A Centralized Peer-to-Peer Architecture for Follow-me Services

Hoon Ko[1], Goreti Marreiros[1], and Jongmyung Choi[2]
[1] Knowledge Engineering & Decision Support Research Group (GECAD),
Institute of Engineering-Polytechnic of Porto (ISEP/IPP),
R. Dr. Antonio Bernardino de Almeida, 431, 4200-072, Porto, Portugal
{hko, mgt}@isep.ipp.pt
[2] Mokpo National University, Muangun, JeonNam, S. Korea,
jmchoi@mokpo.ac.kr

Abstract. Follow-me services will be popular in the near future. In this paper, we present modeling of follow-me services and a centralized peer-to-peer architecture for follow-me systems. Our modeling and architecture encompass two types of follow-me services: network style and mobile agent and have advantages including supporting the existing applications and flexibility. Our modeling and will help follow-me system stakeholders to understand the services and to build the systems efficiently.

1 Introduction

In the pervasive computing field, follow-me services are attractive because these provide services to users consistently as following them. What makes it possible is the popularity of computing devices and the advance of network technologies. These computing devices include desktop computers, PDAs, cell phones, and navigators, and they are so popular that users can utilize them almost anywhere. Furthermore, these devices start to have wireless communication facilities to communicate each other and share data. In the literatures [1][4], there are two types of follow-me services according to whether application code moves or not. First type provides services without code migration [1]. Second one is mobile agent applications with code migration [4]. Though both types of applications provide users with follow-me services, they have different concepts, requirements, technologies, and architectures. However, until now only few have paid attentions to these facts, so there have been only few research on the modeling, requirements, and architecture for follow-me systems that encompass two different approaches. These research topics should be studied for the advance of follow-me services. In this paper, we present a centralized peer-to-peer architecture that supports two types of follow-me services simultaneously. To develop the architecture, we follow these steps: identifying features of follow-me services, eliciting requirement, modeling follow-me service, and designing the architecture. We identify nine interesting features of follow-me services from the published literatures. We also elicit functional/non-functional requirements of follow-me systems. Based on the features, we model follow-me service and identify subcomponents of them. We design

a centralized peer-to-peer architecture that supports two types of follow-me services. Our work contributes to follow-me services in two folds. First, we elicit general features and requirements of follow-me services. These requirements will help analysts understand and analyze their working projects of developing follow-me services. Second, we present the modeling of follow-me services. It will be helpful for stakeholders to understand their own follow-me services. It can be used in system design and implementation phase, and it can also be used as a communication vocabulary. The rest of the paper is organized as follows. In Section 2, we discuss the works that are closely related to our work. Then we discuss features of follow-me services and modeling of those in Section 3. After that, we present our software architecture pattern for follow-me systems in Section 4, and evaluation of our work in Section 5. Finally, we reveal the conclusions of our work in Section 6.

2 Related Work

There have been studies on various topics related to follow-me service, but we focus on the researches on follow-me system framework and architecture because they have close relationship with our work. We classify those researches into two categories according to their concepts on mobile applications in follow-me systems. First group considers mobile applications as distributed applications with client-server architecture. The representative works are Bennett's [1], Roy Want's [8], and Steggles' [3]. Bennett introduces follow-me services by displaying user interface on the displays near users. It is an X-window based client-server architecture. Steggles introduces CORBA based 3-tier architecture: centralized database, a middle layer, and sensor systems. Systems in this group have limits on using local resources and they provide limited services. Second group considers mobile applications as software modules that are able to move from hosts to hosts. El-Khatib et al. [5] introduces agent architecture for follow-me applications. In their work, a software agent runs on a portable device and identifies all services available in vicinity of the user. Satoh [6] introduces a framework that consists of three parts: location information server, mobile agents, and agent host. Takashio [7] introduces a layered architecture for mobile agent framework. It is based on the Java VM plus Agent Space for java mobile agents. Agents can move from one host to other host with their contexts. This approach has a serious drawback that it cannot use the existing applications. Our work is different from the existing studies because those support only one type of services but ours supports both types of services. It means our work enable applications to be used widely in follow-me systems.

