(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
29 September 2011 (29.09.2011)

PCT

(10) International Publication Number
WO 2011/116819 A1

(54) Title: CACHING IN MOBILE NETWORKS
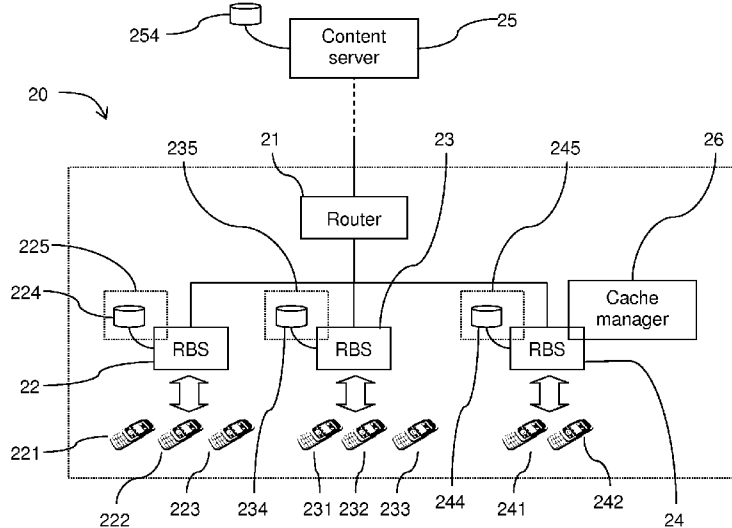
Figure 2

(57) **Abstract**: There is described a method for optimising the distribution of data objects between caches in a cache domain of a resource limited network. User requests for data objects are received at caches (225, 235, 245) in the cache domain. A notification is sent from each cache at which a request is received to a cache manager (26). The notification reports the user request and identifies the requested data object. At the cache manager, object information including the request frequency of each requested data object and the locations of the caches at which the requests were received is collated and stored. At the cache manager, objects for distribution within the cache domain are identified on the basis of the object information. Instructions are sent from the cache manager to the caches to distribute data objects stored in those caches between themselves. The data objects are distributed between the caches using transmission capacity of the network that would otherwise be unused.

1

CACHING IN MOBILE NETWORKS

Technical Field

5   The present invention relates to a system for caching data in mobile packet data networks. In particular, the invention relates to a caching architecture suitable for optimising the distribution of cached data in a resource limited network.

Background

10

Typical file caching methods include a cache receiving a file from a file server, and storing the entire file. Later when a client desires the file, instead of serving the file from the file server, the file is served from the cache. Because the cache is typically a server that is closer to the client, or has higher bandwidth than the file server, the file is

15   served to the client quickly from the cache.

This can be understood with reference to Figure 1, which is a schematic illustration of an exemplary architecture of a network 10 having an origin server 11 and a number of caches 12-16. Clients 17-19 are configured to receive files and/or streaming data from

20   the origin server 11 or the caches 12-16.

In order to reduce the load on the origin server 11 and to save bandwidth in the delivery network 10, some of the content is stored in caches 12-16 closer to the end users 17-19. It is desirable to push these caches as close to the end user as possible.

25

For example, mobile network architectures generally comprise a Core Network (CN), to which are attached a number of radio access networks (RANs) in which terminals are connected to radio base stations (RBSs). The terminals may move and as they move they may be connected wirelessly to different RBSs. The transmission links between

30   the RBSs and the rest of the network are usually rented and limited in capacity. One way of reducing the demands on these links is to use caching and, as described above, caching should preferably take place as close to the terminals as possible to minimize the need for transmission.

2

One problem with caching in the RAN (with the cache being a part of the RBS or other cell node) is that each cache will receive traffic from only a small population of terminals. In general, the content stored by a cache will be the content which has been requested by the end-users. If the population using the cache is small, the volume of
5   cached content stored in that cache is small. Statistically, the probability that someone else wants to download the same content from that cache, *i.e.* the "cache hit rate", decreases with a smaller population. One large cache for a large population is therefore statistically more efficient than many small caches serving different smaller subsets of the population.
10

Summary

It is an object of the present invention to obviate at least some of the above disadvantages. In general, it is desirable to generate a large, virtual cache out of many
15   small ones to solve the problem above. However, the transmission is limited in terms of capacity and must be efficiently used and therefore a distributed cache architecture must consider that limitation.

In accordance with one aspect of the present invention there is provided a network
20   element configured to act as a cache for sending cached data in a session to a mobile terminal in a packet data network. The network element is associated with a cache storage unit for storing data objects. The network element comprises a communications system for communication with other elements in the network. A DPI or proxy agent is configured to inspect packets from the terminal passing through the
25   network element to determine if they include a request for a data object, determine whether the requested data object is stored in the cache storage unit and, if so, send the requested data object towards the terminal. An object agent is configured to receive instructions from a cache manager controlling caches in a cache domain of the network, and in response to send data objects to other caches in the cache domain
30   using transmission capacity that would otherwise remain unused.

The object agent may also be configured to receive data objects not requested by the mobile terminal from other caches in the cache domain, and to store the received data objects in the cache storage unit. This may again use transmission capacity that would

3

otherwise remain unused, and may be in response to instructions from the cache manager to other caches in the cache domain.

The object agent may be configured to capture each event of a request for a particular
5   data object, and pass this information to the cache manager to assist in identifying which data objects should be distributed between caches.

In other words, requested data objects may be distributed between caches in a cache domain, under the control of a cache manager, only when there is available capacity in
10   the network, so that the sending and/or receiving does not interfere with transmission of higher priority data transmission, *e.g.* requests of data objects from clients or non-cached data objects.

The DPI or proxy agent may be configured so that, if the requested data object is not
15   stored in the cache storage unit, the request is passed towards an upstream network element or the cache manager. When the requested data object passes through the network element when it is returned towards the terminal, it may be saved in the cache storage unit. The network element may be a Radio Base Station.

20   The object agent may be configured to use transmission capacity that would otherwise remain unused by sending data objects (optionally, only low priority data objects) to the other caches during off-peak data traffic periods. The object agent may also be configured so that high priority objects may still be sent to other caches during peak periods in response to instructions from the cache manager. This means that high
25   priority data objects can still be cached in other caches even during peak periods. High priority objects may be those requested by another mobile terminal through another cache.

The object agent may be configured to capture each event of a request for a particular
30   data object, and pass this information to the cache manager to assist in identifying which data objects should be distributed between caches.

The cache storage unit may be included in the network element.

4

In accordance with another aspect of the present invention there is provided a cache manager for controlling the distribution of content stored in a set of caches called a cache domain of a packet data network (optionally a resource-limited network). The cache manager comprises an object monitor for receiving notifications from caches in

5   the cache domain about data objects requested at those caches. An object database is configured to collate and store object information about the requested data objects, the object information including information about the caches at which the data objects have been requested and frequency with which the requests have been made. An object distributor is configured to identify objects for distribution on the basis of the

10   object information and instruct the caches in the network to distribute data objects stored in those caches between themselves using transmission capacity of the network that would otherwise be unused. It will be appreciated that the network may include one or more cache domains.

15   The cache manager may be configured to control the distribution of the data objects between caches in the cache domain so that the cache domain functions as a virtually large cache. The cache domain may correspond to a radio access network in the mobile network. The object distributor may be configured to instruct the caches in the cache domain to distribute high priority objects between themselves during peak

20   periods.

The cache manager may be a discrete entity and include a communications system for communicating with elements in the network. It may be associated with one of the caches in the cache domain. Alternatively, the functionality of the cache manager may

25   be distributed between network elements in the cache domain.

The cache manager may further comprise a topology monitor for retrieving topology information about the topology of the cache domain. A topology database for storing said topology information may also be provided. The object distributor may be

30   configured to make use of the topology information in the determination of how the objects should be distributed between the caches. In more detail, the object distributor may decide which objects should be stored at which caches, and when and from where a necessary fetch should occur (*i.e.* suitable times and places to distribute content from one or more local caches to one or more other local caches).

