A fault tolerant external application server. The external application server is a web based system that allows a user of a client computing device to work with a file over a network via a general client application communicating with a host. The host brokers the functionality and provides a platform for interacting with the external application server. The external application server is implemented as a server farm. A fault tolerant farm system combines latent configuration replication between farm members, interchangeable farm members, and optional health monitoring to allow the external application server farm to provide on-the-fly configuration while maintaining full functionality without requiring a real time state management database.
Fig. 2
Fig. 4
Fig. 5B
RELATED APPLICATIONS


BACKGROUND

[0002] Enterprises often maintain various types of documents that are stored in different places for different purposes. In many cases, such documents are created and stored according to a variety of different software applications and storage systems. For example, documents may be generated using word processing applications, spreadsheet applications, presentation applications, note applications, graphic design applications, photographic applications, and the like. Generated documents may be stored within a variety of storage systems, including one or more content servers used for storing documents of various types, servers for storing documents as attachments to electronic mail items (e-mail), storage systems for storing documents as attachments to meetings, customer relationship management (CRM) systems for storing documents as attachments to leads or customer data, general purpose document stores for storing documents for routine use, and/or specialized document stores (e.g., Documentum® from Documentum, Inc.) for storing documents for specific, highly regulated needs.

[0003] In a typical server farm, proper operation requires all members of the farm to share an up-to-date and perfect understanding of the farm topology and configuration in order to avoid a loss of functionality. In order to provide an up-to-date understanding, a typical server farm relies on a state configuration database; however, the use of a state configuration database generally adds complexity to the management, requirements, administration, and cost of the server farm.

[0004] It is with respect to these and other considerations that the present invention has been made.

BRIEF SUMMARY

[0005] The following Brief Summary is provided to introduce a selection of concepts in a simplified form that are further described below in the Detailed Description. This Brief Summary is not intended to identify key features or essential features of the claimed subject matter, nor is it intended to be used to limit the scope of the claimed subject matter.

[0006] According to the embodiments, the external application server farm includes a fault tolerant farm (FTF) system. The external application server farm achieves a low administrative burden, minimal supporting software requirements, and excellent robustness, reliability, and scalability as the result of the FTF system. The major components of the FTF system include the farm management component, the fungible server component, and the health monitoring component. In general, the FTF system manages the farm state, which describes the current state of the external application server farm without the cost and complexity of a state management database. To accomplish this, the FTF system includes a variety of agents that run on each member of the external application server farm. An agent is a system service that provides a distinct piece of functionality for the farm member running the agent.

[0007] The farm management component includes the farm state manager and the farm state replicator. The primary functions of the farm management component are managing configuration changes to the farm state and distributing the farm state to the farm members. The farm state manager is an agent running on each farm member. Although each farm member runs the farm state manager only one instance of the farm state manager is designated as the master farm state manager at any given time. Any instances of the farm state manager that are not designated as the master (e.g., the secondary instances) generally continue to run but have no responsibilities (i.e., they do not do anything). The master farm state manager maintains the “official” (i.e., true) version of the farm state for the entire external application server farm. The FTF system stores the official farm state in the central configuration store. Each farm member caches a local copy of the farm state in a local configuration store.

[0008] The FTF system allows the external application server farm to continue to function even if the active master farm state manager fails. In that event, the administrator manually designates another farm state manager as the master. Because of the fluid design of the farm state handling by the FTF system, it is not necessary to promote another farm state manager. Until a new master farm state manager is specified, each separate instance of the farm state manager continues to operate independently using the last known farm state cached in the local configuration store of that particular computing device.

[0009] The farm state replicator is an agent that makes sure each farm member has a reasonably current copy of the official version of the farm state. Each computing device runs an instance of the farm state replicator. To keep the locally cached version of the farm state reasonably current, the farm state replicator periodically contacts the master farm state manager and copies the official farm state to the computing device’s local configuration store.

[0010] The amount of time between updates to the local configuration store represents an inherent latency in the farm management component. Because of the latency involved in the propagation of the official farm state to the farm members introduced by the farm management component, the overall farm state is presumed to be incoherent (i.e., the farm state known to individual farm members can differ from the official farm state and/or the farm state known to other farm members). As the latency period passes without any configuration changes, the overall farm state becomes coherent (i.e., the external application server operates as the administrator intends) once each farm member gets the official farm state from central configuration store. It is not necessary that the latency period be short for the external application server farm to remain functional. An extended latency period simply increases the amount of time required before all farm members operate as intended by the administrator.

[0011] The external application server farm exhibits a high degree of fault tolerance due to the fungible server component. Among other benefits, the fungible server component allows the external application server farm to operate with an incoherent farm state without a loss of functionality. A first aspect of the fungible server component is the intended role component. The members of the farm and the role of each farm member are specified in the farm configuration; how-
ever, the external application server roles are intended roles that describe the primary focus of that particular member without restricting the functionality of the member. The fungible server component requires all farm members to run all agents, or at least all agents specific to the functionality of the external application server, at all times. Thus, each member is capable of performing any action normally handled by the external application server. Moreover, each farm member will perform any action that is requested of it regardless of the intended role assigned to that member.

**0012** A second aspect of the fungible server component is the fallback behavior component. The external application server receives requests from outside sources and those requests are distributed to a farm member based on the configuration of the external application server. Additionally, intra-farm requests can be made between the farm members. If a request made to a farm member but the request is not filled for any reason within the control of the external application server, the request will be made to other farm members, including the requestor, until the request is fulfilled. The fallback behavior component handles the response of the external application server farm to configuration problems in the farm state. If a setting is insolubly misconfigured by the administrator or not understood by the external application server, the external application server reverts to default values for the setting or ignores the setting, as appropriate.

