Figure 1

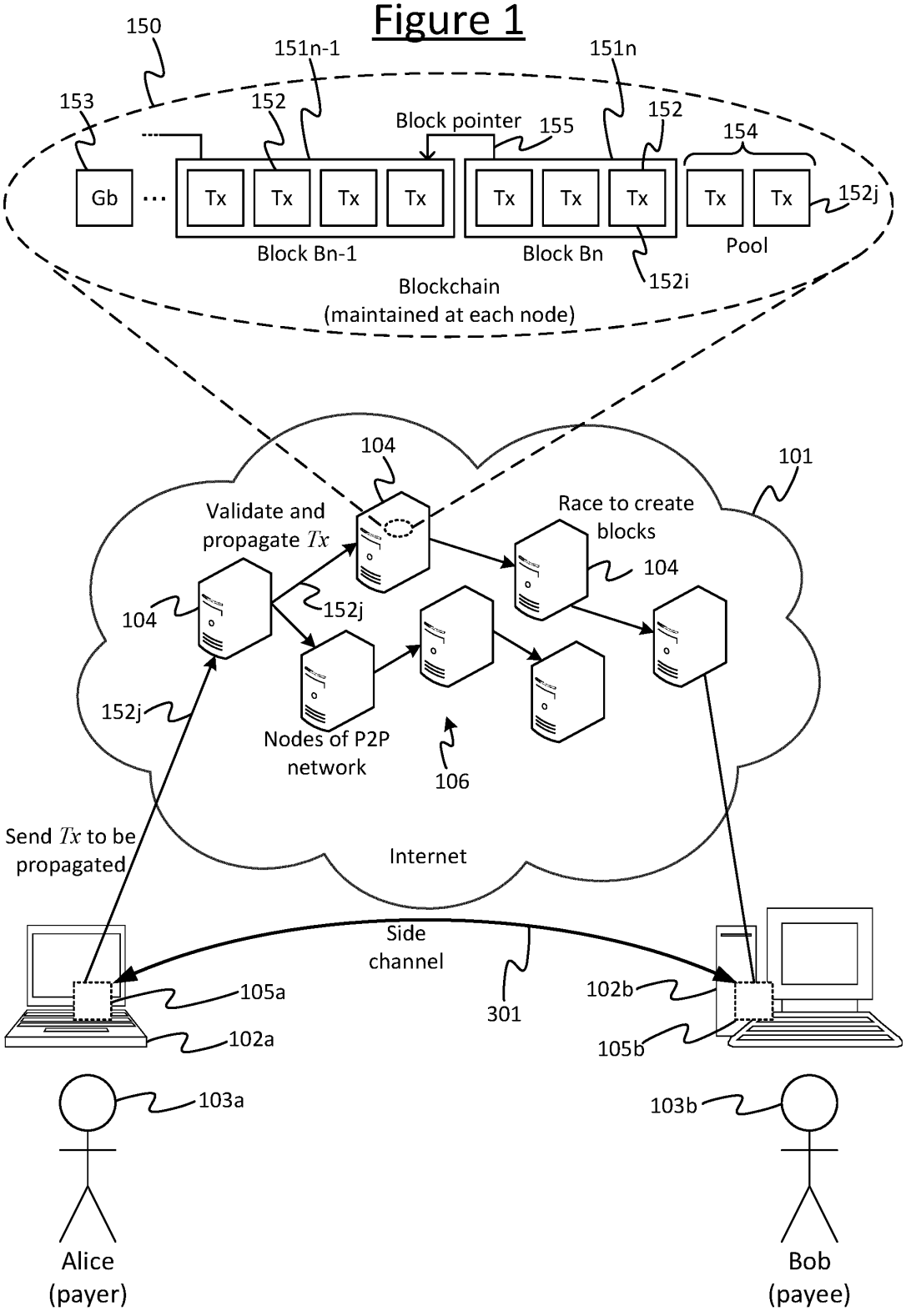
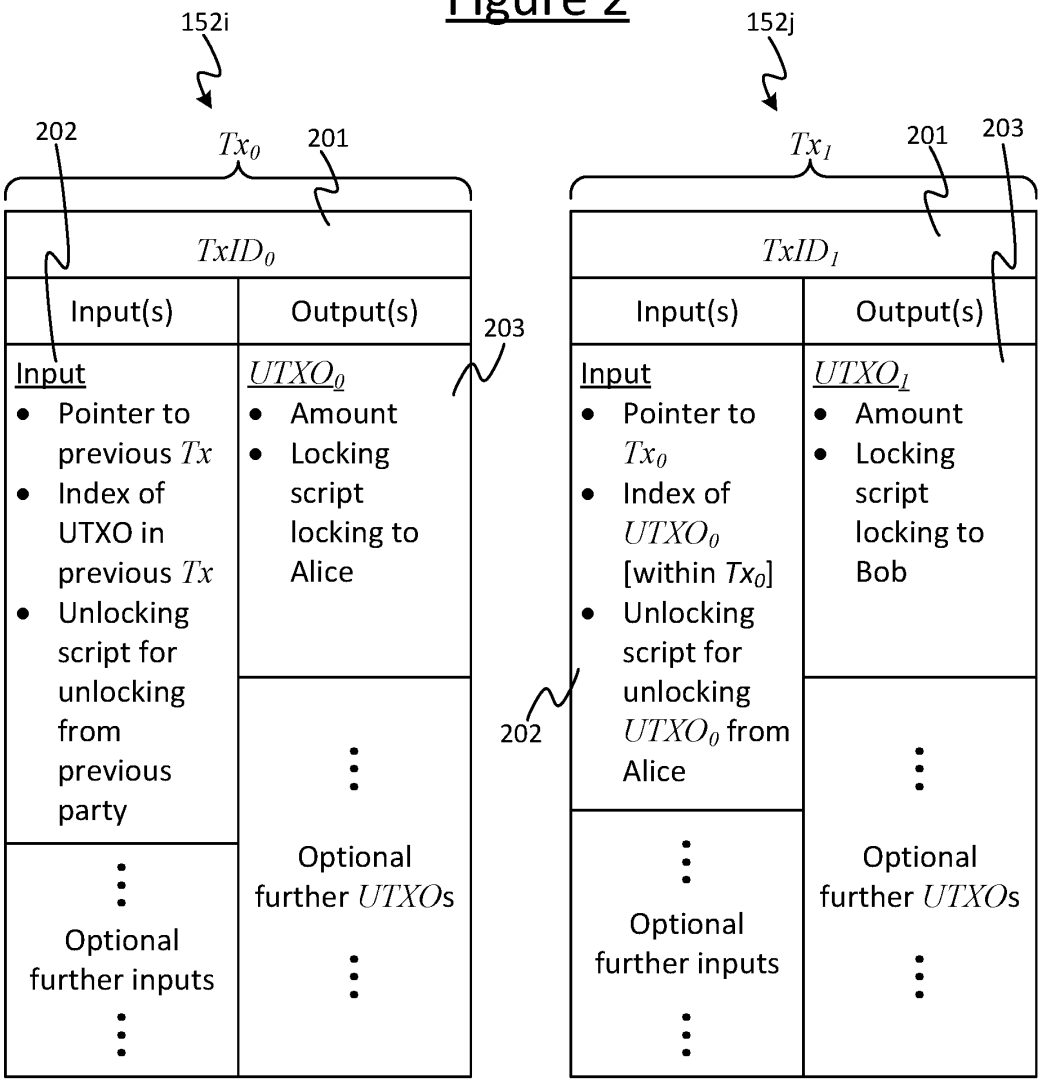