35

5

The topology information may include data relating to one or more of the following: a topology structure of the cache domain; bandwidth limitations of links between elements in the topology structure; location of caches in the topology structure; current loads sustained by the links between elements in the topology structure; and transport types for data within the cache domain.

In accordance with another aspect of the present invention there is provided a method for optimising the distribution of data objects between caches in a cache domain of a resource limited network. User requests for data objects are received at caches in the cache domain. A notification is sent from the cache at which a request is received to a cache manager. The notification reports the user request and identifies the requested data object. At the cache manager, object information including the request frequency of each requested data object and the locations of the caches at which the requests were received is collated and stored. At the cache manager, objects for distribution within the cache domain are identified on the basis of the object information. Instructions are sent from the cache manager to the caches to distribute data objects stored in those caches between themselves. The data objects are distributed between the caches using transmission capacity of the network that would otherwise be unused.

The functionality of the cache manager may be distributed between network elements in the cache domain. The cache manager may take into account the topology of the network and/or the priority of the requested data objects and/or the frequency of requests for particular data objects when identifying the data objects for distribution.

The network may be a mobile network, and the caches may be associated with or part of RBSs.

The data objects may be distributed between caches during off-peak periods. The method may also comprise labelling a data object as a high priority data object if it is not stored in the cache at which the user request for that data objects is received, and sending instructions, from the cache manager to a cache at which the high priority data object is stored, to send the high priority data object to the cache at which the user request was received during a peak period.

6

In accordance with a further aspect of the present invention there is provided a computer program product comprising code adapted to be executed on a network element associated with a cache storage unit in a resource limited network. The code is operable to cause the network element to inspect packets, from a terminal in the network, passing through the network element. The code is further operable to cause the network element to determine if said packets include a request for a data object. The code is further operable to cause the network element to determine whether the requested data object is stored in the cache storage unit and, if so, send the requested object towards the terminal. The code is further operable to cause the network element to receive instructions from a cache manager controlling caches in a cache domain of the network to send data objects to other caches in the cache domain using transmission capacity that would otherwise remain unused. The code is further operable to cause the network element to send the data objects in accordance with the received instructions. The code may also be operable to report request statistics to cache managers and to receive and store (and possibly delete) objects as decided by the cache manager.

In accordance with a further aspect of the present invention there is provided a computer program product comprising code adapted to be executed on a cache manager in a resource limited network. The code is operable to cause the cache manager to retrieve notifications from caches in a cache domain controlled by the cache manager, the notifications including details of data objects recently requested at those caches. The code is further operable to cause the cache manager to collate and store object information about the requested data objects, the object information including the location and popularity of requested data objects. The code is further operable to cause the cache manager to identify objects for distribution on the basis of the object information. The code is further operable to cause the network element to instruct the caches in the network to distribute data objects stored in those caches between themselves using transmission capacity of the network that would otherwise be unused.

The invention also provides a computer program product as described above carried on a carrier medium.

Brief Description of the Drawings

Some preferred embodiments will now be described by way of example only and with reference to the accompanying drawings, in which:

5   Figure 1 is a schematic illustration of a network having an origin server and a number of caches;

Figure 2 is a schematic illustration of a Radio Access Network with Radio Base Stations configured to act as caching servers;

10

Figure 3 is a flow diagram illustrating a procedure for serving a user request;

Figure 4 is a schematic diagram illustrating an information model for a cache manager;

15   Figure 5 is a schematic diagram of an RBS configured to act as a cache server; and

Figure 6 is a schematic diagram illustrating the management of cached data in the network of Figure 2.

20   Detailed Description

As discussed in the background above, when designing a caching system, there is typically a choice between a centrally placed cache or locally placed caches . In both cases there is little gain in transmission, either because the placement is high up in the
25   network (in the case of the centrally placed cache) or because the hit rate becomes too low (in the case of local caches).

It is therefore desirable to combine the hit rate of a central cache with the transmission advantages of local caches. This can be achieved by the use of a "cache manager"
30   which ensures that local caches enjoy a hit rate close or equal to that of a central cache. It does so by pre-loading the local caches according to global statistics (hence the high hit rate) and ensures that this preloading takes place at off-peak times (so as not to violate the bandwidth savings).

Effectively, therefore, a large, virtual cache is generated out of many small ones, in which the limitations of the transmission are taken into consideration. This can be achieved by "cache equalization" which distributes content between the caches. Each small cache stores content requested by a large population (detected and cached by the other small caches) and this increases the local hit rate for popular content. The object data distribution between the small caches can be seen as a form of predictive caching, where the predictions are based on the behaviour of users receiving data through other caches.

The price to be paid is that the volume of data transferred between the caches will increase. If the bandwidth is restricted between the caches, the additionally transferred data will increase the load loads and eventually cause congestion. Methods that can help to solve content distribution in a resource limited network are thus of importance.

Traffic varies over a typical 24 hour period and there is usually a significant amount of free capacity during night hours. Furthermore, even at peak hours there are periods during which free capacity is available. It is thus possible to distribute low volumes of high priority data objects during peak hours (*e.g.*during the day) and larger volumes of less critical data objects during off-peak hours (*e.g.,* at night). In this context, a data object will be understood as meaning any set of data requested by a terminal and/or stored by a cache.

Figure 2 is a schematic illustration of a RAN 20 comprising a router 21 connected to three RBSs 22, 23, 24, each connected to terminals 221, 222, 223; 231, 232, 233; 241, 242. Each RBS 22, 23, 24 is associated with a cache storage unit 224, 234, 244 which may or may not be built into the RBSs themselves. The RBSs 22, 23, 24 and router 21 are interconnected by transmission links. Each RBS is enabled to act as a cache server – *i.e.* it includes a logical entity that acts like an embedded proxy or uses deep packet inspection (DPI) to detect and direct end-user requests such as HTTP-GET. For convenience, the combination of the caching functionality in a cache server (*e.g.* RBS 22) and cache storage unit (*e.g.* 224) will be referred to as a "cache" in this document. This applies whether or not the cache storage unit is physically built in to the cache server. This means that there is effectively a cache 225, 235, 245 associated with each RBS 22, 23, 24.

9

Although Figure 2 shows a tree-topology with caches 225, 235, 245 operated by RBSs 22, 23, 24, it will be appreciated that this is an example only. As will become apparent, the mechanisms for distributed cached data described herein may apply to many other topologies and scenarios where there is limited bandwidth between distributed caches

5   225, 235, 245. The major reason for distributing the caches in this way is that they can provide a (significant) transmission gain from the *first* hop (between the router 21 and RBSs 22, 23, 24 in Figure 2).

As indicated above, the problem with such caches is related to the fact that, since each

10   RBS 22, 23, 24 only supplies data to a few terminals 221, 222, 223; 231, 232, 233; 241, 242, the amount of traffic from each RBS 22, 23, 24 is small. The "hit-rate" from local cache traffic is small. The solution is to increase the hit rate by distributing content that is locally cached by any of the small caches 225, 235, 245 in the group to other small caches in that group. The caches in the group will then contain content

15   requested at other caches and this will increase the hit rate.

Thus if content is requested by one of the terminals 222, this request passes through the RBS 22 to which the terminal 222 is connected. Content data is delivered from a content server 25 (associated with a content storage unit 254) upstream in the network

20   (for example in a core network, not shown) through the RBS 22 to the terminal 222. As well as delivering the content data to the terminal 222, the RBS 22 saves the content data in its associated cache storage unit 224. Then the next time the terminal 222 (or any of the other terminals 221, 223 attached to that RBS 22) requests that content, it can be extracted from the cache storage unit 224 by the RBS 22 and supplied to the

25   terminal 222. In addition, the content data is copied to the other RBSs 23, 24 in the RAN 20 and saved in their associated cache storage units 234, 244 so that it can also be supplied, if requested, to any of the other terminals 231, 232, 233; 241, 242 in the RAN.