**0013** The combination of the intended role component and the fallback behavior component, the fungible server system makes the external application server extremely fault tolerant of individual farm members failing or otherwise not behaving. The intended role subsystem and the fallback behavior subsystem ensure that all requests made to the external application server are handled without any outwardly apparent (i.e., apparent to the user) loss of functionality.

**0014** The health monitoring component monitors the health of each farm member and the various agents running on each member. Each farm member runs a number of watchdog agents and a health assessment agent. Each watchdog agent is responsible for monitoring the status of a non-watchdog agent running on the farm member. If there is a problem with the non-watchdog agent being monitored, the responsible watchdog agent will take an appropriate action. In various embodiments, appropriate actions include, but are not limited to, attempting to restart the associated non-watchdog agent that is not running and reporting a problem with the associated non-watchdog agent to the health assessment agent. The health assessment agent determines whether or not a farm member is healthy based on reports from the watchdog agents and reports the health determination (i.e., the member health report) to the master farm state manager. In turn, the master farm state manager records each member health report as part of the farm state.

**BRIEF DESCRIPTION OF THE DRAWINGS**

**0015** Further features, aspects, and advantages of the present disclosure will become better understood by reference to the following detailed description, appended claims, and accompanying figures, wherein elements are not to scale so as to more clearly show the details, wherein like reference numbers indicate like elements throughout the several views, and wherein:

**0016** FIG. 1 illustrates a block diagram of an enterprise network employing one embodiment of the host agnostic document access system;

**0017** FIG. 2 illustrates a block diagram of one embodiment of the architecture of the external application server;

**0018** FIG. 3 illustrates a block diagram of an external application server showing one embodiment of the components of the fault tolerant farm (FTF) system;

**0019** FIG. 4 is a simplified block diagram of a computing device with which embodiments of the present invention may be practiced;

**0020** FIGS. 5A and 5B are simplified block diagrams of a mobile computing device with which embodiments of the present invention may be practiced; and

**0021** FIG. 6 is a simplified block diagram of a distributed computing system in which embodiments of the present invention may be practiced.

**DETAILED DESCRIPTION**

**0022** A fault tolerant external application server is described herein and illustrated in the accompanying figures. The external application server is a web-based system that allows a user of a client computing device to work with a file over a network via a general client application communicating with a host. The host brokers the functionality of and provides a platform for interacting with the external application server. The external application server is implemented as a server farm. A fault tolerant farm system combines latent configuration replication between farm members, interchangeable farm members, and optional health monitoring to allow the external application server farm to provide on-the-fly configuration while maintaining full functionality without requiring a real-time state management database.

**0023** FIG. 1 illustrates one embodiment of an exemplary enterprise network 100 including one or more hosts 102 and one or more external application servers 104. A user 106 accesses the host 102 using a web browser 108 from a client computing device 110. The host 102 is, most generally, a content server that stores documents in a document storage system 126 and manages permissions for the user 106. Generally, the host 102 runs a host application 112 providing a host user interface 114 that handles the normal functions of the host 102. At least some of the normal functions of the host 102 involve providing the user 106 with access to stored documents 116 contained in a content store 126 and intended to be viewed and/or edited using a supporting application. The host 102 also provides a generic platform to make the services of the external application server 104 available to user 106. The external application server 104 provides browser-based web applications that allow the user 106 to interact with documents available through the host. The open platform interface defines and directs document operations between the host 102 and the external application server 104. The host 102 also implements an endpoint 120 to receive communications from the external application server 104. Although the host 102 initiates scenarios involving the services of the external application server 104, the host 102 does not make calls to the external application server 104. Instead, the external application server 104 exposes supported functionality to operate on supported document types using callbacks.

**0024** The external application server 104 runs one or more web-based service applications 118 enabling the user 106 to access, view, edit, and, optionally, perform other operations on content (i.e., a file or document) and perform folder (i.e., directory) management over the network from the client computing device 110 without requiring a local instal-
lation of the appropriate application(s) needed to work with a particular document type. The operation and output of the external application server 104 is not specific to the host 102 that invokes the functionality of the external application server 104. Each service application 118 generally runs as a service on the external application server 104. The external application server 104 uses the open platform interface and the set of open platform interface conventions to integrate with the host 102. The external application server 104 provides the necessary operations and functionality to work with a document of a selected file type. The external application server 104 is host agnostic. In other words, the operation and/or output of the external application server 104 are not specific to the host 102 that facilitates access to the services of the external application server 104. Examples of the service applications 118 to handle various document types include the online (i.e., web based) companions to the standard (i.e., locally installed) applications for working with word processing documents, spreadsheets, presentations, and notes.

[0025] The operations provided by each service application 118 are typically specific to a selected file type or related to folder management. The core operations provided by the external application server 104 are viewing and editing documents. In various embodiments, the service application 118 provides one or more additional operations including, but not limited to, reformatting a document for viewing on a mobile device, creating a new document, converting a document, embedding a document, and broadcasting a document. From the perspective of the external application server 104, broadcasting and embedding are particular interactive user flows. In the case of broadcasting, the external application server 104 displays the document on multiple client computing devices 110, and in one embodiment, and keeps track of the current page being view at each multiple client computing devices. The host 102 manages the document upload, broadcast initiation, and the rich client entry points. In another embodiment of the broadcasting operation, page tracking is handled by the host 102.

[0026] The supported operations for each service application 118 are accessed through one or more service application entry URLs. Each service application entry URL serves as an entry point to the external application server 104 for a particular operation on a particular document type. Generally, each service application entry URL includes the address of the external application server 104 and specifies both the task (e.g., embedded edit using the spreadsheet application) and the required data to accomplish the requested task, namely the metadata URL for the document and the access token authorizing access.

