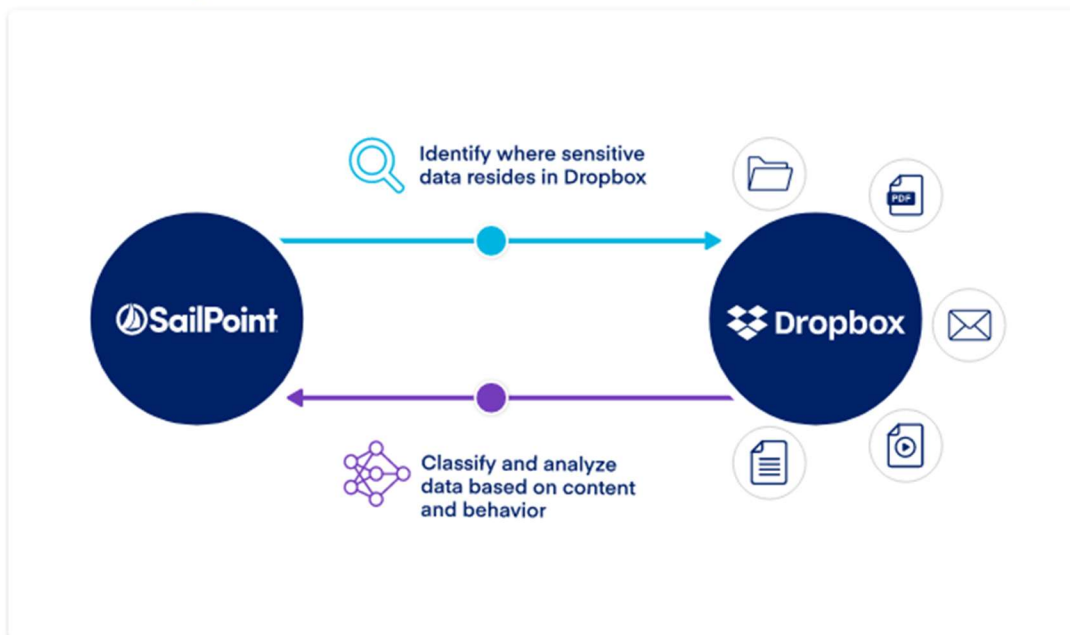
- Find and address your compliance gaps
- Ensure consistent governance across all data types
- Be alerted if inappropriate access is detected
- Minimize exposure to data leakage



Learn more

- How identity governance helps ensure GDPR compliance
- Governing unstructured data and data access
- Watch our Dropbox integration demo

Data discovery and classification



<https://www.sailpoint.com/integrations/dropbox/> (last visited December 16, 2021)

SailPoint is the leading provider of enterprise identity governance solutions. SailPoint's open identity platform enables organizations to manage the entire identity lifecycle of users accessing Dropbox Business, more efficiently control user access, securely prepare your data for migration to Dropbox Business, identify sensitive data, and monitor for malicious behavior.

- **Oversee and streamline who accesses Dropbox Business.** SailPoint identity governance solutions allow you to manage and govern users, groups, and entitlements for DropBox and automate the requesting and provisioning of user access. Improve security and audit performance by instantly reviewing and remediating access.
- **Monitor inappropriate access.** Centralize your visibility to users and their access across Dropbox and other cloud and on-premises applications. Admins can easily audit and ensure access is within corporate policy.
- **Gain greater data visibility.** Identify and classify data prior to migration to gain clear visibility across your data assets and optimize what data should be moved to Dropbox.
- **Clean up permissions.** Collect and analyze permissions to files for deeper insight into who has access and remediate inappropriate access issues to mitigate security risk to your content.
- **Establish data ownership.** Empower data owners who have the most intelligence about the data to take on a key role in managing access and ensure the right users have access to the right data.
- **Support hybrid environments.** Whether your data resides within your datacenter or in Dropbox Business, govern it with a centralized set of controls and policies.

<https://www.dropbox.com/app-integrations/sailpoint> (last visited December 16, 2021)

62. On information and belief, SailPoint, in order to develop, support, and provide its identity governance solutions, has directly infringed and continues to infringe, literally and/or

under the doctrine of equivalents, one or more claims of the '942 Patent under 35 U.S.C. § 271(a) by using Dropbox products and services in the United States and abroad without authority. Included in these acts of infringement are situations where devices associated with SailPoint interact with Dropbox servers to perform file sync operations.

DEFENDANT CLEAR CHANNEL

63. Clear Channel uses Dropbox's online document storage and synchronization products and services, including those products and services that infringe Plaintiff's patents identified above, through Dropbox's online platforms and mobile applications to customers, which practice each and every limitation of one or more claims of the '942 patent.


64. On its website, Dropbox identifies Clear Channel as a user of Dropbox Business.

**Businesses across the world of media trust
Dropbox**




<https://www.dropbox.com/business/solutions/media> (last visited December 16, 2021)



65. In 2017, Clear Channel along with Dropbox released a study disclosing Clear Channel's purchase of Dropbox Business licenses for its employees and summarizing the commercial benefits from the use of Dropbox Business. A copy of Clear Channel's study is attached as Exhibit 2.



Clear Channel Outdoor



In January 2017, Clear Channel purchased 1,200 enterprise Dropbox licenses to provide an enhanced security and collaboration experience for end users. Through three business cases, this study outlines the total return on investment achieved by Clear Channel as a result of its deployment. The study's findings are summarized below.

	\$677k total financial benefit		541% return on investment
---	--	---	-------------------------------------

<https://www.insightsforprofessionals.com/management/leadership/dropbox-businness-and-clear-channel-outdoor/download> (last visited December 16, 2021)

66. On information and belief, Clear Channel has directly infringed and continues to infringe, literally and/or under the doctrine of equivalents, one or more claims of the '942 Patent under 35 U.S.C. § 271(a) by using Dropbox products and services in the United States and abroad without authority. Included in these acts of infringement are situations where devices associated with Clear Channel interact with Dropbox servers to perform file sync operations.

COUNT THREE

INFRINGEMENT OF U.S. PATENT NO. 10,289,607

67. Plaintiff incorporates paragraphs 1 through 66 as though fully set forth herein.

68. Plaintiff is the owner by assignment of U.S. Patent No. 10,289,607 (the "'607 Patent"), entitled "Architecture For Management of Digital Files Across Distributed Network" issued on May 14, 2019. A copy of the '607 Patent is attached as Exhibit 4.

69. The '607 Patent is generally directed to systems and methods for sharing electronic files between multiple devices, wherein when a user modifies an electronic file on a device, a copy of the modified electronic file is automatically transferred to other devices, and wherein metadata associated with the modified electronic file is assigned a greater priority than the copy of the modified electronic file, and the metadata is automatically transferred to the other devices prior to the copy of the modified electronic file.

70. Plaintiff is the owner by assignment of all rights, title, and interest in and to the '607 Patent, including the right to assert all causes of action arising under the '607 Patent and the right to all remedies for the infringement of the '607 Patent.

71. For example, claim 1 of the '607 Patent states:

1. A system, comprising:

server system comprising one or more processors programmed with computer

program instructions that, when executed, cause the server system to:

receive, over a network, a copy of a first file from a first client device associated

with a user, wherein the copy of the first file is automatically received from

the first client device responsive to the user modifying a content of the first

file stored on the first client device, the copy of the first file being a version

of the first file that is generated from the user modifying the content of the

first file;

receive, from the first client device, first metadata associated with the version of

the first file that is generated from the user modifying the content of the first

file, the first metadata being assigned a first priority greater than a second

priority assigned to the copy of the first file;

determine that the server system is not in communication with a second client device associated with the user;

store the copy of the first file on the server system;

automatically transfer the first metadata to the second client device based on the first priority being greater than the second priority such that the first metadata is transferred to the second client device prior to the copy of the first file being transferred to the second client device; and

automatically transfer, over a network, the copy of the first file to the second client device associated with the user to replace an older version of the first file stored on the second client device, responsive to (i) resuming communication with the second client device and (ii) receiving the copy of the first file from the first client device.

DEFENDANT DROPBOX

72. In addition to the description in paragraphs 54-60, the Dropbox products and services include systems and methods for sharing electronic files between multiple devices, wherein when a user modifies an electronic file on a device, a copy of the modified electronic file is automatically transferred to other devices, and wherein metadata associated with the modified electronic file is assigned a greater priority than the copy of the modified electronic file, and the metadata is automatically transferred to the other devices prior to the copy of the modified electronic file.

73. Dropbox's server system receives, over a network, from a client device, metadata associated with the updated version of a file that is generated from the user modifying the content of the file, the metadata being assigned a priority greater than a priority assigned to the copy of the file.

74. Dropbox tracks and stores multiple types of metadata associated with stored files:

How to view metadata for a file in Dropbox

Metadata is information you can see about a file. It includes information like the date your file was created and the size of the file.

To see metadata about your file:

1. [Sign in](#) to dropbox.com.
2. Hover over the file and click the checkbox.
3. The file information will be located in the details pane of the right sidebar. If the metadata does not appear, click the arrow or **Info** button in the right sidebar.

In the details pane, you'll see different types of information depending on the type of file you're previewing. All files display the following metadata:

- The location of the file (**Saved In**)
- When the file was last modified (**Modified**)
- The size of the file in bytes (**Size**)
- The type of file (**Type**)

File-specific metadata

Some types of files display specific metadata in addition to the basic metadata listed above. For more information, refer to the lists of metadata by file type below.

- Image files:**
- Original date
 - Focal length
 - Shutter speed
 - Aperture
 - ISO
 - Metering
 - Flash
 - White balance
 - Camera make
 - Camera model
 - Lens model
 - Dimensions
 - Dots per Inch
 - Color profile
 - Artist
 - Copyright

- Microsoft Office files:**
- Title
 - Copyright Notice
 - Revision Number
 - Pages
 - Paragraphs
 - Words
 - Characters

<https://help.dropbox.com/files-folders/sort-preview/file-info> (last visited December 16, 2021):

75. As demonstrated by the Dropbox API, Dropbox tracks and stores additional types of metadata for files, including metadata (*e.g.*, `client_modified`) that is supplied by a client prior to uploading a modified file:

/get_metadata

VERSION

DESCRIPTION

Returns the metadata for a file or folder.

Note: Metadata for the root folder is unsupported.

FileMetadata

name *String* The last component of the path (including extension). This never contains a slash.

id *String(min_length=1)* A unique identifier for the file.

client_modified *Timestamp(format="%Y-%m-%dT%H:%M:%SZ")* For files, this is the modification time set by the desktop client when the file was added to Dropbox. Since this time is not verified (the Dropbox server stores whatever the desktop client sends up), this should only be used for display purposes (such as sorting) and not, for example, to determine if a file has changed or not.

server_modified *Timestamp(format="%Y-%m-%dT%H:%M:%SZ")* The last time the file was modified on Dropbox.

rev *String(min_length=9, pattern="[0-9a-f]+")* A unique identifier for the current revision of a file. This field is the same rev as elsewhere in the API and can be used to detect changes and avoid conflicts.

size *UInt64* The file size in bytes.

path_lower *String?* The lowercased full path in the user's Dropbox. This always starts with a slash. This field will be null if the file or folder is not mounted. This field is optional.

path_display *String?* The cased path to be used for display purposes only. In rare instances the casing will not correctly match the user's filesystem, but this behavior will match the path provided in the Core API v1, and at least the last path component will have the correct casing. Changes to only the casing of paths won't be returned by [list_folder/continue](#). This field will be null if the file or folder is not mounted. This field is optional.

parent_shared_folder_id deprecated *String(pattern="[-_0-9a-zA-Z:]+")?* Field is deprecated. Please use `FileSharingInfo.parent_shared_folder_id` or `FolderSharingInfo.parent_shared_folder_id` instead. This field is optional.

media_info *MediaInfo?* Additional information if the file is a photo or video. This field will not be set on entries returned by [list_folder](#), [list_folder/continue](#), or [get_thumbnail_batch](#), starting December 2, 2019. This field is optional.

symlink_info *SymlinkInfo?* Set if this file is a symlink. This field is optional.

sharing_info *FileSharingInfo?* Set if this file is contained in a shared folder. This field is optional.

is_downloadable *Boolean* If true, file can be downloaded directly; else the file must be exported. The default for this field is True.

export_info *ExportInfo?* Information about format this file can be exported to. This field must be set if `is_downloadable` is set to false. This field is optional.

property_groups *List of (PropertyGroup)?* Additional information if the file has custom properties with the property template specified. This field is optional.

has_explicit_shared_members *Boolean?* This flag will only be present if `include_has_explicit_shared_members` is true in `list_folder` or `get_metadata`. If this flag is present, it will be true if this file has any explicit shared members. This is different from `sharing_info` in that this could be true in the case where a file has explicit members but is not contained within a shared folder. This field is optional.

content_hash *String(min_length=64, max_length=64)?* A hash of the file content. This field can be used to verify data integrity. For more information see our [Content hash](#) page. This field is optional.

file_lock_info *FileLockMetadata?* If present, the metadata associated with the file's current lock. This field is optional.

https://www.dropbox.com/developers/documentation/http/documentation#files-get_metadata

(last visited December 16, 2021)

76. Metadata associated with files is stored separately from the file contents, and with a higher priority. When a file is modified on a client device, the client device will also upload new metadata for the file. Dropbox's notification system enforces a priority system where changes to file metadata are propagated faster than, and downloaded by other client devices before, changes to file content. Dropbox provides the following architecture:

Our file infrastructure is comprised of the following components:

Metadata servers

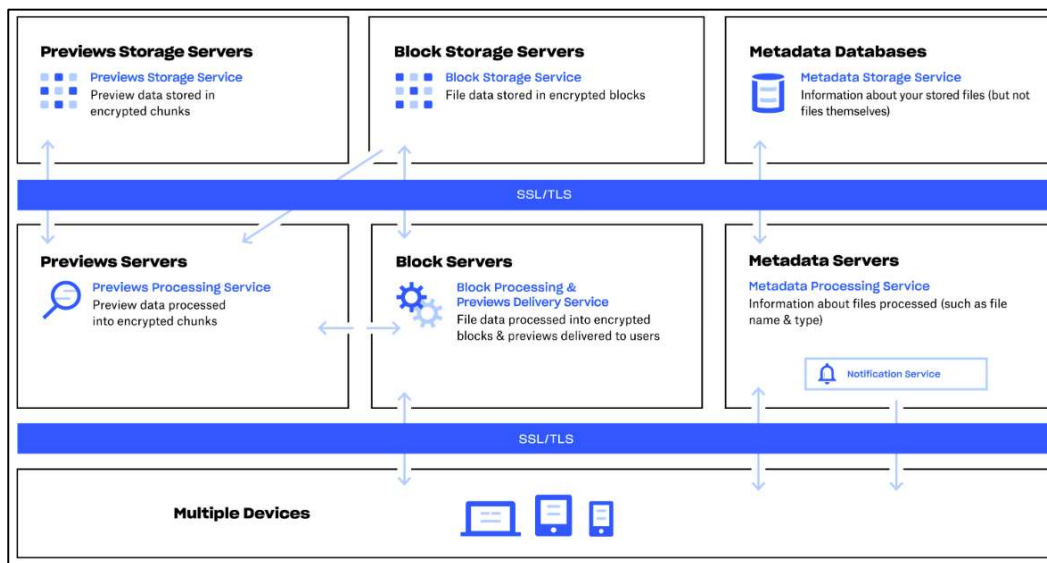
Certain basic information about user data, called metadata, is kept in its own discrete storage service and acts as an index for the data in users' accounts. Metadata includes basic account and user information, like email address, name, and device names. Metadata also includes basic information about files, including file names and types, that helps support features like version history, recovery, and sync.

Metadata Databases

File metadata is stored in a MySQL-backed database service, and is sharded and replicated as needed to meet performance and high availability requirements.

Notification service

This is a separate service dedicated to monitoring if changes have been made to Dropbox accounts. No file data or metadata is stored or transferred here. Each client establishes a long poll connection to the notification service and waits. When a change to any file in Dropbox takes place, the notification service signals a change to the relevant client(s) by closing the long poll connection. Closing the connection signals that the client must connect to the Metadata Servers securely to synchronize any changes.



<https://www.dropbox.com/business/trust/security/architecture> (last visited December 16, 2021)

77. Using the Dropbox API, metadata for a modified file will be uploaded with a higher priority from the sending client device to Dropbox's servers before the file contents, either using a single file upload call or sequential upload calls for larger files. The higher-priority metadata may be passed in an HTTP request header while the lower-priority file content may follow in the HTTP request body:

Content-upload endpoints

These endpoints accept file content in the request body, so their arguments are instead passed as JSON in the `Dropbox-API-Arg` request header or `arg` URL parameter. These endpoints are on the `content.dropboxapi.com` domain.

<https://www.dropbox.com/developers/documentation/http/documentation> (last visited December 16, 2021)

78. When a client device comes online in contact with Dropbox's servers, changed files will be automatically transferred to the device. This automatic transfer is responsive to determining that the client device is online and receiving the copy of the modified file from the source device:

Does Dropbox update and sync files automatically?

The Dropbox desktop app will update and sync files automatically any time you're connected to the internet. This makes sure you always have the latest version of your file across all linked devices. However, you can set specific files to not sync when you're online. By turning on selective sync, you can choose which files will update and sync automatically whenever an internet connection is detected.

<https://www.dropbox.com/features/sync/work-remotely-offline> (last visited December 16, 2021)

79. Upon information and belief, Dropbox has infringed and continues to infringe, literally or under the doctrine of equivalents, one or more claims of the '607 Patent by developing, distributing, operating, using, selling, and/or offering to sell Dropbox products and services in the United States without authority.

80. On information and belief, Dropbox is liable for infringement, literally and/or under the doctrine of equivalents, of one or more claims of the '607 Patent under 35 U.S.C. § 271(b) based on its active marketing and promotion of its Dropbox products and services in the United States to its customers and prospective customers. On information and belief Dropbox has, and will continue to, intentionally encourage acts of direct infringement with knowledge of the '607 Patent and knowledge that its acts are encouraging infringement.

81. On information and belief, Dropbox is liable for infringement, literally and/or under the doctrine of equivalents, of one or more claims of the '607 Patent under 35 U.S.C. §

271(c), because Dropbox has had, and continues to have, knowledge that its Dropbox products are especially developed or adapted for a use that infringes the '607 Patent and constitute a material part of the claimed systems and methods. Dropbox has had, and continues to have, knowledge that there are no substantial noninfringing uses for these Products. Dropbox has infringed and continues to infringe the '607 Patent directly or indirectly in violation of 35 U.S.C. § 271(c).

DEFENDANT SAILPOINT

82. SailPoint makes and sells valued added integration services and products for Dropbox's online document storage and synchronization products and services, including those products and services that infringe Plaintiff's patents identified above, through Dropbox's online platforms and mobile applications to customers, which practice each and every limitation of one or more claims of the '607 patent.

83. More specifically, SailPoint's Identity Governance Platform provides identity governance solutions that are integrated with Dropbox products and services, enabling users of Dropbox products and services to, among other things, control user access, identify sensitive data, and monitor for malicious behavior.

How does SailPoint integrate with Dropbox?

SailPoint collects and analyzes the data and file permissions in Dropbox to identify sensitive data and determine who has access to it. We can also alert you about inappropriate access to help you maintain security.

Examples

Data discovery and classification ▶

Permission analysis ▶


Access monitoring and alerting ▶

Identity for Dropbox

Could some of your files in Dropbox be a compliance risk?

SailPoint's integration with Dropbox helps you identify sensitive information, manage data ownership and alert you to any suspicious access activity. As more people collaborate and share information in the cloud, protecting access to your Dropbox files has never been more important.

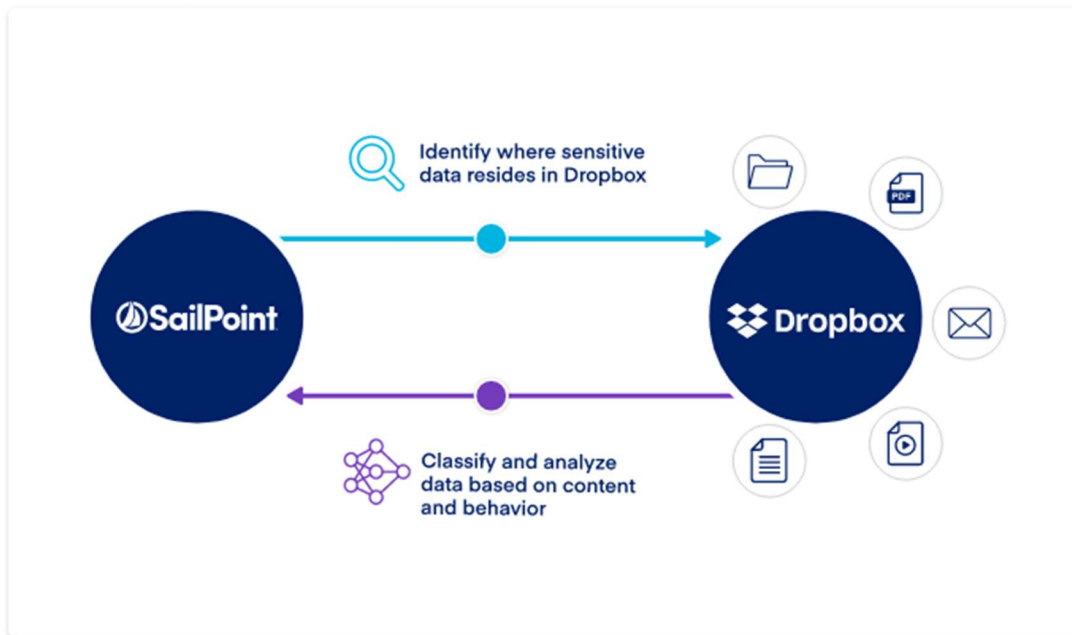
- Find and address your compliance gaps
- Ensure consistent governance across all data types
- Be alerted if inappropriate access is detected
- Minimize exposure to data leakage



Learn more

- How identity governance helps ensure GDPR compliance
- Governing unstructured data and data access
- Watch our Dropbox integration demo

Data discovery and classification



<https://www.sailpoint.com/integrations/dropbox/> (last visited December 16, 2021)

SailPoint is the leading provider of enterprise identity governance solutions. SailPoint's open identity platform enables organizations to manage the entire identity lifecycle of users accessing Dropbox Business, more efficiently control user access, securely prepare your data for migration to Dropbox Business, identify sensitive data, and monitor for malicious behavior.

- **Oversee and streamline who accesses Dropbox Business.** SailPoint identity governance solutions allow you to manage and govern users, groups, and entitlements for DropBox and automate the requesting and provisioning of user access. Improve security and audit performance by instantly reviewing and remediating access.
- **Monitor inappropriate access.** Centralize your visibility to users and their access across Dropbox and other cloud and on-premises applications. Admins can easily audit and ensure access is within corporate policy.
- **Gain greater data visibility.** Identify and classify data prior to migration to gain clear visibility across your data assets and optimize what data should be moved to Dropbox.
- **Clean up permissions.** Collect and analyze permissions to files for deeper insight into who has access and remediate inappropriate access issues to mitigate security risk to your content.
- **Establish data ownership.** Empower data owners who have the most intelligence about the data to take on a key role in managing access and ensure the right users have access to the right data.
- **Support hybrid environments.** Whether your data resides within your datacenter or in Dropbox Business, govern it with a centralized set of controls and policies.

<https://www.dropbox.com/app-integrations/sailpoint> (last visited December 16, 2021)

84. On information and belief, SailPoint, in order to develop, support, and provide its identity governance solutions, has directly infringed and continues to infringe, literally and/or

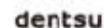
under the doctrine of equivalents, one or more claims of the '607 Patent under 35 U.S.C. § 271(a) by using Dropbox products and services in the United States and abroad without authority. Included in these acts of infringement are situations where devices associated with SailPoint interact with Dropbox servers to perform file sync operations.

DEFENDANT CLEAR CHANNEL

85. Clear Channel uses Dropbox's online document storage and synchronization products and services, including those products and services that infringe Plaintiff's patents identified above, through Dropbox's online platforms and mobile applications to customers, which practice each and every limitation of one or more claims of the '607 patent.


86. On its website, Dropbox identifies Clear Channel as a user of Dropbox Business.

**Businesses across the world of media trust
Dropbox**




<https://www.dropbox.com/business/solutions/media> (last visited December 16, 2021)



87. In 2017, Clear Channel along with Dropbox released a study disclosing Clear Channel's purchase of Dropbox Business licenses for its employees and summarizing the commercial benefits from the use of Dropbox Business. A copy of Clear Channel's study is attached as Exhibit 2.



Clear Channel Outdoor



In January 2017, Clear Channel purchased 1,200 enterprise Dropbox licenses to provide an enhanced security and collaboration experience for end users. Through three business cases, this study outlines the total return on investment achieved by Clear Channel as a result of its deployment. The study's findings are summarized below.

	\$677k total financial benefit		541% return on investment
---	--	---	-------------------------------------

<https://www.insightsforprofessionals.com/management/leadership/dropbox-businness-and-clear-channel-outdoor/download> (last visited December 16, 2021)

88. On information and belief, Clear Channel has directly infringed and continues to infringe, literally and/or under the doctrine of equivalents, one or more claims of the '607 Patent under 35 U.S.C. § 271(a) by using Dropbox products and services in the United States and abroad without authority. Included in these acts of infringement are situations where devices associated with Clear Channel interact with Dropbox servers to perform file sync operations.

COUNT FOUR

INFRINGEMENT OF U.S. PATENT NO. 10,642,787

89. Plaintiff incorporates paragraphs 1 through 88 as though fully set forth herein.

90. Plaintiff is the owner by assignment of U.S. Patent No. 10,642,787 (the "'787 Patent"), entitled "Pre-file-transfer update based on prioritized metadata" issued on May 5, 2020.

A copy of the '787 Patent is attached as Exhibit 5.

91. The '787 Patent is generally directed to systems and methods for sharing electronic files between multiple devices, wherein when a user modifies an electronic file on a device, metadata associated with the modified electronic file is automatically transferred to other devices with higher priority before a copy of the modified electronic file is transferred to the other devices, which causes a user interface of the other devices to indicate the updated version of the modified electronic file.

92. Plaintiff is the owner by assignment of all rights, title, and interest in and to the '787 Patent, including the right to assert all causes of action arising under the '787 Patent and the right to all remedies for the infringement of the '787 Patent.

93. For example, claim 1 of the '787 Patent states:

1. A system comprising:

a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to:

receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being an updated version of the first file that is generated from the user modifying the content of the first file;

receive, over a network, from the first client device, first metadata associated with the updated version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file; and

automatically transfer, based on the first priority being greater than the second priority, the first metadata over a network to a second client device associated with the user such that the first metadata is transferred to the second client device before the copy of the first file is transferred to the second client device,

wherein, before the copy of the first file is transferred to the second client device:

- (i) the transfer of the first metadata to the second client device causes a file representation of the first file presented on a user interface of the second client device to be updated based on the first metadata, and
- (ii) instead of the updated file representation of the first file representing a version of the first file currently stored on the second client device, the updated file representation represents the updated version of the first file that is currently stored on the first client device and not currently stored on the second client device, and

wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.

DEFENDANT DROPBOX

94. In addition to the description in paragraphs 73-82, Dropbox's server system comprises one or more processors running Dropbox server software, and Dropbox's file sharing software running on a first client device and a second client device configure the first client

device and the second client device to sync with each other. When a user modifies a content of a first file stored on the first client device, Dropbox's server system receives metadata, with higher priority, about the modified first file from the first client device, and automatically transfers the metadata to the second client device before a copy of the modified first file, with lower priority, is transferred to the second client device.

95. Based on the higher-priority metadata transferred from the Dropbox server system to the second client device, a file representation of the first file presented on a Dropbox user interface of the second client device is updated, before the copy of the modified first file is transferred to the second client device. The updated file representation represents the updated version of the first file that is currently stored on the first client device and not currently stored on the second client device.

96. Based on the metadata downloaded to the second client device in association with the modified first file, and prior to receiving the modified first file, a Dropbox user interface of the second client device displays a graphical annotation, indicating that the updated version of the first file is available for download. Dropbox provides the following sync icons and symbols in the Dropbox folder and in the taskbar (Windows) or menu bar (Mac):

The sync icons in the Dropbox folder

Below are the sync icons that appear on files and folders in your Dropbox folder in File Explorer (Windows) or Finder (Mac).



Synced and local

A solid green circle with a white checkmark means your file or folder is fully synced and local. "Synced" means that any changes you made to this file or folder are reflected everywhere you access your files in Dropbox. "Local" means that your file or folder is available when you're not connected to internet.



Sync in progress

A solid blue circle with two white arrows going in a circle means that your file or folder is in the process of updating. If you chose to add it to your hard drive with the selective sync feature, this icon could mean that it's still in the process of syncing to your hard drive. If you use the Smart Sync feature, this icon could mean that your file is in the process of changing its sync status between online-only and local.

The sync icons in the taskbar or menu bar

Below are the different Dropbox icons in your taskbar (Windows) or menu bar (Mac), which is visible when the Dropbox desktop app is open on your computer. The icon color may vary, depending on your operating system.



Fully synced

A solid black Dropbox icon with no other icon means the Dropbox files and folders on your computer are fully up to date. Any changes you made are reflected everywhere you use Dropbox.

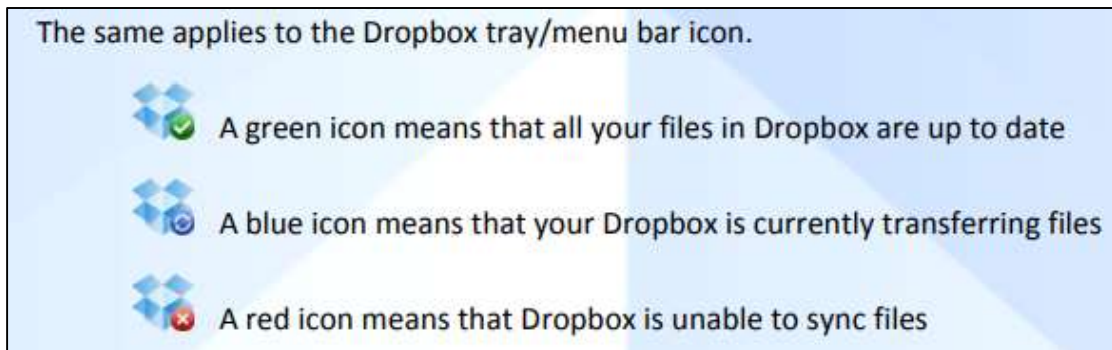
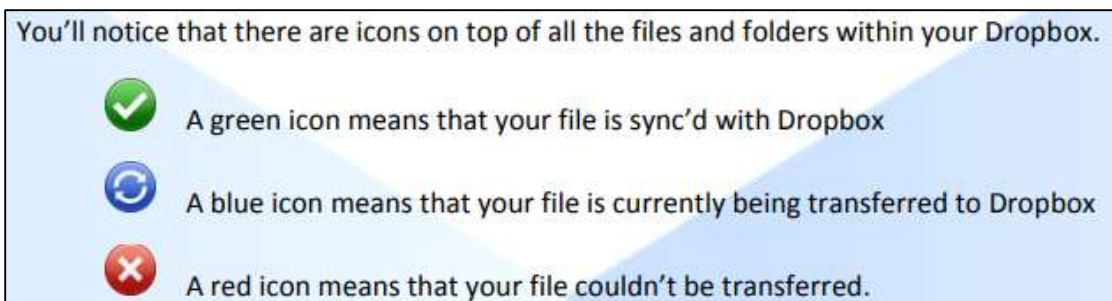


Sync in progress

A solid black circle with two white arrows going in a circle means that the Dropbox files and folders on your computer are in the process of updating. Any changes you made are updating everywhere you access your files in Dropbox.

<https://help.dropbox.com/installs-integrations/sync-uploads/sync-icons> (last visited

December 16, 2021).



https://www.totalestimating.com/Dropbox_guide.pdf (last visited October 13, 2021)

97. The updated file representation indicates that the updated version of the first file from the first client device is available for download from the Dropbox's server system to the second client device.

98. As described in paragraphs 77-79, Dropbox's server system has a configuration to assign greater priority to metadata than priority assigned to the files. Dropbox's notification system enforces a priority system where changes to file metadata are propagated faster than, and downloaded by other client devices before, changes to file content.

99. Upon information and belief, Dropbox has infringed and continues to infringe, literally or under the doctrine of equivalents, one or more claims of the '787 Patent by developing, distributing, operating, using, selling, and/or offering to sell Dropbox products and services in the United States without authority.

100. On information and belief, Dropbox is liable for infringement, literally and/or under the doctrine of equivalents, of one or more claims of the '787 Patent under 35 U.S.C. §

271(b) based on its active marketing and promotion of its Dropbox products and services in the United States to its customers and prospective customers. On information and belief Dropbox has, and will continue to, intentionally encourage acts of direct infringement with knowledge of the '787 Patent and knowledge that its acts are encouraging infringement.

101. On information and belief, Dropbox is liable for infringement, literally and/or under the doctrine of equivalents, of one or more claims of the '787 Patent under 35 U.S.C. § 271(c), because Dropbox has had, and continues to have, knowledge that its Dropbox products are especially developed or adapted for a use that infringes the '787 Patent and constitute a material part of the claimed systems and methods. Dropbox has had, and continues to have, knowledge that there are no substantial noninfringing uses for these Products. Dropbox has infringed and continues to infringe the '787 Patent directly or indirectly in violation of 35 U.S.C. § 271(c).

DEFENDANT SAILPOINT

102. SailPoint makes and sells valued added integration services and products for Dropbox's online document storage and synchronization products and services, including those products and services that infringe Plaintiff's patents identified above, through Dropbox's online platforms and mobile applications to customers, which practice each and every limitation of one or more claims of the '787 patent.

103. More specifically, SailPoint's Identity Governance Platform provides identity governance solutions that are integrated with Dropbox products and services, enabling users of Dropbox products and services to, among other things, control user access, identify sensitive

data, and monitor for malicious behavior.

How does SailPoint integrate with Dropbox?

SailPoint collects and analyzes the data and file permissions in Dropbox to identify sensitive data and determine who has access to it. We can also alert you about inappropriate access to help you maintain security.

Examples

Data discovery and classification



Permission analysis



Access monitoring and alerting




Identity for Dropbox

Could some of your files in Dropbox be a compliance risk?

SailPoint's integration with Dropbox helps you identify sensitive information, manage data ownership and alert you to any suspicious access activity. As more people collaborate and share information in the cloud, protecting access to your Dropbox files has never been more important.

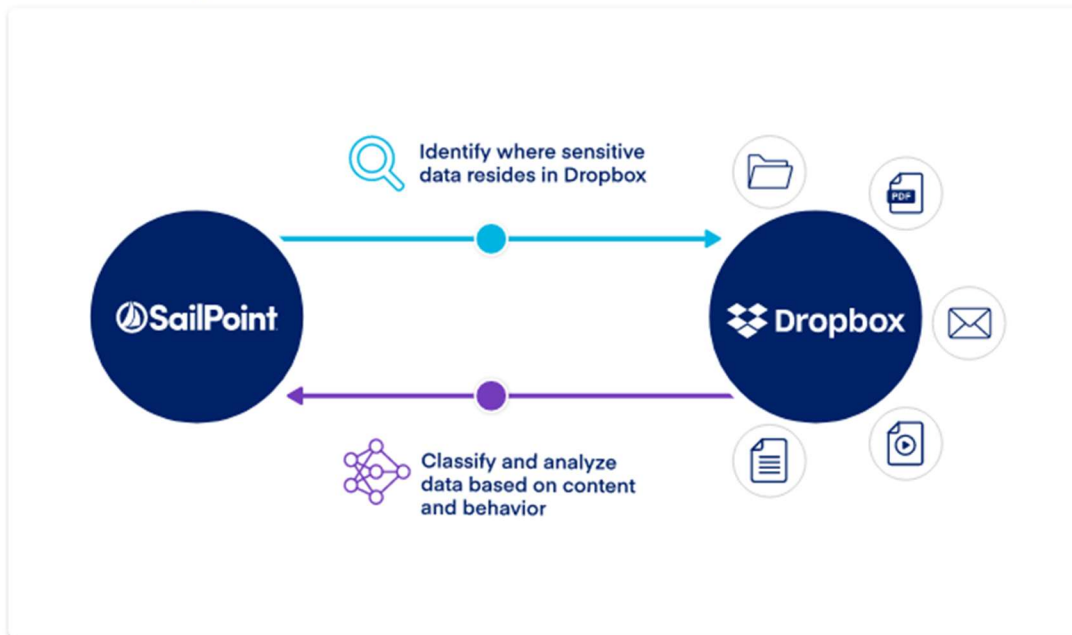
- Find and address your compliance gaps
- Ensure consistent governance across all data types
- Be alerted if inappropriate access is detected
- Minimize exposure to data leakage



Learn more

- How identity governance helps ensure GDPR compliance
- Governing unstructured data and data access
- Watch our Dropbox integration demo

Data discovery and classification



<https://www.sailpoint.com/integrations/dropbox/> (last visited December 16, 2021)

SailPoint is the leading provider of enterprise identity governance solutions. SailPoint's open identity platform enables organizations to manage the entire identity lifecycle of users accessing Dropbox Business, more efficiently control user access, securely prepare your data for migration to Dropbox Business, identify sensitive data, and monitor for malicious behavior.

- **Oversee and streamline who accesses Dropbox Business.** SailPoint identity governance solutions allow you to manage and govern users, groups, and entitlements for DropBox and automate the requesting and provisioning of user access. Improve security and audit performance by instantly reviewing and remediating access.
- **Monitor inappropriate access.** Centralize your visibility to users and their access across Dropbox and other cloud and on-premises applications. Admins can easily audit and ensure access is within corporate policy.
- **Gain greater data visibility.** Identify and classify data prior to migration to gain clear visibility across your data assets and optimize what data should be moved to Dropbox.
- **Clean up permissions.** Collect and analyze permissions to files for deeper insight into who has access and remediate inappropriate access issues to mitigate security risk to your content.
- **Establish data ownership.** Empower data owners who have the most intelligence about the data to take on a key role in managing access and ensure the right users have access to the right data.
- **Support hybrid environments.** Whether your data resides within your datacenter or in Dropbox Business, govern it with a centralized set of controls and policies.

<https://www.dropbox.com/app-integrations/sailpoint> (last visited December 16, 2021)

104. On information and belief, SailPoint, in order to develop, support, and provide its identity governance solutions, has directly infringed and continues to infringe, literally and/or

under the doctrine of equivalents, one or more claims of the '787 Patent under 35 U.S.C. § 271(a) by using Dropbox products and services in the United States and abroad without authority. Included in these acts of infringement are situations where devices associated with SailPoint interact with Dropbox servers to perform file sync operations.

DEFENDANT CLEAR CHANNEL

105. Clear Channel uses Dropbox's online document storage and synchronization products and services, including those products and services that infringe Plaintiff's patents identified above, through Dropbox's online platforms and mobile applications to customers, which practice each and every limitation of one or more claims of the '787 Patent.


106. On its website, Dropbox identifies Clear Channel as a user of Dropbox Business.

**Businesses across the world of media trust
Dropbox**




<https://www.dropbox.com/business/solutions/media> (last visited December 16, 2021)



107. In 2017, Clear Channel along with Dropbox released a study disclosing Clear Channel's purchase of Dropbox Business licenses for its employees and summarizing the commercial benefits from the use of Dropbox Business. A copy of Clear Channel's study is attached as Exhibit 2.



Clear Channel Outdoor



In January 2017, Clear Channel purchased 1,200 enterprise Dropbox licenses to provide an enhanced security and collaboration experience for end users. Through three business cases, this study outlines the total return on investment achieved by Clear Channel as a result of its deployment. The study's findings are summarized below.

	\$677k total financial benefit		541% return on investment
---	--	---	-------------------------------------

<https://www.insightsforprofessionals.com/management/leadership/dropbox-businness-and-clear-channel-outdoor/download> (last visited December 16, 2021)

108. On information and belief, Clear Channel has directly infringed and continues to infringe, literally and/or under the doctrine of equivalents, one or more claims of the '787 Patent under 35 U.S.C. § 271(a) by using Dropbox products and services in the United States and abroad without authority. Included in these acts of infringement are situations where devices associated with Clear Channel interact with Dropbox servers to perform file sync operations.

COUNT FIVE

INFRINGEMENT OF U.S. PATENT NO. 10,754,823

109. Plaintiff incorporates paragraphs 1 through 108 as though fully set forth herein.

110. Plaintiff is the owner, by assignment of U.S. Patent No. 10,754,823 (the "'823 Patent"), entitled "Pre-file-transfer availability indication based on prioritized metadata" issued on August 25, 2020. A copy of the '823 Patent is attached as Exhibit 6.

111. The '823 Patent is generally directed to systems and methods for sharing electronic files between multiple devices, wherein when a user modifies an electronic file on a device, metadata associated with the modified electronic file is automatically transferred to other devices with higher priority, which causes a graphical availability indication of the updated version of the modified electronic file to be presented on the other devices, and subsequently the copy of the modified electronic file is transferred to the other devices.

112. Plaintiff is the owner by assignment of all rights, title, and interest in and to the '823 Patent, including the right to assert all causes of action arising under the '823 Patent and the right to all remedies for the infringement of the '823 Patent.

113. For example, claim 1 of the '823 Patent states:

1. A system comprising:

a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to:

receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being an updated version of the first file that is generated from the user modifying the content of the first file;

receive, over a network, from the first client device, first metadata associated with the updated version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;

automatically transfer, based on the first priority being greater than the second priority, the first metadata over a network to a second client device associated with the user such that the first metadata is transferred to the second client device before the copy of the first file is transferred to the second client device,

wherein, before the copy of the first file is transferred to the second client device:

- (i) the transfer of the first metadata to the second client device causes a graphical availability indication of the updated version of the first file to be presented at the second client device based on the first metadata, and
- (ii) the graphical availability indication is presented proximate a file icon representing the first file on a user interface of the second client device, and

wherein the graphical availability indication indicates that the updated version of the first file generated from the user modifying the content of the first file is available to be downloaded from the server system to the second client device; and

subsequent to the transfer of the first metadata to the second client device, transfer the copy of the first file to the second client device,

wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.

DEFENDANT DROPBOX

114. In addition to the description in paragraphs 95-102, Dropbox's server system includes one or more processors running Dropbox server software and Dropbox's file sharing software running on a first client device and a second client device configure the first client device and the second client device to sync with each other. When a user modifies a content of a first file stored on the first client device, Dropbox's server system receives metadata, with higher priority, about the modified first file from the first client device, and automatically transfers the metadata to the second client device before a copy of the modified first file, with lower priority, is transferred to the second client device.

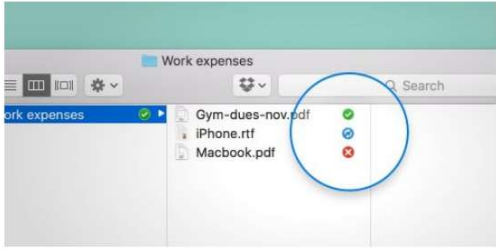
115. Based on the higher-priority metadata transferred from the Dropbox server system to the second client device, a graphical availability indication of the updated version of the first file is presented proximate a file icon representing the first file on a Dropbox user interface of the second client device, before the copy of the modified first file is transferred to the second client device. The graphical availability indication represents that the updated version of the first file is available to be downloaded from the Dropbox server system to the second client device.

116. Based on the metadata downloaded from Dropbox's servers to the second client device, and prior to receiving the modified file, graphical availability indications are presented next to file icons on Dropbox user interface of the second client device to notify the status of syncing:

Understanding Dropbox syncing icons

We add status icons next to your files and folders so you know they'll be accessible on your other devices. Here's what each icon means:

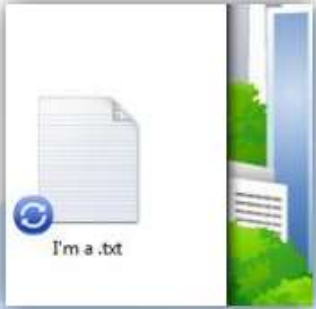
- **Green tick:** all of your files have been saved on our website and are accessible from any device.
- **Blue arrows:** your files are in the process of being saved to our website and your other devices. The speed of your Internet connection, the size of the files and the number of files all affect how long this takes.
- **Red X:** something isn't working properly and your files aren't being synced. Give it a bit of time and check the status later. In most cases, the problem will fix itself but, if it persists, please contact our customer support team.




https://www.dropbox.com/en_GB/lp/pro/pro_onboarding_desktop_app (last visited December 16, 2021)

117. Subsequent to the metadata transfer, Dropbox's server system transfers the modified file to the linked device. Once the transfer is completed, the blue icon with two white arrows going in a circle indicating sync in progress is changed to a green icon with a white checkmark indicating that the file has been updated.

Step 2 The blue icon means your file is syncing with Dropbox. You can check your progress by clicking on the Dropbox tray/menu bar icon.



Step 3 A green icon means that your file has finished syncing and is now available from your other computers and the [web](#).



https://www.total estimating.com/Dropbox_guide.pdf (last visited December 16, 2021)

118. Upon information and belief, Dropbox has infringed and continues to infringe, literally or under the doctrine of equivalents, one or more claims of the '823 Patent by developing, distributing, operating, using, selling, and/or offering to sell Dropbox products and services in the United States without authority.

119. On information and belief, Dropbox is liable for infringement, literally and/or under the doctrine of equivalents, of one or more claims of the '823 Patent under 35 U.S.C. § 271(b) based on its active marketing and promotion of its Dropbox products and services in the United States to its customers and prospective customers. On information and belief Dropbox has, and will continue to, intentionally encourage acts of direct infringement with knowledge of the '823 Patent and knowledge that its acts are encouraging infringement.

120. On information and belief, Dropbox is liable for infringement, literally and/or under the doctrine of equivalents, of one or more claims of the '823 Patent under 35 U.S.C. § 271(c), because Dropbox has had, and continues to have, knowledge that its Dropbox products are especially developed or adapted for a use that infringes the '823 Patent and constitute a material part of the claimed systems and methods. Dropbox has had, and continues to have, knowledge that there are no substantial noninfringing uses for these Products. Dropbox has infringed and continues to infringe the '823 Patent directly or indirectly in violation of 35 U.S.C. § 271(c).

DEFENDANT SAILPOINT

121. SailPoint makes and sells valued added integration services and products for Dropbox's online document storage and synchronization products and services, including those products and services that infringe Plaintiff's patents identified above, through Dropbox's online platforms and mobile applications to customers, which practice each and every limitation of one or more claims of the '823 patent.

122. More specifically, SailPoint’s Identity Governance Platform provides identity governance solutions that are integrated with Dropbox products and services, enabling users of Dropbox products and services to, among other things, control user access, identify sensitive data, and monitor for malicious behavior.

How does SailPoint integrate with Dropbox?

SailPoint collects and analyzes the data and file permissions in Dropbox to identify sensitive data and determine who has access to it. We can also alert you about inappropriate access to help you maintain security.

Examples

Data discovery and classification ▶

Permission analysis ▶


Access monitoring and alerting ▶

Identity for Dropbox

Could some of your files in Dropbox be a compliance risk?

SailPoint's integration with Dropbox helps you identify sensitive information, manage data ownership and alert you to any suspicious access activity. As more people collaborate and share information in the cloud, protecting access to your Dropbox files has never been more important.

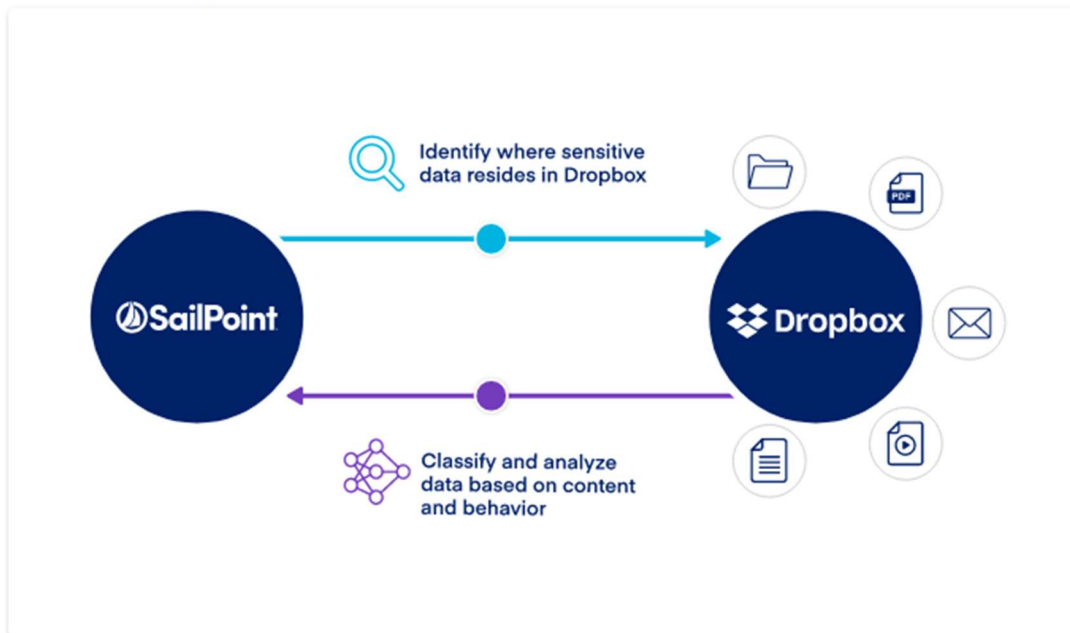
- Find and address your compliance gaps
- Ensure consistent governance across all data types
- Be alerted if inappropriate access is detected
- Minimize exposure to data leakage



Learn more

- How identity governance helps ensure GDPR compliance
- Governing unstructured data and data access
- Watch our Dropbox integration demo

Data discovery and classification



<https://www.sailpoint.com/integrations/dropbox/> (last visited December 16, 2021)

SailPoint is the leading provider of enterprise identity governance solutions. SailPoint's open identity platform enables organizations to manage the entire identity lifecycle of users accessing Dropbox Business, more efficiently control user access, securely prepare your data for migration to Dropbox Business, identify sensitive data, and monitor for malicious behavior.

