US 20160057077A1

(19) **United States**

(12) **Patent Application Publication**
Gomatam et al.

(10) **Pub. No.: US 2016/0057077 A1**

(43) **Pub. Date: Feb. 25, 2016**

(54) **INTELLIGENT DATA CENTER SELECTION**

(71) Applicant: **Oracle International Corporation**, Redwood Shores, CA (US)

(72) Inventors: **Anand Mandayam Gomatam**, Bangalore (IN); **Praneet Tiwari**, Bangalore (IN); **Kunal Shrivastava**, Darbhanga (IN); **Karthic Viswanathan**, Vellore (IN); **Soumitra Kumar Gupta**, Kanpur (IN); **Tejkumar Arora**, Bangalore (IN)

(21) Appl. No.: **14/833,985**

(22) Filed: **Aug. 24, 2015**

Related U.S. Application Data

(60) Provisional application No. 62/040,779, filed on Aug. 22, 2014.

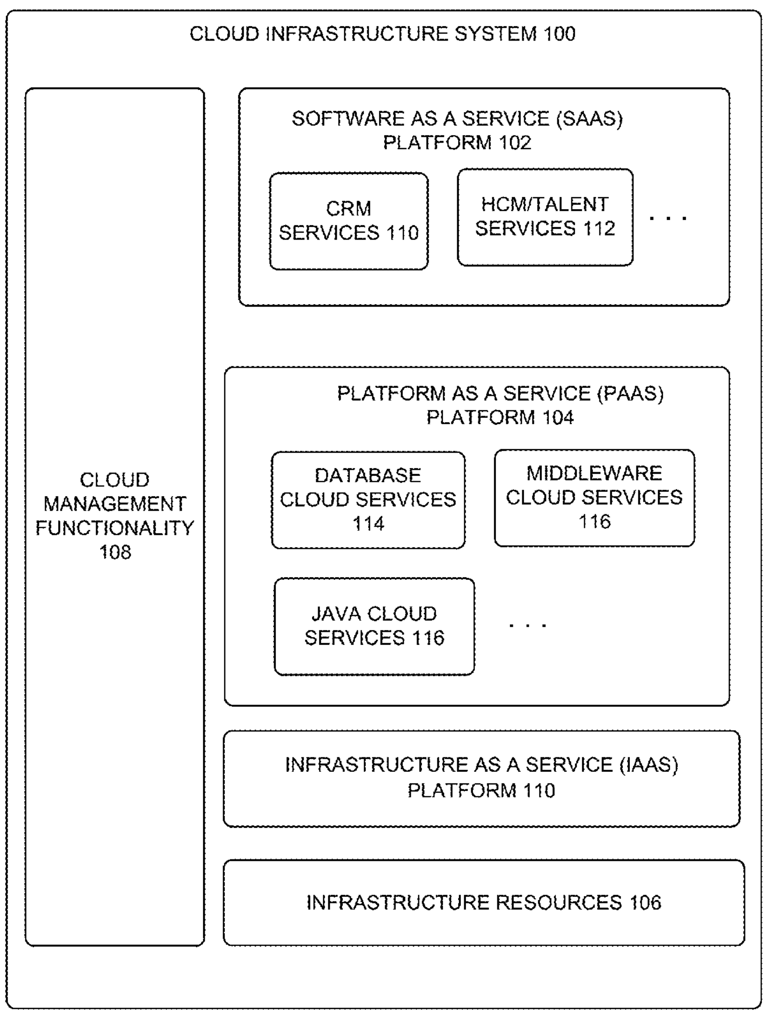
Publication Classification

(51) **Int. Cl.**
H04L 12/911 (2006.01)

(52) **U.S. Cl.**
CPC ***H04L 47/788*** (2013.01)

(57) **ABSTRACT**

In embodiments, a data center selection system can select a chosen data center (DC) for an order submitted to a cloud computing system using a preprocessing layer and a rules engine that incorporates action/algorithm-based selection using data center metrics to determine the chosen DC. In various embodiments, the data center selection system retrieves order information, objectives, rules, algorithms, and other data defined by an administrator. The data center selection system can then retrieve data center information aggregated from various data centers. Using order information received from an order management system, the data center selection system can utilizes the preprocessing layer, the rules engine, and the algorithm-based selection to select the chosen DC. The data center selection system can send an order request to the chosen DC in which to provision services for the order request.



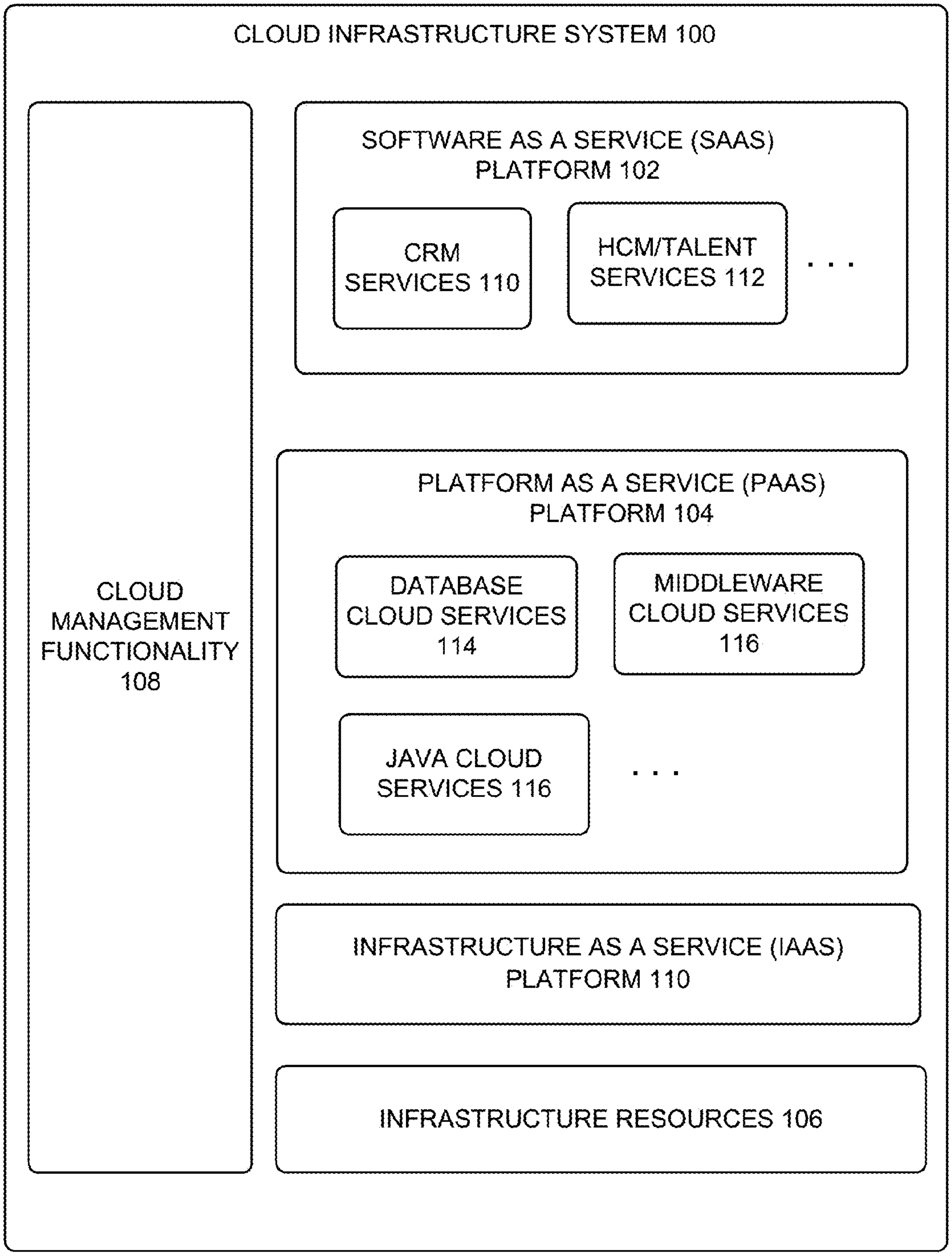
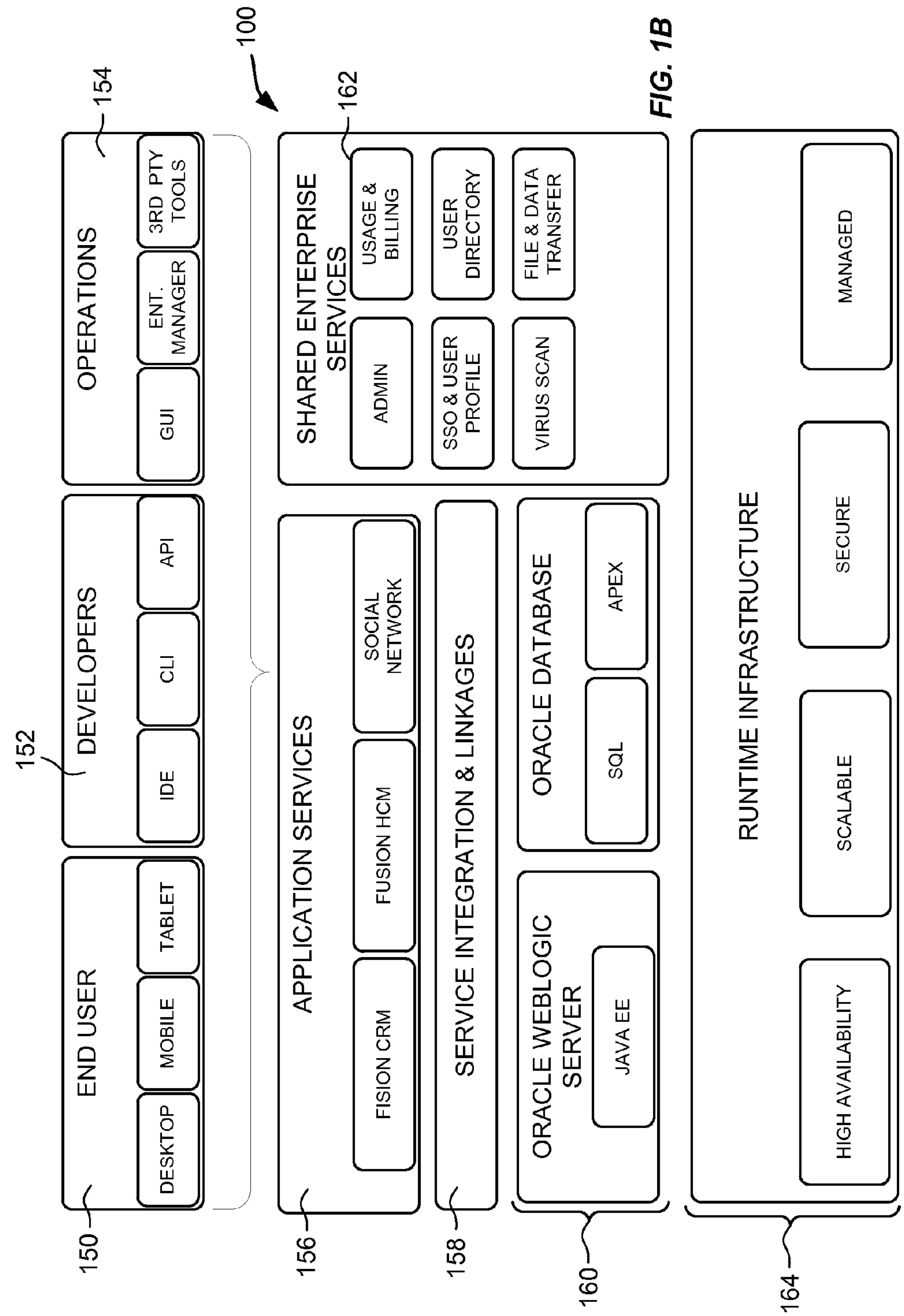

CLOUD INFRASTRUCTURE SYSTEM 100
CLOUD MANAGEMENT FUNCTIONALITY 108
SOFTWARE AS A SERVICE (SAAS) PLATFORM 102
CRM SERVICES 110
HCM/TALENT SERVICES 112
. . .
PLATFORM AS A SERVICE (PAAS) PLATFORM 104
DATABASE CLOUD SERVICES 114
MIDDLEWARE CLOUD SERVICES 116
JAVA CLOUD SERVICES 116
. . .
INFRASTRUCTURE AS A SERVICE (IAAS) PLATFORM 110
INFRASTRUCTURE RESOURCES 106

FIG. 1A



100
END USER
DESKTOP
MOBILE
TABLET
150
DEVELOPERS
IDE
CLI
API
152
OPERATIONS
GUI
ENT. MANAGER
3RD PTY TOOLS
154
APPLICATION SERVICES
FISION CRM
FUSION HCM
SOCIAL NETWORK
156
SERVICE INTEGRATION & LINKAGES
158
ORACLE WEBLOGIC SERVER
JAVA EE
ORACLE DATABASE
SQL
APEX
160
SHARED ENTERPRISE SERVICES
ADMIN
USAGE & BILLING
SSO & USER PROFILE
USER DIRECTORY
VIRUS SCAN
FILE & DATA TRANSFER
162
RUNTIME INFRASTRUCTURE
HIGH AVAILABILITY
SCALABLE
SECURE
MANAGED
164

FIG. 1B

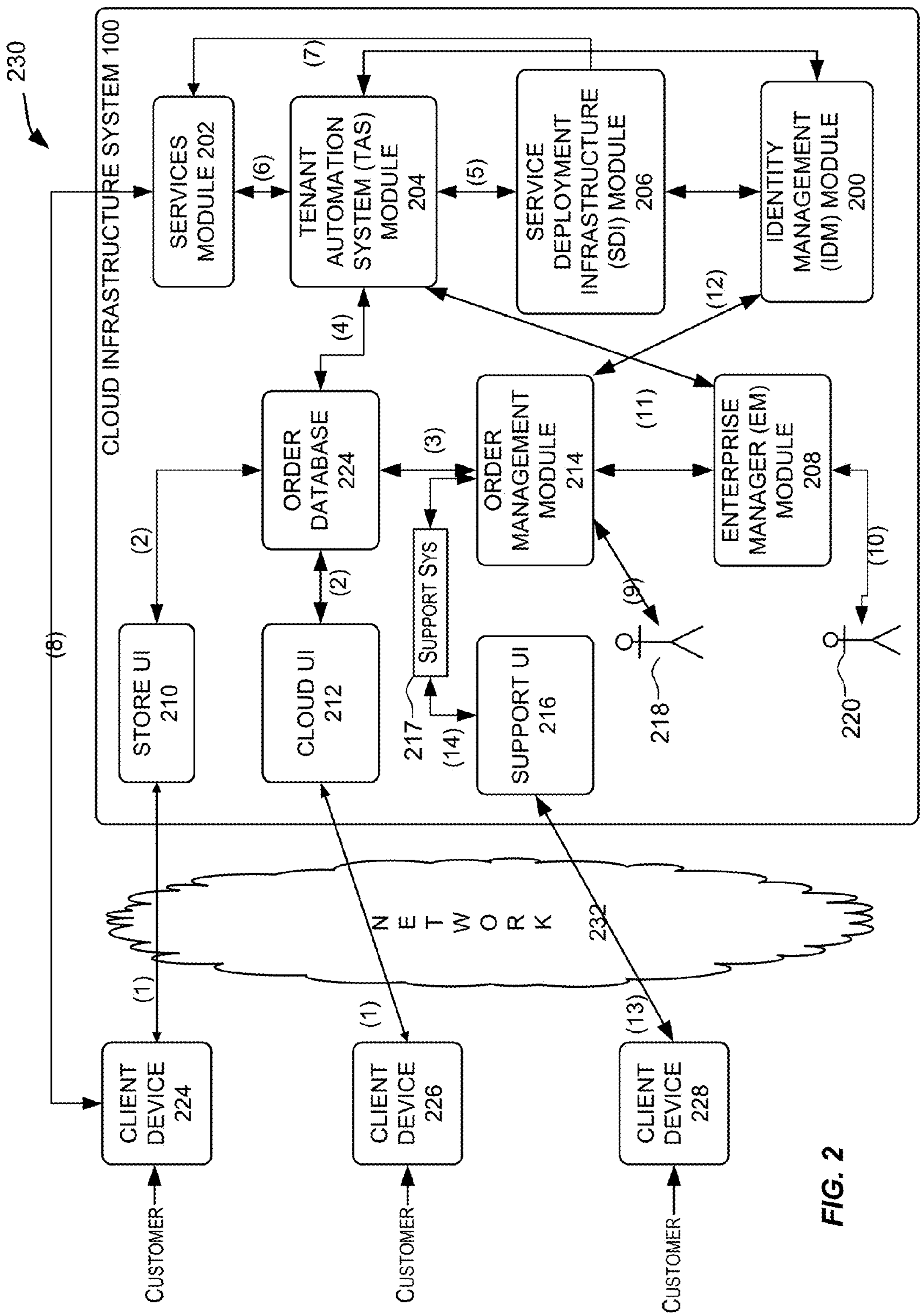

230
CLOUD INFRASTRUCTURE SYSTEM 100
SERVICES MODULE 202
TENANT AUTOMATION SYSTEM (TAS) MODULE 204
SERVICE DEPLOYMENT INFRASTRUCTURE (SDI) MODULE 206
IDENTITY MANAGEMENT (IDM) MODULE 200
ORDER DATABASE 224
ORDER MANAGEMENT MODULE 214
ENTERPRISE MANAGER (EM) MODULE 208
SUPPORT SYS
217
STORE UI 210
CLOUD UI 212
SUPPORT UI 216
218
220
NETWORK
232
CLIENT DEVICE 224
CLIENT DEVICE 226
CLIENT DEVICE 228
CUSTOMER
CUSTOMER
CUSTOMER
(1)
(1)
(2)
(2)
(3)
(4)
(5)
(6)
(7)
(8)
(9)
(10)
(11)
(12)
(13)
(14)

FIG. 2

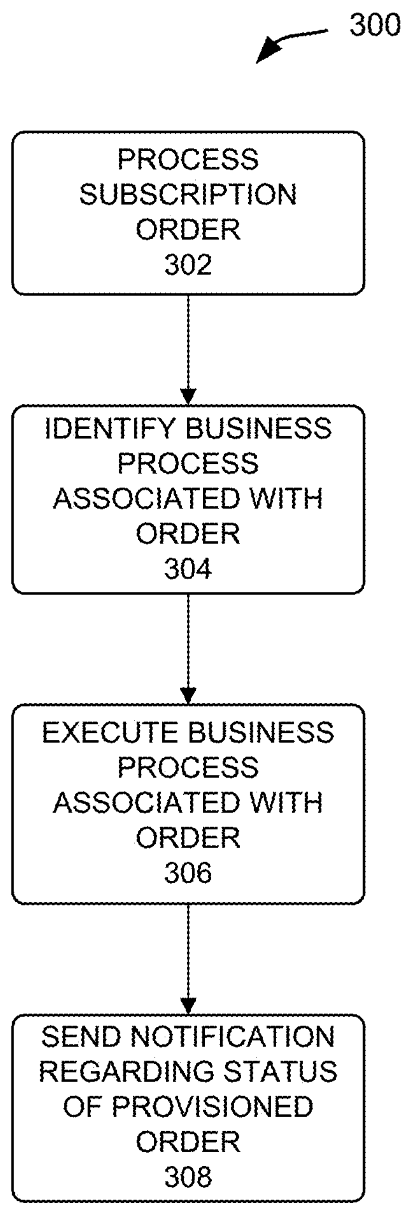

300
PROCESS SUBSCRIPTION ORDER 302
IDENTIFY BUSINESS PROCESS ASSOCIATED WITH ORDER 304
EXECUTE BUSINESS PROCESS ASSOCIATED WITH ORDER 306
SEND NOTIFICATION REGARDING STATUS OF PROVISIONED ORDER 308

