The abstract of the document describes a system for managing and delivering digital services through computer networks. It mentions that a device acts as a gateway between end-users (human or machine) and various enterprise applications (e.g., file servers, data servers, etc.), providing core digital services such as secure, authenticated access to those enterprise applications. The device can use outside or third-party entitlement stores for authentication and application of access policy for those enterprise applications to provide single sign-on and end-to-end transaction view capabilities. Clustering, replication, and load balancing (both front-end and back-end) of a plurality of these devices allows for highly scalable and reliable service to end-users. Any access unit through any network of choice may access the device.
SYSTEM FOR MANAGING AND DELIVERING DIGITAL SERVICES THROUGH COMPUTER NETWORKS

CROSS-REFERENCES TO RELATED APPLICATIONS

The present application claims priority from related U.S. Provisional Applications Serial No. 60/374,708, filed April 23, 2002, and 60/374,658, filed April 23, 2002; the disclosures of both of which are incorporated herein by reference.

BACKGROUND OF THE INVENTION

1. Field of the Invention

The present invention relates to the field of enterprise technology infrastructure and more particularly to the management and delivery of software applications over networks.

2. Description of the Related Art

Some enterprise technology services have a highly standardized functionality and are required by virtually all enterprises and users. Many organizations are faced with the continually increasing cost of maintaining these basic information technology services such as for example, access to central file storage, email, calendar, and web enabling enterprise applications. This results
in less money to invest on strategic initiatives and projects. These basic enterprise infrastructure and services, which may be regarded as core digital services, require annual operational expenditures to maintain and manage, while adding no strategic value. That is, they now provide essential services for an enterprise to function, collaborate, and communicate, as opposed to providing competitive advantage. Thus, it would be desirable to have a new model for delivering infrastructure services that would drastically lower the cost of those services, while increasing their reliability and providing wider accessibility.

These goals have not been achieved partially because core digital services are delivered today by a combination of hardware and software systems that require custom engineering and configuration. Purchasing the software and hardware necessary to deliver file, email, and calendar services is only a small part of the total cost. Deploying, maintaining, and managing these systems is costly, uses scarce IT talent, and restricts the ability of the enterprise to flexibly respond to changing business and technology conditions.

Currently, in order for an organization to support remote end-users, remote offices, and distributed web services over the Internet, additional hardware and software must be delivered at those locations or to end users PCs. The result is slow, expensive and not always secure systems. In addition, this
process may require changes to the source application to enable the internal resources to be accessible over the Internet.

Fig. 1 illustrates a general view of services and security provided by the prior art systems. The system provides an initial route of firewall protection mechanism 6 for applications 10 from end-users 4. In general, application services 10 tend to comprise of network oriented services such as load balancers and actual applications and data services. Security concerns with the prior art systems are in general split between the network (e.g. firewall 6) and various application developers that typically write applications with unique security and policy rules embedded therein. From a security stand point, the network (e.g. firewall 6 and the router 8) control whether the end-user 4 can see the application services 10 from an end-user's access unit such as a PDA or a computer. The application developer writes the software that would than control if the end-user 4 can retrieve data therefrom.

Each application developed is unique, and has its own unique security and policy concerns that are relevant to that application and its developer. In other words, the security and policy concerns 20 developed and embedded in a Web Server application 24 by a third party software developer will be different from the security and policy concerns 26 developed and embedded in an Application Server unit 28. There is no synchronization or consolidation for user level control
of these applications because not all applications are written equally, with each application done differently with different policies and rules. There is no consistency of security and access policy applied across applications, resulting in fragmented security solutions that are hard to manage and expensive to maintain.

The above security model has grown out of an environment where the network was traditionally a finite, well-understood, and well-controlled environment. Traditional enterprise networks were designed to provide connectivity and to transport information within a physically secure building to a set of well-known endpoints (systems and users) located at the corporate desktop or within a corporate data center. This well controlled environment no longer exists as organizations recognize and embrace the business need to communicate and deliver information from anywhere, to anybody, using any device.

In the absence of any other common point of control, the network has been called upon to provide secure access to corporate information in addition to moving data between endpoints. As organizations place this additional responsibility on an ill prepared network infrastructure, they have been forced to drastically increase spending on security with little to show for the expenditure.
While the security exposure related to end user system access continues to increase, other, perhaps higher risk technology innovations are on the rise — XML and Web services. The use of XML to exchange critical business information coupled with exposing access to corporate applications through the increasing use of Web services represents a critical shift in the way corporate information is accessed and distributed. In order to properly secure this new form of information access and distribution, the content of each network message must be examined at a highly detailed level, with security policy applied to the individual message components. More technically, the individual field, field content, XML tag, application method call, and application method parameters contained within the request must each be examined and the appropriate policy applied. Relying on traditional mechanisms to secure these new advances is unrealistic; thus, a new approach is required.

Several essential trends affect the above security model, including, but not limited to, for example, the role of the network, changing user, introduction of Web services, or the adoption of various standards. Each of these trends, on their own, requires the adoption of new security approaches. In combination, they require a realization that a fundamentally new approach to securing the organization's information is needed.
With respect to the role of the network, as companies moved from mainframe systems to client server systems, and eventually to Web based systems, security became a major responsibility of the network. The network was the common infrastructure of the organization, while diverse departments and groups controlled client server applications. The only way to achieve a consistent and standardized security infrastructure was to place security in the network. However, the results of this have not been good, and very difficult to manage.

As the business needs dictated increased information and system access both internally and externally, small private networks expanded to become national and international networks. Connectivity to the outside world grew and the boundary between the public and private networks blurred. Wireless connectivity has made the physical characteristics of the private network even more blurred and difficult to control. Most organizations can no longer look at the network as finite, with well-controlled end points.

Through this transition, most organizations have maintained a security model based on the old concept that the network was private and the endpoints could be physically secured. The result is a highly segmented network, multiple De-militarized Zones (DMZ’s), and point solutions (i.e., Virtual Private Networks) (VPNs) in an attempt to maintain a secure environment. What has resulted is an
increasingly costly and complex environment that is not meeting current security needs and putting organization information and resources at risk.

As to the user, traditionally the user was an employee of the organization and worked inside a fixed physical location (the corporate office) and at a desktop machine provided by the IT department. Both the employee's access and the device were highly controlled. Over the past few years IT organizations have had to deploy and support different sets of systems and even establish unique organizations to serve an increasingly diverse set of users including employees, customers, and business partners.

The definition of a user today is defined by the business, not the Human Resource department. In today's environment, a user is anyone that the business needs to communicate. The users are varied by nature and the number, and the types and location of devices that they use to communicate are ever increasing. The IT department no longer has strong control over the user or the devices they use. Consequently, the network centric security model based on the idea of a trusted user in a trusted environment using a trusted device is failing.

Web services will have an extraordinary impact on the way organizations interact with each other. The initial benefit of Web services provides improved integration of applications based on emerging standards such as Simple Object
Access Protocol (SOAP), Web Service Definition Language (WSDL), and Universal Description, Discovery and Integration (UDDI). Web services provide a standardized way to address integration.

As organizations open up their internal networks to suppliers, partners and customers, it will be important for organizations to build upon their existing network security controls. While security standards have yet to be determined at this point, organizations will need to implement advanced authentication, firewall, and public essential infrastructure (PKI) systems to ensure their Web service applications are protected.

