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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
16 August 2012 (16.08.2012)

WIPO | PCT

(10) International Publication Number
WO 2012/107341 A2

(51) International Patent Classification:
G06F 17/30 (2006.01)

(21) International Application Number:
PCT/EP2012/051732

(22) International Filing Date:
2 February 2012 (02.02.2012)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
11290076.6 7 February 2011 (07.02.2011) EP

(71) Applicant *(for all designated States except US)*: **ALCATEL LUCENT** [FR/FR]; 3, avenue Octave Gréard, F-75007 Paris (FR).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: **DE VLEESCHAUWER, Bart** [BE/BE]; Begijnhoflaan 43, B-9200 Dendermonde (BE). **HUYSEGEMS, Rafaël** [BE/BE]; Weverstraat 11 bis, B-2800 Walem (Mechelen) (BE). **WU, Tingyao** [CN/BE]; Celestijnenlaan 13/0302, B-3001 Leuven (BE).

(74) Agent: **ALU ANTW PATENT ATTORNEYS (No 365)**; Copernicuslaan 50, B-2018 Antwerp (BE).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

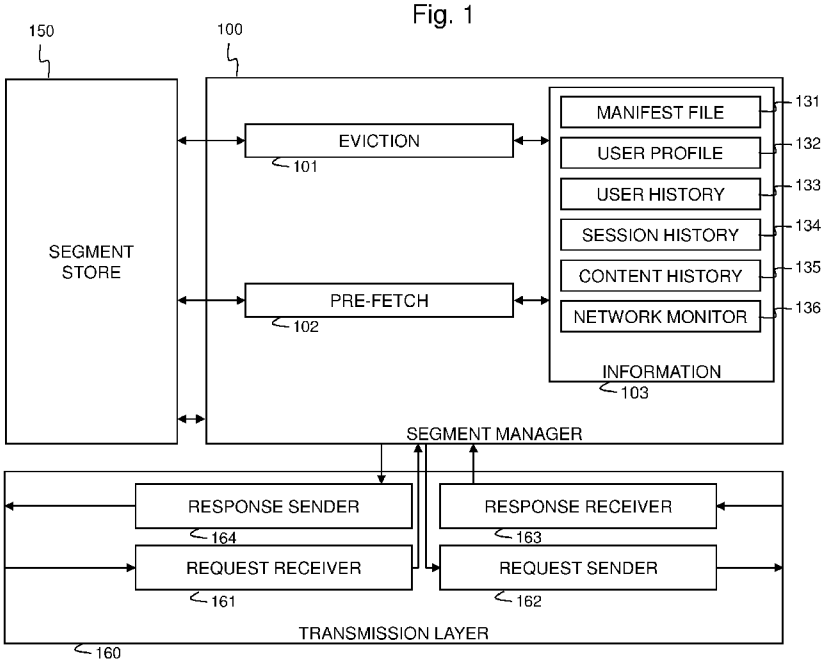
(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

— *without international search report and to be republished upon receipt of that report (Rule 48.2(g))*

(54) **Title**: A CACHE MANAGER FOR SEGMENTED MULTIMEDIA AND CORRESPONDING METHOD FOR CACHE MANAGEMENT

Fig. 1



(57) **Abstract**: The invention concerns a cache manager (100) for managing the intermediate caching of segmented multimedia items. Multiple versions are available of each multimedia item, each version representing the multimedia item with a different quality. The cache manager (100) comprises control means (101, 102) to control the pre-fetching and eviction of segments. The control means (101, 102) are at least responsive to temporal and quality related inter-segment relationships (131).

WO 2012/107341 A2

A CACHE MANAGER FOR SEGMENTED MULTIMEDIA AND CORRESPONDING METHOD FOR CACHE MANAGEMENT

Field of the Invention

[01] The present invention generally relates to caching of segmented multimedia items that are available in multiple qualities, e.g. HTTP Adaptive Streaming (HAS) video segments that are encoded with different resolutions in different versions. The invention in particular attempts to improve the cache hit rate of rather small, intermediate caches in between the multimedia server and the client requesting the multimedia segments.

Background of the Invention

[02] Existing HyperText Transfer Protocol (HTTP) caches treat segmented video in a manner similar to HTTP web content. Typical algorithms that are used by the cache manager for eviction of segments from the cache memory or for pre-fetching segments from the content server for storage in the cache memory are based on least-recently-used (LRU) or least-frequently-used (LFU) mechanisms.

[03] The main drawback of LRU or LFU based cache management when applied to segmented multimedia that is available in multiple qualities, resides in the fact that temporal dependencies and quality-based dependencies between the segments are not exploited, as a result of which the cache hit rate is not optimized.

[04] European Patent Application EP 0 702 491 entitled "Video Optimized Media Streamer with Cache Management" describes an improved video cache that takes into account temporal dependencies between video segments for scheduling transfers between mass storage and a cache buffer. Thus, the nature of video streams with temporally ordered segments is used in pre-fetching video segments in order to improve the cache efficiency.

[05] EP 0 702 491 however does not consider the availability of multiple qualities of a single video item and the possibility for the client to dynamically switch between

these qualities depending for instance on the availability of resources. As a result, the cache algorithm disclosed in EP 0 702 491 does not optimize the cache hit rate when multimedia items can be requested in different qualities and the requested quality may adaptively change during consumption of the multimedia like for instance with HTTP Adaptive Streaming (HAS) of video.

[06] In particular for intermediate cache nodes with limited capacity, e.g. HTTP caches in access nodes, improvements in the caching algorithms may result in significant increase of the cache hit rates and consequently of the benefit/cost ratio when deploying such caches.