FIG. 3A

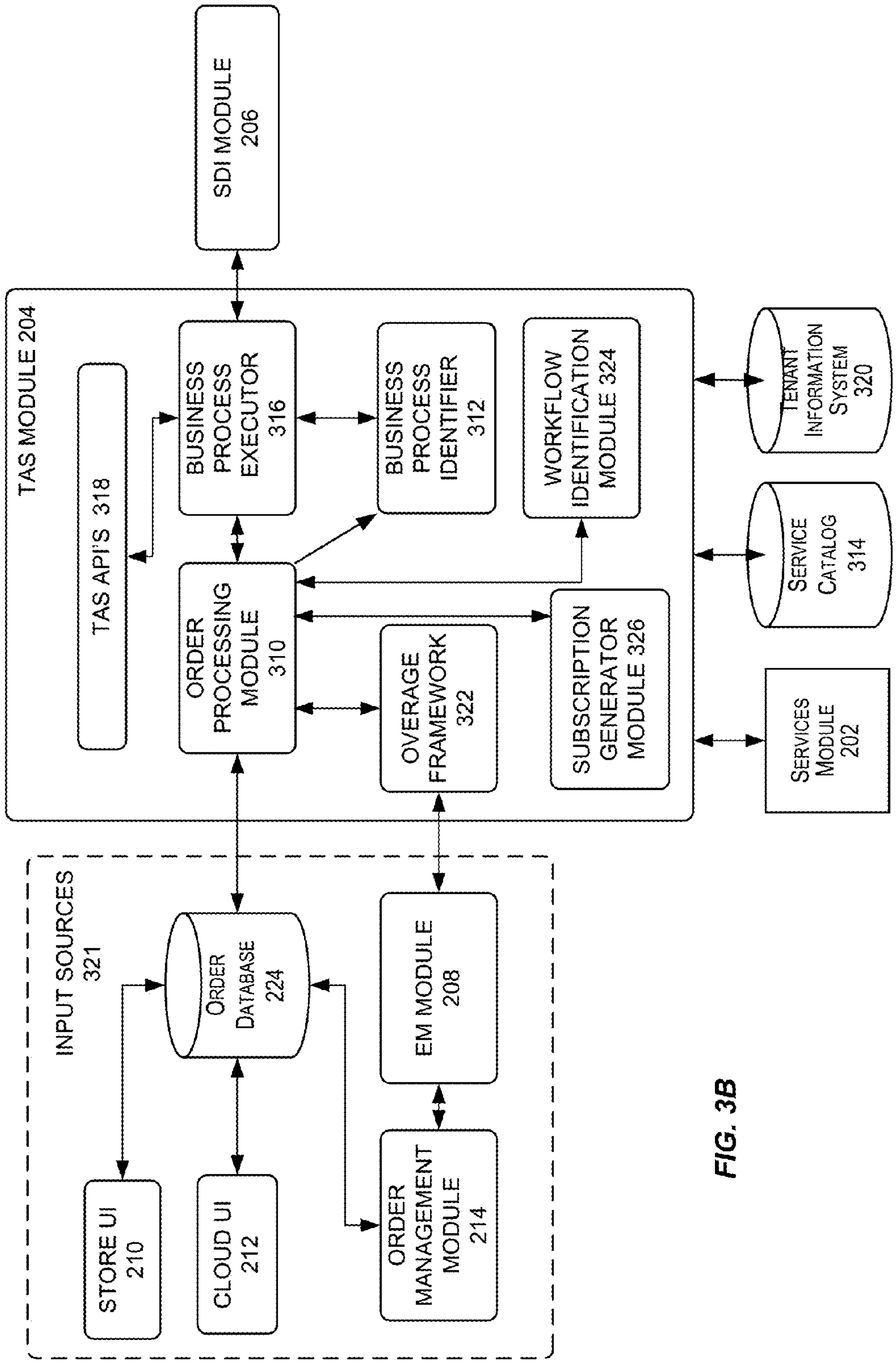

SDI MODULE 206
TAS MODULE 204
TAS API'S 318
BUSINESS PROCESS EXECUTOR 316
BUSINESS PROCESS IDENTIFIER 312
WORKFLOW IDENTIFICATION MODULE 324
ORDER PROCESSING MODULE 310
OVERAGE FRAMEWORK 322
SUBSCRIPTION GENERATOR MODULE 326
TENANT INFORMATION SYSTEM 320
SERVICE CATALOG 314
SERVICES MODULE 202
INPUT SOURCES 321
ORDER DATABASE 224
EM MODULE 208
STORE UI 210
CLOUD UI 212
ORDER MANAGEMENT MODULE 214

FIG. 3B

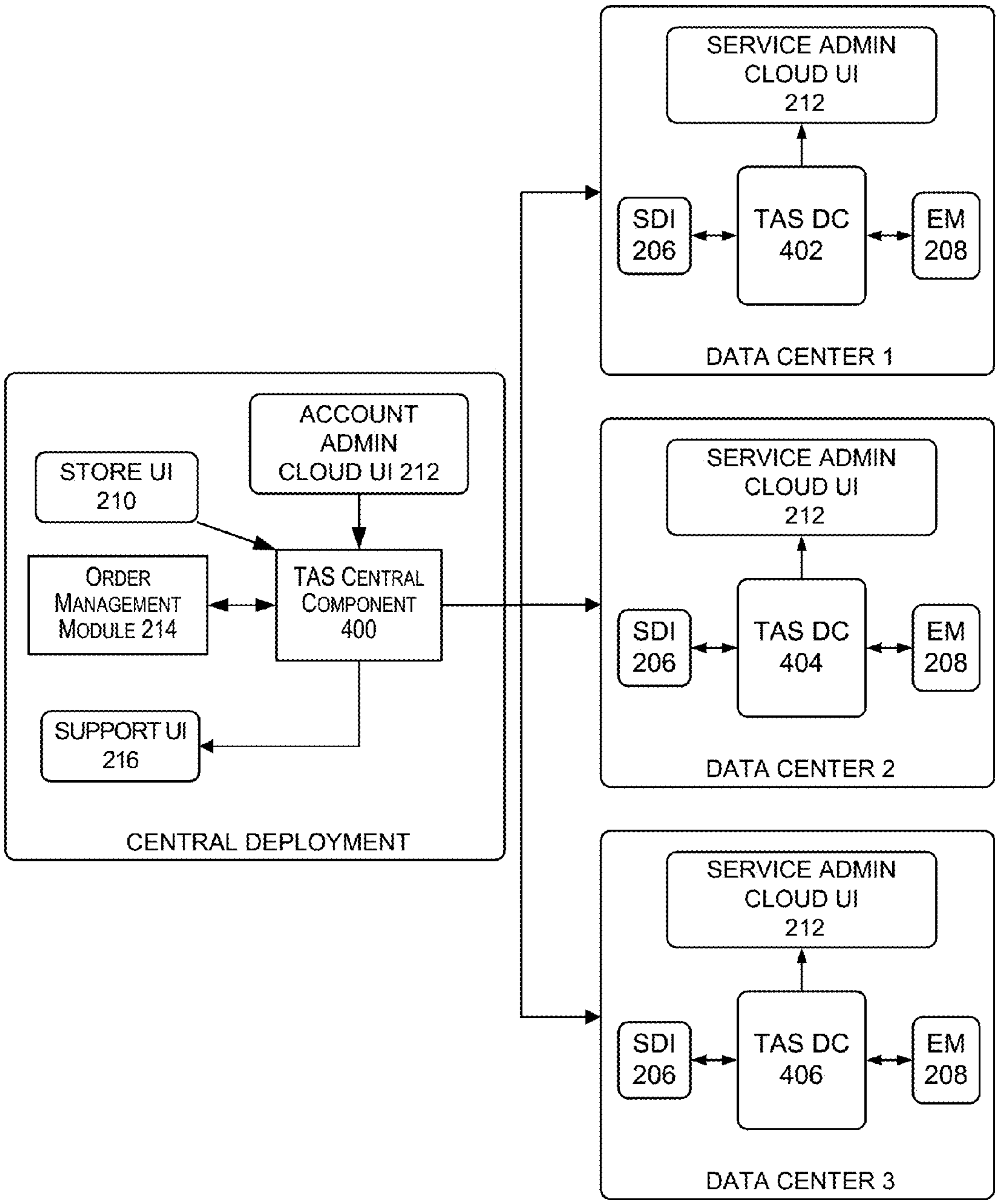

SERVICE ADMIN CLOUD UI 212
SDI 206
TAS DC 402
EM 208
DATA CENTER 1
ACCOUNT ADMIN CLOUD UI 212
STORE UI 210
ORDER MANAGEMENT MODULE 214
TAS CENTRAL COMPONENT 400
SUPPORT UI 216
CENTRAL DEPLOYMENT
SERVICE ADMIN CLOUD UI 212
SDI 206
TAS DC 404
EM 208
DATA CENTER 2
SERVICE ADMIN CLOUD UI 212
SDI 206
TAS DC 406
EM 208
DATA CENTER 3

FIG. 4

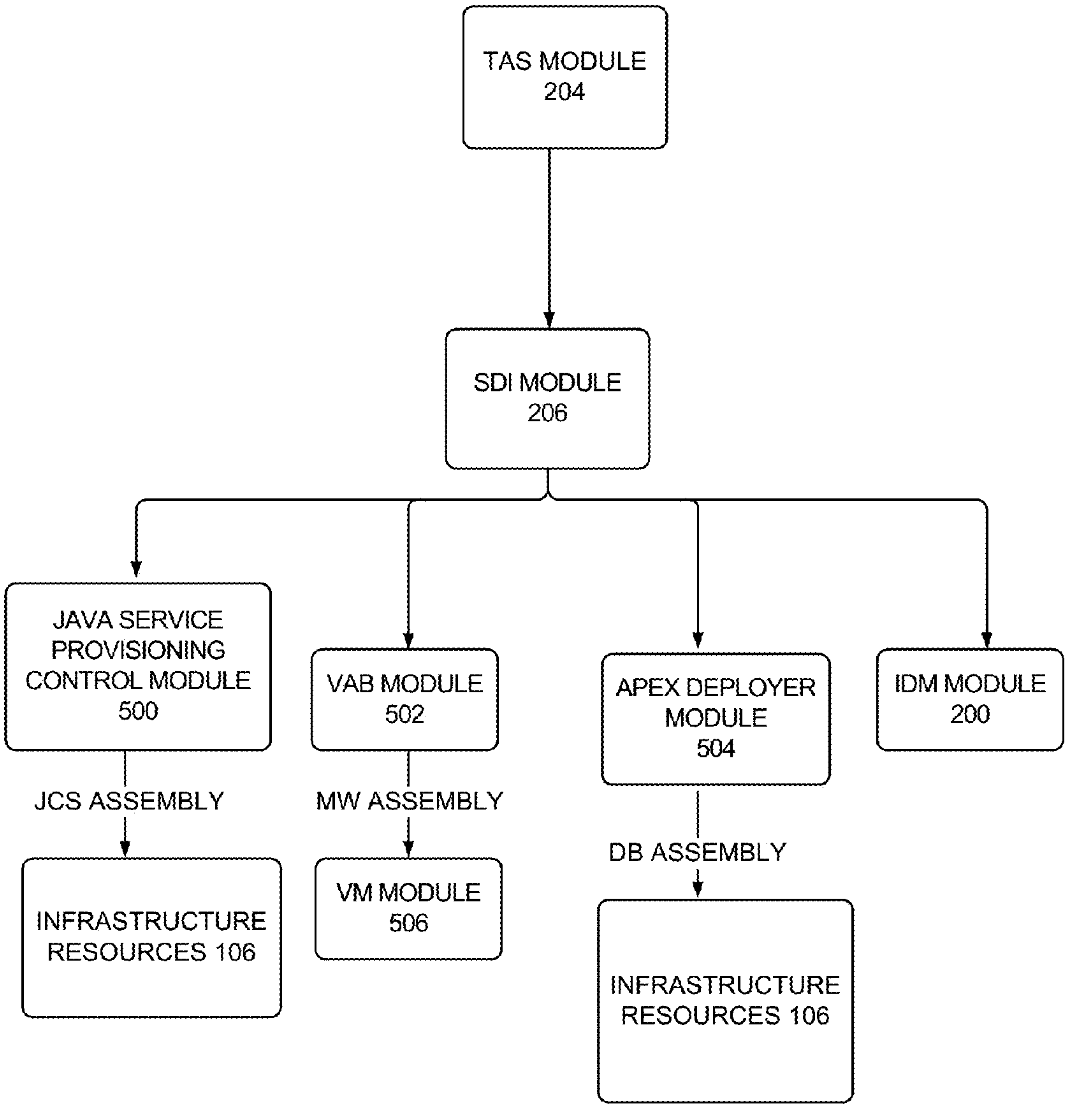

TAS MODULE
204
SDI MODULE
206
JAVA SERVICE
PROVISIONING
CONTROL MODULE
500
VAB MODULE
502
APEX DEPLOYER
MODULE
504
IDM MODULE
200
JCS ASSEMBLY
MW ASSEMBLY
DB ASSEMBLY
INFRASTRUCTURE
RESOURCES 106
VM MODULE
506
INFRASTRUCTURE
RESOURCES 106

FIG. 5

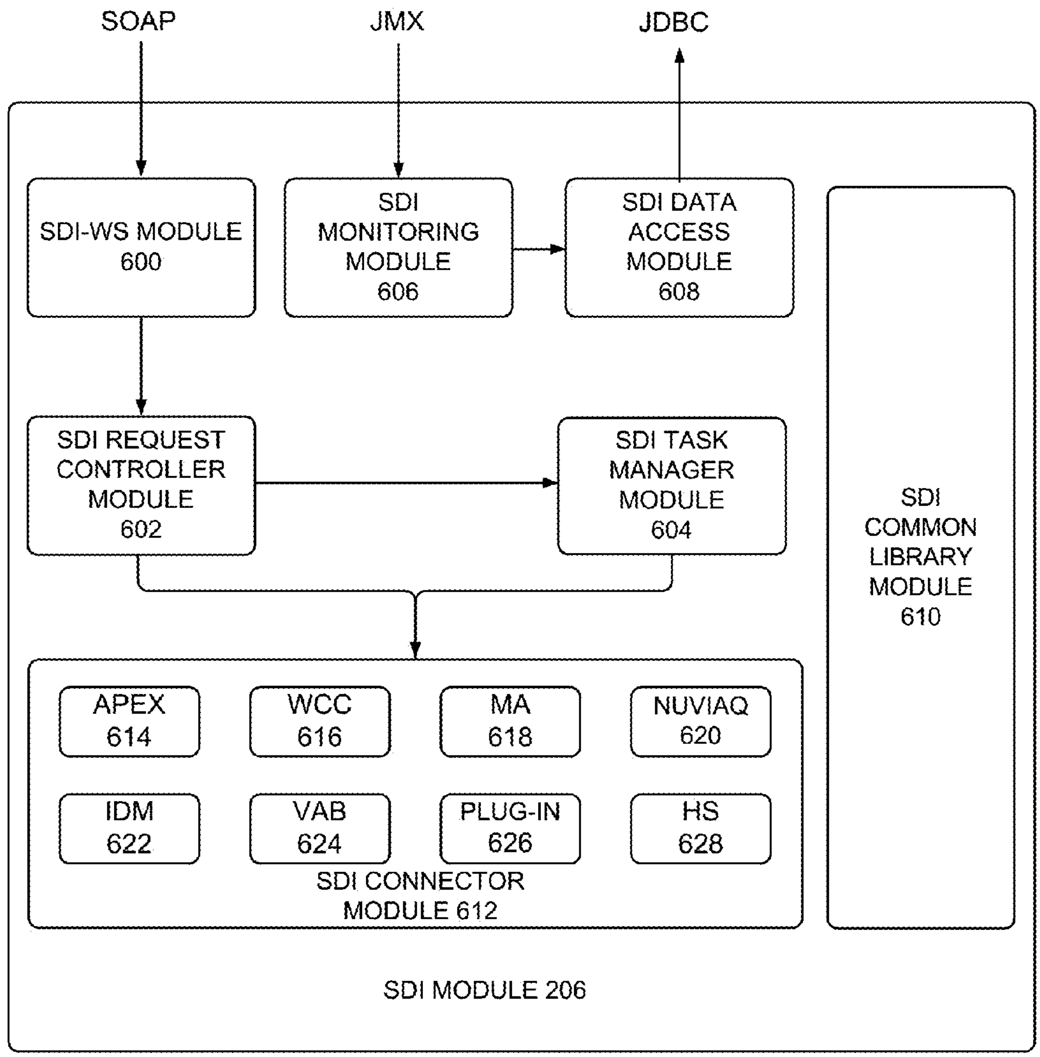

SOAP
JMX
JDBC
SDI-WS MODULE 600
SDI MONITORING MODULE 606
SDI DATA ACCESS MODULE 608
SDI REQUEST CONTROLLER MODULE 602
SDI TASK MANAGER MODULE 604
SDI COMMON LIBRARY MODULE 610
APEX 614
WCC 616
MA 618
NUVIAQ 620
IDM 622
VAB 624
PLUG-IN 626
HS 628
SDI CONNECTOR MODULE 612
SDI MODULE 206

FIG. 6

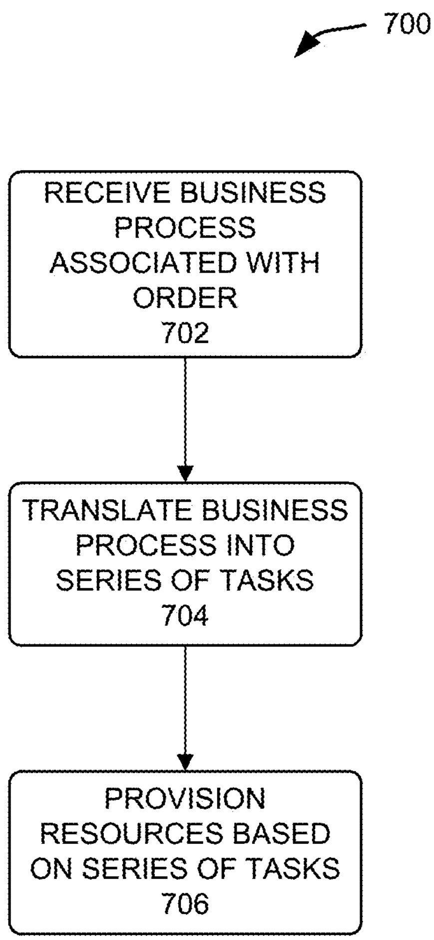
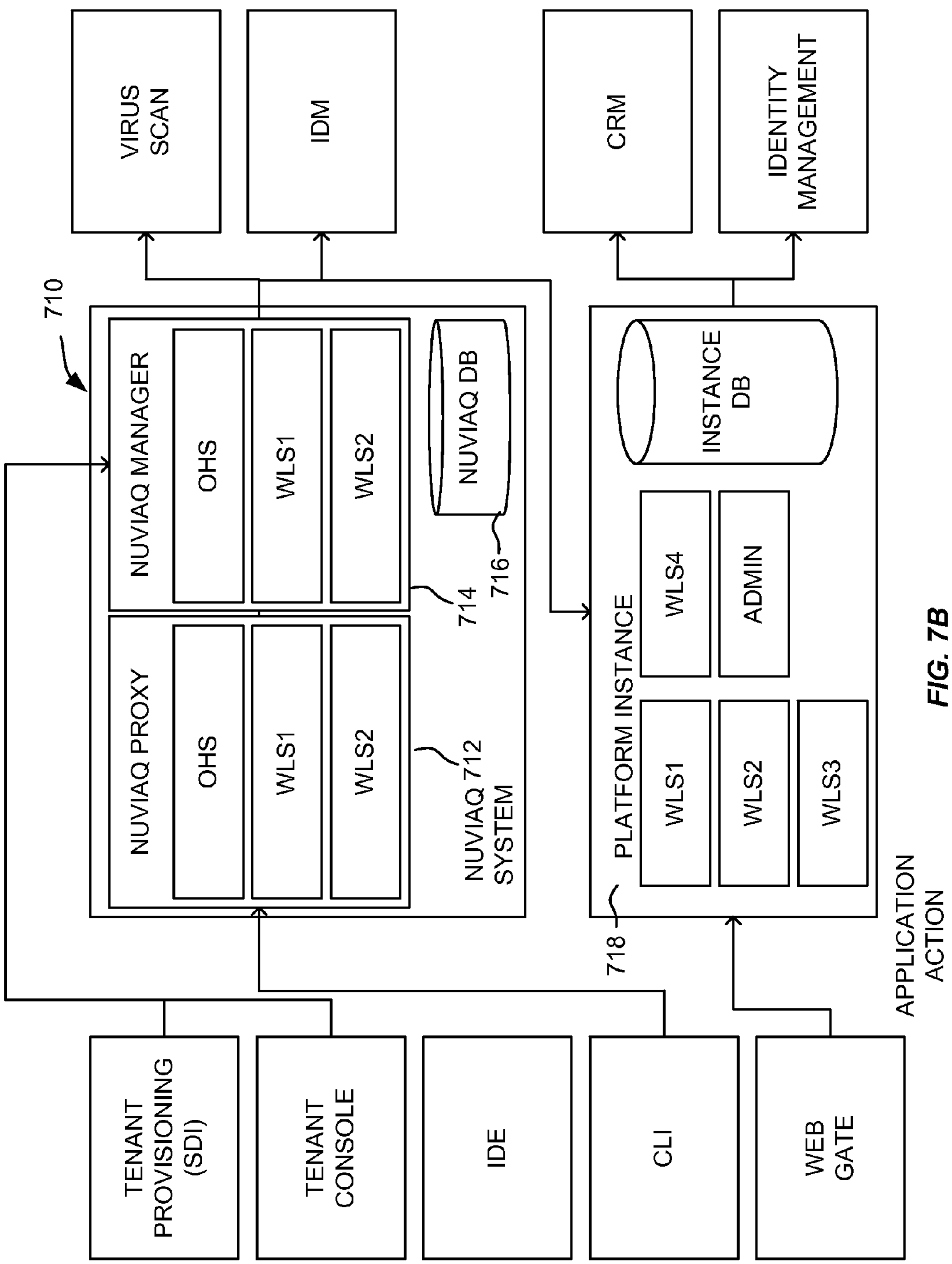
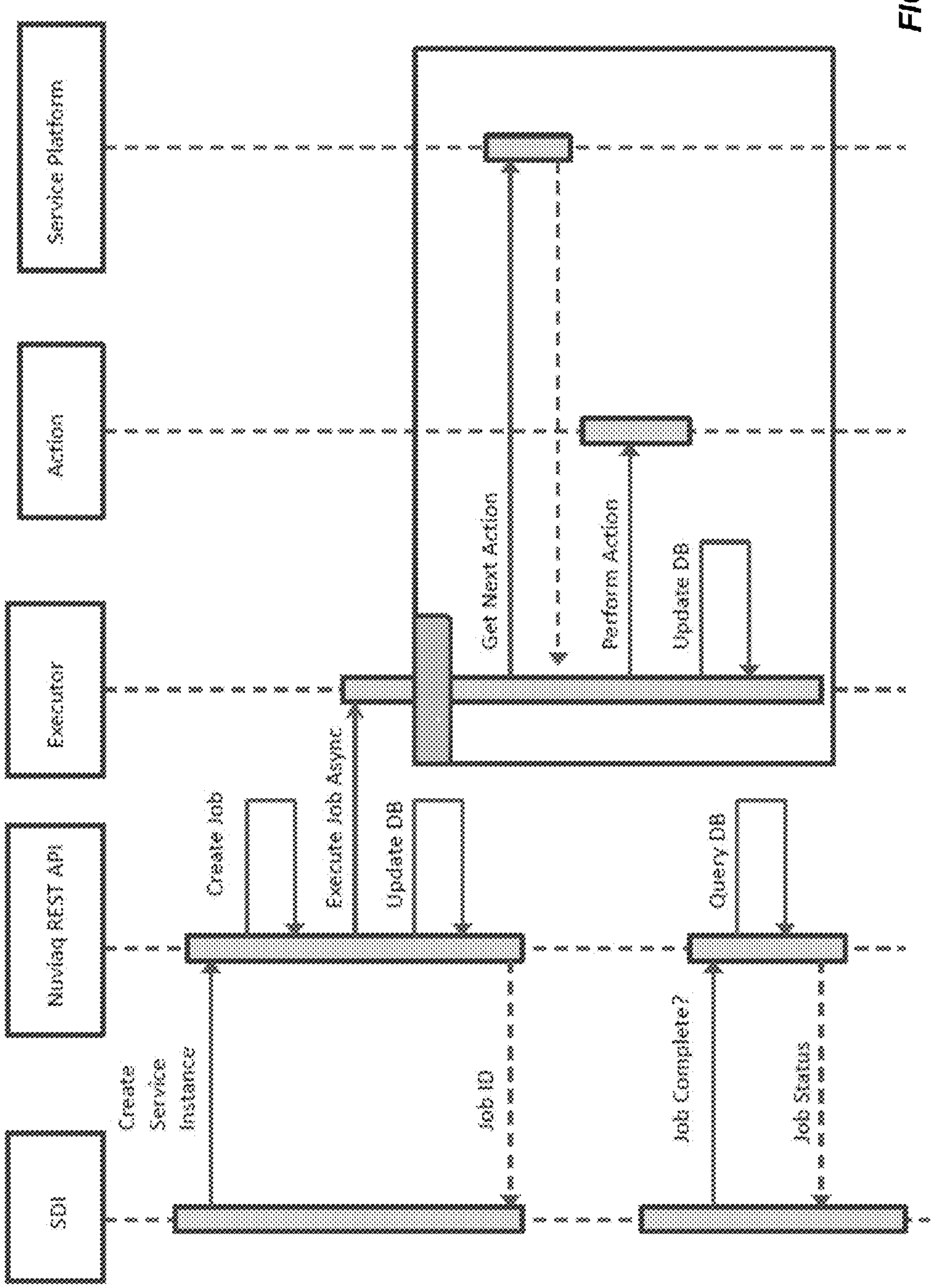