30   The problem with the distribution is that the links are limited in bandwidth, such that updates can cause congestion, in particular if they take place during busy hours. Therefore, the updates should be controlled so as to optimise the use of the bandwidth. This optimisation may include some or all of the following:

- The use of otherwise unutilised transmission capacity (*e.g.* during off-peak
35   times).

- Potentially popular data is prioritised. For example, data objects for distribution can be queued and served as bandwidth becomes available and according to a queuing discipline based on data object popularity.

- The distribution of rarely used data objects is restricted in cases where storage capacity is a limiting factor. Some data may not be distributed. If required, it can be extracted from the cache in which it has been saved.

Furthermore, it may be that a request is made *via* an RBS 22 for a data object not stored in the cache 225 associated with that RBS, but which is stored in a nearby cache 235, 245. In this situation the data object may be marked as high priority and fetched from the nearby RBS 235, 245 immediately, rather than waiting for bandwidth to become available.

In order to control the distribution of data between the caches 225, 235, 245, a centralised cache manager unit 26 is operatively attached to one of the nodes in the RAN 20. In Figure 2, it is shown as being associated with one of the RBSs 24, but it will be appreciated that it could also be associated with any of the RBSs 22, 23, 24 or any other network element such as the router 21, for example. The cache manager unit 26 controls the spreading of information among a set of distributed caches. It will be appreciated that it is also possible to devise distributed cache manager solutions which provide the same functionality. The important feature is that the cache manager (whether it is a single unit 26 as shown in Figure 2 or a distributed system) is able to optimise the distribution of data between caches 225, 235, 245.

The set of caches controlled by a particular manager may be considered as a "caching domain" (CD). CDs may be mutually exclusive or overlapping. The manager will be placed at or associated with a site with a central position in the topology of its domain, and it may be part of an RBS (or any other cache entity) at that site. It will be appreciated that, in the example shown in Figure 2, distribution of cached data is described as being shared between RBSs 22, 23, 24 of a single RAN, but a CD need not be restricted to a RAN, or may be only part of a RAN.

The cache manager 26 is aware of the popularities of all data objects requested at any RBS 22, 23, 24, all possible occurrences in caches 225, 235, 245, and bandwidth limitations in the network, and it is configured to use this information to manage cache

11

updates in accordance with the above. In addition, it is configured to detect and redirect "failed" data requests from users such that requested data objects which cannot be delivered by a cache immediately adjacent to a user may be fetched from other caches in the same CD in preference to the content server 25. In this way the cache manager

5   26 can provide a (significant) transmission gain from the *second* hop (between the router 21 and higher layers in Figure 2) and thus again use links more efficiently. It will be noted that this aspect is particularly attractive where the connections are provided by rented transmission or microwave links

10   In other words, the cache manager 26 increase the "effective" hit rate of the small local caches 225, 235, 245 by distributing content among them during times when there is free bandwidth. The content to be pushed is determined by analyzing data requests from the different caches 225, 235, 245. Content which has been requested on one or more RBSs (and thus cached at the corresponding cache storage units 224, 234, 244)

15   during peak hours will be distributed to the other caches (and thus available for delivery when requested on these RBSs) during off peak hours. In this way some traffic is moved from peak hours (the time when requests are made) to off peak hours (the time when the caches are pre-populated). The content of the different cache storage units 224, 234, 244 will thus diverge during the peak hours (since users at different RBSs 22,

20   23, 24 request different data objects) and converge during off peak hours (since the caches are equalized by copying content in any cache to all caches).

The cache manager knows where data objects are found and decides what to store on each cache. It is therefore possible that it may decide not to store a data object at all,

25   or to store it but only at some caches but not at others. The choice of which caches to store in may depend on bandwidth, disc space, popularity in general and popularity in certain user groups.

A typical procedure for serving a user request is shown in Figure 3 and proceeds as

30   follows:

S1   A user of a mobile terminal 222 issues a request (*e.g.* a HTTP-GET) for piece of content.

S2   The request is intercepted at the site of the RBS 22 (or other network element) by a proxy/DPI logic unit 53 (see Figure 5) of the cache 225 (which may or may not be built into the RBS 22).

S3   The logic unit 53 interrogates the cache storage unit 224 associated with the RBS 22 to see if the content is stored in that cache storage unit 224

S4   If so, this known as a "primary cache hit": the content is retrieved from the cache storage unit 224 and a local delivery to the terminal 222 takes place. In this case bandwidth savings have been made on all the links in Figure 2.

S5   If there is a first cache miss, *i.e.* if the requested content is not available in the cache storage unit 224, the request will be forwarded, either to a "real" content server 25, to a higher layer proxy, or to the cache manager 26.

S6   The request is intercepted (on its path to the content server 25 or a higher layer proxy) or arrives at the cache manager 26 (as the chosen destination).

S7   The cache manager 26 analyses the request and examines if the requested content is found in any other local cache storage unit 234, 244 in the CD.

S8   If so (*i.e.* there is a secondary cache hit), the request is forwarded to whichever cache 235 contains the content.

S9   The content is retrieved from the cache storage unit 234 and delivered to the user.

S10   The first cache (RBS 22) may intercept the delivery and store the content locally in its associated cache storage unit 224 for future requests. In this case bandwidth savings are made on all links above the router 21 in Figure 2.

S11   If there is a secondary cache miss (*i.e.* if the requested content is not available in any of the other cache storage units 234, 244 in the CD), or if dictated by link load conditions towards other local caches, the request for the content is forwarded to the content server 25 or to some higher layer proxy.

S12   The content is delivered to the user from the content server 25 or higher layer proxy.

S13   The first cache (RBS 22) may intercept the delivery and store the content locally in its associated cache storage unit 224 for future requests.

In this way local caches 225, 235, 245 will be filled with content that has been requested by the local users. The content is delivered from different sources; (i) from the local cache itself (if it has been requested by other users of that RBS or site or pushed to that site during off-peak hours, which can be any period of time during which

13

free bandwidth is available), (ii) from another local cache in the CD (if it has recently been requested by another user in the CD), or (iii) from more distant sources (if it has not been requested by any user in the CD). The largest transmission gains are obtained when content is fetched from the local cache, significant gains are obtained

5   when content is fetched from another local cache, and small or no gains are obtained when the content is fetched from a distant source (although caching the data at this stage may lead to gains in the future).

It is thus clear that this process is most efficient if each local cache storage unit 224,

10 234, 244 is filled with content that has been requested by other users in the entire CD, and which therefore with a high probability will be requested by the local users. If this is the case, primary cache hits (S4) will occur more frequently, reducing the need to carry out steps S7-S13 with their associated use of transmission links and bandwidth. In other words, this increases the fraction of requests for which the highest

15 transmission gains (i) can be obtained.

In order to achieve this, the cache manager 26 collects information about such requests (which predominantly occur during peak hours) and then uses this information to pre-populate caches (predominantly during off peak hours).

20

For example, when an data object X is requested for the first time in the CD (*e.g.* by terminal 221), X is fetched from the content server 25 or a higher layer proxy and stored in the cache 225 through with the request is routed. If X is requested soon after by the same terminal 221 or another terminal 222, 223 *via* the same RBS 22, it is

25 fetched from the cache 225 and delivered to the user. If X is requested soon afterwards *via* a different RBS 23 in the same CD, it is fetched from the cache storage unit 224 associated with the first RBS 22 and stored also in the cache storage unit 234 of the second RBS 23. Then, at some off peak time, the cache manager 26 ensures that X is distributed to all the caches 225, 235, 245, so that subsequent requests for X

30 are served by the cache at which the request occurred.

