Smart Data Caching in Archeological Wireless Applications: the PAST Solution

Massimo Ancona Walter Cazzola Daniele D'Agostino
DISI-Department of Informatics and Computer Science
University of Genova, Italy
E-mail: {ancona|cazzola|dagostino}@disi.unige.it

Abstract

Wireless computing, because of the limited memory capacity of the palmtops, forces to separate data (stored on a remote server) from the application (running on the palmtops) that uses them. In applications working on data that frequently change, several kilobytes of data are exchanged between the server and the client palmtops. It is fairly evident that a similar tight coupling may easily saturate the network bandwidth when many palmtops are used in parallel, thus degrading the performances of the application running on it. This paper shows a way to reduce the waste of bandwidth by exploiting at the best the palmtop memory to deal with data caching. The proposed smart data caching is based on context information. We have also applied our method and analyzed its features in a specific application: electronic guide to archeological sites in the PAST EC IT project.

Keywords: Wireless, Caching, Context Awareness, Location Awareness.

1. Introduction

In these last few years there has been a considerable penetration of technology in several branches of the human sciences, like archeology. Archeology may benefit from it by adopting a computer-oriented approach in order to disseminate knowledge or promote archaeological resources.

PAST: exPeriencing Archaeology across Space and Time (IST 20805) [2] is a project aimed at exploiting wireless computer networks in archaeological fields. PAST consists of a system that provides an electronic handy guide to visitors of an archeological site. This electronic guide can adapt the visit to specific time requirements and cultural interests of the visitor and in dependence of his current location.

A wireless application running on palmtops, such as PAST, has several bottlenecks to deal with, e.g., temporary disconnection from the server, bandwidth saturation, and time and power consumption. A peculiarity of this kind of applications is the tight coupling between each palmtop and the remote server which provides the data it has to deal with. This tight coupling and the continuous need for communication involving the palmtop are the main reasons for battery charge and network bandwidth loss and for efficiency losses. Data caching can be the answer to these issues because it can reduce the number of connections to the remote server locally storing data and optimize the data access (local access is faster than remote access). However, current caching algorithms have to be adapted to deal with the limited storage space available on palmtops.

In this paper we are exploiting the opportunity of using contextual information (i.e., information related to a specific topic or environment) and location awareness (every palmtop knows its physical position) to realize a *smart data caching* algorithm. The basic idea consists in prefetching contextual- and location-dependent information and caching them on the palmtop *asynchronously* to the normal computation. Of course, the data that can be stored on the palmtop are limited in size by the memory available on the palmtop itself but, as we will show, our caching approach is more thorough and more efficient than standard approaches. First, we will face the problem in a generic wireless applications; then we will apply our solution on the specific case of PAST.

The rest of the paper is structured as follows: in section 2 we give an overview of the rules that can be used to group data for caching. In section 3 we show our idea of smart caching applied to the archaeological guided tours provided in the PAST project. In section 4 we show the performance improvement we got using our approach and, finally, in sections 5 and 6 we describe some related works and draw our conclusions.

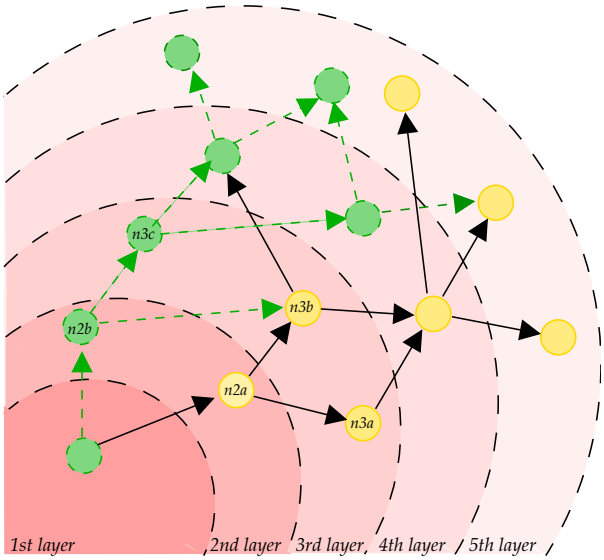

Figure 1. Logical data layerization.