700
RECEIVE BUSINESS PROCESS ASSOCIATED WITH ORDER
702
TRANSLATE BUSINESS PROCESS INTO SERIES OF TASKS
704
PROVISION RESOURCES BASED ON SERIES OF TASKS
706

FIG. 7A



VIRUS SCAN
IDM
CRM
IDENTITY MANAGEMENT
710
NUVIAQ MANAGER
OHS
WLS1
WLS2
NUVIAQ DB
716
714
NUVIAQ PROXY
OHS
WLS1
WLS2
NUVIAQ 712 SYSTEM
INSTANCE DB
PLATFORM INSTANCE
WLS4
ADMIN
WLS1
WLS2
WLS3
718
TENANT PROVISIONING (SDI)
TENANT CONSOLE
IDE
CLI
WEB GATE
APPLICATION ACTION

FIG. 7B



SDI
Nuviaq REST API
Executor
Action
Service Platform
Create Service Instance
Job ID
Job Complete?
Job Status
Create Job
Execute Job Async
Update DB
Query DB
Get Next Action
Perform Action
Update DB

FIG. 7C

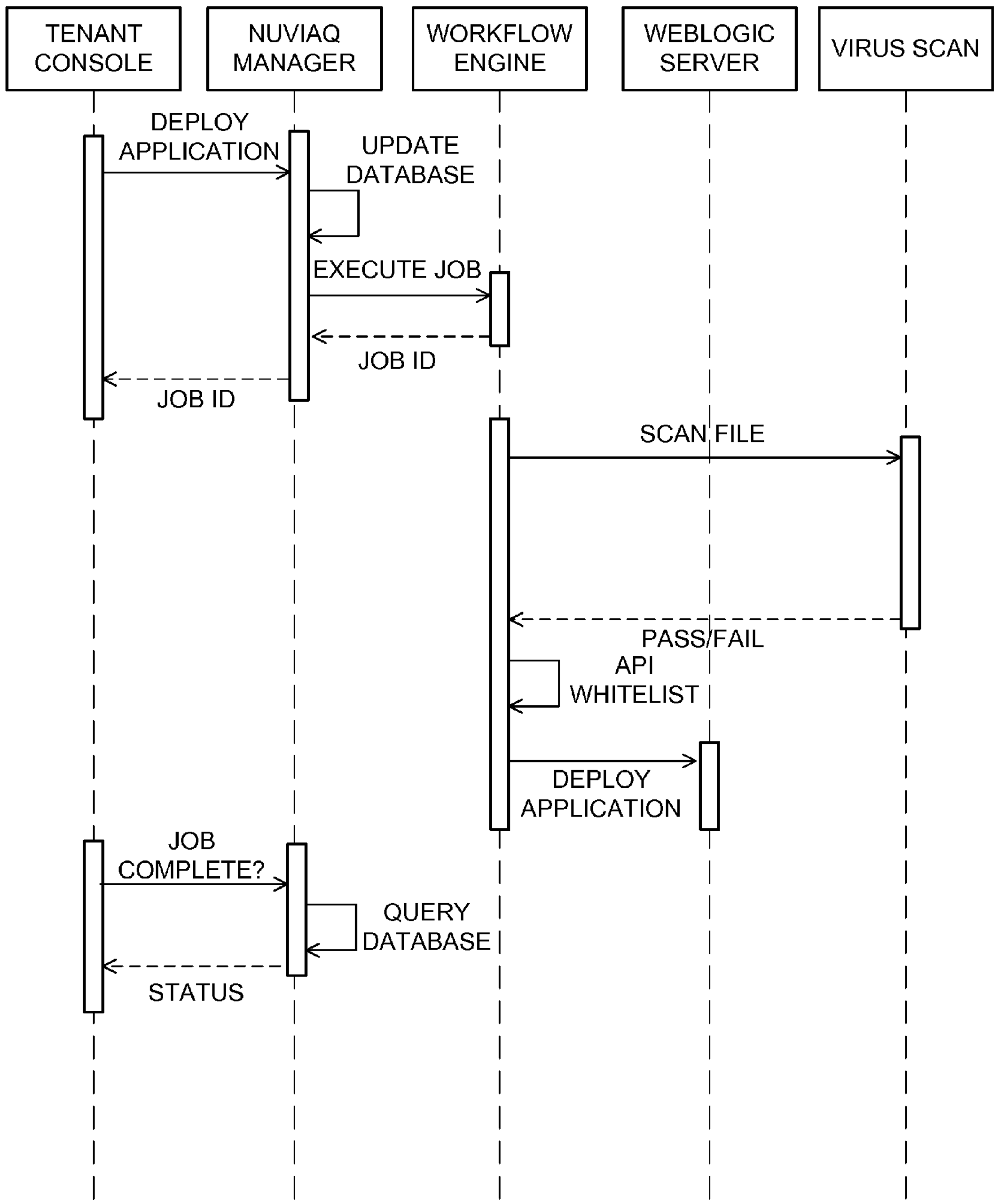

TENANT CONSOLE
NUVIAQ MANAGER
WORKFLOW ENGINE
WEBLOGIC SERVER
VIRUS SCAN
DEPLOY APPLICATION
UPDATE DATABASE
EXECUTE JOB
JOB ID
JOB ID
SCAN FILE
PASS/FAIL
API WHITELIST
DEPLOY APPLICATION
JOB COMPLETE?
QUERY DATABASE
STATUS

FIG. 7D

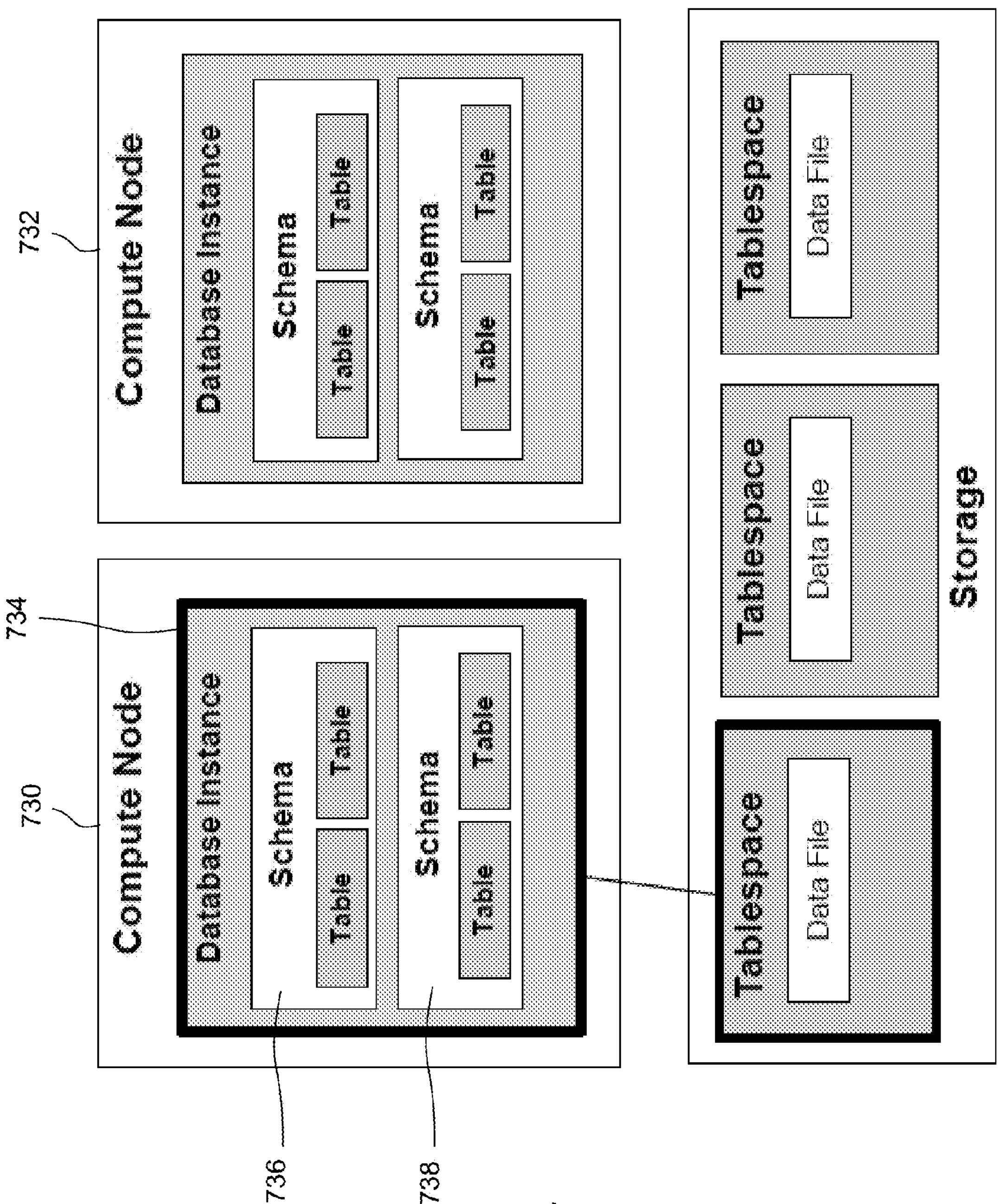

732
Compute Node
Database Instance
Schema
Table
Table
Schema
Table
Table
734
730
Compute Node
Database Instance
Schema
Table
Table
Schema
Table
Table
736
738
Tablespace
Data File
Tablespace
Data File
Tablespace
Data File
Storage

FIG. 7E

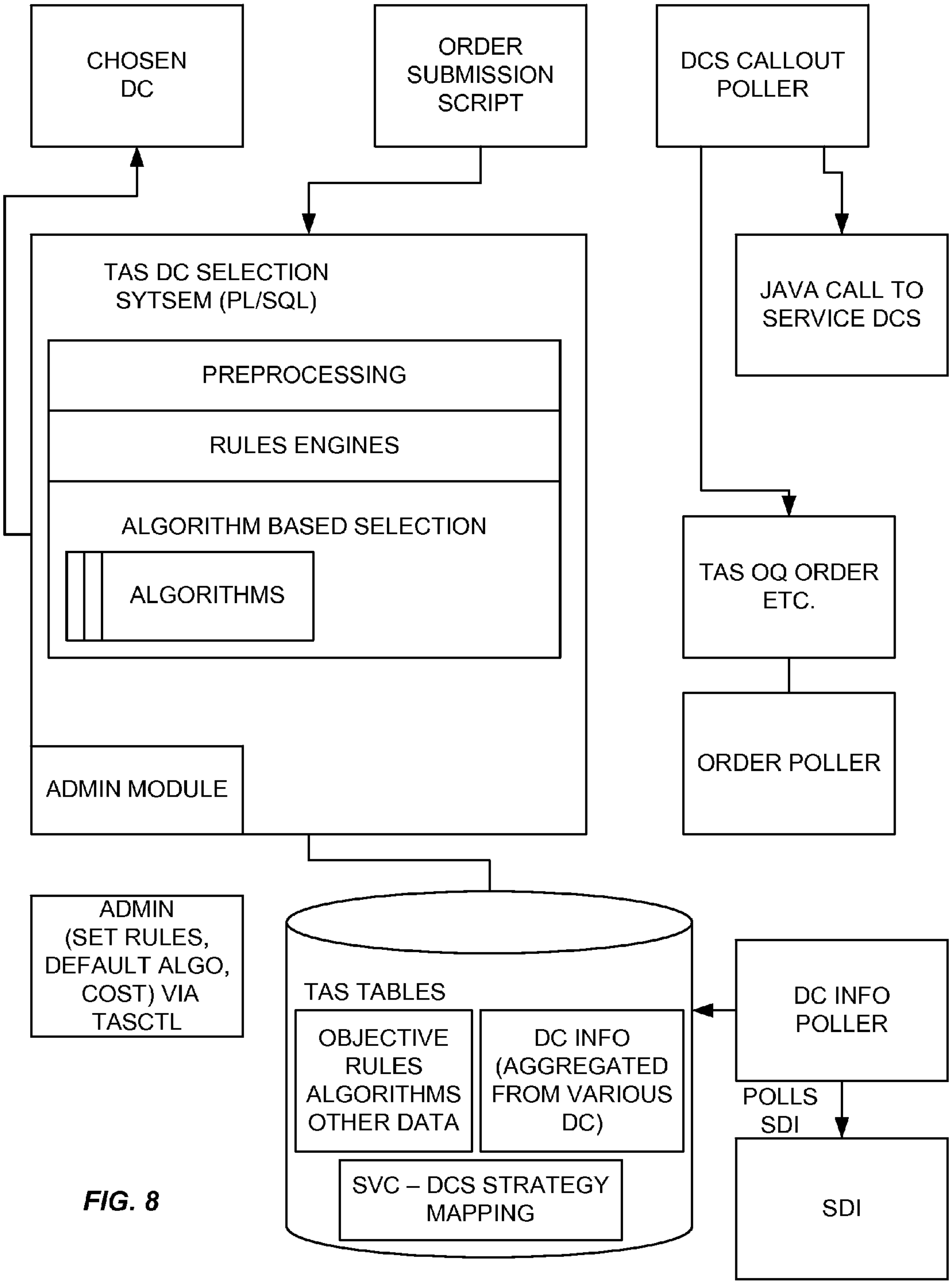

CHOSEN DC
ORDER SUBMISSION SCRIPT
DCS CALLOUT POLLER
TAS DC SELECTION SYTSEM (PL/SQL)
PREPROCESSING
RULES ENGINES
ALGORITHM BASED SELECTION
ALGORITHMS
ADMIN MODULE
JAVA CALL TO SERVICE DCS
TAS OQ ORDER ETC.
ORDER POLLER
ADMIN (SET RULES, DEFAULT ALGO, COST) VIA TASCTL
TAS TABLES
OBJECTIVE RULES ALGORITHMS OTHER DATA
DC INFO (AGGREGATED FROM VARIOUS DC)
SVC – DCS STRATEGY MAPPING
DC INFO POLLER
POLLS SDI
SDI

FIG. 8

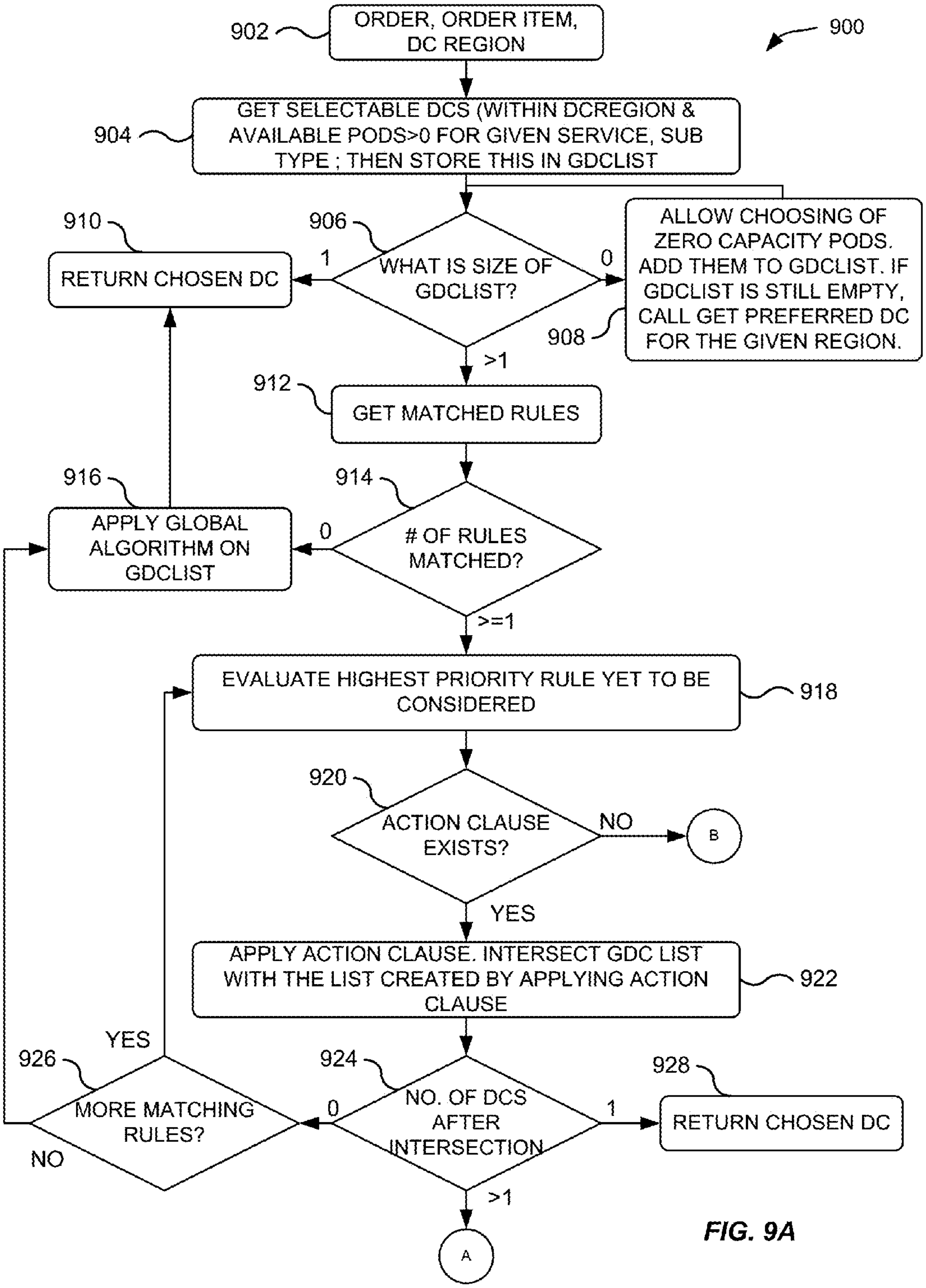

900
902
ORDER, ORDER ITEM, DC REGION
904
GET SELECTABLE DCS (WITHIN DCREGION & AVAILABLE PODS>0 FOR GIVEN SERVICE, SUB TYPE ; THEN STORE THIS IN GDCLIST
906
WHAT IS SIZE OF GDCLIST?
1
0
>1
910
RETURN CHOSEN DC
908
ALLOW CHOOSING OF ZERO CAPACITY PODS. ADD THEM TO GDCLIST. IF GDCLIST IS STILL EMPTY, CALL GET PREFERRED DC FOR THE GIVEN REGION.
912
GET MATCHED RULES
914
OF RULES MATCHED?
0
>=1
916
APPLY GLOBAL ALGORITHM ON GDCLIST
918
EVALUATE HIGHEST PRIORITY RULE YET TO BE CONSIDERED
920
ACTION CLAUSE EXISTS?
NO
B
YES
922
APPLY ACTION CLAUSE. INTERSECT GDC LIST WITH THE LIST CREATED BY APPLYING ACTION CLAUSE
924
NO. OF DCS AFTER INTERSECTION
0
1
>1
A
926
MORE MATCHING RULES?
YES
NO
928
RETURN CHOSEN DC