[0027] When the user 106 selects an operation for a document, the host 102 generates a service application entry URL for that operation against the selected document for the user. More specifically, the external application host page 122 generates the URL parameters used with the service application 118. The parameters generated by external application host page 122 include, but are not limited to, the access token and the source URL. The source URL is the URL that the service application 118 uses to access the host endpoint 120 and the document. The access token is a token that is unique to the user/object pair that the host endpoint 120 uses to authenticate the user 106 and authorize access to the document and/or the service application 118. In various embodiments, the access token is calculated based on a hash of one or more of the user identifier, the time stamp, and the document identifier and is encrypted with a secret known to the host 102 (e.g., stored in the host configuration database).

[0028] The service application entry points are handled through the wrapper provided by the host 102. The wrapper provided by the host 102 is a framework or environment that displays the output of the external application server 104 and accepts input from the client computing device 110 allowing a user 106 to interact with a document using the functionality provided by the service application 118. In one embodiment, the wrapper includes the external application host page 122 and/or an application frame 124. By way of example, one embodiment of the external application host page 122 generated by the host 102 is a single page that hosts all service application pages generated by the application server 104 in a web page container such as an inline frame (iframes) with an edge-to-edge layout. The external application host page 122 has no user interface of its own. Alternate implementations of the external application host page 122 may include a user interface or substitute other web page containers and layouts for those described above.

[0029] The external application server 104 provides the necessary functionality to access, edit, view, and otherwise manipulate or work with various document types. In the described embodiment, the external application server 104 does not include the complexity and overhead associated with network access, user authentication, file storage, network and file security, and other administrative tasks normally handled by other servers within a network and often specific to a particular enterprise. Omitting such features and focusing the external application server 104 on handling document operations through the open platform interface allows the external application server 104 to be used in a wide range of enterprise network scenarios. It should be appreciated that the external application server 104 performing as described herein and assuming additional roles and responsibilities normally handled by other servers on the enterprise network falls within the scope and spirit of the present invention.

[0030] The host is an online server application capable of being accessed over a network using a general client application, such as a web browser. The services provided by the external application server 104 are consumed by the host and made available to a client computing device. When attached to an external application server 104, the host becomes aware of the service application and functionality supported by the external application server 104. The open platform interface is both extensible and provides support for cross-version interface communication. The basic data transport mechanism of the open platform interface facilitates cross-platform communication. In various embodiments, the basic data is transferred in a JavaScript Object Notation (JSON) body, although it should be recognized that other human and/or machine readable data interchange formats fall within the scope and spirit of the present invention. The open platform interface also follows the service oriented architecture principles of “ignore what you weren’t expecting” and “use default values for data you were expecting but didn’t get” used by some application programming interfaces such as the Windows Communication Foundation (WCF). The semantic of “default values must result in acceptable behavior” used by the open platform interface helps maintain functionality in a highly cross-versioned world.

[0031] FIG. 2 illustrates one embodiment of the topology of the external application server 104 implemented as a multi-server farm. It is not necessary for the external application
server 104 to include multiple servers. One embodiment of the external application server 104 is implemented using a single computing device, which effectively operates as a single server “farm”. In one embodiment, each member 202, 204a, 204b, 204c of the external application server farm 200 is assigned a role as a front end server 202 or a back end server 204a, 204b, 204c. A front end server 202 is responsible for routing document handling requests to the back end server 204a, 204b, 204c associated with the selected document type. When more than one front end server 202 is used, improved performance is achieved by using an optional load balancer 206 to distribute traffic between the front end servers 202. The back end servers 204a, 204b, 204c provide the document handling services for supported documents. In the illustrated embodiment, each back end server 204a, 204b, 204c provides document handling services for a specific document type.

[0032] In the illustrated embodiment, the farm state 208 of the external application server 104 is stored in one or more files contained in the central configuration store 210. The information contained in the farm state 208 includes, but is not limited to, one or more of the following items: farm topology (i.e., identification of the members of the farm), farm member roles, configuration settings for the farm members, external application server configuration settings, service application configuration settings, and farm member health. Each farm member 202, 204a, 204b, 204c must have read access to the central configuration store 210. In the illustrated embodiment, the central configuration store 210 is a shared folder readable by each computing device that makes up the external application server. In the various embodiments, the files storing the farm state 208 are machine readable and, optionally, human readable.

[0033] The external application server 104 is accessible through standard network resource addressing schemes. Suitable network addressing schemes include, but are not limited to, uniform naming convention (UNC) paths, drive mappings, and uniform resource indicators (URI), such as uniform resource names (URN) and uniform resource locators (URL). The externally visible addresses either face the internet or an intranet and are commonly accessed using a URL. The internal addresses of the external application server 104, such as the configuration folder path, are commonly accessed using a UNC path.

[0034] In one embodiment, intra-farm communications are secure to ensure that only members of the farm make requests against the cache and the back end servers. This is accomplished by checking the identity of the requesting machine. Intra-farm communication is optionally encrypted for enhanced security.

[0035] FIG. 3 illustrates an external application server farm 200 with one embodiment of the fault tolerant farm (FTF) system 300. The external application server farm 200 achieves a low administrative burden, minimal supporting software requirements, and excellent robustness, reliability, and scalability as the result of the FTF system 300. The major components of the FTF system 300 include the farm management component 302, the fusable server component 304, and the health monitoring component 306. In general, the FTF system 300 manages the farm state 208, which describes the current state of the external application server farm 200, without the cost and complexity of a state management database. To accomplish this, the FTF system 300 includes a variety of agents that run on each member of the external application server farm 200. An agent is a system service that provides a distinct piece of functionality for the farm member running the agent. In the illustrated embodiment, the external application server farm 200 shows a variable set of farm members 308a-308n.

[0036] The farm management component 302 includes the farm state manager 310a, 310b and the farm state replicator 312. The primary functions of the farm management component 302 are managing configuration changes to the farm state and distributing the farm state to the farm members 308a-308n.

