US 20100100967A1

(19) **United States**

(12) **Patent Application Publication**

DOUGLAS et al.

(10) **Pub. No.: US 2010/0100967 A1**

(43) **Pub. Date: Apr. 22, 2010**

(54) **SECURE COLLABORATIVE ENVIRONMENT**

(76) Inventors: **JAMES E. DOUGLAS**, San Diego, CA (US); **Charles R. White**, Charles Town, WV (US); **Melvin D. Satterwhite, JR.**, Gainesville, VA (US)

Correspondence Address:
PATENT GROUP
C/O DLA PIPER US LLP
203 N. LASALLE ST., SUITE 1900
CHICAGO, IL 60601 (US)

(21) Appl. No.: **12/475,028**

(22) Filed: **May 29, 2009**

Related U.S. Application Data

(63) Continuation-in-part of application No. 12/142,232, filed on Jun. 19, 2008, which is a continuation-in-part of application No. 11/824,694, filed on Jul. 2, 2007, which is a continuation-in-part of application No. 11/257,421, filed on Oct. 24, 2005, which is a continuation-in-part of application No. 11/077,948, filed on Mar. 11, 2005, which is a continuation-in-part of application No. 10/892,584, filed on Jul. 15, 2004, now Pat. No. 7,676,834.

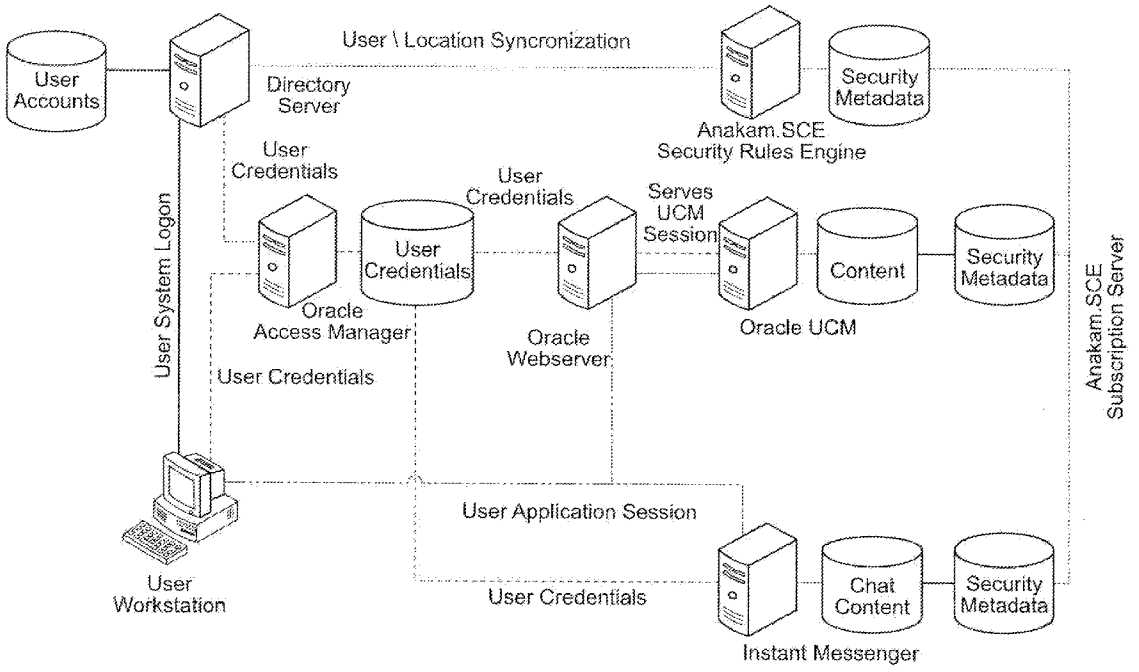
Publication Classification

(51) **Int. Cl.**
G06F 21/00 (2006.01)

(52) **U.S. Cl.** **726/27**; 726/26

(57) **ABSTRACT**

A secure collaborative environment to facilitate the sharing of confidential information between organizations, which can be used in conjunction with existing infrastructure.



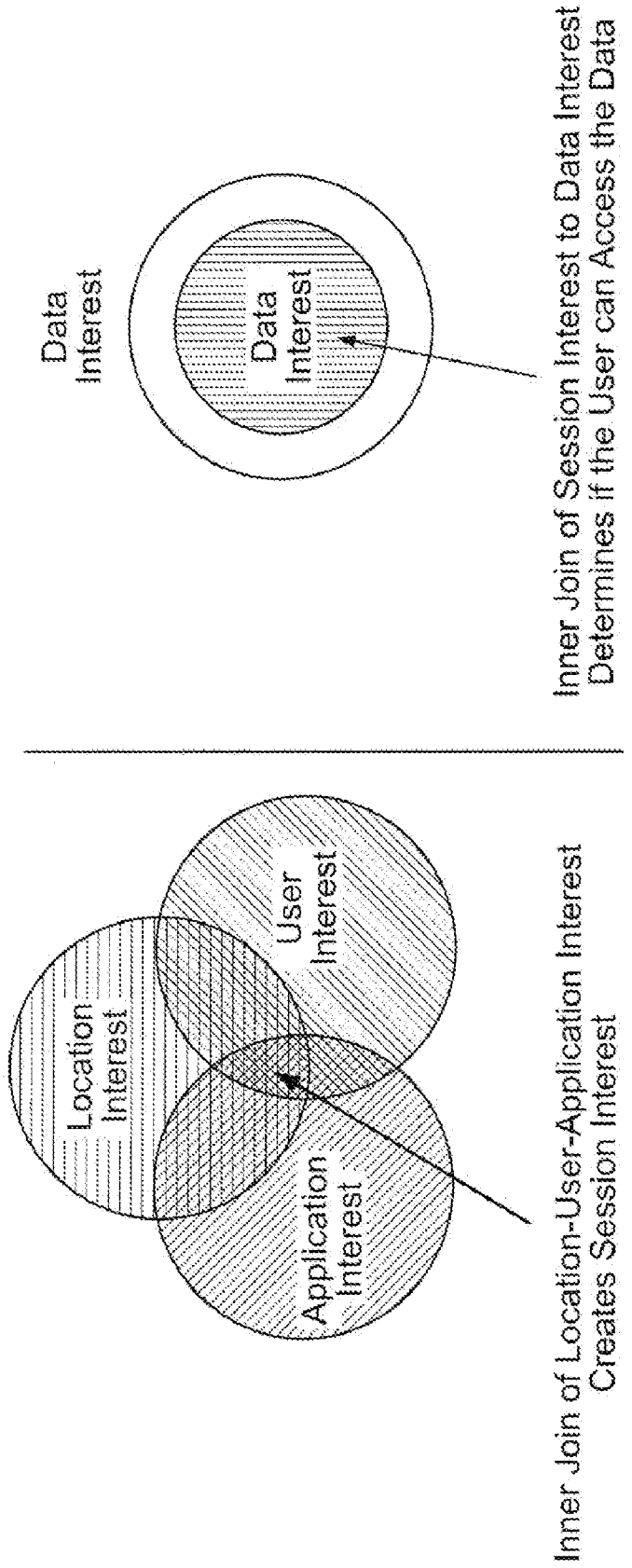

Data Interest
Data Interest
Inner Join of Session Interest to Data Interest
Determines if the User can Access the Data
Location Interest
User Interest
Application Interest
Inner Join of Location-User-Application Interest
Creates Session Interest

FIG. 1

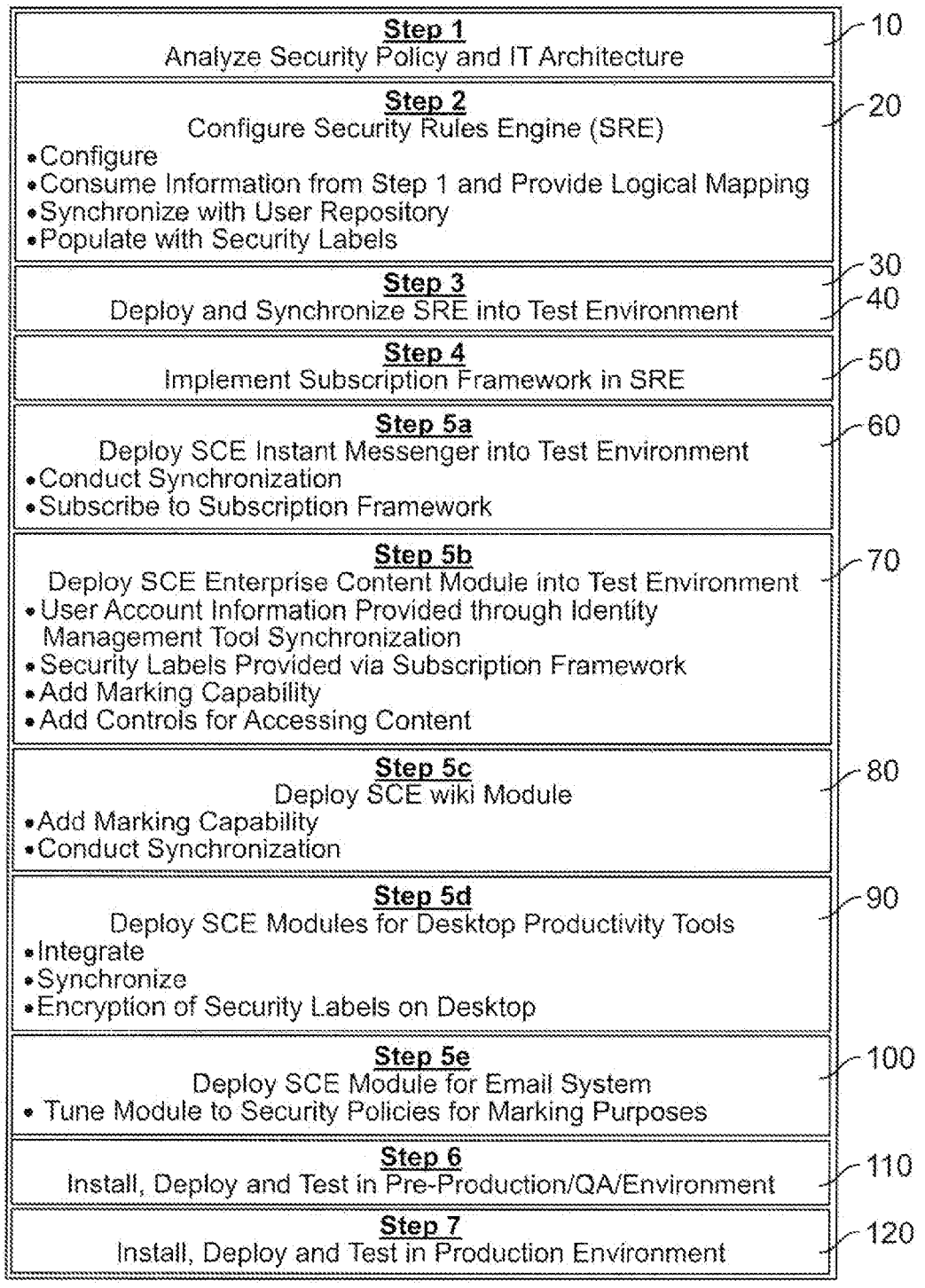

Step 1
Analyze Security Policy and IT Architecture
10
Step 2
Configure Security Rules Engine (SRE)
•Configure
•Consume Information from Step 1 and Provide Logical Mapping
•Synchronize with User Repository
•Populate with Security Labels
20
Step 3
Deploy and Synchronize SRE into Test Environment
30
40
Step 4
Implement Subscription Framework in SRE
50
Step 5a
Deploy SCE Instant Messenger into Test Environment
•Conduct Synchronization
•Subscribe to Subscription Framework
60
Step 5b
Deploy SCE Enterprise Content Module into Test Environment
•User Account Information Provided through Identity Management Tool Synchronization
•Security Labels Provided via Subscription Framework
•Add Marking Capability
•Add Controls for Accessing Content
70
Step 5c
Deploy SCE wiki Module
•Add Marking Capability
•Conduct Synchronization
80
Step 5d
Deploy SCE Modules for Desktop Productivity Tools
•Integrate
•Synchronize
•Encryption of Security Labels on Desktop
90
Step 5e
Deploy SCE Module for Email System
• Tune Module to Security Policies for Marking Purposes
100
Step 6
Install, Deploy and Test in Pre-Production/QA/Environment
110
Step 7
Install, Deploy and Test in Production Environment
120

