Title: RESOURCE PROVISIONING IN DISTRIBUTED COMPUTING ENVIRONMENTS

Abstract: A computer implemented method to provide a resource to a virtualised software installation in a distributed computing environment, the method comprising: receiving a functional requirement for the software installation; determining an extent of the resource required to implement the functional requirement; identifying one or more resource providers in the distributed computing environment having availability of the required resource to the determined extent; and communicating an indication of the identified resource providers to the software installation in order that the resource can be provided to the software installation to the determined extent to provide the functional requirement.

**Figure 2**
Resource Provisioning in Distributed Computing Environments

The present invention relates to the provision of resources. In particular it relates to the provision of resources in distributed computing environments.

Distributed computing environments are environments in which computer systems, services and/or resources (whether hardware, software or a combination) are distributed physically and/or virtually with a dependence on communications networks for interoperability. Further, there is increasing deployment of software installations such as applications or entire virtualised computer systems to service based environments, such as Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and cloud computing environments.

Software installations such as virtualised computer systems, cloud computing deployments and the like, implement functional requirements as part of the installation. For example functional requirements can include: security functionality such as anti-malware, encryption, firewall or intrusion detection; communications functionality such as network communication, encrypted or otherwise secure communication; operating system services and functions; data storage facilities including disk, memory, database and the like; middleware services such as message handling, transaction handling and the like; and other functional requirements. Functional requirements themselves employ computing resources such as storage, processors, security functions, communication resources such as networking and the like that are increasingly provided by "supplier" or "provider" entities (whether third-party or not) either as actual resources or virtualised resources. Thus in implementing or providing a functional requirement a software installation consumes requisite computing resources from resource providers. The consumption of such resources is increasingly dynamic, on-demand and/or dynamically scalable. Resource providers and consumers in such environments can be unrelated, anonymous, untrusted and/or unknown to each other.

A challenge arises for a resource consumer such as a software installation seeking to implement a functional requirement in that it is necessary to identify resource providers able to accommodate the resource demands of the functional requirement. On the other hand a challenge arises for resource providers such as server or cloud service computer systems in that there is no distinction between the resource requirements of seemingly identical functional requirements where the functional requirements are implemented differently such that one uses more resource than the other. Accordingly resource providers apply a one-size-fits-all approach to resource provision and resource consumers can find that they are
over or under provisioned. The challenges are particularly acute where resource consumers seek to change resource consumption characteristics of their functional requirements or change functional requirements altogether, in which case the resource consumption can deviate from a historical norm.

Accordingly it would be beneficial to provide resource provisioning for software installations without the aforementioned disadvantages.

The present invention accordingly provides, in a first aspect, a computer implemented method to provide a resource to a virtualised software installation in a distributed computing environment, the method comprising: receiving a functional requirement for the software installation; determining an extent of the resource required to implement the functional requirement; identifying one or more resource providers in the distributed computing environment having availability of the required resource to the determined extent; and communicating an indication of the identified resource providers to the software installation in order that the resource can be provided to the software installation to the determined extent to provide the functional requirement.

Preferably the functional requirement has associated one or more characteristics and determining an extent of a resource required includes: accessing a directory of functional requirement characteristics, the directory defining, for each characteristic, an extent of the resource required to provide a functional requirement having the characteristic.

Preferably determining an extent of a resource required includes accessing a directory of functional requirements, the directory defining, for each functional requirement, an extent of the resource required to implement the function.

Preferably the functional requirement includes a function of a security policy.

Preferably the functional requirement is an entry in a deployment descriptor for one or more software components for execution for the software installation.

Preferably identifying resource providers includes the steps of: polling one or more resource providers for availability of the resource at each provider; receiving resource availability information from the one or more resource providers; and determining one or more resource providers suitable for providing the resource to the required extent.