[07] It is therefore an objective of the present invention to disclose a cache manager and a corresponding cache management method that resolves the above mentioned shortcomings of existing solutions. More particularly, it is an objective to disclose a cache manager and corresponding cache management method with improved cache hit rate for segmented multimedia available in multiple qualities.

Summary of the Invention

[08] According to the present invention, the above defined objective is realized by a cache manager for managing the intermediate caching of segments of multimedia items of which multiple versions are available, each version representing a multimedia item with a different quality, the cache manager comprising control means to control pre-fetching of segments of the multimedia items into a cache memory and eviction of segments of the multimedia items from the cache memory, these control means being responsive to information indicative for temporal and quality related inter-segment relationships between segments of the multiple versions of the multimedia items. Such cache manager is defined by claim 1.

[09] Thus, the basic idea underlying the current invention is to exploit specific knowledge on inter-segment relationships, both temporal and quality-related, to improve the cache eviction and pre-fetching efficiency. Indeed, when a user is for instance watching a video at a certain point in time by downloading a segment with quality level Q and segment identifier ID, the likelihood that the user will be interested

in viewing segment ID+1 with a quality level that is close to Q, is relatively high. More generally, segments with IDs that are subsequent to the ID of the multimedia segment that is currently consumed are more likely to be requested by the user than segments that are further away in time from the currently consumed segment. Moreover, the quality level of future requested multimedia segments is more likely to be close to the currently consumed quality level. The combined temporal and quality-dependent relationships between segments therefore shall enable to pre-fetch multimedia segments that are more likely to be requested in the near future, and to remove segments that are less likely to be requested in the near future. Obviously, this will result in improved cache hit rates, i.e. higher availability of requested segments in the cache.

[10] It is noticed that the temporal and quality related inter-segment relationships may be combined with traditional criteria such as popularity, time and date of last usage, etc., in order to optimize the cache hit rate. As will be explained below, the temporal and quality related inter-segment relationship knowledge may further be supplemented with less traditional criteria such as user profile information, history information, and network load information to further optimize the cache hit rate. Additionally, the system could use information that is bound to the content itself, e.g. a sports game or talk show, but also information indicative for typical jump-points in time, e.g. the start of a commercial, the end of an interview, etc.

[11] Optionally, as defined by claim 2, the cache manager according to the current invention comprises means to extract the temporal and quality related inter-segment relationship information from one or more multimedia presentation description file, e.g. the manifest files related to the multimedia items.

[12] Indeed, the relationship between segments when using adaptive streaming technology is typically described in the so called multimedia presentation description (MPD) file, e.g. the manifest file. This manifest or MPD file could therefore be exploited by the cache manager according to the present invention to establish knowledge on the relationship between segments in an HTTP adaptive streaming session.

[13] Further optionally, as defined by claim 3, the control means may be adapted to be responsive at start-up to popularity information of the multimedia items.

[14] Indeed, at the start of a movie for instance, little information will be available on the active sessions and the inter-segment relationships. Therefore, general popularity information could be taken into account at this point in time to control the pre-fetching and eviction of segments.

[15] According to a further optional aspect defined by claim 4, the means to control pre-fetching and eviction of segments may be adapted to be responsive to one or more of the following:

- user profile information;
- segment popularity information;
- user history information:
- session history information;
- content history information;
- content information supplied by a content owner/producer;
- network load between a multimedia server and said cache memory;
- network load between said cache memory and end user equipment;
- client side algorithm information.

[16] Thus, additional information can be taken into account by the segment pre-fetching and eviction algorithm. The extra information can be a user profile describing for instance typical bandwidth available for a given user, segment popularity information indicative for the amount of times a given segment has been requested, user history describing for instance the zapping behaviour of a given user, content history describing for instance the typical points in time where the user stops consuming a given multimedia item, session history describing for instance the typical quality level(s) that are requested in a given session, the load or available resources in the part of the network between the multimedia server and the cache and/or in the part of the network between the cache and client, and knowledge on specific client side algorithmic strategies. The latter may for instance include information on the client side buffering. When a Smoothstreaming client for instance notices that its buffer fill level for viewing video is lower than a threshold value, it may

switch automatically to the lowest possible quality in order to avoid interruptions. Keeping the corresponding low quality segments in cache may therefore be a valuable strategy that optimizes the cache hit rate for such clients. Knowledge on the load or available resources in the network part between server and cache may be exploited to determine how aggressively segments will be pre-fetched.

[17] In order to control the pre-fetching of segments and the eviction of segments, the cache manager according to the current invention hence may not only consider temporal and quality dependent inter-segment relationships, but may also exploit any combination of the aforementioned information types in order to further optimize the cache hit rate and benefit/cost ratio of multimedia cache deployment.

[18] As is further specified by claim 5, the cache manager according to the current invention may further comprise means for machine learning the temporal and quality related inter-segment relationships.

[19] Thus, the software/hardware that is used to determine which segments to remove from the cache and which segments to pre-fetch from the mass store, could be based on a machine learning algorithm that dynamically adapts itself to the environment wherein it is deployed. An advantage thereof is that updates to existing clients or the behaviour of new clients in the network can be learned. This way, excessive software upgrades for intermediate caching nodes are avoided.

[20] In an embodiment of the cache manager according the invention, defined by claim 6, the means to control pre-fetching and eviction of segments may be further adapted to maintain and/or assign values to segments stored in the cache memory, each value representing a likelihood that a corresponding segment of a version of a multimedia item will be requested, considering the temporal and quality related inter-segment relationships.

[21] Indeed, a simple way to implement the current invention consists in maintaining values for each segment in each version of the multimedia item. These values represent the likelihood that a segment will be requested and consequently also the interest for keeping a segment in cache. Each value is determined in

function of the temporal relationship and quality relationship between the segment under consideration and the recently requested segments of the same multimedia item. Eventual additional information may be exploited to determine/fine-tune the values. The values are updated each time one or more segments are requested.