150
151n-1
151n
153
152
Block pointer
155
152
154
Gb
Tx
Tx
Tx
Tx
Tx
Tx
Tx
Tx
Tx
152j
Block Bn-1
Block Bn
Pool
152i
Blockchain
(maintained at each node)
104
101
Validate and
propagate Tx
Race to create
blocks
104
104
152j
152j
Nodes of P2P
network
106
Send Tx to be
propagated
Internet
Side
channel
301
105a
102a
102b
105b
103a
103b
Alice
(payer)
Bob
(payee)

Figure 2



Transaction from Alice to Bob

Validated by running: Alice's locking script (from output of Tx_0), together with Bob's unlocking script (as input to Tx_1). This checks that Bob's unlocking script in Tx_1 meets the condition(s) defined in Alice's locking script in Tx_0.

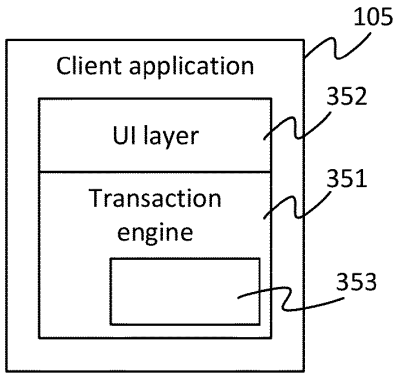


FIG. 3A

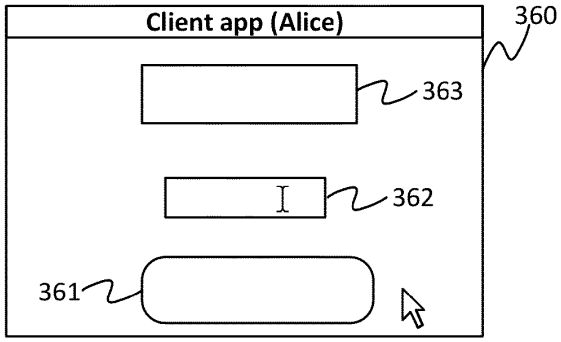


FIG. 3B

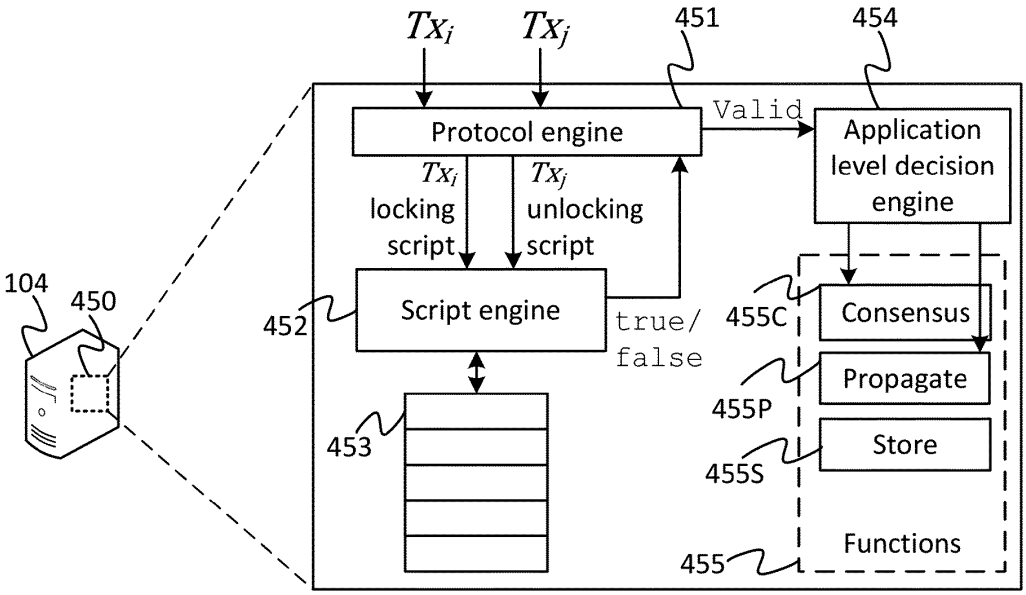


FIG. 4

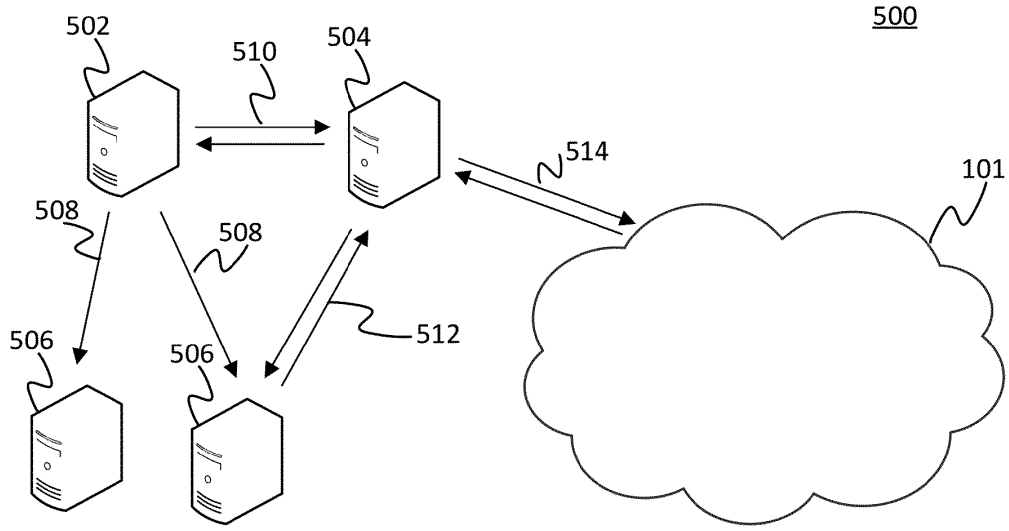


FIG. 5

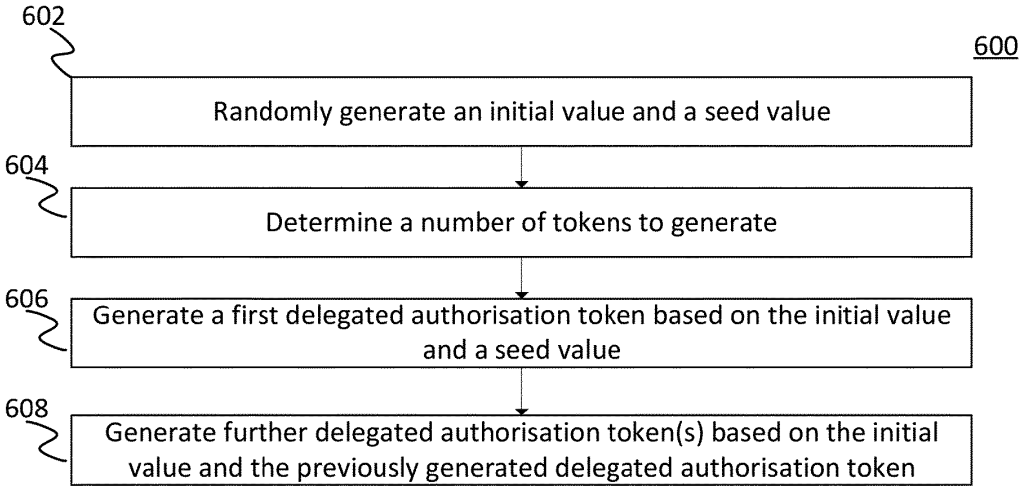


FIG. 6

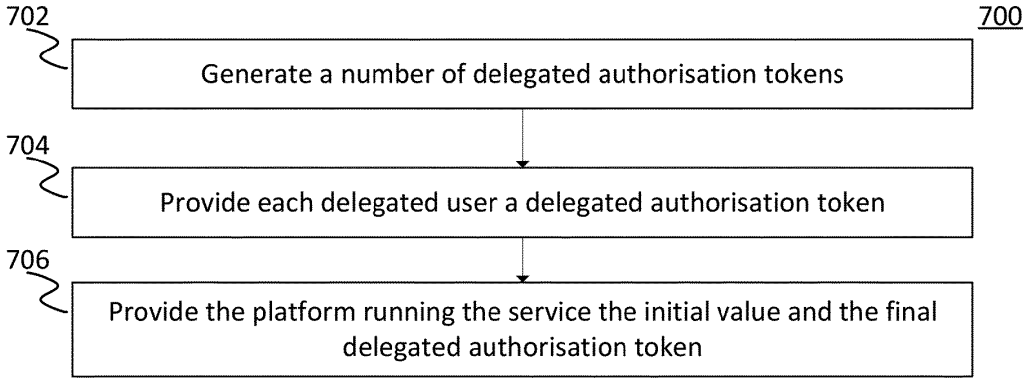


FIG. 7

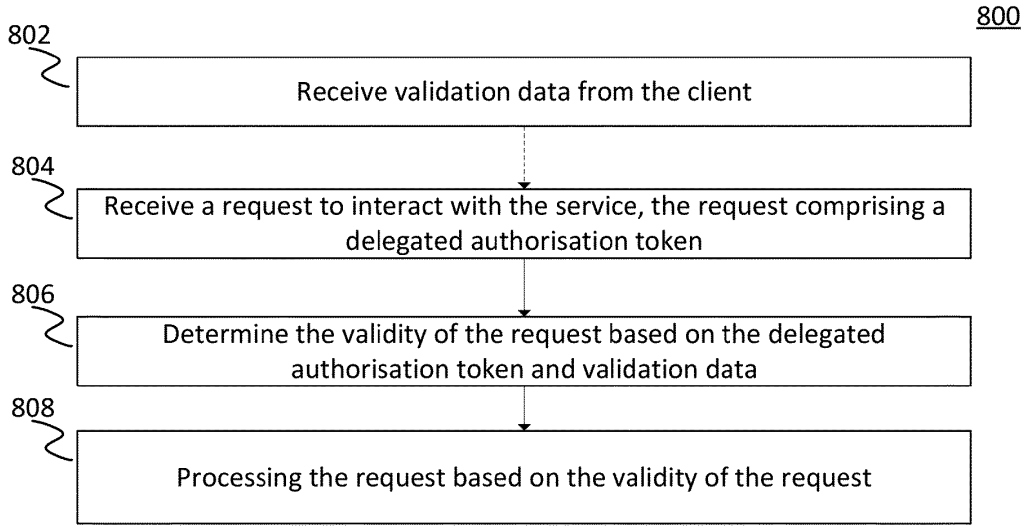


FIG. 8

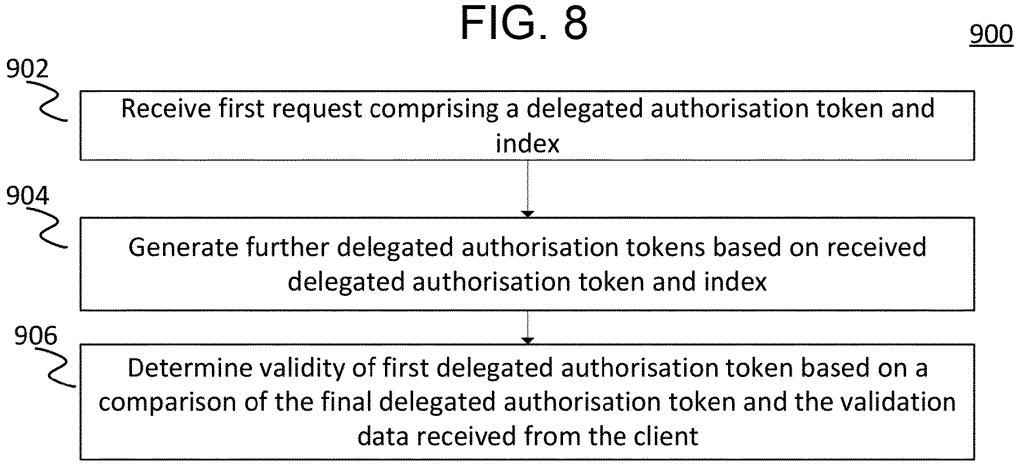


FIG. 9A

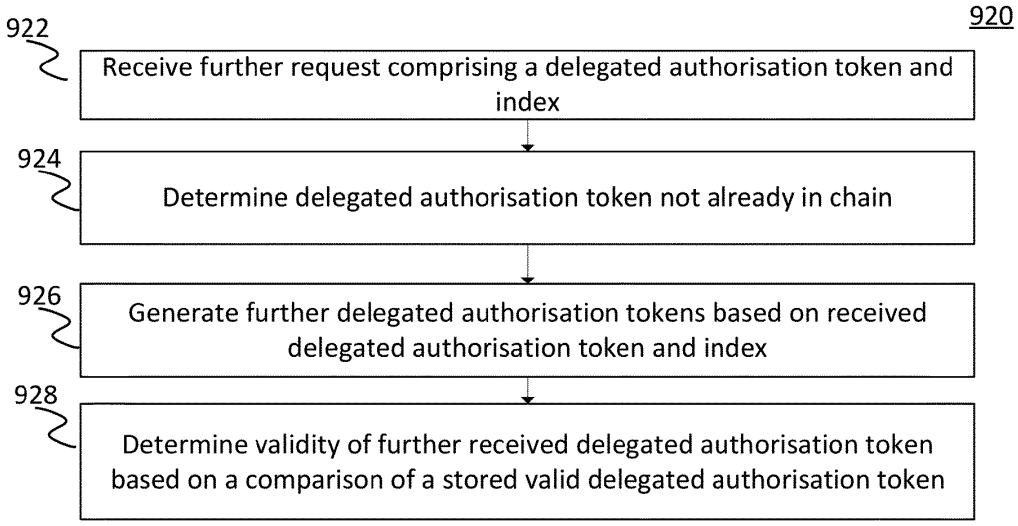


FIG. 9B

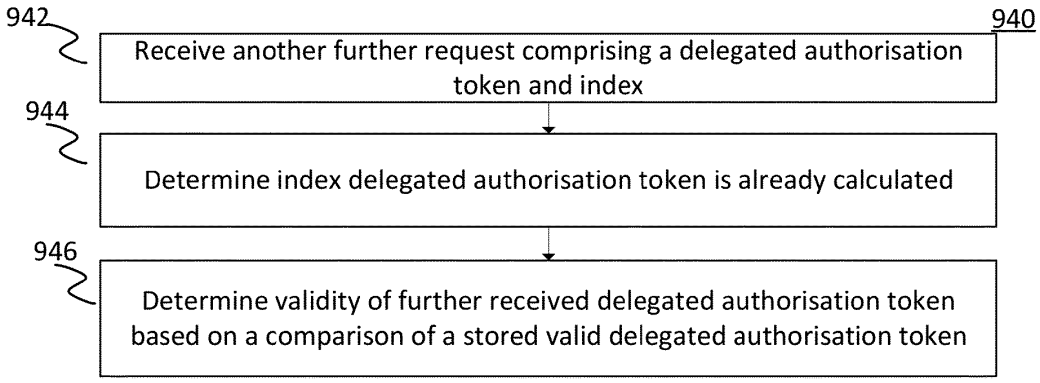

942
940
