US 20160197835A1

(19) **United States**

(12) **Patent Application Publication**

(10) **Pub. No.: US 2016/0197835 A1**

(43) **Pub. Date: Jul. 7, 2016**

Luft

(54) **ARCHITECTURE AND METHOD FOR VIRTUALIZATION OF CLOUD NETWORKING COMPONENTS**

(71) Applicant: **Siegfried Luft**, Vancouver (CA)

(72) Inventor: **Siegfried Luft**, Vancouver (CA)

(21) Appl. No.: **14/588,632**

(22) Filed: **Jan. 2, 2015**

Publication Classification

(51) **Int. Cl.**

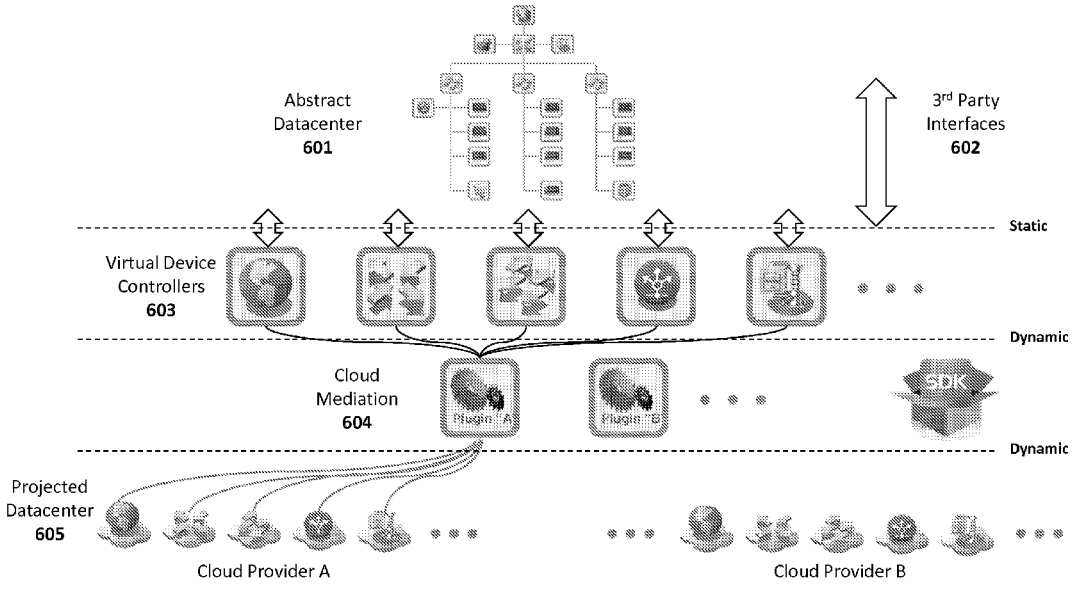
H04L 12/851	(2006.01)
H04L 12/28	(2006.01)
H04L 29/08	(2006.01)
H04L 12/863	(2006.01)
H04L 12/46	(2006.01)

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

CPC ***H04L 47/2425*** (2013.01); ***H04L 47/50*** (2013.01); ***H04L 12/46*** (2013.01); ***H04L 67/10*** (2013.01); ***H04L 12/2854*** (2013.01)

(57) **ABSTRACT**

An architecture and method for traffic engineering between diverse clouds. For example, one embodiment of an apparatus comprises: a virtual device controller to define traffic engineering functions to be performed for communicatively coupling a first cloud provider and a second cloud provider; a mediation layer to map the virtual device controller to a traffic engineering component within the first cloud provider and/or the second cloud provider; and wherein the traffic engineering component comprises a traffic scheduler and a plurality of queues, each queue associated with one or more applications hosted by the first and/or second cloud providers, the traffic scheduler to schedule packets within the queues in accordance with bandwidth and/or latency requirements for each of the applications.



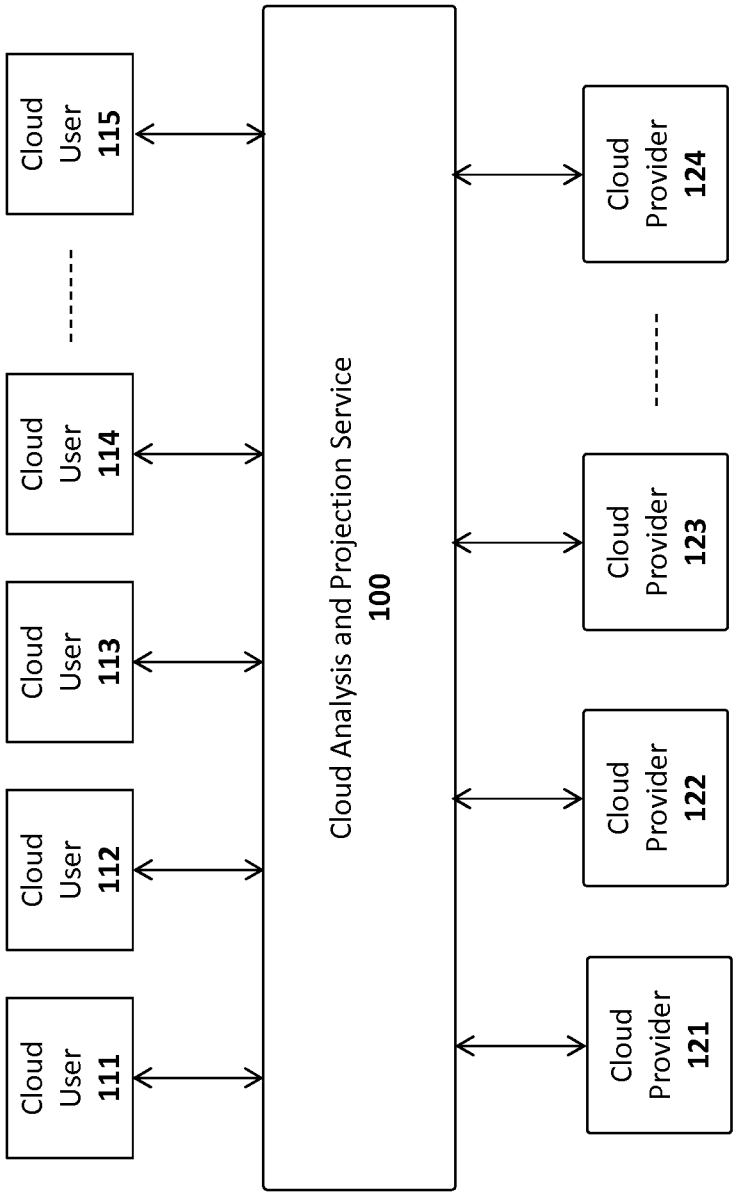

Cloud User 111
Cloud User 112
Cloud User 113
Cloud User 114
Cloud User 115
Cloud Analysis and Projection Service 100
Cloud Provider 121
Cloud Provider 122
Cloud Provider 123
Cloud Provider 124

FIG. 1A

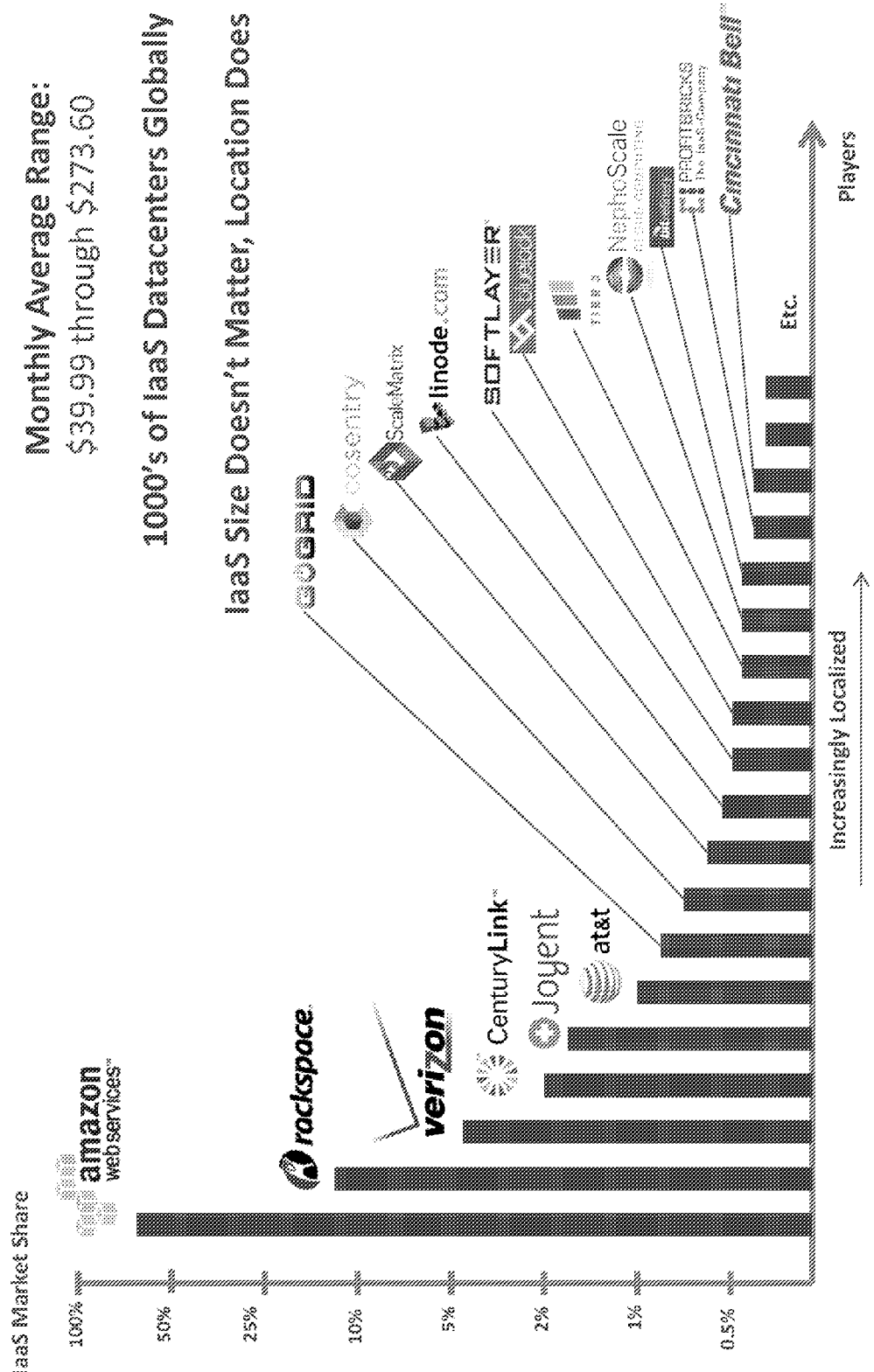

Monthly Average Range:
$39.99 through $273.60
1000's of IaaS Datacenters Globally
IaaS Size Doesn't Matter, Location Does
IaaS Market Share
100%
50%
25%
10%
5%
2%
1%
0.5%
amazon web services
rackspace
verizon
CenturyLink
Joyent
at&t
GoGRID
ServePath
ScaleMatrix
linode.com
SOFTLAYER
NaphoScale
Etc.
Cincinnati Bell
Increasingly Localized
Players

FIG. 1B

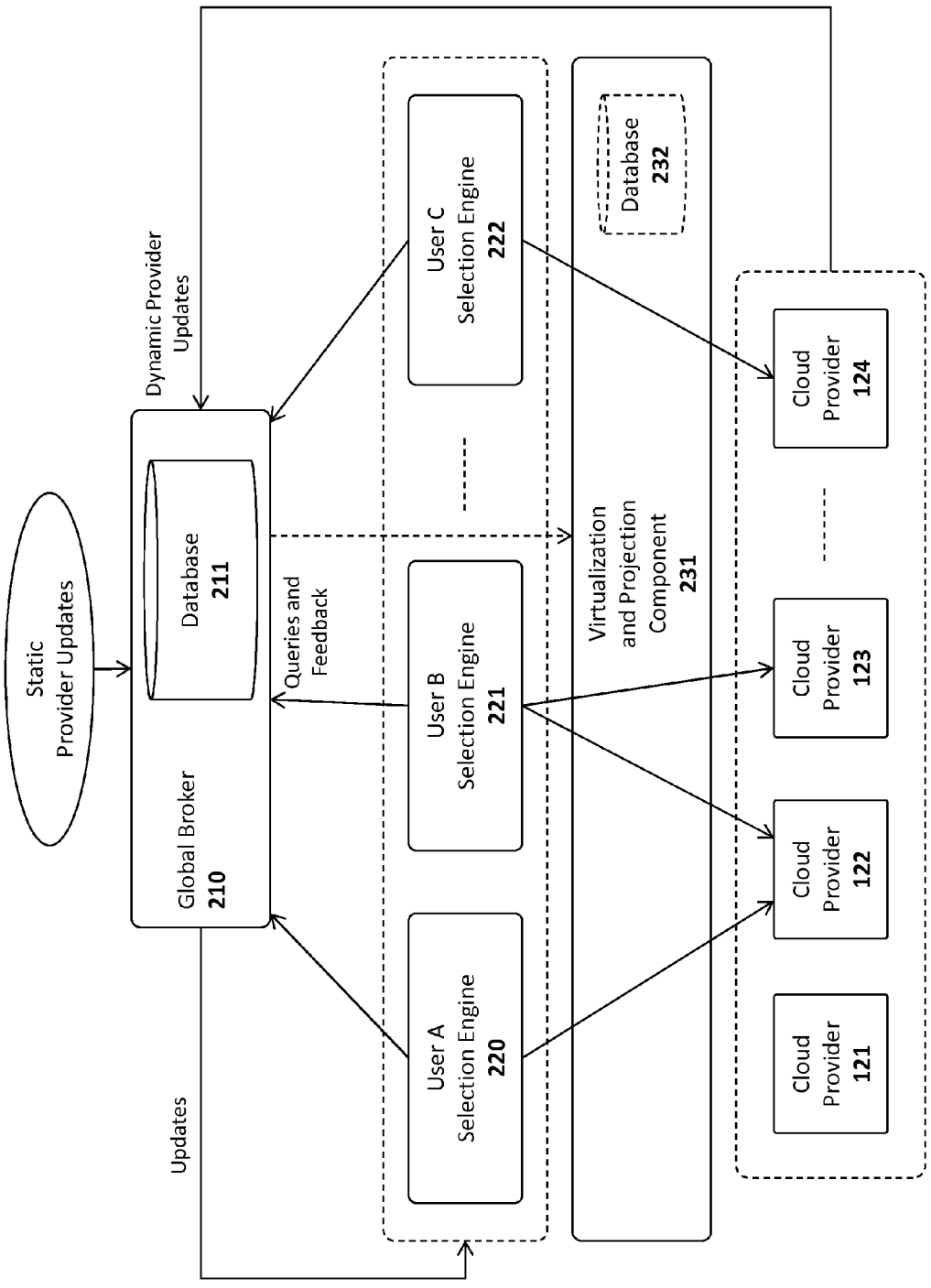

Static Provider Updates
Global Broker 210
Database 211
Dynamic Provider Updates
Queries and Feedback
Updates
User A Selection Engine 220
User B Selection Engine 221
User C Selection Engine 222
Virtualization and Projection Component 231
Database 232
Cloud Provider 121
Cloud Provider 122
Cloud Provider 123
Cloud Provider 124

FIG. 2A

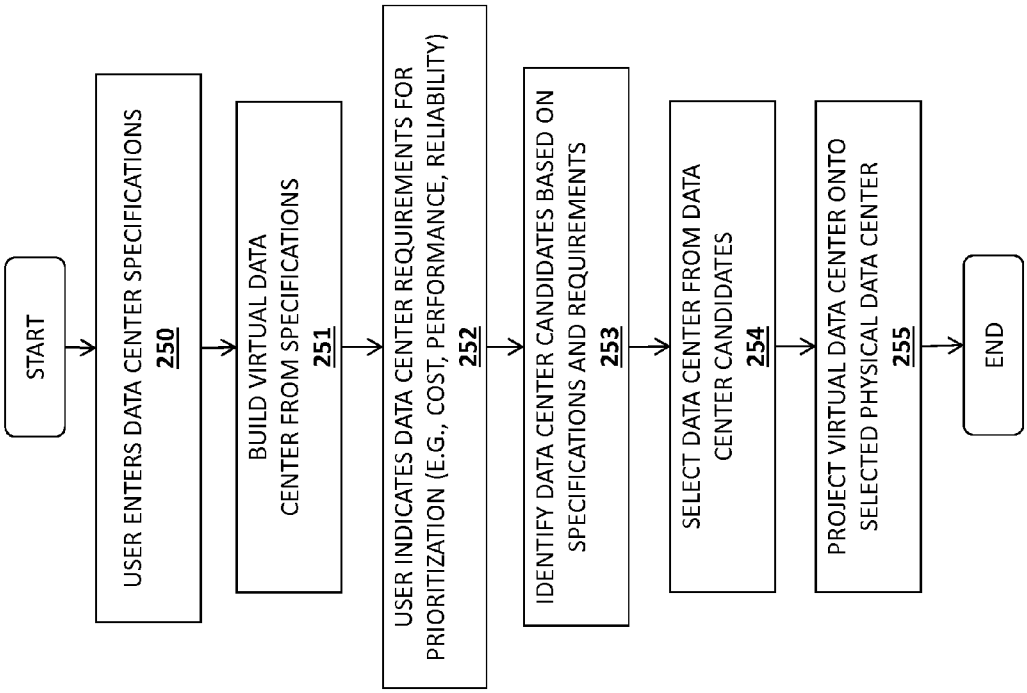

START
USER ENTERS DATA CENTER SPECIFICATIONS
250
BUILD VIRTUAL DATA CENTER FROM SPECIFICATIONS
251
USER INDICATES DATA CENTER REQUIREMENTS FOR PRIORITIZATION (E.G., COST, PERFORMANCE, RELIABILITY)
252
IDENTIFY DATA CENTER CANDIDATES BASED ON SPECIFICATIONS AND REQUIREMENTS
253
SELECT DATA CENTER FROM DATA CENTER CANDIDATES
254
PROJECT VIRTUAL DATA CENTER ONTO SELECTED PHYSICAL DATA CENTER
255
END

FIG. 2B

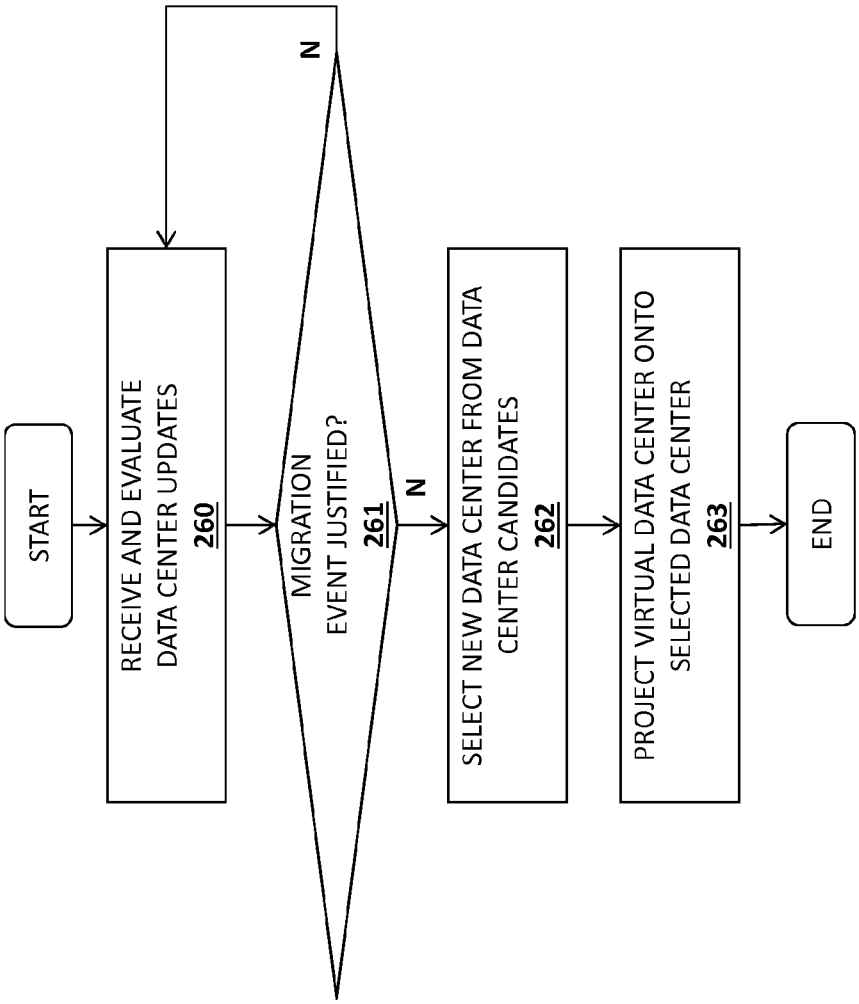

START
RECEIVE AND EVALUATE DATA CENTER UPDATES 260
MIGRATION EVENT JUSTIFIED? 261
N
N
SELECT NEW DATA CENTER FROM DATA CENTER CANDIDATES 262
PROJECT VIRTUAL DATA CENTER ONTO SELECTED DATA CENTER 263
END

FIG. 2C

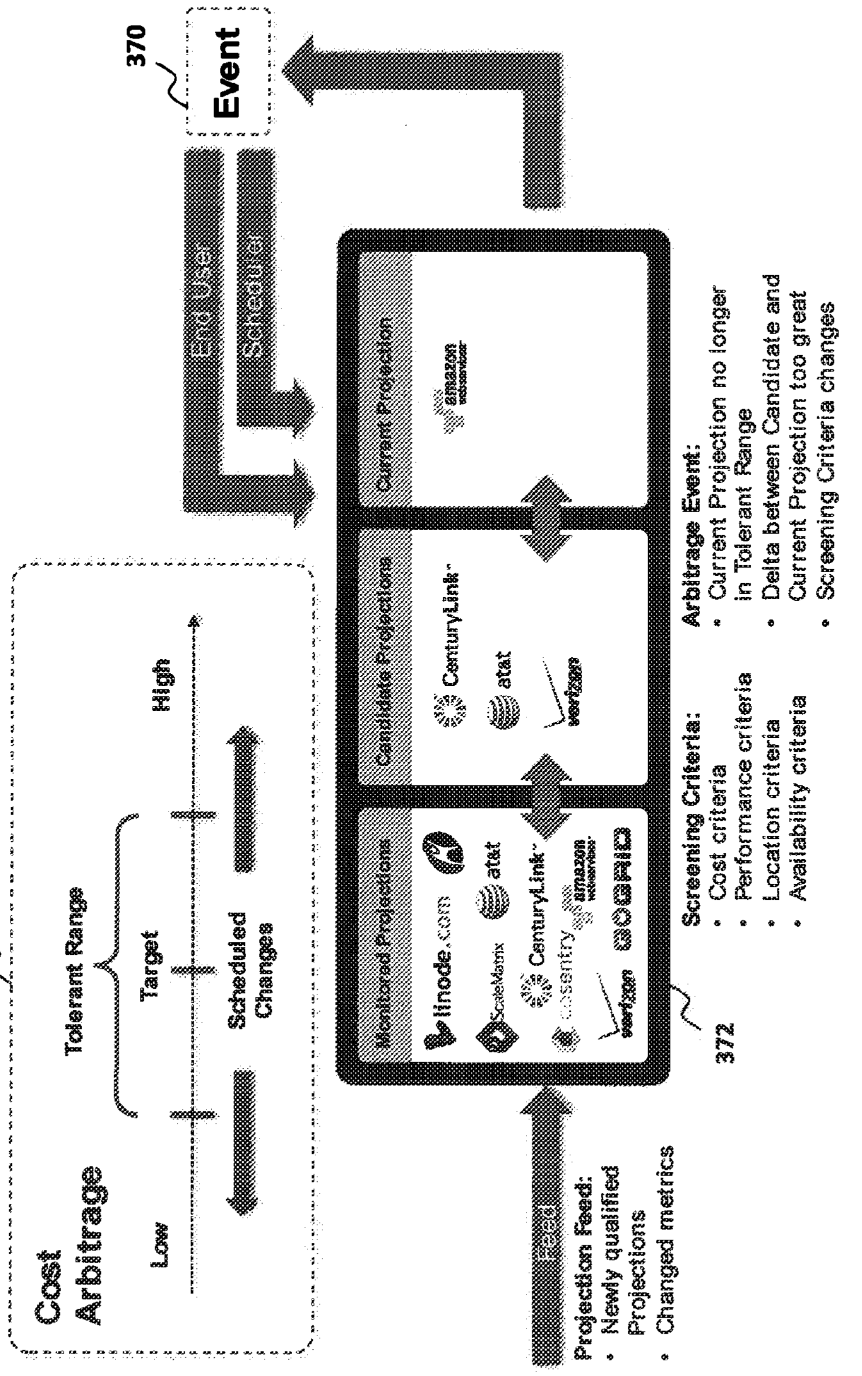

370
Event
End User
Scheduler
371
Cost Arbitrage
Low
High
Target
Tolerant Range
Scheduled Changes
Monitored Projections
linode.com
ScaleMatrix
CenturyLink
at&t
amazon webservices
savvis
GOGRID
verizon
Candidate Projections
Current Projection
372
Feed
Projection Feed:
• Newly qualified Projections
• Changed metrics
Screening Criteria:
• Cost criteria
• Performance criteria
• Location criteria
• Availability criteria
Arbitrage Event:
• Current Projection no longer in Tolerant Range
• Delta between Candidate and Current Projection too great
• Screening Criteria changes

