Charging for web content pre-fetching in 3G networks[1]

David Larrabeiti[1], Ricardo Romeral[1], Manuel Urueña[1], Arturo Azcorra[1], Pablo Serrano[1]

[1] Universidad Carlos III de Madrid, Av. Universidad 30, Leganés 28670, Spain
{dlarra,rromeral,muruenya,azcorra,pablo}@it.uc3m.es
http://www.it.uc3m.es

Abstract. Web pre-fetching is a technique that tries to improve the QoS perceived by a user when surfing the web. Previous studies show that the cost of an effective hit rate is quite high in terms of bandwidth. This may be the reason why pre-fetching has not been commonly deployed in web proxies. Nevertheless, the situation can change in the context of 3G, where the radio access is a shared scarce resource and the operator may find useful to exchange *fixed-network bandwidth* by *perceived QoS* for subscribed customers. Most importantly, in UMTS it is possible to charge for this service even though pre-fetching is provided by a third party. This paper studies this scenario, identifying the conditions where pre-fetching makes sense, describes the way OSA/Parlay could be used to enable charging, presents a tool developed for this purpose and analyses several issues related to charging for this service.

1 Introduction

Web pre-fetching is a well-known technology that has met a singular market niche in so-called internet boosters. The target of this sort of applications is to increase the effective QoS perceived by the end user by making use of spare access bandwidth to pre-fetch and cache those web pages most probable to be visited by the user in his/her next HTTP request. The average performance gain is driven by the ability of the prediction module to foresee the next link selected by the user. The context of this application is dial-up internet access over a low speed modem (e.g. V.34 or V.90). At first sight a reader may think that these internet boosters could be incorporated in 3G terminals and thus enhance the access bandwidth and virtually remove currently high RTTs (Round-Trip Times) from UMTS terminal to the Internet. However the charging schemes applicable in 3G that rate the volume of carried traffic make this option not realistic.

[1] This work was partially funded by the IST project Opium (Open Platform for Integration of UMTS Middleware) IST-2001-36063 and the Spanish MCYT under project AURAS TIC2001-1650-C02-01. This paper reflects the view of the authors and not necessarily the view of the referred projects.

An alternative to this is the deployment of a proxy cache in the network that features pre-fetching (Fig.1). This way, over-fetching does not happen on the limited-bandwidth radio segment and, consequently, this service can actually be delivered in a cost-effective way by the network operator and even exploited by a third party as discussed in this paper. Therefore pre-fetching can be used as a way to provide differentiated application-specific QoS to a number of users on a subscription basis.

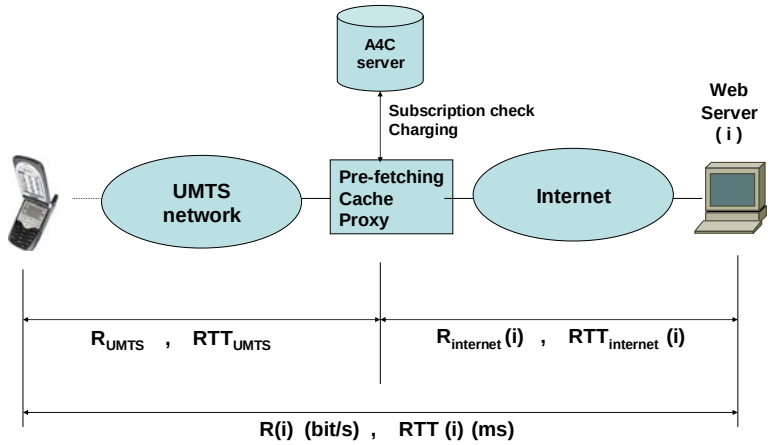

Fig. 1. Network-based pre-fetching in UMTS

The purpose of this paper is to show that this is possible and to identify key issues in its deployment, including the study of charging strategies for this specific QoS provisioning mechanism. On this sense, this paper is organized as follows. Section 2 provides a quick overview of pre-fetching and its practical limits. Once described the nature of the technique to be exploited, section 3 discusses how this service could be exploited externally via OSA/Parlay. Section 4 describes a test scenario deployed in a real UMTS network, that uses a pre-fetcher and charges for its usage. Finally a number of conclusions are drawn from this experience in section 5.

2 Web pre-fetching

The research carried out in techniques to optimize web caching is very extensive. Therefore we shall try to cite only those works that bring key ideas required to understand the process, and how to charge for it in 3G. A broader survey can be found in [1].