One of the essential issues today is that the end points of a distributed network, typically the client PC, are particularly vulnerable to malicious intruders. End point attacks normally originate from trusted computers and allow the perpetrator to enter an established “secure” tunnel and intercept all secure, encrypted traffic and decrypt it. As organizations move towards distributed Web services based computing model, policy management and authentication solutions that determine the appropriate level of security on the endpoints will be critical.

Another issue that arises with the Web services model is that users from any client terminal will be able to access Web services. To ensure that only
legitimate clients are able to access the Web service, organizations will need to implement software that validates the user identity and determines what the users are allowed to do within the system. Authentication will also be required on the server side to validate the identity of the Web services with which they are communicating.

Distributed Denial of Service (DDoS) attacks strike at the very existence of Web services. While current DDoS attacks are targeted at crippling entire networks and servers, a new form of DDoS attacks is expected to emerge against applications. DDoS attacks clog the Internet and, consequently, deny access to Web service applications. DDoS attacks targeted at applications could result in reliability and quality of service vulnerabilities. It will be important for organizations to implement security solutions that guard against such attacks.

Thus, there is a need to bridge the gap between all networks (public, private, corporate, etc.), and between individual applications services for secure and uniform delivery of services to employees, customers, and business collaborate. There is also a need for such a system that operates independently of particular network requirements, is reliable, manageable, and can be implemented and maintained at a low cost.

SUMMARY OF THE INVENTION
At least some embodiments of the present invention seeks to provide an infrastructure services layer for delivering core digital services at reduced cost and with increased reliability, accessibility and security, while meeting the functionality requirements of an enterprise or end-users.

At least some embodiments of the invention seeks to provide secure access to all enterprise applications, including, but not limited to, for example email, calendar, file and directory services, and Intranet applications, by dynamically rewriting the application interface so that it can securely run over the Internet.

One goal of at least some of the embodiments of the present invention is to provide multiple devices, groups of which provide distinct services, with the entire environment managed from a single interface or a single management framework.

Another goal of at least some embodiments of the present invention is to provide a system with the ability of self-configuration through its interface with directory systems and other information accessible by the device(s) of the present invention.
At least one embodiment of the present invention further seeks to provide the ability for component modeling, that is, the infrastructure services layer of the present invention works within an overall existing computing environment such as existing storage systems, file systems, application servers, and directory systems.

In addition, at least of the embodiments the present invention seeks to provide scalability, reliability, and load balancing based on a true clustering model, where multiple devices of the present invention may deliver the same services.

At least some of the embodiments of the present invention further seek to leverage products and technologies that exist in the marketplace, including support for standard protocols for communication wherever possible.

In addition, at least some of the embodiments of the present invention also seeks to provide delivery of users services that include, but are not limited to, for example, access provisioning, single sign-on, fine-grain access controls without requiring application changes, or deployment of additional plug-ins.

In keeping with the principles of the present invention, in one embodiment of the invention, a system for providing and managing digital services includes a
central gateway module and a plurality of central digital services modules at a central location connected to the central gateway module through a first local area network (LAN). A public computer network is also connected to the central gateway module. A remote gateway module is connected to the public computer network and one or more remote user computers are connected to the remote gateway module through a second LAN. As a result of this configuration digital services from the central digital services modules may be provided to the remote user computers through a path that includes the first LAN, the central gateway module, the public network, the remote gateway, and the second LAN.

These and other features, aspects, and advantages of the invention will be apparent to those skilled in the art from the following detailed description of preferred non-limiting embodiments, taken together with the drawings and the claims that follow.

BRIEF DESCRIPTION OF THE DRAWINGS

It is to be understood that the drawings are to be used for the purposes of exemplary illustration only and not as a definition of the limits of the invention.

Referring to the drawings in which like reference numbers present corresponding parts throughout:
Fig. 1 is a general view illustration of services and security provided by the prior art systems;

Fig. 2 is a schematic illustration of systems with the infrastructure services layer of the present invention;

Fig. 3 is a general view illustration of services and security in accordance with the present invention;

Fig. 4 is a general physical overview of the device and overall systems used therewith in accordance with the present invention;

Fig. 5 is a logic view illustration of the infrastructure services layer shown as a gateway device in accordance with the present invention;

Fig. 6 illustrates a typical flow of a request and its corresponding response in accordance with the present invention;

Fig. 7 illustrates the overall general architecture of the gateway device of the present invention;
Fig. 8 is a general view of clustering method for the gateway devices in accordance with the present invention;

Fig. 9 illustrates methods of clustering, front-end, and back-end load balancing of the gateway devices in accordance with the present invention;

Fig. 10 is a schematic illustration of two or more gateway devices of the present invention in communication with one another;

Fig. 11 is a flow diagram illustrating an exemplary procedure for the process of end-user initiated request to access a gateway device, and for checking connection state in accordance with the present invention;

Fig. 12 is a flow diagram illustrating an example procedure for the process of handling a request by the end-user in a cluster in accordance with the present invention;

Fig. 13 is a flow diagram illustrating an exemplary procedure for the process of checking for valid session and authorization of a request by an end-user in accordance with the present invention;
Fig. 14 is a flow diagram illustrating an exemplary procedure for the process of authenticating an end-user and creating a session therefor in accordance with the present invention;

Fig. 15 is a flow diagram illustrating an exemplary procedure for the process of capturing a credential change event and updating new credentials of an end-user in accordance with the present invention;

Fig. 16 is a flow diagram illustrating an exemplary procedure for the process of a general request from an end-user in accordance with the present invention;

Fig. 17 is a flow diagram illustrating an exemplary procedure for the process of finding a primary gateway device in case of failure in accordance with the present invention;

Fig. 18 is a flow diagram illustrating an exemplary procedure for the process of finding a secondary gateway device in case of failure in accordance with the present invention;
Fig. 19 is a flow diagram illustrating an exemplary procedure for the process of load balancing among gateway devices in accordance with the present invention;

Fig. 20 is a flow diagram illustrating an exemplary procedure for the process error handling in the device of the present invention;

Fig. 21 is a flow diagram illustrating an exemplary procedure for the process of validating and creating a user session for an administrator in accordance with the present invention;

Fig. 22 is a flow diagram illustrating an exemplary procedure for the process of changing credentials for an administrator in accordance with the present invention;

Fig. 23 is a flow diagram illustrating an exemplary procedure for the process of adding a cluster to a domain by an administrator in accordance with the present invention;

Fig. 24 is a flow diagram illustrating an exemplary procedure for the process of configuring a cluster by an administrator in accordance with the present invention;
Fig. 25 is a flow diagram illustrating an exemplary procedure for the process of making changing to a configuration by an administrator in accordance with the present invention;

Fig. 26 is a flow diagram illustrating an exemplary procedure for the process starting a device in a cluster by an administrator in accordance with the present invention;

Fig. 27 is a flow diagram illustrating an exemplary procedure for the process of stopping a device running in a cluster by an administrator in accordance with the present invention;

Fig. 28 is a flow diagram illustrating an exemplary procedure for the process of adding a new application on a device by an administrator in accordance with the present invention;

Fig. 29 is a flow diagram illustrating an exemplary procedure for the process of starting an added application in a cluster by an administrator in accordance with the present invention;
Fig. 30 is a flow diagram illustrating an exemplary procedure for the process of stopping a running application in a cluster by an administrator in accordance with the present invention.

