DATAMAN project: Towards a Mosaic-like Location-Dependant Information Service for Mobile Clients

Arup ACHARYA, Tomasz IMIELINSKI and B. R. BADRINATH
Department of Computer Science, Rutgers University, New Brunswick, NJ 08903

Introduction

The goal of the DATAMAN project is to develop primitives and build an environment for location dependant applications . In such applications, information requested by the user depends on the location from which it is accessed, and which may be invalidated or continue to be valid when a user changes location. Examples include directions to a static/mobile entity, tracking current location of mobile objects, accessing proximate services, and as indicated in [10], binding user application to nearby resources. The communication between the client and the information source/server does not necessarily have to occur over a wireless link. In fact, a mobile client may simply connect to different ports of a fixed network from different locations, and be disconnected in the interim periods. However, the typical scenario will consist of a wireless connection from a mobile client to the information service residing in the static portion of the network.

The goals of this paper are twofold: first, we would like to conceptually describe the main *primitives* necessary to build location-dependant and wireless applications, and second, we describe the *preliminary implementation* of an intelligent building that demonstrates some of the primitives.

We will start with describing the proposed primitives. The implementation is described in the last section.

Overall Design

Determining Location

We require that the mobile client be equipped with a mechanism to obtain its current location. This mechanism could either be tied to the tracking scheme used by the static network (to setup a communication path to the client) or be completely independent of it. For example, the client may determine its location via GPS in terms of its absolute geographic coordinates. Alternatively, in a environment with predefined cells, i.e. areas of wireless coverage with a common access point to the fixed network, the client's location can only be pinned down to a cell-id, thereby defining its location relative to the fixed network. Therefore, ''location'' has different connotations depending on the mechanism in use, with a range of granularity from a few square miles in the case of macrocellular structure (as in a cellular telephony/AMPS setting) to a cellsize of the order of meters, e.g. infrared cells within a building as in [4].

Page

The information service that we envision is organised as a collection of pages , with a hyperlink-based "Mosaic like" interface on the client. Pages may either be active or passive. An active page is one whose contents are sensitive to change of location or time of access.

Pages are delivered to a client from a server. The point of access of a page not only determines its contents, but also the server that delivers those contents. This is done in a manner transparent to the client, i.e. the client should not first need to determine which server must it contact to refresh a page upon page-invalidation due to a move.

Scope The set of locations in which the contents of a page are valid is termed its *scope*. Scope of a page will vary substantially depending on its contents, and in many cases mirror the geographic region in which it is valid. For example, pages that reflect

– *wide-area services* such as weather information, are associated with a scope spanning cells covering tens if not hundreds, of miles,
– traffic conditions will encompass cell(s) within a few square miles,
– availability of empty spaces in a parking lot, will have a scope spanning cell(s) within a few hundred metres, and

– at the extreme, cells that span single rooms within a building will likely have pages, such as a list of occupants, whose scope will be limited to that cell alone.

The concept of scope is useful to efficiently implement page invalidation due to location changes.

There will also be pages that reflect services such as stock market information which will not depend on location.

In addition to pages being location dependent, contents of pages provided to mobile clients will also vary with time. Thus, a client may find a page invalidated either due to a spatial change (e.g. directions) or temporal change (e.g. a stock quote).

Mobile clients are faced with two chief resource constraints: 1) power consumption and 2) low-bandwidth of wireless connection. Additionally, the wireless link is usually asymmetric, i.e. power consumed for packet transmission is higher than reception, and the downlink bandwidth (from the fixed server to the mobile client) is higher than the uplink bandwidth. One of our goals is to develop primitives for detecting invalidation of page contents that utilize the asymmetric nature of the wireless links. In particular, these primitives should eliminate the need

– for a client to poll a server for verifying validity of time-dependent pages, or
– to fetch a redundant copy of a page from the server in its current cell whose contents are not invalidated by a move.

For location-dependent pages, the associated *scope* is sent to the client along with the *contents*. The scope information is sufficient for the client to locally detect page invalidation. Similarly, for time dependent pages, there is an associated *expiration period* attached to the page after which the page is invalid, e.g delay in flight departure.