As already defined, web pre-fetching is a technique that tries to improve the quality of service perceived in web browsing. The idea behind pre-fetching is enhancing an HTTP cache with initiative to retrieve in advance those web objects most likely to be downloaded by its users. This way, the hit ratio is increased and the effective latency perceived by the user is reduced, under the premise that there is an excess of bandwidth available.

An important early key work in pre-fetching techniques is [2] where the authors analyze the latency reduction and network traffic for personal local caches, and introduce the idea of measuring conditional html link access probabilities and compressing this information with Prediction by Partial Matching. With this method and by using web-server-trace-driven simulation they obtain a reduction of 45% in latency at the cost of doubling the traffic. This can be considered a practical bound for prediction based on *client access probabilities*, which is the probability that the user accesses the link from a given page based on the user's personal navigation history.

With regard to pre-fetching with shared network caches, as mentioned above the scenario applicable to 3G, [3] estimates the theoretical limits of perfect caching plus perfect pre-fetching in a reduction of client latency of 60%. More realistic results (40-50%) are documented in other works [4] that study the origin of the limit of improvement: it comes from the increase of delay and burstiness caused by the extra traffic put on the network by over-fetching. Regarding mobility, [5] analyses the effect of moving from one access technology to another on the effectiveness of pre-fetching.

As a conclusion from this brief survey techniques and bounds, it can be said that:

- The improvement obtained by pre-fetching is not fixed and depends on the predictability of the user behaviour.
- The gain is measured in reduction of retrieval latency. The maximum performance is around 50%. This means that the perceived improvement is actually determined by round trip times, size and speed of our context. In a context with low RTTs and high speeds the gain can be negligible.
- Effective pre-fetching is expensive in terms of bandwidth consumption. The optimum is obtained at 100% of over-fetching. Hence different strategies will be required depending on whether the link bandwidth is shared or not, and its cost.

3 Charging for third-party provided pre-fetching service in 3G

The conclusions derived from the above observations seem to justify why, nowadays, pre-fetching is not enabled in web proxies. In fact, the main reason for web proxies campus and corporative networks is not just the reduction of latency, but the saving of a fair amount of bandwidth in the shared internet access link. Pre-fetching would reverse the latter positive effect (bandwidth saving) for the sake of the varying no-

ticeability former effect (latency). Moreover, in the fixed internet access business model there is not an easy framework to charge the user –not a terminal- according to complex rules.

On the other hand, 3G subscribers are identified individually after they turn on their terminal and type in their PIN, and there is a working billing system available. The following situation may be frequent. The subscriber has paid for a high class ubiquitous wireless Internet access, and finds that the perceived QoS is under the real access capacity, due to bottlenecks and long delays not (only) in the access but somewhere in the Internet. In this 2G-3G context, network-based pre-fetching may be a tool to cover this demand, improve delivered QoS and charge accordingly.

3.1 Charging schemes for pre-fetching

Once described the nature of the service, its context and theoretical bounds, several charging policies for pre-fetching can be studied:

- Flat rate. This option is the simplest to deploy as it only requires subscription checking. Its drawback is that it relays strongly on the confidence of the subscriber about the promise performance gain. This is a problem that also suffer some commercial offers for differentiated services. In fact, as already described, pre-fetching can not guarantee a pre-determined QoS due to the random variables involved (mainly, the user will and the conditions of the path to the chosen web servers). Therefore, such charging may be rendered unfair by the user, due to the uncertainty of the obtained QoS.
- Charging per GGSN bandwidth devoted to pre-fetching on behalf of a given subscriber. This is directly not feasible due to the high number of subscribers sharing the Internet access of the 3G network. It is not possible to pre-allocate bandwidth for all. An alternative is to pay for sharing the bandwidth proportionally among all active pre-fetching subscribers. Again the charging procedure is very simple, yet the problem comes from the fact that more bandwidth share simply means more probability of obtaining an uncertain amount of extra QoS.
- Charging per pre-fetched information. The rationale for not recommending this option is the same as the previous one. In both cases, however, there is a direct relation between the extra communication costs caused by each subscriber and the charged amount.
- Charging per saved delay. This is the fairest approach from the subscriber perspective as the charging is directly related, not only to the success of the prediction module in terms of hits, but to the exact gain achieved by each hit. On the other hand, it is the most difficult to implement. The client is charged only for the accumulated amount of saved delay. The service provider must have careful control of incurred costs, but has the flexibility to allocate to the pre-fetching activity just unused resources.

- Fixed quota plus per-saved-delay charging. Given the operation costs, a fixed subscription fee is actually necessary, complementary to charging per saved delay, in order to establish a point where service provider and client viewpoints meet.