DETAILED DESCRIPTION OF THE INVENTION

As illustrated in Fig. 2, the present invention provides functionality between the traditional network layer 40 and the application layer 36. It has taken a set of standardized services (core digital services) 42 that exist in most organizations, removing them from the complex application layer 36 of the organization, and creating an infrastructure services layer 38 of easily implemented and managed devices. The infrastructure layer 38 is designed to work within current enterprise technology environments as well as conform to strategic trends in the technology marketplace. This allows the infrastructure technology to be both complimentary to current environments and provide a strategic roadmap for the future. Core digital services 42 may include, but are not limited to, for example systems such as email, calendaring, directories, voice over IP, printing, enterprise security systems, etc.

Figure 3 provides a more detailed, but simplified illustration of the infrastructure layer 38 (as a gateway) incorporated between the enterprise (or network) layer 40 and the application services layer 36. The gateway device 60 of the present invention illustrated in this figure detail only those core digital
services 81 that relate to network oriented services. The infrastructure layer 38 allows the removal and placement of all of the security and policy concerns from both the applications layer 36 and to some degree the network layer 40 into device 60. Complex inconsistent patch work of security and policy for each individual application, with different groups of individual rights are synchronized and consolidates in the gateway device 60 for user level access and user level control for applications residing in the applications services layer 36. The exemplary core digital services 81 are functions related to security that enable uniform security and policy enforcement, consistently applied across all applications for any organization. The gateway device 60 replaces secondary firewalls and routers used in most networks, including various sub-net Authentications for each enterprise application. In addition, both client's and server's side Virtual Private Networks (VPNs) for accessing enterprise applications will no longer be needed. Hence, the gateway device 60 provides a simplification and reduction in parts, complexity and cost of the prior art systems. With the processes of the present invention, the responsibility of security and policy development and enforcement is no longer delegated to the individual application developers.

The physical illustration of one specific functional aspect of the gateway device 60 is shown in Fig. 4, including its physical interactions with other existing systems. The figure illustrates the use of multiple devices 60 interconnected or
clustered to share sessions or information thereacross, creating a more robust, reliable, enterprise class of solutions. As further illustrated, the gateway devices 60 do not care about the physical network that information from various access units 100 is traveling over, whether public 90, private 92 or any other network of choice 94. In addition, the type of access unit 100 used is also of no concern to the device 60.

The illustrated exemplary entitlement stores 96 shown in this figure may comprise of directories or policy stores that a company may own and operate to provide the rules about end-user access privileges to the exemplary enterprise applications 98 via the gateway device 60. In general, companies owning the enterprise applications 98 will develop these rules. The gateway device 60 includes the same kind of functionality (as that of an entitlement store) built into it, and also has the ability to connect to any third party entitlement stores 96, as illustrated.

As shown, the gateway device 60 represents a cost-effective implementation of new application centric security model in an easy to deploy, highly available, highly scalable security appliance. The device 60 enables an organization to deploy existing Web and intranet applications, email, calendaring, file shares and web services securely over any network of choice 90, 92, 94 to remote users 100. The device 60 secures information access and delivery, and
operates as a network service. That is, it can intercept application layer protocols and intelligently route requests. It can also secure the organization's resources, independent of system or application, and provides a single point of administration and control for all applications within and across organizations. The device 60 can further support technology advancements (e.g., Web Services, XML, etc.) that assume security is provided by some other service, and enhance user services such as access provisioning, single sign-on, and fine-grained access controls without requiring application changes, or deployment of additional plug-ins. The end result is a single infrastructure layer 38 that makes the Web and client/server applications available to authorized customers, partners, and employees securely over any network while enhancing their value, providing control and flexibility to easily extend new applications or give access to new user groups.

The logical view of the physical structure of the gateway device shown in Fig. 4 for various process flows is illustrated in Fig. 5. End-users 4 (shown in Fig. 3) may place (or input) user requests 110 through various access units 100 (shown in Fig. 4) to gain access to any number of enterprise (or source) applications 114. The term domain illustrated in this figure is a generic term, and may be replaced by any term relevant to the organization. In addition, although only five- (5) user requests 110 and five- (5) source applications are illustrated for clarity, the system is obviously made to handle any number of users and or
applications. The user requests 110 may be communicated by a variety of protocols 116, depending on the request and the enterprise application 114 being accessed. For example, if secure connection protocols are required for accessing a particular type of an enterprise application 114, the gateway device 60 can provide an encryption type connection protocol, such as for example, a Secure Socket Layer (SSL) connection protocol. Non-secure connection protocols may for example include HTTP type connection protocols. The gateway device 60 of the present invention is capable of brokering between different protocols. That is, it has an understanding of several protocols and can act as a "translator" between them. This capability is obviously extensible.

In general, upon receiving a user-request 110, the gateway device 60 accesses entitlement stores 112 and retrieves authentication type information therefrom, processing it to determine appropriate connection (if any) between the source applications 114 and end-user 4. Although only a few entitlement stores 112 with their respective communications protocols 118 are illustrated for clarity, it is obvious that any third party or internal entitlement store, with its appropriately corresponding communication protocol may be used. The processing system uses connectors to create a path to source applications 114, and uses encoders for dynamic adjustment of the resultant content of the enterprise applications 114 to be secure and Internet ready.
The process illustrated enforces existing policy in a central location or a small number of well controlled strategic locations. It also provides a platform/infrastructure for policy consolidation and enforcement. More importantly, it provides a single point of control for application access policy enforcement. However, policy creation and management may still be decentralized or distributed by organizations. The process also helps improve usage and utilization monitoring, i.e. for the first time there is an end-to-end view of a transaction (e.g. transaction integrity, security, audit-ability, etc.). No application changes are required to enforce policy because the gateway device 60 enforces policy on a request basis, based on message header information. The process provides for strong protection of servers because the core assets 114 and 112 of an organization are no longer directly exposed to user requests 110, i.e. only an approved transaction is passed to the core assets. The process further enables secure access to previously inaccessible applications from the Internet, and enables Single Sign On (SSO) capabilities for users, which is a more secure solution then the existing solutions. It also provides client-side security without Virtual Private Network (VPN) solution, and in fact may be applied across any implementation - network topography, location, or application infrastructure. The gateway device 60 also provides session management (e.g. fail-over, load balancing, etc.), functions that most network devices typically do not provide.
Fig. 6 illustrates detailed steps for the processing of a specific user-request. The request 130 and response 150 generated may be by an end-user or other automation system or application 110. Each process generates audit information available to internally hosted and external analysis application programs. This information may be used to provide input to billing and accounting systems, resource utilization systems, security and audit systems, or other data processing applications that would find an end-to-end detailed view of a message processing useful.

The requester 110 may be an end user attempting to access an application or application system or may be a program or automated process sending requests 130 in a variety of formats or protocols (including for example XML messages, Web Services, etc.). The gateway device 60 receives the request 130 at interface module 131 and terminates the inbound request 130 at module 132, guaranteeing that the original request 130 is never forwarded to the target application system 114 prior to appropriate processing. This process greatly enhances application security and protects corporate assets. The requester information and the request 130 are then routed via module 134 to any external or internal entitlement store 112 to be authenticated against a directory or other authentication source. Once the user or requester information 110 authenticated, the specific request 130 is then examined. The request 130 is validated to make sure that the user 110 is authorized to send requests 130 for
the target resource 114. For example, in the case of a Web Services request, the specific method requested is examined and validated based on the role assigned to the Requester 110. If any of the authentication or authorization checks fail or if the request is for a system resource that is not recognized or configured within the gateway device 60, the initial request 130 is not forwarded to the target source, and is rejected. No further processing is performed on the request 130, with the exception of generating audit information regarding the transaction.