[22] According to a further advantageous aspect of the invention, defined by claim 7, the cache manager may comprise re-direction means adapted to instruct re-direction of a multimedia session from a first cache node to a second cache node, the re-direction means being responsive at least to network topology information mapping clients to one or more caching nodes.

[23] Thus, the information that is used to optimize the pre-fetching and segment eviction decisions according to the current invention may further be exploited to re-direct a multimedia session, e.g. a video streaming session by a client, to another cache node in the network. This may be advantageous for instance in a situation where the same multimedia session is already delivered by another cache node to another client, eventually slightly delayed in time. When a cache node is handling a multimedia session and watches the segment with identifier ID and quality level Q, the likelihood that in the session a request for the segment with identifier ID+1 and quality level Q will also be issued will be high. Thus it may be advantageous to redirect this session request to a cache node that handles a session that is already handling a session for the same content and that is watching the segment identifier ID+1 and quality level Q.

[24] For determining which caching node should handle a session, typically topology information is used in the prior art and no dedicated information on the HAS session is taken into account. As a result, sessions are traditionally redirected to a nearby caching node. In order to be able to efficiently re-direct multimedia sessions, the cache manager according to the invention could use network topology information, e.g. a list mapping clients to one or several cache nodes that are closely located. Additional information like the actual load of the cache nodes, i.e. the cache node receiving the request and the cache node whereto the session will be re-directed, as well as particular information on the sessions handled by the different caching nodes, including time and quality information of these sessions, will improve

the efficiency of the re-directing algorithm. Additional information may include prior knowledge on the expected popularity of the session. In order to share the network topology information and eventual additional information on the load and sessions handled by the different caching nodes, a protocol enabling to communicate this information between the caching nodes or a protocol enabling to distribute this information through mediation of a central platform, must be implemented.

[25] It is further noticed that the re-direction of a multi-media session according to the current invention may be controlled at segment level, i.e. one or more segments of a session are delivered through another caching node, or may be controlled at session level, i.e. the entire multimedia session is re-directed to another caching node.

[26] In addition to the cache manager defined by claim 1, the current invention also relates to a corresponding method for managing the intermediate caching of segments of multimedia items of which multiple versions are available, each version representing a multimedia item with a different quality, the method comprising controlling pre-fetching of segments of the multimedia items into a cache memory and eviction of segments of the multimedia items from the cache memory in response to information indicative for temporal and quality related inter-segment relationships between segments of the multiple versions of the multimedia items. This corresponding method is defined by claim 8.

[27] As is indicated by claim 9, the method according to the invention optionally may comprise re-directing a multimedia session from a first cache node to a second cache node thereby using at least network topology information mapping clients to one or more caching nodes.

Brief Description of the Drawings

[28] Fig. 1 is a functional block scheme of an embodiment of the cache manager 100 according to the present invention;

[29] Fig. 2 illustrates operation of a second embodiment of the cache manager according to the present invention when a single session wherein a multimedia item available in multiple qualities Q is downloaded, is active;

[30] Fig. 3 illustrates operation of a third embodiment of the cache manager according to the present invention when two sessions wherein a multimedia item available in multiple qualities Q is downloaded, are active; and

[31] Fig. 4 illustrates a functional block scheme of an embodiment of the cache manager according to the present invention with re-direct component.

Detailed Description of Embodiment(s)

[32] Fig. 1 depicts a segment store or cache 150, a segment manager 100, and the underlying transmission layer 160.

[33] The segment store 150 is a simple memory component that contains a number of multimedia segments. Segments can be requested from it and injected in it. Additionally, the memory component can be monitored and will return information on the segments that are available.

[34] The segment manager 100 represents a first embodiment of the cache manager according to the present invention. The segment manager 100 is responsible for answering requests for segments from clients, contains a pre-fetch component 102 for pre-fetching and inserting segments in the segment store 150, and an eviction component 101 for eviction or removal of segments from the segment store 150. Additionally, the segment manager 100 is also responsible for requesting segments from a remote entity, a mass multimedia store, when these segments are not present in the segment store 150 and will have the possibility to store these segments in the segment store 150 when it decides to do so. At last, the segment manager 150 will also store information on the requests it sees into various information components 103 that are responsible for maintaining relevant data on the multimedia traffic. These information components 103 may be internal or external to the segment manager 100.

[35] In the information component 103 contains a variety of components that essentially capture global and local information to aid the segment eviction and pre-fetch algorithms in making their decisions. The examples shown in Fig. 1 include the manifest file(s) 131 with information on how the segments of a given session are linked temporally and quality-wise, user profile information 132, user history information 133 on a particular user, session history information 134 on a specific session, content history information 135 on how specific content is typically consumed, and information on the monitored network load 136. However, this is a non-exhaustive list and other additional information could be used in the eviction and pre-fetch algorithms of alternative embodiments of the current invention.

[36] The eviction component 101 is responsible for determining which segments should be removed from the segment cache 150. To do this, the eviction component 101 makes use of the various types of information 131, 132, 133, 134, 135 and 136 that are available in the information component 103.

[37] The pre-fetch component 102 is responsible for determining which segments to pre-fetch from the multimedia mass store. To do this, the pre-fetch component 102 also makes use of the various types of information 131, 132, 133, 134, 135 and 136 that are available in the information component 103.

[38] The request receiver 161, the request sender 162, the response receiver 163, and the response sender 164 constitute the message interface to the external world. Basically, these provide the transmission layer 160 for communicating with other entities in the network, like clients and mass stores.

