US 20200082106A1

(19) United States

(12) Patent Application Publication (10) Pub. No.: US 2020/0082106 A1

FOX et al. (43) Pub. Date: Mar. 12, 2020

(54) MANAGEMENT OF CONFIDENTIAL DOCUMENT DISTRIBUTION AND SECURITY

(71) Applicant: **Icon Clinical Research Limited**, Leopardstown (IE)

(72) Inventors: **Ronan FOX**, Leopardstown (IE); **Sean KELLY**, Leopardstown (IE); **Thomas O'LEARY**, Leopardstown (IE); **Anthony CLARKE**, Leopardstown (IE)

(73) Assignee: **Icon Clinical Research Limited**

(21) Appl. No.: **16/560,181**

(22) Filed: **Sep. 4, 2019**

(30) **Foreign Application Priority Data**

Sep. 6, 2018	(EP)	18193030.6
Jan. 30, 2019	(EP)	19154593.8
Aug. 1, 2019	(EP)	19189553.1

Publication Classification

(51) **Int. Cl.**

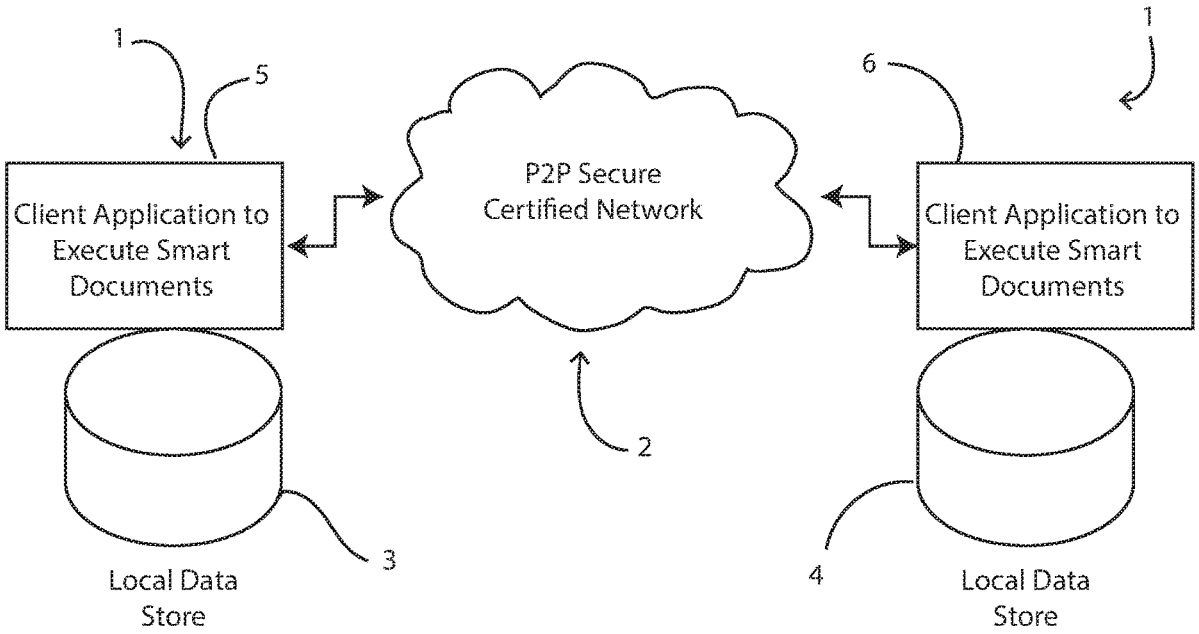
G06F 21/62	(2006.01)
H04L 29/06	(2006.01)
G06F 21/60	(2006.01)

(52) **U.S. Cl.**

CPC ***G06F 21/6209*** (2013.01); *H04L 67/104* (2013.01); ***G06F 21/602*** (2013.01); ***H04L 63/101*** (2013.01)

(57) **ABSTRACT**

A method of managing distribution of a document and access to its contents with security in a network having nodes is described. The nodes may have digital processors for remote communication over a wide area network and local data stores. The exemplary method includes a node sending a document in encrypted format and including a payload, and control data including destination data, access control data, and signature data, and a document destination list. The sending node may only send the document to nodes on the destination list. A node may receive the document, decrypt at least some of the document, and process the document according to the control data by extracting the control data and managing document payload access and document storage and user access according to the control data.