Scope is useful to invalidate a page based purely on changes in client location, just as whether a page has crossed its expiration period can be determined locally using the client's clock. For pages whose validity depends on location/events external to the client, we propose a server-based callback/trigger facility to notify a client. Thus, for active pages, we have two modes of invalidation: client-based and server-based.

Resource-adaptive links We propose *resource-adaptive* hyperlinks as a primitive for delivering pages whose contents and presentation are governed by the resources available at the client. Each page is now associated with a set of resource *prerequisites* that must be satisfied at the client end in order to receive the page. These prerequisites relate to both features local to the client such as CPU speed, memory size, display resolution, as well as to the network connection[1].

Location-independant addressing of servers

A page can be provided by a single server common to all clients regardless of their location. To access the page, clients specify their current location as a parameter in the URL, which is then used by the server to determine the (location-dependant) contents of the page. For example, the URL
http://128.6.4.7/cgi-bin/floor-map?128.6.157.131
encodes the address of the server (128.6.4.7), the service desired (/cgi-bin/floor-map) and the present location of the client (the cell under MSS 128.6.157.131, using the Mobile-IP scheme of [7]).

The above centralized scheme has the disadvantage that the server must store page contents and scope for each location/cell, in the system. The advantage is that clients can use the same address to contact the server from any location. A decentralized scheme, on the other hand, allows the set of all cells to be partitioned amongst multiple servers, with each server providing page contents and scope for only those cells that are ''under'' it. However, this implies that before a client can retrieve the page corresponding its current location, it must discover the address of the server to use. This entails additional communication from the client violating energy and (wireless) bandwidth constraints, which we wish to avoid.

We propose narrowcast as a network-layer primitive for mobile clients to locate and access the server covering its present location. The requirements of such a primitive are:

– Multiple servers that provide a common service(page) are reachable via the same (group) address.
– A narrowcast request for a page originating from a mobile client be routed to the specific server responsible for handling page requests originating from the cell.[2]

[1] This is a generalization of the feature provided in HTML for displaying alternate text in place of images for text-only client browsers, e.g. <img alt="..." src="...">

[2] In general, it is possible that multiple hosts are configured as servers to cover a common set of cells so that a narrowcast request originating from any of these cells will be routed to all the hosts. We restrict the number of such hosts to one, since it allows a one-to-one transport-layer connection to be set-up between a mobile client and the nearest server.

This is similar to the concept of *anycast* proposed in [9], but with a critical difference, viz. a anycast packet may be routed to any of the servers listening to a common anycast address, while a narrowcast packet will be routed to a specific server depending on the source of the packet. Details of the protocol to implement narrowcasting will be published separately.

Modes of page delivery

Finally, the *mode of delivery* of a page may vary between servers. In general, each server has the following choices in the mode of disseminating a page to clients within its cell:

- **On-Demand Mode** In this mode, the client explicitly accesses a page by transmitting an uplink message to the server.
- **Docking Mode** This mode is designed for clients that operate primarily in disconnected mode, and ''dock'' with the fixed network only to download a set of pages; thus, within the union of the scope of downloaded pages, the client has local access to the service even while disconnected from the network. This mode cannot support server-based invalidation of the active pages downloaded, and requires a higher bandwidth downlink in the docking cell and sufficient on-board memory at the client.
- **Publication Mode** If a page is requested often and by multiple clients, the server instead of replying to individual requests, can periodically multicast the page at a predefined address [6]. We propose using the narrowcast primitive to support this mode as well. A page is narrowcast from a server (periodically) to reach all mobile clients in the cells covered by the server; thus, the clients listen to the narrowcast address. This is symmetric to the on-demand mode, where a request is narrowcast from a client (source) and is routed to the appropriate server (destination). Thus, both modes are supported by a common addressing and routing mechanism, viz. narrowcasting. However, while the on-demand mode requires a one-to-one transport-layer connection between the client and the server, the publication mode requires a one-to-many ''connection'' from the server to multiple clients. So, while TCP can be used at the transport-layer for supporting on-demand mode, there is no commonly accepted reliable transport-layer mechanism for one-to-many connections and needs further investigation.
  - *Delta Publication Mode* The server will not multicast entire pages and instead, send only the *delta* or difference in page contents. Alternatively, it could send an invalidation notification on receipt of which interested clients can individually access the updated page if desired, i.e switch to an on demand mode [1].