FIG. 2

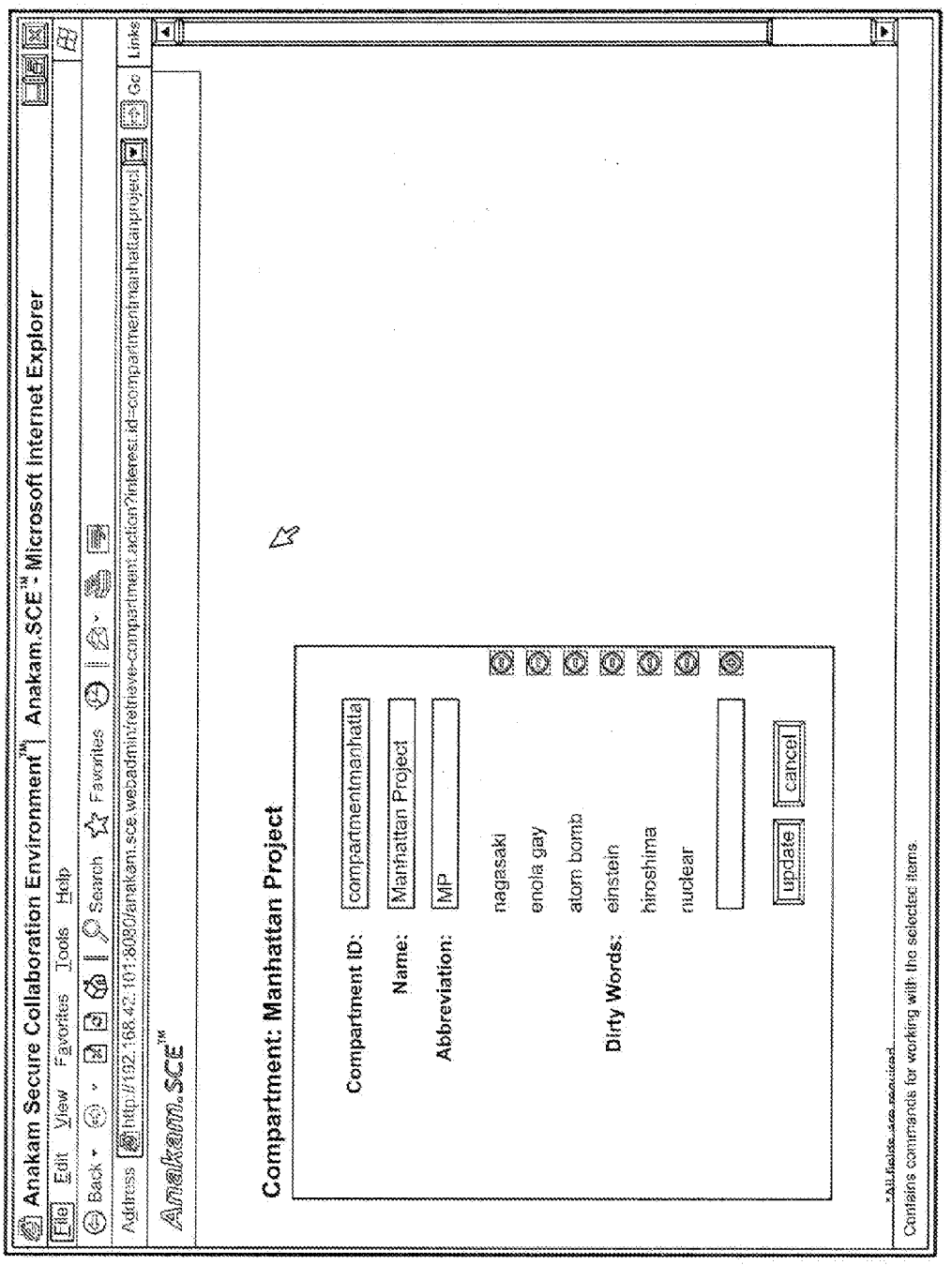

Anakam Secure Collaboration Environment™ | Anakam.SCE™ - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://192.168.42.101:8080/anakam.sce.webadmin/retrieve-compartment.action?interest.id=compartmentmanhattanproject
Links
Anakam.SCE™
Compartment: Manhattan Project
Compartment ID: compartmentmanhatta
Name: Manhattan Project
Abbreviation: MP
Dirty Words:
nagasaki
enola gay
atom bomb
einstein
hiroshima
nuclear
update
cancel
Contains commands for working with the selected items.

FIG. 3

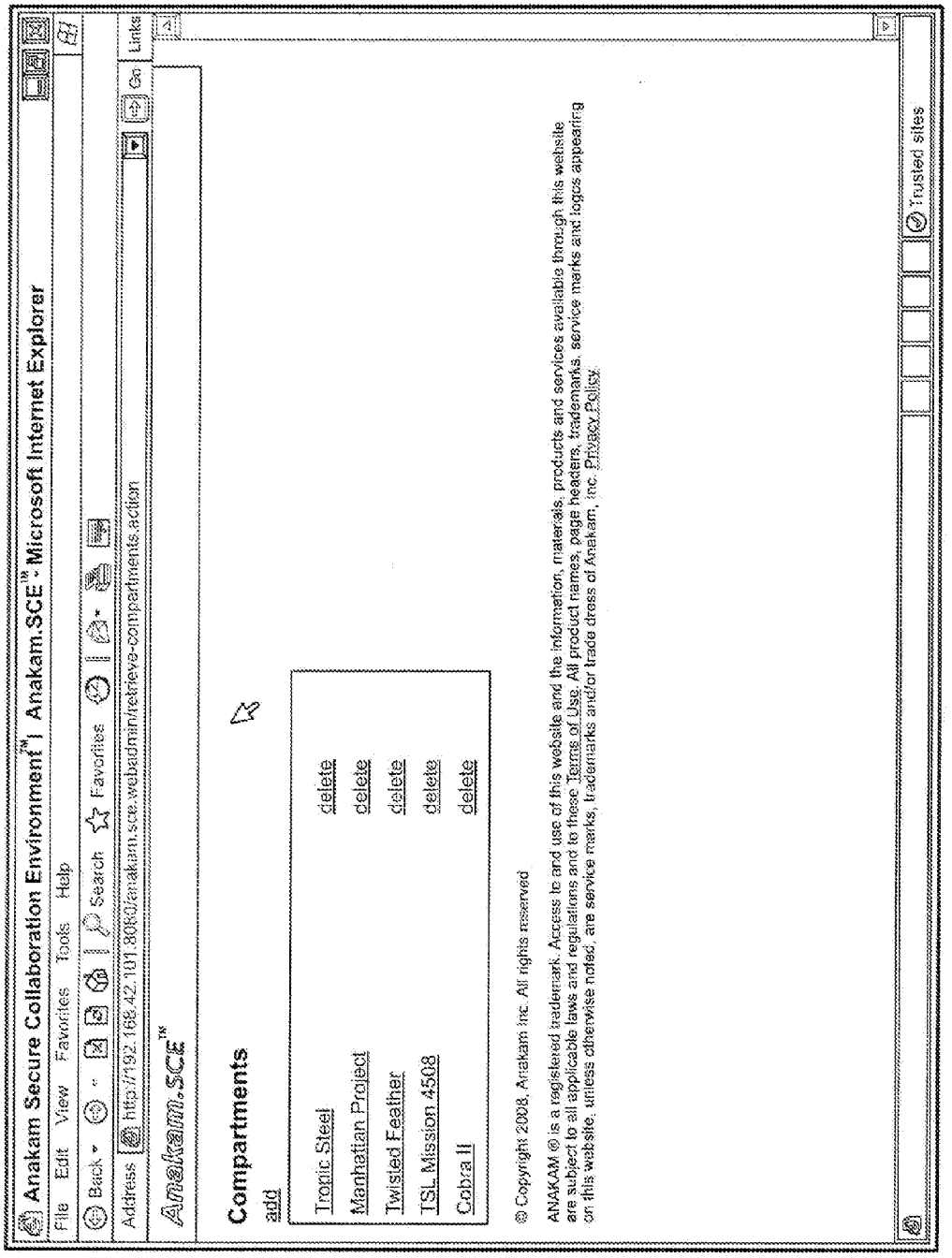

Anakam Secure Collaboration Environment™ | Anakam.SCE™ - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://192.168.42.101:8080/anakam.sce.webadmin/retrieve-compartments.action
Go Links
Anakam.SCE™
Compartments
add
Tropic Steel delete
Manhattan Project delete
Twisted Feather delete
TSL Mission 4508 delete
Cobra II delete
© Copyright 2008, Anakam Inc. All rights reserved
ANAKAM ® is a registered trademark. Access to and use of this website and the information, materials, products and services available through this website are subject to all applicable laws and regulations and to these Terms of Use. All product names, page headers, trademarks, service marks and logos appearing on this website, unless otherwise noted, are service marks, trademarks and/or trade dress of Anakam, Inc. Privacy Policy
Trusted sites

FIG. 4

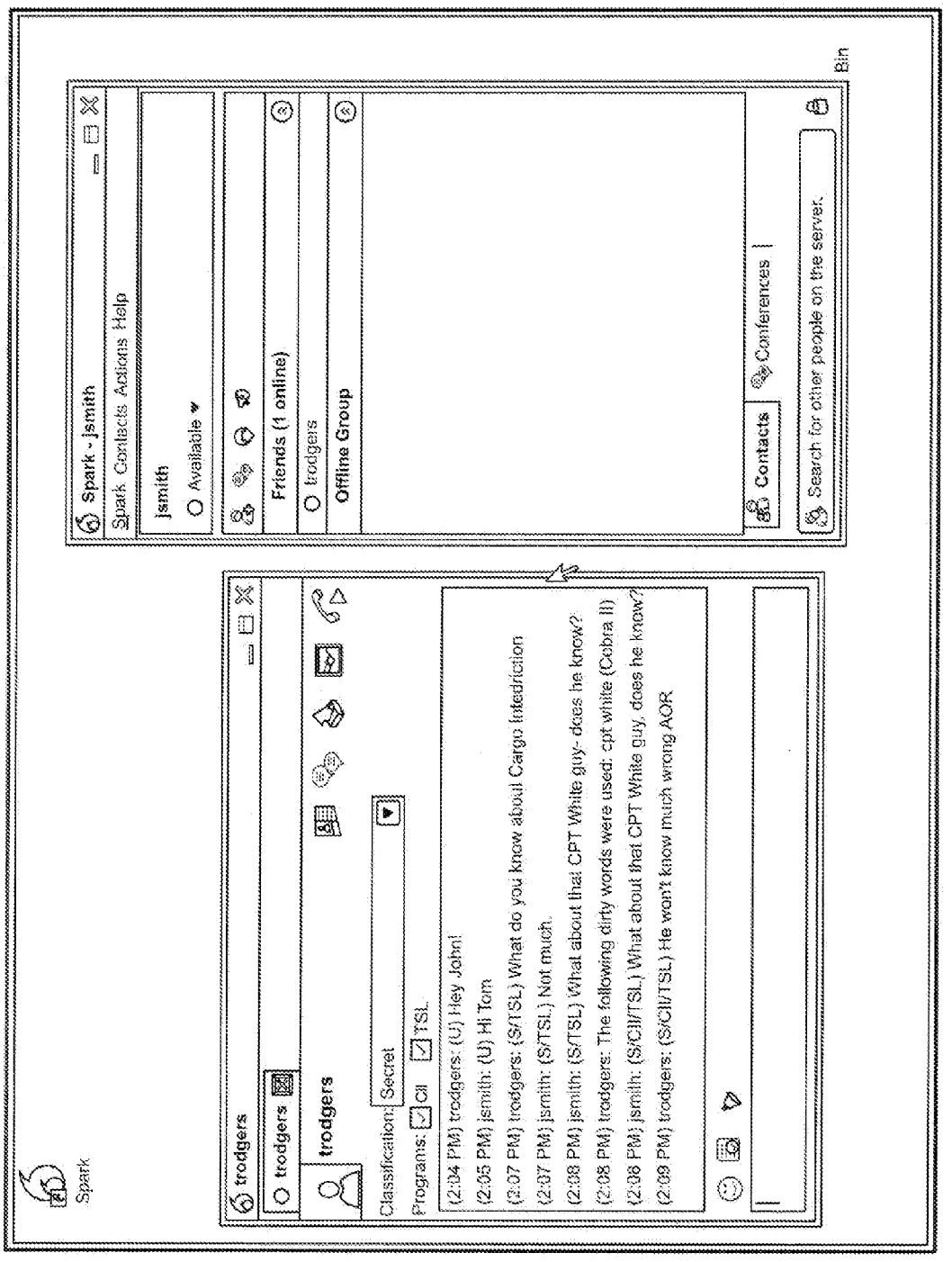

Spark
trodgers
trodgers
trodgers
Classification: Secret
Programs: CII TSL
(2:04 PM) trodgers: (U) Hey John!
(2:05 PM) jsmith: (U) Hi Tom
(2:07 PM) trodgers: (S/TSL) What do you know about Cargo Infestriction
(2:07 PM) jsmith: (S/TSL) Not much.
(2:08 PM) jsmith: (S/TSL) What about that CPT White guy- does he know?
(2:08 PM) trodgers: The following dirty words were used: cpt white (Cobra II)
(2:08 PM) jsmith: (S/CII/TSL) What about that CPT White guy, does he know?
(2:09 PM) trodgers: (S/CII/TSL) He won't know much wrong AOR
Spark - jsmith
Spark Contacts Actions Help
jsmith
Available
Friends (1 online)
trodgers
Offline Group
Contacts
Conferences
Search for other people on the server.
Bin