Receive another further request comprising a delegated authorisation token and index
944
Determine index delegated authorisation token is already calculated
946
Determine validity of further received delegated authorisation token based on a comparison of a stored valid delegated authorisation token

FIG. 9C

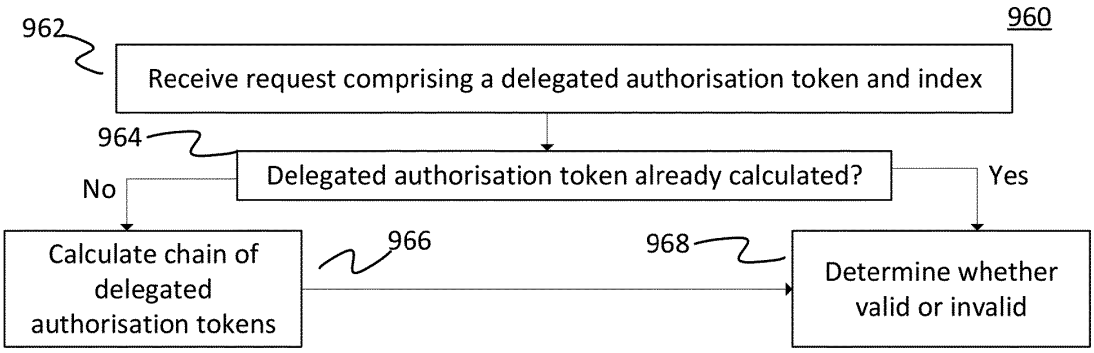

962
960
Receive request comprising a delegated authorisation token and index
964
Delegated authorisation token already calculated?
No
Yes
Calculate chain of delegated authorisation tokens
966
968
Determine whether valid or invalid

