United States Patent	12388808
Kind Code	B2
Date of Patent	August 12, 2025
Inventor(s)	Gupta; Virender et al.

Security and governance policies in electronic signature systems

Abstract

Policy-informed e-signature processing. An electronic signature system (ESS) is interfaced to a content management system (CMS) that stores instances of shared files and coordinates policy-informed interactions with the shared files. Upon detection of an occurrence of an e-signature request event, electronic signature processing is carried out in a manner that observes one or more governance policies that control handling of shared files. When responding to an e-signature request event, the CMS observes the governance policies by issuing an electronic query to a database of parameters, wherein the parameters define metes and bounds of the governance policies. Upon determination that at least some of the governance policies at least potentially apply to the e-signature processing, the ESS and/or the CMS modifies metadata pertaining to one or more workflow objects that are associated with a computer-implemented workflow. The workflow may terminate or be terminated based upon the one or more governance policies.

Inventors: **Gupta; Virender (Morganville, NJ), Bakshi; Rohit (Campbell, CA), Zhao; Yi (Redwood City, CA), Shay; Shawn (Castro Valley, CA), Carlson; William Ernest (Austin, TX)**

Applicant: **Box, Inc.** (Redwood City, CA)

Family ID: **1000008749358**

Assignee: **Box, Inc. (Redwood City, CA)**

Appl. No.: **17/958154**

Filed: **September 30, 2022**

Prior Publication Data

Document Identifier	Publication Date
US 20230025808 A1	Jan. 26, 2023

Related U.S. Application Data

continuation-in-part parent-doc US 16948828 20201001 US 11616782 child-doc US 17958154
continuation-in-part parent-doc US 16553057 20190827 PENDING child-doc US 16948828
us-provisional-application US 63262133 20211005
us-provisional-application US 62723314 20180827
us-provisional-application US 62723435 20180827

Publication Classification

Int. Cl.: **H04L9/40** (20220101)

U.S. Cl.:

CPC **H04L63/08** (20130101); **H04L63/20** (20130101);

Field of Classification Search

CPC: H04L (63/205); H04L (63/10); H04L (63/101); H04L (63/105); H04L (63/20)

References Cited

U.S. PATENT DOCUMENTS

Patent No.	Issued Date	Patentee Name	U.S. Cl.	CPC
10552590	12/2019	Hamlin et al.	N/A	N/A
10726152	12/2019	Durham et al.	N/A	N/A
10902072	12/2020	Anders et al.	N/A	N/A
10943030	12/2020	Guymon, Jr.	N/A	G06F 21/64
10979432	12/2020	Frank et al.	N/A	N/A
11689517	12/2022	Styliadis	726/7	G06F 21/30
2003/0018890	12/2002	Hale et al.	N/A	N/A
2009/0307746	12/2008	Di et al.	N/A	N/A
2014/0208425	12/2013	Palomaki	N/A	N/A
2015/0089575	12/2014	Vepa et al.	N/A	N/A
2015/0227756	12/2014	Barbas	N/A	N/A
2016/0048696	12/2015	Follis	726/28	G06F 21/645
2016/0070758	12/2015	Thomson et al.	N/A	N/A
2016/0117495	12/2015	Li et al.	N/A	N/A
2017/0083867	12/2016	Saxena	N/A	G06Q 10/103
2017/0223093	12/2016	Peterson	N/A	G06F 21/62
2018/0114015	12/2017	Nuseibeh et al.	N/A	N/A
2018/0124609	12/2017	Ciano et al.	N/A	N/A
2019/0220550	12/2018	Arasachetty	N/A	G06F 16/907
2020/0092337	12/2019	Ojha et al.	N/A	N/A

OTHER PUBLICATIONS

Non-Final Office Action for U.S. Appl. No. 16/948,828 dated Jul. 22, 2022. cited by applicant
J. Alqatawna, E. Rissanen and B. Sadighi, “Overriding of Access Control in XACML,” Eighth IEEE International Workshop on Policies for Distributed Systems and Networks (Policy'07), 2007, pp. 87-95 (Year: 2007). cited by applicant
International Search Report and Written Opinion dated Dec. 11, 2019 for PCT Appln. No.

PCT/US19/48435. cited by applicant
Non-Final Office Action Dated Feb. 17, 2022 U.S. Appl. No. 16/553,057. cited by applicant
Non-Final Office Action for U.S. Appl. No. 16/553,063 dated Mar. 3, 2022. cited by applicant
Final Office Action dated Oct. 7, 2021 U.S. Appl. No. 16/553,063. cited by applicant
Non-Final Office Action dated Jun. 8, 2021 U.S. Appl. No. 16/553,063. cited by applicant
European Search Report dated Sep. 20, 2021 for related EP Applicatio No. 19853611.2. cited by applicant
Notice of Allowance dated Jun. 24, 2022 for U.S. Appl. No. 16/553,063. cited by applicant
Final Office Action dated Jul. 21, 2022; U.S. Appl. No. 16/553,057. cited by applicant
Chen, Y., et al., "What's New About Cloud Computing Security?," Electrical Engineering and Computer Sciences, University of California at Berkeley, dated Jan. 20, 2010. cited by applicant
Filippi, D., et al., "Cloud Computing: Centralization and Data Sovereignty," European Journal of Law and Technology, vol. 3, No. 2, 2012. cited by applicant
Notice of Allowance dated Apr. 30, 2024 for related U.S. Appl. No. 16/553,057. cited by applicant
Notice of Allowance dated Aug. 14, 2024 for related U.S. Appl. No. 16/553,057. cited by applicant
Notice of Allowance for U.S. Appl. No. 16/948,828 dated Nov. 16, 2022. cited by applicant
Notice of Allowance for U.S. Appl. No. 16/553,063 dated Feb. 1, 2023. cited by applicant
Notice of Allowance for U.S. Appl. No. 16/553,063 dated Sep. 21, 2022. cited by applicant
Notice of Allowance dated Dec. 20, 2024 for related U.S. Appl. No. 16/553,057. cited by applicant

---

Primary Examiner: Vaughan; Michael R

Attorney, Agent or Firm: Vista IP Law Group, LLP

Background/Summary

RELATED APPLICATIONS (1) The present application is a continuation in part of, and claims the benefit of priority to co-pending U.S. patent application Ser. No. 16/948,828 titled "CONTEXT-AWARE CONTENT OBJECT SECURITY", filed on Oct. 1, 2020, which claims the benefit of priority to co-pending U.S. patent application Ser. No. 16/553,057 titled "CONTEXT-AWARE CONTENT OBJECT SECURITY", filed on Aug. 27, 2019, which claims the benefit of priority to U.S. Provisional Patent Application Ser. No. 62/723,314 titled "COLLABORATION SYSTEM SECURITY", filed on Aug. 27, 2018; and which claims the benefit of priority to U.S. Provisional Patent Application Ser. No. 62/723,435 titled "COLLABORATION SYSTEMS", filed on Aug. 27, 2018; and this application claims the benefit of priority to U.S. Provisional Patent Application Ser. No. 63/262,133 titled "COLLABORATION SYSTEM SECURITY", filed on Oct. 5, 2021, all of which are hereby incorporated by reference in their entirety.

FIELD
(1) This disclosure relates to content management systems, and more particularly to techniques for enforcing policies that govern the handling of electronic signatures over content objects of a content management system.
BACKGROUND
(2) Cloud-based content management services and systems have impacted the way personal and enterprise computer-readable content objects (e.g., documents, spreadsheets, images, programming code files, etc.) are stored, and has also impacted the way such personal and enterprise content objects are shared and managed. Content management systems provide the ability to securely share large volumes of content objects among trusted users (e.g., collaborators) on a variety of user devices such as mobile phones, tablets, laptop computers, desktop computers, and/or other devices.

(3) In some systems, sharing and collaboration activities include one or more interactions (e.g., authoring, editing, viewing, etc.) that are performed over the content objects. The interactions may be performed by one user, or by multiple users, and/or even autonomously by one or more computing entities (e.g., processes, agents, applications, etc.). Moreover, such interactions may span across multiple departments and/or multiple enterprises.
(4) The interactions by users over content objects in such content management systems are governed in accordance with certain resource access permissions associated with the various combinations of users and content objects. In some settings, resource access permissions may correspond to not only users and content objects, but also may correspond to connections, services, URLs, and/or other resources associated with the content management system.
(5) Resource access permissions for specific resources often comply with a global permissions model that is implemented and managed at the content management system. A global permissions model defines a single set of semantics and workflows to efficiently manage (e.g., assign, lookup, etc.) resources permissions over the many resources of a content management system. As one example, a global permissions model might predefine a set of user roles (e.g., owner, co-owner, editor, viewer, previewer, uploader, etc.), a set of permitted content object interactions (e.g., edit, download, preview, view/add comments, view metadata, etc.), and a set of mappings between the aforementioned user roles and the permitted content object interactions. In this case, a user can be assigned a particular user role for a particular content object so as to restrict the user to performing only those permitted interactions associated with that user's role over that content object.
(6) Further, in some other settings, content management systems interface with, or otherwise provide certain capabilities that facilitate use of electronic signatures over electronic documents. In particular, once advised of the existence of a document to be signed, such application programming interfaces can be called to (1) assemble the document to be signed together with any one or more further electronic documents to be reviewed and/or signed (e.g., PDFs, notarial forms, etc.), (2) designate the signatories, and (3) send a request out to the signatories for their individual e-signatures.
(7) Unfortunately, in today's dynamic collaboration environments, the aforementioned global permissions model is deficient. Specifically, the static nature of the global permissions model is deficient in addressing the many dynamic contexts of the interactions with the content objects performed at a content management system. Such interaction contexts are defined not only by a user's role as pertains to a content object, but also by the enterprise of the user, the application being used for the interaction, the security level applied to the content object, and/or by other policies and/or rules of the content management system. What is needed is a way to overcome the foregoing deficiencies.

SUMMARY

(8) The present disclosure describes techniques used in systems, methods, and in computer program products for enforcing security and governance policies in electronic signature systems, which techniques advance the relevant technologies to address technological issues with legacy approaches. More specifically, the present disclosure describes techniques used in systems, methods, and in computer program products for context-aware extensible content object access permissions. Certain embodiments are directed to technologies for analyzing the attributes of interactions with content objects to dynamically determine extensible access permissions that override a global permissions model.
(9) The disclosed embodiments modify and improve over legacy approaches. In particular, the herein-disclosed techniques provide technical solutions that address the technical problems attendant to the limited applicability of global permissions models in collaboration systems. Such technical solutions involve specific implementations (i.e., data organization, data communication paths, module-to-module interrelationships, etc.) that relate to the software arts for improving computer functionality.

(10) The ordered combination of steps of the embodiments serve in the context of practical applications that perform steps to analyze various attributes of user interactions with content objects to dynamically determine extensible access permissions. The disclosed techniques for analyzing the attributes of interactions with content objects to dynamically determine extensible access permissions serve to overcome long standing yet heretofore unsolved technological problems that arise in the realm of computerized collaboration systems. Aspects of the present disclosure achieve performance and other improvements in peripheral technical fields including (but not limited to) computer security and distributed storage systems.

(11) Some embodiments perform a series of steps for establishing a global permissions model that applies a first set of resource access permissions to a shared content object; establishing one or more context-aware access policies that govern user interactions over the shared content object; gathering one or more interaction attributes associated with a particular user interaction over the shared content object; applying the context-aware access policies to the one or more interaction attributes to determine a set of extensible access permissions; and overriding the first set of resource access permissions with the set of extensible access permissions.

(12) Some embodiments perform further steps including generating a challenge that is issued to an authority to justify or otherwise allow a particular user interaction that would be disallowed in absence of a user-justified override. The determination of when to carry out a challenge—and the determination of when to allow the particular user interaction to proceed—can be based on the set of permissions that are in place at the time of the particular user interaction request. In some cases, the response corresponds to at least one of: allowing an interaction, allowing the interaction based on the validity of a particular justification, blocking the interaction, or taking no overriding action. In some cases, at least a portion of the challenge is presented to an authorizing user via a user interface.

(13) Some embodiments process context-aware access policies wherein the context-aware access policies and/or extensible permissions rules are considered when responding to an e-signature request event.

(14) In various embodiments, a context associated with the particular user interaction is characterized by one or more of the one or more interaction attributes, which interaction attributes comprise one or more user-justifications, one or more event attributes, and/or one or more object attributes, and/or one or more user attributes. User-justifications for override can be specified by a user at a user interface. Specifically, some embodiments include a user interface for selecting or otherwise inputting a justification for a particular requested action. In some embodiments, the acceptability of a justification is determined based on sentiment analysis and/or natural language processing techniques. In some embodiments, sentiment analysis and/or natural language processing is provided by one or more external data processing modules.

(15) Some embodiments include a sequence of instructions that are stored on a non-transitory computer readable medium. Such a sequence of instructions, when stored in memory and executed by one or more processors, causes the one or more processors to perform a set of acts for enforcing security and governance policies in electronic signature systems.

(16) Some embodiments include the aforementioned sequence of instructions that are stored in a memory, which memory is interfaced to one or more processors such that the one or more processors can execute the sequence of instructions to cause the one or more processors to enforce security and governance policies in electronic signature systems.

(17) In various embodiments, any combinations of any of the above can be combined to perform any variations of acts for implementing security and governance policies in electronic signature systems, and many such combinations of aspects of the above elements are contemplated.

(18) Further details of aspects, objectives, and advantages of the technological embodiments are described herein, and in the drawings and claims.

Description

BRIEF DESCRIPTION OF THE DRAWINGS

(1) The drawings described below are for illustration purposes only. The drawings are not intended to limit the scope of the present disclosure.

(2) FIG. **1**A exemplifies a computing environment in which embodiments of the present disclosure can be implemented.

(3) FIG. **1**B**1** shows a security and governance engine within a content management system, according to an embodiment.

(4) FIG. **1**B**2** illustrates how a set of context-aware policies can inform any/all steps of an e-signature process, according to an embodiment.

(5) FIG. **1**B**3** shows examples of rules that may be introduced into the system, according to an embodiment.

(6) FIG. **1**B**4** shows an example envelope data structure, according to an embodiment.

(7) FIG. **2** exemplifies a context-aware extensible access permissions assignment technique as implemented in systems that enforce context-aware extensible content object access permissions, according to an embodiment.

(8) FIG. **3**A presents a block diagram of a system that implements and enforces context-aware extensible content object access permissions, according to some embodiments.

(9) FIG. **3**B presents a block diagram of a system that reconciles context-aware extensible content object access permissions, according to some embodiments.