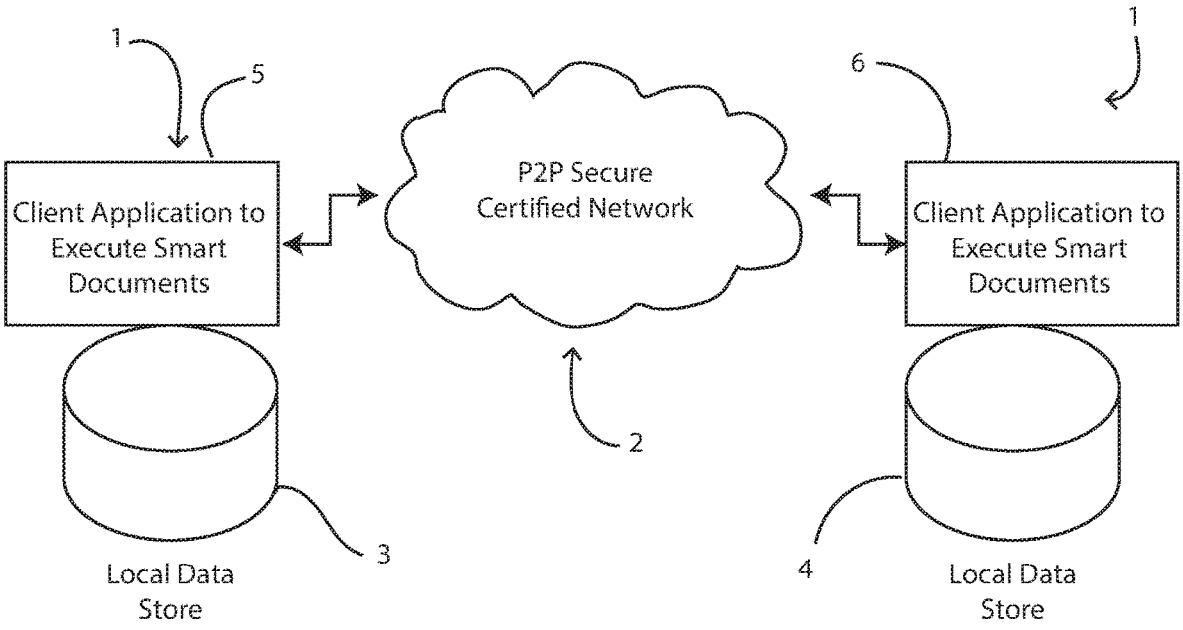

1
5
Client Application to Execute Smart Documents
P2P Secure Certified Network
2
1
6
Client Application to Execute Smart Documents
3
Local Data Store
4
Local Data Store

Fig.1

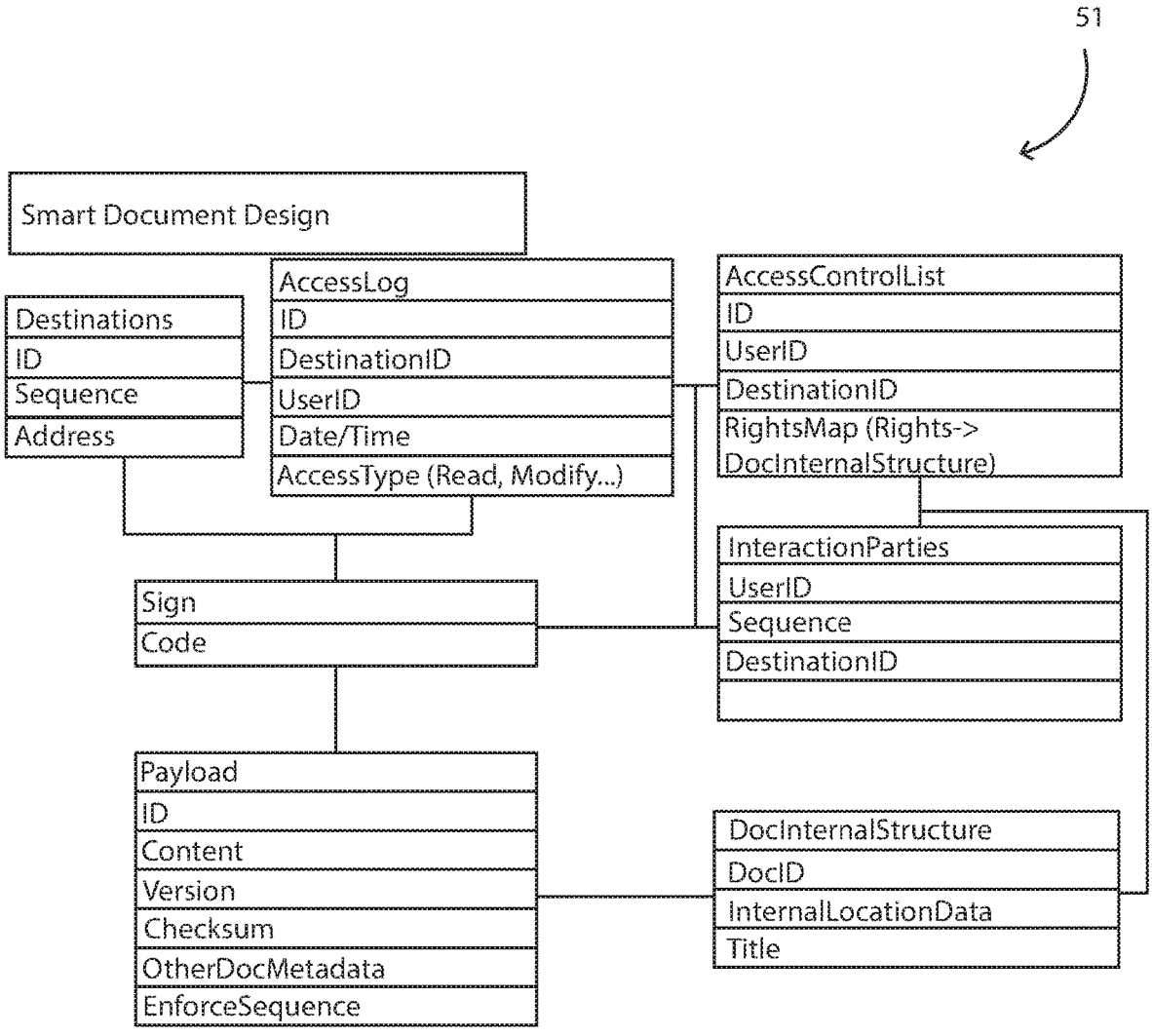

51
Smart Document Design
Destinations
ID
Sequence
Address
AccessLog
ID
DestinationID
UserID
Date/Time
AccessType (Read, Modify...)
AccessControlList
ID
UserID
DestinationID
RightsMap (Rights-> DocInternalStructure)
InteractionParties
UserID
Sequence
DestinationID
Sign
Code
Payload
ID
Content
Version
Checksum
OtherDocMetadata
EnforceSequence
DocInternalStructure
DocID
InternalLocationData
Title