- **Oversee and streamline who accesses Dropbox Business.** SailPoint identity governance solutions allow you to manage and govern users, groups, and entitlements for DropBox and automate the requesting and provisioning of user access. Improve security and audit performance by instantly reviewing and remediating access.
- **Monitor inappropriate access.** Centralize your visibility to users and their access across Dropbox and other cloud and on-premises applications. Admins can easily audit and ensure access is within corporate policy.
- **Gain greater data visibility.** Identify and classify data prior to migration to gain clear visibility across your data assets and optimize what data should be moved to Dropbox.
- **Clean up permissions.** Collect and analyze permissions to files for deeper insight into who has access and remediate inappropriate access issues to mitigate security risk to your content.
- **Establish data ownership.** Empower data owners who have the most intelligence about the data to take on a key role in managing access and ensure the right users have access to the right data.
- **Support hybrid environments.** Whether your data resides within your datacenter or in Dropbox Business, govern it with a centralized set of controls and policies.

<https://www.dropbox.com/app-integrations/sailpoint> (last visited December 16, 2021)

123. On information and belief, SailPoint, in order to develop, support, and provide its identity governance solutions, has directly infringed and continues to infringe, literally and/or

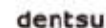
under the doctrine of equivalents, one or more claims of the '823 Patent under 35 U.S.C. § 271(a) by using Dropbox products and services in the United States and abroad without authority. Included in these acts of infringement are situations where devices associated with SailPoint interact with Dropbox servers to perform file sync operations.

DEFENDANT CLEAR CHANNEL

124. Clear Channel uses Dropbox's online document storage and synchronization products and services, including those products and services that infringe Plaintiff's patents identified above, through Dropbox's online platforms and mobile applications to customers, which practice each and every limitation of one or more claims of the '823 patent.


125. On its website, Dropbox identifies Clear Channel as a user of Dropbox Business.

**Businesses across the world of media trust
Dropbox**




<https://www.dropbox.com/business/solutions/media> (last visited December 16, 2021)



126. In 2017, Clear Channel along with Dropbox released a study disclosing Clear Channel's purchase of Dropbox Business licenses for its employees and summarizing the commercial benefits from the use of Dropbox Business. A copy of Clear Channel's study is attached as Exhibit 2.



Clear Channel Outdoor



In January 2017, Clear Channel purchased 1,200 enterprise Dropbox licenses to provide an enhanced security and collaboration experience for end users. Through three business cases, this study outlines the total return on investment achieved by Clear Channel as a result of its deployment. The study's findings are summarized below.

	\$677k total financial benefit		541% return on investment
---	--	---	-------------------------------------

<https://www.insightsforprofessionals.com/management/leadership/dropbox-businness-and-clear-channel-outdoor/download> (last visited December 16, 2021)

127. On information and belief, Clear Channel has directly infringed and continues to infringe, literally and/or under the doctrine of equivalents, one or more claims of the '823 Patent under 35 U.S.C. § 271(a) by using Dropbox products and services in the United States and abroad without authority. Included in these acts of infringement are situations where devices associated with Clear Channel interact with Dropbox servers to perform file sync operations.

COUNT SIX

INFRINGEMENT OF U.S. PATENT NO. 11,003,622

128. Plaintiff incorporates paragraphs 1 through 127 as though fully set forth herein.

129. Plaintiff is the owner by assignment of U.S. Patent No. 11,003,622 (the "'622 Patent"), entitled "Architecture For Management of Digital Files Across Distributed Network" issued on May 11, 2021. A copy of the '622 Patent is attached as Exhibit 7.

130. The '622 Patent is generally directed to systems and methods for sharing electronic files between multiple devices, wherein when a user modifies an electronic file on a device, metadata associated with the modified electronic file is first automatically transferred to other devices with higher priority and a copy of the modified file is automatically transferred to the other devices to replace an older version of the electronic file stored on the other devices.

131. Plaintiff is the owner by assignment of all rights, title, and interest in and to the '622 Patent, including the right to assert all causes of action arising under the '622 Patent and the right to all remedies for the infringement of the '622 Patent.

132. For example, claim 1 of the '622 Patent states:

1. A system comprising:

a server system comprising one or more processors programmed with computer

program instructions that, when executed, cause the server system to:

receive, over a network, a copy of a first file from a first client device associated

with a user, wherein the copy of the first file is automatically received from

the first client device responsive to the user modifying a content of the first

file stored on the first client device, the copy of the first file being a version

of the first file that is generated from the user modifying the content of the

first file;

store the copy of the first file on the server system;

receive, from the first client device, first metadata associated with the version of

the first file that is generated from the user modifying the content of the first

file, the first metadata being assigned a first priority greater than a second

priority assigned to the copy of the first file;

automatically transfer, based on the first priority being greater than the second priority, the first metadata to the second client device such that the first metadata is transferred to the second client device prior to the copy of the first file being transferred to the second client device; and

automatically transfer, over a network, the copy of the first file to the second client device associated with the user to replace an older version of the first file stored on the second client device, responsive to receiving the copy of the first file from the first client device.

DEFENDANT DROPBOX

133. In addition to the description in paragraphs 115-121, Dropbox's server system comprises one or more processors running Dropbox server software and Dropbox's file sharing software running on a first client device and a second client device configure the first client device and the second client device to sync with each other. When a user modifies a content of a first file stored on the first client device, Dropbox's server system receives metadata, with higher priority, about the modified first file from the first client device, and automatically transfers the metadata to the second client device before a copy of the modified first file, with lower priority, is transferred to the second client device. Subsequently, Dropbox's server system automatically transfers the copy of the modified first file to the second client device to replace an older version of the first file stored on the second client device.

134. Dropbox's servers automatically update and sync modified files across linked devices, responsive to receiving the copy of the modified file from any one of the linked devices:

Does Dropbox update and sync files automatically?

The Dropbox desktop app will update and sync files automatically any time you're connected to the internet. This makes sure you always have the latest version of your file across all linked devices. However, you can set specific files to not sync when you're online. By turning on selective sync, you can choose which files will update and sync automatically whenever an internet connection is detected.

<https://www.dropbox.com/features/sync/work-remotely-offline> (last visited December 16, 2021)

135. Dropbox's file sync feature automatically transfers the updated version of the file, received from the first client device, to the second client device to replace an older version of the file stored on the second client device. Thus, any changes to a file made in the first client device is automatically transferred to the second client device to replace the previous version of that file in the second client device. The synced file is saved on the hard drive of the second client device:

Can I use Dropbox syncing to move my files to a new computer?

After you follow the steps above to get started, your files are synced to Dropbox, so you don't need to move or transfer them manually between computers or devices anymore. You can access them from any device through [dropbox.com](https://www.dropbox.com) or the Dropbox desktop and mobile apps.

If you'd like your files saved on a computer's hard drive (or multiple computers' hard drives), as well as to your Dropbox account online, you can choose to do so when you download the Dropbox desktop app. When prompted, choose "local" instead of "online-only". You can also change a computer's Smart Sync settings to "local" in your desktop app preferences at any time.

<https://help.dropbox.com/installs-integrations/sync-uploads/sync-overview> (last visited December 16, 2021)

136. Upon information and belief, Dropbox has infringed and continues to infringe, literally or under the doctrine of equivalents, one or more claims of the '622 Patent by developing, distributing, operating, using, selling, and/or offering to sell Dropbox products and services in the United States without authority.

137. On information and belief, Dropbox is liable for infringement, literally and/or under the doctrine of equivalents, of one or more claims of the '622 Patent under 35 U.S.C. § 271(b) based on its active marketing and promotion of its Dropbox products and services in the United States to its customers and prospective customers. On information and belief Dropbox has, and will continue to, intentionally encourage acts of direct infringement with knowledge of the '622 Patent and knowledge that its acts are encouraging infringement.

138. On information and belief, Dropbox is liable for infringement, literally and/or under the doctrine of equivalents, of one or more claims of the '622 Patent under 35 U.S.C. § 271(c), because Dropbox has had, and continues to have, knowledge that its Dropbox products are especially developed or adapted for a use that infringes the '622 Patent and constitute a material part of the claimed systems and methods. Dropbox has had, and continues to have, knowledge that there are no substantial noninfringing uses for these Products. Dropbox has infringed and continues to infringe the '622 Patent directly or indirectly in violation of 35 U.S.C. § 271(c).

DEFENDANT SAILPOINT

139. SailPoint makes and sells valued added integration services and products for Dropbox's online document storage and synchronization products and services, including those products and services that infringe Plaintiff's patents identified above, through Dropbox's online platforms and mobile applications to customers, which practice each and every limitation of one or more claims of the '622 patent.

140. More specifically, SailPoint's Identity Governance Platform provides identity governance solutions that are integrated with Dropbox products and services, enabling users of Dropbox products and services to, among other things, control user access, identify sensitive data, and monitor for malicious behavior.

How does SailPoint integrate with Dropbox?

SailPoint collects and analyzes the data and file permissions in Dropbox to identify sensitive data and determine who has access to it. We can also alert you about inappropriate access to help you maintain security.

Examples

Data discovery and classification ▶

Permission analysis ▶


Access monitoring and alerting ▶

Identity for Dropbox

Could some of your files in Dropbox be a compliance risk?

SailPoint's integration with Dropbox helps you identify sensitive information, manage data ownership and alert you to any suspicious access activity. As more people collaborate and share information in the cloud, protecting access to your Dropbox files has never been more important.

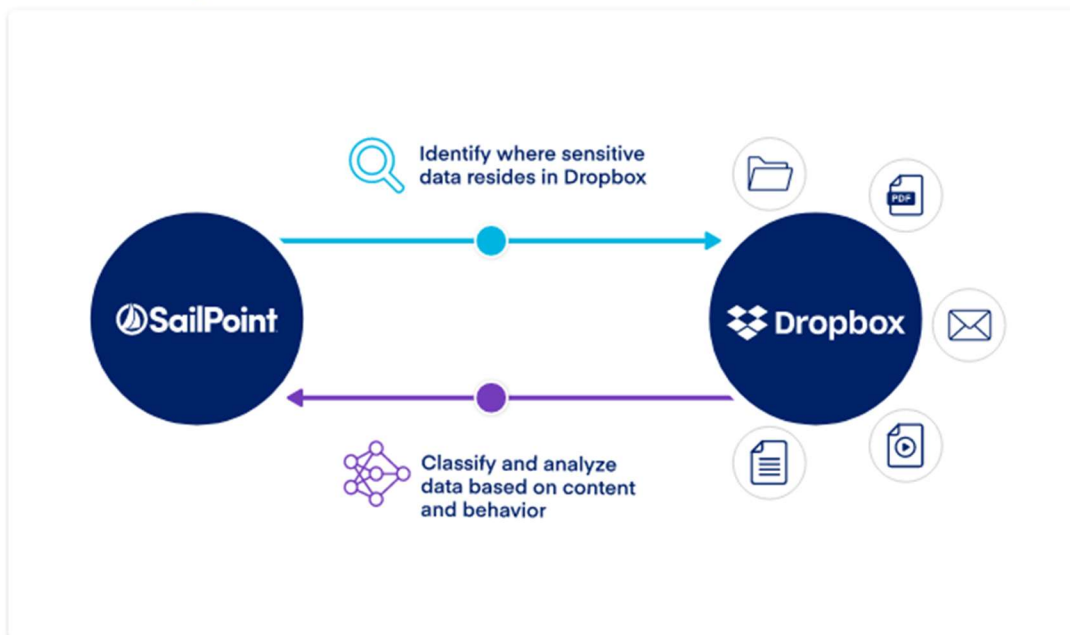
- Find and address your compliance gaps
- Ensure consistent governance across all data types
- Be alerted if inappropriate access is detected
- Minimize exposure to data leakage



Learn more

- How identity governance helps ensure GDPR compliance
- Governing unstructured data and data access
- Watch our Dropbox integration demo

Data discovery and classification



<https://www.sailpoint.com/integrations/dropbox/> (last visited December 16, 2021)

SailPoint is the leading provider of enterprise identity governance solutions. SailPoint's open identity platform enables organizations to manage the entire identity lifecycle of users accessing Dropbox Business, more efficiently control user access, securely prepare your data for migration to Dropbox Business, identify sensitive data, and monitor for malicious behavior.

- **Oversee and streamline who accesses Dropbox Business.** SailPoint identity governance solutions allow you to manage and govern users, groups, and entitlements for DropBox and automate the requesting and provisioning of user access. Improve security and audit performance by instantly reviewing and remediating access.
- **Monitor inappropriate access.** Centralize your visibility to users and their access across Dropbox and other cloud and on-premises applications. Admins can easily audit and ensure access is within corporate policy.
- **Gain greater data visibility.** Identify and classify data prior to migration to gain clear visibility across your data assets and optimize what data should be moved to Dropbox.
- **Clean up permissions.** Collect and analyze permissions to files for deeper insight into who has access and remediate inappropriate access issues to mitigate security risk to your content.
- **Establish data ownership.** Empower data owners who have the most intelligence about the data to take on a key role in managing access and ensure the right users have access to the right data.
- **Support hybrid environments.** Whether your data resides within your datacenter or in Dropbox Business, govern it with a centralized set of controls and policies.

<https://www.dropbox.com/app-integrations/sailpoint> (last visited December 16, 2021)

141. On information and belief, SailPoint, in order to develop, support, and provide its identity governance solutions, has directly infringed and continues to infringe, literally and/or

under the doctrine of equivalents, one or more claims of the '622 Patent under 35 U.S.C. § 271(a) by using Dropbox products and services in the United States and abroad without authority. Included in these acts of infringement are situations where devices associated with SailPoint interact with Dropbox servers to perform file sync operations.

DEFENDANT CLEAR CHANNEL

142. Clear Channel uses Dropbox's online document storage and synchronization products and services, including those products and services that infringe Plaintiff's patents identified above, through Dropbox's online platforms and mobile applications to customers, which practice each and every limitation of one or more claims of the '622 patent.


143. On its website, Dropbox identifies Clear Channel as a user of Dropbox Business.

**Businesses across the world of media trust
Dropbox**




<https://www.dropbox.com/business/solutions/media> (last visited December 16, 2021)



144. In 2017, Clear Channel along with Dropbox released a study disclosing Clear Channel's purchase of Dropbox Business licenses for its employees and summarizing the commercial benefits from the use of Dropbox Business. A copy of Clear Channel's study is attached as Exhibit 2.



Clear Channel Outdoor



In January 2017, Clear Channel purchased 1,200 enterprise Dropbox licenses to provide an enhanced security and collaboration experience for end users. Through three business cases, this study outlines the total return on investment achieved by Clear Channel as a result of its deployment. The study's findings are summarized below.

	\$677k total financial benefit		541% return on investment
---	--	---	-------------------------------------

<https://www.insightsforprofessionals.com/management/leadership/dropbox-businness-and-clear-channel-outdoor/download> (last visited December 16, 2021)

145. On information and belief, Clear Channel has directly infringed and continues to infringe, literally and/or under the doctrine of equivalents, one or more claims of the '622 Patent under 35 U.S.C. § 271(a) by using Dropbox products and services in the United States and abroad without authority. Included in these acts of infringement are situations where devices associated with Clear Channel interact with Dropbox servers to perform file sync operations.

PRAYER FOR RELIEF

WHEREFORE, Plaintiff respectfully requests the Court enter judgement against the Defendants:

- a. Declaring that the Defendants have infringed the '561, '942, '607, '787, '823, and '622 Patents;

- b. Awarding Plaintiff its damages suffered as a result of the Defendants' infringement of the '561, '942, '607, '787, '823, and '622 Patents, including pre-judgement and post-judgement interest and supplemental damages for any continuing post-verdict or post-judgement infringement with an accounting as needed;
- c. An order enjoining Defendants, their officers, agents, employees, attorneys, and all other persons or entities acting in concert, participation or in privity with one or more of them, and their successors and assigns, from infringing the '561, '942, '607, '787, '823, and '622 Patents;
- d. A judgment declaring that this is an exceptional case and awarding Plaintiff's reasonable attorneys' fees and costs in this action, as provided by 35 U.S.C. § 285;
- e. A judgment, declaration or order that Defendant's infringement is willful and increasing damages under 35 U.S.C. § 284;
- f. Awarding Plaintiff its costs, attorney's fees, expenses, and interest; and
- g. Granting Plaintiff such further relief which may be requested and as the Court find appropriate.

DEMAND FOR A JURY TRIAL

Plaintiff demands a trial by jury on all issues so triable in this Complaint.

Dated: December 29, 2021

OF COUNSEL:

Raja N. Saliba
Michael R. Dzwonczyk
Chidambaram S. Iyer
Mark Boland

SUGHRUE MION, PLLC
2000 Pennsylvania Ave., NW
Washington, DC 20037
(202) 293-7060
mboland@sughrue.com
rsaliba@sughrue.com
mdzwonczyk@sughrue.com
ciyer@sughrue.com

Respectfully submitted,

/s/ Raymond W. Mort, III
Raymond W. Mort, III
Texas State Bar No. 00791308
raymort@austinlaw.com

THE MORT LAW FIRM, PLLC
100 Congress Ave, Suite 2200
Austin, Texas 78701
Tel/Fax: (512) 865-7950

ATTORNEYS FOR PLAINTIFF
TOPIA TECHNOLOGY, INC.

CIVIL COVER SHEET

The S 44 civil cover sheet and the information contained herein neither replace nor supplement the filing and service of pleadings or other papers as required by law, except as provided by local rules of court. This form, approved by the Judicial Conference of the United States in September 1974, is required for the use of the Clerk of Court for the purpose of initiating the civil docket sheet. (SEE INSTRUCTIONS ON NEXT PAGE OF THIS FORM.)

I. PLAINTIFFS DEFENDANTS
Topia Technology, Inc.
Dropbox, Inc., Sailpoint Technologies Holdings, Inc., and Clear Channel Outdoor Holdings, Inc.
County of Residence of First Listed Plaintiff
County of Residence of First Listed Defendant
Attorneys (Firm Name, Address, and Telephone Number)
The Mort Law Firm, PLLC
100 Congress Ave, Suite 2000, Austin, TX 78701
(512) 865-7950

II. BASIS OF JURISDICTION (Place an "X" in One Box Only)
III. CITIZENSHIP OF PRINCIPAL PARTIES (Place an "X" in One Box for Plaintiff and One Box for Defendant)
1 U.S. Government Plaintiff
2 U.S. Government Defendant
3 Federal Question (U.S. Government Not a Party)
4 Diversity (Indicate Citizenship of Parties in Item III)
Citizen of This State
Citizen of Another State
Citizen or Subject of a Foreign Country
PTF DEF
Incorporated or Principal Place of Business In This State
Incorporated and Principal Place of Business In Another State
Foreign Nation

IV. NATURE OF SUIT (Place an "X" in One Box Only)
CONTRACT
REAL PROPERTY
TORTS
CIVIL RIGHTS
PRISONER PETITIONS
FORFEITURE PENALTY
LABOR
IMMIGRATION
BAN RUPTCY
PROPERTY RIGHTS
SOCIAL SECURITY
FEDERAL TAX SUITS
OTHER STATUTES

V. ORIGIN (Place an "X" in One Box Only)
1 Original Proceeding
2 Removed from State Court
3 Remanded from Appellate Court
4 Reinstated or Reopened
5 Transferred from Another District (specify)
6 Multidistrict Litigation - Transfer
8 Multidistrict Litigation - Direct File

VI. CAUSE OF ACTION
Cite the U.S. Civil Statute under which you are filing (Do not cite jurisdictional statutes unless diversity):
35 U.S.C. § 1 et seq.
Brief description of cause:
Patent Infringement

VII. REQUESTED IN COMPLAINT
CHEC IF THIS IS A CLASS ACTION UNDER RULE 23, F.R.Cv.P. DEMAND
CHEC YES only if demanded in complaint:
JURY DEMAND Yes No

VIII. RELATED CASE S IF ANY
(See instructions):
JUDGE DOC ET NUMBER

DATE December 29, 2021
SIGNATURE OF ATTORNEY OF RECORD /s/ Raymond W. Mort, III

FOR OFFICE USE ONLY
RECEIPT # AMOUNT APPLYING IFP JUDGE MAG. JUDGE

EXHIBIT 1



US009143561B2

(12) **United States Patent**
Manzano

(10) **Patent No.:** **US 9,143,561 B2**
(45) **Date of Patent:** ***Sep. 22, 2015**

(54) **ARCHITECTURE FOR MANAGEMENT OF DIGITAL FILES ACROSS DISTRIBUTED NETWORK**

(75) Inventor: **Michael R. Manzano**, Seattle, WA (US)

(73) Assignee: **TOPIA TECHNOLOGY, INC.**, Tacoma, WA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 860 days.

This patent is subject to a terminal disclaimer.

(21) Appl. No.: **12/267,852**

(22) Filed: **Nov. 10, 2008**

(65) **Prior Publication Data**
US 2009/0138528 A1 May 28, 2009

Related U.S. Application Data

(60) Provisional application No. 60/986,896, filed on Nov. 9, 2007.

(51) **Int. Cl.**
G06F 17/00 (2006.01)
G06F 17/30 (2006.01)
(Continued)

(52) **U.S. Cl.**
CPC **H04L 67/1095** (2013.01); **G06F 15/16** (2013.01); **G06F 17/30** (2013.01); **G06F 17/30174** (2013.01)

(58) **Field of Classification Search**
CPC G06F 17/00; G06F 17/30; G06F 15/16
USPC 707/608-609, 638, 657, 695, 806, 818, 707/821-822, 825; 709/203, 206; 715/229
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,600,834 A * 2/1997 Howard 713/178
5,806,078 A * 9/1998 Hug et al. 715/205

(Continued)

FOREIGN PATENT DOCUMENTS

EP 1130511 * 9/2001
WO WO 2007047302 * 4/2007

OTHER PUBLICATIONS

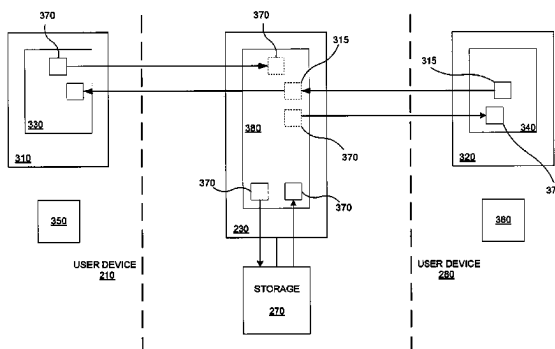
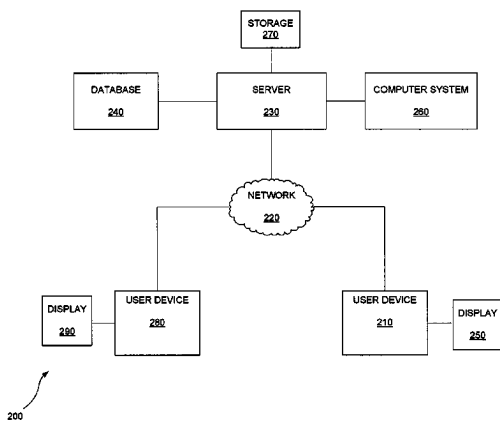
U.S. Office Action dated Aug. 13, 2014 for U.S. Appl. No. 11/739,083.

Primary Examiner — Srirama Channavajjala
(74) *Attorney, Agent, or Firm* — Pillsbury Winthrop Shaw Pittman LLP

(57) **ABSTRACT**

A system includes a first application executable on a first electronic device. The system further includes a second application executable on a second electronic device in communication with the first electronic device. The second electronic device is configured to store a first electronic file. Subsequent to a user modifying the first electronic file, the second application is operable to automatically transfer the modified first electronic file, or a copy thereof, to the first electronic device. The system further includes a third application executable on a third electronic device in communication with the first electronic device. The third electronic device is configured to store a second electronic file. Subsequent to the user modifying the second electronic file, the third application is operable to automatically transfer the modified second electronic file, or a copy thereof, to the first electronic device. The first application is operable to automatically transfer the modified first electronic file or copy to the third electronic device, and automatically transfer the modified second electronic file or copy to the second electronic device.

13 Claims, 3 Drawing Sheets



US 9,143,561 B2

Page 2

(51)	Int. Cl.								
	<i>G06F 15/16</i>	(2006.01)		7,325,038	B1	1/2008	Wang		
	<i>H04L 29/08</i>	(2006.01)		7,788,303	B2 *	8/2010	Mikesell et al.	707/828	
				2002/0026478	A1	2/2002	Rodgers et al.		
				2002/0087588	A1 *	7/2002	McBride et al.	707/204	
				2003/0028542	A1 *	2/2003	Muttik et al.	707/100	
				2004/0093361	A1 *	5/2004	Therrien et al.	707/204	
				2004/0107225	A1 *	6/2004	Rudoff	707/204	
				2004/0172424	A1 *	9/2004	Edelstein et al.	707/201	
				2005/0091316	A1	4/2005	Ponce et al.		
				2006/0010150	A1 *	1/2006	Shaath et al.	707/102	
				2007/0027936	A1 *	2/2007	Stakutis et al.	707/204	
				2007/0100913	A1 *	5/2007	Sumner et al.	707/204	
				2007/0180084	A1	8/2007	Mohanty		
				2008/0005114	A1 *	1/2008	Li	707/9	
				2013/0226871	A1 *	8/2013	Sarnowski	707/637	
									* cited by examiner
(56)	References Cited								
	U.S. PATENT DOCUMENTS								
	6,026,414	A *	2/2000	Anglin			1/1		
	6,154,817	A *	11/2000	Mohan et al.			711/162		
	6,260,069	B1 *	7/2001	Anglin			709/229		
	6,449,624	B1 *	9/2002	Hammack et al.			1/1		
	6,606,646	B2 *	8/2003	Feigenbaum			709/203		
	7,024,428	B1 *	4/2006	Huang et al.			1/1		
	7,136,934	B2 *	11/2006	Carter et al.			709/248		
	7,224,973	B2 *	5/2007	Tsutazawa et al.			455/437		
	7,260,646	B1	8/2007	Stefanik et al.					

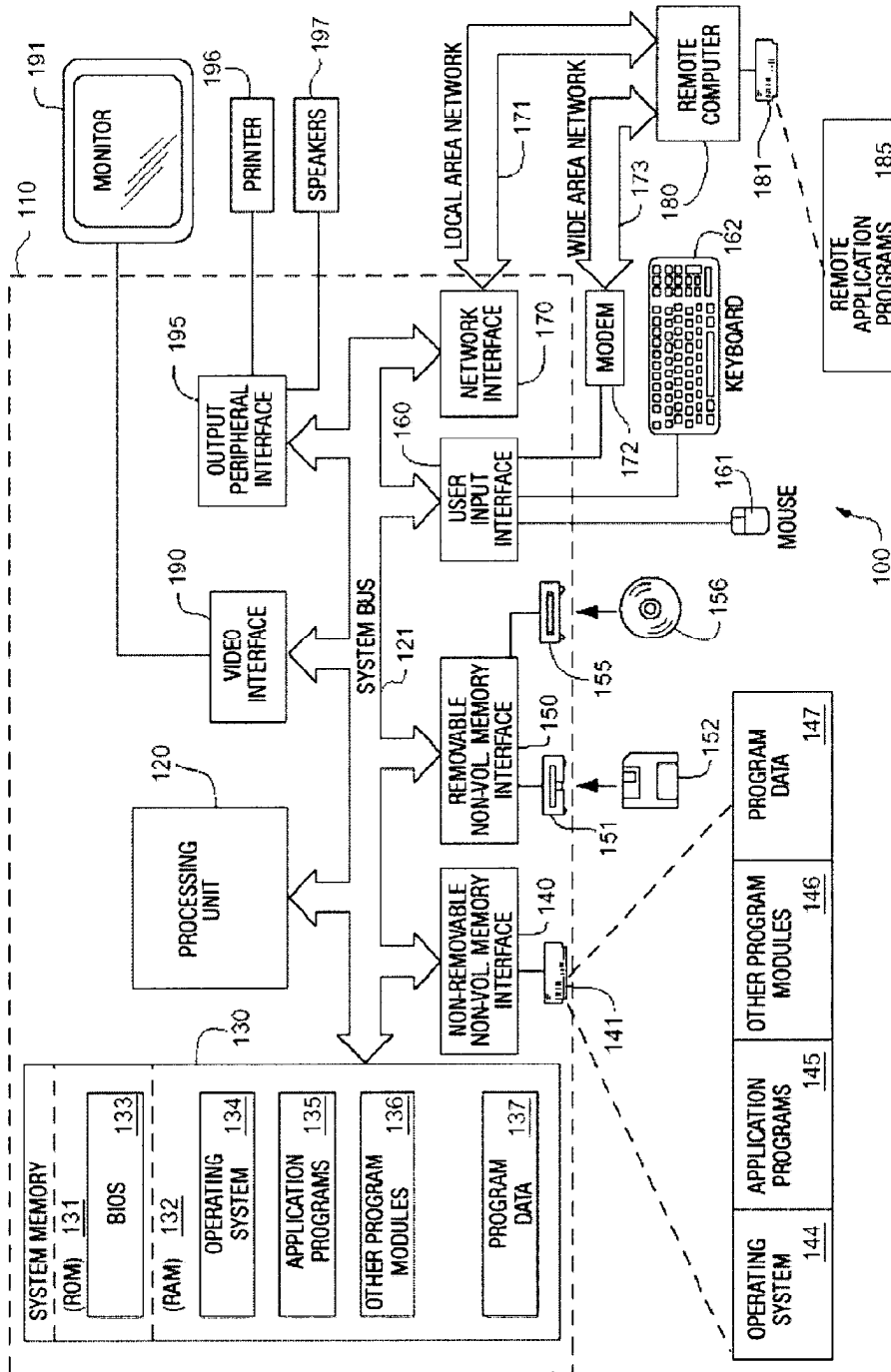


FIG. 1

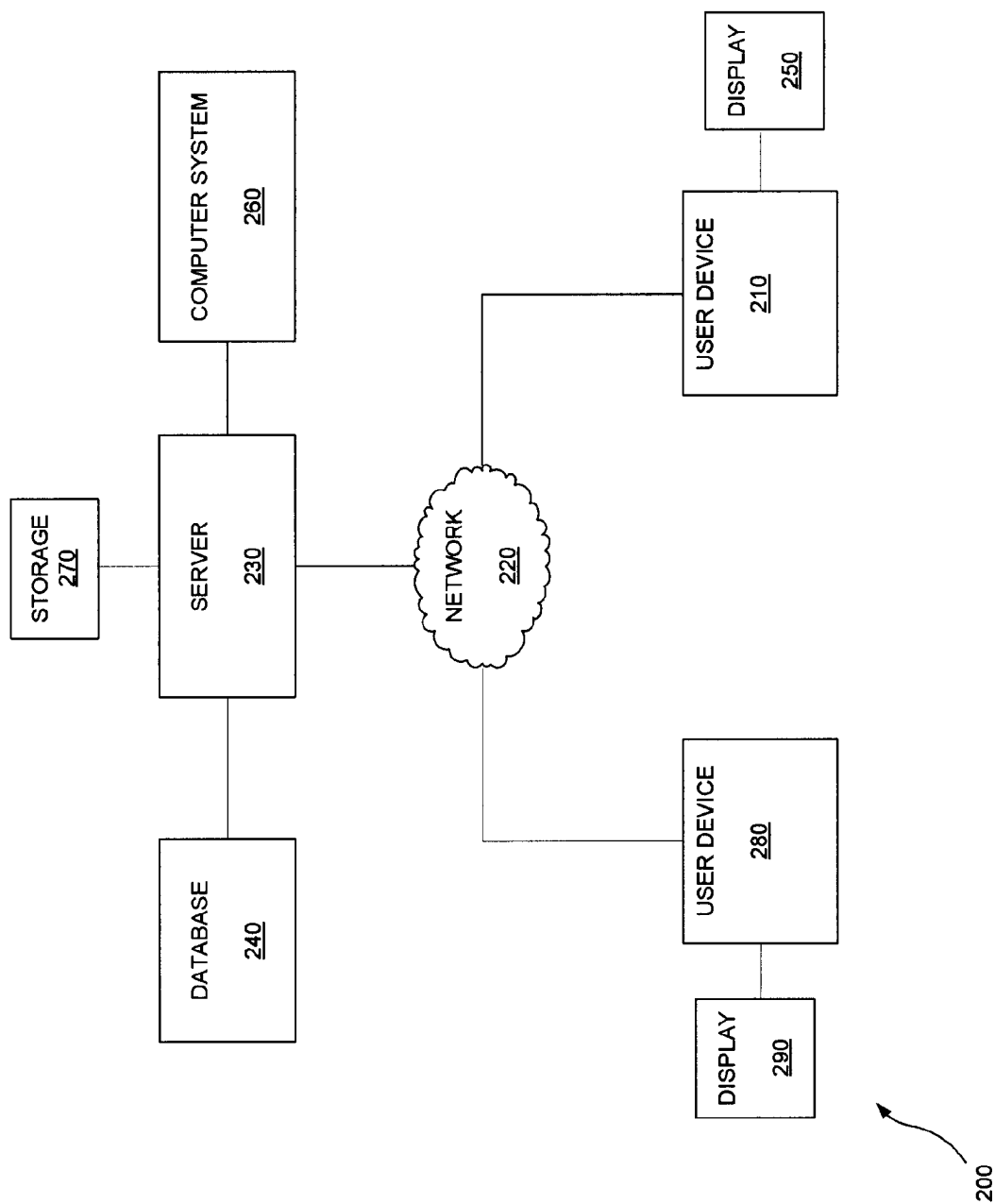


FIG. 2

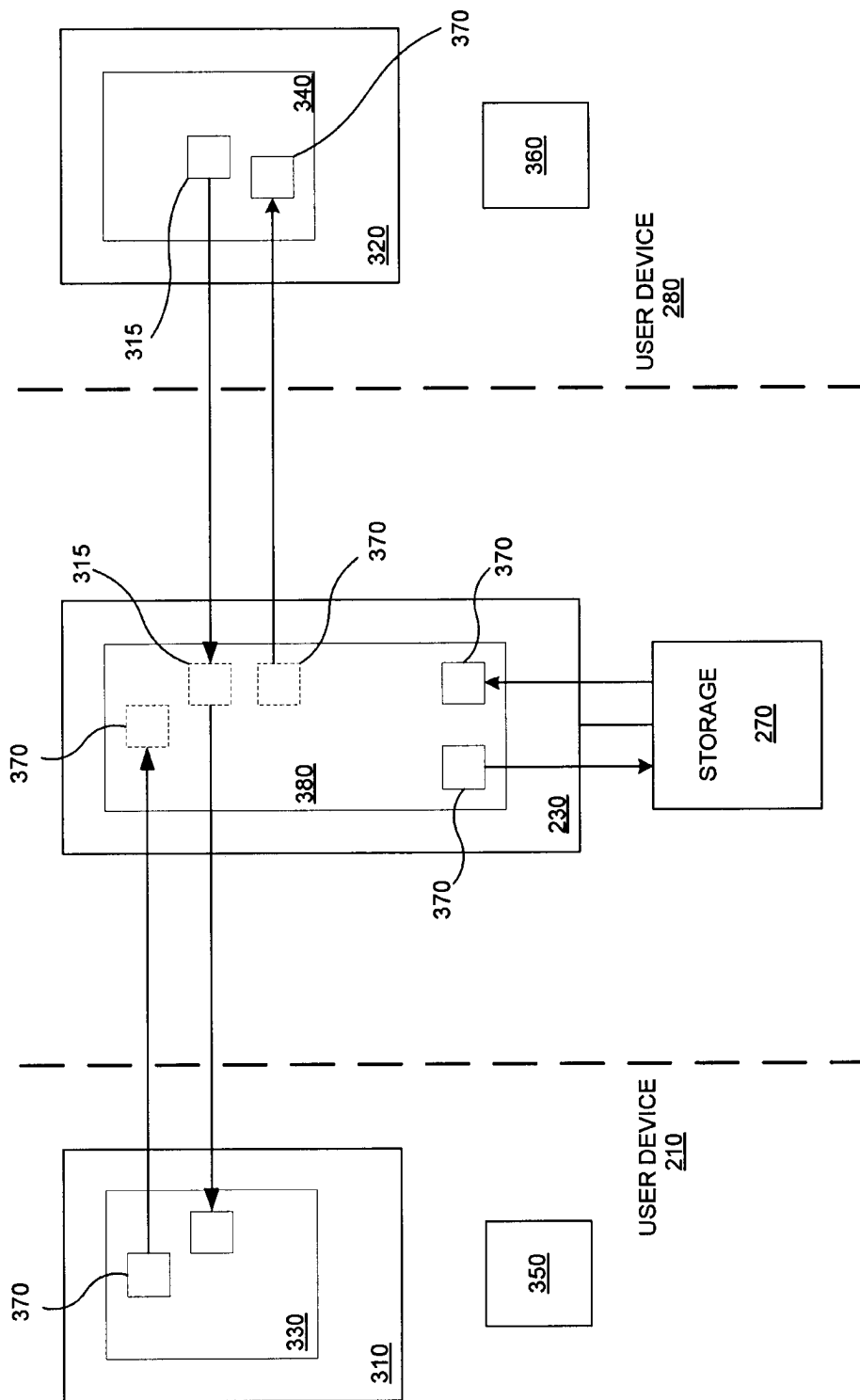


FIG. 3

US 9,143,561 B2

1

ARCHITECTURE FOR MANAGEMENT OF DIGITAL FILES ACROSS DISTRIBUTED NETWORK

CROSS-REFERENCE TO RELATED APPLICATIONS

This application claims priority to U.S. Provisional Appl. No. 60/986,896 entitled "ARCHITECTURE FOR MANAGEMENT OF DIGITAL FILES ACROSS DISTRIBUTED NETWORK" and filed Nov. 9, 2007, which is hereby incorporated by reference in its entirety.

FIELD OF THE INVENTION

This invention relates generally to computer-implemented processes and, more specifically, to sharing of electronic files among electronic devices.

BACKGROUND OF THE INVENTION

Users of modern computing systems are increasingly finding themselves in constantly-connected, high-speed networked environments. The Web continues to be a killer application, second only to email, on the Internet. Further, customers are increasingly using more than one computing device; a customer may have a desktop computer at home, one at work, and a constantly connected "smart phone". Due to the confluence of these two trends, file management across these devices has become a problem.

Although modern devices are easily connected, they do not provide the customer a seamless environment; the customer must manually handle many aspects of that connection. With regards to file management, customers must manually move files between their devices using some protocol like email, ftp, or by posting them on the Web. These practices lead to problems that include:

The proliferation of redundant file copies. This proliferation creates a confusing environment where the customer is unclear where the "official" or newest version of a file exists.

The creation of an error-prone environment. Some documents, such as those associated with word processing and desktop publishing, externally reference other files. Copying such a document can break these references causing errors that the customer has to handle manually. An example of such a document is a desktop publishing document that contains a reference to an image. If that image file is not transferred along with the desktop publishing file, the image will appear as a broken link.

Unnecessary complexity. Because devices tend to have their own filing system, customers must manage a different filing model on each of his devices. For example, instead of having a single "Movies" folder, he may have to deal with many "Movies" folders, which may be in different locations on each of his devices. Each device may also have its own security model, further complicating the matter.

That a customer has to manually move files around to ensure their accessibility on his devices is unnecessary, and is an indicator of a lack of customer-focused design in modern file systems. File systems in use today are direct offspring of systems used when graphical customer interfaces were nonexistent. Modern file system customer interfaces, such as Windows® Explorer and Mac OS X's Finder are just now starting to provide experiences that are more in line to a customer's workflow. Whereas, before, these interfaces were

2

concerned with representing files with abstracted icons, the file's actual contents are becoming paramount in how files are organized and presented.

Problems still exist with how these newer customer interfaces are implemented. They are not completely integrated with applications, suffer from performance problems, and do not generally work well outside of a device's local file system.

There are several solutions to this problem that are in one way or another inadequate to the task:

Remote Desktop software allows a customer to remotely "see" his desktop. Remote desktop software screen-scrapes a remote machine's screen (a "server") and displays it on a screen local to the customer (a "client"). Remote desktop gives a customer access to not only his files, but also to his applications. However, this approach requires that the host machine be turned on and connected to the internet at all times. Consequently, this approach would not be appropriate for mobile hosts such as laptops. Remote desktop does not use the resources of a local machine. For full accessibility, the customer would have to keep all files and application on the host machine as any files stored on a client are not guaranteed to be accessible.

Distributed File Systems, like remote desktop software, place data on an always-connected host machine. Unlike remote desktop software, the host machine is not one on which the customer performs computing tasks. The host machine is used as a storage mechanism, and any computation performed on that machine serves to support its use as such. Distributed file systems generally provide the right functionality for customers to share files between their devices. However, distributed file systems are usually deployed as a shared resource; that is, other customers have access to it. Because of this sharing, a customer's files may be buried deep in a filing structure, and it may not always be immediately evident to customers what kind of access they have to a particular file. Further, to use a distributed file system, the customer must always be connected to it. Files stored on a distributed file system are generally inaccessible if the customer's machine is not connected to it, unless the customer has copied or moved the files to his machine's local hard drive. However, doing so immediately creates the problem of having two filing systems for the same file, creating a mental burden on the customer.

Additionally, accessing a file located on a distributed file system tends to be slower than accessing files on the local hard drive. Modern applications are usually written to assume that the files they access are located locally, and thus are not optimized to access remote files. When these applications are used with remote files, they can lose performance by an order of magnitude. This problem can be fixed by automatically caching often-used files on the local file system, and only synchronizing them when they have been changed. However, this separate synchronization step introduces another problem: because the synchronization process can be lengthy, the customer is never entirely sure if the file he is remotely accessing is the latest version of the file, versus an earlier one that has been marked to be updated. Further, the directory may not reflect the existence of the file at all until synchronization finishes.

FTP is similar to a distributed file system with regards to files being hosted on a remote server. However FTP generally does manifest as a "disk drive" on the customer's desktop; the customer must use special FTP client software to access an FTP server. It shares the same problem as distributed file systems, with the additional problem of weak integration with applications. Applications can generally write and read files directly to and from a distributed file system. This is not the

US 9,143,561 B2

3

case with FTP, as the customer has to manually use the client software to perform these operations as a separate task.

Email was originally invented for messaging. From the beginning, the model it employs to make files accessible remotely is necessarily inefficient. Email's model for making files accessible is in the form of an email "attachment". Attachments are so named because they piggy-back on a message sent from one customer to another. A customer can make a file remotely available using email by attaching the file to an email and sending it to himself. He can then retrieve the file from a remote location by accessing the message on the email server. Email used in this way is even worse than FTP as the process is even more manual: a customer must find the message containing the file before he can even access it. Further, the location in which the attachment lives is read only. If the customer, for example, were to open the file, change it, then save it back out, the results would be ambiguous to the user because the email application, not the user, specified its location. Usually, the saved file would end up buried in an email file cache in an undisclosed area of the file system.

Flash Drives and External Disk Drives, although seemingly the most "primitive" way to ensure file availability, avoid all the problems related to network latency. However, these devices must be physically connected to the computer on which the files will be accessed. These restrictions preclude the customer from employing several effective workflows including: using more than one computer to complete a single task (the files can only be accessed on one computer) and setting up an automated backup (the computer running the backup can't guarantee that the storage device will be connected come backup time). Further, to ensure full availability of the files, the customer must carry the device with them at all times, and must follow the associated protocols for mounting and dismounting the device.

Other problems with the prior art not described above can also be overcome using the teachings of embodiments of the present invention, as would be readily apparent to one of ordinary skill in the art after reading this disclosure.

SUMMARY OF THE INVENTION

In an embodiment, a system includes a first application executable on a first electronic device. The system further includes a second application executable on a second electronic device in communication with the first electronic device. The second electronic device is configured to store a first electronic file. Subsequent to a user modifying the first electronic file, the second application is operable to automatically transfer the modified first electronic file, or a copy thereof, to the first electronic device. The system further includes a third application executable on a third electronic device in communication with the first electronic device. The third electronic device is configured to store a second electronic file. Subsequent to the user modifying the second electronic file, the third application is operable to automatically transfer the modified second electronic file, or a copy thereof, to the first electronic device. The first application is operable to automatically transfer the modified first electronic file or copy to the third electronic device, and automatically transfer the modified second electronic file or copy to the second electronic device.

BRIEF DESCRIPTION OF THE DRAWING

Preferred and alternative embodiments of the present invention are described in detail below with reference to the following drawings.

4

FIG. 1 is a schematic view of an exemplary operating environment in which an embodiment of the invention can be implemented;

FIG. 2 is a functional block diagram of an exemplary operating environment in which an embodiment of the invention can be implemented; and

FIG. 3 is a functional block diagram illustrating file sharing and/or synchronization according to an embodiment of the invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

An embodiment of the invention leverages remote programming concepts by utilizing processes called mobile agents (sometimes referred to as mobile objects or agent objects). Generally speaking, these concepts provide the ability for an object (the mobile agent object) existing on a first ("host") computer system to transplant itself to a second ("remote host") computer system while preserving its current execution state. The operation of a mobile agent object is described briefly below.

The instructions of the mobile agent object, its preserved execution state, and other objects owned by the mobile agent object are packaged, or "encoded," to generate a string of data that is configured so that the string of data can be transported by all standard means of communication over a computer network. Once transported to the remote host, the string of data is decoded to generate a computer process, still called the mobile agent object, within the remote host system. The decoded mobile agent object includes those objects encoded as described above and remains in its preserved execution state. The remote host computer system resumes execution of the mobile agent object which is now operating in the remote host environment.

While now operating in the new environment, the instructions of the mobile agent object are executed by the remote host to perform operations of any complexity, including defining, creating, and manipulating data objects and interacting with other remote host computer objects.

File transfer and/or synchronization, according to an embodiment, may be accomplished using some or all of the concepts described in commonly owned U.S. patent application Ser. No. 11/739,083, entitled "Electronic File Sharing," the entirety of which is incorporated by reference as if fully set forth herein.

FIG. 1 illustrates an example of a suitable computing system environment **100** in which one or more embodiments of the invention may be implemented. The computing system environment **100** is only one example of a suitable computing environment and is not intended to suggest any limitation as to the scope of use or functionality of the invention. Neither should the computing environment **100** be interpreted as having any dependency or requirement relating to any one or combination of components illustrated in the exemplary operating environment **100**.

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

US 9,143,561 B2

5

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

With reference to FIG. 1, an exemplary system for implementing the invention includes a general purpose computing device in the form of a computer 110. Components of computer 110 may include, but are not limited to, a processing unit 120, a system memory 130, and a system bus 121 that couples various system components including the system memory to the processing unit 120. The system bus 121 may be any of several types of bus structures including a memory bus or memory controller, a peripheral bus, and a local bus using any of a variety of bus architectures. By way of example, and not limitation, such architectures include Industry Standard Architecture (ISA) bus, Micro Channel Architecture (MCA) bus, Enhanced ISA (EISA) bus, Video Electronics Standards Association (VESA) local bus, and Peripheral Component Interconnect (PCI) bus also known as Mezzanine bus.

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

The system memory 130 includes computer storage media in the form of volatile and/or nonvolatile memory such as read only memory (ROM) 131 and random access memory (RAM) 132. A basic input/output system 133 (BIOS), containing the basic routines that help to transfer information between elements within computer 110, such as during start-up, is typically stored in ROM 131. RAM 132 typically contains data and/or program modules that are immediately

6

accessible to and/or presently being operated on by processing unit 120. By way of example, and not limitation, FIG. 1 illustrates operating system 134, application programs 135, other program modules 136, and program data 137.

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

The drives and their associated computer storage media discussed above and illustrated in FIG. 1, provide storage of computer readable instructions, data structures, program modules and other data for the computer 110. In FIG. 1, for example, hard disk drive 141 is illustrated as storing operating system 144, application programs 145, other program modules 146, and program data 147. Note that these components can either be the same as or different from operating system 134, application programs 135, other program modules 136, and program data 137. Operating system 144, application programs 145, other program modules 146, and program data 147 are given different numbers here to illustrate that, at a minimum, they are different copies. A user may enter commands and information into the computer 110 through input devices such as a keyboard 162 and pointing device 161, commonly referred to as a mouse, trackball or touch pad. Other input devices (not shown) may include a microphone, joystick, game pad, satellite dish, scanner, or the like. These and other input devices are often connected to the processing unit 120 through a user input interface 160 that is coupled to the system bus, but may be connected by other interface and bus structures, such as a parallel port, game port or a universal serial bus (USB). A monitor 191 or other type of display device is also connected to the system bus 121 via an interface, such as a video interface 190. In addition to the monitor, computers may also include other peripheral output devices such as speakers 197 and printer 196, which may be connected through an output peripheral interface 190.