(10) FIG. **4**A depicts a context-aware access policy specification technique as implemented in systems that facilitate use of context-aware extensible content object access permissions, according to an embodiment.

(11) FIG. **4**B depicts a shared link restriction specification technique as implemented in systems that facilitate use of context-aware extensible content object access permissions, according to an embodiment.

(12) FIG. **4**C depicts an external collaboration restriction specification technique as implemented in systems that facilitate use of context-aware extensible content object access permissions, according to an embodiment.

(13) FIG. **5** depicts an interaction event context identification technique as implemented in systems that facilitate use of context-aware extensible content object access permissions, according to an embodiment.

(14) FIG. **6** depicts a rule-based extensible access permissions override technique as implemented in systems that facilitate use of context-aware extensible content object access permissions, according to an embodiment.

(15) FIG. **7**A depicts an interaction event response technique as implemented in systems that facilitate use of context-aware extensible content object access permissions, according to an embodiment.

(16) FIG. **7**B and FIG. **7**C depict aspects of a processing flow to reach an allow determination or a deny determination for a particular collaboration activity request, according to an embodiment.

(17) FIG. **7**D depicts aspects of a processing flow for propagating results of an allow or deny determination for a particular collaboration activity request, according to an embodiment.

(18) FIG. **7**E depicts a justification user interface, according to an embodiment.

(19) FIG. **8**A, FIG. **8**B, FIG. **8**C, and FIG. **8**D depict system components as arrangements of computing modules that are interconnected so as to implement certain of the herein-disclosed embodiments.

(20) FIG. **9**A and FIG. **9**B present block diagrams of computer system architectures having components suitable for implementing embodiments of the present disclosure, and/or for use in the

herein-described environments.

DETAILED DESCRIPTION

(21) Aspects of the present disclosure solve problems associated with collaboration systems that suffer from the limited applicability of global permissions models. These problems are unique to, and may have been created by, various computer-implemented permissions models used in content management systems. Some embodiments are directed to approaches for analyzing the attributes of interactions with content objects to dynamically determine extensible access permissions that override other permissions models. The accompanying figures and discussions herein present example environments, systems, methods, and computer program products for context-aware extensible content object access permissions.

(22) Overview

(23) Disclosed herein are techniques for analyzing the attributes of content object interactions so as to dynamically determine sets of extensible access permissions that govern user interactions over shared content objects. In exemplary embodiments, such techniques are implemented in a content management system (CMS) that includes or is interfaced with an electronic signature system (ESS). Deployment of the combination of a CMS that is interfaced with an ESS facilitates many user-friendly workflows. Certain of these workflows are configured to address the fact that certain users of the ESS (e.g., signatories, administrators, etc.) might not be users of the CMS.

(24) Leakage Prevention

(25) The mere possibility that some users of the ESS might not be users of the CMS introduces risk of leakage of sensitive information. This possibility arises when an ESS user does not have a CMS user profile. In absence of a CMS user profile, the CMS is unable to vet the intended recipient with respect to the ESS user's and/or the ESS user's computing platform's trustworthiness. To address this vulnerability, context-aware access policies are established at the content management system. These context-aware policies govern the interactions of the users over the content objects. More specifically, the foregoing context-aware policies are driven, at least in part, by rules that protect against unintended dissemination of CMS documents merely because a particular subject document is a subject of an ESS transaction.

(26) Context-aware policies are enforced such that, when interaction events associated with the user and content objects are detected (e.g., a request for the user to participate in an e-signing protocol, a request to upload a content object, a request to download a content object, etc.), certain event attributes corresponding to the events are identified. The event attributes, and other attributes derived from the event attributes, constitute the interaction attributes that characterize the respective contexts of the interaction events. The interaction attributes are subjected to the context-aware access policies to determine sets of extensible access permissions that govern further interaction events. Specifically, responses to the interaction events are issued in accordance with the sets of extensible access permissions. As an example, for a particular interaction event (e.g., a sign request, a download request, etc.), a response might allow an action (e.g., “OK to preview”, “OK to download”, etc.) associated with the event, or the action might be blocked. In certain situations, the system might allow the action only if accompanied with some administrative intercession (e.g., an access grant or other justified exception).

(27) In some embodiments, classification labels are assigned to content objects to facilitate the determination of sets of extensible access permissions. In certain embodiments, the context-aware access policies comprise one or more extensible permissions rules that facilitate the determination of the sets of extensible access permissions. In certain embodiments, a conflict remediation capability is implemented in the content management system to resolve conflicts between different permissions models and/or conflicts between multiple access policies. In certain embodiments, invocation and/or completion of certain operations involved in the electronic signing protocol are governed by the extensible access permissions.

Definitions and Use of Figures

(28) Some of the terms used in this description are defined below for easy reference. The presented terms and their respective definitions are not rigidly restricted to these definitions—a term may be further defined by the term's use within this disclosure. The term “exemplary” is used herein to mean serving as an example, instance, or illustration. Any aspect or design described herein as “exemplary” is not necessarily to be construed as preferred or advantageous over other aspects or designs. Rather, use of the word exemplary is intended to present concepts in a concrete fashion. As used in this application and the appended claims, the term “or” is intended to mean an inclusive “or” rather than an exclusive “or”. That is, unless specified otherwise, or is clear from the context, “X employs A or B” is intended to mean any of the natural inclusive permutations. That is, if X employs A, X employs B, or X employs both A and B, then “X employs A or B” is satisfied under any of the foregoing instances. As used herein, at least one of A or B means at least one of A, or at least one of B, or at least one of both A and B. In other words, this phrase is disjunctive. The articles “a” and “an” as used in this application and the appended claims should generally be construed to mean “one or more” unless specified otherwise or is clear from the context to be directed to a singular form.

(29) Various embodiments are described herein with reference to the figures. It should be noted that the figures are not necessarily drawn to scale, and that elements of similar structures or functions are sometimes represented by like reference characters throughout the figures. It should also be noted that the figures are only intended to facilitate the description of the disclosed embodiments—they are not representative of an exhaustive treatment of all possible embodiments, and they are not intended to impute any limitation as to the scope of the claims. In addition, an illustrated embodiment need not portray all aspects or advantages of usage in any particular environment.

(30) An aspect or an advantage described in conjunction with a particular embodiment is not necessarily limited to that embodiment and can be practiced in any other embodiments even if not so illustrated. References throughout this specification to “some embodiments” or “other embodiments” refer to a particular feature, structure, material or characteristic described in connection with the embodiments as being included in at least one embodiment. Thus, the appearance of the phrases “in some embodiments” or “in other embodiments” in various places throughout this specification are not necessarily referring to the same embodiment or embodiments. The disclosed embodiments are not intended to be limiting of the claims.

Descriptions of Example Embodiments

(31) FIG. **1**A exemplifies a computing environment **1**A**00** in which embodiments of the present disclosure can be implemented. As an option, one or more variations of computing environment **1**A**00** or any aspect thereof may be implemented in the context of the architecture and functionality of the embodiments described herein.

(32) FIG. **1**A illustrates aspects pertaining to analyzing the attributes of interactions with content objects to dynamically determine extensible access permissions. Specifically, the figure presents a logical depiction of how the herein disclosed techniques can be used to observe (e.g., enforce) extensible access permissions that are derived from the contexts of content object interaction events. A representative set of high order operations are presented to illustrate how the herein disclosed techniques might be applied in a computing environment.

(33) FIG. **1**A depicts a representative set of users **102** (e.g., user “u**1**”, . . . , user “uN”) who interact (e.g., using user device **103**.sub.1, user device **103**.sub.N) with various instances of content objects **106** managed at a content management system **104**. Users **102** may be users of content management system **104** which facilitates interactions (e.g., authoring, editing, viewing, etc.) over content objects **106** for sharing, collaboration, electronic signing, and/or other purposes. The interactions may be performed by one user, or by multiple users, and/or even autonomously by one or more computing entities (e.g., processes, agents, applications, etc.). Moreover, such interactions may span across multiple departments and/or multiple enterprises.

(34) As illustrated, the interactions by users **102** over content objects **106** in content management

system **104** are governed at least in part by sets of global access permissions **112** associated with the various combinations of users and content objects. In some settings, global access permissions may correspond to not only users and content objects, but also may correspond to connections, services, URLs, and/or other resources associated with the content management system. Such global access permissions may comply with a global permissions model that is implemented and managed at content management system **104**. As used herein, a global permissions model is a scheme that defines an initial set (e.g., a default set) of resource permissions to the many resources of a content management system. In today's dynamic collaboration and shared content management environments, however, using such a global permissions model is deficient. Specifically, the limited flexibility of global access permissions **112** as derived from a global permissions model is deficient in addressing the many dynamic contexts in which user interactions with content objects **106** can occur. Some mechanism is needed to define a second set (e.g., an override set) of resource permissions. Moreover, such a mechanism needs to be flexible such that the second set (e.g., the override set) of resource permissions can be flexibly extended in accordance with a policy that can in turn be used to determine the specifics of the second set of permissions (e.g., the overridden permissions) to apply to the various types of resources and/or interactions in the content management system. As used herein, the act of overriding means to modify a previously established permission that was set by a global permissions model. Such modification serves to address the limited applicability of the aforementioned global permissions model.

(35) Various challenges pertaining to the limited applicability of global permissions models are addressed at least in part by analyzing the attributes of interactions between users **102** and content objects **106** so as to dynamically determine sets of extensible access permissions **122** that govern the interactions. In the embodiment of FIG. **1**A, the techniques are facilitated at least in part by an extensible permissions engine **120** and a set of context-aware access policies **124** implemented at content management system **104**. As used herein, extensible access permissions refer to characteristics that apply to then-current interactions between users and content objects, which characteristics are considered after a global permissions model has been established.

(36) As shown, context-aware access policies **124** may be established by one or more of the users **102** (e.g., enterprise system administrators, etc.). When users **102** invoke interaction events over content objects **106** (operation **1**), global access permissions **112** are first identified to govern the interaction events (operation **2**). Extensible permissions engine **120** accesses the context-aware access policies **124** to retrieve any sets of extensible access permissions **122** to apply to the interaction events, which extensible access permissions are based at least in part on the context of the interaction events (operation **3**).

(37) The context of an interaction event, as used herein, comprises the set of conditions that correspond to the interaction event. Such conditions are often described by one or more interaction attributes. The interaction attributes can comprise event attributes and other attributes derived from the event attributes that correspond to a particular interaction event. In this case, the interaction attributes are applied to context-aware access policies **124** to determine the sets of extensible access permissions **122** for the respective interaction events.

(38) As used herein, context-aware access policies are groups of extensible permissions rules, which extensible permissions rules are evaluated against at least a portion of the one or more interaction attributes to determine a set of extensible access permissions, which extensible access permissions allow or deny access to a content object that already has an assigned access permission. Such interaction attributes might derive from a particular context associated with the particular user interaction and/or such interaction attributes might derive from one or more event attributes, or might derive from one or more object attributes, or might derive from one or more user attributes. The heretofore mentioned contexts are defined at least in part by one or more context-aware access policies, which context-aware access policies can be specified by a user at a user interface.

(39) It can happen that the semantics of extensible access permissions that derive from any one or more context-aware access policies may conflict with the semantics of the global access permissions. As such, the sets of extensible access permissions **122** are reconciled with the global access permissions (operation **4**) to facilitate access to content objects **106** in accordance with the extensible access permissions **122** (operation **5**).
(40) The specific embodiment of FIG. **1**A, includes an e-signature system **105** that is interfaced to the CMS. As shown, the interface is configured such that the CMS **104** can send CMS-initiated commands **119** as well as CMS-derived information **117** to the ESS **105**. Similarly, the shown interface is configured such that the ESS can send ESS-initiated commands **125** as well as ESS-derived information **121** to the CMS. Both the ESS and the CMS can perform document processing (e.g., as depicted by document processing **107**.sub.ESS and document processing **107**.sub.CMS. Such document processing can be implemented as a series of steps that are triggered (or not triggered) based on document processing conditions (e.g., conditions **118**.sub.ESS, conditions **118**.sub.CMS).
(41) CMS document processing triggers and/or ESS document processing triggers (or absence of triggers) can be informed by ESS-derived information or CMS-derived information. In some situations, ESS document processing can be informed by an external user through an unregistered user device (e.g., user device **103**.sub.X). Using the foregoing interface, the ESS can pass ESS-derived information **121** (e.g., information pertaining to the external user or unregistered user device) to the CMS. The CMS can in turn process that ESS-derived information against the context aware policies. In certain situations, many of which are the subject of additional disclosure hereunder, document processing **107**.sub.ESS and/or document processing **107**.sub.CMS might be terminated based on consideration of the context aware policies with respect to characteristics of the external user or unregistered user device. Strictly as one example, document processing **107**.sub.ESS and/or any step or operation within document processing **107**.sub.ESS might not be triggered, or might be expressly terminated based on the determination that one of the signatories is an external user or that one of the signatories is using an unregistered user device.
(42) As used herein, an external user is a user for whom the CMS does not have a user profile record that associates the external user with an internal user. An internal user is a person for whom the CMS has a profile record associating the person with a particular organization (e.g., company). A different person who does not have a CMS user profile that associates that different person with the same particular organization is deemed to be an external user.
(43) Given that modern e-signing protocols allow for e-signing by many signatories from many different organizations, there is often high risk—at least inasmuch as not every organization and its members can be vetted by the CMS. Nevertheless, means for avoiding leakage needs to be provided. To address leakage prevention, the shown CMS implements automated security and governance controls.
(44) FIG. **1**B**1** shows a security and governance engine within a content management system. The figure is being presented to show how a security and governance engine can be integrated into a content management system in a manner that facilitates control over e-signing processing such that context-aware access policies can be enforced. More specifically, the shown governance engine can allow or deny any aspect, step or activity invocation corresponding to an e-signature request event **149** in order to prevent inadvertent leakage of sensitive information.
(45) To illustrate with merely one example, consider that an administrator or alternate signatory might be named in an e-signing envelope, yet it can happen that that administrator does not have sufficient security clearance to be able to view the materials that are mentioned in the e-signing envelope. Accordingly, the security and governance engine **111** provides the means to automatically apply governance policies **109** so as to prevent leakage (e.g., unauthorized access) to the security-classified materials that are mentioned in the e-signing envelope.
(46) Automatic Observance of Such Governance Policies to Prevent Leakage.

