Gaia Mobility: Extending Active Space Boundaries to Everyday Devices

James Bresler *Jalal Al-Muhtadi* *Roy Campbell*

Department of Computer Science,
University of Illinois at Urbana-Champaign,
{jbresler, almuhtad, rhc}@uiuc.edu

Abstract

In this paper we introduce an architecture for extending the reach of ubiquitous computing environments, or active spaces, to resource-stripped mobile devices, cell phones, and wearable computers. The architecture allows lightweight output and/or input elements of ubiquitous applications to run on the mobile devices, allowing users to interact with active spaces remotely while using minimal processing, power, and network bandwidth.

Keywords

Ubiquitous computing, mobility, Gaia, Java, Mobile phones.

1. Introduction

Ubiquitous computing is poised to have a profound effect on how humans interact with machines, physical spaces, services, everyday devices, and other humans. Ubiquitous computing envisions a world where a plethora of embedded processors, wearable computers, smart consumer devices, sensors, and digital communications are tightly coupled to form a convenient, information-rich environment. This environment merges the physical and computational infrastructures into a single integrated "active space." We define an active space as a physical space coordinated by a context-based software infrastructure that enhances the ability of mobile users to interact and configure their physical and digital environments seamlessly. Active spaces provide users with an environment where data, applications, and digital services are omnipresent and accessible anytime and anywhere. Gaia [1] is a distributed middleware that provides the core support necessary to construct general-purpose active spaces.

Gaia utilizes a layered architecture, which facilitates the abstraction of physical spaces and the various entities contained within as a single programmable entity. Figure 1 illustrates the layered Gaia infrastructure. The low-level layer consists of the basic functionality, which includes component management, resource discovery, security, and other core services necessary for any general-purpose distributed computing environment. We refer to this layer as Gaia's kernel. Building applications using only the low-level support is tedious and inflexible. In Gaia, we identified six patterns [2] that are needed for typical ubiquitous computing applications. These are multi-device support, user-centrism, run-time adaptation, mobility, context-awareness, and environment independence. The application layer of Gaia provides a framework that automates the identified patterns and facilitates the creation and management of ubiquitous applications and their components. The *Unified Object Bus* (UOB) was introduced [3] to support device heterogeneity and mobility. UOB provides common tools for managing components running in an active space. By porting the UOB to different platforms, it was possible to provide a unified interface to Gaia services even on heterogeneous devices. It was also possible to port the UOB host to some high-end PDA devices running MS Pocket PC or Windows™ CE, which allowed mobile users to interact with active spaces using their mobile devices. However, full scale deployment of ubiquitous computing is hindered by the need for an infrastructure of high-tech devices and networks. While processing power and digital communication are becoming cheap commodities available in many places, they are still not as pervasive as mobile phones or consumer devices. Typical mobile phones used by average users are resource-stripped devices with limited memory, power, and processing capabilities. Indeed, more powerful mobile phones or PDA–mobile phone hybrids are becoming available; however, because of their higher prices and larger sizes, they are not as wide-spread as standard resource-stripped cell phones. Currently, the UOB is too heavyweight to port to such resource-stripped mobile phones or consumer devices, and therefore, these devices cannot become native Gaia execution nodes.

In this paper we present the Gaia Proxy service, which enables resource-stripped mobile devices, cell phones, and wearable computers to utilize Gaia services, interact with active spaces, and run input and/or output elements without the extra overhead of installing large, processor-intensive components.

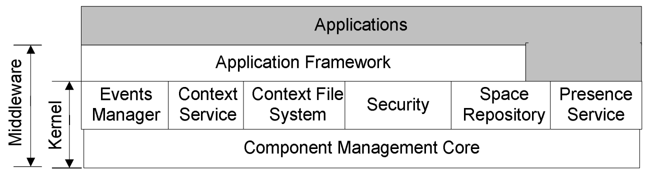

Figure 1 – Gaia Layered Architecture

The Gaia Proxy Service will enable remote mobile users to communicate and interact with active spaces; even if they were located in an environment where rich device connectivity is not available. This can be used in many business scenarios, such as providing an interface for stock traders "on the floor" and stock analysts to communicate securely using common devices such as mobile phones.

The remainder of this paper is divided as follows. Section 2 describes some simple motivating scenarios. Section 3 discusses the Gaia Proxy architecture. Section 4 evaluates the performance of the proxy. Section 5 discusses future work. Finally, Section 6 concludes.