FIG. 5

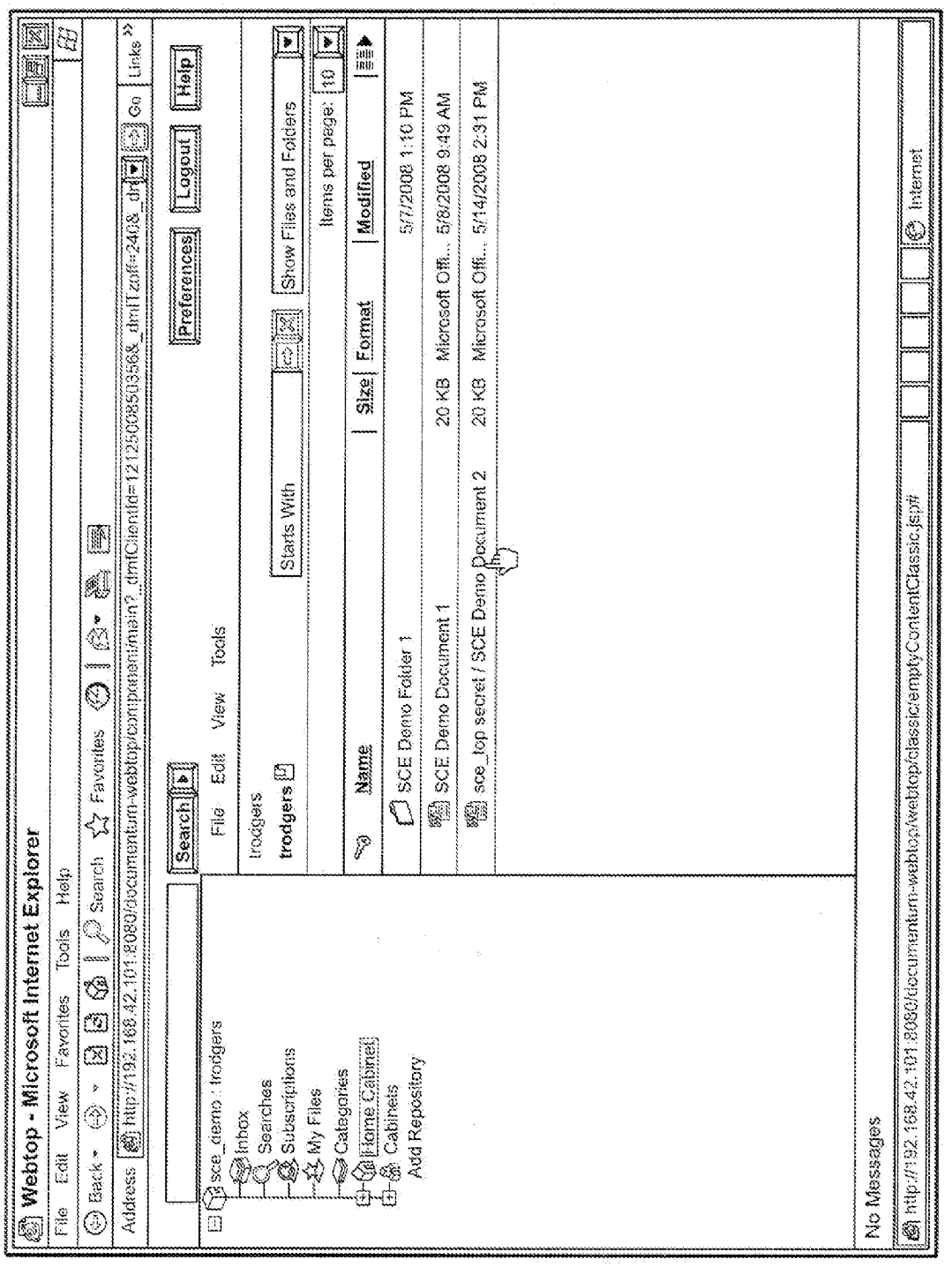

Webtop - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://192.168.42.101:8080/documentum-webtop/component/main?_dmfClientId=12125008503568_dmfTzoff=2408_dr
Links
Search
File Edit View Tools
Preferences
Logout
Help
sce_demo : trodgers
Inbox
Searches
Subscriptions
My Files
Categories
Home Cabinet
Cabinets
Add Repository
trodgers
trodgers
Starts With
Show Files and Folders
Items per page: 10
Name
Size
Format
Modified
SCE Demo Folder 1
5/7/2008 1:10 PM
SCE Demo Document 1
20 KB Microsoft Offi... 5/8/2008 9:49 AM
sce_top secret / SCE Demo Document 2
20 KB Microsoft Offi... 5/14/2008 2:31 PM
No Messages
http://192.168.42.101:8080/documentum-webtop/webtop/classic/emptyContentClassic.jsp#
Internet

FIG. 6

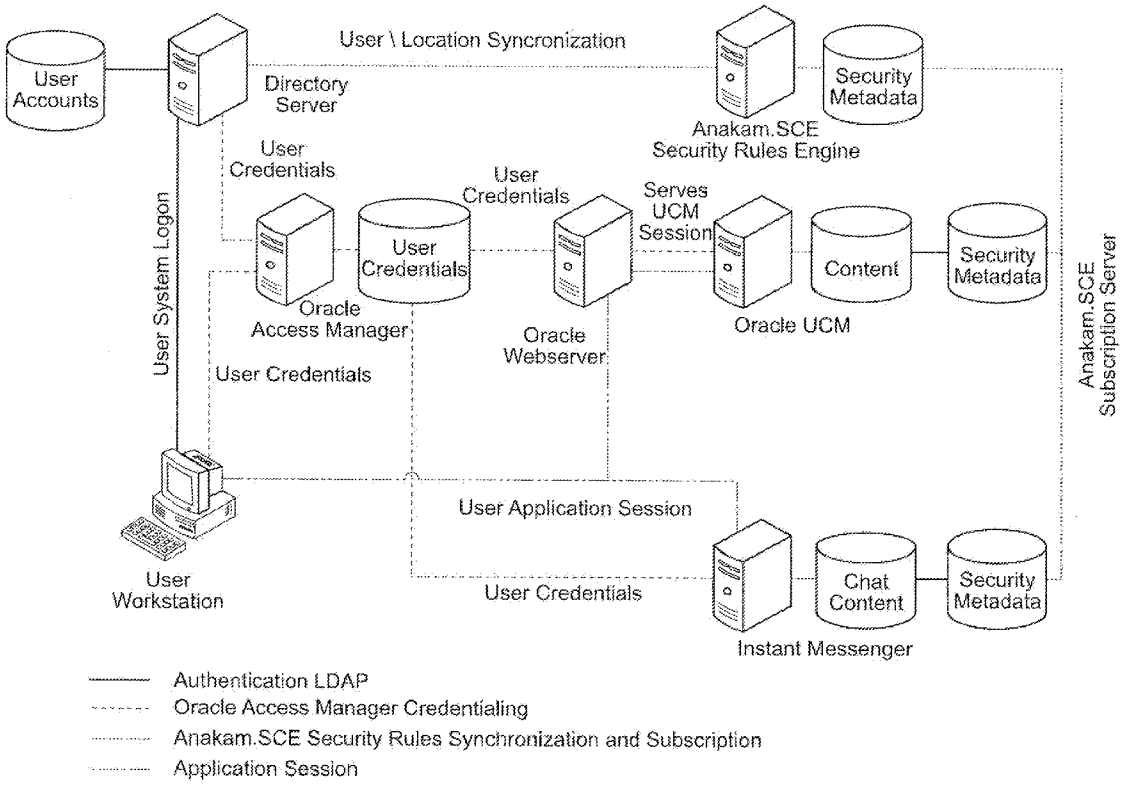

User Accounts
Directory Server
User \ Location Syncronization
Anakam.SCE Security Rules Engine
Security Metadata
User Credentials
User Credentials
Serves UCM Session
User Credentials
Oracle Access Manager
Oracle Webserver
Oracle UCM
Content
Security Metadata
User System Logon
User Credentials
Anakam.SCE Subscription Server
User Application Session
User Workstation
User Credentials
Chat Content
Security Metadata
Instant Messenger
Authentication LDAP
Oracle Access Manager Credentialing
Anakam.SCE Security Rules Synchronization and Subscription
Application Session