FIG. 9D

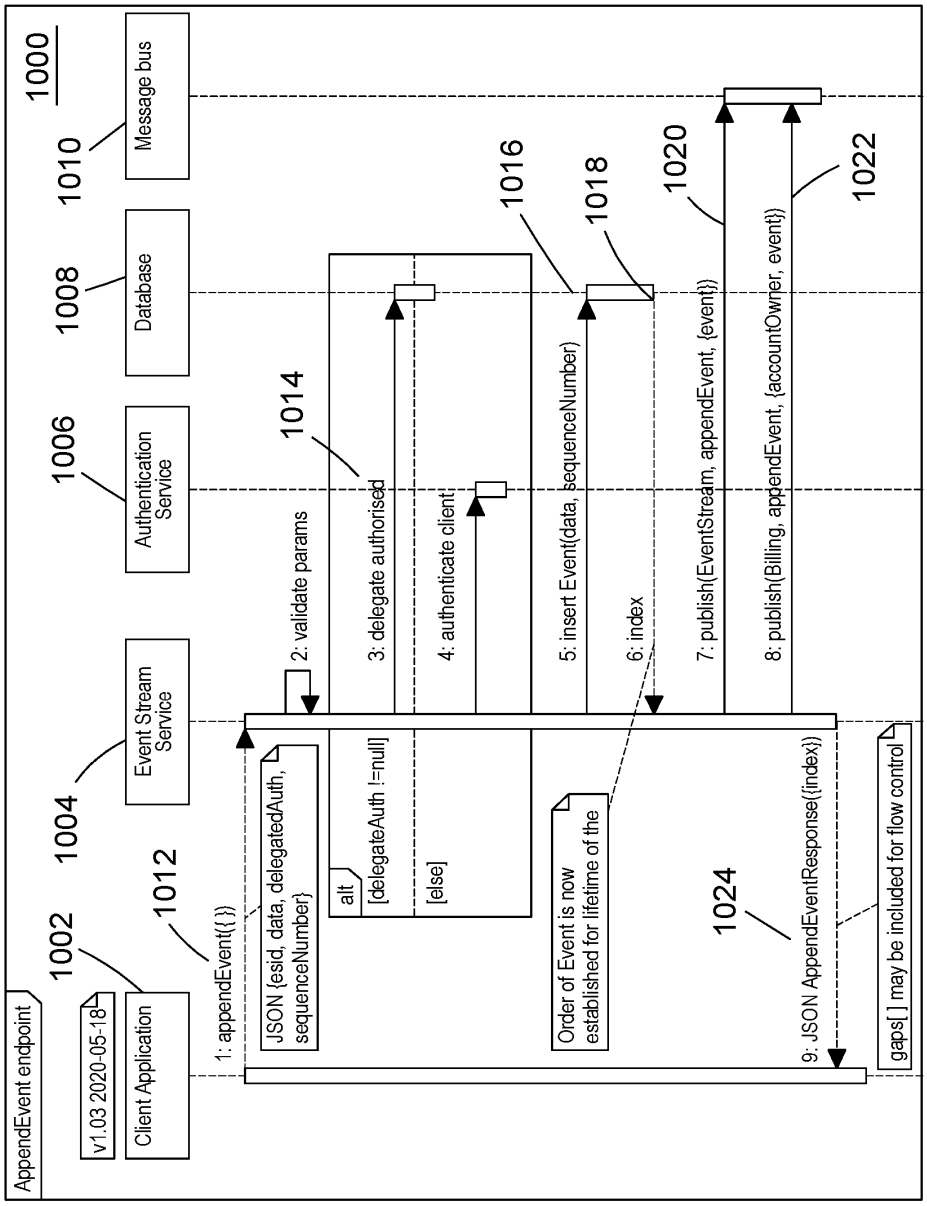

AppendEvent endpoint
1000
v1.03 2020-05-18
Client Application
Event Stream Service
Authentication Service
Database
Message bus
1002
1004
1006
1008
1010
1012
1014
1016
1018
1020
1022
1024
1: appendEvent({})
JSON {esid, data, delegatedAuth, sequenceNumber}
alt
[delegateAuth !=null]
[else]
2: validate params
3: delegate authorised
4: authenticate client
5: insert Event(data, sequenceNumber)
6: index
Order of Event is now established for lifetime of the
7: publish(EventStream, appendEvent, {event})
8: publish(Billing, appendEvent, {accountOwner, event})
9: JSON AppendEventResponse({index})
gaps[ ] may be included for flow control

