UbiData: Requirements and Architecture for Ubiquitous Data Access*

Abdelsalam Helal
University of Florida
helal@cise.ufl.edu

Joachim Hammer
University of Florida
jhammer@cise.ufl.edu

Abstract

Mobile users today demand ubiquitous access to their data from any mobile device and under variable connection quality. We refer to this requirement as *any-time, any-where* data access whose realization requires much more support for asynchronous and disconnected operation than is currently available from existing research prototypes or commercial products. Furthermore, the proliferation of mobile devices and applications, forges the additional requirement of device- and application-transparent data access. Support for such *any-device*, *any-application* computing paradigm requires the ability to store and manage data in a generic representation and to transform it for usage in different applications, which may also be running on different platforms. In this article, we give an overview of the UbiData architecture and prototype system and show how it addresses these challenging requirements. We also summarize our ongoing and future efforts.

1. Introduction

The proliferation of mobile devices and wireless networks has created new challenges for mobile computing research. Mobile users demand constant availability of data and information, which is typically stored on their workstations and corporate file servers, even when only limited network bandwidth is available. Moreover, given the growing popularity of portables and personal digital assistants (PDA), mobile users are requiring access to important data regardless of the form-factor, rendering capabilities, and computing power of the device they are using at the time.

In response to these challenges, which are described in more detail in a recent special issue on ubiquitous computing [15], we have identified three goals to pursue and to address these challenges:

1. *Any-time, any-where access* to user data, regardless of whether the user is disconnected, weakly connected via high-latency, low-bandwidth network (e.g., cellular), or completely (yet temporarily) disconnected.

2. *Device-independent data access*, which allows users to switch among different mobile devices of different capabilities without the capability to access the data.

3. *Application-independent access and update* to data to allow users to modify portions of documents and files belonging to classes of related applications (e.g., the ability to modify parts of a document irrespective of the word processing application that was originally used to create the file).

The UbiData (Ubiquitous Data Access) project (*http://www.harris.cise.ufl.edu/projects/ubidata*) [6][8][10] [27][28][29] addresses these three challenges using an architecture and sophisticated hoarding, synchronization, and transcoding algorithms to enable continuous availability of data regardless of user mobility and disconnection, and regardless of the mobile device and its data viewing/processing applications. Furthermore, UbiData automates the task of maintaining currency (up-to-dateness) and consistency of data on mobile devices. For example, our Format-Independent Change Detection and Propagation (FCDP) algorithm allows users to edit and use different versions of the same document using multiple devices and word processing applications.

2. The UbiData Architecture

A conceptual overview of the UbiData architecture is shown is Fig. 1. In this figure, several mobile and fixed clients (henceforth referred to as client devices) are connected to a UbiData Server via the Internet. The UbiData Server consists of a *Mobile Data Server* (MDS) and a middleware component called *Fixed Mobile Environment Manager* (Fixed-MEM or F-MEM). Each of the *UbiData-enabled* client devices has a MEM counterpart called Client-MEM or C-MEM. Note that C-MEM refers to the MEM component on both the mobile as well as fixed clients (see Fig. 1). C-MEMs and F-MEM are often connected through the Internet or Intranet via a wire-line or wireless network.

* This research was supported by the National Science Foundation under grant CCR-0100770.

C-MEM may sometimes be disconnected from any network.

Together, C-MEM and F-MEM eliminate the manual and tedious synchronization of data between the mobile and fixed devices that are employed by a user. We will describe the functionalities of both F-MEM and C-MEM in more detail later in this section.