FIG. 7

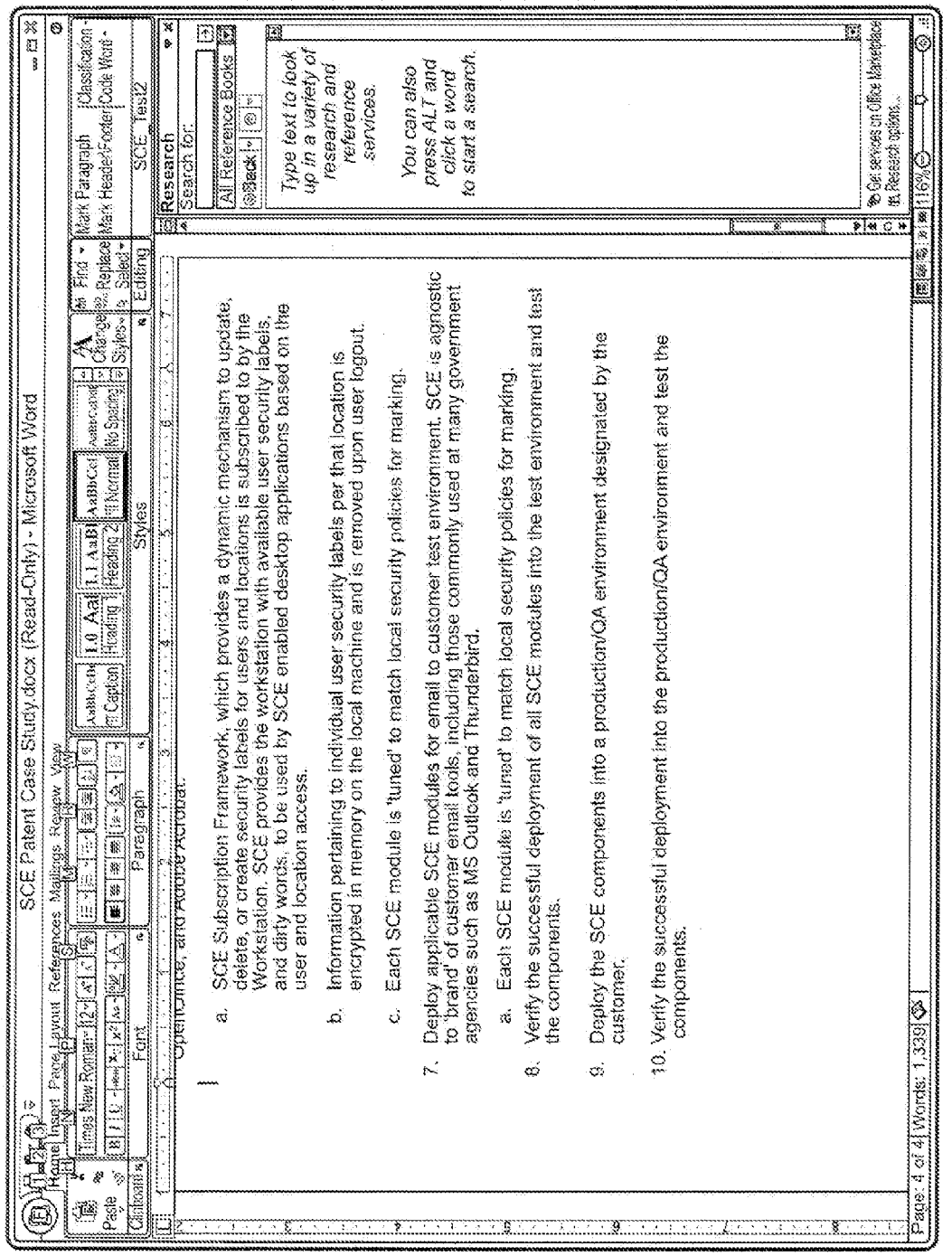

SCE Patent Case Study.docx (Read-Only) - Microsoft Word
Optimize, and Acrobat
a. SCE Subscription Framework, which provides a dynamic mechanism to update, delete, or create security labels for users and locations is subscribed to by the Workstation. SCE provides the workstation with available user security labels, and dirty words, to be used by SCE enabled desktop applications based on the user and location access.
b. Information pertaining to individual user security labels per that location is encrypted in memory on the local machine and is removed upon user logout.
c. Each SCE module is 'tuned' to match local security policies for marking.
7. Deploy applicable SCE modules for email to customer test environment. SCE is agnostic to 'brand' of customer email tools, including those commonly used at many government agencies such as MS Outlook and Thunderbird.
a. Each SCE module is 'tuned' to match local security policies for marking.
8. Verify the successful deployment of all SCE modules into the test environment and test the components.
9. Deploy the SCE components into a production/QA environment designated by the customer.
10. Verify the successful deployment into the production/QA environment and test the components.
Research
Search for:
All Reference Books
Type text to look up in a variety of research and reference services.
You can also press ALT and click a word to start a search.
Get services on Office Marketplace
Research options...
Page: 4 of 4 Words: 1,339

FIG. 8

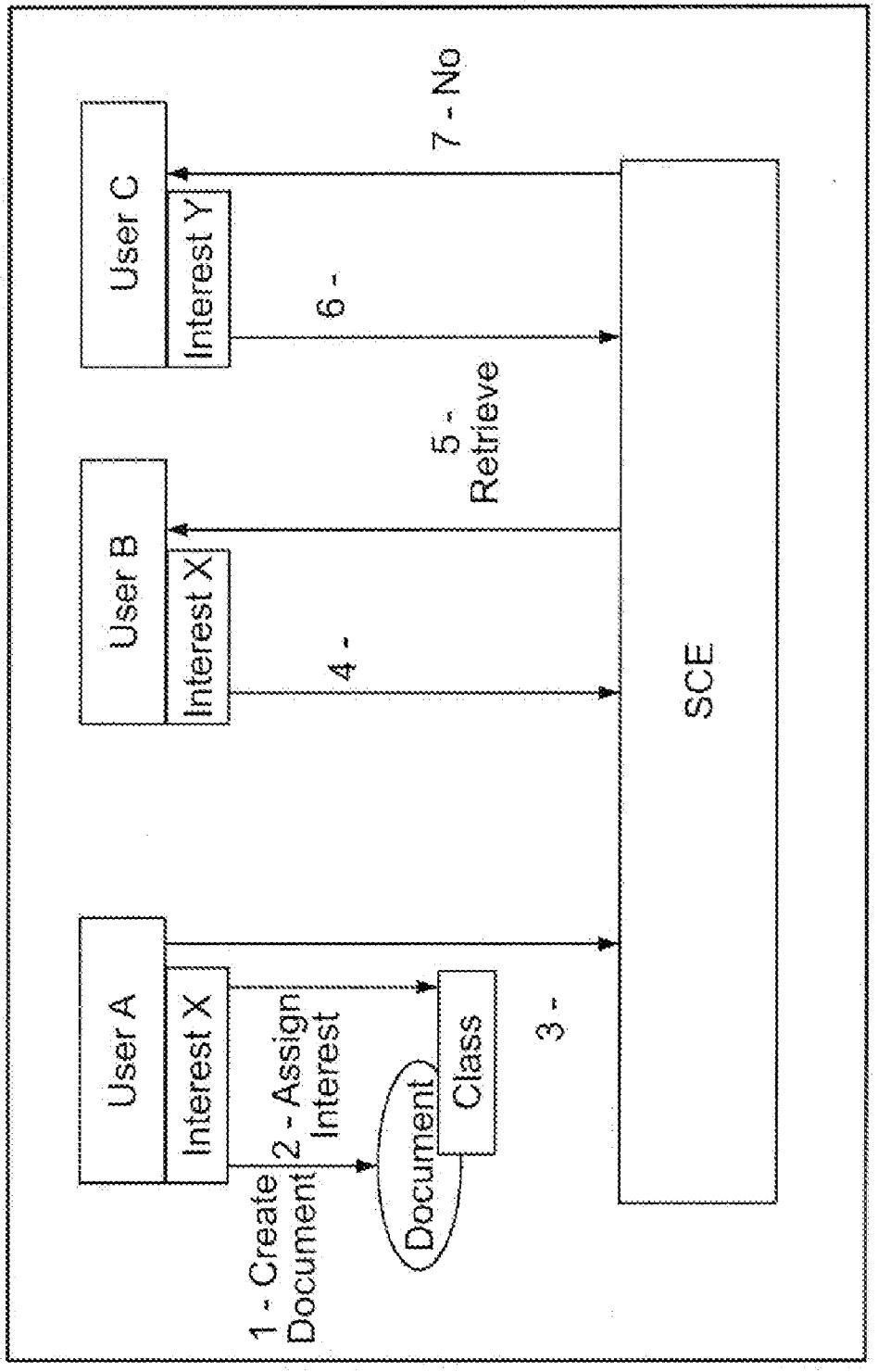

User A
Interest X
1 - Create Document
2 - Assign Interest
Document
Class
3 -
SCE
User B
Interest X
4 -
5 - Retrieve
User C
Interest Y
6 -
7 - No

FIG. 9

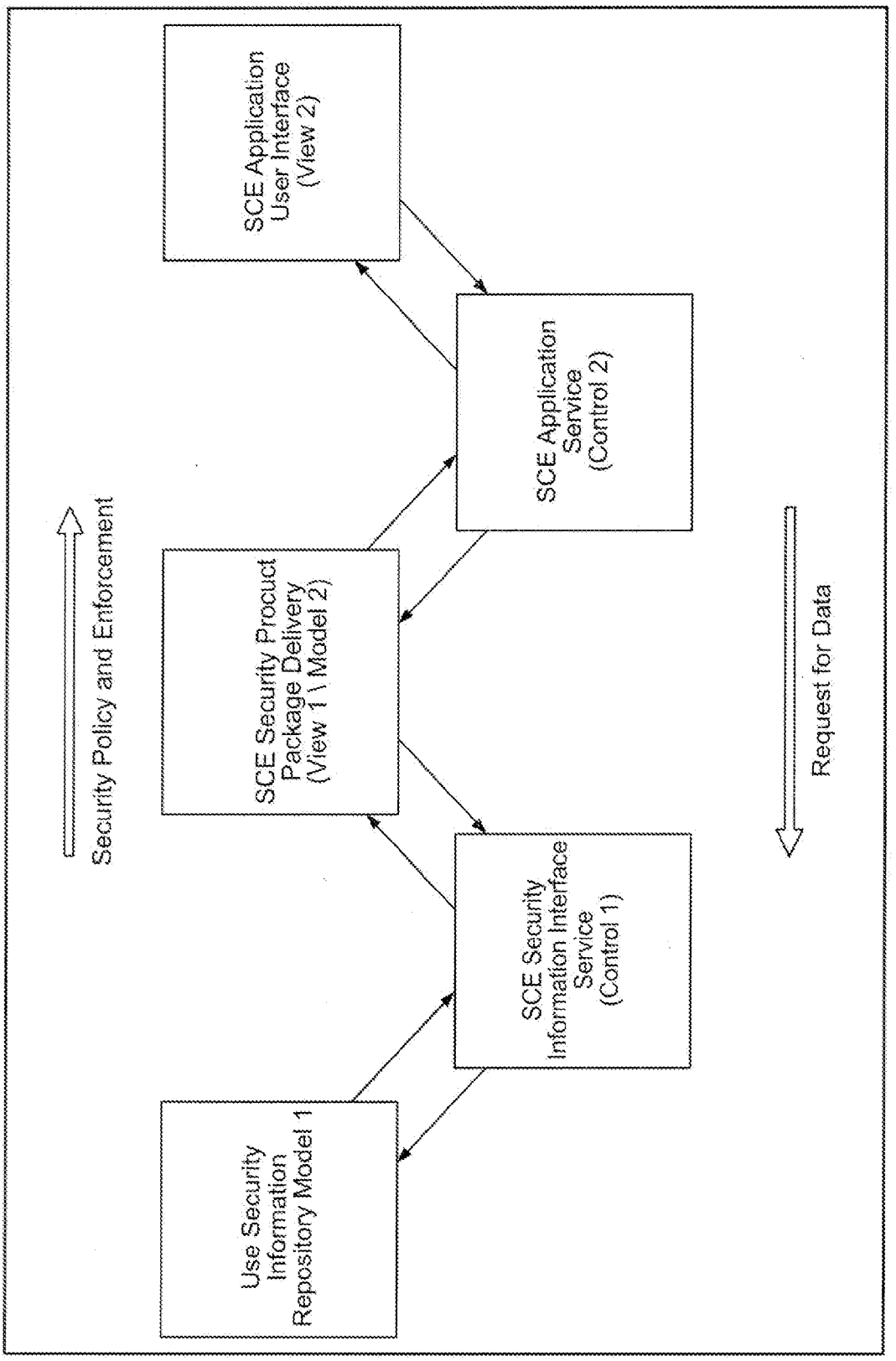

SCE Application User Interface (View 2)
SCE Application Service (Control 2)
SCE Security Procuct Package Delivery (View 1 \ Model 2)
SCE Security Information Interface Service (Control 1)
Use Security Information Repository Model 1
Security Policy and Enforcement
Request for Data

FIG. 10

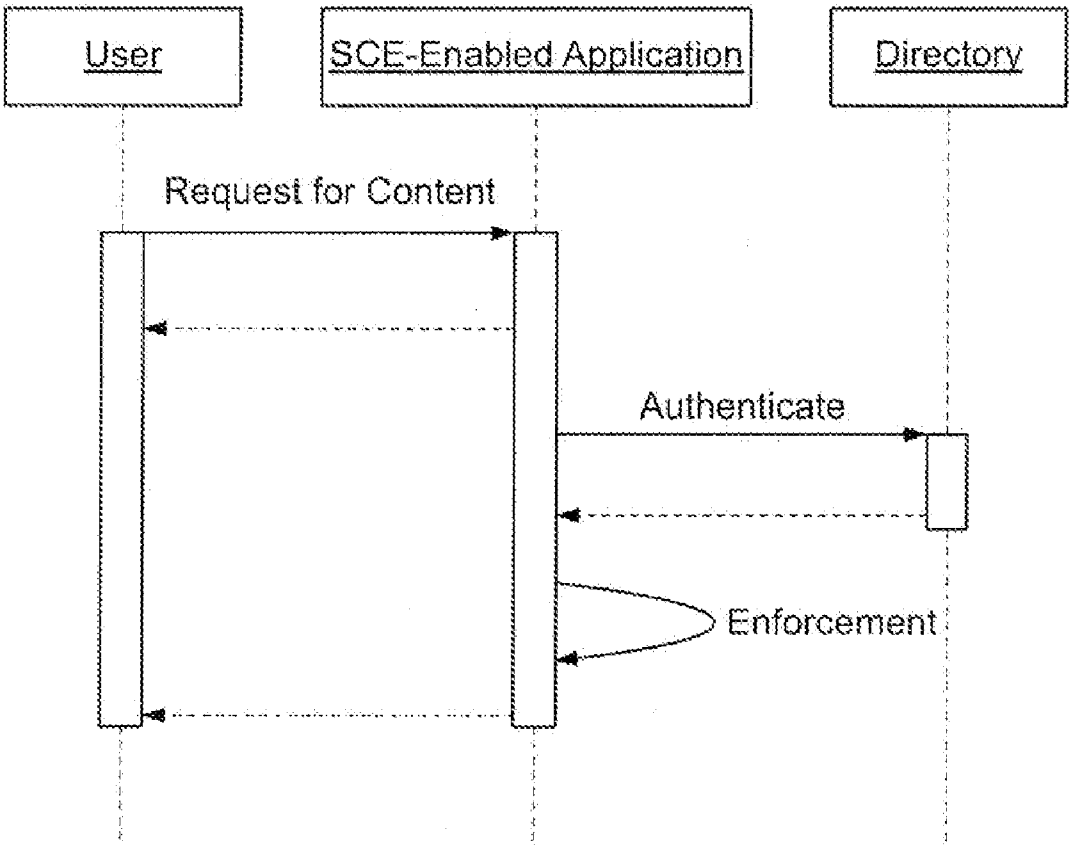

User
SCE-Enabled Application
Directory
Request for Content
Authenticate
Enforcement

