PosCFS: An Advanced File Management Technique for the Wearable Computing Environment

Woojoong Lee, Shine Kim, Jonghwa Shin, and Chanik Park

Department of CSE/GSIT Pohang University of Science and Technology,
San 31 Hyoja-dong Pohang, Kyungbuk, Korea,
{wjlee, postshin, shinjh00, cipark}@postech.ac.kr,
WWW home page: http://sslab.postech.ac.kr

Abstract. In the ubiquitous computing environment, wearable computers are becoming more and more important because they provide an interface between human and computers. However, there are many challenges that come from its distributed, dynamic, heterogeneous computing environment. Most of all, an wearable computer has limited I/O devices for user interface. Thus, there are many context-aware applications to provide convenience for users. But each application should suffer from finding corresponding data. So, it is very important to support context-awareness in a file service for these applications through maintaining user's activities and metadata derived from them.
In this paper, we present a context-aware file service suitable for the wearable computing environment to provide a convenient data sharing service among devices and users. To realize our work, we present an ontology based file metadata management scheme for context awareness. Finally, we implement the early prototype by using the UPnP [7] and WebDAV [8] technologies and define source ontology by OWL [9] and are still refining the system design. We believe that this work could be a predominant reference for a new generation of file management technology in the ubiquitous computing environment.

1 Introduction

In the ubiquitous computing environment, various devices are connected through a heterogeneous network and communicate with each other constantly. Therefore, an wearable computer as an interface between human and computers is becoming more and more important in the environment. Through the interface, users can collect and exchange information. However, due to the dynamic, distributed and heterogeneous computing environment, such an information network is difficult to construct.

Currently, we have various distributed file systems such as NFS [12], AFS [13], CODA [14] and Microsoft DFS [15], but they are not suitable because they work by a static mount mechanism and depend on a specific platform. Additionally, another critical problem is that wearable computers have limited I/O devices.

This affects user's ability to browse data. To provide a convenient data browsing and managing method for the user, it is very useful to support context-awareness on a file service layer.

In this paper, we present a novel distributed file service which is not only platform independent but also applicable to an ad-hoc network environment. We also present an ontology-based file metadata management scheme for context-awareness support on a file service layer. Then, we and apply them to the file service.

We summarize details of the characteristics of our file service below.

- **Heterogeneous network and platform support:** In the wearable computing environment, not only are there diverse gadgets running on specific operating systems such as Linux, MS Windows, Palm OS and so on, but also networks based on specific network protocols such as Bluetooth, IEEE 802.11b and so on.
- **Network Plug-and-Play support:** In order to provide convenient file access for users of wearable computers, a network plug-and-play and self-configuration mechanism must be supported.
- **Mobility support:** Mobility support must be provided by the file service because users wander move constantly.
- **Context-awareness support:** As we mentioned above, an wearable computer has limited I/O devices for user interface. Thus, there are many context-aware applications to provide convenience for users. But each application should suffer from finding corresponding data. So, it is very important to support context-awareness in a file service for these applications through maintaining user's activities and metadata derived from them.

In this paper, we present two use case scenarios to show how our file service works. The first scenario shows a feature of an integrated data management based on a user's profile. In this scenario, a user can manage his data in PAN with his profile. For example, if the user wants to browse his music files by a categorizing rule, 'genre-artist-album', our file service will provide a virtual file tree by the manner. See Figure 1.

The second scenario shows a case of context-aware file browsing. In this scenario, our file service provides a virtual file tree based on user's events such as a project meeting. See Figure 2.

The rest of this paper is constructed as follows. Works related to our research are described in Section 2. The key issues for context-awareness support are clarified and our proposed file service is introduced in Section 3. We describe the details of the service in Section 4. Finally, in Section 5, we present our concluding comments and describe future works.