Module 136 retrieves specific content rules (policies) associated with the target resource applications 114 the requester intends to access, the validated requester 110, and the request message 130 from an available entitlement store 112. The applicable content rules are applied to the request message 130 at module 136, which may include any combination of INCLUDE, EXCLUDE, ENCRYPT, DECRYPT, or other filter actions. The content rules will cause the original request content to be modified in various forms including (but not limited to), for example, rewriting URL or URI values to correspond to the target application system. Additional content modifications could include, for example, removing unknown XML tags and associated content, applying message encryption or decryption rules to the message prior to forwarding, or modifying input Web Services method calls and associated method parameters.
Upon modifications of the original request 130, a new request based on the modified message content is generated by module 138, and forwarded via the interface module 140 to a target resource 114 "on behalf" of the original requester 110. The target application or resource 114 receives the modified request, processes it, and returns a response to the interface module 140 of gateway device 60 for further handling. The processing performed by the gateway device 60 on the inbound request 130 and on the resulting generated response 150 is transparent to the target application 114. This allows the introduction of additional security checks and the introduction of additional value added services (such as Single Sign On, Content Acceleration, Message Routing, Load Balancing, etc.) without any requirement for target system modification or for the installation of "agent" software. The inbound target application response is terminated at module 142, with all appropriate acknowledgements returned to the target application 114 (through the interface module 140). The response message content is checked against the content policy rules at module 146, and modified based on the applicable content rules. The modifications performed include (but are not limited to), for example, adjusting all response URL and URI values so that resource references are redirected to the device 60, identifying and removing specific XML response tags and values, identifying and removing specific Web Services response content, or applying encryption to message content based on the role of the original requester 110. The new response message 150 is generated and matched with
the original inbound request 130 at module 148, and returned to the requester 110 through the network interface module 131 of gateway device 60.

The architecture 200 of the gateway device 60 illustrated in Fig. 7 highlights the architecture layers and component of the present invention. The Platform Services architectural layer 202 provides the physical and operational base for the gateway device 60 functionality and includes the Physical Services layer 204 and the Logical Services layer 206. The Physical Services layer 204 includes the actual hardware such as massive storage unit, at least one or more processing components, memory to support processing, and hardware to connect to one or more networks (communication hardware). This layer also includes defined set of application protocols over its communication hardware, including, but not limited to, for example, WAN Ports, LAN Ports, etc. The Physical services layer 204 enables the implementation of the gateway device within a variety of network configurations such as those behind an Internet Router as the first logical device to process, for example, HTTP and other web traffic. The gateway device 60 may also be placed behind firewalls within a network De-militarized Zone (DMZ) configuration, or behind hardware-based load-balancers, or besides other web-servers within the DMZ configuration, or besides applications servers, mail servers, database servers, or any other combinations thereof. The Logical Services layer 206 may include, for example, the Operating System, Application Server, Encryption, De-cryption or other
services or combinations thereof. Any hardened (secure) platform may be used to run the various software applications therein.

The general-purpose services 210 to 232 of the Core Services architectural layer 208 allow the gateway device 60 to function as an enterprise class machine. The Audit Control service 210 provides detailed end-to-end view or audit for each request by tracking requests and responses. Clustering and Replication Service 212 enable load balancing and session replication across active devices and clusters, with the Logging-Services 214 provide all of the detailed elements of all of the request and response. The latter service allows viewing information regarding the devices that a requester was logged-on, the time, the duration, types of requests, and so on. The Licensing Service 216 manages all software and hardware related licensing matters when new software is added or old ones removed (deleted). Exception Handling Services 218 provide registration (or log) of peculiar encounters. Secure Storage Services 220 secure storage space for proprietary applications, information, etc, with Caching Services 222 caching program modules or software used on a regular basis, or images or any form of content, improving speed and performance of the system. The I18N Support Services 226 supports internationalization characters (spelling out internationalization is 18 characters, and hence I18N). The system may also be configured for specific use by an organization through the System
Configuration Services 228, and managed by the Management Services 232. Various system wide notifications are provided by the Notification Services 230.

The Management Services 232 program coordinates and controls all the external devices or a cluster of the gateway devices 60, ensuring appropriate configuration and operation. Management Services 232 provides various interfaces, including a web based interface for systems administration to add, update, remove various services, modules, connectors, or users and groups from the gateway device 60. It allows for addition of protocols and connectors to the device 60, and configures connections to network resources as well as define and update hardware communication connectors, such as port settings, high availability settings, including for example, virtual IP addresses. The Management Services 232 also co-ordinates and manages access to the device Configuration Repository for devices and clustering. The repository stores device and cluster specific configuration information along with configuration information associated with each application service. User ID and username mappings may also be stored within the repository when an existing authoritative source (an entitlement store) is not available.

Application Services architectural layer 240 comprise of functional services provided to end-users (human or machine), including access to applications, security, session management, common navigation feature, etc.
The sub-component layers within layer 240 are extensible, allowing ease of expansion for new functionality. The User Interface sub-component architectural layer 242 and Application Handlers sub-component architectural layer 246 of the Applications Services layer 240 enable handling of requests 110 (shown in Fig. 6) from end-users (human or machine). The Content Encoders sub-component architectural layer 248 and the Protocol Connector sub-component architectural layer 250 handle a set of enterprise application services, such as for example, databases, Web servers, and all applications that are generally called enterprise applications. Each of the enterprise applications have special nuances regarding communication with the outside world, and do not “talk” to one another. The sub-component layers 248 and 250 enable communications between all enterprise applications. The Security & Policy Engine sub-component architectural layer 252 links or brings together the end-user services (242, 246) with the enterprise services (248, 250).

The Pluggable Provider Services architectural layer 256 allows the gateway device 60 to integrate and extend existing investments through reuse. It enables the gateway device 60 not to be a stand-alone machine and provides the ability for connectivity and communication with enterprise applications of a company. The pluggable components within this layer also allow the gateway device 60 to integrate third party products or components, such as those from strategic partners or independent vendors. The Security Providers 258 of
Pluggable Provider Services layer 256 work with authentication type devices (entitlement stores), with the Other Provider Services 260 of the Pluggable Provider Services layer 256 enabling connections (communications) with the actual enterprise applications. This may include passing onto an enterprise application of a company management (through the management plug 260 of the Pluggable Provider Services layer 256) all log-in information from the Log-in Services 214 of the Core Services layer 208. The user or (an administrator) can access any of the core or others services by a variety of methods, including, for example, web pages, with the overall system monitored by the Performance Monitoring Services 228. Any other services meeting the definition of core digital services may be added to the architecture 200 of the gateway device 60, hence the architecture of this device is extensible.

Figs. 8 and 9 illustrate local and wide area clustering, replication, and load balancing services capabilities of devices 60. Fig. 8 illustrates clustering of Management Services (MS) 232 (shown in Fig. 7) amongst a cluster 350 of devices 60 (Non-management service clustering, replication, and load balancing is described in more details in Fig. 9). Within a cluster environment 350, the MS service is implemented as primary controller 358, backup controller 352, and listener 356 configurations, with all devices 60 being identical. The Management Services primary controller 358 provides updates to all active cluster members 60. Hence, the dynamically assigned titles 352, 356, and 358 enable only one
MS service (the primary controller 358) to update the configuration of the cluster 350 at any one point in time, controlling the integrity of the cluster wide configurations. The network 360 used for clustering 350 may be any network of choice, but should preferably be a secure. This may for example, include, but not be limited to SSL type networks. Clustering 350 illustrated in Fig. 8 only applies to Management Services configuration updates. Session data clustering and replication is handled by the security and clustering services describe with respect to Fig. 9.

