(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
6 April 2006 (06.04.2006)

PCT

(10) International Publication Number
WO 2006/036656 A2

(51) International Patent Classification:
G06F 17/21 (2006.01) *G06F 3/00* (2006.01)
G06Q 99/00 (2006.01)

(21) International Application Number:
PCT/US2005/033638

(22) International Filing Date:
21 September 2005 (21.09.2005)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
60/612,699 24 September 2004 (24.09.2004) US

(71) Applicant *(for all designated States except US)*: **ENCOMIA, L.P.** [US/US]; 3701 Kirby Drive, Suite 450, Houston, TX 77098 (US).

(72) Inventor; and
(75) Inventor/Applicant *(for US only)*: **DUBINSKY, Andrew, M.** [US/US]; 4935 Braesheatner Drive, Houston, TX 77098 (US).

(74) Agent: **REPASS, James, W.**; Fulbright & Jaworski L.L.P., Fulbright Tower, Suite 5100, 1301 McKinney, Houston, TX 77010-3095 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, LY, MA, MD, MG, MK, MN, MW, MX, MZ, NA, NG, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RU, SC, SD, SE, SG, SK, SL, SM, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, YU, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IS, IT, LT, LU, LV, MC, NL, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:
— *without international search report and to be republished upon receipt of that report*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

WO 2006/036656 A2

(54) Title: A METHOD AND SYSTEM FOR BUILDING AUDIT RULE SETS FOR ELECTRONIC AUDITING OF DOCUMENTS

(57) Abstract: A software-based method for building audit rule sets for auditing electronic documents. In an exemplary embodiment, methods and systems including an editor, schemas, a processing engine, and sample documents are provided. The invention gives users a visual interface for building audit rule sets to examine electronic documents for compliance with legal and regulatory requirements.

A METHOD AND SYSTEM FOR BUILDING AUDIT RULE SETS FOR ELECTRONIC AUDITING OF DOCUMENTS

CROSS-REFERENCE TO RELATED APPLICATIONS

[0001] This application claims priority to U.S. Provisional Application No. 60/612,699, filed September 24, 2004, which is incorporated by references herein in its entirety.

BACKGROUND OF THE INVENTION

[0002] The present invention relates generally to user interfaces in computer systems and more particularly to the use of extensible markup language to determine and describe allowable and forbidden validation of data contained within a legal document describing regulatory structure as evinced by legal text.

[0003] When legislators and government entities draft statutes to regulate commerce, a certain type of legal language is employed to describe the various entities, their relationships, and constraints on behavior of said entities. These drafters often use limiting language and purposeful descriptions to specify the method to which laws, rules, and regulations must be obeyed. Entities affected by these regulations must comply using various means, including the use of computer processing systems to analyze information and determine with simplistic Boolean algebra if the presented information falls within the regulatory guidelines. These tests are authored by computer programmers using a multitude of computer languages on systems of varying size and complexity. All these systems have commonalities, including a limitation that the executable language is transmitted in whole apart from the data it contains. The data may be part (a resource) of the underlying executable program.

[0004] By using a compound document type based on XML, or eXtensible Markup Language and a description of the Boolean logic, a document can be created that contains all needed components for validating and processing a particular information sets' compliance with regulatory information directly relevant to the aforementioned document. This compound document contains sections that describe meta-information about the document, data in the form of discrete values, legal terminology, and legal conditions. The complexity of developing such a system involves the production of legal terms and conditions capable of satisfying the demanding exactitude of computer processing systems while maintaining human readability of legal terminology. This legal terminology must be proved exactly and faithfully represented by

the computer system in order to substantiate the human-readable legal text. The legal text binds the actions of the parties to a particular course or courses of action and represents a meeting of the minds. Computer systems must be able to elucidate the intent of those minds and discern from those legal terms an exact and uncompromising Boolean argument capable of providing two determinate outcomes, true or false. In the event that the computer system cannot discern proper logic structure capable of resolving a binary condition, the resulting indeterminate argument must also be provided. However, indeterminate results are not of themselves proper results from a Boolean condition and represent only an erroneous or unexpected condition.

[0005] What is needed is a method for providing computer processing systems a mechanism for elucidating and validating regulatory statutes and legal constraints.

BRIEF SUMMARY OF THE INVENTION

[0006] The invention is a software program, represented by a graphical user interface (GUI) that presents information and enables users to determine Boolean conditions for data values. A method for creating human-readable legal structures, underpinned by extensible markup language, and a processor capable of accepting the aforementioned documents to enable processing and reporting of compound Boolean tests. In a standard legal document common terms and terms of art gird the structure. In a computer processing system, this structure is created through the use of programming language compiled into binary codes. The invention creates this linkage between legal text and binary codes through the use of extensible markup language. The extensible markup language is not apparent to experts in the state of the art due to the complex knowledge of regulatory structures and legally binding terminology required to construct and develop such as system.