Preferably identifying resource providers includes sending a request to each of one or more resource providers including an indication of the required resource and the determined extent; and receiving an indication from each of one or more providers that the provider has availability of the required resource to the determined extent.
The present invention accordingly provides, in a second aspect, a computer system to provide a resource to a virtualised software installation in a distributed computing environment, the system comprising: a receiver including logic adapted to receive a functional requirement for the software installation; a resource assessor including logic adapted to determine an extent of the resource required to implement the functional requirement; a resource provider identifier including logic adapted to identify one or more resource providers in the distributed computing environment having availability of the required resource to the determined extent; and an interface via which an indication of the identified resource providers is communicated to the software installation in order that the resource can be provided to the software installation to the determined extent to provide the functional requirement.

The present invention accordingly provides, in a third aspect, a computer program element comprising computer program code to, when loaded into a computer system and executed thereon, cause the computer to perform the steps of a method as described above.

A preferred embodiment of the present invention will now be described, by way of example only, with reference to the accompanying drawings, in which:

Figure 1 is a block diagram of a computer system suitable for the operation of embodiments of the present invention;

Figure 2 is a component diagram of an exemplary computer system arrangement for provisioning a resource to a virtualised software installation in accordance with an embodiment of the present invention;

Figure 3 is an exemplary schematic data definition of a directory in accordance with an embodiment of the present invention;

Figure 4 is an alternative exemplary schematic data definition of a directory in accordance with an embodiment of the present invention; and

Figure 5 is a flowchart of an exemplary method of the computer system of Figure 2 in accordance with an embodiment of the present invention.

Figure 1 is a block diagram of a computer system suitable for the operation of components in embodiments of the present invention. A central processor unit (CPU) 102 is communicatively connected to storage 104 and an input/output (I/O) interface 106 via a data bus 108. The storage 104 can be any read/write storage device such as a random access memory (RAM) or a non-volatile storage device. An example of a non-volatile storage device includes a disk or tape storage device. The I/O interface 106 is an interface to devices for the
input or output of data, or for both input and output of data. Examples of I/O devices
connectable to I/O interface 106 include a keyboard, a mouse, a display (such as a monitor)
and a network connection.

Figure 2 is a component diagram of an exemplary computer system arrangement 200 for
provisioning a resource to a virtualised software installation 214 in accordance with an
embodiment of the present invention. A virtualised computing environment 212 is an
environment for the deployment of software installations such as installation 214. Software
installations can include software virtualisations of physical machines, application
deployments, services and the like. For example the virtualised computing environment 212
can be provided as a particular operating system executing within a virtual machine with a
hypervisor on a hardware device or, potentially, a distributed arrangement of hardware
devices. The virtualised computing environment 212 can be provided as a service-based
technology such that the environment 212 is delivered as a service for the installation and
execution of one or more software applications or services. In a preferred embodiment, the
virtualised environment 212 is provided as part of a Cloud Computing service provided by a
Cloud Computing service provider such as BT Cloud Compute available from British
Telecommunications plc. Additionally or alternatively, the virtualised computing environment
212 can be provided as, or operate with, a service based infrastructure and/or platform such
as IaaS and/or PaaS.

Deployment of the software installation 214 includes any or all of installing, configuring,
arranging and adapting one or more software components such that the installation 214 is
executable with the virtualised computing environment 212. For example, a web based
software application can be installed to execute with an operating system executing on a
virtual machine, the virtual machine being configured to include networking facilities and the
virtual machine also having installed thereon a web server having a certain configuration, a
database and certain other requirements defined for the application. All such installation and
configuration such that the web based software application is executable in the virtualised
computing environment 212 is part of the deployment of the application as a software
installation 214.

The software installation 214 has associated a deployment specification such as a
functional definition, component list or similar that is suitable for use in deploying, updating,
refreshing or executing the software installation 214 with the virtualised computing
environment 212. For example, a deployment specification can include a specification of an
architecture of the software installation 214 and/or an architecture of software components
required for the installation 214. Additionally or alternatively, the deployment specification
204 can include specifiers or descriptors of application or other software or platform components that are required for the deployment of the installation 214.

In some embodiments the virtualised computing environment 212 is provided as, or operates with, a service based infrastructure and/or platform such a Cloud Computing service, an IaaS service and/or a PaaS service. In such embodiments a deployment specification is further suitable for use in deploying, refreshing, updating or executing the software installation 214 with such services.