Figure 4 is a schematic diagram illustrating a suitable information model for the cache manager 26. The manager includes the following information stores:

35 - A topology database 41 which stores information related to:

- Topology structure of CD: (*e.g.* tree, star, *etc.*)
- Bandwidth limitations of the links in the topology.
- Location of caches in the topology.
- Current link loads. The topology may support multiple traffic types (*e.g.* interactive traffic and conversational traffic). The link load monitors the load on the links between network elements to avoid the distribution of data objects causing congestion.
- Transport types: Multicast or unicast.

- An object data-base 42 which stores information related to:
- Locations of data objects, *i.e.* the cache(s), if any, in which different data objects are stored.
- Popularity of data objects including historic information such as previous requests for the data object.
- Cached data objects, *i.e.* a list of recently cached data objects.

- An object distribution status database 43 which includes:
- Object distribution status: this may be a status flag recording the suitability for distribution of data objects: (*i.e.* "should be distributed", or "should not be distributed") or a parameter with a higher resolution so that distribution can be decided on a cache-by-cache basis.
- Priority: data object distribution priority.
- Wall-clock time: timing information for determining the point at which distribution should start.

The functionality in the manager 26 may be divided into the following blocks:

- Topology monitor 44: a function that retrieves or collects information about the topology of the distributed caches 225, 235, 245.

- Object monitor 45: a function that retrieves or collects information from local caches 22, 23, 24 within the CD about recently cached data objects. The information may be obtained, for example, by regular polling of the caches, by subscribing to information from the caches, or by monitoring and analyzing the traffic between caches and servers. Other possibilities will be apparent to one skilled in the art.

15

- Object distributor 46: a function that identifies data objects which should be redistributed and performs the redistributions. These actions may be a continuous process or they may take place at a certain frequency, at certain times and/or at certain load conditions.

- Object prioritizer 47: a function that uses popularity statistics, for example, to estimate the probability of multiple requests of an data object in order to prioritize between different data objects which have been identified for redistribution.

- Resource manager 48: a function that ensures that redistributions do not drive the network into overload. This can be achieved, for example, by limiting the number of data objects to be redistributed for a single specific off-peak hour but then continue distribution the next off-peak hours or, when applicable, by selecting from which cache an data object should be delivered.

- Communications system 49: a function that controls communication with other network elements in the CD.

It will be appreciated that the logical elements shown in Figure 4 all interact with each other to enable the cache manager 26 to perform the functions described.

In addition (and not shown in Figure 4), there may be a request match maker, the function of which is to redirect requests from caches where a miss occurred to caches where the data object is cached and thus a hit will occur. In this way bandwidth can be saved at substantial bandwidth savings even in the presence of cache equalization delays.

Figure 5 is a schematic diagram illustrating the functionality of a RBS 22 and associated cache storage unit 224 acting as a cache 225. It will be appreciated that a cache can be provided at, or associated with, any suitable network element as well as or instead of a RBS, and that functions described as being operated by the RBS 22 may be operated by a separate entity.

16

The RBS 22 includes a controller 51 and communications system 52 for operating the usual functions of an RBS (*e.g.* communications with terminals, routers, forwarding traffic *etc.*). Where the RBS is configured to act as a cache 225 it is also associated with a cache storage unit 224, and includes two further functions; a DPI 53 and an object agent 54, described below.

The functionality in the cache 225 may be described as:

- DPI 53: a function that inspects passing packets to find those packets that include requests for information such as HTTP-GET. It can be implemented as a HTTP-proxy or a Deep Packet Inspection device.

- Cache storage unit 225: a storage space such as a hard-disk. This may be separate from the RBS 22 (as shown in Figure 5) or an integral part of the RBS 22.

- Object agent 54: an interface towards the cache manager 26 (*via* the communications system 52 if the cache manager 26 is not part of the RBS 22). The object agent 54 can also distribute information over a TCP/UDP session with low priority such that other traffic (e.g. interactive and conversational) is not disturbed during the transfer. This can be done in two ways: Using a "lower than best-effort" priority class, or a TCP session that backs off faster that usual TCP.

Figure 6 is a schematic diagram illustrating one example of how the distribution of cached data can be managed in the network shown in Figure 2. The information flow is as follows:

S61   The topology monitor 44 monitors the topology of the network elements in the CD, for example using the Simple Network Management Protocol (SNMP), and stores the updated knowledge into the topology database 41.

S62   The object monitor 45 continuously fetches from the local caches 225, 235, 245, *via* the object agent 54 in those caches or by intercepting content requests, information about data objects recently stored in the caches 225, 235, 245 (for example as a result of a request S62a from a terminal 222).

S63   This information is stored into the object database 42.

17

S64   At a specific time or at specific load condition, the object distributor 46 inspects the object database 42 for new data objects and compiles a list of new data objects.

S65   The list of new data objects is passed to the object-prioritizer 47. The "object prioritizer" assigns priorities to the data objects and creates a prioritized list of data objects.

S66   The prioritized list is then transferred to the resource manager 48.

S67   The resource manager 48 fetches information from the topology database 41 about the topology status and sets the number of data objects to be transferred based on the topology information.

S68   A new list is then transferred back to the object distributor 46.

S69   Based on the processed list, the object distributor 46 creates a set of object distribution requests which are sent to the object agent 54 in one or more of the caches 225. Several requests can be sent which may be distributed over longer periods of time. The requests can include information about the source (a local cache at which the data object is found), the sink(s) (the local caches at which the data object should be stored) and about distribution method (unicast, multicast).

S70   The object agent 54 initiates transfers after receiving such requests. Such transfers may be implemented in many ways. As a first, straightforward example, the object distributor 46 at the manager may instruct object agents at destination caches to issue ordinary request message (i.e. HTTP-GET messages) and to ensure that these requests be sent to the object agents of the source local caches. As a second, more advanced, example the object distributor 46 at the manager may instruct object agents at source caches to set up broadcasts and instruct object agents at specified sets of destination caches to listen to these broadcasts.

The sequence above is an example of updates in a distributed caching architecture. As a complementary example, some of the functionally provided by the object monitor 45, object distributor 46, object prioritizer 47 and resource manager 49 may be implemented as one or more object transfer queues (OTQs).

18

OTQs contain data objects (or references to data objects) which are to be updated at one or more caches. The requests are served according to a priority based discipline where the most urgent updates are served first.

5

An data object in an OTQ may be an update including a pointer to a source (*e.g.* cache 225) and one or more pointers to receivers (*e.g.* caches 235, 245). One can envisage a single, global OTQ; one OTQ for fetching data objects from sources and another OTQ for dispatching data objects to receivers; or, optionally, a "fetch-OTQ" and a "dispatch-

10   OTQ" for each local cache. It will be noted that separate queues are a logical notation rather than a physical implementation.

The object monitor 45 and object prioritizer 47 may be responsible for adding elements to the OTQs as new or updated data objects are observed. Queued entries may be

15   replaced or preceded by other entries as more recent updates or more popular data objects are observed.

The resource manager 48 may supervise the traffic from each cache and activate the first entry in the fetch-OTQ of the corresponding cache when the traffic is below a

20   certain threshold. Similarly, it may supervise the traffic to each cache and activate the first entry in the dispatch-OTQ of the corresponding cache when the traffic is below a certain threshold. Data objects which have been fetched but not yet dispatched would in the meantime be stored by the manager itself. This "global stand-by scheduling" should use all available bandwidth in order to minimise the transfer times and thus

25   maximise the hit rates at the local caches.

As noted above, the term "stand-by" is related to the fact that requests are served as free capacity (which may be identified, for example, in terms of link load) is detected. The meaning may, however, be expanded and not only refer to requests but to single

30   packets. This can be accomplished either by traffic differentiation in MPLS, in the link scheduling at the two end points or by means of a "soft" end-to-end protocol which yields to congestion much faster and stronger than other traffics (i.e., a protocol which is more responsive than TCP).