The computation of saved delay is not straightforward. It could be estimated by this formula:

$$\Delta\delta(I) = \sum_{i \in I} (sizeof(i)(\frac{1}{\min(R_{\text{internet}}(i), R_{UMTS})} - \frac{1}{R_{UMTS}}) + k(RTT_{\text{internet}}(i))) \quad \textbf{(1)}$$

Where I is the set of retrieved web objects during a navigation session and function *sizeof()* returns their respective size. Bit rates R and round trip times RTT are defined by Fig. 1. R_{UMTS} is considered constant, and $R_{internet}$ must be estimated and recorded by the pre-fetcher when the object is downloaded (the simplest implementation will log the size and download duration, integrating delay due to connection setup and data transfer). The reader must note that the RTT term is particularly important when retrieving small objects or when the bottleneck in the path is the UMTS access. Factor $k \geq 1$ models the times the RTT is required for the TCP slow-start mechanism to reach the capacity of the path. A linear charging scheme proposed by the authors by this service would be given by:

$$\alpha = K_{subscription} + K_t \sum_{I} \Delta\delta(I) \quad \textbf{(2)}$$

Where $K_{subscription}$ and K_t are expressed in *monetary unit* and *monetary unit/ms* respectively.

3.2 Provisioning and charging by a third party

The open network interface promoted by 3GPP[6] for UMTS (Universal Mobile Telecommunication System, the ETSI European standard of the 3G IMT-2000 system launched commercially in Europe in the fourth quarter of 2003) provides a framework for A4C, along with a significant number of network control functions, that makes it possible for third-party service providers the delivery of a wide-range transaction-oriented services upon this telecommunications network. This open network interface is named OSA (Open Service Access) [7] and adopts work from Parlay [8]. In this section we study how OSA can be used to enable the provision of pre-fetching by a third party.

Figure 2 shows the main elements involved in this scenario. The pre-fetcher keeps track of the user navigation and, when a given link access probability threshold is exceeded, the associated object is pre-fetched. Based upon the time and object size log available at the cache, it is possible to estimate the perceived delay gain, and

charge the user accordingly through the OSA interface. Note that provisioning of pre-fetching by an external entity implies that all the internet traffic is handled by this entity. This requires low delay operator- service provider and may require NAT (Network Address Translation).

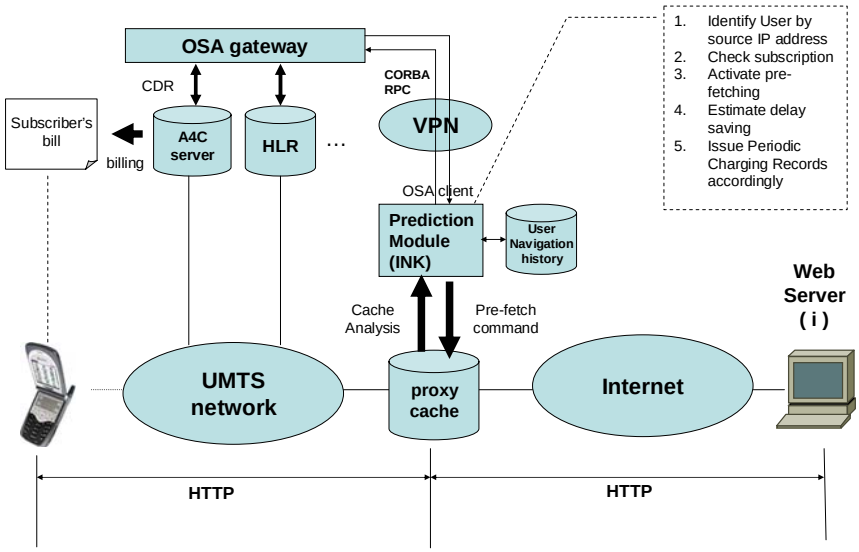

Fig. 2. Charging via OSA/Parlay

The interaction with OSA works as follows. Once the application, in this case the pre-fetching proxy, has been successfully authenticated by the Framework SCF (Service Capability Functions) of the Parlay Gateway, it is able to access to the authorized SCFs. The SCFs required for this service are the Charging SCF and a non-standard Terminal Session SCF which will be described in the next section. In particular, all the communication between the application and the Parlay gateway employs CORBA, although it is modelled as Java classes. For example, in the case of the Charging SCF, the relevant API calls are: from the initial `ChargingManager` object, applications may create several `ChargingSession` objects that refer to a concrete user and merchant. This class implements the interface to perform the most relevant operations of credit/debit to request charging the user for some amount of money or in some volume unit as in bytes, directly (`directCreditUnitReq()`) or towards a reservation (`debitAmountReq()`) for pre-paid services.