The Management Services primary controller 358 broadcast configuration updates to all other devices 60 within the cluster 350. It initiates periodic heartbeat queries to active devices 60 in a device table (not shown). A failure by an active device 60 to respond to a heartbeat request within an allowed time will result in the device being marked as inactive within the device table, generating an alert to indicate that the device is down. If a gateway device 60 fails, it is reconfigured or updated by the Management Services primary controller 358 once it has been restarted. If an update is required, the controller provides an update configuration message to the device 60. The device updates its configuration (within the device Configuration Repository), and prepares to receive requests. The Management Services primary controller 358 may, depending on the status of the cluster 350 and any other active devices 60, designate (entitle) the restarted device as a backup controller 352.
A backup controller 352 listens to configuration updates and monitors primary controller 358. In case of failure of primary controller 358, the backup controller 352 takes over as primary controller, and assigns next available device as backup. Therefore, a backup controller performs a dual role within the cluster 350. It listens for updates from the Management Services controller 358 and upon receiving them, updates its local configuration store. It also provides heartbeat queries to the Management Services controller 358. A failure by the Management Services controller 358 to respond to a backup controller 352 heartbeat request within a prescribed time will activate the backup takeover process, whereby the backup assumes the Management Services controller title, and marks the old Management Services controller 358 as inactive within its device table. Although only three devices are illustrated for clarity, plurality of such devices may be used. All the other devices 60 within the Management cluster 350 will be listeners 356 to configuration updates and to controller updating local repository.

The Management Services 232 maintains a dynamic table of active devices 60. This allows for fail-over of Management Services primary controllers and backup controllers in the even of failure, and provides a method for directing the clustering service for replication. To synchronize the Configuration Repository (CR) of a device, serialized update broadcasts are established. The controller
358 will broadcast to individual cluster members checking to ensure that the last update is sequentially consistent, i.e. current update is for example 1002, therefore last committed update cluster member should have been 1001. If the cluster member’s last committed update is not sequentially consistent, then the controller will need to either rollback and apply the updates required to bring the device current, or force a CR rebuild. In the latter instance, the device is temporary brought offline, the CR rebuilt from the controller configuration, and the device then is brought back online.

Replication of Management Service updates are done by the Management Services (MS) coordinating and physically updating the configuration of individual devices 60 and cluster wide updates. When MS has an update to an individual device or cluster-wide configuration, the MS service creates a replication request to the clustering service. The clustering service then controls the distribution of this update to the specific other devices, ensuring that they receive the update (and acknowledge it). It is MS’s responsibility to ensure that the replication request identifies the correct devices to replicate the change, which is based on the nature of the change and the MS Controllers active device list. It is the responsibility of the clustering device to notify MS of any failed updates.

Fig. 9 illustrates both local 380, 382 and wide 390 clustering of devices 60 that may span across a wi...
area (non-management service) clustering involves three main events, including replication of data to all local cluster members 60, replication of user session state and other data to ensure that in the event of failure, the user is unaffected by the device failure, and load balancing of requests across cluster members to ensure distributed load balancing. Both Local 380, 382 and wide 390 area clustering use point-to-point-clustering algorithms to replicate, allow fail-over Session State, and distribute Management Services updates. Point to point clustering eliminates the need for IP multicasting, allowing devices 60 to support wide area clustering and fail-over. Replication uses sequential serial numbering of updates to ensure synchronization of all devices. This includes logging updates during the event or rollback or fail-over of individual devices 60.

Session State replication is the process of ensuring that user session data and attributes are distributed across multiple devices 60 to take over device failures. In case of an individual device failure, end-users (human or machine) authenticated to the failed device can transparently be serviced by another device 60 within the clusters. This means that the end-user does not know that the device has failed - the user is not prompt to re-authenticate. This assumes there is more than one active device at that location (380, 382, or 390). In addition, changes to users Session States (e.g. adding, updating, or deleting attributes or session objects) are replicated to at least one other cluster member to provide fail-over, using point-to-point fail-over methods.

35
Load balancing is another aspect of Clustering and Replications Services 212 provided by the Core Services architectural layer 208 of device 60. It may be accomplished both at front and back ends of a device 60. Front-end load balancing involves balancing of specific incoming (or inbound) requests (from a plurality of end users - human or machine via the network of choice 376) within a cluster area 380, 382 or 390 of devices 60. Back-end load balancing involves balancing of outbound requests from at least one device 60 to a plurality of enterprise application clusters 372. Hence, with back-end load balancing, all requests received by a cluster of enterprise applications 372 are equally distributed among the servers in the cluster 372 by the device 60.

The gateway device 60 of the present invention processes a request from a browser or other type of input mechanism in two ways, from an end-user looking to access an application that is mounted on the device, and for systems administrators and managers to configure and manage the device environment. Fig. 10 illustrates the setup and flow of information between user services 406, 408 (available to end-users) and the administrator services 410 (available to administrator users). In every instance of a start of a device 400 or 402, requests for configuration information for performing various tasks are forwarded to Admin Services 410 of a Management Service 404. These requests are processed (responded) through a user interface 412 and returned to the User
Services 406 or 408 of a unit device 400 or 402 requesting the information. This figure simply illustrates how user services communicate with administrator service to obtain configuration data such as types of enterprise applications available, user that can get access to them, etc. User services are responsible for interface with the end-users, where as admin Services are responsible for interface with administrators.

From an end-user looking to access an application, the gateway device 60 acts as a gateway for all requests given by the end-user through User Services. The flowchart diagrams of Figs. 11 - 20 illustrate logic flows for various User Services exemplary functions.

Fig. 11 is a flow chart illustrating an exemplary procedure for processing an initial request by a user accessing gateway device 60 with a check for the user connection state. A user initiates a request, and at step 450, the gateway device 60 handles it by providing various graphic user interfaces (GUI) to the end-user. As describe in Fig. 10, the User Services communicate the request with the Administrative Services to obtain application configuration information. The information from step 450 is parsed in step 452 and modified to an appropriate comprehensible form for the intended enterprise application and or an entitlement store by the Admin Services of the gateway device 60 the end-user is trying to access. At step 454, Admin Services of the gateway device 60
retrieves and reads the application data. At the next step 456, the process retrieves various application security parameters, including the type of user allowed to access the application and or the type of connection required between the end-user and the application itself. Upon retrieving the security parameters of an application at step 456, if the application does not require authentication (determined at step 458), then the next item determined by the procedure at step 460 includes encryption requirements for the application. If encryption is required, the device checks for the existing state of user connection at step 462 to determine if the connection is encrypted. If so, the request is redirected to appropriate container at step 464 (group, application, etc.), else, the gateway device 60 creates an encrypted connection for the end-user.