[0007] A further understanding of the nature and advantages of the inventions herein may be realized by reference to the remaining portions of the specification and the attached drawings.

DETAILED DESCRIPTION OF THE INVENTION

SUMMARY OF SYSTEM FUNCTIONS

[0008] Operating as a software program on a computer, system functions include:

[0009] 1. Audit rule and rule-set construction, using a visually-oriented interface that shows how audit rules and rule-sets will look in a hierarchical arrangement.

[0010] 2. Set custom properties and definitions for individual audit rules, allowing for maximum flexibility in document evaluation.

[0011] 3. Integration with secondary application(s) to evaluate documents for legal and regulatory compliance.

TERMS DEFINED

[0012] audit: a method of evaluating any unique data. Very much like a corporation's financial data are "audited" to ensure that financial reporting is correct, a single document or collection of documents of any type can also be "audited" for compliance with legal or regulatory expectations or compliance. Specific to the mortgage industry, mortgage documents are audited regularly for various reasons, including lender compliance, data mining, trend-reporting, compliance with federal, state, and local laws, pooling of loans with similar characteristics for sale to investors, and various other needs.

[0013] audit rule: a specific instructional or conditional parameter based on regulatory or legal requirements that evaluates the document for compliance based on a pass/fail condition.

[0014] audit rule set: a collection of various rules and testing parameters gathered together as a coherent and unified set of rules against which a document is tested for regulatory compliance.

[0015] document: for purposes of this patent application, the words "document" and "electronic document" are used interchangeably. See "electronic document."

[0016] electronic document: the electronic representation of a document as a "file" by a computer system (e.g. a Microsoft Word "file" is a document file, because it represents a textual file in electronic format), vs. a printed document.

[0017] extensible Markup Language (XML): XML is a derivative of hypertext markup language (HTML), a unique programming language used primarily in the World Wide Web environment to represent "web pages." Unlike HTML, which is relatively simple and

constrained, XML is a highly sophisticated and flexible programming language used primarily to build World Wide Web-based software applications that run primarily in World Wide Web browsers.

[0018] LDD (Logical Data Dictionary): a Logical Data Dictionary (LDD) is a compilation of methods, classes, objects, and various other code-based constructs used in a software application. The LDD generally defines the methods, classes. objects, and code-based constructs used in the software application and explains how they are used.

[0019] MISMO: Mortgage Industry Standards and Maintenance Organization (MISMO). MISMO is a standards body organized by the Mortgage Bankers Association (MBA) to establish, and ultimately enforce, standards for electronic mortgages. MISMO has developed a Logical Data Dictionary (LDD) of code-based objects and a code-based hierarchy for populating SMARTDocs with variable source data. http://www.mismo.org.

[0020] SDK (Software Developer's Kit): a Software Developer's Kit (SDK) is a code-based package of tools, methods, classes, and objects provided with a software application that makes it possible for a customer to develop custom software extensions or tools. An SDK is generally licensed with the sold software application.

[0021] SMARTDoc: used interchangeably with "electronic document." A SMARTDoc is an XML-based construct in which "layers" are used to separate static, unchangeable text from variable document-specific data.

[0022] WYSIWYG: "What You See Is What You Get." WYSIWYG editors are common throughout industry and lauded for their ease-of-use and their method of helping the user build documents with a visual interface that allows users to see what their end-document(s) will look like while the documents are being constructed. Examples of WYSIWYG editors include Microsoft Word, Microsoft PowerPoint, and others.

[0023] XPATH: an XML-specific construct that defines where variable objects may be placed in an XML-based SMARTDoc. Requires use of an LDD to define XPATHs and ensure that data objects can be properly matched with their corresponding XPATHs.

PROCESS FLOW

[0024] The process flow that follows is written from a user's perspective and provides insight into how the invention would be used and/or implemented in a business environment.

AUDIT RULE SET CONSTRUCTION WITH THE AUDIT FLOW EDITOR

[0025] The first step in using the invention to examine documents for compliance with legal and regulatory requirements is to construct the audit rule set that will be used to evaluate the SMARTDoc for compliance. Audit rule construction is completed using the Audit Flow Editor, a proprietary custom software application whose sole purpose is to construct audit rule-sets.

[0026] In a hypothetical scenario, a user (e.g. a loan processing clerk in a mortgage organization) would set up the audit rule sets using the following criteria and priorities: (1) bank/mortgage organization business rules; (2) bank/mortgage organization processing & approval rules; (3) legal requirements specific to the state in which the mortgage is being processed.

