社団法人　情報処理学会　研究報告
IPSJ SIG Technical Report

2005－UBI－8（70）
2005／6／10

AIR: Ambient/Abstract/Atmosphere-like Information Representator

Atsushi Kanda† Yoshihiro Kawahara† Hiroyuki Morikawa† Tomonori Aoyama†

The University of Tokyo†

Abstract

In everyday life, anyone is spoiled by information, which often seems annoying even though its content might be of some relative importance. In light of this, "Ambient Displays" constitute an attempt to maintain a relative user attention, where they present information through subtle changes of devices' state surrounding a user. For example, it can tell you today's weather forecast using the color changes of a light, rather than using text.

Such a system can prove its merits when a user can bind input information to an output device easily. Existing ambient display systems, however, lack flexibility when mapping information to devices, because they're developed in a more hard-wired way.

Ambient/Abstract/Atmosphere-like Information Representator(AIR) system applies RSS mechanism to provide a flexible I/O link configuration and uses dimension abstraction to generate a flexible I/O mapping* Thus, AIR aims at providing two levels of flexibility to the user. In this paper we report on its design and implementation

1 Introduction

In everyday life, people get informed from surrounding sources regardless of whether or not they focus on their content. For example, the light that penetrates a window definitely triggers a time and weather indicator. Sounds of footsteps announce another person's movements. You might also discover today's menu from the tantalizing smell of a kitchen.

Since the advent of embedded computing, pluralities of information sources are available from the environment. Indeed, our surrounding becomes a source to send information to the world.

However, this increase in information involves a risk of distraction at the user level. The environment needs to remain subtle enough for the users to preserve a high concentration level on their primary activities [1]. In this sense, User Attention plays an important role in ubiquitous environment. Nowadays, it is one of the most challenging issues in ubiquitous computing.

In relation with User Attention, Wisneski and his colleagues divided the world into two areas, namely "Foreground" and "Background" [2]. "Foreground" is the area to which a user is paying attention to, and "Background" is the area that is out of the user's attention span. They also named these related information output devices for the background area as "Ambient Displays". An ambient display presents information with a subtle change in the environment. Thus a user can get information in a casual manner without being disturbed. In this paper, we use the term "Ambient Display" as a generic name for media designed for the same purpose as the original "Ambient Display". Although some ambient displays might integrate sensors and processes for information acquisition and its presentation, we restrict the scope of "Ambient Displays" to devices which obtain information from a network.

Because "Attention" is a very personal issue, user's specifications shall be strongly considered for optimizing a system in the field of ambient displays. Thus the flexibility to accommodate users' demands is critically important in ambient display systems.

However, current ambient display systems have been developed in a more hard-wired way. In fact, ambient display system developers have focused on accurately providing information in the periphery of a user. Our research focused on the implementation of a flexible system instead of making the user ' s attention the corner stone of our study. As the concept of ambient display reaches the daily life of individuals, practical issues such as the system 's flexibility definitely becomes a requirement.

Our ultimate goal is to build a framework for

*We use the term "I/O mapping" to describe the correspondence between information and its output presentation

ambient displays, which will easily accommodate a user's demands. During the initial stage of the study, we focused on two levels of flexibility.

(1) A flexible link configuration between the information source[1] and the output device
(2) A flexible I/O mapping

In this paper, we present the design and the implementation of our ambient display system, which allows users to personalize ambient displays and map multiple information sources. The prototype is named Ambient/Abstract/Atmosphere-like Information Representator (AIR).

In Section2, we introduce related works. Then in Section3 we discuss the requirements of an ambient display systems. Section4 illustrates AIR's architecture, and Section5 describes AIR's prototype. Finally we conclude in Section6 with our future intentions.

2 Related works

In this section, we introduce several related works and discuss their flexibility issues.

2.1 ambientROOM

AmbientROOM [2] presents an environment where a user is surrounded by various ambient displays. For instance, the ambient room could accommodate a lamp reflecting water producing rippling shadows on the ceiling, or a projector creating patterns of illuminated patches onto an inner wall.

Unfortunately, AmbinetROOM has been designed on the basis that the link between the information source and the ambient display never changes. Changes in either the information source or the ambient display require an entire reconfiguration of the system.

2.2 AROMA

By abstracting the sensors' information, AROMA [3] shows the potential of converting data across several media. Its prototype implementation depicts information from microphones, video cameras and several sensors being output via either the sounds on speakers, or the temperature changes of peltier elements, or the rotations of a merry-go-round or even the animations on a monitor.

Although AROMA has the potential to provide flexible I/O link configurations and I/O mapping, there is no consideration for flexibility at the user's level. In other words, there is no mechanism to support dynamic modifications of the system's settings.