FIG. 9A

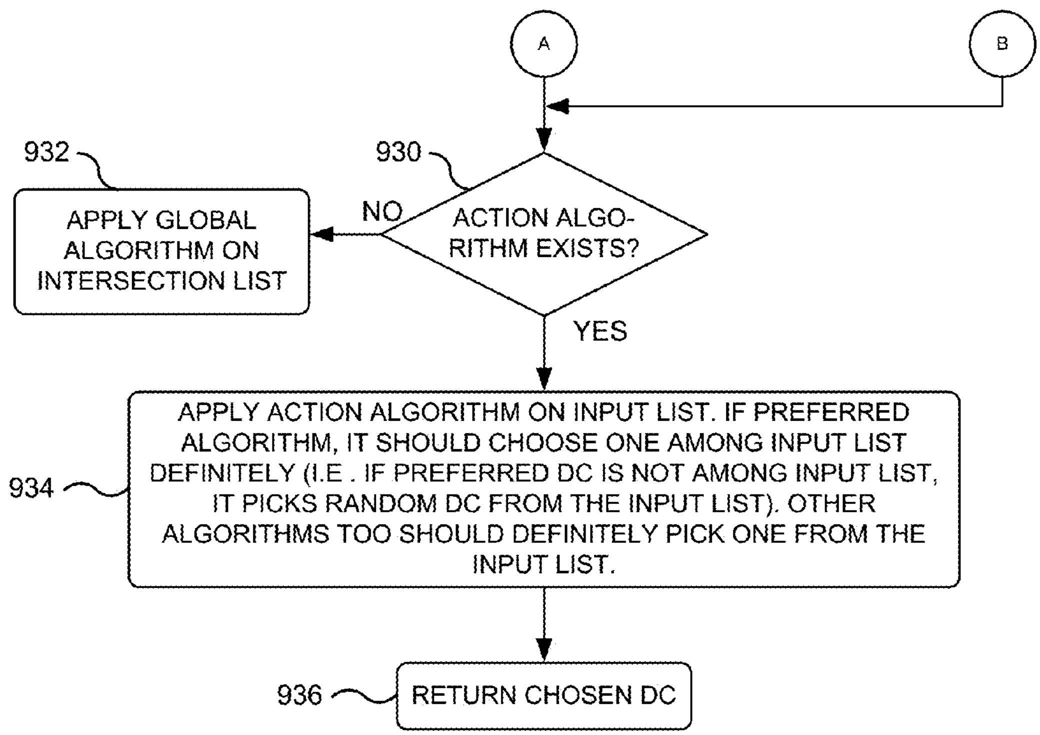

A
B
930
ACTION ALGO-
RITHM EXISTS?
NO
932
APPLY GLOBAL
ALGORITHM ON
INTERSECTION LIST
YES
934
APPLY ACTION ALGORITHM ON INPUT LIST. IF PREFERRED
ALGORITHM, IT SHOULD CHOOSE ONE AMONG INPUT LIST
DEFINITELY (I.E . IF PREFERRED DC IS NOT AMONG INPUT LIST,
IT PICKS RANDOM DC FROM THE INPUT LIST). OTHER
ALGORITHMS TOO SHOULD DEFINITELY PICK ONE FROM THE
INPUT LIST.
936
RETURN CHOSEN DC

FIG. 9B

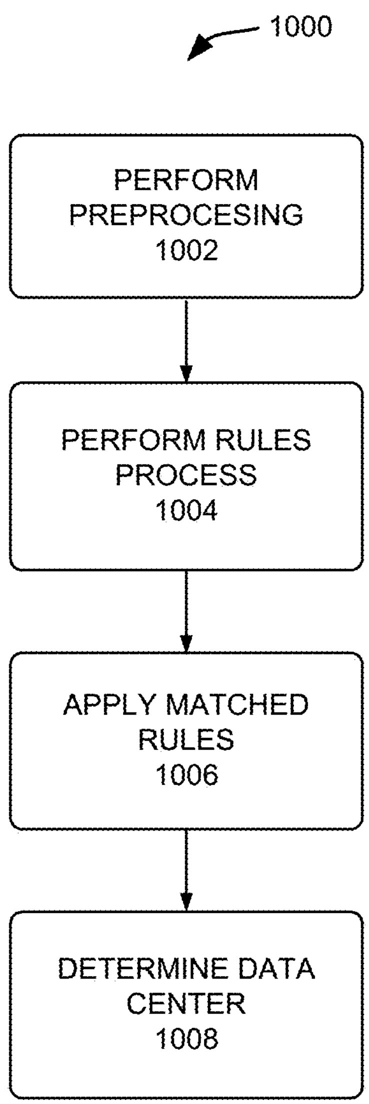

1000
PERFORM PREPROCESING 1002
PERFORM RULES PROCESS 1004
APPLY MATCHED RULES 1006
DETERMINE DATA CENTER 1008

FIG. 10

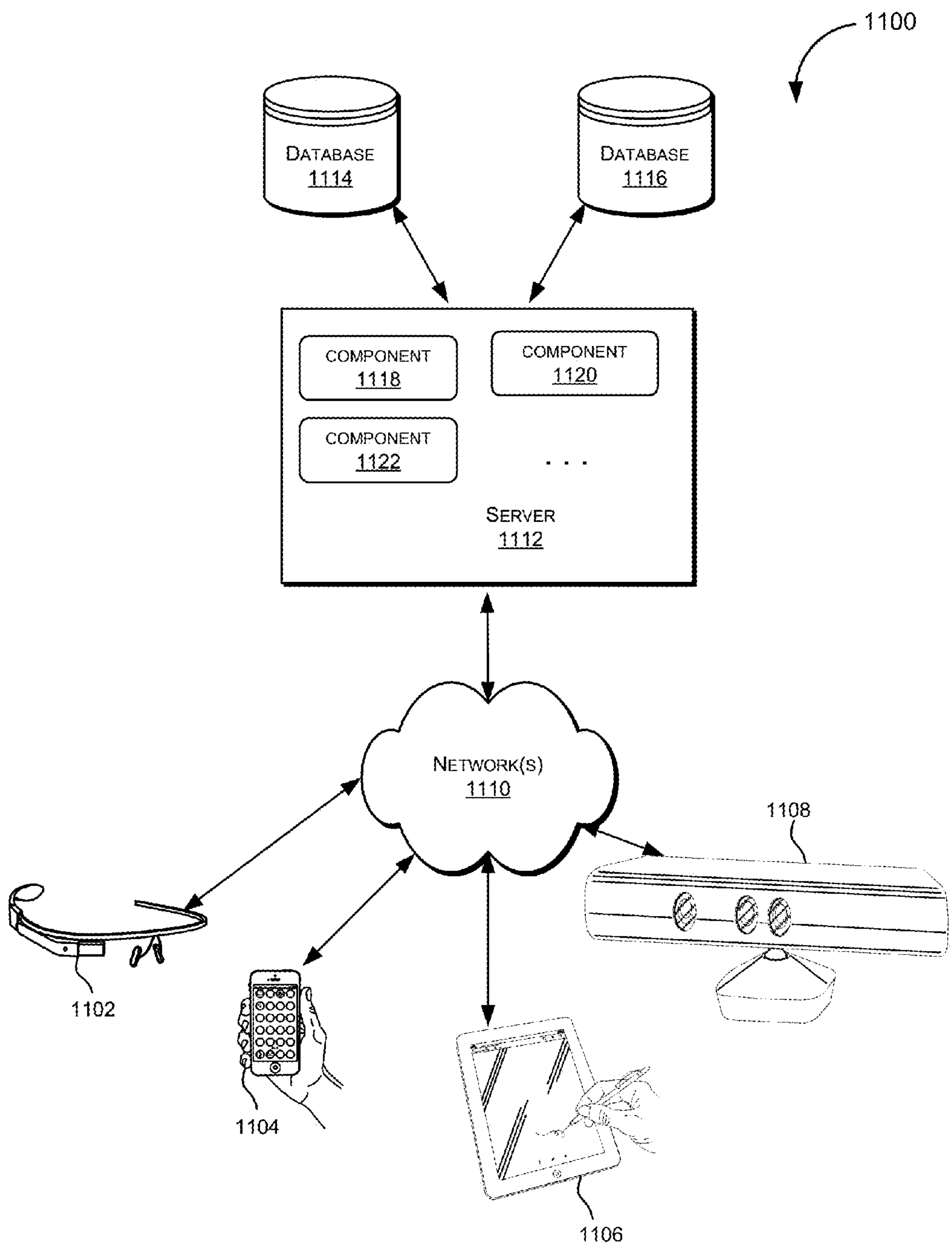

1100
DATABASE
1114
DATABASE
1116
COMPONENT
1118
COMPONENT
1120
COMPONENT
1122
. . .
SERVER
1112
NETWORK(S)
1110
1108
1102
1104
1106

FIG. 11

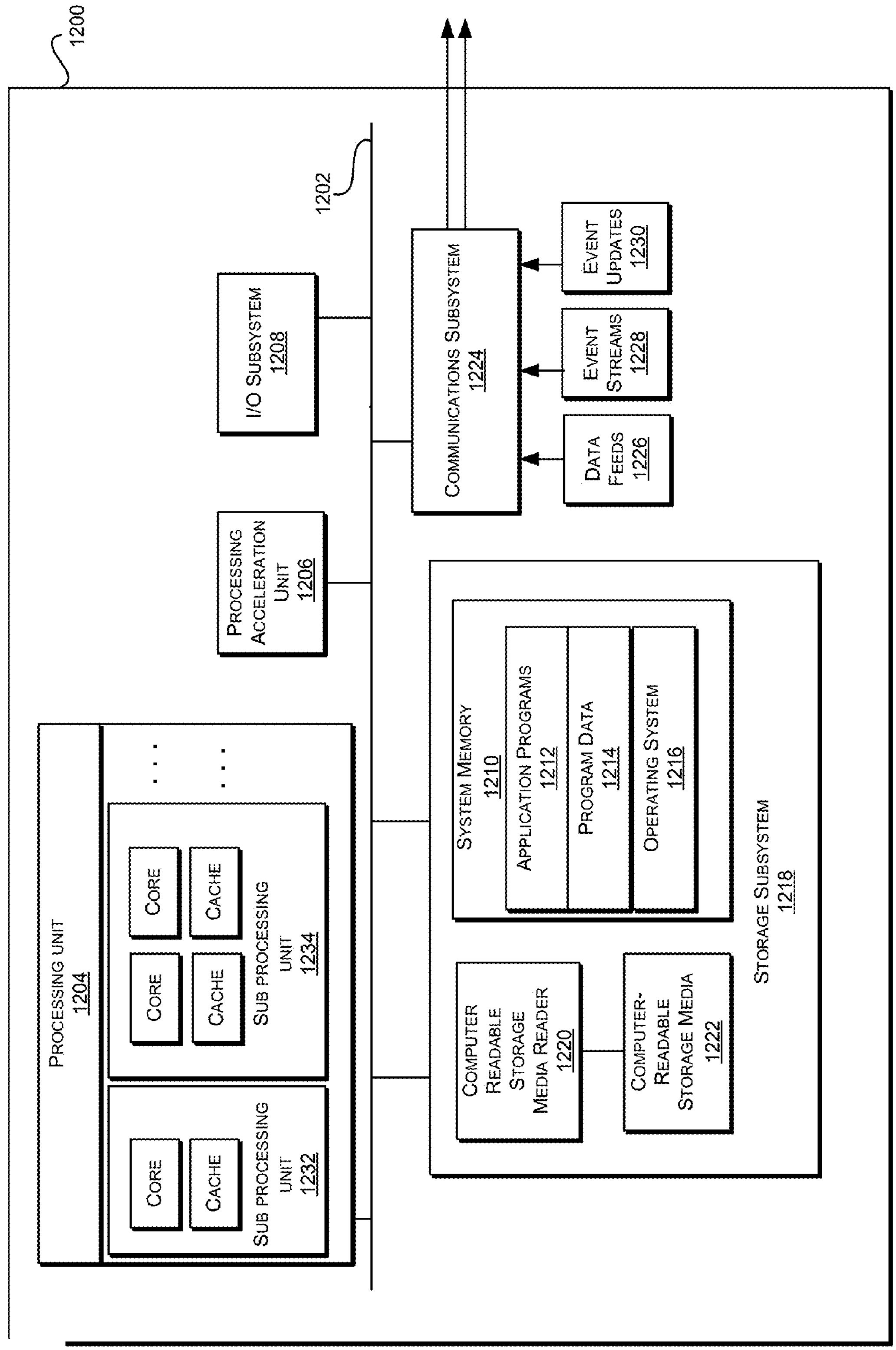

1200
1202
I/O SUBSYSTEM
1208
PROCESSING ACCELERATION UNIT
1206
COMMUNICATIONS SUBSYSTEM
1224
EVENT UPDATES
1230
EVENT STREAMS
1228
DATA FEEDS
1226
PROCESSING UNIT
1204
CORE
CACHE
SUB PROCESSING UNIT
1232
CORE
CACHE
CORE
CACHE
SUB PROCESSING UNIT
1234
SYSTEM MEMORY
1210
APPLICATION PROGRAMS
1212
PROGRAM DATA
1214
OPERATING SYSTEM
1216
COMPUTER READABLE STORAGE MEDIA READER
1220
COMPUTER-READABLE STORAGE MEDIA
1222
STORAGE SUBSYSTEM
1218

FIG. 12

INTELLIGENT DATA CENTER SELECTION

CROSS-REFERENCES TO RELATED APPLICATIONS

[0001] This Applications claims priority to and the benefit of U.S. Provisional Patent Application No. 62/040,779, filed Aug. 22, 2014 and entitled "INTELLIGENT DATA CENTER SELECTION," which is hereby incorporate by reference for all purposes.

BACKGROUND

[0002] The present disclosure relates to computer systems and software, and more particularly to techniques for facilitating and automating the provision of services in a cloud environment.

[0003] Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services). The services provided or accessed through the cloud (or network) are referred to as cloud services. There is a lot of processing that needs to be performed by a cloud service provider to make cloud services available to a subscribing customer. Due to its complexity, much of this processing is still done manually. For example, provisioning resources for providing such cloud services can be a very labor-intensive process.

[0004] Consumers and businesses have an expectation that ordering and starting the use of computer network cloud-based services be seamless. However, many cloud services are difficult to start up, requiring the provisioning of resources that might be shared. For example, servers in a server farm may run cloud services for many customers at once, and adding another customer may require shifting around resources. Different users may wish to employ different services. For example, some users may only want raw storage on the cloud, while other users may want to use sophisticated database storage. Still others may want the use of graphical user interface (GUI) software applications running on the cloud for its employees or customers.

[0005] There exists a need in the art for faster, less expensive, and easier-to-use cloud based computer systems.

SUMMARY

[0006] Generally, aspects of the present disclosure relate to enabling already provisioned services, such as database services and Java® services, in a computer network cloud infrastructure system to be reused in order to reduce the time to provision new services.

[0007] In embodiments, a data center selection system can select a chosen data center (DC) for an order submitted to a cloud computing system using a preprocessing layer and a rules engine that incorporates action/algorithm-based selection using data center metrics to determine the chosen DC. In various embodiments, the data center selection system retrieves order information, objectives, rules, algorithms, and other data defined by an administrator. The data center selection system can then retrieve data center information aggregated from various data centers. Using order information received from an order management system, the data center selection system can utilizes the preprocessing layer, the rules engine, and the algorithm-based selection to select the chosen DC. The data center selection system can send an order request to the chosen DC in which to provision services for the order request.

[0008] This summary is not intended to identify key or essential features of the claimed subject matter, nor is it intended to be used in isolation to determine the scope of the claimed subject matter. The subject matter should be understood by reference to appropriate portions of the entire specification of this patent, any or all drawings, and each claim.

BRIEF DESCRIPTION OF THE DRAWINGS

[0009] Illustrative embodiments of the present invention are described in detail below with reference to the following drawing figures:

[0010] FIG. **1**A is a logical view of a cloud infrastructure system according to one embodiment of the present invention.

[0011] FIG. **1**B is a simplified block diagram of a hardware/software stack that may be used to implement a cloud infrastructure system according to an embodiment of the present invention.

[0012] FIG. **2** is a simplified block diagram of a system environment for implementing the cloud infrastructure system shown in FIG. **1**A.

[0013] FIG. **3**A depicts a simplified flowchart **300** depicting processing that may be performed by the TAS module in the cloud infrastructure system, in accordance with an embodiment of the present invention.

[0014] FIG. **3**B depicts a simplified high-level diagram of one or more sub-modules in the TAS module in the cloud infrastructure system, in accordance with an embodiment of the present invention.

[0015] FIG. **4** depicts an exemplary distributed deployment of the TAS component, according to an embodiment of the present invention.

[0016] FIG. **5** is a simplified block diagram illustrating the interactions of the SDI module with one or more modules in the cloud infrastructure system, in accordance with an embodiment of the present invention.

[0017] FIG. **6** depicts a simplified high-level diagram of sub-modules of the SDI module according to an embodiment of the present invention.

[0018] FIG. **7**A depicts a simplified flowchart depicting processing that may be performed by the SDI component in the cloud infrastructure system, in accordance with an embodiment of the present invention.

[0019] FIG. **7**B depicts a simplified block diagram showing the high-level architecture of a Nuviaq system **710** and its relationships with other cloud infrastructure components according to an embodiment of the present invention.

[0020] FIG. **7**C depicts an example sequence diagram illustrating steps of a provisioning process using a Nuviaq system according to an embodiment of the present invention.

[0021] FIG. **7**D depicts an example sequence diagram illustrating steps of a deployment process using a Nuviaq system according to an embodiment of the present invention.

[0022] FIG. **7**E depicts an example of database instances provisioned for a database service according to an embodiment of the present invention.

[0023] FIG. **8** illustrates a high level overview of the various interactions involved to provide intelligent data center selection in accordance with an embodiment of the present invention.

[0024] FIGS. **9**A-**9**B are a flowchart of a method for selecting a data center to host a cloud subscription according to various embodiments.

[0025] FIG. **10** is a simplified flowchart of a method for selecting a data center to host a cloud subscription according to various embodiments.

[0026] FIG. **11** depicts a simplified diagram of a distributed system for implementing one of the embodiments.

[0027] FIG. **12** illustrates an exemplary computer system, in which various embodiments of the present invention may be implemented.

DETAILED DESCRIPTION

[0028] In the following description, for the purposes of explanation, specific details are set forth in order to provide a thorough understanding of embodiments of the invention. However, it will be apparent that various embodiments may be practiced without these specific details. The figures and description are not intended to be restrictive.

[0029] Certain embodiments of the present invention provide techniques for automating the provisioning, managing and tracking of services provided by a cloud infrastructure system.

Introduction

[0030] In certain embodiments, a cloud infrastructure system may include a suite of applications, middleware and database service offerings that are delivered to a customer in a self-service, subscription-based, elastically scalable, reliable, highly available, and secure manner. An example of such a cloud infrastructure system is the Oracle Public Cloud provided by the present assignee.

[0031] A cloud infrastructure system may provide many capabilities including, but not limited to, provisioning, managing and tracking a customer's subscription for services and resources in the cloud infrastructure system, providing predictable operating expenses to customers utilizing the services in the cloud infrastructure system, providing robust identity domain separation and protection of a customer's data in the cloud infrastructure system, providing customers with a transparent architecture and control of the design of the cloud infrastructure system, providing customers assured data protection and compliance with data privacy standards and regulations, providing customers with an integrated development experience for building and deploying services in the cloud infrastructure system and providing customers with a seamless integration between business software, middleware, database and infrastructure services in the cloud infrastructure system.

[0032] In certain embodiments, services provided by the cloud infrastructure system may include a host of services that are made available to users of the cloud infrastructure system on demand such as online data storage and backup solutions, Web-based e-mail services, hosted office suites and document collaboration services, database processing, managed technical support services and the like. Services provided by the cloud infrastructure system can dynamically scale to meet the needs of its users. A specific instantiation of a service provided by cloud infrastructure system is referred to herein as a service instance. In general, any service made available to a user via a communication network such as the Internet from a cloud service provider's system is referred to as a cloud service. Typically, in a public cloud environment, servers and systems that make up the cloud service provider's system are different from the customer's own on-premises servers and systems. For example, a cloud service provider's system may host an application and a user may, via a communication network such as the Internet, on demand, order and use the application.

[0033] A service in a computer network cloud infrastructure includes protected computer network access to storage, a hosted database, a hosted web server, a software application, or other service provided by a cloud vendor to a user, or as otherwise known in the art. For example, a service can include password-protected access to remote storage on the cloud through the Internet. As another example, a service can include a web service-based hosted relational database and script-language middleware engine for private use by a networked developer. As another example, a service can include access to an email software application hosted on a cloud vendor's web site.

[0034] FIG. **1**A is a logical view of a cloud infrastructure system according to one embodiment of the present invention. Cloud infrastructure system **100** may provide a variety of services via a cloud or networked environment. These services may include one or more services provided under Software as a Service (SaaS) category, Platform as a Service (PaaS) category, Infrastructure as a Service (IaaS) category, or other categories of services including hybrid services. A customer, via a subscription order, may order one or more services provided by cloud infrastructure system **100**. Cloud infrastructure system **100** then performs processing to provide the services in the customer's subscription order.