2. Context- and Location-Aware Caching

In client/server applications, such as web or database based applications, servers store data and provide such data to clients. In many cases the size and the number of exchanged data can be very large and the downloading time may represent the larger part of the execution time on the mobile unit.

Data can be *sequenced* by the *proximity* relation, i.e., data are logically correlated by the order that the client uses to access them, e.g., in a web application, the client has to access (and then to download) page `A` before accessing page `B` if page `A` has a link to page `B`. Moreover, data can be grouped by *similarity* of contents, e.g., all data related to a specific topic.

Last but not least, in many applications, data are tightly coupled with the physical location of the device using them, e.g., audio and video streaming, car navigators and interactive life simulators. In this cases, data can be grouped on the basis of the physical location they are correlated with. For example, information about the route we are following with the help of a car navigator.

Similarity, proximity and location provide criteria to identify and organize data that can be cached together by a client unit saving downloading time and bandwidth.

Proximity Relation. All data exchanged between client and server can be logically structured, following the order used by the client to access them, in a connected, directed, acyclic graph (we will unfold all cycles). Nodes represent exchanged data whereas edges denote the proximity relation among data. This logical organization is fairly evident when the exchanged data are multimedia hypertexts, e.g., an e-book, every node represents a page of the e-book whereas every edge represents an hyperlink between two pages of the e-book.

Applying the topological sorting algorithm [8] we can unfold the graph and impose a layerization on it (see figure 1). First layer contains the starting datum, then i-th layer is composed of nodes that can be reached from nodes of $i-1$-th layer with exactly a step. This layerization provides both a fair representation of the proximity relation and a general criterion for caching data. In fact, given a datum belonging to the i-th layer we are sure that the next accessed datum belongs to the $i+1$th layer. A criterion to realize a smart caching consists of prefetching all data belonging to layers adjacent to the layer currently visited before the next download. Further optimization can be represented by caching only data belonging in a layer and reachable from the currently examined datum, e.g., considering the situation in figure 1 if we are visiting the node n_2a then we will cache data related to nodes n_3a and n_3b and not to node n_3c that is unreachable from node n_2a.

Of course, this kind of caching can be carried out by a specific process when the client side application is idle, spending its latency time for housekeeping. Unfortunately, this technique does not take care of optimizing also the waste of bandwidth, and could not be exploited by devices with limited memory capacity like palmtops. In fact, palmtops (and in some cases also desktops) cannot store all the data directly reachable from the currently examined datum, hence, we need a mechanism which further filters eligible data for caching in order to fit in the available memory.

Similarity Relation. A way to overcome the limits of previous techniques consists of using information obtained by the user, e.g., information related to user's interests, cultural heritage or geographical position. This information help partitioning data by similarity and skimming them to reduce both size and number of information to cache. This extra information mainly provide a criterion to subdivide reachable data within a layer into chunks ordered by the probability of being downloaded, e.g., given a technical e-book about programming languages, knowing the reader interest for type systems with this approach the caching system can label as "very unlikely" hyperlinks related to untyped programming languages and avoiding to cache them.

Moreover, caching mechanisms based on the similarity relation take particular advantage of using user-definable description languages, such as XML [9], to structure data. Altering the *document type description (DTD)* semantics/interpretation, through an XSL [1]-based approach, the caching system can go a step deeply by purging each not required piece of information from composite and therefore

larger data. In this case the caching system discriminates among nodes as well as among parts of each single node. This approach, with similar results, can be also applied when data are dynamically generated, e.g., nodes are PHP or ASP pages.

Location Awareness. Constraints introduced by information on client's location can be considered a special case of the proximity relation. The knowledge of the location adds an extra rule for filtering information to be cached, but its usability is limited. In fact, we can only carry out location-aware caching when the client's physical location can change (e.g., when the client part runs on a palmtop) and when there is a correspondence between the exchanged data and the client's physical location, e.g., virtual reality.