2.3 Peripheral Display Toolkit

As for research related to ambient display development environments, Peripheral Display[2] Toolkit(PTK) [4] describes an interesting avenue. PTK is a Java library to facilitate the prototyping of ambient display systems. It allows developers of ambient displays to design systems focusing on the appropriate user attention level.

However, PTK doesn't support the development of flexible systems, which enables users to define I/O mapping by themselves.

3 Discussion on the system flexibility issues

As for our prototype, here are the scenarios we envisioned.

Senario1

Ashley is sitting home. Besides him, a lamp is glowing blue — its purpose is to announce the weather forecast for the next day. However, Ashley is more preoccupied about the reception of an important e-mail that should arrive soon or later. Thus he reconfigures the lamp so that it depicts the newly arrived e-mails.

Senario2

On Ashley's desk at his office lies a small alarm. It usually makes a subtle sound when the outside temperature exceeds 30 ℃ *or falls below 10* ℃*. However, because the summer is approaching, Ashley reconfigures the thresholds to levels of 35* ℃ *and 20* ℃.

Two different types of flexibility are included in the above application scenarios. In the first scenario, the background system has to provide a mechanism to flexibly reconfigure an information source to a novel output device. In the second scenario, the background system needs to allow the user to define I/O mappings flexibly.

[1] In this paper, we define the term "information source" as a terminal that retains the source of information available for an ambient display.

[2] Note that, in [4], the term "Peripheral Display" is used in the same sense as we used the term "Ambient Display".

3.1 Flexible I/O link configuration

With the proliferation of online information sources and the advent of ambient displays, more and more people will be eager to link an information source to an output freely.

However, current systems do not enable easy bindings between information sources and their output devices since they have been developed in a more hard-wired way. Each system has its own interface to gather and display information.

To address this issue, we should be able to describe information sources and ambient displays in a more generic perspective to facilitate their integration into our surrounding.

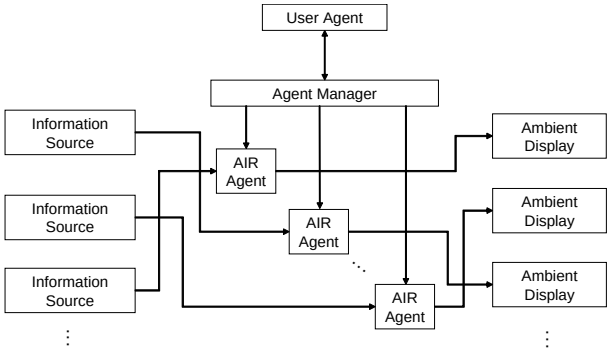

Figure 1. AIR architecture

3.2 Flexible I/O mapping

Even if an ambient display represents the abstraction of information, different users might interpret information in a different way, if they are not aware of the abstraction rules.

In current systems, information is tightly associated to an output device and it makes any reconfiguration very cumbersome.

To address this challenge, one solution is to project all kinds of information into one common dimension.

3.3 Dimension

There exists a tradeoff between the amount of information a user receives and the amount of effort required to retrieve that piece of information [5, 6].

Background information is not always important to a user until he suddenly pays attention to it. In other words, background displays must remain background in order to minimize the possible distractions from foreground activities.

In light of this, decryption of detailed information tends to require more attention compared to abstracted information. In this case, it might be beneficial to abstract information so that it becomes easily understandable and thus requires less attention span from the viewer.

We defined two types of information abstraction namely, Numeric information and State information. Numeric information is information that can be described as a one-dimensional number. Examples of Numeric information are the temperature, the number of persons in particular room, the number of newly arrived e-mails.

State information is information composed of finite states. For instance, information such as the weather or the on/off state of a switch can be defined as State information. Because State information is composed of finite states, we can assign a number to each state.

When defining information as either one of the two types, we can treat it as a one-dimensional value.

4 Architecture of AIR

In this section we describe our approach to address system flexibility issues. The architecture of AIR is shown in Figure 1

The main components of the system are *Information Source*, *Ambient Display*, *AIR Agent*, *Agent Manager* and *User Agent*.

4.1 Information Source

The *Information Source* component retains data for the *Ambient Display*. Although *Information Source* contain information in a format specific (native) to AIR, we assume that data is a heterogeneous entity. Examples of the non-native *Information Sources* includes WWW servers, POP servers and sensor network Sink nodes.

In response to a request from an *AIR Agent*, an *Information Source* returns either Numeric information or State information. Native *Information Source* uses a proprietary data format defined for AIR communication, while non-native *Information Source* translates data into an appropriate format via transcoder if necessary.

The definition of information in *Information Sources* components includes types (Numeric / State) and the scope of the value.

4.2 Ambient Display

The *Ambient Display* component presents information in an ambient manner. It is implemented

as either a desktop application or a daily object in the real world.