The deployment specification identifies one or more functional requirements for the deployment, updating, refreshing or executing of the installation 214 such that the application executes with the virtualised computing environment 212. Functional requirements are specifications of functions, features, components, facilities or services required for the software installation 214. For example, functional requirement can include: application of a particular security policy; a particular middleware architecture and configuration; a particular file storage mechanism such as a particular file system or data storage arrangement; a particular database architecture; a particular processing capability such as a particular processing performance or capacity; and other such functional requirements. By way of example, a functional requirement can be defined to implement a security policy for a software installation 214 in which all data is encrypted using a particular standard of encryption. Computing resources in the distributed computing environment employed to satisfy functional requirements of the installation 214. Thus a functional requirement to encrypt data can be satisfied by an encryption service, function, module or component of a resource provider 210 from potentially many such resource providers 210 in the distributed computing environment.

A resource provider 210 is a physical or virtualised computer system or multiple computer systems communicatively connected in or to the distributed computing environment and adapted to provide resources to consuming systems such as the software installation 214. The resources provided by resource providers 210 are employed by the software installation 214 to satisfy functional requirements of the software installation 214.

Resources can include functions, dataflows and/or technologies. Examples of function resources include bespoke functions, procedures, modules or components provided for resource consumers, such as a library containing functions embodying or supporting the a software component or a class of instantiable objects providing methods and routines of or for resource consumers. Examples of dataflow resources include communications between software components such as the invocation of a function, routine or method of a first component by a facility of a second component. A further example of a dataflow resource is
a coupling between two or more components such that messages are passed, requests are sent or data is shared between the two components. Such components can be internal to a resource consumer, part of a virtualised computing environment 212 or external to the a resource consumer and the virtualised computing environment 212. Examples of technology resources include particular software components, applications or facilities to be installed to deploy a software installation 214. For example, a technology resource can be a database software component from a particular technology vendor at a particular version, release or level. Further examples of technology resources include intrusion detection or prevention technologies, virus scanning technologies such as antivirus software, web servers, operating systems, middleware, encryption and message handling technologies.

Thus, resources can include, inter alia, software or hardware components, software packages, modules, applications, services or solutions, networking facilities, protocols, storage facilities including databases, middleware facilities, user interface facilities and connectivity services. A deployment specification defining functional requirements may explicitly identify resources such as an explicit identification of a particular database or web server facility. Alternatively or additionally, a deployment specification can be suitable for identifying a resource such that the identification is not explicit but is discernable. For example, an explicit identification of a web server resource for a software installation 214 further identifies dataflows between web page repositories, server side script repositories and the web server. Such dataflows are identified by a deployment specification while such identification is not necessarily explicit. In all cases a deployment specification identifies functional requirements which require resources for their satisfaction.

In one exemplary embodiment a functional requirement for a software installation 214 has associated one or more characteristics. For example, a functional requirement to implement an encryption security policy may further characterise the security policy by specifying characteristics of the policy, such as an encryption algorithm and key size. Such characteristics associated with a functional requirements can require specific resources. For example, an encryption service operating with a 128bit key length will be less computationally intensive than an encryption service operating with a 256bit key length. Accordingly a functional requirement for an encryption service having a characteristic defining a 256bit key length requires more processor resource than one having a characteristic defining a 128bit key length.

The computer system 200 of Figure 2 includes an interface 202 for communication with resource providers 210 and resource consumers such as the software installation 214. The computer system 200 further includes a receiver 204 as a software, hardware, firmware or
combination component for receiving a functional requirement for the software installation 214. The functional requirement may be received directly from the software installation 214 or from the virtualised computing environment in which the software installation 214 is deployed or to be deployed. Alternatively the functional requirement can be received from a different component or system that is functional to configure, deploy, execute or service the software installation 214 such as a deployment component or the like. The functional requirement requires one or more resources of one or more resource providers 210 for fulfilment.