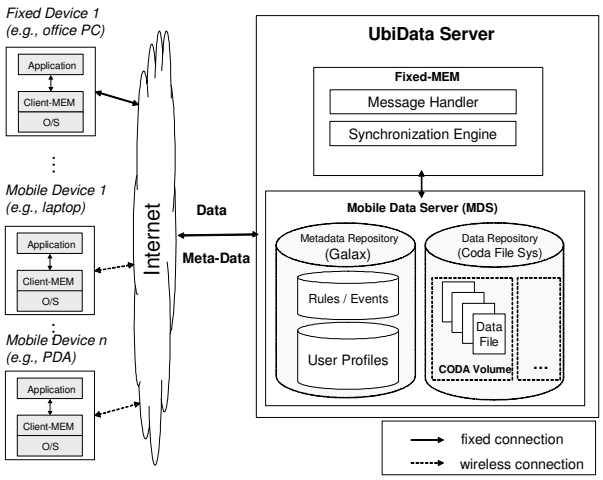

Figure 1. Conceptual overview of the UbiData architecture

Continuing on the server side, the MDS consists of the *Metadata Repository* that manages information about UbiData users, their data, and the devices on which users manage their data. Specifically, for each user, the metadata repository stores a *user profile* that contains information about the user including user preferences, authorization and password information, a list of the devices owned by the user, including their configurations and capabilities, as well as descriptions and usage information about the data files that are stored on the various devices owned by the user. Besides the user profiles, the Metadata Repository also contains the rules and filters used by the FCDP for converting data into various formats to support interoperability among different devices and platforms (*see Sec. 4.2 for ongoing research*). All metadata is represented and stored as XML data, which is queried and updated using the Galax query processor[2] (*see Sec. 4.3 for ongoing research including our work on adding update functionality to Galax*).

User profiles also contain links to the most recent and updated copies of the data files, which are managed by CODA [18] as CODA volumes. The collection of CODA volumes makes up the *Data Repository* shown inside MDS in Fig 1. In a sense, the CODA file system becomes an extension of the disk space on the user's client device(s), which may not be able to always hold all of the data that is part of the user's immediate working set (especially when using a mobile device with limited capabilities). For each user, the most-frequently used files, called a *hoarding set,* are automatically maintained in the data repository on the UbiData server. In addition, in order to prepare a new device for usage or an existing one for a disconnection (e.g., during travel), the latest version of the files that make up a user's hoarding set are *incrementally* copied (hoarded) [27] onto the device to increase data availability during disconnection and to reduce dependence on the network. Membership in the hoarding set is predicted based on past behavior by the C-MEM, which monitors user access patterns on each device. In addition, users can request that certain files be added to/removed from the hoarding list in anticipation of future data needs.

UbiData operations are supported by MEM as follows: On each mobile device, C-MEM continuously monitors the user's data access patterns in order to automatically determine the *working set*. This monitoring occurs regardless of whether C-MEM is connected, weakly connected or disconnected. Accesses to system files and other unrelated data are filtered out and removed from the working set using a filtering mechanism [27][29]. The working set is used by F-MEM to incrementally hoard data to the mobile device. During weak connections or when a mobile device reconnects to the network, C-MEM propagates file accesses and any updates to data in the hoarding set to F-MEM, which initiates synchronization procedures with the corresponding copy in the MDS. If the data item is new and does not have a counterpart on the MDS (e.g., when the mobile user creates a new file), it will be inserted into the hoarding set on the server (*see Sec. 4.3 for our ongoing research in hoarding and synchronization and Sec. 4.4 for mobile transactions*).

The architecture in Fig. 1 has been implemented in a prototype system that is running in the Harris Lab at the University of Florida. The prototype, which consists of the UbiData server and several clients, supports the automatic selection and efficient hoarding and synchronization of data files on mobile clients. The prototype also supports basic user profile management. The integration of application- and format-independent hoarding and re-integration capabilities is under development.

3. Related Work

Most of the research related to ubiquitous data access falls under the broad category of mobile or ubiquitous computing. Due to space constraints, we can only highlight some of the most important efforts.

In the area of infrastructure support for mobile computing, the Coda system [21], which is based on the client/server model, provides a mobile environment by supporting a variety of operations such as disconnected operation, file hoarding, weak connection adaptation, etc. Coda also tracks local file modifications in a Client Modification Log to save bandwidth. The Intermezzo

[2] http://db.bell-labs.com/galax/

file system [2] provides similar functionalities. The Roma personal meta-data service [23] provides a Peer-to-Peer based file information service.