FIG. 3

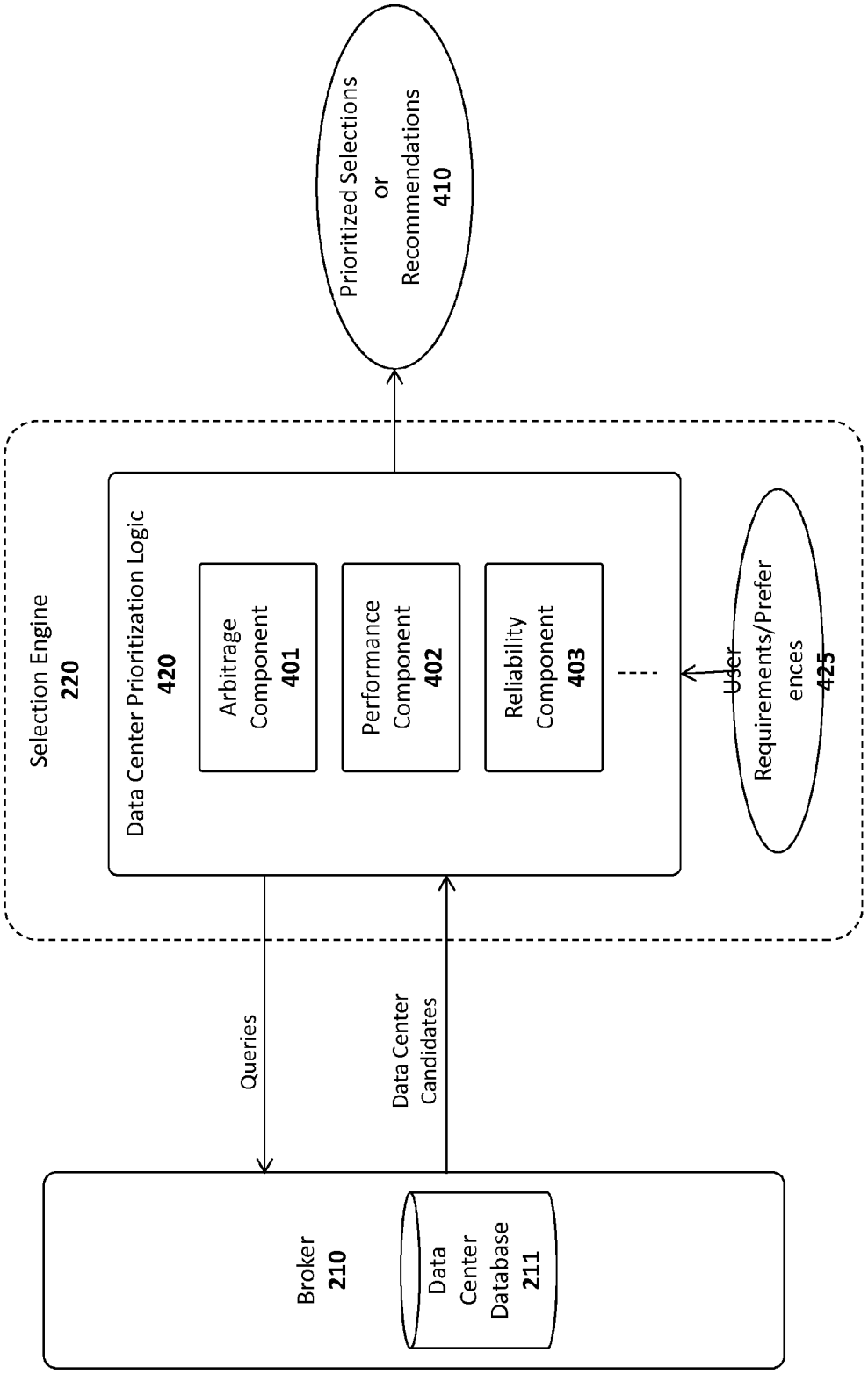

Prioritized Selections or Recommendations 410
Selection Engine 220
Data Center Prioritization Logic 420
Arbitrage Component 401
Performance Component 402
Reliability Component 403
User Requirements/Preferences 425
Queries
Data Center Candidates
Broker 210
Data Center Database 211

FIG. 4

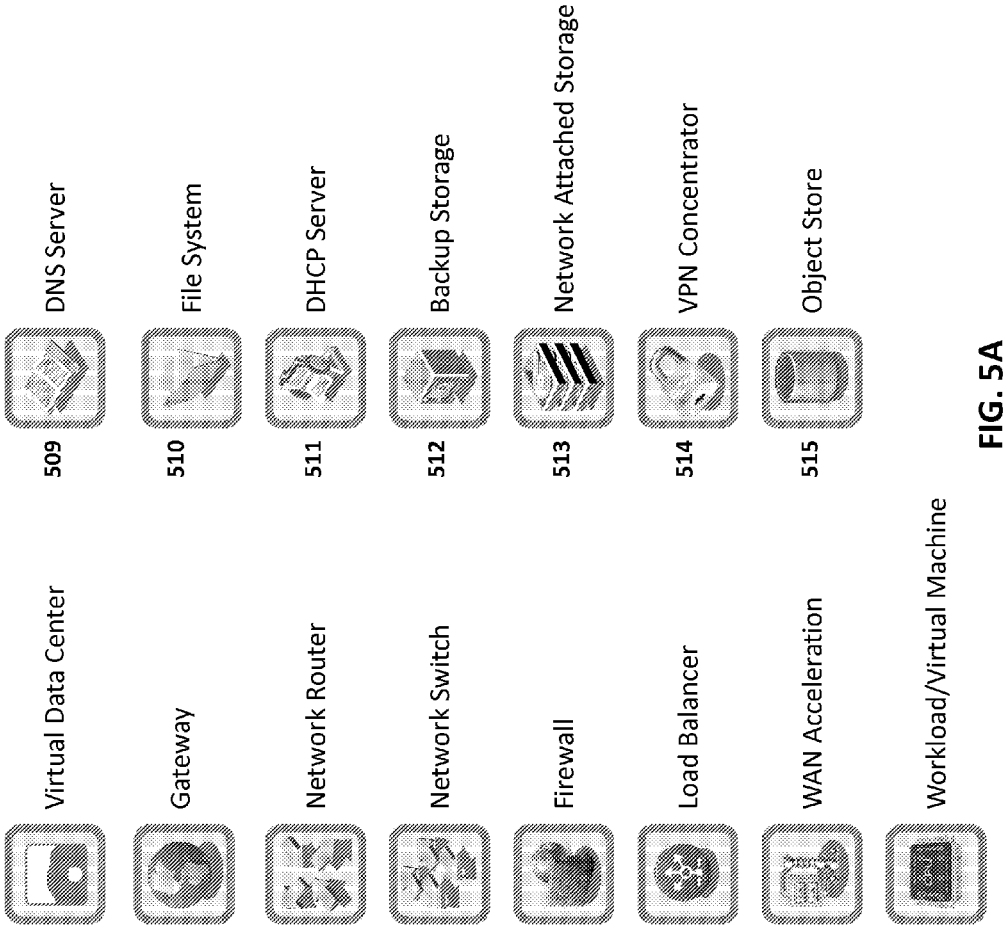

501 Virtual Data Center
502 Gateway
503 Network Router
504 Network Switch
505 Firewall
506 Load Balancer
507 WAN Acceleration
508 Workload/Virtual Machine
509 DNS Server
510 File System
511 DHCP Server
512 Backup Storage
513 Network Attached Storage
514 VPN Concentrator
515 Object Store

FIG. 5A

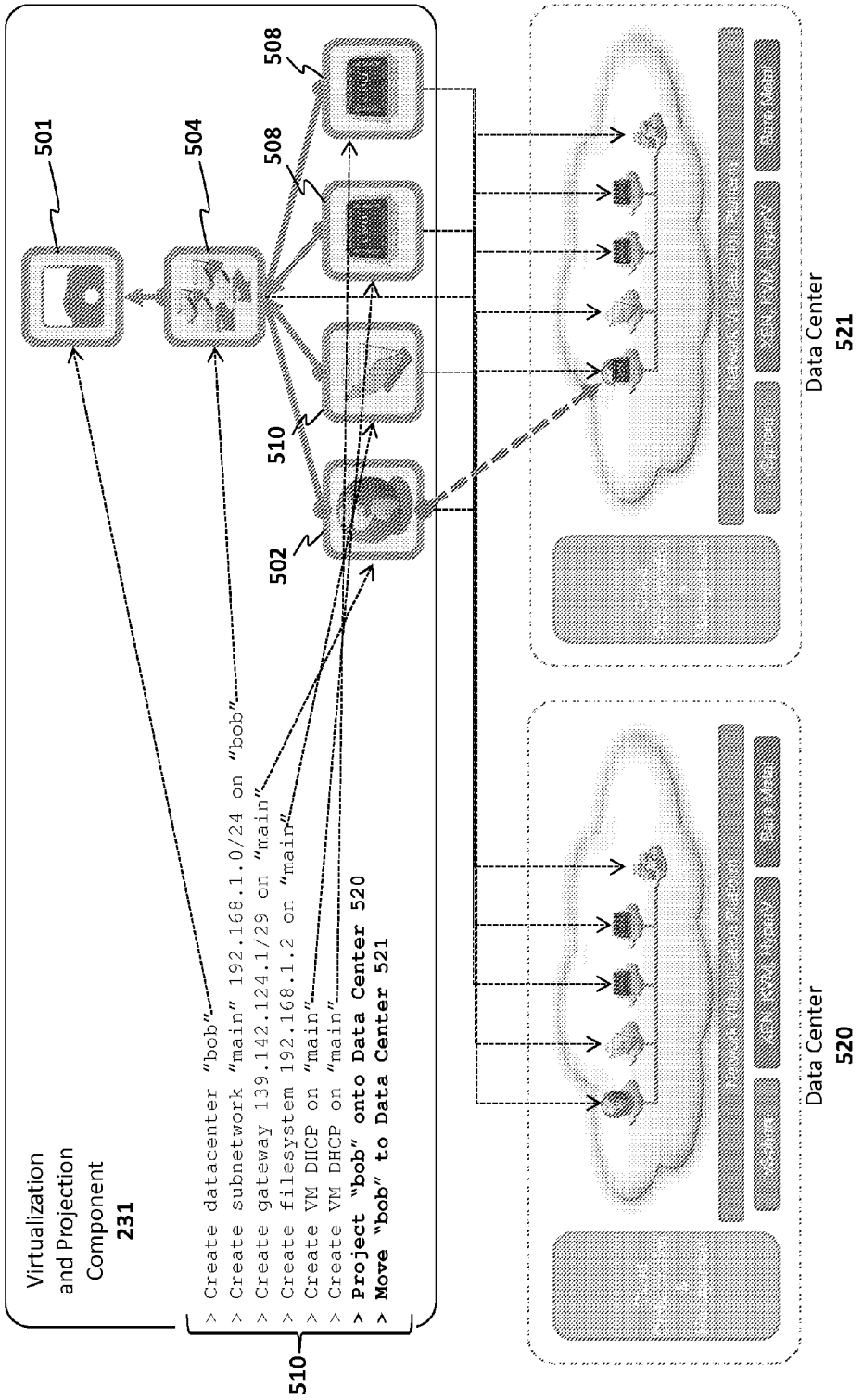

Virtualization and Projection Component 231
> Create datacenter "bob"
> Create subnetwork "main" 192.168.1.0/24 on "bob"
> Create gateway 139.142.124.1/29 on "main"
> Create filesystem 192.168.1.2 on "main"
> Create VM DHCP on "main"
> Create VM DHCP on "main"
> Project "bob" onto Data Center 520
> Move "bob" to Data Center 521
510
501
504
508
508
510
502
Data Center 520
Data Center 521

FIG. 5B

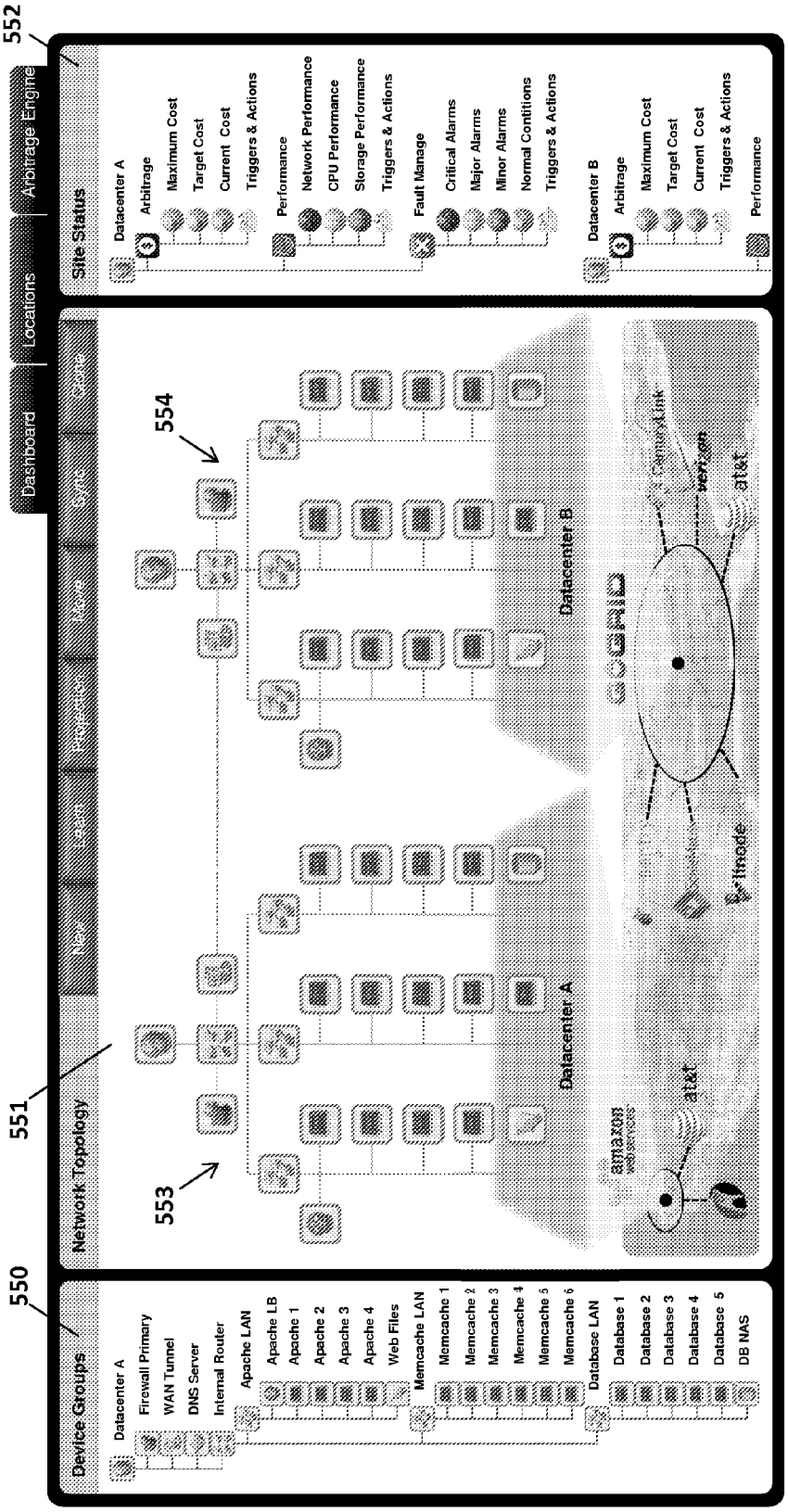

552
Dashboard
Locations
Arbitrage Engine
Site Status
Datacenter A
Arbitrage
Maximum Cost
Target Cost
Current Cost
Triggers & Actions
Performance
Network Performance
CPU Performance
Storage Performance
Triggers & Actions
Fault Manage
Critical Alarms
Major Alarms
Minor Alarms
Normal Contitions
Triggers & Actions
Datacenter B
Arbitrage
Maximum Cost
Target Cost
Current Cost
Triggers & Actions
Performance
551
Network Topology
554
553
Datacenter A
Datacenter B
amazon web services
at&t
verizon
at&t
Linode
550
Device Groups
Datacenter A
Firewall Primary
WAN Tunnel
DNS Server
Internal Router
Apache LAN
Apache LB
Apache 1
Apache 2
Apache 3
Apache 4
Web Files
Memcache LAN
Memcache 1
Memcache 2
Memcache 3
Memcache 4
Memcache 5
Memcache 6
Database LAN
Database 1
Database 2
Database 3
Database 4
Database 5
DB NAS

FIG. 5C

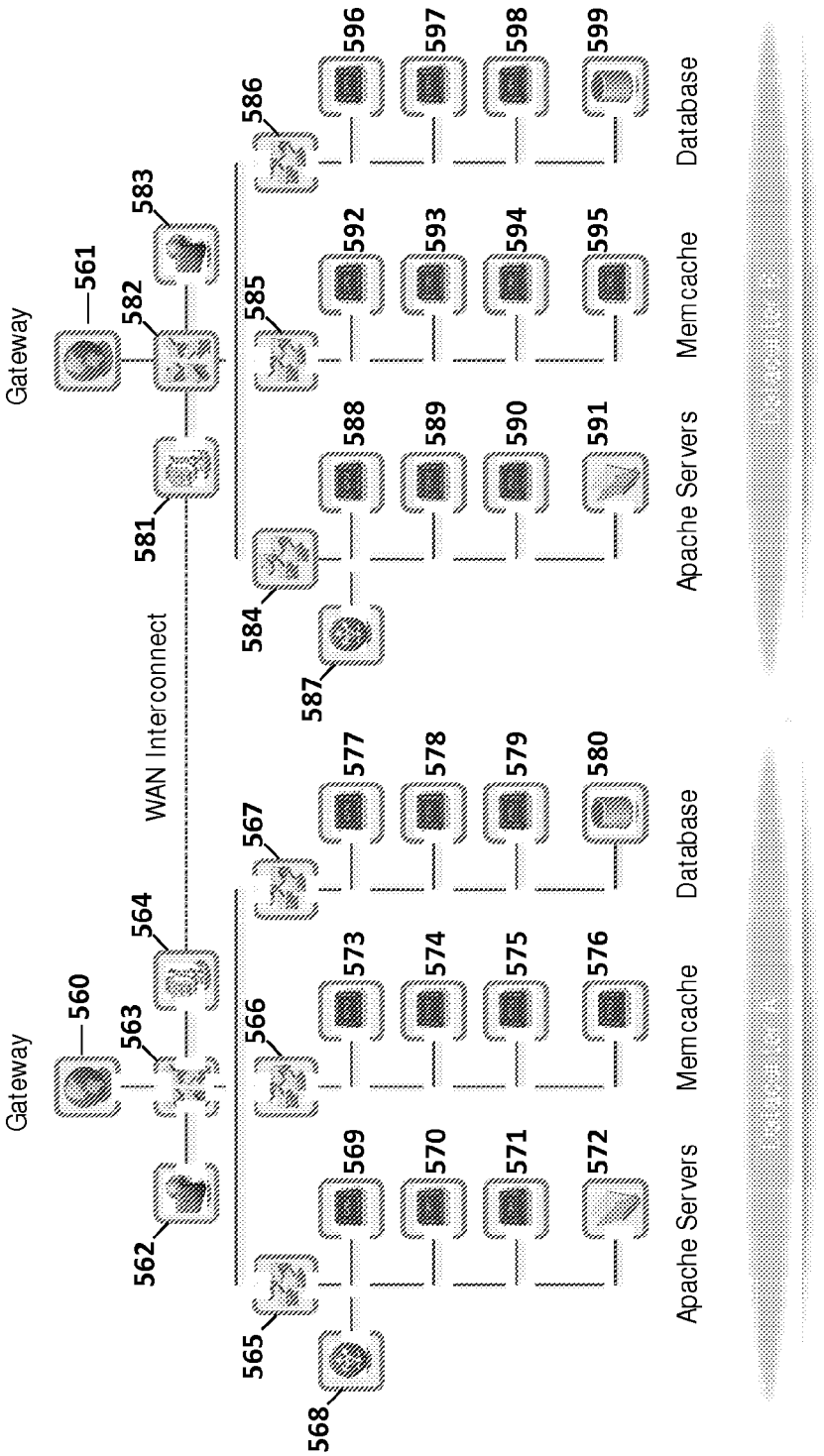

Gateway
Gateway
WAN Interconnect
Apache Servers
Memcache
Database
Apache Servers
Memcache
Database
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599