By using information related to the physical location of the client, only the data concerning objects physically close to the mobile unit should be cached. Stored information only cover a fixed area enclosing the client's location (e.g., a ten meters wide circle centered in client's location). Covered area is wide as well as the palmtop memory is large. The covered area can be widen by using the orientation besides the location: in this case, only the data concerning the *visible* area in front of the client, i.e., a cone originated in the client location, has to be cached.

This particular technique offers a starting point for caching. In fact, when a mobile client is turned on the only useful data we have are its location, hence the caching system will start caching information steered by its location, in the case this approach is feasible. Moreover it allows the caching system to deal with sudden variation on user's requirements, e.g., the user leaves from the guided tour to see something not listed.

The main problem of this technique is the absence of a standardized location-sensing technology to support location-aware applications. Despite the critical need for location information, each ubiquitous computing system typically treats location data in its own way, or rather many systems build the entire implementation (sensing, representation and application logic) as a monolithic structure. For a deeper discussion of these problems and a complete survey of localization systems survey we refer to [5].

3. Smart Data Caching in PAST

The criteria, we discussed in section 2, for grouping data to get an efficient caching are applicable under specific conditions and the improvement they provide depends on the application, moreover to deal with the size of palmtop memory presents a severe limitation for simply applying only one of the showed techniques. Hence, to get the best result, in implementing the caching system for the project PAST we exploited all these techniques.

In the next of this section, we will briefly describe the application, our approach to caching in this particular application and its architecture.

3.1. PAST: the Electronic Guide.

PAST is a large project whose intent consists of supporting tourists and specialists in visiting an archeological site. The application is composed of two main parts: a database containing all the data related to the site and an electronic guide retrieving such data from the database and showing them in a suitable form to the user. The electronic guide directly works on the field during the visitor's tour, this is possible thanks to a wireless network which connects the electronic guides to the database running on a relatively remote workstation.

The client-side application is a context-dependent application because it adapts its behavior (i.e., the guided tour) in accordance with the environment (i.e., both the user's preferences and its physical location) [7]. The visitor specifies which path he will follow and his/her own cultural interests (*visitor's profile*) at the beginning of the guided tour. On this basis, the application queries the database for organizing and obtaining the required data, that are displayed as (dynamic) HTML pages by a browser and can be navigated by the user, proceeding at the same rate of the tour (proximity relation). The interface provided to the user can be stylized by two buttons: `back` and `next`. These buttons are the unique mechanism available to the visitor for steering his tour.

3.2. PAST Caching System.

Each data page displayed by the system, describes part of the archaeological site and includes text, photos, brief documentaries and short speeches. Text and links referring to it are dynamically generated with the HTML code, whereas, the other information are external `jpeg` and `mpeg` files. Of course, text is not the most heavy part to download and it is a nonsense to modify the PAST system to use statically generated HTML pages only for caching them. Whereas the multimedia part of each page can reach 600-700 Kbytes and it is very convenient to speed up the access to these data. The caching system will focus is efforts on efficiently and transparently rendering available this information to the client.

Caching System Composition. The caching system is a module of the PAST system which works transparently to the rest of the system. The caching module is logically structured in three submodules: *localizer*, *filter*, and *cacher*. Each of these submodules works like a daemon. They col-

laborate together to keep updated the cache of the palmtops without the knowledge of the rest of the PAST system.

Cacher. The cacher daemon is responsible of caching the multimedia data. It receives from the filter daemon a list containing the URLs of the data it has to download. A cooky traces the visit and the cacher exploits this cooky to store in the browser cache all data that the user is likely to use in the near future without interfering with the normal sequence of the visit. Moreover, this daemon deals with the limited capacity of the palmtop purging useless files from the cache before storing new data.

Filter. Each filter daemon is associated to an access point and represents the kernel of the caching system. Its main task consists of choosing the data that are to be cached with respect to context and location information. It obtains palmtop location from the localizer and the context information from the original application (that is, in our case, from the *profile* of the user). It queries the central database to get a preview of all the files that could be downloaded by the user, then it skims them in accordance with similarity, proximity, and location-awareness criteria. The result is a list of URLs of multimedia data with a very high probability to be required. This list is passed to the cacher.