3 Follow-me Service Modeling

3.1 Follow-me system terminology

In the researches on follow-me systems, there are no common terminologies, and this causes some confusions. Therefore, we start our work with defining terminologies on follow-me systems. Fig. 1 shows the conceptual diagram of a follow-me service. When a user moves from place A to place B, a follow-me service application provides services to the user continuously or continually while his/her moving or after moving.

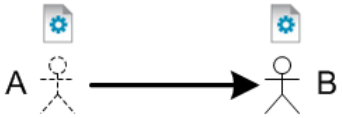

Fig. 1. Conceptual Follow-me Service

From this conceptual service, we define following terminologies.

- Origin: User's location before moving, location A in Fig. 1.
- Here: User's current location after moving, location B in Fig. 1
- Transfer: User movement from 'origin' to 'here.'
- Follow-me service: A service that is provided to users continuously or continually while his/her moving or after moving.
- Follow-me application: An application that provides follow-me services.
- Device: An instrument with computing power which follow-me applications run on.
- Peripheral: An instrument that is separated with 'devices' and has no computing power but special functions such as input or output

There have been different concepts for follow-me applications, also known as mobile applications in some literatures [1][4]. Bennett [1] argues that the interface of follow-me application "follows users rather than being carried by them." However, Scott [4] defines that follow-me applications are "programs which are able to move themselves between hosts on the network." Since these two concepts are reasonable, we combine the two definitions, and we define those terminologies.

3.2 Follow-me system features

Basically, follow-me systems follow users and provide services continually or continuously. These services have interesting features compared to the traditional systems:

- (F1) User identification: Follow-me systems identify users, and provide their own services whiling moving or after moving.
- (F2) Continual/Continuous service: Follow-me systems track users and application status and provide services continually or continuously.

- (F3) Least effort for services: Users can use the services while moving or after moving without complex setting for the services such as saving data or applications on a server.
- (F4) Tracking users: Follow-me systems grasp user's current location, and his/her arrival and leave.
- (F5) Movement of application code or data: To provide services to moving users, the follow-me application should be able to move around from hosts to hosts on the network. Sometimes, they need to move data from hosts to hosts following users.

We can get more features by looking into follow-me service categories. We can classify follow-me services into three groups according to the connectivity to outside of a system: documentation, network, and entertainment. Document services include editing documents, writing programs, and drawing Fig. s. These services have close relationship with file systems and operating systems. If the work is saved properly, it is accessible any place, so it is rather static and session-less. It needs some data format conversion capabilities for different platforms. For example, a user left his office after working with MS word, and he/she wants to finish the work at home with open office. In this case, a follow-me system should be able to convert the file from MS word format to open office word format.

- (F6) Data Format Transparency: Follow-me system can provide services data format independently.

Network services are applications running connected to the network. It includes web surfing, conferencing, and internet shopping. According to the advance of web technologies, many business applications have been transformed into web applications, so that there are many applications in this type. These services require session management that enables users to use services without breaking sessions.

- (F7) Session: Follow-me system can sustain network sessions.

Entertainment services include listening music, seeing movies, and watching TVs. These services use external peripheral devices such as speakers or/and screens. In home, these services are connected to the home theater system, so they use audio speakers and large TV screens.

- (F8) Device Transparency: Any computing device around a user can provide services to him/her.
- (F9) Adapting services to available resources: Follow-me services utilize available resources around users to provide services. In this process, they may change data format, display format, resolution, or QoS

3.3 Requirements of Follow-me services

Follow-me systems in literature have their own aims: making application mobile [1], improving accessibility of information [2], and providing services without break during a user's moving around [5]. In this work, our aim for follow-me systems is to provide users with continual or continuous services with location, device, and data format transparency. These three kinds of transparency mean that a user can use the services continually or continuously while/after moving through different computing

devices around him/her. Specific follow-me systems have their own specific requirements, but we can draw some common requirements for general follow-me systems. These requirements should meet the features described above. First, the functional requirements are following:

- (R1) Identification of User Location: Follow-me systems should grasp user's location and notice his/her arrival and leave.
- (R2) User Identification: The systems should identify users and keep their information about service usage.
- (R3) Service migration: The systems should follow user movement.
- (R4) Data format conversion: The systems should convert data in a format into data in other format if necessary.
- (R5) Network enabled: The systems should be connected to network whether wire or wireless.
- (R6) Auto configuration: Follow-me services should be able to configure their configuration automatically. They should be able to adopt their configuration according to the available resources.
- (R7) Demand Loading: Follow-me services move from hosts to hosts according to users' movement if necessary. Therefore, the applications should be able to be loaded dynamically on demand.

Nonfunctional requirements are very important for determining system architecture. Nonfunctional requirements are following:

- (NR1) Encompassing existing applications: For popularity, follow-me service should encompass the existing applications. Therefore the systems should provide follow-me services for the existing applications without developing or modifying the existing systems or the existing applications.
- (NR2) Code reuse: Follow-me service should be designed to increase the reusability.
- (NR3) Platform Independence: Follow-me services should be able to run on multiple platforms. The applications should be platform independent.
- (NR4) Standard Protocol: Follow-me services should use standard protocol for communication. These communication facilities are needed for application code and data migration.
- (NR5) Adaptability: It should be adaptable enough to adapt its services according to the available resources surrounding users.
- (NR6) Flexibility: Follow-me service should be flexible enough to change its configuration according to its surrounding conditions. For example, if it runs on mobile device, it should be able to change its contents to fit on small screen.

3.4 Follow-me services scenarios and modeling

In this section, we model two types of follow-me services. The first one is client/server style, and there is a centralized server for services. Fig. 2 shows how it works on user movement. A user works on A devices and he/she moves from A to B. If the user leaves A, then A notices user's movement and saves data in the server, C. On arrival, B notices the user's location, and gets data from the server for the user. B

creates processes for stored in the server and starts services. The other is peer-to-peer style, and it uses mobile agent. When a user moves from A to B, then the applications also move from A to B. Fig. 3 shows this type of services. The applications are developed with mobile agent technologies.

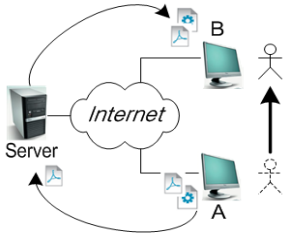

Fig. 1. Client/Server Style

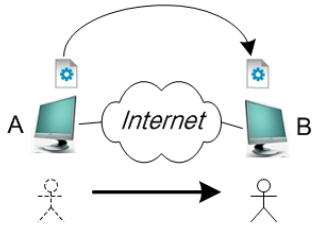

Fig. 2. Mobile Agent Style

The computing device that supports follow-me service consists of five subcomponents as shown in Fig. 4: peripheral platform, device platform, software runtime, and services.

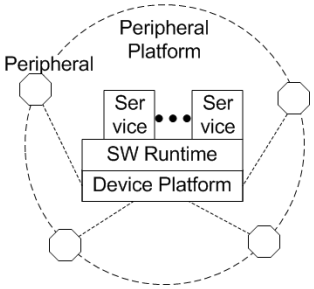

Fig. 4. A Follow-me Service System

Peripheral platform is the composition of peripherals and a device platform for a follow-me service. Peripheral platform may be complex when entertainment and game services are used, because these services utilize external peripherals such as display system, audio system, and camera. The peripheral are connected to the device with wire or wireless network. Furthermore, the composition and connection can be established in ad-hoc manner with users' least efforts or no efforts. Device platform is the device hardware plus the operating system runs on it. It can be mobile cell phone, desktop computer, or car navigation. It must be able to communicate with peripheral platform and it has a sensor to identify user and to note user's arrival and leave. Furthermore, it has some sensors to identify users and track user movement. Software runtime is a software platform that manages the lifecycle of services (create, run, destroy, and schedule). It should be able to scan and identify available physical peripheral devices. Furthermore, it negotiates service quality vs. the facility of physical devices. Service is a follow-me service. It can be a process of follow-me application of type 2 or a process of thin-client presentation application of type 1. It contains application data. We can divide a follow-me service into three blocks conceptually: computation, application data, and user interface. Computation is the program code that implements business logic or service algorithm. Application data mean the data that computation needs for proper services to users. For music service, the application data

are music files. User interface consists of input and output. Keyboard, mouse, and microphone are examples of input device. Display and speaker are examples of output. Fig. 5 shows that three parts of follow-me services.