In the area of hoarding and synchronization, the SEER system [9] correlates the user-accessed files into clusters by calculating the semantic distance between referenced files, rather than considering them as discrete files. A cluster is hoarded as a whole unit to mobile devices by the hoarding system. Lei and Duchamp proposed a tree-based algorithm to track and analyze program executions in [14]. SyncML [24] is a specification designed for mobile data replication and synchronization between heterogeneous mobile devices. Incremental updates are also used in Rsync [25] and XDFS [16].

In terms of data and metadata representation, XML is often the data representation of choice due to its flexible modeling capabilities, its support for heterogeneous environments, as well as its improved integration with data management software. For instance, a number of XML formats have been proposed to represent user profiles (e.g., GUP from 3GPP[3], Liberty Profile Specification 1.1 from Liberty Alliance[4]).

XML querying and, more recently updates are a very active area of research and subject of intense competition between vendors. Several products include update capabilities: the Tamino XML Server from Software AG, Xindice from Apache, and X-Hive/DB XML from the X-Hive Corporation to name a few. Update capabilities of those products focus on simple forms of updates, making them difficult to use for applications requiring support for complex updates such as user profile management described below.

Much research has also been done in the areas of XML difference algorithms and bandwidth adaptation. Researchers in the Lore project [17] were one of the first to develop difference algorithms for hierarchically structured data. Other XML specific diff algorithms include laDiff [3], IBM's *XML Diff Merge Tool* and *XML treeDiff* [7]. In addition, the open source community and Linux developers have created numerous XML converters for popular word processing formats [26] that we are leveraging in FCDP.

The Puppeteer [11] project at Rice University is the closest to our vision of systems that support ubiquitous computing without adapting the user's applications. Puppeteer offers the ability to transmit content in low- and high-fidelity modes, and switch between the two. Puppeteer also is beginning the process of allowing edits to the low-fidelity components on a mobile device and integrating those changes into the high-fidelity version. A related line of research in bandwidth adaptation is conducted by the Odyssey project [18]. Odyssey also adapts application data to the current state of the network connection. Unlike Puppeteer, which uses public APIs of applications to manipulate that application's data files, Odyssey requires applications to be customized to support its implementation scheme. This is also in contrast to FCDP, which utilizes XML as a common format for supported applications.

Finally, in the area of mobile transactions, several research efforts and new transaction models have been introduced. In [5] the authors provide a comprehensive survey on mobile transaction models.

[3] http://www.3gpp.org/
[4] http://www.projectliberty.org/

4. UbiData Research Efforts

To date, we have focused our research efforts on four important problems: (1) Policies and algorithms in support of automatic selection, filtering, and hoarding of user working sets; (2) hoarding and synchronization of data across platforms and applications; (3) metadata management; and (4) support for mobile transactions. We briefly describe these efforts in the following subsections.

4.1 Automatic Selection and Hoarding

One of UbiData's goals is to support any time access and update to data from heterogeneous sources (e.g., files belonging to different file systems). To achieve this goal, we introduced and developed automatic and device-independent selection, hoarding, and synchronization of data [27] (also see Fig 2). Since this requires adding hooks to the operating system, we have focused only on the Linux and Windows platforms. The hooks enable UbiData to sense and collect events related to data access. We have developed filtering mechanisms to sift through the events in real-time and to collapse them into a minimal set that can be later analyzed to extract the working set (hoarding set) [27].

We have designed several algorithms and heuristics to perform the filtering and to determine the hoarding set. We have also conducted experimental studies to evaluate their effectiveness and the overall mobile access performance. We have used trace-based simulation and have shown how to design and configure hybrid priority hoarding policies to maximize the hit ratio of data access during disconnections.

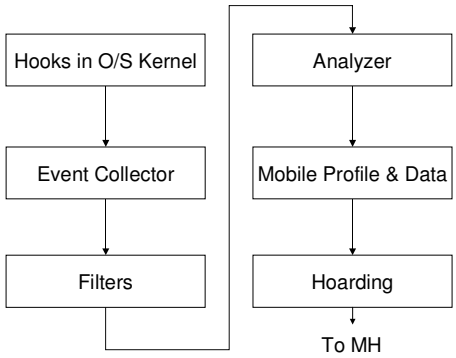

Figure 2. Automated Selection and Hoarding in UbiData.