2 Related works

In this section, we describe the state of the art of context-aware data management technologies. The GAIA's context-aware file system [1] proposed by the System

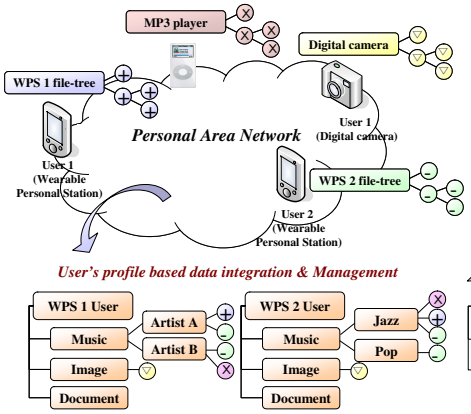

Fig. 1. Integrated data management in PAN

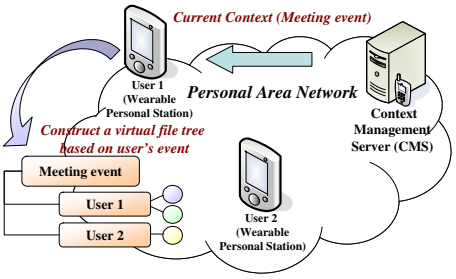

Fig. 2. File browsing based on user's event

Software Research Group at university of Illinois was the first approach which tried to adapt a context-aware concept to a file system. It not only provided a novel concept as a well-defined middleware component but also was applicable to diverse computing environments. However, it is not suitable to the wearable computing environment because of its centralized construction mechanism and lack of representations for describing file metadata. The MoGATU system [2] proposed by UMBC provides an intelligent data service. (i.e. no file I/O interface) It keeps track of a user's situation and gathers information corresponding to the situation. Finally, the information is provided for the user according to his preference. We summarize details of the characteristics of these systems below.

Table 1. Characteristics of GAIA's CFS and MoGATU system

	GAIA's CFS	Mogatu
Type of service	File service	Data service
Construction method	Centralized	Distributed
Metadata	Keyword based	Ontology based
Context-awareness	support	support

For the semantic file addressing [3], there are some previous works such as SFS [3], LISFS [4], CONNECTIONS [5] and LIFS [6]. However they are not suitable to the wearable computing environment because of the lack of the important fuctionalities such as platform independency and network plug-and-play.

3 Concept of the context-aware file service

As we mentioned before, one of the most important challenges in the wearable computing environment is to make a system correspond with a users context (situation or intention). In order to realize this, integration of context information must be achieved. Through this, a service or application can interact with users of the wearable computers.

In terms of file service, the integration is more important than other services in the environment because it lays the groundwork for realizing an intelligent file search and retrieval mechanism. In this section, we define and describe the concept of our context-aware file service and file metadata management scheme.

3.1 Concept of the context-aware file service

The purpose of our file service is to provide an intelligent file search and retrieval mechanism for a user or an application. Thus, it is very important to keep track of the user's context. However, we focus on the viewpoint of the file service interacting with it.

For example, When a user comes home, the context management server becomes aware of his entrance with RFID sensor and notifies the current context information to all nodes of Personal Area Network (PAN). If the user says, "I'd like to listen to music." the home theater generates a file request with the current context information and the user's preferences. This simple scenario clearly illustrates our goal and why a novel concept of file request is required. In the remainder of this section, we show other concepts which must be specified to realize our ideals. For more details, please refer to [16].

3.2 Ontology-based file metadata management

In this section, we present how to describe metadata for files using ontology and how to manage them. First of all, we define the source ontology for our file service using W3Cs OWL [9].

For our concept map of file metadata, please refer to [16]. It can be classified into two parts. One is a group of concepts related to the file (e.g. file name, file size, creation time, file path, and information extracted from standard file meta-tag like the ID3 of the mp3), the other is a group of concepts related to user (e.g. user profile, user preference, emotional status). We name the former passive metadata and the latter active metadata.

There is a radical difference between managing passive and active metadata.

Passive metadata: The passive metadata is extracted from file system attributes and standard file meta-tag like ID3 of the mp3 format during file creation. Then, it is assigned to the file based on our source ontology. In this paper, we implement a metadata generator which is able to extract metadata from various file formats such as mp3, ppt, doc, jpg, mpeg and so on.

Active metadata: The active metadata is constantly changed by user activities such as accessing, copying and registering files on his schedule.

For instance, when a user copies or moves a file from a source directory to a destination directory, an active metadata can be extracted from its destination directory or neighbor files - Most users usually want to manage files in the same directory, if they have a similar semantics. In the case of registering a file on a schedule, the event information can be added to the files metadata as an active metadata.