FIG. 5D

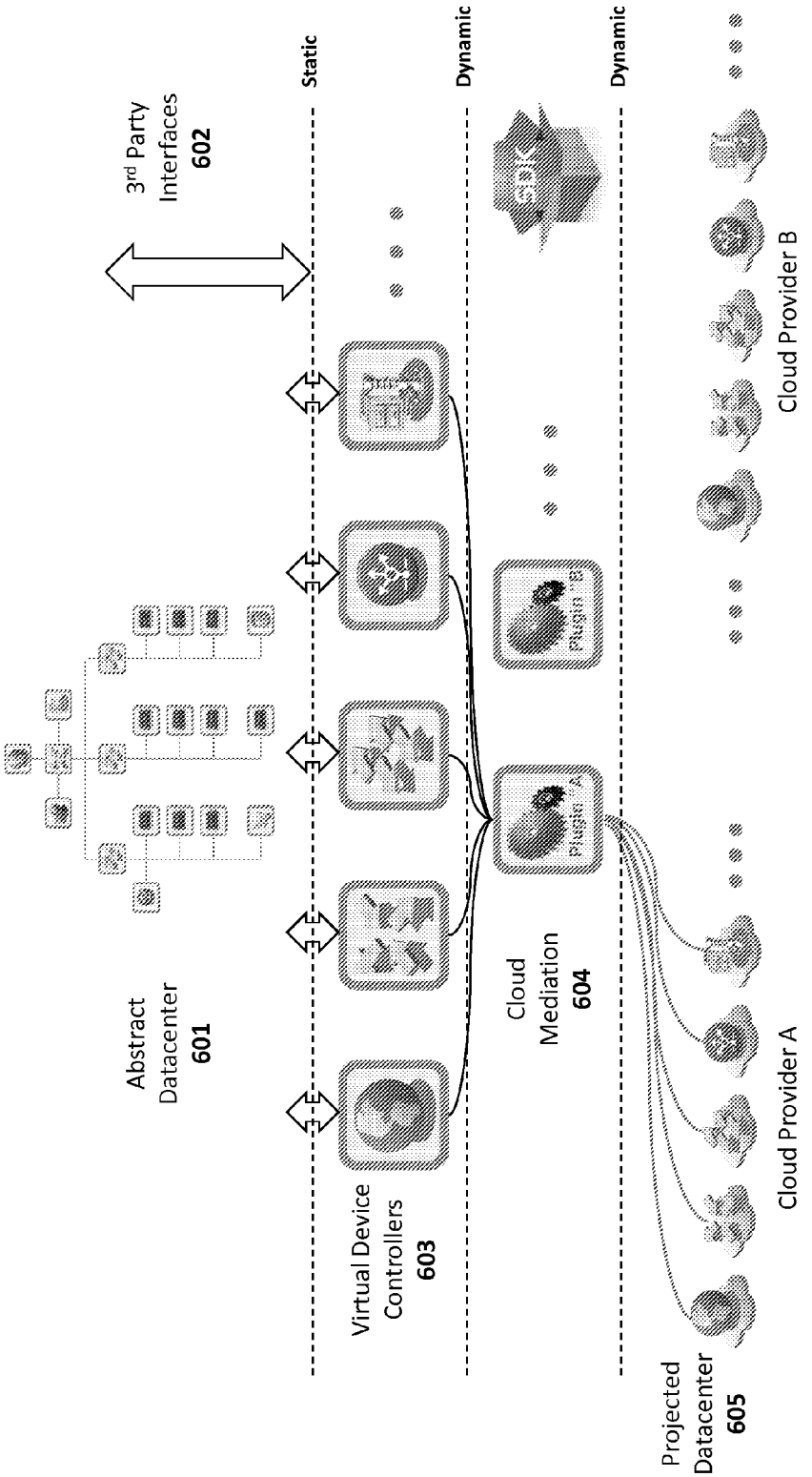

3rd Party Interfaces 602
Static
Dynamic
Dynamic
Abstract Datacenter 601
Virtual Device Controllers 603
Cloud Mediation 604
Plugin A
Plugin B
SDK
Cloud Provider A
Cloud Provider B
Projected Datacenter 605

FIG. 6

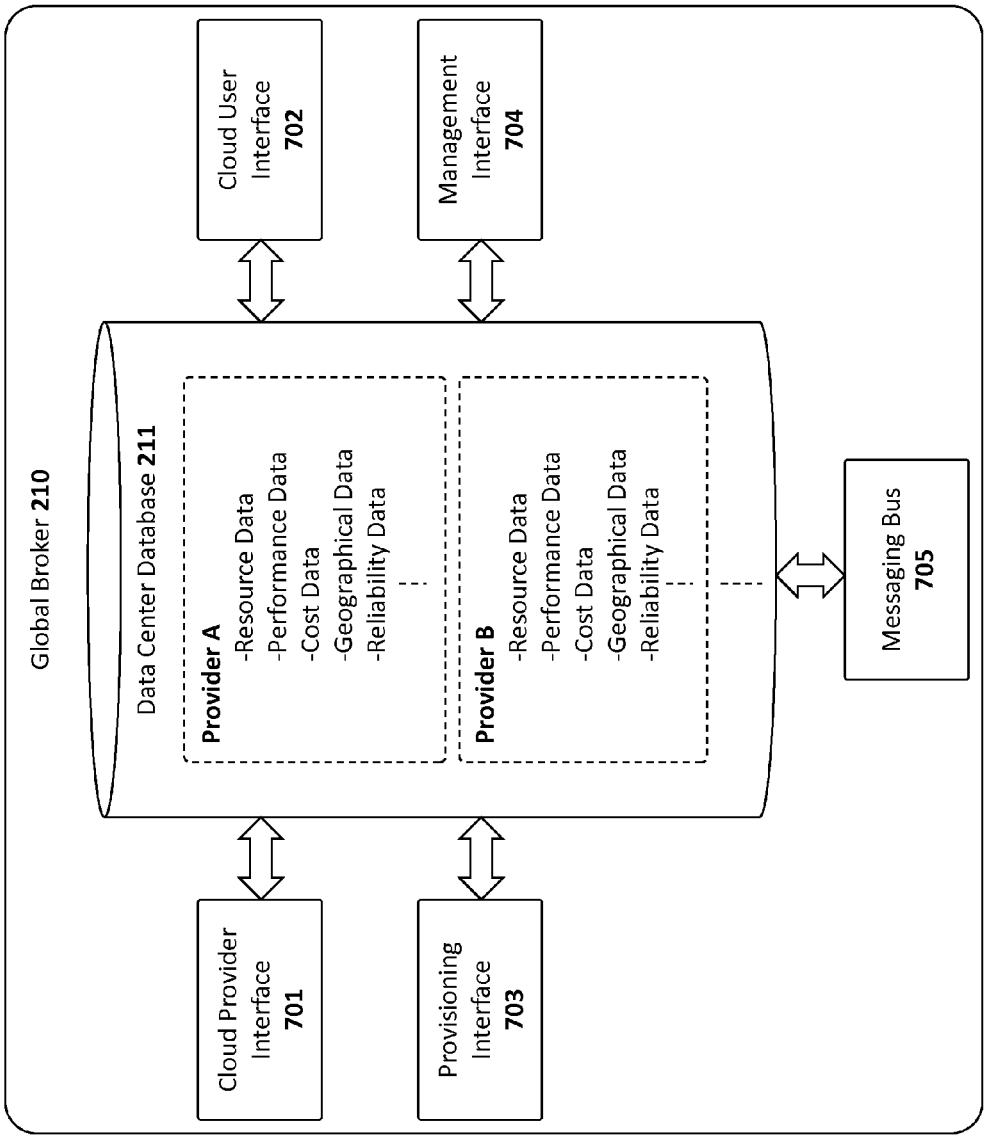

Global Broker 210
Cloud User Interface 702
Management Interface 704
Data Center Database 211
Provider A
-Resource Data
-Performance Data
-Cost Data
-Geographical Data
-Reliability Data
Provider B
-Resource Data
-Performance Data
-Cost Data
-Geographical Data
-Reliability Data
Messaging Bus 705
Cloud Provider Interface 701
Provisioning Interface 703

FIG. 7

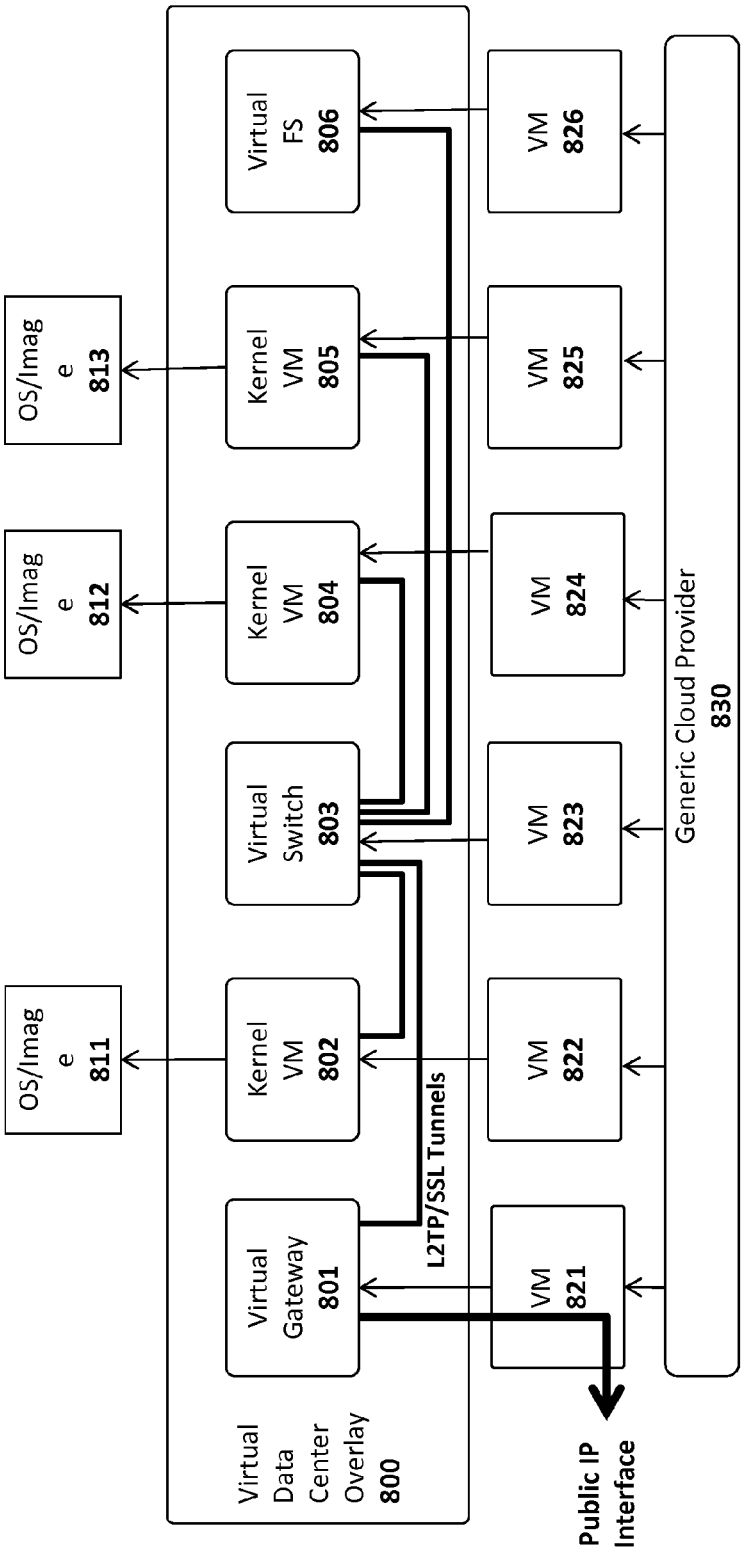

Virtual Data Center Overlay 800
Virtual Gateway 801
Virtual Switch 803
Virtual FS 806
Kernel VM 802
Kernel VM 804
Kernel VM 805
OS/Image 811
OS/Image 812
OS/Image 813
VM 821
VM 822
VM 823
VM 824
VM 825
VM 826
L2TP/SSL Tunnels
Public IP Interface
Generic Cloud Provider 830

FIG. 8

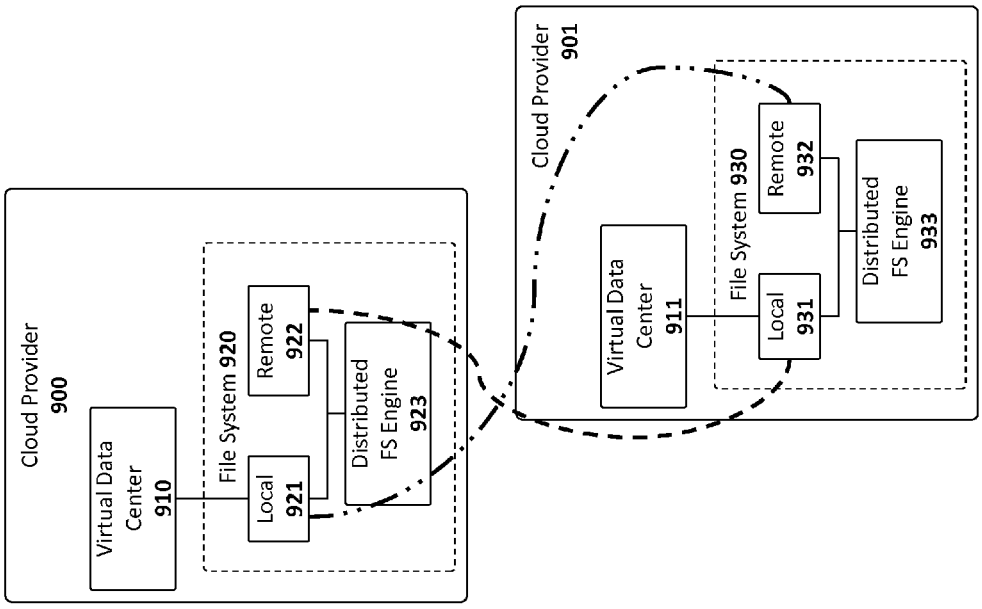

Cloud Provider
900
Virtual Data Center
910
File System 920
Local
921
Remote
922
Distributed FS Engine
923
Cloud Provider
901
Virtual Data Center
911
File System 930
Local
931
Remote
932
Distributed FS Engine
933

FIG. 9

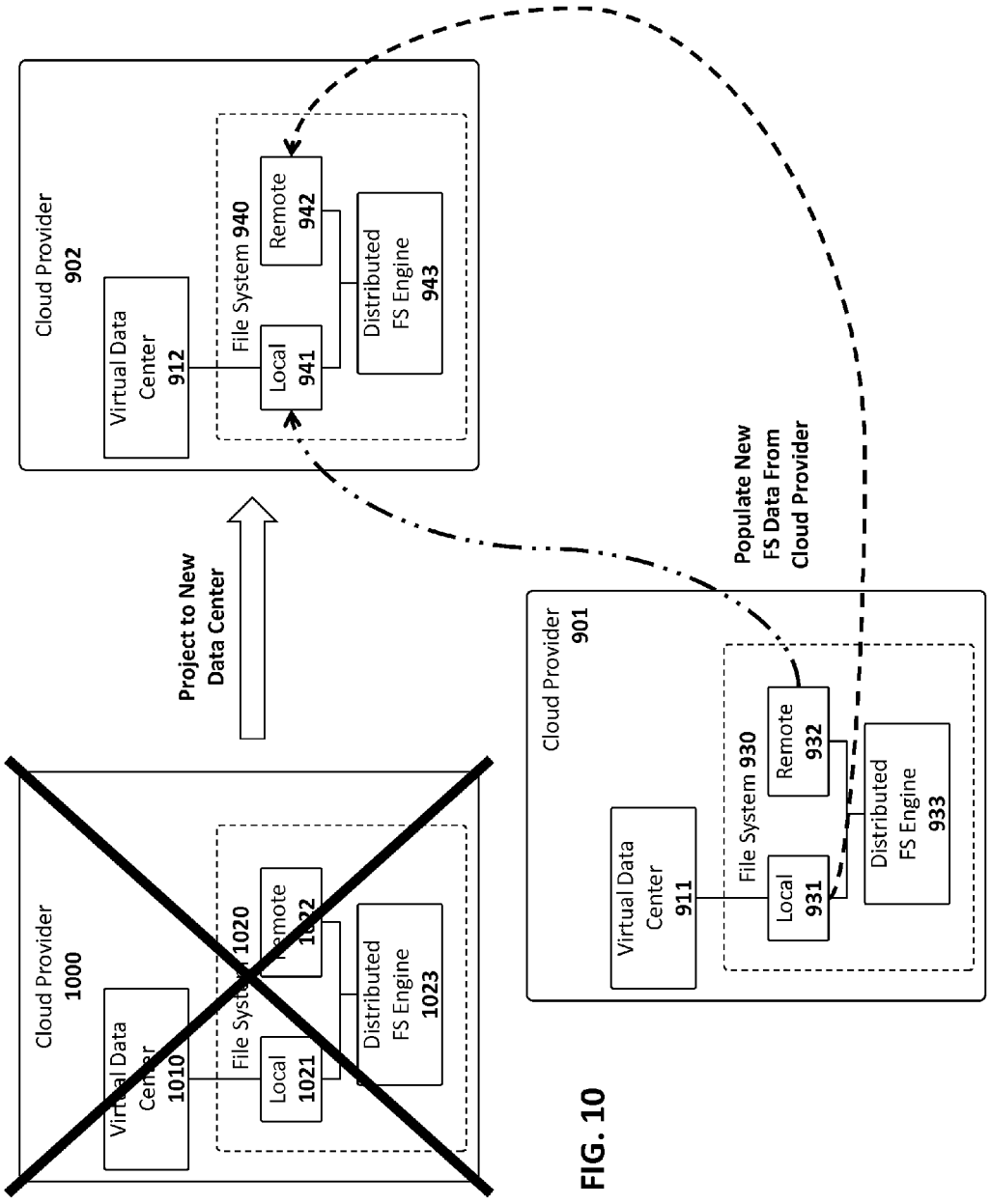

Cloud Provider
902
Virtual Data Center
912
File System 940
Local
941
Remote
942
Distributed FS Engine
943
Project to New Data Center
Populate New FS Data From Cloud Provider
Cloud Provider
901
Virtual Data Center
911
File System 930
Local
931
Remote
932
Distributed FS Engine
933
Cloud Provider
1000
Virtual Data Center
1010
File System 1020
Local
1021
Remote
1022
Distributed FS Engine
1023

FIG. 10

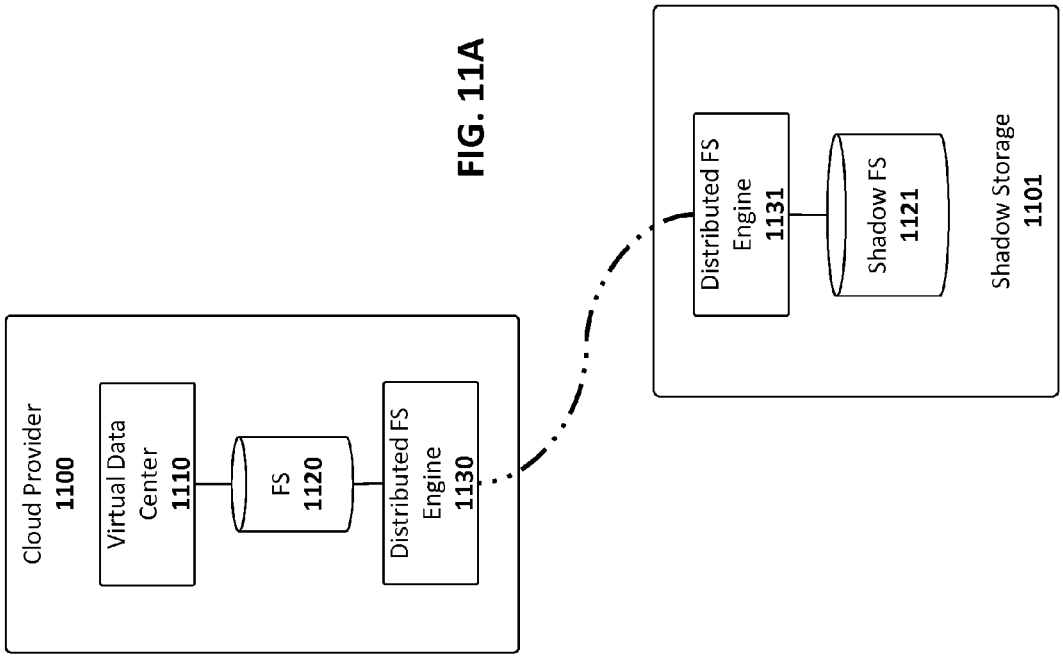

Cloud Provider
1100
Virtual Data Center
1110
FS
1120
Distributed FS Engine
1130
Distributed FS Engine
1131
Shadow FS
1121
Shadow Storage
1101

FIG. 11A

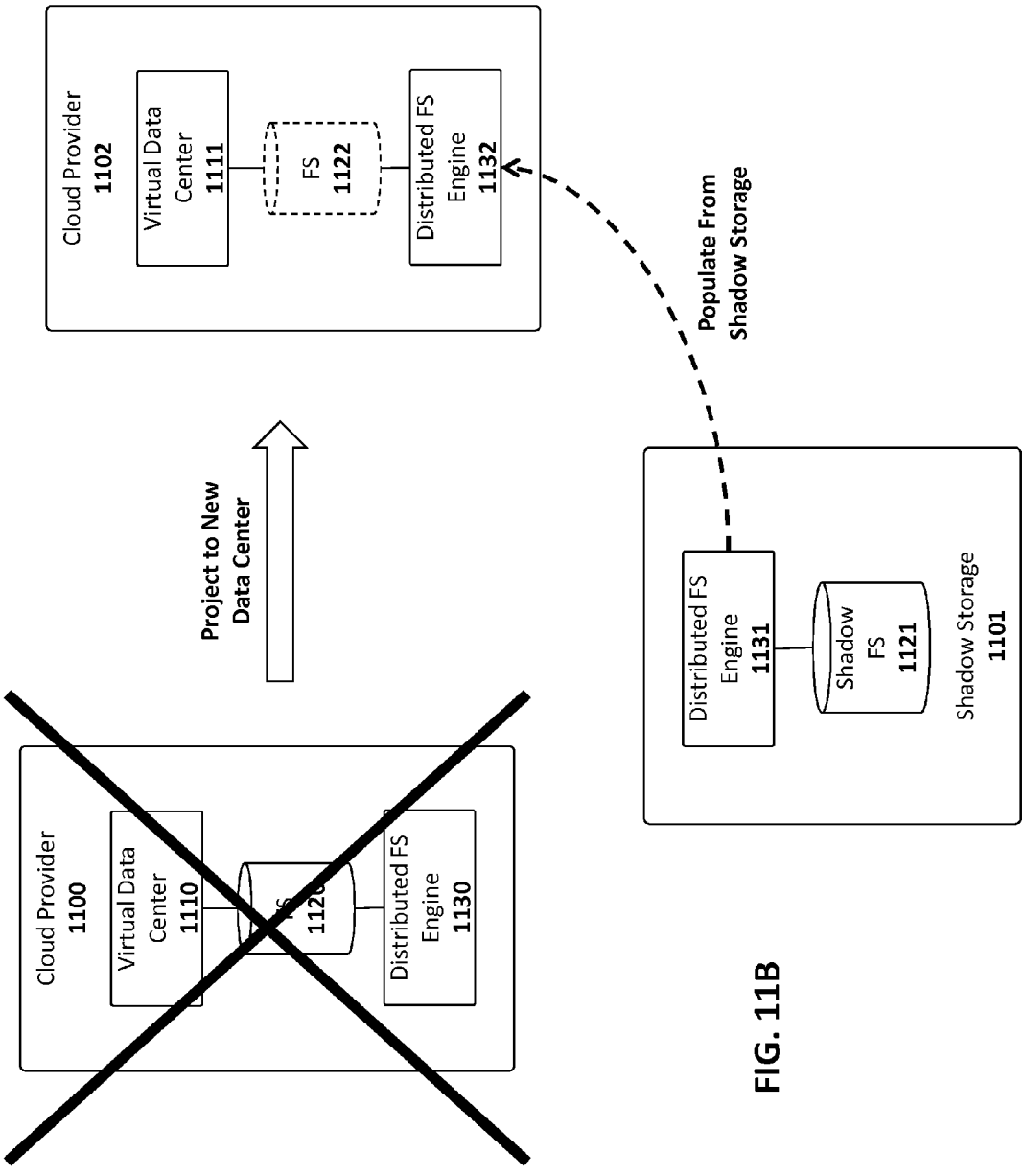

Cloud Provider
1102
Virtual Data Center
1111
FS
1122
Distributed FS Engine
1132
Populate From Shadow Storage
Project to New Data Center
Cloud Provider
1100
Virtual Data Center
1110
Distributed FS Engine
1130
Distributed FS Engine
1131
Shadow FS
1121
Shadow Storage
1101