Similar to information type, representation of the information is defined in one of two ways, Numeric presentation and State presentation. Numeric presentation is a presentation that can be described as linear function, and State presentation is composed of finite states of presentation. Needless to say, each state in State presentation can be numbered.

With respect to data acquisition, *Ambient Display* doesn't fetch the information by itself. It receives the data pushed from *Agent Manager* and behaves according to the data received. However, in respond to a request, the *Ambient Display* component is able to return Meta information such as the presentation definition.

4.3 AIR Agent

The *AIR Agent* acts as user proxy. It acquires information from an *Information Source*, translates the acquired value according to the mapping given by the user and sends it to the *Ambient Display*. Each *AIR Agent* contains information about an association between an *Information Source* and an *Ambient Display*. The information includes a list of information provided by the *Information Source*, the output capability of the *Ambient Display*, and mapping information provided by a user.

The *AIR Agent* processes the given information as follows.

(1) Obtain information from the *Information Source*
(2) If the information type is Numeric, use the value as it is. If the information type is State information, number it so that each data component can be treated as one dimensional value.
(3) Translate the input and output value based on the user-provided Mapping
(4) If the output information is of type State, one dimensional values are then translated to strings (function name) that represents the given state
(5) Send an execute command to the *Ambient Display*

4.4 Agent Manager & User Agent

The *Agent Manager* is responsible for creation, configuration, activation, and deactivation of an *AIR Agent*. The *User Agent* is the AIR system's user interface. Users access to the *Agent Manager* through the *User Agent* to control *AIR Agents*. The primary role of the *User Agent* is to provide an interface which enables a user to arrange the *AIR Agent* easily and instinctually.

5 Prototype implementation

Based on the design described in Section4, we are now going ahead with prototyping. In this section, we report on the details of a prototype implementation.

5.1 Protocol

Recently, RSS (RDF Site Summary) has become a common format for describing Site summaries of websites, news headlines, entry contents of weblogs, etc. RSS is XML-based and allows networked computers to easily exchange website title, content, and Meta information. As a result, automatic content retrieval tools can be implemented by simply specifying the target URL of RSS.

We apply this mechanism to our system. By assigning a unique URL to an *Information Source* and *Ambient Display*, a user can link the output of the *Information Source* and the output of the *Ambient Display* by URL. In addition, both informational data and Meta data such as the information definition and output method can be described in the same XML content.

5.2 Information Source

As the WWW is the largest information repository available, we decided to use HTTP for the communication protocol of *Information Sources*. However, because most WWW information is not encoded in XML format, we transcode the information by way of the information trancoding server scripting system WeBee [7].

Figure 2 shows the example data described in AIR data format. Meta data such as definition and description are described in the channel element and actual data is described in the item element. Information is defined in the definition element. In this example, Numeric information is defined and scope of the variable (-50 to 50) is described. In case of State information, state names are listed in the description.

In the current implementation, weather forecast, air temperature, stock information, and presence information provided by MSN Messenger [8] is implemented as *Information Sources*.

5.3 Ambient Display

In the current AIR system, each *Ambient Display* implements HTTP server and waits for execu-

```
<?xml version="1.0" encoding="utf-8" ?>
<RDF>
  <channel>
    <title>Temperature in Tokyo</title>
    <link>
      http://air.h.mlab.t.u-tokyo.ac.jp/IS/
        Temperature?area=tokyo&type=high
    </link>
    <description>
      Tommorow's highest temperature in Tokyo
    </description>
    <language>en</language>
    <definition>
      <numeric unit="℃">
        <lowerLimit>-50</lowerLimit>
        <upperLimit>50</upperLimit>
      </numeric>
    </definition>
    <updatePeriod>daily</updatePeriod>
    <updateFrequency>3</updateFrequency>
  </channel>
  <item>
    <value>26</value>
    <date>2005-04-01T12:00:00+09:00</date>
  </item>
</RDF>
```

Figure 2. Description of information data

tion command that come by way of HTTP GET requests. However, we are considering implementing an RPC mechanism other than HTTP because strict HTTP functionality is not necessary.

Figure 3 presents a Meta data example of an *Ambient Display*. The data format of the *Ambient Display* differs from the one of *Information Source* in two elements: "methodName" and "paramName. '' These elements are used for specifying the actuation method of the *Ambient Display*. For instance, an HTTP request for requesting a "WashFace" method for this *Ambient Display* is described in Figure 4.