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

When used in a LAN networking environment, the computer 110 is connected to the LAN 171 through a network interface or adapter 170. When used in a WAN networking

7

environment, the computer **110** typically includes a modem **172** or other means for establishing communications over the WAN **173**, such as the Internet. The modem **172**, which may be internal or external, may be connected to the system bus **121** via the user input interface **160**, or other appropriate mechanism. In a networked environment, program modules depicted relative to the computer **110**, or portions thereof, may be stored in the remote memory storage device. By way of example, and not limitation, FIG. **1** illustrates remote application programs **185** as residing on memory device **181**. It will be appreciated that the network connections shown are exemplary and other means of establishing a communications link between the computers may be used.

Referring now to FIG. **2**, an embodiment of the present invention can be described in the context of an exemplary computer network system **200** as illustrated. System **200** includes electronic user devices **210**, **280**, such as personal computers or workstations, that are linked via a communication medium, such as a network **220** (e.g., the Internet), to an electronic device or system, such as a server **230**. The server **230** may further be coupled, or otherwise have access, to a database **240**, electronic storage **270** and a computer system **260**. Although the embodiment illustrated in FIG. **2** includes one server **230** coupled to two user devices **210**, **280** via the network **220**, it should be recognized that embodiments of the invention may be implemented using two or more such user devices coupled to one or more such servers.

In an embodiment, each of the user devices **210**, **280** and server **230** may include all or fewer than all of the features associated with the computer **110** illustrated in and discussed with reference to FIG. **1**. User devices **210**, **280** include or are otherwise coupled to a computer screen or display **250**, **290**, respectively. User devices **210**, **280** can be used for various purposes including both network- and local-computing processes.

The user devices **210**, **280** are linked via the network **220** to server **230** so that computer programs, such as, for example, a browser or other applications, running on one or more of the user devices **210**, **280** can cooperate in two-way communication with server **230** and one or more applications running on server **230**. Server **230** may be coupled to database **240** and/or electronic storage **270** to retrieve information therefrom and to store information thereto. Additionally, the server **230** may be coupled to the computer system **260** in a manner allowing the server to delegate certain processing functions to the computer system.

Referring now to FIG. **3**, illustrated is functionality of an embodiment of the invention allowing a user (not shown) who owns or otherwise controls devices **210**, **280** to automatically maintain file synchronization between at least devices **210**, **280**, or any other user devices on which principles of the present invention are implemented. In an embodiment, an administrator (not shown) of the server **230** or other appropriate electronic device transfers a file-transfer and/or synchronization application to the user devices **210**, **280** for installation thereon. Once installed on the user devices **210**, **280**, the file-transfer application provides file-transfer clients **310**, **320** executable by the user devices **210**, **280**, respectively. Each of the file-transfer clients **310**, **320** may, but need not, include a respective mobile-agent runtime environment **330**, **340**. The mobile-agent runtime environment **330**, **340** include portions of memory of the user devices **210**, **280** dedicated to allowing a mobile object the ability to perform operations that the mobile object is programmed to carry out. Also included in the file-transfer application are user interfaces **350**, **360** that are displayable on the displays **250**, **290**, respectively. In an embodiment, the interfaces **350**, **360** allow

8

a user to view, access and/or organize files to be synched among the various user devices.

Generally, all files that the user desires to be synched or shared may at some point be uploaded by one or more of the user devices **210**, **280** and stored in storage **270**. Upon receiving the files to be synched, the server **230** can store such files in the storage **270** and/or transfer the files to one or more of the respective hard drives of the user devices **210**, **280**, thereby enabling each respective user device to access such files. In this manner, the server **230** is operable to treat each hard drive of the respective user devices **210**, **280** as a local document cache for files received by the server. Typically, the server **230** will store one or more of the received files to the storage **270** only if the destination user device is offline or otherwise temporarily not in communication with the server **230**. Upon resuming communication with the destination user device, the server **230** will transfer the temporarily stored files to the destination device.

In operation, according to an embodiment, the user may open and modify a file **370**, such as a word-processing document or other electronic file. Alternatively, the user may create a first instance of the file **370**. The user may have previously have associated, or may now associate, the file **370** with the transfer client **310**. Upon a predetermined and user-configurable triggering event, the transfer client **310** transfers the modified file **370**, or a copy of the modified file, to the server **230**. Such a triggering event may include, but be not limited to, the user saving the file, the elapsing of a predetermined amount of time during which the file has been opened, or the re-initiation of a communication session between the device **210** and the server **230**.

The file **370** is transferred to the server **230** on which is executing a synchronization application **380**, which may include a mobile-agent runtime environment. Through user configuration, the synch application **380** monitors a set of user devices to which the file **370** should be transferred to effect file synchronization. In the illustrated embodiment, this set of user devices includes the user device **280**. The synch application **380** polls the device **280** to determine whether the device **280** is in communication with the server **230**. If the device **280** is in communication with the server **230**, the synch application **380** transfers the file **370** to the device **280**, whereupon the transfer client **320** resident on the device **280** replaces the previous version of the file **370**, previously cached on the device **280**, with the latest version of the file **370** modified on the user device **210**. If the device **280** is not currently in communication with the server **230**, the synch application **380** may store the file **370** in the storage **270** until such time as communication between the device **280** and server **230** is reestablished. As illustrated in FIG. **3**, a similar reverse-direction synchronization process may be performed by the synch application **380** and the transfer clients **310**, **320** with regard to a file **315** modified on device **280** and synchronized to device **210**.

In an embodiment, the user interfaces **350**, **360** may include a list of the customer's documents and related metadata, as well as any one-to-one or one-to-many relationships between the documents and metadata. An embodiment can always provide customers with an accurate "picture" of their document collection, regardless of whether their devices physically contain the documents. As alluded to earlier, a problem with distributed file systems and FTP is the latency between a file being put onto a file system and it showing up on a remote machine. To prevent this problem, an embodiment directory is decoupled from the movement of files. An embodiment's directory update system updates at a higher priority than the documents to be synchronized. This feature

ensures that when a customer browses or searches through his set of documents, they appear even if they have not yet been cached locally on the user device. An indicator signifying a document's availability may be prominently displayed adjacent to the document's representation so that customers are aware of the document's availability.

An embodiment may include a stand-alone application that allows customers to find and manage documents associated with transfer clients **310**, **320** by visualizing relationships between documents and their metadata. It allows customers to tag documents with any number of identifiers. Customers can relate both documents and tags with each other in any number of user-specified one-to-one and one-to-many relationships, and an embodiment provides a user interface to browse and search on these relationships. To mitigate the customers' learning curve, an embodiment can implement relationships common to contemporary file systems, including a folder hierarchy. In addition to this, an embodiment provides direct support for methods that the customer uses to organize documents by manifesting them as user interface idioms. This is unlike conventional document filing systems which require the customer to work within a strict folder metaphor for organization.

Some alternate methods that an embodiment supports for organizing documents include:

Allow customers to organize their documents by application. Many times customers remember the application used to create a document instead of the document's name or its location in a hierarchy.

Allow customers to organize their documents by most recent access. Customers are likely to access a document they've accessed in the near past. Usually, such documents are part of a task that the customer is actively working.

Allow customers to organize their documents by project or subproject.

Allow customers to organize their documents by people. Many times, especially in the context of a collaboration, a document is directly related to one or more people other than the customer.

Allow the customer to organize their document by process stage. Documents may represent one or more stages of a process. Customers need a method for organizing documents by process stage, and a mechanism for moving the document through a set of predefined stages.

Allow customers to organize their documents by any of the aforementioned methods concurrently. These organization methods are not mutually exclusive.

An embodiment presents an interface that allows a customer to locate one or more documents associated with the transfer clients **310**, **320** and open such document into a separate software application. Since this interface is intended to be used from within the separate application, that application may need to know how to invoke such interface. Advantageously, this invocation behavior can be provided to the application using the application's plug-in API.

An embodiment presents an interface that allows a customer to synchronize a currently opened document according to processes described elsewhere herein. This interface can be invoked within an application and can be made available to the application in the manner described above in connection with the application's plug-in API.

Some files associated with the transfer clients **310**, **320** are dependent on other files associated with the transfer clients **310**, **320**. For example, a desktop publishing document may include images that are stored in files separate from the main document. Previous file-synching solutions treat these files as

separate. Because of this, for example, a document synchronized from the device **210** to the device **280** may be opened by the user of the device **280** before the image files have been fully transferred to the device **280**. This causes the document to fail to open, or break, since the image files don't exist or are incomplete. An embodiment prevents this by: (1) always ensuring the file catalog (e.g., the stand-alone application that allows customers to find and manage documents associated with transfer clients **310**, **320**, as discussed above herein) is synchronized before any file data is synchronized, and (2) pausing any file access by any program until the file contents have been fully synchronized. In such an embodiment, if a user attempts, using a software program, to open a file whose related files haven't yet finished transferring to the local (hard drive) cache, if that software attempts to open the related files, the software program is blocked by an embodiment until the requested files are downloaded and ready to access.

Other file sending and synchronizing software requires the user to upload their data to a storage device owned by the operator of the service. An embodiment treats storage as a participant in the synchronization process; this means that the user can choose the service or device where their files will be stored. The file transfer/synching is abstracted from the storage system allowing any storage to be used. An embodiment treats storage like any other synch target, such as a desktop computer, or a cell phone. As such, any device owned or otherwise controlled by the user and running a synch application, such as synch application **380**, as provided in an embodiment of the invention can perform the storage and/or synching functions described elsewhere herein. That is, the user device **280** or user device **210**, rather than the server **230**, may perform such functions.

While a preferred embodiment of the invention has been illustrated and described, as noted above, many changes can be made without departing from the spirit and scope of the invention. For example, as an alternative to the approach described with reference to FIG. 3, wherein the transfer clients **310**, **320** function to "push" modified or created files to the synch application **380**, the synch application **380** may instead function to periodically "pull" or otherwise actively retrieve such files from the transfer clients **310**, **320**. Instead, the invention should be determined entirely by reference to the claims that follow.

The embodiments of the invention in which an exclusive property or privilege is claimed are defined as follows:

1. A system, comprising:

a first electronic device configured to selectively execute a first application, the first electronic device being in communication with a second electronic device and a third electronic device, each associated with a user wherein the first electronic device is configured to:

receive from a second application executable on the second electronic device a copy of a first electronic file automatically transferred from the second application when the user modifies a content of the first electronic file; and wherein the first electronic device is further configured to receive from a third application executable on the third electronic device a copy of a second electronic file automatically transferred from the third application when the user modifies a content of the second electronic file; and wherein the first application is further configured to automatically transfer the modified first electronic file copy to the third electronic device to replace an older version of the first electronic file stored on the third electronic device with the modified first electronic file copy having the content modified by the user; and

US 9,143,561 B2

11

automatically transfer the modified second electronic file copy to the second electronic device to replace an older version of the second electronic file stored on the second electronic device with the modified second electronic file copy having the content modified by the user;

wherein the second application automatically transfers the copy of the modified first electronic file to the first electronic device upon determining that a save operation has been performed on the modified first electronic file.

2. The system of claim 1 wherein at least one of the second and third applications comprises a runtime environment for mobile-agent objects.

3. The system of claim 1 wherein the first application is configured to store the modified first electronic file copy to a memory device associated with the first electronic device.

4. The system of claim 3, wherein the first application is configured to store the modified first electronic file copy to the memory device associated with the first electronic device when the third electronic device is not in communication with the first electronic device.

5. The system of claim 1 wherein the second application is operable to create a first mobile object, the first mobile object being operable to create a proxy object at the first electronic device.

6. The system of claim 5 wherein the first mobile object is operable to provide the copy of the modified first electronic file to the proxy object.

7. The system of claim 6 wherein the proxy object is operable to store the copy of the modified first electronic file in a storage device coupled to the first electronic device.

8. A method implementable in an electronic system having a storage component, the electronic system being coupled to a first electronic device having stored thereon a first electronic file and a second electronic device having stored thereon a second electronic file, the first electronic device and the second electronic device each being associated with a user, the method comprising:

receiving from the first electronic device a copy of a first electronic file modified by the user, the copy of the modified first electronic file being automatically pro-

12

vided to the electronic system by the first electronic device when a content of the first electronic file is modified by the user;

receiving from the second electronic device a copy of a second electronic file modified by the user, the copy of the modified second electronic file being automatically provided to the electronic system by the second electronic device when a content of the second electronic file is modified by the user;

automatically transferring the modified first electronic file copy to the second electronic device to replace an older version of the first electronic file stored on the second electronic device with the modified first electronic file copy having the content modified by the user; and

automatically transferring the modified second electronic file copy to the first electronic device to replace an older version of the second electronic file stored on the first electronic device with the modified second electronic file copy having the content modified by the user;

wherein the first electronic device automatically transfers the copy of the modified first electronic file to the electronic system upon determining that a save operation has been performed on the modified first electronic file.

9. The method of claim 8 wherein at least one of the first and second electronic devices includes a runtime environment for mobile-agent objects.

10. The method of claim 8, wherein the electronic system is configured to store the modified first electronic file copy to the storage component.

11. The method of claim 10 wherein the electronic system is configured to store the modified first electronic file copy to the storage component when the second electronic device is not in communication with the electronic system.

12. The method of claim 8 wherein the first electronic device is operable to create a first mobile object, the first mobile object being operable to create a proxy object at the electronic system.

13. The method of claim 12 wherein the first mobile object is operable to provide the copy of the modified first electronic file to the proxy object.

* * * * *

EXHIBIT 2



Clear Channel Outdoor



In January 2017, Clear Channel purchased 1,200 enterprise Dropbox licenses to provide an enhanced security and collaboration experience for end users. Through three business cases, this study outlines the total return on investment achieved by Clear Channel as a result of its deployment. The study's findings are summarized below.



\$677k
total financial benefit



541%
return on investment



1,400
days of working time saved per year

Dropbox is giving Clear Channel's end users time to focus on more important work. With each end user saving 9 hours per year on collaboration tasks and with 800 total hours of downtime avoided, Dropbox is driving **\$428k of total value** in productivity gains per year



63
servers removed from use

Moving data to Dropbox is allowing Clear Channel to deprecate 63 servers across over 30 markets. This is driving enhanced security and data recoverability, while producing **\$249k in annual cost savings**



66 TB
of data secured in Dropbox

By moving to Dropbox, Clear Channel is protecting critical assets, including **49 TB** of data previously saved to on-prem servers and **17 TB** of data previously saved to personal Dropbox accounts

“At Clear Channel, we strive to create a world-class work environment with the top technologies available. We've achieved a return on investment of over 500% with Dropbox, while taking a giant step to modernize our IT strategy and enhance collaboration across our company.”

Nichole Boatsman
IT Director

For more information on Dropbox Business, contact sales@dropbox.com or visit dropbox.com/business.



Business case #1

Team folder deployment



10
minutes saved per
end user per week



1,300
total days of
working time
saved per year



\$398k
total financial
benefit

Team Folder allows users to be organized in groups and share files internally or externally at any level of folder structure. Clear Channel estimates that through its ability to enable remote collaboration, maintain reliable connectivity and make file recovery and permissioning self-serve, Team Folder saves each of its users 10 minutes a week.

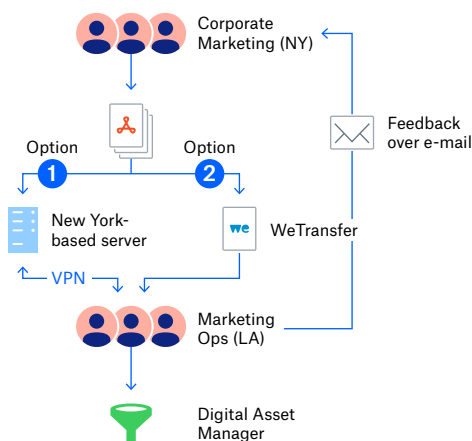
Below is an example of the kinds of process improvements driving this benefit.

“Collaborating with Team Folder makes colleagues located across the country feel like they’re a couple desks away. I’m not only saving time, but also working with these colleagues more frequently.”

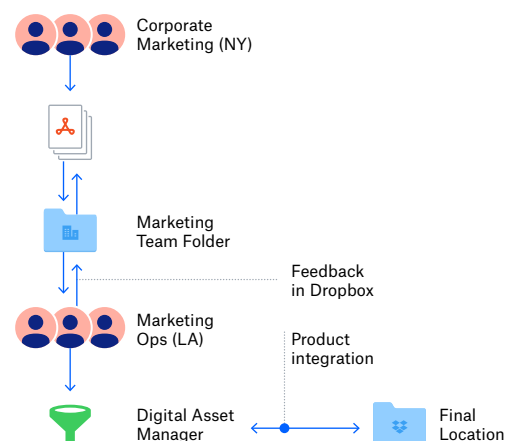
Jennifer Hurley

Director of Marketing Operations

Process before Dropbox



Process with Dropbox



- 2x per week, the LA-based marketing ops team collects assets from the NY-based corporate marketing team and adds them to the DAM
- Connecting to a NY-based file server to download the assets is challenging due to a faulty VPN
- When that fails, the corporate marketing team uses WeTransfer to share the files
- Either option requires logging in, downloading the asset and uploading it to the DAM
- Feedback happens over e-mail, creating additional delays and confusion to share the files

- All assets are created in a Marketing team folder, which both teams have access to
- As work is completed, the team folder automatically stays updated, so assets are accessible to the marketing ops team at any time
- Feedback is given using Dropbox comments, keeping work and communication in one place and limiting the use of e-mail
- A process that once took 5 minutes now takes a few seconds to complete, saving each marketing ops user **10 minutes per week**



Business case #2

Data migration to cloud

- 

63
total file servers removed from use
- 

100
total days of working time saved per year
- 

\$279k
total financial benefit

Results

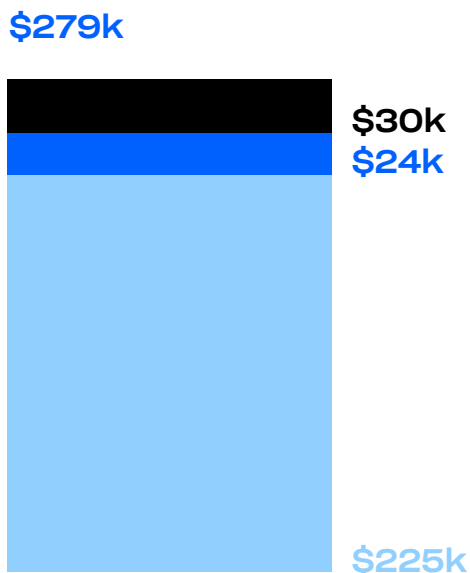
- 

30k
value of time saved
- 

24k
total SharePoint cost savings
- 

225k
server cost savings per year

Financial Impact



Business Impact

Before Dropbox, 7% of Clear Channel employees experienced a server outage and 1% of employees experienced a computer crash once per year, resulting in **808 hours** of lost work time. Dropbox allows work to be done on any device, producing **\$30k per year** of value in additional work time

Clear Channel previously hosted **43 GB** of data on SharePoint and paid **\$24k** annually in hosting fees

Before deploying Dropbox, Clear Channel maintained **63 on-prem servers**. With Dropbox, these servers will be deprecated, saving **\$225k per year** in server replacement and space costs

“I am thrilled with the amount of money we’re saving year-over-year by deprecating 60+ file servers and deploying Dropbox. But what is even better is that we’re no longer wasting resources maintaining hardware across 30 locations. That’s now in expert hands at Dropbox, allowing us to focus on our jobs.”

Nichole Boatsman
IT Director



Business case #3

Security improvements

-  **550** personal Dropbox accounts secured
-  **75** devices remote wiped by IT
-  **66 TB** of data secured with Dropbox

“With over 17 TB of data in personal accounts, our employees made it clear that Dropbox is an important space for getting work done at Clear Channel. And as CTO, it’s my responsibility to ensure this work gets done securely.”

Christian Aaselund
CTO



49 TB on-prem server data

Before deploying Dropbox Enterprise, Clear Channel had **49 TB** of data saved to on-prem servers across **34 locations**.

Migrating all this data to Dropbox will improve security by:

- Increasing corporate visibility and control of user data
- Improving IT’s ability to recover lost user data
- Enabling remote data wipe of employee devices

17 TB end user data in personal accounts

Before deploying Dropbox Enterprise, **550 employees** created personal Dropbox accounts with their Clear Channel e-mail addresses and saved **17 TB of company data** to these accounts.

With the account capture feature, Clear Channel was able to secure this data by immediately adding all these users to their Dropbox Business account.

Feature deep dive

User suspension and remote wipe

With Dropbox Business, it takes just a few clicks to suspend a user, wipe all Dropbox data from their devices and prevent him from logging into his account. This ensures company data lost to stolen devices or former employees doesn’t end up in the wrong hands.

Since purchasing Dropbox Enterprise, Clear Channel’s IT team has remote wiped over 75 employee devices, ensuring that Clear Channel’s data stays where it belongs.

EXHIBIT 3



US010067942B2

(12) **United States Patent**
Manzano

(10) **Patent No.:** **US 10,067,942 B2**
(45) **Date of Patent:** ***Sep. 4, 2018**

(54) **ARCHITECTURE FOR MANAGEMENT OF DIGITAL FILES ACROSS DISTRIBUTED NETWORK**

(71) Applicant: **Topia Technology, Inc.**, Tacoma, WA (US)

(72) Inventor: **Michael R. Manzano**, Seattle, WA (US)

(73) Assignee: **TOPIA TECHNOLOGY**, Tacoma, WA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 345 days.

This patent is subject to a terminal disclaimer.

(21) Appl. No.: **14/860,289**

(22) Filed: **Sep. 21, 2015**

(65) **Prior Publication Data**

US 2016/0012067 A1 Jan. 14, 2016

Related U.S. Application Data

(63) Continuation of application No. 12/267,852, filed on Nov. 10, 2008, now Pat. No. 9,143,561.
(Continued)

(51) **Int. Cl.**
G06F 17/30 (2006.01)
G06F 15/16 (2006.01)
H04L 29/08 (2006.01)

(52) **U.S. Cl.**
CPC **G06F 17/30082** (2013.01); **G06F 15/16** (2013.01); **G06F 17/30** (2013.01);
(Continued)

(58) **Field of Classification Search**
CPC .. G06F 17/30; G06F 17/132; G06F 17/30017; G06F 17/30067; G06F 17/3007;
(Continued)

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,600,834 A 2/1997 Howard
5,806,078 A * 9/1998 Hug G06F 17/2288
707/999.202

(Continued)

FOREIGN PATENT DOCUMENTS

EP 1130511 9/2001
WO WO 98/56149 * 12/1998
WO WO 2007047302 4/2007

OTHER PUBLICATIONS

U.S. Office Action dated Aug. 13, 2014 for U.S. Appl. No. 11/739,083.

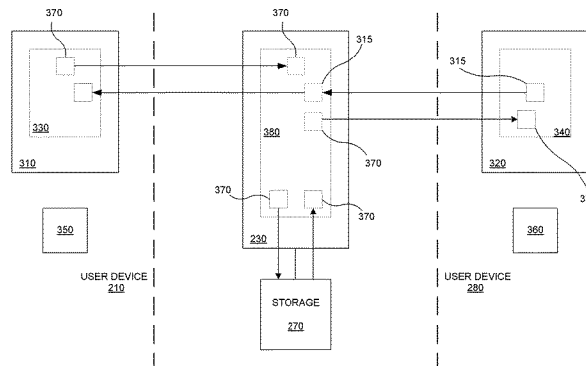
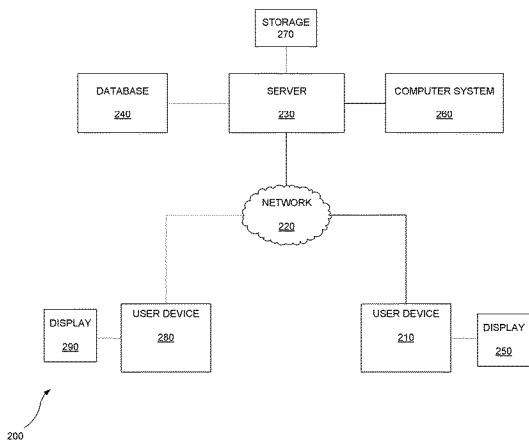
(Continued)

Primary Examiner — Srirama Channavajjala
(74) *Attorney, Agent, or Firm* — Pillsbury Winthrop Shaw Pittman LLP

(57) **ABSTRACT**

A system includes a first application executable on a first electronic device. The system further includes a second application executable on a second electronic device in communication with the first electronic device. The second electronic device is configured to store a first electronic file. Subsequent to a user modifying the first electronic file, the second application is operable to automatically transfer the modified first electronic file, or a copy thereof, to the first electronic device. The system further includes a third application executable on a third electronic device in communication with the first electronic device. The third electronic device is configured to store a second electronic file. Subsequent to the user modifying the second electronic file, the third application is operable to automatically transfer the modified second electronic file, or a copy thereof, to the first electronic device. The first application is operable to automatically transfer the modified first electronic file or copy to the third electronic device, and automatically transfer the

(Continued)



US 10,067,942 B2

Page 2

modified second electronic file or copy to the second electronic device.

18 Claims, 3 Drawing Sheets**Related U.S. Application Data**

(60) Provisional application No. 60/986,896, filed on Nov. 9, 2007.

(52) **U.S. Cl.**

CPC **G06F 17/301** (2013.01); **G06F 17/30088** (2013.01); **G06F 17/30091** (2013.01); **G06F 17/30165** (2013.01); **G06F 17/30174** (2013.01); **H04L 67/1095** (2013.01)

(58) **Field of Classification Search**

CPC G06F 17/3023; G06F 17/2288; G06F 17/30091; G06F 17/30176; G06F 17/30194; G06F 8/71; G06F 17/30082; G06F 17/30165; G06F 17/301; G06F 17/30174; G06F 17/30088

See application file for complete search history.

(56)

References Cited

U.S. PATENT DOCUMENTS

5,909,581 A * 6/1999 Park G06F 8/65
717/170

6,026,414 A 2/2000 Anglin
6,154,817 A 11/2000 Mohan et al.
6,260,069 B1 7/2001 Anglin
6,449,624 B1 9/2002 Hammack et al.
6,463,463 B1 10/2002 Godfrey et al.
6,504,994 B2 1/2003 Kawamura et al.
6,505,200 B1 1/2003 Ims et al.
6,606,646 B2 8/2003 Feigenbaum
6,611,849 B1 8/2003 Raff et al.
6,671,700 B1 12/2003 Creemer et al.
6,708,221 B1 3/2004 Mendez et al.
6,757,696 B2 6/2004 Multer et al.
6,760,759 B1 7/2004 Chan
6,810,405 B1 10/2004 Larue et al.
6,829,622 B2 12/2004 Beyda
6,874,037 B1 3/2005 Abram et al.
6,931,454 B2 8/2005 Deshpande et al.
6,990,522 B2 1/2006 Wu
7,024,428 B1 4/2006 Huang et al.
7,054,594 B2 5/2006 Bloch et al.
7,065,658 B1 6/2006 Baraban et al.
7,089,307 B2 8/2006 Zintel et al.
7,136,934 B2 11/2006 Carter et al.
7,155,488 B1 12/2006 Lunsford et al.
7,224,973 B2 5/2007 Tsutazawa et al.
7,243,163 B1 7/2007 Friend et al.
7,260,646 B1 8/2007 Stefanik et al.
7,269,433 B2 9/2007 Vargas et al.
7,290,244 B2 10/2007 Peck et al.
7,325,038 B1 1/2008 Wang
7,340,534 B2 3/2008 Cameron et al.
7,398,327 B2 7/2008 Lee
7,415,615 B2 8/2008 Skygebjær
7,457,631 B2 11/2008 Yach et al.
7,467,353 B2 12/2008 Kurlander et al.
7,483,925 B2 1/2009 Koskimies et al.
7,526,575 B2 4/2009 Rabbers et al.
7,574,711 B2 8/2009 Zondervan et al.
7,584,186 B2 9/2009 Chen et al.
7,587,446 B1 9/2009 Onyon et al.
7,613,773 B2 11/2009 Watt
7,639,116 B2 12/2009 Saunders
7,657,271 B2 2/2010 Kim

7,680,885 B2 3/2010 Schausser et al.
7,752,166 B2 7/2010 Quinlan et al.
7,761,414 B2 7/2010 Freedman
7,788,303 B2 8/2010 Mikesell et al.
7,895,334 B1 2/2011 Tu et al.
7,987,420 B1 7/2011 Kloba et al.
8,009,966 B2 8/2011 Bloom et al.
8,112,549 B2 2/2012 Srinivasan et al.
8,244,288 B2 8/2012 Chipchase
8,321,534 B1 11/2012 Roskind et al.
8,370,423 B2 2/2013 Ozzie et al.
8,386,558 B2 2/2013 Schleifer et al.
9,143,561 B2 * 9/2015 Manzano G06F 17/30174
2002/0026478 A1 2/2002 Rodgers et al.
2002/0035697 A1 * 3/2002 McCurdy G06F 17/30011
726/3

2002/0087588 A1 7/2002 McBride et al.
2003/0028514 A1 * 2/2003 Lord G06F 11/2064
2003/0028542 A1 2/2003 Muttik et al.
2003/0038842 A1 2/2003 Peck et al.
2003/0078946 A1 * 4/2003 Costello G06F 11/2064
2003/0125057 A1 7/2003 Pesola
2003/0135565 A1 7/2003 Estrada
2004/0049345 A1 3/2004 McDonough et al.
2004/0093361 A1 5/2004 Therrien et al.
2004/0107225 A1 6/2004 Rudoff
2004/0133629 A1 * 7/2004 Reynolds G06F 17/30902
709/202

2004/0158817 A1 * 8/2004 Okachi G06F 8/65
717/122

2004/0172424 A1 9/2004 Edelstein et al.
2005/0091316 A1 4/2005 Ponce et al.
2005/0097225 A1 5/2005 Glatt et al.
2005/0220080 A1 10/2005 Ronkainen et al.
2006/0010150 A1 1/2006 Shaath et al.
2006/0058907 A1 3/2006 Suderman
2006/0074985 A1 * 4/2006 Wolfish G06Q 20/322
2006/0129627 A1 * 6/2006 Phillips H04L 63/10
709/200

2006/0143129 A1 * 6/2006 Holm G06F 8/61
705/52

2006/0168118 A1 * 7/2006 Godlin G06F 17/30212
709/218

2006/0189348 A1 8/2006 Montulli et al.
2007/0014314 A1 1/2007 O'Neil
2007/0016629 A1 * 1/2007 Reinsch G06F 8/71
2007/0027936 A1 * 2/2007 Stakutis G06F 11/1451
2007/0100913 A1 5/2007 Sumner et al.
2007/0180084 A1 * 8/2007 Mohanty G06F 11/1451
709/223

2007/0191057 A1 8/2007 Kamada
2007/0238440 A1 10/2007 Sengupta et al.
2008/0005114 A1 * 1/2008 Li G06F 17/30209
2008/0005280 A1 1/2008 Adams
2008/0086494 A1 4/2008 Heller et al.
2008/0168526 A1 7/2008 Robbin et al.
2008/0288578 A1 11/2008 Silverberg
2009/0013009 A1 * 1/2009 Nakayama G06F 11/1453
2009/0063711 A1 3/2009 Finkelstein
2013/0226871 A1 8/2013 Sarnowski

OTHER PUBLICATIONS

FolderShare—Secure Remote Access VPN Solution; Document Management & Real-time File Mirroring Solution; < <https://web.archive.org/web/20040804020435/http://www.foldershare.com:80/>>; Aug. 4, 2004.

FolderShare; Your Files Anywhere; < <https://web.archive.org/web/20030808183932/http://www.foldershare.com:80/>>; Aug. 8, 2003.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; < <https://web.archive.org/web/20040814015727/http://www.foldershare.com:80/>>; Aug. 14, 2004.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; < <https://web.archive.org/web/20040820052105/http://www.foldershare.com:80/>>; Aug. 20, 2004.

US 10,067,942 B2

Page 3

(56)

References Cited

OTHER PUBLICATIONS

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20041211020957/http://foldershare.com:80/>>; Dec. 11, 2004.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20041217041726/http://foldershare.com:80/>>; Dec. 17, 2004.

FolderShare; Your Smart File Transfer Solution; <<https://web.archive.org/web/20031220151508/http://www.foldershare.com:80/>>; Dec. 20, 2003.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20041230211050/http://www.foldershare.com:80/>>; Dec. 30, 2004.

FolderShare—Secure Remote Access VPN Solution; Document Management & Real-time File Mirroring Solution; <<https://web.archive.org/web/20040701113739/http://foldershare.com:80/>>; Jul. 1, 2004.

FolderShare—Secure Remote Access VPN Solution; Document Management & Real-time File Mirroring Solution; <<https://web.archive.org/web/20040711062548/http://www.foldershare.com:80/>>; Jul. 11, 2004.

FolderShare; Your Files Anywhere; <<https://web.archive.org/web/20030722054342/http://foldershare.com:80/>>; Jul. 22, 2003.

FolderShare—Secure Remote Access VPN Solution; Document Management & Real-time File Mirroring Solution; <<https://web.archive.org/web/20040730030655/http://www.foldershare.com:80/>>; Jul. 30, 2004.

FolderShare—Secure Remote Access VPN Solution; Document Management & Real-time File Mirroring Solution; <<https://web.archive.org/web/20040603205113/http://www.foldershare.com:80/>>; Jun. 3, 2004.

FolderShare—Secure Remote Access VPN Solution; Document Management & Real-time File Mirroring Solution; <<https://web.archive.org/web/20040613161906/http://www.foldershare.com:80/>>; Jun. 13, 2004.

FolderShare—Secure Remote Access VPN Solution; Document Management & Real-time File Mirroring Solution; <<https://web.archive.org/web/20040629075057/http://www.foldershare.com:80/>>; Jun. 29, 2004.

FolderShare—Secure Remote Access VPN Solution; Your Smart File Transfer & Real-time File Mirroring Solution; <<https://web.archive.org/web/20040316235151/http://foldershare.com:80/>>; Mar. 16, 2004.

FolderShare—Secure Remote Access VPN Solution; Your Smart File Transfer & Real-time File Mirroring Solution; <<https://web.archive.org/web/20040325034239/http://www.foldershare.com:80/>>; Mar. 25, 2004.

FolderShare—Secure Remote Access VPN Solution; Document Management & Real-time File Mirroring Solution; <<https://web.archive.org/web/20040512211417/http://www.foldershare.com:80/>>; May 12, 2004.

FolderShare; Your Files Anywhere; <<https://web.archive.org/web/20030531180252/http://www.foldershare.com:80/>>; May 31, 2003.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20041104031510/http://www.foldershare.com:80/>>; Nov. 4, 2004.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20041117092357/http://www.foldershare.com:80/>>; Nov. 17, 2004.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20041123085254/http://www.foldershare.com:80/>>; Nov. 23, 2004.

FolderShare; Your Files Anywhere; <<https://web.archive.org/web/20031128143634/http://foldershare.com:80/>>; Nov. 28, 2003.

FolderShare; Your Files Anywhere; <<https://web.archive.org/web/20031001071631/http://foldershare.com:80/>>; Oct. 1, 2003.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20041012083127/http://www.foldershare.com:80/>>; Oct. 12, 2004.

FolderShare—Secure Remote Access VPM Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20041029085820/http://www.foldershare.com:80/>>; Oct. 29, 2004.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20040901034646/http://www.foldershare.com:80/>>; Sep. 1, 2004.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20040909075254/http://www.foldershare.com:80/>>; Sep. 9, 2004.

FolderShare; Your Files Anywhere; <<https://web.archive.org/web/20030920051943/http://www.foldershare.com:80/>>; Sep. 20, 2003.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20040924032146/http://www.foldershare.com:80/>>; Sep. 24, 2004.

* cited by examiner

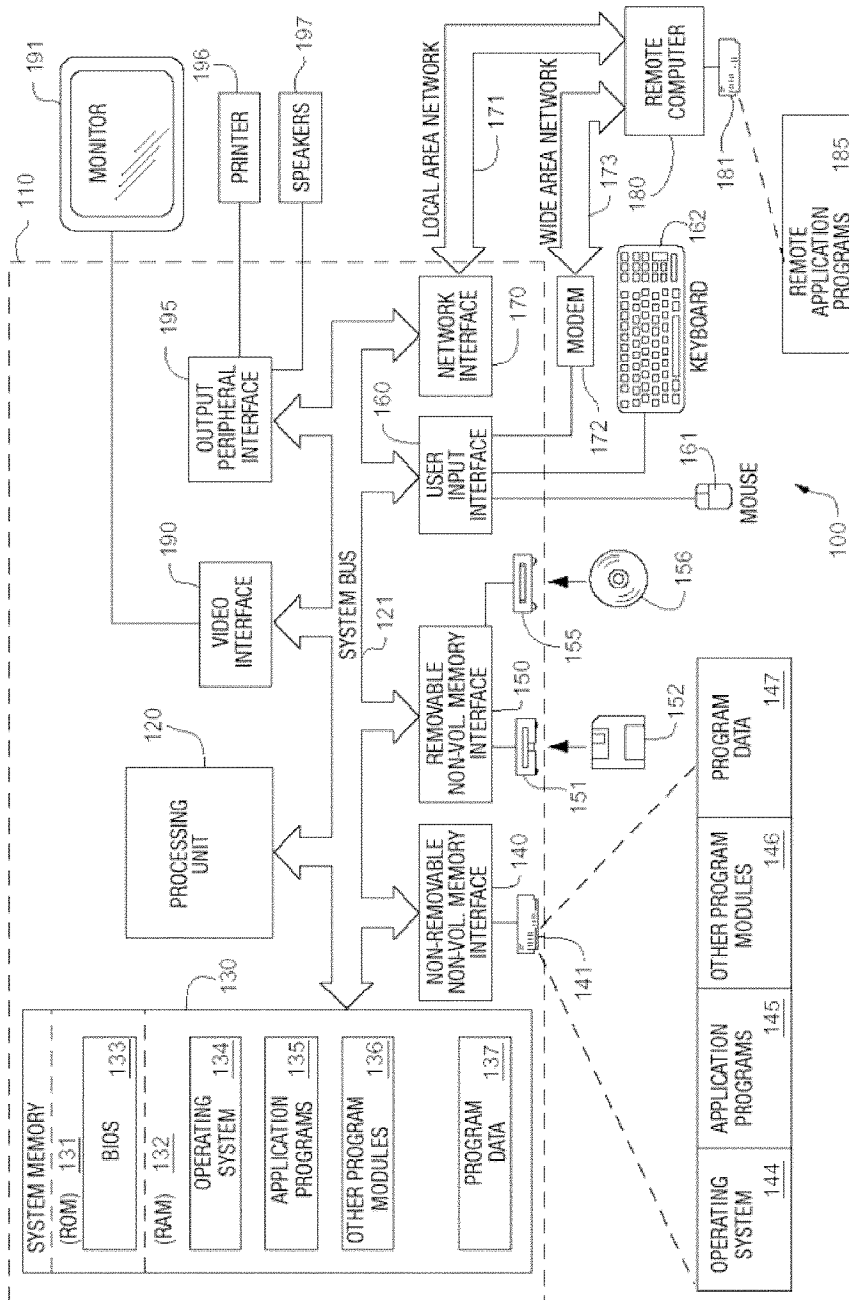


FIG. 1

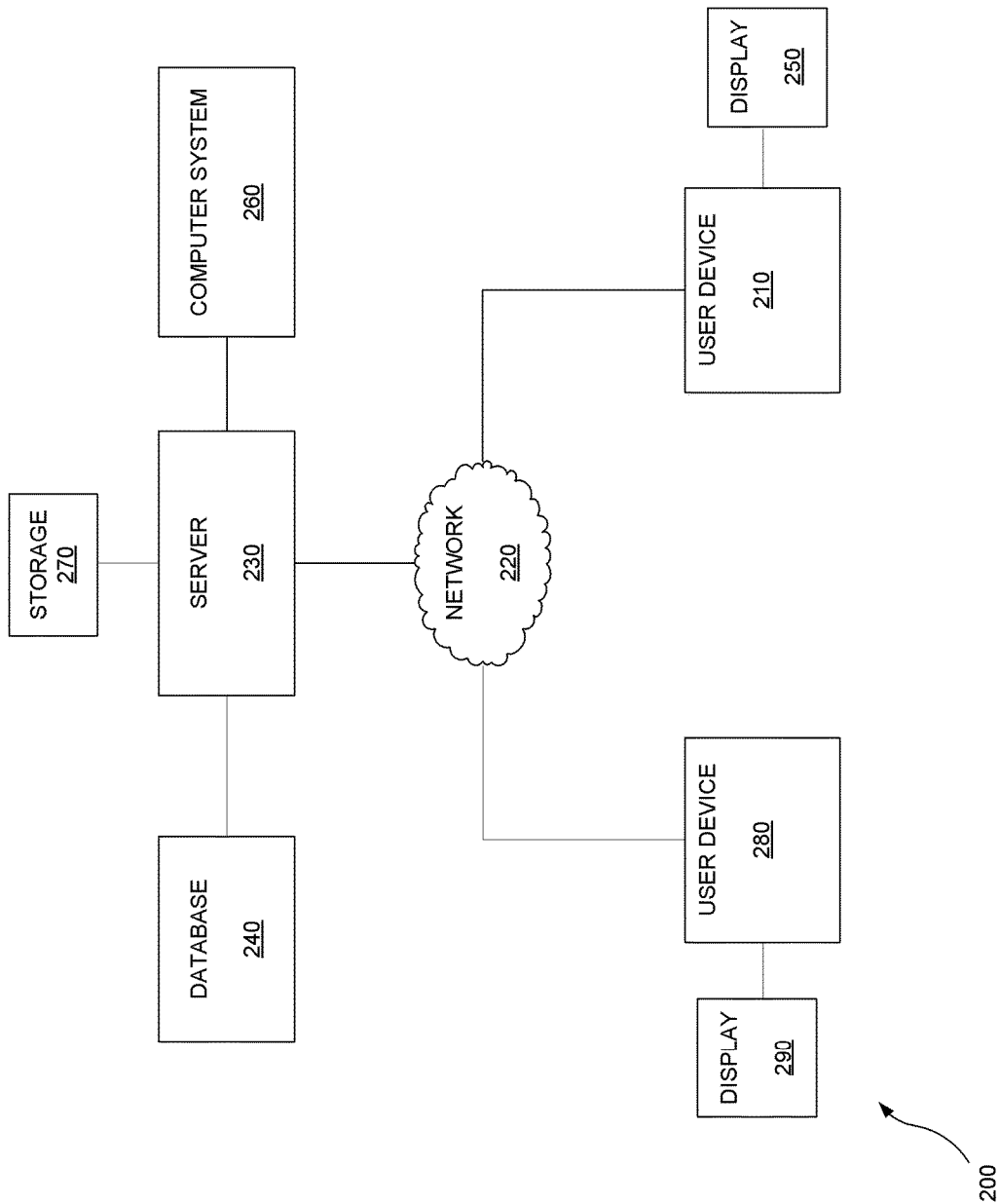


FIG. 2

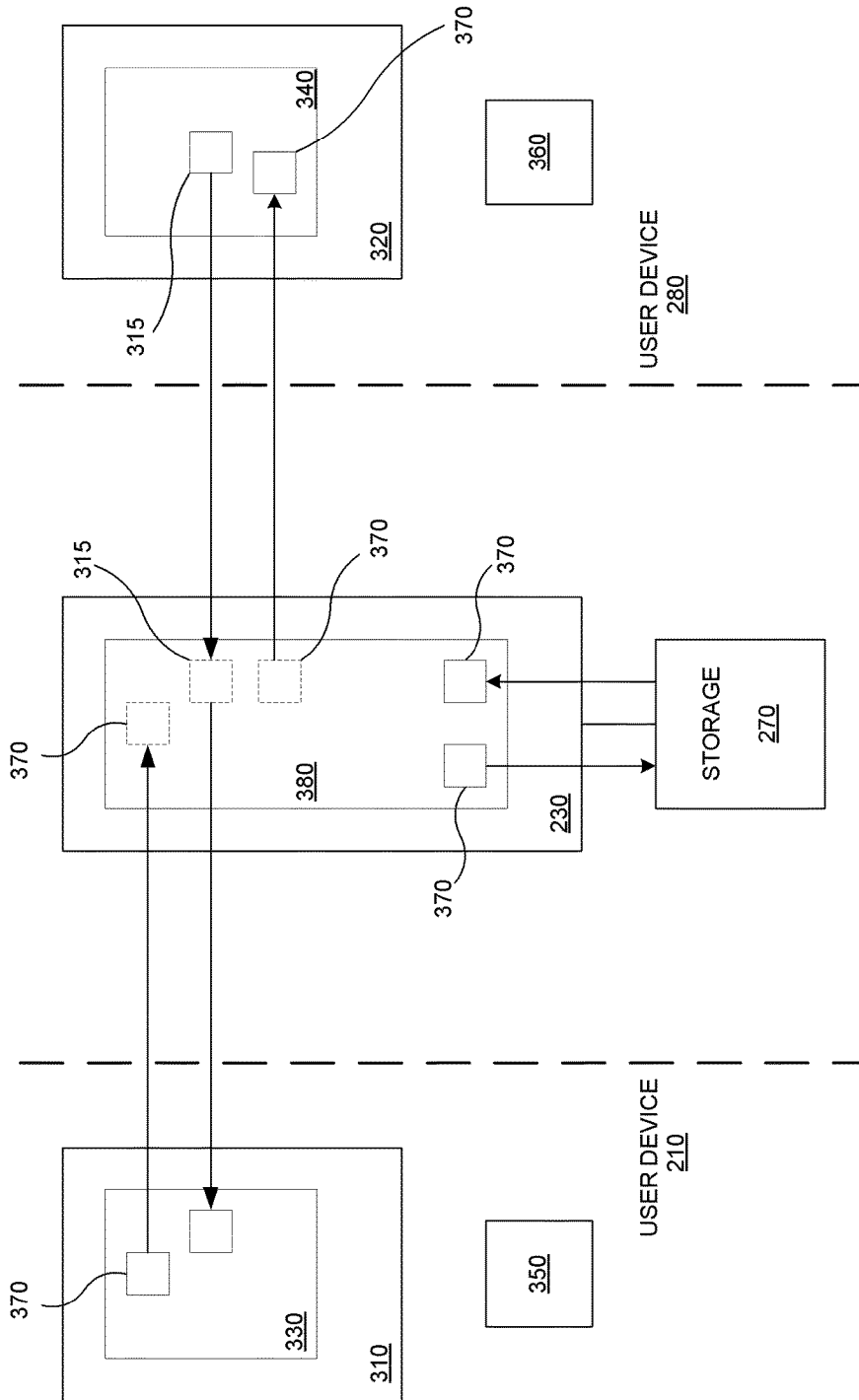


FIG. 3

US 10,067,942 B2

1

ARCHITECTURE FOR MANAGEMENT OF DIGITAL FILES ACROSS DISTRIBUTED NETWORK

CROSS-REFERENCE TO RELATED APPLICATIONS

This application is a continuation of U.S. patent application Ser. No. 12/267,852, filed Nov. 10, 2008, which claims priority to U.S. Provisional Application No. 60/986,896 entitled "ARCHITECTURE FOR MANAGEMENT OF DIGITAL FILES ACROSS DISTRIBUTED NETWORK" and filed Nov. 9, 2007, the contents of which are hereby incorporated by reference in their entirety.

FIELD OF THE INVENTION

This invention relates generally to computer-implemented processes and, more specifically, to sharing of electronic files among electronic devices.

BACKGROUND OF THE INVENTION

Users of modern computing systems are increasingly finding themselves in constantly-connected, high-speed networked environments. The Web continues to be a killer application, second only to email, on the Internet. Further, customers are increasingly using more than one computing device; a customer may have a desktop computer at home, one at work, and a constantly connected "smart phone". Due to the confluence of these two trends, file management across these devices has become a problem.

Although modern devices are easily connected, they do not provide the customer a seamless environment; the customer must manually handle many aspects of that connection. With regards to file management, customers must manually move files between their devices using some protocol like email, ftp, or by posting them on the Web. These practices lead to problems that include:

The proliferation of redundant file copies. This proliferation creates a confusing environment where the customer is unclear where the "official" or newest version of a file exists.

The creation of an error-prone environment. Some documents, such as those associated with word processing and desktop publishing, externally reference other files. Copying such a document can break these references causing errors that the customer has to handle manually. An example of such a document is a desktop publishing document that contains a reference to an image. If that image file is not transferred along with the desktop publishing file, the image will appear as a broken link.

Unnecessary complexity. Because devices tend to have their own filing system, customers must manage a different filing model on each of his devices. For example, instead of having a single "Movies" folder, he may have to deal with many "Movies" folders, which may be in different locations on each of his devices. Each device may also have its own security model, further complicating the matter.

That a customer has to manually move files around to ensure their accessibility on his devices is unnecessary, and is an indicator of a lack of customer-focused design in modern file systems. File systems in use today are direct offspring of systems used when graphical customer interfaces were nonexistent. Modern file system customer inter-

2

faces, such as Windows® Explorer and Mac OS X's Finder are just now starting to provide experiences that are more in line to a customer's workflow. Whereas, before, these interfaces were concerned with representing files with abstracted icons, the file's actual contents are becoming paramount in how files are organized and presented.

Problems still exist with how these newer customer interfaces are implemented. They are not completely integrated with applications, suffer from performance problems, and do not generally work well outside of a device's local file system.

There are several solutions to this problem that are in one way or another inadequate to the task:

Remote Desktop software allows a customer to remotely "see" his desktop. Remote desktop software screen-scrapes a remote machine's screen (a "server") and displays it on a screen local to the customer (a "client"). Remote desktop gives a customer access to not only his files, but also to his applications. However, this approach requires that the host machine be turned on and connected to the internet at all times. Consequently, this approach would not be appropriate for mobile hosts such as laptops. Remote desktop does not use the resources of a local machine. For full accessibility, the customer would have to keep all files and application on the host machine as any files stored on a client are not guaranteed to be accessible.