In brief, we define two types of metadata. These metadata are stored and managed in a metadata repository. One of the notable features of our work is that user activity is reflected in it

3.3 File Browsing and Searching mechanism

Since file metadata is managed as a form of ontology and can reflect user activity as stated above, we can search for a file corresponding with the current context. In this section we demonstrate how these files are retrieved.

Query language: This feature is not fully implemented currently and we are still refining it. In our implementation of prototype, we use RDQL [11]. However, as we mentioned in Section 3.2, users cannot input a semantic addresses which are fully expressed by ontology or complex query language such as RDQL. Thus, another special form to describe these semantic addresses is required. Additionally, the query language is able to specify dynamic file graph construction rule.

Below we briefly describe our query form with some examples.

a) Music (type = music & artist = Ella Fitzerald & album = *)
b) Music (type = music & artist = Ella Fitzerald)
c) Presentation (type = presentation & author = wjlee) & Event (name = project meeting or name = conference)
d) Event (person (name = wjlee) & time = * & location = home)

These query examples show the same semantics of the conventional ontology languages such as DAML+OIL or OWL. Additionally, it contains file graph construction rule. In case of a) and b) above, there is a subtle difference. The result set of files is same each other. However, the former make a file graph categorized by a rule, type = music and artist = Ella Fitzerald and her albums on run-time. The latter does not contain the albums on the graph.

Initialization step In the initialization step, each device exchanges its service description which contains person, device information and a category list being able to provide to other devices. Through this step, a file search query can be only delivered to some corresponding devices.

Dynamic file graph construction and file browsing: In this section, we present our browsing manner with the context-aware file service. Figure 3 shows the construction mechanism of a virtual file tree. The tree represents a semantic space for data browsing, which is generated on run-time by a user's intention. Using the mapping table of (node, category list), a query is sent to corresponding nodes and the results are returned to the request device by a form of a graph. Then, the results are merged based on its query form. The example in Figure 3 shows the processing of a query, Music (type = music & artist = Ella & album = *). From the construction rule in the query, the result is generated and categorized by order of the sequence, music, artist and album

This mechanism is very useful for browsing data in wearable computing environment. if we want to show all files by a view, person and device, we can just use the query form, Person (name = *) & Device (name = *).

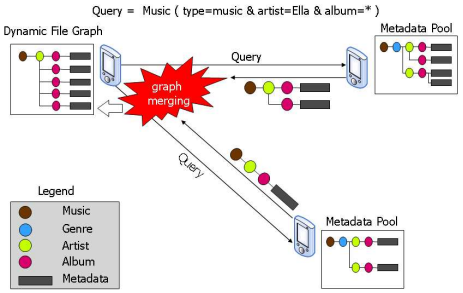

Fig. 3. Virtual file tree construction

File searching with current context: The current context gathered from sensing information, user request, and user preference can be transformed to a file request. For example, a query, Presentation (type = presentation) & Event (location = Conference Room 222) reflects the aspect.

In our approach, every context is defined by an event. In order to get files corresponding current context, the current context information must be transformed to a query form described above. Then, the query is transferred to other devices which were selected by the mapping table of (node, category list).

As we mentioned, the query processing is performed by selecting sub-category graphs and by merging them. However, if it was requested by an application, only one file should be selected. In this case, we can use the user preference to choose one from the files maintained in the merged graph and to decide the order of precedence among them.

4 Implementation of the context-aware file service

4.1 PosCFS Architecture and implementation

We implement a prototype of our file service named PosCFS on Linux. The details of our implementation environment appear in Table 2.

Table 2. Implementation environment

Platform	Sharp Zaurus SL-5500
CPU	206MHz Intel StrongARM
Memory	64MB DRAM, 16MB Flash, 512MB CF-Type SDRAM
OS	Linux Kernel 2.4.6
Library	libupnp 1.2.1, Apache mod_dav, ixml parser v1.2, librdf, sqlite3
Compiler	GNU G++ 2.95

Figure 4 shows the PosCFS Architecture. Our service is implemented as an UPnP service and WebDAV technology is used to support a platform-independent file I/O mechanism.