The computer system 200 further includes a resource assessor 206 as a software, hardware, firmware or combination component including logic adapted to determine an amount of each resource required to fulfil or satisfy the functional requirement as an extent of each resource. In an exemplary embodiment the resource assessor 206 is further adapted to determine which resources are required to fulfil or satisfy the functional requirement, such as where the resource requirements are not explicitly identified in a functional requirement or where resource requirements are ambiguous based on the functional requirement.

In a preferred embodiment the resource assessor 206 determines the extent of each resource required (and optionally which resources are required) using a directory 216. The directory is a data store suitable for holding functional requirement information such that a set of one or more resources required to fulfil or satisfy a resource requirement can be determined along with an extent (or amount) of each resource that is required to fulfil or satisfy the resource requirement. For example the directory can be stored in a volatile or non-volatile storage, as files, in a registry or a database or other suitable directory storage means. Examples of suitable data structures of the directory 216 are considered below with reference to Figures 3 and 4.

Figure 3 is an exemplary schematic data definition of a directory 316 in accordance with an embodiment of the present invention. The data structure of the directory 316 of Figure 3 associates a functional requirement 320 directly with any number of ResourceExtent 326 data structures. A ResourceExtent 326 data structure is a holding data structure associating a single resource 322 with a single extent 324 (indicating an amount of the resource 322).

Accordingly, by a set of resources 322 required to fulfil or satisfy a functional requirement 320 can be determined by reference to the set of ResourceExtent 326 data structures. Thus each resource 322 indicated in a ResourceExtent 326 for a functional requirement 320 has associated a resource extent 324 indicating an amount, extent or degree of the resource 322 required to fulfil or satisfy the functional requirement 320. It will be appreciated that the measure of an extent 324 of a resource 322 required to fulfil or satisfy a functional
requirement 320 will depend on the nature of the resource. For example, a processor resource may be consumed according to a particular processing rate, for a particular period of time (whether in one block, spread over multiple operations or repeatedly), a number of processors, a number of processor cores, or other extents of consumption of processor resource. On the other hand, for example, a storage device may be consumed up to a certain amount of data. An extent of consumption of other resources can be expressed similarly depending on the resource type. Thus the directory of Figure 3 provides a relationship between a functional requirement, resources and an extent of consumption of those resources such that the resource assessor 206 is able to determine an extent of a resource required to implement the functional requirement.

Figure 4 is an alternative exemplary schematic data definition of a directory 416 in accordance with an embodiment of the present invention. Many of the features of Figure 4 are identical to those described with respect to Figure 3. Figure 4 differs to Figure 3 in that a functional requirement 420 is not directly associated with a ResourceExtent 426. In contrast, in Figure 4, a functional requirement is associated with any number of characteristics 430 which are each associated with any number of ResourceExtent 426 data structures. Each characteristic 430 corresponds to a feature, attribute, sub-requirement or parameter of a functional requirement 420 such that the functional requirement 420 is characterised. Accordingly, the resources 422 and the extent 424 of their consumption required for a resource requirement 420 can vary in dependence on the particular characteristics 430 selected for the resource requirement. By way of practical example, a functional requirement 420 can be a requirement for encryption, having a characteristic 430 of a key length of 256bits which requires a processor resource 422 operating at a particular speed for a particular duration (extent 424).

It will be appreciated by those skilled in the art that the directories of Figure 3 and Figure 4 are purely exemplary and alternative arrangements could be employed. Further, it will be appreciated that a combination of the data schemas of Figures 3 and 4 could be employed such that certain functional requirements have a direct mapping to a resource (such as via only a single placeholder characteristic) while others have associated characteristics.

Returning to Figure 2, the resource assessor 206 thus determines an extent of the resource required to implement the functional requirement for the software installation 214 and provides resource and extent information to a resource provider identifier 208.

The resource provider identifier 208 is a software, hardware, firmware or combination component including logic adapted to identify resource providers 210 in the distributed computing environment having availability of each of the required resources to the
determined extent. It will be appreciated that different resources could be provided by
different resource providers 210 such that a combination of the resources from different
providers 210, when taken together, provide all the resources required to fulfil or satisfy a
functional requirement. In an exemplary embodiment, the resource provider identifier 208
identifies resource providers having availability of the required resources to the required
extent by polling the resource providers 210 for availability of each resource and receiving
resource availability information from resource providers 210. Accordingly, in such an
exemplary embodiment, the resource provider identifier 208 is further adapted to determine
one or more resource providers 210 suitable for providing the required resources to the
required extent based on the information returned thereby.