Distributed File Systems, like remote desktop software, place data on an always-connected host machine. Unlike remote desktop software, the host machine is not one on which the customer performs computing tasks. The host machine is used as a storage mechanism, and any computation performed on that machine serves to support its use as such. Distributed file systems generally provide the right functionality for customers to share files between their devices. However, distributed file systems are usually deployed as a shared resource; that is, other customers have access to it. Because of this sharing, a customer's files may be buried deep in a filing structure, and it may not always be immediately evident to customers what kind of access they have to a particular file. Further, to use a distributed file system, the customer must always be connected to it. Files stored on a distributed file system are generally inaccessible if the customer's machine is not connected to it, unless the customer has copied or moved the files to his machine's local hard drive. However, doing so immediately creates the problem of having two filing systems for the same file, creating a mental burden on the customer.

Additionally, accessing a file located on a distributed file system tends to be slower than accessing files on the local hard drive. Modern applications are usually written to assume that the files they access are located locally, and thus are not optimized to access remote files. When these applications are used with remote files, they can lose performance by an order of magnitude. This problem can be fixed by automatically caching often-used files on the local file system, and only synchronizing them when they have been changed. However, this separate synchronization step introduces another problem: because the synchronization process can be lengthy, the customer is never entirely sure if the file he is remotely accessing is the latest version of the file, versus an earlier one that has been marked to be updated. Further, the directory may not reflect the existence of the file at all until synchronization finishes.

FTP is similar to a distributed file system with regards to files being hosted on a remote server. However FTP generally does manifest as a "disk drive" on the customer's desktop; the customer must use special FTP client software

US 10,067,942 B2

3

to access an FTP server. It shares the same problem as distributed file systems, with the additional problem of weak integration with applications. Applications can generally write and read files directly to and from a distributed file system. This is not the case with FTP, as the customer has to manually use the client software to perform these operations as a separate task.

Email was originally invented for messaging. From the beginning, the model it employs to make files accessible remotely is necessarily inefficient. Email's model for making files accessible is in the form of an email "attachment". Attachments are so named because they piggy-back on a message sent from one customer to another. A customer can make a file remotely available using email by attaching the file to an email and sending it to himself. He can then retrieve the file from a remote location by accessing the message on the email server. Email used in this way is even worse than FTP as the process is even more manual: a customer must find the message containing the file before he can even access it. Further, the location in which the attachment lives is read only. If the customer, for example, were to open the file, change it, then save it back out, the results would be ambiguous to the user because the email application, not the user, specified its location. Usually, the saved file would end up buried in an email file cache in an undisclosed area of the file system.

Flash Drives and External Disk Drives, although seemingly the most "primitive" way to ensure file availability, avoid all the problems related to network latency. However, these devices must be physically connected to the computer on which the files will be accessed. These restrictions preclude the customer from employing several effective work-flows including: using more than one computer to complete a single task (the files can only be accessed on one computer) and setting up an automated backup (the computer running the backup can't guarantee that the storage device will be connected come backup time). Further, to ensure full availability of the files, the customer must carry the device with them at all times, and must follow the associated protocols for mounting and dismounting the device.

Other problems with the prior art not described above can also be overcome using the teachings of embodiments of the present invention, as would be readily apparent to one of ordinary skill in the art after reading this disclosure.

SUMMARY OF THE INVENTION

In an embodiment, a system includes a first application executable on a first electronic device. The system further includes a second application executable on a second electronic device in communication with the first electronic device. The second electronic device is configured to store a first electronic file. Subsequent to a user modifying the first electronic file, the second application is operable to automatically transfer the modified first electronic file, or a copy thereof, to the first electronic device. The system further includes a third application executable on a third electronic device in communication with the first electronic device. The third electronic device is configured to store a second electronic file. Subsequent to the user modifying the second electronic file, the third application is operable to automatically transfer the modified second electronic file, or a copy thereof, to the first electronic device. The first application is operable to automatically transfer the modified first electronic file or copy to the third electronic device, and auto-

4

matically transfer the modified second electronic file or copy to the second electronic device.

BRIEF DESCRIPTION OF THE DRAWING

Preferred and alternative embodiments of the present invention are described in detail below with reference to the following drawings.

FIG. 1 is a schematic view of an exemplary operating environment in which an embodiment of the invention can be implemented;

FIG. 2 is a functional block diagram of an exemplary operating environment in which an embodiment of the invention can be implemented; and

FIG. 3 is a functional block diagram illustrating file sharing and/or synchronization according to an embodiment of the invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

An embodiment of the invention leverages remote programming concepts by utilizing processes called mobile agents (sometimes referred to as mobile objects or agent objects). Generally speaking, these concepts provide the ability for an object (the mobile agent object) existing on a first ("host") computer system to transplant itself to a second ("remote host") computer system while preserving its current execution state. The operation of a mobile agent object is described briefly below.

The instructions of the mobile agent object, its preserved execution state, and other objects owned by the mobile agent object are packaged, or "encoded," to generate a string of data that is configured so that the string of data can be transported by all standard means of communication over a computer network. Once transported to the remote host, the string of data is decoded to generate a computer process, still called the mobile agent object, within the remote host system. The decoded mobile agent object includes those objects encoded as described above and remains in its preserved execution state. The remote host computer system resumes execution of the mobile agent object which is now operating in the remote host environment.

While now operating in the new environment, the instructions of the mobile agent object are executed by the remote host to perform operations of any complexity, including defining, creating, and manipulating data objects and interacting with other remote host computer objects.

File transfer and/or synchronization, according to an embodiment, may be accomplished using some or all of the concepts described in commonly owned U.S. patent application Ser. No. 11/739,083, entitled "Electronic File Sharing," the entirety of which is incorporated by reference as if fully set forth herein.

FIG. 1 illustrates an example of a suitable computing system environment **100** in which one or more embodiments of the invention may be implemented. The computing system environment **100** is only one example of a suitable computing environment and is not intended to suggest any limitation as to the scope of use or functionality of the invention. Neither should the computing environment **100** be interpreted as having any dependency or requirement relating to any one or combination of components illustrated in the exemplary operating environment **100**.

Embodiments of the invention are operational with numerous other general purpose or special purpose computing system environments or configurations. Examples of

US 10,067,942 B2

5

well known computing systems, environments, and/or configurations that may be suitable for use with the invention include, but are not limited to, personal computers, server computers, hand-held or laptop devices, multiprocessor systems, microprocessor-based systems, set top boxes, programmable consumer electronics, network PCs, minicomputers, mainframe computers, distributed computing environments that include any of the above systems or devices, and the like.

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

With reference to FIG. 1, an exemplary system for implementing the invention includes a general purpose computing device in the form of a computer 110. Components of computer 110 may include, but are not limited to, a processing unit 120, a system memory 130, and a system bus 121 that couples various system components including the system memory to the processing unit 120. The system bus 121 may be any of several types of bus structures including a memory bus or memory controller, a peripheral bus, and a local bus using any of a variety of bus architectures. By way of example, and not limitation, such architectures include Industry Standard Architecture (ISA) bus, Micro Channel Architecture (MCA) bus, Enhanced ISA (EISA) bus, Video Electronics Standards Association (VESA) local bus, and Peripheral Component Interconnect (PCI) bus also known as Mezzanine bus.

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

6

acoustic, RF, infrared and other wireless media. Combinations of the any of the above should also be included within the scope of computer readable media.

The system memory 130 includes computer storage media in the form of volatile and/or nonvolatile memory such as read only memory (ROM) 131 and random access memory (RAM) 132. A basic input/output system 133 (BIOS), containing the basic routines that help to transfer information between elements within computer 110, such as during start-up, is typically stored in ROM 131. RAM 132 typically contains data and/or program modules that are immediately accessible to and/or presently being operated on by processing unit 120. By way of example, and not limitation, FIG. 1 illustrates operating system 134, application programs 135, other program modules 136, and program data 137.

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

The drives and their associated computer storage media discussed above and illustrated in FIG. 1, provide storage of computer readable instructions, data structures, program modules and other data for the computer 110. In FIG. 1, for example, hard disk drive 141 is illustrated as storing operating system 144, application programs 145, other program modules 146, and program data 147. Note that these components can either be the same as or different from operating system 134, application programs 135, other program modules 136, and program data 137. Operating system 144, application programs 145, other program modules 146, and program data 147 are given different numbers here to illustrate that, at a minimum, they are different copies. A user may enter commands and information into the computer 110 through input devices such as a keyboard 162 and pointing device 161, commonly referred to as a mouse, trackball or touch pad. Other input devices (not shown) may include a microphone, joystick, game pad, satellite dish, scanner, or the like. These and other input devices are often connected to the processing unit 120 through a user input interface 160 that is coupled to the system bus, but may be connected by other interface and bus structures, such as a parallel port, game port or a universal serial bus (USB). A monitor 191 or other type of display device is also connected to the system bus 121 via an interface, such as a video interface 190. In addition to the monitor, computers may also include other peripheral output devices such as speakers 197 and printer 196, which may be connected through an output peripheral interface 190.

The computer 110 may operate in a networked environment using logical connections to one or more remote computers, such as a remote computer 180. The remote computer 180 may be a personal computer, a server, a router,

US 10,067,942 B2

7

a network PC, a peer device or other common network node, and typically includes many or all of the elements described above relative to the computer 110, although only a memory storage device 181 has been illustrated in FIG. 1. The logical connections depicted in FIG. 1 include a local area network (LAN) 171 and a wide area network (WAN) 173, but may also include other networks. Such networking environments are commonplace in offices, enterprise-wide computer networks, intranets and the Internet.

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

Referring now to FIG. 2, an embodiment of the present invention can be described in the context of an exemplary computer network system 200 as illustrated. System 200 includes electronic user devices 210, 280, such as personal computers or workstations, that are linked via a communication medium, such as a network 220 (e.g., the Internet), to an electronic device or system, such as a server 230. The server 230 may further be coupled, or otherwise have access, to a database 240, electronic storage 270 and a computer system 260. Although the embodiment illustrated in FIG. 2 includes one server 230 coupled to two user devices 210, 280 via the network 220, it should be recognized that embodiments of the invention may be implemented using two or more such user devices coupled to one or more such servers.

In an embodiment, each of the user devices 210, 280 and server 230 may include all or fewer than all of the features associated with the computer 110 illustrated in and discussed with reference to FIG. 1. User devices 210, 280 include or are otherwise coupled to a computer screen or display 250, 290, respectively. User devices 210, 280 can be used for various purposes including both network- and local-computing processes.

The user devices 210, 280 are linked via the network 220 to server 230 so that computer programs, such as, for example, a browser or other applications, running on one or more of the user devices 210, 280 can cooperate in two-way communication with server 230 and one or more applications running on server 230. Server 230 may be coupled to database 240 and/or electronic storage 270 to retrieve information therefrom and to store information thereto. Additionally, the server 230 may be coupled to the computer system 260 in a manner allowing the server to delegate certain processing functions to the computer system.

Referring now to FIG. 3, illustrated is functionality of an embodiment of the invention allowing a user (not shown) who owns or otherwise controls devices 210, 280 to automatically maintain file synchronization between at least devices 210, 280, or any other user devices on which principles of the present invention are implemented. In an embodiment, an administrator (not shown) of the server 230 or other appropriate electronic device transfers a file-transfer

8

and/or synchronization application to the user devices 210, 280 for installation thereon. Once installed on the user devices 210, 280, the file-transfer application provides file-transfer clients 310, 320 executable by the user devices 210, 280, respectively. Each of the file-transfer clients 310, 320 may, but need not, include a respective mobile-agent runtime environment 330, 340. The mobile-agent runtime environment 330, 340 include portions of memory of the user devices 210, 280 dedicated to allowing a mobile object the ability to perform operations that the mobile object is programmed to carry out. Also included in the file-transfer application are user interfaces 350, 360 that are displayable on the displays 250, 290, respectively. In an embodiment, the interfaces 350, 360 allow a user to view, access and/or organize files to be synched among the various user devices.

Generally, all files that the user desires to be synched or shared may at some point be uploaded by one or more of the user devices 210, 280 and stored in storage 270. Upon receiving the files to be synched, the server 230 can store such files in the storage 270 and/or transfer the files to one or more of the respective hard drives of the user devices 210, 280, thereby enabling each respective user device to access such files. In this manner, the server 230 is operable to treat each hard drive of the respective user devices 210, 280 as a local document cache for files received by the server. Typically, the server 230 will store one or more of the received files to the storage 270 only if the destination user device is offline or otherwise temporarily not in communication with the server 230. Upon resuming communication with the destination user device, the server 230 will transfer the temporarily stored files to the destination device.

In operation, according to an embodiment, the user may open and modify a file 370, such as a word-processing document or other electronic file. Alternatively, the user may create a first instance of the file 370. The user may have previously have associated, or may now associate, the file 370 with the transfer client 310. Upon a predetermined and user-configurable triggering event, the transfer client 310 transfers the modified file 370, or a copy of the modified file, to the server 230. Such a triggering event may include, but be not limited to, the user saving the file, the elapsing of a predetermined amount of time during which the file has been opened, or the re-initiation of a communication session between the device 210 and the server 230.

The file 370 is transferred to the server 230 on which is executing a synchronization application 380, which may include a mobile-agent runtime environment. Through user configuration, the synch application 380 monitors a set of user devices to which the file 370 should be transferred to effect file synchronization. In the illustrated embodiment, this set of user devices includes the user device 280. The synch application 380 polls the device 280 to determine whether the device 280 is in communication with the server 230. If the device 280 is in communication with the server 230, the synch application 380 transfers the file 370 to the device 280, whereupon the transfer client 320 resident on the device 280 replaces the previous version of the file 370, previously cached on the device 280, with the latest version of the file 370 modified on the user device 210. If the device 280 is not currently in communication with the server 230, the synch application 380 may store the file 370 in the storage 270 until such time as communication between the device 280 and server 230 is reestablished. As illustrated in FIG. 3, a similar reverse-direction synchronization process may be performed by the synch application 380 and the transfer clients 310, 320 with regard to a file 315 modified on device 280 and synchronized to device 210.

In an embodiment, the user interfaces **350**, **360** may include a list of the customer's documents and related metadata, as well as any one-to-one or one-to-many relationships between the documents and metadata. An embodiment can always provide customers with an accurate "picture" of their document collection, regardless of whether their devices physically contain the documents. As alluded to earlier, a problem with distributed file systems and FTP is the latency between a file being put onto a file system and it showing up on a remote machine. To prevent this problem, an embodiment directory is decoupled from the movement of files. An embodiment's directory update system updates at a higher priority than the documents to be synchronized. This feature ensures that when a customer browses or searches through his set of documents, they appear even if they have not yet been cached locally on the user device. An indicator signifying a document's availability may be prominently displayed adjacent to the document's representation so that customers are aware of the document's availability.

An embodiment may include a stand-alone application that allows customers to find and manage documents associated with transfer clients **310**, **320** by visualizing relationships between documents and their metadata. It allows customers to tag documents with any number of identifiers. Customers can relate both documents and tags with each other in any number of user-specified one-to-one and one-to-many relationships, and an embodiment provides a user interface to browse and search on these relationships. To mitigate the customers' learning curve, an embodiment can implement relationships common to contemporary file systems, including a folder hierarchy. In addition to this, an embodiment provides direct support for methods that the customer uses to organize documents by manifesting them as user interface idioms. This is unlike conventional document filing systems which require the customer to work within a strict folder metaphor for organization.

Some alternate methods that an embodiment supports for organizing documents include:

Allow customers to organize their documents by application. Many times customers remember the application used to create a document instead of the document's name or its location in a hierarchy.

Allow customers to organize their documents by most recent access. Customers are likely to access a document they've accessed in the near past. Usually, such documents are part of a task that the customer is actively working.

Allow customers to organize their documents by project or subproject.

Allow customers to organize their documents by people. Many times, especially in the context of a collaboration, a document is directly related to one or more people other than the customer.

Allow the customer to organize their document by process stage. Documents may represent one or more stages of a process. Customers need a method for organizing documents by process stage, and a mechanism for moving the document through a set of predefined stages.

Allow customers to organize their documents by any of the aforementioned methods concurrently. These organization methods are not mutually exclusive.

An embodiment presents an interface that allows a customer to locate one or more documents associated with the transfer clients **310**, **320** and open such document into a separate software application. Since this interface is intended to be used from within the separate application, that

application may need to know how to invoke such interface. Advantageously, this invocation behavior can be provided to the application using the application's plug-in API.

An embodiment presents an interface that allows a customer to synchronize a currently opened document according to processes described elsewhere herein. This interface can be invoked within an application and can be made available to the application in the manner described above in connection with the application's plug-in API.

Some files associated with the transfer clients **310**, **320** are dependent on other files associated with the transfer clients **310**, **320**. For example, a desktop publishing document may include images that are stored in files separate from the main document. Previous file-synching solutions treat these files as separate. Because of this, for example, a document synchronized from the device **210** to the device **280** may be opened by the user of the device **280** before the image files have been fully transferred to the device **280**. This causes the document to fail to open, or break, since the image files don't exist or are incomplete. An embodiment prevents this by: (1) always ensuring the file catalog (e.g., the stand-alone application that allows customers to find and manage documents associated with transfer clients **310**, **320**, as discussed above herein) is synchronized before any file data is synchronized, and (2) pausing any file access by any program until the file contents have been fully synchronized. In such an embodiment, if a user attempts, using a software program, to open a file whose related files haven't yet finished transferring to the local (hard drive) cache, if that software attempts to open the related files, the software program is blocked by an embodiment until the requested files are downloaded and ready to access.

Other file sending and synchronizing software requires the user to upload their data to a storage device owned by the operator of the service. An embodiment treats storage as a participant in the synchronization process; this means that the user can choose the service or device where their files will be stored. The file transfer/synching is abstracted from the storage system allowing any storage to be used. An embodiment treats storage like any other synch target, such as a desktop computer, or a cell phone. As such, any device owned or otherwise controlled by the user and running a synch application, such as synch application **380**, as provided in an embodiment of the invention can perform the storage and/or synching functions described elsewhere herein. That is, the user device **280** or user device **210**, rather than the server **230**, may perform such functions.

While a preferred embodiment of the invention has been illustrated and described, as noted above, many changes can be made without departing from the spirit and scope of the invention. For example, as an alternative to the approach described with reference to FIG. 3, wherein the transfer clients **310**, **320** function to "push" modified or created files to the synch application **380**, the synch application **380** may instead function to periodically "pull" or otherwise actively retrieve such files from the transfer clients **310**, **320**. Instead, the invention should be determined entirely by reference to the claims that follow.

What is claimed is:

1. A system comprising:

a first electronic device, associated with a user, configured to:

receive, via a first application at the first electronic device, a copy of a modified first electronic file from a second application at a second electronic device associated with the user, wherein the modified first electronic file copy is automatically received from

US 10,067,942 B2

11

the second application responsive to the user modifying a content of the first electronic file; determine whether the first electronic device is in communication with a third electronic device; automatically send, via the first application, the modified first electronic file copy to a third application at the third electronic device responsive to the determination that the first electronic device is in communication with the third electronic device and responsive to receiving the modified first electronic file copy from the second electronic device; receive, via the first application, a copy of a modified second electronic file from the third application at the third electronic device associated with the user, wherein the modified second electronic file copy is automatically received from the third application responsive to the user modifying a content of the second electronic file; determine whether the first electronic device is in communication with the second electronic device; and automatically send, via the first application, the modified second electronic file copy to the second application at the second electronic device responsive to the determination that the first electronic device is in communication with the second electronic device and responsive to receiving the modified second electronic file copy from the third electronic device, wherein, responsive to sending the modified first electronic file copy to the third electronic device, an older version of the first electronic file stored on the third electronic device is automatically caused to be replaced with the modified first electronic file copy such that the modified first electronic file copy is stored on the third electronic device in lieu of the older version of the first electronic file, and wherein, responsive to sending the modified second electronic file copy to the second electronic device, an older version of the second electronic file stored on the second electronic device is automatically caused to be replaced with the modified second electronic file copy such that the modified second electronic file copy is stored on the second electronic device in lieu of the older version of the second electronic file.

2. The system of claim 1, wherein at least one of the second application or the third application comprises a runtime environment for mobile-agent objects.

3. The system of claim 1, wherein the first electronic device is configured to:

store, via the first application, the modified first electronic file copy to a memory device associated with the first electronic device.

4. The system of claim 3, wherein the modified first electronic file copy is stored on the memory device associated with the first electronic device in lieu of an older version of the first electronic file copy that was stored on the memory device associated with the first electronic device.

5. The system of claim 3, wherein the modified first electronic file copy is stored on the memory device associated with the first electronic device responsive to a determination that the first electronic device is not in communication with the third electronic device.

6. The system of claim 5, wherein the first electronic device is configured to:

responsive to resuming communication with the third electronic device, automatically transfer, via the first

12

application, the modified first electronic file copy to the third electronic device to cause the older version of the first electronic file stored on the third electronic device to be replaced with the modified first electronic file copy.

7. The system of claim 1, wherein the second application is operable to create a first mobile object, and wherein the first mobile object is operable to create a proxy object at the first electronic device.

8. The system of claim 7, wherein the first mobile object is operable to provide the modified first electronic file copy to the proxy object.

9. The system of claim 8, wherein the proxy object is operable to store the modified first electronic file copy on a memory device associated with the first electronic device.

10. A method comprising:

receiving, via a first application at a first electronic device, a copy of a modified first electronic file from a second application at a second electronic device associated with a user, wherein the modified first electronic file copy is automatically received from the second application responsive to the user modifying a content of the first electronic file;

determining whether the first electronic device is in communication with a third electronic device;

automatically sending, via the first application, the modified first electronic file copy to a third application at the third electronic device responsive to the determination that the first electronic device is in communication with the third electronic device and responsive to receiving the modified first electronic file copy from the second electronic device;

receiving, via the first application, a copy of a modified second electronic file from the third application at the third electronic device associated with the user, wherein the modified second electronic file copy is automatically received from the third application responsive to the user modifying a content of the second electronic file;

determining whether the first electronic device is in communication with the second electronic device; and automatically sending, via the first application, the modified second electronic file copy to the second application at the second electronic device responsive to the determination that the first electronic device is in communication with the second electronic device and responsive to receiving the modified second electronic file copy from the third electronic device,

wherein, responsive to sending the modified first electronic file copy to the third electronic device, an older version of the first electronic file stored on the third electronic device is automatically caused to be replaced with the modified first electronic file copy such that the modified first electronic file copy is stored on the third electronic device in lieu of the older version of the first electronic file, and

wherein, responsive to sending the modified second electronic file copy to the second electronic device, an older version of the second electronic file stored on the second electronic device is automatically caused to be replaced with the modified second electronic file copy such that the modified second electronic file copy is stored on the second electronic device in lieu of the older version of the second electronic file.

11. The method of claim 10, wherein at least one of the second application or the third application comprises a runtime environment for mobile-agent objects.

12. The method of claim 10, further comprising:
storing, via the first application, the modified first elec-
tronic file copy to a memory device associated with the
first electronic device.

13. The method of claim 12, wherein the modified first 5
electronic file copy is stored on the memory device associ-
ated with the first electronic device in lieu of an older
version of the first electronic file copy that was stored on the
memory device associated with the first electronic device.

14. The method of claim 12, wherein the modified first 10
electronic file copy is stored on the memory device associ-
ated with the first electronic device responsive to a deter-
mination that the first electronic device is not in communi-
cation with the third electronic device.

15. The method of claim 14, further comprising: 15
responsive to resuming communication with the third
electronic device, automatically transferring, via the
first application, the modified first electronic file copy
to the third electronic device to cause the older version
of the first electronic file stored on the third electronic 20
device to be replaced with the modified first electronic
file copy.

16. The method of claim 10, wherein the second appli-
cation is operable to create a first mobile object, and wherein
the first mobile object is operable to create a proxy object at 25
the first electronic device.

17. The method of claim 16, wherein the first mobile
object is operable to provide the modified first electronic file
copy to the proxy object.

18. The method of claim 17, wherein the proxy object is 30
operable to store the modified first electronic file copy on a
memory device associated with the first electronic device.

* * * * *

EXHIBIT 4



US010289607B2

(12) **United States Patent**
Manzano

(10) **Patent No.:** **US 10,289,607 B2**
(45) **Date of Patent:** ***May 14, 2019**

(54) **ARCHITECTURE FOR MANAGEMENT OF DIGITAL FILES ACROSS DISTRIBUTED NETWORK**

(58) **Field of Classification Search**
CPC G06F 17/30082; G06F 17/30067; G06F 17/3023; G06F 17/30194; G06F 17/30174;

(71) Applicant: **TOPIA TECHNOLOGY, INC.**,
Tacoma, WA (US)

(Continued)

(72) Inventor: **Michael R. Manzano**, Seattle, WA (US)

(56) **References Cited**

U.S. PATENT DOCUMENTS

(73) Assignee: **TOPIA TECHNOLOGY, INC.**,
Tacoma, WA (US)

5,600,834 A 2/1997 Howard
5,675,802 A * 10/1997 Allen G06F 8/71
707/999.202

(Continued)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

This patent is subject to a terminal disclaimer.

FOREIGN PATENT DOCUMENTS

EP 1 130 511 A2 9/2001
WO WO 98/56149 A1 12/1998
WO WO 2007/047302 A2 4/2007

OTHER PUBLICATIONS

FolderShare—Secure Remote Access VPN Solution; Document Management & Real-time File Mirroring Solution; < <https://web.archive.org/web/20040804020435/http://www.foldershare.com:80/>>; Aug. 4, 2004.

(Continued)

Primary Examiner — Srirama Channavajjala

(74) *Attorney, Agent, or Firm* — Pillsbury Winthrop Shaw Pittman LLP

(21) Appl. No.: **16/017,348**

(22) Filed: **Jun. 25, 2018**

(65) **Prior Publication Data**

US 2018/0307698 A1 Oct. 25, 2018

Related U.S. Application Data

(63) Continuation of application No. 14/860,289, filed on Sep. 21, 2015, now Pat. No. 10,067,942, which is a (Continued)

(51) **Int. Cl.**

G06F 16/00 (2019.01)

G06F 16/11 (2019.01)

(Continued)

(52) **U.S. Cl.**

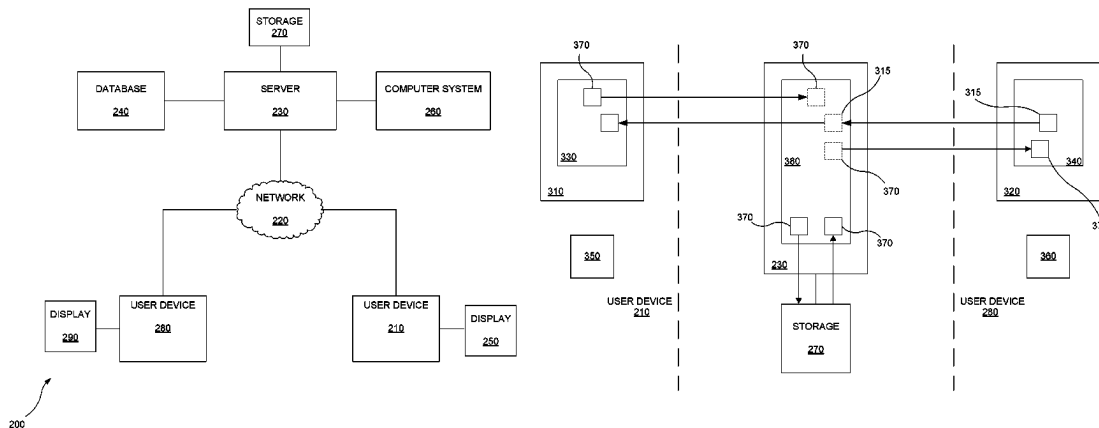
CPC **G06F 16/122** (2019.01); **G06F 15/16** (2013.01); **G06F 16/00** (2019.01); **G06F 16/128** (2019.01);

(Continued)

(57) **ABSTRACT**

In certain embodiments, automatic modification-triggered transfer of a file among two or more computer systems associated with a user. In some embodiments, a copy of a first file may be received, via a first application at a first computer system, from a second application at a second computer system associated with a user. The first file copy may be automatically received from the second application responsive to the user modifying a content of the first file, where the first file copy is a version of the first file that is generated from the user modifying the content of the first file. Responsive to receiving the first file copy from the

(Continued)



US 10,289,607 B2

Page 2

second computer system, the first file copy may be automatically transferred via the first application to a third computer system associated with the user to replace an older version of the first file stored on the third computer system.

21 Claims, 3 Drawing Sheets**Related U.S. Application Data**

continuation of application No. 12/267,852, filed on Nov. 10, 2008, now Pat. No. 9,143,561.

(60) Provisional application No. 60/986,896, filed on Nov. 9, 2007.

(51) Int. Cl.

G06F 15/16 (2006.01)
G06F 16/13 (2019.01)
G06F 16/14 (2019.01)
G06F 16/176 (2019.01)
G06F 16/178 (2019.01)
H04L 29/08 (2006.01)

(52) U.S. Cl.

CPC **G06F 16/13** (2019.01); **G06F 16/14** (2019.01); **G06F 16/176** (2019.01); **G06F 16/178** (2019.01); **H04L 67/1095** (2013.01)

(58) Field of Classification Search

CPC .. G06F 17/30362; G06F 16/122; G06F 16/10; G06F 16/16; G06F 16/13 14; G06F 16/128; G06F 16/176; G06F 16/178
 See application file for complete search history.

(56)**References Cited**

U.S. PATENT DOCUMENTS

5,806,078 A 9/1998 Hug
 5,909,581 A 6/1999 Park
 5,920,867 A * 7/1999 Van Huben G06F 17/30362
 6,026,414 A 2/2000 Anglin
 6,154,817 A 11/2000 Mohan et al.
 6,260,069 B1 7/2001 Anglin
 6,449,624 B1 9/2002 Hammack et al.
 6,463,463 B1 10/2002 Godfrey et al.
 6,504,994 B2 1/2003 Kawamura et al.
 6,505,200 B1 1/2003 Ims et al.
 6,526,418 B1 * 2/2003 Midgley G06F 11/1451
 707/640
 6,606,646 B2 8/2003 Feigenbaum
 6,611,849 B1 8/2003 Raff et al.
 6,671,700 B1 12/2003 Creemer et al.
 6,708,221 B1 3/2004 Mendez et al.
 6,757,696 B2 6/2004 Multer et al.
 6,757,893 B1 * 6/2004 Haikin G06F 8/71
 717/170
 6,760,759 B1 7/2004 Chan
 6,810,405 B1 10/2004 Larue et al.
 6,829,622 B2 12/2004 Beyda
 6,874,037 B1 3/2005 Abram et al.
 6,931,454 B2 8/2005 Deshpande et al.
 6,990,522 B2 1/2006 Wu
 7,024,428 B1 4/2006 Huang et al.
 7,054,594 B2 5/2006 Bloch et al.
 7,065,658 B1 6/2006 Baraban et al.
 7,089,307 B2 8/2006 Zintel et al.
 7,136,934 B2 11/2006 Carter et al.
 7,155,488 B1 12/2006 Lunsford et al.
 7,224,973 B2 5/2007 Tsutazawa et al.
 7,243,163 B1 7/2007 Friend et al.
 7,260,646 B1 8/2007 Stefanik et al.
 7,269,433 B2 9/2007 Vargas et al.

7,290,244 B2 10/2007 Peck et al.
 7,325,038 B1 * 1/2008 Wang G06F 9/543
 707/999.01
 7,340,534 B2 3/2008 Cameron et al.
 7,398,327 B2 7/2008 Lee
 7,415,588 B2 8/2008 Hong et al.
 7,415,615 B2 8/2008 Skygebjær
 7,457,631 B2 11/2008 Yach et al.
 7,467,353 B2 12/2008 Kurlander et al.
 7,483,925 B2 1/2009 Koskimies et al.
 7,526,575 B2 4/2009 Rabbers et al.
 7,574,711 B2 8/2009 Zondervan et al.
 7,584,186 B2 9/2009 Chen et al.
 7,587,446 B1 9/2009 Onyon et al.
 7,613,773 B2 11/2009 Watt
 7,639,116 B2 12/2009 Saunders
 7,657,271 B2 2/2010 Kim
 7,680,885 B2 3/2010 Schausser et al.
 7,752,166 B2 7/2010 Quinlan et al.
 7,761,414 B2 7/2010 Freedman
 7,788,303 B2 8/2010 Mikesell et al.
 7,895,334 B1 2/2011 Tu et al.
 7,987,420 B1 7/2011 Kloba et al.
 8,009,966 B2 8/2011 Bloom et al.
 8,112,549 B2 2/2012 Srinivasan et al.
 8,244,288 B2 8/2012 Chipchase
 8,321,534 B1 11/2012 Roskind et al.
 8,370,423 B2 2/2013 Ozzie et al.
 8,386,558 B2 2/2013 Schleifer et al.
 8,565,729 B2 10/2013 Moseler et al.
 8,595,182 B1 * 11/2013 Nelson H04L 67/1095
 707/602
 9,143,561 B2 * 9/2015 Manzano G06F 17/30174
 10,067,942 B2 * 9/2018 Manzano G06F 17/30082
 2002/0026478 A1 2/2002 Rodgers et al.
 2002/0035697 A1 3/2002 McCurdy
 2002/0055942 A1 * 5/2002 Reynolds G06F 21/64
 2002/0087588 A1 7/2002 McBride et al.
 2002/0143795 A1 * 10/2002 Fletcher G06F 8/61
 2003/0028514 A1 2/2003 Lord
 2003/0028542 A1 2/2003 Muttik et al.
 2003/0038842 A1 2/2003 Peck et al.
 2003/0078946 A1 4/2003 Costello
 2003/0125057 A1 7/2003 Pesola
 2003/0135565 A1 7/2003 Estrada
 2003/0233241 A1 * 12/2003 Marsh G06F 16/48
 725/14
 2004/0049345 A1 3/2004 McDonough et al.
 2004/0093361 A1 5/2004 Therrien et al.
 2004/0107225 A1 6/2004 Rudoff
 2004/0133629 A1 7/2004 Reynolds
 2004/0158817 A1 8/2004 Okachi
 2004/0172424 A1 9/2004 Edelstein et al.
 2005/0027757 A1 * 2/2005 Kiessig G06F 16/10
 2005/0091316 A1 4/2005 Ponce et al.
 2005/0097225 A1 5/2005 Glatt et al.
 2005/0165722 A1 * 7/2005 Cannon G06F 11/1451
 2005/0192966 A1 * 9/2005 Hilbert H04L 51/22
 2005/0220080 A1 10/2005 Ronkainen et al.
 2006/0004698 A1 * 1/2006 Pyhalammii G06F 16/44
 2006/0010150 A1 1/2006 Shaath et al.
 2006/0058907 A1 3/2006 Suderman
 2006/0074985 A1 4/2006 Wolfish
 2006/0129627 A1 6/2006 Phillips
 2006/0136446 A1 * 6/2006 Hughes G06F 16/10
 2006/0143129 A1 6/2006 Holm
 2006/0168118 A1 7/2006 Godlin
 2006/0189348 A1 8/2006 Montulli et al.
 2007/0014314 A1 1/2007 O'Neil
 2007/0016629 A1 1/2007 Reinsch
 2007/0022158 A1 * 1/2007 Vasa H04M 1/72522
 709/204
 2007/0027936 A1 2/2007 Stakutis
 2007/0100913 A1 5/2007 Sumner et al.
 2007/0180084 A1 * 8/2007 Mohanty G06F 11/1451
 709/223
 2007/0191057 A1 8/2007 Kamada
 2007/0238440 A1 10/2007 Sengupta et al.
 2008/0005114 A1 1/2008 Li

US 10,289,607 B2

Page 3

(56)

References Cited

U.S. PATENT DOCUMENTS

2008/0005280	A1	1/2008	Adams	
2008/0086494	A1	4/2008	Heller et al.	
2008/0168526	A1	7/2008	Robbin et al.	
2008/0288578	A1	11/2008	Silfverberg	
2009/0013009	A1	1/2009	Nakayama	
2009/0037718	A1*	2/2009	Ganesh	G06F 9/4416 713/2
2009/0063711	A1	3/2009	Finkelstein	
2009/0282050	A1	11/2009	Thomas et al.	
2012/0192064	A1*	7/2012	Antebi	G06F 17/2288 715/255
2013/0226871	A1	8/2013	Sarnowski	
2013/0226872	A1*	8/2013	Barefoot	G06F 17/30575 707/638
2014/0053227	A1*	2/2014	Ruppin	G06F 21/10 726/1

OTHER PUBLICATIONS

FolderShare; Your Files Anywhere; <<https://web.archive.org/web/20030808183932/http://www.foldershare.com:80/>>; Aug. 8, 2003.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20040814015727/http://www.foldershare.com:80/>>; Aug. 14, 2004.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20040820052105/http://www.foldershare.com:80/>>; Aug. 20, 2004.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20041211020957/http://foldershare.com:80/>>; Dec. 11, 2004.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20041217041726/http://foldershare.com:80/>>; Dec. 17, 2004.

FolderShare; Your Smart File Transfer Solution; <<https://web.archive.org/web/20031220151508/http://www.foldershare.com:80/>>; Dec. 20, 2003.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20041230211050/http://www.foldershare.com:80/>>; Dec. 30, 2004.

FolderShare—Secure Remote Access VPN Solution; Document Management & Real-time File Mirroring Solution; <<https://web.archive.org/web/20040701113739/http://foldershare.com:80/>>; Jul. 1, 2004.

FolderShare—Secure Remote Access VPN Solution; Document Management & Real-time File Mirroring Solution; <<https://web.archive.org/web/20040711062548/http://www.foldershare.com:80/>>; Jul. 11, 2004.

FolderShare; Your Files Anywhere; <<https://web.archive.org/web/20030722054342/http://foldershare.com:80/>>; Jul. 22, 2003.

FolderShare—Secure Remote Access VPN Solution; Document Management & Real-time File Mirroring Solution; <<https://web.archive.org/web/20040730030655/http://www.foldershare.com:80/>>; Jul. 30, 2004.

FolderShare—Secure Remote Access VPN Solution; Document Management & Real-time File Mirroring Solution; <<https://web.archive.org/web/20040603205113/http://www.foldershare.com:80/>>; Jun. 3, 2004.

FolderShare—Secure Remote Access VPN Solution; Document Management & Real-time File Mirroring Solution; <<https://web.archive.org/web/20040613161906/http://www.foldershare.com:80/>>; Jun. 13, 2004.

FolderShare—Secure Remote Access VPN Solution; Document Management & Real-time File Mirroring Solution; <<https://web.archive.org/web/20040629075057/http://www.foldershare.com:80/>>; Jun. 29, 2004.

FolderShare—Secure Remote Access VPN Solution; Your Smart File Transfer & Real-time File Mirroring Solution; <<https://web.archive.org/web/20040316235151/http://foldershare.com:80/>>; Mar. 16, 2004.

FolderShare—Secure Remote Access VPN Solution; Your Smart File Transfer & Real-time File Mirroring Solution; <<https://web.archive.org/web/20040325034239/http://www.foldershare.com:80/>>; Mar. 25, 2004.

FolderShare—Secure Remote Access VPN Solution; Document Management & Real-time File Mirroring Solution; <<https://web.archive.org/web/20040512211417/http://www.foldershare.com:80/>>; May 12, 2004.

FolderShare; Your Files Anywhere; <<https://web.archive.org/web/20030531180252/http://www.foldershare.com:80/>>; May 31, 2003.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20041104031510/http://www.foldershare.com:80/>>; Nov. 4, 2004.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20041117092357/http://www.foldershare.com:80/>>; Nov. 17, 2004.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20041123085254/http://www.foldershare.com:80/>>; Nov. 23, 2004.

FolderShare; Your Files Anywhere; <<https://web.archive.org/web/20031128143634/http://foldershare.com:80/>>; Nov. 28, 2003.

FolderShare; Your Files Anywhere; <<https://web.archive.org/web/20031001071631/http://foldershare.com:80/>>; Oct. 1, 2003.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20041012083127/http://www.foldershare.com:80/>>; Oct. 12, 2004.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20041029085820/http://www.foldershare.com:80/>>; Oct. 29, 2004.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20040901034646/http://www.foldershare.com:80/>>; Sep. 1, 2004.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20040909075254/http://www.foldershare.com:80/>>; Sep. 9, 2004.

FolderShare; Your Files Anywhere; <<https://web.archive.org/web/20030920051943/http://www.foldershare.com:80/>>; Sep. 20, 2003.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20040924032146/http://www.foldershare.com:80/>>; Sep. 24, 2004.

Marshall, M., “The Y Combinator List,” Venture Beat, Aug. 2007, Retrieved from the Internet: URL: <<https://venturebeat.com/2007/08/16/the-y-combinator-list/>>, 4 pages.

Jarvis, A., “Dropbox pitch deck to raise seed capital investment,” Medium, Mar. 2018, Retrieved from the Internet: URL: <<https://medium.com/@adjblog/dropbox-pitch-deck-to-raise-seed-capital-investment-6a6cd6517e56>>, 12 pages.

* cited by examiner

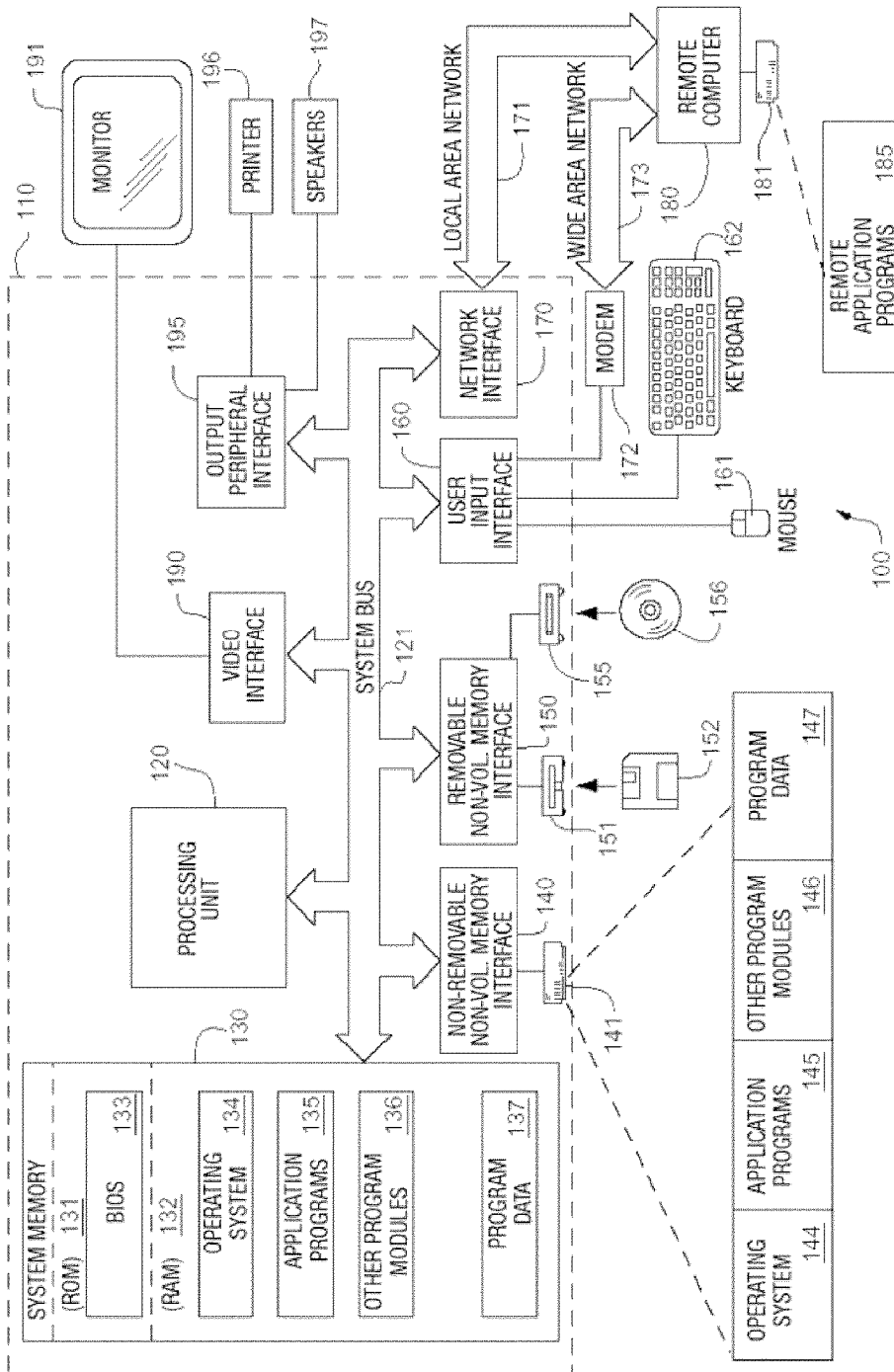


FIG. 1

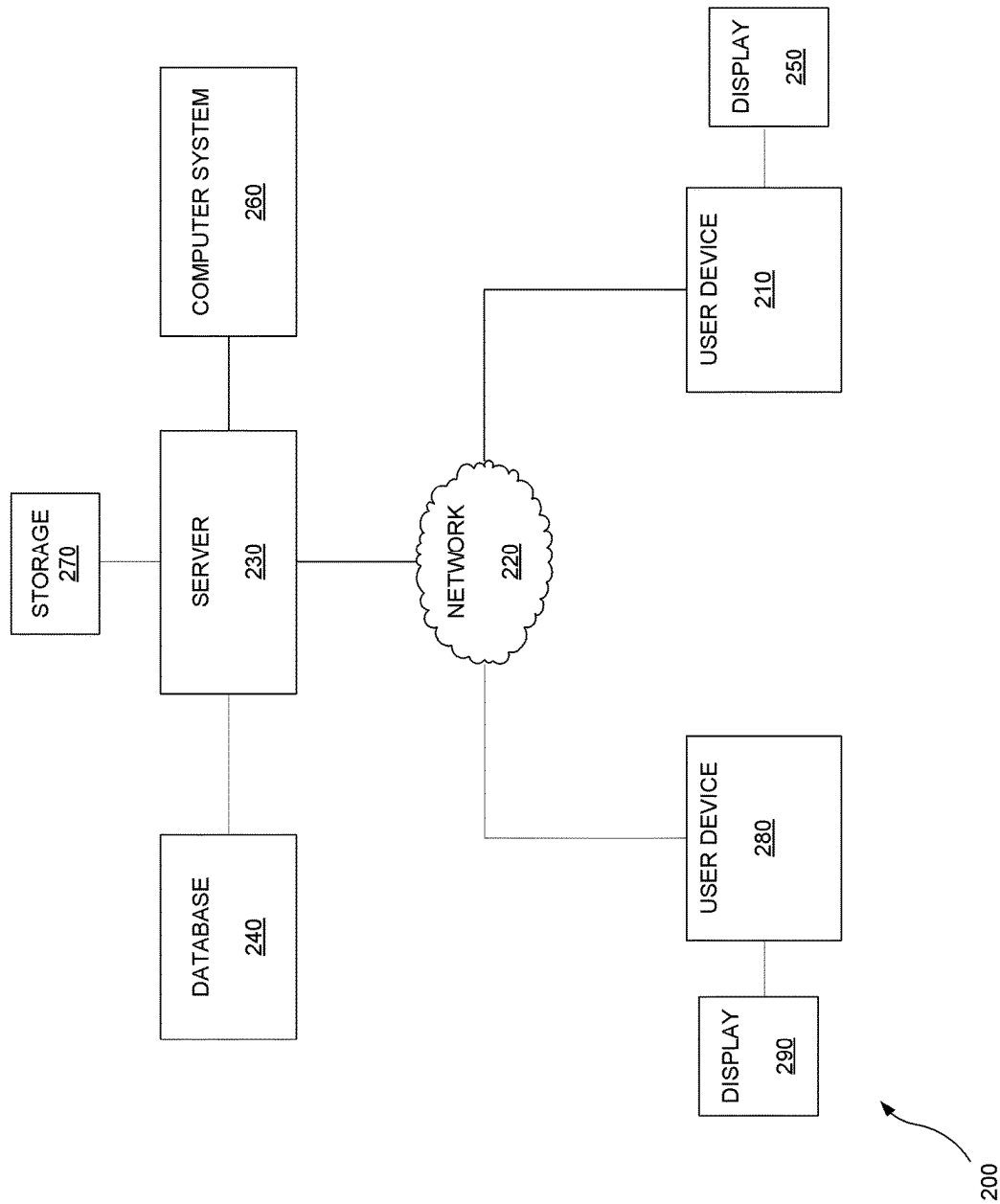


FIG. 2

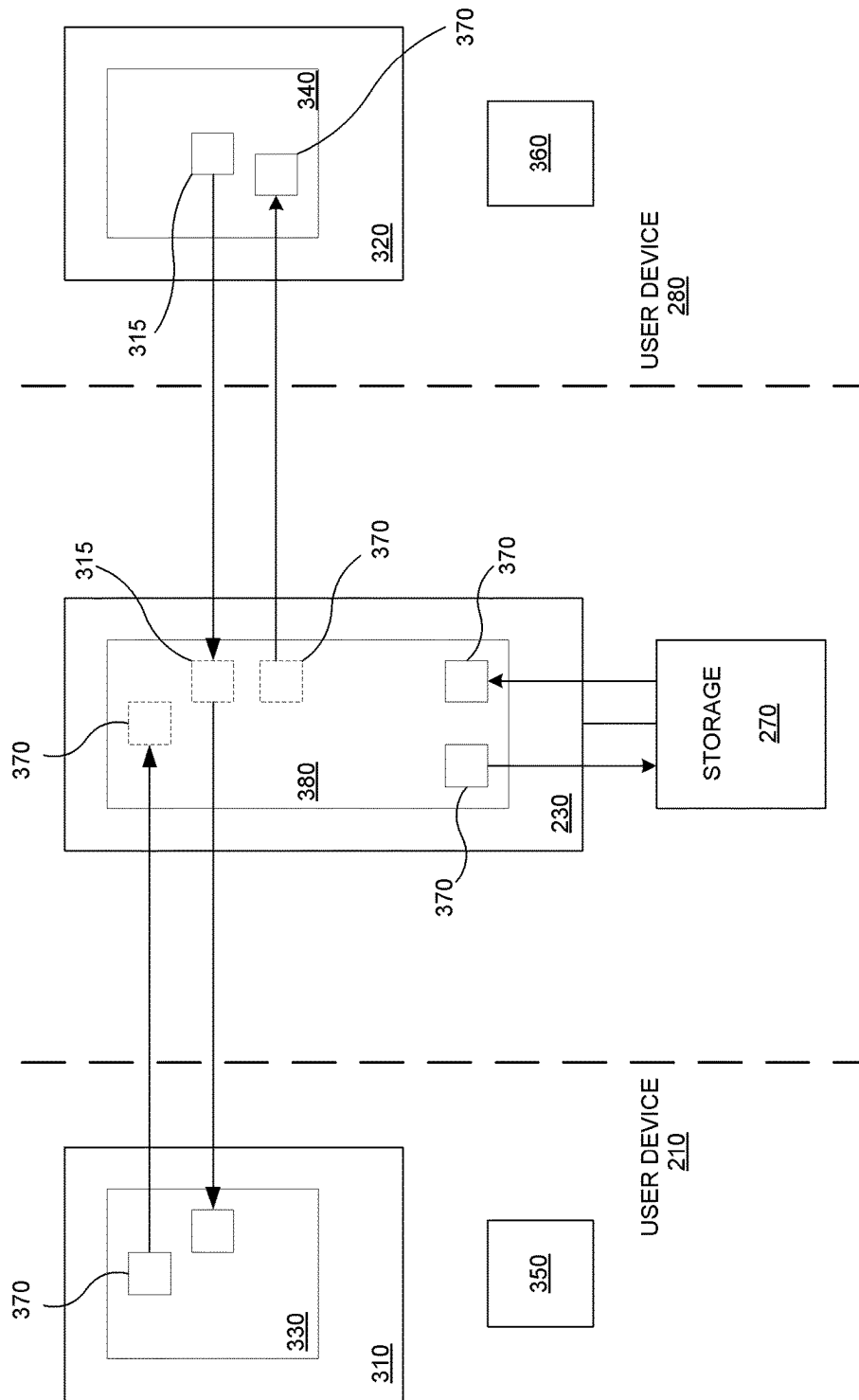


FIG. 3

US 10,289,607 B2

1

ARCHITECTURE FOR MANAGEMENT OF DIGITAL FILES ACROSS DISTRIBUTED NETWORK

CROSS-REFERENCE TO RELATED APPLICATIONS

This application is a continuation of U.S. patent application Ser. No. 14/860,289, filed Sep. 21, 2015, which is a continuation of U.S. patent application Ser. No. 12/267,852, filed Nov. 10, 2008, which claims priority to U.S. Provisional Application No. 60/986,896 entitled “ARCHITECTURE FOR MANAGEMENT OF DIGITAL FILES ACROSS DISTRIBUTED NETWORK” and filed Nov. 9, 2007, the contents of which are hereby incorporated by reference in their entirety.