[0037] The farm state manager 310a, 310b is an agent running on each farm member 308a-308n. Although each farm member 308a-308n runs the farm state manager 310a, 310b only one instance of the farm state manager is designated as the master farm state manager 310a at any given time. Any instances of the farm state manager 310b that are not designated as the master (i.e., the secondary instances) generally continue to run but have no responsibilities (i.e., they do not do anything). In an alternate embodiment, the secondary instances of the farm state manager 310b are stopped when a master farm state manager 310a is found and only start if no master farm state manager 310a is available or at the manual request of the administrator.

[0038] The master farm state manager 310a maintains the “official” (i.e., true) version of the farm state 208 for the entire external application server farm 200. The FTF system 300 stores the official farm state 208 in the central configuration store 210. Each farm member caches a local copy of the farm state in a local configuration store 314.

[0039] The master farm state manager 310a accepts configuration changes made from any farm member and updates the official version of the configuration state in the central configuration store 210. In order to prevent conflicts from arising when multiple configuration changes are contemporaneously attempted, the master farm state exercises access control over the central configuration store 210. In one embodiment, the master farm state manager 310a writes the configuration state change to the configuration files in the central configuration store 210 and controls conflicts by locking the configuration files when a configuration change is received. In one embodiment, the central configuration store 210 is the local configuration store 314 on the farm member running the master farm state manager 310a. The master farm state manager 310a also accepts requests for a current copy of the farm state from any farm member 308a-308n. In some embodiments, the master farm state manager 310a only accepts of configuration changes and requests from selected farm members and/or selected computing devices which are not members of the external application server farm 200.

[0040] The FTF system 300 allows the external application server farm 200 to continue to function even if the active master farm state manager 310a fails. In that event, the administrator manually designates another farm state manager 310b as the master. Because of the fluid design of the farm state handling by the FTF system 300, it is not necessary to promote another farm state manager 310b. Until a new master farm state manager 310a is specified, each separate instance of the farm state manager 310b continues to operate independently using the last known farm state cached in the local configuration store 314 of that particular computing device. In an alternate embodiment, the external application server automatically promotes another farm state manager to master. In the various embodiments, the automatic decision to automatically promote one instance of the farm state manager
is entirely arbitrary or is based on selection criteria (e.g., considering the role, load, and/or resources of the computing device and/or the time of the last farm state replication operation).

[0041] The farm state replicator 312 is an agent that makes sure each farm member 308a-308n has a reasonably current copy of the official version of the farm state. Each computing device runs an instance of the farm state replicator 312; however, unlike the farm state manager, there is no master farm state replicator. To keep the locally cached version of the farm state reasonably current, the farm state replicator 312 periodically contacts the master farm state manager 310a and copies the official farm state to the computing device’s local configuration store 314. In one embodiment, the farm state replicator 312 updates the local farm state of the farm member on a schedule (e.g., every 30 seconds). In some embodiments, each individual farm state replicator 312 operates on an independent schedule. In other embodiments, each individual farm state replicator 312 operates on a common schedule based on a master clock (e.g., at the start of each minute of a synchronized clock). In certain embodiments, the farm state replicator 312 updates the local configuration state in response to an event or other similar trigger.

[0042] The amount of time between updates to the local configuration store represents an inherent latency in the farm management component 302. Because of the latency involved in the propagation of the official farm state to the farm members 308a-308n introduced by the farm management component 302, the overall farm state is presumed to be incoherent (i.e., the farm state known to individual farm members can differ from the official farm state and/or the farm state known to other farm members). As the latency period passes without any configuration changes, the overall farm state becomes coherent (i.e., the external application server operates as the administrator intends) once each farm member 308a-308n gets the official farm state from central configuration store 210. It is not necessary that the latency period be short for the external application server farm 200 to remain functional. An extended latency period simply increases the amount of time required before all farm members operate as intended by the administrator.

[0043] The external application server 104 exhibits a high degree of fault tolerance due to the fungible server component 304. Among other benefits, the fungible server component 304 allows the external application server farm 200 to operate with an incoherent farm state without a loss of functionality. A first aspect of the fungible server component 304 is the intended role component. The members of the farm and the role of each farm member 308a-308n are specified in the farm configuration; however, the external application server roles are intended roles rather than strict roles commonly used with conventional servers. A strict role defines the services that are run on a particular server. Assigning a strict role to a server is often done to optimize the performance of the server or tailor the functions handled by a particular server to that server’s resources. In contrast, the intended role describes the primary focus of that particular member without restricting the functionality of the member. Although an intended role does not directly impact the functionality of the member, the use of an intended role provides other benefits, such as performance improvements as a result of load balancing between the farm members.

[0044] The fungible server component 304 requires all farm members 308a-308n to run all agents, or at least all agents specific to the functionality of the external application server 104, at all times. This includes, but is not limited to, farm management agents (e.g., the farm state manager 310 and the farm state replicator 312), document handling agents 316a-316f (e.g., word processing viewer agents, spreadsheet editor agents, and presentation broadcast agents), and monitoring agents (e.g., watchdog agents and health assessment agents). Thus, each member is capable of performing any action normally handled by the external application server. Moreover, each farm member 308a-308n will perform any action that is requested of it regardless of the intended role assigned to that member. In other words, the members of the external application server farm 200 are fungible. Even if a request is sent to the wrong farm member 308a-308n because the farm state known by the requester is out of date, that “wrong” member will handle the request.

[0045] A second aspect of the fungible server component 304 is the fallback behavior component. The external application server 104 receives requests from outside sources and those requests are distributed to a farm member based on the configuration of the external application server. Additionally, intra-farm requests can be made between the farm members 308a-308n. If a request made to a farm member but the request is not filled for any reason within the control of the external application server, the request will be made to other farm members, including the requester, until the request is fulfilled. In an extreme case, the fungible server system allows a member (i.e., the requester) to handle its own request. The fallback behavior component handles the response of the external application server farm 200 to configuration problems in the farm state. If a setting is insolubly misconfigured by the administrator or not understood by the external application server 104, the external application server 104 reverts to default values for the setting or ignores the setting, as appropriate.