FIG. 11B

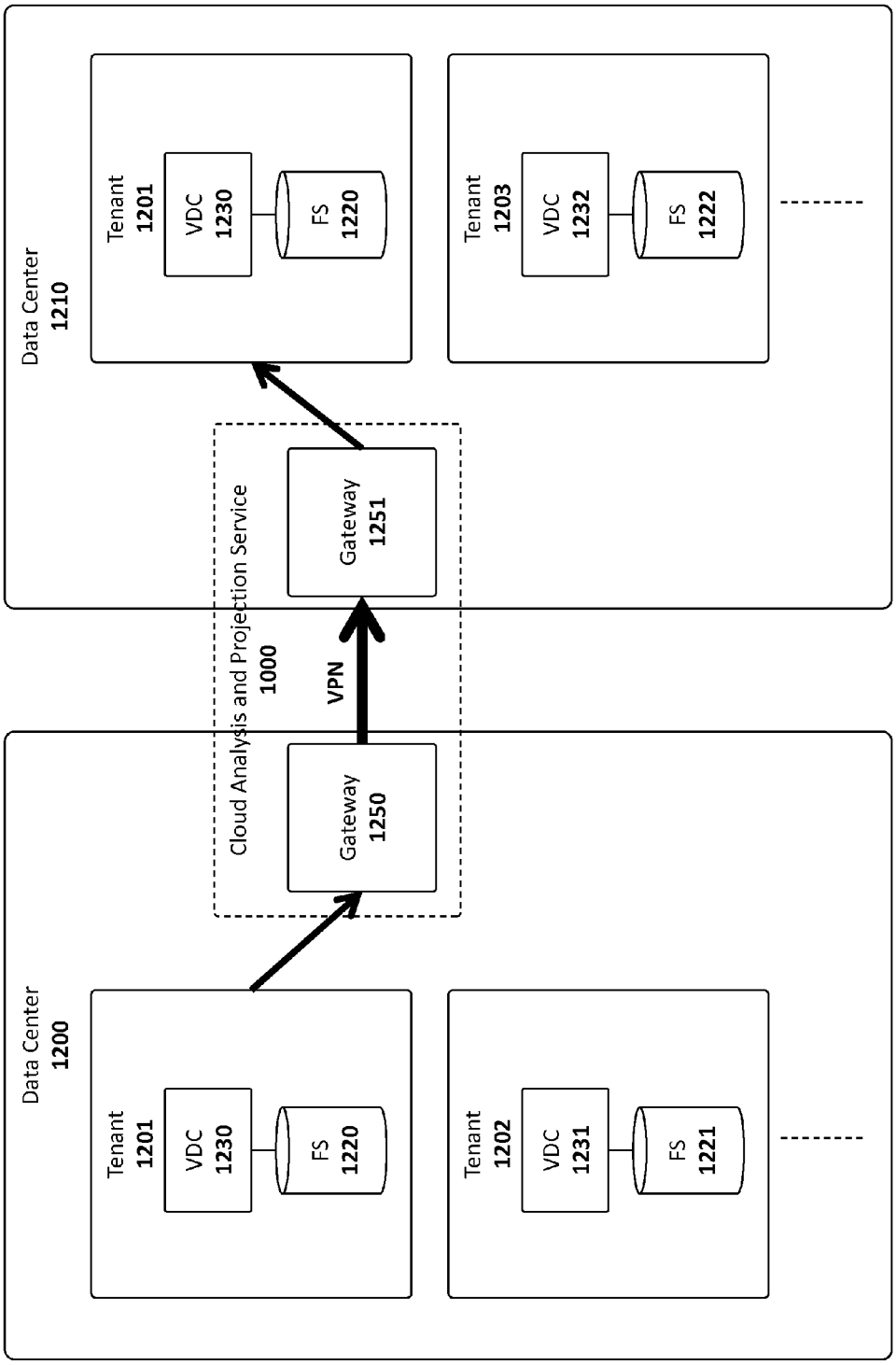

Data Center
1200
Tenant
1201
VDC
1230
FS
1220
Tenant
1202
VDC
1231
FS
1221
Cloud Analysis and Projection Service
1000
Gateway
1250
VPN
Gateway
1251
Data Center
1210
Tenant
1201
VDC
1230
FS
1220
Tenant
1203
VDC
1232
FS
1222

FIG. 12A

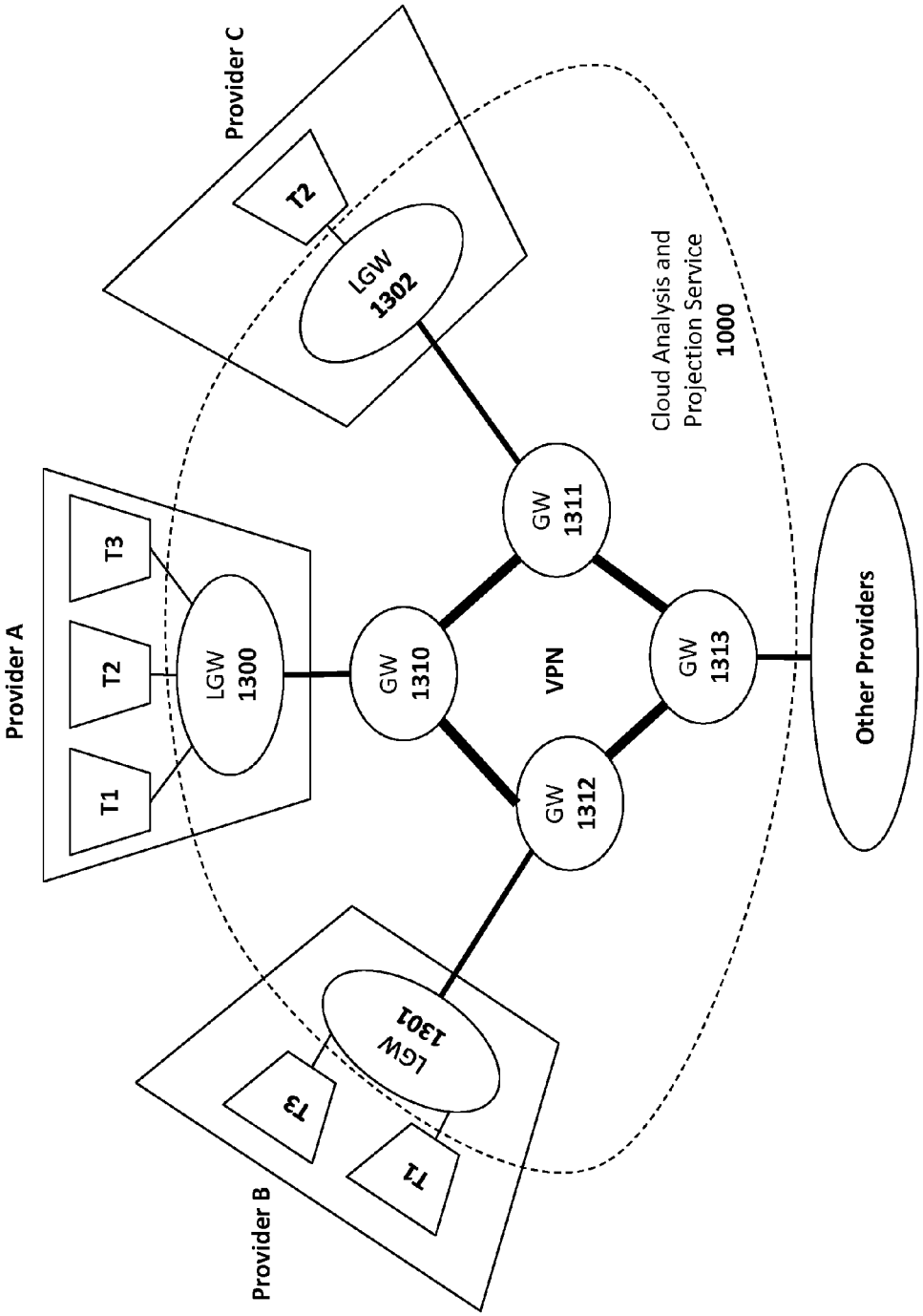

Provider C
T2
LGW
1302
Cloud Analysis and
Projection Service
1000
Provider A
T3
T2
T1
LGW
1300
GW
1310
GW
1311
VPN
GW
1312
GW
1313
Other Providers
Provider B
T3
T1
LGW
1301

FIG. 12B

Cloud Provider Connectivity Table **1290**

Provider	1	2	3	4	5
1	NA	Y	Y	N	N
2	Y	NA	Y	N	N
3	Y	Y	NA	N	N
4	N	N	N	NA	N
5	N	N	N	N	NA

FIG. 12C

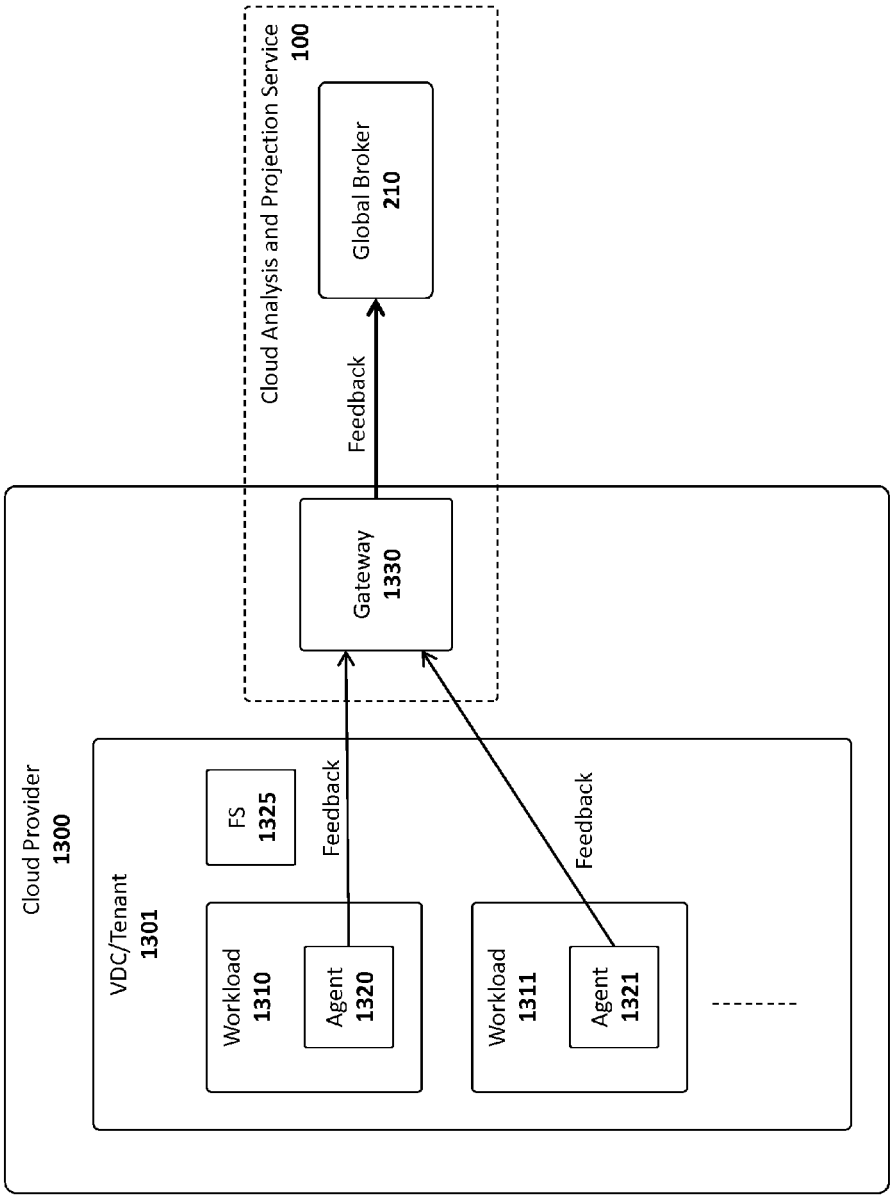

Cloud Analysis and Projection Service
100
Global Broker
210
Feedback
Gateway
1330
Cloud Provider
1300
VDC/Tenant
1301
FS
1325
Feedback
Feedback
Workload
1310
Agent
1320
Workload
1311
Agent
1321

FIG. 13A

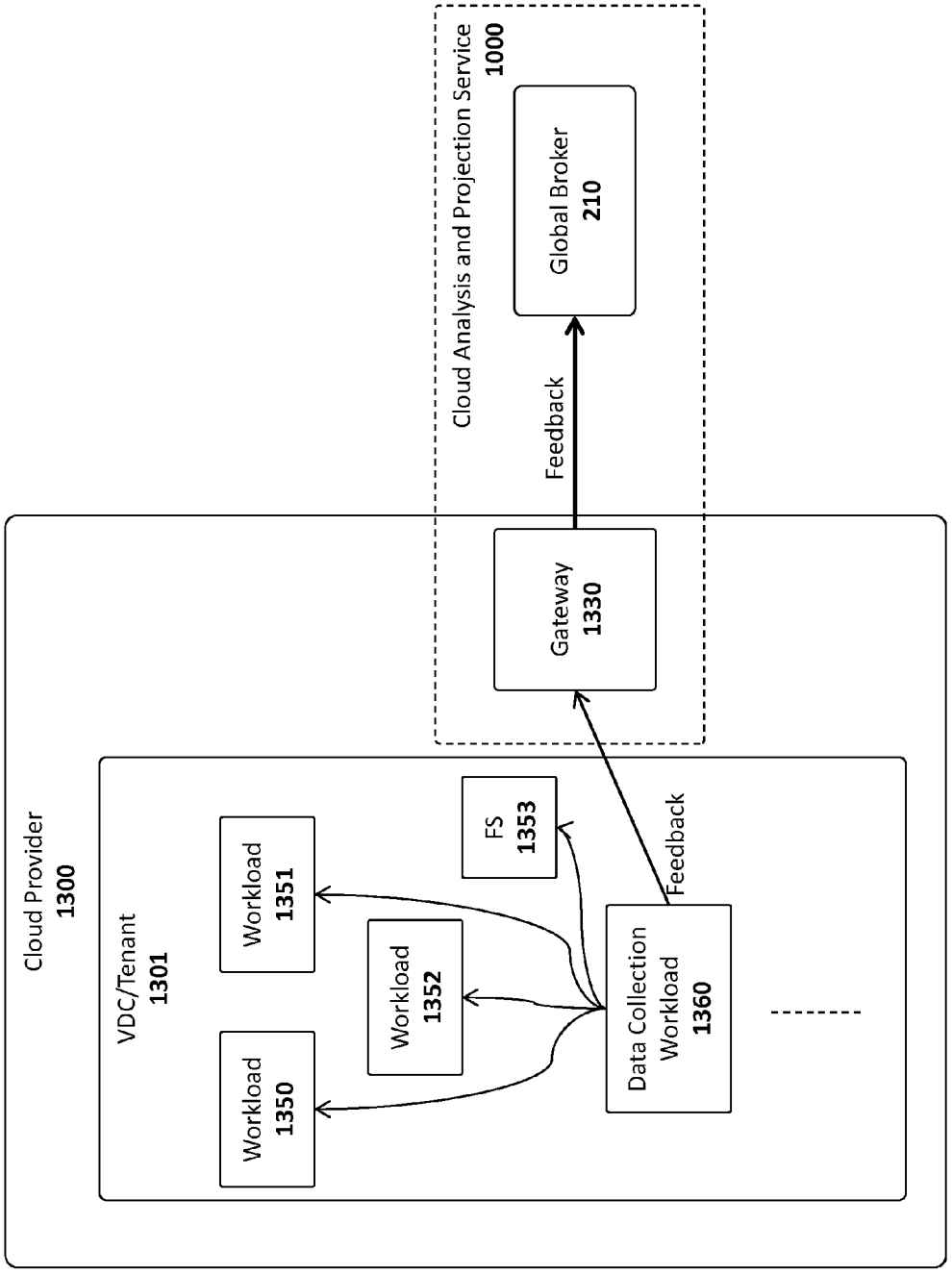

Cloud Analysis and Projection Service
1000
Global Broker
210
Feedback
Gateway
1330
Cloud Provider
1300
VDC/Tenant
1301
Workload
1351
FS
1353
Feedback
Workload
1352
Data Collection
Workload
1360
Workload
1350

FIG. 13B

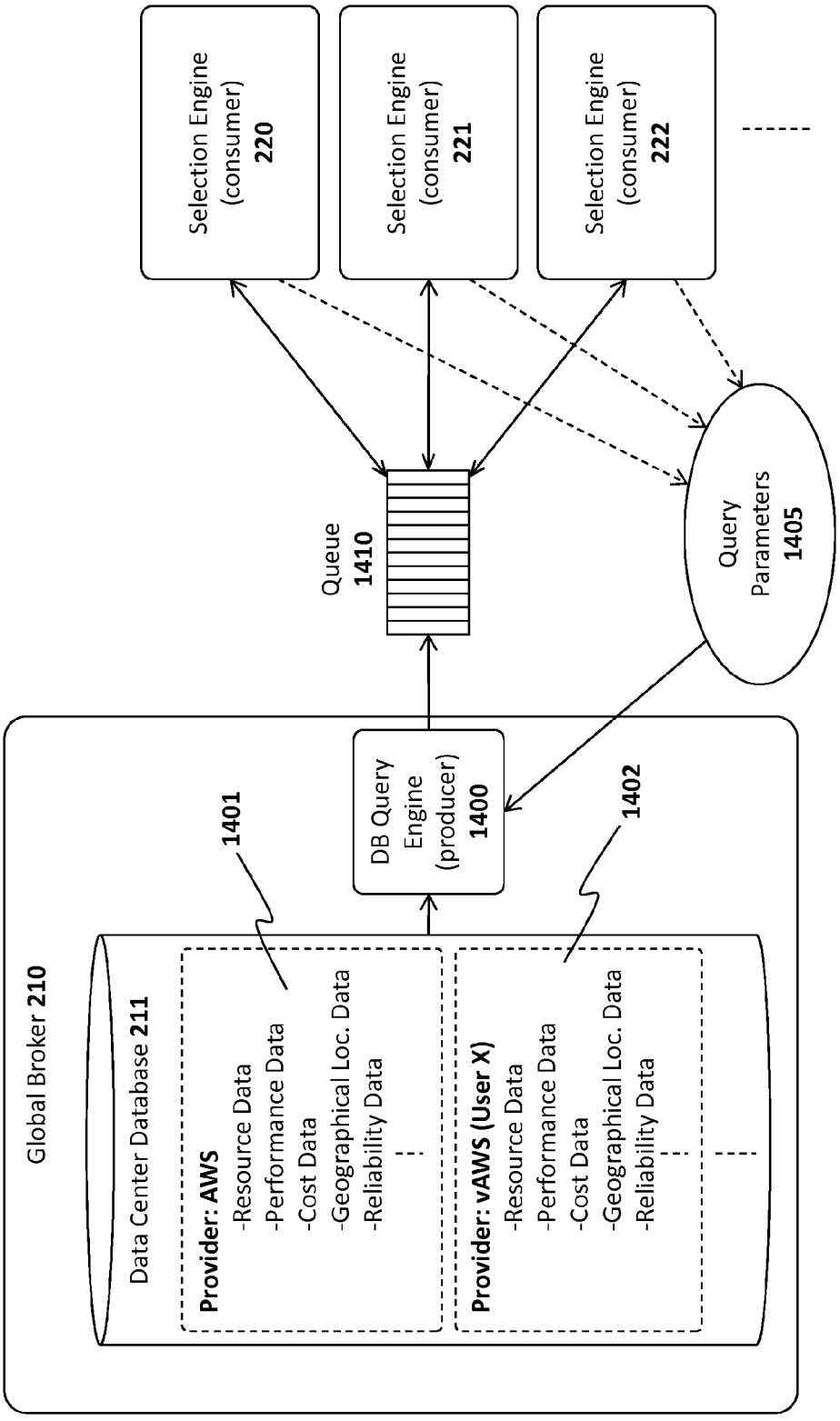

Selection Engine (consumer) 220
Selection Engine (consumer) 221
Selection Engine (consumer) 222
Queue 1410
Query Parameters 1405
DB Query Engine (producer) 1400
Global Broker 210
Data Center Database 211
1401
1402
Provider: AWS
-Resource Data
-Performance Data
-Cost Data
-Geographical Loc. Data
-Reliability Data
Provider: vAWS (User X)
-Resource Data
-Performance Data
-Cost Data
-Geographical Loc. Data
-Reliability Data

FIG. 14

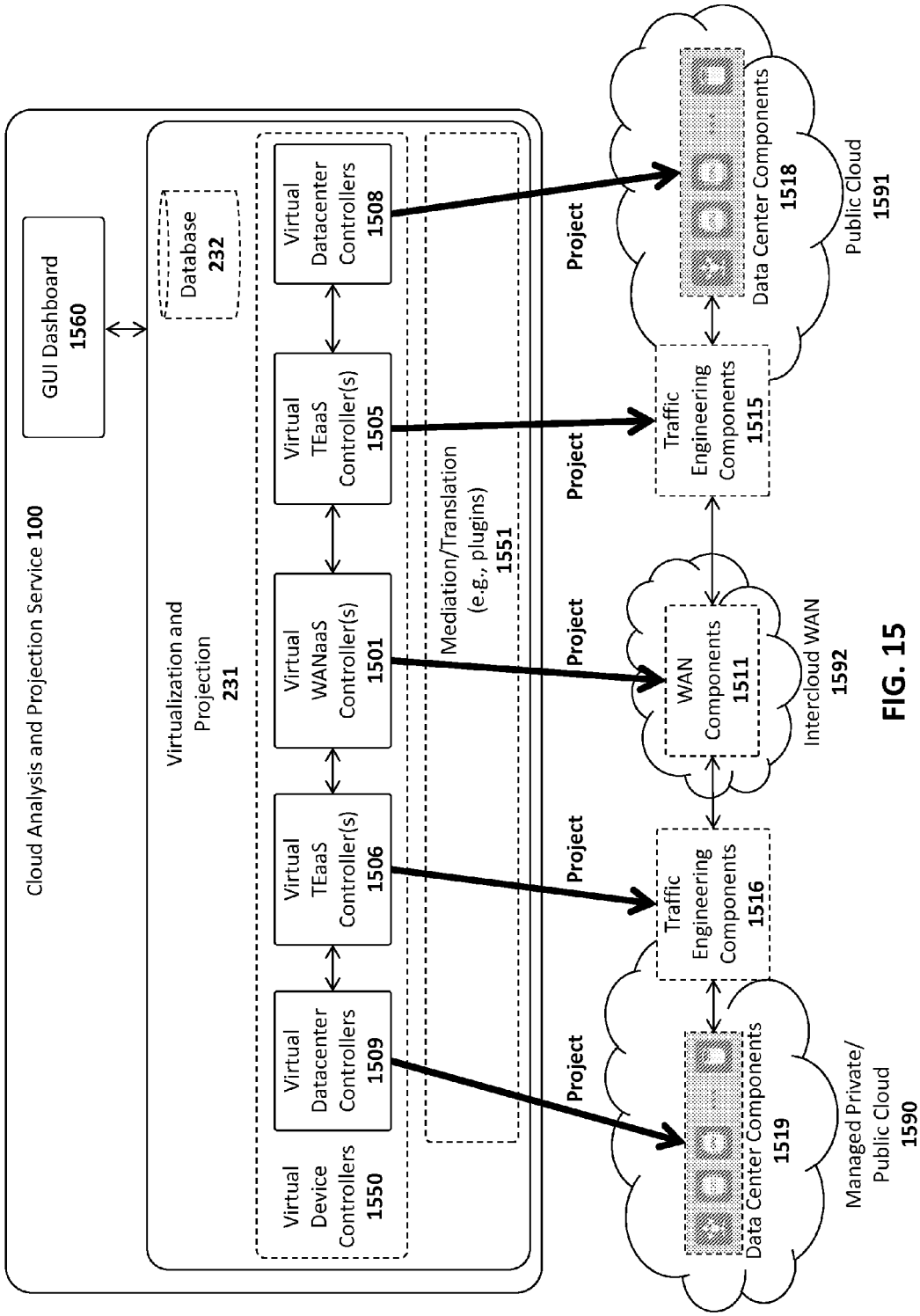

Cloud Analysis and Projection Service 100
GUI Dashboard 1560
Database 232
Virtualization and Projection 231
Virtual Device Controllers 1550
Virtual Datacenter Controllers 1509
Virtual TEaaS Controller(s) 1506
Virtual WANaaS Controller(s) 1501
Virtual TEaaS Controller(s) 1505
Virtual Datacenter Controllers 1508
Mediation/Translation (e.g., plugins) 1551
Project
Project
Project
Project
Project
Traffic Engineering Components 1516
WAN Components 1511
Traffic Engineering Components 1515
Data Center Components 1519
Intercloud WAN 1592
Data Center Components 1518
Managed Private/ Public Cloud 1590
Public Cloud 1591