FIG. 10A

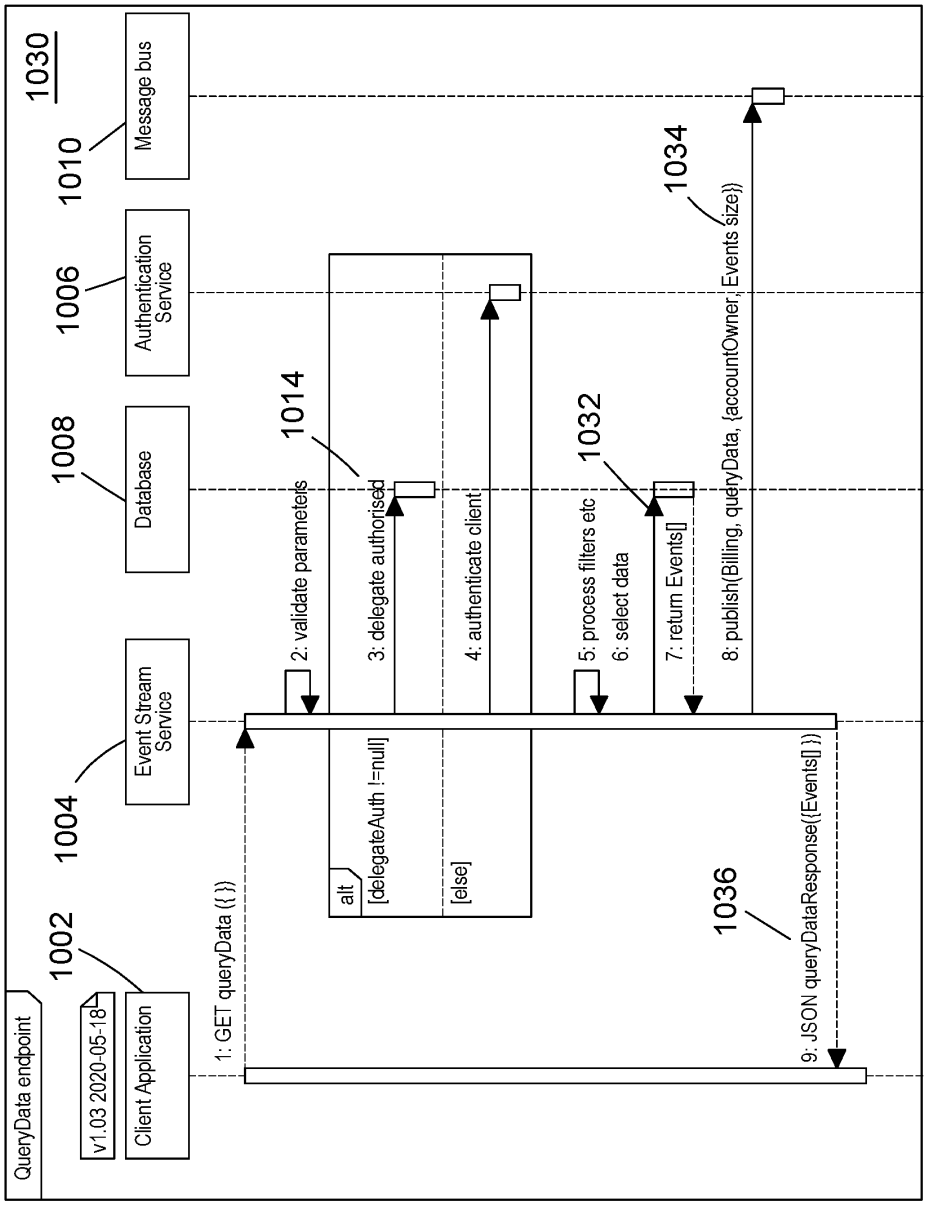

QueryData endpoint
v1.03 2020-05-18
Client Application
Event Stream Service
Database
Authentication Service
Message bus
1002
1004
1008
1006
1010
1030
1: GET queryData ({})
alt
[delegateAuth !=null]
[else]
2: validate parameters
3: delegate authorised
4: authenticate client
5: process filters etc
6: select data
7: return Events[]
8: publish(Billing, queryData, {accountOwner, Events size})
9: JSON queryDataResponse({Events[] })
1014
1032
1034
1036

FIG. 10B

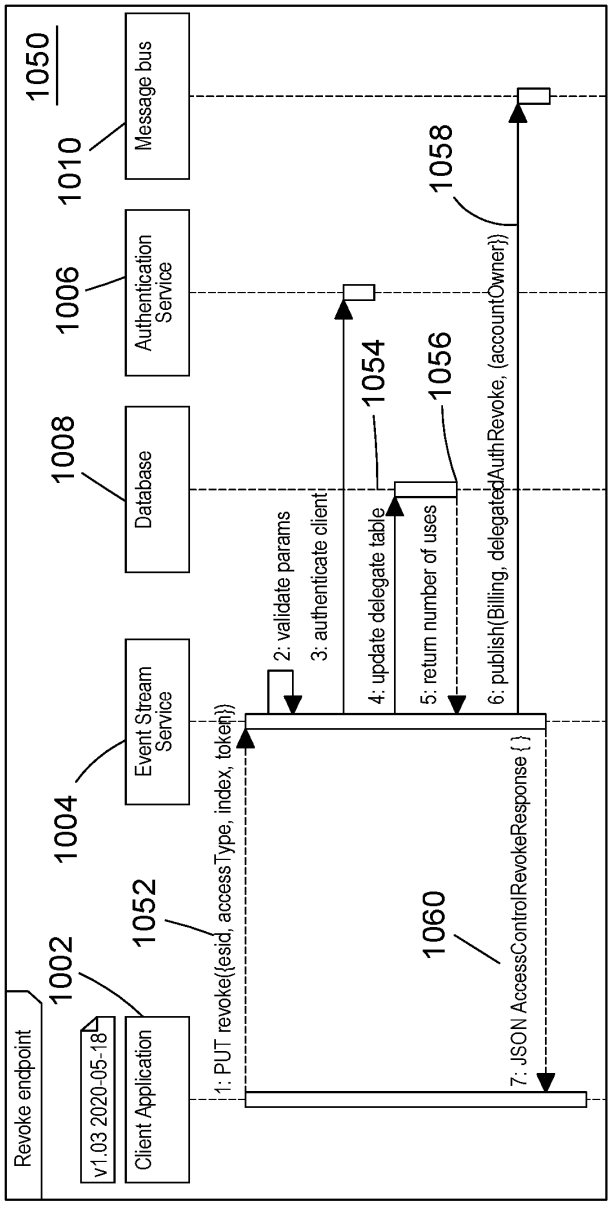

Revoke endpoint
v1.03 2020-05-18
Client Application
Event Stream Service
Database
Authentication Service
Message bus
1050
1002
1004
1008
1006
1010
1052
1054
1056
1058
1060
1: PUT revoke({esid, accessType, index, token})
2: validate params
3: authenticate client
4: update delegate table
5: return number of uses
6: publish(Billing, delegatedAuthRevoke, {accountOwner})
7: JSON AccessControlRevokeResponse { }