[0046] The combination of the intended role component and the fallback behavior component, the fungible server system makes the external application server extremely fault tolerant of individual farm members failing or otherwise not behaving. The intended role subsystem and the fallback behavior subsystem ensure that all requests made to the external application server are handled without any outwardly apparent (i.e., apparent to the user) loss of functionality.

[0047] The health monitoring component 306 monitors the health of each farm member and the various agents running on each member. Each farm member runs a number of watchdog agents 318a-318g and a health assessment agent 320. Each watchdog agent 318a-318g is responsible for monitoring the status of a non-watchdog agent (e.g., the farm management agents 310a, 310b, 312 and the document handling agents 316a-316f) running on the farm member 308a-308n. In one embodiment, each watchdog agent is uniquely paired (i.e., associated) with one of the non-watchdog agents. If there is a problem with the non-watchdog agent being monitored, the responsible watchdog agent will take an appropriate action. In various embodiments, appropriate actions include, but are not limited to, attempting to restart the associated non-watchdog agent that is not running and reporting a problem with the associated non-watchdog agent to the health assessment agent 320. In some embodiments, the watchdog agent will report a problem with the associated non-watchdog agent directly to the master farm state manager 310a. In other embodiments, the responsible watchdog agent periodically
reports the status of the associated non-watchdog agent regardless of whether or not a problem is detected.

[0048] The health assessment agent 320 determines whether or not a farm member is healthy based on reports from the watchdog agents and reports the health determination (i.e., the member health report) to the master farm state manager 310a. In turn, the master farm state manager 310a records each member health report as part of the farm state. In one embodiment, a farm member is considered healthy if all agents on that member are running properly. In another embodiment, a farm member is considered healthy if all agents specific to the functionality of the external application server are running properly. In a still further embodiment, a farm member is considered healthy if selected agents on that member are running properly. In various embodiments, the health assessment agent 320 includes the watchdog agents when determining whether the farm member is healthy. In such embodiments, the health assessment agent 320 is aware of the watchdog agents running on the farm member and determines that a watchdog agent has failed when that watchdog agent does not make a report. In one embodiment, the health assessment agent 320 only reports to the master farm state manager 310a if the farm member is unhealthy.

[0049] Aside from the reliability benefits, the FTF system 300 provides additional advantages to the administrator of the external application server farm 200. First, by aggregating the health reports from each farm member in the farm state, the administrative burden is simplified because there is a single place to look to check the health of the farm. It is not necessary to check the health at each farm member individually. Second, the aggregated health data is easily monitored by separate monitoring tools. For example, the health data can be incorporated into an event log or a separate monitoring tool (e.g., a comprehensive enterprise or network monitoring system) and an alert sent to the external application server administrator in the event of a problem. Third, because the farm state is cached by all farm members, a true copy or, at least a reasonably recent copy, of the official farm state is available even in the event that the central configuration store is lost. This includes valuable health data that may be useful when investigating the cause of failure.

[0050] The FTF system 300 provides robust fault tolerance in a low cost, easily administered system by leveraging some of the efficiency provided by a highly regulated state management system. When the farm state becomes incoherent, whether as a result of the latency introduced by the FTF system 300 or as a result of a farm member failing, the external application system continues to operate, albeit with a potential loss of efficiency due to a lack of load balancing or having to make multiple fallback requests. In an extreme case where no master farm state manager 310a is available, the FTF system 300 allows the external application server to continue to function (although likely inefficiently) with each farm member operating autonomously based on the cached farm state in the member's local configuration store. As the farm state becomes coherent, the external application server farm 200 settles down and operates as intended by the administrator.

[0051] The embodiments and functionalities described herein may operate via a multitude of computing systems such as the host 102, the external application server 104, and the client device 110 described above with reference to FIG. 1, including wired and wireless computing systems, mobile computing systems (e.g., mobile telephones, tablet or slate type computers, laptop computers, etc.). In addition, the embodiments and functionalities described herein may operate over distributed systems (e.g., cloud-based computing systems), where application functionality, memory, data storage and retrieval and various processing functions may be operated remotely from each other over a distributed computing network, such as the Internet or an intranet. User interfaces and information of various types may be displayed via on-board computing device displays or via remote display units associated with one or more computing devices. For example, user interfaces and information of various types may be displayed and interacted with on a wall surface onto which user interfaces and information of various types are projected. Interaction with the multitude of computing systems with which embodiments of the invention may be practiced include, keystroke entry, touch screen entry, voice or other audio entry, gesture entry where an associated computing device is equipped with detection (e.g., camera) functionality for capturing and interpreting user gestures for controlling the functionality of the computing device, and the like. FIGS. 4 through 6 and the associated descriptions provide a discussion of a variety of operating environments in which embodiments of the invention may be practiced. However, the devices and systems illustrated and discussed with respect to FIGS. 4 through 6 are for purposes of example and illustration and are not limiting of a vast number of computing device configurations that may be utilized for practicing embodiments of the invention, described herein.

[0052] FIG. 4 is a block diagram illustrating example physical components of a computing device 400 with which embodiments of the invention may be practiced. The computing device components described below may be suitable for the computing devices described above, for example, the host 102, the external application server 104, and the client computing device 110. In a basic configuration, computing device 400 may include at least one processing unit 402 and a system memory 404. Depending on the configuration and type of computing device, system memory 404 may comprise, but is not limited to, volatile (e.g. random access memory (RAM)), non-volatile (e.g. read-only memory (ROM)), flash memory, or any combination. System memory 404 may include operating system 406, one or more programming modules 408, which are suitable for running many types of applications (e.g., the user agent/web browser 108) or server applications (e.g., the host application 112 or the service applications 118). Operating system 405, for example, may be suitable for controlling the operation of computing device 400. Furthermore, embodiments of the invention may be practiced in conjunction with a graphics library, other operating systems, or any other application program and is not limited to any particular application or system. This basic configuration is illustrated in FIG. 4 by those components within a dashed line 408.