[39] It is noticed that the invention can be used in a system where a combination of HTTP Adaptive Streaming (HAS) with Scalable Video Coding (SVC) is used. This is for instance the case in the MPEG DASH (Dynamic Adaptive Streaming of HTTP) standard specification. SVC encodes different versions of a video file in hierarchically structured layers. The base layer represents a low quality version of the video file. Enhancement layers are built on the base layer to generate higher quality versions of the video file. In such a system, the benefit brought by the cache manager according

to the present invention is even bigger because lower layers, e.g. the base layer, are always required for a segment. Whereas higher, enhancement layers may be removed from the cache, the base layer may be kept since it is needed for all versions of the video file.

[40] Fig. 2 and Fig. 3 illustrate the operation of a second embodiment of the cache manager operating according to the principles of the invention. In this second embodiment a value is maintained for each segment of a video file (horizontal axis T) at each quality level (vertical axis Q). Fig. 2 illustrates a situation where a single user downloads at a given point in time a particular segment, i.e. the shaded segment in Fig. 2. The horizontal dashed line shows the consecutive segments of the video file that the user is assumed to download. At the point in time where the user downloads the shaded segment, the cache manager according to the invention assigns values to other segments. These values depend on temporal and quality related inter-segment relationships, and represent a measure for the likeliness that a segment will be requested by the user. A simple strategy to assign such values to the segments could be as follows:

- the first three segments of the same quality level, following the segment that is currently being downloaded are increased by 3.
- the subsequent three segments at the same quality level are increased by 2;
- the subsequent three segments at the same quality level are increased by 1;
- the segments with a quality level difference of 1, and a segment identifier that differs at most 5 from the currently downloaded segment identifier get an increase of 2;
- the subsequent 4 segments at a quality level difference of 1 are increased by 1; and
- the segments with a quality level difference of 2, and a segment identifier that differs at most 5 from the currently downloaded segment identifier get an increase of 1.

[41] If it is assumed that the values of all segments were 0 before downloading the shaded segment, Fig. 2 represents the values of the segments at the different quality levels after downloading the shaded segment.

[42] Fig. 3 illustrates the same embodiment in a situation where a first user downloads a first shaded segment, and a second user downloads a second shaded segment. The segment downloaded by the second user constitutes a later segment, and the second user downloads this later segment with a higher quality level than the first user is viewing. When it is again assumed that the values of all segments were 0 before the first and second users were downloading the shaded segments, Fig. 3 represents the values of the segments at the different quality levels after user and user 2 have downloaded their respective segment at their respective desired quality. The dotted lines in Fig. 3 further represent the assumed sequence of segments that will be downloaded by the first and second user.

[43] The above explained assignment of values is repeated for all active sessions and the final value assigned to the segments is normalized. The eviction component and pre-fetching component of the cache manager according to the invention shall base their decisions for eviction of segments from the cache and pre-fetching of segments into the cache on these values. Segments with a high value are least likely to be removed from the cache by the eviction component. Segments with high values that are not yet in the cache are likely to be pre-fetched from mass store by the pre-fetching component.

[44] In an alternative embodiment that also maintains importance values for segments, the cache manager assigns a value to specific segments that are in the cache. It is assumed that all the segments have the same duration. When a segment is removed from the cache, the eviction component in the cache manager selects the one with the lowest value. In the algorithm, S(m) denotes the set of all segments of a movie m. Each segment is identified by an integer value s representing the order in which the segments are typically downloaded and Q(s) represents the quality corresponding to a segment s. The cache importance of a segment is the value that represents to what extent it is interesting to keep this segment in the cache. The scheme to assign these values in the third embodiment is as follows:

```
For all the active users u that are watching a movie m {
For all i belonging to S(m), where i>s {
i_importance = i_importance + f(i, s);
```

}

}

Herein:

m represents the movie being watched by user u;

s represents the identifier of the segment of movie m being watched by user u;

q represents the quality of segment s;

i represents an integer index;

i_importance represents the importance value of the segment with identifier i; and

f(i,s) is a function used to calculate the increase in importance for a segment i when it is known that user u is currently watching segment s of the same movie m.

[45] An example function f(i,s) could be the following:

$$f(i,s) = 1 + \frac{|100 - (i - s)|}{1 + |Q(i) - Q(s)|} + \min\left(20, \frac{(i - s)|Q(i) - Q(s)|}{10}\right)$$

This function f(i,s) assigns a higher importance to segments that are close in time to the currently watched segments. This function f(i,s) also allows for a varying quality by letting the estimation of the expected quality to be higher for segments that are near in time and assuming that segments that are further away in time would have a much more equal importance for the quality versions that are available.

[46] In this third embodiment it is preferred to reserve part of the cache memory for the initial segments belonging to the video streams. This would ensure that segments at the start of a movie, which would have a lower likelihood to be in the cache when the previous algorithm is used, are kept available.

[47] In a fourth embodiment of the cache manager according to the present invention, a learning mechanism is used to learn the likelihoods of requesting segments close to the currently watched segments, either in time or in quality. One plausible technique consists in training a Markov chain for different historical requesting paths. Given a training set, the probability of requesting segment Q(s+1) can be estimated as P(Q(s+1)|Q(s),Q(s-1),…, Q(1)), which assumes that the

requesting quality by HTTP adaptive streaming in the near future (s+1) does not only depend on the current quality, but also rely on the historical requesting qualities.

[48] To simplify the computation and avoid the data sparseness, P(Q(s+1)|Q(s), Q(s-1), ..., Q(1)) can be approximated as P(Q(s+1)|Q(s), Q(s-1), ..., Q(s-n)), where n is denoted as the length of the retrieved quality path. The parameter n needs to be optimized. Similarly, it is possible to statistically estimate P(Q(s+2),Q(s+1)| Q(s), Q(s-1), ..., Q(s-n)), ..., P(Q(s+t),..., Q(s+1)|Q(s), Q(s-1), ..., Q(s-n)) from the training set.