In an alternative exemplary embodiment the resource provider identifier 208 identifies
resource providers having availability of the required resources to the required extent by
sending a request to each of at least a subset of the resource providers 210 indicating one or
more of the required resources and corresponding extents and receiving an indication
directly from resource providers 210 indicating availability of the resource and extent.

On identification of resource providers 210 having available resources to the required
extent the computer system 200 communicates an indication of the identified resource
providers 210 to the software installation 214 (or, in alternative arrangements, the component
from which the functional requirement was received) in order that the resources can be
provided to the software installation 214 to fulfil or satisfy the functional requirement of the
software installation 214. Thus in this way embodiments of the present invention determine
an amount or extent of resources required to fulfil or satisfy a functional requirement of a
software installation 214. Accordingly over-consumption of resources can be mitigated based
on the determined extent of resource consumption required. Furthermore overprovision of
resources can be mitigated by indication to resource providers 210 of an extent of
consumption required to fulfil or satisfy a functional requirement.

In one embodiment the computer system 200 makes or proposes an association between
one or more identified resource providers 210 and the software installation 214 such that the
software installation 214 consumes resources from the identified resource providers 210.
The association can be formalised as a data structure defining the service provider(s) 210,
the software installation 214, the resource(s) for consumption and the extent of such
consumption. Such a data structure can constitute an agreement between the resource
provider(s) 210 and the software installation 214. Thus the data structure defines, on one
hand, the resources and extent (amount) of those resources assured to be made available
by the resource provider(s) 210, and on the other hand, the resources and extent (amount) of
those resources that will be consumed by the software installation 214. Such a data structure can be recorded in a manner that mitigates repudiation of this association or agreement, for example by recording in a distributed sequential transactional database such as a blockchain data structure.

Distributed sequential transactional databases are well known in the field of cryptocurrencies and are documented, for example, in "Mastering Bitcoin. Unlocking Digital Crypto-Currencies." (Andreas M. Antonopoulos, O'Reilly Media, April 2014). For convenience, such a database is herein referred to as a blockchain though it will be appreciated that other suitable databases, data structures or mechanisms possessing the characteristics essential for embodiments of the present invention could alternatively be used. A blockchain is a distributed chain of block data structures accessed by a network of nodes, referred to here as a miner network. Each block in the blockchain includes a plurality of transaction data structures, each transaction referring or relating to a prior transaction. For example, in a preferred embodiment each blockchain includes a Merkle of hash or digest values for transactions included in the block to arrive at a hash value for the block, which is itself combined with a hash value for a preceding block to generate a chain of blocks (blockchain). A new block of transactions is added to the blockchain by miner software, hardware, firmware or combination systems in a miner network. The miners are communicatively connected to sources of transactions (such as the computer system 200 generating associations between resource provider(s) 210 and a consuming software installation 214) and access or copy the blockchain. A miner undertakes validation of the substantive content of a transaction (such as the criteria defined therein) and adds a block of new transactions to the blockchain when a challenge is satisfied, typically such challenge involving a combination hash or digest for a prospective new block and a preceding block in the blockchain and some challenge criterion. Thus miners in the miner network may each generate prospective new blocks for addition to the blockchain. Where a miner satisfies or solves the challenge and validates the transactions in a prospective new block such new block is added to the blockchain. Accordingly a blockchain provides a distributed mechanism for reliably verifying a data entity such as an entity constituting or representing an association between resource provider(s) 210 and resource consumers such as software installation 214. The detailed operation of such blockchains and the function of miners in a miner network is beyond the scope of this specification. The manner in which a blockchain and network of miners operate ensures that only valid transactions are added within blocks to the blockchain in a manner that is persistent within the blockchain. Transactions added erroneously or maliciously are not verifiable by other miners in the network and cannot persist in the blockchain. This attribute of blockchains can be exploited by embodiments of
the present invention to provide a distributed and reliable assurance for resource provider(s) 210 and resource consumers such as software installation 214 that resource consumption by a consumer is in accordance with an association or agreement recorded in the blockchain.