In principle, a server can *autonomously* select any of the desired dissemination modes. However, when a client moves outside of the scope of a page, it must first determine the mode in which the server covering its current cell is disseminating the page. This requires support for *mode handoff* on relocation: on detecting that a page is invalid, the client narrowcasts to the group address representing the service. If the (nearest) server supports on-demand mode, it will respond with the updated page; else, it will inform the client to switch to publishing mode, i.e. listen on the narrowcast address and wait for the next publication of the page.

Summary of proposed primitives

- A page defines a service. Contents of an active page change with location or time.
- Invalidation of pages is either client-based (scope/ expiration-period) or server based (trigger/callback) to reduce redundant uplink communication.
- A page may either be delivered on-demand or published periodically from the server.
- The narrowcast primitive is proposed to support on-demand as well as publishing modes of page dissemination.

Map-centric Approach

We take a map-centric approach both in terms of the user interface at the mobile client as well as for application development.

We view the user's terminal as a "magnified eye" which allows him/her to see and interpret "what is around" his/her current location. There is a hierarchy of annotated maps, starting from the most specific map of the area (such as a floor) surrounding the user. The user can "zoom out" to obtain a more global view of the surrounding area. The most specific map, though, serves as a "background" on the the screen's platform. It is possible that different types of clients will see different "views" of the same area. For example a plumber may see the locations of pipes in the walls, a student may be interested in seminar rooms and a computer hardware technician in locations of individual computers. This set of pages will constitute a basic shell of any location dependent applications. We will call it the "*What's around*" application.

Additional applications will be provided on the top of *What's around* application in the form of ''*direction*'' service or ''*nearest*'' service. While the former provides directions to a given location from the user's current location, the latter provides the nearest instance of the object specified by the client.

Map-centric Development Platform The application development interface will be built analogously to the user's interface in a map centric fashion. In this way, the developer will be able to put himself/herself into the "user's skin" and visually debug the application.

As an illustration of how a location dependant application will be built, consider an intelligent building. In the intelligent-building application (which we are developing here at Rutgers), the building is divided into floors and each floor is covered by one or more wireless cells. The floor plan is provided as the most specific view of the cell, with campus maps and township maps at successive zoom levels. How would such an application be developed using the concept of a map centric development tool? First, the developer will obtain a set of maps, one per floor, and associate a map with one server covering all cells within a floor. Then, other points of interest on each map will be linked to additional pages.

Implementation Status

Portions of the overall project have been implemented separately. The canonical application is a *What's around me?* interface, providing locations of proximate services such as nearby printers and directions to various offices from the user's current location within a building. Client mobility is supported via Mobile-IP[7], and the building is divided into cells with the IP address of the MSS identifying a location. On startup, the user's current location is encoded as a parameter in the URL to access a central server, which returns the floor plan for the user's current location (Fig. 1).

The user can zoom up to view a campus map, and then to a map of Piscataway township (Fig. 2). Users receive different floor maps depending on the user type they select, viz. student, utility-worker etc.

Each location is associated with a set of nearest services (Fig. 3) such as printers, vending machines. This is implemented by encoding the desired service and the user's current location within the URL, e.g.
http://rags.rutgers.edu/cgi-bin/printer?128.6.157.130
Each page returned to the client is associated with a scope, so that the page providing a service, e.g. nearest printer, can be invalidated locally at the client.

The directions service is implemented by maintaining a ''routing'' table at the server, which is an adjacency matrix of cells used to route users from their current location to any desired location. To provide directions (Fig. 4) to an entity A, the server first looks up a location directory to determine the IP address of the cell covering A. The routing table then supplies the next cell (Fig. 5) that the user should visit enroute to A, and this process continues till the user's current location(cell) matches that of A. Note that the size routing table is independent of the number of entities to which we provide directions, and is proportional to the square of number of cells in the building. Further, the incremental directions are tagged with a scope (Fig. 6) to provide directions in increments greater than one ''hop'', i.e. guide a user between non-adjacent cells.