Fig. 12 is a flow chart illustrating an exemplary procedure for handling a request in a cluster (group of gateway devices 60 as shown in Figs. 8 and 9). Upon receiving a request from an end-user (as illustrated in Fig. 11), the first step 470, in handling a request in a cluster is to determine if the request has been handled by other devices in a cluster. If so, existence of a valid session and authorization of the request must then be determined (Fig. 13); else, a determination is made in the next step 472 to find the number of devices in a cluster. If this number equals one, then no clustering of devices exists. In the case where the number of devices 60 is greater than one, the procedure determines at step 474 if the request by an end-user is a new request and if so,
determines at step 476 if load balancing is required. If at step 474 it is determinate that the request made by an end user is not a new request, then the device interrogates itself at step 478 to determine if itself is a Primary Session Device (PSD). If not, the session ID for the device is parsed for PSD and Secondary Session Device (SSD) at step 480, and a determination is made for a list of active devices at step 482. The procedure having obtained session IDs and a list of active devices, determine at step 484 if the PSD is active. If so, a logical flag is set at step 486 to indicate that the request has been handled, and then at step 490 the request is forwarded to appropriate device (in this case either a second device 60, a device where a requested enterprise application resides, or an entitlement store). If the active device is determined not to be a PSD at step 484, then the next step 488 determines if the device itself is a SSD. If it is not, similar to step 484, the process determines in step 492 if a SSD device is active. If so, the steps 496 and 498, which are identical to steps 486, 490, execute. If at step 492 it is determined that the SSD is not active, then a logical flag at step 494 is set, requiring a log-in, with the information passed to an error handling procedure.

Fig. 13 is a flow chart illustrating an exemplary procedure for examining valid session and authorization of a request. In order to determine that the existence of a valid session or an authorized request, the access rights of the request must be determined, at step 500. If access rights are invalid, that is, the
end-user request does not have appropriate "clearance", the next step 502 generates an invalid access right exception. Requests or sessions with appropriate access rights are processed at step 504 to determine if the end-user authorization for the specific application accessed, exists. The user may be authorized to access the system, but may not be authorized to access a particular application. If the user has no authorization to access a particular application, the next step at 506 generates and forwards a message to the end-user stating that the user has no authorization for that particular application. On the other hand, if a user is authorized for accessing the requested application, then before the system mounts the application for the end-user to access, it determines at step 508 if the application itself requires encrypted secure connection. If no, the process at step 510 redirects the request to an appropriate container (another device 60, devices where enterprise applications reside, entitlement stores, etc.) If an application requires an encrypted connection, the next step 512 determines and checks for the existing state of user connection. If the user connection is not secure (or encrypted), the next step 513 generates and forwards the message to the user stating the required need for a secure or encrypted connection. If a secure connection exists, then the process at step 514 determines any certification requirements. With such a determination made, the next step at 516 accepts and validates the certification. If valid (determined at step 518), it is processed; else, the process at the next step 520 forwards an error message to the end user stating "Invalid Certificate."
Fig. 14 is a flow chart illustrating an exemplary procedure for authenticating a user and creating a session therefor. Before creation of a session for an end-user, the process at step 522 determines if the number of session in existence exceeds the maximum number of sessions allowed by the system. The process step at 523 generates an error message if such a scenario exists. If a correct number of sessions exist, the security manager at the next step 526 is called to determine at step 528 if a change of credential is requested. If so, the request is processed through the exemplary procedure illustrated in Fig. 15 below. If no such request is made, then the process determines at step 530 the validity of the end-user credentials input. If valid, the process at step 534 checks for an encrypted connection. If invalid, the next process step at 532 generates an error message. The process at the next step 536 determines the result of the check for a secure connection. If it is determined at step 536 that a secure connection is not required, the process at step 540 generates a session ID with combination of Primary Session Device and Secondary Session Device for the end-user, loads user profiles from the Configuration Repository at step 542, increases session count in the device at step 544, and displays the default user interface for the end-user at step 546. If at step 536 it is determined that a secure connection is required, the process will validate the client (end-user) Certificate at step 538, and at step 548, based on the results, either notifies and
generates an error message at step 550 (if invalid), or further processes the request.

Fig. 15 is a flow chart illustrating an exemplary procedure for capturing a credential change event and updating the new credentials. The system at step 552 forwards to the end-user a credential change graphic user interface (GUI). At step 554, the process retrieves the old and the new credentials from the user, and at step 556 checks for the credential rules (e.g. maximum number of characters in a password or other rules if credentials are based on biometrics). The processing step 560 generates an error message for any invalid credentials (determined at step 558). The next processing step 562 forwards the old and the new valid credentials (determined at step 558) to a repository. The next processing step 564 receives (or waits) for a response (acknowledgement) from the repository regarding the updates. With an updated repository (determined at step 566), the next process step 568 forwards a confirmation message to the end-user. The step 560 executes if updating the repository was not successful.

Fig. 16 is a flow chart illustrating an exemplary procedure for processing a general request from an end-user. At step 570, the process instantiated appropriate connection (secure, non-secure, etc.) for the end-user based on the user request. At the next step 572, the process checks for end (or source) application authorization method. This step allows the system to determine the
"security clearance" level requirements of an application requested by an end-user. If the application requires an authorization (determined at step 574), then the process in step 576 forwards the user credentials to the source application, and executes the request at step 578. If no requirements for authorization exist, the process simply executes the end-user request (at step 578). In either case, the next step 580 parses (Encodes) the response from the application, with the user session, the gateway device 60 session ID, and the application session ID (all) updated at step 582. In addition, the next step 584 updates any local session data. After appropriate update of the devices, the number of devices is needed to determine if the device is a Primary Session or a Secondary Session device. If there is only one device, the application forwards the findings to the browser at step 588 and the end-user request processing ends at step 590. If more than one device is available (for example in a cluster setting), then the process determines (at step 592) if the device is a Secondary Session Device. If it is not, the process at step 594 parses the session ID determined at step 582 to retrieve the Secondary Session Device. The process then continues with step 596 with an active device list check to determine at the next step 598 if a Secondary Session Device is active. With a Secondary Session Device active, the next process at step 600 forwards the session data and count to the SSD device. With a successful forwarding of the session data and count (determined at step 602), the application at step 588 then forwards a response to the browser and the process ends.
Fig. 17 is a flow chart illustrating an exemplary procedure for finding a primary device in case of a failure. The process at step 604 attempts to find an active device that has the least number of active sessions, as a Primary Session Device (PSD) for an end-user. If no such device is found (determined at step 606), the process at the next step 608 updates the session ID with a new Primary Session Device, and forwards a response to the browser at the next step 610, ending the entire procedure at step 624. If an active device with the least active sessions as a Primary Session Device is found, the process at step 612 forwards the session data to PSD. The process then determines at step 614 the success of forwarding the data. If successful, the process at step 610 forwards a response to the browser, and the process ends at step 624. If the forwarding of session data was not successful, the process at step 616 attempts the forwarding again, for a prescribed number of times (preferably three). If all attempts fail, the next step 618 updates local active device list, and a the next step 620 forwards a failure message to steps 604 to restart the process.