(47) The figure further shows how extensible permissions rules **146** and context aware access policies **124** can inform the security and governance engine so as to automatically prevent leakage of the security-classified materials even in settings where permissions, rules and/or policies are changing dynamically as users interact with the CMS.
(48) As used herein, a governance policy is a computer representation of an organizational decision that has been adopted by the organization. Within a particular organization, governance policies define capabilities and/or limits of the decision-adopting organization. Governance documents that contain organizational decisions from which computer representations of the decisions are formed may include corporate bylaws, articles of incorporation, shareholder agreements, etc. The decisions that are documented in such governance documents can be codified as rules. Such rules, individually or in groups, define governance policies. Governance policies in turn form a portion of the foregoing context-aware access policies **124**.
(49) Based on consideration of the context aware policies with respect to characteristics of the external user or unregistered user device, a particular interaction request (e.g., a sign request, a download request, etc.) might be allowed, or might be blocked. Enforcement of context aware policies serves to eliminate vulnerabilities that would otherwise exist in the absence of such enforcement. This is because, since the CMS is able to command the ESS, in the event that the CMS determines the existence of a vulnerability endemic to the ESS, the CMS can (for example) halt further progression of document processing **107**.sub.CMS, and/or can halt further progression of document processing **107**.sub.ESS. In this way, risk of leakage, where an unvetted user could potentially gain unauthorized access to a document (e.g., for previewing the document, or for downloading the document) can be avoided.
(50) If/when a risk item is identified, a leakage risk assessment **115** as well as corresponding administrative commands **113** can be delivered to an administrator or other user. In some cases, a risk assessment is associated with a quantitative risk score. If the risk score breaches a risk threshold, then the risk is mitigated by changing the way an e-signature is sought. In some cases a risk score that breaches a threshold terminates any e-signing process that is in progress. In some cases a risk score that breaches a risk threshold prevents any new e-signing process from being initiated until such time that the risk assessment is recalculated and the risk score does not breach the risk threshold.
(51) Dynamically changing contexts, such as when additional organizational decisions are taken, can be codified into context-aware policies, which in turn can inform any/all steps that comprise e-signature processes. A selection of such e-signature processes, as well as how governance policies inform the steps of the e-signature processes, are shown and described as pertains to FIG. **1**B**2**.
(52) FIG. **1**B**2** illustrates how a set of context-aware policies can inform any/all steps of an e-signature process. Further, the figure depicts one possible configuration where an e-signing security and governance overlay **146** is implemented as a controlling layer that determines how to respond to (e.g., to process or not to process, how to process, etc.) a particular e-signature request event.
(53) In this embodiment, the aforementioned security and governance engine is interfaced directly with a repository of context-aware access policies **124**. As such, the e-signing security and governance overlay **146** controls, at least in part, the progression or termination of steps of the e-signing process. In this embodiment, a subject document is processed through a series of e-signing steps (e.g., step **132** to finalize the document, step **134** to get a certificate for the document, step **136** to generate an envelope, and step **138** to send out the envelope).
(54) Step **136** operates to control generation of an envelope. Step **136** interfaces to the e-signing security and governance overlay **146** in a manner that can allow or deny progression to next steps in the e-signing process. For example, a determination to allow or deny progression to next steps in the e-signing process can be based on security clearance levels of signatories of the e-signing request. In some cases, a security level assigned to a particular signatory might be lower than the security level assigned to a particular item identified in the envelope, in which case the ESS will

deny progression to next steps in the e-signing process.
(55) In some embodiments, progression to next steps in the e-signing process comports to a strict sequence where a next step does not commence until the previous step has completed. In other embodiments, one or more next steps in the e-signing process can be carried out in parallel, where at least portions of two or more steps are executed at the same time.
(56) The e-signing security and governance overlay **146** might control which steps are performed sequentially, and/or which steps can be performed in parallel, and/or which steps are either terminated or not initiated at all. In some cases the e-signing security and governance overlay **146** informs step **136** not to produce an envelope (e.g., via envelope generation rules **137**) and/or not to trigger next steps in the process (e.g., via trigger **135**). Decisions to terminate an e-signing process and/or to not trigger next steps in the process can be taken for a multitude of reasons. As examples, a new rule might cause invalidation of the envelope, or it might be because the security clearance of one of the signatories has been lowered, or it might be because one of the content objects referenced in the envelope has been designated with a higher security level indication, etc., or it might be that a security classification value has been applied to a document (e.g., as a result of a security scan).
(57) The e-signing security and governance overlay **146** can also be configured to prevent addition of a recipient to an envelope (e.g., by informing step **136** of a potential security rule violation). If it happens that a problem is detected, or if it happens that some other event occurs that would at least potentially change how the e-signature process is being carried out, then it can be seen that it is much more efficient and is much less time consuming if on-the-fly change loop **151** is taken rather than restarting at the beginning of the process.
(58) Strictly as one non-limiting example, and strictly for comparative purposes, suppose it were determined that one of the designated e-signatories is “on vacation” or otherwise unavailable. Now, further suppose that the e-signatory who is on vacation has a deputy or other alternate signatory who would be available to e-sign in absence of the e-signatory who is on vacation. In such a case, notice the long loop (e.g., restart loop **151**) incurred by restarting the e-signing process, this time with the deputy named as an e-signatory. It can happen that, even though the alternate signatory might indeed be available, the alternate signatory might not have sufficient security clearances to be able to even view one or more of the content objects that are identified in the envelope.
(59) Step **138** operates to send out the envelope that was generated in step **136**. The act of sending out the envelope that was generated in step **136** can be informed by information from the CMS. In the shown example, a set of permitted channels **139** are provided to step **138**. This can, for instance, avoid vulnerabilities that might exist if unsecured channels (e.g., public WiFi channels) or channels other than the permitted channels were to be used to send the envelope.
(60) The figure also shows how, while waiting for the last signatory, it is possible that new permissions or rules or policies might be introduced into the CMS. Such new permissions or rules or policies would need to control ongoing e-signing processing. This is shown by the introduction of CMS rules inheritance **141** into step **140**. One way to control ongoing e-signing processing is embodied in step **140**, where step **140** enforces that a signatory be actually signed into the CMS before permitting any e-signature activity. In some cases, authentication of a signatory's identity as being the intended signatory is carried out through use of multi-factor authentication regime, a first factor being confirmation of the signatory's CMS user identity and a second factor being the signatory's response to a challenge on the device of record in the CMS. In some cases, the value of a risk score is used to determine whether or not to uplevel confidence of the authentication regime. In some cases, a second or Nth factor confirmation of the signatory's user identity is used to increase the risk score.
(61) Further, to ensure that even though some time might have elapsed while waiting for the last signatory, step **142** is configured to process integrity checks **143** before deeming the e-signing process to be complete and progressing on to post processing.

(62) Post processing (shown as step **144**) may include any sort of processing that would modify content objects or metadata that had been accessed or modified during the e-signing process. Such post processing might include deep content inspection, scanning for applicability of governance rules, scanning for personally identifiable information (PII), or other post-processing operations.
(63) Strictly as one example, it might have been determined during the e-signing process that a particular content object satisfies conditions that would require a particular retention period. As such, post processing might establish the determined retention period designation **145** to the affected content object or objects. In some cases, many content object are assigned the same retention period. In some cases, different content object are assigned different respective retention periods. A retention period establishes, directly (e.g., by a date value) or indirectly (e.g., through use of a calculation), a timeframe during which a content object that is associated with the retention period cannot be deleted.
(64) As yet another example, there might be a rule or policy that explicitly covers (e.g., prevents) alterations of content objects named in the envelope or corresponding manifest until such time as the e-signing process has completed. In such a case, post processing (e.g., processing undertaken after successful or unsuccessful completion of the e-signing process) might be undertaken to release the content objects that were subject to the foregoing rule or policy.
(65) Any of the foregoing e-signing processes might be carried out in whole or in part within a collaboration system that includes ongoing management of content objects such that a plurality of collaborators can concurrently access the content objects from their respective user devices. As a result, the policy environment is dynamically changing. The security and governance engine **111** and other constituent components of the e-signing security and governance overlay **146** serve to ensure that there is no leakage of sensitive information. More specifically, in the dynamically changing environment, new permissions, rules and policies can arise at any moment in time. In some cases, new rules are introduced into the CMS by users of the CMS.
(66) FIG. **1**B**3** shows examples of rules that may be introduced into the system by way of introduction into a repository for extensible permissions that may in turn be used to govern the progression through any one or more e-signing processes. The shown rules are numbered as Rule #**1**, Rule #**2**, etc. through Rule #**6**. Any or all of the rules can be evaluated by the CMS, and the result(s) of evaluation of the rules can in turn be used to control how the e-signing process is carried out (e.g., to prevent leakage).
(67) Strictly as one example, and referring to the shown "Rule #**3**", a potential recipient of an envelope can be deemed to be ineligible to receive such an envelope or can be deemed to be ineligible to be named as a signatory by operation of "Rule #**3**". Specifically, if the potential recipient of an envelope does not have sufficient permissions to download one or more of the content objects named in the envelope, then the potential recipient can be marked as "Ineligible".
(68) Ineligibility of a particular potential recipient to be added to an envelope might be because the particular potential recipient does not have sufficient permissions to access one or more of the documents that are included in the envelope or manifest.
(69) In the event that there are one or more potential recipients who are deemed to be ineligible to download any one or more of the content objects, subsequent steps in the e-signing process might not be triggered. More particularly, when generating an envelope (step **136** of FIG. **1**B**2**), if there are one or more potential recipients who are deemed to be ineligible to download any one or more of the content objects, then processing within the e-signing security and governance overlay **146** can (1) avoid adding the ineligible percipient, and/or (2) emit an error notification and/or (3) terminate progression of the e-signing process.
(70) The foregoing are merely examples. Any number of rules that characterize any conditions and/or actions can be entered into the extensible permissions rules **146**. In some exemplary cases, a rule can address security-related requirements such as enforcing a security time window (e.g., Rule #**4**) and/or a rule can consider a security risk score and take actions when the security risk score is

deemed to be too high (e.g., Rule #**5**).
(71) FIG. **1**B**4** shows an example envelope data structure **1**B**400**. As an option, one or more variations of envelope data structure or any aspect thereof may be implemented in the context of the architecture and functionality of the embodiments described herein and/or in any environment.
(72) As shown, the envelope data structure **160** includes an indication of the user who is the envelope originator **161**, an indication of the user who is the document originator **164**, and an array of e-signatories **166**. In some cases, the array of e-signatories **166** may include only a single e-signatory. This can happen, for example, when a document is self-signed. There are many scenarios where a document is self-signed. Strictly as an illustrative example, consider the case of an offer letter to an employment candidate, where the offer letter is prepared by an HR professional and where the act of e-signing the offer letter by the candidate is merely to acknowledge the candidate's receipt of the offer letter.
(73) In other cases, the array of e-signatories **166** may include not only multiple e-signatories, but also one or more delegates **167** who are considered as alternate or secondary e-signatories. In the event that a primary e-signatory is unavailable, then a delegate can be selected in his/her stead. In some cases, a delegate might not initially have access rights to the attachments referenced in the envelope. In this situation, the delegate might be granted temporary access to such attachments. For example, a delegate might be granted document access privileges for a time period that begins when the envelope is created and ends when the e-signing process has completed.
(74) As heretofore indicated, there are many situations where participants of the e-signing process are not e-signatories. As such, the data structure for envelope **123** includes an array of recipients **168**, each of whom is identified by a name or alias and a role. As used herein, a recipient is a participant of the e-signing process but is not an e-signatory. A recipient can at least potentially make changes to one or more documents to be e-signed, and such a change can be detected as an on-the-fly event that can be remediated using the heretofore disclosed on-the-fly remediation techniques.
(75) Further, the documents are identified in the envelope. More specifically, the envelope data structure includes attachments **170**, which attachments can include identification of a subject document as well as identification of any number of additional attachments such as images, evidence, addenda or rider documents, etc. The attachments themselves need not be included in the envelope, but rather, the identification of any attachment might be in the form of a network access point into the CMS that handles access to a location where the actual bits of the attachments are stored. The nature of such additional attachments might depend on the use case. For example, in insurance claims processing scenarios, an agent of the insured (e.g., an insurance agent) might provide an image (e.g., a photo) or other evidence that substantiates the extent of the insurance and/or the circumstances surrounding the cause of the loss for which the insured is filing (and e-signing) an insurance claim.
(76) In many scenarios, some portion of the e-signing process is carried out by CMS-native workflows or by third-party provided workflows **172**, which workflows can be triggered on the basis of the nature and/or occurrence of e-sign system events. As such, the envelope might specify one or more triggered workflows **174**. Similarly, it can sometimes happen that initiation of processing within an e-signature system arises due to events within and/or emitted from a triggering workflow **176**. Accordingly, the envelope data structure includes an array that facilitates identification and registration of such triggering workflows within the e-signature system. Processing of identified triggering workflows by the e-signature system might include registering specific events of the triggering workflows with event processors of the e-signature system.
(77) There are many situations where one or more of the attachments (e.g., subject documents, images, evidence, etc.) are content objects of the CMS. As such, there may be many situations where there are a potentially large number of CMS users who have at least some access to the content objects that comprise the attachments. In some embodiments, the envelope includes an

array that holds designations of one or more collaboration groups **178**. In some embodiments, the designation(s) codified into an envelope can be accessed by any CMS user device that has access to the envelope, and such a CMS user can expand the designation into the collaboration group constituents.

(78) As used herein, a collaboration group refers to any set of identifiers pertaining to users of a content management system. Such identifiers may include usernames, email aliases, user device identification information, etc. A collaboration group can be associated with any number of attributes and attribute values, and such attributes and attribute values can be inherited by the members of a particular collaboration group. The constituency of a collaboration group serves to aid in cooperative activities over content management system documents and metadata. The constituency of a collaboration group may include collaborators who are not e-signatories and/or who are not named participants in a particular e-signing process.

(79) As used herein, a collaborator refers to a content management system user profile record (e.g., corresponding to a registered user profile) that comprises metadata that, directly or indirectly, defines that user's access to content objects of the content management system. In some situations an e-signatory is a collaborator of the CMS. In other situations an e-signatory is not a collaborator of the CMS.

(80) In certain embodiments, a particular envelope is delivered only to those who are named participants (e.g., e-signatories, delegates, administrators, etc.) in a particular e-signing process. Additionally or alternatively, the envelope is made available to certain collaboration groups (e.g., groups that include at least some of the e-signatories, their delegates, and/or administrators). As such, any user who is a member of one or more of the aforementioned certain collaboration groups can raise one or more events that might in turn affect the e-signing process. In fact, it can happen that a user who is a member of one or more of the aforementioned certain collaboration groups, but is not one of the named participants (e.g., e-signatories, delegates, administrators, etc.), can nevertheless raise an event that might in turn affect the e-signing process for those who are named in the envelope.