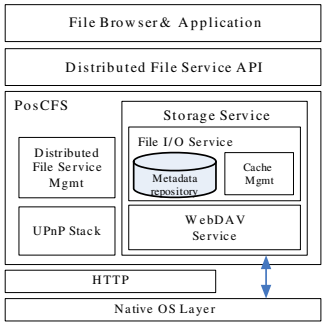

Fig. 4. PosCFS Architecture

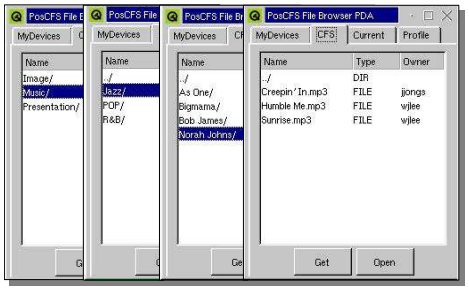

Fig. 5. PosCFS File Browser

Distributed File Service API: It describes an interface between applications and the context-aware file service.

Distributed File Service Management Component: The Distributed File Service Management Component is the key component which takes charge of system initialization, file service management and context-aware file I/O management. At the initialization step, it makes a list of UPnP devices in the PAN by

the UPnP discovery process and exchange IDs and storage service information. When a new device joins/leaves the network, it detects the device and updates information such as its UPnP device list, file service status, et al.

It also provides an interface for the File I/O service between PosCFS services as an UPnP service. Through the interface, a file request is sent to a remote file service and its results are returned.

Storage Service Component: The Storage Service Component manages low-level file I/O, file metadata and a file cache. The Storage Service consists of two sub-components, the File I/O Service and the WebDAV Service. The File I/O Service manages file metadata which is stored in local system, and provides a cache management service. The WebDAV Service provides a low-level network file I/O based on the WebDAV protocol. The low-level network file I/O is used for file transmission.

4.2 CFS Browser

We design and implement a file browser named CFS Browser as a user interface. The CFS Browser is implemented on top of Embedded/QT 2.3.7. Figure 5 shows our CFS Browser.

There are four kinds of operating modes in our CFS Browser - My Device, CFS, Current and Profile. In the 'My Device' mode, the CFS browser provides a view of files in PAN with devices which belong to a user. In this mode, a user can export or import his/her files and attach events. If files are exported, they can be shared and managed by our semantic rules. The metadata, ontology instances extracted from files, are created at that time and maintained by the CFS.

The 'CFS' mode provide a view of integrated data management based on user's profiles. See Figure 5. a user can edit his profile and manage his/her files based on the profiles.

In the current context mode, the CFS browser provides a view of files corresponding with the current context. For the details please refer to the Section 3.

5 Performance Evaluation

In this section, we present performance evaluation focused on the devices discovery time (initialization time) and query response time. For the evaluation, we use four Jaurus-SL5500 PDAs, three PCs (800Mhz Duron, 512MB RAM) and two laptop computer (1.8 GHz P4, 512 MB RAM). Figure 6 shows the result of device discovery time. From the result, we see that device discovery time is convergence to 7 seconds regardless of the number of devices. Because it depends on the expire time of the UPNP discovery process.

Figure 7 shows our results for the query processing time. In the figure, we show the increase response time as increasing the number of devices. From these evaluations, we show our approach is acceptable to real environment. However, the query processing overhead should be reduced for scalability.

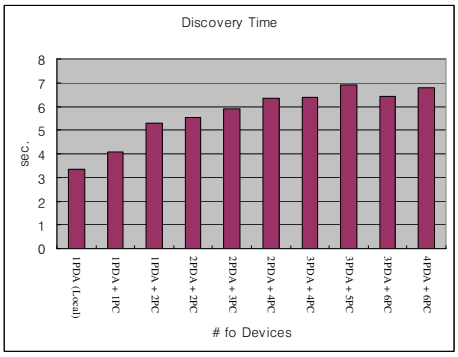

Fig. 6. Device discovery time

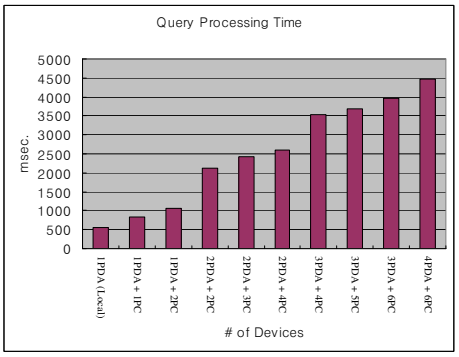

Fig. 7. Query processing time

6 Conclusions and Future works

In conclusion, we have proposed a ontology-based metadata management scheme and a context-aware file service suitable for the wearable computing environment.