A quick overview of some UMTS aspects is required to understand a few implementation issues of charging via OSA of IP-address-identified services.

The UMTS architecture is strongly influenced by compatibility with the 2G digital telephony system (GSM) and the switched packet data service evolved from it GPRS (General Packet Radio Service). Two conceptually new elements have been introduced: the SGSN (Serving GPRS Support Node) and the GGSN (Gateway GPRS

Support Node). These devices are in charge of data packet switching. In outline, the SGSN deals with mobility across RNCs (Radio Network Controller), following mobile stations in its service area and with AAA functions; whereas the GGSN is the actual gateway to Internet (see UMTS forum [6] documents for further details).

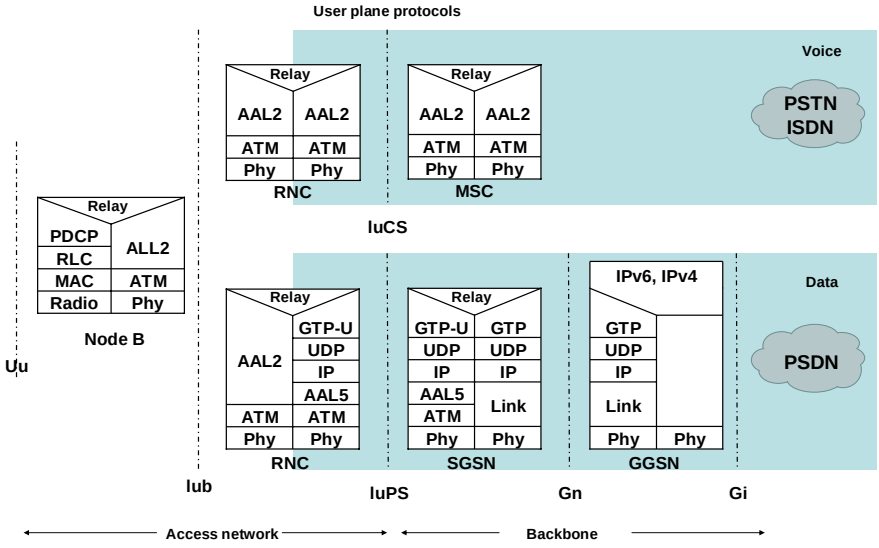

Fig. 3. UMTS protocol architecture

Fig. 3 shows the respective protocol stacks running in each of these elements to transport IP packets. The purpose of this figure is to reflect that the transport of IP packets inside the UMTS network is complex and that the only fixed point in the network where persistent caching is possible is just behind the GGSN, as this is the single internet access for the UMTS subnetwork. Another important issue when trying to charge for pre-fetching is that the "always-on" feature (a fixed global public IP address permanently allocated to the terminal) implies a non-scalable resource consumption in UMTS nodes. Therefore, IP addresses are provided on demand via the dynamic creation of *contexts* for each data session. This means that the implicit authentication given by the source IP address is not valid all the time and the binding <MSISDN , IP address> must be checked whenever the proxy observes a new data flow from an IP address inactive for a period longer than the guard time given by UMTS to reassign the address T_{guard}. Furthermore, since the release of an IP address is not conveyed to the proxy, charging requires periodic charging transactions on the OSA gateway interfacing the UMTS network. Just a leaky bucket whose period is less than the IP address reassignment guard time T_{guard} is enough to guarantee that the identity of the user is still valid and the charge goes to the right bill. This rate for CDR generation also determines the throughput required at the OSA gateway to ac-

complish a target Grade of Service. This can be determined by well known traffic engineering formulae such as Erlang-B or Engset, where the maximum number of active data sessions that can be charged (the number of servers in that formula) is given by $m = C_{server}/C_{client}$, where C_{server} is the capacity of the server in transactions/minute, and $C_{client} > 1/T_{guard}$ is the CDR generation rate of a single data session.

4 A prototype

In the context of project [9], a testing scenario for the concepts developed in this paper was set up. The targets were to assess the viability and performance of pre-fetching in UMTS, to evaluate provisioning by a third-party provider, and to test alternatives for charging for this service via OSA/Parlay. The OSA gateway employed was AePONA Causeway, set up at Nortel Networks Hispania premises, Vodafone provided its UMTS network and UC3M developed a web pre-fetcher [10]. Commercial exploitation of UMTS services had not yet started at the time of testing, and, therefore, the system was not tested with real subscribers. The pre-fetcher was connected through a VPN to the OSA gateway for authentication and charging purposes according to the criterium defined in section 3. The maximum delay gain (over 40%) was easily achieved after a number of code optimizations and training of the prediction module. This gain translated into tens of seconds when browsing medium-sized pages at distant servers (e.g. Australia from Spain) even though the pre-fetcher was suboptimally located outside the operator's premises.