FIELD OF THE INVENTION

This invention relates generally to computer-implemented processes and, more specifically, to sharing of electronic files among computer systems.

BACKGROUND OF THE INVENTION

Users of modern computing systems are increasingly finding themselves in constantly-connected, high-speed networked environments. The Web continues to be a killer application, second only to email, on the Internet. Further, customers are increasingly using more than one computing device; a customer may have a desktop computer at home, one at work, and a constantly connected “smart phone”. Due to the confluence of these two trends, file management across these devices has become a problem.

Although modern devices are easily connected, they do not provide the customer a seamless environment; the customer must manually handle many aspects of that connection. With regards to file management, customers must manually move files between their devices using some protocol like email, ftp, or by posting them on the Web. These practices lead to problems that include:

The proliferation of redundant file copies. This proliferation creates a confusing environment where the customer is unclear where the “official” or newest version of a file exists.

The creation of an error-prone environment. Some documents, such as those associated with word processing and desktop publishing, externally reference other files. Copying such a document can break these references causing errors that the customer has to handle manually. An example of such a document is a desktop publishing document that contains a reference to an image. If that image file is not transferred along with the desktop publishing file, the image will appear as a broken link.

Unnecessary complexity. Because devices tend to have their own filing system, customers must manage a different filing model on each of his devices. For example, instead of having a single “Movies” folder, he may have to deal with many “Movies” folders, which may be in different locations on each of his devices. Each device may also have its own security model, further complicating the matter.

That a customer has to manually move files around to ensure their accessibility on his devices is unnecessary, and is an indicator of a lack of customer-focused design in modern file systems. File systems in use today are direct

2

offspring of systems used when graphical customer interfaces were nonexistent. Modern file system customer interfaces, such as Windows® Explorer and Mac OS X’s Finder are just now starting to provide experiences that are more in line to a customer’s workflow. Whereas, before, these interfaces were concerned with representing files with abstracted icons, the file’s actual contents are becoming paramount in how files are organized and presented.

Problems still exist with how these newer customer interfaces are implemented. They are not completely integrated with applications, suffer from performance problems, and do not generally work well outside of a device’s local file system.

There are several solutions to this problem that are in one way or another inadequate to the task:

Remote Desktop software allows a customer to remotely “see” his desktop. Remote desktop software screen-scrapes a remote machine’s screen (a “server”) and displays it on a screen local to the customer (a “client”). Remote desktop gives a customer access to not only his files, but also to his applications. However, this approach requires that the host machine be turned on and connected to the internet at all times. Consequently, this approach would not be appropriate for mobile hosts such as laptops. Remote desktop does not use the resources of a local machine. For full accessibility, the customer would have to keep all files and application on the host machine as any files stored on a client are not guaranteed to be accessible.

Distributed File Systems, like remote desktop software, place data on an always-connected host machine. Unlike remote desktop software, the host machine is not one on which the customer performs computing tasks. The host machine is used as a storage mechanism, and any computation performed on that machine serves to support its use as such. Distributed file systems generally provide the right functionality for customers to share files between their devices. However, distributed file systems are usually deployed as a shared resource; that is, other customers have access to it. Because of this sharing, a customer’s files may be buried deep in a filing structure, and it may not always be immediately evident to customers what kind of access they have to a particular file. Further, to use a distributed file system, the customer must always be connected to it. Files stored on a distributed file system are generally inaccessible if the customer’s machine is not connected to it, unless the customer has copied or moved the files to his machine’s local hard drive. However, doing so immediately creates the problem of having two filing systems for the same file, creating a mental burden on the customer.

Additionally, accessing a file located on a distributed file system tends to be slower than accessing files on the local hard drive. Modern applications are usually written to assume that the files they access are located locally, and thus are not optimized to access remote files. When these applications are used with remote files, they can lose performance by an order of magnitude. This problem can be fixed by automatically caching often-used files on the local file system, and only synchronizing them when they have been changed. However, this separate synchronization step introduces another problem: because the synchronization process can be lengthy, the customer is never entirely sure if the file he is remotely accessing is the latest version of the file, versus an earlier one that has been marked to be updated. Further, the directory may not reflect the existence of the file at all until synchronization finishes.

FTP is similar to a distributed file system with regards to files being hosted on a remote server. However FTP gener-

US 10,289,607 B2

3

ally does manifest as a “disk drive” on the customer’s desktop; the customer must use special FTP client software to access an FTP server. It shares the same problem as distributed file systems, with the additional problem of weak integration with applications. Applications can generally write and read files directly to and from a distributed file system. This is not the case with FTP, as the customer has to manually use the client software to perform these operations as a separate task.

Email was originally invented for messaging. From the beginning, the model it employs to make files accessible remotely is necessarily inefficient. Email’s model for making files accessible is in the form of an email “attachment”. Attachments are so named because they piggy-back on a message sent from one customer to another. A customer can make a file remotely available using email by attaching the file to an email and sending it to himself. He can then retrieve the file from a remote location by accessing the message on the email server. Email used in this way is even worse than FTP as the process is even more manual: a customer must find the message containing the file before he can even access it. Further, the location in which the attachment lives is read only. If the customer, for example, were to open the file, change it, then save it back out, the results would be ambiguous to the user because the email application, not the user, specified its location. Usually, the saved file would end up buried in an email file cache in an undisclosed area of the file system.

Flash Drives and External Disk Drives, although seemingly the most “primitive” way to ensure file availability, avoid all the problems related to network latency. However, these devices must be physically connected to the computer on which the files will be accessed. These restrictions preclude the customer from employing several effective work-flows including: using more than one computer to complete a single task (the files can only be accessed on one computer) and setting up an automated backup (the computer running the backup can’t guarantee that the storage device will be connected come backup time). Further, to ensure full availability of the files, the customer must carry the device with them at all times, and must follow the associated protocols for mounting and dismounting the device.

Other problems with the prior art not described above can also be overcome using the teachings of embodiments of the present invention, as would be readily apparent to one of ordinary skill in the art after reading this disclosure.

SUMMARY OF THE INVENTION

In certain embodiments, automatic modification-triggered transfer of a file among two or more computer systems associated with a user. In some embodiments, a copy of a first file may be received, via a first application at a first computer system, from a second application at a second computer system associated with a user. The first file copy may be automatically received from the second application responsive to the user modifying a content of the first file, where the first file copy is a version of the first file that is generated from the user modifying the content of the first file. Responsive to receiving the first file copy from the second computer system, the first file copy may be automatically transferred via the first application to a third computer system associated with the user to replace an older version of the first file stored on the third computer system.

4

BRIEF DESCRIPTION OF THE DRAWING

Preferred and alternative embodiments of the present invention are described in detail below with reference to the following drawings.

FIG. 1 is a schematic view of an exemplary operating environment in which an embodiment of the invention can be implemented;

FIG. 2 is a functional block diagram of an exemplary operating environment in which an embodiment of the invention can be implemented; and

FIG. 3 is a functional block diagram illustrating file sharing and/or synchronization according to an embodiment of the invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

An embodiment of the invention leverages remote programming concepts by utilizing processes called mobile agents (sometimes referred to as mobile objects or agent objects). Generally speaking, these concepts provide the ability for an object (the mobile agent object) existing on a first (“host”) computer system to transplant itself to a second (“remote host”) computer system while preserving its current execution state. The operation of a mobile agent object is described briefly below.

The instructions of the mobile agent object, its preserved execution state, and other objects owned by the mobile agent object are packaged, or “encoded,” to generate a string of data that is configured so that the string of data can be transported by all standard means of communication over a computer network. Once transported to the remote host, the string of data is decoded to generate a computer process, still called the mobile agent object, within the remote host system. The decoded mobile agent object includes those objects encoded as described above and remains in its preserved execution state. The remote host computer system resumes execution of the mobile agent object which is now operating in the remote host environment.

While now operating in the new environment, the instructions of the mobile agent object are executed by the remote host to perform operations of any complexity, including defining, creating, and manipulating data objects and interacting with other remote host computer objects.

File transfer and/or synchronization, according to an embodiment, may be accomplished using some or all of the concepts described in commonly owned U.S. patent application Ser. No. 11/739,083, entitled “Electronic File Sharing,” the entirety of which is incorporated by reference as if fully set forth herein.

FIG. 1 illustrates an example of a suitable computing system environment **100** in which one or more embodiments of the invention may be implemented. The computing system environment **100** is only one example of a suitable computing environment and is not intended to suggest any limitation as to the scope of use or functionality of the invention. Neither should the computing environment **100** be interpreted as having any dependency or requirement relating to any one or combination of components illustrated in the exemplary operating environment **100**.

Embodiments of the invention are operational with numerous other general purpose or special purpose computing system environments or configurations. Examples of well known computing systems, environments, and/or configurations that may be suitable for use with the invention include, but are not limited to, personal computers, server

US 10,289,607 B2

5

computers, hand-held or laptop devices, multiprocessor systems, microprocessor-based systems, set top boxes, programmable consumer electronics, network PCs, minicomputers, mainframe computers, distributed computing environments that include any of the above systems or devices, and the like.

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

With reference to FIG. 1, an exemplary system for implementing the invention includes a general purpose computing device in the form of a computer 110. Components of computer 110 may include, but are not limited to, a processing unit 120, a system memory 130, and a system bus 121 that couples various system components including the system memory to the processing unit 120. The system bus 121 may be any of several types of bus structures including a memory bus or memory controller, a peripheral bus, and a local bus using any of a variety of bus architectures. By way of example, and not limitation, such architectures include Industry Standard Architecture (ISA) bus, Micro Channel Architecture (MCA) bus, Enhanced ISA (EISA) bus, Video Electronics Standards Association (VESA) local bus, and Peripheral Component Interconnect (PCI) bus also known as Mezzanine bus.

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

6

The system memory 130 includes computer storage media in the form of volatile and/or nonvolatile memory such as read only memory (ROM) 131 and random access memory (RAM) 132. A basic input/output system (BIOS), containing the basic routines that help to transfer information between elements within computer 110, such as during start-up, is typically stored in ROM 131. RAM 132 typically contains data and/or program modules that are immediately accessible to and/or presently being operated on by processing unit 120. By way of example, and not limitation, FIG. 1 illustrates operating system 134, application programs 135, other program modules 136, and program data 137.

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

The drives and their associated computer storage media discussed above and illustrated in FIG. 1, provide storage of computer readable instructions, data structures, program modules and other data for the computer 110. In FIG. 1, for example, hard disk drive 141 is illustrated as storing operating system 144, application programs 145, other program modules 146, and program data 147. Note that these components can either be the same as or different from operating system 134, application programs 135, other program modules 136, and program data 137. Operating system 144, application programs 145, other program modules 146, and program data 147 are given different numbers here to illustrate that, at a minimum, they are different copies. A user may enter commands and information into the computer 20 through input devices such as a keyboard 162 and pointing device 161, commonly referred to as a mouse, trackball or touch pad. Other input devices (not shown) may include a microphone, joystick, game pad, satellite dish, scanner, or the like. These and other input devices are often connected to the processing unit 120 through a user input interface 160 that is coupled to the system bus, but may be connected by other interface and bus structures, such as a parallel port, game port or a universal serial bus (USB). A monitor 191 or other type of display device is also connected to the system bus 121 via an interface, such as a video interface 190. In addition to the monitor, computers may also include other peripheral output devices such as speakers 197 and printer 196, which may be connected through an output peripheral interface 190.

The computer 110 may operate in a networked environment using logical connections to one or more remote computers, such as a remote computer 180. The remote computer 180 may be a personal computer, a server, a router, a network PC, a peer device or other common network node, and typically includes many or all of the elements described above relative to the computer 110, although only a memory

US 10,289,607 B2

7

storage device **181** has been illustrated in FIG. **1**. The logical connections depicted in FIG. **1** include a local area network (LAN) **171** and a wide area network (WAN) **173**, but may also include other networks. Such networking environments are commonplace in offices, enterprise-wide computer networks, intranets and the Internet.

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

Referring now to FIG. **2**, an embodiment of the present invention can be described in the context of an exemplary computer network system **200** as illustrated. System **200** includes electronic user devices **210**, **280**, such as personal computers or workstations, that are linked via a communication medium, such as a network **220** (e.g., the Internet), to an electronic device or system, such as a server **230**. The server **230** may further be coupled, or otherwise have access, to a database **240**, electronic storage **270** and a computer system **260**. Although the embodiment illustrated in FIG. **2** includes one server **230** coupled to two user devices **210**, **280** via the network **220**, it should be recognized that embodiments of the invention may be implemented using two or more such user devices coupled to one or more such servers.

In an embodiment, each of the user devices **210**, **280** and server **230** may include all or fewer than all of the features associated with the computer **110** illustrated in and discussed with reference to FIG. **1**. User devices **210**, **280** include or are otherwise coupled to a computer screen or display **250**, **290**, respectively. User devices **210**, **280** can be used for various purposes including both network- and local-computing processes.

The user devices **210**, **280** are linked via the network **220** to server **230** so that computer programs, such as, for example, a browser or other applications, running on one or more of the user devices **210**, **280** can cooperate in two-way communication with server **230** and one or more applications running on server **230**. Server **230** may be coupled to database **240** and/or electronic storage **270** to retrieve information therefrom and to store information thereto. Additionally, the server **230** may be coupled to the computer system **260** in a manner allowing the server to delegate certain processing functions to the computer system.

Referring now to FIG. **3**, illustrated is functionality of an embodiment of the invention allowing a user (not shown) who owns or otherwise controls devices **210**, **280** to automatically maintain file synchronization between at least devices **210**, **280**, or any other user devices on which principles of the present invention are implemented. In an embodiment, an administrator (not shown) of the server **230** or other appropriate electronic device transfers a file-transfer and/or synchronization application to the user devices **210**, **280** for installation thereon. Once installed on the user devices **210**, **280**, the file-transfer application provides file-

8

transfer clients **310**, **320** executable by the user devices **210**, **280**, respectively. Each of the file-transfer clients **310**, **320** may, but need not, include a respective mobile-agent runtime environment **330**, **340**. The mobile-agent runtime environment **330**, **340** include portions of memory of the user devices **210**, **280** dedicated to allowing a mobile object the ability to perform operations that the mobile object is programmed to carry out. Also included in the file-transfer application are user interfaces **350**, **360** that are displayable on the displays **250**, **290**, respectively. In an embodiment, the interfaces **350**, **360** allow a user to view, access and/or organize files to be synched among the various user devices.

Generally, all files that the user desires to be synched or shared may at some point be uploaded by one or more of the user devices **210**, **280** and stored in storage **270**. Upon receiving the files to be synched, the server **230** can store such files in the storage **270** and/or transfer the files to one or more of the respective hard drives of the user devices **210**, **280**, thereby enabling each respective user device to access such files. In this manner, the server **230** is operable to treat each hard drive of the respective user devices **210**, **280** as a local document cache for files received by the server. Typically, the server **230** will store one or more of the received files to the storage **270** only if the destination user device is offline or otherwise temporarily not in communication with the server **230**. Upon resuming communication with the destination user device, the server **230** will transfer the temporarily stored files to the destination device.

In operation, according to an embodiment, the user may open and modify a file **370**, such as a word-processing document or other electronic file. Alternatively, the user may create a first instance of the file **370**. The user may have previously have associated, or may now associate, the file **370** with the transfer client **310**. Upon a predetermined and user-configurable triggering event, the transfer client **310** transfers the modified file **370**, or a copy of the modified file, to the server **230**. Such a triggering event may include, but be not limited to, the user saving the file, the elapsing of a predetermined amount of time during which the file has been opened, or the re-initiation of a communication session between the device **210** and the server **230**.

The file **370** is transferred to the server **230** on which is executing a synchronization application **380**, which may include a mobile-agent runtime environment. Through user configuration, the synch application **380** monitors a set of user devices to which the file **370** should be transferred to effect file synchronization. In the illustrated embodiment, this set of user devices includes the user device **280**. The synch application **380** polls the device **280** to determine whether the device **280** is in communication with the server **230**. If the device **280** is in communication with the server **230**, the synch application **380** transfers the file **370** to the device **280**, whereupon the transfer client **320** resident on the device **280** replaces the previous version of the file **370**, previously cached on the device **280**, with the latest version of the file **370** modified on the user device **210**. If the device **280** is not currently in communication with the server **230**, the synch application **380** may store the file **370** in the storage **270** until such time as communication between the device **280** and server **230** is reestablished. As illustrated in FIG. **3**, a similar reverse-direction synchronization process may be performed by the synch application **380** and transfer clients **310**, **320** with regard to a file **315** modified on device **280** and synchronized to device **210**.

In an embodiment, the user interfaces **350**, **360** may include a list of the customer's documents and related metadata, as well as any one-to-one or one-to-many rela-

tionships between the documents and metadata. An embodiment can always provide customers with an accurate “picture” of their document collection, regardless of whether their devices physically contain the documents. As alluded to earlier, a problem with distributed file systems and FTP is the latency between a file being put onto a file system and it showing up on a remote machine. To prevent this problem, an embodiment directory is decoupled from the movement of files. An embodiment’s directory update system updates at a higher priority than the documents to be synchronized. This feature ensures that when a customer browses or searches through his set of documents, they appear even if they have not yet been cached locally on the user device. An indicator signifying a document’s availability may be prominently displayed adjacent to the document’s representation so that customers are aware of the document’s availability.

An embodiment may include a stand-alone application that allows customers to find and manage documents associated with transfer clients **310, 320** by visualizing relationships between documents and their metadata. It allows customers to tag documents with any number of identifiers. Customers can relate both documents and tags with each other in any number of user-specified one-to-one and one-to-many relationships, and an embodiment provides a user interface to browse and search on these relationships. To mitigate the customers’ learning curve, an embodiment can implement relationships common to contemporary file systems, including a folder hierarchy. In addition to this, an embodiment provides direct support for methods that the customer uses to organize documents by manifesting them as user interface idioms. This is unlike conventional document filing systems which require the customer to work within a strict folder metaphor for organization.

Some alternate methods that an embodiment supports for organizing documents include:

Allow customers to organize their documents by application. Many times customers remember the application used to create a document instead of the document’s name or its location in a hierarchy.

Allow customers to organize their documents by most recent access. Customers are likely to access a document they’ve accessed in the near past. Usually, such documents are part of a task that the customer is actively working.

Allow customers to organize their documents by project or subproject.

Allow customers to organize their documents by people. Many times, especially in the context of a collaboration, a document is directly related to one or more people other than the customer.

Allow the customer to organize their document by process stage. Documents may represent one or more stages of a process. Customers need a method for organizing documents by process stage, and a mechanism for moving the document through a set of predefined stages.

Allow customers to organize their documents by any of the aforementioned methods concurrently. These organization methods are not mutually exclusive.

An embodiment presents an interface that allows a customer to locate one or more documents associated with the transfer clients **310, 320** and open such document into a separate software application. Since this interface is intended to be used from within the separate application, that application may need to know how to invoke such interface. Advantageously, this invocation behavior can be provided to the application using the application’s plug-in API.

An embodiment presents an interface that allows a customer to synchronize a currently opened document according to processes described elsewhere herein. This interface can be invoked within an application and can be made available to the application in the manner described above in connection with the application’s plug-in API.

Some files associated with the transfer clients **310, 320** are dependent on other files associated with the transfer clients **310, 320**. For example, a desktop publishing document may include images that are stored in files separate from the main document. Previous file-synching solutions treat these files as separate. Because of this, for example, a document synchronized from the device **210** to the device **280** may be opened by the user of the device **280** before the image files have been fully transferred to the device **280**. This causes the document to fail to open, or break, since the image files don’t exist or are incomplete. An embodiment prevents this by: (1) always ensuring the file catalog (e.g., the stand-alone application that allows customers to find and manage documents associated with transfer clients **310, 320**, as discussed above herein) is synchronized before any file data is synchronized, and (2) pausing any file access by any program until the file contents have been fully synchronized. In such an embodiment, if a user attempts, using a software program, to open a file whose related files haven’t yet finished transferring to the local (hard drive) cache, if that software attempts to open the related files, the software program is blocked by an embodiment until the requested files are downloaded and ready to access.

Other file sending and synchronizing software requires the user to upload their data to a storage device owned by the operator of the service. An embodiment treats storage as a participant in the synchronization process; this means that the user can choose the service or device where their files will be stored. The file transfer/synching is abstracted from the storage system allowing any storage to be used. An embodiment treats storage like any other synch target, such as a desktop computer, or a cell phone. As such, any device owned or otherwise controlled by the user and running a synch application, such as synch application **380**, as provided in an embodiment of the invention can perform the storage and/or synching functions described elsewhere herein. That is, the user device **280** or user device **210**, rather than the server **230**, may perform such functions.

While a preferred embodiment of the invention has been illustrated and described, as noted above, many changes can be made without departing from the spirit and scope of the invention. For example, as an alternative to the approach described with reference to FIG. **3**, wherein the transfer clients **310, 320** function to “push” modified or created files to the synch application **380**, the synch application **380** may instead function to periodically “pull” or otherwise actively retrieve such files from the transfer clients **310, 320**. Instead, the invention should be determined entirely by reference to the claims that follow.

What is claimed is:

1. A system comprising:

a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to:

receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being a version

US 10,289,607 B2

11

of the first file that is generated from the user modifying the content of the first file;
 receive, from the first client device, first metadata associated with the version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;
 determine that the server system is not in communication with a second client device associated with the user;
 store the copy of the first file on the server system;
 automatically transfer the first metadata to the second client device based on the first priority being greater than the second priority such that the first metadata is transferred to the second client device prior to the copy of the first file being transferred to the second client device; and
 automatically transfer, over a network, the copy of the first file to the second client device associated with the user to replace an older version of the first file stored on the second client device, responsive to (i) resuming communication with the second client device and (ii) receiving the copy of the first file from the first client device.

2. The system of claim 1, wherein the computer program instructions, when executed, cause the server system to:
 store the copy of the first file to a memory device associated with the server system, wherein the copy of the first file is stored on the memory device associated with the server system responsive to determining that the server system is not in communication with the second client device.

3. The system of claim 1, wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.

4. The system of claim 1, wherein availability of the version of the first file is presented at the second client device based on the first metadata.

5. The system of claim 1, wherein the server system is caused to:
 receive a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the second client device responsive to the user modifying a content of the second file stored on the second client device, the copy of the second file being a version of the second file that is generated from the user modifying the content of the second file;
 determine that the server system is in communication with the first client device associated with the user; and
 automatically transfer the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client device and (ii) receiving the copy of the second file from the second client device.

6. The system of claim 1, wherein the computer program instructions, when executed, cause the server system to:
 periodically perform a pull request, wherein the copy of the first file is automatically received from the first

12

client device responsive to (i) the pull request and (ii) the user modifying the content of the first file stored on the first client device.

7. The system of claim 1, wherein the copy of the first file is automatically received from the first client device responsive to (i) a push request of the first client device and (ii) the user modifying the content of the first file stored on the first client device.

8. The system of claim 1, wherein the copy of the first file is automatically received from a first application at the first client device, and wherein the first application comprises a runtime environment for one or more mobile-agent objects.

9. The system of claim 8, wherein the first application is configured to create a first mobile object, and wherein the first mobile object is configured to create a proxy object at the server system.

10. The system of claim 9, wherein the first mobile object is configured to provide the copy of the first file to the proxy object.

11. The system of claim 10, wherein the proxy object is configured to store the copy of the first file on a memory device associated with the server system.

12. A method being implemented by a server system comprising one or more processors executing computer program instructions that, when executed, perform the method, the method comprising:
 receiving, by the server system, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being a version of the first file that is generated from the user modifying the content of the first file;
 receiving, by the server system, from the first client device, first metadata associated with the version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;
 determining, by the server system, that the server system is not in communication with a second client device associated with the user;
 store the copy of the first file on the server system;
 automatically transferring, by the server system, the first metadata to the second client device based on the first priority being greater than the second priority such that the first metadata is transferred to the second client device prior to the copy of the first file being transferred to the second client device; and
 automatically transferring, by the server system, over a network, the copy of the first file to the second client device associated with the user to replace an older version of the first file stored on the second client device, responsive to (i) resuming communication with the second client device and (ii) receiving the copy of the first file from the first client device.

13. The method of claim 12, further comprising:
 storing, by the server system, the copy of the first file to a memory device associated with the server system, wherein the copy of the first file is stored on the memory device associated with the server system responsive to determining that the server system is not in communication with the second client device.

14. The method of claim 12, wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to meta-

13

data associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.

15. The method of claim 12, wherein the copy of the first file is automatically received from a first application at the first client device, wherein the first application comprises a runtime environment for one or more mobile-agent objects, and wherein the first application is configured to create a first mobile object, and the first mobile object is configured to create a proxy object at the server system and to provide the copy of the first file to the proxy object.

16. The method of claim 15, wherein the proxy object is configured to store the copy of the first file on a memory device associated with the server system.

17. A system comprising:

a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to:

receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being a version of the first file that is generated from the user modifying the content of the first file;

receive, from the first client device, first metadata associated with the version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file, wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration;

determine that the server system is not in communication with a second client device associated with the user;

store the copy of the first file on the server system; automatically transfer the first metadata to the second client device based on the first priority being greater than the second priority such that the first metadata is transferred to the second client device prior to the copy of the first file being transferred to the second client device; and

automatically transfer, over a network, the copy of the first file to the second client device associated with the user, responsive to (i) resuming communication with the second client device and (ii) receiving the copy of the first file from the first client device, wherein, responsive to transferring the copy of the first file to the second client device, an older version of the first file stored on the second client device is automatically caused to be replaced with the copy of the first file such that the copy of the first file is stored on the second client device in lieu of the older version of the first file.

18. The system of claim 17, wherein the copy of the first file is automatically received from a first application at the

14

first client device, wherein the first application comprises a runtime environment for one or more mobile-agent objects, and wherein the first application is configured to create a first mobile object, the first mobile object is configured to create a proxy object at the server system and to provide the copy of the first file to the proxy object, and the proxy object is configured to store the copy of the first file on a memory device associated with the server system.

19. One or more non-transitory machine-readable media storing instructions that, when executed by one or more processors of a server system, cause operations comprising:

receiving, by the server system, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being a version of the first file that is generated from the user modifying the content of the first file;

receiving, by the server system, from the first client device, first metadata associated with the version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;

determining, by the server system, that the server system is not in communication with a second client device associated with the user;

storing, by the server system, the copy of the first file on the server system;

automatically transferring, by the server system, the first metadata to the second client device based on the first priority being greater than the second priority such that the first metadata is transferred to the second client device prior to the copy of the first file being transferred to the second client device; and

automatically transferring, by the server system, over a network, the copy of the first file to the second client device associated with the user to replace an older version of the first file stored on the second client device, responsive to (i) resuming communication with the second client device and (ii) receiving the copy of the first file from the first client device.

20. The one or more non-transitory machine-readable media of claim 19, wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.

21. The one or more non-transitory machine-readable media of claim 19, wherein the copy of the first file is automatically received from a first application at the first client device, wherein the first application comprises a runtime environment for one or more mobile-agent objects, and wherein the first application is configured to create a first mobile object, the first mobile object is configured to create a proxy object at the server system and to provide the copy of the first file to the proxy object, and the proxy object is configured to store the copy of the first file on a memory device associated with the server system.

EXHIBIT 5



US010642787B1

(12) **United States Patent**
Manzano

(10) **Patent No.:** **US 10,642,787 B1**
(45) **Date of Patent:** ***May 5, 2020**

(54) **PRE-FILE-TRANSFER UPDATE BASED ON PRIORITIZED METADATA**

(71) Applicant: **TOPIA TECHNOLOGY, INC.**,
Tacoma, WA (US)

(72) Inventor: **Michael R. Manzano**, Seattle, WA (US)

(73) Assignee: **TOPIA TECHNOLOGY, INC.**,
Tacoma, WA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

This patent is subject to a terminal disclaimer.

(21) Appl. No.: **16/750,399**

(22) Filed: **Jan. 23, 2020**

Related U.S. Application Data

(63) Continuation of application No. 16/361,641, filed on Mar. 22, 2019, which is a continuation of application (Continued)

(51) **Int. Cl.**
G06F 16/00 (2019.01)
G06F 16/11 (2019.01)
(Continued)

(52) **U.S. Cl.**
CPC **G06F 16/122** (2019.01); **G06F 15/16** (2013.01); **G06F 16/00** (2019.01); **G06F 16/128** (2019.01);
(Continued)

(58) **Field of Classification Search**
CPC **G06F 16/178**; **G06F 16/27**; **G06F 16/128**; **G06F 16/182**; **G06F 16/1844**;
(Continued)

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,600,834 A 2/1997 Howard
5,806,078 A 9/1998 Hug
(Continued)

FOREIGN PATENT DOCUMENTS

EP 1 130 511 A2 9/2001
WO 98/56149 A1 12/1998
WO 2007/047302 A2 4/2007

OTHER PUBLICATIONS

FolderShare—Secure Remote Access VPN Solution; Document Management & Real-time File Mirroring Solution; < <https://web.archive.org/web/20040804020435/http://www.foldershare.com:80/>>; Aug. 4, 2004.

(Continued)

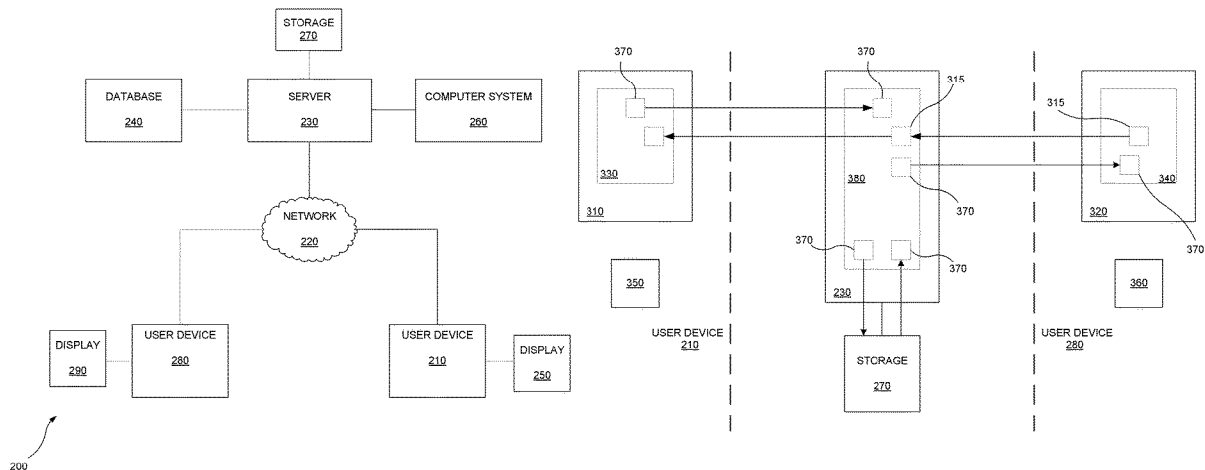
Primary Examiner — Srirama Channavajjala

(74) *Attorney, Agent, or Firm* — Pillsbury Winthrop Shaw Pittman LLP

(57) **ABSTRACT**

In some embodiments, responsive to a user modifying a content of the file at a first client device (associated with the user), a server system may automatically receive a copy of the file from the first client device, where the file copy may be an updated version of the file that is generated from the user modifying the content of the file. After receiving metadata associated with the updated version of the file from the first client device, the server system may automatically transfer the metadata to a second client device associated with the user such that, before the file copy is transferred to the second client device, the transfer of the metadata to the second client device causes a file representation of the file presented on a user interface of the second client device to be updated based on the metadata.

17 Claims, 3 Drawing Sheets



US 10,642,787 B1

Page 2

Related U.S. Application Data						
No. 16/017,348, filed on Jun. 25, 2018, now Pat. No. 10,289,607, which is a continuation of application No. 14/860,289, filed on Sep. 21, 2015, now Pat. No. 10,067,942, which is a continuation of application No. 12/267,852, filed on Nov. 10, 2008, now Pat. No. 9,143,561.			7,415,615 B2	8/2008	Skygebjær	
			7,457,631 B2	11/2008	Yach et al.	
			7,467,353 B2	12/2008	Kurlander et al.	
			7,483,925 B2	1/2009	Koskimies et al.	
			7,526,575 B2	4/2009	Rabbers et al.	
			7,529,778 B1 *	5/2009	Dewey	G06F 11/1451
			7,574,711 B2	8/2009	Zondervan et al.	
			7,584,186 B2	9/2009	Chen et al.	
			7,587,446 B1	9/2009	Onyon et al.	
			7,613,773 B2	11/2009	Watt	
			7,639,116 B2	12/2009	Saunders	
(60) Provisional application No. 60/986,896, filed on Nov. 9, 2007.			7,657,271 B2	2/2010	Kim	
			7,680,838 B1	3/2010	Shaw	
			7,680,885 B2	3/2010	Schauser et al.	
(51) Int. Cl.			7,707,165 B1 *	4/2010	Jiang	G06F 16/10 707/806
H04L 29/08 (2006.01)						
G06F 16/178 (2019.01)			7,752,166 B2	7/2010	Quinlan et al.	
G06F 16/176 (2019.01)			7,761,414 B2	7/2010	Freedman	
G06F 15/16 (2006.01)			7,788,303 B2	8/2010	Mikesell et al.	
G06F 16/14 (2019.01)			7,885,925 B1	2/2011	Strong et al.	
G06F 16/13 (2019.01)			7,895,334 B1	2/2011	Tu et al.	
(52) U.S. Cl.			7,987,420 B1	7/2011	Kloba et al.	
CPC	G06F 16/13 (2019.01); G06F 16/14 (2019.01); G06F 16/176 (2019.01); G06F 16/178 (2019.01); H04L 67/1095 (2013.01)		8,009,966 B2	8/2011	Bloom et al.	
			8,019,900 B1	9/2011	Sekar et al.	
			8,112,549 B2	2/2012	Srinivasan et al.	
			8,244,288 B2	8/2012	Chipchase	
			8,321,534 B1	11/2012	Roskind et al.	
(58) Field of Classification Search			8,370,423 B2	2/2013	Ozzie et al.	
CPC	G06F 16/176 ; G06F 16/10 ; G06F 16/1873 ; G06F 16/13 ; G06F 16/152 ; G06F 16/164		8,386,558 B2	2/2013	Schleifer et al.	
See application file for complete search history.			8,504,519 B1 *	8/2013	Sachs	G06F 8/71 707/616
			8,565,729 B2	10/2013	Moseler et al.	
			8,874,534 B2 *	10/2014	March	G06F 11/1435 707/695
(56) References Cited						
U.S. PATENT DOCUMENTS			9,143,561 B2 *	9/2015	Manzano	H04L 67/1095
			10,067,942 B2 *	9/2018	Manzano	H04L 67/1095
			10,289,607 B2 *	5/2019	Manzano	H04L 67/1095
5,909,581 A	6/1999	Park	2002/0026478 A1	2/2002	Rodgers et al.	
6,026,414 A	2/2000	Anglin	2002/0035697 A1	3/2002	McCurdy	
6,088,693 A	7/2000	Van Huben	2002/0087588 A1	7/2002	McBride et al.	
6,154,817 A	11/2000	Mohan et al.	2002/0194382 A1	12/2002	Kausik	
6,260,069 B1	7/2001	Anglin	2003/0028514 A1	2/2003	Lord	
6,449,624 B1	9/2002	Hammack et al.	2003/0028542 A1	2/2003	Muttik et al.	
6,463,463 B1	10/2002	Godfrey et al.	2003/0038842 A1	2/2003	Peck et al.	
6,504,994 B2	1/2003	Kawamura et al.	2003/0078946 A1	4/2003	Costello	
6,505,200 B1	1/2003	Ims et al.	2003/0115547 A1	6/2003	Ohwada	
6,606,646 B2	8/2003	Feigenbaum	2003/0125057 A1	7/2003	Pesola	
6,611,849 B1	8/2003	Raff et al.	2003/0135565 A1	7/2003	Estrada	
6,671,700 B1	12/2003	Creemer et al.	2004/0031027 A1 *	2/2004	Hiltgen	G06F 8/65 717/170
6,708,221 B1	3/2004	Mendez et al.				
6,757,696 B2	6/2004	Multer et al.	2004/0049345 A1	3/2004	McDonough et al.	
6,760,759 B1	7/2004	Chan	2004/0093361 A1	5/2004	Therrien et al.	
6,810,405 B1	10/2004	Larue et al.	2004/0107225 A1	6/2004	Rudoff	
6,826,626 B1	11/2004	McManus	2004/0133629 A1	7/2004	Reynolds	
6,829,622 B2	12/2004	Beyda	2004/0158817 A1	8/2004	Okachi	
6,874,037 B1	3/2005	Abram et al.	2004/0172424 A1	9/2004	Edelstein et al.	
6,931,454 B2	8/2005	Deshpande et al.	2005/0091316 A1	4/2005	Ponce et al.	
6,990,522 B2	1/2006	Wu	2005/0097225 A1	5/2005	Glatt et al.	
7,024,428 B1	4/2006	Huang et al.	2005/0210371 A1	9/2005	Pollock	
7,035,847 B2	4/2006	Brown et al.	2005/0220080 A1	10/2005	Ronkainen et al.	
7,051,364 B1	5/2006	Tackman	2005/0223047 A1	10/2005	Shah et al.	
7,054,594 B2	5/2006	Bloch et al.	2006/0010150 A1	1/2006	Shaath et al.	
7,058,667 B2	6/2006	Goldick	2006/0058907 A1	3/2006	Suderman	
7,065,658 B1	6/2006	Baraban et al.	2006/0074985 A1	4/2006	Wolfish	
7,089,307 B2	8/2006	Zintel et al.	2006/0101064 A1	5/2006	Strong et al.	
7,136,934 B2	11/2006	Carter et al.	2006/0129627 A1	6/2006	Phillips	
7,149,760 B1	12/2006	Breuer	2006/0143129 A1	6/2006	Holm	
7,155,488 B1	12/2006	Lunsford et al.	2006/0168118 A1	7/2006	Godlin	
7,162,501 B2	1/2007	Kupkova	2006/0189348 A1	8/2006	Montulli et al.	
7,224,973 B2	5/2007	Tsutazawa et al.	2007/0014314 A1	1/2007	O'Neil	
7,243,163 B1	7/2007	Friend et al.	2007/0016629 A1	1/2007	Reinsch	
7,260,646 B1	8/2007	Stefanik et al.	2007/0027936 A1	2/2007	Stakutis	
7,263,493 B1	8/2007	Provost	2007/0100913 A1	5/2007	Summer et al.	
7,269,433 B2	9/2007	Vargas et al.	2007/0180084 A1 *	8/2007	Mohanty	G06F 11/1451 709/223
7,290,244 B2	10/2007	Peck et al.				
7,325,038 B1	1/2008	Wang	2007/0191057 A1	8/2007	Kamada	
7,340,534 B2	3/2008	Cameron et al.	2007/0238440 A1	10/2007	Sengupta et al.	
7,398,327 B2	7/2008	Lee	2008/0005114 A1	1/2008	Li	
7,415,588 B2	8/2008	Hong et al.	2008/0005195 A1	1/2008	Li	

US 10,642,787 B1

Page 3

(56)

References Cited

U.S. PATENT DOCUMENTS

2008/0005280	A1	1/2008	Adams
2008/0086494	A1	4/2008	Heller et al.
2008/0168526	A1	7/2008	Robbin et al.
2008/0288578	A1	11/2008	Silverberg
2009/0013009	A1	1/2009	Nakayama
2009/0024922	A1	1/2009	Markowitz et al.
2009/0063711	A1	3/2009	Finkelstein
2009/0282050	A1	11/2009	Thomas et al.
2013/0226871	A1	8/2013	Sarnowski

OTHER PUBLICATIONS

FolderShare; Your Files Anywhere; <<https://web.archive.org/web/20030808183932/http://www.foldershare.com:80/>>; Aug. 8, 2003.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20040814015727/http://www.foldershare.com:80/>>; Aug. 14, 2004.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20040820052105/http://www.foldershare.com:80/>>; Aug. 20, 2004.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20041211020957/http://foldershare.com:80/>>; Dec. 11, 2004.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20041217041726/http://foldershare.com:80/>>; Dec. 17, 2004.

FolderShare; Your Smart File Transfer Solution; <<https://web.archive.org/web/20031220151508/http://www.foldershare.com:80/>>; Dec. 20, 2003.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20041230211050/http://www.foldershare.com:80/>>; Dec. 30, 2004.

FolderShare—Secure Remote Access VPN Solution; Document Management & Real-time File Mirroring Solution; <<https://web.archive.org/web/20040701113739/http://foldershare.com:80/>>; Jul. 1, 2004.

FolderShare—Secure Remote Access VPN Solution; Document Management & Real-time File Mirroring Solution; <<https://web.archive.org/web/20040711062548/http://www.foldershare.com:80/>>; Jul. 11, 2004.

FolderShare; Your Files Anywhere; <<https://web.archive.org/web/20030722054342/http://foldershare.com:80/>>; Jul. 22, 2003.

FolderShare—Secure Remote Access VPN Solution; Document Management & Real-time File Mirroring Solution; <<https://web.archive.org/web/20040730030655/http://www.foldershare.com:80/>>; Jul. 30, 2004.

FolderShare—Secure Remote Access VPN Solution; Document Management & Real-time File Mirroring Solution; <<https://web.archive.org/web/20040603205113/http://www.foldershare.com:80/>>; Jun. 3, 2004.

FolderShare—Secure Remote Access VPN Solution; Document Management & Real-time File Mirroring Solution; <<https://web.archive.org/web/20040613161906/http://www.foldershare.com:80/>>; Jun. 13, 2004.

FolderShare—Secure Remote Access VPN Solution; Document Management & Real-time File Mirroring Solution; <<https://web.archive.org/web/20040629075057/http://www.foldershare.com:80/>>; Jun. 29, 2004.

FolderShare—Secure Remote Access VPN Solution; Your Smart File Transfer & Real-time File Mirroring Solution; <<https://web.archive.org/web/20040316235151/http://foldershare.com:80/>>; Mar. 16, 2004.

FolderShare—Secure Remote Access VPN Solution; Your Smart File Transfer & Real-time File Mirroring Solution; <<https://web.archive.org/web/20040325034239/http://www.foldershare.com:80/>>; Mar. 25, 2004.

FolderShare—Secure Remote Access VPN Solution; Document Management & Real-time File Mirroring Solution; <<https://web.archive.org/web/20040512211417/http://www.foldershare.com:80/>>; May 12, 2004.

FolderShare; Your Files Anywhere; <<https://web.archive.org/web/20030531180252/http://www.foldershare.com:80/>>; May 31, 2003.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20041104031510/http://www.foldershare.com:80/>>; Nov. 4, 2004.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20041117092357/http://www.foldershare.com:80/>>; Nov. 17, 2004.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20041123085254/http://www.foldershare.com:80/>>; Nov. 23, 2004.

FolderShare; Your Files Anywhere; <<https://web.archive.org/web/20031128143634/http://foldershare.com:80/>>; Nov. 28, 2003.

FolderShare; Your Files Anywhere; <<https://web.archive.org/web/20031001071631/http://foldershare.com:80/>>; Oct. 1, 2003.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20041012083127/http://www.foldershare.com:80/>>; Oct. 12, 2004.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20041029085820/http://www.foldershare.com:80/>>; Oct. 29, 2004.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20040901034646/http://www.foldershare.com:80/>>; Sep. 1, 2004.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20040909075254/http://www.foldershare.com:80/>>; Sep. 9, 2004.

FolderShare; Your Files Anywhere; <<https://web.archive.org/web/20030920051943/http://www.foldershare.com:80/>>; Sep. 20, 2003.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20040924032146/http://www.foldershare.com:80/>>; Sep. 24, 2004.

Marshall, M., “The Y Combinator List,” Venture Beat, Aug. 2007, Retrieved from the Internet: URL: <<https://venturebeat.com/2007/08/16/the-y-combinator-list/>>, 4 pages.

Jarvis, A., “Dropbox pitch deck to raise seed capital investment,” Medium, Mar. 2018, Retrieved from the Internet: URL: <<https://medium.com/@adjblog/dropbox-pitch-deck-to-raise-seed-capital-investment-6a6cd6517e56>>, 12 pages.