FIG. 10C

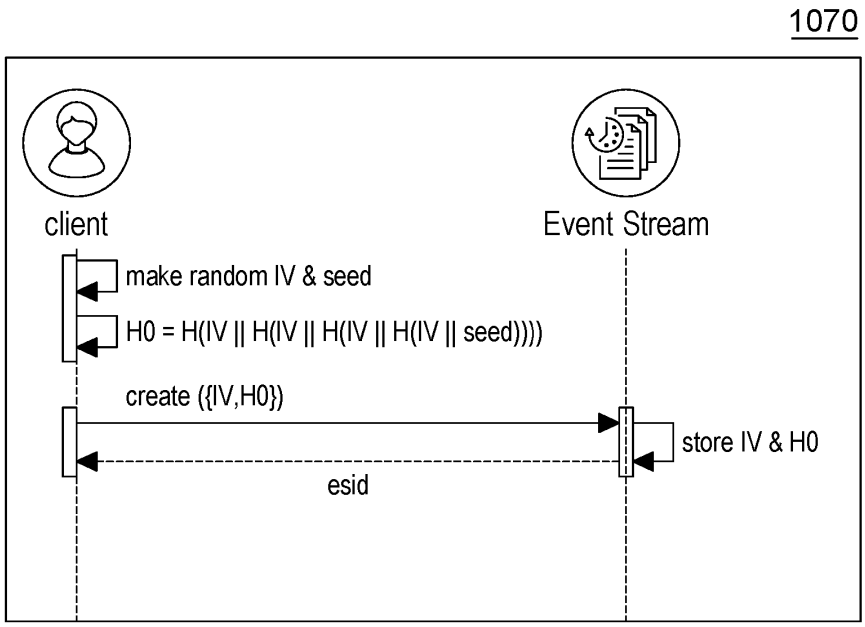


FIG. 10D

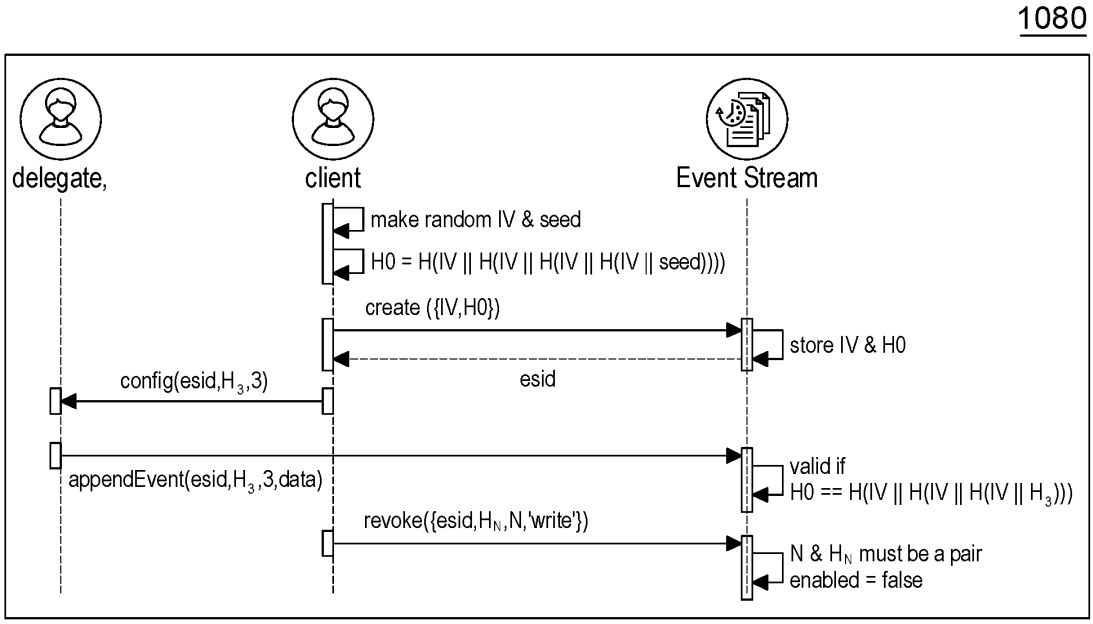


FIG. 10E

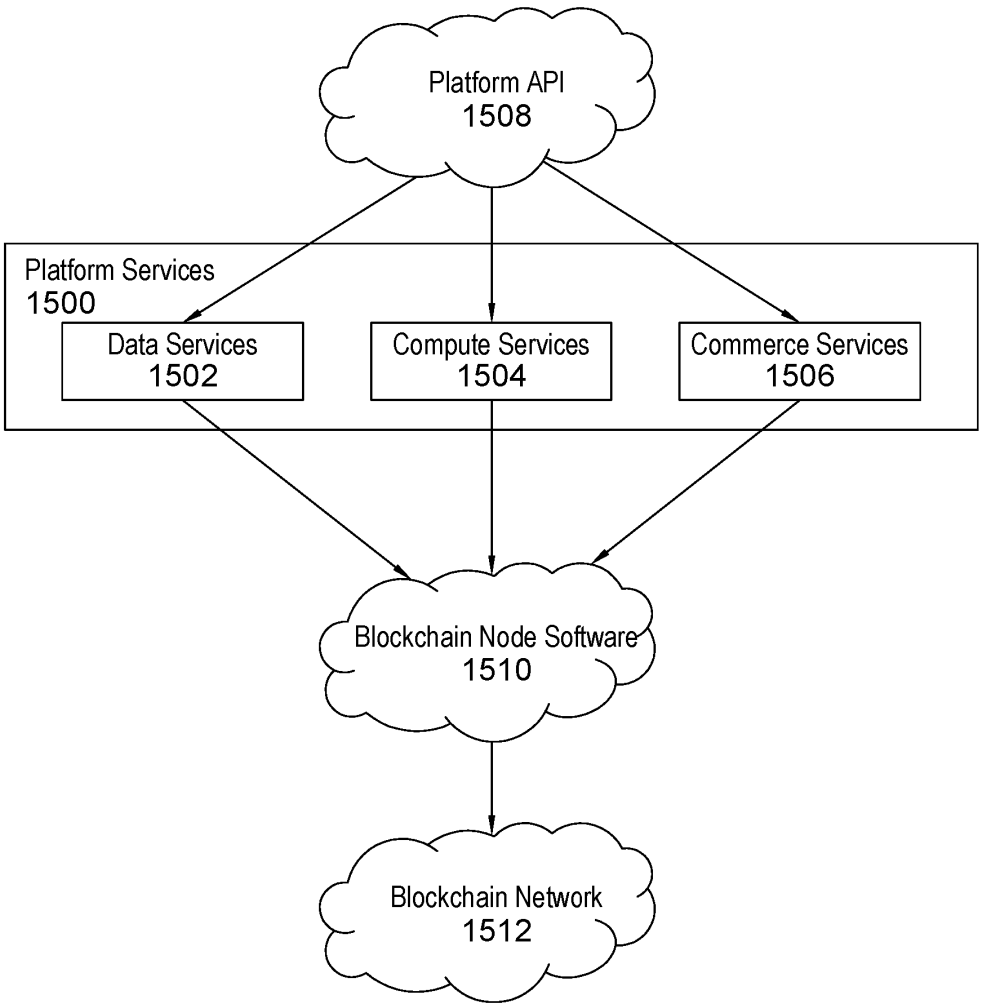

Platform API
1508
Platform Services
1500
Data Services
1502
Compute Services
1504
Commerce Services
1506
Blockchain Node Software
1510
Blockchain Network
1512