In connection with our work in automated selection and hoarding, we conducted research to adapt the mobile network environment to minimize the use of bandwidth in both directions (hoarding and synchronization). This research is critical to the ability of UbiData to maintain its operation and properties under weak type of network connections. We have employed message queue optimizations and incremental and lazy update propagation. Some messages may be cancelled or merged. Thus, after every modification, only the minimal differences are shipped scheduled for future communication. This adaptation required some infrastructure support, such as file versioning and stale file management. Our work in [6, 8] quantified the impact of incremental hoarding and re-integration on overall performance.

4.2 Format-Independent Change Detection and Propagation

We have developed change detection and propagation methods to synchronize documents, which are stored on various computing devices. Our approach, called Format-Independent Change Detection and Propagation (FCDP), is capable of computing the changes that have been made to a document on one device and applying the net effect to an unedited copy, which uses a different format and representation. This allows users of mobile devices to modify documents, which have been generated by more powerful applications on different platforms, and to have the modifications reflected in the originating documents [10].

Specifically, we have developed algorithms and techniques for converting XML documents into a client-usable format (in this case an ASCII text editor). We have also developed algorithms and techniques to convert client-usable formats into minimalist XML documents based on the originating documents' XML DTDs/Schemas. We have also developed tools to track where and how the minimalist XML representations intersect with the XML tag and attribute set of the originating document. We developed a technique to infer, whenever appropriate, attributes for new XML nodes when the modifying application cannot generate those attributes. Finally, we have developed a heuristic to integrate shared and unshared nodes across XML documents to ensure useful ordering of document content.

Despite the promising results of our experiments, our FCDP component system is still work in progress. We are currently refining the approach to provide more capabilities and improve performance. To improve performance we are developing customizable optimization routines that rely on specific knowledge of the XML data structure. We are also experimenting with placeholders [28] in the bandwidth-adapted files to mark the location and existence of stripped elements. That step will help us improve performance (both in time, and size of the edit script). Placeholders will allow movement, deletion, resizing, and other manipulation of the placeholder (though not its actual contents).

4.3 Metadata Management in MDS

With the increasing popularity of network-based services (e.g., voice and e-mail messaging, Web portals), the management of user profile information has become an important issue for service providers and users alike. In particular, easy yet controlled access and sharing of user profile information is an important prerequisite for building so-called integrated or converged networks in which users can freely use a host of services from cooperating providers through a single sign-on [20].

Our research in metadata management falls into two related areas: Design of an integrated schema for storing user profile information, and development of a metadata management server and interfaces for accessing, storing, and manipulating user profiles.

We have developed an XML-based user profile management system that uses our update language integrated with XQuery 1.0 for manipulating profile information. We chose XML to represent all data including user profiles, because it facilitates device and application independence. Also, XML makes it possible to define constraints and complex queries in a flexible and convenient manner even in presence of heterogeneities and distribution of the underlying data.

In UbiData, a profile contains information about the user (e.g., name, id, password) and descriptions of his/her mobile devices currently in use including their capabilities (e.g., O/S, RAM, disk space). In addition, the profile also stores information about the most frequently used files owned by the user as well as how frequently they are accessed, whether they are shared or not, etc. The profile information enables UbiData to automatically maintain the up-to-dateness and consistency of frequently used files across the mobile devices that are registered with the server. An excerpt of a sample UbiData user profile is shown in Fig 3.

```
<user_profile userID="gsur" status="active">
 <user_info>
  <username>
   <firstname>Gargi</firstname>
   <lastname>Sur</lastname>
  </username>
  <password>Psdf84n87</password>
 </user_info>
 <device_list>
  <device id="laptop">Dell Inspiron4100</device>
  <device id="PDA">Compaq iPAQ 3890</device>
 </device_list>
 <file_list>
  <file>mfs://MDS/gsur/docs/todo.txt</file>
  <file>mfs://MDS/gsur/ubidata/arch.doc</file>
 </file_list>
</user_profile>
```

Figure 3. A portion of a sample user profile.