It will be appreciated that there is a sense in which all local caches 225, 235, 245 have agents, that is to say "slave agents" which listen to and obey a "master", *i.e.* the cache manager 26. A distributed implementation of the cache manager may extend these local slave agents or implement new local managers as "master agents".

Furthermore, it will be appreciated that slightly different modes of communication between RBSs 22, 23, 24 and cache manager 26 are possible. For example, consider two cases, a primary cache hit and a primary cache miss (as in steps S4 and S5 of Figure 3), in a local cache at RBS 22.

If there is a hit (*i.e.* content is requested which is present in the cache storage unit 224), the RBS 22 may or may not inform the cache manager 26 of the request due to size or security reasons. Some data objects may contain viruses or that the portal is not trusted. This can be described by local policy-rules in the cache. Informing the manager may help the manager to improve its plans for placing content at other caches (more popular content should perhaps be shared to other caches sooner) and, in particular, to improve its suggestions for identifying data objects which can be deleted when caches are full (less popular content should go first when a cache is full and the "global" view of the cache manager may help in identifying these data objects in a statistically accurate way).

If there as a miss (*i.e.* the requested content is not present in the cache storage unit 224) the requesting local cache could fetch the content from somewhere else (typically a higher layer cache or the content server 25) and then inform the cache manager 26 about its new acquisition. Alternatively, prior to the request it could ask the cache manager 26 where the content should be fetched from, and the cache manager 26 can direct the RBS 22 to another local cache. In a further alternative, all requests could be handled through the cache manager 26 so that the requesting, local cache is unaware of the source of the content.

The arrangements described above enable a distributed caching architecture to perform a similar function to a large cache, even where resources are limited. High caching hit-rates can be provided independently of where in the topology the cache is located.

The updates of the caches can be performed without disturbing other traffic, *e.g.* during non-busy hours, to ensure that the distribution does not require more capacity in the transport network.

5   The effectiveness of this approach can be illustrated by simulation of two scenarios.

The first scenario was designed to resemble video on demand, and assumes a system with 16,000 data objects (movies) where content servers operate on the level of 100 RBSs each of which receives about 100 requests per day (hence the server is exposed

10   to a total of 10,000 requests per day). Each day 16 new data objects (movies) are added. The request rate of the different data objects follows a Zipf-Mandelbrot distribution such that the request rate of the r:th most popular data object is proportional to $(r+q)^{-\beta}$ with $\beta = 0.75$ and $q = 80$.

15   In this example, the intention was to compare three different cases, ignoring possible memory limitations. In the first case a central cache was placed at a content server which also acts as a cache manager (hence its caching gain is determined by the request rates of all RBSs); in the second case distributed caches were placed at the RBSs and updated once per day (it will be noted that this is a relatively pessimistic

20   assumption about updates); and in the third case distributed caches were placed at the RBSs, but without controlled updates.

The results of the simulation of this scenario are illustrated in Figure 7, which shows the variation of hit rate over time for the three cases 71, 72, 73. It was found that the

25   hit rate in the second case 72 was almost the same as the first case 71, but was much lower in the third case 73. In other words the hit rate at the distributed caches is almost the same as that of a central cache if updates are used, but much lower when updates are not used.

30   The second scenario was designed to resemble web data objects, and assumes a system with 1,600,000 data objects (pages *etc.*) where managers operate on the level of 100 RBSs each of which receives about 10,000 requests per day (so the manager is exposed to a total of 1,000,000 requests per day). Each day 16,000 new data objects (pages *etc.*) are added and 32,000 data objects (pages *etc.*) are updated. The request

35   rate of the different data objects follows a Zipf-Mandelbrot distribution such that the

21

request rate of the r:th most popular data object is proportional to $(r+q)^{-\beta}$ with $\beta = 1.00$ and $q = 0$ while the ranks of data objects to be inserted or updated are drawn from a similar distribution with $\beta = 0.25$ and $q = 0$.

5   Again the intention is to compare different cases and so possible memory limitations are again ignored. The same three cases as above are compared, and the results are shown in Figure 8: in the first case 81 a central cache is placed at a control server which also operates as a cache manager (hence its caching gain is determined by the request rates of all RBSs); in the second case 82 distributed caches are placed at the

10   RBSs and updated once per day; and in the third case 83 distributed caches are again placed at the RBSs but this time without controlled updates.

Again, it was found that the hit rate of distributed caches is almost the same as that of a central cache if updates are used, but much lower if updates are not used.

CLAIMS:

1.   A network element (22) configured to act as a cache for sending cached data in a session to a mobile terminal in a packet data network, the network element being associated with a cache storage unit (224) for storing data objects, the network element comprising:

a communications system (52) for communication with other elements in the network;

a DPI or proxy agent (53) configured to inspect packets from the terminal passing through the network element to determine if they include a request for a data object, determine whether the requested data object is stored in the cache storage unit and, if so, send the requested data object towards the terminal; and

an object agent (54) configured to receive instructions from a cache manager (26) controlling caches in a cache domain of the network, and in response to send data objects to other caches in the cache domain using transmission capacity that would otherwise remain unused.

2.   The network element of claim 1, wherein the object agent (54) is further configured to receive data objects not requested by the mobile terminal from other caches (23, 24) in the cache domain, and to store the received data objects in the cache storage unit (224).

3.   The network element of claim 1 or 2, wherein the DPI or proxy agent (53) is configured so that, if the requested data object is not stored in the cache storage unit, the request is passed towards an upstream network element or the cache manager (26) and, when the requested data object passes through the network element when it is returned towards the terminal, it is saved in the cache storage unit.

4.   The network element of claim 1, 2 or 3, wherein the network element is a Radio Base Station (22).

5.   The network element of any of claims 1 to 4, wherein the object agent (54) is configured to use transmission capacity that would otherwise remain unused by sending data objects to other caches during off-peak data traffic periods.

6.   The network element of claim 5, wherein the data objects sent to the other caches during off-peak data traffic periods are not high priority data objects, and are sent in response to instructions from the cache manager (26).

5   7.   The network element of any preceding claim, wherein the object agent (54) is further configured to send high priority data objects to other caches in the cache domain during peak periods in response to instructions from the cache manager (26).

8.   The network element of claim 6 or 7, wherein high priority data objects are
10   those requested by another mobile terminal through another cache.

9.   The network element of any preceding claim, wherein the object agent (54) is configured to capture each event of a request for a particular data object, and pass this information to the cache manager (26) to assist in identifying which data objects should
15   be distributed between caches.

10.   The network element of any preceding claim, wherein the cache storage unit (224) is included in the network element (22).

20   11.   A cache manager (26) for controlling the distribution of content stored in caches (225, 235, 245) in a cache domain in a packet data network, the cache manager comprising:

        an object monitor (45) for receiving notifications from caches in the cache domain about data objects requested at those caches;
25   an object database (42) for collating and storing object information about the requested data objects, the object information including information about the caches at which the data objects have been requested and frequency with which the requests have been made; and

        an object distributor (46) for identifying objects for distribution on the basis of
30   the object information and instructing the caches in the cache domain to distribute data objects stored in those caches between themselves using transmission capacity of the network that would otherwise be unused.

24

12.   The cache manager of claim 11, wherein the cache manager is a discrete entity and includes a communications system (49) for communicating with elements in the network.

5   13.   The cache manager of claim 12, wherein the cache manager is associated with one of the caches (245) in the cache domain.

14.   The cache manager of claim 11, wherein the functionality of the cache manager is distributed among network elements in the cache domain.

10

15.   The cache manager of any of claims 9 to 12, further comprising:
        a topology monitor (44) for retrieving topology information about the topology of the cache domain; and
        a topology database (41) for storing said topology information;
15   wherein the object distributor is configured to make use of the topology information in the determination of how the objects should be distributed between the caches.