FIG. 15

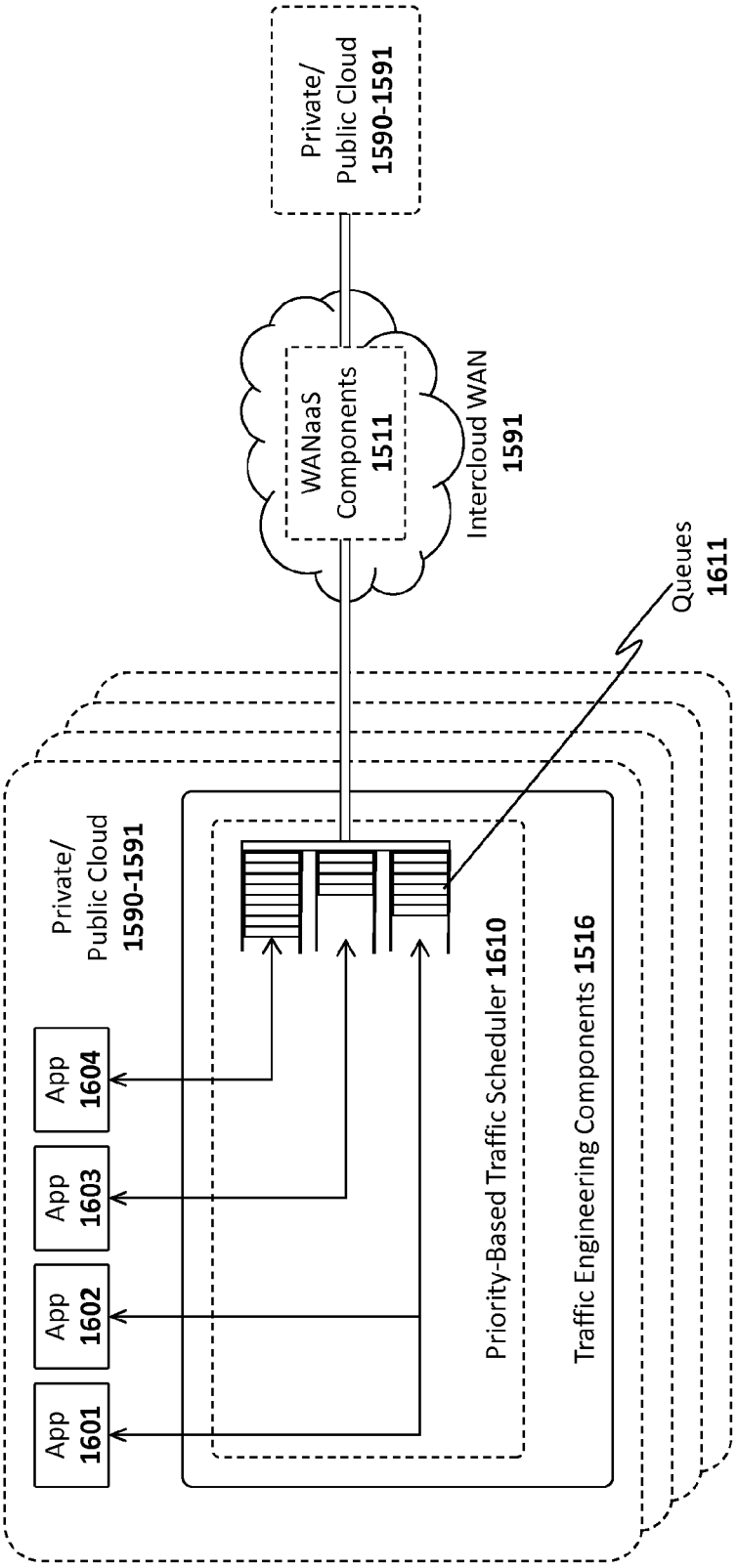

Private/
Public Cloud
1590-1591
WANaaS
Components
1511
Intercloud WAN
1591
Queues
1611
Private/
Public Cloud
1590-1591
App
1604
App
1603
App
1602
App
1601
Priority-Based Traffic Scheduler 1610
Traffic Engineering Components 1516

FIG. 16

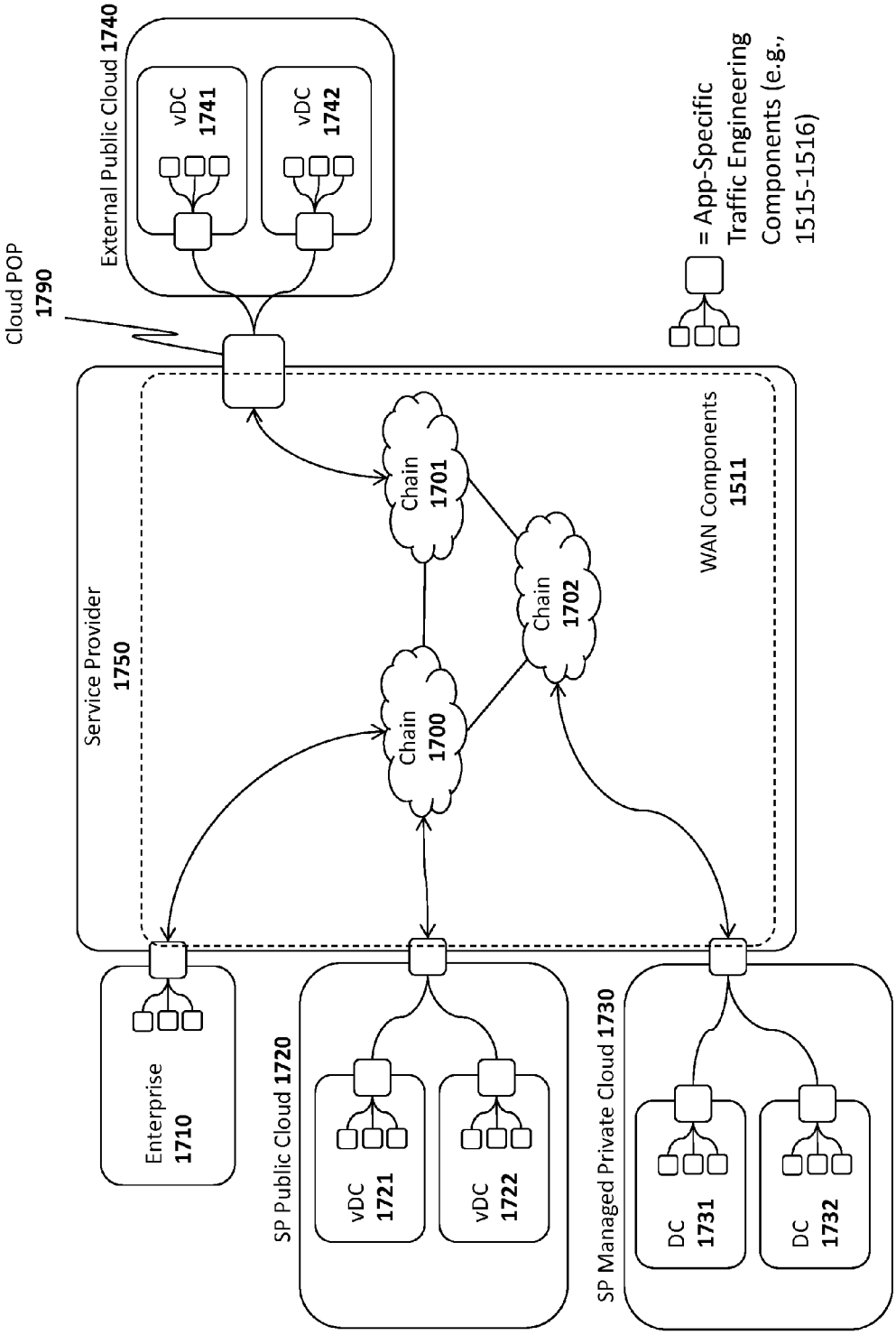

Cloud POP
1790
External Public Cloud 1740
vDC
1741
vDC
1742
= App-Specific
Traffic Engineering
Components (e.g.,
1515-1516)
Service Provider
1750
WAN Components
1511
Chain
1701
Chain
1702
Chain
1700
Enterprise
1710
SP Public Cloud 1720
vDC
1721
vDC
1722
SP Managed Private Cloud 1730
DC
1731
DC
1732

FIG. 17

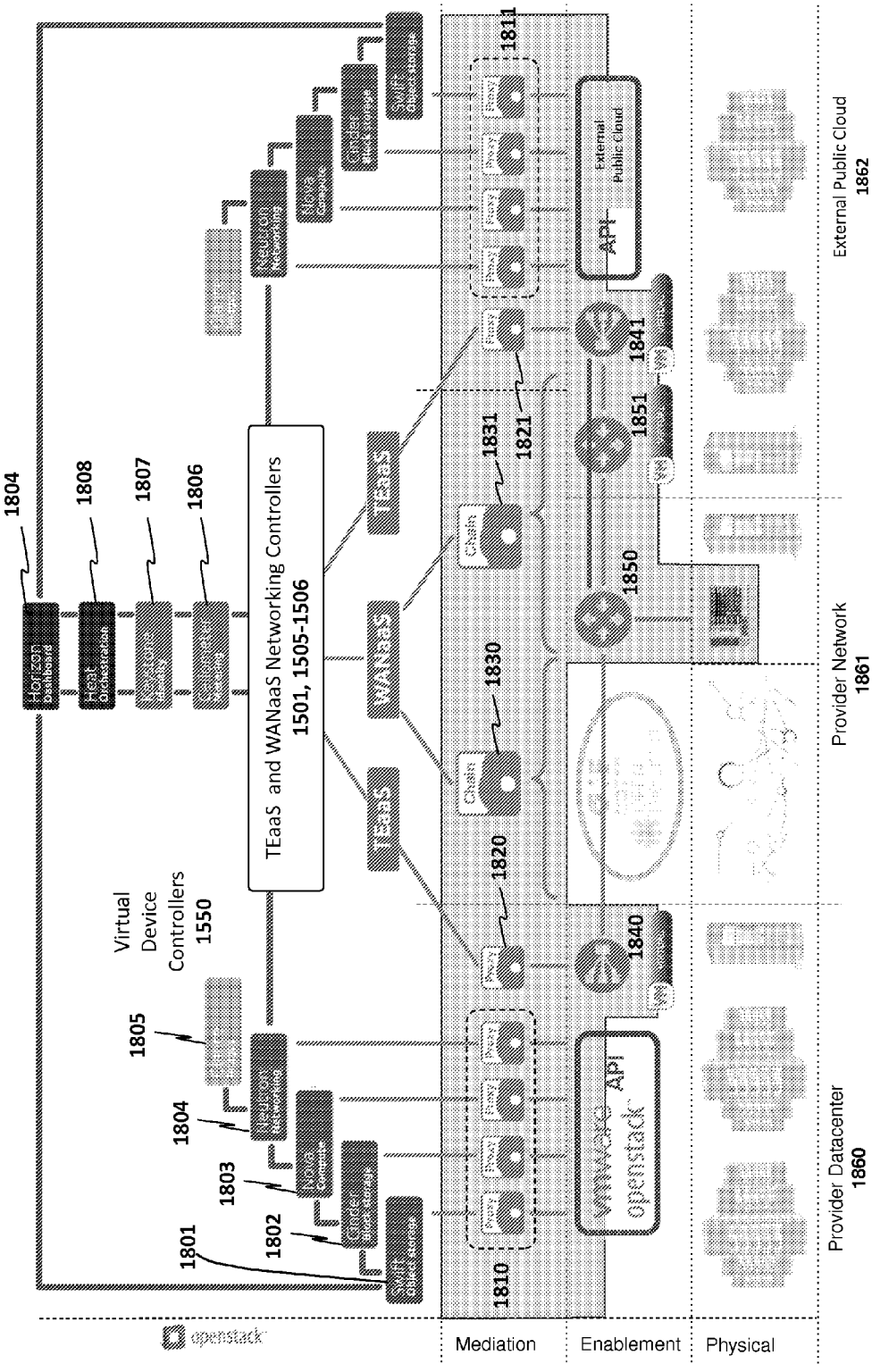

1804
1808
1807
1806
TEaaS and WANaaS Networking Controllers
1501, 1505-1506
Virtual Device Controllers 1550
1805
1804
1803
1802
1801
1810
1820
1830
1831
1821
1811
1840
1850
1851
1841
API
External Public Cloud
1862
Provider Network
1861
Provider Datacenter
1860
Mediation
Enablement
Physical

FIG. 18

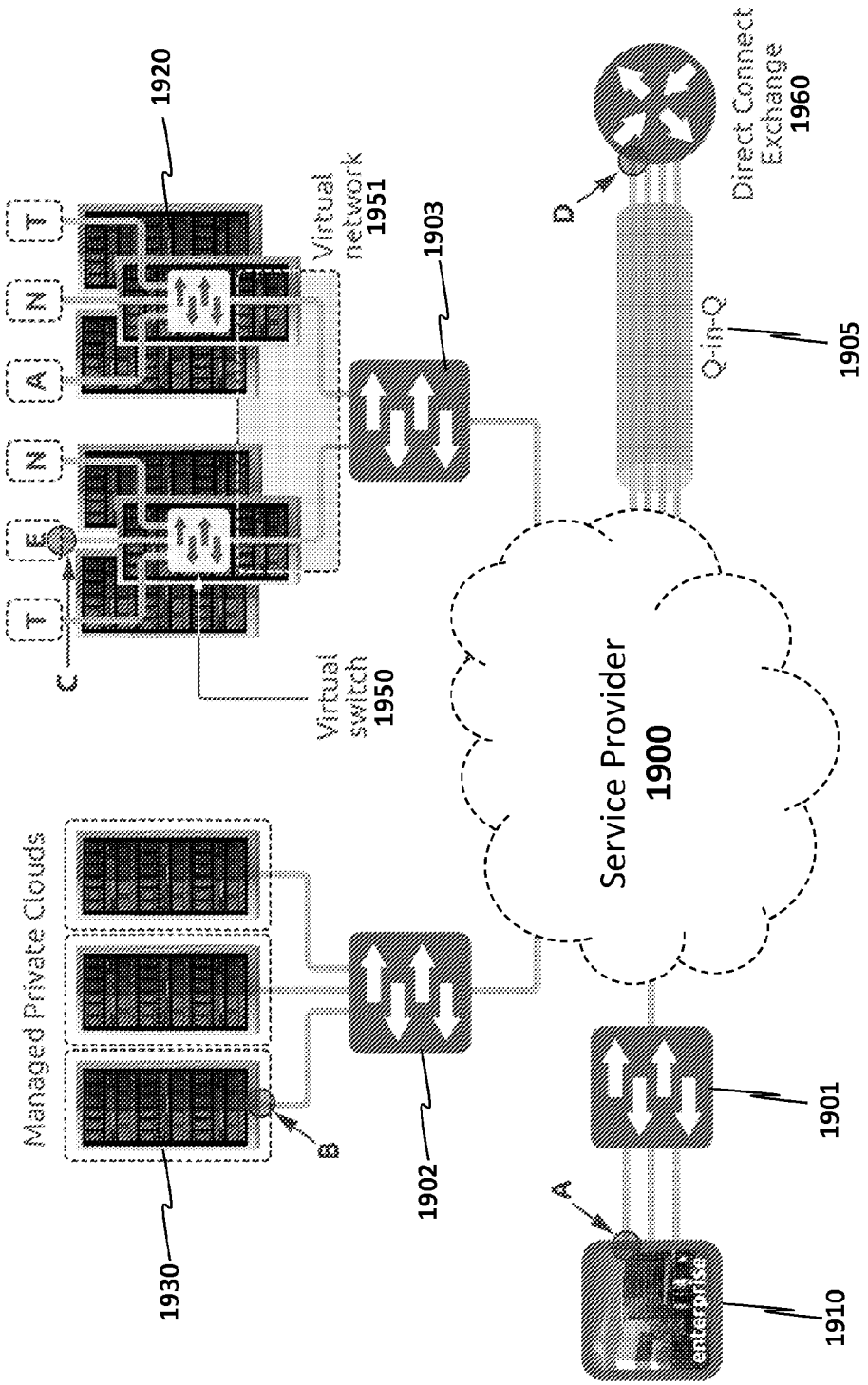

1920
Virtual network 1951
1903
Direct Connect Exchange 1960
D
Q-in-Q
1905
T
N
A
N
E
T
C
Virtual switch 1950
Service Provider 1900
Managed Private Clouds
B
1902
1901
A
enterprise
1930
1910

FIG. 19

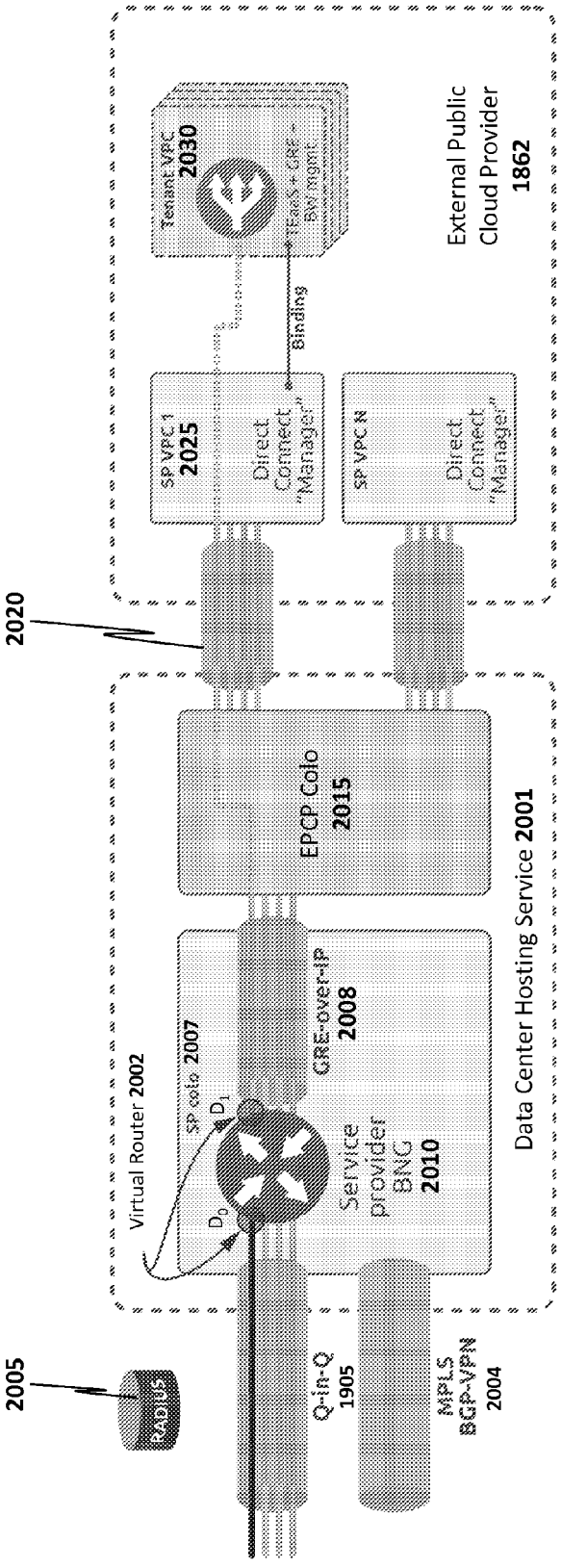

External Public Cloud Provider 1862
Tenant VPC 2030
TEaaS + CRE + BW mgmt
Binding
SP VPC 1 2025
Direct Connect "Manager"
SP VPC N
Direct Connect "Manager"
2020
EPCP Colo 2015
Data Center Hosting Service 2001
Virtual Router 2002
SP edge 2007
D1
D0
CRE-over-IP 2008
Service provider BNG 2010
2005
Radius
Q-in-Q 1905
MPLS BGP-VPN 2004

FIG. 20

ARCHITECTURE AND METHOD FOR VIRTUALIZATION OF CLOUD NETWORKING COMPONENTS

BACKGROUND

[0001] 1. Field of the Invention

[0002] This invention relates generally to the field of data processing systems. More particularly, the invention relates to a system and method for virtualization of cloud networking components.

[0003] 2. Description of Related Art

[0004] Cloud computing may be provided using the models of infrastructure as a service (IaaS), platform as a service (PaaS), and software as a service (SaaS). Any of these models may be implemented within a cloud-based "data center" comprised of various computing resources (e.g., servers, routers, load balancers, switches, etc).

[0005] IaaS is the most basic model. Providers of IaaS offer physical or virtual computers (i.e., using virtual machines) and other resources such as a virtual-machine disk image library, storage resources, including file-based storage, firewalls, load balancers, IP addresses, virtual local area networks (VLANs), and software bundles. IaaS providers may supply these resources dynamically, from large pools installed in data centers. To deploy their applications, cloud users install operating system images and application software on the cloud resources. In this model, the cloud user maintains the operating systems and the application software. Cloud providers typically bill users based on the amount of resources allocated and consumed.

[0006] In the PaaS model, cloud providers deliver a complete computing platform, typically including an operating system, Web server, programming language execution environment, and database. Application developers develop and run software solutions on this cloud platform without the cost and complexity associated with buying and managing the underlying hardware and software layers. In some PaaS implementations, the underlying resources (e.g., computing, storage, etc) scale automatically to match application demand so that the cloud user does not have to allocate resources manually.

[0007] In the SaaS model, cloud providers install and maintain application software in the cloud and cloud users access the software from cloud clients (sometimes referred to as an "on-demand software" model). This eliminates the need to install and run the application on the cloud user's own computers, which simplifies maintenance and support. Cloud applications offer virtually unlimited scalability (in contrast to locally executed applications) which may be achieved by cloning tasks onto multiple virtual machines during run-time to meet changing work demand. Load balancers distribute the work over the set of virtual machines transparently to the user (who sees only a single access point).