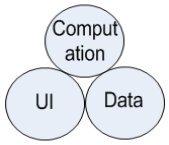

Fig. 5. Three Components of a Follow-me Service

In follow-me systems, we can think of four usage scenarios. For four scenarios, the systems behave differently and they require different technologies.

Scenario 1. *Mr. X works using application A, B, and C on his computer in his office. And he goes home, where there is a home computer in which there are A, B, and C applications. In this case, he can continue his work if the application data for A, B, and C are available. Therefore, it is the application data what moves with him.* The first scenario is the case where same applications are installed both devices in 'origin' and 'here.' In this case, the device in 'here' has its own computation and UI, and it gets the data from the device in 'origin'.	Scenario 2. *Mr. X works using application A, B, and C in his office, and he moves to Jack's office, where there is a computer in which there are AA, BB, and CC applications. These applications are similar to A, B, and C respectively, but their data format are different from X's applications. In this case, he can continue his work if the application data for A, B, and C are transformed to suitable formats for AA, BB, and CC applications. Therefore, the data transformation is required for follow-me services.* The second scenario is the case where two devices have similar applications for follow-me services, but they don't support same data format. In this case, the device in 'here' has its own computation and UI, and it gets the transformed data from some server which takes care of data conversion.
Scenario 3. *Mr. X works using application A, B, and C on his computer in his office. And he gets in his car, where there is a navigation that has computing power. He wants to watch the result done at his office in his car. But, the navigator doesn't have the A application. In this case, the A application should move from his computer to his navigator. Therefore, it is the application itself with data what moves*	Scenario 4. *Mr. X watches his company's advertisement movie for marketing on his computer at work. He wants to watch the movie continually at home. His home is equipped with home theater system. The movie displays on TV screen, the sound plays on his audio speakers, and he controls the movie play with his mobile phone. In this case, the computing device can set up ad-hoc computing system by*

with him. The third scenario is the case where the current device doesn't have some applications, so that the applications are migrated from the device in 'origin'. The current device gets the application code and the data from the device in 'origin'.	*combining available resources and devices.* The fourth scenario is the case where the current device orchestrates the surrounding peripherals. In this case, the current device has its own computation and data, and it sends UI to the surrounding peripherals.

4 Follow-me System Architecture

Software architecture should be designed to meet requirements of the application. Since there are two types of follow-me services and both of them have its own advantages and limits, we design software architecture to meet two types of follow-me services at the same time. It is centralized peer-to-peer architecture as shown in Fig. 6. There is a centralized server, and it keeps the status of follow-me services and user information. It also stores application data and application code for migration of data and code. Sometimes, it converts data format according to the request of devices. Each computing device identifies a user using its sensors, and it gets user's last status from the central server. If it finds out the origin device, it requests the data from the last device. At this step, software runtime determines which scenario is suitable for this service. If it follows scenario 1 and 2 mentioned in Section 3.4, it requests only application data. However, if it follows scenario 3, it starts application migration process. Anyway if it receives data, it creates new processes and starts services for the user. If the user leaves, it notifies the server about the current status, and sends data to the server to save.

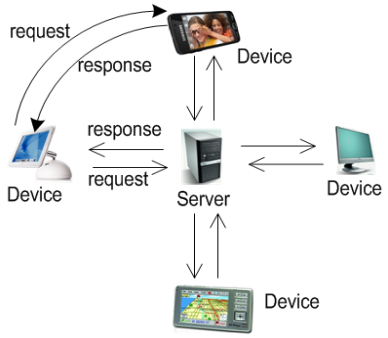

Fig. 6. Centralized Peer-to-peer Architecture for Follow-me Services

A computing device follows a layered architecture as shown in Fig. 7. In the hardware layer, there are peripheral devices and sensors for detecting user location. In the runtime layer, there are several sub-components: process management, resource management, managing peripheral devices, and detecting user location sub-component.