FIG. 11

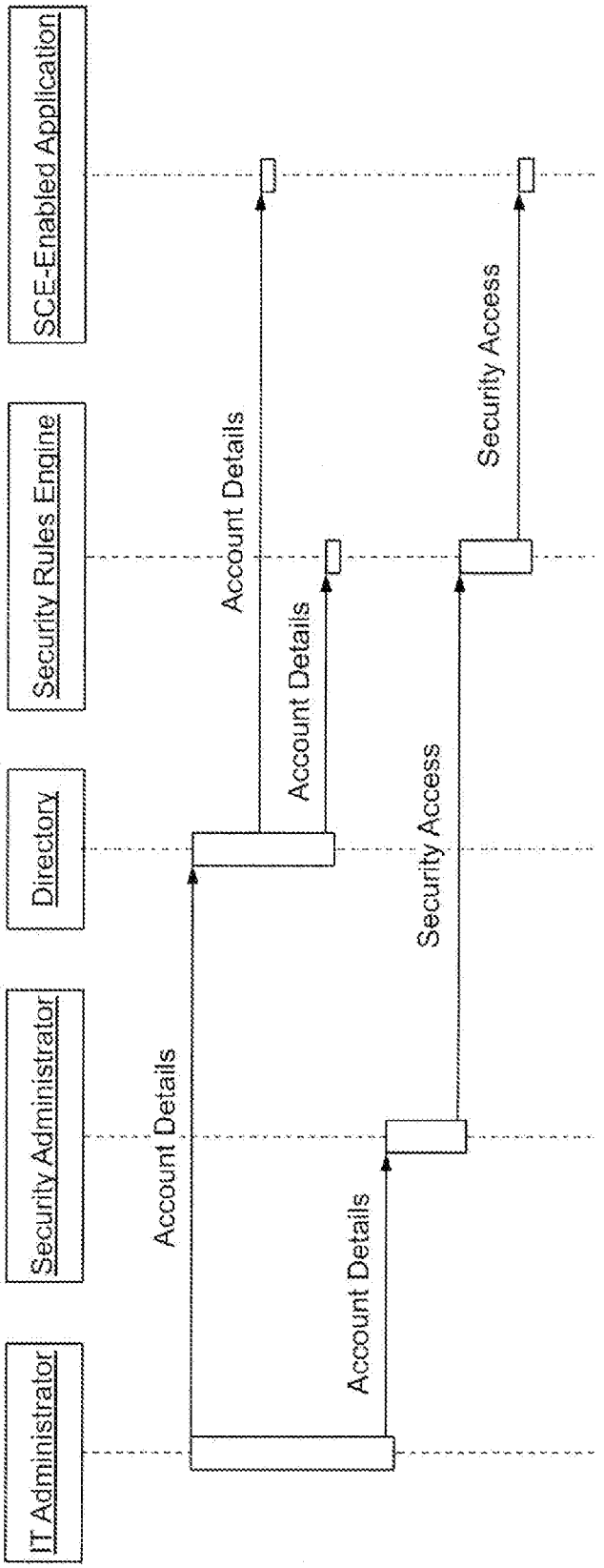

IT Administrator
Security Administrator
Directory
Security Rules Engine
SCE-Enabled Application
Account Details
Account Details
Account Details
Account Details
Security Access
Security Access

FIG. 12

SECURE COLLABORATIVE ENVIRONMENT

RELATED APPLICATIONS

[0001] This application is a continuation-in-part of and claims priority from 12/142,232 filed Jun. 19, 2008, which is a continuation in part of U.S. patent application Ser. No. 11/824,694, filed Jul. 2, 2007, which is a continuation-in-part of U.S. patent application Ser. No. 11/257,421, filed Oct. 24, 2005, which is a continuation-in-part of U.S. patent application Ser. No. 11/077,948, flied Mar. 11, 2005, which in turn is a CIP of Ser. No. 10/892,584, filed Jul. 15, 2004, all of which are incorporated herein by reference and from all of which is priority claimed.

FIELD OF THE INVENTION

[0002] The systems, methods, and graphical user interfaces disclosed herein relate to communications generally, and more specifically to a Secure Collaborative Environment comprising a Security Rules Engine in which access to secure information is granted or denied based on the application of an organization's security policy. The security policy may include authorization criteria for users, physical and logical access locations, and computing applications, as well as descriptive security details of the target information. The Security Rules Engine interprets and enforces the security policy to appropriately allow, restrict, or deny access to information and interaction between users.

[0003] The present invention provides the means to utilize web services and application modifications to apply authorization based access control to items (users, facilities, data, and applications) by security specific interest in secure data networks by establishing programmatic mechanisms that compare items based on a union of item accesses. Additionally the invention provides a subscription framework to provide real-time updates for security specific interests for users, facilities and applications in secure data networks. In one embodiment, the present invention may comprise a secure Instant Messaging function. This system integrates the Security Rules Engine into the customary instant messaging paradigm, allowing the organization's security policy to be applied to instant messaging by appropriately allowing, restricting, or denying access to instant messages and communication between users.

[0004] In another embodiment the present invention may comprise a secure wiki system. This system integrates the Security Rules Engine into the known wiki paradigm, allowing security policy to be applied to wiki entries by appropriately allowing, restricting, or denying access to wiki content and communication between users.

[0005] In another embodiment the present invention may comprise a Secure Blog system. This system integrates the security rules engine into the known blog paradigm, allowing security policy to be applied to blogging by appropriately allowing, restricting, or denying access to blog entries and communication between users. These three factors (instant messaging, wiki, and blog) may well be present on the same embodiment as is desired or in desired combination.

BACKGROUND OF THE INVENTION

[0006] Collaborative tools are an integral part of commercial business and government operations, facilitating communication through organized information sharing, rapid information access, and real-time information availability. Across both commercial and government organizations, information is commonly made available and on a "need to know" or other similar basis in order to help prevent the compromise of sensitive information. Projects of all scales require a degree of collaboration among participants, whom may span across multiple organizations. With a myriad of different participants, each of whom with their own particular need to know, there arises the need to provide a secure, adaptable environment in which participants can share potentially sensitive information.

[0007] Numerous known approaches exist to collaboration including email, teleconferencing, video conferencing, and File Transfer Protocol (FTP). While these solutions do much to facilitate communication and information sharing, they do little or nothing to enforce security.

[0008] There, thus, is a need for a well-developed, widely acceptable system with the capacity to dynamically interpret and implement the security policy of commercial and government organizations in a collaborative environment. This has led to the development of numerous one-off solutions, none of which scale to the usability and compatibility extents necessary for widespread adoption. These systems typically require significant administrative overhead and are time-consuming to maintain. They often rely mainly or entirely on the auditability of information access to enforce security policy and limit access to information, significantly limiting their ability to provide a real time information exchange. Existing secure collaboration products are therefore either point solutions limited to a single operating system, desktop applications, enterprise content management (ECM) systems, or stovepipe systems that segregate data into common user groups, minimizing the ability for users to collaborate and share information across groups.

[0009] The prevalence of collaborative tools combined with the insufficient means of securing collaboration presents a security risk so severe that collaborative tools are prohibitively expensive to secure, or are simply not used. This results in both loss of security and loss of the ability to collaborate. Thus, there is a need for a better SCE that solves these problems.

SUMMARY OF THE INVENTION

[0010] Aspects of the systems, methods, and user interfaces disclosed herein comprise a Secure Collaborative Environment that is integrated into an existing organization's infrastructure to allow secure information sharing by prohibiting access to information to those users whom do not have a "need to know." The systems, methods, and user interfaces disclosed herein may be implemented in existing collaborative software environments as well as future collaborative software environments.

[0011] The Secure Collaborative Environment (SCE) of the present invention is a security framework that is integrated into an existing collaborative environment in order to facilitate secure compartmented information sharing. Its use in document management systems, typically referred to as Enterprise Content Management (ECM) systems, leverage the inherent ECM benefits and capabilities while adding integrated security policy enforcement and management. The SCE ensures that information is available only to those who are allowed to view it and only at locations they are allowed to view it from by enforcing the appropriate entity security policy at the object level.

[0012] By using such a system users are provided with the ability to easily create and share sensitive information in a controlled, audited and appropriate manner. This information is protected by prohibiting unauthorized access as governed by a user's "need to know" while at the same time maximizing the ability of the users to collaborate, thereby recognizing and encouraging the required "responsibility-to-share" that is necessary in collaborative ventures.

[0013] Another fundamental of the SCE is abstraction of interests and items. SCE capabilities define clearances, programs, and handling instructions as interests. In practice an interest can be considered a security label. An item is defined as a user, location, system, and application and has an interest or set of interests associated with it. In the simplest case, access is the process where the interest of items is compared and access is granted based on the logical union of the interests of items. If the interest of the first item is equal to or greater than the interest of the item being accessed than access is granted.

[0014] A more realistic case is defined as follows. As shown in Figure One, the interest of a user, location, and application is compared. The union of those interests is compared to the interest of a file. If the union of interest from the user-location-application to the file is sufficient then access is granted to the user at that location with the application accessing the file.

[0015] The Secure Collaborative Environment described and disclosed herein is agnostic to the platform on which it is used, and can be used in conjunction with various desktop productivity tools such as Microsoft Office, Open Office, and Adobe Acrobat; email tools such as Microsoft Outlook and Thunderbird; web browsers such as Internet Explorer and Firefox; Operating Systems such as Windows, UNIX, and LINUX; and ECM solutions such as Open Text Livelink, EMC^2 Documentation, and Oracle Universal Content Management (UCM) system, as well as open source collaboration tools such as Drupal.

[0016] Additionally, the SCE disclosed herein can be used in conjunction with automated tools such as wikis, XMPP-based (also known as Jabber-based) instant messaging/secure chat systems, secure blogs, and other processes that improve productivity for users and administrators.

[0017] The SCE disclosed herein also can be used with ECM systems to leverage inherent ECM benefits and capabilities while adding integrated security policy enforcement and management. Its can be integrated with a user's desktop software to assist and enforce proper sensitivity marking and identification of documents that users are manipulating. The SCE ensures that information is sent only to those users that are allowed to view it and only at locations at which they are allowed to view it by initiating security policy at the object level.

[0018] Moreover, the SCE disclosed herein implements an organization's security policy at the object level through the installation of a Security Rules Engine, as described in more detail below.

[0019] The SCE disclosed herein also provides automated security tools for document and email security marking and "dirty word" searches. In this context a "dirty word" is a word, phrase, expression or concept that requires security control based on the organization's security policy and is detected by pattern recognition algorithms. This automation of policy and functionality minimizes the user's responsibility of focusing on security policy. It also separates information technology administration from information assurance/security administration, allowing for rigorous certification and accreditation such as Director of Central Intelligence Directive (DCID) 6/3 Protection Level 4. In this context certification is a measure of the effectiveness of the technical and non-technical security features within or supporting the information system. Accreditation is a formal declaration by an organization that an information system is approved to operate in a particular security mode using a prescribed set of safeguards at an acceptable level of risk. DCID 6/3 establishes the security policy and procedures for storing, processing, and communicating classified U.S. intelligence information in information systems:

[0020] Yet another aspect of the methods and systems disclosed herein is the abstraction of interests and items. An interest is a clearance, program, handling instruction, or other criteria related to defining security. An item is a user, location, system, or application. Interests are applied to items to determine whether access is allowed. In the simplest case, access is the process where the interests of items are compared and access is granted based on a logical union of the interest of the items. That is, if the interest of the first item if equal to or greater than the interest of the item that is being accessed, then the access is granted.