(81) As the actions that are encoded into the envelope and/or its corresponding workflow are carried out, and/or as the actions inherent in the context defined by the envelope are carried out, a log of events pertaining to the envelope items is maintained by audit trail events **180**. Events that control ineligibility or that change eligibility to ineligibility (e.g., ineligible array **181**) and/or events that arise due to timeouts (e.g., expired array **182**), or that implicate impossible travel scenarios can be included in the audit trail.

(82) Further details regarding general approaches to assessing ineligibility or impossible travel scenarios, are described in U.S. patent application Ser. No. 16/553,106 titled “AVOIDING USER SESSION MISCLASSIFICATION USING CONFIGURATION AND ACTIVITY FINGERPRINTS”, filed on Aug. 27, 2019, which is hereby incorporated by reference in its entirety.

(83) In some cases a document retention period is established and codified into a document retention array **183**. The document retention array can be configured to codify individual document retention periods for any of the various attachments, or for any other documents of the CMS.

(84) As shown, the data structure for envelope **123** can include a series of audit trail events **180**. In some cases, rather than including the series of audit trail events in the envelope itself, one or more levels of indirection are provided (e.g., via pointers or links) such that the audit trail events can be reconstructed at any moment in time. In some cases, storage of the actual audit trail events is in a repository that has access controls to ensure that the audit trail cannot be altered except by vetted and authenticated users or programs.

(85) In some implementations, the envelope and its contents and/or the audit trail entries inherit security settings from the environment. In some implementations, audit trail events can include an ineligible array **181** that captures determinations that any user is ineligible to take an e-signing

action, and/or audit trail events can include a timestamp and characterization of an expiration event **182**. In some implementations the audit trail can include a document retention array **183** that characterizes a document retention value (e.g., number of years or months, or legal hold, etc.)
(86) Further, in some implementations, the envelope includes authentication seals **186**, together with audit trail events corresponding to failed authentications **187** and/or authentication step-up events **188**. Authentication step-up events can be raised when the CMS and/or ESS determines that authentication is incomplete or suspect. This can happen, for example, if the authentication protocol was being carried out using a public WiFi facility. More particularly, if the CMS and/or ESS had determined that authentication might have been vulnerable to a man-in-the-middle attack, authentication step-up might take the form of requiring a third party or other multi-factor corroboration that the respondent is verifiably the entity being authenticated. In some cases, the authentication step-up might require that a temporary proxy for a registered CMS user profile is generated (e.g., by an authenticatable agent or module of the CMS) and that the respondent be required to log in using a newly-generated credential corresponding to the temporarily registered CMS user.
(87) In some situations, authentication seals can be made available to participants in the e-signing process such that any particular participant can self-verify that the envelope or the subject document(s) or other attachments have not been tampered with. On the other hand, certain governance policies might require that a third party or specific authentication service is to be accessed repeatedly at each step in the e-signing process such that even if some conditions had changed during the e-signing processing (e.g., one or more alternate signatories had been added to the envelope) the e-signing process can nevertheless be guaranteed to comport with the applicable governance policies.
(88) Further details regarding general approaches to specifying policies and observing security controls are described in U.S. patent application Ser. No. 17/706,500 titled “HANDLING ELECTRONIC SIGNATURES IN COLLABORATION SYSTEMS”, filed on Mar. 28, 2022, which is hereby incorporated by reference in its entirety.
(89) As depicted in FIG. **1**A, extensible access permissions **122** serves as a permissions layer (e.g., permission wrapper) that is managed, determined, and/or applied separately from the global access permissions **112**. Moreover, this extensible context-aware permissions capability facilitated by the herein disclosed techniques serves to address the problems attendant to the limited applicability of global permissions models. As such, application of the techniques disclosed herein facilitate improvements in computer functionality that serve to reduce the demand for computer memory, reduce the demand for computer processing power, reduce network bandwidth use, and reduce the demand for intercomponent communication. Specifically, consumption of such computing resources to facilitate manual permissions adjustments and/or global permissions models updates so as to accommodate newly discovered user-to-user and/or user-to-content interactions is eliminated.
(90) One embodiment of techniques for such context-aware extensible access permissions assignment is disclosed in further detail as follows.
(91) FIG. **2** exemplifies a context-aware extensible access permissions assignment technique **200** as implemented in systems that enforce context-aware extensible content object access permissions. As an option, one or more variations of context-aware extensible access permissions assignment technique **200** or any aspect thereof may be implemented in the context of the architecture and functionality of the embodiments described herein. The context-aware extensible access permissions assignment technique **200** or any aspect thereof may be implemented in any environment.
(92) FIG. **2** illustrates aspects pertaining to analyzing the attributes of interactions with content objects to dynamically determine extensible access permissions. Specifically, the figure is presented to illustrate one embodiment of certain steps and/or operations performed over a network

of devices (e.g., user devices, computing systems, etc.) to determine sets of extensible access permissions for the interaction events based at least in part on the context of the interaction events. As can be observed, the steps and/or operations can be grouped into a set of setup operations **210** and a set of ongoing operations **220**.

(93) Setup operations **210** of context-aware extensible access permissions assignment technique **200** commences by identifying a content management system that facilitates interactions over a plurality of users and a plurality of content objects (step **212**). Such interactions can involve both user-to-user interactions and user-to-content interactions. One or more context-aware access policies are established at the content management system to govern interactions between the user and the content objects in accordance with the context of the interactions (step **214**). As used herein, a policy, such as a context-aware access policy, comprises a collection of electronically-stored parameters that serve to constrain one or more computing system operations. More specifically, a context-aware access policy comprises a set of policy parameters that are evaluated subject to the context of an invoked interaction event to determine certain permissions to govern over the interaction event.

(94) As depicted in ongoing operations **220**, when any such interaction events associated with the users and the content objects are detected (step **222**), the respective contexts of the interaction events are determined (step **224**). Such contexts are often described by one or more interaction attributes that can comprise event attributes and other attributes derived from the event attributes (e.g., user attributes, object attributes, resource attributes, etc.) that correspond to the interaction events.

(95) The contexts (e.g., interaction attributes) of the interaction events are applied to the context-aware access policies to determine respective sets of extensible access permissions that correspond to the interaction events (step **226**). As merely one example, the policy parameters of the context-aware access policies might codify sets of extensible permission rules that are evaluated subject to the interaction attributes describing the contexts to determine the sets of extensible access permissions. Then, the extensible access permissions are applied to govern interactions between the users and the content objects (step **228**). Application of the extensible access permissions often raise responses to specific interaction events. The responses are then processed within the content management system. As examples, such responses might allow an interaction, or might block an interaction, or might allow an interaction with justification (e.g., as might be specified by a user desiring to perform the interaction).

(96) One embodiment of a system, data flows, and data structures for implementing the context-aware extensible access permissions assignment technique **200** and/or other herein disclosed techniques, is disclosed as follows.

(97) FIG. **3**A presents a block diagram of a system **3**A**00** that implements and enforces context-aware extensible content object access permissions. As an option, one or more variations of system **3**A**00** or any aspect thereof may be implemented in the context of the architecture and functionality of the embodiments described herein. The system **3**A**00** or any aspect thereof may be implemented in any environment.

(98) FIG. **3**A illustrates aspects pertaining to analyzing the attributes of interactions with content objects to dynamically determine extensible access permissions. Specifically, the figure is being presented to show one embodiment of certain representative components and associated data structures and data flows implemented in a computing environment to facilitate the herein disclosed techniques. As shown, the components, data flows, and data structures are associated with a set of users (e.g., user **102**.sub.1, . . . , user **102**.sub.N) that interact with each other and a set of content objects **106** managed at a content management system **104**. The components, data flows, and data structures shown in FIG. **3**A present one partitioning and associated data manipulation approach. The specific example shown is purely exemplary, and other subsystems, data structures, and/or partitionings are reasonable.

(99) As shown, system **3**A**00** comprises an instance of a content management server **310** operating at content management system **104**. Content management server **310** comprises a message processor **312**, an instance of a global permissions framework **314** that comprises a response generator **318**, and an instance of an extensible permissions engine **120** that comprises a policy conflict remediator **316**. Policy conflicts can be considered whenever an attempt is made to introduce a new policy into the system. Strictly as one example, and as shown, a new policy to be introduced into the system has a set of parameters (e.g., incoming policy parameters **317**), which policy parameters can be considered with respect to previously seen policies. If a conflict is detected, the incoming policy is either rejected (e.g., remanded back to the submitter) or remediated. Strictly as one example of remediation, a policy conflict remediator **316** can resolve apparent conflicts by applying selection criteria and/or conflict remediation rules (e.g., by applying a rule to select the most restrictive or conservative parameters or settings).
(100) A plurality of instances of the foregoing components might operate at a plurality of instances of servers (e.g., content management server **310**) at content management system **104** and/or any portion of system **3**A**00**. Such instances can interact with a communications layer **320** to access each other and/or a set of storage devices **330** that store various information to support the operation of the components of system **3**A**00** and/or any implementations of the herein disclosed techniques.
(101) For example, the servers and/or storage devices of content management system **104** might facilitate interactions over content objects **106** by the users (e.g., user **102**.sub.1, . . . , user **102**.sub.N) from a respective set of user devices (e.g., user device **302**.sub.1, . . . , user device **302**.sub.N). A content management system "manages" a plurality of content objects at least in part by maintaining (e.g., storing, updating, resolving interaction conflicts, etc.) the content objects subject to the various interactions performed over the content objects by the users of the content objects at their respective user devices. The content objects (e.g., files, folders, etc.) in content objects **106** are characterized at least in part by a set of object attributes **340** (e.g., content object metadata) stored at storage devices **330**. Furthermore, the users are characterized at least in part by a set of user attributes **342** stored in a set of user profiles **332** at storage devices **330**.
(102) The users access instances of applications (e.g., app **304**.sub.11, . . . , app **304**.sub.1N) operating at their respective user devices to interact with the content objects **106** managed by content management system **104**. Such applications might comprise native applications provided by content management system **104**, or third-party applications that are integrated with content management system **104**, which applications facilitate initiation and performance of interactions over the content objects. Various information pertaining to such integration with content management system **104** might be codified in a registry at content management system **104**. As indicated, the applications often present user interfaces (e.g., user interface **306**.sub.11, . . . , user interface **306**.sub.1N) to the users to facilitate the interactions with content objects **106** and/or other communications with content management server **310** and/or content management system **104**.
(103) Specifically, the instances of the applications operating at the user devices send or receive various instances of messages **322** that are received or sent by message processor **312** at content management server **310**. In some cases, messages **322** are sent to or received from content management server **310** without human interaction. One class of messages **322** pertains to the policy parameters that constitute various instances of context-aware access policies **124** stored in storage devices **330**.
(104) Such messages might be invoked, for example, by a system administrator (e.g., admin) submitting a policy creation form at a user interface of an application. In this case, the policy parameters derived from the information specified by the system administrator in the policy creation form are embedded in one or more instances of messages **322** and delivered to message processor **312**. The policy parameters are then forwarded to extensible permissions engine **120** for analysis and storage in context-aware access policies **124**.

(105) Such analyses may include certain conflict remediation operations performed by policy conflict remediator **316**. Conflict remediation may be applied when multiple parties specify conflicting policy parameters. The multiples parties may be within one enterprise or over two or more enterprises. As merely one example, a first administrator in an enterprise might specify that no users external to the enterprise can download a particular content object, while a second administrator might specify that external users can download the content object. Policy conflict remediator **316** will identify such conflicts and resolve them according to certain criteria (e.g., always select the most restrictive or conservative setting).
(106) Another class of messages **322** corresponds to interaction events that are invoked by users when they interact with or attempt to interact with content objects **106**. As examples, users might log in to any of the applications to interact with content objects they own or that are shared with them, to invite other users to collaborate on content objects, and/or to perform other collaboration activities. Any of the foregoing interactions or collaboration activities can be characterized as interaction events.
(107) Message processor **312** at content management server **310** monitors the messages **322** to detect interaction events performed over the plurality of users and/or the plurality of content objects. Message processor **312** codifies certain interaction attributes **344** pertaining to interaction events in a set of event records **334** stored in storage devices **330**. In some cases, message processor **312** will access any instances of user attributes **342** (e.g., user enterprise identifiers, etc.) stored in user profiles **332** and/or will access any instances the object attributes **340** (e.g., content object types, etc.) stored in content objects **106** to facilitate the populating of interaction attributes **344** that are included in event records **334**.
(108) Event records **334** and/or any other data described herein can be organized and/or stored using various techniques. For example, event records **334** might be organized and/or stored in a tabular structure (e.g., relational database table) that has rows that relate various interaction attributes with a particular interaction event. As another example, the event data might be organized and/or stored in a programming code object that has instances corresponding to a particular interaction event and properties that describe the various attributes associated with the event.
(109) When interaction events are detected at message processor **312**, the global permissions framework **314** assigns respective sets of global access permissions **112**, as determined by a global permissions model **324**, to govern the interaction events. Global permissions framework **314** then calls extensible permissions engine **120** to retrieve any instances of extensible access permissions **122** that are to be assigned to the interaction events. Extensible permissions engine **120** applies the interaction attributes that describe the respective contexts of the interaction events to the context-aware access policies **124** to determine the extensible access permissions for the events.
(110) As shown, some or all of the interaction attributes may be applied to sets of extensible permissions rules **346** associated with respective instances of the context-aware access policies **124** to facilitate determination of the extensible access permissions. In some cases, extensible access permissions **122** will override global access permissions **112** to govern the interaction events whereas, in other cases, a combination of extensible access permissions **122** and global access permissions **112** may govern the interaction events.
(111) Response generator **318** receives sets of extensible access permissions **122** determined by extensible permissions engine **120** to generate and issue responses to the sources of the interaction events. Specifically, interaction responses generated by response generator **318** are issued as instances of messages **322** to the user devices of the users. For example, consider an interaction event that pertains to a preview of a content object by a user. A set of extensible access permissions corresponding to this interaction event indicates that the content object can be downloaded by the user if a business justification is provided. In this case, response generator **318** will receive the set of extensible access permissions and generate an interaction response that presents a modal to the user to request selection by the user of one of several business justification descriptions. When an

acceptable business justification is submitted by the user, the content object is provisioned (e.g., in an instance of messages **322**) to the user.

(112) Further details pertaining to techniques for establishing (e.g., specifying, recording, etc.) context-aware access policies to govern interactions over content objects (e.g., step **214** of FIG. **2**) are disclosed infra.

(113) Now, returning to the topic of policy conflicts that can at least potentially occur when new policies are introduced into the system, FIG. **3**B discloses a system for reconciling a new policy with respect to previously seen policies. The system of FIG. **3**B is one possible representation of the aforementioned policy conflict remediator **316**.