Inheriting terminology from the Telephony service, the OSA gateway issued Call Detail Records (CDR) with a flexible arbitrary format that enabled the production of a detailed charging log containing application-specific information about the charged event. As explained before, the experiment required an extension to the OSA API: the Terminal Session SCF. This extension permits to find out what user (i.e. which MSISDN number) is making a given HTTP request by simply checking the packet source IP address. This new functionality is very important in order to make OSA fully transparent to the end user.

A main practical result of the experience is that, regardless of the usage of pre-fetching and caching, the utilisation of a network proxy in UMTS is always advisable. The main reason is the speed of proxies. Proxies have multi-threading retrieval capabilities not usually available at UMTS terminal's web browsers. Furthermore, in the scenario deployed for the trial, the proxy was located 200ms away from the terminal (RTT=400ms) and still the improvement was significantly high in sequential retrievals.

5 Conclusions

Several conclusions can be drawn from the previous discussion and from the practical experience obtained with the test platform.

- Today the mechanics of pre-fetching are well known and it is clear that the performance gain is expensive in terms of bandwidth consumption and added traffic burstiness when performed on behalf of a large population of clients. However, in the context of internet access through 3G networks, where the business model is quite different from the classical accesses, pre-fetching can be an added-value service that can be offered to a certain type of users on a subscription basis.
- Network-based pre-fetching is viable in UMTS. A rough estimation of the maximum performance gain obtained is given by a 100% hit ratio, which, in the typical delay-bandwidth UMTS-Internet scenarios, leads to up to 40% of delay reduction (accessing far away or low speed web servers) at a cost of double bandwidth consumption. This means tens of seconds of saving when downloading a medium complexity web page. As demonstrated in IST project Opium, OSA enables a business model for pre-fetching in UMTS very difficult to deploy for fixed internet access subscribers, and whose low scalability must be controlled by subscription.
- It must be remarked that a network-based prefetcher does not improve the throughput in the radio segment, but the perceived end-to-end bandwidth due to bottlenecks and delay existing in the fixed part of the network. In other words, the only case when this pre-fetching scenario is really cost-effective is when accessing far away servers. Otherwise, it is the optimized multithreading capabilities of the proxy what predominates and still justifies the insertion of a proxy.
- Thanks to OSA/Parlay it is possible to authenticate, check the subscription, and charge subscribers of web content pre-fetching. This worked as expected and all tests passed. The range of tarification models applicable is very wide (charging for effective pre-fetching, for average virtual bandwidth excess granted, for the amount of pre-fetched information, etc), and its granularity is limited by the rate of charging transactions at the OSA/Parlay gateway, which is constrained itself by a lineal communications/computation overhead. CDRs were generated at a maximum rate of 2 CDR/minute in our tests. CDRs were issued only when the user selected a link that had been pre-fetched on his/her behalf, and the amount of money charged was proportional to the real delay saving. Due to the unpredictable effectiveness of pre-fetching this seems to be the fairest cost model applicable to this scenario.

References

1. Jia Wang. A survey of Web caching schemes for the Internet. ACM Computer Communication Review, 25(9):36{46, 1999.
2. J. G. Cleary and I. H. Witten. Data compression using adaptive coding and partial string matching. IEEE Transactions on Communication, 32:396{402, 1984.
3. Tom M. Kroeger, Darrell D. E. Long, and Jerey C. Mogul. Exploring the bounds of web latency reduction from caching and pre-fetching. In USENIX Symposium on Internet Technologies and Systems, 1997.
4. Venkata N. Padmanabhan and Jerey C. Mogul. Using predictive pre-fetching to improve World-Wide Web latency. In Proceedings of the ACM SIGCOMM '96 Conference, Stanford University, CA, 1996.
5. Z. Jiang and L. Kleinrock. Web pre-fetching in a mobile environment. IEEE Personal Communications, 5:25{34, October 1998.
6. 3rd Generation Partnership Project (3GPP). http://www.3gpp.org, July 2003.
7. 3GPP. Open Service Access (OSA) Application Programming Interface (API). Technical Specification 29.198+.
8. Parlay Group. http://www.parlay.org.
9. Opium project site. http://www.ist-opium.org/.
10. INK tool web page. http://matrix.it.uc3m.es/opium