BRIEF DESCRIPTION OF THE DRAWINGS

[0008] A better understanding of the present invention can be obtained from the following detailed description in conjunction with the following drawings, in which:

[0009] FIG. **1**A illustrates one embodiment of a cloud analysis and projection service;

[0010] FIG. **1**B illustrates a graph showing details associated with the cloud provider market;

[0011] FIG. **2**A illustrates a system architecture in accordance with one embodiment of the invention;

[0012] FIGS. **2**B-C illustrates methods in accordance with one embodiment of the invention;

[0013] FIG. **3** illustrates a graphical example of data center arbitrage employed in one embodiment of the invention;

[0014] FIG. **4** illustrates one embodiment of a selection engine architecture;

[0015] FIGS. **5**A-D illustrate additional details associated with one embodiment of a virtualization and projection component including a graphical user interface;

[0016] FIG. **6** illustrates a plurality of logical layers employed in one embodiment for projecting a virtual data center to a physical data center;

[0017] FIG. **7** illustrates additional details associated with one embodiment of a global broker;

[0018] FIG. **8** illustrates one embodiment of a virtual data center overlay;

[0019] FIGS. **9-10** illustrate one embodiment of a distributed file system engine used for migrating a data center;

[0020] FIGS. **11**A-B illustrate one embodiment of a shadow storage system for migrating a data center;

[0021] FIGS. **12**A-C illustrate gateways and network infrastructure employed to migrate data centers in one embodiment of the invention;

[0022] FIGS. **13**A-B illustrate agents and data collection processes in accordance with one embodiment of the invention;

[0023] FIG. **14** illustrates additional details of one embodiment of a global broker and communication with selection engines;

[0024] FIG. **15** illustrates one embodiment of the invention which virtualizes and projects network connectivity and traffic management components;

[0025] FIG. **16** illustrates one embodiment of a traffic cloud engineering component which includes a priority-based traffic scheduler;

[0026] FIG. **17** illustrates one embodiment in which a service provider is interconnected to diverse cloud providers via a series of WAN components;

[0027] FIG. **18** illustrates one embodiment of the invention which utilizes Openstack to define virtual device controllers;

[0028] FIG. **19** illustrates termination points between various cloud providers and a service provider; and

[0029] FIG. **20** illustrates exemplary details showing how a service provider may be communicatively coupled through a data center hosting service to an external public cloud provider.

DETAILED DESCRIPTION OF PREFERRED EMBODIMENTS

[0030] Described below are embodiments of an apparatus, method, and machine-readable medium for cloud service selection and projection. Throughout the description, for the purposes of explanation, numerous specific details are set forth in order to provide a thorough understanding of the present invention. It will be apparent, however, to one skilled in the art that the present invention may be practiced without some of these specific details. In other instances, well-known structures and devices are not shown or are shown in a block diagram form to avoid obscuring the underlying principles of the present invention.

[0031] The embodiments of the invention described herein take advantage of the growing number of cloud service pro-

viders on behalf of those users migrating to the cloud. In particular, these embodiments include mechanisms to manage and move data centers independent of which cloud service provider is actually providing the cloud footprint. In one embodiment, the cloud footprint is an IaaS footprint; however, the underlying principles of the invention may also be implemented across data centers offering PaaS or SaaS services.

[0032] FIG. **1**A illustrates a high level architecture of a cloud analysis and projection service (CAPS) **100** in accordance with one embodiment of the invention. As described in detail below, the CAPS **100** enables a number of powerful models, including the ability to arbitrage diverse datacenters offered by cloud providers **121-124** to create the optimal price, performance, availability and/or geographical reach for virtual datacenters. In particular, one embodiment of the CAPS **100** analyzes the cost data, resource data, performance data, geographical reach data, reliability data, and/or any other pertinent cloud provider variables in view of the requirements specified by cloud users **111-115**. Once the relevant data has been evaluated, the CAPS **100** automatically selects one or more cloud provider on behalf of the cloud user. Alternatively, or in addition, the CAPS **100** may recommend a set of "candidate" cloud providers by performing cloud arbitrage, exploiting measurable differences between the cloud providers (e.g., striking a combination of matching deals that capitalize upon imbalances between cloud providers including the differences between cloud provider pricing, performance, service level agreements, or other measurable metrics). The end user may then select among the recommended cloud provider candidates.

[0033] As discussed in detail below, one embodiment of the CAPS **100** includes virtualization and projection logic to virtualize data center resources and enable data center migration once a decision has been made to migrate from one cloud provider to another (see, e.g., virtualization and projection component **231** illustrated in FIG. **2**A). Specifically, one embodiment of the CAPS **100** generates a virtualized or logical representation of all data center resources including (but not limited to) routers, switches, load balancers, WAN accelerators, firewalls, VPN concentrators, DNS/DHCP servers, workload/virtual machines, file systems, network attached storage systems, object storage, and backup storage, to name a few. This "virtual data center" representation reflects the atomic components that comprise the datacenter and manages the basic commissioning state of each logical device. The CAPS **100** then projects the virtual data center on the new physical data center by either translating the virtualized representation into a format necessary for implementing the physical data center, or by executing the virtual data center directly on the cloud provider (e.g., using a fully virtualized implementation such as discussed below with respect to FIG. **9**).

[0034] There are thousands of small cloud service providers who are just as capable of delivering IaaS to their customers as the large providers, but are seen as fragmented or regional. Consider the chart shown in FIG. **1**B that illustrates the current top North American cloud service providers. Note the logarithmic degrade of market share. By soliciting the long tail on the curve, the CAPS **100** becomes a "market maker" for aggregating data center services. In one embodiment, the CAPS **100** employs a brokerage model for buying and selling these data center services—i.e., arranges transactions between a buyer and a seller, and receives a commission when the deal is executed. This is beneficial to both cloud users **111-115** and cloud providers **121-124** because it enables movement throughout a fragmented market, and creates a conglomerated market place, without the need for outright acquisitions.

[0035] Cloud sellers **121-124** may bid into the CAPS platform **100** based on cost and other variables such as duration (a particular cost for a limit period), service level agreements, geographical location, network resources (suitability for certain distribution applications), and/or dedicated hardware resources for certain applications. This data may be further augmented with historical records from past customer transactions (e.g., using a customer feedback or other rating system). The notion of "arbitrage" is thus expanded to match the cloud buyer's requirements with a greater list of cloud seller characteristics. A simple example is price and duration. A seller may have capacity at a discounted rate but only for a specific duration (e.g., because another customer has reserved the capacity at a given point in the future). This may work for some buyers who either have more mobile datacenter architectures or only have short-term requirements for their datacenter. Overall, however, as the law of large numbers comes into effect, there will be an increased probability that the CAPS **100** can find buyers for each seller—thereby benefitting both.

[0036] As illustrated in FIG. **2**A, one embodiment of the CAPS **100** includes a plurality of components including a global broker **210**, a set of user selection engines **220-222** (e.g., one for each cloud user), and a virtualization and projection component **231**. The global broker **210** manages a database **211** of all available cloud providers **121-124** and the attributes of the data centers offered by the cloud providers. By way of example, the database **211** may include the cost data, resource data, performance data, geographical reach data, reliability data, and/or any other pertinent information associated with the data centers operated by the cloud providers **121-124**. The database **211** may be updated dynamically by the cloud providers **121-124** themselves or statically, by members of the CAPS **100**. For example, if a particular cloud provider has changed the cost structure for its data center resources for a particular duration, it may update the cost/duration for this change in the database **211**. Similarly, if a cloud provider has upgraded its hardware/software, or opened a new data center in a new location, it may update the database **211** to reflect the changes.

[0037] In one embodiment, the global broker **210** exposes an application programming interface (API) to enable the updates to the database **211**. The cloud providers **121-124** may then utilize the API to dynamically update the database **211**. This may be accomplished, for example, via CAPS software installed at each of the cloud providers **121-124** and/or via a Web server accessible by browser clients at the cloud providers. Static provider updates may also be implemented via the API.

[0038] In one embodiment, user selection engines **220-222** are executed to perform data center selections/recommendations for each cloud user. In the example shown in FIG. **2**A, selection engine **220** is executed to render data center selections/recommendations for User A; selection engine **221** is executed to render data center selections/recommendations for User B; and selection engine **222** is executed to render data center selections/recommendations for User C. In operation, each selection engine **220-222** is provided with the data center requirements for a corresponding cloud user (e.g., cost,

performance, reliability, geographical location, etc) and then identifies candidate cloud providers from the broker database which match those requirements.

[0039] As discussed in greater detail below, each selection engine **220-222** may generate a prioritized list of cloud providers **121-124** which match the user's requirements (e.g., with those at the top of the list being a closer match than those at the bottom of the list). The list may be provided to the end user as a set of cloud provider "recommendations" along with a comparative analysis explaining the reasoning for the recommendations. Alternatively, in one embodiment, a selection engine may automatically select one of the cloud providers on behalf of the user (and perform migrations between data centers as discussed below).

[0040] In one embodiment, the selection engines **220-222** receive updates from the global broker database **211** periodically and/or automatically to render new data center selections and/or recommendations. For example, if a particular cloud provider **121-124** has significantly reduced the cost of its services, then this may cause the selection engine to select the cloud provider and/or to place the cloud provider at the top of the prioritized list (thereby justifying a migration to the new cloud provider). The frequency with which a selection engine receives updates and generates prioritized list of data centers may vary depending on the implementation of the CAPS **100** and/or the preferences of each cloud user.

[0041] FIG. **4** illustrates additional details associated with a selection engine **220** including data center prioritization logic **420** for sending queries to the broker **210** indicating data center requirements (e.g., as specified by user requirements/preferences **425**). For example, based on user input **425**, the data center prioritization logic **420** may send a query specifying that it is only interested in data centers within a specific geographic region and having certain capabilities (e.g., load balancing, automatic failover capabilities, etc). As a result, the data center candidates provided by the broker **210** will be limited to those with the required parameters.

[0042] As indicated in FIG. **4**, the data center prioritization logic **420** may then prioritize the data center candidates based on various weighted components including an arbitrage component **401**, a performance component **402** and a reliability component **403**. While only three components are shown in FIG. **3**, various other/additional components may be included in the data center prioritization logic **420** while still complying with the underlying principles of the invention (e.g., such as geographical location, data center ratings from end users, etc).

[0043] In one embodiment, weights are assigned to each component **401-403** based on the user-specified requirements/preferences **425**. For example, if a particular cloud user is primarily interested in low cost data center services, then the arbitrage component **401** may be weighted more heavily than the performance component **402** and the availability component **403**. Another cloud user may also be interested in low cost but may specify minimum performance **402** and/or reliability **403** requirements. In such a case, the data center prioritization logic **420** will filter out those data centers which do not meet the minimum requirements and will then prioritize the remaining data center candidates based on cost. Yet another cloud user may be primarily concerned with data center reliability **403** and, as a result, the reliability component **403** may be weighted more heavily than the arbitrage **401** or performance **402** components. Various different/additional algorithms may be implemented by the data center prioritization logic **420** to generate the prioritized selections or recommendations **410** based on relative component weights.

[0044] Returning to FIG. **2**A, once a new cloud provider has been selected, the virtualization and projection component **231** manages the migration of the data center to the new cloud provider. As mentioned above, one embodiment of the virtualization and projection component **231** maintains a "virtual data center" representation of data center resources required by each cloud user such as routers, switches, load balancers, WAN accelerators, firewalls, VPN concentrators, DNS/DHCP servers, workload/virtual machines, file systems, network attached storage systems, object storage, and backup storage, to name a few. This "virtual data center" representation reflects the atomic components that comprise the datacenter and manages the basic commissioning state of each logical device (e.g., the user-specific configuration for each device). In one embodiment, the virtualization and projection component **231** maintains its own database **232** of all of the necessary data for managing/migrating each virtual data center. Alternatively, the virtualization and projection component **231** may rely on the global broker database **211** to store this data.

[0045] Once the new data center is selected, the virtualization and projection component **231** projects the virtual data center on the new data center. As mentioned, the projection to the new data center may involve translating the virtualized representation into a format necessary for implementing the physical data center (e.g., based on the specific hardware/software resources of the cloud provider) or by executing the virtual data center directly on the cloud provider (e.g., using a fully virtualized implementation such as discussed below with respect to FIG. **9**). Once the projection is complete, the old data center may be taken offline.

[0046] FIG. **2**B illustrates one embodiment of a method for selecting a new data center based on user-specified data center specifications and requirements and FIG. **2**C illustrates one embodiment of a method for migrating from one data center to another.

[0047] Turning first to FIG. **2**B, at **250**, the user enters the specifications for the data center. As used herein, the "specifications" comprise the particular components and architecture of the data center including, for example, the arrangements of routers, switches, load balancers, WAN accelerators, firewalls, VPN concentrators, DNS/DHCP servers, workload/virtual machines, file systems, network attached storage systems, object storage, and backup storage. In one embodiment, the virtualization and projection component **231** may provide the user with a graphical user interface (GUI) for graphically selecting the data center components and the interconnections between components (see, e.g., FIG. **8** and associated text). The GUI may be Web-based (e.g., provided via Web pages accessible via a browser) or may be implemented as a stand-alone application. In one embodiment, the virtualization and projection component **231** determines the data center specifications by asking the cloud user a series of questions related to the data center architecture. Alternatively, or in addition, the virtualization and projection component **231** may provide the user with a set of pre-constructed data center templates from which to select based on the user's data center requirements. Each of the templates may be associated with a certain set of required resources and/or have particular parameters associated therewith.

[0048] Regardless of how the user enters the data center specifications, at **251**, the virtualization and projection component builds the virtual data center representation using the specifications. As discussed above, in one embodiment, the virtual representation comprises an abstract representation of all required data center resources and architectural layout (e.g., interconnections between the resources). The virtual representation reflects the atomic components that comprise the datacenter and manages the basic commissioning state of each logical device.

[0049] At **252**, the user indicates the various factors to be considered for prioritizing data center candidates. As discussed above, this may involve associating weights with variables such as data center cost, performance, and/or reliability a data based on the user's preferences/requirements. At **253**, a set of data center candidates are identified based on the specifications and requirements. For example, as previously discussed, if a particular cloud user is primarily interested in low cost data center services, then the cost variable may be weighted more heavily than the performance and the availability variables. Various different prioritization algorithms may be implemented to generate the prioritized selections or recommendations based on relative component weights.

[0050] At **254**, a data center is selected from the identified data center candidates. In one embodiment, the selection is performed by the cloud user (e.g., after reviewing the prioritized list of candidates). In another embodiment, the selection is performed automatically on behalf of the cloud user.

[0051] Regardless of how the data center is selected, at **255**, the virtual data center is projected onto the selected physical data center. As mentioned, the projection to the new data center may involve translating the virtualized representation into a format necessary for implementing the physical data center (e.g., based on the specific hardware/software resources of the cloud provider) or by executing the virtual data center directly on the cloud provider (e.g., using a fully virtualized implementation such as discussed below with respect to FIG. **9**). Once the projection is complete, the data center may be placed online.

[0052] FIG. **2**C illustrates one embodiment of a method in which an existing data center is migrated to a new data center. At **260** data center updates are received and evaluated and, at **261**, a determination is made as to whether migration to a new data center is justified (e.g., based on price, performance and/or reliability considerations). As mentioned, the global broker **210** may receive continuous dynamic updates from cloud providers **121-124** and/or may be updated statically (i.e., by members of the CAPS **100**). As these updates are stored in the global broker **210**, each selection engine **220-222** may execute its selection policy to determine whether migrating to a new data center would benefit the end user. For example, the decision to migrate may be based on changes to the current data center and/or the other candidate data centers (e.g., changes to cost, SLA, tier of datacenter in addition to cost, time of day, performance for limited periods of time, availability, etc).

[0053] A graphical depiction of one embodiment of the decision-making process for selecting a new data center is illustrated in FIG. **3**. As illustrated, a migration event **370** may be generated upon detecting that the cost associated with the current data center has moved outside of a tolerant range (e.g., as indicated in cost arbitrage box **371**). The event may be a scheduled event (e.g., the cloud provider may provide advance notice of the cost change), in which case, the selection engine may employ a scheduler to trigger a migration event to move the data center at a particular point in time (e.g., the point at which the cost is anticipated to rise above the target range). As indicated in box **372**, the global broker may be continually updated via a projection feed with data related to all monitored data centers. The set of all monitored data centers are filtered by the selection engines based on screening criteria such as cost, performance, location, and/or availability to arrive at a set of candidate data center projections. One particular data center is then selected from the set of candidates based on an "event" such as the current projection failing to fall within the tolerant range, the difference between a candidate and a current projection rising above a threshold, and/or changes to the screening criteria. The end result is that an event is triggered (either automatically or manually by the cloud user) to migrate the data center.

[0054] Returning to FIG. **2**C, at **262**, the new data center is selected from the set of candidates and, at **263**, the virtual data center is projected onto the newly selected data center (e.g., using the various projection techniques described herein).

[0055] As illustrated in FIGS. **5**A-D, one embodiment includes a graphical user interface and a command line interface for creating and managing the virtual data center. In one embodiment, each of the virtual controllers employed in the virtual data center are represented by unique graphics. FIG. **5**A illustrates one particular set of graphics to represent virtual controllers including a virtual data center graphic **501**; a gateway graphic **502**; a network router graphic **503**; a network switch graphic **504**; a firewall graphic **505**; a load balancer graphic **506**; a WAN acceleration graphic **507**; a workload/virtual machine graphic **508**; a DNS server graphic **509**; a file system graphic **510**; a DHCP server graphic **511**; a backup storage graphic **512**; a network attached storage graphic **513**; a VPN concentrator graphic **514**; and an object store graphic **515**.

[0056] A brief description of each of the virtual controllers represented by these graphic images is set forth below. In one embodiment, access to the underlying resources is provided via a management interface exposed by the virtualization and projection component **231**.

[0057] The Virtual Datacenter **501** is a process that captures high level attributes of the projection such as geographical location, SLA, tier, etc. This is a non-operational object that is used to group geographic disparate datacenters at a top level view. The attributes of the virtual data center may include location, service level agreement, data center tier, pricing category.

[0058] The Gateway Router **502** is responsible for the public routes over the Internet. The attributes include WAN Configuration, Route Entries, Route Protocols, Interface monitoring, DNS Properties, Topology/Routing Information, and Affinity Rules.

[0059] The network router **503** is responsible for routing between all subnetworks within the virtual datacenter. Multiple network routers may be instantiated with different interfaces tied to different subnetworks, much like a real router. The attributes may include Network Configuration, Route entries, Route protocols, Interface monitoring, and Affinity rules.

[0060] The network switch **504** embodies the notion of a subnetwork. Interfaces connecting different devices to each other within a subnetwork are modeled by this entity. In cases where telemetry for each connected device is collected, the network switch can be the managed entity to identify the

usage and therefore cost and performance of the datacenter. The attributes may include Network Configuration, Monitoring, and Affinity Rules.

[0061] The firewall **505** is a feature that typically is provided by the cloud provider, but could be an additive feature offered by the CAPS **100** (either directly or through an App Store concept). The Firewall can provide a range of potential offerings including, but not limited to network address translation (NAT), distributed denial of service (DDOS) protection, and flow monitoring. Attributes may include Network configuration, Firewall Policies, Monitoring Policies, and Affinity Rules.

[0062] The load balancer **506** is a device used to map a number of identical workloads together for scale out purposes. The attributes may include Network configuration, Addressable End Stations, Balancing Policies, Monitoring, and Affinity Rules.

[0063] The WAN accelerator **507** is a service available for interconnecting datacenters over the WAN. This element may include devices such as Riverbed which offers deduplication compression algorithms. These services may be offered by cloud providers as a virtual workload to cloud users. Between two or more virtual datacenters, an instance of a WAN accelerator may be employed (one at each site) to compress data heading across the WAN. Attributes may include Network Configuration, End-Point-Configuration, SLA, Base User Privileges, Monitoring, and Affinity Rules.

[0064] The Workload/Virtual Machine **508** maintains the generic configuration of the OS image. It is responsible for transposing these images for a variety of VM formats such as VMDK, ISO, etc. By maintaining these images at all times, the migration process is greatly streamlined. The attributes may include CPU Class and Quantity, Memory, Local Storage, Operating System Image, Network Configuration, and Affinity Rules.

[0065] The DNS Server **509** provides a method for the virtual datacenter to offer naming both internal and external to the IaaS Datacenter. It should tie into both the naming services of the hosting IaaS service provider and the Siaras Global Directory/Broker. The attributes may include Domain Names, Addressing, Migration Features, Monitoring, and Affinity Rules.

[0066] The File System **510** may be associated with network attached storage (NAS). It may be a shared resource but can have some associated privileges, either through addressability or potentially user-based privileges. A core feature of the migration component of the file system includes the migration of data. In one embodiment, the virtual controller supports the ability to transfer data from one file system instance to another. Migration may be orchestrated at a higher layer than the virtual file system controller, but the controller should offer at minimum a "Sink" and "Source" mechanism for the transfer. As discussed in greater detail below, in one embodiment, a distributed file system (e.g., such as Hadoop) is employed that does not require the manual transfer of data. Each instance of the file system automatically binds to the existing nodes and downloads the necessary local data. The attributes of the file system may include Size, Network Configuration, SLA, Base User Privileges, Backup Policies, and Affinity Rules.

[0067] The DHCP Server **511** allows the datacenter provider to define addressing schemes and ACLs, and other controls over devices within the logical datacenter. The attributes may include Sub-Network Configuration, Monitoring, and Affinity Rules.

[0068] Backup storage **512** is a core attribute of any High Availability application and may be attributed features of the local IaaS service provider, or possibly a value add feature of the CAPS. In the later case, at issue would be the amount of data transferred out of physical datacenters, and the cost associated with them.

[0069] Network Attached Storage **513** may be a high performance storage methodology that can be available in Tier 1 IaaS Cloud datacenters or within private datacenters. These controllers are used to manage these resources. The attributes may include LUNs & Size, RAID Policies, Network Configuration, SLA, Base User Privileges, Backup Policies, and Affinity Rules.

[0070] The VPN concentrator **514** is the end station that remote clients will use to connect to the datacenter. VDI applications and other basic secure connectivity would utilize a VPN concentrator or act as a simple secure VPN end point. Attributes may include Network configuration, Firewall Policies, Monitoring Policies, and Affinity Rules.

[0071] IaaS Cloud providers may offer object storage capabilities, represented by object store graphic **515**. Optimally, the object store virtual controllers will offer a transformation function to map one object storage facility to another. It may be the responsibility of the end applications to utilize Cloud PaaS abstraction solutions, such as Chef or Cloud Foundry to deal with the API changes. In one embodiment, the CAPS' role is to ensure the move is done effectively, and the data is available for the new project to continue processing. The attributes may include Size, Network Configuration, SLA, Base User Privileges, Backup Policies, and Affinity Rules.

[0072] FIG. **5**B illustrates an exemplary series of commands **510** which may be executed by the virtualization and projection component **231** (e.g., via a command line accessible to the exposed management interface) to build a virtual data center. While a command line interface is illustrated for the purposes of explanation, the same set of commands may be executed in response to a user manipulating elements within a graphical user interface (e.g., as shown in FIGS. **5**C-D). In this example, a "create datacenter 'Bob'" command creates a virtual controller for a datacenter named "Bob" represented by graphic **501**. The command "create subnetwork 'main' 192.168.1.0/24 on 'bob'" creates a network switch virtual controller **504** under the virtual data center "Bob" and the additional set of "create" commands create a gateway virtual controller **502**, file system virtual controller **510**, and two virtual machine virtual controllers **508** under the network switch **504**. The resulting virtual data center is then projected to data center **520** via the "Project" command. As described herein, various different techniques may be employed for projecting the virtual data center to a physical data center. After the virtual data center has been successfully projected, a "Move" command may be executed to migrate the virtual data center to a new physical data center **521**.