2. Scenario

The motivating thrust behind the Gaia proxy service came after considering real-life deployment scenarios of ubiquitous computing, particularly, in classroom environments, seminars, and meetings. For example, we have developed a suite of ubiquitous applications that utilizes active spaces to create seamless and rich environments that improve the learning experience of students in a lecture or a seminar setting. These applications included a distributed slideshow that can broadcast notes and content to all participants, an automated method for recording attendance, a scribble application to add comments or annotations to the class notes, as well as several interactive features, such as allowing the instructor to poll or quiz students and allowing students to send questions to, or chat with, the instructor. Similarly, a suite of ubiquitous applications is developed to support group meetings and other scenarios. However, many of these scenarios assume that participants have access to high network bandwidth and personal devices that can act as Gaia execution nodes. In reality, most students do not carry laptops or high-end PDAs with them when going to classes. On the other hand, mobile phones are very common and truly pervasive. The Gaia Proxy Service allows students to use more common devices to utilize the services and capabilities provided by active spaces.

Furthermore, we would like to extend the services provided by active spaces so that remote users who are located at areas with no networking or computational infrastructures other than simple cellular network coverage, can still interact and join remote active spaces.

3. Gaia Proxy Architecture and Design

Gaia Mobility Support is implemented by using a proxy that sits between Gaia and mobile clients. The proxy is responsible for encoding and delivering messages to devices that cannot directly communicate with the active space. The proxy provides mobile clients access to some of the low-level services of Gaia. Mobile clients and their users are authenticated to Gaia through the proxy. By utilizing Gaia's distributed PAM (Pluggable Authentication Modules) architecture [4] it is possible to support a lightweight authentication mechanism between the mobile client and the proxy, then allowing the proxy to authenticate to the authentication service on behalf of the client. The authentication credentials in this case can be stored at the proxy. In addition, the proxy enables mobile clients to access other Gaia services like the file system, discovery and presence, etc.

In addition, the proxy enables mobile clients to utilize some of the functionality provided by Gaia's application framework. For instance, the proxy along with the mobile client can act as an input element for an application (input sensor) or as an output element for another application (presentation). The proxy changes the state of an application when an input event occurs on a mobile device. The mobile client software is designed to be very simple and can be implemented on a large variety of devices.

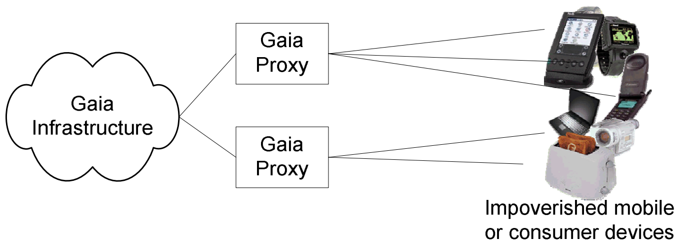

Figure 2 – Gaia Mobility Architecture

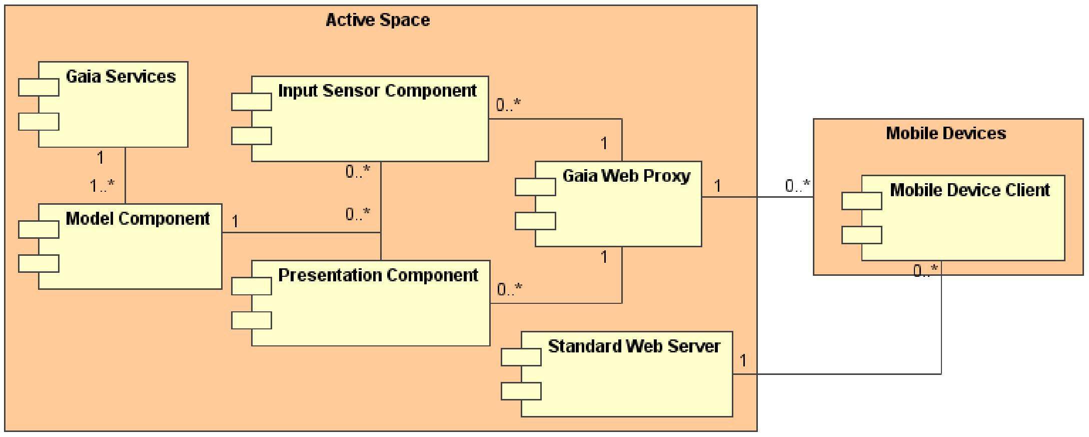

Figure 3 – Gaia Mobility Design

The Gaia Proxy is primarily responsible for bidirectional forwarding and encoding of information needed for a mobile device to interact with Gaia. To obtain this information, it is necessary for the proxy to own several Gaia application components. The proxy updates the state of mobile clients when an event occurs on one of these application components that cause the view of the application's data to change.