Fig.2

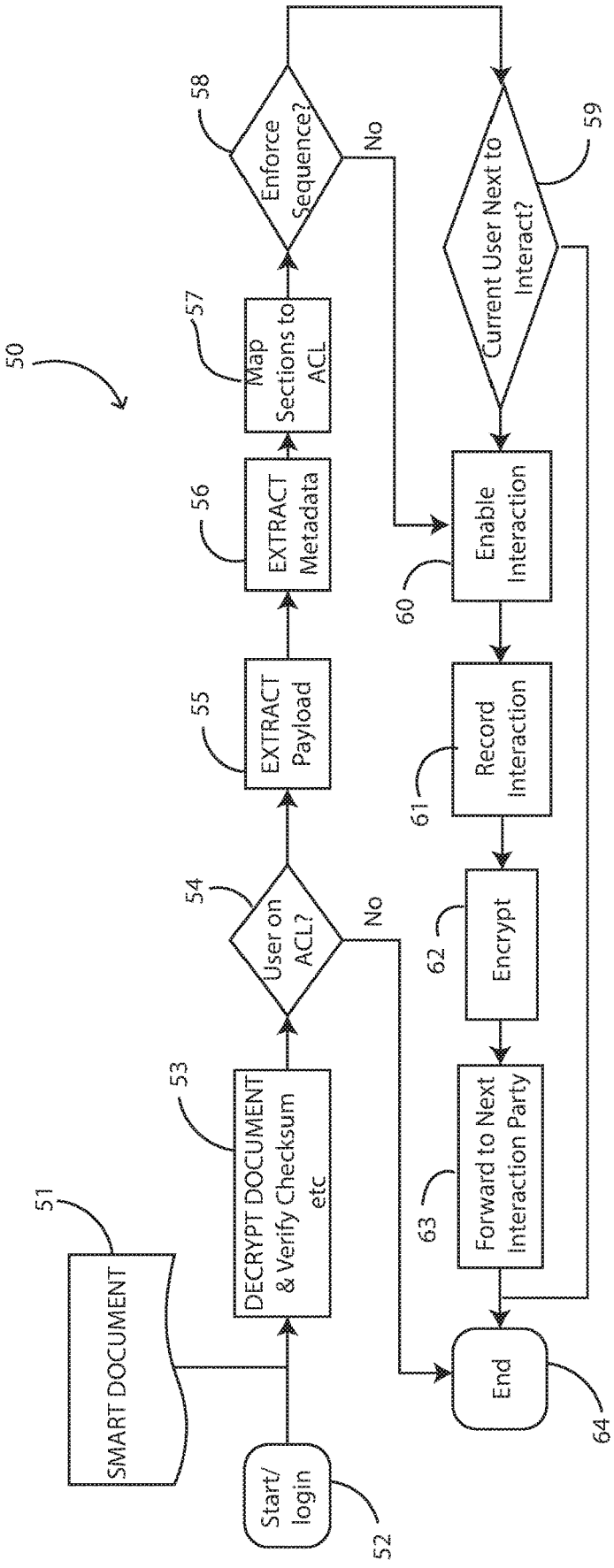

50
51
SMART DOCUMENT
52
Start/ login
53
DECRYPT DOCUMENT & Verify Checksum etc
54
User on ACL?
No
55
EXTRACT Payload
56
EXTRACT Metadata
57
Map Sections to ACL
58
Enforce Sequence?
No
59
Current User Next to Interact?
60
Enable Interaction
61
Record Interaction
62
Encrypt
63
Forward to Next Interaction Party
64
End

Fig.3

MANAGEMENT OF CONFIDENTIAL DOCUMENT DISTRIBUTION AND SECURITY

CROSS-REFERENCE TO RELATED APPLICATIONS

[0001] This patent application claims the benefit of priority under 35 U.S.C. § 119 to European Patent Application No. EP18193030.6 filed on Sep. 6, 2018, European Patent Application No. EP 19154593.8 filed on Jan. 30, 2019, and European Patent Application No. EP19189553.1 filed on Aug. 1, 2019, the entireties of which are incorporated herein by reference.

INTRODUCTION

[0002] The present disclosure relates to management of distribution and security of documents which have confidential information and which must be signed or approved by a number of geographically spread parties.

[0003] Peer-to-peer ("P2P") networks may be provided, in which data is shared between nodes using security such as asymmetric cryptography. Each node may have a local store of documents relevant to workflows and collaborations to which the users have been invited.

[0004] The present disclosure is directed towards achieving improved control of distribution and security of such documents.

SUMMARY OF THE DISCLOSURE

[0005] The present embodiments describe a method of managing distribution of a document and access to its contents with security in a network comprising nodes with digital processors for remote communication over a wide area network and local data stores, the method being implemented by said processors. The method may comprise steps of:

- [0006] a node, acting as a sending node, sending a document in encrypted format into the network and said document including:
  - [0007] a payload with document content, and
  - [0008] control data including destination data, an access control rights map, and signature data, and a document destination list, and
  - [0009] finite state machine FSM executable software code for managing document state.

[0010] A node, acting as a receiving node and executing a software client application, may receive the document, decrypt at least some of it, and process the document according to said control data by extracting the control data and managing document payload access and document storage and user access according to the control data.