[0073] FIG. **5**C illustrates an exemplary graphical user interface (GUI) "dashboard" for creating and managing virtual data centers. The network topologies for two datacenters, datacenter A **553** and datacenter B **554**, coupled together via a WAN accelerator are illustrated within a GUI window **551**. In one embodiment, the user may create, edit, and delete the virtual controllers within the displayed network topology by

selecting and dragging the graphical elements representing the virtual controllers. For example, virtual controller elements displayed within region **550** of the GUI may be selected and the configuration data associated with each of the elements may be edited. Alternatively, the user may directly select a virtual controller from the GUI window **551** to edit the variables associated with the controller.

[0074] A site status window **552** is also illustrated to provide data related to arbitrage (i.e., data center cost), performance, and reliability. Under the graphical arbitrage element, the user may access various cost information including a maximum specified cost, a target cost, and a current cost associated with the usage of each data center. In addition, under the graphical arbitrage element, the user may specify triggers and actions in response to changes in cost values (referred to in FIG. **2**D as "events"). For example, the user may specify that a migration should occur if the cost value rises above a particular threshold.

[0075] Under the graphical performance element, the user may review current performance measurements including network performance, CPU performance, and storage performance. The user may also may specify triggers and actions in response to changes in performance values. For example, the user may specify that the data center should be migrated if performance of any particular variable drops below a specified threshold.

[0076] Under the graphical fault management element, the user may access various fault/reliability data including different types of system alarms (e.g., critical alarms, major alarms, minor alarms, etc). Once again, the user may specify triggers and actions in response to changes in reliability. For example, the user may specify that the data center should be migrated if the number of critical or major alarms rises above a specified threshold.

[0077] In one embodiment, the management GUI shown in FIG. **5**C provides the following functions/features:

- **[0078]** the distributed datacenters should be visible over their respective geographies;
- **[0079]** the active projects show which IaaS service provider is offering service;
- **[0080]** the ability to define policies under which the virtual datacenters should be projected, including policies based on Location, Cost, Time, Performance, SLA, Tier, Replication;
- **[0081]** the alternative datacenters (monitored sites) should be visible to the end user;
- **[0082]** the ability to clearly see where issues exist including performance issues, cost issues, and availability issues (e.g. an IaaS provider who is offering limited times for their assets, might have a count down timer)
- **[0083]** the ability to plan moves ahead of time, and perhaps monitor that resources remain available at the destination location;
- **[0084]** the ability to clearly see where costs are within different datacenter instances, and where the costs are within a given datacenter (e.g., an hourly reporting mechanism may be maintained to facility financial forensics at the end of a month, quarter, or year); and/or
- **[0085]** the ability to configure policies around a datacenter move. This may include alarm notifications requiring manual intervention, and the ability to schedule migrations based on predetermined maintenance windows (e.g. if an arbitrage event occurs, move at 2 am the next morning). A special case might be that if an existing projection dies due to a failure within the datacenter, then move immediately.

[0086] FIG. **5**D illustrates the hierarchical virtual controller arrangement shown in window **551** in greater detail. As mentioned, the user may design a virtual data center simply by selecting and dragging the graphical elements representing each of the virtual controllers **560-599**. In the particular topology shown in FIG. **5**D, a gateway **560** is associated with Datacenter A and gateway **561** is associated with Datacenter B. A network router **563**, firewall **562**, and WAN accelerator **564** are logically positioned directly under gateway **560** and another network router **582**, firewall **583**, and WAN accelerator **581** are positioned directly under gateway **561** in the hierarchical arrangement. As illustrated, a dedicated WAN interconnect communicatively coupling the two WAN accelerators, **564** and **581** may be used to ensure redundancy and/or failover between the two data centers. In one embodiment, as discussed below, the WAN interconnect may be used to streamline the migration process (i.e., when migrating a virtual datacenter to a new physical datacenter).

[0087] A first set of network switches **565-567** are logically positioned beneath the network router **563** of Datacenter A and a second set of network switches **584-586** are logically positioned beneath the network router **582** of Datacenter B. Switch **565** couples a set of Apache servers to the local network comprised of a load balancer **568**, a set of workload/VM units **569-571** (for executing processing tasks), and a file system **572**. Switch **566** couples a second set of workload/VM units **573-576** for implementing a memory cache subsystem ("Memcache") and switch **567** couples another set of workload/VM units **577-579** and an object store **580** for implementing a database.

[0088] In the example shown in FIG. **5**D, a mirrored set of components is configured in Datacenter B. For example, switch **584** couples a set of Apache servers to the local network comprised of a load balancer **587**, a set of workload/VM units **588-590** (for executing processing tasks), and a file system **591**. Switch **585** couples a second set of workload/VM units **592-595** for implementing a memory cache subsystem ("Memcache") and switch **586** couples another set of workload/VM units **596-598** and an object store **599** for implementing a database.

[0089] In one embodiment, additional data center elements such as processing, networking, and storage resources may be added simply by clicking and dragging virtual controllers within the illustrated hierarchical architecture. For example, additional workload/VM controllers may be added under each respective switch to increase processing capabilities. Similarly, additional switches may be added under the routers **563**, **582** to add new subsystems to the datacenter topology.

[0090] In some of the embodiments described below, the file systems **572** and **591** are distributed file systems which have built in capabilities for maintaining synchronization between the two datacenters across the WAN interconnect (e.g., such as Hadoop).

[0091] As a specific example using the architecture shown in FIG. **5**D, web services may be offered by the groups of Apache servers load balanced by load balancing appliances **568**, **587**. These may be within a single subnetwork. The Memcache servers may form a second subnetwork to maintain active caches of their respective databases. The group of database servers **577-579**, **596-598** each operate from a common data store **580**, **599**.

[0092] In operation, when a URL request enters the datacenter through the Gateway **560**, **561**, it is screened by the Firewall **562**, **583**, and then forwarded to the Load Balancer **568**, **587** which redirects the request to one of the Apache servers. In a heavily loaded condition, the number of servers may be automatically spun up in this case. For example, ranges of servers may be defined, and triggers may be used to expand and contract those server pools. What is unique here is that the illustrated architecture is an actual logical datacenter that is orthogonal to any given cloud provider offering—thus making it inherently portable.

[0093] Returning to the above example, once the URL is processed, the active Apache server will forward the request to the Memcache servers. If there is a dirty bit in the Memcache (e.g. data is out of date), in one embodiment, the Memcache will respond right away (e.g., within 200 ms), with out-of-date data, rather than waiting multiple seconds for a refresh. When the event occurs, the Memcache will trigger a database query from the next bank of servers. In doing so, when the end user clicks refresh on their browser, they typically get the up-to-date data. In other words, leaving it to the end user to re-request data, gives the Memcache the necessary time to update "behind the scenes."

[0094] FIG. **6** illustrates various layers which are used in one embodiment of the virtualization and projection logic **231** to project a virtual data center to a cloud provider. In particular, an abstract, virtual datacenter representation **601** may be built using the GUIs shown in FIGS. **5**A-D and/or via a third party user interface **602** (e.g., using a GUI designed by a third party). In one embodiment, each object from the abstract GUI layer **601** (e.g., such as the graphical objects **560-599** shown in FIG. **5**D) maps to a particular controller within the virtual device controller layer **603**. Each virtual device controller comprises a virtual data structure containing the data required to implement the underlying piece of hardware (e.g., a router, switch, gateway, etc) as well as an interface to provide access to the data. For example, the interface may be implemented using representation state transfer (REST) or any other interface model.

[0095] The resulting set of virtual device controllers **603** may be mapped to corresponding physical devices within the projected datacenter **605** via a cloud mediation layer **604**, which may be implemented using a Cloud API (e.g., JClouds). In one embodiment, a separate "plugin" module is provided to map and/or translate the virtual device controller representation into a format capable of being implemented on the resources provided by the cloud provider. Consequently, in FIG. **6**, Plugin A is used to map and/or translate the virtual datacenter representation to Cloud Provider A and Plugin B may be used to translate and project the virtual datacenter representation to Cloud Provider B. Thus, when a new datacenter registers its services with the broker **210**, the underlying virtual datacenter representation does not need to be modified. Rather, only a new plugin is required to map and/or translate the existing virtual datacenter to the new physical datacenter. The plugins may be implemented on a server within the cloud provider premises (e.g., on the cloud provider LAN) and/or at the CAPS **100**.

[0096] Additional details associated with one embodiment of a global broker **210** are illustrated in FIG. **7**. As previously described, the global broker **210** includes a data center database **211** containing data related to each provider including, but not limited to, resource data (e.g., specifying the types of processing, networking, and storage platforms available), performance data (e.g., measured based on latency associated with processing tasks or network communication tasks), cost (e.g., in dollars per day or other unit of usage), geographical data (e.g., indicating geographical locations), and reliability date (e.g., based on the average number of significant alarms over a particular period of time).

[0097] In one embodiment, various application programming interfaces (API) are exposed to provide access the data center database **211** including a cloud provider interface **701**, a cloud user interface **702**, a provisioning interface **703**, and a management interface **704**. In one embodiment, each interface includes a set of commands to perform operations on the database (e.g., to create records, delete records, modify records, etc.). Cloud providers are provided with access to the data center database **211** via the cloud provider interface **701**, cloud users are provided with access via the cloud user interface **702**, database provisioning is performed via the provisioning interface **703**, and management operations are provided via the management interface **704**. In addition, a messaging bus **705** is provided which allows all cloud users to maintain up-to-date views of available resources (e.g., by providing data center results to a queue to which the cloud users listen as discussed below).

[0098] In one embodiment, the management interface **704** is used by the CAPS **100** to perform any and all house keeping functionality required for the sustained execution of the system. Functions that may be supported by the management interface include the ability to:

[0099] view and modify data related to active Buyers and Sellers;

[0100] provide access control lists (ACLs) that limit the accessibility of Buyers and Sellers;

[0101] monitor the throughput of the message bus **705**;

[0102] spin up or down additional computational and storage elements to handle different loads;

[0103] manage SaaS-based datacenter manager clients;

[0104] drop into low-level command line debug and diagnostics tools;

[0105] shut down the system and prepare for a move to a new data center;

[0106] see all running instances of the system, including those running in multiple datacenters;

[0107] manage failover solutions; and/or

[0108] statically manage customers for debugging purposes.

[0109] In one embodiment, the provisioning interface **703** allows the CAPS **100** to provide updates to the data center database **211** (e.g., adding new entries for vetted datacenters and/or partner datacenters and removing data centers which are no longer in service or undesirable). In the non-partner category (e.g. IaaS providers that are not actively aware that the CAPS **100** utilizing their resources), it is up to the CAPS **100** to provide updates to the data center database **211** as changes occur.

[0110] In one embodiment, configuration techniques such as data center "Zones" (as used by Amazon) are not made transparent to cloud users. For example, Zones may simply be identified as different datacenters. Consequently, one virtual datacenter may be employed in a Zone of a cloud provider (e.g., Amazon), and one virtual datacenter may be employed in a different cloud provider (e.g., Rackspace). A corollary to this is that cloud users may be allowed to specify that they wish to have a single provider, but different sites/zones (e.g., using affinity rules indicating affinity for certain sites).

[0111] In one embodiment, the functions included by the provisioning interface **703** include the ability to:

[0112] add/remove/view IaaS Datacenter (Seller) records;

[0113] update static/dynamic Seller records;

[0114] force a push of records to Buyers;

[0115] create a new Buyer and optionally the associated SaaS infrastructure (using Siaras template); and/or

[0116] view statistics and reports relating to registered Buyers and Sellers

[0117] In one embodiment, the cloud provider interface **701** is open to cloud providers wishing to post details of their available services. The API may be tied to a SaaS Web Portal for manual entry and open for M2M integration with automated systems. The functions of this interface may include the ability to add/remove/view/update cloud provider records.

[0118] In one embodiment, the cloud user management interface **704** is open to cloud users running managed virtual datacenters. These systems may either be running in clouds themselves (as an SaaS) or as an enterprise application. The purpose of this interface is to provide a methodology for the managed virtual datacenters to report on their current execution experiences. The interface may be implemented as a closed system that is activated only through the management software provided to cloud users by the CAPS **100** (i.e., customers do not have direct access to this interface). In one embodiment, the functions included in the cloud user interface include the ability to:

[0119] report updates to the observed performance of a virtual datacenter;

[0120] report any outages observed by specific service provider;

[0121] report any failures to configure within virtual datacenter (e.g. incompatibilities between our mediation systems, or lack of reported capabilities or available resources); and/or

[0122] provide a customer satisfaction reporting methodology and trouble ticket mechanism.

[0123] In one embodiment, the global broker is responsible for redirecting DNS entries for cloud users. In doing so, migration of datacenters may be instantly updated. In addition, one embodiment of the global broker **210** is designed to support scale. In particular, any interface requiring multiple clients supports scale out architectures.

[0124] As mentioned above, the virtualization and projection component **231** may project the virtual data center on a physical data center by either translating the virtualized representation into a format necessary for implementing the physical data center (i.e., a plugin which converts the abstract data into a format usable by the data center's physical resources), or by executing the virtual data center directly on the cloud provider.

[0125] FIG. **8** illustrates the latter scenario which utilizes a fully virtualized implementation comprising a virtual data center overlay **800** running on a plurality of virtual machines **821-826** provided by a generic cloud provider **830**. Thus, this embodiment comprises a fully re-virtualized cloud layer in which each component **801-806** of the virtual datacenter **800** may be projected on any cloud provider—from the most generic to the most sophisticated. In the specific example shown in FIG. **8**, each component runs on a different VM exposed by the generic cloud provider **830**. In particular, the virtual gateway **801** runs on VM **821**. Three different Kernel VMs **802**, **804**, and **805** comprising virtual kernels run on VMs **822**, **824**, and **825**, respectively. As illustrated, operating systems or other software images **811**, **812**, and **813** may be executed on top of the kernel VMs **802**, **804**, and **805**, respectively. A virtual switch **803** runs on VM **823** and a virtual file system **806** runs on VM **826**.

[0126] As indicated in FIG. **8**, each of the virtual components **801-806** forming the virtual data center **800** may communicate using the Layer 2 Tunneling Protocol (L2TP) tunnels, Secure Sockets Layer (SSL) tunnels, or another secure inter-process protocol to create secure tunnels between components. In addition, as illustrated, the virtual gateway **801** communicatively couples the other components **802-806** to a public network via a public IP interface provided via VM **821**.

[0127] There are several benefits to utilizing a virtual data center overlay. First, because no translation is required, the virtual data center overlay may be deployed seamlessly on any physical data center capable of providing a consistent VM. A more homogenous SLA and security profile may be imposed over various tiers of datacenter services. In addition, far greater control and visibility of the actual datacenter may be provided, resulting in a more homogenous offering over time. For example, agents could be included in every entity (e.g. vGateway, KVM, vSwitch, vFileSystem, etc), to continuously measure performance and cost.

[0128] As mentioned above, the file systems **510**, **572**, **591** implemented in one embodiment of the invention are distributed file systems which have built-in capabilities for maintaining synchronization between the two (or more) datacenters across a WAN interconnect. FIG. **9** illustrates one such embodiment which includes a first virtual data center **910** utilizing a file system **920** on cloud provider **900** and a second virtual data venter **911** utilizing an associated file system **930** on cloud provider **901**. Each of the file systems, **920** and **930**, include distributed file system engines, **923** and **933**, respectively, for synchronizing a location portion **921** of file system **920** with a remote portion **932** of file system **930** and for synchronizing a location portion **931** of file system **930** with a remote portion **922** of file system **920**. As a result of the synchronization, any changes made to local portion **921** of file system **920** are automatically reflected in remote portion **932** of file system **930** and any changes made to local portion **931** of file system **930** are automatically reflected in remote portion **922** of file system **920**. In one embodiment, the "local" components **921**, **931** of the file systems are those components which are created, edited and/or otherwise accessed locally by the respective virtual data center **910**, **911**. In contrast, the "remote" components are those which are created, edited and/or otherwise accessed from a different virtual data center **910**, **911**.

[0129] In one embodiment, the distributed file system engines **923**, **933** are Hadoop Distributed File System (HDFS) engines and the local and remote portions are implemented as Hadoop nodes. HDFS is a distributed, scalable, and portable file-system which stores large files across multiple machines and achieves reliability by replicating the data across multiple hosts. As a result Hadoop instances do not require RSID storage on hosts. Data nodes can talk to each other to rebalance data, move copies around, and maintain a high replication of data. In the implementation shown in FIG. **9**, the Hadoop protocol may be used to synchronize the local **921** and **931** and remote **932** and **922** portions, respectively, of the file systems **920**, **930**. It should be noted, however, that the

underlying principles of the invention are not limited to any particular distributed file system protocol.

[0130] In one embodiment of the invention, a distributed file system (such as described above) is used to streamline the data center migration process. For example, as illustrated in FIG. **10**, if the user chooses to migrate from cloud provider **900** to a new cloud provider **902**, then once the new projection is complete (e.g., using the virtual data center techniques described herein), the underlying file system **940** for the virtual data center **912** may be populated using the protocol of the distributed file system engine **943**. For example, in FIG. **10**, the local portion **941** of the file system **940** may be populated from the remote portion **932** of the existing file system **930** of cloud provider **901**. Similarly, the remote portion **942** of the file system **940** may be populated from the local portion **931** of the existing file system **930** of cloud provider **901**. In one embodiment, the entire contents of the local **941** and remote **942** portions of file system **940** do not need to be completely populated before the cloud provider **902** is put online. Rather, the local **941** and remote **942** distributed file system nodes may be created and populated during runtime (after the virtual data center **912** is placed online). When a request for data is received at local node **941** or remote node **942**, which is not available locally, the data may be retrieved from remote node **932** or **931** respectively, thereby populating the local node **941** and remote node **942** during runtime. As a result, the virtual data center **912** may be spun up on cloud provider **902** much more efficiently than in prior implementations (i.e., where all of the file system **940** data is required in advance).

[0131] FIG. **11**A illustrates another embodiment which includes a could provider **1100** is running a virtual data center **1110** coupled to a file system **1120** managed by a distributed file system engine **1130** (as in the prior embodiment). In this embodiment, a shadow storage system **1101** is used for storing a shadow copy **1121** of the virtual data center file system **1120**. A distributed file system engine **1131** communicatively coupled to the distributed file system engine **1130** of the virtual data center is configured to maintain a synchronized, shadow copy **1121** of file system **1120**. In one embodiment, as changes are made to the local file system **1120**, the distributed file system engines **1130-1131** coordinate to reflect those changes in the shadow file system **1121**. The difference in this embodiment is that the shadow storage **1101** is not itself a data center accessible by end users. Rather, it is used only for shadow storage.

[0132] In one embodiment, the shadow storage system **1101** is used to streamline the data center migration process. For example, as illustrated in FIG. **11** B, if the user chooses to migrate from cloud provider **1100** to a new cloud provider **1102**, then once the new projection is complete (e.g., using the virtual data center techniques described herein), the underlying file system **1122** for the virtual data center **1111** may be populated using the protocol of the distributed file system engine **1132**. For example, in FIG. **11** B, file system **1122** may be populated from the shadow copy of the file system **1121** stored on the shadow storage system **1101**.

[0133] As in the embodiments described above, the entire contents of the file system **1121** need not be completely populated to file system **1122** before the cloud provider **1102** is placed online. Rather, the distributed file system node **1122** may be created and populated during runtime (after the virtual data center **1111** is placed online). When a request for data is received at the file system **1122** which is not available locally, the data may be retrieved from the shadow file system **1121**, thereby populating the file system **1122** during runtime. As a result, the virtual data center **1111** may be spun up on cloud provider **1102** much more efficiently than in prior implementations.

[0134] As illustrated in FIG. **12**A, in one embodiment, the cloud analysis and projection service **1000** may be implemented using its own gateway devices, **1250** and **1251**, at different data centers **1200** and **1210**, respectively. The gateways **1250** may then be used to establish a secure connection such as a virtual private network (VPN) connection between the data centers when performing intra-data center migrations. In one embodiment, the VPN connection comprises a purchased WAN accelerator link such as those which provide deduplication compression algorithms (e.g., Riverbed).

[0135] Two tenants are illustrated in data center **1200**: tenant **1201** with virtual data center **1230** and file system **1220** and tenant **1202** with virtual data center **1231** and file system **1221**. An additional tenant **1203** with virtual data center **1232** and file system **1222** is located within data center **1210**. As used herein, the "tenants" are cloud users who have signed up for the virtual data center services described herein. In the illustrated example, the dedicated VPN connection is used to migrate the virtual data center **1230** of tenant **1201** from data center **1200** to data center **1210**. In this case, because the VPN connection is a dedicated link between the data centers (purchased by the CAPS **100**), no additional cost is incurred by the tenant **1201** for the data center migration.

[0136] As illustrated in FIG. **12**B, in one embodiment, in addition to providing local gateway devices **1300-1302** at each of the cloud providers A-C, the cloud analysis and projection service **1000** may build/purchase a network fabric (e.g., a dedicated network infrastructure/backbone, etc) comprising additional gateway/router devices **1310-1313** to support high speed, secure interconnections between the various providers.

[0137] In one embodiment, the cloud analysis and projection service **1000** maintains a provider connection table **1290** such as shown in FIG. **12**C to determine whether a dedicated, high speed connection exists between the data centers of any two cloud providers. In the particular example shown in FIG. **12**C, providers **1-3** are all interconnected via a dedicated network infrastructure (e.g., maintained/purchased by the CAPS **100**) whereas no such connection exists to cloud providers **4** and **5**. In one embodiment, the cloud analysis and projection service **1000** and/or selection engines **220-222** may consult the table **1290** when rendering data center recommendations or selections. For example, if data centers **3** and **5** have similar cost, performance, and reliability characteristics but data center **3** is coupled to the current data center via a dedicated, high speed connection, then the selection engine may recommend migrating to data center **3** above data center **5**.

[0138] As mentioned above, the Global Broker **210** may be updated dynamically by feedback from the cloud providers which may include cost, performance and/or reliability updates. FIG. **13**A illustrates one embodiment in which performance and/or reliability updates are dynamically provided by agents **1320-1321** executed within the virtual data center **1301** for a particular tenant. In this particular embodiment, an agent **1320-1321** is inserted into each workload **1310-1311**, respectively, being executed on the resources of the virtual data center **1301**. The agents **1320-1321** monitor the execution of their respective workloads **1310-1311** and collect per-

formance data. For example, the agents **1320-1321** may measure the time required to complete execution of program code by the workloads **1310-1311** and/or may ping other components within the virtual data center (e.g., the file system **1325**) and measure the time taken to receive a response from each component. This information may then be reported back to the global broker **210** via a gateway **1330** and used to calculate a normalized performance measurement for the cloud provider **1300**.

[0139] FIG. **13**B illustrates another embodiment in which a separate data collection workload **1360** is executed in parallel with the other workloads **1350-1352** within the virtual data center **1301** to collect performance and/or reliability data. The data collection workload **1360** may ping the other workloads **1350-1352** and other data center components such as the file system **1353** and measure the amount of time taken to receive a response (with longer time periods indicating relatively lower performance). The data collection workload **1360** may also calculate the amount of time required to execute its own program code (e.g., inserting tags in the program code or performing other program tracing techniques). Because it is executed as another workload on the resources of the cloud provider **1300**, the performance associated with its own execution is indicative of the performance of the other workloads **1350-1352**. It may then feed back the performance results to the global broker **210** via the gateway **1330**.