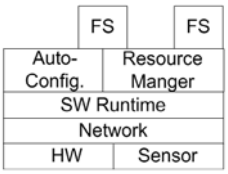

Fig. 7. Layered Architecture of a Computing Device

5 Evaluation

Our centralized peer-to-peer architecture is suitable for meeting requirements of follow-me systems. It meets functional requirements for two types of follow-me services. It also meets non-functional requirements such as supporting the existing applications, platform independence, adaptability, and mobility. TABLE 1 shows its evaluation for nonfunctional requirements.

TABLE 1. Evaluation of Our Work

	Steggles [3]	ElKhatib [5]	Satoh [6]	Takashio [7]	Our Work
Existing App.	△	X	X	X	O
Plt Independ.	△	X	X	X	O
Adaptability	X	X	X	X	O
Mobility	X	O	O	O	O

As shown in TABLE 1, it can get the popularity of follow-me services without modification of the applications or with the least modification by using the existing applications. Furthermore, by allowing code migration, it takes the advantages of mobility and flexibility. Server-based data conversion also supports adaptability and flexibility.

6 Conclusions

In ubiquitous computing environments, users will want to use their services continually or continuously regardless their locations and computing devices. For these needs, follow-me services are very important to ubiquitous computing system designers and developers. In this paper, we introduced interesting features and requirements of follow-me services, and we proposed follow-me system modeling and a centralized peer-to-peer architecture for follow-me systems. Our modeling and architecture meet the two concepts on follow-me services and they meet supporting the existing applications, platform independence, flexibility, adaptability, and mobility. Our work will help system architects, and designers to understand and design follow-me systems.

Acknowledgement

This work is supported by FEDER Funds through the "Programa Operacional Factores de Competitividade - COMPETE" program and by National Funds through FCT "Fundação para a Ciência e a T ecnologia" under the project: FCOMP-01-0124-FEDER-PEst-OE/EEI/UI0760/2011.

References

1. Frazer Bennett, Tristan Richardson, and Andy Harter, "Teleporting – Making Applications Mobile," *Proc. of the 1st Workshop on Mobile Computing Systems and Applications*, pp. 82-84, 1994.
2. William Gibson, and Mike Bursell, *Follow Me White Paper*, APM Ltd., 1996.
3. Pete Steggles, Paul Webster, and Andy Harter, "The Implementation of a Distributed Framework to support 'Follow Me' Applications," *Proc. of the Int'l Conf. on Parallel and Distributed Processing Technique and Applications*, pp. 1381-1388, 1998.
4. David Scott, Alastair Beresford, and Alan Mycroft, "Spatial Policies for Sentient Mobile Applications," *Proc. of the 4th International Workshop on Policies for Distributed Systems and Networks*, IEEE, 2003.
5. K. El-Khatib, N. Hadibi, and G. v. Bochmann, "Support for Personal and Service Mobility in Ubiquitous Computing Environments," *LNCS 2790*, Springer, pp. 1046-1055, 2004.
6. Ichiro Satoh, "Location-based Services in Ubiquitous Computing Environments," *Int'l Journal on Digital Libraries*, Springer, Vol. 6, No. 3, pp. 280-291, Jun., 2006.
7. Kazunori Takashio, Gakuya Seoda, and Hideyuki Tokuda, "A Mobile Agent Framework for Follow-Me Applications in Ubiquitous Computing Environment," *Proc. of Distributed Computing Systems Workshop*, IEEE, pp. 202-207, 2001.
8. Roy Want, Andy Hopper, Veronica Falcão, and Jonathan Gibbons, "The Active Badge location system". *ACM Transactions on Information Systems*, Vol. 10, No. 1, pp. 91-102, Jan., 1992.