[0011] Also, said software client application may act as a listener for documents directed towards that receiving node, and act as a compiler and executor of the control data embedded within the received document, and/or

- [0012] the receiving node software client application may process the control data in a manner transparently to authorised users, and/or
- [0013] the receiving node may execute the finite state machine FSM executable software code to allow it and the document to access a current document state, and to enable the document to move through a document-specific set of allowable states.

[0014] The receiving node may extract the payload and outputs data to authorised users, by mapping sections of the payload to an access control list (ACL), in which each authorized user participant has access to some or all of the document.

[0015] The receiving node may store in an access log of the document a record of authorised user access to the payload.

[0016] The control data may include mapped document sections including:

- [0017] an interested parties section defining authorised users,
- [0018] an access control section defining, for each authorised user, a user identifier and a rights map governing access to the document internal structure,
- [0019] an access log section recording historical user access data including identifier of said user and time and nature of each access.

[0020] The nodes may perform encryption and decryption using public and private keys in which the public key is used to verify and encrypt and the private key is used to decrypt.

[0021] The document may include one or more of:

- [0022] network configuration data including destination nodes and optimisation settings relating to network load for distributing the document in an efficient manner,
- [0023] an access control list specifying levels of authorisation for recipients of the document, an access log which records access to the document, both machine access and human user access,
- [0024] payload attributes including one or more of version numbers, checksums, and priorities, and
- [0025] signature-relations records includes settings relating to a map of signatures and workflow for a sequence of signatures and verifications to be enforced.

[0026] The document may include executable code for dynamic instantiation of the document by the sending node.

[0027] The document may be processed by the receiving node as an instantiation of the FSM which understands, through its executable code, a sequence of document-specific allowable states for that document and provides document-specific features for use by the authorized users.

[0028] The receiving node may determine a next user according to a next actor in a workflow which may be determined according to a map within the document, and said map may be aligned to the current state of the document as held in the FSM code to ensure that the document goes to the next agreed-upon step in the process.

[0029] The document payload may be sent a plurality of times each with a different set of participating parties, and each FSM state in the document may be known ahead of its distribution, and FSM state data may be included in the control data when the document is distributed.

[0030] The method may comprise the sending node performing a look-up of an array of participants to determine at document run time.

[0031] Said look-up may provide instructions for a level of granularity of controlling interaction of a receiving node with specific parts of a document, according to metadata about the document and its constituent parts.

[0032] The document may include semantic definitions of the sections within the document.

[0033] The semantic definitions may be expressed as RDF and/or XML/JSON representations. The rights map may be

included in an access control list providing a desired fine grained and deep level of control.

[0034] The method may comprise allowing an authorized user to sign in with access to one document section, but make no other alterations, or to allow an update to a specific section of the document but not to another section.

[0035] The FSM code and the access control data may be dynamically modified to enable intelligent document analysis at run time to determine any shifts in access control requirements over its lifetime, in which the document may have M states and N authorized users who have read/write access. An authorized user may act on the document and move it from a first state to a second state which causes that user to lose write access, but a subsequent state change caused by a second user may cause said first user to re-gain write access.

[0036] The FSM executable software code may manage the movement of the document from one document state to another document state by providing standard services to the underlying receiving node processor and the document may allow a consistent approach to state changes across all document types, in which the FSM executable code maintains the state of the document.

[0037] Updates to the document may be tracked and controlled by the nodes in a distributed audit trail, while guaranteeing that only those authorized users who have access rights to relevant document sections can interact.

[0038] Additional Statements

[0039] The present disclosure describes a method of managing distribution of a document and access to its contents with security in a network comprising nodes with digital processors for remote communication over a wide area network and local data stores, the method being implemented by said processors and comprising steps of a node sending a document in encrypted format and including: a payload, and control data including destination data, access control data, and signature data, and a document destination list. The sending node may only send the document to nodes on the destination list. A node may receive the document, decrypt at least some of it, and process the document according to said control data by extracting the control data and managing document payload access and document storage and user access according to the control data.

[0040] A dedicated client application on each receiving node may perform said document processing, in which the application may act as a listener for documents directed towards that node, and also act as a compiler and executor of any configuration or code embedded within a received.

[0041] The receiving node may process the control data in a manner transparently to the authorised users. The receiving node may extract the payload and outputs to authorised users, by mapping sections of the payload to an access control list (ACL), in which each authorized user participant has access to some or all of the document.

[0042] The receiving node may store in an access log of the document, a record of authorised user access to the payload.

[0043] The control data may include mapped document sections including:

[0044] an interested parties section defining authorised users,

[0045] an access control section defining, for each authorised user, a user identifier and a rights map governing access to the document internal structure,

[0046] an access log section recording historical user access data including identifier of said user and time and nature of each access.

[0047] The nodes may perform encryption and decryption using public and private keys in which the public key is used to verify and encrypt and the private key is used to decrypt.

[0048] The receiving node may include a finite state machine (FSM) execution code, and the document may include executable code allowing the document to access an underlying state from the node.

[0049] The node executable code may be embedded in the node as part of instantiation of a document before it is sent to the receiving node.

[0050] The document may include one or more of:

[0051] network configuration data including destination nodes and optimisation settings relating to network load for distributing the document in an efficient manner,

[0052] an access control list specifying levels of authorisation for recipients of the document, an access log which records access to the document, both machine access and human user access,

[0053] a payload, and payload attributes including one or more of version numbers, checksums, and priorities, and

[0054] signature-relations records includes settings relating to a map of signatures and workflow for a sequence of signatures and verifications to be enforced.

[0055] The document may include executable code for dynamic instantiation of the document.

[0056] The receiving node may determine a next user according to a next actor in a workflow which is determined either through a FSM embedded executable code in the node or according to a map within the received document.