[0053] Computing device 400 may have additional features or functionality. For example, computing device 400 may also include additional data storage devices (removable and/or non-removable) such as, for example, magnetic disks, optical disks, or tape. Such additional storage is illustrated in FIG. 4 by a removable storage 409 and a non-removable storage 410.

[0054] As stated above, a number of program modules and data files may be stored in system memory 404, including operating system 405. While executing on processing unit 402, programming modules 406 may perform processes
including, for example, one or more of the stages of the methods of the fault tolerant farm system 300. The aforementioned process is an example, and processing unit 402 may perform other processes. Other programming modules that may be used in accordance with embodiments of the present invention may include electronic mail and contacts applications, word processing applications, spreadsheet applications, database applications, slide presentation applications, drawing or computer-aided application programs, etc.

Generally, consistent with embodiments of the invention, program modules may include routines, programs, components, data structures, and other types of structures that may perform particular tasks or that may implement particular abstract data types. Moreover, embodiments of the invention may be practiced with other computer system configurations, including hand-held devices, multiprocessor systems, microprocessor-based or programmable consumer electronics, mini computers, mainframe computers, and the like. Embodiments of 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 memory storage devices.

Furthermore, embodiments of the invention may be practiced in an electrical circuit comprising discrete electronic elements, packaged or integrated electronic chips containing logic gates, a circuit utilizing a microprocessor, or on a single chip containing electronic elements or microprocessors. For example, embodiments of the invention may be practiced via a system-on-a-chip (SOC) where each or many of the components illustrated in FIG. 4 may be integrated onto a single integrated circuit. Such an SOC device may include one or more processing units, graphics units, communications units, system virtualization units and various application functionality all of which are integrated (or “burned”) onto the chip substrate as a single integrated circuit. When operating via an SOC, the functionality of the various client and/or server applications may be implemented via application-specific logic integrated with other components of the computing device 400 on the single integrated circuit (chip).

Embodiments of the invention may also be practiced using other technologies capable of performing logical operations such as, for example, AND, OR, and NOT, including but not limited to mechanical, optical, fluidic, and quantum technologies. In addition, embodiments of the invention may be practiced within a general purpose computer or in any other circuits or systems.

Embodiments of the invention, for example, may be implemented as a computer process (method), a computing system, or as an article of manufacture, such as a computer program product or computer readable media. The computer program product may be a computer storage media readable by a computer system and encoding a computer program of instructions for executing a computer process.

The term computer readable media as used herein may include computer storage media. Computer storage media may include volatile and nonvolatile, removable and non-removable media implemented in any method or technology for storage of information, such as computer system instructions, data structures, program modules, or other data. System memory 404, removable storage 409, and non-removable storage 410 are all computer storage media examples (i.e., memory storage.) Computer storage media may include, but is not limited to, RAM, ROM, electrically erasable read-only memory (EEPROM), flash memory or other memory technology, CD-ROM, digital versatile disks (DVD) or other optical storage, magnetic cassettes, magnetic tape, magnetic disk storage or other magnetic storage devices, or any other medium which can be used to store information and which can be accessed by computing device 400. Any such computer storage media may be part of device 400. Computing device 400 may also have input device(s) 412 such as a keyboard, a mouse, a pen, a sound input device, a touch input device, etc. Output device(s) 414 such as a display, speakers, a printer, etc. may also be included. The aforementioned devices are examples and others may be used.

The term computer readable media as used herein may also include communication media. Communication media may be embodied by 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” may describe a signal that has one or more characteristics set or changed in such a manner as to encode information in the signal. By way of example, and not limitation, communication media may include wired media such as a wired network or direct-wired connection, and wireless media such as acoustic, radio frequency (RF), infrared, and other wireless media. Computing device 400 may include communication connections 16 allowing communications with other computing devices 418. Examples of suitable communication connections 416 include, but are not limited to, RF transmitter, receiver, and/or transceiver circuitry; universal serial bus (USB), parallel, or serial ports, and other connections appropriate for use with the applicable computer readable media.

FIGS. 5A and 5B illustrate a suitable mobile computing environment, for example, a mobile telephone 500, a smart phone, a tablet personal computer, a laptop computer, and the like, with which embodiments of the invention may be practiced. With reference to FIG. 5A, an example mobile computing device 500 for implementing the embodiments is illustrated. In a basic configuration, mobile computing device 500 is a handheld computer having both input elements and output elements. Input elements may include touch screen display 505 and input buttons 510 that allow the user to enter information into mobile computing device 500. Mobile computing device 500 may also incorporate an optional side input element 515 allowing further user input. Optional side input element 515 may be a rotary switch, a button, or any other type of manual input element. In alternative embodiments, mobile computing device 500 may incorporate more or less input elements. For example, display 505 may not be a touch screen in some embodiments. In yet another alternative embodiment, the mobile computing device is a portable phone system, such as a cellular phone having display 505 and input buttons 510. Mobile computing device 500 may also include an optional keypad 535. Optional keypad 535 may be a physical keypad or a “soft” keypad generated on the touchscreen display.

Mobile computing device 500 incorporates output elements, such as display 505, which can display a graphical user interface (GUI). Other output elements include speaker 525 and LED light 520. Additionally, mobile computing device 500 may incorporate a vibration module (not shown), which causes mobile computing device 500 to vibrate to notify the user of an event. In yet another embodiment,
mobile computing device 500 may incorporate a headphone jack (not shown) for providing another means of providing output signals.