[0027] The user would open the Audit Flow Editor from the Start | Programs | Encomia menu in a Microsoft Windows environment, and would see the initial program application screen as shown in Figure 1. The initial program application screen displays the application in a standard Microsoft Windows tri-pane format, with the Common Rules pane on the left, the Editor pane in the center, and the Properties pane on the right. Microsoft Windows standard menu and toolbars appear at the top of the application window.

[0028] The user would begin building the audit rule set by dragging a common rule from the Rules pane into the editor pane. The Rules pane, shown in Figure 2, contains pre-formatted common rules used in audit rule construction as basic templates on which to build a complex rule-set.

[0029] The Common Rules are explained in detail in Table 1. The Common Rules are provided in "default" mode with basic structure and few specific properties set so that the user can define rule properties as needed.

Rule	Description
 Default rule	The Default rule, or base rule is an empty rule container that can be configured and named as needed. It is a general rule with no specific properties.
 Comparison	The Comparison rule is used to compare two or more conditions or specific properties in the document on a pass/fail basis.
 Include	The Include rule is used to specify inclusion of an external rule-set or file during rule-processing.
 Script	The Script rule is used to specify inclusion of an external script during rule-processing.
 EMail	The EMail rule is used to specify that notification emails be sent to any number of recipients during or after rule-processing. This rule could be used to notify persons that rule-processing has passed or failed, and could be used to notify persons elsewhere in the mortgage process that the subject document has been audited for compliance.
 Application	The Application rule is used to specify processing of an external application during rule-processing.
 Assign agent	The Assign Agent rule is used to assign a specific individual to the mortgage application and rule-processing event. Assignment takes place during rule-processing, and this rule would be most commonly used with the EMail rule to notify the individual of their assignment.

Table 1: Common Rule Descriptions

[0030] As the user drags the first rule from the Rules pane to the Editor pane, the Audit rule-set begins to take shape. The user would note the Properties pane on the right (Figure 3), where rule-specific properties are displayed and changed as needed to suit the specific business and legal environment.

[0031] The user can "drag-n-drop" rules from the Rules pane to the Editor pane, where the rule-set is constructed in a visual environment that clearly shows the relationships between individual rule elements, including processing order.

[0032] FIGS. 4 - 12 illustrate a representative example of an audit rule construction process. These examples of the audit rule construction process show the visually-oriented drag-and-drop audit rule construction process, step by step. Audit rule objects are dragged-and-

dropped into the audit rule editing window, where the rules are "assembled" visually, prepared, and tested prior to use. These diagrams are merely examples which should not limit the scope of the claims herein. One of ordinary skill in the art would recognize many other variations, alternatives, and modifications.

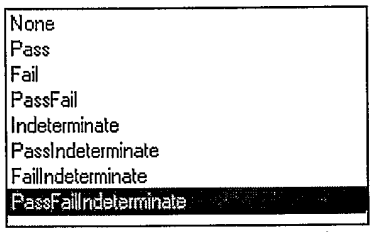
[0033] In Figure 4, the base rule has been dragged into the Editor pane. This base rule serves as the "top-level" of the rule-set, and is the predecessor for all subsequent sub-rules in the rule-set. The BaseRule is a generic container rule that can be defined in a very broad sense as a top-level container rule for subsequent sub-rules.

PROCESSING ORDER

[0034] Rules are processed in sequential order beginning with the base rule, followed by sub-rules.

[0035] In Figure 5, the first sub-rule (ComparePropertyRule) has been dragged into the Editor pane and added as a "child" of the base rule. The three colored symbols next to the sub-rule icon indicate the designated and desired result(s) of the sub-rule processing. Designated results include None, Pass, Fail, PassFail, Indeterminate, PassIndeterminate, FailIndeterminate, and PassFailIndeterminate.

[0036] The ComparePropertyRule is used to compare two or more properties or objects in a SMARTDoc for agreement. The ComparePropertyRule would generally be used in a boolean comparison operation, yielding a Pass/Fail or Yes/No result.



[0037] In Figure 6, a second sub-rule (IncludeRule) has been added at the same level as the first sub-rule (ComparePropertyRule). Because rules are processed sequentially, this rule will be the next rule processed unless sub-rules are added to the ComparePropertyRule.

[0038] The IncludeRule is used to specify inclusion of third-party source data or other exterior data file, etc. during rule-set processing.