* cited by examiner

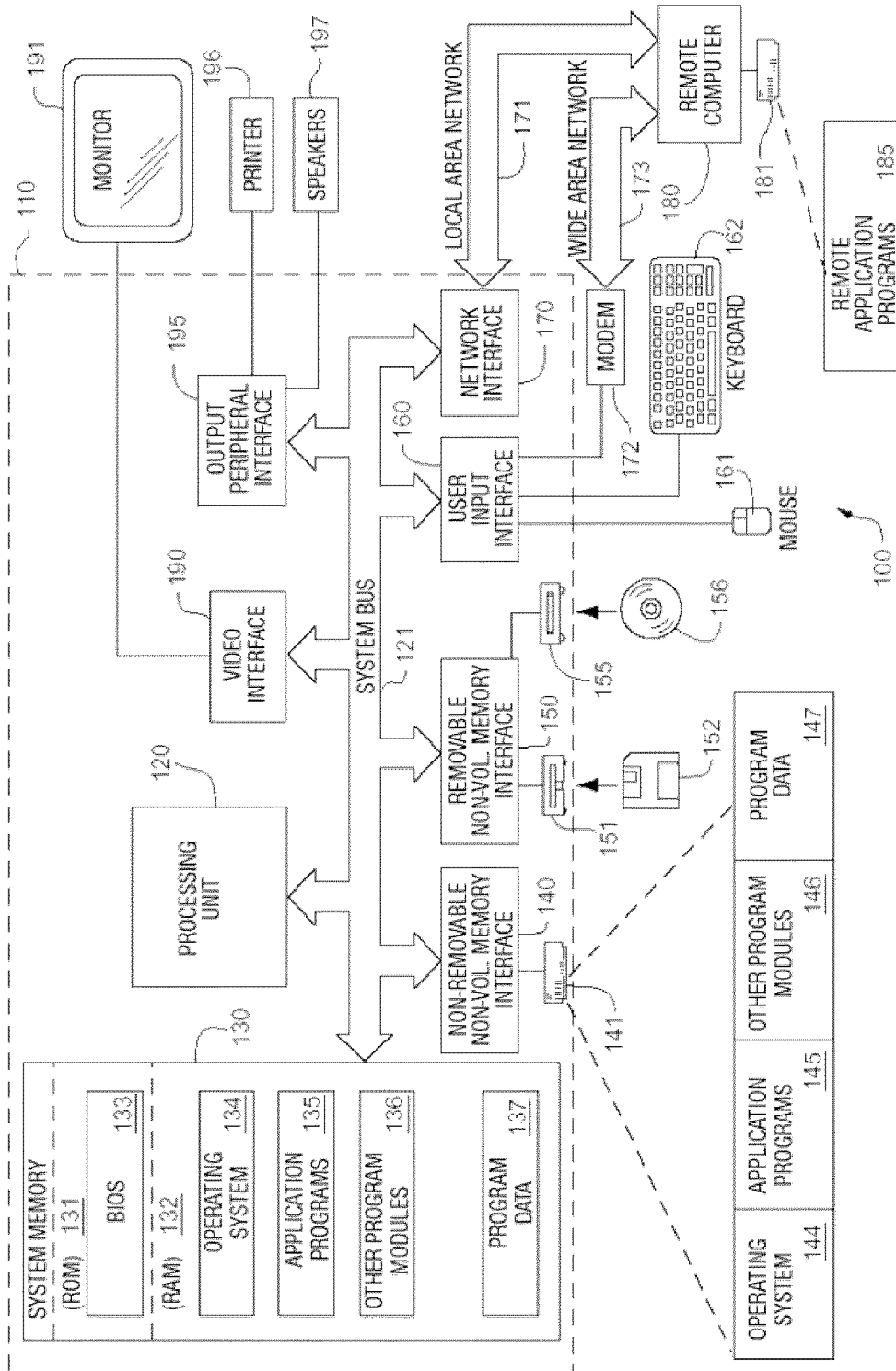


FIG. 1

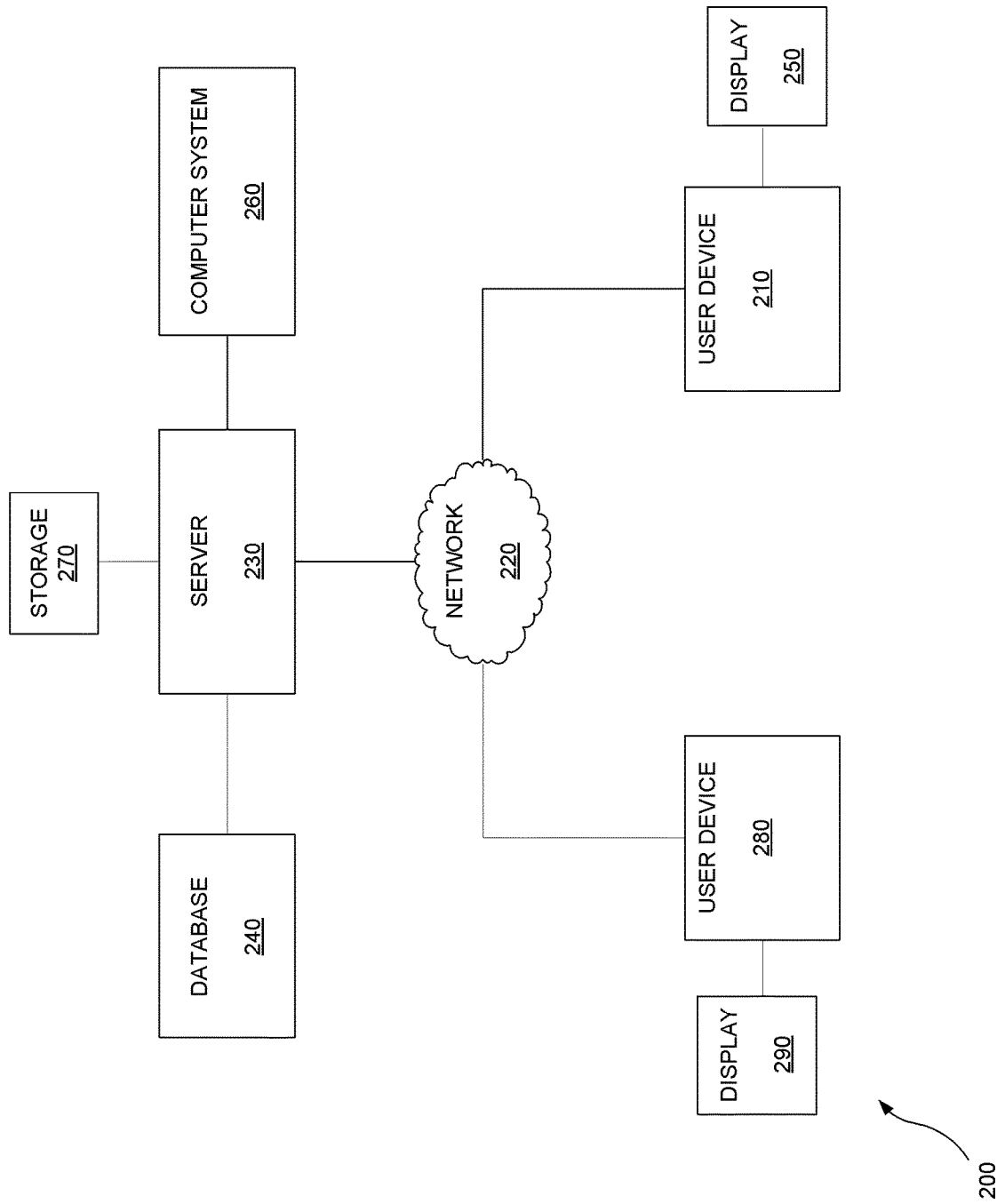


FIG. 2

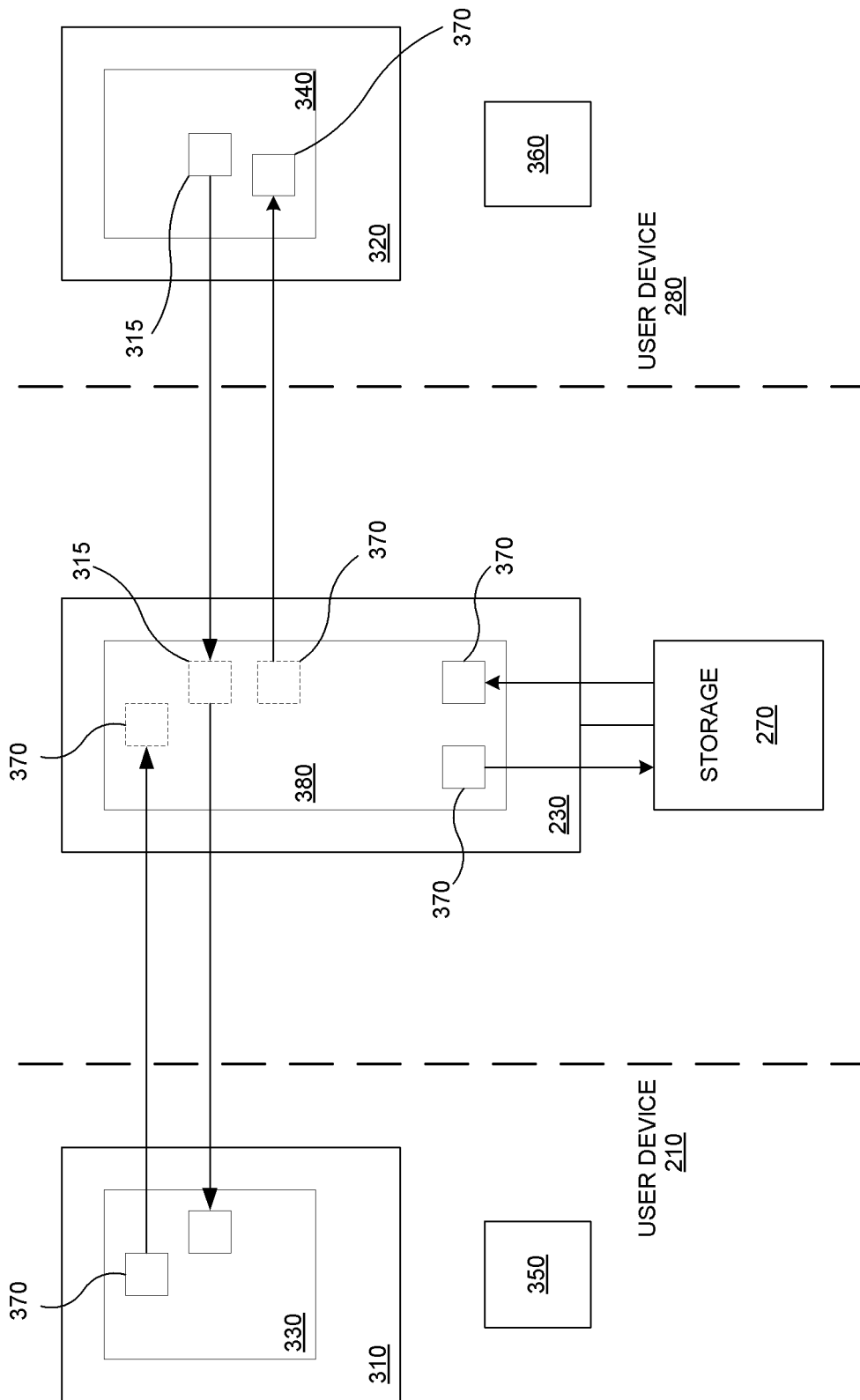


FIG. 3

US 10,642,787 B1

1

**PRE-FILE-TRANSFER UPDATE BASED ON
PRIORITIZED METADATA****CROSS-REFERENCE TO RELATED
APPLICATIONS**

This application is a continuation of U.S. patent application Ser. No. 16/361,641, filed Mar. 22, 2019, which is a continuation of U.S. patent application Ser. No. 16/017,348, filed Jun. 25, 2018, which is a continuation of U.S. patent application Ser. No. 14/860,289, filed Sep. 21, 2015, now U.S. Pat. No. 10,067,942, which is a continuation of U.S. patent application Ser. No. 12/267,852, filed Nov. 10, 2008, now U.S. Pat. No. 9,143,561, which claims priority to U.S. Provisional Application No. 60/986,896 entitled “ARCHITECTURE FOR MANAGEMENT OF DIGITAL FILES ACROSS DISTRIBUTED NETWORK” and filed Nov. 9, 2007, the contents of which are hereby incorporated by reference in their entirety.

FIELD OF THE INVENTION

This invention relates generally to computer-implemented processes and, more specifically, to sharing of electronic files among computer systems.

BACKGROUND OF THE INVENTION

Users of modern computing systems are increasingly finding themselves in constantly-connected, high-speed networked environments. The Web continues to be a killer application, second only to email, on the Internet. Further, customers are increasingly using more than one computing device; a customer may have a desktop computer at home, one at work, and a constantly connected “smart phone”. Due to the confluence of these two trends, file management across these devices has become a problem.

Although modern devices are easily connected, they do not provide the customer a seamless environment; the customer must manually handle many aspects of that connection. With regards to file management, customers must manually move files between their devices using some protocol like email, ftp, or by posting them on the Web. These practices lead to problems that include:

The proliferation of redundant file copies. This proliferation creates a confusing environment where the customer is unclear where the “official” or newest version of a file exists.

The creation of an error-prone environment. Some documents, such as those associated with word processing and desktop publishing, externally reference other files. Copying such a document can break these references causing errors that the customer has to handle manually. An example of such a document is a desktop publishing document that contains a reference to an image. If that image file is not transferred along with the desktop publishing file, the image will appear as a broken link.

Unnecessary complexity. Because devices tend to have their own filing system, customers must manage a different filing model on each of his devices. For example, instead of having a single “Movies” folder, he may have to deal with many “Movies” folders, which may be in different locations on each of his devices. Each device may also have its own security model, further complicating the matter.

2

That a customer has to manually move files around to ensure their accessibility on his devices is unnecessary, and is an indicator of a lack of customer-focused design in modern file systems. File systems in use today are direct offspring of systems used when graphical customer interfaces were nonexistent. Modern file system customer interfaces, such as Windows® Explorer and Mac OS X’s Finder are just now starting to provide experiences that are more in line to a customer’s workflow. Whereas, before, these interfaces were concerned with representing files with abstracted icons, the file’s actual contents are becoming paramount in how files are organized and presented.

Problems still exist with how these newer customer interfaces are implemented. They are not completely integrated with applications, suffer from performance problems, and do not generally work well outside of a device’s local file system.

There are several solutions to this problem that are in one way or another inadequate to the task:

Remote Desktop software allows a customer to remotely “see” his desktop. Remote desktop software screen-scrapes a remote machine’s screen (a “server”) and displays it on a screen local to the customer (a “client”). Remote desktop gives a customer access to not only his files, but also to his applications. However, this approach requires that the host machine be turned on and connected to the internet at all times. Consequently, this approach would not be appropriate for mobile hosts such as laptops. Remote desktop does not use the resources of a local machine. For full accessibility, the customer would have to keep all files and application on the host machine as any files stored on a client are not guaranteed to be accessible.

Distributed File Systems, like remote desktop software, place data on an always-connected host machine. Unlike remote desktop software, the host machine is not one on which the customer performs computing tasks. The host machine is used as a storage mechanism, and any computation performed on that machine serves to support its use as such. Distributed file systems generally provide the right functionality for customers to share files between their devices. However, distributed file systems are usually deployed as a shared resource; that is, other customers have access to it. Because of this sharing, a customer’s files may be buried deep in a filing structure, and it may not always be immediately evident to customers what kind of access they have to a particular file. Further, to use a distributed file system, the customer must always be connected to it. Files stored on a distributed file system are generally inaccessible if the customer’s machine is not connected to it, unless the customer has copied or moved the files to his machine’s local hard drive. However, doing so immediately creates the problem of having two filing systems for the same file, creating a mental burden on the customer.

Additionally, accessing a file located on a distributed file system tends to be slower than accessing files on the local hard drive. Modern applications are usually written to assume that the files they access are located locally, and thus are not optimized to access remote files. When these applications are used with remote files, they can lose performance by an order of magnitude. This problem can be fixed by automatically caching often-used files on the local file system, and only synchronizing them when they have been changed. However, this separate synchronization step introduces another problem: because the synchronization process can be lengthy, the customer is never entirely sure if the file he is remotely accessing is the latest version of the file, versus an earlier one that has been marked to be updated.

US 10,642,787 B1

3

Further, the directory may not reflect the existence of the file at all until synchronization finishes.

FTP is similar to a distributed file system with regards to files being hosted on a remote server. However FTP generally does manifest as a "disk drive" on the customer's desktop; the customer must use special FTP client software to access an FTP server. It shares the same problem as distributed file systems, with the additional problem of weak integration with applications. Applications can generally write and read files directly to and from a distributed file system. This is not the case with FTP, as the customer has to manually use the client software to perform these operations as a separate task.

Email was originally invented for messaging. From the beginning, the model it employs to make files accessible remotely is necessarily inefficient. Email's model for making files accessible is in the form of an email "attachment". Attachments are so named because they piggy-back on a message sent from one customer to another. A customer can make a file remotely available using email by attaching the file to an email and sending it to himself. He can then retrieve the file from a remote location by accessing the message on the email server. Email used in this way is even worse than FTP as the process is even more manual: a customer must find the message containing the file before he can even access it. Further, the location in which the attachment lives is read only. If the customer, for example, were to open the file, change it, then save it back out, the results would be ambiguous to the user because the email application, not the user, specified its location. Usually, the saved file would end up buried in an email file cache in an undisclosed area of the file system.

Flash Drives and External Disk Drives, although seemingly the most "primitive" way to ensure file availability, avoid all the problems related to network latency. However, these devices must be physically connected to the computer on which the files will be accessed. These restrictions preclude the customer from employing several effective work-flows including: using more than one computer to complete a single task (the files can only be accessed on one computer) and setting up an automated backup (the computer running the backup can't guarantee that the storage device will be connected come backup time). Further, to ensure full availability of the files, the customer must carry the device with them at all times, and must follow the associated protocols for mounting and dismounting the device.

Other problems with the prior art not described above can also be overcome using the teachings of embodiments of the present invention, as would be readily apparent to one of ordinary skill in the art after reading this disclosure.

SUMMARY OF THE INVENTION

In certain embodiments, automatic modification-triggered transfer of a file among two or more computer systems associated with a user. In some embodiments, a copy of a first file may be received, via a first application at a first computer system, from a second application at a second computer system associated with a user. The first file copy may be automatically received from the second application responsive to the user modifying a content of the first file, where the first file copy is a version of the first file that is generated from the user modifying the content of the first file. Responsive to receiving the first file copy from the second computer system, the first file copy may be automatically transferred via the first application to a third

4

computer system associated with the user to replace an older version of the first file stored on the third computer system.

In some embodiments, responsive to a user modifying a content of the file at a first client device (associated with the user), a server system may automatically receive a copy of the file from the first client device, where the file copy may be an updated version of the file that is generated from the user modifying the content of the file. After receiving metadata associated with the updated version of the file from the first client device, the server system may automatically transfer the metadata to a second client device associated with the user such that, before the file copy is transferred to the second client device, the transfer of the metadata to the second client device causes a file representation of the file presented on a user interface of the second client device to be updated based on the metadata.

BRIEF DESCRIPTION OF THE DRAWING

Preferred and alternative embodiments of the present invention are described in detail below with reference to the following drawings.

FIG. 1 is a schematic view of an exemplary operating environment in which an embodiment of the invention can be implemented;

FIG. 2 is a functional block diagram of an exemplary operating environment in which an embodiment of the invention can be implemented; and

FIG. 3 is a functional block diagram illustrating file sharing and/or synchronization according to an embodiment of the invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

An embodiment of the invention leverages remote programming concepts by utilizing processes called mobile agents (sometimes referred to as mobile objects or agent objects). Generally speaking, these concepts provide the ability for an object (the mobile agent object) existing on a first ("host") computer system to transplant itself to a second ("remote host") computer system while preserving its current execution state. The operation of a mobile agent object is described briefly below.

The instructions of the mobile agent object, its preserved execution state, and other objects owned by the mobile agent object are packaged, or "encoded," to generate a string of data that is configured so that the string of data can be transported by all standard means of communication over a computer network. Once transported to the remote host, the string of data is decoded to generate a computer process, still called the mobile agent object, within the remote host system. The decoded mobile agent object includes those objects encoded as described above and remains in its preserved execution state. The remote host computer system resumes execution of the mobile agent object which is now operating in the remote host environment.

While now operating in the new environment, the instructions of the mobile agent object are executed by the remote host to perform operations of any complexity, including defining, creating, and manipulating data objects and interacting with other remote host computer objects.

File transfer and/or synchronization, according to an embodiment, may be accomplished using some or all of the concepts described in commonly owned U.S. patent appli-

US 10,642,787 B1

5

cation Ser. No. 11/739,083, entitled “Electronic File Sharing,” the entirety of which is incorporated by reference as if fully set forth herein.

FIG. 1 illustrates an example of a suitable computing system environment **100** in which one or more embodiments of the invention may be implemented. The computing system environment **100** is only one example of a suitable computing environment and is not intended to suggest any limitation as to the scope of use or functionality of the invention. Neither should the computing environment **100** be interpreted as having any dependency or requirement relating to any one or combination of components illustrated in the exemplary operating environment **100**.

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

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

With reference to FIG. 1, an exemplary system for implementing the invention includes a general purpose computing device in the form of a computer **110**. Components of computer **110** may include, but are not limited to, a processing unit **120**, a system memory **130**, and a system bus **121** that couples various system components including the system memory to the processing unit **120**. The system bus **121** may be any of several types of bus structures including a memory bus or memory controller, a peripheral bus, and a local bus using any of a variety of bus architectures. By way of example, and not limitation, such architectures include Industry Standard Architecture (ISA) bus, Micro Channel Architecture (MCA) bus, Enhanced ISA (EISA) bus, Video Electronics Standards Association (VESA) local bus, and Peripheral Component Interconnect (PCI) bus also known as Mezzanine bus.

Computer **110** typically includes a variety of computer readable media. Computer readable media can be any available media that can be accessed by computer **110** and includes both volatile and nonvolatile media, removable and non-removable media. By way of example, and not limitation, computer readable media may comprise computer storage media and communication media. Computer storage media includes both volatile and nonvolatile, removable and non-removable media implemented in any method or technology for storage of information such as computer readable instructions, data structures, program modules or other data. Computer storage media includes, but is not limited to,

6

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

The system memory **130** includes computer storage media in the form of volatile and/or nonvolatile memory such as read only memory (ROM) **131** and random access memory (RAM) **132**. A basic input/output system **133** (BIOS), containing the basic routines that help to transfer information between elements within computer **110**, such as during start-up, is typically stored in ROM **131**. RAM **132** typically contains data and/or program modules that are immediately accessible to and/or presently being operated on by processing unit **120**. By way of example, and not limitation, FIG. 1 illustrates operating system **134**, application programs **135**, other program modules **136**, and program data **137**.

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

The drives and their associated computer storage media discussed above and illustrated in FIG. 1, provide storage of computer readable instructions, data structures, program modules and other data for the computer **110**. In FIG. 1, for example, hard disk drive **141** is illustrated as storing operating system **144**, application programs **145**, other program modules **146**, and program data **147**. Note that these components can either be the same as or different from operating system **134**, application programs **135**, other program modules **136**, and program data **137**. Operating system **144**, application programs **145**, other program modules **146**, and program data **147** are given different numbers here to illustrate that, at a minimum, they are different copies. A user may enter commands and information into the computer **20** through input devices such as a keyboard **162** and pointing device **161**, commonly referred to as a mouse, trackball or touch pad. Other input devices (not shown) may include a

US 10,642,787 B1

7

microphone, joystick, game pad, satellite dish, scanner, or the like. These and other input devices are often connected to the processing unit 120 through a user input interface 160 that is coupled to the system bus, but may be connected by other interface and bus structures, such as a parallel port, game port or a universal serial bus (USB). A monitor 191 or other type of display device is also connected to the system bus 121 via an interface, such as a video interface 190. In addition to the monitor, computers may also include other peripheral output devices such as speakers 197 and printer 196, which may be connected through an output peripheral interface 190.

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

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

Referring now to FIG. 2, an embodiment of the present invention can be described in the context of an exemplary computer network system 200 as illustrated. System 200 includes electronic user devices 210, 280, such as personal computers or workstations, that are linked via a communication medium, such as a network 220 (e.g., the Internet), to an electronic device or system, such as a server 230. The server 230 may further be coupled, or otherwise have access, to a database 240, electronic storage 270 and a computer system 260. Although the embodiment illustrated in FIG. 2 includes one server 230 coupled to two user devices 210, 280 via the network 220, it should be recognized that embodiments of the invention may be implemented using two or more such user devices coupled to one or more such servers.

In an embodiment, each of the user devices 210, 280 and server 230 may include all or fewer than all of the features associated with the computer 110 illustrated in and discussed with reference to FIG. 1. User devices 210, 280 include or are otherwise coupled to a computer screen or display 250, 290, respectively. User devices 210, 280 can be used for various purposes including both network- and local-computing processes.

The user devices 210, 280 are linked via the network 220 to server 230 so that computer programs, such as, for example, a browser or other applications, running on one or

8

more of the user devices 210, 280 can cooperate in two-way communication with server 230 and one or more applications running on server 230. Server 230 may be coupled to database 240 and/or electronic storage 270 to retrieve information therefrom and to store information thereto. Additionally, the server 230 may be coupled to the computer system 260 in a manner allowing the server to delegate certain processing functions to the computer system.

Referring now to FIG. 3, illustrated is functionality of an embodiment of the invention allowing a user (not shown) who owns or otherwise controls devices 210, 280 to automatically maintain file synchronization between at least devices 210, 280, or any other user devices on which principles of the present invention are implemented. In an embodiment, an administrator (not shown) of the server 230 or other appropriate electronic device transfers a file-transfer and/or synchronization application to the user devices 210, 280 for installation thereon. Once installed on the user devices 210, 280, the file-transfer application provides file-transfer clients 310, 320 executable by the user devices 210, 280, respectively. Each of the file-transfer clients 310, 320 may, but need not, include a respective mobile-agent runtime environment 330, 340. The mobile-agent runtime environment 330, 340 include portions of memory of the user devices 210, 280 dedicated to allowing a mobile object the ability to perform operations that the mobile object is programmed to carry out. Also included in the file-transfer application are user interfaces 350, 360 that are displayable on the displays 250, 290, respectively. In an embodiment, the interfaces 350, 360 allow a user to view, access and/or organize files to be synched among the various user devices.

Generally, all files that the user desires to be synched or shared may at some point be uploaded by one or more of the user devices 210, 280 and stored in storage 270. Upon receiving the files to be synched, the server 230 can store such files in the storage 270 and/or transfer the files to one or more of the respective hard drives of the user devices 210, 280, thereby enabling each respective user device to access such files. In this manner, the server 230 is operable to treat each hard drive of the respective user devices 210, 280 as a local document cache for files received by the server. Typically, the server 230 will store one or more of the received files to the storage 270 only if the destination user device is offline or otherwise temporarily not in communication with the server 230. Upon resuming communication with the destination user device, the server 230 will transfer the temporarily stored files to the destination device.

In operation, according to an embodiment, the user may open and modify a file 370, such as a word-processing document or other electronic file. Alternatively, the user may create a first instance of the file 370. The user may have previously have associated, or may now associate, the file 370 with the transfer client 310. Upon a predetermined and user-configurable triggering event, the transfer client 310 transfers the modified file 370, or a copy of the modified file, to the server 230. Such a triggering event may include, but be not limited to, the user saving the file, the elapsing of a predetermined amount of time during which the file has been opened, or the re-initiation of a communication session between the device 210 and the server 230.

The file 370 is transferred to the server 230 on which is executing a synchronization application 380, which may include a mobile-agent runtime environment. Through user configuration, the synch application 380 monitors a set of user devices to which the file 370 should be transferred to effect file synchronization. In the illustrated embodiment, this set of user devices includes the user device 280. The

synch application **380** polls the device **280** to determine whether the device **280** is in communication with the server **230**. If the device **280** is in communication with the server **230**, the synch application **380** transfers the file **370** to the device **280**, whereupon the transfer client **320** resident on the device **280** replaces the previous version of the file **370**, previously cached on the device **280**, with the latest version of the file **370** modified on the user device **210**. If the device **280** is not currently in communication with the server **230**, the synch application **380** may store the file **370** in the storage **270** until such time as communication between the device **280** and server **230** is reestablished. As illustrated in FIG. 3, a similar reverse-direction synchronization process may be performed by the synch application **380** and the transfer clients **310**, **320** with regard to a file **315** modified on device **280** and synchronized to device **210**.

In an embodiment, the user interfaces **350**, **360** may include a list of the customer's documents and related metadata, as well as any one-to-one or one-to-many relationships between the documents and metadata. An embodiment can always provide customers with an accurate "picture" of their document collection, regardless of whether their devices physically contain the documents. As alluded to earlier, a problem with distributed file systems and FTP is the latency between a file being put onto a file system and it showing up on a remote machine. To prevent this problem, an embodiment directory is decoupled from the movement of files. An embodiment's directory update system updates at a higher priority than the documents to be synchronized. This feature ensures that when a customer browses or searches through his set of documents, they appear even if they have not yet been cached locally on the user device. An indicator signifying a document's availability may be prominently displayed adjacent to the document's representation so that customers are aware of the document's availability.

An embodiment may include a stand-alone application that allows customers to find and manage documents associated with transfer clients **310**, **320** by visualizing relationships between documents and their metadata. It allows customers to tag documents with any number of identifiers. Customers can relate both documents and tags with each other in any number of user-specified one-to-one and one-to-many relationships, and an embodiment provides a user interface to browse and search on these relationships. To mitigate the customers' learning curve, an embodiment can implement relationships common to contemporary file systems, including a folder hierarchy. In addition to this, an embodiment provides direct support for methods that the customer uses to organize documents by manifesting them as user interface idioms. This is unlike conventional document filing systems which require the customer to work within a strict folder metaphor for organization.

Some alternate methods that an embodiment supports for organizing documents include:

Allow customers to organize their documents by application. Many times customers remember the application used to create a document instead of the document's name or its location in a hierarchy.

Allow customers to organize their documents by most recent access. Customers are likely to access a document they've accessed in the near past. Usually, such documents are part of a task that the customer is actively working.

Allow customers to organize their documents by project or subproject.

Allow customers to organize their documents by people. Many times, especially in the context of a collabora-

tion, a document is directly related to one or more people other than the customer.

Allow the customer to organize their document by process stage. Documents may represent one or more stages of a process. Customers need a method for organizing documents by process stage, and a mechanism for moving the document through a set of predefined stages.

Allow customers to organize their documents by any of the aforementioned methods concurrently. These organization methods are not mutually exclusive.

An embodiment presents an interface that allows a customer to locate one or more documents associated with the transfer clients **310**, **320** and open such document into a separate software application. Since this interface is intended to be used from within the separate application, that application may need to know how to invoke such interface. Advantageously, this invocation behavior can be provided to the application using the application's plug-in API.

An embodiment presents an interface that allows a customer to synchronize a currently opened document according to processes described elsewhere herein. This interface can be invoked within an application and can be made available to the application in the manner described above in connection with the application's plug-in API.

Some files associated with the transfer clients **310**, **320** are dependent on other files associated with the transfer clients **310**, **320**. For example, a desktop publishing document may include images that are stored in files separate from the main document. Previous file-synching solutions treat these files as separate. Because of this, for example, a document synchronized from the device **210** to the device **280** may be opened by the user of the device **280** before the image files have been fully transferred to the device **280**. This causes the document to fail to open, or break, since the image files don't exist or are incomplete. An embodiment prevents this by: (1) always ensuring the file catalog (e.g., the stand-alone application that allows customers to find and manage documents associated with transfer clients **310**, **320**, as discussed above herein) is synchronized before any file data is synchronized, and (2) pausing any file access by any program until the file contents have been fully synchronized. In such an embodiment, if a user attempts, using a software program, to open a file whose related files haven't yet finished transferring to the local (hard drive) cache, if that software attempts to open the related files, the software program is blocked by an embodiment until the requested files are downloaded and ready to access.

Other file sending and synchronizing software requires the user to upload their data to a storage device owned by the operator of the service. An embodiment treats storage as a participant in the synchronization process; this means that the user can choose the service or device where their files will be stored. The file transfer/synching is abstracted from the storage system allowing any storage to be used. An embodiment treats storage like any other synch target, such as a desktop computer, or a cell phone. As such, any device owned or otherwise controlled by the user and running a synch application, such as synch application **380**, as provided in an embodiment of the invention can perform the storage and/or synching functions described elsewhere herein. That is, the user device **280** or user device **210**, rather than the server **230**, may perform such functions.

While a preferred embodiment of the invention has been illustrated and described, as noted above, many changes can be made without departing from the spirit and scope of the invention. For example, as an alternative to the approach

US 10,642,787 B1

11

described with reference to FIG. 3, wherein the transfer clients 310, 320 function to “push” modified or created files to the synch application 380, the synch application 380 may instead function to periodically “pull” or otherwise actively retrieve such files from the transfer clients 310, 320. Instead, the invention should be determined entirely by reference to the claims that follow.

What is claimed is:

1. A system comprising:

a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to:

receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being an updated version of the first file that is generated from the user modifying the content of the first file;

receive, over a network, from the first client device, first metadata associated with the updated version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file; and

automatically transfer, based on the first priority being greater than the second priority, the first metadata over a network to a second client device associated with the user such that the first metadata is transferred to the second client device before the copy of the first file is transferred to the second client device, wherein, before the copy of the first file is transferred to the second client device:

(i) the transfer of the first metadata to the second client device causes a file representation of the first file presented on a user interface of the second client device to be updated based on the first metadata, and

(ii) instead of the updated file representation of the first file representing a version of the first file currently stored on the second client device, the updated file representation represents the updated version of the first file that is currently stored on the first client device and not currently stored on the second client device, and

wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.

2. The system of claim 1, wherein, before the copy of the first file is transferred to the second client device, the transfer of the first metadata to the second client device causes a graphical availability indication of the updated version of the first file to be presented on the user interface of the second client device based on the first metadata.

3. The system of claim 1, wherein the computer program instructions, when executed, cause the server system to:

receive a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the second client device responsive to the user modifying a content

12

of the second file stored on the second client device, the copy of the second file being an updated version of the second file that is generated from the user modifying the content of the second file;

determine that the server system is in communication with the first client device associated with the user; and automatically transfer the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client device and (ii) receiving the copy of the second file from the second client device.

4. The system of claim 1, wherein the copy of the first file is automatically received from the first client device responsive to (i) a push request of the first client device and (ii) the user modifying the content of the first file stored on the first client device.

5. The system of claim 1, wherein the copy of the first file is automatically received from a first application at the first client device, and wherein the first application comprises a runtime environment for one or more mobile-agent objects.

6. The system of claim 5, wherein the first application is configured to create a first mobile object, and wherein the first mobile object is configured to create a proxy object at the server system.

7. The system of claim 6, wherein the first mobile object is configured to provide the copy of the first file to the proxy object, and wherein the proxy object is configured to store the copy of the first file on a memory device associated with the server system.

8. A method being implemented by a server system comprising one or more processors executing computer program instructions that, when executed, perform the method, the method comprising:

receiving, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being an updated version of the first file that is generated from the user modifying the content of the first file;

receiving, over a network, from the first client device, first metadata associated with the updated version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file; and

automatically transferring, based on the first priority being greater than the second priority, the first metadata over a network to a second client device associated with the user such that the first metadata is transferred to the second client device before the copy of the first file is transferred to the second client device,

wherein, before the copy of the first file is transferred to the second client device:

(i) the transfer of the first metadata to the second client device causes a file representation of the first file presented on a user interface of the second client device to be updated based on the first metadata, and

(ii) the updated file representation represents the updated version of the first file that is currently stored on the first client device and not currently stored on the second client device, and

wherein at least one of the server system or the first client device comprises a priority assignment configuration to

US 10,642,787 B1

13

assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration. 5

9. The method of claim 8, wherein, before the copy of the first file is transferred to the second client device, the transfer of the first metadata to the second client device causes a graphical availability indication of the updated version of the first file to be presented on the user interface of the second client device based on the first metadata. 10

10. The method of claim 8, further comprising:

receiving a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the second client device responsive to the user modifying a content of the second file stored on the second client device, the copy of the second file being an updated version of the second file that is generated from the user modifying the content of the second file; 15 20

determining that the server system is in communication with the first client device associated with the user; and automatically transferring the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client device and (ii) receiving the copy of the second file from the second client device. 25 30

11. The method of claim 8, wherein the copy of the first file is automatically received from the first client device responsive to (i) a push request of the first client device and (ii) the user modifying the content of the first file stored on the first client device. 35

12. The method of claim 8, wherein the copy of the first file is automatically received from a first application at the first client device, wherein the first application comprises a runtime environment for one or more mobile-agent objects, wherein the first application is configured to create a first mobile object, wherein the first mobile object is configured to create a proxy object at the server system, wherein the first mobile object is configured to provide the copy of the first file to the proxy object, and wherein the proxy object is configured to store the copy of the first file on a memory device associated with the server system. 40 45

13. One or more non-transitory machine-readable media storing instructions that, when executed by one or more processors of a server system, cause operations comprising:

receiving, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being an updated version of the first file that is generated from the user modifying the content of the first file; 50 55

receiving, over a network, from the first client device, first metadata associated with the updated version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file; and 60

automatically transferring, based on the first priority being greater than the second priority, the first metadata over a network to a second client device associated 65

14

with the user such that the first metadata is transferred to the second client device before the copy of the first file is transferred to the second client device,

wherein, before the copy of the first file is transferred to the second client device:

- (i) the transfer of the first metadata to the second client device causes a file representation of the first file presented on a user interface of the second client device to be updated based on the first metadata, and
- (ii) the updated file representation represents the updated version of the first file that is currently stored on the first client device and not currently stored on the second client device, and

wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration. 70

14. The media of claim 13, wherein, before the copy of the first file is transferred to the second client device, the transfer of the first metadata to the second client device causes an availability indication of the updated version of the first file to be presented on the user interface of the second client device based on the first metadata. 75

15. The media of claim 13, the operations further comprising:

receiving a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the second client device responsive to the user modifying a content of the second file stored on the second client device, the copy of the second file being an updated version of the second file that is generated from the user modifying the content of the second file; 80 85

determining that the server system is in communication with the first client device associated with the user; and automatically transferring the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client device and (ii) receiving the copy of the second file from the second client device. 90 95

16. The media of claim 13, wherein the copy of the first file is automatically received from the first client device responsive to (i) a push request of the first client device and (ii) the user modifying the content of the first file stored on the first client device. 100

17. The media of claim 13, wherein the copy of the first file is automatically received from a first application at the first client device, wherein the first application comprises a runtime environment for one or more mobile-agent objects, wherein the first application is configured to create a first mobile object, wherein the first mobile object is configured to create a proxy object at the server system, wherein the first mobile object is configured to provide the copy of the first file to the proxy object, and wherein the proxy object is configured to store the copy of the first file on a memory device associated with the server system. 105 110

* * * * *

EXHIBIT 6



US010754823B2

(12) **United States Patent**
Manzano

(10) **Patent No.:** **US 10,754,823 B2**
(45) **Date of Patent:** ***Aug. 25, 2020**

(54) **PRE-FILE-TRANSFER AVAILABILITY INDICATION BASED ON PRIORITIZED METADATA**

(71) Applicant: **TOPIA TECHNOLOGY, INC.**, Tacoma, WA (US)

(72) Inventor: **Michael R. Manzano**, Seattle, WA (US)

(73) Assignee: **TOPIA TECHNOLOGY, INC.**, Tacoma, WA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

This patent is subject to a terminal disclaimer.

(21) Appl. No.: **16/750,435**

(22) Filed: **Jan. 23, 2020**

(65) **Prior Publication Data**

US 2020/0159695 A1 May 21, 2020

Related U.S. Application Data

(63) Continuation of application No. 16/361,641, filed on Mar. 22, 2019, which is a continuation of application (Continued)

(51) **Int. Cl.**
G06F 16/00 (2019.01)
G06F 16/11 (2019.01)
(Continued)

(52) **U.S. Cl.**
CPC **G06F 16/122** (2019.01); **G06F 15/16** (2013.01); **G06F 16/00** (2019.01); **G06F 16/128** (2019.01);
(Continued)

(58) **Field of Classification Search**
CPC G06F 16/213; G06F 16/583; G06F 16/68; G06F 16/14; G06F 16/1873; G06F 16/27;
(Continued)

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,600,834 A 2/1997 Howard
5,806,078 A 9/1998 Hug
(Continued)

FOREIGN PATENT DOCUMENTS

EP 1 130 511 A2 9/2001
WO 98/56149 A1 12/1998
WO 2007/047302 A2 4/2007

OTHER PUBLICATIONS

FolderShare—Secure Remote Access VPN Solution; Document Management & Real-time File Mirroring Solution; < <https://web.archive.org/web/20040804020435/http://www.foldershare.com:80/>>; Aug. 4, 2004.

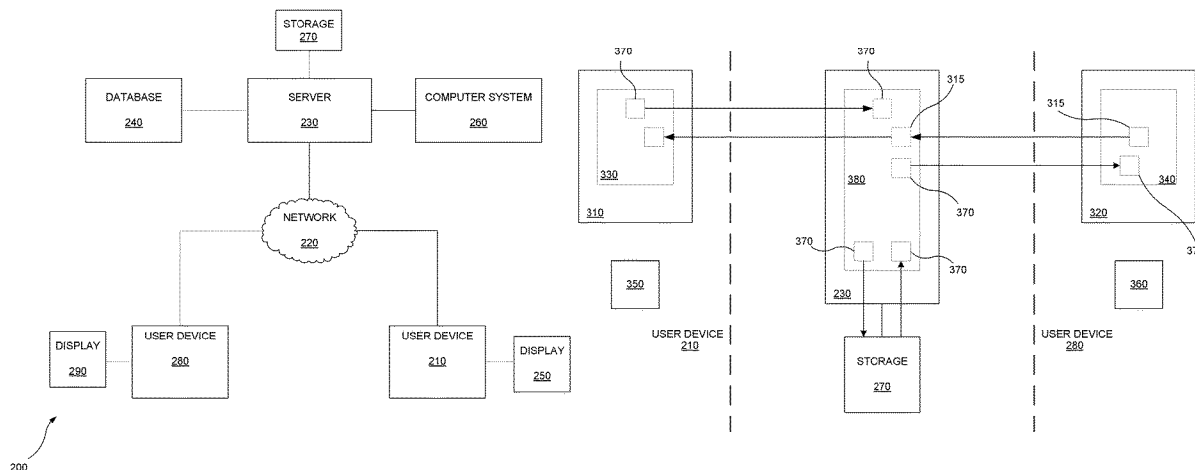
(Continued)

Primary Examiner — Srirama Channavajjala
(74) *Attorney, Agent, or Firm* — Pillsbury Winthrop Shaw Pittman LLP

(57) **ABSTRACT**

In some embodiments, responsive to a user modifying a content of the file at a first client device (associated with the user), a server system may automatically receive a copy of the file from the first client device, where the file copy may be an updated version of the file that is generated from the user modifying the content of the file. After receiving metadata associated with the updated version of the file from the first client device, the server system may automatically transfer the metadata to a second client device associated with the user such that, before the file copy is transferred to the second client device, the transfer of the metadata to the second client device causes a graphical availability indication of the updated version of the file to be presented (e.g., proximate a file icon representing the file) at the second client device based on the metadata.

17 Claims, 3 Drawing Sheets



US 10,754,823 B2

Page 2

Related U.S. Application Data					
	No. 16/017,348, filed on Jun. 25, 2018, now Pat. No. 10,289,607, which is a continuation of application No. 14/860,289, filed on Sep. 21, 2015, now Pat. No. 10,067,942, which is a continuation of application No. 12/267,852, filed on Nov. 10, 2008, now Pat. No. 9,143,561.	7,398,327 B2	7/2008	Lee	
		7,415,588 B2	8/2008	Hong et al.	
		7,415,615 B2	8/2008	Skygebjerg	
		7,457,631 B2	11/2008	Yach et al.	
		7,467,353 B2	12/2008	Kurlander et al.	
		7,483,925 B2	1/2009	Koskimies et al.	
		7,526,575 B2	4/2009	Rabbers et al.	
		7,574,711 B2	8/2009	Zondervan et al.	
		7,584,186 B2	9/2009	Chen et al.	
		7,587,446 B1	9/2009	Onyon et al.	
(60)	Provisional application No. 60/986,896, filed on Nov. 9, 2007.	7,613,773 B2	11/2009	Watt	
		7,639,116 B2	12/2009	Saunders	
		7,657,271 B2	2/2010	Kim	
(51)	Int. Cl.	7,680,838 B1	3/2010	Shaw	
	G06F 15/16 (2006.01)	7,680,885 B2	3/2010	Schauser et al.	
	G06F 16/13 (2019.01)	7,752,166 B2	7/2010	Quinlan et al.	
	G06F 16/14 (2019.01)	7,761,414 B2	7/2010	Freedman	
	G06F 16/176 (2019.01)	7,788,303 B2	8/2010	Mikesell et al.	
	G06F 16/178 (2019.01)	7,885,925 B1	2/2011	Strong et al.	
	G06F 16/18 (2019.01)	7,895,334 B1	2/2011	Tu et al.	
	H04L 29/08 (2006.01)	7,987,420 B1	7/2011	Kloba et al.	
(52)	U.S. Cl.	8,009,966 B2	8/2011	Bloom et al.	
	CPC G06F 16/13 (2019.01); G06F 16/14 (2019.01); G06F 16/176 (2019.01); G06F 16/178 (2019.01); G06F 16/1873 (2019.01); H04L 67/1095 (2013.01)	8,019,900 B1	9/2011	Sekar et al.	
		8,112,549 B2	2/2012	Srinivasan et al.	
		8,208,792 B2 *	6/2012	Morioka G11B 27/034 386/248	
		8,244,288 B2	8/2012	Chipchase	
(58)	Field of Classification Search	8,321,534 B1	11/2012	Roskind et al.	
	CPC G06F 16/122; G06F 16/13-14; G06F 16/128; G06F 16/176; G06F 16/178	8,370,423 B2	2/2013	Ozzie et al.	
	See application file for complete search history.	8,386,558 B2	2/2013	Schleifer et al.	
		8,565,729 B2	10/2013	Moseler et al.	
		9,143,561 B2 *	9/2015	Manzano G06F 16/176	
		10,067,942 B2 *	9/2018	Manzano G06F 15/16	
		10,289,607 B2 *	5/2019	Manzano G06F 16/176	
(56)	References Cited	2002/0026478 A1	2/2002	Rodgers et al.	
	U.S. PATENT DOCUMENTS	2002/0035697 A1	3/2002	McCurdy	
		2002/0087588 A1	7/2002	McBride et al.	
		2002/0184318 A1 *	12/2002	Pineau H04L 29/06 709/206	
		2002/0194382 A1	12/2002	Kausik	
		2003/0028514 A1	2/2003	Lord	
		2003/0028542 A1	2/2003	Muttik et al.	
		2003/0038842 A1	2/2003	Peck et al.	
		2003/0074376 A1 *	4/2003	Benayoun G06F 16/10	
		2003/0078946 A1	4/2003	Costello	
		2003/0115547 A1	6/2003	Ohwada	
		2003/0125057 A1	7/2003	Pesola	
		2003/0135565 A1	7/2003	Estrada	
		2004/0003013 A1 *	1/2004	Coulthard H04L 67/2823	
		2004/0049345 A1	3/2004	McDonough et al.	
		2004/0093361 A1	5/2004	Therrien et al.	
		2004/0107225 A1	6/2004	Rudoff	
		2004/0133629 A1	7/2004	Reynolds	
		2004/0158817 A1	8/2004	Okachi	
		2004/0172424 A1	9/2004	Edelstein et al.	
		2005/0091316 A1	4/2005	Ponce et al.	
		2005/0097225 A1	5/2005	Glatt et al.	
		2005/0177602 A1 *	8/2005	Kaler H04L 63/04	
		2005/0210371 A1	9/2005	Pollock	
		2005/0220080 A1	10/2005	Ronkainen et al.	
		2005/0223047 A1	10/2005	Shah et al.	
		2006/0010150 A1	1/2006	Shaath et al.	
		2006/0026567 A1 *	2/2006	McVoy G06F 16/213 717/120	
		2006/0058907 A1	3/2006	Suderman	
		2006/0074985 A1	4/2006	Wolfish	
		2006/0101064 A1	5/2006	Strong et al.	
		2006/0129627 A1	6/2006	Phillips	
		2006/0143129 A1	6/2006	Holm	
		2006/0168118 A1	7/2006	Godlin	
		2006/0189348 A1	8/2006	Montulli et al.	
		2006/0224626 A1 *	10/2006	Lakshminath G06F 16/1873	
		2007/0014314 A1	1/2007	O'Neil	
		2007/0016629 A1	1/2007	Reinsch	
		2007/0027936 A1	2/2007	Stakutis	
		2007/0100913 A1	5/2007	Sumner et al.	
		2007/0124334 A1 *	5/2007	Pepin G06F 8/73	
		2007/0180084 A1 *	8/2007	Mohanty G06F 11/1451 709/223	

US 10,754,823 B2

Page 3

(56)

References Cited

U.S. PATENT DOCUMENTS

2007/0191057	A1	8/2007	Kamada	
2007/0203927	A1*	8/2007	Cave	G06F 16/14
2007/0238440	A1	10/2007	Sengupta et al.	
2008/0005114	A1	1/2008	Li	
2008/0005195	A1	1/2008	Li	
2008/0005280	A1	1/2008	Adams	
2008/0086494	A1	4/2008	Heller et al.	
2008/0168526	A1	7/2008	Robbin et al.	
2008/0288578	A1	11/2008	Silfverberg	
2009/0013009	A1	1/2009	Nakayama	
2009/0024922	A1	1/2009	Markowitz et al.	
2009/0063711	A1	3/2009	Finkelstein	
2009/0282050	A1	11/2009	Thomas et al.	
2013/0226871	A1	8/2013	Sarnowski	
2016/0125058	A1*	5/2016	Jain	G06F 16/27 707/639

OTHER PUBLICATIONS

FolderShare; Your Files Anywhere; <<https://web.archive.org/web/20030808183932/http://www.foldershare.com:80/>>; Aug. 8, 2003.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20040814015727/http://www.foldershare.com:80/>>; Aug. 14, 2004.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20040820052105/http://www.foldershare.com:80/>>; Aug. 20, 2004.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20041211020957/http://foldershare.com:80/>>; Dec. 11, 2004.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20041217041726/http://foldershare.com:80/>>; Dec. 17, 2004.

FolderShare; Your Smart File Transfer Solution; <<https://web.archive.org/web/20031220151508/http://www.foldershare.com:80/>>; Dec. 20, 2003.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20041230211050/http://www.foldershare.com:80/>>; Dec. 30, 2004.

FolderShare—Secure Remote Access VPN Solution; Document Management & Real-time File Mirroring Solution; <<https://web.archive.org/web/20040701113739/http://foldershare.com:80/>>; Jul. 1, 2004.

FolderShare—Secure Remote Access VPN Solution; Document Management & Real-time File Mirroring Solution; <<https://web.archive.org/web/20040711062548/http://www.foldershare.com:80/>>; Jul. 11, 2004.