[0057] The document may be deployed a plurality of times, in which each deployment has a different set of participating parties and each FSM state in the document is known ahead of its distribution and FSM state data is included in the control data when the document is created. When each instance of that document is subsequently deployed, the list of participating parties may change in at least one time that the document is deployed.

[0058] The method may comprise performing a look-up of an array of participants to determine at document run time. Said look-up may provide instructions for a level of granularity of controlling interaction of a node with specific parts of a document, according to meta data about the document and its constituent parts.

[0059] In various embodiments:

[0060] the document includes semantic definitions of the sections within the document, and/or the semantic definitions are expressed as RDF and/or XML/JSON representations; and/or the document structure definition is in in fine-grained control of access right as defined in a rights map; and/or

[0061] the rights map is included in an access control list providing a desired fine grained and deep level of control; and/or

[0062] the method comprises allowing a user to sign in one section, but make no other alterations, or to allow an update to a specific part of the document but not to another section; and/or

[0063] the document structure is machine readable to enable intelligent document analysis at run time to determine any shifts in access control requirements over its lifetime.

[0064] A network comprising a plurality of nodes with digital data processors and storage may be configured to perform a method of any embodiment.

[0065] In another aspect, the present disclosure describes a non-transitory computer storage medium comprising software code for implementing a method of any embodiment when executed by digital data processors.

BRIEF DESCRIPTION OF THE DRAWINGS

[0066] The invention will be more clearly understood from the following description of some embodiments thereof, given by way of example only with reference to the accompanying drawings, in which:

[0067] FIG. **1** is a block diagram showing major components of a document management network;

[0068] FIG. **2** is a diagram illustrating components of a document; and

[0069] FIG. **3** is a flow diagram illustrating a workflow implemented by the network.

DETAILED DESCRIPTION OF THE DISCLOSURE

[0070] The present disclosure describes a method and network for securely distributing documents to remote consumers, along with certifying that the consumers have read those documents (for such things as protocols and standard operating procedures ("SOPs")), and have agreed to the contents within them (as part of a collaborative workflow), or that they have consented to the content using electronic signatures.

[0071] The method and network perform such control to, for example, retain the source documents at a particular site (or a location at which they were created). Also, the nodes on a P2P network synchronise to enable centralised data tracking for reporting and compliance enforcement if required. The network may allow data to be stored at multiple locations, giving a distribution of the documents and associated data set between nodes in the network, and also records auditable transactions against those data sets for a recreation of the data set, regulatory reporting, network resilience, and secure distributed storage.

[0072] The network described herein may include a development of P2P networks in that, while using much of the technology of a P2P network underlying infrastructure, not all data is shared between all nodes in the network. This is very advantageous for applications such as in management of clinical trials. In the case of a clinical trial application, the network may be used by a sponsor, a contract research organisation ("CRO"), and sites across studies. For specific studies and use cases within the performance of a clinical trial study, different partitions may be created in this described network of the invention to allow the sharing of documents between specific nodes in the network to ensure that those who need access to the data have access, without the overhead of all documents being duplicated across all nodes in the network, when only a very small percentage would need access (given that redundancy would be built in between those nodes that need that access).

[0073] The network topology is shown at a high level in FIG. **1** which represents an embodiment of the main network elements involved in a document exchange at the clinical trial study level. Each node **1** accessing the network **2** and declared to be a member of the network may have a local store **3** and **4** respectively of documents relevant to the workflows and collaborations to which they have been invited. The local document store may be provided by a browser-based HTML **5** document store, or through a larger scale document store dependent on their overall document store requirements. The store (at rest) and all transmissions of data (in flight) on the network may be encrypted. Local clients may only be able to decrypt the data if they are in possession of correct cryptographic keys.

[0074] Each node **1** may have a client application **5** and **6**, respectively, which acts as a listener for documents directed towards that node when the node is acting as a receiving node, and also acts as a compiler and executor of any configuration or code embedded with the document being distributed. When a first, sending, node pushes a document into the network, this code and configuration may determine the high-level destination nodes to which the document should be distributed.

[0075] The document may be sent to all nodes, but only those nodes who need to see it will process it, and these nodes may be referred to as receiving nodes. The FSM software pre-installed on each node may make that determination by analysis of the list of node destinations.

[0076] The client application **5** and client application **6** workflow may execute to unwrap, determine the relevance to the current logged-in user, enable the interaction (sign, edit, acknowledge receipt), and forward the document to the next recipient, as shown in exemplary FIG. **2**.

[0077] As part of the underlying infrastructure, an encryption mechanism is provided. This may be asymmetric. Some specific examples include Elliptic Curve Cryptography (ECC) and Elliptical Curve Digital Signature Algorithm (ECDSA). Both may be used to validate the origin and integrity of messages.

[0078] The basis of signing may be the creation of a public and private key. For example, Ethereum or Bitcoin use the hashed version of the public key to identify an address. The public key may be used to verify and encrypt, and the private key may be used to decrypt.

[0079] Setup

[0080] Not every member of the network needs access to every document in that network. For example, a patient may not need access to study-specific protocols. So, each document may have a high-level routing table (itself encrypted), indicating to the network which high-level network members should receive the distributed document.

[0081] Also, the network may determine elements of a workflow indicating a sequence in which actions need to take place across the network. For example, if a consent form has been distributed to the network and a patient has been included in that distribution, the system may need to control who signs that consent and in which order. The system may prevent an investigator signing until the patient and/or his/her guardian has signed.