[0039] In Figure 7, a sub-rule (ProcessScriptRule) has been added to the IncludeRule. The ProcessScript rule is used to embed processing of an exterior script in the rule-set. An example of script embedding might include the processing of an exterior JavaScript to "clean up" unpopulated or unneeded data objects in the SMARTDoc so that errors are not generated.

[0040] In Figure 8, a sub-rule (Application/ShellExecuteRule) has been added to the ProcessScriptRule. The Application/ShellExecuteRule is used to initiate execution of a third-party software application from inside the rule-set. For example, the Application/ShellExecuteRule could be used to start a particular Microsoft Word file for comparison purposes during the execution of the rule-set.

[0041] In Figure 9, a sub-rule (SendMailRule) has been added to the IncludeRule. Processing order is now as follows: BaseRule, ComparePropertyRule, IncludeRule, ProcessScriptRule, ShellExecuteRule, SendMailRule. The SendMailRule is used to send a plain-text electronic mail (eMail) message to a particular recipient or group of recipient before, during, or after rule processing. An example implementation of this rule might include invoking the SendMailRule to send an eMail message to the loan processing technician to let them know that the rule-set has been successfully processed.

[0042] In Figure 10, another sub-rule (AssignAgent) has been added to the IncludeRule. The AssignAgent rule is used to assign a particular individual to responsibility for the loan package. Combined with the SendMailRule, the AssignAgent rule could be used to assign a particular individual to responsibility for the loan package, and inform them of that responsibility by eMail during processing.

[0043] In Figure 11, additional sub-rules have been added, completing the audit rule set.

[0044] The Audit Rule Set is saved as an Audit file (*.audf).

CREATING A NEW DOCUMENT WITH THE SMARTDOC EDITOR

[0045] To audit a document, the user must create a new document or open an existing document in the SMARTDoc Editor to have a document against which to test the audit rule set.

[0046] The SMARTDoc Editor is a WYSIWYG editor that allows for the creation and formatting of SMARTDocs and SMARTDoc templates. The SMARTDoc Editor is shown in Figure 12.

[0047] To create a new document, the user would start the New Document Wizard by selecting File | New | New Document. The New Document Wizard walks the user through the document creation process step-by-step.

AUDITING DOCUMENTS

[0048] From an end-user's perspective, auditing a document is a simple process once the audit rules and rule-sets have been constructed. Audit rule-sets must have been constructed first, using the Audit Flow Editor as described in AUDIT RULE SET CONSTRUCTION WITH THE AUDIT FLOW EDITOR. A new SMARTDoc or SMARTDoc template must have been created with the SMARTDoc Editor as described in CREATING A NEW DOCUMENT WITH THE SMARTDOC EDITOR.

[0049] The invention is designed to be integrated with other software applications, such as the SMARTDoc Editor and the eVault. Documents are audited directly from these secondary software applications using custom menu items to invoke the integrated audit processing engine.

AUDITING A DOCUMENT IN THE SMARTDOC EDITOR

[0050] The SMARTDoc Editor is a WYSIWYG editor that is used to develop SMARTDoc templates for mortgage and electronic mortgage applications. An example SMARTDoc template is shown in Figure 12. The audit processing engine is integrated with the SMARTDoc Editor, enabling the user to audit SMARTDoc templates for legal and regulatory compliance from the SMARTDoc Editor.

[0051] The audit processing engine is embedded in the SMARTDoc Editor at the code level as an integrated processor, and is available to the user from a standard menu item in the function menu bar. See Figure 14.

[0052] The user must have set up audit rule-sets as described in AUDIT RULE SET CONSTRUCTION WITH THE AUDIT FLOW EDITOR, and must have created a new

document with the SMARTDoc Editor as described in CREATING A NEW DOCUMENT WITH THE SMARTDOC EDITOR.

[0053] The user selects the document to be audited and the audit file to use for auditing the document, then clicks Audit. The Audit processing engine reports results, including errors, in the Audit Document dialog as shown in Figure 14.

[0054] Figure 15 shows a simplified workflow illustration of steps in the process to create a document, populate it with variable data (an optional step not covered in this patent application), and audit the document for legal and regulatory compliance.

[0055] The following diagram illustrates an overview workflow diagram of steps in the process to create a document, populate it with variable data (an optional step not covered in this patent application), and audit the document for legal and regulatory compliance.

SOURCE DATA

[0056] "Source data" as shown in Figure 15 refers to variable loan-specific data obtained from the lender. This source data is used to populate SMARTDocs with loan-specific data, and may be tested against in the current invention for regulatory and legal compliance.