In UbiData, update operations range from simple inserts (e.g., a new profile is created when a new user initially joins UbiData) to conditional and repeated modifications involving multiple profiles and requiring the appropriate integration with XQuery (e.g., when updating information for files which are shared by multiple users). To the best of our knowledge, none of the existing XML-based query processors can handle the updates needed to support our user profile application.

We are implementing a query processor for an XML update language which is based on the soon-to-be released W3C working draft specifying update extensions for XQuery. Although still in flux, this working draft consolidates the features of several early proposals (notably, Lehti [13], Microsoft [19]). Our implementation is being carried out on top of the *Galax* XQuery 1.0 query processor and tightly integrates the new update capabilities with the existing XQuery language [1]. [22] provides details on the management of user profile data in the UbiData project.

4.4 Mobile Transactions

We are currently investigating a novel mobile transaction model in UbiData. We build on previous work on High Commit Mobile Transactions (HiCoMo) [12] and the Kangaroo mobile transaction model [4]. The goal of transaction management in UbiData is similar to the non-transactional case: increase the availability of data for query and update under disconnections. To achieve this goal, we are developing a variable, fine-grain locking scheme based on several heuristics and on mobile user access and update behavior. By distributing fine-grain locks and associating them with parts of the data items (e.g., paragraphs of a document) and by adapting the lock boundaries on reconnection, it will be possible to increase the commit rate of transactions under high degree of concurrency or prolonged periods of disconnections.

5. Conclusion

Recent technological advances in wireless networks and portable information appliances have engendered a new computing paradigm that will enable users carrying portable devices to access data and shared information services regardless of the their physical location and device used. However, the use of this impressive technology will remain limited unless we develop the necessary concepts, theory and system support that will seamlessly integrate mobility and disconnection into networked computing.

In this report we have described ongoing research in policies and algorithms in support of automatic selection, filtering, hoarding and synchronization of data and files across platforms and applications, user profile and metadata management, and support for mobile transactions. The overall goal of our UbiData project is to make mobile computing available to a broader range of users and applications by automating the hoarding of a wide variety of data from multiple heterogeneous sources into mobile devices, and to facilitate the synchronization between the mobile devices and the fixed networks where the sources reside.

Acknowledgements

The authors would like to thank the UbiData team (past and present – Allan Zhang, Young Park, Ajay Kang, Abhinav Khoshraj, Michael Lanham, and Gargi Sur) for their contributions to the research and prototype implementation of UbiData. The authors are also thankful to the National Science Foundation for supporting this research.

References

[1] S. Boag, D. Chamberlin, M. F. Fernandez, D. Florescu, J. Robie, and J. Siméon, "Xquery 1.0: An XML query language," W3C Working Draft, 15 November 2002.

[2] P. J. Braam, M. Callahan, and P. Schwan, "The intermezzo file system," www.inter-mezzo.org/docs/perlintermezzo.pdf.

[3] S. S. Chawathe and H. Garcia-Molina, "Meaningful change detection in structured data," *SIGMOD Record (ACM Special Interest Group on Management of Data)*, vol. 26, pp. 26-37, 1997.

[4] M. Dunham, A. Helal, and S. Balakrishnan, "A mobile transaction model that captures both the data and movement behavior," *ACM-Baltzer Jounal on Mobile Networks and Applications*, vol. 2, pp. 149-162, 1997.

[5] A. Helal, S. Balakrishnan, M. Dunham, and R. Li, "A survey on mobile transaction models," in *Mobile and wireless computing and communication*, A. Boukerche, Ed.: Chapman Hall/CRC, 2002, pp. 345.

[6] A. Helal, J. Hammer, A. Khushraj, and J. Zhang, "A three-tier architecture for ubiquitous data access," *First ACS/IEEE International Conference on Computer Systems and Applications*, Beirut, 2001.

[7] IBM Corp., "IBM alphaworks emerging technologies - XML utilities," Web Site, www.alphaworks.ibm.com.