Localizer. This daemon locates each palmtops with respect to the access point networks. Each access point internally keeps a list of palmtops in its zone of coverage. When a palmtop moves from a zone of coverage to another the access points involved update their lists. The localizer polls each access point looking for a variation in such a list. Every change is notified to the filter associated to the access point. The localization is quite raw but sufficient for this kind of application, where moving from and access point to another means that a different set of data are likely to be accessed.

In order to complete its task, our caching system needs a mapping between each page and the files it links. Such a mapping is stored in the database and retrieved from the filter daemon. The filter daemon uses this information to generated the list of URLs for the cacher.

Location and Context Aware Caching in PAST. To determine a reasonable subset of data to store in the local memory (to get a really smart caching) is convenient to implement a mechanism that exploits all the criteria exposed in section 2.

Context aware caching is straightforward realized in PAST. The similarity relation is well represented by the user's profile and helps in filtering data by user's interests, e.g., caching data concerning stone age rather than bronze age. Whereas the proximity relation is represented by the page organization and by the interface provided to the user, i.e., by the buttons `next` and `prev`. The caching system

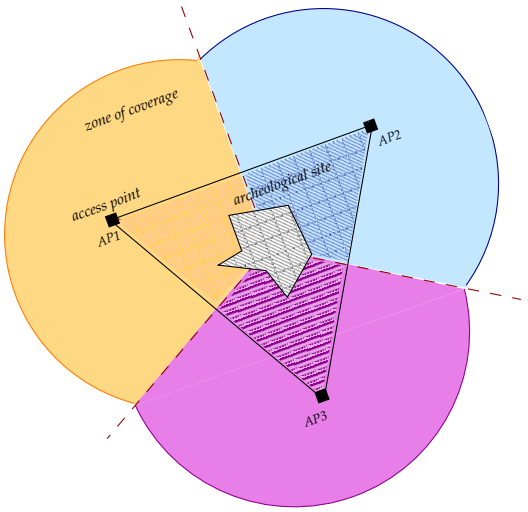

Figure 2. Zone of coverage and localization in PAST.

will keep in the local cache only the data related to pages reachable from the current page clicking n times either on the button `next` or the button `prev`. The value of n depends both on the estimated size of each page (S) and on the capacity (C) of the palmtop memory, as follows:

$$n = \left\lfloor \frac{1}{2}\left(\frac{C}{S} - 1\right)\right\rfloor$$

In this way, our system caches a *virtual window* of $2n+1$ pages on the given route; in our experiments $n = 4$. When the users moves too near the border of the window the caching system will update the window removing the data related to pages far from the current location and loading the new data. The window and the location of the user inside the window is determined by a *cooky* updated at each step.

Location aware caching is tailored to the archaeological sites involved in the project. Monuments in Bibracte (France), Toumba (Greece), and Passo di Corvo (Italy) are sparse on a wide area and without obstacles near the monuments, so, we do not need a particularly thorough localization mechanism. A more precise localization can be get by using a specific hardware such as a *GPS* [6] card, but this means to buy such an extra hardware for each mobile device increasing the cost of the electronic guides. A more precise localization may have a large impact on the PAST software organization and on the granularity of our caching system without changing its basic structure. We have split each site in several areas of coverage depending on the number of available access points and on the site breadth (see Figure 2). In a radio cellular system like IEEE 802.11 a mobile device has to be registered from only one local base sta-

	elapsed time	*battery consumption*
Desktop	13.52 min	not available
iPaq (deactivated)	18.13 min	21%
iPaq (activated)	15.58 min	14%

Table 1. Summary of the Experiments.

tion (in our case an access point), exploiting this property it is easy to detect which palmtops are in a specific area of coverage. This approach provides the caching system with a starting point when the system is initialized and with a mechanism for covering gaps in the visit (e.g., due to the non-adjacency of two monuments).