[0057] Variable loan-specific source data may be obtained from the lender in any number of methods or formats. An example of this source data would be "Borrower Name" and "Borrower Phone Number." A lender might obtain this data directly from a loan application, or might obtain this data from a loan officer following an interview with a loan applicant. Regardless of method for obtaining the data, the data is entered into a Loan Origination System (LOS) by most lenders. There are many LOS in the mortgage market today, including (for example) Calyx Point, Genesis2000, and Byte, to name but a few.

[0058] Documents are populated with variable source data using the Encomia Logical Data Dictionary (LDD), which is based on the MISMO LDD. For example, the XPATH for the Subject Property Address is /SMART_DOCUMENT/DATA/MAIN/LOAN/_APPLICATION/PROPERTY/_StreetAddress. When a new SMARTDoc is populated with variable source data, the Populate processing engine examines the source data file for XPATHs, then examines the SMARTDoc and looks for corresponding XPATH data objects in the SMARTDoc. If the source data file contains an

XPATH for Subject Property Address and corresponding data, the Populate processing engine matches the XPATH in the source data file with the matching XPATH data object in the SMARTDoc and inserts the variable source data from the XPATH Subject Property Address field in the source data file into the corresponding data object placeholder in the SMARTDoc. Encomia's Framework SDK (Software Developer's Kit) also plays a role in this process, providing the "bridge" between the MISMO LDD and the Encomia LDD, acting as a sort of "translator" for data objects.

[0059] In Figure 17, a sample SMARTDoc, representative of the type(s) of documents that would be used in the software application, is shown. Common source data for this document would include (but is not limited to) BorrowerName, PropertyAddress, PropertyCity, PropertyState, PropertyZipCode, BorrowerPhone, LoanAmount, InterestRate, LenderName, PaymentDate, and PaymentAddress. Common rules that would be used in auditing this document include (but are not limited to) BaseRule, Comparison, Include, and Script.

[0060] Figure 18 is a flow chart of one embodiment of create a document with audit rules.

[0061] The present invention is typically embodied as software programming code which may be stored in permanent storage of some type such as the permanent storage of the workstation. In a client server environment, however, such software programming code could be stored with the storage associated with a server such as storage associated with computer operating as a server. The software programming code embodying techniques according to the present invention can itself be implemented on any of a variety of known media for use with a data processing system such as a removable disk/ette, cassette tape, hard drive or CD ROM. The code may be distributed on such media or distributed to users from the memory or storage of one computer system over a communications network of any given type to other computer systems for use by users of such systems. The techniques and method of embodying software program code on physical media and for distributing or embodying the code via networks are well known and will not be further discussed herein.

[0062] In conclusion, the present invention provides for a method to determine and describe allowable and forbidden validation of data contained within a legal document describing regulatory structure as evinced by legal text. In the foregoing specification, the invention has

been described with reference to specific exemplary embodiments thereof. Many changes or modifications are readily envisioned. For example, different types of regulation may allow for multiple resolvable solutions all perfectly valid, among other changes, are included within other embodiments of the present invention.

[0063] The specification and drawings are, accordingly, to be regarded in an illustrative rather than in a restrictive sense. It will, however, be evident that various modifications and changes may be made thereunto without departing from the broader spirit and scope of the invention as set forth in the claims.

[0064] Moreover, the embodiments described are further intended to explain the best modes for practicing the invention, and to enable others skilled in the art to utilize the invention in such, or other, embodiments and with various modifications required by the particular applications or uses of the present invention. It is intended that the appending claims be construed to included alternative embodiments to the extent that it is permitted by the prior art.

CLAIMS

What is claimed is:

1. A computer software program comprising a method for binding a workflow method with legal text using extensible markup language in which

   a. Use of the invention allows workflow methods and legal text to be bound together for the purpose of creating audit rule sets that can examine electronic document files for compliance with legal requirements and contractual stipulations. Based on these audit rule sets, a computer system is able to determine whether legal and contractual obligations are met in any document file.

   b. Use of the invention allows business logic to be executed by a computer system for the purpose of defining legal contract terms and validating legal contracts for adherence to stated contractual obligations.

   c. Use of the inventions' constraining markup language operating within a computing environment allows computers to transpose regulatory markup into a processing system.

   d. The embedding of the invention into extensible markup language documents and files enables documents to provide additional capability to allow for compliance analysis and communication of regulatory constraints.

   e. The embedding of the invention into secondary software systems enables additional capability to allow for compliance analysis and communication of regulatory constraints.

2. A method for using extensible markup language to accomplish the objectives named in claim 1, in which extensible markup language is used to create a multi-layer SMARTDocument, in which

   a. one layer of the SMARTDocument contains the static legal text, and

   b. a secondary layer of the SMARTDocument contains unique variable data specific to the legal contract.