[49] In test, given the requesting path Q(s), Q(s-1), ..., Q(s-n), the maximum likelihood criterion is used to select the least important segment based on the Markov chain obtained from the training set. An incremental learning can also be incorporated to adaptively update the parameters in the Markov chain in order to fit the new network conditions. This learning could be done for each video individually, for an individual client, but also on aggregates.

[50] Fig. 4 depicts a block diagram of a cache node wherein the cache manager 100 has a redirection component 401. The reference numbers used in Fig. 1 are re-used in Fig. 4 to refer to functional blocks that implement the same functionality. The cache node illustrated by Fig. 4 further contains information 403 on the network topology and information 402 on other cache nodes to which session requests can be redirected. A monitor daemon component 404 takes care of sending monitor information to other cache nodes and retrieving monitor information from other cache nodes. An example algorithm would work as follows: the cache node depicted in Fig. 4 determines the cache nodes that are closer to the client issuing the request. It then orders these nodes based on their proximity to the client and starts with the cache node closest to the client. If this node, below denoted as c, handles sessions for the same content, the cache node of Fig. 4 determines whether this node would be a valid candidate for redirection. This could be done in the following way. Suppose the request S is for a segment with temporal identifier ID and quality identifier Q, go through all of the sessions for the same content. The memory available in the candidate cache node is X.

c = candidate cache node

Decision_value = 0.0

```
For all sessions s, handled by c, for the same content do
    If identifier (s) < ID + X do
        If quality(s) = quality (S) do
            Decision_value = 1.0
        done
        else do
            Decision _value_+ = 1/(10*|(quality(s)-quality (S)|)
        done
    done
done
If Decision_value > threshold_value do
    Redirect (S, c)
done
```

[51] The initial cache node could follow this procedure for all the candidate cache nodes and when it would find a node c where the redirect decision is reached, redirect the session. If no suitable candidate node c would be found, it would handle the request itself.

[52] Although the present invention has been illustrated by reference to specific embodiments, it will be apparent to those skilled in the art that the invention is not limited to the details of the foregoing illustrative embodiments, and that the present invention may be embodied with various changes and modifications without departing from the scope thereof. The present embodiments are therefore to be considered in all respects as illustrative and not restrictive, the scope of the invention being indicated by the appended claims rather than by the foregoing description, and all changes which come within the meaning and range of equivalency of the claims are therefore intended to be embraced therein. In other words, it is contemplated to cover any and all modifications, variations or equivalents that fall within the scope of the basic underlying principles and whose essential attributes are claimed in this patent application. It will furthermore be understood by the reader of this patent application that the words "comprising" or "comprise" do not exclude other elements or steps, that the words "a" or "an" do not exclude a plurality, and that a single element, such as a computer system, a processor, or another integrated unit may

fulfil the functions of several means recited in the claims. Any reference signs in the claims shall not be construed as limiting the respective claims concerned. The terms "first", "second", third", "a", "b", "c", and the like, when used in the description or in the claims are introduced to distinguish between similar elements or steps and are not necessarily describing a sequential or chronological order. Similarly, the terms "top", "bottom", "over", "under", and the like are introduced for descriptive purposes and not necessarily to denote relative positions. It is to be understood that the terms so used are interchangeable under appropriate circumstances and embodiments of the invention are capable of operating according to the present invention in other sequences, or in orientations different from the one(s) described or illustrated above.

CLAIMS

1. A cache manager (100) for managing the intermediate caching of segments of multimedia items of which multiple versions are available, each version representing a multimedia item with a different quality,

CHARACTERIZED IN THAT said cache manager (100) comprises control means (101, 102) to control pre-fetching of segments of said multimedia items into a cache memory (150) and eviction of segments of said multimedia items from said cache memory (150), said control means (101, 102) being responsive to information (131) indicative for temporal and quality related inter-segment relationships between segments of said multiple versions of said multimedia items.

2. A cache manager (100) according to claim 1,

comprising means to extract said information (131) from one or more multimedia presentation description file related to said multimedia items.

3. A cache manager (100) according to claim 1,

wherein said control means (101, 102) are further adapted to be responsive at start-up to popularity information of said multimedia items.

4. A cache manager (100) according to claim 1,

wherein said control means are further adapted to be responsive to one or more of the following:

- user profile information (132);
- segment popularity information;
- user history information (133):
- session history information (134);
- content history information (135);
- content information supplied by a content owner/producer;
- network load between a multimedia server and said cache memory (136);
- network load between said cache memory and end user equipment (136);
- client side algorithm information.

5. A cache manager (100) according to claim 1,

further comprising means for machine learning said temporal and quality related inter-segment relationships.

6. A cache manager (100) according to claim 1,

wherein said control means are further adapted to maintain and/or assign values to segments stored in said cache memory (150), each value representing a likelihood that a corresponding segment of a version of a multimedia item will be requested, considering said temporal and quality related inter-segment relationships.

7. A cache manager (100) according to claim 1,

further comprising re-direction means (401) adapted to instruct re-direction of a multimedia session from a first cache node to a second cache node, said re-direction means (401) being responsive at least to network topology information (403) mapping clients to one or more caching nodes.

8. A method for managing the intermediate caching of segments of multimedia items of which multiple versions are available, each version representing a multimedia item with a different quality,

CHARACTERIZED IN THAT said method comprises controlling pre-fetching of segments of said multimedia items into a cache memory (150) and eviction of segments of said multimedia items from said cache memory (150) in response to information (131) indicative for temporal and quality related inter-segment relationships between segments of said multiple versions of said multimedia items.