BRIEF DESCRIPTION OF THE DRAWINGS

[0021] Aspects of the present invention are illustrated by way of example, and not by way of limitation, in the accompanying drawings in which like reference numerals refer to similar elements. In some instances structures and devices that are known in the art are shown in block form for simplicity's sake. In these drawings:

[0022] Figure One depicts the relationship between abstracts and interest.

[0023] Figure Two depicts the steps required to implement the Secure Collaborative Environment (SCE).

[0024] Figures Three and Four provides SCE Security Rules Engine (SRE) Graphical User Interface (GUI) screen shot examples.

[0025] Figure Five provides an SCE Instant Messenger (IM) screen shot example.

[0026] Figure Six provides an SCE document control screen shot example.

[0027] Figure Seven is a drawing depicting the components of an SCE enabled IM and Enterprise Content Management (ECM) system—in this case Oracle Universal Content Manager—deployment.

[0028] Figure Eight depicts the integration of SCE with a desktop application—in this case Microsoft Office 2007.

[0029] Figure Nine shows how the SCE ECM service determines the range of information that a user is permitted to access.

[0030] Figure Ten shows how a Nested Model-View-Controller (NMVC) design pattern is utilized for SCE functionality.

[0031] Figure Eleven is a cross functional flowchart demonstrating how content is accessed.

[0032] Figure Twelve is a cross functional flowchart demonstrating how account management is accomplished.

DETAILED DESCRIPTION

[0033] A Secure Collaborative Environment (SCE) is described herein. In the following description, illustrative

examples are provided for purposes of explanation and are not intended to be limiting. Numerous details are provided but it will be apparent that aspects of the present invention can be practiced without these specific details or in other appropriate and suitable ways.

[0034] The system, method, and user interface of aspects of the present invention will be disclosed generally through explanation of a typical SCE deployment, for illustrative purposes only. Obviously requirements vary based on an agency's or organization's specific security policies and information technology architecture, but the example below is representative of a general implementation.

[0035] The primary steps typically taken during an SCE implementation are shown in Figure Two and described as follows.

[0036] First, at **10**, the customer's existing Security Policy and Information Technology (IT) architecture is analyzed. This provides a foundation for populating the Security Rules Engine (SRE) of the SCE and allows for the development of basic work flows that represent the customer's business environment.

[0037] The SCE SRE, which provides a framework for defining and implementing an agency's security policy, is next configured for the specific customer environment based on organizationally specified sensitive words whose dissemination needs to be controlled. See **20**. Through this configuration process, the SRE allows enabled applications to search for these patterns of fragments, words, expressions and phrases as a method of data spill prevention, i.e. preventing the unintentional release of secure information into an insecure environment.

[0038] The Security Rules Engine provides a logical mapping of user security labels to user accounts for consumption of enabled applications. Security labels are assigned for all "dirty words" which are words, phrases or regular expressions that directly relate to the security label. The SCE enabled applications have the ability to search content for "dirty words" and restrict or deny transmittal of content based on "dirty words".

[0039] The SRE is populated with the organization's users and locations through synchronization with the organization's user repository. The user repository is generally a database that is embodied via a tool such as a Lightweight Directory Access Protocol (LDAP) repository. The Security Rules Engine uses network items in the form of Internet Protocol (IP) ranges, device tokens, and geospatial coordinates to represent locations. The SRE uses application names to represent organization applications.

[0040] The Security Rules Engine is populated with the previously mentioned security labels (classifications, compartments, personal health information tags, etc.) to represent attributes which form a means for marking and controlling content within SCE enabled applications. The SRE provides interfaces to apply security labels to users, locations, and applications for the enforcement of access control. Access control is the mechanism by which services know whether to honor or deny access requests. Authorization in this context is the means of expressing access policy by explicitly granting a right of an entity to access the resource based on security attributes (interests) that are designated for the given customer environment in the form of security labels.

[0041] Next, at **30**, the Security Rules Engine is deployed into a customer test environment that simulates its operational environment, before actually being implemented into the actual environment. This step consists of implementing the associated rules and processes of the customer security policy within the SRE Administrative Graphical User Interface (GUI). The SRE Admin GUI is the conduit for the user to implement customer security policy into the SRE. Figures Three and Four demonstrate interfaces for establishing and manipulating a security attribute (in this case a Compartment). Figure Four specifically demonstrates an interface for deleting Compartments from the Security Rules Engine. The SRE in turn provides a method of interpreting customer security policy in order to implement customer security policy with SCE enabled applications.

[0042] Customers use the Security Rules Engine via the SRE Admin GUI to assign security labels to users, locations and applications and to assign security controls into their environment based on those labels. The Security Rules Engine shows customer security policy in a logical form that can be implemented by SCE enabled applications as a method of Authorization Based Access Control which can work in tandem with an existing user Role Based Access Control mechanism—usually through an implementation of LDAP.

[0043] Next, at **40**, after the Security Rules Engine is shown to meet administrative and operational requirements in the test environment, the SRE is synchronized with the customer SRE environment. User account and security information is typically managed by Identity Management tools such as Oracle Access Manager. This information is consumed by the SRE through a customizable synchronization connecter. The connecter is matched to the specific identity management tools employed in the customer environment and, by adhering to a common interface for representing and sharing security information, enables synchronization with the SRE through a simple mapping of the customer security model.

[0044] SRE location association is established by IP address, device tag, or geospatial coordinates for the customer environment. User locations are mapped to program accesses through the Security Rules Engine for consumption of enabled applications through the SCE Subscription Framework, which synchronizes real-time security information between the SCE SRE and SCE-enabled applications.

[0045] Next, at **50**, the SCE Subscription Framework is implemented. It provides a dynamic mechanism to update, delete, or create security labels for users and locations for SCE enabled applications, and is implemented in the existing customer environment. The Subscription Framework is implemented in SCE application modules and the Security Rules Engine. These services propagate real-time changes to published security products. The Subscription Framework is the mechanism in which SCE enabled applications consume security labels, users, locations, and applications from implementation of customer security policy. Appendix A is a sample listing of codes the SCE Subscription Framework, written in Java that can be used for a variety of SCE enabled applications that support Java.

[0046] Auditing and audit reduction for the Security Rules Engine are also provided. Audit reduction provides customers with a specific view of SCE audit information determined on criteria such as applications, users or locations for the purpose of a targeted search of SCE.

[0047] Next, at **60**, an SCE XMPP-based Instant Messenger (IM) feature may be deployed into the customer test environment. The SCE Instant Messenger is a real time communications tool that implements customer security policy based on security rules applied to users, and locations within

the SCE Security Rules Engine. Appendix B includes code for implementation of the SCE IM.

[0048] As is seen in Figure Five, SCE IM provides current user and facility information through the Subscription Framework. When a user starts an SCE IM client, it initially contacts the SRE, and requests a subscription to the user's security information. This information, which includes the secure channels to which the user has access, allows the SCE IM client to enforce marking of outgoing data, as well as prevents data leaks to destinations with insufficient security channels. The subscription is kept up to date in real time, so any changes to the user's security channels in the SRE will be propagated to and updated in the SCE IM client while it is in use.

[0049] The SCE IM is synchronized with the customer environment through its own LDAP (or comparable software) synchronization mechanism. User account information is provided through Oracle Access Manager (or comparable software) synchronization with the Instant Messenger. Users are provided with the ability to mark outgoing chat based on their current access. Additionally users have "dirty word" pattern recognition searches conducted against message traffic to ensure that inappropriate information is not sent to recipient(s). The SCE IM message traffic is encrypted at the transport level via Transport Layer Security (TLS) encryption or any other appropriate encryption.

[0050] The SCE Subscription Framework, which provides a dynamic mechanism to update, delete, or create security labels for users and locations for SCE IM, is subscribed to by SCE IM in the customer environment. The Subscription Framework provides real-time updates to user access to the SCE IM client and to the SCE IM server through subscriptions to interest and item services. Auditing and audit reduction for the secure Instant Messenger are also normally provided.

[0051] Next, at **70**, the SCE module for the ECM solution chosen by the customer is deployed into the test environment. The SCE of the present invention is agnostic to the 'brand' of customer document management tools, including those commonly used at many government agencies and corporations such as Open Text Livelink, EMC^2 Documentum, and Oracle Universal Content Management (UCM) system—as well as open source collaboration tools such as Drupal. An example screen shot of an EMC^2 Documentum implementation is shown in Figure Six. This figure demonstrates the Documentum Graphic User Interface with content marked with security attributes—in this case classifications and code words. This marking is enforced with Security Controls which are part of the SCE module for Documentum.

[0052] As shown in Figure Seven, user account information is provided through Oracle Access Manager (OAM) (or equivalent Identity Management tool) synchronization with Oracle UCM (the ECM tool chosen for this deployment example). Users are able to view, modify, and create content in Oracle UCM based on authentication from OAM and access control services through SCE. Oracle Access Manager provides user authentication credentials for Oracle Universal Content Management where as SCE provides a method of assigning security attributes and controlling content access in Oracle UCM based on those security attributes.

[0053] The SCE Subscription Framework is the mechanism in which the SCE Enabled Application server—in this example an Oracle Universal Content Management server—is provided security labels for users and locations as defined in the Security Rules Engine. By virtue of this process, table schemas are implemented for SCE Labels, users are able to mark content based on their current access (a union of user, location, and application access), users who do not have appropriate access are unable to view or modify content in which they do not have access, and data at rest is encrypted. Having users denied access as a result of having insufficient access enforces authorization based access control. Encrypting the data at rest provides a mechanism of protecting the data from exploitation from malicious users and provides compliance for customer security policy. The logon process does not change from a user perspective.

[0054] What does change is the user's experience with Oracle UCM. The user logs into the environment utilizing Oracle Access Manager as a method of providing user and password information to the Oracle UCM server. The SCE Module for Oracle UCM has been configured to allow or deny access to content stored on the Oracle UCM Server based on user and location security labels. Oracle UCM now has the ability to label content and search uploaded content for "dirty words" based on user and location access. It then advises the user on appropriate courses of action (re-label or not upload). The user information for the session is supplied by Oracle Access Manager and the location is taken from the requesting IP address of the user workstation.

[0055] Further, a marking capability is added through the SCE ECM module. Marking for classification banners is added based on the user's highest classification or program access allowed and based on available content marking on current folder, project, and/or cabinet.

[0056] Controls are also added for accessing ECM content. Controls include location sensitivity. These controls can allow, deny, or reduce access based on the organization security policy. Users can further modify associated ECM collaboration tools for SCE. The document control mechanism may be an SCE module which is a part of the content management solution. This SCE module provides an interface for the Subscription Framework to receive information as described previously from the Security Rules Engine. Additionally this module controls access to the content within the system based on the information provided by the Security Rules Engine. This control results in reducing, restricting or allowing access based on the logical union of user, location, and application security labels in comparison to the security label of the content. Additionally content being uploaded into the content management solution is scanned for dirty words and content transmission is restricted or denied based on dirty words existing in the content presented to the content management solution. Restriction of transmission of content to the content management solution includes giving the user the option to apply appropriate security labels to the content based on the logical union of user, location, and application authorized security labels. If the user does not have appropriate access as defined as the logical union of user, location, and application security labels, then the user will not be able to access content within the content management system that has security labels that are not part of the logical union of user, location and application security labels.

[0057] It should also be noted that the SCE Subscription Framework provides a dynamic mechanism to update, delete, or create security labels for users and locations as subscribed to by the SCE ECM module through subscriptions to interest and item services. SCE provides Oracle UCM with current user and facility information through the SCE Subscription Framework. The Subscription Framework provides real-time

updates to user access to the UCM server. Auditing and audit reduction is provided for the SCE module for UCM.

[0058] An SCE wiki module may also be deployed, **80**, to the customer test environment. This SCE wiki module may be ECM or a stand-alone wiki tool based. A wiki in this context is a page or collection of web pages designed to enable anyone who accesses it to contribute or modify content, using a simplified markup language. Wikis are often used to create collaborative websites. Some Enterprise Content Management systems incorporate wikis into their feature set. The SCE is designed to work with either these ECM wiki tools or stand alone tools available in the commercial marketplace.

[0059] Marking capability is added through the SCE wiki module. In one embodiment, ECM wiki information access and marking is paragraph based. Information is viewable based on user's highest classification/program access; each browse event requires the creation of banners on current wiki paragraph and page marking.