FolderShare; Your Files Anywhere; <<https://web.archive.org/web/20030722054342/http://foldershare.com:80/>>; Jul. 22, 2003.

FolderShare—Secure Remote Access VPN Solution; Document Management & Real-time File Mirroring Solution; <<https://web.archive.org/web/20040730030655/http://www.foldershare.com:80/>>; Jul. 30, 2004.

FolderShare—Secure Remote Access VPN Solution; Document Management & Real-time File Mirroring Solution; <<https://web.archive.org/web/20040603205113/http://www.foldershare.com:80/>>; Jun. 3, 2004.

FolderShare—Secure Remote Access VPN Solution; Document Management & Real-time File Mirroring Solution; <<https://web.archive.org/web/20040613161906/http://www.foldershare.com:80/>>; Jun. 13, 2004.

FolderShare—Secure Remote Access VPN Solution; Document Management & Real-time File Mirroring Solution; <<https://web.archive.org/web/20040629075057/http://www.foldershare.com:80/>>; Jun. 29, 2004.

FolderShare—Secure Remote Access VPN Solution; Your Smart File Transfer & Real-time File Mirroring Solution; <<https://web.archive.org/web/20040316235151/http://foldershare.com:80/>>; Mar. 16, 2004.

FolderShare—Secure Remote Access VPN Solution; Your Smart File Transfer & Real-time File Mirroring Solution; <<https://web.archive.org/web/20040325034239/http://www.foldershare.com:80/>>; Mar. 25, 2004.

FolderShare—Secure Remote Access VPN Solution; Document Management & Real-time File Mirroring Solution; <<https://web.archive.org/web/20040512211417/http://www.foldershare.com:80/>>; May 12, 2004.

FolderShare; Your Files Anywhere; <<https://web.archive.org/web/20030531180252/http://www.foldershare.com:80/>>; May 31, 2003.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20041104031510/http://www.foldershare.com:80/>>; Nov. 4, 2004.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20041117092357/http://www.foldershare.com:80/>>; Nov. 17, 2004.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20041123085254/http://www.foldershare.com:80/>>; Nov. 23, 2004.

FolderShare; Your Files Anywhere; <<https://web.archive.org/web/20031128143634/http://foldershare.com:80/>>; Nov. 28, 2003.

FolderShare; Your Files Anywhere; <<https://web.archive.org/web/20031001071631/http://foldershare.com:80/>>; Oct. 1, 2003.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20041012083127/http://www.foldershare.com:80/>>; Oct. 12, 2004.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20041029085820/http://www.foldershare.com:80/>>; Oct. 29, 2004.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20040901034646/http://www.foldershare.com:80/>>; Sep. 1, 2004.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20040909075254/http://www.foldershare.com:80/>>; Sep. 9, 2004.

FolderShare; Your Files Anywhere; <<https://web.archive.org/web/20030920051943/http://www.foldershare.com:80/>>; Sep. 20, 2003.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20040924032146/http://www.foldershare.com:80/>>; Sep. 24, 2004.

Marshall, M., “The Y Combinator List,” Venture Beat, Aug. 2007, Retrieved from the Internet: URL: <<https://venturebeat.com/2007/08/16/the-y-combinator-list/>>, 4 pages.

Jarvis, A., “Dropbox pitch deck to raise seed capital investment,” Medium, Mar. 2018, Retrieved from the Internet: URL: <<https://medium.com/@adjblog/dropbox-pitch-deck-to-raise-seed-capital-investment-6a6cd6517e56>>, 12 pages.

* cited by examiner

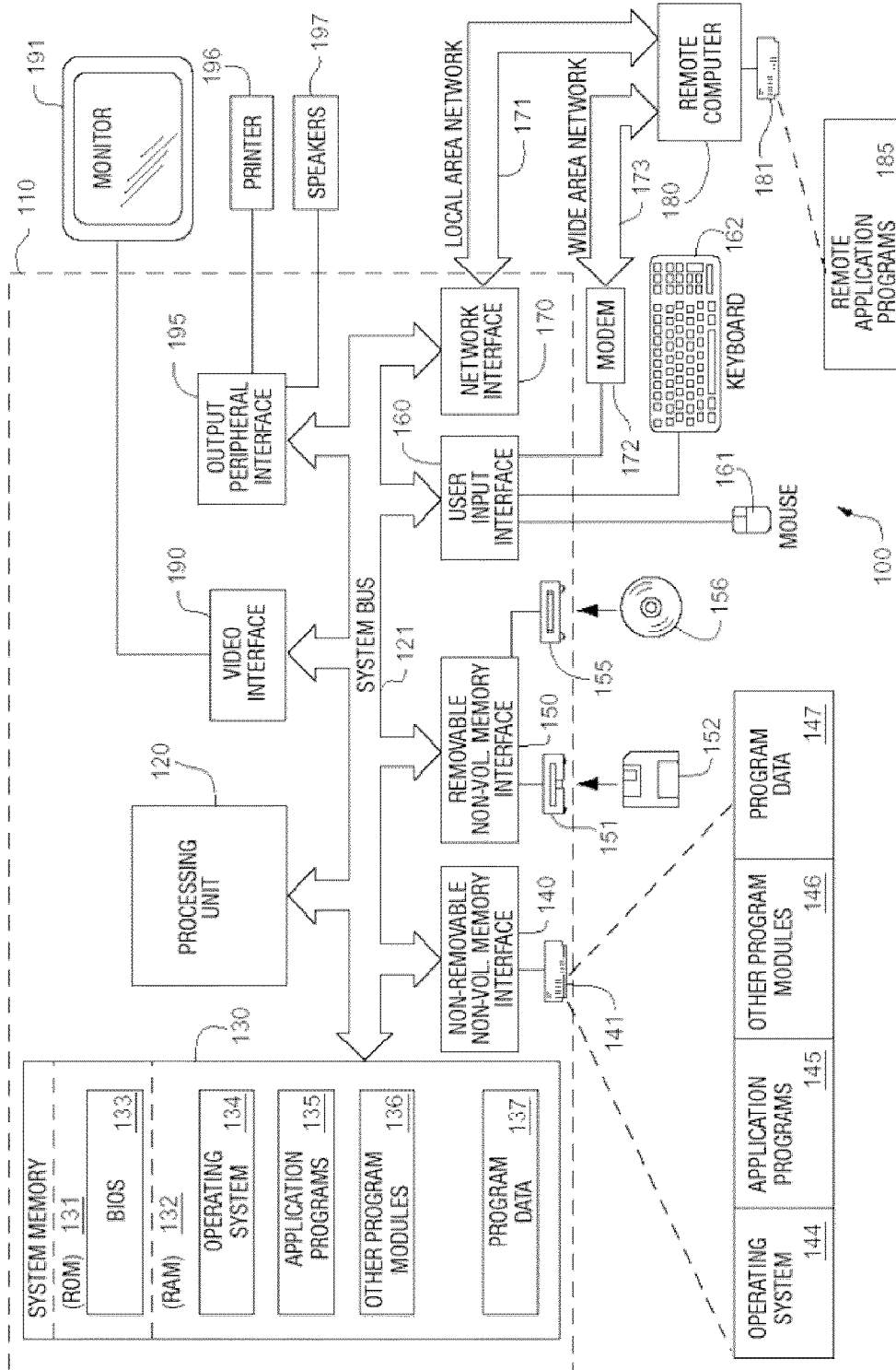


FIG. 1

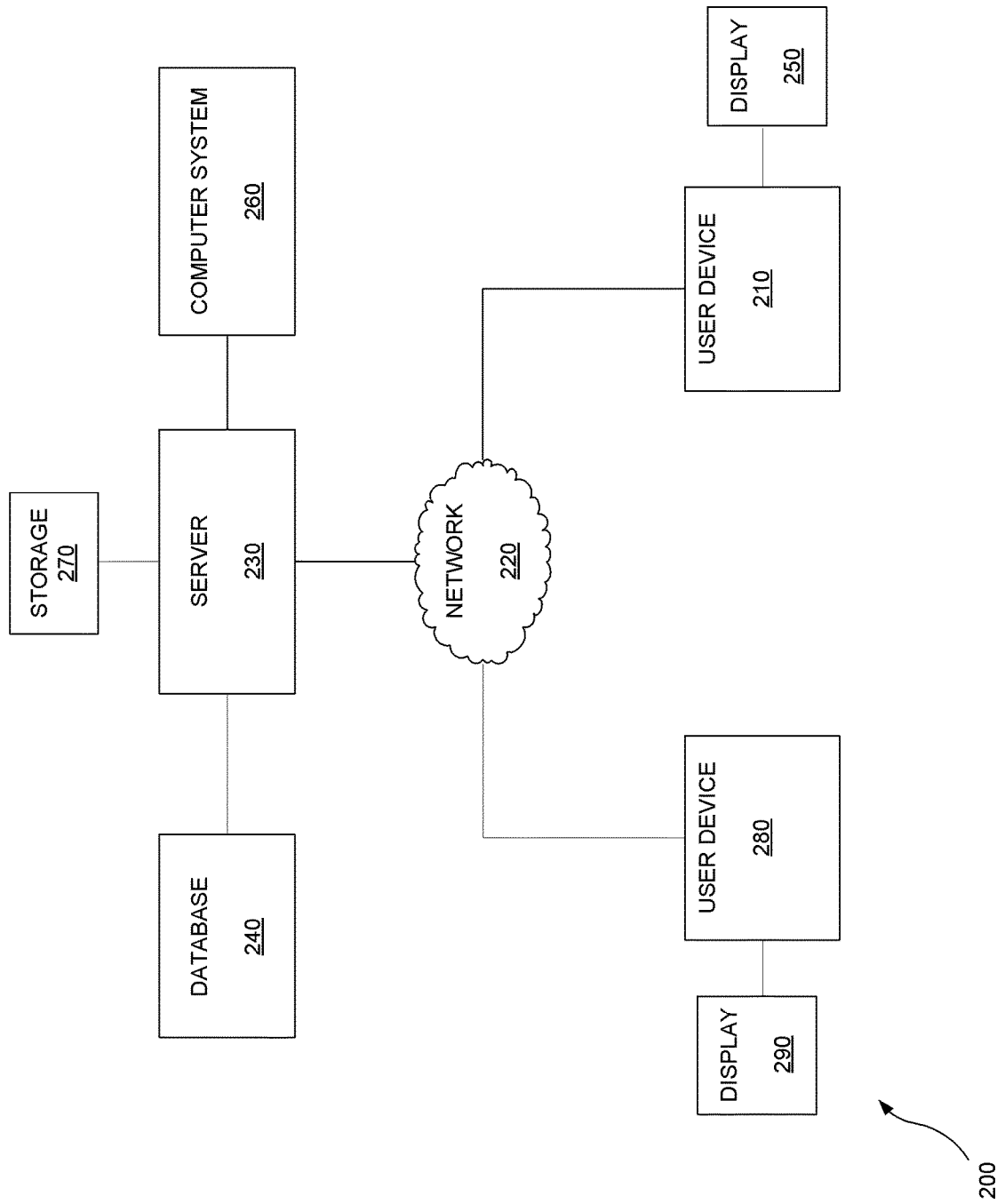


FIG. 2

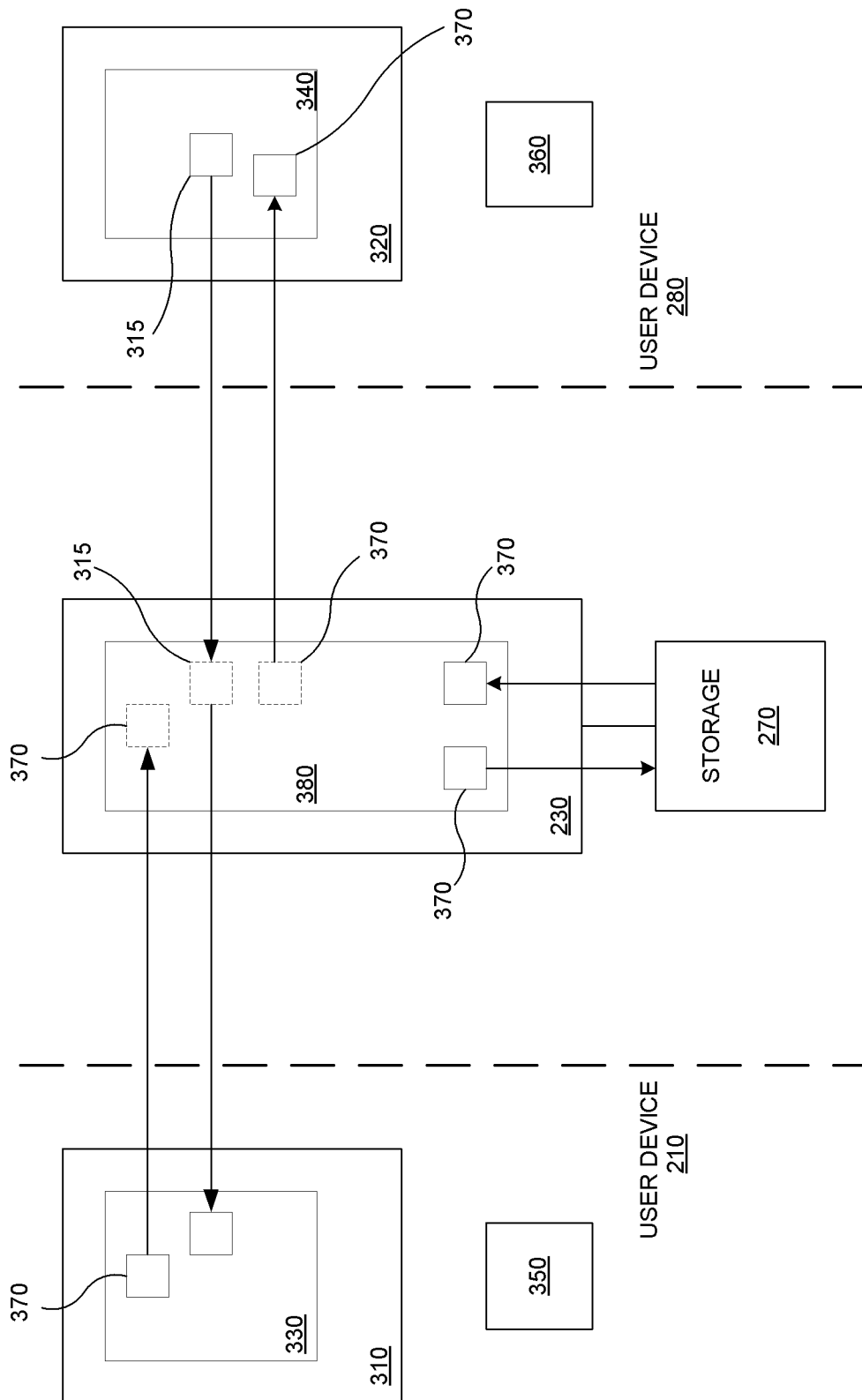


FIG. 3

US 10,754,823 B2

1

**PRE-FILE-TRANSFER AVAILABILITY
INDICATION BASED ON PRIORITIZED
METADATA**

CROSS-REFERENCE TO RELATED
APPLICATIONS

This application is a continuation of U.S. patent application Ser. No. 16/361,641, filed Mar. 22, 2019, which is a continuation of U.S. patent application Ser. No. 16/017,348, filed Jun. 25, 2018, which is a continuation of U.S. patent application Ser. No. 14/860,289, filed Sep. 21, 2015, now U.S. Pat. No. 10,067,942, which is a continuation of U.S. patent application Ser. No. 12/267,852, filed Nov. 10, 2008, now U.S. Pat. No. 9,143,561, which claims priority to U.S. Provisional Application No. 60/986,896 entitled “ARCHITECTURE FOR MANAGEMENT OF DIGITAL FILES ACROSS DISTRIBUTED NETWORK” and filed Nov. 9, 2007, the contents of which are hereby incorporated by reference in their entirety.

FIELD OF THE INVENTION

This invention relates generally to computer-implemented processes and, more specifically, to sharing of electronic files among computer systems.

BACKGROUND OF THE INVENTION

Users of modern computing systems are increasingly finding themselves in constantly-connected, high-speed networked environments. The Web continues to be a killer application, second only to email, on the Internet. Further, customers are increasingly using more than one computing device; a customer may have a desktop computer at home, one at work, and a constantly connected “smart phone”. Due to the confluence of these two trends, file management across these devices has become a problem.

Although modern devices are easily connected, they do not provide the customer a seamless environment; the customer must manually handle many aspects of that connection. With regards to file management, customers must manually move files between their devices using some protocol like email, ftp, or by posting them on the Web. These practices lead to problems that include:

The proliferation of redundant file copies. This proliferation creates a confusing environment where the customer is unclear where the “official” or newest version of a file exists.

The creation of an error-prone environment. Some documents, such as those associated with word processing and desktop publishing, externally reference other files. Copying such a document can break these references causing errors that the customer has to handle manually. An example of such a document is a desktop publishing document that contains a reference to an image. If that image file is not transferred along with the desktop publishing file, the image will appear as a broken link.

Unnecessary complexity. Because devices tend to have their own filing system, customers must manage a different filing model on each of his devices. For example, instead of having a single “Movies” folder, he may have to deal with many “Movies” folders, which may be in different locations on each of his devices. Each device may also have its own security model, further complicating the matter.

2

That a customer has to manually move files around to ensure their accessibility on his devices is unnecessary, and is an indicator of a lack of customer-focused design in modern file systems. File systems in use today are direct offspring of systems used when graphical customer interfaces were nonexistent. Modern file system customer interfaces, such as Windows® Explorer and Mac OS X’s Finder are just now starting to provide experiences that are more in line to a customer’s workflow. Whereas, before, these interfaces were concerned with representing files with abstracted icons, the file’s actual contents are becoming paramount in how files are organized and presented.

Problems still exist with how these newer customer interfaces are implemented. They are not completely integrated with applications, suffer from performance problems, and do not generally work well outside of a device’s local file system.

There are several solutions to this problem that are in one way or another inadequate to the task:

Remote Desktop software allows a customer to remotely “see” his desktop. Remote desktop software screen-scrapes a remote machine’s screen (a “server”) and displays it on a screen local to the customer (a “client”). Remote desktop gives a customer access to not only his files, but also to his applications. However, this approach requires that the host machine be turned on and connected to the internet at all times. Consequently, this approach would not be appropriate for mobile hosts such as laptops. Remote desktop does not use the resources of a local machine. For full accessibility, the customer would have to keep all files and application on the host machine as any files stored on a client are not guaranteed to be accessible.

Distributed File Systems, like remote desktop software, place data on an always-connected host machine. Unlike remote desktop software, the host machine is not one on which the customer performs computing tasks. The host machine is used as a storage mechanism, and any computation performed on that machine serves to support its use as such. Distributed file systems generally provide the right functionality for customers to share files between their devices. However, distributed file systems are usually deployed as a shared resource; that is, other customers have access to it. Because of this sharing, a customer’s files may be buried deep in a filing structure, and it may not always be immediately evident to customers what kind of access they have to a particular file. Further, to use a distributed file system, the customer must always be connected to it. Files stored on a distributed file system are generally inaccessible if the customer’s machine is not connected to it, unless the customer has copied or moved the files to his machine’s local hard drive. However, doing so immediately creates the problem of having two filing systems for the same file, creating a mental burden on the customer.

Additionally, accessing a file located on a distributed file system tends to be slower than accessing files on the local hard drive. Modern applications are usually written to assume that the files they access are located locally, and thus are not optimized to access remote files. When these applications are used with remote files, they can lose performance by an order of magnitude. This problem can be fixed by automatically caching often-used files on the local file system, and only synchronizing them when they have been changed. However, this separate synchronization step introduces another problem: because the synchronization process can be lengthy, the customer is never entirely sure if the file he is remotely accessing is the latest version of the file, versus an earlier one that has been marked to be updated.

US 10,754,823 B2

3

Further, the directory may not reflect the existence of the file at all until synchronization finishes.

FTP is similar to a distributed file system with regards to files being hosted on a remote server. However FTP generally does manifest as a “disk drive” on the customer’s desktop; the customer must use special FTP client software to access an FTP server. It shares the same problem as distributed file systems, with the additional problem of weak integration with applications. Applications can generally write and read files directly to and from a distributed file system. This is not the case with FTP, as the customer has to manually use the client software to perform these operations as a separate task.

Email was originally invented for messaging. From the beginning, the model it employs to make files accessible remotely is necessarily inefficient. Email’s model for making files accessible is in the form of an email “attachment”. Attachments are so named because they piggy-back on a message sent from one customer to another. A customer can make a file remotely available using email by attaching the file to an email and sending it to himself. He can then retrieve the file from a remote location by accessing the message on the email server. Email used in this way is even worse than FTP as the process is even more manual: a customer must find the message containing the file before he can even access it. Further, the location in which the attachment lives is read only. If the customer, for example, were to open the file, change it, then save it back out, the results would be ambiguous to the user because the email application, not the user, specified its location. Usually, the saved file would end up buried in an email file cache in an undisclosed area of the file system.

Flash Drives and External Disk Drives, although seemingly the most “primitive” way to ensure file availability, avoid all the problems related to network latency. However, these devices must be physically connected to the computer on which the files will be accessed. These restrictions preclude the customer from employing several effective work-flows including: using more than one computer to complete a single task (the files can only be accessed on one computer) and setting up an automated backup (the computer running the backup can’t guarantee that the storage device will be connected come backup time). Further, to ensure full availability of the files, the customer must carry the device with them at all times, and must follow the associated protocols for mounting and dismounting the device.

Other problems with the prior art not described above can also be overcome using the teachings of embodiments of the present invention, as would be readily apparent to one of ordinary skill in the art after reading this disclosure.

SUMMARY OF THE INVENTION

In certain embodiments, automatic modification-triggered transfer of a file among two or more computer systems associated with a user. In some embodiments, a copy of a first file may be received, via a first application at a first computer system, from a second application at a second computer system associated with a user. The first file copy may be automatically received from the second application responsive to the user modifying a content of the first file, where the first file copy is a version of the first file that is generated from the user modifying the content of the first file. Responsive to receiving the first file copy from the second computer system, the first file copy may be automatically transferred via the first application to a third

4

computer system associated with the user to replace an older version of the first file stored on the third computer system.

In some embodiments, responsive to a user modifying a content of the file at a first client device (associated with the user), a server system may automatically receive a copy of the file from the first client device, where the file copy may be an updated version of the file that is generated from the user modifying the content of the file. After receiving metadata associated with the updated version of the file from the first client device, the server system may automatically transfer the metadata to a second client device associated with the user such that, before the file copy is transferred to the second client device, the transfer of the metadata to the second client device causes a graphical availability indication of the updated version of the file to be presented (e.g., proximate a file icon representing the file) at the second client device based on the metadata.

BRIEF DESCRIPTION OF THE DRAWING

Preferred and alternative embodiments of the present invention are described in detail below with reference to the following drawings.

FIG. 1 is a schematic view of an exemplary operating environment in which an embodiment of the invention can be implemented;

FIG. 2 is a functional block diagram of an exemplary operating environment in which an embodiment of the invention can be implemented; and

FIG. 3 is a functional block diagram illustrating file sharing and/or synchronization according to an embodiment of the invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

An embodiment of the invention leverages remote programming concepts by utilizing processes called mobile agents (sometimes referred to as mobile objects or agent objects). Generally speaking, these concepts provide the ability for an object (the mobile agent object) existing on a first (“host”) computer system to transplant itself to a second (“remote host”) computer system while preserving its current execution state. The operation of a mobile agent object is described briefly below.

The instructions of the mobile agent object, its preserved execution state, and other objects owned by the mobile agent object are packaged, or “encoded,” to generate a string of data that is configured so that the string of data can be transported by all standard means of communication over a computer network. Once transported to the remote host, the string of data is decoded to generate a computer process, still called the mobile agent object, within the remote host system. The decoded mobile agent object includes those objects encoded as described above and remains in its preserved execution state. The remote host computer system resumes execution of the mobile agent object which is now operating in the remote host environment.

While now operating in the new environment, the instructions of the mobile agent object are executed by the remote host to perform operations of any complexity, including defining, creating, and manipulating data objects and interacting with other remote host computer objects.

File transfer and/or synchronization, according to an embodiment, may be accomplished using some or all of the concepts described in commonly owned U.S. patent appli-

US 10,754,823 B2

5

cation Ser. No. 11/739,083, entitled “Electronic File Sharing,” the entirety of which is incorporated by reference as if fully set forth herein.

FIG. 1 illustrates an example of a suitable computing system environment **100** in which one or more embodiments of the invention may be implemented. The computing system environment **100** is only one example of a suitable computing environment and is not intended to suggest any limitation as to the scope of use or functionality of the invention. Neither should the computing environment **100** be interpreted as having any dependency or requirement relating to any one or combination of components illustrated in the exemplary operating environment **100**.

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

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

With reference to FIG. 1, an exemplary system for implementing the invention includes a general purpose computing device in the form of a computer **110**. Components of computer **110** may include, but are not limited to, a processing unit **120**, a system memory **130**, and a system bus **121** that couples various system components including the system memory to the processing unit **120**. The system bus **121** may be any of several types of bus structures including a memory bus or memory controller, a peripheral bus, and a local bus using any of a variety of bus architectures. By way of example, and not limitation, such architectures include Industry Standard Architecture (ISA) bus, Micro Channel Architecture (MCA) bus, Enhanced ISA (EISA) bus, Video Electronics Standards Association (VESA) local bus, and Peripheral Component Interconnect (PCI) bus also known as Mezzanine bus.

Computer **110** typically includes a variety of computer readable media. Computer readable media can be any available media that can be accessed by computer **110** and includes both volatile and nonvolatile media, removable and non-removable media. By way of example, and not limitation, computer readable media may comprise computer storage media and communication media. Computer storage media includes both volatile and nonvolatile, removable and non-removable media implemented in any method or technology for storage of information such as computer readable instructions, data structures, program modules or other data. Computer storage media includes, but is not limited to,

6

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

The system memory **130** includes computer storage media in the form of volatile and/or nonvolatile memory such as read only memory (ROM) **131** and random access memory (RAM) **132**. A basic input/output system **133** (BIOS), containing the basic routines that help to transfer information between elements within computer **110**, such as during start-up, is typically stored in ROM **131**. RAM **132** typically contains data and/or program modules that are immediately accessible to and/or presently being operated on by processing unit **120**. By way of example, and not limitation, FIG. 1 illustrates operating system **134**, application programs **135**, other program modules **136**, and program data **137**.

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

The drives and their associated computer storage media discussed above and illustrated in FIG. 1, provide storage of computer readable instructions, data structures, program modules and other data for the computer **110**. In FIG. 1, for example, hard disk drive **141** is illustrated as storing operating system **144**, application programs **145**, other program modules **146**, and program data **147**. Note that these components can either be the same as or different from operating system **134**, application programs **135**, other program modules **136**, and program data **137**. Operating system **144**, application programs **145**, other program modules **146**, and program data **147** are given different numbers here to illustrate that, at a minimum, they are different copies. A user may enter commands and information into the computer **20** through input devices such as a keyboard **162** and pointing device **161**, commonly referred to as a mouse, trackball or touch pad. Other input devices (not shown) may include a

US 10,754,823 B2

7

microphone, joystick, game pad, satellite dish, scanner, or the like. These and other input devices are often connected to the processing unit **120** through a user input interface **160** that is coupled to the system bus, but may be connected by other interface and bus structures, such as a parallel port, game port or a universal serial bus (USB). A monitor **191** or other type of display device is also connected to the system bus **121** via an interface, such as a video interface **190**. In addition to the monitor, computers may also include other peripheral output devices such as speakers **197** and printer **196**, which may be connected through an output peripheral interface **190**.

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

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

Referring now to FIG. 2, an embodiment of the present invention can be described in the context of an exemplary computer network system **200** as illustrated. System **200** includes electronic user devices **210**, **280**, such as personal computers or workstations, that are linked via a communication medium, such as a network **220** (e.g., the Internet), to an electronic device or system, such as a server **230**. The server **230** may further be coupled, or otherwise have access, to a database **240**, electronic storage **270** and a computer system **260**. Although the embodiment illustrated in FIG. 2 includes one server **230** coupled to two user devices **210**, **280** via the network **220**, it should be recognized that embodiments of the invention may be implemented using two or more such user devices coupled to one or more such servers.

In an embodiment, each of the user devices **210**, **280** and server **230** may include all or fewer than all of the features associated with the computer **110** illustrated in and discussed with reference to FIG. 1. User devices **210**, **280** include or are otherwise coupled to a computer screen or display **250**, **290**, respectively. User devices **210**, **280** can be used for various purposes including both network- and local-computing processes.

The user devices **210**, **280** are linked via the network **220** to server **230** so that computer programs, such as, for example, a browser or other applications, running on one or

8

more of the user devices **210**, **280** can cooperate in two-way communication with server **230** and one or more applications running on server **230**. Server **230** may be coupled to database **240** and/or electronic storage **270** to retrieve information therefrom and to store information thereto. Additionally, the server **230** may be coupled to the computer system **260** in a manner allowing the server to delegate certain processing functions to the computer system.

Referring now to FIG. 3, illustrated is functionality of an embodiment of the invention allowing a user (not shown) who owns or otherwise controls devices **210**, **280** to automatically maintain file synchronization between at least devices **210**, **280**, or any other user devices on which principles of the present invention are implemented. In an embodiment, an administrator (not shown) of the server **230** or other appropriate electronic device transfers a file-transfer and/or synchronization application to the user devices **210**, **280** for installation thereon. Once installed on the user devices **210**, **280**, the file-transfer application provides file-transfer clients **310**, **320** executable by the user devices **210**, **280**, respectively. Each of the file-transfer clients **310**, **320** may, but need not, include a respective mobile-agent runtime environment **330**, **340**. The mobile-agent runtime environment **330**, **340** include portions of memory of the user devices **210**, **280** dedicated to allowing a mobile object the ability to perform operations that the mobile object is programmed to carry out. Also included in the file-transfer application are user interfaces **350**, **360** that are displayable on the displays **250**, **290**, respectively. In an embodiment, the interfaces **350**, **360** allow a user to view, access and/or organize files to be synched among the various user devices.

Generally, all files that the user desires to be synched or shared may at some point be uploaded by one or more of the user devices **210**, **280** and stored in storage **270**. Upon receiving the files to be synched, the server **230** can store such files in the storage **270** and/or transfer the files to one or more of the respective hard drives of the user devices **210**, **280**, thereby enabling each respective user device to access such files. In this manner, the server **230** is operable to treat each hard drive of the respective user devices **210**, **280** as a local document cache for files received by the server. Typically, the server **230** will store one or more of the received files to the storage **270** only if the destination user device is offline or otherwise temporarily not in communication with the server **230**. Upon resuming communication with the destination user device, the server **230** will transfer the temporarily stored files to the destination device.

In operation, according to an embodiment, the user may open and modify a file **370**, such as a word-processing document or other electronic file. Alternatively, the user may create a first instance of the file **370**. The user may have previously have associated, or may now associate, the file **370** with the transfer client **310**. Upon a predetermined and user-configurable triggering event, the transfer client **310** transfers the modified file **370**, or a copy of the modified file, to the server **230**. Such a triggering event may include, but be not limited to, the user saving the file, the elapsing of a predetermined amount of time during which the file has been opened, or the re-initiation of a communication session between the device **210** and the server **230**.

The file **370** is transferred to the server **230** on which is executing a synchronization application **380**, which may include a mobile-agent runtime environment. Through user configuration, the synch application **380** monitors a set of user devices to which the file **370** should be transferred to effect file synchronization. In the illustrated embodiment, this set of user devices includes the user device **280**. The

synch application **380** polls the device **280** to determine whether the device **280** is in communication with the server **230**. If the device **280** is in communication with the server **230**, the synch application **380** transfers the file **370** to the device **280**, whereupon the transfer client **320** resident on the device **280** replaces the previous version of the file **370**, previously cached on the device **280**, with the latest version of the file **370** modified on the user device **210**. If the device **280** is not currently in communication with the server **230**, the synch application **380** may store the file **370** in the storage **270** until such time as communication between the device **280** and server **230** is reestablished. As illustrated in FIG. 3, a similar reverse-direction synchronization process may be performed by the synch application **380** and the transfer clients **310**, **320** with regard to a file **315** modified on device **280** and synchronized to device **210**.

In an embodiment, the user interfaces **350**, **360** may include a list of the customer's documents and related metadata, as well as any one-to-one or one-to-many relationships between the documents and metadata. An embodiment can always provide customers with an accurate "picture" of their document collection, regardless of whether their devices physically contain the documents. As alluded to earlier, a problem with distributed file systems and FTP is the latency between a file being put onto a file system and it showing up on a remote machine. To prevent this problem, an embodiment directory is decoupled from the movement of files. An embodiment's directory update system updates at a higher priority than the documents to be synchronized. This feature ensures that when a customer browses or searches through his set of documents, they appear even if they have not yet been cached locally on the user device. An indicator signifying a document's availability may be prominently displayed adjacent to the document's representation so that customers are aware of the document's availability.

An embodiment may include a stand-alone application that allows customers to find and manage documents associated with transfer clients **310**, **320** by visualizing relationships between documents and their metadata. It allows customers to tag documents with any number of identifiers. Customers can relate both documents and tags with each other in any number of user-specified one-to-one and one-to-many relationships, and an embodiment provides a user interface to browse and search on these relationships. To mitigate the customers' learning curve, an embodiment can implement relationships common to contemporary file systems, including a folder hierarchy. In addition to this, an embodiment provides direct support for methods that the customer uses to organize documents by manifesting them as user interface idioms. This is unlike conventional document filing systems which require the customer to work within a strict folder metaphor for organization.

Some alternate methods that an embodiment supports for organizing documents include:

Allow customers to organize their documents by application. Many times customers remember the application used to create a document instead of the document's name or its location in a hierarchy.

Allow customers to organize their documents by most recent access. Customers are likely to access a document they've accessed in the near past. Usually, such documents are part of a task that the customer is actively working.

Allow customers to organize their documents by project or subproject.

Allow customers to organize their documents by people. Many times, especially in the context of a collabora-

tion, a document is directly related to one or more people other than the customer.

Allow the customer to organize their document by process stage. Documents may represent one or more stages of a process. Customers need a method for organizing documents by process stage, and a mechanism for moving the document through a set of predefined stages.

Allow customers to organize their documents by any of the aforementioned methods concurrently. These organization methods are not mutually exclusive.

An embodiment presents an interface that allows a customer to locate one or more documents associated with the transfer clients **310**, **320** and open such document into a separate software application. Since this interface is intended to be used from within the separate application, that application may need to know how to invoke such interface. Advantageously, this invocation behavior can be provided to the application using the application's plug-in API.

An embodiment presents an interface that allows a customer to synchronize a currently opened document according to processes described elsewhere herein. This interface can be invoked within an application and can be made available to the application in the manner described above in connection with the application's plug-in API.

Some files associated with the transfer clients **310**, **320** are dependent on other files associated with the transfer clients **310**, **320**. For example, a desktop publishing document may include images that are stored in files separate from the main document. Previous file-synching solutions treat these files as separate. Because of this, for example, a document synchronized from the device **210** to the device **280** may be opened by the user of the device **280** before the image files have been fully transferred to the device **280**. This causes the document to fail to open, or break, since the image files don't exist or are incomplete. An embodiment prevents this by: (1) always ensuring the file catalog (e.g., the stand-alone application that allows customers to find and manage documents associated with transfer clients **310**, **320**, as discussed above herein) is synchronized before any file data is synchronized, and (2) pausing any file access by any program until the file contents have been fully synchronized. In such an embodiment, if a user attempts, using a software program, to open a file whose related files haven't yet finished transferring to the local (hard drive) cache, if that software attempts to open the related files, the software program is blocked by an embodiment until the requested files are downloaded and ready to access.

Other file sending and synchronizing software requires the user to upload their data to a storage device owned by the operator of the service. An embodiment treats storage as a participant in the synchronization process; this means that the user can choose the service or device where their files will be stored. The file transfer/synching is abstracted from the storage system allowing any storage to be used. An embodiment treats storage like any other synch target, such as a desktop computer, or a cell phone. As such, any device owned or otherwise controlled by the user and running a synch application, such as synch application **380**, as provided in an embodiment of the invention can perform the storage and/or synching functions described elsewhere herein. That is, the user device **280** or user device **210**, rather than the server **230**, may perform such functions.

While a preferred embodiment of the invention has been illustrated and described, as noted above, many changes can be made without departing from the spirit and scope of the invention. For example, as an alternative to the approach

US 10,754,823 B2

11

described with reference to FIG. 3, wherein the transfer clients 310, 320 function to “push” modified or created files to the synch application 380, the synch application 380 may instead function to periodically “pull” or otherwise actively retrieve such files from the transfer clients 310, 320. Instead, the invention should be determined entirely by reference to the claims that follow.

What is claimed is:

1. A system comprising:

a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to:

receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being an updated version of the first file that is generated from the user modifying the content of the first file;

receive, over a network, from the first client device, first metadata associated with the updated version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;

automatically transfer, based on the first priority being greater than the second priority, the first metadata over a network to a second client device associated with the user such that the first metadata is transferred to the second client device before the copy of the first file is transferred to the second client device;

(i) the transfer of the first metadata to the second client device causes a graphical availability indication of the updated version of the first file to be presented at the second client device based on the first metadata, and

(ii) the graphical availability indication is presented proximate a file icon representing the first file on a user interface of the second client device, and

wherein the graphical availability indication indicates that the updated version of the first file generated from the user modifying the content of the first file is available to be downloaded from the server system to the second client device; and

subsequent to the transfer of the first metadata to the second client device, transfer the copy of the first file to the second client device,

wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.

2. The system of claim 1, wherein, before the copy of the first file is transferred to the second client device, the transfer of the first metadata to the second client device causes a file representation of the first file presented on the user interface of the second client device to be updated based on the first metadata.

12

3. The system of claim 1, wherein the computer program instructions, when executed, cause the server system to:

receive a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the second client device responsive to the user modifying a content of the second file stored on the second client device, the copy of the second file being an updated version of the second file that is generated from the user modifying the content of the second file;

determine that the server system is in communication with the first client device associated with the user; and

automatically transfer the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client device and (ii) receiving the copy of the second file from the second client device.

4. The system of claim 1, wherein the copy of the first file is automatically received from the first client device responsive to (i) a push request of the first client device and (ii) the user modifying the content of the first file stored on the first client device.

5. The system of claim 1, wherein the copy of the first file is automatically received from a first application at the first client device, and wherein the first application comprises a runtime environment for one or more mobile-agent objects.

6. The system of claim 5, wherein the first application is configured to create a first mobile object, and wherein the first mobile object is configured to create a proxy object at the server system.

7. The system of claim 6, wherein the first mobile object is configured to provide the copy of the first file to the proxy object, and wherein the proxy object is configured to store the copy of the first file on a memory device associated with the server system.

8. A method being implemented by a server system comprising one or more processors executing computer program instructions that, when executed, perform the method, the method comprising:

receiving, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being an updated version of the first file that is generated from the user modifying the content of the first file;

receiving, over a network, from the first client device, first metadata associated with the updated version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;

automatically transferring, based on the first priority being greater than the second priority, the first metadata over a network to a second client device associated with the user such that the first metadata is transferred to the second client device before the copy of the first file is transferred to the second client device,

wherein, before the copy of the first file is transferred to the second client device:

(i) the transfer of the first metadata to the second client device causes a graphical availability indi-

US 10,754,823 B2

13

cation of the updated version of the first file to be presented at the second client device based on the first metadata, and

(ii) the graphical availability indication is presented proximate a graphical file representation of the first file on a user interface of the second client device, and

wherein the graphical availability indication indicates that the updated version of the first file generated from the user modifying the content of the first file is available to be downloaded from the server system to the second client device; and

subsequent to the transfer of the first metadata to the second client device, transferring the copy of the first file to the second client device,

wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.

9. The method of claim 8, wherein, before the copy of the first file is transferred to the second client device, the transfer of the first metadata to the second client device causes a file representation of the first file presented on the user interface of the second client device to be updated based on the first metadata.

10. The method of claim 8, further comprising:

receiving a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the second client device responsive to the user modifying a content of the second file stored on the second client device, the copy of the second file being an updated version of the second file that is generated from the user modifying the content of the second file;

determining that the server system is in communication with the first client device associated with the user; and automatically transferring the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client device and (ii) receiving the copy of the second file from the second client device.

11. The method of claim 8, wherein the copy of the first file is automatically received from the first client device responsive to (i) a push request of the first client device and (ii) the user modifying the content of the first file stored on the first client device.

12. The method of claim 8, wherein the copy of the first file is automatically received from a first application at the first client device, wherein the first application comprises a runtime environment for one or more mobile-agent objects, wherein the first application is configured to create a first mobile object, wherein the first mobile object is configured to create a proxy object at the server system, wherein the first mobile object is configured to provide the copy of the first file to the proxy object, and wherein the proxy object is configured to store the copy of the first file on a memory device associated with the server system.

13. One or more non-transitory machine-readable media storing instructions that, when executed by one or more processors of a server system, cause operations comprising:

14

receiving, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being an updated version of the first file that is generated from the user modifying the content of the first file;

receiving, over a network, from the first client device, first metadata associated with the updated version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;

automatically transferring, based on the first priority being greater than the second priority, the first metadata over a network to a second client device associated with the user such that the first metadata is transferred to the second client device before the copy of the first file is transferred to the second client device,

wherein, before the copy of the first file is transferred to the second client device:

(i) the transfer of the first metadata to the second client device causes an availability indication of the updated version of the first file to be presented at the second client device based on the first metadata, and

(ii) the availability indication is presented proximate a graphical file representation of the first file on a user interface of the second client device, and

wherein the availability indication indicates that the updated version of the first file generated from the user modifying the content of the first file is available to be downloaded from the server system to the second client device; and

subsequent to the transfer of the first metadata to the second client device, transferring the copy of the first file to the second client device,

wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.

14. The media of claim 13, wherein, before the copy of the first file is transferred to the second client device, the transfer of the first metadata to the second client device causes a file representation of the first file presented on the user interface of the second client device to be updated based on the first metadata.

15. The media of claim 13, the operations further comprising:

receiving a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the second client device responsive to the user modifying a content of the second file stored on the second client device, the copy of the second file being an updated version of the second file that is generated from the user modifying the content of the second file;

determining that the server system is in communication with the first client device associated with the user; and automatically transferring the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the

server system is in communication with the first client device and (ii) receiving the copy of the second file from the second client device.

16. The media of claim 13, wherein the copy of the first file is automatically received from the first client device responsive to (i) a push request of the first client device and (ii) the user modifying the content of the first file stored on the first client device. 5

17. The media of claim 13, wherein the copy of the first file is automatically received from a first application at the first client device, wherein the first application comprises a runtime environment for one or more mobile-agent objects, wherein the first application is configured to create a first mobile object, wherein the first mobile object is configured to create a proxy object at the server system, wherein the first mobile object is configured to provide the copy of the first file to the proxy object, and wherein the proxy object is configured to store the copy of the first file on a memory device associated with the server system. 10 15

* * * * *

EXHIBIT 7



US011003622B2

(12) **United States Patent**
Manzano

(10) **Patent No.:** **US 11,003,622 B2**

(45) **Date of Patent:** ***May 11, 2021**

(54) **ARCHITECTURE FOR MANAGEMENT OF DIGITAL FILES ACROSS DISTRIBUTED NETWORK**

(56) **References Cited**

U.S. PATENT DOCUMENTS

(71) Applicant: **TOPIA TECHNOLOGY, INC.**,
Tacoma, WA (US)

5,600,834 A 2/1997 Howard
5,675,802 A 10/1997 Allen
(Continued)

(72) Inventor: **Michael R. Manzano**, Seattle, WA (US)

FOREIGN PATENT DOCUMENTS

(73) Assignee: **TOPIA TECHNOLOGY, INC.**,
Tacoma, WA (US)

EP 1 130 511 A2 9/2001
WO WO 98/56149 A1 12/1998
WO WO 2007/047302 A2 4/2007

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

OTHER PUBLICATIONS

This patent is subject to a terminal disclaimer.

FolderShare—Secure Remote Access VPN Solution; Document Management & Real-time File Mirroring Solution; < <https://web.archive.org/web/20040804020435/http://www.foldershare.com:80/>>; Aug. 4, 2004.

(21) Appl. No.: **16/361,641**

(Continued)

(22) Filed: **Mar. 22, 2019**

Primary Examiner — Srirama Channavajjala

(65) **Prior Publication Data**

US 2019/0220442 A1 Jul. 18, 2019

(74) *Attorney, Agent, or Firm* — Pillsbury Winthrop Shaw Pittman LLP

Related U.S. Application Data

(57) **ABSTRACT**

(63) Continuation of application No. 16/017,348, filed on Jun. 25, 2018, now Pat. No. 10,289,607, which is a (Continued)

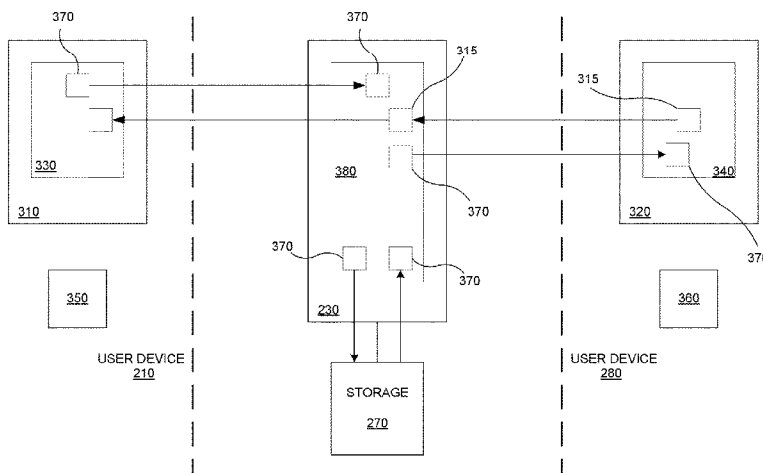
In certain embodiments, automatic modification-triggered transfer of a file among two or more computer systems associated with a user. In some embodiments, a copy of a first file may be received, via a first application at a first computer system, from a second application at a second computer system associated with a user. The first file copy may be automatically received from the second application responsive to the user modifying a content of the first file, where the first file copy is a version of the first file that is generated from the user modifying the content of the first file. Responsive to receiving the first file copy from the second computer system, the first file copy may be automatically transferred via the first application to a third computer system associated with the user to replace an older version of the first file stored on the third computer system.