16.   The cache manager of claim 15, wherein the topology information includes data
20   relating to one or more of the following:
        a topology structure of the cache domain;
        bandwidth limitations of links between elements in the topology structure;
        location of caches in the topology structure;
        current loads sustained by the links between elements in the topology structure;
25   and
        transport types for data within the cache domain.

17.   The cache manager of any of claims 11 to 16, configured to control the distribution of the data objects between caches in the cache domain so that the cache
30   domain functions as a large virtual cache.

18.   The cache manager of any of claims 11 to 17, wherein the network is a mobile network.

25

19.   The cache manager of claim 18, wherein the cache domain corresponds to a radio access network (20) in the mobile network.

20.   The cache manager of any of claims 11 to 19, wherein the object distributor (46) is configured to instruct the caches in the cache domain to distribute high priority objects between themselves during peak periods.

21.   A method for optimising the distribution of data objects between caches (225, 235, 245) in a cache domain of a resource limited network, the method comprising:

    receiving user requests for data objects at caches in the cache domain (S62a);

    sending a notification from the cache at which a request is received to a cache manager (26) reporting the user request and identifying the requested data object (S62);

    at the cache manager, collating and storing object information including the request frequency of each requested data object and the locations of the caches at which the requests were received (S63, S64);

    at the cache manager, identifying objects for distribution within the cache domain on the basis of the object information (S65, S66, S67);

    sending instructions from the cache manager to the caches to distribute data objects stored in those caches between themselves (S68, S69); and

    distributing the data objects between the caches (S70) using transmission capacity of the network that would otherwise be unused.

22.   The method of claim 21, wherein functionality of the cache manager (26) is distributed between network elements in the cache domain.

23.   The method of claim 21 or 22, wherein the cache manager (26) further takes into account the topology of the network and/or the priority of the requested data objects and/or the frequency of requests for particular data objects when identifying the data objects for distribution.

24.   The method of claim 21, 22 or 23, wherein the network is a mobile network.

25.   The method of claim 24, wherein the caches are associated with or part of Radio Base Stations.

26

26.   The method of any of claims 21 to 25, wherein the data objects are distributed between the caches (S70) during off-peak periods.

5   27.   The method of any of claims 21 to 26, further comprising:

labelling a data object as a high priority data object if it is not stored in the cache at which the user request for that data objects is received; and

sending instructions, from the cache manager to a cache at which the high priority data object is stored, to send the high priority data object to the cache at which
10   the user request was received during a peak period.

28.   A computer program product comprising code adapted to be executed on a network element (22) associated with a cache storage unit (224) in a resource limited network, the code operable to cause the network element to:
15   inspect packets, from a terminal in the network, passing through the network element;

determine if said packets include a request for a data object;

determine whether the requested data object is stored in the cache storage unit and, if so, send the requested object towards the terminal;
20   receive instructions from a cache manager (26) controlling caches in a cache domain of the network to send data objects to other caches in the cache domain using transmission capacity that would otherwise remain unused; and

send the data objects in accordance with the received instructions.

25   29.   A computer program product comprising code adapted to be executed on a cache manager in a resource limited network, the code operable to cause the cache manager to:

retrieve notifications from caches in a cache domain controlled by the cache manager, the notifications including details of data objects recently requested at those
30   caches;

collate and store object information about the requested data objects, the object information including the location and popularity of requested data objects;

identify objects for distribution on the basis of the object information; and

27

instruct the caches in the network to distribute data objects stored in those caches between themselves using transmission capacity of the network that would otherwise be unused.