[0082] A finite state machine (FSM) may be distributed to each node in the P2P network. This may allow the smart document to access the underlying state from the infrastructure rather than implementing the state machine each time a new document is created by users. The FSM may be an

engine that is installed on each node which will manage the movement of a smart document from state to state. This FSM engine may provide standard services to the underlying infrastructure and the smart document that will utilise the P2P network, thus simplifying the smart documents themselves and allowing a consistent approach to state changes across all smart document types. The FSM engine may maintain the state of the document. The smart document may then an instantiation of an individual state machine which understands, through its own code, the sequence of document-specific valid states for that document as well as providing document-specific features for use by the participants in the P2P network.

[0083] The generic FSM distributed to each node as part of the infrastructure set up might look like the following (as part of the underlying Blockchain or P2P infrastructure):

```
FSM {
    Structure stateFunctions [ ]
    Void updateState( )
        currentState = getCurrentState( )
        Execute (currentState)
    Void removeTopState( )
        stateFunctions.remove( )
    void addState(newState)
        stateFunctions.add(newState)
    function getCurrentState( )
        return stateFunctions.top( )
}
```

[0084] This pseudo code above is an example of a generic state machine, allowing the application domain implement and maintain the allowable set of states and the logical transitions between them. This "application" may be the Smart Document which is distributed to the nodes in a P2P/Blockchain network, given that each node already has the FSM executable code ("brain") in place as part of the infrastructure. Consensus and sequencing can also be leveraged through existing infrastructure in HyperLedger (Burrow and Consensus).

[0085] Smart Document

[0086] A Smart Document is a document that contains a document or data set and code and configuration data to enable collaboration, distribution and certification within an encrypted P2P network.

[0087] An example of a smart document which is capable of executing the workflow shown as part of the architecture and using the node FSM executable code detailed above is detailed below:

```
'Network Configuration
Structure Destinations [ ]
Structure AccessLog [ ]
Structure ACL [ ]
'Document
Structure Payload bytes[ ]
Structure DocInternalStructure[ ]
'Payload Attributes
Version string
Checksum string
'Interaction-Relations
Map InteractionParties bytes[ ][ ]
Boolean EnforceSequence = False
'Instantiation of the document
```

-continued

```
SmartDocument(Particpants string[ ], EnforceSequence Boolean))
    Copy Particpants into InteractionParties[ ][1]
    Create New FSM( ) 'Creating the "brain"
    Initialise the FSM(signPatient)
Void signPatient(Participant string, Signature bytes[ ])
    performPatientSignigOperation( )
    FSM.removeTopState( )
    FSM.addState(signInvestigator)
Void signInvestigator(Participant string, Signature bytes[ ])
    performInvestigatorSignigOperation( )
    FSM.removeTopState( )
    FSM.addState(makeConsentAvailable)
Void makeConsentAvailable( )
    Publish Consent to Persistent Storage
    Notify InteractionParties
```

[0088] The pseudo code above is one example of a "Smart Contract" and how one might be used with the FSM to enable a distributed workflow within the network. For simple interaction workflows, each state in the Smart Document may be known beforehand, including the participating parties. These could be effectively "hard coded" in the document as the document is created and deployed onto the network. In other cases, the document could be generic and created once. Then the document could be deployed several times, with each deployment potentially having a different set of participating parties. In either case each state in the Smart Document may be known ahead of its distribution. The FSM states may be included in the control data when the document is created and when each instance of that document is subsequently deployed. The list of participating parties may, and probably will, change each time the document is deployed.

[0089] For example, in the pseudo code shown above, the "performPatientSigningOperation" would check that the node and participant match and can sign the document, based on the current state of the document itself. Further, the "performInvestigatorSigningOperation" would most likely also check that the node and participant match and can sign the document, based on the current state of the document itself—in this case the state of the document must be that the patient has already signed it. In other words the investigator, in this example, could not perform the signing operation until after the patient has signed it.

[0090] In the case where participating parties are not known ahead of time, arrays of Participants would be used as a lookup to determine based on the specific scenario (during the individual deployment of the document at run time). Further flexible granularity may also be required where interaction with specific parts of a Smart Document may be controlled, which will require metadata about the document and its constituent parts. This more complete structure is shown in exemplary FIG. **2**.

[0091] As shown in FIG. **2** a Smart Document **51** includes the following types of data:

[0092] Network configuration may include elements such as the destinations relevant to this document and some optimisation settings relating to network load and getting the documents distributed in the most efficient way (**1**). Some of this configuration may be used by the underlying P2P network while others may be used by the Smart Document itself

[0093] Access Control List may allow the specification of levels of Authorisation for recipients of the document. The highest level of security may access the

complete distributed ledger. Others having less access may only be permitted see what they need to see in the ledger. Thus, the ledger may become secured.

[0094] Access Log may record who has read the document, opened, etc. as part of the shared ledger. This may embed the concept of knowing, not alone what machine has received the document, but who has also read it within the ledger.

[0095] The Document content—which may be a PDF, document, image, or other form of media

[0096] Payload attributes may, for example, include version numbers, checksums, priorities, and other elements which can be used by the application user interface and underlying architecture to add value to the business processes built on the document.

[0097] Signature-Relations may include settings relating to the map of signatures and other elements relating to workflow where the sequence of signatures/verification is to be enforced.

[0098] DocInternalStructure may include semantic definitions of the internal sections within the document. RDF and XML/JSON are representations which could be used. One use of this internal structure definition may be finer grained control of access right as defined in the RightsMap under the AccessControlList. This may allow finer grained and a deeper level of control, thus allowing a user to sign in one section, but make no other alterations, or to allow an update to a specific part of the document but no other. Allowing the document structure to be machine readable may enable intelligent document analysis at run time to determine any shifts in access control requirements over its lifetime, to the point of workflow mutations that could be needed depending on ACL-to-semantic mappings.