9. A method according to claim 8,

further comprising re-directing a multimedia session from a first cache node to a second cache node thereby using at least network topology information (403) mapping clients to one or more caching nodes.

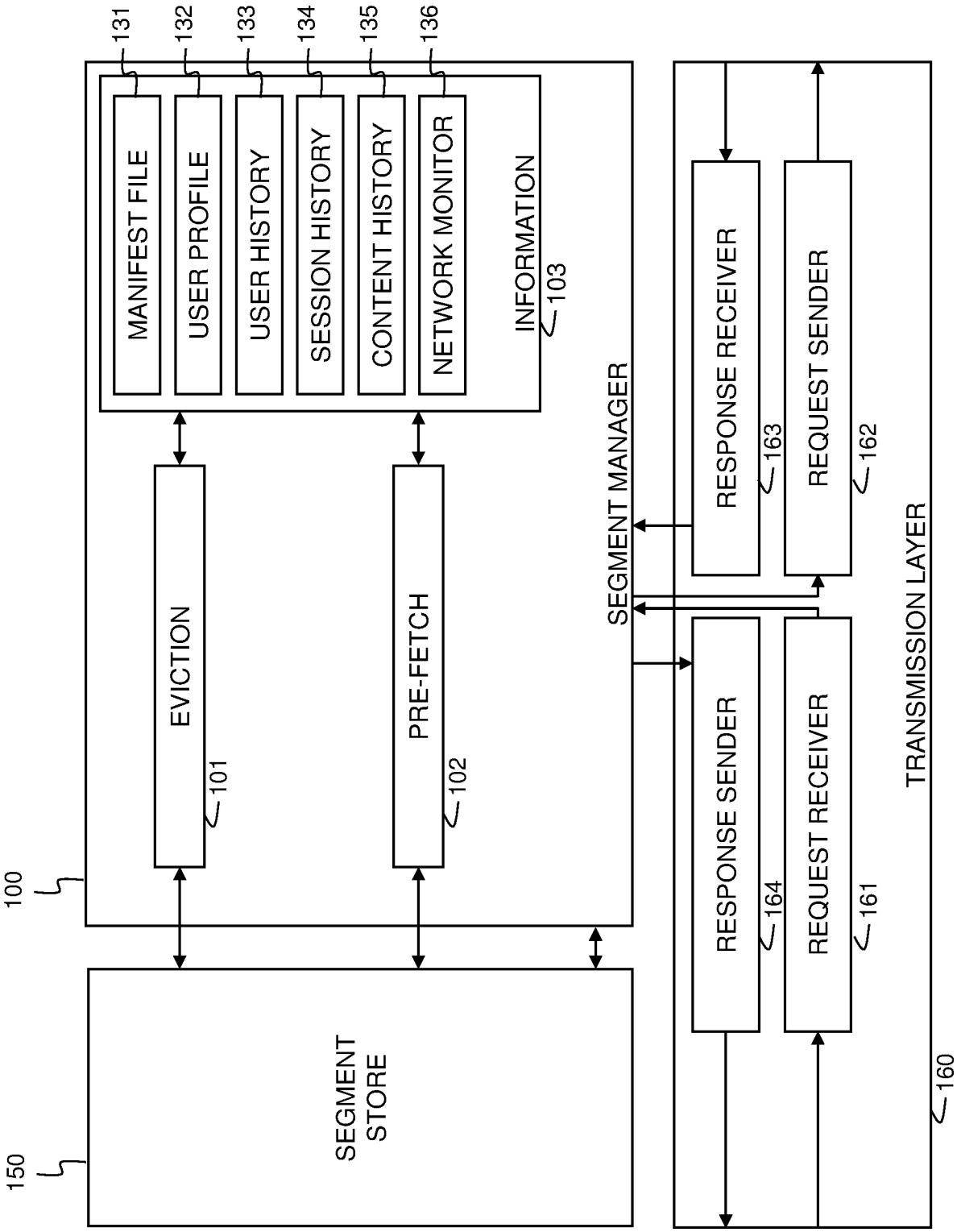

131
132
133
134
135
136
MANIFEST FILE
USER PROFILE
USER HISTORY
SESSION HISTORY
CONTENT HISTORY
NETWORK MONITOR
INFORMATION
103
EVICTION
101
PRE-FETCH
102
SEGMENT MANAGER
RESPONSE RECEIVER
163
REQUEST SENDER
162
RESPONSE SENDER
164
REQUEST RECEIVER
161
TRANSMISSION LAYER
160
100
150
SEGMENT STORE