[0035] Cloud infrastructure system **100** may provide the cloud services via different deployment models. For example, services may be provided under a public cloud model where cloud infrastructure system **100** is owned by an organization selling cloud services (e.g., owned by Oracle) and the services are made available to the general public or different industry enterprises. As another example, services may be provided under a private cloud model where cloud infrastructure system **100** is operated solely for a single organization and may provide services for one or more entities within the organization. The cloud services may also be provided under a community cloud model where cloud infrastructure system **100** and the services provided by system **100** are shared by several organizations in a related community. The cloud services may also be provided under a hybrid cloud model, which is a combination of two or more different models.

[0036] As shown in FIG. **1**A, cloud infrastructure system **100** may comprise multiple components, which working in conjunction, enable provision of services provided by cloud infrastructure system **100**. In the embodiment illustrated in FIG. **1**A, cloud infrastructure system **100** includes a SaaS platform **102**, a PaaS platform **104**, an IaaS platform **110**, infrastructure resources **106**, and cloud management functionality **108**. These components may be implemented in hardware, or software, or combinations thereof.

[0037] SaaS platform **102** is configured to provide cloud services that fall under the SaaS category. For example, SaaS platform **102** may provide capabilities to build and deliver a suite of on-demand applications on an integrated development and deployment platform. SaaS platform **102** may manage and control the underlying software and infrastructure for providing the SaaS services. By utilizing the services provided by SaaS platform **102**, customers can utilize applications executing on cloud infrastructure system **100**. Customers can acquire the application services without the need for customers to purchase separate licenses and support.

[0038] Various different SaaS services may be provided. Examples include without limitation services that provide solutions for sales performance management, enterprise integration, and business flexibility for large organizations, and the like. In one embodiment, the SaaS services may include Customer Relationship Management (CRM) services **110** (e.g., Fusion CRM services provided by the Oracle cloud), Human Capital Management (HCM)/Talent Management services **112**, and the like. CRM services **110** may include services directed to reporting and management of a sales activity cycle to a customer, and others. HCM/Talent services **112** may include services directed to providing global workforce lifecycle management and talent management services to a customer.

[0039] Various different PaaS services may be provided by PaaS platform **104** in a standardized, shared and elastically scalable application development and deployment platform. Examples of PaaS services may include without limitation services that enable organizations (such as Oracle) to consolidate existing applications on a shared, common architecture, as well as the ability to build new applications that leverage the shared services provided by the platform. PaaS platform **104** may manage and control the underlying software and infrastructure for providing the PaaS services. Customers can acquire the PaaS services provided by cloud infrastructure system **100** without the need for customers to purchase separate licenses and support. Examples of PaaS services include without limitation Oracle Java Cloud Service (JCS), Oracle Database Cloud Service (DBCS), and others.

[0040] By utilizing the services provided by PaaS platform **104**, customers can utilize programming languages and tools supported by cloud infrastructure system **100** and also control the deployed services. In some embodiments, PaaS services provided by the cloud infrastructure system **100** may include database cloud services **114**, middleware cloud services (e.g., Oracle Fusion Middleware services) **116** and Java cloud services **117**. In one embodiment, database cloud services **114** may support shared service deployment models that enable organizations to pool database resources and offer customers a database-as-a-service in the form of a database cloud, middleware cloud services **116** provides a platform for customers to develop and deploy various business applications and Java cloud services **117** provides a platform for customers to deploy Java applications, in the cloud infrastructure system **100**. The components in SaaS platform **102** and PaaS platform **104** illustrated in FIG. **1**A are meant for illustrative purposes only and are not intended to limit the scope of embodiments of the present invention. In alternate embodiments, SaaS platform **102** and PaaS platform **104** may include additional components for providing additional services to the customers of cloud infrastructure system **100**.

[0041] Various different IaaS services may be provided by IaaS platform **110**. The IaaS services facilitate the management and control of the underlying computing resources such as storage, networks, and other fundamental computing resources for customers utilizing services provided by the SaaS platform and the PaaS platform.

[0042] In certain embodiments, cloud infrastructure system **100** includes infrastructure resources **106** for providing the resources used to provide various services to customers of the cloud infrastructure system **100**. In one embodiment, infrastructure resources **106** includes pre-integrated and optimized combinations of hardware such as servers, storage and networking resources to execute the services provided by the PaaS platform and the SaaS platform.

[0043] In certain embodiments, cloud management functionality **108** provides comprehensive management of cloud services (e.g., SaaS, PaaS, IaaS services) in the cloud infrastructure system **100**. In one embodiment, cloud management functionality **108** includes capabilities for provisioning, managing, and tracking a customer's subscription received by the cloud infrastructure system **100**, and the like.

[0044] FIG. **1**B is a simplified block diagram of a hardware/software stack that may be used to implement cloud infrastructure system **100** according to an embodiment of the present invention. It should be appreciated that implementation depicted in FIG. **1**B may have other components than those depicted in FIG. **1**B. Further, the embodiment shown in FIG. **1**B is only one example of a cloud infrastructure system that may incorporate an embodiment of the invention. In some other embodiments, cloud infrastructure system **100** may have more or fewer components than shown in FIG. **1**B, may combine two or more components, or may have a different configuration or arrangement of components. In certain embodiments, the hardware and software components are stacked so as to provide vertical integration that provides optimal performance.

[0045] Various types of users may interact with cloud infrastructure system **100**. These users may include, for example, end users **150** that can interact with cloud infrastructure system **100** using various client devices such as desktops, mobile devices, tablets, and the like. The users may also include developers/programmers **152** who may interact with cloud infrastructure system **100** using command line interfaces (CLIs), application-programming interfaces (APIs), through various integrated development environments (IDEs), and via other applications. User may also include operations personnel **154**. These may include personnel of the cloud service provider or personnel of other users.

[0046] Application services layer **156** identifies various cloud services that may be offered by cloud infrastructure system **100**. These services may be mapped to or associated with respective software components **160** (e.g., Oracle WebLogic server for providing Java services, oracle database for providing database services, and the like) via a service integration and linkages layer **158**.

[0047] In certain embodiments, a number of internal services **162** may be provided that are shared by different components or modules of cloud infrastructure system **100** and by the services provided by cloud infrastructure system **100**. These internal shared services may include, without limitation, a security and identity service, an integration service, an enterprise repository service, an enterprise manager service, a virus scanning and white list service, a high availability, backup and recovery service, service for enabling cloud support in IDEs, an email service, a notification service, a file transfer service, and the like.

[0048] Runtime infrastructure layer **164** represents the hardware layer on which the various other layers and components are built. In certain embodiments, runtime infrastructure layer **164** may comprise one Oracle's Exadata machines for providing storage, processing, and networking resources. An Exadata machine may be composed of various database servers, storage Servers, networking resources, and other components for hosting cloud-services related software layers. In certain embodiments, the Exadata machines may be designed to work with Oracle Exalogic, which is an engineered system providing an assemblage of storage, compute, network, and software resources. The combination of Exadata and Exalogic provides a complete hardware and software-engineered solution that delivers high-performance, highly available, scalable, secure, and a managed platform for providing cloud services.

[0049] FIG. **2** is a simplified block diagram of a system environment for implementing the cloud infrastructure system shown in FIG. **1**A according to an embodiment of the present invention. In the illustrated embodiment, system environment **230** includes one or more client computing devices **224**, **226**, and **228** that may be used by users to interact with cloud infrastructure system **100**. A client device may be configured to operate a client application such as a web browser, a proprietary client application (e.g., Oracle Forms), or some other application, which may be used by a user of the client device to interact with cloud infrastructure system **100** to utilize services provided by cloud infrastructure system **100**.

[0050] It should be appreciated that cloud infrastructure system **100** depicted in FIG. **2** may have other components than those depicted in FIG. **2**. Further, the embodiment shown in FIG. **2** is only one example of a cloud infrastructure system that may incorporate an embodiment of the invention. In some other embodiments, cloud infrastructure system **100** may have more or fewer components than shown in FIG. **2**, may combine two or more components, or may have a different configuration or arrangement of components.

[0051] Client computing devices **224**, **226** and **228** may be general purpose personal computers (including, by way of example, personal computers and/or laptop computers running various versions of Microsoft Windows and/or Apple Macintosh operating systems), cell phones or PDAs (running software such as Microsoft Windows Mobile and being Internet, e-mail, SMS, Blackberry, or other communication protocol enabled), workstation computers running any of a variety of commercially-available UNIX or UNIX-like operating systems (including without limitation the variety of GNU/Linux operating systems), or any other computing device. For example, client computing devices **224**, **226** and **228** may be any other electronic device, such as a thin-client computer, Internet-enabled gaming system, and/or personal messaging device, capable of communicating over a network (e.g., network **232** described below). Although exemplary system environment **230** is shown with three client computing devices, any number of client computing devices may be supported. Other devices such as devices with sensors, etc. may interact with cloud infrastructure system **100**.

[0052] A network **232** may facilitate communications and exchange of data between clients **224**, **226** and **228** and cloud infrastructure system **100**. Network **232** may be any type of network familiar to those skilled in the art that can support data communications using any of a variety of commercially available protocols, including without limitation TCP/IP, SNA, IPX, AppleTalk, and the like. Merely by way of example, network **232** can be a local area network (LAN) such as an Ethernet network, a Token-Ring network and/or the like, a wide-area network, a virtual network, including without limitation a virtual private network (VPN), the Internet, an intranet, an extranet, a public switched telephone network (PSTN), an infra-red network, a wireless network (e.g., a network operating under any of the IEEE 802.1X suite of protocols, the Bluetooth protocol known in the art, and/or any other wireless protocol), and/or any combination of these and/or other networks.

[0053] Cloud infrastructure system **100** may comprise one or more computers and/or servers which may be general purpose computers, specialized server computers (including, by way of example, PC servers, UNIX servers, mid-range servers, mainframe computers, rack-mounted servers, etc.), server farms, server clusters, or any other appropriate arrangement and/or combination. The computing devices that make up cloud infrastructure system **100** may run any of operating systems or a variety of additional server applications and/or mid-tier applications, including HTTP servers, FTP servers, CGI servers, Java servers, database servers, and the like. Exemplary database servers include without limitation those commercially available from Oracle, Microsoft, Sybase, IBM and the like.

[0054] In various embodiments, cloud infrastructure system **100** may be adapted to automatically provision, manage and track a customer's subscription to services offered by cloud infrastructure system **100**. In one embodiment, as depicted in FIG. **2**, the components in cloud infrastructure system **100** include an Identity Management (IDM) module **200**, a services module **202**, a Tenant Automation System (TAS) module **204**, a Service Deployment Infrastructure (SDI) module **206**, an Enterprise Manager (EM) module **208**, one or more front-end web interfaces such as a store user interface (UI) **210**, a cloud user interface (UI) **212**, and a support user interface (UI) **216**, an order management module **214**, sales personnel **218**, operator personnel **220** and an order database **224**. These modules may include or be provided using one or more computers and/or servers which may be general purpose computers, specialized server computers, server farms, server clusters, or any other appropriate arrangement and/or combination. In one embodiment, one or more of these modules can be provided by cloud management functionality **108** or IaaS platform **110** in cloud infrastructure system **100**. The various modules of the cloud infrastructure system **100** depicted in FIG. **2** are meant for illustrative purposes only and are not intended to limit the scope of embodiments of the present invention. Alternative embodiments may include more or fewer modules than those shown in FIG. **2**.

[0055] In an exemplary operation, at (1) a customer using a client device such as client device **224** or **226** may interact with cloud infrastructure system **100** by browsing the various services provided by cloud infrastructure system **100** and placing an order for a subscription for one or more services offered by cloud infrastructure system **100**. In certain embodiments, the customer may access store UI **210** or cloud UI **212** and place a subscription order via these user interfaces.

[0056] The order information received by cloud infrastructure system **100** in response to the customer placing an order may include information identifying the customer and one or more services offered by the cloud infrastructure system **100** to which the customer intends to subscribe. A single order may include orders for multiple services. For instance, a customer may login to cloud UI **212** and request a subscription for a CRM service and a Java cloud service in the same order.

[0057] Additionally, the order may also include one or more service levels for the ordered services. As used herein, and as will be discussed in greater detail below, a service level for a service determines the amount of resources to be allocated for providing the requested service in the context of the subscription, such as the amount of storage, amount of computing resources, data transfer facilities, and the like. For example, a basic service level may provide a minimum level of storage, data transmission, or number of users, and higher service levels may include additional resources.

[0058] In addition, in some instances, the order information received by cloud infrastructure system **100** may include information indicative of a customer level, and the time period during which the service is desired. The customer level specifies the priority of the customer making the subscription request. In one example, the priority may be determined based on the quality of service that the cloud infrastructure system **100** guarantees or promises the customer as specified by a Service Level Agreement (SLA) agreed to between the customer and the provider of the cloud services. In one example, the different customer levels include a basic level, a silver level, and a gold level. The time period for a service may specify the start date and time for the service and the time period for which the service is desired (e.g., a service end date and time may be specified).

[0059] In one embodiment, a customer may request a new subscription via store UI **210** or request for a trial subscription via cloud UI **212**. In certain embodiments, store UI **210** may represent the service provider's eCommerce storefront (e.g., www.oracle.com/store for Oracle Cloud services). Cloud UI **212** may represent a business interface for the service provider. Consumer can explore available services and sign up for interested services through cloud UI **212**. Cloud UI **212** captures user input necessary for ordering trial subscriptions provided by cloud infrastructure system **100**. Cloud UI **212** may also be used to view account features and configure the runtime environment located within cloud infrastructure system **100**. In addition to placing an order for a new subscription, store UI **210** may also enable the customer to perform other subscription-related tasks such as changing the service level of a subscription, extending the term of the subscription, increasing the service level of a subscription, terminating an existing subscription, and the like.

[0060] After an order has been placed per (1), at (2), the order information that is received via either store UI **210** or cloud UI **212** is stored in order database **224**, which can be one of several databases operated by cloud infrastructure system **100** and utilized in conjunction with other system elements. While order database **224** is shown logically as a single database in FIG. **2**, in actual implementation, this may comprise one or more databases.

[0061] At (3), the order is forwarded to order management module **214**. Order management module **214** is configured to perform billing and accounting functions related to the order such as verifying the order and upon verification, booking the order. In certain embodiments, order management module **214** may include a contract management module and an install base module. The contract management module may store contract information associated with the customer's subscription order such as the customer's service level agreement (SLA) with cloud infrastructure system **100**. The install base module may include detailed descriptions of the services in the customer's subscription order. In addition to order information, the install base module may track installation details related to the services, product status and support service history related to the services. As a customer orders new services or upgrades existing ones, the install base module may automatically add new order information.

[0062] At (4), information regarding the order is communicated to TAS module **204**. In one embodiment, TAS module **204** utilizes the order information to orchestrate the provisioning of services and resources for the order placed by the customer. At (5), TAS component **204** orchestrates the provisioning of resources to support the subscribed services using the services of SDI module **206**. At (6) TAS module **204** provides information related to the provisioned order received from SDI module **206** to services module **202**. In some embodiments, at (7), SDI module **206** may also use services provided by services module **202** to allocate and configure the resources needed to fulfill the customer's subscription order.

[0063] At (8), services module **202** sends a notification to the customers on client devices **224**, **226** and **228** regarding the status of the order.

[0064] In certain embodiments, TAS module **204** functions as an orchestration component that manages business processes associated with each order and applies business logic to determine whether an order should proceed to provisioning. In one embodiment, upon receiving an order for a new subscription, TAS module **204** sends a request to SDI module **206** to allocate resources and configure those resources needed to fulfill the subscription order. SDI module **206** enables the allocation of resources for the services ordered by the customer. SDI module **206** provides a level of abstraction between the cloud services provided by cloud infrastructure system **100** and the physical implementation layer that is used to provision the resources for providing the requested services. TAS module **204** may thus be isolated from implementation details such as whether or not services and resources are actually provisioned on the fly or pre-provisioned and only allocated/assigned upon request.

[0065] In certain embodiments, a user may use store UI **210** to directly interact with order management module **214** to perform billing and accounting related functions such as verifying the order and upon verification, booking the order. In some embodiments, instead of a customer placing an order, at (9), the order may instead be placed by sales personnel **218** on behalf of the customer such as a customer's service representative or sales representative. Sales personnel **218** may directly interact with order management module **214** via a user interface (not shown in FIG. **2**) provided by order management module **214** for placing orders or for providing quotes for the customer. This, for example, may be done for large customers where the order may be placed by the customer's sales representative through order management module **214**. The sales representative may set up the subscription on behalf of the customer.

[0066] EM module **208** is configured to monitor activities related to managing and tracking a customer's subscription in cloud infrastructure system **100**. EM module **208** collects usage statistics for the services in the subscription order such as the amount of storage used, the amount data transferred, the number of users, and the amount of system up time and system down time. At (10), a host operator personnel **220**, who may be an employee of a provider of cloud infrastructure system **100**, may interact with EM module **208** via an enterprise manager user interface (not shown in FIG. **2**) to manage systems and resources on which services are provisioned within cloud infrastructure system **100**.

[0067] Identity management (IDM) module **200** is configured to provide identity services such as access management and authorization services in cloud infrastructure system **100**.

In one embodiment, IDM module **200** controls information about customers who wish to utilize the services provided by cloud infrastructure system **100**. Such information can include information that authenticates the identities of such customers and information that describes which actions those customers are authorized to perform relative to various system resources (e.g., files, directories, applications, communication ports, memory segments, etc.) IDM module **200** can also include the management of descriptive information about each customer and about how and by whom that descriptive information can be accessed and modified.

[0068] In one embodiment, information managed by the identity management module **200** can be partitioned to create separate identity domains. Information belonging to a particular identity domain can be isolated from all other identity domains. In addition, an identity domain can be shared by multiple separate tenants. Each such tenant can be a customer subscribing to services in the cloud infrastructure system **100**. In some embodiments, a customer can have one or many identity domains, and each identity domain may be associated with one or more subscriptions, each subscription having one or many services. For example, a single customer can represent a large entity and identity domains may be created for divisions/departments within this large entity. EM module **208** and IDM module **200** may in turn interact with order management module **214** at (11) and (12) respectively to manage and track the customer's subscriptions in cloud infrastructure system **100**.

[0069] In one embodiment, at (13), support services may also be provided to the customer via a support UI **216**. In one embodiment, support UI **216** enables support personnel to interact with order management module **214** via a support backend system to perform support services at (14). Support personnel in the cloud infrastructure system **100** as well as customers can submit bug reports and check the status of these reports via support UI **216**.

[0070] Other interfaces, not shown in FIG. **2** may also be provided by cloud infrastructure system **100**. For example, an identity domain administrator may use a user interface to IDM module **200** to configure domain and user identities. In addition, customers may log into a separate interface for each service they wish to utilize. In certain embodiments, a customer who wishes to subscribe to one or more services offered by cloud infrastructure system **100** may also be assigned various roles and responsibilities. In one embodiment, the different roles and responsibilities that may be assigned for a customer may include that of a buyer, an account administrator, a service administrator, an identity domain administrator or a user who utilizes the services and resources offered by cloud infrastructure system **100**. The different roles and responsibilities are described more fully in FIG. **4** below.

[0071] FIG. **3**A depicts a simplified flowchart **300** depicting processing that may be performed by the TAS module in the cloud infrastructure system, in accordance with an embodiment of the present invention. The processing depicted in FIG. **3**A may be implemented in software (e.g., code, instructions, program) executed by one or more processors, hardware, or combinations thereof. The software may be stored in memory (e.g., on a memory device, on a non-transitory computer-readable storage medium). The particular series of processing steps depicted in FIG. **3**A is not intended to be limiting. Other sequences of steps may also be performed according to alternative embodiments. For example, alternative embodiments of the present invention may perform the steps outlined above in a different order. Moreover, the individual steps illustrated in FIG. **3**A may include multiple sub-steps that may be performed in various sequences as appropriate to the individual step. Furthermore, additional steps may be added or removed depending on the particular applications. One of ordinary skill in the art would recognize many variations, modifications, and alternatives. In one embodiment, the processing depicted in FIG. **3**A may be performed by one or more components in TAS component **204** as will be described in detail in FIG. **3**B.

[0072] At **302**, a customer's subscription order is processed. The processing may include validating the order, in one example. Validating the order includes ensuring that the customer has paid for the subscription and ensuring that the customer does not already have subscriptions with the same name or that the customer is not attempting to create multiple subscriptions of the same type in the same identity domain for subscription types for which this is disallowed (such as, in the case of a CRM service). Processing may also include tracking the status of an order for each order that is being processed by cloud infrastructure system **100**.

[0073] At **304**, a business process associated with the order is identified. In some instances, multiple business processes may be identified for an order. Each business process identifies a series of steps for processing various aspects of the order. As an example, a first business process may identify one or more steps related to provisioning physical resources for the order, a second business process may identify one or more steps related to creating an identity domain along with customer identities for the order, a third business process may identify one or more steps for related to performing back office functions such as creating a customer record for the user, performing accounting functions related to the order, and the like. In certain embodiments, different business processes may also be identified for processing different services in an order. For example, different business process may be identified to process a CRM service and a database service.

[0074] At **306**, the business process identified for the order in **304** is executed. Executing the business process associated with the order may include orchestrating the series of steps associated with the business process identified in step **304**. For example, executing a business process related to provisioning physical resources for the order may include sending a request to SDI module **206** to allocate resources and configure those resources needed to fulfill the subscription order.

[0075] At **308**, a notification is sent to the customer regarding the status of the provisioned order. Additional description related to performing steps **302**, **304**, **306** and **308** is provided in detail in FIG. **3**B.