[0099] Instantiation. Each individual deployed version document and document intelligence may include self-contained explicit code and configuration elements which are compiled and executed by the underlying P2P client software made available either through a light client, browser add-on, or installed software applications. This could be implemented via JavaScript and Node.js, a code generation tool, or a configuration tool, or directly on Blockchain (**2**)/Ethereum (**3**). The instantiation may refer to the specific deployment of an instance of the document type into the network. For example, an eConsent document may be designed for a trial (may be regarded as a template). Then that eConsent document may be deployed multiple times—for each combination of investigator/patient and that is what actually is deployed and is signed by each participating party.

[0100] Regarding access control by authorized users, the document structure may be machine readable to enable intelligent document analysis at run time to determine any shifts in access control requirements over its lifetime. Access control may change over the lifetime of a document. For example, there may be three states that the document could be in and they are sequentially A, B, C and two authorized users (User**1** and User**2**) who have read/write access at the start. If User **1** acts on the document and moves it from State A to State B, then it could occur that User **1** loses write access. Once User **2** moves the state from State B to State C, then it could arise that User **1** regains write access. These states and access rights could be predetermined, or there could be a more complex sequences of states which, when reached in a specified amount of time or set of states results in different access right being granted to specific users.

[0101] Referring again to FIG. **3**, the following are the main steps of a method **50** performed by a client application at a node upon receipt of a smart document **51**:

[0102] Decrypt document and verify checksum, **52**, **53**, respectively.

[0103] User on Access Control List (ACL)? (**54**). The application determines if the user has the right to perform this interaction on this part of the document (or the document as a whole)

[0104] Extract payload, **55**.

[0105] Extract metadata, **56**.

[0106] Map Sections to ACL, **57**.

[0107] Enforce Sequence (**58**) using the embedded FSM executable code in the network node **1**, **2** in conjunction with the specific Smart Document **51** which includes information as to where it should be routed and what the states mean to the document in terms of the domain requirements, notifications on completion, persistence, and any other relevant activities

[0108] Current user next to interact, **59**. This can be defined as the next actor in a workflow which is determined either through a "hard coded" FSM or one that relies on a more extensible and flexible map of Interaction Parties

[0109] Enable interaction, **60**.

[0110] Record interaction, **61**.

[0111] Encrypt, **62**.

[0112] Forward to next interaction party, **63** and end **64**.**0**

[0113] An example of the DocInternalStructure is laid out below in RDF as an exemplar representation model:

```
<rdf:Description
Rdf:about="http://www.sourcelocation.com/important/DocToSign">
  <document:author>Dr. Principal Investogator</ document:author >
  <document:source>{7892341n-nkmgdfshj-453782p5}</
  document:author >
  <document:sections>
    <rdf:bag>
      <rdf:li rdf:id="123">Understanding a Clinical Trial</rdf:li>
      <rdf:li rdf:id="124">Risks in this Clinical Trial</rdf:li>
      <rdf:li rdf:id="125">Steps in this Clinical Trial</rdf:li>
    </rdf:bag>
  <document:sections>
</rdf:Description>
```

[0114] The above document has three sections with ids of "123", "124", and "125" respectively An example of a RightsMap to be used in conjunction with the DocInternalStructure is laid out below:

```
@prefix acl: <http://www.w3.org/ns/auth/acl#> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
<acl:accessTo <DocToSign>; acl:mode acl:Write; acl:agentClass
foaf:PrincipalInvestigator>
<acl:accessTo <DocToSign>; acl:mode acl:Control; acl:agent
<PrincipalInvestigator#999>>
<acl:accessTo <DocToSign>; acl: mode acl: Read;
acl:agentClass foaf:StudyNurse>
<acl:accessTo <DocToSign#123>; acl: mode acl: Sign;
```

-continued

```
acl:agent<Patient#11>>
<acl:accessTo <DocToSign#124>; acl: mode acl: Sign;
acl:agent<Patient#11>>
<acl:accessTo <DocToSign#125>; acl: mode acl: Sign;
acl:agent<Patient#11>>
```

[0115] The above ACL representation shows that any PrincipalInvestigator can write to the document, PrincipalInvestigator#999 can control who has access to the document, and any StudyNurse can read the document. Patient11 can sign the three sections but cannot modify the document.

[0116] It will be appreciated that the network may distribute documents to remote consumers, and that it may provide a mechanism to manage protocol and SOP distribution. It may also provide for collaborative editing and updating of joint documents. All edits and updates to the document may be tracked and controlled through the use of the distributed audit trail, while guaranteeing that only those who have access rights to the relevant sections (by virtue of authority or capability) can interact, and while the next step the process can only be executed in sequence with the steps before and after, all may combine to give a self-contained, secure, distributed workflow engine, enacted by the document definition, which may provide for local access to data in a controlled manner.

[0117] Also, the network may provide a means to allow electronic informed consent comply with regulations where local copies of documents must be held at site in some jurisdictions. It can be used to determine the nodes and the people who have access and/or updated the document in any way. The following are example applications of the method and network of the present disclosure.

[0118] Example Application of the Technology: Process Management of a Clinical Trial

[0119] An Investigator at a clinical trial site may be dependent on getting the results of blood draws or biopsies before enrolling a patient into a trial to ensure that the patient meets the inclusion/exclusion criteria. The method described above helps to ensure that the Investigator has reviewed the report results from the screening laboratory.

[0120] The method provides, in one example application, for sending multiple thousands of documents to regulatory authorities each day; with confirmation that documents have been received/opened by the authorities and thereby triggering exception reporting if a document has not been received or opened.