Although described herein in combination with mobile computing device 500, in alternative embodiments the invention is used in combination with any number of computer systems, such as in desktop environments, laptop or notebook computer systems, multiprocessor systems, microprocessor based or programmable consumer electronics, network PCs, mini computers, main frame computers and the like. Embodiments of 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; programs may be located in both local and remote memory storage devices. To summarize, any computer system having a plurality of environment sensors, a plurality of output elements to provide notifications to a user and a plurality of notification event types may incorporate embodiments of the present invention.

FIG. 51 is a block diagram illustrating components of a mobile computing device used in one embodiment, such as the computing device shown in FIG. 5A. That is, mobile computing device 500 can incorporate system 502 to implement some embodiments. For example, system 502 can be used in implementing a "smart phone" that can run one or more applications similar to those of a desktop or notebook computer such as, for example, browser, e-mail, scheduling, instant messaging, and media player applications. In some embodiments, system 502 is integrated as a computing device, such as an integrated personal digital assistant (PDA) and wireless phone.

One or more application programs 566 may be loaded into memory 562 and run on or in association with operating system 564. Examples of application programs include phone dialer programs, e-mail programs, personal information management (PIM) programs, word processing programs, spreadsheet programs, Internet browser programs, messaging programs, and so forth. System 502 also includes non-volatile storage 568 within memory 562. Non-volatile storage 568 may be used to store persistent information that should not be lost if system 502 is powered down. Applications 566 may use and store information in non-volatile storage 568, such as e-mail or other messages used by an e-mail application, and the like. A synchronization application (not shown) also resides on system 502 and is programmed to interact with a corresponding synchronization application resident on a host computer to keep the information stored in non-volatile storage 568 synchronized with corresponding information stored at the host computer. As should be appreciated, other applications may be loaded into memory 562 and run on the device 500, including the various client and server applications described herein.

System 502 has a power supply 570, which may be implemented as one or more batteries. Power supply 570 might further include an external power source, such as an AC adapter or a powered docking cradle that supplements or recharges the batteries.

System 502 may also include a radio 572 that performs the function of transmitting and receiving radio frequency communications. Radio 572 facilitates wireless connectivity between system 502 and the "outside world", via a communications carrier or service provider. Transmissions to and from radio 572 are conducted under control of the operating system 564. In other words, communications received by radio 572 may be disseminated to application programs 566 via operating system 564, and vice versa.

Radio 572 allows system 502 to communicate with other computing devices, such as over a network. Radio 572 is one example of communication media. Communication media may typically be embodied by 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. The term computer readable media as used herein includes both storage media and communication media.

This embodiment of system 502 is shown with two types of notification output devices; light emitting diode (LED) 520 that can be used to provide visual notifications and an audio interface 574 that can be used with speaker 525 to provide audio notifications. These devices may be directly coupled to power supply 570 so that when activated, they remain on for a duration dictated by the notification mechanism even though processor 560 and other components might shut down for conserving battery power. LED 520 may be programmed to remain on indefinitely until the user takes action to indicate the powered-on status of the device. Audio interface 574 is used to provide audible signals to and receive audible signals from the user. For example, in addition to being coupled to speaker 525, audio interface 574 may also be coupled to a microphone to receive audible input, such as to facilitate a telephone conversation. In accordance with embodiments of the present invention, the microphone may also serve as an audio sensor to facilitate control of notifications, as will be described below. System 502 may further include video interface 576 that enables an operation of onboard camera 530 to record still images, video stream, and the like.

A mobile computing device implementing system 502 may have additional features or functionality. For example, the device may also include additional data storage devices (removable and/or non-removable) such as, magnetic disks, optical disks, or tape. Such additional storage is illustrated in FIG. 51 by storage 568. Computer storage media may include 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.

Data/information generated or captured by the device 500 and stored via the system 502 may be stored locally on the device 500, as described above, or the data may be stored on any number of storage media that may be accessed by the device via the radio 572 or via a wired connection between the device 500 and a separate computing device associated with the device 500, for example, a server computer in a distributed computing network, such as the Internet. As should be appreciated such data/information may be accessed via the device 500 via the radio 572 or via a distributed computing network. Similarly, such data/information may be readily transferred between computing devices for storage and use according to well-known data/informa-
tion transfer and storage means, including electronic mail and collaborative data/information sharing systems.

[0071] FIG. 6 illustrates a system architecture for providing the browser application 108, the host application 112, and/or the service applications 118 to one or more client devices, as described above. Content developed, interacted with or edited in association with the host application 112, and/or the service applications 118 may be stored in different communication channels or other storage types. For example, various documents may be stored using directory services 622, web portals 624, mailbox services 626, instant messaging stores 628 and social networking sites 630. The host application 112 and/or the service applications 118 may use any of these types of systems or the like for enabling data utilization, as described herein. A server 620 may provide the host application 112 and/or the service applications 118 to clients. As one example, server 620 may be a web server providing the host application 112, and/or the service applications 118 over the web. Server 620 may provide the host application 112 and/or the service applications 118 over the web to clients through a network 615. Examples of clients that may access the host agnostic document access system 100 include computing device 400, which may include any general purpose personal computer 110a, a tablet computing device 110b and/or mobile computing device 110c such as smart phones. Any of these devices may obtain content from the store 616.

[0072] Embeddings of the present invention, for example, are described above with reference to block diagrams and/or operational illustrations of methods, systems, and computer program products according to embodiments of the invention. The functions/acts noted in the blocks may occur out of the order as shown in any flowchart. For example, two blocks shown in succession may in fact be executed substantially concurrently or the blocks may sometimes be executed in the reverse order, depending upon the functionality/acts involved.

[0073] While certain embodiments of the invention have been described, other embodiments may exist. Furthermore, although embodiments of the present invention have been described as being associated with data stored in memory and other storage mediums, data can also be stored on or read from other types of computer-readable media, such as secondary storage devices, like hard disks, floppy disks, or a CD-ROM, or other forms of RAM or ROM. Further, the disclosed methods’ stages may be modified in any manner, including by reordering stages and/or inserting or deleting stages, without departing from the invention.