(114) FIG. **3**B presents a block diagram of a system **3**B**00** that reconciles context-aware extensible content object access permissions. The figure shows a policy that is codified as a set of incoming policy parameters **317**, which policy parameters can be considered with respect to previously seen policies. Consideration of an incoming policy can include determining if the newly-incoming policy is in any way in conflict with previously considered policies. This particular embodiment compares the newly-incoming policy with all previously considered policies by: (1) formulating (step **350**) the newly-incoming policy into a canonical representation of the set of policy parameters; (2) retrieving (step **352**) a list of policies **327**; then (3) for each policy of the list, comparing (step **354**) the canonical representation of the newly-incoming policy to the canonical representation of the other policies.

(115) If, while considering any particular policy, it is found that there is a conflict (decision **356**), then the conflict is remediated or, if remediation is not provided for, then the newly-incoming policy is rejected (step **358**). In some cases, remediation in event of a conflict can be implemented by changing a policy parameter to a value that is more restrictive or more conservative. In some cases, remediation in event of a conflict can be implemented by introducing a new incoming rule. For example, in event of specification of a retention period of 3 years, but there is a pre-existing policy or rule that requires a retention plan of 5 years, then the processing of step **358** might impute a 5-year retention period. In some situations, the canonical representation of the newly-incoming rule or policy in its original form is retained for future consideration.

(116) FIG. **4**A depicts a context-aware access policy specification technique **4**A**00** as implemented in systems that facilitate use of context-aware extensible content object access permissions. As an option, one or more variations of context-aware access policy specification technique **4**A**00** or any aspect thereof may be implemented in the context of the architecture and functionality of the embodiments described herein. The context-aware access policy specification technique **4**A**00** or any aspect thereof may be implemented in any environment.

(117) FIG. **4**A illustrates aspects pertaining to analyzing the attributes of interactions with content objects to dynamically determine extensible access permissions. Specifically, the figure is presented to illustrate one embodiment of a user interface and data structure for specifying and recording policy parameters of instances of context-aware access policies. The flow depicted in the figure can represent an embodiment of step **214** of FIG. **2**.

(118) The context-aware access policy specification technique **4**A**00** is facilitated at least in part by a policy specification view **422** presented at user interface **306**.sub.1N of user device **302**.sub.N associated with user **102**.sub.N. User **102**.sub.N might be a system administrator of an enterprise who is responsible for establishing at least some of the context-aware access policies for the enterprise. To assist such system administrators, a context management system might provide the policy specification view **422** or other views to collect the user inputs that specify the policy parameters of the context-aware access policies.

(119) Specifically, policy specification view **422** comprises user interface elements (e.g., text boxes, radio buttons, dropdown selection lists, switches, checkboxes, etc.) to collect such user inputs. As can be observed, user **102**.sub.N can manipulate the user interface elements to specify a policy name (e.g., entered in a “Policy Name” text box), a policy description (e.g., entered in a

"Description" text box), a classification label associated with the policy (e.g., enabled by a radio button and selected in a dropdown in a "Content Type" section), one or more security controls (e.g., as selected in a "Security Controls" section), and/or other policy attributes. A classification label, as used herein, is a data object that is assigned to one or more content objects to identify a respective context-aware access policy to apply to the content objects having that classification label.

(120) As can be observed in policy specification view **422**, for example, a classification label defined by the text string "Confidential" is associated with the "Classified Files" policy. As such, any interactions with any content objects having the classification label of "Confidential" will be governed by the "Classified Files" policy. In some cases, multiple classification labels might be associated with a particular context-aware access policy.

(121) Further details regarding general approaches to specifying policies and observing security controls are described in U.S. patent application Ser. No. 16/553,063 titled "POLICY-BASED USER DEVICE SECURITY CHECKS", filed on Aug. 27, 2019, which is hereby incorporated by reference in its entirety.

(122) When user **102**.sub.N clicks the "Save" button in policy specification view **422**, the user inputs from the view are submitted to extensible permissions engine **120**. After certain policy conflict remediation operations are performed by policy conflict remediator **316** as earlier described, the policy parameters associated with the "Classified Files" policy are recorded in context-aware access policies **124**. As depicted in a set of representative policy parameters **424**, each policy in context-aware access policies **124** might be describe by a policy identifier (e.g., stored in a "policyID" field), a policy name (e.g., stored in a "name" field), a policy description (e.g., stored in a "description" field), an enterprise identifier (e.g., stored in an "enterpriseID" field), one or more classification labels (e.g., stored in a "classLabel[ ]" object), one or more extensible permissions rules (e.g., stored in a "rules[ ]" object), and/or other parameters.

(123) As facilitated by the foregoing representative policy parameters, a particular context-aware access policy can be uniquely selected for a subject content object using the classification label (e.g., as specified in the "classLabel[ ]" object) and enterprise identifier (e.g., as specified in the "enterpriseID" field) that corresponds to the subject content object.

(124) A set of rules (e.g., rule base), such as the extensible permissions rules **346** stored in the "rules[ ]" object or any other rules described herein, comprises data records storing various information that can be used to form one or more constraints to apply to certain functions and/or operations. For example, information pertaining to a rule in the rule base might comprise conditional logic operands (e.g., input variables, conditions, constraints, etc.) and/or conditional logic operators (e.g., "if", "then", "and", "or", "greater than", "less than", etc.) for forming a conditional logic statement that returns one or more results. In some cases, the information pertaining to a rule might comprise conditions (e.g., predicates, conditional expressions, field names, field values, etc.) and commands and/or clauses (e.g., "select", "where", "order by", etc.) for forming a data statement (e.g., query) that returns one or more results. As shown in representative policy parameters **424**, the aforementioned conditional logic operands may comprise various interaction attributes (e.g., stored in an "iAttributes[ ]" object) and the aforementioned conditional logic operators may be incorporated into one or more conditional logic statements (e.g., codified in instances of "ruleLogic") that have placeholder for the interaction attributes.

(125) One example of such rules is depicted in a representative extensible permissions rule **426**. The representative extensible permissions rule **426** is formed by extensible permissions engine **120** in response to the "Download restriction" specified in policy specification view **422** "For mobile" devices and/or applications. More specifically, the shown user inputs of policy specification view **422** indicate that any downloads of content objects classified as "Confidential" by an enterprise are allowed with justification when the download requests are from the mobile applications of users external to the enterprise. In response to this download restriction as specified in policy specification view **422**, the representative extensible permissions rule **426** is formed and recorded

in context-aware access policies **124**. The rule specifically states that if a content object has a classification label that is equal to any of the classification labels associated with the policy (e.g., “objLabel==classLabel (any)”), and the application is a mobile application (e.g., “appID== (ALL_MOBILE_APP_IDS)”), and the user is not from the enterprise associated with the content object (e.g., “userEntID!=objEntID”), then access is allow with justification. For all other conditions, access to the content object is allowed.

(126) The foregoing pertains to merely one illustrative set of scenarios. Other scenarios include shared link permissions/restrictions and/or external collaboration permissions/restrictions. Some of such scenarios can be based on user-provided and/or administrator-provided specifications that apply to handling of shared links, and some of such scenarios can be based on user-provided and/or administrator-provided specifications that apply to handling of external collaborators. Strictly as examples, sample user interfaces that support specification of permissions/restrictions and/or specification of external collaboration permissions/restrictions are shown and described as pertains to FIG. **4**B and FIG. **4**C.

(127) FIG. **4**B depicts a shared link restriction specification technique **4**B**00** as implemented in systems that facilitate use of context-aware extensible content object access permissions. As an option, one or more variations of shared link restriction specification technique **4**B**00** or any aspect thereof may be implemented in the context of the architecture and functionality of the embodiments described herein. The shared link restriction specification technique **4**B**00** or any aspect thereof may be implemented in any environment.

(128) As shown, a shared link restrictions modal **432** is included in a user interface. The user interface might be presented to a policy owner who might be a system administrator of an enterprise, and who is responsible for establishing at least some of the context-aware access policies for the enterprise. The user interface facilitates specification of shared link restrictions. Strictly as examples, the user interface might include several options for the policy owner to select. The particular semantics of the options, and the particular graphical user interface mechanism for allowing unambiguous specification of shared link restrictions, are implementation specific and may vary depending on the range of options that are applicable in any given content management system.

(129) Strictly as an illustrative example, the shown shared link restrictions modal includes radio buttons that facilitate specification of one particular choice. In this case, the choices are (1) a shared link restriction whereby links can be accessed by anyone including people outside of the enterprise (no sign-in is required), or (2) a shared link restriction whereby links can be accessed by anyone within the enterprise or by people who are invited to the file or folder, or (3) a shared link restriction whereby links can be accessed only by people invited to the file or folder.

(130) FIG. **4**C depicts an external collaboration restriction specification technique **4**C**00** as implemented in systems that facilitate use of context-aware extensible content object access permissions. As an option, one or more variations of external collaboration restriction specification technique **4**C**00** or any aspect thereof may be implemented in the context of the architecture and functionality of the embodiments described herein. The external collaboration restriction specification technique **4**C**00** or any aspect thereof may be implemented in any environment.

(131) As shown, an external collaborations restrictions modal **442** is included in a user interface. The user interface might be presented to a policy owner who might be a system administrator of an enterprise and who is responsible for establishing at least some of the context-aware access policies for the enterprise.

(132) Strictly as an illustrative example, the shown external collaboration restrictions modal includes radio buttons that facilitate specification of one particular choice. In this case, the choices are (1) block all external collaboration, or (2) allow collaboration events for everyone except certain organizations, or (3) allow collaboration events only for approved organizations. The example further depicts a user interface widget that is configured to facilitate whitelisted domains

(e.g., example.com) and/or whitelists (e.g., WhiteList.docx) and/or specific users (e.g., Bob@zz.com).
(133) As used herein, a whitelist **434** is a collection of specific domain names and/or specific document names, and/or specific usernames or aliases. In the situation where an access event is subject to evaluation with respect to extensible permissions, the determination to allow or deny the access is made after considering whether or not the user (or computer program) is reliably associated with any of (1) the whitelisted domains, and/or (2) the identities given in a specified whitelist, and/or (3) identified as a whitelisted user.
(134) The foregoing discussions pertaining to FIG. **4**A, FIG. **4**B, and FIG. **4**C include techniques for detecting interaction events, techniques for specifying restrictions that influence what types of interactions and/or what users are allowed or denied to complete the interactions, and techniques for determining the contexts of those interactions. Certain aspects of such techniques are disclosed in further detail as follows.
(135) FIG. **5** depicts an interaction event context identification technique **500** as implemented in systems that facilitate use of context-aware extensible content object access permissions. As an option, one or more variations of interaction event context identification technique **500** or any aspect thereof may be implemented in the context of the architecture and functionality of the embodiments described herein. The interaction event context identification technique **500** or any aspect thereof may be implemented in any environment.
(136) FIG. **5** illustrates aspects pertaining to analyzing the attributes of interactions with content objects to dynamically determine extensible access permissions. Specifically, the figure is presented to illustrate one embodiment of certain steps and/or operations that facilitate detecting interaction events and determining the contexts of those events. As depicted in the figure, the steps and/or operations are associated with step **222** and step **224** of FIG. **2**. A representative scenario is also shown in the figure to illustrate an example application of interaction event context identification technique **500**.
(137) The interaction event context identification technique **500** commences by monitoring a content management system for interaction events associated with content objects managed by the system (step **502**). As illustrated, one or more instances of message processor **312** might monitor all interaction events invoked by users of the content management system. In response to any one of the interaction events, an interaction event message is received (step **504**). Referring to the scenario of FIG. **5**, consider an interaction event that pertains to user **102**.sub.1 clicking on a download icon in a download view **530**.sub.1 to download a file named “Strategy.pptx”. As shown, the download view **530**.sub.1 might be presented in a user interface **306**.sub.11 of app **304**.sub.11. In response to the foregoing user action, an interaction event message **520** is received at message processor **312**.
(138) The interaction event message is parsed to retrieve the event attributes of the interaction event (step **506**). As indicated in a set of select event attributes **522**, interaction event message **520** might be characterized by attributes that describe an event identifier (e.g., stored in an “eventID” field), a user identifier (e.g., stored in a “userID” field), a content object identifier (e.g., stored in an “objID” field), an application identifier (e.g., stored in an “appID” field), an interaction type (e.g., stored in an “interaction” field), and/or other attributes. In some situations, additional attributes might be required to characterize the interaction event (“Yes” path of decision **508**), in which case various other attributes derived from the event attributes are retrieved (step **510**).
(139) As shown, certain content objects attributes (e.g., content object metadata) might be retrieved from content objects **106** using the content object identifier (e.g., “objID” attribute) from interaction event message **520**, and/or certain user attributes might be retrieved from user profiles **332** using the user identifier (e.g., “userID” attribute) from interaction event message **520**. As depicted in a set of select object attributes **524**, the retrieved object attributes might include a security classification label assigned to the content object (e.g., stored in an “objLabel” field). A security classification label might be assigned to the content object by operation of a rule or parser

that considers the subject matter of the content object. For example, a particular content object might be associated with subject matter that is known to be sensitive, and as such is labeled such that the attribute “objLabel” takes on the value “sensitive”. Or, a particular content object might be deemed to be a legal document of a nature that is automatically labeled as “sensitive”.

(140) Alternatively, or in addition to the foregoing event attributes, additional event attributes might include an enterprise identifier associated with the content object (e.g., stored in an “objEntID” field), a content object owner identifier (e.g., stored in an “ownerID” field), and/or other object attributes. Furthermore, and as depicted in a set of select user attributes **526**, the retrieved user attributes might include a user name (e.g., stored in a “userName” field), an enterprise identifier associated with the user (e.g., stored in a “userEntID” field), one or more roles assigned to the user (e.g., stored in a “role[ ]” object), and/or other user attributes. As shown, each role (e.g., owner, co-owner, editor, viewer, previewer, uploader, etc.) assigned to a particular user can be represented by a role identifier (e.g., stored in a “roleID”) and the object identifier (e.g., stored in an “objID” field) of the content object to which the role identifier applies. Such roles can be accessed to apply, for example, a specific set of access permissions that correspond to specific types of interaction events. Application of such a specific set of access permissions may be in accordance with a global permissions model.

(141) When all desired attributes associated with an interaction event are retrieved (e.g., “No” path of decision **508** or completion of step **510**), then the retrieved attributes are recorded as a set of interaction attributes that characterize the context of the interaction event (step **512**). As depicted in the scenario of FIG. **5**, message processor **312** can record a set of subject interaction attributes **544** associated with interaction event message **520** in event records **334**. In this case, subject interaction attributes **544** characterize an interaction context **540** that corresponds to the download request from user **102**.sub.1 at app **304**.sub.11.