The Gaia Proxy needs several device parameters to provide an encoding of data that makes the mobile device software as simple as possible. To facilitate this, mobile clients are required to provide basic device information including the type of device, screen resolution and color depth. We envision scenarios where we may have hundreds of mobile devices connected to Gaia through the proxy service. For this reason, the proxy supports grouping clients into several pools based on the device parameters and every client in a pool will receive the same data-encoding message. The client pooling significantly decreases the encoding costs when there are large number of clients connected to a single application.

Because many mobile devices support a very small network bandwidth, it is possible that a client cannot receive every message. To address this, the Gaia Proxy flags every message that determines the *primary state* of a client. These messages are idempotent messages that change the state of an application but do not require any user action. If there are multiple queued messages that determine the primary state of a client, the proxy will drop all primary state messages except the message that is currently in transit and the last message added to the queue. This allows applications such as a slideshow viewer to only start sending a client the most current information, while guaranteeing clients receive all other messages.

The clients used on mobile devices consist of simple software that connects to the Gaia proxy, sends information about the application or service the user wants to access, sends basic device parameters, and waits for messages. When the client receives a message, the user's view of the application will be updated. The format of the message includes constraints specific to each service or application, allowing it to be processed easily on the mobile device. For example, the slideshow application in Gaia guarantees the device will receive messages consisting of PNG files with a resolution that is suitable for the target device. Depending on the application and security policies, the user may be able to perform events that affect the state of the application. To support this, the mobile application sends a message to the proxy when an input event occurs. The proxy will respond by notifying Gaia. Gaia then updates the state of the application, causing the mobile device to see the updated state.

The client software is designed to be implemented on a wide variety of devices, including devices that support J2ME or the .NET compact framework. The client software for communicating with the Gaia proxy is less than 200 lines of Java code and uses less then 5 KB of device space for a mobile slideshow viewer.

4. Gaia Proxy Evaluations

To evaluate the Gaia Proxy architecture, we tested a seminar class scenario where the instructor utilizes several Gaia applications to present a paper or teach a class. These applications include:

- *The slideshow application*, which uses Gaia facilities to programmatically control which displays and devices are used for displaying content, synchronize between different displays, and move content from one display to another.
- *The attendance application*, which allows the active space to record attendance automatically.
- *The poll application*, which allows the instructor to interact with the students by creating polls or multiple choice quizzes and pushing them to users' personal devices, then getting replies back.

The client software on the mobile phone is written in Java, and can be run on any device supporting Java Mobile Information Device Profile with sockets support. Furthermore, the client software is very lightweight and requires minimal space and processing footprint. We tested the scenario above with a large number of students using mixed devices, including high-end PDAs running as native Gaia nodes, mobile phones with data transmissions of about 2 Mbps (comparable to 3G mobile phones), and mobile phones with limited bandwidth and processing power (comparable to 2G GSM mobile phones). The results are shown in Table 1. Figure 4 shows how the different components of the slideshow application are distributed.

Mobile devices connecting to the Gaia Proxy can be forced to authenticate to Gaia through the authentication service so that the attendance can be recorded automatically.

Table 1: Performance measurements.
(a) Avg. time to display next slide. (b) Avg. time to load a 5 question interactive poll (c) Avg. time for input to take effect (choosing 'next' to go to next slide, or sending back poll results). All results are in seconds.