[0074] In various embodiments, the types of networks used for communication between the computing devices that make up the present invention include, but are not limited to, an internet, an intranet, a wide area network (WAN), local area networks (LAN), and virtual private networks (VPN). In the present application, the networks include the enterprise network and the network through which the client computing device accesses the enterprise network (i.e., the client network). In one embodiment, the client network is part of the enterprise network. In another embodiment, the client network is a separate network accessing the enterprise network through externally available entry points, such as a gateway, a remote access protocol, or a public or private internet address.

[0075] The description and illustration of one or more embodiments provided in this application are not intended to limit or restrict the scope of the invention as claimed in any way. The embodiments, examples, and details provided in this application are considered sufficient to convey possession and enable others to make and use the best mode of claimed invention. The claimed invention should not be construed as being limited to any embodiment, example, or detail provided in this application. Regardless of whether shown and described in combination or separately, the various features (both structural and methodological) are intended to be selectively included or omitted to provide an embodiment with a particular set of features. Having been provided with the description and illustration of the present application, one skilled in the art may envision variations, modifications, and alternate embodiments falling within the spirit of the broader aspects of the claimed invention and the general inventive concept embodied in this application that do not depart from the broader scope.

What is claimed is:

1. An external application server comprising a farm configuration;
   one or more computing devices, each computing device being a member of a server farm, each said member having a local configuration storage location for storing a copy of said farm configuration;
   a management agent for each said member, each said member running one instance of said management agent, one said management agent designated as a master management agent, said local configuration storage location on said member running said master management agent designated as a central configuration storage location for holding a master copy of said farm configuration, said master management agent maintaining said master copy of said farm configuration;
   a replication agent for each said member, each said member running one instance of said replication agent, said replication agent periodically requesting a copy of said farm configuration from said master management agent and storing said copy said farm configuration in said local configuration storage location on said member; and
   a document handling agent for each said member, each said member running one instance of said document handling agent, said document handling agent providing functionality to interact with a document of a selected file type.

2. The external application server of claim 1 characterized in that said farm configuration comprises a plurality of files, each said file being human readable and machine readable.

3. The external application server of claim 2 characterized in that said plurality of files are formatted using a structured markup language.

4. The external application server of claim 1 further comprising a plurality of watchdog agents, each said member running one instance of each watchdog agent, said plurality of watchdog agents comprising a watchdog agent for a watchdog agent, said watchdog agent comprising another agent running on said member.

5. The external application server of claim 4 characterized in that said plurality of watchdog agents comprises a document handling watchdog agent, a replication watchdog agent, and a management watchdog agent.

6. The external application server of claim 1 further comprising a plurality of health assessment agents, each said member running one instance of said health assessment agent.
7. The external application server of claim 6 characterized in that each said watchdog agent reports a watched agent status of the corresponding said watched agent to said health assessment agent, said health assessment agent generating a member health report based on said watched agent status.

8. The external application server of claim 6 characterized in that each said health assessment agent reports said member health report to said master management agent.

9. The external application server of claim 7 characterized in that said master management agent stores said member health report in said farm configuration.

10. The external application server of claim 1 characterized in that each said member is assigned an intended role.

11. The external application server of claim 10 characterized in that each said member responds to any request made to said member regardless of said intended role assigned to said member.

12. A method of providing fault tolerance in an external application server farm, said method comprising the steps of: providing one or more computing devices, each said computing device being a member of a server farm; running a management agent on each said member; Designating said management agent on one said member as a master management agent; storing an official farm configuration in a central farm configuration store using said master management agent; running a replication agent on each said member; requesting a copy of said official farm configuration from said master management agent using said replication agent; storing said copy of said official farm configuration in a local configuration store on said member; and running a document handling agent on each said member, said document handling agent providing functionality to interact with a document of a selected file type.

13. The method of claim 12 characterized in that each said management agent, each said replication agent, and each said document handling agent are watched agents, said method further comprising the step of running a watchdog agent for each watched agent on each said member.

14. The method of claim 13 further comprising the step of running a health assessment agent on each said member.

15. The method of claim 14 further comprising the step of reporting an agent status of the corresponding said watched agent to said health assessment agent using each said watchdog agent.

16. The method of claim 15 further comprising the step of sending a member health report based on each said agent status to said master management agent using said health assessment agent.

17. The method of claim 16 further comprising the step of storing said member health status in said farm configuration using said master management agent.

18. The method of claim 12 further comprising the step assigning an intended role to each said member.

19. The method of claim 12 further comprising the step of allowing each said member to respond to all requests made to said member regardless of said intended role of said member.

20. An external application server comprising: a farm configuration; one or more computing devices, each computing device being a member of a server farm, each said member having a local configuration storage location for storing a copy of said farm configuration; a plurality of watched agents comprising:

(i) a management agent running on each said member, each said management agent designated as a master management agent, said local configuration storage location on said member running said master management agent designated as a central configuration storage location for holding a master copy of said farm configuration, said master management agent maintaining said master copy of said farm configuration;

(ii) a replication agent running on each said member, each said replication agent periodically requesting a copy of said farm configuration from said master management agent and storing said copy said farm configuration in said local configuration storage location on said member; and

(iii) a document handling agent running on each said member, each said document handling agent providing functionality to interact with a document of a selected file type;

a plurality of watchdog agents running on each member, each said watchdog agent uniquely associated with one said watched agent, each said watchdog agent reporting a watched agent status of the associated said watched agent; and

a health assessment agent running on each member, each said health assessment agent receiving said watched agent status and producing a member health report, said health assessment agent sending said member health report to said master management agent for inclusion in said farm configuration.

* * * * *