(142) The foregoing discussions include techniques for applying the contexts (e.g., as characterized by interaction attributes) of interaction events to context-aware access policies to determine sets of extensible permissions to govern the interaction events (e.g., step **226** of FIG. **2**), which techniques are disclosed in further detail as follows.

(143) FIG. **6** depicts a rule-based extensible access permissions override technique **600** as implemented in systems that facilitate use of context-aware extensible content object access permissions. As an option, one or more variations of rule-based extensible access permissions override technique **600** or any aspect thereof may be implemented in the context of the architecture and functionality of the embodiments described herein. The rule-based extensible access permissions override technique **600** or any aspect thereof may be implemented in any environment.

(144) FIG. **6** illustrates aspects pertaining to analyzing attributes of interactions with content objects to dynamically determine extensible access permissions. Specifically, the figure is presented to illustrate one embodiment of certain steps and/or operations that facilitate applying the contexts (e.g., as characterized by interaction attributes) of interaction events to context-aware access policies to determine sets of extensible permissions to govern the interaction events. As depicted in the figure, the steps and/or operations are associated with step **226** of FIG. **2**. A representative scenario is also shown in the figure to illustrate an example application of rule-based extensible access permissions override technique **600**.

(145) The rule-based extensible access permissions override technique **600** commences by accessing the interaction attributes that characterize the context of an interaction event (step **602**). Continuing the scenario depicted in FIG. **5**, the subject interaction attributes **544** earlier described are accessed at event records **334** of a content management system. As indicated in a set of select interaction attributes **622**, subject interaction attributes **544** might describe an interaction type (e.g., stored in an “interaction” field), a user identifier (e.g., stored in a “userID” field), a content object identifier (e.g., stored in an “objID” field), an application identifier (e.g., stored in an “appID” field), an enterprise identifier associated with the user (e.g., stored in a “userEntID” field), a role

identifier associated with the user (e.g., stored in a “roleID” field), a classification label assigned to the content object (e.g., stored in an “objLabel” field), an enterprise identifier associated with the content object (e.g., stored in an “objEntID” field), and/or other attributes.

(146) If valid extensible access permissions for the particular set of interaction attributes and associated interaction event are stored in a cache memory (“Yes” path of decision **604**), then the set of extensible access permissions are retrieved from cache memory (step **616**). If no pertinent extensible access permissions are cached (“No” path of decision **604**), then a set of global access permissions are assigned to the interaction event in accordance with a global permissions model (step **606**). As illustrated in the scenario of FIG. **6**, a global permissions framework **314** at the content management system might assign a set of subject resource access permissions **624** to the interaction event that corresponds to subject interaction attributes **544**. Specifically, subject resource access permissions **624** might be derived from the role identifier (e.g., the “roleID” attribute of subject interaction attributes **544**) associated with interaction event. In some cases, subject resource access permissions **624** might be considered the “default” permissions assigned to the interaction event. According to the herein disclosed techniques, such default permissions may be overridden by a set of extensible access permissions as follows.

(147) One or more extensible permissions rules applicable to the interaction attributes are retrieved (step **608**). For example, an instance of extensible permissions engine **120** implemented at the content management system might access a store of context-aware access policies **124** to retrieve a policy that is applicable to the interaction event described by the subject interaction attributes **544**. In some cases, an applicable context-aware access policy from context-aware access policies **124** is identified based at least in part on the classification label assigned to the content object (e.g., the “objLabel” attribute of subject interaction attributes **544**) and the enterprise identifier associated with the content object (e.g., the “objEntID” attribute of subject interaction attributes **544**).

(148) As earlier described, such context-aware access policies often comprise certain respective instances of extensible permissions rules **346**. As such, the applicable rules from among a set of extensible access rules associated with the identified context-aware access policy are evaluated subject to some or all of the interaction attributes to determine a set of rule results (step **610**). Such rule results are, for example, the outcomes of the logical statements that comprise the rules as evaluated in accordance with the interaction attributes. In this case, the rule results describe certain permissions that may or may not override any access permissions that had already been applied to the interaction event. The rule results are reconciled with the resource access permissions to identify a set of extensible access permissions (e.g., subject extensible access permissions **626**) for the interaction event (step **612**).

(149) As one permission reconciliation example, resource access permissions provisioned to a user that permits download of a content object may be overridden by rule results that require downloads of the applicable content are to be blocked. As another permission reconciliation example, resource access permissions provisioned to a user that block downloads of a content object may override a rule that allows downloads. When the set of extensible access permissions for the interaction event are determined, the extensible access permissions (e.g., subject extensible access permissions **626**) are stored in a cache memory (step **614**) for efficient subsequent retrievals (step **616**).

(150) The foregoing discussions include techniques for issuing responses to the interaction events in accordance with the extensible access permissions identified for the events (e.g., step **228** of FIG. **2**), which techniques are disclosed in further detail as follows.

(151) FIG. **7**A depicts an interaction event response technique **7**A**00** as implemented in systems that facilitate use of context-aware extensible content object access permissions. As an option, one or more variations of interaction event response technique **7**A**00** or any aspect thereof may be implemented in the context of the architecture and functionality of the embodiments described herein. The interaction event response technique **7**A**00** or any aspect thereof may be implemented in any environment.

(152) FIG. **7**A illustrates aspects pertaining to analyzing the attributes of interactions with content objects to dynamically determine extensible access permissions. Specifically, the figure is presented to illustrate one embodiment of certain steps and/or operations that facilitate issuing responses to interaction events over content objects in accordance with extensible access permissions identified for the interaction events. As depicted in the figure, the steps and/or operations are associated with step **228** of FIG. **2**. A representative scenario is also shown in the figure to illustrate an example application of interaction event response technique **7**A**00**.
(153) The interaction event response technique **7**A**00** commences by accessing a set of extensible access permissions (step **702**) associated with an interaction event such as may be codified in a collaboration activity request **753**.sub.1. Continuing the scenario depicted in FIG. **5** and FIG. **6**, the subject extensible access permissions **626** earlier described are accessed by an instance of a response generator **318** at a content management system.
(154) It sometimes happens that a single-level inquiry to reach a binary "Yes/No" determination as to whether or not to allow a particular collaboration activity is too restrictive. In some cases a second level of inquiry is undertaken. Strictly as an illustrative example, a second-level inquiry might ask a user for a business reason or other justification as to why a particular collaboration activity should be allowed to proceed for a particular user. The result of such a second-level inquiry can inform one or more extensible access permissions **626**.
(155) Specifically, and as indicated in this particular embodiment, a set of select extensible access permissions attributes **722**, define the scope and applicability of extensible access permissions **626**. Such access permissions attributes might describe an event identifier corresponding to the interaction event (e.g., stored in an "eventID" field), a response action identifier (e.g., stored in an "actionID" field), a response reason (e.g., stored in a "reason" field), and/or other attributes that characterize the extensible access permissions. As merely one example, the "actionID" field might be populated with some identifier that refers to an "allow with justification" action, and the corresponding "reason" field might be populated with message that explains the action (e.g., "This content is classified as [objLabel] and requires business justification to download").
(156) Various embodiments pertain to handling business justifications, which justifications may in turn lead to allowing or denying any particular action. To facilitate allowing or denying any particular action based on a business justification, the "actionID" field might be populated with some identifier that refers to an "allow with justification" action. Furthermore, a corresponding "reason" field might be populated with a message that explains the action (e.g., "The content object referred to at this link is subject to external collaboration restrictions and requires a business justification to access.").
(157) An interaction response is generated based on the collaboration activity request and based on the then-current conditions such as the then-current states of the extensible access permissions (step **703**) and/or based on then-current environment settings (step **705**). As can observed, response generator **318** might access some or all of the subject interaction attributes **544** associated with subject extensible access permissions **626** to generate a subject interaction response **724**. The subject interaction attributes **544** can be retrieved from event records **334** using the event identifier (e.g., stored in the "eventID" field) associated with subject extensible access permissions **626**.
(158) At step **705**, a determination is made as to whether or not an extensible permission can be set in a manner that allows the actions pertaining to the collaboration activity request. In some cases such a determination is made based on interaction with a selected user (e.g., the user that raised the collaboration activity request, and/or a different authorized user). In some cases, the selected user is different from the user that raised the collaboration activity request. More specifically, the particular collaboration activity request might be a collaboration activity that is subjected to an interactive determination (step **707**) as to whether or not the collaboration activity request is to be allowed or denied. Scenarios that invoke such authorizations are shown and described as pertains to FIG. **7**B and FIG. **7**C and FIG. **7**D.

(159) In the particular scenario of FIG. **7**A, app **304**.sub.11 is operated by or otherwise associated with user **102**.sub.1. As such, subject interaction response **724** causes an authorization view **531** to be presented in user interface **306**.sub.11 of app **304**.sub.11. The information contained in subject interaction response **724** is accessed to populate response modal view **720**. Specifically, response modal view **720** facilitates the application of subject extensible access permissions **626** by presenting the reason associated with the granting of permissions. The shown response modal view **720** serves as one possible mechanism for receiving the business justification and for allowing a particular collaboration activity (e.g., download) to be carried out by the requestor. More specifically, if an acceptable business justification is provided by an authorized user, then permission to carry out the particular collaboration activity (e.g., to access the content object "Strategy.pptx" file) is granted when the "Send" button is clicked.

(160) The determinations of step **705** and step **707** might be to allow the action or actions corresponding to the collaboration activity request, or the determinations might be to deny the action or actions corresponding to the collaboration activity request, or the determinations might entail some results other than to allow or deny the action or actions corresponding to the collaboration activity request. In any of the foregoing cases, step **709** serves to propagate the determination. Strictly as one example, a determination to allow the action or actions corresponding to the collaboration activity request might be conditioned with respect to a time window.

(161) Various embodiments of step **705**, step **707** and step **709** are shown and described as pertains to FIG. **7**B, FIG. **7**C, and FIG. **7**D.

(162) FIG. **7**B and FIG. **7**C depict aspects of a processing flow to reach an allow determination or a deny determination for a particular collaboration activity request. As an option, one or more variations of the processing flow to reach an allow or deny determination or any aspect thereof may be implemented in the context of the architecture and functionality of the embodiments described herein. The processing flow to reach an allow or deny determination or any aspect thereof may be implemented in any environment.

(163) As shown in FIG. **7**B, a set of environment-based allow or deny determinations are made based on attributes of potential collaborator **752** and/or any information available in the shown collaboration activity request **753**.sub.2. Specifically, step **754** represents a process to check that environmental preferences, variables, settings, etc. are configured to determine whether or not the content management system has been configured to permit use of whitelists and/or determine whether or not the content management system has been configured to permit overrides.

(164) In the case where whitelists are permitted, decision **756** serves to determine if the potential collaborator is listed on any one or more whitelists. The potential collaborator might be deemed to be on a whitelist if the domain name from which the potential collaborator had raised the collaboration activity request appears on a whitelist. Additionally or alternatively, a potential collaborator might be deemed to be on a whitelist if the username or user alias under which the potential collaborator had raised the collaboration activity request appears on a whitelist. When the "Yes" branch of decision **756** is taken, then step **758** executes and the collaboration activity request is permitted to proceed. On the other hand, when the "No" branch of decision **756** is taken, the collaboration activity request is further considered with the potential that the collaboration activity request might be granted based on some criteria other than the whitelist.

(165) Strictly as one example, while a particular content management system might support one or more features for overriding permissions, it can happen that an administrator of the content management system might decide to disable such features. To accommodate this possibility, decision **760** performs checks to determine whether or not the needed features for overriding permissions is enabled. If not, the "No" branch of decision **760** is taken and the request is denied (step **762**).

(166) Some implementations involve criteria other than environmental settings and/or other than

whitelist-related criteria and/or other than override setting criteria. Some such implementations are shown and described as pertains to FIG. **7**C.

(167) As shown, the processing flow of FIG. **7**C is entered after a determination is made at decision **760** that overrides of permissions is enabled in the content management system. More specifically, when the "Yes" branch of decision **760** is taken, the processing flow returns to the caller. In the foregoing example embodiments, the processing returns to the flow of FIG. **7**A just before step **707**. Step **707** may avail of the processing flow of FIG. **7**C. Moreover, the same collaboration activity request **753**.sub.2 of FIG. **7**B is made available to the processing flow of FIG. **7**C. Certain data items that are part of, or are associated with, the collaboration activity request **753**.sub.2 are used by operations within the processing flow of FIG. **7**C. Strictly as one example, the shown collaboration activity request **753**.sub.2 includes an identification of the subject content object that corresponds to the collaboration activity request (e.g., subject content object name **755**).

(168) Using the identification of the subject content object, decision **764** is able to determine if the collaboration activity request over the particular subject content object had been previously allowed (e.g., via a previous user-provided justification). If so, the "Yes" branch of decision **764** is taken and further processing is undertaken to determine if the relevant states of the subject content object and/or its environment have changed state. This further processing is undertaken because the conditions that were previously present as pertains to the subject content object and/or its environment might have changed, thus serving to revoke privileges or otherwise invalidating the previous allowance. One way to keep track of file or folder states is to save particular state values in the metadata of a particular content object. Another way to keep track of file or folder states is to save particular state values in a cache that holds the metadata of a particular content object. Yet another way to keep track of file or folder states is to save particular state values to an audit journal holds an audit trail of collaboration activity requests, timings, then-present environmental conditions, and the specific value or values of the changes made to the metadata of a particular content object. The metadata of a particular content object can be defined to hold values that pertain to file timestamps, file ownership, policies that apply to one or more content objects, classification labels that apply to one or more content objects, a file- or policy-owner's enterprise affiliation, a hierarchical lineage indication, as well as a sibling or peer content object enumeration. In some cases, metadata is stored separately from a respective content object (e.g., in a different field or row of a database table), whereas in some cases metadata is stored with a respective content object (e.g., in a hard drive partition that holds a file directory and its constituent files).

(169) If the "No" branch of decision **764** is taken, then challenge **767** commences by identifying a permission override authority (step **766**), after which identification, a challenge user interface is presented to the identified permission override authority (step **768**).