Figure 5 is a flowchart of an exemplary method of the computer system of Figure 2 in accordance with an embodiment of the present invention. Initially at step 502 the method receives a functional requirement for the software installation 214. At step 504 the method determines an extent of the resource required to implement the functional requirement. At step 506 the method identifies one or more resource providers in the distributed computing environment having availability of the required resource to the determined extent. Finally at step 508 the method communicates an indication of the identified resource providers to the software installation in order that the resource can be provided to the software installation to the determined extent to provide the functional requirement.

Insofar as embodiments of the invention described are implementable, at least in part, using a software-controlled programmable processing device, such as a microprocessor, digital signal processor or other processing device, data processing apparatus or system, it will be appreciated that a computer program for configuring a programmable device, apparatus or system to implement the foregoing described methods is envisaged as an aspect of the present invention. The computer program may be embodied as source code or undergo compilation for implementation on a processing device, apparatus or system or may be embodied as object code, for example.

Suitably, the computer program is stored on a carrier medium in machine or device readable form, for example in solid-state memory, magnetic memory such as disk or tape, optically or magneto-optically readable memory such as compact disk or digital versatile disk etc., and the processing device utilises the program or a part thereof to configure it for operation. The computer program may be supplied from a remote source embodied in a communications medium such as an electronic signal, radio frequency carrier wave or optical carrier wave. Such carrier media are also envisaged as aspects of the present invention.

It will be understood by those skilled in the art that, although the present invention has been described in relation to the above described example embodiments, the invention is not limited thereto and that there are many possible variations and modifications which fall within the scope of the invention.

The scope of the present invention includes any novel features or combination of features disclosed herein. The applicant hereby gives notice that new claims may be formulated to such features or combination of features during prosecution of this application or of any such
further applications derived therefrom. In particular, with reference to the appended claims, features from dependent claims may be combined with those of the independent claims and features from respective independent claims may be combined in any appropriate manner and not merely in the specific combinations enumerated in the claims.
1. A computer implemented method to provide a resource to a virtualised software installation in a distributed computing environment, the method comprising:

   receiving a functional requirement for the software installation;
   determining an extent of the resource required to implement the functional requirement;
   identifying one or more resource providers in the distributed computing environment having availability of the required resource to the determined extent; and
   communicating an indication of the identified resource providers to the software installation in order that the resource can be provided to the software installation to the determined extent to provide the functional requirement.

2. The method of claim 1 wherein the functional requirement has associated one or more characteristics and determining an extent of a resource required includes:

   accessing a directory of functional requirement characteristics, the directory defining, for each characteristic, an extent of the resource required to provide a functional requirement having the characteristic.

3. The method of claim 2 wherein the functional requirement includes a function of a security policy.

4. The method of claim 2 wherein the functional requirement is an entry in a deployment descriptor for one or more software components for execution for the software installation.

5. The method of any preceding claim wherein identifying resource providers includes the steps of:

   polling one or more resource providers for availability of the resource at each provider;
   receiving resource availability information from the one or more resource providers;
determining one or more resource providers suitable for providing the resource to the required extent.

6. The method of any of claims 1 to 4 wherein identifying resource providers includes sending a request to each of one or more resource providers including an indication of the required resource and the determined extent; and receiving an indication from each of one or more providers that the provider has availability of the required resource to the determined extent.

7. A computer system to provide a resource to a virtualised software installation in a distributed computing environment, the system comprising:

a receiver including logic adapted to receive a functional requirement for the software installation;

a resource assessor including logic adapted to determine an extent of the resource required to implement the functional requirement;

a resource provider identifier including logic adapted to identify one or more resource providers in the distributed computing environment having availability of the required resource to the determined extent; and

an interface via which an indication of the identified resource providers is communicated to the software installation in order that the resource can be provided to the software installation to the determined extent to provide the functional requirement.