Currently, there are many efforts to enhance ubiquity in this area. However, researches related to file services are just beginning despite their importance to the infrastructure of wearable computing.

As an interface for a context-aware network, our context-aware file service provides a data service which adapts to the context of the user or application through the file service interface.

We believe that our work will be a predominant reference for a new generation technology of file management in the ubiquitous computing environment. However, it is a preliminary implementation to achieve a basic concept of a context-aware file service. Therefore, our implementation has some limitations. For instance, it not only includes some restricted concepts related to a user profile and a mandatory file metadata, but also is implemented without a full consideration of efficiency. Furthermore, there are also many other important issues such as mobility support, privacy guarantee and data security. Thus, we will develop a more specific and efficient metadata management scheme which is adaptive to real-world conditions and study about the issues we mentioned above.

7 Acknowledgement

This research was supported in part by the wearable personal station project from the Ministry of Information and Communication and in part by the grant number IITA-2005-C1090-0501-001. This research was also supported in part by the BK21 program from the Ministry of Education of Korea.

References

1. Christopher K. Hess, and Roy H. Campbell: A Context-Aware Data Management System for Ubiquitous Computing Applications. In Proceedings of International Conference on Distributed Computing Systems, 2003
2. Filip Perich, Anupam Joshi, Timothy Finin, and Yelena Yesha: On Data Management In Pervasive Computing Environments. In Proceedings of IEEE Transactions on Knowledge and Data Engineering, 2004.
3. Gifford, D.K., Jouvelot , P., Sheldon, M.A., O'Toole, Jr., J.W.: Semantic File Systems, 13th ACM Symposium on Operating Systems Principles, 1991.
4. Y. Padioleau, O.Ridoux, B. Sigonneau, S. Ferre, M. Ducasse, O. Bedel and P. Cellier: LISFS: a Logical Information System as a File System, 28th International Conference on Software Engineering, 2006.
5. Craig A. Soules and Gregory R. Ganger: Connections: using context to enhance file search, 20th ACM symposium on Operating systems principles, ACM Press, pp.119-132, 2005
6. Alexander Ames, Nikhil Bobb, Scott A. Brandt, Adam Hiatt, Carlos Maltzahn, Ethan L. Miller, Alisa Neeman, and Deepa Tuteja: Richer file system metadata using links and attributes. In Proceedings of the 22nd IEEE / 13th NASA Goddard Conference on Mass Storage Systems and Technologies, Monterey, CA, April 2005.
7. UPnP Forum: UPnP: Universal Plug-and-Play, `http://www.upnp.org`
8. IETF: WebDAV: Web-based Distributed Authoring and Versioning, RFC 2518.
9. W3C: OWL Web Ontology Language, `http://www.w3.org/TR/owl-features/`
10. W3C: RDF: Resource Description Framework, `http://www.w3c.org/RDF`
11. W3C: RDQL: A Query Language for RDF, `http://www.w3.org/Submission/2004/SUBM-RDQL-20040109/`
12. IETF: Network File System (NFS) version 4, RFC 3530.
13. John H. Howard: An Overview of the Andrew File System CMU_ITC-88-062
14. Peter J. Braam: The Coda Distributed File System. Linux Journal 1998
15. Microsoft: Microsoft Distributed File System `http://www.microsoft.com/windowsserver2003/technologies/storage/dfs/default.mspx`
16. Jonghwa Shin, Woojoong Lee, Shine Kim and Chanik Park: PosCFS: A Context-aware File Service for the Wearable Computing Environment, In Proceeding of Next Generation PC International conference, 2005.