4. Performance Evaluation

Our experiments consists of simulating a visit on a subset of the whole PAST database. We have only considered a subset of 18 Mb, subdivided in 61 pages belonging to two points of interest covered by an access point. We have executed the experiment both on the desktop containing the web server (avoiding remote exchange of data), and by using the palmtops iPaq H3630, equipped with Intel StrongARM at 206 MHz and with 32 Mb Ram, WindowsCE and Pocket Internet Explorer. Testing the caching system on palmtops, we have measured the elapsed time and the battery consumption (the battery consumption has been provided by an external application) with and without using our caching system. As reported in Table 1, by using our caching system we get an improvement in saving both time and battery. Better improvements can only get by using a faster mobile device (they are already been announced) because we have detected a delay due to the rendering of images and not in the downloading phase.

5. Related Work.

One system providing caching in a similar contexts is GUIDE [3]. GUIDE is a context aware hand-held tourist guide for visiting Lancaster. GUIDE uses both visitor's profile, like PAST, and environmental information, e.g., it uses the time and the date to inform the user about the closure of the monument/museum he would like to visit. It contains a number of smaller and non overlapping cells, every server is associated with a cell, and is responsible for broadcasting information in its zone of coverage, i.e., to the palmtops in the cell. They approximate the location of each palmtop to the location of the server it is using, like in our case. Location is the key value used by GUIDE caching mechanism. It provides [4] its own broadcasting mechanism, used by the local proxy. Cell servers periodically updates the content of the cache of each palmtop in their coverage zone, broadcasting information relative to cell attractions. In this way, many of the pages that an user can request will already be in the local cache. Local servers are also used as second level cache, but contextual information are not used by the caching system.

6. Conclusions

In the paper we explored some caching techniques for mobile systems, these techniques are based on the concepts of *similarity*, *proximity* and *localization*. A mix of the explored techniques has been implemented in the proposed caching system and used in the project PAST. Moreover, we have carried out some experiments detecting a 7% saving of the battery charge of the mobile unit.

References

[1] S. Adler, A. Berglund, J. Caruso, S. Deach, T. Graham, P. Grosso, E. Gutentag, A. Milowski, S. Parnell, J. Richman, and S. Zilles. Extensible Stylesheet Language (XSL) 1.0. W3C Recommendation available at `http://www.w3.org/TR/xsl/`, Oct. 2001.

[2] M. Ancona, G. Dodero, V. Gianuzzi, O. Bocchini, A. Vezzoso, A. Traverso, and E. Antonacci. Exploiting Wireless Networks for Virtual Archaeology: The PAST Project. In *Proceedings of the Virtual Archaeology between Scientific Research and Territorial Marketing Conference (VAST 2000)*, Arezzo, Italy, Nov. 2000.

[3] K. Cheverst, N. Davies, K. Mitchell, and A. Friday. Experiences of Developing and Deploying a Context-Aware Tourist Guide: The Lancaster Guide Project. In *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking (Mobicom 2000)*, pages 20–31, New York, USA, 2000. ACM Press.

[4] N. Davies, K. Cheverst, K. Mitchell, and A. Friday. Caches in the Air: Disseminating Tourist Information in the Guide System. In *Procedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications (WMCSA'99)*, pages 11–19, New Orleans, Louisiana, USA, Feb. 1999.

[5] J. Hightower and G. Borriello. Location Systems for Ubiquitous Computing. *IEEE Computer*, 34(8):57–66, Aug. 2001.

[6] B. Hoffman-Wellenhof, H. Lichtenegger, and J. Collins. *Global Positioning System: Theory and Practice*. Springer-Verlag, New York, fifth edition, 2001.

[7] A. Vezzoso, A. Romagnoli, O. Bocchini, and S. Locati. Managing and Organizing Archaeological Data Sets with an XML Native Database. In *Proceedings of the Virtual Archaeology between Scientific Research and Territorial Marketing conference (WAST 2001)*, Glyfada, Greece, Nov. 2001.

[8] N. Wirth. *Algorithms + Data Structures = Programs*. Prentice-Hall, Englewood Cliff, 1st edition, 1976.

[9] Word Wide Web Consortium. Extensible Markup Language (XML) 1.0. W3C Recommendation available at `http://www.w3.org/TR/REC-xml`, Feb. 1998.