[0140] In another embodiment, the gateway **1330** itself may collect performance data from the workloads **1350-1352** (e.g., pinging the workloads as described above) and feed the resulting data back to the global broker **210**.

[0141] In both of the embodiments shown in FIGS. **13**A-B, the agents **1320-1321** or data collection workload **1360** may also monitor the reliability of the virtual data center. For example, if a response to a ping is not received after a specified period of time, then a determination may be made that the component being measured has become unavailable. This reliability information may then be transmitted to the global broker **210** via the gateway **1330**.

[0142] As mentioned above, in one embodiment, the CAPS **100** architecture may be used as an online marketplace for buying and selling data center services may be built using the CAPS **100** architecture. Moreover, the marketplace is not limited to buying by cloud users and selling by actual cloud providers. Rather, in one embodiment, any user or entity may buy and sell data center services in the open marketplace. Thus, a particular cloud provider may purchase data center services from another cloud provider (including a virtual provider as discussed below) to meet demand. Conversely, a cloud user may sell data center services to another cloud user or to another cloud provider. For example, a particular cloud provider may offer a sale on data center services several months in advance of anticipated usage of the data center services (e.g., selling in June for data center usage in December/January). Using the open marketplace provided by the CAPS **100** another user or cloud provider may purchase these services and subsequently re-sell them (e.g., at a premium or a loss, depending on the going rate for the services in December/January). In this manner, a futures market for data center services is established by the CAPS **100**.

[0143] FIG. **14** illustrates how the global broker **210** and data center database **211** may be configured to enable such an online marketplace. As previously mentioned, the data center database **211** contains up-do-date records for each of the cloud providers registered with the CAPS **100** which include resource data, performance data, cost data, geographical location data, reliability data, and any other data which may be pertinent to a cloud user. Record **1401** is associated with and includes all of the current information for Amazon Web Services (AWS). In contrast, record **1402** represents a user (or another cloud provider) who has purchased AWS data center services at a particular price for some specified period of time (identified as a virtual AWS (vAWS) database record). Returning to the example mentioned above, this user may have purchased data center services from AWS for December/January several months in advance but does not intend to use these data center services (or perhaps purchased with the intent of using only a portion, or realized some time after purchasing that he/she would not require all of the purchased services). In one embodiment, any user who has purchased future (or current) rights to data center services may register the availability of these services within the data center database **211** of global broker **210**. The global broker **210** may then include these services in response to database queries generated from the various selection engines **220-222**. If the cost of these services is lower than the current market price (all other things being equal), then the selection engines **220-222** will recommend/select these services over those offered at the current market price (thereby resulting in a profit to the seller, assuming that the seller purchased at a lower price).

[0144] FIG. **14** illustrates additional details associated with the communication between the selection engines and the global broker **210**. In particular, query parameters **1405** may be sent from each of the selection engines **220-222** to a database query engine **1400** which then queries the data center database **211** using the parameters. For example, selection engine **220** may be interested in data centers located in New York and Japan; selection engine **221** may be interested in data centers located in California; and selection engine **222** may be interested in data centers located in Europe. In response, the database query engine may perform a query (or a series of queries) and retrieve from the database **211** all data centers meeting the specified criteria. In one embodiment, the results are the "candidate" data centers mentioned above (i.e., those meeting some minimum initial criteria; in this case based on geographic location).

[0145] In one embodiment, the results of the database query engine **1400** are provided as entries in a queue **1410** and each of the selection engines read/filter the entries from the queue **1410**. In one embodiment, a producer/consumer architecture is implemented in which the database query engine **1400** acts as a producer (writing new entries to the queue) and the selection engines **220-222** act as consumers of the queue (also sometimes referred to as "listeners" to the queue). Returning to the above example, selection engine **220** will only retrieve those entries from the queue **1410** associated with data centers in New York and Japan; selection engine **221** will only retrieve those entries from the queue **1410** associated with data centers in California; and selection engine **222** will only retrieve those entries from the queue **1410** associated with data centers in Europe. The various selection engines **220-222** may then perform filtering/weighting operations as discussed above, to further filter the candidate data centers to arrive at recommendations or selections on behalf of the cloud user (e.g., filtering based on cost, performance, reliability, etc). Although not explicitly shown in FIG. **14**, each selection engine may generate queries over the queue to

retrieve only those entries which are relevant to its search (e.g., New York and Japan for selection engine **220**).

Virtualization And Projection of Network Components

[0146] In addition to the techniques described above for virtualizing and projecting data centers, one embodiment of the invention applies these virtualization and projection techniques to implement network connectivity for network service providers, public/private data centers, and/or end users. In particular, as described below, one embodiment of the invention includes techniques for orchestrating and routing multi-chain WAN segments for interconnecting service provider public clouds, managed private clouds, external public clouds, and enterprise clients. In addition, one embodiment of the invention includes an apparatus and method for performing traffic engineering techniques at edge points between diverse cloud types, thereby ensuring that application-specific bandwidth (and other) requirements are met. Finally, one embodiment of the invention includes techniques for sharing dedicated public cloud connectivity by performing traffic engineering techniques.

[0147] As illustrated in FIG. **15**, the embodiments of the invention may be implemented within the context of the virtualization and projection engine **231** within the cloud analysis and projection service **100** described in detail above. In one embodiment, the virtualization and projection component **231** maintains a virtualized representation of both data center resources and network connection resources required to interconnect and manage the various clouds including routers, gateways, traffic engineering components, switches, load balancers, WAN accelerators, firewalls, VPN concentrators, DNS/DHCP servers, workload/virtual machines, file systems, network attached storage systems, object storage, and backup storage, to name just a few. This virtualized network representation reflects the atomic components that comprise the physical network connectivity and manages the basic commissioning state of each logical network device (e.g., the user-specific configuration for each device). In one embodiment, the virtualization and projection component **231** maintains its own database **232** of all of the necessary data for managing/migrating each virtual data center and each virtual network component.

[0148] As illustrated, a graphical user interface (GUI) dashboard **1560** such as that described above with respect to FIG. **5**C may be used to configure and manage the virtualized network resources represented by a plurality of virtual device controllers **1550**. In one embodiment, the virtual device controllers **1550** include the various types of virtual datacenter controllers **1509-1509** described above which provide a generic/virtual representation of the processing resources, storage resources, platform resources, etc, required for a particular data center implementation. In addition, the virtual datacenter controllers in FIG. **15** include virtual traffic engineering-as-a-service (TEaaS) controllers **1505-1506** providing a virtual representation of traffic engineering components **1515-1516** implemented between managed private/public clouds **1590**, public clouds **1591** (including service provider managed public clouds and external public clouds), and enterprise networks (not shown). The virtual datacenter controllers also include virtual WAN-as-a-service (WANaaS) controllers **1501** providing a virtual representation of WAN components **1511** such as network connection chains interconnecting the various cloud endpoints. As mentioned above, In particular, the user may create, edit, and delete the virtual controllers **1550** within the displayed network topology by selecting and dragging the graphical elements representing the virtual controllers.

[0149] Once a set of datacenter requirements and network connectivity and management requirements have been specified via the controllers, the virtualization and projection component **231** projects the virtual data centers **1508-1509** to the physical data center components **1518-1519**, respectively; projects the virtual TEaaS components **1505-1506** to the physical traffic engineering components **1515-1516**, respectively; and projects the WANaaS components **1501** to the physical WAN components **1511**, respectively. As mentioned, the projection to the new data center and network components may involve a mediation/translation layer **1551** translating the virtualized representation into a format necessary for implementing the physical data center and network components (e.g., based on the specific hardware/software resources and networking equipment of the cloud providers and service providers). Alternatively, or in addition, the virtual data center and network components may be executed directly on the cloud provider and/or service provider using a fully virtualized implementation such as discussed above with respect to FIG. **9**. Once the projection is complete, the various clouds **1590-1591** will be interconnected over an intercloud WAN **1592** as specified by the virtual WANaaS controllers **1501** with network traffic managed at the cloud endpoints as specified by the virtual TEaaS controllers **1505-1506**.

[0150] As illustrated in FIG. **16**, in one embodiment, the traffic engineering components **1516** include a priority based traffic scheduler **1610** for managing incoming and/or outgoing packet streams based on priority. For example, a service provider may implement a private or public cloud **1590-1591** to host a plurality of different applications **1601-1604**, each of which may have different network requirements such as minimum bandwidth and latency requirements. As such, the priority-based traffic scheduler may be programmed with these requirements and responsively manage a set of input/output queues **1611** to ensure that the bandwidth/latency requirements are met for each application **1601-1604**. For example, if one of the applications comprises a voice/media application which requires a **100** megabit bandwidth whenever it runs, then the priority-based traffic scheduler **1610** will manage the queues **1611** to ensure that this particular application always has access to a minimum **100** meg bandwidth. It may accomplish this by throttling the queues for other applications with less stringent requirements (e.g., applications which perform data backups).

[0151] In one embodiment, a service provider may utilize the traffic engineering components **1516** and WAN components **1511** described herein to interconnect different data center resources including those located within internal private clouds and external public clouds. For example, a service provider may offer its enterprise customers a single integrated cloud service which it will then implement using the techniques described herein to connect and coordinate between its internally-managed cloud resources and one or more external private/public cloud offerings **1590-1591**. The service provider may lease the external cloud offerings from the external cloud provider and configure the traffic engineering components **1516** in accordance with the lease arrangements. For example, the priority-based traffic scheduler **1610** may manage each of the queues **1611** based on the total available

bandwidth of the data link connecting the service provider and private/public cloud **1590-1591**. For example, the priority-based traffic scheduler may ensure that each application **1601-1604** hosted for each enterprise customer of the service provider is allocated the bandwidth that it needs by intelligently filling and reading packets from each of the queues **1611**. This may be accomplished in part by ensuring that no single application is permitted to monopolize all of the available bandwidth.

[0152] FIG. **17** illustrates one particular arrangement in which a service provider **1750** offers cloud services to an enterprise customer **1710** and coordinates between service provider-managed public clouds **1720**, service provider managed private clouds **1730**, and external public clouds **1740** (e.g., such as Amazon Web Services) using the virtualization and projection techniques described herein. In this specific example, the service provider public cloud **1720** includes virtual data centers **1721-1722**, the managed private cloud **1730** includes data centers **1731-1732**, and the external public cloud **1740** includes virtual data centers **1741-1742**, connected to the service provider via a cloud point of presence (POP) **1790**. As illustrated, the service provider may seamlessly interconnect each of the different types of clouds by specifying WAN components **1511** comprising a plurality of network chains **1700-1702** and may manage the bandwidth, latency, and other requirements for each enterprise customer **1710** using the traffic engineering components **1515-1516** described herein. Each of the chains **1700-1702** may specify an interconnection between each of the various different clouds **1720**, **1730**, **1740** and the enterprise client **1710** and each of the traffic engineering components **1515-1516** may specify bandwidth/latency requirements for each enterprise customer **1710** and, more specifically, for each application hosted for each enterprise customer **1710** (as discussed above with respect to FIG. **16**). Returning to FIG. **15**, all of the configuration information for interconnecting the chains **1700-1702** and implementing the traffic engineering components **1515-1516** may be specified via the (GUI) dashboard **1560** (e.g., by defining a set of virtual TEaaS controllers **1505-1506** and virtual WANaaS controllers **1501** and then translating using the mediation/translation layer **1551** to implement the traffic engineering components **1515-1516** and WAN components **1511**, respectively).

[0153] FIG. **18** illustrates additional details for embodiment of the invention in which the virtual device controllers **1550** are implemented using OpenStack which (as known by those of skill in the art) comprises a free, open-source cloud design platform comprising a series of interrelated "projects" that may be used to control pools of processing, storage, and networking resources through a web-based dashboard, command-line tools, or a RESTful API. Thus, in one embodiment, the TEaaS controllers **1505-1506** and WANaaS controllers **1501** described above may be implemented as Openstack projects, along with various other Openstack virtual components including a Swift object storage component **1801** for defining a scalable, redundant storage system; a Cinder block storage component **1802** for defining persistent block-level storage devices for use with OpenStack compute instances; a Nova compute component **1803** designed to manage and automate pools of computer resources; a Neutron networking component **1804** for managing networks and IP addresses; and a Glance images component for discovery, registration, and delivery services for disk and server images. All of the foregoing components may be used to define the requirements for each data center. For example, each component may be translated by the mediation layer into proxy components **1810**, **1811** for execution on physical or virtual data center platforms **1860**, **1862**, respectively.

[0154] Additional components illustrated in FIG. **18** include a Ceilometer component **1806** for providing telemetry services (e.g., a single point of contact for billing systems); a Keystone identity component **1807** for provides a central directory of users mapped to the services they can access; a Heat component **1808** for orchestrating multiple composite cloud applications using templates; and a Horizon dashboard component **1809** for providing administrators and users a graphical interface to access, provision, and automate cloud-based resources (e.g., analogous to the dashboard GUI shown in FIG. **5**C).

[0155] It should be noted, however, that while an Openstack implementation is shown in FIG. **18**, the underlying principles of the invention are not limited to an Openstack implementation.

[0156] As illustrated, the TEaaS controllers **1505-1506** may be translated by the mediation layer to proxy components **1820-1821** which may then be executed on top of virtual or physical traffic engineering platforms **1840-1841**, respectively. In addition, the WANaaS controller(s) **1501** may be translated by the mediation layer to chain components **1830-1831** specifying physical or virtual routing platforms such as routers **1850-1851** which form interconnections between the various datacenters **1860**, **1862** across the service provider network **1861**. IN the illustrated embodiment, a first router **1850** is implemented on a physical machine platform and a second router **1851** is implemented on a virtual machine platform.

[0157] FIG. **19** illustrates an exemplary embodiment in which the techniques described herein may be used to implement a variety of different network interconnections for a service provider **1900** including a first interconnection point (A) with an enterprise network **1910**; a second interconnection point (B) with a managed private cloud **1930**; a third interconnection point (C) to a data center **1920** coupled through a virtual network **1951** and virtual switch **1950**; and a fourth interconnection to a direct connect exchange point (D) **1960**. In this example, all of the intermediate interconnections needed to form the chains between each of the points A, B, C, and D may be defined by a plurality of virtual WANaaS controllers **1501** discussed above. For example, each chain may define the various switches **1901-1903** and the platform to communicate across the Q-in-Q virtual LAN tunnel **1905** for connecting to the direct connect exchange point **1960**.

[0158] FIG. **20** illustrates additional details of how the service provider endpoint (A) in FIG. **19** may be mapped all the way through an external data center hosting service **2001** into a tenant VPC **2030** of an external public cloud provider **1862**. The Q-in-Q virtual LAN tunnel **1905** is implemented in this embodiment, but the same principles may be applied to a Multiprotocol Label Switching (MPLS) or Border Gateway Protocol (BGP)-VPN tunnel **2004**.

[0159] The Q-to-Q tunnel terminates at point D_0 at a border network gateway (BNG) device **2010** managed as a service provider colocation (colo) platform **2007** within the data center hosting service **2001**. In this example, the service provider may lease a set of computing/networking resources from the data center hosting service **2001** (e.g., such as Equinig™) to implement the service provider colo **2007**. In one embodi-

ment, the BNG device **2010** translates between the Q-in-Q virtual LAN tunnel and a generic routing encapsulation (GRE) over IP protocol. In one embodiment, this involves extracting an STag from the incoming data packets, using the STag to identify the customer and generating GRE packets to identify the corresponding tenant **2030** within the external public cloud provider **1862** (e.g., such as Amazon Web Services™ in one embodiment). In one embodiment, the service provider **1900** has an account with the data center hosting service **2001** that is tied to this direct connect link.

[0160] In one embodiment, the GRE over IP packets are routed from an external public cloud provider colo **2015** at the data center hosting service **2001** over a fiber connection **2020** to a direct connect manager **2025** implemented at the external public cloud provider **1862**. In one embodiment, the direct connect manager **2025** is defined by a implemented as a traffic engineering component such as described above. For example, a separate instance of the direct connect manager **2025** may bind to each instance of a tenant VPC **2030** and perform traffic engineering operations (such as described above with respect to FIG. **16**) to ensure that the available bandwidth across the Q-in-Q tunnel is used effectively. For example, the service provider may pay for a different amount of bandwidth across the Q-in-Q pipe **1905** for each tenant and the direct connect manager **2025** may ensure that the allocated bandwidth is not consumed by a single application. In one embodiment, a radius server **2005** is used to identify each tenant and determine the bandwidth allocated to that tenant. For outgoing data connections, the service provider BNG **2010** may form a virtual router **2002** that terminates the GRE tunnel and maps to an STAG VLAN over the Q-in-Q pipe **1905** (e.g., stripping out the GRE headers and repackaging them using STAGs at D**0**-D**1** in FIG. **20**). Thus, the service provider is effectively connected from point A in FIG. **19** to each individual tenant VPC **203** within the external public cloud provider and is provided with the inherent ability to perform traffic engineering based on the requirements of each tenant application and the available bandwidth.

[0161] In one embodiment, all of the components within the service provider colo **2007**, including the service provider BNG **2010**, as well as the various direct connect managers **2025** within the external public cloud provider **1862** are defined using virtual device controllers **1550** within the virtualization and projection component **231** illustrated in FIG. **15**. The mediation/translation layer **1551** then translates the virtual description of each required component to operate within the specific physical or virtual hardware implemented within the data center hosting service **2001** and external public cloud provider **1862**.

[0162] Embodiments of the invention may include various steps as set forth above. The steps may be embodied in machine-executable instructions which cause a general-purpose or special-purpose processor to perform certain steps. Alternatively, these steps may be performed by specific hardware components that contain hardwired logic for performing the steps, or by any combination of programmed computer components and custom hardware components.

[0163] Elements of the present invention may also be provided as a machine-readable medium for storing the machine-executable program code. The machine-readable medium may include, but is not limited to, floppy diskettes, optical disks, CD-ROMs, and magneto-optical disks, ROMs, RAMs, EPROMs, EEPROMs, magnetic or optical cards, or other type of media/machine-readable medium suitable for storing electronic program code.

[0164] Throughout the foregoing description, for the purposes of explanation, numerous specific details were set forth in order to provide a thorough understanding of the invention. It will be apparent, however, to one skilled in the art that the invention may be practiced without some of these specific details. For example, it will be readily apparent to those of skill in the art that the functional modules and methods described herein may be implemented as software, hardware or any combination thereof. Moreover, although some embodiments of the invention are described herein within the context of a mobile computing environment, the underlying principles of the invention are not limited to a mobile computing implementation. Virtually any type of client or peer data processing devices may be used in some embodiments including, for example, desktop or workstation computers. Accordingly, the scope and spirit of the invention should be judged in terms of the claims which follow.

We claim:

1. An apparatus comprising:

a plurality of virtual device controllers to define a corresponding plurality of networking functions to be performed within a cloud computing architecture, the networking functions including traffic engineering components and wide area network (WAN) connection components;

a mediation layer to map the virtual device controllers to physical or virtual networking components to implement the plurality of networking functions within the cloud computing architecture;

wherein the traffic engineering components perform one or more traffic engineering functions and the WAN connection components perform one or more traffic routing functions within the cloud computing architecture.

2. The apparatus as in claim **1** wherein at least one of the traffic engineering components comprises a traffic scheduler to schedule the packets within a plurality of queues to ensure that the bandwidth and/or latency requirements for each of a plurality of hosted applications are being met.

3. The apparatus as in claim **2** wherein the traffic scheduler is to schedule the packets within the queues in accordance with a maximum amount of bandwidth allocated to a tenant of a first cloud provider.

4. The apparatus as in claim **3** wherein the first cloud provider is interconnected to a second cloud provider, the interconnection being specified by the WAN connection components.

5. The apparatus as in claim **1** wherein the first cloud provider is interconnected to an enterprise customer of a network service provider, the interconnection being specified by the WAN connection components.

6. The apparatus as in claim **1** wherein the WAN connection components comprise a set of one or more network chains defining the traffic routing to be performed to interconnect a first endpoint at an edge of a first cloud provider with a second endpoint at an edge of an enterprise customer.

7. The apparatus as in claim **1** wherein at least one of the traffic engineering components comprise:

a direct connect manager to identify each tenant of a first cloud provider and responsively perform traffic engi-

neering in accordance with requirements specified for each of the applications and bandwidth allocated to each of the tenants; and

a border network gateway to translate packets from a first protocol used on the first cloud provider network to a second protocol used by a service provider communicatively coupled to the border network gateway, thereby establishing a connection between each of the tenants and one or more endpoints on the service provider network.

8. The apparatus as in claim **7** wherein the first protocol comprises generic routing encapsulation (GRE) over IP.

9. The apparatus as in claim **8** wherein the second protocol is selected from a group consisting of Multiprotocol Label Switching (MPLS), Border Gateway Protocol (BGP)-Virtual Private Networking, and Q-in-Q.

10. The apparatus as in claim **7** further comprising:

a radius server to identify each tenant to the border network gateway to perform the translation, wherein upon identifying a tenant, the border network gateway is to determine an identifier associated with that tenant for implementing the second protocol.

11. A method comprising:

defining a plurality of virtual device controllers for a corresponding plurality of networking functions to be performed within a cloud computing architecture, the networking functions including traffic engineering components and wide area network (WAN) connection components;

mapping the virtual device controllers to physical or virtual networking components to implement the plurality of networking functions within the cloud computing architecture;

wherein the traffic engineering components perform one or more traffic engineering functions and the WAN connection components perform one or more traffic routing functions within the cloud computing architecture.

12. The method as in claim **11** wherein at least one of the traffic engineering components comprises a traffic scheduler to schedule the packets within a plurality of queues to ensure that the bandwidth and/or latency requirements for each of a plurality of hosted applications are being met.

13. The method as in claim **12** wherein the traffic scheduler is to schedule the packets within the queues in accordance with a maximum amount of bandwidth allocated to a tenant of a first cloud provider.

14. The method as in claim **13** wherein the first cloud provider is interconnected to a second cloud provider, the interconnection being specified by the WAN connection components.

15. The method as in claim **11** wherein the first cloud provider is interconnected to an enterprise customer of a network service provider, the interconnection being specified by the WAN connection components.

16. The method as in claim **11** wherein the WAN connection components comprise a set of one or more network chains defining the traffic routing to be performed to interconnect a first endpoint at an edge of a first cloud provider with a second endpoint at an edge of an enterprise customer.

17. The method as in claim **11** wherein at least one of the traffic engineering components comprise:

a direct connect manager to identify each tenant of a first cloud provider and responsively perform traffic engineering in accordance with requirements specified for each of the applications and bandwidth allocated to each of the tenants; and

a border network gateway to translate packets from a first protocol used on the first cloud provider network to a second protocol used by a service provider communicatively coupled to the border network gateway, thereby establishing a connection between each of the tenants and one or more endpoints on the service provider network.

18. The method as in claim **17** wherein the first protocol comprises generic routing encapsulation (GRE) over IP.

19. The method as in claim **18** wherein the second protocol is selected from a group consisting of Multiprotocol Label Switching (MPLS), Border Gateway Protocol (BGP)-Virtual Private Networking, and Q-in-Q.

20. The method as in claim **17** further comprising:

a radius server to identify each tenant to the border network gateway to perform the translation, wherein upon identifying a tenant, the border network gateway is to determine an identifier associated with that tenant for implementing the second protocol.

* * * * *