8. A computer program element comprising computer program code to, when loaded into a computer system and executed thereon, cause the computer to perform the steps of a method as claimed in any of claims 1 to 6.
FIGURE 5

START

Receive Specification of Functional Requirement

Determine Extent of Resource Required to Implement Functional Requirement

Identify Resource Providers having Availability of Required Resource & Extent

Communicate Indication of Identified Resource Providers

STOP
**INTERNATIONAL SEARCH REPORT**

**A. CLASSIFICATION OF SUBJECT MATTER**

INV. G06F9/50

ADD.

According to International Patent Classification (IPC) or to both national classification and IPC.

**B. FIELDS SEARCHED**

Minimum documentation searched (classification system followed by classification symbols): G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched:

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used):

EPO-Internal, WPI Data

**C. DOCUMENTS CONSIDERED TO BE RELEVANT**

<table>
<thead>
<tr>
<th>Category</th>
<th>Citation of document, with indication, where appropriate, of the relevant passages</th>
<th>Relevant to claim No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>US 2012/198073 Al (SRI KANTH MUNI RATHNAM) [US] ET AL) 2 August 2012 (2012-08-02)</td>
<td>1-8</td>
</tr>
<tr>
<td></td>
<td>claim 1-3, 9, 10 figure 2 paragraph [0002] - paragraph [0004] paragraph [00027] paragraph [00031] - paragraph [0036]</td>
<td></td>
</tr>
</tbody>
</table>

[X] Further documents are listed in the continuation of Box C.  [X] See patent family annex.

* Special categories of cited documents:

**A** document defining the general state of the art which is not considered to be of particular relevance.

**E** earlier application or patent but published on or after the international filing date.

**L** document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified).

**O** document referring to an oral disclosure, use, exhibition or other means.

**P** document published prior to the international filing date but later than the priority date claimed.

**T** later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention.

**X** document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone.

**Y** document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

**S** document member of the same patent family.

Date of the actual completion of the international search: 22 July 2016

Date of mailing of the international search report: 23/08/2016

Name and mailing address of the ISA:

European Patent Office, P.B. 5818 Patentlaan 2
NL - 2280 HV Rijswijk
Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016

Authorized officer: rtz Hanno

Form PCT/ISA/210 (second sheet) (April 2005)
### DOCUMENTS CONSIDERED TO BE RELEVANT

<table>
<thead>
<tr>
<th>Category</th>
<th>Citation of document, with indication, where appropriate, of the relevant passages</th>
<th>Relevant to claim No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>WO 2014/202934 Al (BRITISH TELECOMM [GB]) 24 December 2014 (2014-12-24) abstract claims 1, 3, 4 figures 4, 5, 6a, 10, 13 page 6, line 1 - line 17 page 19, line 34 - page 20, line 25</td>
<td>1-8</td>
</tr>
<tr>
<td>Patent document cited in search report</td>
<td>Publication date</td>
<td>Patent family member(s)</td>
</tr>
<tr>
<td>----------------------------------------</td>
<td>-----------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>US 2012198073 AI</td>
<td>02-08-2012</td>
<td>CN 103733194 A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>EP 2668600 A2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 2012198073 AI</td>
</tr>
<tr>
<td></td>
<td></td>
<td>WO 2012102863 A2</td>
</tr>
<tr>
<td>US 2011106515 AI</td>
<td>05-05 -2011</td>
<td>NONE</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>US 2013036208 AI</td>
<td>07-02 -2013</td>
<td>CN 103703445 A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>EP 2740035 A1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>JP 2014524608 A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 2013036208 AI</td>
</tr>
<tr>
<td></td>
<td></td>
<td>WO 2013022619 A1</td>
</tr>
<tr>
<td>US 2014136710 AI</td>
<td>15-05 -2014</td>
<td>NONE</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Wo 2014202934 AI</td>
<td>24-12 -2014</td>
<td>EP 2816468 A1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Wo 2014202934 A1</td>
</tr>
</tbody>
</table>