[0076] FIG. **3**B depicts a simplified high-level diagram of one or more sub-modules in the TAS module in the cloud infrastructure system, in accordance with an embodiment of the present invention. In one embodiment, the modules depicted in FIG. **3**B perform the processing described in steps **302-308** discussed in FIG. **3**A. In the illustrated embodiment, TAS module **204** comprises an order-processing module **310**, a business process identifier **312**, a business process executor **316**, an overage framework **322**, a workflow identification module **324**, and a bundled subscription generator module **326**. These modules may be implemented in hardware, or software, or combinations thereof. The various modules of the TAS module depicted in FIG. **3**B are meant for illustrative purposes only and are not intended to limit the scope of embodiments of the present invention. Alternative embodiments may include more or fewer modules than those shown in FIG. 3B.

[0077] In one embodiment, order processing module **310** receives an order from a customer from one or more input sources **321**. For example, order-processing module **310** may directly receive an order via cloud UI **212** or store UI **210**, in one embodiment. Alternatively, order-processing module **310** may receive an order from order management module **214** or order database **224**. Order processing module **310** then processes the order. In certain embodiments, processing the order includes generating a customer record which includes information about the order such as a service type, a service level, a customer level, the type of resources, the amount of the resources to be allocated to the service instance and a time period during which the service is desired. As part of the processing, order-processing module **310** also determines whether the order is a valid order. This includes ensuring that the customer does not already have subscriptions with the same name or that the customer is not attempting to create multiple subscriptions of the same type in the same identity domain for subscription types where this is disallowed (such as, in the case of a fusion CRM service).

[0078] Order processing module **310** may also perform additional processing on the order. Processing may include tracking the status of an order for each order that is being processed by cloud infrastructure system **100**. In one embodiment, order-processing module **310** may process each order to identify a number of states pertaining to the order. In one example, the different states of an order may be an initialized state, a provisioned state, an active state, an administration required state, an error state, and the like. An initialized state refers to the state of a new order; a provisioned state refers to the state of an order once the services and resources for the order have been provisioned. An order is in an active state when the order has been processed by TAS module **204** and a notification to that effect has been delivered to the customer. An order is in an administration-required state when intervention by an administrator is needed to resolve the issue. The order is in an error state when the order cannot be processed. In addition to maintaining the order progress status, order-processing module **310** also maintains detailed information about any failures encountered during process execution. In other embodiments, and as will be discussed in detail below, the additional processing performed by order processing module **310** may also include changing the service level for a service in the subscription, changing the services included in the subscription, extending the time period of the subscription, and canceling the subscription or specifying different service levels for different time periods in the subscription.

[0079] After an order has been processed by order processing module **310**, business logic is applied to determine whether the order should proceed to provisioning. In one embodiment, as part of orchestrating the order, business process identifier **312** receives the processed order from order processing module **310** and applies business logic to identify a particular business process to use for the order being processed. In one embodiment, business process identifier **312** may utilize information stored in a service catalog **314** to determine the particular business process to be used for the order. In one embodiment, and as discussed in FIG. 3A, multiple business processes may be identified for an order and each business process identifies a series of steps for processing various aspects of the order. In another embodiment, and as discussed above, different business processes may be defined for different types of services, or combinations of services such as a CRM service or a database service. In one embodiment, service catalog **314** may store information mapping an order to a particular type of business process. Business process identifier **312** may use this information to identify a specific business process for the order being processed.

[0080] Once a business process has been identified, business process identifier **312** communicates the particular business process to be executed to business process executor **316**. Business process executor **316** then executes steps of the identified business process by operating in conjunction with one or more modules in the cloud infrastructure system **100**. In some embodiments, business process executor **316** acts as an orchestrator for performing the steps associated with a business process. For example, the business process executor may interact with order processing module **310** to execute steps in a business process that identifies workflows related to the order, determines the overage of services in the order, or identifies service components related to the order.

[0081] In one example, business process executor **316** interacts with SDI module **206** to execute steps in a business process for allocating and provisioning resources for services requested in the subscription order. In this example, for each step in the business process, business process executor **316** may send a request to SDI component **206** to allocate resources and configure resources needed to fulfill the particular step. SDI component **206** is responsible for the actual allocation of the resources. Once all the steps of the business processes of an order have been executed, business process executor **316** may send a notification to the customer of the processed order by utilizing the services of services component **202**. The notification may include sending an email notification to the customer with details of the processed order. The email notification may also include deployment information related to the order to enable the customer to access the subscribed services.

[0082] In certain embodiments, TAS module **204** may provide one or more TAS Application Programming Interfaces (APIs) **318** that enable TAS module **204** to interact with other modules in cloud infrastructure system **100** and for other modules to interact with TAS module **204**. For example, the TAS APIs may include a system provisioning API that interacts with SDI module **206** via an asynchronous Simple Object Access Protocol (SOAP) based web services call to provision resources for the customer's subscription order. In one embodiment, TAS module **204** may also utilize the system provisioning API to accomplish system and service instance creation and deletion, switch a service instance to an increased service level, and associate service instances. An example of this is the association of a Java service instance to a fusion applications service instance to allow secure web service communications. The TAS APIs may also include a notification API that interacts with the services module **202** to notify the customer of a processed order. In certain embodiments, the TAS module **204** also periodically propagates subscription information, outages, and notifications (e.g. planned downtime) to services component **202**.

[0083] In certain embodiments, TAS module **204** periodically receives usage statistics for each of the provisioned services such as the amount of storage used, the amount data transferred, the number of users, and the amount of system up time and system down time from EM module **208**. Overage framework **322** utilizes the usage statistics to determine whether over use of a service has occurred, and if so, to determine how much to bill for the overage, and provides this information to order management module **214**.

[0084] In certain embodiments, TAS module **204** includes an order workflow identification module **324** that is configured to identify one or more workflows associated with processing a customer's subscription order. In certain embodiments, TAS module **204** may include a subscription order generation framework **326** for generating subscription orders for a customer when the customer places a subscription order for one or more services offered by the cloud infrastructure system **100**. In one embodiment, a subscription order includes one or more service components responsible for providing the services requested by a customer in the subscription order.

[0085] Additionally, TAS module **204** may also interact with one or more additional databases such as a Tenant Information System (TIS) database **320** to enable the provisioning of resources for one or more services subscribed by the customer while taking into consideration historical information, if any, available for the customer. TIS database **320** may include historical order information and historical usage information pertaining to orders subscribed by the customer.

[0086] TAS module **204** may be deployed using different deployment models. In certain embodiments, the deployment includes a central component that interfaces with one or more distributed components. The distributed components may, for example, be deployed as various data centers and accordingly may also be referred to as data center components. The central component includes capabilities to process orders and co-ordinate services in cloud infrastructure system **100**, while the data center components provide capabilities for provisioning and operating the runtime system that provides the resources for the subscribed services.

[0087] FIG. **4** depicts an exemplary distributed deployment of the TAS module, according to an embodiment of the present invention. In the embodiment depicted in FIG. **4**, the distributed deployment of TAS module **204** includes a TAS central component **400** and one or more TAS Data Centers (DCs) components **402**, **404**, and **406**. These components may be implemented in hardware, or software, or combinations thereof.

[0088] In one embodiment, the responsibilities of TAS central component **400** include, without limitation, to provide a centralized component for receiving customer orders, performing order-related business operations such as creating a new subscription, changing the service level for a service in the subscription, changing the services included in the subscription, and extending the time period of the subscription, or canceling the subscription. The responsibilities of TAS central component **400** may also include maintaining and serving subscription data needed by cloud infrastructure system **100** and interfacing with order management module **214**, support UI **216**, cloud UI **212**, and store UI **210** to handle all the back-office interactions.

[0089] In one embodiment, the responsibilities of TAS DCs **402**, **404** and **406** include, without limitation, performing runtime operations for orchestrating the provisioning the resources for one or more services subscribed by the customer. TAS DCs **402**, **404** and **406** also include capabilities to perform operations such as locking, unlocking, enabling, or disabling a subscription order, collecting metrics related to the order, determining the status of the order, and sending notification events related to the order.

[0090] In an exemplary operation of the distributed TAS system shown in FIG. **4**, TAS central component **400** initially receives an order from a customer via cloud UI **212**, store UI **210**, via order management system **214**, or via order database **224**. In one embodiment, the customer represents a buyer who has financial information and the authority to order and/or change a subscription. In one embodiment, the order information includes information identifying the customer, the type of services that the customer wishes to subscribe to, and an account administrator who will be responsible for handling the request. In certain embodiments, the account administrator may be nominated by the customer when the customer places an order for a subscription to one or more services offered by cloud infrastructure system **100**. Based on the order information, the TAS central component **400** identifies the data region of the world such as Americas, EMEA, or Asia Pacific in which the order originates and the particular TAS DCs (for e.g., **402**, **404** or **406**) that will be deployed for provisioning the order. In one embodiment, the particular TAS DC (for e.g., from among DCs **402**, **404** or **406**) that will be deployed for provisioning the order is determined based on the geographical data region in which the request originated.

[0091] TAS central component **400** then sends the order request to the particular TAS DC in which to provision services for the order request. In one embodiment, TAS DCs **402**, **404** or **406** identify a service administrator and an identity domain administrator responsible for processing the order request at the particular TAS DC. The service administrator and the identity administrator may be nominated by the account administrator identified in the subscription order. TAS DCs **402**, **404** or **406** communicate with SDI module **204** to orchestrate the provisioning of physical resources for the order. SDI component **204** in respective TAS DCs **402**, **404** or **406** allocates resources and configures those resources needed to fulfill the subscription order.

[0092] In certain embodiments, TAS DCs, **402**, **404** or **406** identify an identity domain associated with the subscription. SDI component **206** may provide the identity domain information to IDM component **200** (shown in FIG. **2**) for identifying an existing identity domain or creating a new identity domain. Once the order is provisioned by the SDI module at respective TAS DCs, **402**, **404** or **406**, TAS central component **400** may place information regarding the provisioned resources in a support system, via support UI **216**. Information may include, for example, displaying resource metrics related to the services and usage statistics of the services.

[0093] Once in operation, at each data center, EM module **208** to periodically collects usage statistics for each of the provisioned services provisioned at that data center, such as the amount of storage used, the amount data transferred, the number of users, and the amount of system up time and system down time. These statistics are provided to the TAS DC that is local to EM module **208** (i.e., at the same data center). In an embodiment, the TAS DCs may use the usage statistics to determine whether overuse of a service has occurred, and if so, to determine how much to bill for the overage, and provide the billing information to order management system **214**.

[0094] FIG. **5** is a simplified block diagram illustrating the interactions of the SDI module with one or more modules in the cloud infrastructure system, in accordance with an embodiment of the present invention. In one embodiment, SDI module **206** interacts with TAS module **204** to provision resources for services in a subscription order received by TAS module **204**. In certain embodiments, one or more of the modules illustrated in FIG. **5** may be modules within cloud infrastructure system **100**. In other embodiments, one or more of the modules that interact with SDI module **206** may be outside cloud infrastructure system **100**. In addition, alternative embodiments may have more or less modules than those shown in FIG. **5**. These modules may be implemented in hardware, or software, or combinations thereof.

[0095] In one embodiment, the modules in SDI module **206** may include one or more modules in SaaS platform **102** and PaaS platform **104** in cloud infrastructure system **100**. In order to perform provisioning of resources for various services, SDI module **206** may interact with various other modules, each customized to help with provisioning resources for a particular type of service. For example, as illustrated in FIG. **5**, SDI module **206** may interact with a Java service provisioning control module **500** to provision Java cloud services. In one embodiment, Java service provisioning control component **500** may deploy a Java Cloud Service (JCS) assembly specified by SDI module **206** that includes a set of tasks to be performed to provision Java cloud services. Infrastructure resources **106** then determines the resources needed to provision the Java cloud services.

[0096] As other examples, SDI module **206** may interact with one or more modules such as a Virtual Assembly Builder (VAB) module **502**, an Application Express (APEX) deployer module **504**, a Virtual Machine (VM) module **506**, an IDM module **200**, and a database machine module **118**. VAB module **502** includes capabilities to configure and provision complete multi-tier application environments. In one embodiment, VAB module **502** deploys a Middleware (MW) service assembly specified by SDI module **206** to provision a MW service in cloud infrastructure system **100** using the services provided by VM module **506**. APEX deployer module **504** includes capabilities to configure and provision database services. In one embodiment, APEX deployer module **504** deploys a database service assembly specified by SDI module **206** to provision a database service in cloud infrastructure system **100** using the resources provided by infrastructure resources **106**. SDI module **206** interacts with IDM module **200** to provide identity services such as access management across multiple applications in cloud infrastructure system **100**.

[0097] FIG. **6** depicts a simplified high-level diagram of sub-modules of the SDI module according to an embodiment of the present invention. In the embodiment depicted in FIG. **6**, SDI module **206** includes a SDI-Web Services (WS) module **600**, an SDI request controller module **602**, an SDI task manager module **604**, an SDI monitoring module **606**, an SDI data access module **608**, an SDI common library module **610**, and an SDI connector module **612**. These modules may be implemented in hardware, or software, or combinations thereof. SDI module **206** depicted in FIG. **6** and its various modules are meant for illustrative purposes only and are not intended to limit the scope of embodiments of the present invention. Alternative embodiments may have more or less modules than those shown in FIG. **6**. These modules and their functions are described in detail below.

[0098] SDI-WS module **600** includes capabilities for receiving a step in the business associated with an order from business process executor **316** of TAS component **204**. In one embodiment, SDI-WS module **600** parses each step of the business process and converts the step into an internal representation used by SDI module **206**. In one embodiment, each step of the business process associated with the order arrives through a web service processing layer (for example, via System Provisioning API discussed in FIG. 3B) in the form of a SOAP request to SDI-WS module **600**.

[0099] SDI request controller module **602** is the internal request-processing engine in SDI module **206** and includes capabilities for performing asynchronous request processing, concurrent request processing, concurrent task processing, fault tolerant and recovery and plug-in support related to the order requests. In one embodiment, SDI request controller module **602** accepts each step of the business process associated with the order from SDI-WS module **600** and submits the step to SDI task manager module **604**.

[0100] SDI task manager module **604** translates each step specified in the business process into a series of tasks for provisioning the particular step. Once the set of tasks for a specific step have been provisioned, SDI task manager module **604** responds to business process executor **316** in TAS module **204** with operation results that includes an order payload with details of the resources provisioned to fulfill the particular step. SDI task manager module **604** repeats this process until all the steps of the particular business process associated with the order are complete.

[0101] In certain embodiments, SDI task manager module **604** translates each step specified in the business process into a series of tasks by utilizing the services of SDI connector module **612**. SDI connector module **612** includes one or more connectors for handling the deployment of tasks specified by SDI task manager module **604** to provision one or more services related to the order request. In certain embodiments, one or more of the connectors may handle tasks that are specific to a particular service type while other connectors may handle tasks that are common across different service types. In one embodiment, SDI connector module **612** includes a set of connectors (wrapper APIs) that interface with one or more of the external modules (shown in FIG. **5**) in cloud infrastructure system **100** to provision the services and resources related to the order request. For example, Application Express (APEX) connector **614** interfaces with APEX deployer module **504** to provision database services. Web Center Connector **616** (WCC) interfaces with a web center module in cloud infrastructure system **100** to provision web services. The web center module is a user engagement platform and includes capabilities for delivering connectivity between people and information in cloud infrastructure system **100**.

[0102] In certain embodiments, Middleware Applications (MA) connector **618** interfaces with VAB module **502** in cloud infrastructure system **100** to provision middleware application services. NUVIAQ connector **620** interfaces with VAB module **502** to provision Java services. IDM connector **622** interfaces with IDM module **200** to provide identity and access management for users subscribing to services and resources in cloud infrastructure system **100**. Virtual Assembly Builder (VAB) connector **624** interfaces with VAB module **502** in cloud infrastructure system **100** to configure and provision complete multi-tier application environments. Plug-in connector **626** interfaces with EM module **208** to manage and monitor the components in cloud infrastructure system **100**. HTTP server connector **628** interfaces with one or more web servers in the PaaS platform to provide connection services to users in cloud infrastructure system **100**.

[0103] SDI monitoring module **606** in SDI module **206** provides an inbound interface for receiving Java Management

Extensions (JMX) requests. SDI monitoring module **606** also provides tools for managing and monitoring applications, system objects, and devices in cloud infrastructure system **100**. SDI-data access module **608** provides an inbound interface for receiving Java Database Connectivity (JDBC) requests. SDI-data access module **608** supports data access and provides object relational mapping, java transaction API services, data access objects, and connection pooling in cloud infrastructure system **100**. The SDI-common library module **610** provides configuration support for the modules in SDI module **206**.

[0104] The embodiment of FIG. **6** discussed above describes modules in the SDI module according to an embodiment of the present invention. FIG. **7**A depicts a simplified flowchart **700** depicting processing that may be performed by the modules of the SDI module in the cloud infrastructure system, in accordance with an embodiment of the present invention. The processing depicted in FIG. **7**A may be implemented in software (e.g., code, instructions, program) executed by one or more processors, hardware, or combinations thereof. The software may be stored in memory (e.g., on a memory device, on a non-transitory computer-readable storage medium). The particular series of processing steps depicted in FIG. **7**A is not intended to be limiting. Other sequences of steps may also be performed according to alternative embodiments. For example, alternative embodiments of the present invention may perform the steps outlined above in a different order. Moreover, the individual steps illustrated in FIG. **7**A may include multiple sub-steps that may be performed in various sequences as appropriate to the individual step. Furthermore, additional steps may be added or removed depending on the particular applications. One of ordinary skill in the art would recognize many variations, modifications, and alternatives. In one embodiment, the processing depicted in FIG. **7**A may be performed by one or more modules in the SDI module **206** discussed in detail in FIG. **6**.

[0105] At **702**, a business process associated with a subscription order is received. In one embodiment, SDI-WS module **600** in SDI module **206** receives one or more steps in the business process associated with the subscription order from business process executor **316**. At **704**, each step in the business process is translated into a series of tasks for provisioning resources for the subscription order. In one embodiment, SDI task manager module **604** in SDI module **206** translates each step specified in the business process into a series of tasks by utilizing the services of SDI connector module **612**. At **706**, the subscription order is provisioned based on the series of tasks. In one embodiment, and as discussed in FIG. **6**, SDI connector module **612** includes one or more connectors for handling the deployment of tasks specified by SDI task manager module **604** to provision resources for the services in the subscription order.

[0106] As described above with respect to FIG. **6**, SDI task manager module **604** translates each step specified in a business process into a series of tasks by utilizing the services of SDI connector module **612**, which may include one or more connectors for handling the deployment of tasks specified by SDI task manager module **604** to provision one or more services related to the order request. One or more of the connectors may handle tasks that are specific to a particular service type while other connectors may handle tasks that are common across different service types. In one embodiment, SDI connector module **612** includes a set of connectors (wrapper APIs) that interface with one or more of the external modules (shown in FIG. **5**) in cloud infrastructure system **100** to provision the services and resources related to the order request. For example, a NUVIAQ connector **620** interfaces with VAB module **502** to provision Java services.

[0107] FIG. **7**B depicts a simplified block diagram showing the high-level architecture of a Nuviaq system **710** and its relationships with other cloud infrastructure components according to an embodiment of the present invention. It should be appreciated that Nuviaq system **710** depicted in FIG. **7**B may have other components than those depicted in FIG. **7**B. Further, the embodiment shown in FIG. **7**B is only one example of a cloud infrastructure system that may incorporate an embodiment of the invention. In some other embodiments, Nuviaq system **710** may have more or fewer components than shown in FIG. **7**B, may combine two or more components, or may have a different configuration or arrangement of components.

[0108] In certain embodiments, Nuviaq system **710** may be configured to provide a runtime engine for orchestrating PaaS operations. Nuviaq system **710** may provide a web service API to facilitate integration with other products and services. Nuviaq system **710** also provides support for complex workflows in system provisioning, application deployment and associated lifecycle operations, and integrates with management and monitoring solutions.

[0109] In the embodiment depicted in FIG. **7**B, Nuviaq system **710** comprises a Nuviaq proxy **712**, a Nuviaq manager **714**, and a Nuviaq database **716**. In certain embodiments, Nuviaq manager **714** provides an entry point into Nuviaq system **710**, providing secure access to PaaS operations via the web service API. Internally, it tracks system state in the database and controls job execution on the workflow engine. In a public cloud, Nuviaq manager **714** may be accessed by the Tenant Provisioning system (SDI **206**) and the Tenant Console, to drive provisioning and deployment operations respectively.

[0110] In one embodiment, Nuviaq manager **714** executes jobs asynchronously via an internal workflow engine. A job may be a sequence of actions specific to a given PaaS workflow. Actions may be performed in order, with failure in any step resulting in failure of the overall job. Many workflow actions delegate to external systems relevant to the workflow, such as the EM command line interface (cli). In one implementation, Nuviaq manager **714** application may be hosted in a 2-node WebLogic cluster with associated HTTP server (e.g., Oracle HTTP Server or OHS) instance, running inside a firewall.

[0111] In certain embodiments, Nuviaq proxy **712** is the public access point to the

[0112] Nuviaq API. In one embodiment, only Public API may be exposed here. Requests received by proxy **712** may be forwarded to Nuviaq manager **714**. In one embodiment, Nuviaq proxy **712** runs outside the firewall, whereas manager **714** runs within the firewall. In one implementation, Nuviaq proxy **712** application runs on a WebLogic cluster running outside the firewall.