(170) The permission override authority can, for example, be the owner of the subject content object, or the permission override authority can be, for example, an administrator, or the permission override authority can be, for example, a boss, or can be the owner of a security policy. In some embodiments, the permission override authority can be an identified human who has been sufficiently vetted so as to reduce the likelihood of unauthorized or malicious acts to be taken over subject content objects. In some cases, a secure identification (e.g., an authentication certificate pertaining to an authorized permission override authority) and its association with a policy can be securely stored such that the secure identification (e.g., a certificate) and/or its association with a policy cannot be modified without detection. In some cases, the secure identification (e.g., a certificate pertaining to an authorized permission override authority) is stored together with, or in association with, a whitelist. In some cases, the secure identification is stored together with, or in association with, an override set.

(171) In some cases, the permission override authority can be the collaboration activity requestor, in which case the justification is deemed to be self-certified. In some cases, the permission override authority is an autonomous agent that executes workflows and/or algorithms to determine whether

or not to allow a particular collaboration activity to be performed by a particular collaboration activity requestor. In some cases, such an autonomous agent might deny a particular collaboration activity based on an event-wise behavioral characterization of the particular collaboration activity requestor. In some cases, such an autonomous agent might coordinate resources access permissions across enterprise boundaries. In some cases, such an autonomous agent might be identified with an authentication certificate. Interaction by the identified authority using the authorization user interface, or operation of the aforementioned authenticated agent can result in a particular type of justification, which can in turn be codified into a justification object **769**.

(172) There may be some appreciable amount of time between composition and presentation of the authorization user interface to the identified authority. As such, various temporal checks that would influence whether or not the collaboration activity request should be granted are performed asynchronously, beginning when justification object **769** is received at step **770**. Since it can happen that decision **764** might have been taken in an earlier timeframe, step **770** is carried out to check the then-current temporal currency and other status of the justification at hand. In addition to checking the temporal currency and other status of the justification at hand, further processing might be undertaken to determine whether or not the relevant states of the subject content object and/or its environment have changed state since the performance of decision **764**. If the relevant states of the subject content object and/or its environment have changed state from a previous value, or at least, if the relevant states of the subject content object and/or its environment have changed state in a manner that invalidates the previous allowance, or if the checks of step **770** determine that the justification at hand has expired, then the “Not Current” branch of decision **772** is taken is taken such that the flow control goes to loop **771**, where-after a new challenge is presented.

(173) If the checks of step **770** result in a determination that the authority declines to provide a justification, then the “Not Justified” branch of decision **772** is taken, and the flow control goes to step **773**, where the collaboration activity request is denied.

(174) On the other hand, if it is determined that the allowance is current, and if it is determined that the justification at hand is valid, or for any other reason or combination of reasons, the justification at hand is deemed to be “OK”, then an exception object **723** is populated with the then-current status information as processing proceeds to the “Current and Justified” branch of decision **772**. The aforementioned exception object includes fields to capture details pertaining to the occurrence of the collaboration activity request, fields for environmental and content object states, a field for a final exception determination, fields to capture details pertaining to a time-bounded validity period, etc. Such fields and any populated values are used for further processing of the collaboration activity request and/or for auditing purposes.

(175) In some embodiments a collaboration activity request might be temporally limited, and corresponding checks (e.g., the checks of step **770**) can be undertaken when making allow or deny determinations. Strictly as an example it can happen that a designated authority had limited the allowability of the collaboration activity request to a particular time period (e.g., from May 1 through June 30), and the validity (e.g., currency) of the allowance can be checked against the date range. In some implementations, the time-bounded validity period is stored in the exception object. In other implementations, the time-bounded validity period is stored in metadata pertaining to the subject content object.

(176) After the foregoing tests in which determinations and/or challenge(s) have been accomplished successfully, and if an authority has provided sufficient justification for the override, step **778** is carried out to indicate that the collaboration activity request has been allowed.

(177) FIG. **7**D depicts aspects of a processing flow for propagating results of an allow or deny determination for a particular collaboration activity request. As an option, one or more variations of the processing flow for propagating results or any aspect thereof may be implemented in the context of the architecture and functionality of the embodiments described herein. The processing

flow for propagating results or any aspect thereof may be implemented in any environment.
(178) As shown, exception object **723** is provided to the propagation flow of FIG. **7**D. Aspects of the field values in the exception object are used to inform the operations of FIG. **7**D. Specifically, and as shown, at step **780**, the user interface of the permission override authority is updated to reflect the decisions made in flow **707** of FIG. **7**C. Also, and as shown, at step **782**, the user interface of the requesting user is updated to reflect the result of the challenge that was presented to the override authority. In some cases, the requesting user and the override authority are the same user.
(179) At step **784**, metadata that is associated with the subject content object is updated (e.g., updated metadata **789**). In some situations, such metadata updates serve to override previously indicated permission limitations. In some situations, such metadata updates serve to rescind or revoke previous allowances. Also, additional processing might be carried out to add protections to the subject content object. Strictly as one example, if the subject content object is classified as "high security" then the "Yes" branch of decision **786** is taken and protection facilities are added to the subject content object (step **788**). As another example, if the subject content object is classified as "high security" (decision **786**) then the "Yes" branch of step **786** is taken and the subject content object is automatically watermarked.
(180) Further details regarding general approaches to watermarking are described in U.S. patent application Ser. No. 16/237,552 titled "EVENT-DRIVEN GENERATION OF WATERMARKED PREVIEWS OF AN OBJECT IN A COLLABORATION ENVIRONMENT", filed on Dec. 31, 2018, which is hereby incorporated by reference in its entirety.
(181) In addition to operation of step **784** to apply metadata updates to the subject content object, any content objects that are associated with the subject content object may be conditionally subjected to metadata propagation. Strictly as one example, if the subject content object is a folder, then conditionally, the metadata of the files that are in that folder might be updated (step **790**) to reflect the allowance. The corresponding justification might also be stored in the metadata. Strictly as one example, further content objects that are descendants of a particular content object that has been the subject of an allowance override might be marked in their corresponding metadata with the corresponding justification. In some cases, such as when a security label is applied to a subject descendant, then even though that particular content object had been the subject of an allowance override, that subject descendant might not be marked in their corresponding metadata with the corresponding justification. This is because there are certain security labels than cannot be overridden except by an authorized change that is made to a corresponding security policy. As other examples, if a user has received a justification to collaborate on a folder, then all subfolders and/or child content objects might automatically inherit the justification for that user. Similarly, if user has received a justification to download a folder, then all the child content objects or a subset of the child content objects might inherit the received justification.
(182) Any aspect that pertains to the occurrence of the collaboration activity request, and/or any aspect that pertains to the allowance or denial and/or acceptable justification, or insufficient justification, and/or time-boundedness of any allowance can be delivered to an auditing facility (step **792**). As used herein, an auditing facility is a record of actions or attempted actions made over a content object. The records of an auditing facility may be time stamped, and/or time ordered.
(183) FIG. **7**E depicts a justification user interface **7**E**00**. As an option, one or more variations of the justification user interface or any aspect thereof may be implemented in the context of the architecture and functionality of the embodiments described herein. A justification user interface or any aspect thereof may be implemented in any environment.
(184) As shown, a pulldown user interface widget facilitates choosing from a set of policy-specific justification options **793**. Also shown are user interface widgets that facilitate specification of a start date and end date that defines a time-limited allowability. The end date can be used as a time-based trigger to send a reminder message to the user.

Additional Embodiments of the Disclosure
Additional Practical Application Examples
(185) FIG. **8**A depicts a system **8**A**00** as an arrangement of computing modules that are interconnected so as to operate cooperatively to implement certain of the herein-disclosed embodiments. This and other embodiments present particular arrangements of elements that, individually or as combined, serve to form improved technological processes that address the limited applicability of global permissions models. The partitioning of system **8**A**00** is merely illustrative and other partitions are possible. As an option, the system **8**A**00** may be implemented in the context of the architecture and functionality of the embodiments described herein. Of course, however, the system **8**A**00** or any operation therein may be carried out in any desired environment.
(186) The system **8**A**00** comprises at least one processor and at least one memory, the memory serving to store program instructions corresponding to the operations of the system. As shown, an operation can be implemented in whole or in part using program instructions accessible by a module. The modules are connected to a communication path **8**A**05**, and any operation can communicate with any other operations over communication path **8**A**05**. The modules of the system can, individually or in combination, perform method operations within system **8**A**00**. Any operations performed within system **8**A**00** may be performed in any order unless as may be specified in the claims.
(187) The shown embodiment implements a portion of a computer system, presented as system **8**A**00**, comprising one or more computer processors to execute a set of program code instructions (module **8**A**10**) and modules for accessing memory to hold program code instructions to perform: establishing context-aware access policies that govern user interactions over a plurality of content objects (module **8**A**20**); gathering interaction attributes associated with a particular user interaction over one of the plurality of content objects (module **8**A**30**); applying the context-aware access policies to the interaction attributes to determine a set of extensible access permissions (module **8**A**40**); and generating a response to the particular user interaction, the response being generated in accordance with the set of extensible access permissions (module **8**A**50**).
(188) Some embodiments further comprise acts for applying global resource access permissions to the particular user interaction; and then overriding the resource access permissions with the set of extensible access permissions. In some embodiments, the context-aware access policies are implemented at least in part by evaluation of extensible permissions rules. More specifically, the extensible permissions rules can be evaluated against at least a portion of the interaction attributes to determine a set of extensible access permissions that correspond to the particular interaction.
(189) There are many ways to enforce the permissions. For example, enforcement might take the form of one or more of, taking no action, allowing the interaction, allowing the interaction but only with a user-provided justification, or blocking the interaction.
(190) Variations of the foregoing may include more or fewer of the shown modules. Certain variations may perform more or fewer (or different) steps and/or certain variations may use data elements in more, or in fewer, or in different operations. Still further, some embodiments include variations in the operations performed, and some embodiments include variations of aspects of the data elements used in the operations.
(191) FIG. **8**B depicts a system **8**B**00** as an arrangement of computing modules that are interconnected so as to operate cooperatively to implement certain of the herein-disclosed embodiments. The partitioning of system **8**B**00** is merely illustrative and other partitions are possible. As an option, the system **8**B**00** may be implemented in the context of the architecture and functionality of the embodiments described herein. Of course, however, the system **8**B**00** or any operation therein may be carried out in any desired environment. The system **8**B**00** comprises at least one processor and at least one memory, the memory serving to store program instructions corresponding to the operations of the system. As shown, an operation can be implemented in whole or in part using program instructions accessible by a module. The modules are connected to

a communication path **8**B**05**, and any operation can communicate with any other operations over communication path **8**B**05**. The modules of the system can, individually or in combination, perform method operations within system **8**B**00**. Any operations performed within system **8**B**00** may be performed in any order unless as may be specified in the claims. The shown embodiment implements a portion of a computer system, presented as system **8**B**00**, comprising one or more computer processors to execute a set of program code instructions (module **8**B**10**) and modules for accessing memory to hold program code instructions to perform: establishing a global permissions model that applies a first set of resource access permissions to a shared content object (module **8**B**20**); establishing one or more context-aware access policies that govern user interactions over the shared content object (module **8**B**30**); gathering one or more interaction attributes associated with a particular user interaction over the shared content object (module **8**B**40**); applying the context-aware access policies to the one or more interaction attributes to determine a set of extensible access permissions (module **8**B**50**); and overriding the first set of resource access permissions with the set of extensible access permissions (module **8**B**60**).

(192) FIG. **8**C depicts a system **8**C**00** as an arrangement of computing modules that are interconnected so as to operate cooperatively to implement certain of the herein-disclosed embodiments. The partitioning of system **8**C**00** is merely illustrative and other partitions are possible. As an option, the system **8**C**00** may be implemented in the context of the architecture and functionality of the embodiments described herein. Of course, however, the system **8**C**00** or any operation therein may be carried out in any desired environment. The system **8**C**00** comprises at least one processor and at least one memory, the memory serving to store program instructions corresponding to the operations of the system. As shown, an operation can be implemented in whole or in part using program instructions accessible by a module. The modules are connected to a communication path **8**C**05**, and any operation can communicate with any other operations over communication path **8**C**05**. The modules of the system can, individually or in combination, perform method operations within system **8**C**00**. Any operations performed within system **8**C**00** may be performed in any order unless as may be specified in the claims. The shown embodiment implements a portion of a computer system, presented as system **8**C**00**, comprising one or more computer processors to execute a set of program code instructions (module **8**C**10**) and modules for accessing memory to hold program code instructions to perform: exposing a set of content objects to a user, wherein at least one content object of the set of content objects is associated with a first resource access permission (module **8**C**20**); responsive to an access request to access the at least one content object in a manner that is in conflict with the first resource access permission, and responsive to a user-provided justification, overriding the first resource access permission of the at least one content object with a second resource access permission, wherein the second resource access permission is different from the first resource access permission (module **8**C**30**); and applying the second resource access permission to one or more of the content objects (module **8**C**40**).

(193) FIG. **8**D depicts a system **8**D**00** as an arrangement of computing modules that are interconnected so as to operate cooperatively to implement certain of the herein-disclosed embodiments. The partitioning of system **8**D**00** is merely illustrative and other partitions are possible. As an option, the system **8**D**00** may be implemented in the context of the architecture and functionality of the embodiments described herein. Of course, however, the system **8**D**00** or any operation therein may be carried out in any desired environment. The system **8**D**00** comprises at least one processor and at least one memory, the memory serving to store program instructions corresponding to the operations of the system. As shown, an operation can be implemented in whole or in part using program instructions accessible by a module. The modules are connected to a communication path **8**D**05**, and any operation can communicate with any other operations over communication path **8**D**05**. The modules of the system can, individually or in combination, perform method operations within system **8**D**00**. Any operations performed within system **8**D**00**

may be performed in any order unless as may be specified in the claims. The shown embodiment implements a portion of a computer system, presented as system **8**D**00**, comprising one or more computer processors to execute a set of program code instructions (module **8**D**10**) and modules for accessing memory to hold program code instructions to perform: interfacing an electronic signature system (ESS) to a content management system (CMS) that stores instances of shared files and coordinates user interactions with the shared files, the user interactions being by and between a plurality of CMS users (module **8**D**20**); detecting occurrence of an e-signature request event, wherein at least one signatory corresponds to at least one of the plurality of CMS users (module **8**D**30**); responding to the e-signature request event (module **8**D**40**); by issuing an electronic query to a database of parameters, wherein the database stores a plurality of parameter values that define a set of governance policies, and wherein at least some of the plurality of parameters are selected based on a determination that at least some of the governance policies at least potentially apply to e-signature processing (module **8**D**50**); and by modifying metadata pertaining to a workflow object that is associated with a computer-implemented workflow, wherein an occurrence of a workflow trigger is based upon the determination that at least some of the governance policies at least potentially apply to e-signature processing (module **8**D**60**).