Fig. 18 is a flow chart illustrating an exemplary procedure for finding a secondary device in case of a failure. The step 626 in this process checks to determine if the total number of devices in use (or in a cluster) is less than three. If there are more than three devices, step 628 selects the second device as SSD and forwards session data to the selected SSD device. The process then
determines at step 630 if the update of the session data on SSD device was a success. If so, other processing continue at step 656. If the updating on the Secondary Session Device was not successful, at step 632 the device is removed from the cluster after a predetermined amount of time has passed (preferably after trying 3 times), if the cluster itself is false. In finding a secondary device in case of failure, if the number of devices determined in step 626 is less than three, the process determines at step 638 if the previous SSD device is not available. If so, the next active device is set to previous SSD at step 640, and another active device from the remaining active device list is searched for at step 642. On the other hand, if a previous SSD is determined to be not null, then step 642 executes (without the intervening step 640). In either case, once step 642 executes, if the result of the search at step 644 is determined that no such device is found, the process continues at step 656. If after searching for the next active device from the remaining active device list at step 642 is found at step 644, then a determination must be made to see if this active device is the previous Secondary Session Device at step 646. If so, step 642 is re-executed, else session data of the end user is forwarded to the Secondary Session Device at step 648, and at step 650 a check is made to determine if the update on SSD was successful. If the update was a success, the SSD name is store as previous SSD at step 654, and the processing continues at 656. If the update was not successful, the process at step 652 attempts the updating procedure for a predetermined number of times until the update is completed (preferably after 3
times). If the number of attempts made exceeds the maximum number of attempts allowed by the system, step 634 forwards a device-failed message, and the next step 636 updates the local active device list.

Fig. 19 is a flow chart illustrating an exemplary procedure for load balancing among devices 60, that is, processing for balancing the amount or number of inbound requests amongst a cluster of devices 60. The process at step 658 selects the first device session count as minimum and names it. The next step 600 searches the next active device name and a session count. If a determination is made at step 664 that the device is found, the session count is stored as minimum, including the device name, at step 666. If at step 664, it is determined that the device searched for at step 660 is not found, then a determination at step 668 must be made to find if the first device selected at step 658 is the "next" active device. If the device selected at step 658 is not the "next" active device, the request at step 670 is marked as handled and forwarded to the found device in the above process.

Fig. 20 is a flow chart illustrating an exemplary procedure for processing errors in device 60. The process of error handling commences with retrieving the error code at step 672. The next process step at 674 determines if the error is critical. If so, the process at step 676 notifies an administrator. If a non-critical error occurs, step 678 reads the error message from the resource, and step 680
registers it. Step 682 displays the error message to the end-user, and at step 684 determines if the error requires a log-in. If so, at step 686 the end-user is redirected to a log-in page, and the process ends at step 688. If no log-in is required (determined at step 684), the error handling process ends at step 688.

An administrator can configure a gateway device 60 (or a cluster thereof) from any location using the Admin Services of the Management Services 232 (shown in Fig 7). The next set of flowchart diagrams of Fig. 21 to 30 illustrates logic flows for various Admin Services exemplary functions.

Fig. 21 is a flow chart illustrating an administrative services exemplary procedure for processing an administrator user request and creation of a session. The step 690 determines if the connection made is secured. If connection is not secure, the process at step 692 forwards the administrator’s request to the login page for a new secure log-in session. On the other hand, if the connection is secure, the Amdin Services determine at step 694 if the user is valid. If the user is not valid, the login session of the administrator fails the next at step 968. The credentials of a valid administrator user are checked at step 700 to determine if expired. If the credentials have not been expired, the Admin Services creates an administrator user session at step 702, loads user profiles from Configuration Repository at step 704, and displays console at step 706 for the administration depending on the administrator role.
Fig. 22 is a flow chart illustrating an administrative services exemplary procedure for processing changes in credential request for Administrator user, including the process of providing the administrator a new set of credentials. Step 708 of this exemplary process forwards the Administrator a credential change GUI. The Management Services within device 60 receives at step 710 the old and the new credentials from the user, and checks for credential rules at step 712. If a determination is made at step 714 that the old and the new credentials meet the rules checked at step 712, the Management Services forwards at step 718 the old and the new credentials for the Administrator to a repository. The Management Services at step 720 must receive a response from the repository regarding the updating of the credentials in the repository. If it is determined at step 722 that the update was successful, the process step at 724 forwards a confirmation to the administrator; else, step 716 generates an error message. In addition, step 716 generates an error message if at step 714 it is determined that the credentials are not valid.

Fig. 23 is a flow chart illustrating an administrative services exemplary procedure for the Administrator to add clusters to a domain. The administrator, through the Management Services (MS) interface (GUI) selects at step 726 an "add cluster" GUI. The next processing step 728 shows a screen to the administrator for configuring a cluster. The process at step 730 continues by
asking the administrator parameters regarding new cluster, and at steps 732, the process checks to determine if the parameters entered by the administrator at step 730 are unique. If they are not, the administrator must re-enter new set of parameters, repeating steps 730, 732. If the parameters entered by the administrator at step 730 are confirmed to be unique at step 732 the Management Services creates a cluster at step 734, updates the Configuration Repository (CR) on itself as well as Backup Admin Services at step 736, and shows the administrator options to Configure the cluster at step 738.

Fig. 24 is a flow chart illustrating an administrative services exemplary procedure for configuring a newly created cluster, e.g. add devices, applications, etc. Step 740 provides the administrator a cluster configuration GUI through the Management Services (MS). The next step 742 shows the administrator a list of standalone devices and applications configurable (selectable) on a cluster. After the selection of the desired devices and applications, at step 744 the Management Services (MS) adds those configuration to the devices in the cluster, and the system is updated. If a successful update is determined (at step 746), the next processing step 750 creates a cluster service. The processing step 748 notifies the administrator regarding update failures.

Fig. 25 is a flow chart illustrating an administrative services exemplary procedure for an Administrator to make various changes to a particular
configuration. Obviously, if the number of devices in a cluster is determined at step 752 to be less than or equal to 1, the process stops at step 754. If the device count is greater than one- (1), then the Management Services determines at step 756 if backup Administrative Services exists. If no, step 758 assigns a new Backup Administrative Service to a device in the cluster. Step 762 updates all the devices in a cluster by promulgating the assignment of backup administrative services to the particular device within the cluster. Steps 768 - 764 execute for a predetermined number of times when updating of the cluster fails. Steps 770 - 788 update the User Services of the devices regarding the above-mentioned assignment of the backup administrative services to the selected device in the cluster.

Fig. 26 is a flow chart illustrating an administrative services exemplary procedure for starting a device in a cluster. The Administrator may start a device through the Management Services GUI at step 790, enabling running of various scripts for activating a device in the cluster at step 792. The next step 794 will try to find the Admin Services in the domain. If at step 796 it is determined that Admin Service is not found, the entire process stops with a request for Admin Services IP address at step 798, with the step 794 re-executed. After finding Admin Services, the next step 800 retrieves the latest configurations from it and at step 802 the device starts with those configurations. If at step 804 it is determined that the operation above was a success, an active device list is then
retrieved from the Admin Services at step 808, else the process stops at step 806. Upon retrieval of the active devices list, the active device table of the Management Services itself is updated at step 810, and an update cycle for the updating device tables is executed at step 812.

Fig. 27 is a flow chart illustrating an administrative services exemplary procedure for stopping a device running in a cluster, by the Administrator. Through the Management Services GUI, the administrator selects a "Stop Device in a Cluster" GUI at step 814. At step 816 that device is marked so not to accept any new requests, and a notification is forwarded at step 818 to the rest of the devices in the cluster regarding the stopped device. After a predetermined wait at step 820, the next processing step at 822 runs a shutdown script for shutting down the selected device. If at step 824 it is determined that the operation is successful, the processing step 828 updates the table on the Admin Services; else step 826 notifies administrator that a problem exists with the device. After the update in step 828, the next step 830 notifies the administrator that the device has stopped.