FIG. 11

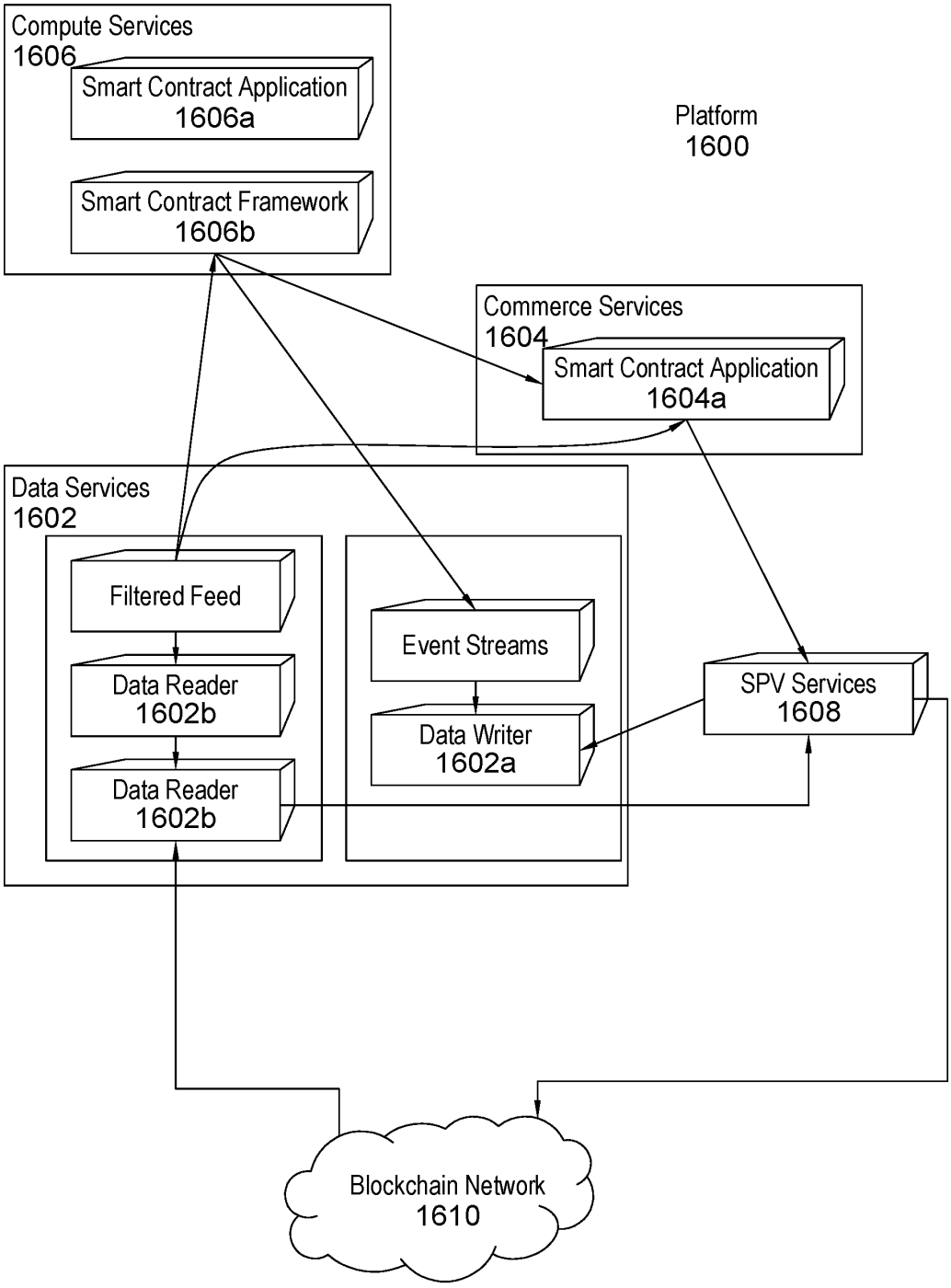

Compute Services
1606
Smart Contract Application
1606a
Smart Contract Framework
1606b
Platform
1600
Commerce Services
1604
Smart Contract Application
1604a
Data Services
1602
Filtered Feed
Data Reader
1602b
Data Reader
1602b
Event Streams
Data Writer
1602a
SPV Services
1608
Blockchain Network
1610

FIG. 12

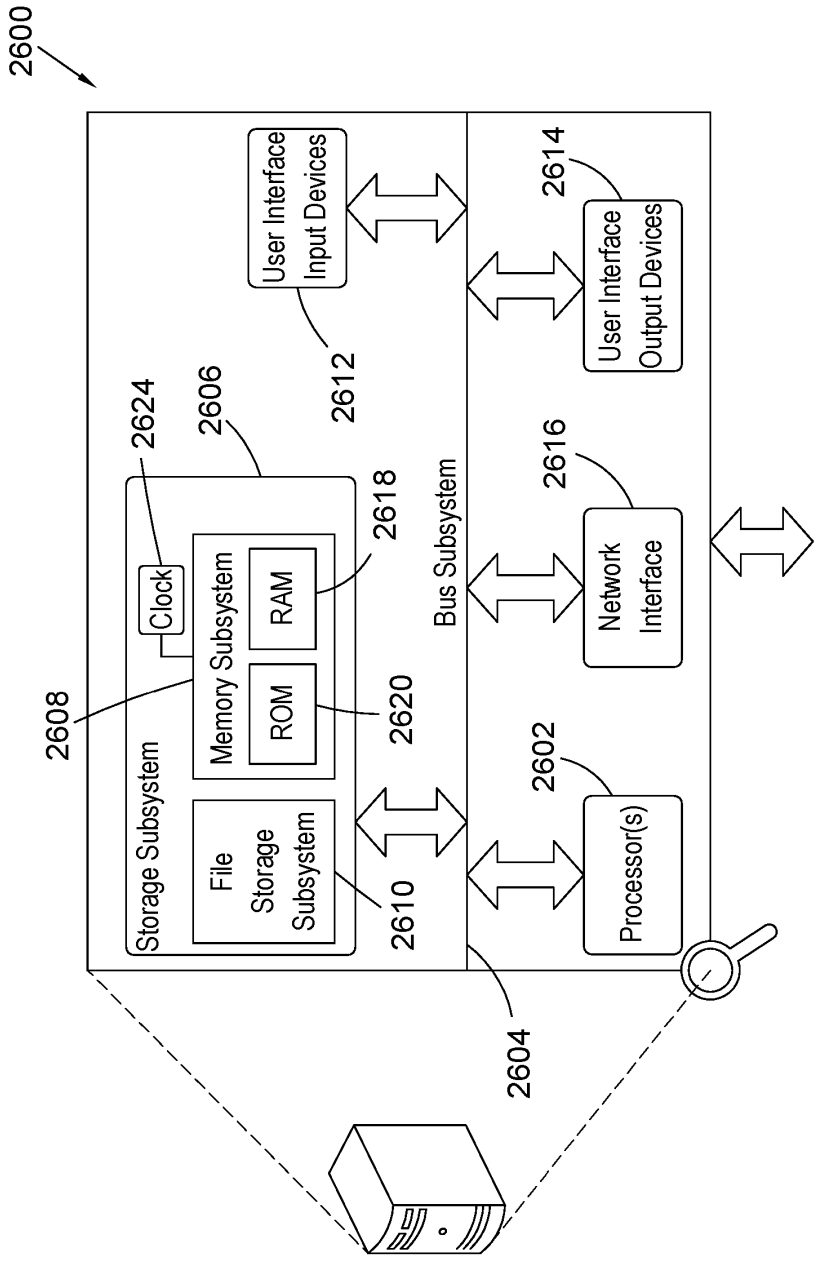

2600
User Interface Input Devices
2612
2614
User Interface Output Devices
2624
2606
2618
Clock
Memory Subsystem
RAM
ROM
2620
2608
Storage Subsystem
File Storage Subsystem
2610
Bus Subsystem
2616
Network Interface
2602
Processor(s)
2604

FIG. 13