[0060] The SCE Subscription Framework, which provides a dynamic mechanism to update, delete, or create security labels for users and locations, is subscribed to by the SCE wiki module through subscriptions to interest and item services. SCE provides Oracle UCM with current user and facility information through the SCE Subscription Framework. The Subscription Framework provides real-time updates to user access to the UCM server. Auditing and audit reduction is provided for the SCE client for UCM.

[0061] In the case of a standalone SCE wiki tool, the wiki is synchronized with the customer environment through its own LDAP synchronization mechanism. User account information is provided through the synchronization of Oracle Access Manager with the wiki, or by any other suitable mechanism.

[0062] Next, at **90**, if desired, applicable SCE modules for desktop productivity tools are deployed to the customer test environment. By adhering to a common model for representing and sharing security information, SCE is agnostic to the 'brand' of customer desktop productivity tools, including those commonly used at many government agencies such as Microsoft Office, OpenOffice, and Adobe Acrobat.

[0063] As shown in Figure Eight, the SCE module is integrated into the customer's existing environment, in this case Microsoft Office 2007. This figure is an illustrative but non-limiting example of an SCE enabled Microsoft application user interface. In this case, a module has been developed that provides a mechanism for marking document content with security labels that are supplied by the Security Rules Engine through the Subscription Framework. Additionally, this module provides a mechanism for checking for "dirty words". This is treated primarily as a marking mechanism with the controlling capabilities happening at SCE enabled application servers.

[0064] The SCE Subscription Framework, which provides a dynamic mechanism to update, delete, or create security labels for users and locations, may be subscribed to by the users' workstation, the personal computer in which a user interacts with SCE enabled server applications, through subscriptions to interest and item services. SCE provides the workstation with available user security labels and dirty words, to be used by SCE enabled desktop applications based on the user and location access. This information is encrypted in the memory of the user's workstation upon the workstation subscribing to SCE information in the Security Rules Engine.

[0065] Further, information pertaining to individual user security labels per that location is encrypted in memory on the local machine and subsequently removed upon user logout. Each SCE module is 'tuned' to match local security policies for marking.

[0066] Next, at **100**, if desired, applicable SCE modules for email are deployed to the customer test environment. Again, by adhering to a common model for representing and sharing security information, SCE is agnostic to 'brand' of customer email tools, including those commonly used at many government agencies such as Microsoft Outlook and Thunderbird.

[0067] During implementation each SCE module discussed above is 'tuned' to match local security policies for marking. The final step is to verify the successful deployment of all SCE modules into the test environment and test the components.

[0068] At **110**, the SCE components are then installed into a pre-production/Quality Assurance (QA) environment designated by the customer. This is done through installing and integrating SCE modules for enabled applications, installing the Security Rules Engine, populating the Security Rules Engine with security labels, assigning dirty words for security labels, and assigning security labels to user, locations, and applications. The complete environment is tested based on customer system requirements and a security evaluation of the environment is performed to ensure compliance with customer security policy.

[0069] Finally, at **120**, the system is installed, tested, and deployed into the operational production environment.

[0070] A user session component of the present invention manages SCE application level user login and provides interfaces to determine the user's current session capabilities. This component manages SCE sign on authentication and provides user verification credentials and system privileges. The SCE user session module utilizes the existing authentication systems for user management and authentication. This existing system is a repository that maintains authorized system users. An interface between SCE and authentication system provides fundamental system access control for users attempting to access the SCE.

[0071] Once the user is authenticated, an SCE login session is established setting the interest limits available to the user for their session. The user session component monitors for changes to user privileges and provides software notification to other SCE components of required updates. A service interface allows SCE components to determine a user's current session interest and privileges.

[0072] The content management or Enterprise Content Management (ECM) portal displays are modified to limit what content is displayed. Content that is presented is labeled to indicate the security interest of the content. Users are prevented from accessing information they are not permitted to access. These limits are based on the granted security interest controlled by system administrators. In addition, users are prevented from determining if information exists for which access has not granted. Users accessing the portal will have access to only information falling within their configured access privileges.

[0073] As shown in Figure Nine, when a user requests to display repository contents, the SCE ECM service determines the range of information that a user is permitted to access and limits the display to documents within that interest range. Interoperating with the SCE security services, the range of information the user is permitted to access is determined and used to limit what documents and directories can

be viewed. The SCE ECM service enforces mandatory security labeling for documents stored in the repository.

[0074] Instant Messaging and Wikis are powerful tools for quickly and effectively managing and communicating information. SCE enhancements to Instant Messenger and Wikis allow users with differing interests to collaborate straightforwardly with assurances that data is distributed only to properly cleared users. The interest of each instant message is determined based on the message content when the message is sent and the message is marked with the appropriate classification. Messages are blocked for recipients without proper clearance to ensure that they do not receive information marked at that classification level. A blocked message is not delivered and a notification is provided to the sender indicating the recipient has insufficient clearance to receive a message with that marking.

[0075] SCE enhanced Wikis function much like other SCE enhanced tools. The interest of data being added to the page is determined automatically using dirty word search tools or can be manually set by the user editing the page. Wiki paragraph markings are displayed and accessible content is limited to the interest of the user's current session.

[0076] Microsoft Office, OpenOffice and Microsoft Outlook are primary user desktop productivity tools. Plug-in modules for the user applications provide built in capabilities to assist users with marking and labeling documents with proper security interest. These modules add menu options and tool bars directly to office applications including Microsoft Word, Excel, PowerPoint and Project as well as OpenOffice Writer, Calc, and Impress applications.

[0077] Document marking tools provide the capability to mark documents according to DoD 5200.1-PH, HIPAA, or user defined standards. Based on user interest selections, document headers and footers are automatically edited to include standard sensitivity markings. In addition, documents are portion marked. By default, the document is marked at the user's highest session interest; however menu options allow the user to manually select a desired classification. Users also have the option to use a dirty word search tool to automatically assign an interest or individually select sections of a document and manually change the interest for the current selection.

[0078] The markings applied by the user become the labeling used by the system to control content distribution. The marking tools make use of XML document file representation formats to provide maximum flexibility and compatibility with other software and file formats.

[0079] Users interact with Email systems such as Microsoft Outlook\Exchange using built in menu options much like working with a document. Messages are assigned a classification and are marked. When the user selects to send a message the recipient's clearance level is tested against the message classification. If the recipient is not cleared to receive the message the message is not delivered and a rejection message is sent to the sender.

[0080] Security administration tools provide an authorized administrator the capabilities to manage interests such as security levels and compartments for the Department of Defense. Tools are provided to create, update, and remove SCE security levels and compartments. These tools also provide the capability to select users, or groups of users, and specify security level and compartment assignments.

[0081] The Security Administration tool utilizes the Subscription Framework which propagates user, location and application interests through web based services that supported applications subscribe to when starting. When an enabled application starts it passes its start state to the Security Rules Engine which in turn passes user, location security metadata updates to interests for users, locations, and applications via web service to the supported application server or workstation.

[0082] A Nested Model-View-Controller (NMVC) design pattern is utilized for SCE functionality as shown in Figure Ten. Ultimately, an organization's security data repository is the model for the SCE. The following describes the steps taken:

[0083] 1. Control 1, Security Information Interface, requests data from Model 1 and then interprets the returned data to provide to View 1/Model 2, the SCE Security Product Package delivery service.

[0084] 2. View 1/Model 2 returns a security metadata product that is relevant for the SCE enabled application Control 2.

[0085] 3. Control 2 translates the security package into relevant information for that application that is displayed via View 2, the SCE application user interface.

[0086] 4. Control 1 and View 1/Model 2 represent the automation/interpretation of security policy for the SCE, whereas as Control 2 and View 2 represent the implementation of policy via an SCE enabled application.

[0087] This approach allows for the interpretation and implementation of policy to be separate in the SCE. By separating policy interpretation and implementation, application specific changes will not impact the collective behavior of the SCE. This allows aspects of the SCE to be swapped out and reconfigured based on customer needs. Additionally this allows minor policy changes to be made without significantly impacting the SCE functionality.

[0088] When a user account is formed for identification and authentication data store, such as LDAP, the Subscription Service synchronizes user account data to the Security Rules Engine much in the same way that supported applications synchronize with LDAP for single sign on. This allows security administrators to apply interests to users or locations that are defined in LDAP or apply interests to locations, and applications independent of LDAP. The subscription service applies the updated interests to enabled applications through encrypted web services to enabled applications in near real time. The cross functional flow chart depicted in Figure Eleven demonstrates the process of user account information being populated into a Directory by an IT administrator, the synchronization of user information to the Security Rules Engine, the Security Administrator applying interests to users, and the Subscription Framework populating SCE Enabled Applications with user interests. Furthermore, this demonstrates the separation of user role based access controls for applications (populated by the IT Administrator) and SCE authorization based access control (populated by the Security Administrator).

[0089] When a user requests to display contents of the enabled application, the enabled application authenticates under normal circumstances. After a valid authentication attempt the SCE ECM service determines the range of information that a user is permitted to access and limits the display to documents within that interest range. Interoperating with the SCE security services, the range of information the user is permitted to access is determined and used to limit what

documents and directories can be viewed. The cross functional flow chart depicted in Figure Twelve demonstrates role based authentication for the SCE application controlled from the Directory Server, and authorization based access being controlled by the SCE Enabled Application.

[0090] It is therefore intended that the foregoing detailed description be regarded as illustrative rather than limiting, and that it be understood that it is the following claims, including all equivalents, that are intended to define the spirit and scope of this invention.

APPENDIX A SECURITY SERVICES AND SUBSCRIPTION FRAMEWORK SOFTWARE LISTING

A.1 SubscribableService Definition and Implementation

[0091] The SubscribableService specifies a service to be implemented by SRE services.

```
public interface SubscribableService {
    public void addSubscription(Subscription subscription);
    public void removeSubscription(Subscription subscription);
    public void updateSubscribers(Collection<String>
updatedProductIds);
}
public abstract class SubscribableServiceImpl implements
SubscribableService {
    // Maps a subscriberServiceUrls to a collection of productIds
    private Map<String, Collection<String>> subscriptions = new
HashMap<String, Collection<String>>( );
    public void addSubscription(Subscription subscription) {
        if (this.subscriptions.get(subscription.getServiceUrl( )) ==
null) {
            this.subscriptions.put(subscription.getServiceUrl( ),
                    new HashSet<String>( ));
        }
        this.subscriptions.get(subscription.getServiceUrl( )).addAll(
                subscription.getProductIds( ));
        new Thread(new SubscriberUpdater(subscription)).start( );
    }
    public void removeSubscription(Subscription subscription) {
        this.subscriptions.remove(subscription.getServiceUrl( ));
    }
    public void updateSubscribers(Collection<String>
updatedProductIds) {
        System.out.println("About to update subscribers. This should
be quick.");
        for (String serviceUrl : this.subscriptions.keySet( )) {
            Collection<String> unionProductIds = new
HashSet<String>( );
            unionProductIds.addAll(updatedProductIds);
    unionProductIds.retainAll(this.subscriptions.get(serviceUrl));
            try {
                Subscription subscription = new Subscription( );
                subscription.setServiceUrl(serviceUrl);
                subscription.setProductIds(updatedProductIds);
                new Thread(new
SubscriberUpdater(subscription)).start( );
            } catch (Throwable t) {
                System.err.println("Error updating subscriber " +
serviceUrl
                        + " with ids " + updatedProductIds);
            }
        }
        System.out.println("Done updating subscribers.");
    }
}
```

A.2 InterestService Definition and Implementation

[0092] InterestService is an implementation of a SubscribableService, and is a service provided by SRE services to provide Interests to subscribers. Interests in this case can be classifications, compartments, programs, etc.

```
public interface InterestService extends SubscribableService {
    public void setInterest(Interest interest);
    public Interest getInterest(String identifier);
    public void setInterests(Collection<Interest> interests);
    public Collection<Interest> getInterests(Collection<String> ids);
    public Collection<Interest> getInterests( );
    public Collection<Interest> getInterests(String type);
}
public class InterestServiceImpl extends SubscribableServiceImpl
implements
        InterestService {
    private InterestDao interestDao;
    @Override
    public void setInterest(Interest interest) {
        this.interestDao.saveInterest(interest);
        Collection<String> updatedProductIds = new
HashSet<String>( );
        updatedProductIds.add(interest.getId( ));
        this.updateSubscribers(updatedProductIds);
    }
    @Override
    public Interest getInterest(String id) {
        return this.interestDao.getInterest(id);
    }
    @Override
    public void setInterests(Collection<Interest> interests) {
        this.interestDao.saveInterests(interests);
        Collection<String> updatedProductIds = new
HashSet<String>( );
        for (Interest updatedInterest : interests) {
            updatedProductIds.add(updatedInterest.getId( ));
        }
        this.updateSubscribers(updatedProductIds);
    }
    @Override
    public Collection<Interest> getInterests(Collection<String> ids) {
        return this.interestDao.getInterests(ids);
    }
    @Override
    public Collection<Interest> getInterests( ) {
        return this.interestDao.getInterests( );
    }
    @Override
    public Collection<Interest> getInterests(String type) {
        return this.interestDao.getInterests(type);
    }
    public InterestDao getInterestDao( ) {
        return interestDao;
    }
    public void setInterestDao(InterestDao interestDao) {
        this.interestDao = interestDao;
    }
}
```