[8] A. Khushraj, A. Helal, and J. Zhang, "Incremental hoarding and reintegration in mobile environments," *IEEE/IPSJ International*

Symposium on Applications and the Internet (SAINT), Nara, Japan, 2002.

[9] G. H. Kuenning, "Seer: Predictive file hoarding for disconnected mobile operation," UCLA, Los Angeles, CA, Technical Report CSD-970015, March 1997.

[10] M. Lanham, A. Kang, J. Hammer, A. Helal, and J. Wilson, "Format-independent change detection and propagation in support of mobile computing," *SBBD2002 - XVII Brazilian Symposium on Databases*, Gramado, Brazil, 2002.

[11] E. d. Lara, D. Wallach, and W. Zwaenepoel, "Position summary: Architectures for adaptation systems," *Eighth IEEE Workshop on Hot Topics in Operating Systems (HotOS-VIII)*, Schloss Elmau, Germany, 2001.

[12] M. Lee and A. Helal, "Hicomo: High commit mobile transactions," *Journal of Distributed and Parallel Databases, Kluwer Academic Publishing*, 2000.

[13] P. Lehti, "Design and implementation of a data manipulation processor for an xml query processor," Technical University of Darmstadt, Darmstadt, Germany, Diplomarbeit, August 2001.

[14] H. Lei and D. Duchamp., "An analytical approach to file prefetching," *USENIX Conference*, Anaheim, California, 1997.

[15] K. Lyytinen and Y. Yoo, "Issues and challenges in ubiquitous computing," in *Communications of the acm*, vol. 45. New York: ACM, 2002, pp. 62-96.

[16] J. P. MacDonald, "File system support for delta compression," Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, Berkeley, CA, Master thesis May 2000.

[17] J. McHugh, S. Abiteboul, R. Goldman, D. Quass, and J. Widom, "Lore: A database management system for semistructured data," *SIGMOD Record (ACM Special Interest Group on Management of Data)*, vol. 26, pp. 54-66, 1997.

[18] B. Noble, M. Satyanarayanan, D. Narayanan, J. Tilton, J. Flinn, and K. Walker, "Agile application-aware adaptation for mobility," *Operating Systems Review (ACM)*, vol. 51, pp. 276-287, 1997.

[19] M. Rys, "Proposal for an xml data modification language," Microsoft, Corp., Redmond, WA, Proposal, May 2002.

[20] A. Sahuguet, R. Hull, D. Lieuwen, and M. Xiong, "Enter once, share everywhere: User profile management in converged networks," *First Biennial Conference on Innovative Data Systems Research*, Asilomar, CA, 2003.

[21] M. Satyanarayanan, J. Kistler, P. Kumar, M. E. Okasaki, E. H. Seigel, and D. C. Steere, "Coda: A highly available file system for a distributed workstation environment," *IEEE Transactions on Computers*, vol. 39, pp. 447-459, 1990.

[22] G. M. Sur and J. Hammer, "Management of user profile information in UbiData," Dept. of CISE, University of Florida, Gainesville, FL, Technical Report TR03-001, January 2003.

[23] E. Swierk, E. Kiciman, V. Laviano, and M. Baker, "The Roma personal metadata service," presented at Third IEEE Workshop on Mobile Computing Systems and Applications, Monterey, California, 2000.

[24] SyncML Consortium, "Syncml specification v. 1.0.1," www.syncml.org/download.html.

[25] A. Tridgell and P. Mackerras, "The Rsync algorithm," Australian National University, Adelaide, Australia, Technical Report TR-CS-96-05, May 1996.

[26] H. Watchorn and P. Daly, "Word and xml: Making the 'twain meet," *XML Europe 2001, Internationales Congress Centrum (ICC)*, Berlin, Germany, 2001.

[27] J. Zhang, A. S. Helal, and J. Hammer, "Ubidata: Ubiquitous mobile file service," *Eighteenth ACM Symposium on Applied Computing*, Melbourne, FL, 2003.

[28] A. Helal, and A. Kang, "Change Detection and Re-integration for Reduced Content Documents," Submitted for publication.

[29] J. Zhang and A. Helal, "Mobility Adaptations in the UbiData System," Submitted for publication.