(194) System Architecture Overview

Additional System Architecture Examples

(195) FIG. **9**A depicts a block diagram of an instance of a computer system **9**A**00** suitable for implementing embodiments of the present disclosure. Computer system **9**A**00** includes a bus **906** or other communication mechanism for communicating information. The bus interconnects subsystems and devices such as a central processing unit (CPU), or a multi-core CPU (e.g., data processor **907**), a system memory (e.g., main memory **908**, or an area of random access memory (RAM)), a non-volatile storage device or non-volatile storage area (e.g., read-only memory **909**), an internal storage device **910** or external storage device **913** (e.g., magnetic or optical), a data interface **933**, a communications interface **914** (e.g., PHY, MAC, Ethernet interface, modem, etc.). The aforementioned components are shown within processing element partition **901**, however other partitions are possible. Computer system **9**A**00** further comprises a display **911** (e.g., CRT or LCD), various input devices **912** (e.g., keyboard, cursor control), and an external data repository **931**.

(196) According to an embodiment of the disclosure, computer system **9**A**00** performs specific operations by data processor **907** executing one or more sequences of one or more program instructions contained in a memory. Such instructions (e.g., program instructions **902**.sub.1, program instructions **902**.sub.2, program instructions **902**.sub.3, etc.) can be contained in or can be read into a storage location or memory from any computer readable/usable storage medium such as a static storage device or a disk drive. The sequences can be organized to be accessed by one or more processing entities configured to execute a single process or configured to execute multiple concurrent processes to perform work. A processing entity can be hardware-based (e.g., involving one or more cores) or software-based, and/or can be formed using a combination of hardware and software that implements logic, and/or can carry out computations and/or processing steps using one or more processes and/or one or more tasks and/or one or more threads or any combination thereof.

(197) According to an embodiment of the disclosure, computer system **9**A**00** performs specific networking operations using one or more instances of communications interface **914**. Instances of communications interface **914** may comprise one or more networking ports that are configurable (e.g., pertaining to speed, protocol, physical layer characteristics, media access characteristics, etc.) and any particular instance of communications interface **914** or port thereto can be configured differently from any other particular instance. Portions of a communication protocol can be carried out in whole or in part by any instance of communications interface **914**, and data (e.g., packets, data structures, bit fields, etc.) can be positioned in storage locations within communications interface **914**, or within system memory, and such data can be accessed (e.g., using random access

addressing, or using direct memory access DMA, etc.) by devices such as data processor **907**.
(198) Communications link **915** can be configured to transmit (e.g., send, receive, signal, etc.) any types of communications packets (e.g., communication packet **938**.sub.1, communication packet **938**.sub.N) comprising any organization of data items. The data items can comprise a payload data area **937**, a destination address **936** (e.g., a destination IP address), a source address **935** (e.g., a source IP address), and can include various encodings or formatting of bit fields to populate packet characteristics **934**. In some cases, the packet characteristics include a version identifier, a packet or payload length, a traffic class, a flow label, etc. In some cases, payload data area **937** comprises a data structure that is encoded and/or formatted to fit into byte or word boundaries of the packet.
(199) In some embodiments, hard-wired circuitry may be used in place of or in combination with software instructions to implement aspects of the disclosure. Thus, embodiments of the disclosure are not limited to any specific combination of hardware circuitry and/or software. In embodiments, the term “logic” shall mean any combination of software or hardware that is used to implement all or part of the disclosure.
(200) The term “computer readable medium” or “computer usable medium” as used herein refers to any medium that participates in providing instructions to data processor **907** for execution. Such a medium may take many forms including, but not limited to, non-volatile media and volatile media. Non-volatile media includes, for example, optical or magnetic disks such as disk drives or tape drives. Volatile media includes dynamic memory such as RAM.
(201) Common forms of computer readable media include, for example, floppy disk, flexible disk, hard disk, magnetic tape, or any other magnetic medium; CD-ROM or any other optical medium; punch cards, paper tape, or any other physical medium with patterns of holes; RAM, PROM, EPROM, FLASH-EPROM, or any other memory chip or cartridge, or any other non-transitory computer readable medium. Such data can be stored, for example, in any form of external data repository **931**, which in turn can be formatted into any one or more storage areas, and which can comprise parameterized storage **939** accessible by a key (e.g., filename, table name, block address, offset address, etc.).
(202) Execution of the sequences of instructions to practice certain embodiments of the disclosure are performed by a single instance of a computer system **9**A**00**. According to certain embodiments of the disclosure, two or more instances of computer system **9**A**00** coupled by a communications link **915** (e.g., LAN, public switched telephone network, or wireless network) may perform the sequence of instructions required to practice embodiments of the disclosure using two or more instances of components of computer system **9**A**00**.
(203) Computer system **9**A**00** may transmit and receive messages such as data and/or instructions organized into a data structure (e.g., communications packets). The data structure can include program instructions (e.g., application code **903**), communicated through communications link **915** and communications interface **914**. Received program instructions may be executed by data processor **907** as it is received and/or stored in the shown storage device or in or upon any other non-volatile storage for later execution. Computer system **9**A**00** may communicate through a data interface **933** to a database **932** on an external data repository **931**. Data items in a database can be accessed using a primary key (e.g., a relational database primary key).
(204) Processing element partition **901** is merely one sample partition. Other partitions can include multiple data processors, and/or multiple communications interfaces, and/or multiple storage devices, etc. within a partition. For example, a partition can bound a multi-core processor (e.g., possibly including embedded or co-located memory), or a partition can bound a computing cluster having plurality of computing elements, any of which computing elements are connected directly or indirectly to a communications link. A first partition can be configured to communicate to a second partition. A particular first partition and particular second partition can be congruent (e.g., in a processing element array) or can be different (e.g., comprising disjoint sets of components).
(205) A module as used herein can be implemented using any mix of any portions of the system

memory and any extent of hard-wired circuitry including hard-wired circuitry embodied as a data processor **907**. Some embodiments include one or more special-purpose hardware components (e.g., power control, logic, sensors, transducers, etc.). Some embodiments of a module include instructions that are stored in a memory for execution so as to facilitate operational and/or performance characteristics pertaining to context-aware extensible content object access permissions. A module may include one or more state machines and/or combinational logic used to implement or facilitate the operational and/or performance characteristics pertaining to forming context-aware extensible content object access permissions.

(206) Various implementations of database **932** comprise storage media organized to hold a series of records or files such that individual records or files are accessed using a name or key (e.g., a primary key or a combination of keys and/or query clauses). Such files or records can be organized into one or more data structures (e.g., data structures used to implement or facilitate aspects of context-aware extensible content object access permissions). Such files, records, or data structures can be brought into and/or stored in volatile or non-volatile memory. More specifically, the occurrence and organization of the foregoing files, records, and data structures improve the way that the computer stores and retrieves data in memory, for example, to improve the way data is accessed when the computer is performing operations pertaining to context-aware extensible content object access permissions, and/or for improving the way data is manipulated when performing computerized operations pertaining to analyzing the attributes of interactions with content objects to dynamically determine extensible access permissions.

(207) FIG. **9**B depicts a block diagram of an instance of a cloud-based environment **9**B**00**. Such a cloud-based environment supports access to workspaces through the execution of workspace access code (e.g., workspace access code **942**.sub.0, workspace access code **942**.sub.1, and workspace access code **942**.sub.2). Workspace access code can be executed on any of access devices **952** (e.g., laptop device **952**.sub.4, workstation device **952**.sub.5, IP phone device **952**.sub.3, tablet device **952**.sub.2, smart phone device **952**.sub.1, etc.), and can be configured to access any type of object. Strictly as examples, such objects can be folders or directories or can be files of any filetype. A group of users can form a collaborator group **958**, and a collaborator group can be composed of any types or roles of users. For example, and as shown, a collaborator group can comprise a user collaborator, an administrator collaborator, a creator collaborator, etc. Any user can use any one or more of the access devices, and such access devices can be operated concurrently to provide multiple concurrent sessions and/or other techniques to access workspaces through the workspace access code.

(208) A portion of workspace access code can reside in and be executed on any access device. Any portion of the workspace access code can reside in and be executed on any computing platform **951**, including in a middleware setting. As shown, a portion of the workspace access code resides in and can be executed on one or more processing elements (e.g., processing element **905**.sub.1). The workspace access code can interface with storage devices such as networked storage **955**. Storage of workspaces and/or any constituent files or objects, and/or any other code or scripts or data can be stored in any one or more storage partitions (e.g., storage partition **904**.sub.1). In some environments, a processing element includes forms of storage, such as RAM and/or ROM and/or FLASH, and/or other forms of volatile and non-volatile storage.

(209) A stored workspace can be populated via an upload (e.g., an upload from an access device to a processing element over an upload network path **957**). A stored workspace can be delivered to a particular user and/or shared with other particular users via a download (e.g., a download from a processing element to an access device over a download network path **959**).

(210) In the foregoing specification, the disclosure has been described with reference to specific embodiments thereof. It will however be evident that various modifications and changes may be made thereto without departing from the broader spirit and scope of the disclosure. For example, the above-described process flows are described with reference to a particular ordering of process

actions. However, the ordering of many of the described process actions may be changed without affecting the scope or operation of the disclosure. The specification and drawings are to be regarded in an illustrative sense rather than in a restrictive sense.

Claims

1. A method for e-signature processing comprising: interfacing an electronic signature system (ESS) to a content management system (CMS) that stores instances of content objects and coordinates user interactions with the content objects, the user interactions being by a plurality of CMS users; detecting occurrence of an e-signature request event associated with a content object, wherein at least one signatory corresponds to at least one of the plurality of CMS users; and responding to the e-signature request event by: issuing an electronic query to a database of parameters, wherein the database stores a plurality of parameter values that define a set of governance policies, and wherein at least some of the plurality of parameters are selected based on a determination that at least some of the governance policies apply to e-signature processing; and modifying metadata pertaining to the content object that is associated with a workflow, wherein the metadata is modified to set permissions for the content object to be used for the e-signature processing, a workflow trigger for the workflow is generated for the e-signature processing, and the workflow trigger is based upon the determination that at least some of the set of governance policies apply to e-signature processing.
2. The method of claim 1 wherein at least one governance policy of the set of governance policies specifies a security classification.
3. The method of claim 2, further comprising terminating at least a portion of the e-signature processing when a security classification value of a particular content object of the CMS is higher than a security clearance of a signatory.
4. The method of claim 2, further comprising determining which ones of a plurality of signatories are registered users of the CMS.
5. The method of claim 4, further comprising terminating at least a portion of the e-signature processing when a participant specified in the e-signature request event is not one of the registered users of the CMS.
6. The method of claim 1 wherein a risk score is calculated based on (1) the at least some of the governance policies, (2) a security classification value of file, or (3) a security clearance level of one or more of the signatories.
7. The method of claim 1, further comprising invoking further authentication of a signatory's identity through use of a multi-factor authentication regime.
8. The method of claim 7, wherein a first factor of the multi-factor authentication regime comprises confirmation of the signatory's identity in the CMS, and a second factor of the multi-factor authentication regime comprises confirmation of the signatory's response to a challenge on a device of record in the CMS.
9. The method of claim 1, further comprising invoking a process to establish a retention period designation.
10. The method of claim 9, further comprising applying the retention period designation to one or more content objects of the content management system.
11. The method of claim 1, further comprising identifying one or more policy conflicts wherein the one or more policy conflicts are identified by comparing parameters of a new policy with respect to parameters of previously stored policies.
12. The method of claim 11, wherein a check for a policy conflict is carried out upon occurrence of an attempt to introduce the new policy.
13. A non-transitory computer readable medium having stored thereon a sequence of instructions which, when stored in memory and executed by one or more processors causes the one or more

processors to perform a set of acts for e-signature processing the set of acts comprising: interfacing an electronic signature system (ESS) to a content management system (CMS) that stores instances of content objects and coordinates user interactions with the content objects, the user interactions being by a plurality of CMS users; detecting occurrence of an e-signature request event associated with a content object, wherein at least one signatory corresponds to at least one of the plurality of CMS users; and responding to the e-signature request event by: issuing an electronic query to a database of parameters, wherein the database stores a plurality of parameter values that define a set of governance policies, and wherein at least some of the plurality of parameters are selected based on a determination that at least some of the governance policies apply to e-signature processing; and modifying metadata pertaining to the content object that is associated with a workflow, wherein the metadata is modified to set permissions for the content object to be used for the e-signature processing, occurrence of a workflow trigger for the workflow is generated for the e-signature processing, and the workflow trigger is based upon the determination that at least some of the set of governance policies apply to the e-signature processing.

14. The non-transitory computer readable medium of claim 13 wherein at least one governance policy of the set of governance policies specifies a security classification.

15. The non-transitory computer readable medium of claim 14, further comprising instructions which, when stored in memory and executed by the one or more processors causes the one or more processors to perform acts of terminating at least a portion of the e-signature processing when a security classification value of a particular content object of the CMS is higher than a security clearance of a signatory.

16. The non-transitory computer readable medium of claim 14, further comprising instructions which, when stored in memory and executed by the one or more processors causes the one or more processors to perform acts of determining which ones of a plurality of signatories are registered users of the CMS.

17. The non-transitory computer readable medium of claim 16, further comprising instructions which, when stored in memory and executed by the one or more processors causes the one or more processors to perform acts of terminating at least a portion of the e-signature processing when a participant specified in the e-signature request event is not one of the registered users of the CMS.

18. The non-transitory computer readable medium of claim 13 wherein a risk score is calculated based on (1) the at least some of the governance policies, (2) a security classification value of file, or (3) a security clearance level of one or more of the signatories.

19. A system for e-signature processing comprising: a storage medium having stored thereon a sequence of instructions; and one or more processors that execute the sequence of instructions to cause the one or more processors to perform a set of acts, the set of acts comprising, interfacing an electronic signature system (ESS) to a content management system (CMS) that stores instances of content objects and coordinates user interactions with the content objects, the user interactions being by a plurality of CMS users; detecting occurrence of an e-signature request event associated with a content object, wherein at least one signatory corresponds to at least one of the plurality of CMS users; and responding to the e-signature request event by: issuing an electronic query to a database of parameters, wherein the database stores a plurality of parameter values that define a set of governance policies, and wherein at least some of the plurality of parameters are selected based on a determination that at least some of the governance policies apply to e-signature processing; and modifying metadata pertaining to the content object that is associated with a workflow, wherein the metadata is modified to set permissions for the content object to be used for the e-signature processing, a workflow trigger for the workflow is generated for the e-signature processing, and the workflow trigger is based upon the determination that at least some of the set of governance policies apply to the e-signature processing.

20. The system of claim 19 wherein at least one governance policy of the set of governance policies specifies a security classification.