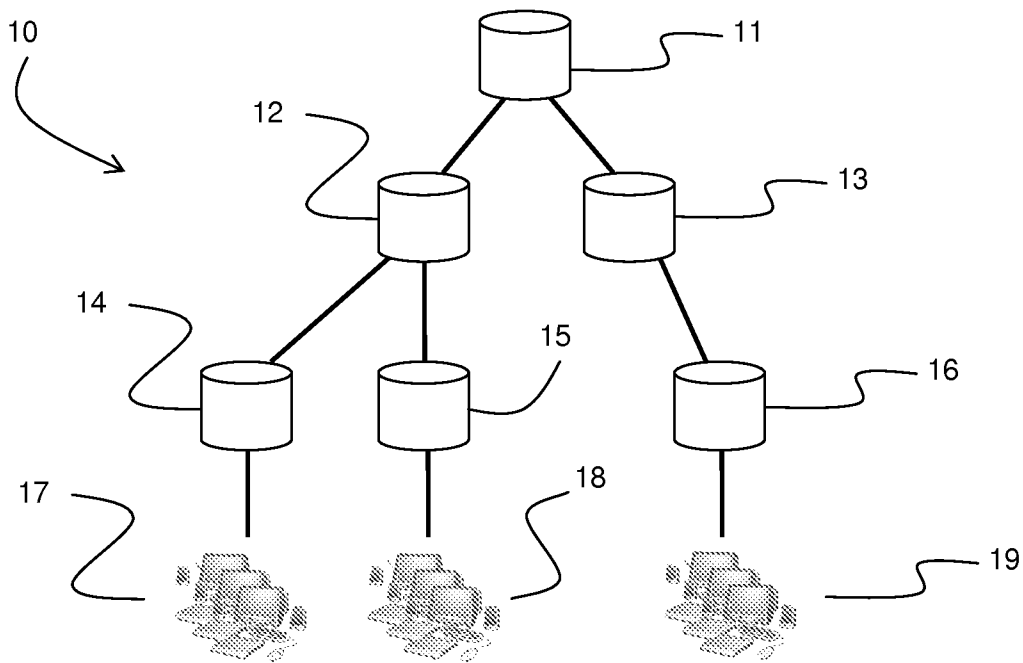5   30.   The computer program product of claim 28 or 29, carried on a carrier medium.
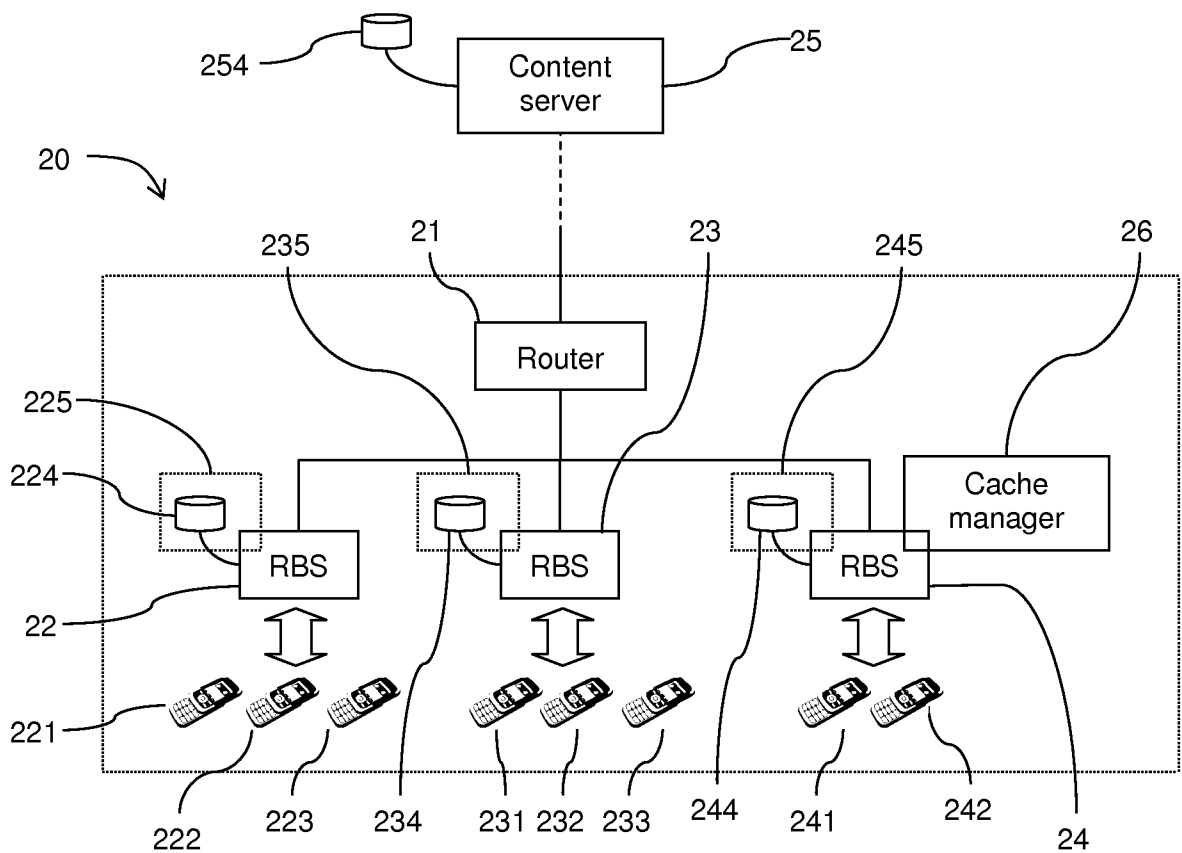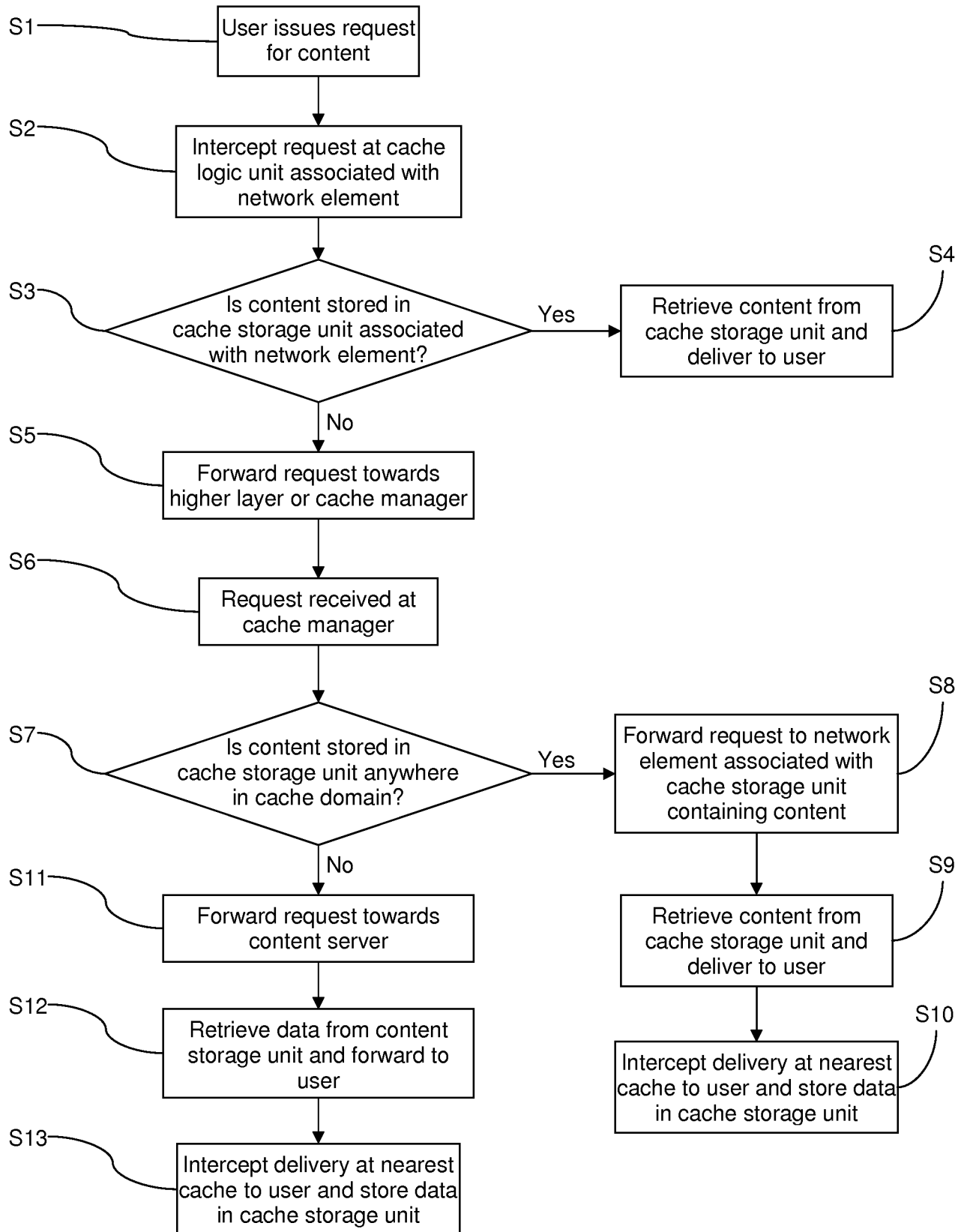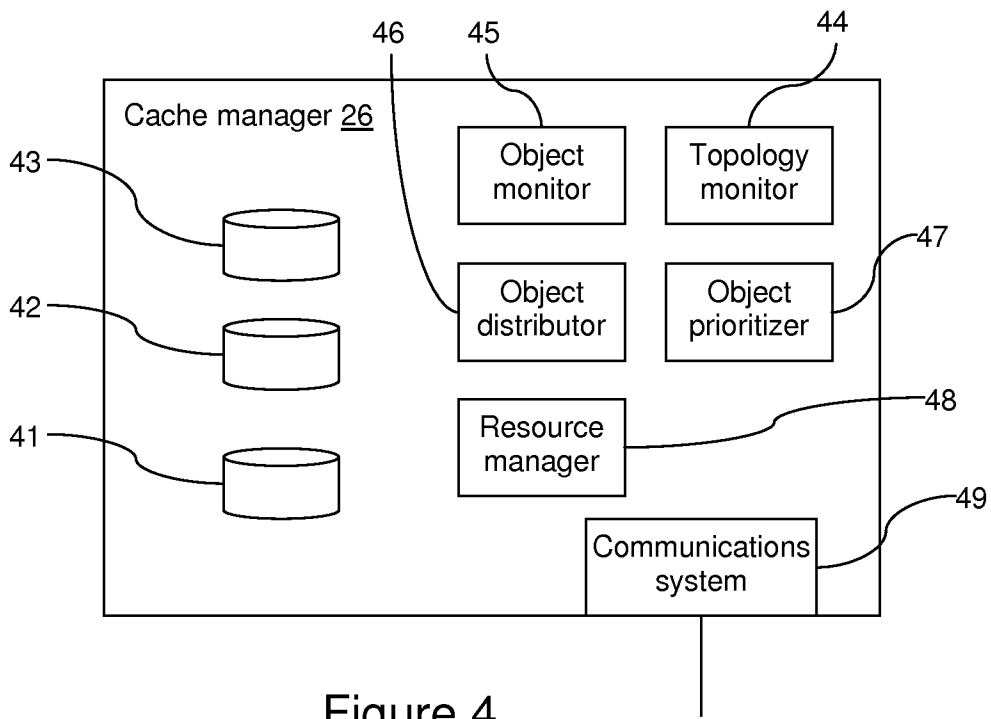
1/5

Figure 1

Figure 2

S1 — User issues request for content

S2 — Intercept request at cache logic unit associated with network element

S3 — Is content stored in cache storage unit associated with network element? — **Yes** → S4 — Retrieve content from cache storage unit and deliver to user

No

S5 — Forward request towards higher layer or cache manager

S6 — Request received at cache manager

S7 — Is content stored in cache storage unit anywhere in cache domain? — **Yes** → S8 — Forward request to network element associated with cache storage unit containing content