(51) **Int. Cl.**
G06F 16/00 (2019.01)
G06F 16/11 (2019.01)
(Continued)

(52) **U.S. Cl.**
CPC **G06F 16/122** (2019.01); **G06F 15/16** (2013.01); **G06F 16/00** (2019.01); **G06F 16/10** (2019.01);
(Continued)

(58) **Field of Classification Search**
CPC G06F 16/00; G06F 16/178; G06F 16/182; G06F 16/10; G06F 16/1873;
(Continued)

17 Claims, 3 Drawing Sheets



US 11,003,622 B2

Page 2

Related U.S. Application Data					
	continuation of application No. 14/860,289, filed on Sep. 21, 2015, now Pat. No. 10,067,942, which is a continuation of application No. 12/267,852, filed on Nov. 10, 2008, now Pat. No. 9,143,561.	7,089,307 B2	8/2006	Zintel et al.	
		7,136,934 B2	11/2006	Carter et al.	
		7,149,760 B1 *	12/2006	Breuer	G06F 16/93
		7,155,488 B1	12/2006	Lunsford et al.	
		7,162,501 B2 *	1/2007	Kupkova	G06F 16/93
		7,224,973 B2	5/2007	Tsutazawa et al.	
		7,243,163 B1	7/2007	Friend et al.	
		7,260,646 B1	8/2007	Stefanik et al.	
(60)	Provisional application No. 60/986,896, filed on Nov. 9, 2007.	7,263,493 B1 *	8/2007	Provost	G06F 19/328 705/2
		7,269,433 B2	9/2007	Vargas et al.	
(51)	Int. Cl.	7,290,244 B2	10/2007	Peck et al.	
	G06F 15/16 (2006.01)	7,325,038 B1	1/2008	Wang	
	G06F 16/13 (2019.01)	7,340,534 B2	3/2008	Cameron et al.	
	G06F 16/14 (2019.01)	7,398,327 B2	7/2008	Lee	
	G06F 16/176 (2019.01)	7,415,588 B2	8/2008	Hong et al.	
	G06F 16/178 (2019.01)	7,415,615 B2	8/2008	Skygebjjer	
	G06F 16/18 (2019.01)	7,457,631 B2	11/2008	Yach et al.	
	G06F 16/182 (2019.01)	7,467,353 B2	12/2008	Kurlander et al.	
	G06F 16/10 (2019.01)	7,483,925 B2	1/2009	Koskimies et al.	
	H04L 29/08 (2006.01)	7,526,575 B2	4/2009	Rabbers et al.	
		7,529,778 B1	5/2009	Dewey	
		7,574,711 B2	8/2009	Zondervan et al.	
(52)	U.S. Cl.	7,584,186 B2	9/2009	Chen et al.	
	CPC G06F 16/128 (2019.01); G06F 16/13 (2019.01); G06F 16/14 (2019.01); G06F 16/176 (2019.01); G06F 16/178 (2019.01); G06F 16/182 (2019.01); G06F 16/1873 (2019.01); H04L 67/1095 (2013.01)	7,587,446 B1	9/2009	Onyon et al.	
		7,613,773 B2	11/2009	Watt	
		7,639,116 B2	12/2009	Saunders	
		7,657,271 B2	2/2010	Kim	
		7,680,838 B1	3/2010	Shaw	
		7,680,885 B2	3/2010	Schauser et al.	
		7,707,165 B1	4/2010	Jiang	
(58)	Field of Classification Search	7,752,166 B2	7/2010	Quinlan et al.	
	CPC G06F 16/1756; G06F 16/1824; G06F 16/1844; G06F 16/23; G06F 8/71; G06F 11/1435; G06F 11/1448; G06F 16/273; G06F 16/275; G06F 16/128; G06F 16/122; G06F 16/16; G06F 16/13-14; G06F 16/176	7,761,414 B2	7/2010	Freedman	
	See application file for complete search history.	7,788,303 B2	8/2010	Mikesell et al.	
		7,796,779 B1	9/2010	Strong et al.	
		7,885,925 B1	2/2011	Strong et al.	
		7,895,334 B1	2/2011	Tu et al.	
		7,987,420 B1	7/2011	Kloba et al.	
		8,009,966 B2	8/2011	Bloom et al.	
		8,019,900 B1	9/2011	Sekar et al.	
		8,112,549 B2	2/2012	Srinivasan et al.	
		8,208,792 B2	6/2012	Morioka	
		8,244,288 B2	8/2012	Chipchase	
(56)	References Cited	8,321,534 B1	11/2012	Roskind et al.	
	U.S. PATENT DOCUMENTS	8,370,423 B2	2/2013	Ozzie et al.	
		8,386,558 B2	2/2013	Schleifer et al.	
		8,504,519 B1	8/2013	Sachs	
		8,565,729 B2	10/2013	Moseler et al.	
		8,595,182 B1	11/2013	Nelson	
		8,874,534 B2	10/2014	March	
		9,143,561 B2	9/2015	Manzano	
		10,067,942 B2	9/2018	Manzano	
		10,289,607 B2	5/2019	Manzano	
		2002/0026478 A1	2/2002	Rodgers et al.	
		2002/0035697 A1	3/2002	McCurdy	
		2002/0055942 A1	5/2002	Reynolds	
		2002/0087588 A1	7/2002	McBride et al.	
		2002/0143795 A1	10/2002	Fletcher	
		2002/0184318 A1	12/2002	Pineau	
		2002/0194382 A1 *	12/2002	Kausik	G06F 9/54 709/246
		2003/0028514 A1	2/2003	Lord	
		2003/0028542 A1	2/2003	Muttik et al.	
		2003/0038842 A1	2/2003	Peck et al.	
		2003/0074376 A1	4/2003	Benayoun	
		2003/0078946 A1	4/2003	Costello	
		2003/0115547 A1 *	6/2003	Ohwada	G06F 16/93 715/229
		2003/0125057 A1	7/2003	Pesola	
		2003/0135565 A1	7/2003	Estrada	
		2003/0233241 A1	12/2003	Marsh	
		2004/0003013 A1	1/2004	Coulthard	
		2004/0031027 A1	2/2004	Hiltgen	
		2004/0049345 A1	3/2004	McDonough et al.	
		2004/0088348 A1 *	5/2004	Yeager	H04L 67/1068 709/202
		6,829,622 B2	12/2004	Beyda	
		6,874,037 B1	3/2005	Abram et al.	
		6,931,454 B2	8/2005	Deshpande et al.	
		6,990,522 B2	1/2006	Wu	
		7,024,428 B1	4/2006	Huang et al.	
		7,035,847 B2	4/2006	Brown et al.	
		7,051,364 B1 *	5/2006	Tackman	G06Q 50/188 705/80
		7,054,594 B2	5/2006	Bloch et al.	
		7,058,667 B2 *	6/2006	Goldick	G06F 16/10
		7,065,658 B1	6/2006	Baraban et al.	

US 11,003,622 B2

Page 3

(56)

References Cited

U.S. PATENT DOCUMENTS

2004/0158817	A1	8/2004	Okachi	
2004/0172424	A1	9/2004	Edelstein et al.	
2005/0027757	A1	2/2005	Kiessig	
2005/0091316	A1	4/2005	Ponce et al.	
2005/0097225	A1	5/2005	Glatt et al.	
2005/0165722	A1	7/2005	Cannon	
2005/0177602	A1	8/2005	Kaler	
2005/0188051	A1*	8/2005	Sneh	G06F 9/44526 709/213
2005/0192966	A1	9/2005	Hilbert	
2005/0210371	A1*	9/2005	Pollock	G06F 17/212 715/212
2005/0220080	A1	10/2005	Ronkainen et al.	
2005/0223047	A1	10/2005	Shah	
2006/0004698	A1	1/2006	Pyhalammi	
2006/0010150	A1	1/2006	Shaath et al.	
2006/0026567	A1	2/2006	McVoy	
2006/0058907	A1	3/2006	Suderman	
2006/0074985	A1	4/2006	Wolfish	
2006/0101064	A1	5/2006	Strong et al.	
2006/0129627	A1	6/2006	Phillips	
2006/0136446	A1	6/2006	Hughes	
2006/0143129	A1	6/2006	Holm	
2006/0168118	A1	7/2006	Godlin	
2006/0189348	A1	8/2006	Montulli et al.	
2006/0224626	A1	10/2006	Lakshminath	
2007/0014314	A1	1/2007	O'Neil	
2007/0016629	A1	1/2007	Reinsch	
2007/0022158	A1	1/2007	Vasa	
2007/0027936	A1	2/2007	Stakutis	
2007/0100913	A1	5/2007	Sumner et al.	
2007/0124334	A1	5/2007	Pepin	
2007/0174246	A1*	7/2007	Sigurdsson	G06F 16/1787
2007/0180084	A1*	8/2007	Mohanty	G06F 11/1464 709/223
2007/0191057	A1	8/2007	Kamada	
2007/0203927	A1	8/2007	Cave	
2007/0238440	A1	10/2007	Sengupta et al.	
2007/0239725	A1*	10/2007	Bhat	H04L 67/1095
2008/0005114	A1	1/2008	Li	
2008/0005195	A1	1/2008	Li	
2008/0005280	A1	1/2008	Adams	
2008/0086494	A1	4/2008	Heller et al.	
2008/0168526	A1	7/2008	Robbin et al.	
2008/0288578	A1	11/2008	Silfverberg	
2009/0013009	A1	1/2009	Nakayama	
2009/0024922	A1	1/2009	Markowitz et al.	
2009/0037718	A1	2/2009	Ganesh	
2009/0063711	A1	3/2009	Finkelstein	
2009/0282050	A1	11/2009	Thomas et al.	
2012/0192064	A1	7/2012	Antebi	
2013/0226871	A1	8/2013	Sarnowski	
2013/0226872	A1	8/2013	Barefoot	
2014/0053227	A1	2/2014	Ruppin	
2016/0125058	A1	5/2016	Jain	

OTHER PUBLICATIONS

FolderShare; Your Files Anywhere; <<https://web.archive.org/web/20030808183932/http://www.foldershare.com:80/>>; Aug. 8, 2003.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20040814015727/http://www.foldershare.com:80/>>; Aug. 14, 2004.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20040820052105/http://www.foldershare.com:80/>>; Aug. 20, 2004.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20041211020957/http://foldershare.com:80/>>; Dec. 11, 2004.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20041217041726/http://foldershare.com:80/>>; Dec. 17, 2004.

FolderShare; Your Smart File Transfer Solution; <<https://web.archive.org/web/20031220151508/http://www.foldershare.com:80/>>; Dec. 20, 2003.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20041230211050/http://www.foldershare.com:80/>>; Dec. 30, 2004.

FolderShare—Secure Remote Access VPN Solution; Document Management & Real-time File Mirroring Solution; <<https://web.archive.org/web/20040701113739/http://foldershare.com:80/>>; Jul. 1, 2004.

FolderShare—Secure Remote Access VPN Solution; Document Management & Real-time File Mirroring Solution; <<https://web.archive.org/web/20040711062548/http://www.foldershare.com:80/>>; Jul. 11, 2004.

FolderShare; Your Files Anywhere; <<https://web.archive.org/web/20030722054342/http://foldershare.com:80/>>; Jul. 22, 2003.

FolderShare—Secure Remote Access VPN Solution; Document Management & Real-time File Mirroring Solution; <<https://web.archive.org/web/20040730030655/http://www.foldershare.com:80/>>; Jul. 30, 2004.

FolderShare—Secure Remote Access VPN Solution; Document Management & Real-time File Mirroring Solution; <<https://web.archive.org/web/20040603205113/http://www.foldershare.com:80/>>; Jun. 3, 2004.

FolderShare—Secure Remote Access VPN Solution; Document Management & Real-time File Mirroring Solution; <<https://web.archive.org/web/20040613161906/http://www.foldershare.com:80/>>; Jun. 13, 2004.

FolderShare—Secure Remote Access VPN Solution; Document Management & Real-time File Mirroring Solution; <<https://web.archive.org/web/20040629075057/http://www.foldershare.com:80/>>; Jun. 29, 2004.

FolderShare—Secure Remote Access VPN Solution; Your Smart File Transfer & Real-time File Mirroring Solution; <<https://web.archive.org/web/20040316235151/http://foldershare.com:80/>>; Mar. 16, 2004.

FolderShare—Secure Remote Access VPN Solution; Your Smart File Transfer & Real-time File Mirroring Solution; <<https://web.archive.org/web/20040325034239/http://www.foldershare.com:80/>>; Mar. 25, 2004.

FolderShare—Secure Remote Access VPN Solution; Document Management & Real-time File Mirroring Solution; <<https://web.archive.org/web/20040512211417/http://www.foldershare.com:80/>>; May 12, 2004.

FolderShare; Your Files Anywhere; <<https://web.archive.org/web/20030531180252/http://www.foldershare.com:80/>>; May 31, 2003.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20041104031510/http://www.foldershare.com:80/>>; Nov. 4, 2004.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20041117092357/http://www.foldershare.com:80/>>; Nov. 17, 2004.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/2004123085254/http://www.foldershare.com:80/>>; Nov. 23, 2004.

FolderShare; Your Files Anywhere; <<https://web.archive.org/web/20031128143634/http://foldershare.com:80/>>; Nov. 28, 2003.

FolderShare; Your Files Anywhere; <<https://web.archive.org/web/20031001071631/http://foldershare.com:80/>>; Oct. 1, 2003.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20041012083127/http://www.foldershare.com:80/>>; Oct. 12, 2004.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20041029085820/http://www.foldershare.com:80/>>; Oct. 29, 2004.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/2004091034646/http://www.foldershare.com:80/>>; Sep. 1, 2004.

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20040909075254/http://www.foldershare.com:80/>>; Sep. 9, 2004.

FolderShare; Your Files Anywhere; <<https://web.archive.org/web/20030920051943/http://www.foldershare.com:80/>>; Sep. 20, 2003.

US 11,003,622 B2

Page 4

(56)

References Cited

OTHER PUBLICATIONS

FolderShare—Secure Remote Access VPN Solution; Need your files on more than one computer?; <<https://web.archive.org/web/20040924032146/http://www.foldershare.com:80/>>; Sep. 24, 2004.

Marshall, M., “The Y Combinator List,” Venture Beat, Aug. 2007, Retrieved from the Internet: URL: <<https://venturebeat.com/2007/08/16/the-y-combinator-list/>>, 4 pages.

Jarvis, A., “Dropbox pitch deck to raise seed capital investment.” Medium, Mar. 2018, Retrieved from the Internet: URL: <<https://medium.com/@adjblog/dropbox-pitch-deck-to-raise-seed-capital-investment-6a6cd6517e56>>, 12 pages.

* cited by examiner

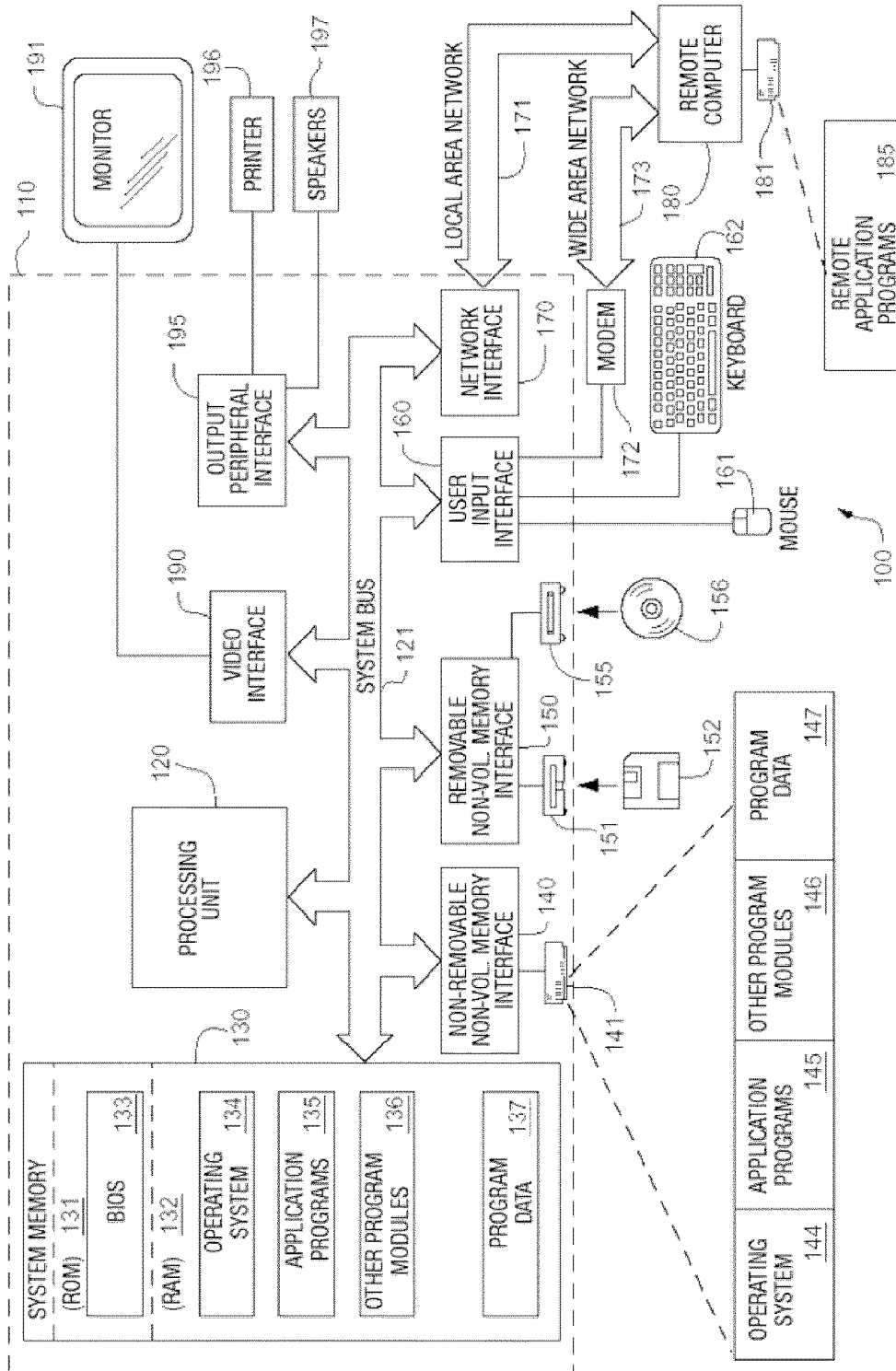


FIG. 1

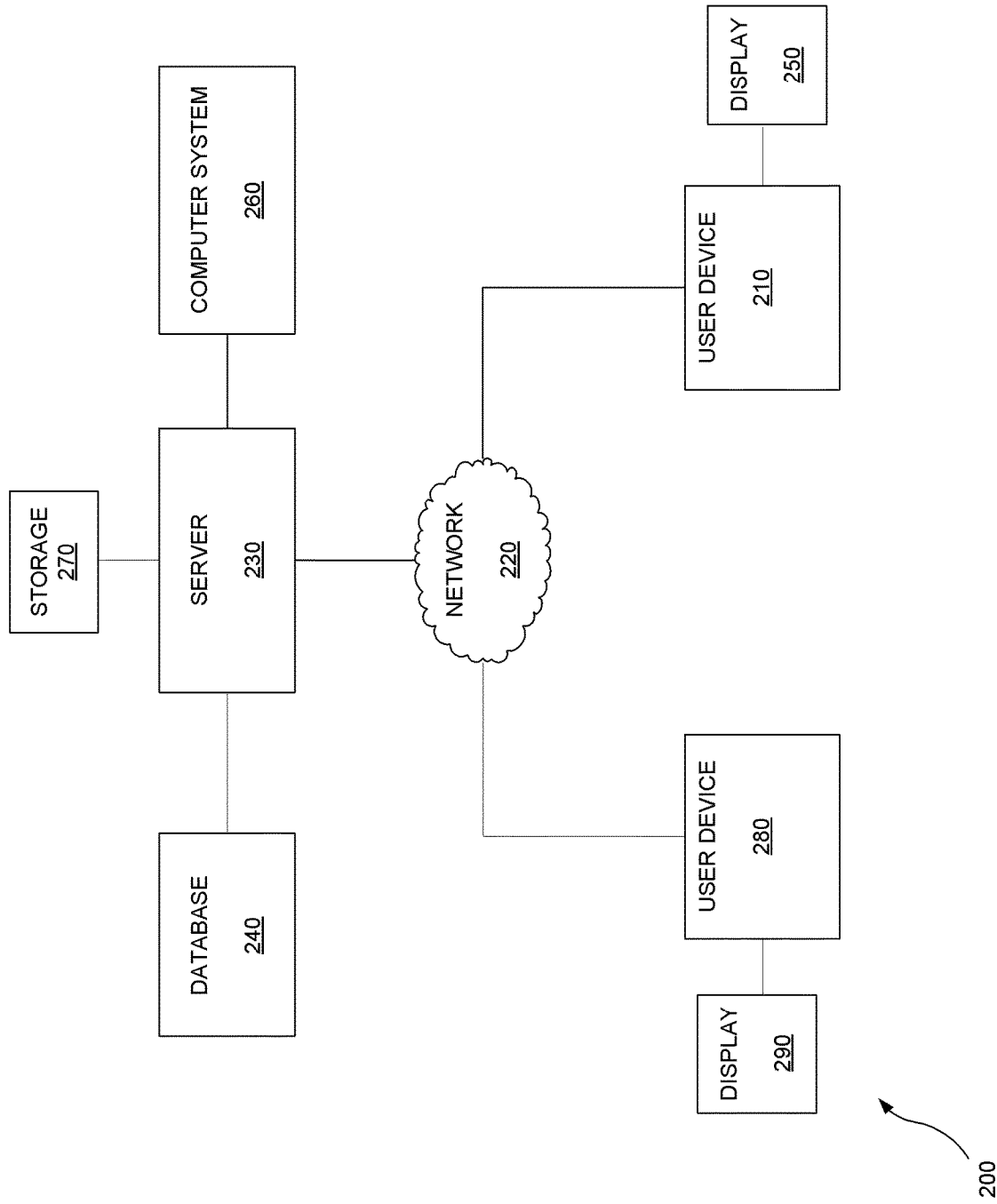


FIG. 2

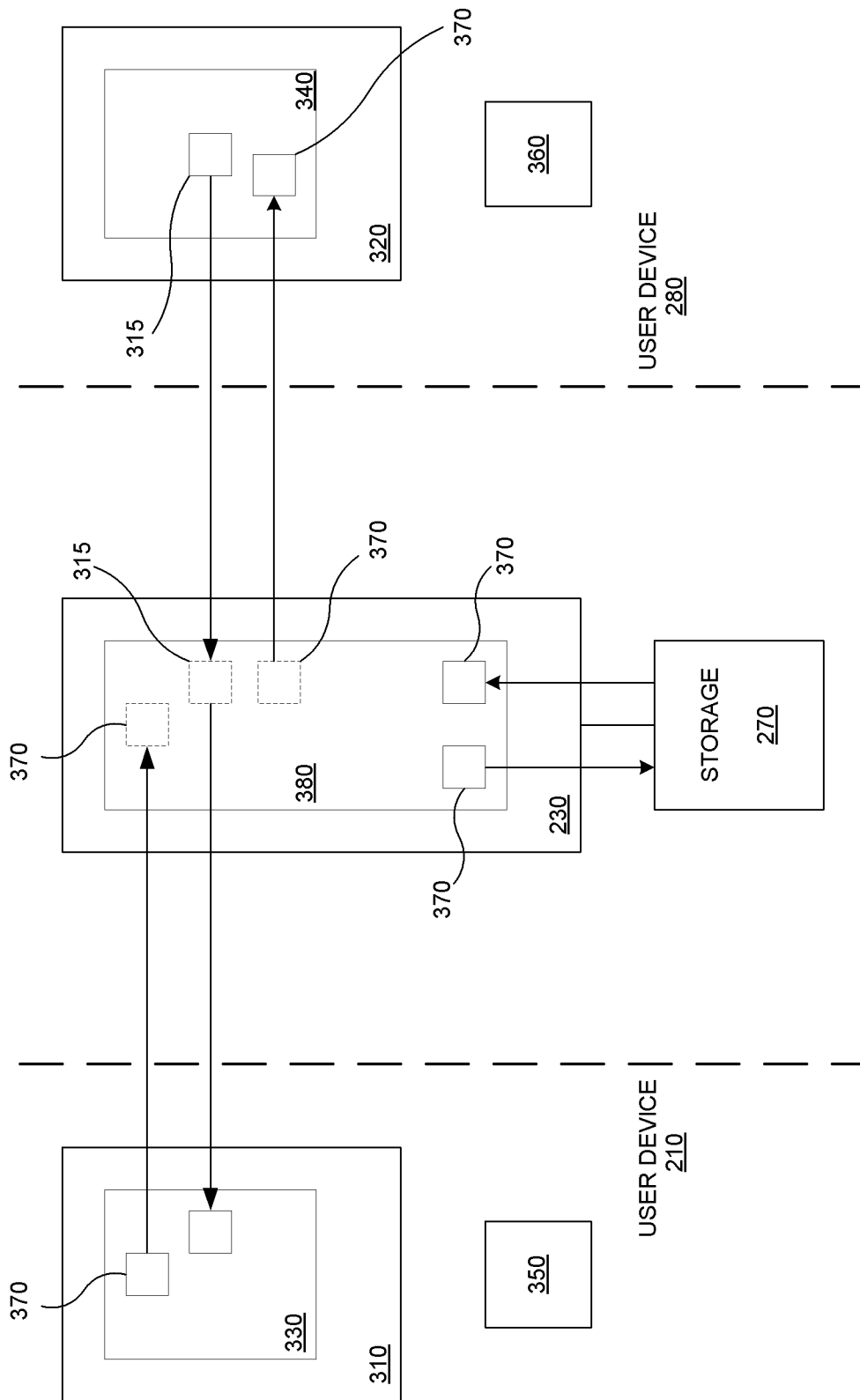


FIG. 3

US 11,003,622 B2

1

ARCHITECTURE FOR MANAGEMENT OF DIGITAL FILES ACROSS DISTRIBUTED NETWORK

CROSS-REFERENCE TO RELATED APPLICATIONS

This application is a continuation of U.S. patent application Ser. No. 16/017,348, filed Jun. 25, 2018, which is a continuation of U.S. patent application Ser. No. 14/860,289, filed Sep. 21, 2015, now U.S. Pat. No. 10,067,942, which is a continuation of U.S. patent application Ser. No. 12/267,852, filed Nov. 10, 2008, now U.S. Pat. No. 9,143,561, which claims priority to U.S. Provisional Application No. 60/986,896 entitled “ARCHITECTURE FOR MANAGEMENT OF DIGITAL FILES ACROSS DISTRIBUTED NETWORK” and filed Nov. 9, 2007, the contents of which are hereby incorporated by reference in their entirety.

FIELD OF THE INVENTION

This invention relates generally to computer-implemented processes and, more specifically, to sharing of electronic files among computer systems.

BACKGROUND OF THE INVENTION

Users of modern computing systems are increasingly finding themselves in constantly-connected, high-speed networked environments. The Web continues to be a killer application, second only to email, on the Internet. Further, customers are increasingly using more than one computing device; a customer may have a desktop computer at home, one at work, and a constantly connected “smart phone”. Due to the confluence of these two trends, file management across these devices has become a problem.

Although modern devices are easily connected, they do not provide the customer a seamless environment; the customer must manually handle many aspects of that connection. With regards to file management, customers must manually move files between their devices using some protocol like email, ftp, or by posting them on the Web. These practices lead to problems that include:

The proliferation of redundant file copies. This proliferation creates a confusing environment where the customer is unclear where the “official” or newest version of a file exists.

The creation of an error-prone environment. Some documents, such as those associated with word processing and desktop publishing, externally reference other files. Copying such a document can break these references causing errors that the customer has to handle manually. An example of such a document is a desktop publishing document that contains a reference to an image. If that image file is not transferred along with the desktop publishing file, the image will appear as a broken link.

Unnecessary complexity. Because devices tend to have their own filing system, customers must manage a different filing model on each of his devices. For example, instead of having a single “Movies” folder, he may have to deal with many “Movies” folders, which may be in different locations on each of his devices. Each device may also have its own security model, further complicating the matter.

That a customer has to manually move files around to ensure their accessibility on his devices is unnecessary, and

2

is an indicator of a lack of customer-focused design in modern file systems. File systems in use today are direct offspring of systems used when graphical customer interfaces were nonexistent. Modern file system customer interfaces, such as Windows® Explorer and Mac OS X’s Finder are just now starting to provide experiences that are more in line to a customer’s workflow. Whereas, before, these interfaces were concerned with representing files with abstracted icons, the file’s actual contents are becoming paramount in how files are organized and presented.

Problems still exist with how these newer customer interfaces are implemented. They are not completely integrated with applications, suffer from performance problems, and do not generally work well outside of a device’s local file system.

There are several solutions to this problem that are in one way or another inadequate to the task:

Remote Desktop software allows a customer to remotely “see” his desktop. Remote desktop software screen-scrapes a remote machine’s screen (a “server”) and displays it on a screen local to the customer (a “client”). Remote desktop gives a customer access to not only his files, but also to his applications. However, this approach requires that the host machine be turned on and connected to the internet at all times. Consequently, this approach would not be appropriate for mobile hosts such as laptops. Remote desktop does not use the resources of a local machine. For full accessibility, the customer would have to keep all files and application on the host machine as any files stored on a client are not guaranteed to be accessible.

Distributed File Systems, like remote desktop software, place data on an always-connected host machine. Unlike remote desktop software, the host machine is not one on which the customer performs computing tasks. The host machine is used as a storage mechanism, and any computation performed on that machine serves to support its use as such. Distributed file systems generally provide the right functionality for customers to share files between their devices. However, distributed file systems are usually deployed as a shared resource; that is, other customers have access to it. Because of this sharing, a customer’s files may be buried deep in a filing structure, and it may not always be immediately evident to customers what kind of access they have to a particular file. Further, to use a distributed file system, the customer must always be connected to it. Files stored on a distributed file system are generally inaccessible if the customer’s machine is not connected to it, unless the customer has copied or moved the files to his machine’s local hard drive. However, doing so immediately creates the problem of having two filing systems for the same file, creating a mental burden on the customer.

Additionally, accessing a file located on a distributed file system tends to be slower than accessing files on the local hard drive. Modern applications are usually written to assume that the files they access are located locally, and thus are not optimized to access remote files. When these applications are used with remote files, they can lose performance by an order of magnitude. This problem can be fixed by automatically caching often-used files on the local file system, and only synchronizing them when they have been changed. However, this separate synchronization step introduces another problem: because the synchronization process can be lengthy, the customer is never entirely sure if the file he is remotely accessing is the latest version of the file, versus an earlier one that has been marked to be updated. Further, the directory may not reflect the existence of the file at all until synchronization finishes.

US 11,003,622 B2

3

FTP is similar to a distributed file system with regards to files being hosted on a remote server. However FTP generally does manifest as a “disk drive” on the customer’s desktop; the customer must use special FTP client software to access an FTP server. It shares the same problem as distributed file systems, with the additional problem of weak integration with applications. Applications can generally write and read files directly to and from a distributed file system. This is not the case with FTP, as the customer has to manually use the client software to perform these operations as a separate task.

Email was originally invented for messaging. From the beginning, the model it employs to make files accessible remotely is necessarily inefficient. Email’s model for making files accessible is in the form of an email “attachment”. Attachments are so named because they piggy-back on a message sent from one customer to another. A customer can make a file remotely available using email by attaching the file to an email and sending it to himself. He can then retrieve the file from a remote location by accessing the message on the email server. Email used in this way is even worse than FTP as the process is even more manual: a customer must find the message containing the file before he can even access it. Further, the location in which the attachment lives is read only. If the customer, for example, were to open the file, change it, then save it back out, the results would be ambiguous to the user because the email application, not the user, specified its location. Usually, the saved file would end up buried in an email file cache in an undisclosed area of the file system.

Flash Drives and External Disk Drives, although seemingly the most “primitive” way to ensure file availability, avoid all the problems related to network latency. However, these devices must be physically connected to the computer on which the files will be accessed. These restrictions preclude the customer from employing several effective work-flows including: using more than one computer to complete a single task (the files can only be accessed on one computer) and setting up an automated backup (the computer running the backup can’t guarantee that the storage device will be connected come backup time). Further, to ensure full availability of the files, the customer must carry the device with them at all times, and must follow the associated protocols for mounting and dismounting the device.

Other problems with the prior art not described above can also be overcome using the teachings of embodiments of the present invention, as would be readily apparent to one of ordinary skill in the art after reading this disclosure.

SUMMARY OF THE INVENTION

In certain embodiments, automatic modification-triggered transfer of a file among two or more computer systems associated with a user. In some embodiments, a copy of a first file may be received, via a first application at a first computer system, from a second application at a second computer system associated with a user. The first file copy may be automatically received from the second application responsive to the user modifying a content of the first file, where the first file copy is a version of the first file that is generated from the user modifying the content of the first file. Responsive to receiving the first file copy from the second computer system, the first file copy may be automatically transferred via the first application to a third

4

computer system associated with the user to replace an older version of the first file stored on the third computer system.

BRIEF DESCRIPTION OF THE DRAWING

Preferred and alternative embodiments of the present invention are described in detail below with reference to the following drawings.

FIG. 1 is a schematic view of an exemplary operating environment in which an embodiment of the invention can be implemented;

FIG. 2 is a functional block diagram of an exemplary operating environment in which an embodiment of the invention can be implemented; and

FIG. 3 is a functional block diagram illustrating file sharing and/or synchronization according to an embodiment of the invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

An embodiment of the invention leverages remote programming concepts by utilizing processes called mobile agents (sometimes referred to as mobile objects or agent objects). Generally speaking, these concepts provide the ability for an object (the mobile agent object) existing on a first (“host”) computer system to transplant itself to a second (“remote host”) computer system while preserving its current execution state. The operation of a mobile agent object is described briefly below.

The instructions of the mobile agent object, its preserved execution state, and other objects owned by the mobile agent object are packaged, or “encoded,” to generate a string of data that is configured so that the string of data can be transported by all standard means of communication over a computer network. Once transported to the remote host, the string of data is decoded to generate a computer process, still called the mobile agent object, within the remote host system. The decoded mobile agent object includes those objects encoded as described above and remains in its preserved execution state. The remote host computer system resumes execution of the mobile agent object which is now operating in the remote host environment.

While now operating in the new environment, the instructions of the mobile agent object are executed by the remote host to perform operations of any complexity, including defining, creating, and manipulating data objects and interacting with other remote host computer objects.

File transfer and/or synchronization, according to an embodiment, may be accomplished using some or all of the concepts described in commonly owned U.S. patent application Ser. No. 11/739,083, entitled “Electronic File Sharing,” the entirety of which is incorporated by reference as if fully set forth herein.

FIG. 1 illustrates an example of a suitable computing system environment **100** in which one or more embodiments of the invention may be implemented. The computing system environment **100** is only one example of a suitable computing environment and is not intended to suggest any limitation as to the scope of use or functionality of the invention. Neither should the computing environment **100** be interpreted as having any dependency or requirement relating to any one or combination of components illustrated in the exemplary operating environment **100**.

Embodiments of the invention are operational with numerous other general purpose or special purpose computing system environments or configurations. Examples of

US 11,003,622 B2

5

well known computing systems, environments, and/or configurations that may be suitable for use with the invention include, but are not limited to, personal computers, server computers, hand-held or laptop devices, multiprocessor systems, microprocessor-based systems, set top boxes, programmable consumer electronics, network PCs, minicomputers, mainframe computers, distributed computing environments that include any of the above systems or devices, and the like.

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

With reference to FIG. 1, an exemplary system for implementing the invention includes a general purpose computing device in the form of a computer 110. Components of computer 110 may include, but are not limited to, a processing unit 120, a system memory 130, and a system bus 121 that couples various system components including the system memory to the processing unit 120. The system bus 121 may be any of several types of bus structures including a memory bus or memory controller, a peripheral bus, and a local bus using any of a variety of bus architectures. By way of example, and not limitation, such architectures include Industry Standard Architecture (ISA) bus, Micro Channel Architecture (MCA) bus, Enhanced ISA (EISA) bus, Video Electronics Standards Association (VESA) local bus, and Peripheral Component Interconnect (PCI) bus also known as Mezzanine bus.

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

6

acoustic, RF, infrared and other wireless media. Combinations of the any of the above should also be included within the scope of computer readable media.

The system memory 130 includes computer storage media in the form of volatile and/or nonvolatile memory such as read only memory (ROM) 131 and random access memory (RAM) 132. A basic input/output system 133 (BIOS), containing the basic routines that help to transfer information between elements within computer 110, such as during start-up, is typically stored in ROM 131. RAM 132 typically contains data and/or program modules that are immediately accessible to and/or presently being operated on by processing unit 120. By way of example, and not limitation, FIG. 1 illustrates operating system 134, application programs 135, other program modules 136, and program data 137.

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

The drives and their associated computer storage media discussed above and illustrated in FIG. 1, provide storage of computer readable instructions, data structures, program modules and other data for the computer 110. In FIG. 1, for example, hard disk drive 141 is illustrated as storing operating system 144, application programs 145, other program modules 146, and program data 147. Note that these components can either be the same as or different from operating system 134, application programs 135, other program modules 136, and program data 137. Operating system 144, application programs 145, other program modules 146, and program data 147 are given different numbers here to illustrate that, at a minimum, they are different copies. A user may enter commands and information into the computer 20 through input devices such as a keyboard 162 and pointing device 161, commonly referred to as a mouse, trackball or touch pad. Other input devices (not shown) may include a microphone, joystick, game pad, satellite dish, scanner, or the like. These and other input devices are often connected to the processing unit 120 through a user input interface 160 that is coupled to the system bus, but may be connected by other interface and bus structures, such as a parallel port, game port or a universal serial bus (USB). A monitor 191 or other type of display device is also connected to the system bus 121 via an interface, such as a video interface 190. In addition to the monitor, computers may also include other peripheral output devices such as speakers 197 and printer 196, which may be connected through an output peripheral interface 190.

The computer 110 may operate in a networked environment using logical connections to one or more remote computers, such as a remote computer 180. The remote computer 180 may be a personal computer, a server, a router,

a network PC, a peer device or other common network node, and typically includes many or all of the elements described above relative to the computer 110, although only a memory storage device 181 has been illustrated in FIG. 1. The logical connections depicted in FIG. 1 include a local area network (LAN) 171 and a wide area network (WAN) 173, but may also include other networks. Such networking environments are commonplace in offices, enterprise-wide computer networks, intranets and the Internet.

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

Referring now to FIG. 2, an embodiment of the present invention can be described in the context of an exemplary computer network system 200 as illustrated. System 200 includes electronic user devices 210, 280, such as personal computers or workstations, that are linked via a communication medium, such as a network 220 (e.g., the Internet), to an electronic device or system, such as a server 230. The server 230 may further be coupled, or otherwise have access, to a database 240, electronic storage 270 and a computer system 260. Although the embodiment illustrated in FIG. 2 includes one server 230 coupled to two user devices 210, 280 via the network 220, it should be recognized that embodiments of the invention may be implemented using two or more such user devices coupled to one or more such servers.

In an embodiment, each of the user devices 210, 280 and server 230 may include all or fewer than all of the features associated with the computer 110 illustrated in and discussed with reference to FIG. 1. User devices 210, 280 include or are otherwise coupled to a computer screen or display 250, 290, respectively. User devices 210, 280 can be used for various purposes including both network- and local-computing processes.

The user devices 210, 280 are linked via the network 220 to server 230 so that computer programs, such as, for example, a browser or other applications, running on one or more of the user devices 210, 280 can cooperate in two-way communication with server 230 and one or more applications running on server 230. Server 230 may be coupled to database 240 and/or electronic storage 270 to retrieve information therefrom and to store information thereto. Additionally, the server 230 may be coupled to the computer system 260 in a manner allowing the server to delegate certain processing functions to the computer system.

Referring now to FIG. 3, illustrated is functionality of an embodiment of the invention allowing a user (not shown) who owns or otherwise controls devices 210, 280 to automatically maintain file synchronization between at least devices 210, 280, or any other user devices on which principles of the present invention are implemented. In an embodiment, an administrator (not shown) of the server 230 or other appropriate electronic device transfers a file-transfer

and/or synchronization application to the user devices 210, 280 for installation thereon. Once installed on the user devices 210, 280, the file-transfer application provides file-transfer clients 310, 320 executable by the user devices 210, 280, respectively. Each of the file-transfer clients 310, 320 may, but need not, include a respective mobile-agent runtime environment 330, 340. The mobile-agent runtime environment 330, 340 include portions of memory of the user devices 210, 280 dedicated to allowing a mobile object the ability to perform operations that the mobile object is programmed to carry out. Also included in the file-transfer application are user interfaces 350, 360 that are displayable on the displays 250, 290, respectively. In an embodiment, the interfaces 350, 360 allow a user to view, access and/or organize files to be synched among the various user devices.

Generally, all files that the user desires to be synched or shared may at some point be uploaded by one or more of the user devices 210, 280 and stored in storage 270. Upon receiving the files to be synched, the server 230 can store such files in the storage 270 and/or transfer the files to one or more of the respective hard drives of the user devices 210, 280, thereby enabling each respective user device to access such files. In this manner, the server 230 is operable to treat each hard drive of the respective user devices 210, 280 as a local document cache for files received by the server. Typically, the server 230 will store one or more of the received files to the storage 270 only if the destination user device is offline or otherwise temporarily not in communication with the server 230. Upon resuming communication with the destination user device, the server 230 will transfer the temporarily stored files to the destination device.

In operation, according to an embodiment, the user may open and modify a file 370, such as a word-processing document or other electronic file. Alternatively, the user may create a first instance of the file 370. The user may have previously have associated, or may now associate, the file 370 with the transfer client 310. Upon a predetermined and user-configurable triggering event, the transfer client 310 transfers the modified file 370, or a copy of the modified file, to the server 230. Such a triggering event may include, but be not limited to, the user saving the file, the elapsing of a predetermined amount of time during which the file has been opened, or the re-initiation of a communication session between the device 210 and the server 230.

The file 370 is transferred to the server 230 on which is executing a synchronization application 380, which may include a mobile-agent runtime environment. Through user configuration, the synch application 380 monitors a set of user devices to which the file 370 should be transferred to effect file synchronization. In the illustrated embodiment, this set of user devices includes the user device 280. The synch application 380 polls the device 280 to determine whether the device 280 is in communication with the server 230. If the device 280 is in communication with the server 230, the synch application 380 transfers the file 370 to the device 280, whereupon the transfer client 320 resident on the device 280 replaces the previous version of the file 370, previously cached on the device 280, with the latest version of the file 370 modified on the user device 210. If the device 280 is not currently in communication with the server 230, the synch application 380 may store the file 370 in the storage 270 until such time as communication between the device 280 and server 230 is reestablished. As illustrated in FIG. 3, a similar reverse-direction synchronization process may be performed by the synch application 380 and the transfer clients 310, 320 with regard to a file 315 modified on device 280 and synchronized to device 210.

In an embodiment, the user interfaces **350**, **360** may include a list of the customer's documents and related metadata, as well as any one-to-one or one-to-many relationships between the documents and metadata. An embodiment can always provide customers with an accurate "picture" of their document collection, regardless of whether their devices physically contain the documents. As alluded to earlier, a problem with distributed file systems and FTP is the latency between a file being put onto a file system and it showing up on a remote machine. To prevent this problem, an embodiment directory is decoupled from the movement of files. An embodiment's directory update system updates at a higher priority than the documents to be synchronized. This feature ensures that when a customer browses or searches through his set of documents, they appear even if they have not yet been cached locally on the user device. An indicator signifying a document's availability may be prominently displayed adjacent to the document's representation so that customers are aware of the document's availability.

An embodiment may include a stand-alone application that allows customers to find and manage documents associated with transfer clients **310**, **320** by visualizing relationships between documents and their metadata. It allows customers to tag documents with any number of identifiers. Customers can relate both documents and tags with each other in any number of user-specified one-to-one and one-to-many relationships, and an embodiment provides a user interface to browse and search on these relationships. To mitigate the customers' learning curve, an embodiment can implement relationships common to contemporary file systems, including a folder hierarchy. In addition to this, an embodiment provides direct support for methods that the customer uses to organize documents by manifesting them as user interface idioms. This is unlike conventional document filing systems which require the customer to work within a strict folder metaphor for organization.

Some alternate methods that an embodiment supports for organizing documents include:

Allow customers to organize their documents by application. Many times customers remember the application used to create a document instead of the document's name or its location in a hierarchy.

Allow customers to organize their documents by most recent access. Customers are likely to access a document they've accessed in the near past. Usually, such documents are part of a task that the customer is actively working.

Allow customers to organize their documents by project or subproject.

Allow customers to organize their documents by people. Many times, especially in the context of a collaboration, a document is directly related to one or more people other than the customer.

Allow the customer to organize their document by process stage. Documents may represent one or more stages of a process. Customers need a method for organizing documents by process stage, and a mechanism for moving the document through a set of predefined stages.

Allow customers to organize their documents by any of the aforementioned methods concurrently. These organization methods are not mutually exclusive.

An embodiment presents an interface that allows a customer to locate one or more documents associated with the transfer clients **310**, **320** and open such document into a separate software application. Since this interface is intended to be used from within the separate application, that

application may need to know how to invoke such interface. Advantageously, this invocation behavior can be provided to the application using the application's plug-in API.

An embodiment presents an interface that allows a customer to synchronize a currently opened document according to processes described elsewhere herein. This interface can be invoked within an application and can be made available to the application in the manner described above in connection with the application's plug-in API.

Some files associated with the transfer clients **310**, **320** are dependent on other files associated with the transfer clients **310**, **320**. For example, a desktop publishing document may include images that are stored in files separate from the main document. Previous file-synching solutions treat these files as separate. Because of this, for example, a document synchronized from the device **210** to the device **280** may be opened by the user of the device **280** before the image files have been fully transferred to the device **280**. This causes the document to fail to open, or break, since the image files don't exist or are incomplete. An embodiment prevents this by: (1) always ensuring the file catalog (e.g., the stand-alone application that allows customers to find and manage documents associated with transfer clients **310**, **320**, as discussed above herein) is synchronized before any file data is synchronized, and (2) pausing any file access by any program until the file contents have been fully synchronized. In such an embodiment, if a user attempts, using a software program, to open a file whose related files haven't yet finished transferring to the local (hard drive) cache, if that software attempts to open the related files, the software program is blocked by an embodiment until the requested files are downloaded and ready to access.

Other file sending and synchronizing software requires the user to upload their data to a storage device owned by the operator of the service. An embodiment treats storage as a participant in the synchronization process; this means that the user can choose the service or device where their files will be stored. The file transfer/synching is abstracted from the storage system allowing any storage to be used. An embodiment treats storage like any other synch target, such as a desktop computer, or a cell phone. As such, any device owned or otherwise controlled by the user and running a synch application, such as synch application **380**, as provided in an embodiment of the invention can perform the storage and/or synching functions described elsewhere herein. That is, the user device **280** or user device **210**, rather than the server **230**, may perform such functions.

While a preferred embodiment of the invention has been illustrated and described, as noted above, many changes can be made without departing from the spirit and scope of the invention. For example, as an alternative to the approach described with reference to FIG. 3, wherein the transfer clients **310**, **320** function to "push" modified or created files to the synch application **380**, the synch application **380** may instead function to periodically "pull" or otherwise actively retrieve such files from the transfer clients **310**, **320**. Instead, the invention should be determined entirely by reference to the claims that follow.

What is claimed is:

1. A system comprising:

a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to: receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modi-

US 11,003,622 B2

11

fying a content of the first file stored on the first client device, the copy of the first file being a version of the first file that is generated from the user modifying the content of the first file;
 store the copy of the first file on the server system;
 receive, from the first client device, first metadata associated with the version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;
 automatically transfer, based on the first priority being greater than the second priority, the first metadata to the second client device such that the first metadata is transferred to the second client device prior to the copy of the first file being transferred to the second client device; and
 automatically transfer, over a network, the copy of the first file to the second client device associated with the user to replace an older version of the first file stored on the second client device, responsive to receiving the copy of the first file from the first client device.

2. The system of claim 1, wherein the computer program instructions, when executed, cause the server system to:
 store the copy of the first file to a memory device associated with the server system, wherein the copy of the first file is stored on the memory device associated with the server system responsive to determining that the server system is not in communication with the second client device; and
 automatically transfer the copy of the first file to the second client device to replace the older version of the first file stored on the second client device with the copy of the first file, responsive to (i) resuming communication with the second client device and (ii) receiving the copy of the first file from the first client device.

3. The system of claim 1, wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.

4. The system of claim 1, wherein availability of the version of the first file is presented at the second client device based on the first metadata.

5. The system of claim 1, wherein the computer program instructions, when executed, cause the server system to:
 receive a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the second client device responsive to the user modifying a content of the second file stored on the second client device, the copy of the second file being a version of the second file that is generated from the user modifying the content of the second file;
 determine that the server system is in communication with the first client device associated with the user; and
 automatically transfer the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server

12

system is in communication with the first client device and (ii) receiving the copy of the first file from the second client device.

6. The system of claim 1, wherein the computer program instructions, when executed, cause the server system to:
 periodically perform a pull request, wherein the copy of the first file is automatically received from the first client device responsive to (i) the pull request and (ii) the user modifying the content of the first file stored on the first client device.

7. The system of claim 1, wherein the copy of the first file is automatically received from the first client device responsive to (i) a push request of the first client device and (ii) the user modifying the content of the first file stored on the first client device.

8. The system of claim 1, wherein the copy of the first file is automatically received from a first application at the first client device, and wherein the first application comprises a runtime environment for one or more mobile-agent objects.

9. The system of claim 8, wherein the first application is configured to create a first mobile object, and wherein the first mobile object is configured to create a proxy object at the server system.

10. The system of claim 9, wherein the first mobile object is configured to provide the copy of the first file to the proxy object, and wherein the proxy object is configured to store the copy of the first file on a memory device associated with the server system.

11. A method being implemented by a server system comprising one or more processors executing computer program instructions that, when executed, perform the method, the method comprising:
 receiving, by the server system, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being a version of the first file that is generated from the user modifying the content of the first file;
 receiving, by the server system, from the first client device, first metadata associated with the version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;
 automatically transferring, by the server system, based on the first priority being greater than the second priority, the first metadata to the second client device such that the first metadata is transferred to the second client device prior to the copy of the first file being transferred to the second client device; and
 automatically transferring, by the server system, over a network, the copy of the first file to the second client device associated with the user to replace an older version of the first file stored on the second client device, responsive to receiving the copy of the first file from the first client device.

12. The method of claim 11, further comprising:
 initially determining, by the server system, that the server system is not in communication with the second client device associated with the user; and
 automatically transferring, by the server system, the copy of the first file to the second client device to replace the older version of the first file stored on the second client device with the copy of the first file, responsive to (i) resuming communication with the second client device,

US 11,003,622 B2

13

(ii) determining that the server system is in communication with the second client device, (iii) and receiving the copy of the first file from the first client device.

13. The method of claim 11, further comprising:

storing, by the server system, the copy of the first file to a memory device associated with the server system, wherein the copy of the first file is stored on the memory device associated with the server system responsive to initially determining that the server system is not in communication with the second client device.

14. The method of claim 11, wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.

15. The method of claim 11, wherein the copy of the first file is automatically received from a first application at the first client device, wherein the first application comprises a runtime environment for one or more mobile-agent objects, and wherein the first application is configured to create a first mobile object, and the first mobile object is configured to create a proxy object at the server system and to provide the copy of the first file to the proxy object.

16. One or more non-transitory machine-readable media storing instructions that, when executed by one or more processors of a first computer system, cause operations comprising:

receiving, by the first computer system, over a network, a copy of a first file from a second computer system associated with a user, wherein the copy of the first file is automatically received from the second computer system responsive to the user modifying a content of

14

the first file stored on the second computer system, the copy of the first file being a version of the first file that is generated from the user modifying the content of the first file;

storing, by the first computer system, the copy of the first file on the first computer system;

receiving, by the first computer system, from the second computer system, first metadata associated with the version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;

automatically transferring, by the first computer system, based on the first priority being greater than the second priority, the first metadata to a third computer system associated with the user such that the first metadata is transferred to the third computer system prior to the copy of the first file being transferred to the third computer system; and

automatically transferring, by the first computer system, over a network, the copy of the first file to the third computer system associated with the user to replace an older version of the first file stored on the third computer system, responsive to receiving the copy of the first file from the second computer system.

17. The one or more non-transitory machine-readable media of claim 16, wherein at least one of the first computer system or the second computer system comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the first computer system or the second computer system assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.

* * * * *