Fig. 28 is a flow chart illustrating an administrative services exemplary procedure for adding new applications on a device by an Administrator, before they are available to end users. Through the Management Services GUI, the administrator selects an "Add New Mount" GUI at step 832. At step 834, the
Administrator will then be forwarded a question page regarding connector details and mount (add or file) name for the application. At step 836, Administrator completes the form and press an "OK" type GUI to forward the completed application to the Management Services. One of the parameters required for mounting an application is the file name, which is asked in the form generated at step 834. If the name completed by the Administrator in step 836 has a duplicate in the system, the Administrator must re-name the application, starting at step 834. If at step 838 it is determined that no duplicate name exist, then the next step 840 displays a page regarding the details of the application configuration. Upon completion of the configuration information at step 842, the administrator updates the system. After the update, the next step 844 displays another page with the completed information requesting the user to test the application mount (addition). If it is determined at step 846 that the test at step 844 is successful, then at step 848 all configuration details are added to the repository of the Admin Services. The Admin Services also initiates an update cycle. If the test is not successful, the steps 840 on down re-execute. If the update cycle at step 848 is a success, the administrator is informed at step 854 that the application was mounted successfully, else and error page is displayed at step 852 on the console of the Administrator starting that the mount (or addition) of the application was unsuccessful.
Fig. 29 is a flow chart illustrating an administrative services exemplary procedure for starting mounted (added) applications in a cluster by an Administrator. Through the Management Services GUI, the administrator selects a "Start Application in a cluster" GUI at step 856. At step 858, a check is made of number of devices in a cluster and the configuration of respective applications thereon. If it is determined at step 860 that all contain the same configuration, then at step 864 the application service is made "Ready" in all devices in the cluster. If devices do not contain the same configuration, then step 862 executes to update configuration of all the respective devices. If all updates are successful on the devices, as determined by step 866, the application service will start in a cluster at step 870; else the step 868 informs administrator that a problem may exist with the cluster itself.

Fig. 30 is a flow chart illustrating an administrative services exemplary procedure for stopping a running application on a device by an Administrator. Through the Management Services GUI, the administrator selects a "Stop a device in a Cluster" GUI at step 872. At step 874, the device is marked so that it will not accept new request, and the next step 876 notifies all devices in the cluster accordingly. After a predetermined wait at step 878, step 880 runs shutdown script for shutting down the selected device. If at step 882 it is determined that, the shutdown was a success, the step 886 updates the table on the Admin Services, and step 888 notifies the administrator that the device has
stopped. If the shutdown of the device was not a success, step 884 notifies the administrator regarding some problem with the selected device.

Although the invention has been described in language specific to structural features and or methodological steps, it is to be understood that the invention defined in the appended claims is not necessarily limited to the specific features or steps described. Rather, the specific features and steps are disclosed as preferred forms of implementing the claimed invention.
WE CLAIM:

1. A method of processing user requests and a corresponding response to said user requests for accessing enterprise applications residing on at least one device, comprising:
   a. Receiving said user requests;
   b. Terminating said user requests for authentication and validation against an entitlement store before forwarding said request for further processing;
   c. Retrieving from said entitlement stores content policy rules associated with said enterprise application that said user request intends to access;
   d. Modifying said user request according to said content policy rules to generate a modified user request;
   e. Forwarding said modified user request to said intended enterprise application;
   f. Said intended enterprise application returning a response to said modified user request;
   g. Applying content policy rules to said returned response from said intended enterprise application;
   h. Modifying said return response based on said applicable content policy rules to generate a modified returned response; and
   g. Forwarding said modified returned response to said user request.
2. The method of claim 1, wherein each of a. through g. generates an audit information.

3. A method for process of providing and managing digital services comprising:
   a. Providing a central gateway module;
   b. Providing a plurality of central digital services modules at a central location connected to the central gateway module through a first local area network (LAN);
   c. Connecting the central gateway module to a public network;
   d. Providing a remote gateway module connected to the public network;
   e. Connecting one or more remote user access devices to the remote gateway module through a second LAN; and
   f. Providing digital services from the central digital services modules to the remote user access devices through a path that includes the first LAN, the central gateway module, the public network, and the second LAN.

4. The method of Claim 3, further comprising:
Dynamically re-writing application interface software of the central digital services modules, whereby said application is securely forwarded over any network of choice.

5. The method of Claim 3, further comprising:
Providing digital services using the digital services module that comprise at least one of the following core digital services: core digital services, directory services, enterprise storage, enterprise applications, Intranets, instant messaging, voice services, and security services.

6. The method of Claim 3, wherein the core digital services module comprises a module for file services, email services, and calendaring services.

7. The method of Claim 3 further comprising delivering the digital services over the public network using standard protocols.

8. The method of Claim 3, wherein the public network comprises the Internet.

9. The method of Claim 3 further comprising, providing a network services module connected to the central gateway module through the public network.
10. The method of Claim 3 further comprising collecting a fee for the services of providing and managing said digital services.

11. A system for providing and managing digital services, comprising:
    a central gateway module;
    a plurality of central digital services at a central location connected to the central gateway module through a first local area network (LAN);
    a network of choice connected to the central gateway module;
    remote gateway module connected to the network of choice;
    one or more remote user access units connected to the remote gateway module through a second LAN, wherein digital services from the central digital services modules are provided to the remote user access units through a path that comprises the first LAN, the central gateway module, the network of choice, the remote gateway module and the second LAN.
Fig. 2
FIG. 4
Fig. 5
Fig. 7

SUBSTITUTE SHEET (RULE 26)
Fig. 10
User initiates the request

UserService

Parse req

Read application data

Get the Application Security Params

Requires Auth, or not

Yes

No

Does Client Require Encryption or not

Yes

No

Check for existing state of user connection

Insecure No Encryption

Create a new connection & Redirect the request to container

Redirect the request to container

Fig. 11
Send the user a credential change
Get the old & new credential from the user
Check for the credential rules
Valid or not
No → Notify the error message
Yes → Send old & new credentials to repository
Get the response
Updated or not
Yes → Send confirmation msg. to the user
No →
Fig. 16
Fig. 17
Fig. 19
Fig. 20
Fig. 21

1. Is Secure Connection
   - Yes: Forward the request to login page.
   - No: Check user credentials.

2. Check user credentials
   - Yes: Is Valid User
     - Yes: Log in.
     - No: Login Failed
   - No: Is credential expired?
     - Yes: Change credential.
     - No: Create user session.

3. Create user session
   - Load user profile from CRL.
   - Display Console for Administration depending on his role.

4. IC
Fig. 22
Fig. 23
Fig 24
Fig. 25

SUBSTITUTE SHEET (RULE 26)
Via IMS Admin say:
Start a Device.

Run scripts for starting a device in cluster.

Go to Admin Services In Domain.

Adopt Service Not Found

Stop and Ask for Admin IP.

Get the latest Configuration from Admin Service.

Device starts with configuration.

Operation successful

Got active device list from Admin Service.

Update active device table of own.

Fig. 26

Update cycle in the cluster

Run Update Cycle for updating device tables.
Fig. 27

**SUBSTITUTE SHEET (RULE 26)**
Fig. 29

Via MS admin starts application service in cluster.

Check number of devices in cluster, check for configuration respective application.

All are contain same Config.

Make the application service 'Ready' in all devices in cluster.

Update Success on devices

Application service has started in a cluster.

Update configuration respective devices.

Some problem in cluster.
Fig. 30