Related Work

The work that is perhaps closest to ours, is Mobisaic [11]. Mobisaic extends Mosaic as a information-browser for mobile clients by introducing two new constructs: *dynamic URLs* and *active documents*. Dynamic URLs reference environment variables whose values change dynamically, either due to change in user locations or time. Dynamic URLs are resolved at the client by instantiating the referenced variables with their current values, and sent to the server as a ''standard'' URL. Dynamic URLs thus provide a construct for users to specify location-dependent pages. After a page is accessed via a dynamic URL, it may be invalidated due to a change in any of the environment variables used to specify the dynamic URL. To detect such invalidations, an active document lists the environment variables which affect its validity. Since these variables are ''global'', i.e. their values can be changed by entities other than the local client, Mobisaic uses the Zephyr notification system [3] as a publication-subscription mechanism to propagate changes to dynamic variables to interested parties.

Though similar in aim, viz.location-dependant information browsing, our proposed approach is complementary to Mobisaic. [11] presents new language-level (HTML) constructs for writing active pages and specifying dynamic URLs and implements the necessary interaction between various entities via Zephyr. In our approach, both clients and servers are aware of mobility and the goals are:

- Structure services as pages distributed across multiple servers with each server covering a set of locations
- Network-layer primitives for routing pages requests to nearest server from the client's current location

– Different modes of page dissemination
– Augmenting pages with primitives such as adaptive links and scope, that make it possible for clients to decide (without communicating with a server) if and when a page can/should be accessed/refreshed.

Other related work includes [8] where *dynamic documents* are sent as Tcl scripts and interpreted at the client side to generate the actual contents of the document. The techniques of caching and prefetching pages at the client are used in [8] and also in web clients such as W* [12] and W4[2] that are designed expressly to handle bandwidth and latency constraints of the wireless link. While W* and W4 partition the necessary processing (to access a document) between the (wireless) client and a workstation on the wired network, [5]performs all processing at the client (to the extent allowed by the client's hardware capabilities).

Bibliography

[1] D. Barbara and T. Imielinski. Sleepers and workaholics: Caching strategies in mobile environments. In *Proc. of the ACM SIGMOD Intl. Conference on Management of Data*, May 1994.

[2] Joel F. Bartlett. W4 — the wireless world wide web. In *Workshop on Mobile Computing Systems and Applications*, 1994.

[3] C. Anthony DellaFera et. al. The zephyr notification service. In *Proceedings of the USENIX 1988 Winter Conference*.

[4] N. Adams et. al. An infrared network for mobile computers. In *USENIX Symposium on Mobile and Location-independent Computing*, August 1993.

[5] S. Gessler and A. Kotulla. Pdas as mobile www browsers. In *2nd WWW Conference '94: Mosaic and the Web*, October '94.

[6] T. Imielinski and S. Viswanathan. Adaptive wireless information systems. In *Proc. of SIGDBS Conference, Japan*, Oct. '94.

[7] J. Ioannidis, D. Duchamp, and G. Q. Maguire. Ip-based protocols for mobile internetworking. In *Proc. of ACM SIGCOMM Symposium on Communication, Architectures and Protocols*, pages 235–245, September 1991.

[8] F. Kaashoek, T. Pickney, and J. Tauber. Dynamic documents: Mobile wireless access to the www. In *Workshop on Mobile Computing Systems and Applications*, 1994.

[9] C. Partridge, T. Mendez, and W. Milliken. Host anycasting service. *RFC 1546*.

[10] Bill Schilit, Norman Adams, and Roy Want. Context-aware computing applications. In *Workshop on Mobile Computing Systems and Applications*, 1994.

[11] G. Voelker and B. Bershad. Mobisaic: An information system for a mobile wireless computing environment. In *Workshop on Mobile Computing Systems and Applications*, 1994.

[12] Terri Watson. Application design for wireless computing. In *Workshop on Mobile Computing Systems and Applications*, 1994.