No

S8 → S9 — Retrieve content from cache storage unit and deliver to user

S9 → S10 — Intercept delivery at nearest cache to user and store data in cache storage unit

S11 — Forward request towards content server

S12 — Retrieve data from content storage unit and forward to user

S13 — Intercept delivery at nearest cache to user and store data in cache storage unit

Figure 3

Figure 4

Figure 5

4/5
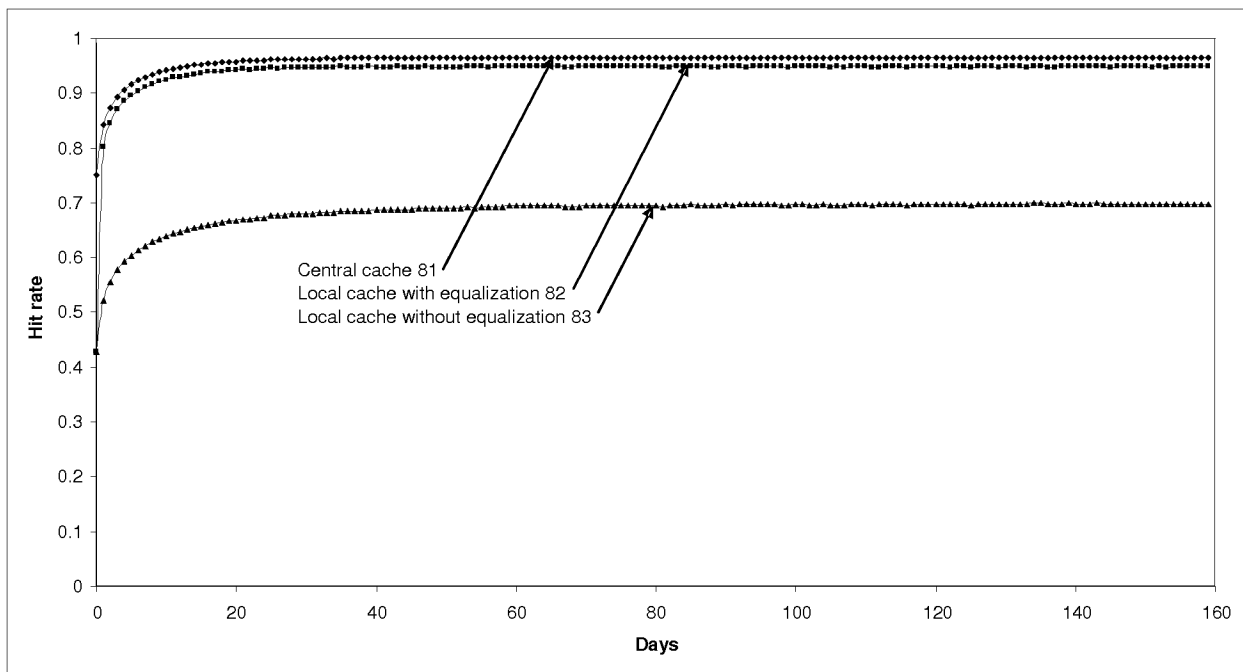

Figure 6

5/5
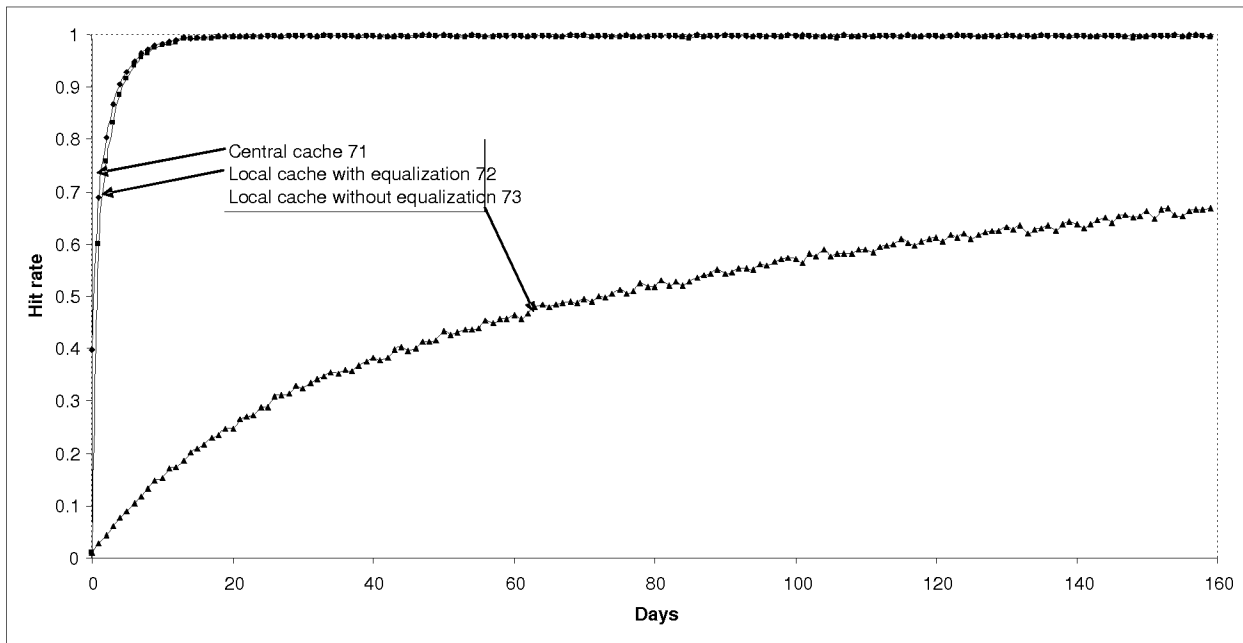
Figure 7

Figure 8

INTERNATIONAL SEARCH REPORT

A. CLASSIFICATION OF SUBJECT MATTER
INV. H04L29/08
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)

H04L

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 practical, search terms used)

EPO-Internal, COMPENDEX, WPI Data

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2004/260769 A1 (YAMAMOTO JUNJI [JP]) 23 December 2004 (2004-12-23) paragraphs [0006] - [0011], [0044] - [0046], [0056] - [0063]; figures 1-3	1-30
X	US 2003/115421 A1 (MCHENRY STEPHEN T [US] ET AL) 19 June 2003 (2003-06-19) paragraphs [0012] - [0018]	1-30
X	EP 2 053 831 A1 (ALCATEL LUCENT [FR]) 29 April 2009 (2009-04-29) paragraphs [0012] - [0022], [0030], [0031]	1-30
A	US 2004/103437 A1 (ALLEGREZZA FRED J [US] ET AL) 27 May 2004 (2004-05-27) paragraphs [0019] - [0025]; figure 1	1-30

-/--

| 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 document 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

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
11 January 2011	19/01/2011

| 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

Biro, Udo Bela |

Form PCT/ISA/210 (second sheet) (April 2005)

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT		
Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 2009/291696 A1 (CORTES MAURICIO [US] ET AL) 26 November 2009 (2009-11-26) paragraphs [0007] – [0010]; figure 4	1-30

Patent document cited in search report		Publication date	Patent family member(s)		Publication date
US 2004260769	A1	23-12-2004	JP	2005010970 A	13-01-2005
US 2003115421	A1	19-06-2003	NONE		
EP 2053831	A1	29-04-2009	CN	101431530 A	13-05-2009
			WO	2009052963 A1	30-04-2009
			KR	20100084179 A	23-07-2010
			US	2009119455 A1	07-05-2009
US 2004103437	A1	27-05-2004	AU	2003287725 A1	18-06-2004
			CA	2532424 A1	10-06-2004
			EP	1570671 A2	07-09-2005
			WO	2004049717 A2	10-06-2004
US 2009291696	A1	26-11-2009	WO	2009142685 A1	26-11-2009