Configuration	(a)	(b)	(c)
Native Gaia node using WiFi at 11 Mbps	<1	<1	<1
Basic mobile phone, 20 MHz processor, ~ 2 Mbps connection through carrier	2	<1	1
Basic mobile phone with 20 MHz processor, 14.4 Kbps connection through carrier	12	2	3

5. Future Work

Future work includes creating thin mobile clients for other Gaia applications, as well as tools for developers to facilitate the creation of input and output elements that can be used over the proxy services. We plan to investigate the possibility of allowing the Gaia proxy to push Java applets

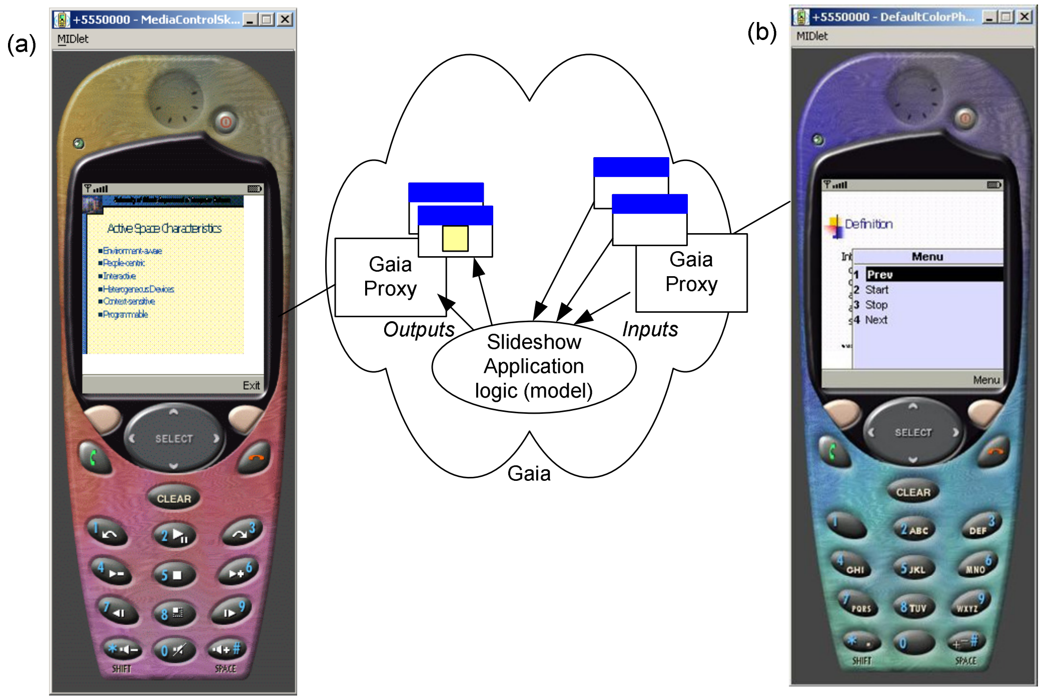
(a)
+5550000 - MediaControlSk...
MIDlet
Active Space Characteristics
Exit
(b)
+5550000 - DefaultColorPh...
MIDlet
Definition
Menu
1 Prev
2 Start
3 Stop
4 Next
Menu
Gaia Proxy
Outputs
Slideshow Application logic (model)
Inputs
Gaia Proxy
Gaia

Figure 4: (a) Output of the slideshow application on a students' devices. (b) The input element for the slideshow application as it appears on the instructor device. (Pictures are taken from J2ME simulator)

or WAP content dynamically to mobile phones. We plan to assess the usability of the system in large classroom scenarios and in distant learning scenarios.

6. Conclusion

The Gaia Proxy is a lightweight service that allows basic J2ME enabled devices to interact with active spaces. Mobile users who reside in “dumb” environments with limited or no computational or communication infrastructures can now exploit typical cellular phones and cellular phone networks for interactions with active spaces.

7. References

[1] M. Roman, C. K. Hess, R. Cerqueira, A. Ranganat, R. H. Campbell, and K. Nahrstedt, "Gaia: A Middleware Infrastructure to Enable Active Spaces," *IEEE Pervasive*, vol. 1, pp. 74-82, 2002.

[2] M. Roman and R. H. Campbell, "Providing Middleware Support for Active Space Applications," presented at ACM/IFIP/USENIX International Middleware Conference, Rio de Janeiro, Brazil, 2003.

[3] M. Roman and R. H. Campbell, "GAIA: Enabling Active Spaces," presented at 9th SIGOPS European Workshop, Kolding, Denmark, 2000.

[4] J. Al-Muhtadi, A. Ranganathan, R. Campbell, and M. D. Mickunas, "Cerberus: A Context-Aware Security Scheme for Smart Spaces," presented at the First IEEE Annual Conference on Pervasive Computing and Communications (PerCom 2003), Fort Worth, Texas, 2003.