Fig. 1 Top-level Screen: floor-map and services

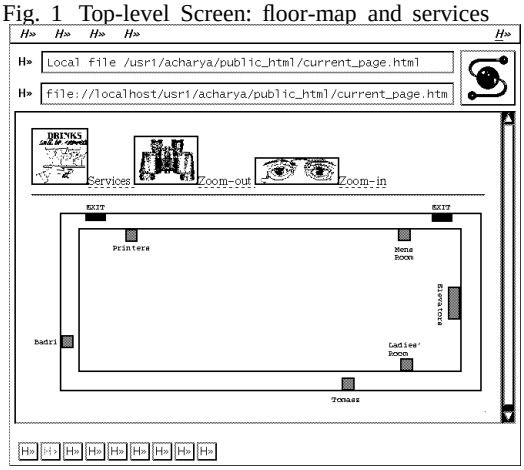

Fig. 2 *What's around me?* zoom to township map

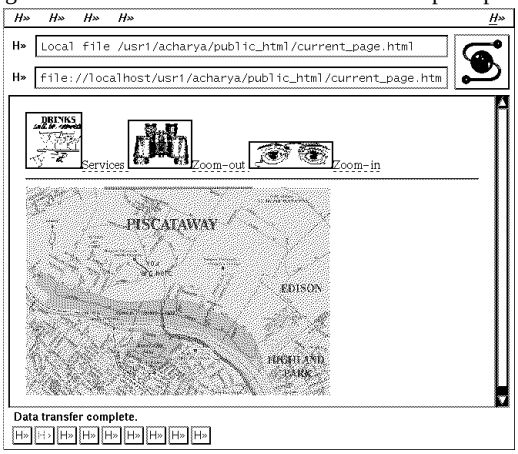

Fig. 3 *Proximate services*

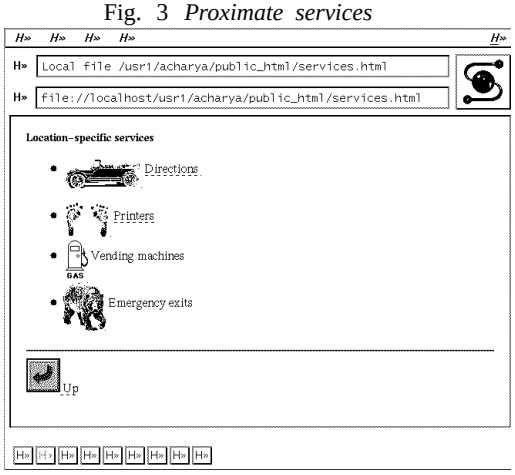

Fig. 4 *Directions*

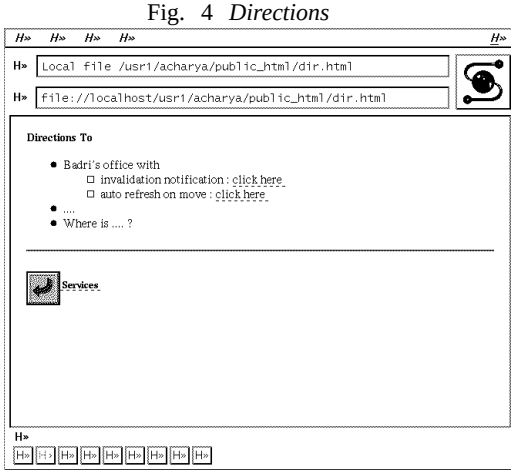

Fig. 5 *Directions to Badri's office from current position*

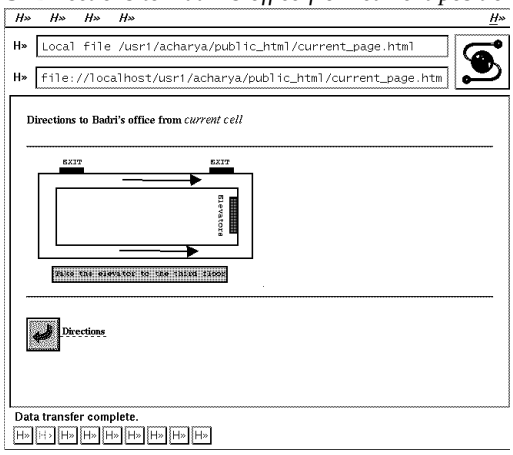

Fig. 6 *Invalidate directions locally (exit scope)*