Currently, several *Ambient Display* functionalities are implemented on both the AIBO [9] and desktop applications. The AIBO performs various actions when it receives commands through a IEEE 802.11 wireless LAN interface. (See Figure 5)

```
<?xml version="1.0" encoding="utf-8" ?>
<RDF>
  <channel>
    <title>AIBO Motion</title>
    <link>
      http://aibo.k.mlab.t.u-tokyo.ac.jp/
    </link>
    <description>AIBO's motion</description>
    <language>en</language>
    <methodName>AIBO</methodName>
    <paramName>Motion</paramName>
    <definition>
      <states>
        <state>WashFace</state>
        <state>RobotDance</state>
        <state>Shudder</state>
                  ⋮
      </states>
    </definition>
  </channel>
</RDF>
```

Figure 3. Description of Ambient Display

5.4 Mapping Server

The *Agent Manager* works as a program in the Mapping Server. When it receives a request to create a new *AIR Agent*, it generates an Agent thread, which is equivalent to an *AIR Agent*, and executes it. In reconfiguration of *AIR Agent*, the *Agent Manager* suspends the corresponding Agent thread and updates its configuration data.

5.5 User Agent

The *User Agent* implemented as a desktop application. The *Mapping Server* and *User Agent* communicate over a network via TCP.

In the agent program, users can draw a curve on the Input/Output graph allowing them to describe the correspondence between the input information and output representation freely. Figure 6 is a screenshot of Mapping Form. By adopting intuitive and flexible representation inherent in the graph description, this system accommodates complex abstraction.

$$\text{http://}\underbrace{\text{aibo.k.mlab.t.u-tokyo.ac.jp}}_{hostname}/\overbrace{\text{AIBO}}^{methodName}?\underbrace{\text{Motion}}_{paramName}=\overbrace{\text{WashFace}}^{parameter}$$

Figure 4. Request URL for Ambient Display

Figure 5. AIBO action (shuddering)

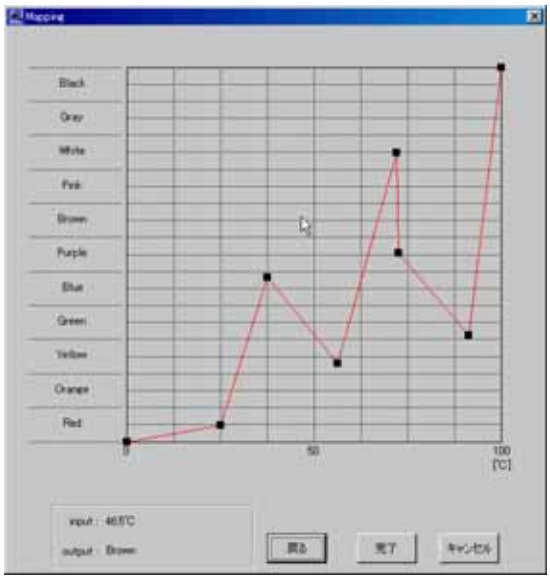

Figure 6. Mapping Form

6 Conclusion

In this paper, we presented the design and the implementation of an ambient display system AIR. Unlike existing ambient display systems, AIR allows a user to flexibly associate any information sources with any ambient displays. In AIR system, an information source on the network and an ambient display are identified by URL. This URL is used for configuring the link between the information source and the display. Moreover, AIR allows a user to configure the I/O mapping by simply editing "I/O curve" so that a user can determine "how to display" on the ambient display.

As a future work, we will demonstrate our system in the real working environment to search for further technical challenges.

Acknowledgment

This work is supported by Ministry of Public Management, Home Affairs, Posts and Telecommunications.

References

[1] Mark Wesier and John Seely Brown, "The Coming Age of Calm Technology," http://www.ubiq.com/hypertext/weiser/acmfuture2endnote.htm , October 1996

[2] Craig Wisneski, Hiroshi Ishii, Andrew Dahley, Matt Gorbet, Scott Brave, Brygg Ullmer and Paul Yarin, "Ambient Displays: Turning Architectural Space into an Interface between People and Digital Information," In Proceedings of CoBuild, February 1998

[3] Elin Rønby Pedersen and Tomas Sokoler, "AROMA: abstract representation of presence supporting mutual awareness," In Proceedings of CHI, March 1997

[4] Tara Matthews, Anind K. Dey, Jennifer Mankoff, Scott Carter and Tye Rattenbury, "A Toolkit for Managing User Attention in Peripheral Displays," In Proceedings of UIST, October 2004

[5] Scott E. Hudson and Ian Smith, "Techniques for addressing fundamental privacy and disruption tradeoffs in awareness support systems," In Proceedings of CSCW, 1996

[6] Paul P. Maglio and Christopher S. Campbell, "Tradeoffs in Displaying Peripheral Information," In Proceedings of CHI, April 2000

[7] WebBee, http://webbee.eecs.umich.edu/

[8] MSN Messenger, http://messenger.msn.co.jp/

[9] SONY AIBO, http://www.sony.net/Products/aibo/