A.3 Subscription, Subscriber, and SubscriberUpdater

[0093] A Subscriber subscribes via a Subscription to a SubscribableService. The Subscriber's Subscription is updated by a SubscriberUpdater.

```
public class Subscription implements Serializable {
    private static final long serialVersionUID = 1L;
    private String serviceUrl;
    private Collection<String> productIds;
    public String getServiceUrl( ) {
        return serviceUrl;
    }
    public void setServiceUrl(String serviceUrl) {
```

-continued

```
        this.serviceUrl = serviceUrl;
    }
    public void setProductIds(Collection<String> productIds) {
        this.productIds = productIds;
    }
    public Collection<String> getProductIds( ) {
        return this.productIds;
    }
}
public interface Subscriber {
    public void update(Collection<String> updatedProductIds);
}
public class SubscriberUpdater implements Runnable {
    private Subscription subscription;
    public SubscriberUpdater(Subscription subscription) {
        this.subscription = subscription;
    }
    private boolean updateSubscriber( ) {
        try {
            HttpInvokerProxyFactoryBean
httpInvokerProxyFactoryBean = new HttpInvokerProxyFactoryBean( );
            httpInvokerProxyFactoryBean.setServiceInterface(Subscriber.class);
            httpInvokerProxyFactoryBean.setServiceUrl(this.subscription
                    .getServiceUrl( ));
            httpInvokerProxyFactoryBean.afterPropertiesSet( );
            Subscriber subscriber = (Subscriber)
httpInvokerProxyFactoryBean
                    .getObject( );
            subscriber.update(this.subscription.getProductIds( ));
            return true;
        } catch (Throwable t) {
            System.err.println("Error updating subscriber " +
this.subscription.getServiceUrl( ));
            return false;
        }
    }
    @Override
    public void run( ) {
        boolean success = false;
        int attempts = 10;
        while (success == false && attempts > 0) {
            success = this.updateSubscriber( );
            attempts--;
            try {
                Thread.sleep(5 * 1000);
            } catch (Exception e) {
                System.err.println(e);
            }
        }
    }
}
```

A.4 Security Rules Engine Spring Configuration

[0094] This is where SRE services are tied together and published.

```
<beans>
    <bean name="testSubscriber"
class="com.anakam.sce.securityservices.TestSubscriber" />
    <bean name="/TestSubscriber"
    class="org.springframework.remoting.httpinvoker.HttpInvokerService
Exporter">
        <property name="service">
            <ref bean="testSubscriber" />
        </property>
        <property name="serviceInterface">
    <value>com.anakam.sce.securityservices.Subscriber</value>
        </property>
    </bean>
    <bean name="persister"
```

-continued

```
class="com.anakam.sce.persist.HibernateUtil"
        init-method="populateTestData" />
    <bean id="itemDao" class="com.anakam.sce.persist.ItemDao">
        <property name="persister" ref="persister" />
    </bean>
    <bean id="interestDao"
class="com.anakam.sce.persist.InterestDao">
        <property name="persister" ref="persister" />
    </bean>
    <bean id="itemService"
class="com.anakam.sce.securityservices.ItemServiceImpl">
        <property name="itemDao" ref="itemDao" />
    </bean>
    <bean id="interestService"
class="com.anakam.sce.securityservices.InterestServiceImpl">
        <property name="interestDao" ref="interestDao" />
    </bean>
    <bean name="/ItemService"
    class="org.springframework.remoting.httpinvoker.-
HttpInvokerServiceExporter">
        <property name="service">
            <ref bean="itemService" />
        </property>
        <property name="serviceInterface">
    <value>com.anakam.sce.securityservices.ItemService</value>
        </property>
    </bean>
    <bean name="/InterestService"
    class="org.springframework.remoting.httpinvoker.-
HttpInvokerServiceExporter">
        <property name="service">
            <ref bean="interestService" />
        </property>
        <property name="serviceInterface">
    <value>com.anakam.sce.securityservices.InterestService</value>
        </property>
    </bean>
</beans>
```

APPENDIX B INSTANT MESSENGER SOFTWARE LISTING

B.1 AccessProvider Definition and SSAccessProvider Implementation

[0095] An AccessProvider specifies a means of retrieving an Access. An Access in this case is the type of secure information a given individual is cleared to see. An Access includes collateral classifications, compartments, programs, etc. SSAccessProvider is the SRE services implementation of an AccessProvider.

```
public interface AccessProvider {
    public Access getAccess(String username) throws
AccessUnavailableException;
}
public class SSAccessProvider implements AccessProvider {
    private Document getXML(String urlString) throws Exception {
        java.net.URL url = new java.net.URL(urlString);
        HttpURLConnection m_con = (HttpURLConnection)
url.openConnection( );
        m_con.setDoInput(true);
        SAXReader saxReader = new SAXReader( );
        Document document =
saxReader.read(m_con.getInputStream( ));
        return document;
    }
    private Document getXML(String urlString, String name) throws
Exception {
        return this.getXML(urlString + "?name=" + name);
```

-continued

```
    }
    public Access getAccess(String username) throws
AccessUnavailableException {
        SafeJiveLog.info("About to retrieve Access for " + username);
        HttpInvokerProxyFactoryBean httpInvokerProxyFactoryBean
= new HttpInvokerProxyFactoryBean( );
    httpInvokerProxyFactoryBean.setServiceInterface(ItemService.class);
        httpInvokerProxyFactoryBean
    .setServiceUrl("http://localhost:46220/sre/ItemService");
        httpInvokerProxyFactoryBean.afterPropertiesSet( );
        ItemService itemService = (ItemService)
httpInvokerProxyFactoryBean
                .getObject( );
        Item user = itemService.getItem(username);
        Access access = new Access( );
        access.setUsername(username);
        List<Classification> classifications = new
ArrayList<Classification>( );
        for (Interest interest : user.getRegard( )) {
            if (interest.getType( ).equals("Classification")) {
                Classification classification = new
ClassificationImpl(null,
                        null, null, null);
                classification.setName(interest.getName( ));
                classification.setCode(interest.getShortName( ));
                classification.setUniqueID(interest.getId( ));
                classification.setPrograms(new
ArrayList<Program>( ));
                classifications.add(classification);
            }
        }
        for (Interest interest : user.getRegard( )) {
            if (interest.getType( ).equals("Compartment")) {
                Program program = new ProgramImpl(null, null,
null, null, null);
                program.setUniqueID(interest.getId( ));
                program.setName(interest.getName( ));
                program.setCode(interest.getShortName( ));
                program.setDirtyWords(new
ArrayList<String>( ));
                program.setHandlingCaveats(new
ArrayList<HandlingCaveat>( ));
                for (String dirtyWord : interest.getTriggers( )) {
                    program.getDirtyWords( ).add(dirtyWord);
                }
                for (Classification classification : classifications)
{
    classification.getPrograms( ).add(program);
                }
            }
        }
        access.setClassifications(classifications);
        SafeJiveLog.info("Done retrieving Access for " + username);
        return access;
    }
}
```

B.2 MessageClassificationFilter

[0096] A MessageClassificationFilter performs the marking of instant message packets with classification information based on the current selection by the user. The current selection by the user depends on which checkboxes he has selected, indicating the collateral classification level, the applicable compartments, programs, etc.

```
public class MessageClassificationFilter implements MessageFilter {
    private ChatRoom chatRoom;
    private ImprovedClassificationPanel classificationPanel;
    public MessageClassificationFilter(ChatRoom chatRoom,
```

-continued

```
            ImprovedClassificationPanel classificationPanel) {
        this.setChatRoom(chatRoom);
        this.setClassificationPanel(classificationPanel);
    }
    public void filterOutgoing(ChatRoom chatRoom, Message
message) {
        if (chatRoom == this.getChatRoom( )) {
            Classification selectedClassification = this
    .getClassificationPanel( ).getSelectedClassification( );
            StringBuffer newMessageSB = new StringBuffer( );
            newMessageSB.append("(");
    newMessageSB.append(selectedClassification.getCode( ));
            for (Program program :
selectedClassification.getPrograms( )) {
                newMessageSB.append("/");
                newMessageSB.append(program.getCode( ));
            }
            newMessageSB.append(") ");
            newMessageSB.append(message.getBody( ));
            message.setBody(newMessageSB.toString( ));
            message.addExtension(new
PacketClassificationExtension(
                    selectedClassification));
        }
    }
    public void filterIncoming(ChatRoom chatRoom, Message
message) {
        PacketExtension packetExtension = message.getExtension(
                PacketClassificationExtension.getElemName( ),
    PacketClassificationExtension.getElemNamespace( ));
        if (packetExtension != null) {
            PacketClassificationExtension
messageClassificationExtension = (PacketClassificationExtension)
packetExtension;
            Classification selectedClassification =
messageClassificationExtension
                    .getClassification( );
            this.getClassificationPanel( ).setSelectedClassification(
                    selectedClassification);
        }
    }
    public ChatRoom getChatRoom( ) {
        return chatRoom;
    }
    public void setChatRoom(ChatRoom chatRoom) {
        this.chatRoom = chatRoom;
    }
    public ImprovedClassificationPanel getClassificationPanel( ) {
        return classificationPanel;
    }
    public void setClassificationPanel(
            ImprovedClassificationPanel classificationPanel) {
        this.classificationPanel = classificationPanel;
    }
                    }
```

We claim:

1. A secure collaborative environment for an organization comprising a security rules engine in which access to secure information is granted or denied based on the application of an organization's security policy.

2. The environment of claim **1** wherein the security policy includes authorization criteria for users, physical and logical access locations, and computing applications, as well as descriptive security details of the target information.

3. A method for selectively granting access to data in an application based on the comparison of item interests.

4. The method of claim **3** wherein comparison of item interests is controlled through the application of rules to allow, restrict or deny interaction between item interests.

5. The method of claim **3** where item interaction is controlled by the application of interests to items.

6. The method of claim **3** wherein information that defines a logical limitation of need to share is applied to an item interest.

7. The method of claim **3** wherein interests are applied to items.

8. The method of claim **4** wherein interests are applied to items.

9. The method of claim **3** where the logical union of a user's interest is less that the interest of the requested data or communication transaction with another user.

10. The method of claim **3** wherein the application is a content management system.

11. The method of claim **10** that provides for reduced or restricted searching, browsing, and storing of data.

12. The method of claim **3** wherein items with interest for uses may be marked.

13. The method of claim **3** further comprising dirty word searches for submitted content items by interest.

14. The method of claim **3** further comprising streaming context based search of submitted content items.

15. The method of claim **3** wherein the application is a messaging system.

16. The method of claim **15** that allows for marking messages with interests for users.

17. The method of claim **15** that allows for dirty word searches for submitted messages between items by interest.

18. The method of claim **15** that allows for streaming context based search of submitted messages between items.

19. The method of claim **15** further comprising streaming context based search of messages.

20. The method of claim **3** wherein the application is a collaboration tool.

21. The method of claim **20** further comprising marking of session with interests for users.

22. The method of claim **20** further comprising dirty word searches for sessions between items of interest.

23. The method of claim **20** further comprising streaming context based search of content between items.

24. The method of claim **3** wherein the interests are applied to items where the items are produced through desktop productivity software.

25. The method of claim **24** wherein the items are Adobe. pdf content.

26. The method of claim **24** wherein the items are Adobe. pdf forms.

27. The method of claim **24** wherein the items are OpenOffice content.

28. The method of claim **24** wherein the items are Microsoft Office content.

29. The method of claim **24** wherein the items are messages between items.

30. The method of claim **24** wherein dirty word searches are applied to items.

* * * * *