[0113] In certain embodiments, Nuviaq database **716** tracks various domain entities such as, without limitation, platform instance, deployment plan, application, WebLogic domain, jobs, alerts, and the like. Primary keys may be aligned with the Service Database where appropriate.

[0114] In one embodiment, Platform Instance **718** may contain all resources required for a WebLogic service for a given tenant.

[0115] Nuviaq system **710** may rely on additional systems of cloud infrastructure system **100** to carry out the workflows used the WebLogic cloud service. These dependencies may include dependencies on SDI **206**, IDM **200**, a virus scan system, a service database, CRM instances, and the like. For example, Nuviaq system **710** may depend upon functions performed by an Assembly Deployer in SDI **206**. In one embodiment, the Assembly Deployer is a system to manage interactions with OVAB (Oracle Virtual Assembly Builder) and OVM (Oracle Virtual Machine). Capabilities of the Assembly Deployer used by Nuviaq system **710** may include, without limitation, functions for deploying an assembly, un-deploying an assembly, describing assembly deployment, scaling appliance, and the like. In one implementation, Nuviaq system **710** accesses the Assembly Deployer via a web service API.

[0116] In certain embodiments, security policies may require certain artifacts to be scanned for viruses before being deployed to an application. Cloud infrastructure system **100** may provide a virus scan system for this purpose that provides scanning as a service for multiple components of the public cloud.

[0117] In certain embodiments, a public cloud infrastructure may maintain a Service Database containing information about tenants (e.g., customers) and their service subscriptions. Nuviaq workflows may access to this data in order to properly configure a WebLogic service as a client to other services that the tenant also subscribes to.

[0118] Nuviaq system **710** may depend on IDM **200** for its security integration. In certain embodiments, Java Service instances can be associated with a CRM instance. The association allows user applications deployed to their Java Service instance to access a CRM instance though Web Service calls.

[0119] Various entities may use services provided by Nuviaq system **710**. These clients of Nuviaq system **710** may include: a Tenant Console, which is an management server (e.g., Oracle Management Server) based user interface that customers may access to manage their applications on their platform instances; several IDEs such as Oracle IDEs (JDeveloper, NetBeans, and OEPE) have been extended to offer access to application lifecycle management operations; one or more Command Line Interfaces (CLIs) that are available to access lifecycle operations on the platform instances.

[0120] Provisioning use case for Nuviaq system **710**—A Provision Platform Instance use case is realized via the Create Platform Instance operation of the Nuviaq API. In the context of cloud infrastructure system **100**, a service instance with respect to the Nuviaq system corresponds to a Nuviaq platform instance. A platform instance is assigned a unique identifier is used on all subsequent operations related to this instance. A Platform Deployment descriptor provided to the Create Platform Instance action allows for properties to be set that modify the configuration of the platform instance to meet the subscription requirements of the tenant. These properties may include for example:

[0121] Property#1: oracle.cloud.service.weblogic.size

[0122] Values: BASIC, STANDARD, ENTERPRISE

[0123] Description: Specifies the subscription type. This impacts the number of servers, database limits and quality of service settings.

[0124] Property#2: oracle.cloud.service.weblogic.trial

[0125] Values: TRUE, FALSE

[0126] Description: Indicates whether or not this is a trial subscription.

[0127] Property#**3**: oracle.cloud.service.weblogic.crm

[0128] Values: CRM Service ID

[0129] Description: Identifies a CRM service to be associated with this WebLogic service instance.

[0130] FIG. **7**C depicts an example sequence diagram illustrating steps of a provisioning process using a Nuviaq system according to an embodiment of the present invention. The sequence diagram depicted in FIG. **7**C is only an example and is not intended to be limiting.

[0131] Install/Update Application use case—The Install Application operation deploys an application to a running WebLogic Server after validating that the application archive meets the security requirements of the Public Cloud. In one embodiment, the Application Deployment descriptor provided to the Install Application action allows for properties to be set that modify the configuration of the application to meet the subscription requirements of the tenant. These properties may include for example:

[0132] Property: oracle.cloud.service.weblogic.state

[0133] Values: RUNNING, STOPPED

[0134] Description: Specifies the initial state of the application after deployment.

[0135] FIG. **7**D depicts an example sequence diagram illustrating steps of a deployment process using a Nuviaq system according to an embodiment of the present invention. The sequence diagram depicted in FIG. **7**D is only an example and is not intended to be limiting.

[0136] Referring back to FIG. **2**, in certain embodiments, TAS **204** and SDI **206** working in cooperation are responsible for provisioning resources for one or more services ordered by a customer from a set of services offered by cloud infrastructure system **100**. For example, in one embodiment, for provisioning a database service, the automated provisioning flow may be as follows for a paid subscription:

[0137] (1) Customer places an order for a paid subscription to a service via Store UI **210**.

[0138] (2) TAS **204** receives the subscription order.

[0139] (3) When services are available TAS **204** initiates provisioning by using the services of SDI **206**. TAS **204** may perform business process orchestration, which will execute the relevant business process to complete the provisioning aspect of the order. In one embodiment, TAS **204** may use a BPEL (Business Process Execution Language) Process Manager to orchestrate the steps involved in the provisioning and handle the lifecycle operations.

[0140] (4) In one embodiment, to provision a database service, SDI **206** may call PLSQL APIs in the CLOUD_UI to associate a schema for the requesting customer.

[0141] (5) After successful association of a schema to the customer, SDI signals TAS and TAS send a notification to the customer that the database service is now available for use by the customer.

[0142] (6) The customer may log into cloud infrastructure system **100** (e.g., using an URAL such as cloud.oracle.com) and activate the service.

[0143] In some embodiments, a customer may also be allowed to subscribe to a service on a trial basis. For example, such a trial order may be received via cloud UI **212** (e.g., using cloud.oracle.com).

[0144] In certain embodiments, cloud infrastructure system **100** enables underlying hardware and service instances to be shared between customers or tenants. For example, the database service may be provisioned as shown in FIG. **7**E in one embodiment. FIG. **7**E depicts multiple Exadata compute nodes **730** and **732**, each providing a database instance provisioned for the database service. For example, compute node **730** provides a database instance **734** for a database service. Each Exadata compute node may have multiple database instances.

[0145] In certain embodiments, each database instance can comprise multiple schemas and the schemas may be associated with different customers or tenants. For example, in FIG. **7**E, database instance **734** provides two schemas **736** and **738**, each with its own tables. Schema **736** may be associated with a first customer or tenant subscribing to a database service and schema **738** may be associated with a second customer or tenant subscribing to the database service. Each tenant gets a completely isolated schema. Each schema acts like a container that can manage database objects including tables, views, stored procedures, triggers, etc. for the associated tenant. Each schema may have one dedicated tablespace, with each tablespace having one data file.

[0146] In this manner, a single database instance can provide database services to multiple tenants. This not only enables sharing of underlying hardware resources but also enables sharing of service instance between tenants.

[0147] In certain embodiments, such a multi-tenancy system is facilitated by IDM **200**, which beneficially enables multiple separate customers, each having their own separate identity domains, to use hardware and software that is shared in the cloud. Consequently, there is no need for each customer to have its own dedicated hardware or software resources, and in some cases resources that are not being used by some customers at a particular moment can be used by other customers, thereby preventing those resources from being wasted. For example, as depicted in FIG. **7**E, a database instance can service multiple customers each with their respective identity domains. Although each such database service instance can be a separate abstraction or view of a single physical multi-tenant database system that is shared among the many separate identity domains, each such database service instance can have a separate and potentially different schema than each other database service instance has. Thus, the multi-tenant database system can store mappings between customer-specified database schemas and the identity domains to which those database schemas pertain. The multi-tenant database system can cause the database service instance for a particular identity domain to use the schema that is mapped to that particular identity domain.

[0148] The multi-tenancy can also be extended to other services such as the Java Service. For example, multiple customers can have a JAVA service instance placed within their respective identity domains. Each such identity domain can have a JAVA virtual machine, which can be viewed as being a virtual "slice" of hardware. In one embodiment, a job-monitoring service (e.g., Hudson) can be combined with a JAVA enterprise edition platform (e.g., Oracle WebLogic) in the cloud to enable each separate identity domain to have its own separate virtual "slice" of the JAVA enterprise edition platform. Such a job-monitoring service can, for example, monitor the execution of repeated jobs, such as building a software project or jobs run by an operating system's time-based job scheduler. Such repeated jobs can include the continuous building and/or testing of software projects. Additionally or alternatively, such repeated jobs can include the monitoring of executions of operating system-run jobs that are executed on machines that are remote from the machine on which the job-monitoring service executes.

Intelligent Data Center Selection

[0149] In certain embodiments, cloud infrastructure system **100** enables selection of appropriate data centers to host subscriptions. In the embodiment depicted in FIG. **4**, the distributed deployment of TAS module **204** includes TAS central component **400** and one or more TAS DCs **402**, **404** and **406** within data centers **1**, **2**, and **3** respectively. In one aspect, TAS central component **400** receives customer orders and performs order-related business operations such as creating subscriptions.

[0150] The efficacy of TAS central component **400** lies in its flexibility and ability to handle a wide variety of inputs. Inputs can be processed through an intelligent workflow using order-related information allowing TAS central component **400** to determine which data center will host the subscriptions. In one aspect, some types of inputs to the workflow include static inputs and dynamic inputs.

[0151] Static inputs as used herein generally refer to scriptable custom rules and algorithms provided by a system administrator. These rules use dynamic parameters and result in a list of data centers (DCs). Rules can be assigned a priority and can be executed in priority order. Additionally, the algorithms to be applied to a list of DCs work on a list to output a DC. An algorithm can be applied at two levels—at each rule level, and at a global level.

[0152] Dynamic inputs as used herein generally refer to customer order parameters, DC availability metrics, and DC capacity metrics. Customer order parameters can include customer name and other attributes, service type being ordered, and other service attributes, etc. DC availability metrics and levels can include the available capacity for each service type in a DC, and can be used to filter a list of DCs. DC capacity metrics and levels cans include the total capacity for each service type in a DC, and can be used to filter a list of DCs.

[0153] In various embodiments, the responsibilities of TAS central component **400** may also include maintaining metrics needed by cloud infrastructure system **100** to make appropriate selections between data centers. Based on metrics about each data center, TAS central component **400** can determine which data center will host the subscriptions. The responsibilities of TAS central component **400** may also include maintaining additional requirements needed by cloud infrastructure system **100** to make appropriate selections between data centers. Based on these additional requirements, such as local laws or regulations, TAS central component **400** can determine which data center will host the subscriptions.

[0154] FIG. **8** illustrates a high level overview of the various interactions involved to provide intelligent data center selection in accordance with an embodiment of the present invention. According to this embodiment, the distributed TAS system shown in FIG. **4**, TAS central component **400** includes a TAS DC Selection System that initially receives an order made by a customer via cloud UI **212**, store UI **210**, via order management system **214**, or via order database **224**. In this example, TAS DC Selection System receives order information from an order submission script. The customer may be ordering one or more subscriptions. The order information can include information identifying the customer, the type of services that the customer wishes to subscribe to, user information, and any other information relevant to the customer and subscription. Based on the order information, the TAS DC Selection System can identify one or more regions of the world (such as Americas, EMEA, or Asia Pacific) from which the order originates or to which region a subscription may be designated.

[0155] The TAS DC Selection System can then select a Chosen DC for the order using a preprocessing layer, a rules engine, an algorithm-based selection layer, and TAS Tables to determine the Chosen DC (e.g., which particular TAS DCs (for e.g., **402**, **404** or **406**) will be employed for provisioning the order). In various embodiments, the TAS DC Selection System retrieves objectives, rules, algorithms, and other data from the TAS Tables. An administrator can define the objectives, create rules, and specify selection algorithms using an Admin module of the TAS DC Selection System.

[0156] In some embodiments, the TAS DC Selection System retrieves data center information aggregated from various data centers. A DC Info Poller can connect to each of the data centers (e.g., to SDI **206** of each data center) to obtain information about the operation of the data center, services offered, subscriptions hosted, resource allocations, resource availability, as well as other information related the data center. In some embodiments, the TAS DC Selection System can retrieve service to data center strategy mappings from the TAS Tables.

[0157] Based on information retrieved from the TAS Tables and the order information received from the Order submission script, the TAS DC Selection System utilizes the preprocessing layer, the rules engine, and the algorithm-based selection layer to select the Chosen DC. The TAS DC Selection System can then send an order request to the Chosen DC in which to provision services for the order request.

[0158] In some embodiments, the TAS DC Selection System receives the order information from an Order Poller. For services that are provisioned but have a custom DCS mechanism, the Order Poller can identify orders where a data center is not set. The Order Poller can call an appropriate API and get the Chosen DC from the TAS DC Selection System.

[0159] FIGS. **9**A-**9**B are a flowchart of method **900** for selecting a data center to host a cloud subscription according to various embodiments. The processing depicted in FIGS. **9**A-**9**B may be implemented in software (e.g., code, instructions, program) executed by one or more processors, hardware, or combinations thereof. The software may be stored in memory (e.g., on a memory device, on a non-transitory computer-readable storage medium). The particular series of processing steps depicted in FIGS. **9**A-**9**B is not intended to be limiting. Other sequences of steps may also be performed according to alternative embodiments. For example, alternative embodiments of the present invention may perform the steps outlined above in a different order. Moreover, the individual steps illustrated in FIGS. **9**A-**9**B may include multiple sub-steps that may be performed in various sequences as appropriate to the individual step. Furthermore, additional steps may be added or removed depending on the particular applications. One of ordinary skill in the art would recognize many variations, modifications, and alternatives. FIG. **9**A begins in step **902**.

[0160] In step **902**, order information is received. For example, the TAS DC Selection System of FIG. **8** can receive an order, order item, and DC region. In step **904**, a list of selectable DCs is obtained, for example, within DCRegion & available pods>0 for given service, sub type; then stored in a GDCList (global data center list).

[0161] In step **906**, a determination is made as to the size of the GDCList. If the size of the GDCList is 0, in step **908**, a choice among zero capacity pods is made. The data centers having zero capacity pods are added to the GDCList. Additionally, if GDCList is still empty, the preferred DC for the given region is added to the GDCList. Processing then returns to step **906**. If the size of the GDCList is 1, in step **910**, the chose DC is returned allowing TAS DC Selection System to request that the chosen DC host the subscription.

[0162] If the size of the GDCList in step **906** is greater than 1, in step **912**, a set of matched rules is obtained. As indicated above, TAS DC Selection System utilizes a rules engine to process the GDCList according to a set of rules. A rule as used in embodiments of the present invention includes a condition clause and one or more action clauses. The condition clause specifies one or more conditions that must be satisfied in order for the rule to match. An action clause defines one or more actions to be taken for a matching rule. The actions are used to filter out DCs from the GDCList. In one embodiment, an action specifically excludes a DC from the GDCList. In one embodiment, an action applies a predetermined algorithm to the GDCList to filter the GDCList.

[0163] Each rule in the set of rules can be identified (matched) using properties of the order, customer, DC region, or the like to satisfy the condition clauses. In step **914**, a determination is made as to the size of the set of matched rules. If there are no matched rules, in step **916**, a global algorithm is used on the GDCList to determine a chosen DC (which is returned in step **910**).

[0164] If there are one or more rules in the set, a determination is made as to the ordering of processing of the rules in the set. For example, in step **918**, the highest priority rule yet to be considered is then evaluated. In step **920**, a determination is made whether the rule includes an action clause. If no action clause is present, processing continues in FIG. **9**B via reference "B."

[0165] If an action clause is present, the action clause is applied in step **922**. TAS DC Selection System can intersect the GDCList with a list of DCs created by applying action clause. In step **924**, a determination is made as to the number of DCs after the intersection. If there are 0 DCs, a determination is made in step **926** whether there are any more matching rules. If so, processing continues in step **918**. If not, processing continues in step **916** where the global selection algorithm is applied for choosing the appropriate DC.

[0166] If there is one (1) DC left in the determination is step **924**, in step **928**, the remaining DC is returned as the chosen DC. If there is more than one DC left in the list, processing continues in FIG. **9**B via reference "A."

[0167] Referring to FIG. **9**B, a determination is made whether an action algorithm exists in step **930**. A rule can specify an action algorithm that further filters a list of DCs. If no action algorithm is specified, in step **932**, the global algorithm is then applied to determine the appropriate DC. If an action algorithm does exist, in step **934**, the action algorithm is applied on input list. In one embodiment, if there is a preferred algorithm, it should choose one among input list definitively (i.e., if preferred DC is not among input list, it picks random DC from the input list). Other algorithms should definitively pick one from the input list. In step **936**, the chosen DC is returned from the list.

[0168] In some embodiments, an administrator can add data center selection rules at will. The following are a few examples of commands that add data center selection rules and show what the rules can look like.

[0169] Rule #1—Always select a particular data center for a particular customer

[0170] add_data_center_selection_rule—name us3_for_bmwcustomer—rule_clause

[0171] "orders.customer name='bmw'"—action "id='US003'"—status ACTIVE—priority 5

[0172] Rule #2—Always select least cost data center for a particular customer (overrides global algorithm)

```
add_data_center_selection_rule --name leastcost_dc_for_bmwcustomer
--rule_clause "orders.customer_name='bmw'" --action_algo "LEAST_
COST"--status ACTIVE --priority 5
```

[0173] Rule #3—Always select least cost data center for TRIAL orders

```
add_data_center_selection_rule --name leastcost_dc_fortrial
--rule_clause " order_items.subscription_type='TRIAL'" --action_algo
"LEAST_COST"
```

[0174] Rule #4—Not select a particular data center

```
add_data_center_selection_rule --name donot_select_us1_for_trials --rule_
clause " order_items.subscription_type ='TRIAL'" --action "id<>'US001'"
```

[0175] Rule #5—Action containing multiple properties

```
add_data_center_selection_rule --name select_below35USdc_for_java
--rule_clause "service_type like 'JAVA'" --action "id like 'US%' and
devops_weight < 35" --status ACTIVE --priority 5
```

[0176] Rule #6—Specifying action algorithm for the list of DCs filtered by action

```
add_data_center_selection_rule --name lru_algo_among_sub200dcs_for_
java --rule_clause "service_type_like 'JAVA'" --action "devops_weight
< 200" --action_algo LFU --status ACTIVE --priority 5
```

[0177] FIG. **10** is a flowchart of method **1000** for selecting a data center to host a cloud subscription according to various embodiments. The processing depicted in FIG. **10** may be implemented in software (e.g., code, instructions, program) executed by one or more processors, hardware, or combinations thereof. The software may be stored in memory (e.g., on a memory device, on a non-transitory computer-readable storage medium). The particular series of processing steps depicted in FIG. **10** is not intended to be limiting. Other sequences of steps may also be performed according to alternative embodiments. For example, alternative embodiments of the present invention may perform the steps outlined above in a different order. Moreover, the individual steps illustrated in FIG. **10** may include multiple sub-steps that may be performed in various sequences as appropriate to the individual step. Furthermore, additional steps may be added or removed depending on the particular applications. One of ordinary skill in the art would recognize many variations, modifications, and alternatives. FIG. **10** begins in step **1002**.

[0178] In step **1002**, preprocessing is performed. In various embodiments, a preprocessing layer of the TAS DC Selection System of FIG. **8** provides flexibility to choose among only 'valid' DCs. Based on a requested service/subscription-type from the order information, the TAS DC Selection System can obtain a list of all the available data centers that can serve any ordered services or service-subscription types. In one aspect, the list of all the available data centers is filtered down using a set of active rules and algorithms to determine at least one appropriate data center.

[0179] In step **1004**, rules processing is performed. In various aspects, the TAS DC Selection System parses or processes a set of active rules to determine a set of matched rules. A rule may be assigned a priority and the TAS DC Selection System may apply matched rules based on their assigned priority. Each rule can define criteria that need to be satisfied in order for the rule to be considered a match. Criteria that may be used to match a rule can include criteria related to the order information, criteria related to each data center, criteria related to cloud infrastructure system **100**, or the like.

[0180] A rule can also define, reference, or otherwise indicate an action, algorithm, logic, or combinations thereof. In one aspect, a rule action defines, refers to, or otherwise identifies a set of one or more operations, instructions, logic, steps, or the like to be performed or applied to the list of all the available data centers determine in the preprocessing of step **1002** to filter out data centers and otherwise determine a new list of available data centers. A rule algorithm defines one or more algorithms that based on a set of inputs to be performed or implemented to filter out data centers and otherwise determine a new list of available data centers. In some embodiments, a rule defines only a rule action. A default algorithm or universal selection mechanism is applied to the new set of data centers resulting from application of the rule action.

[0181] In step **1006**, the set of matched rules are applied. For example, when a rule action is defined, TAS DC Selection System computes an appropriate data center or list of appropriate data centers from the action. When a rule algorithm is defined, TAS DC Selection System selects that specific algorithm for computing an appropriate data center or list of appropriate data centers. If no data centers are available for selection or if no data centers are obtained as a result of applying the set of matched rules, TAS DC Selection System can select a default data center for the service.

[0182] In various embodiments, if a matched rule when applied resulted in a list of data centers, a default algorithm for determining an appropriate data center is run against that list of data centers. If an algorithm is defined by or otherwise chosen by a matched rule when applied, that algorithm is run instead of any default selection algorithms. In some embodiments, a rule specifies or otherwise indicates only a rule algorithm. Application of the rule causes TAS DC Selection System to apply the provided rule algorithm to obtain a new set of data centers and select from among the data centers in the new set of data centers. In further embodiments, a rule may define both a rule action and a rule algorithm. Application of the rule causes TAS DC Selection System to apply the provided rule algorithm to a new set of data centers resulting from application of the rule action.