[0121] Virtual Trials:

[0122] Ensuring and validating the origin of data coming from patients—for example remote signature, biometrics, or measurements taken at home

[0123] The exemplary method and network allow improved decentralised data storage in a more general sense, enabling better adherence to privacy and GDPR requirements. The data stays where it is supposed to stay—and users can determine directly who has access to selected data in the relevant context of a process such as a clinical trial.

[0124] The present disclosure is not limited to the embodiments described but may be varied in construction and detail.

1. A method of managing distribution of a document and access to its contents with security in a network comprising nodes with digital processors for remote communication over a wide area network and local data stores,

the method being implemented by said processors and comprising steps of:

a node, acting as a sending node, sending a document in encrypted format into the network and said document including:

a payload with document content, and

control data including destination data, an access control rights map, and

signature data, and a document destination list, and

finite state machine (FSM) executable software code for managing document state,

a node, acting as a receiving node and executing a software client application, receiving the document, decrypting at least some of it, and processing the document according to said control data by extracting the control data and managing document payload access and document storage and user access according to the control data, and in which:

said software client application acts as a listener for documents directed towards that receiving node, and also acts as a compiler and executor of the control data embedded within the received document,

the receiving node software client application processes the control data in a manner transparently to authorised users, and

the receiving node executes the finite state machine FSM executable software code to allow it and the document to access a current document state, and to enable the document to move through a document-specific set of allowable states.

2. The method as claimed in claim **1**, wherein the receiving node extracts the payload and outputs data to authorised users, by mapping sections of the payload to an access control list ACL, in which each authorized user participant has access to some or all of the document.

3. The method as claimed in claim **1**, wherein the receiving node stores in an access log of the document a record of authorised user access to the payload.

4. The method as claimed in claim **1**, wherein the control data includes mapped document sections including:

an interested parties section defining authorised users,

an access control section defining, for each authorised user, a user identifier and a rights map governing access to the document internal structure,

an access log section recording historical user access data including identifier of said user and time and nature of each access.

5. The method as claimed in claim **1**, wherein the nodes perform encryption and decryption using public and private keys in which the public key is used to verify and encrypt and the private key is used to decrypt.

6. The method as claimed in claim **1**, wherein the document includes one or more of:

network configuration data including destination nodes and optimisation settings relating to network load for distributing the document in an efficient manner,

an access control list specifying levels of authorisation for recipients of the document,

an access log which records access to the document, both machine access and human user access,

payload attributes including one or more of version numbers, checksums, and priorities, and

signature-relations records includes settings relating to a map of signatures and workflow for a sequence of signatures and verifications to be enforced.

7. The method as claimed in claim **1**, wherein the document includes executable code for dynamic instantiation of the document by the sending node.

8. The method as claimed in claim **1**, wherein the document is processed by the receiving node as an instantiation of the FSM which understands, through its executable code, a sequence of document-specific allowable states for that document and provides document-specific features for use by the authorized users.

9. The method as claimed in claim **1**, wherein the receiving node determines a next user according to a next actor in a workflow which is determined according to a map within the document, and said map is aligned to the current state of the document as held in the FSM code to ensure that the document goes to the next agreed-upon step in the process.

10. The method as claimed in claim **1**, wherein the receiving node determines a next user according to a next actor in a workflow which is determined according to a map within the document, and said map is aligned to the current state of the document as held in the FSM code to ensure that the document goes to the next agreed-upon step in the process; and wherein the document payload is sent a plurality of times each with a different set of participating parties, and each FSM state in the document is known ahead of its distribution, and FSM state data is included in the control data when the document is distributed.

11. The method as claimed in claim **1**, comprising the sending node performing a look-up of an array of participants to determine at document run time.

12. The method as claimed in claim **1**, comprising the sending node performing a look-up of an array of participants to determine at document run time; and wherein said look-up provides instructions for a level of granularity of controlling interaction of a receiving node with specific parts of a document, according to meta data about the document and its constituent parts.

13. The method as claimed in claim **1**, wherein the document includes semantic definitions of the sections within the document.

14. The method as claimed in claim **1**, wherein the document includes semantic definitions of the sections within the document; and wherein the semantic definitions are expressed as RDF and/or XML/JSON representations.

15. The method as claimed in claim **1**, wherein the rights map is included in an access control list providing a desired fine grained and deep level of control.

16. The method as claimed in claim **1**, comprising allowing an authorized user to sign in with access to one document section, but make no other alterations, or to allow an update to a specific section of the document but not to another section.

17. The method as claimed in claim **1**, wherein the FSM code and the access control data are dynamically modified to enable intelligent document analysis at run time to determine any shifts in access control requirements over its lifetime, in which the document has M states and N authorized users who have read/write access, an authorized user acts on the document and moves it from a first state to a second state which causes that user to lose write access, but a subsequent state change caused by a second user causes said first user to re-gain write access.

18. The method as claimed in claim **1**, wherein the FSM executable software code manages the movement of the document from one document state to another document state by providing standard services to the underlying receiving node processor and the document allows a consistent approach to state changes across all document types, in which the FSM executable code maintains the state of the document.

19. The method as claimed in claim **1**, wherein updates to the document are tracked and controlled by the nodes in a distributed audit trail, while guaranteeing that only those authorized users who have access rights to relevant document sections can interact.

20. A network comprising a plurality of nodes with digital data processors and storage and being configured to perform a method of claim **1**.

21. A non-transitory computer storage medium comprising software code for implementing a method of claim **1** when executed by digital data processors.

* * * * *