Fig. 1

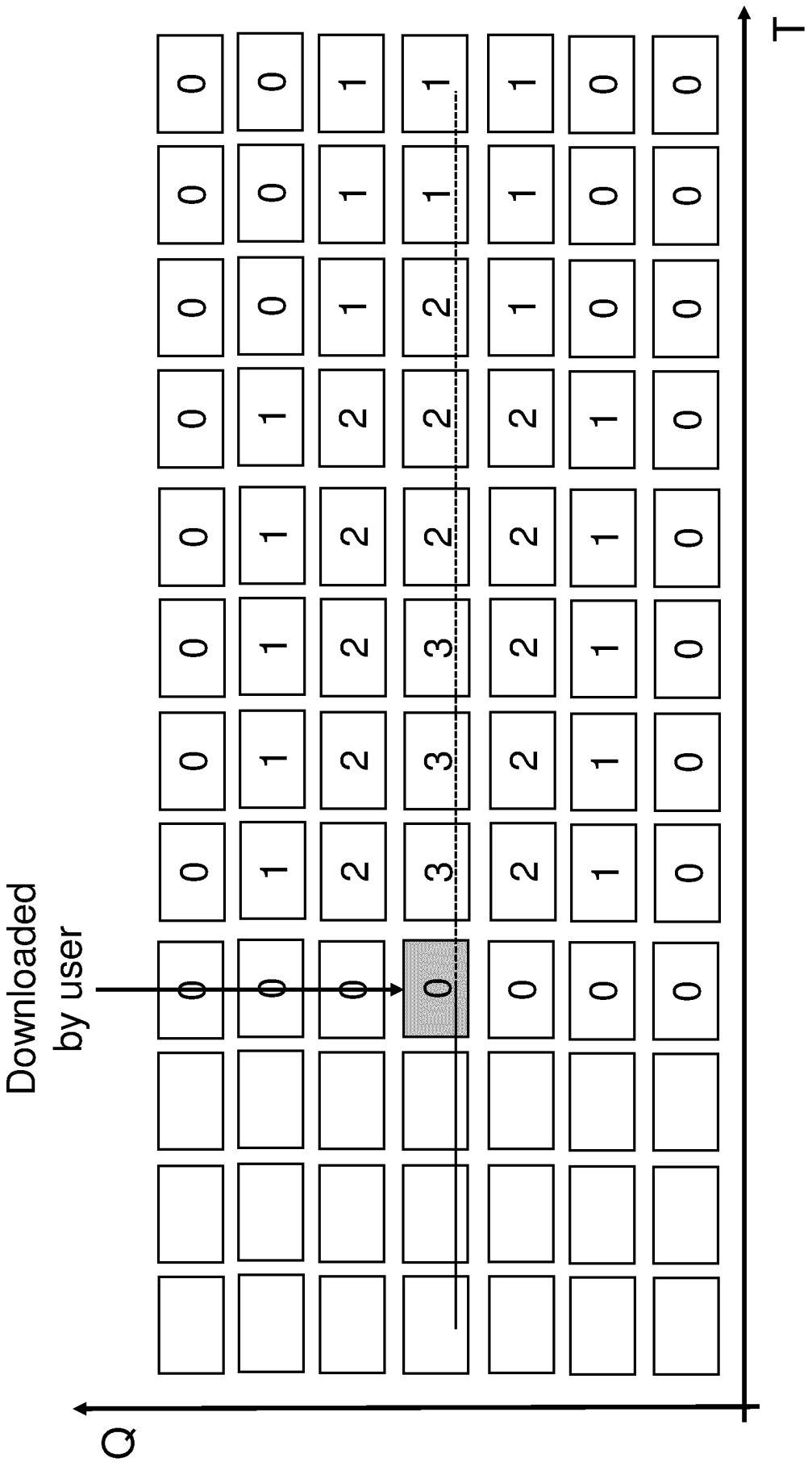


Fig. 2

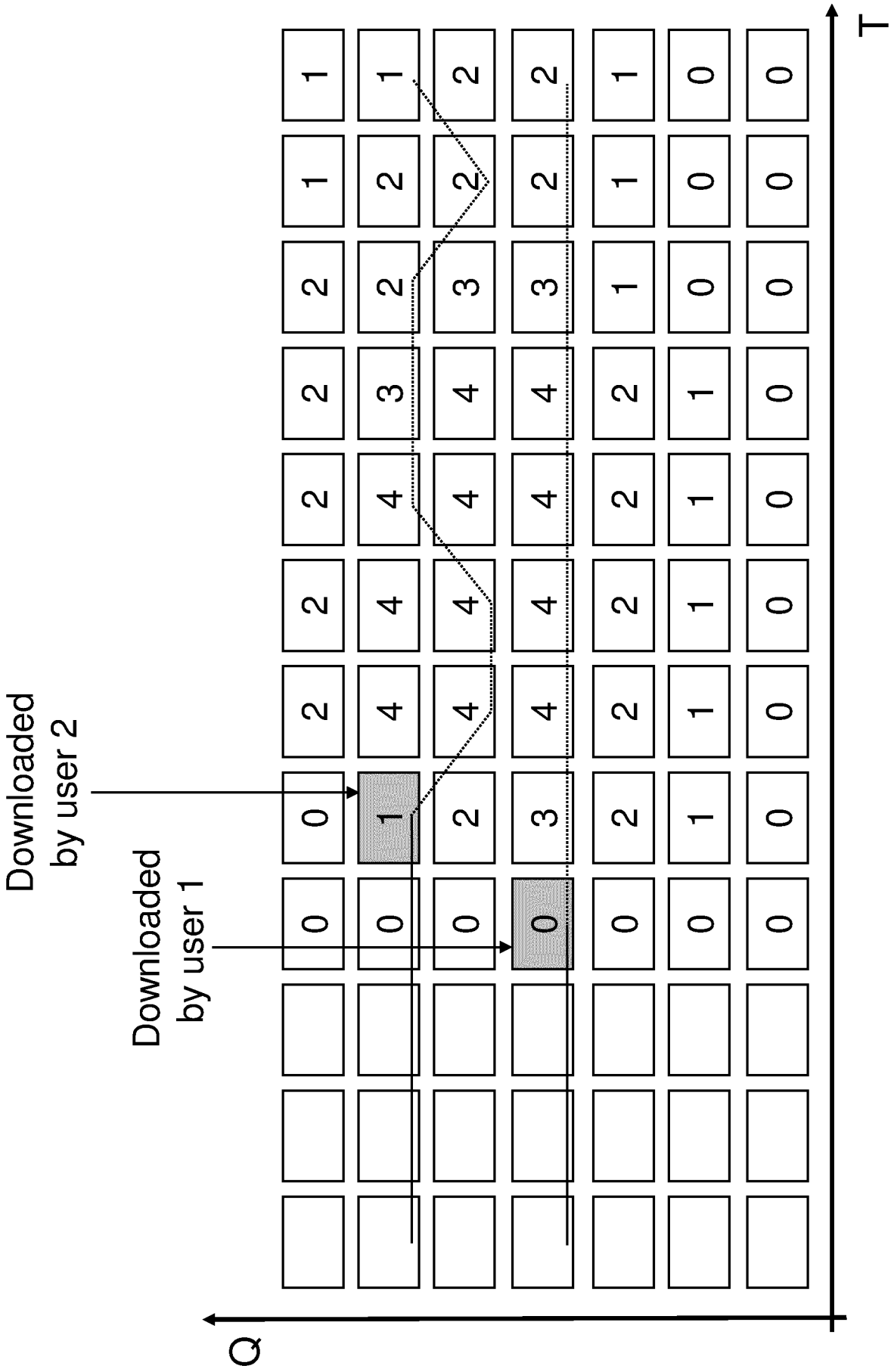


Fig. 3

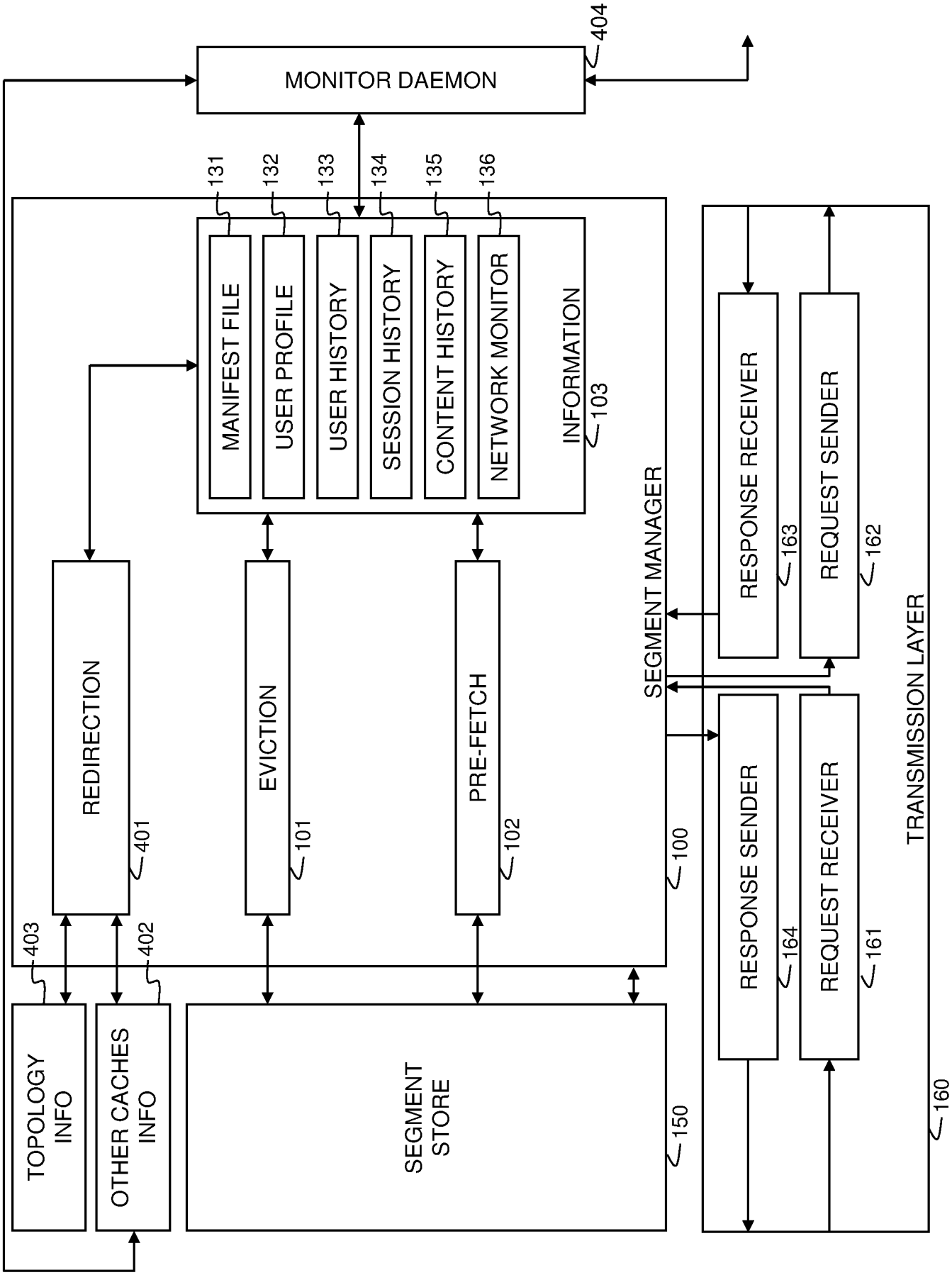

MONITOR DAEMON
404
131
132
133
134
135
136
MANIFEST FILE
USER PROFILE
USER HISTORY
SESSION HISTORY
CONTENT HISTORY
NETWORK MONITOR
INFORMATION
103
SEGMENT MANAGER
REDIRECTION
401
EVICTION
101
PRE-FETCH
102
100
RESPONSE RECEIVER
163
REQUEST SENDER
162
RESPONSE SENDER
164
REQUEST RECEIVER
161
TRANSMISSION LAYER
160
403
402
TOPOLOGY INFO
OTHER CACHES INFO
SEGMENT STORE
150

Fig. 4