[0183] In step **1008**, a data center is determined. In various embodiments, TAS DC Selection System begins with a list of data centers that may be appropriate to host the subscription and filters down the list to select an appropriate data center to identify one data center or at least a smaller subset of data centers in the list. FIG. **10** ends in step **1008**.

CONCLUSION

[0184] FIG. **11** depicts a simplified diagram of a distributed system **1100** for implementing one of the embodiments. In the illustrated embodiment, distributed system **1100** includes one or more client computing devices **1102**, **1104**, **1106**, and **1108**, which are configured to execute and operate a client application such as a web browser, proprietary client (e.g., Oracle Forms), or the like over one or more network(s) **1110**. Server **1112** may be communicatively coupled with remote client computing devices **1102**, **1104**, **1106**, and **1108** via network **1110**.

[0185] In various embodiments, server **1112** may be adapted to run one or more services or software applications provided by one or more of the components of the system. In some embodiments, these services may be offered as web-based or cloud services or under a Software as a Service (SaaS) model to the users of client computing devices **1102**, **1104**, **1106**, and/or **1108**. Users operating client computing devices **1102**, **1104**, **1106**, and/or **1108** may in turn utilize one or more client applications to interact with server **1112** to utilize the services provided by these components.

[0186] In the configuration depicted in the figure, the software components **1118**, **1120** and **1122** of system **1100** are shown as being implemented on server **1112**. In other embodiments, one or more of the components of system **1100** and/or the services provided by these components may also be implemented by one or more of the client computing devices **1102**, **1104**, **1106**, and/or **1108**. Users operating the client computing devices may then utilize one or more client applications to use the services provided by these components. These components may be implemented in hardware, firmware, software, or combinations thereof. It should be appreciated that various different system configurations are possible, which may be different from distributed system **1100**. The embodiment shown in the figure is thus one example of a distributed system for implementing an embodiment system and is not intended to be limiting.

[0187] Client computing devices **1102**, **1104**, **1106**, and/or **1108** may be portable handheld devices (e.g., an iPhone®, cellular telephone, an iPad®, computing tablet, a personal digital assistant (PDA)) or wearable devices (e.g., a Google Glass® head mounted display), running software such as Microsoft Windows Mobile®, and/or a variety of mobile operating systems such as iOS, Windows Phone, Android, BlackBerry 10, Palm OS, and the like, and being Internet, e-mail, short message service (SMS), Blackberry®, or other communication protocol enabled. The client computing devices can be general purpose personal computers including, by way of example, personal computers and/or laptop computers running various versions of Microsoft Windows®, Apple Macintosh®, and/or Linux operating systems. The client computing devices can be workstation computers running any of a variety of commercially-available UNIX® or UNIX-like operating systems, including without limitation the variety of GNU/Linux operating systems, such as for example, Google Chrome OS. Alternatively, or in addition, client computing devices **1102**, **1104**, **1106**, and **1108** may be any other electronic device, such as a thin-client computer, an Internet-enabled gaming system (e.g., a Microsoft Xbox gaming console with or without a Kinect® gesture input device), and/or a personal messaging device, capable of communicating over network(s) **1110**.

[0188] Although exemplary distributed system **1100** is shown with four client computing devices, any number of client computing devices may be supported. Other devices, such as devices with sensors, etc., may interact with server **1112**.

[0189] Network(s) **1110** in distributed system **1100** may be any type of network familiar to those skilled in the art that can support data communications using any of a variety of commercially-available protocols, including without limitation TCP/IP (transmission control protocol/Internet protocol), SNA (systems network architecture), IPX (Internet packet exchange), AppleTalk, and the like. Merely by way of example, network(s) **1110** can be a local area network (LAN), such as one based on Ethernet, Token-Ring and/or the like. Network(s) **1110** can be a wide-area network and the Internet. It can include a virtual network, including without limitation a virtual private network (VPN), an intranet, an extranet, a public switched telephone network (PSTN), an infra-red network, a wireless network (e.g., a network operating under any of the Institute of Electrical and Electronics (IEEE) 802.11 suite of protocols, Bluetooth®, and/or any other wireless protocol); and/or any combination of these and/or other networks.

[0190] Server **1112** may be composed of one or more general purpose computers, specialized server computers (including, by way of example, PC (personal computer) servers, UNIX® servers, mid-range servers, mainframe computers, rack-mounted servers, etc.), server farms, server clusters, or any other appropriate arrangement and/or combination. In various embodiments, server **1112** may be adapted to run one or more services or software applications described in the foregoing disclosure. For example, server **1112** may correspond to a server for performing processing described above according to an embodiment of the present disclosure.

[0191] Server **1112** may run an operating system including any of those discussed above, as well as any commercially available server operating system. Server **1112** may also run any of a variety of additional server applications and/or mid-tier applications, including HTTP (hypertext transport protocol) servers, FTP (file transfer protocol) servers, CGI (common gateway interface) servers, JAVA® servers, database servers, and the like. Exemplary database servers include without limitation those commercially available from Oracle, Microsoft, Sybase, IBM (International Business Machines), and the like.

[0192] In some implementations, server **1112** may include one or more applications to analyze and consolidate data feeds and/or event updates received from users of client computing devices **1102**, **1104**, **1106**, and **1108**. As an example, data feeds and/or event updates may include, but are not limited to, Twitter® feeds, Facebook® updates or real-time updates received from one or more third party information sources and continuous data streams, which may include real-time events related to sensor data applications, financial tickers, network performance measuring tools (e.g., network monitoring and traffic management applications), click-stream analysis tools, automobile traffic monitoring, and the like. Server **1112** may also include one or more applications to display the data feeds and/or real-time events via one or more display devices of client computing devices **1102**, **1104**, **1106**, and **1108**.

[0193] Distributed system **1100** may also include one or more databases **1114** and **1116**. Databases **1114** and **1116** may reside in a variety of locations. By way of example, one or more of databases **1114** and **1116** may reside on a non-transitory storage medium local to (and/or resident in) server

1112. Alternatively, databases **1114** and **1116** may be remote from server **1112** and in communication with server **1112** via a network-based or dedicated connection. In one set of embodiments, databases **1114** and **1116** may reside in a storage-area network (SAN). Similarly, any necessary files for performing the functions attributed to server **1112** may be stored locally on server **1112** and/or remotely, as appropriate. In one set of embodiments, databases **1114** and **1116** may include relational databases, such as databases provided by Oracle, that are adapted to store, update, and retrieve data in response to SQL-formatted commands.

[0194] FIG. **12** illustrates an exemplary computer system **1200**, in which various embodiments of the present invention may be implemented. The system **1200** may be used to implement any of the computer systems described above. As shown in the figure, computer system **1200** includes a processing unit **1204** that communicates with a number of peripheral subsystems via a bus subsystem **1202**. These peripheral subsystems may include a processing acceleration unit **1206**, an I/O subsystem **1208**, a storage subsystem **1218**, and a communications subsystem **1224**. Storage subsystem **1218** includes tangible computer-readable storage media **1222** and a system memory **1210**.

[0195] Bus subsystem **1202** provides a mechanism for letting the various components and subsystems of computer system **1200** communicate with each other as intended. Although bus subsystem **1202** is shown schematically as a single bus, alternative embodiments of the bus subsystem may utilize multiple buses. Bus subsystem **1202** may be any of several types of bus structures including a memory bus or memory controller, a peripheral bus, and a local bus using any of a variety of bus architectures. For example, such architectures may include an Industry Standard Architecture (ISA) bus, Micro Channel Architecture (MCA) bus, Enhanced ISA (EISA) bus, Video Electronics Standards Association (VESA) local bus, and Peripheral Component Interconnect (PCI) bus, which can be implemented as a Mezzanine bus manufactured to the IEEE P1386.1 standard.

[0196] Processing unit **1204**, which can be implemented as one or more integrated circuits (e.g., a conventional microprocessor or microcontroller), controls the operation of computer system **1200**. One or more processors may be included in processing unit **1204**. These processors may include single core or multicore processors. In certain embodiments, processing unit **1204** may be implemented as one or more independent processing units **1232** and/or **1234** with single or multicore processors included in each processing unit. In other embodiments, processing unit **1204** may also be implemented as a quad-core processing unit formed by integrating two dual-core processors into a single chip.

[0197] In various embodiments, processing unit **1204** can execute a variety of programs in response to program code and can maintain multiple concurrently executing programs or processes. At any given time, some or all of the program code to be executed can be resident in processor(s) **1204** and/or in storage subsystem **1218**. Through suitable programming, processor(s) **1204** can provide various functionalities described above. Computer system **1200** may additionally include a processing acceleration unit **1206**, which can include a digital signal processor (DSP), a special-purpose processor, and/or the like.

[0198] I/O subsystem **1208** may include user interface input devices and user interface output devices. User interface input devices may include a keyboard, pointing devices such as a mouse or trackball, a touchpad or touch screen incorporated into a display, a scroll wheel, a click wheel, a dial, a button, a switch, a keypad, audio input devices with voice command recognition systems, microphones, and other types of input devices. User interface input devices may include, for example, motion sensing and/or gesture recognition devices such as the Microsoft Kinect® motion sensor that enables users to control and interact with an input device, such as the Microsoft Xbox® 360 game controller, through a natural user interface using gestures and spoken commands. User interface input devices may also include eye gesture recognition devices such as the Google Glass® blink detector that detects eye activity (e.g., 'blinking' while taking pictures and/or making a menu selection) from users and transforms the eye gestures as input into an input device (e.g., Google Glass®). Additionally, user interface input devices may include voice recognition sensing devices that enable users to interact with voice recognition systems (e.g., Siri® navigator), through voice commands.

[0199] User interface input devices may also include, without limitation, three dimensional (3D) mice, joysticks or pointing sticks, gamepads and graphic tablets, and audio/visual devices such as speakers, digital cameras, digital camcorders, portable media players, webcams, image scanners, fingerprint scanners, barcode reader 3D scanners, 3D printers, laser rangefinders, and eye gaze tracking devices. Additionally, user interface input devices may include, for example, medical imaging input devices such as computed tomography, magnetic resonance imaging, position emission tomography, medical ultrasonography devices. User interface input devices may also include, for example, audio input devices such as MIDI keyboards, digital musical instruments and the like.

[0200] User interface output devices may include a display subsystem, indicator lights, or non-visual displays such as audio output devices, etc. The display subsystem may be a cathode ray tube (CRT), a flat-panel device, such as that using a liquid crystal display (LCD) or plasma display, a projection device, a touch screen, and the like. In general, use of the term "output device" is intended to include all possible types of devices and mechanisms for outputting information from computer system **1200** to a user or other computer. For example, user interface output devices may include, without limitation, a variety of display devices that visually convey text, graphics and audio/video information such as monitors, printers, speakers, headphones, automotive navigation systems, plotters, voice output devices, and modems.

[0201] Computer system **1200** may comprise a storage subsystem **1218** that comprises software elements, shown as being currently located within a system memory **1210**. System memory **1210** may store program instructions that are loadable and executable on processing unit **1204**, as well as data generated during the execution of these programs.

[0202] Depending on the configuration and type of computer system **1200**, system memory **1210** may be volatile (such as random access memory (RAM)) and/or non-volatile (such as read-only memory (ROM), flash memory, etc.) The RAM typically contains data and/or program modules that are immediately accessible to and/or presently being operated and executed by processing unit **1204**. In some implementations, system memory **1210** may include multiple different types of memory, such as static random access memory (SRAM) or dynamic random access memory (DRAM). In some implementations, a basic input/output system (BIOS), containing the basic routines that help to transfer information between elements within computer system **1200**, such as during start-up, may typically be stored in the ROM. By way of example, and not limitation, system memory **1210** also illustrates application programs **1212**, which may include client applications, Web browsers, mid-tier applications, relational database management systems (RDBMS), etc., program data **1214**, and an operating system **1216**. By way of example, operating system **1216** may include various versions of Microsoft Windows®, Apple Macintosh®, and/or Linux operating systems, a variety of commercially-available UNIX® or UNIX-like operating systems (including without limitation the variety of GNU/Linux operating systems, the Google Chrome® OS, and the like) and/or mobile operating systems such as iOS, Windows® Phone, Android® OS, BlackBerry® 12 OS, and Palm® OS operating systems.

[0203] Storage subsystem **1218** may also provide a tangible computer-readable storage medium for storing the basic programming and data constructs that provide the functionality of some embodiments. Software (programs, code modules, instructions) that when executed by a processor provide the functionality described above may be stored in storage subsystem **1218**. These software modules or instructions may be executed by processing unit **1204**. Storage subsystem **1218** may also provide a repository for storing data used in accordance with the present invention.

[0204] Storage subsystem **1200** may also include a computer-readable storage media reader **1220** that can further be connected to computer-readable storage media **1222**. Together and, optionally, in combination with system memory **1210**, computer-readable storage media **1222** may comprehensively represent remote, local, fixed, and/or removable storage devices plus storage media for temporarily and/or more permanently containing, storing, transmitting, and retrieving computer-readable information.

[0205] Computer-readable storage media **1222** containing code, or portions of code, can also include any appropriate media known or used in the art, including storage media and communication media, such as but not limited to, volatile and non-volatile, removable and non-removable media implemented in any method or technology for storage and/or transmission of information. This can include tangible computer-readable storage media such as RAM, ROM, electronically erasable programmable ROM (EEPROM), flash memory or other memory technology, CD-ROM, digital versatile disk (DVD), or other optical storage, magnetic cassettes, magnetic tape, magnetic disk storage or other magnetic storage devices, or other tangible computer readable media. This can also include nontangible computer-readable media, such as data signals, data transmissions, or any other medium which can be used to transmit the desired information and which can be accessed by computing system **1200**.

[0206] By way of example, computer-readable storage media **1222** may include a hard disk drive that reads from or writes to non-removable, nonvolatile magnetic media, a magnetic disk drive that reads from or writes to a removable, nonvolatile magnetic disk, and an optical disk drive that reads from or writes to a removable, nonvolatile optical disk such as a CD ROM, DVD, and Blu-Ray® disk, or other optical media. Computer-readable storage media **1222** may include, but is not limited to, Zip® drives, flash memory cards, universal serial bus (USB) flash drives, secure digital (SD) cards, DVD disks, digital video tape, and the like. Computer-readable storage media **1222** may also include, solid-state drives (SSD) based on non-volatile memory such as flash-memory based SSDs, enterprise flash drives, solid state ROM, and the like, SSDs based on volatile memory such as solid state RAM, dynamic RAM, static RAM, DRAM-based SSDs, magnetoresistive RAM (MRAM) SSDs, and hybrid SSDs that use a combination of DRAM and flash memory based SSDs. The disk drives and their associated computer-readable media may provide non-volatile storage of computer-readable instructions, data structures, program modules, and other data for computer system **1200**.

[0207] Communications subsystem **1224** provides an interface to other computer systems and networks. Communications subsystem **1224** serves as an interface for receiving data from and transmitting data to other systems from computer system **1200**. For example, communications subsystem **1224** may enable computer system **1200** to connect to one or more devices via the Internet. In some embodiments communications subsystem **1224** can include radio frequency (RF) transceiver components for accessing wireless voice and/or data networks (e.g., using cellular telephone technology, advanced data network technology, such as 3G, 4G or EDGE (enhanced data rates for global evolution), WiFi (IEEE 802.11 family standards, or other mobile communication technologies, or any combination thereof), global positioning system (GPS) receiver components, and/or other components. In some embodiments communications subsystem **1224** can provide wired network connectivity (e.g., Ethernet) in addition to or instead of a wireless interface.

[0208] In some embodiments, communications subsystem **1224** may also receive input communication in the form of structured and/or unstructured data feeds **1226**, event streams **1228**, event updates **1230**, and the like on behalf of one or more users who may use computer system **1200**.

[0209] By way of example, communications subsystem **1224** may be configured to receive data feeds **1226** in real-time from users of social networks and/or other communication services such as Twitter® feeds, Facebook® updates, web feeds such as Rich Site Summary (RSS) feeds, and/or real-time updates from one or more third party information sources.

[0210] Additionally, communications subsystem **1224** may also be configured to receive data in the form of continuous data streams, which may include event streams **1228** of real-time events and/or event updates **1230**, that may be continuous or unbounded in nature with no explicit end. Examples of applications that generate continuous data may include, for example, sensor data applications, financial tickers, network performance measuring tools (e.g. network monitoring and traffic management applications), clickstream analysis tools, automobile traffic monitoring, and the like.

[0211] Communications subsystem **1224** may also be configured to output the structured and/or unstructured data feeds **1226**, event streams **1228**, event updates **1230**, and the like to one or more databases that may be in communication with one or more streaming data source computers coupled to computer system **1200**.

[0212] Computer system **1200** can be one of various types, including a handheld portable device (e.g., an iPhone® cellular phone, an iPad® computing tablet, a PDA), a wearable device (e.g., a Google Glass® head mounted display), a PC, a workstation, a mainframe, a kiosk, a server rack, or any other data processing system.

[0213] Due to the ever-changing nature of computers and networks, the description of computer system **1200** depicted in the figure is intended only as a specific example. Many other configurations having more or fewer components than the system depicted in the figure are possible. For example, customized hardware might also be used and/or particular elements might be implemented in hardware, firmware, software (including applets), or a combination. Further, connection to other computing devices, such as network input/output devices, may be employed. Based on the disclosure and teachings provided herein, a person of ordinary skill in the art will appreciate other ways and/or methods to implement the various embodiments.

[0214] In the foregoing specification, aspects of the invention are described with reference to specific embodiments thereof, but those skilled in the art will recognize that the invention is not limited thereto. Various features and aspects of the above-described invention may be used individually or jointly. Further, embodiments can be utilized in any number of environments and applications beyond those described herein without departing from the broader spirit and scope of the specification. The specification and drawings are, accordingly, to be regarded as illustrative rather than restrictive.

That which is claimed is:

1. A method performed by a computer having a processor and a memory storing a set of data center selection rules, the method comprising:
- receiving, at the computer, order information from an order management server of a cloud computing system, the order information specifying a customer and a product to be instantiated at at least one data center of the cloud computing system;
- determining, by the processor, a set of one or more data center selection rules using the order information; and
- evaluating, by the processor, the set of data center selection rules using a first plurality of data centers associated with the cloud computing system, wherein evaluating the set of data center selection rules comprises:
  - determining that a data center selection rule includes an action clause,
  - filtering the first plurality of data centers by applying the action clause to generate a second plurality of data centers,
  - determining that the data center selection rule includes an algorithm clause, and
  - selecting a data center from the second plurality of data centers by applying a selection algorithm identified by the data center selection rule.

2. The method of claim **1** wherein determining, by the processor, the set of one or more data center selection rules using the order information comprising:
- matching at least one data center selection rule having a condition clause specifying one or more properties of the customer.

3. The method of claim **1** wherein determining, by the processor, the set of one or more data center selection rules using the order information comprising:
- matching at least one data center selection rule having a condition clause specifying one or more properties of the product.

4. The method of claim **1** wherein filtering the first plurality of data centers by applying the action clause to generate the second plurality of data centers comprises selecting the second plurality of data centers using one or more properties of the second plurality of data centers defined in the rule and one or more metrics associated with the second plurality of data centers.

5. The method of claim **1** wherein filtering the first plurality of data centers by applying the action clause to generate the second plurality of data centers comprises excluding at least one data center form the first plurality of data centers using one or more properties of the at least one data center defined in the rule and one or more metrics associated with the at least one data center.

6. The method of claim **1** wherein filtering the first plurality of data centers by applying the action clause to generate the second plurality of data centers comprises selecting at least one data center based on a type of service associated with the product.

7. The method of claim **1** wherein filtering the first plurality of data centers by applying the action clause to generate the second plurality of data centers comprises selecting at least one data center based on a cost of instantiating the product at the at least one data center.

8. The method of claim **1** wherein filtering the first plurality of data centers by applying the action clause to generate the second plurality of data centers comprises selecting at least one data center based on a predefined mapping between the customer or the product and the at least one data center.

9. A non-transitory computer-readable medium storing a computer program executable by a computer having a processor and a memory storing a set of data center selection rules, the non-transitory computer-readable medium comprising:
- code for receiving order information from an order management server of a cloud computing system, the order information specifying a customer and a product to be instantiated at at least one data center of the cloud computing system;
- code for determining a set of one or more data center selection rules using the order information; and
- code for evaluating the set of data center selection rules using a first plurality of data centers associated with the cloud computing system, wherein the code for evaluating the set of data center selection rules comprises:
  - code for determining that a data center selection rule includes an action clause,
  - code for filtering the first plurality of data centers by applying the action clause to generate a second plurality of data centers,
  - code for determining that the data center selection rule includes an algorithm clause, and
  - code for selecting a data center from the second plurality of data centers by applying a selection algorithm identified by the data center selection rule.

10. A system comprising:
- a hardware processor; and
- a memory storing a set of instructions which when executed by the processor cause the processor to:
  - receive order information from an order management server of a cloud computing system, the order information specifying a customer and a product to be instantiated at at least one data center of the cloud computing system;
  - determine a set of one or more data center selection rules using the order information; and
  - evaluate the set of data center selection rules using a first plurality of data centers associated with the cloud

computing system, wherein to evaluate the set of data center selection rules the processor is caused to:

determine that a data center selection rule includes an action clause,

filter the first plurality of data centers by applying the action clause to generate a second plurality of data centers,

determine that the data center selection rule includes an algorithm clause, and

select a data center from the second plurality of data centers by applying a selection algorithm identified by the data center selection rule.

* * * * *