Abstract—A pervasive computing system has to be context-aware because it cannot distract users. However, user’s context information can be quite rich and unpredictable. Therefore, how to design an extensible protocol to achieve these requirements and implement a context-aware Internet service is very important.

In this paper, we propose a SIP-compatible protocol for context-aware services, and this protocol is easy to understand, implement, and deploy. We also demonstrate some scenarios about this protocol to show transactions between client and server as well as behaviors of proxy/redirect servers. Finally, our projects implemented by this protocol are introduced.

This study also provides a further insight on the issues of context-aware concept in pervasive computing, and demonstrates a referable methodology to propose and implement a context-aware framework in pervasive computing, which can promote the technology of future Internet.

Keywords- Context-aware services; Session Initiation Protocol; Session Description Protocol; Pervasive computing; Smart appliance; CASP

I. INTRODUCTION

Future environments will be saturated with computing and communication capability, and these pervasive computing [1] environments will offer a variety of services beyond our imagination. How to gracefully integrate the environments, services, and human users is a very important research issue, especially when the pervasive computing era is coming faster than our expectation.

In a pervasive computing environment, technologies are expected to physically and psychologically invisible; that is, users are not conscious of the existence of these technologies because human attention is the scarcest resource. In practice, minimizing user distraction and simplifying user’s interaction to the system reasonably approximate this ideal.

A large portion of interaction between user and system is that the user tells the system about his/her state and surroundings, or operating the system based on this information, such as physical location, surrounding noise level, and so on. Hence, a pervasive computing system that strives to be minimally intrusive has to be context-aware. In other words, it must modify its behavior based on the cognizance of users’ state and surroundings. It even can make decisions in a proactive fashion, anticipating user needs, without disturbing users at inopportune moments except in an emergency.

A user’s context can be quite rich, consisting of attributes such as physical location, orientation, walking/moving speed, physiological state (e.g., body temperature and heart rate), emotional state (e.g., angry, distraught, or calm), personal history, daily behavioral patterns, and so on. Surroundings also consist of many attributes, such as weather rainfall and temperature data, remaining energy of wireless devices, current time, lightness, the density of crowd, landform, and so on. A smart appliance and sensors should cognize users’ state and surroundings [2], and implicitly obtain context information through awareness. Context information can be applied to filter the flow of information from application to user, to partially solve the problem of information overload. Context information can also give additional meaning to the user’s input, which combines explicit and implicit human-computer interaction. For example, a web server of night market guiding system could know about the user’s location. It can give the user a smaller map of nearby area because of the smaller display of PocketPC. Thus, the user doesn’t need to find a small region in the overall map, especially when the user uses a small display.

However, Internet application and services diversify everyday, so we need a public interface to convey context information and communicate between several components in such a pervasive computing system. This interface should be easy to understand, and the framework should implement new context-aware services easily. This interface also needs to be extensible and configurable because of rapid deployment of Internet services. In addition, it needs to be low effort, and compatible with current systems and services.

There are some researches on the issue about how to activate context-aware services, such as context-aware packets [3]. It is a document-based approach whose packets contain context constraints and data for describing an entire service request. This approach requires intermediary network nodes route and apply the data to services, which match the embedded constraints. This approach is not easy to do service (re)negotiation because it is not based on a transaction concept. In addition, the approach is new protocol and hard to deploy.
Based on the above fact, we propose a SIP-compatible protocol, called context-aware service protocol (CASP). This protocol is based on the session initiation protocol (SIP), which is designed for Internet telephony signaling and call control [4]. It is a pure text protocol, and easy to understand as well as implement. This protocol is extensible and configurable, especially for future various context awareness services of pervasive computing.

This paper explains the concept of context-aware service. Then, CASP is introduced in section II. Components and operations of CASP are described in this section. After that, we present some scenarios in section III, including the case of proxy and redirect server. Section IV introduces related work, which is about our initial implementation of CASP. Concluding remarks is finally made in the last section.

In this study, we not only propose an extensible and configurable solution for context-aware services, but also demonstrate a referable process to propose and implement a context-aware concept in pervasive computing, which can promote the technology of future Internet.

## II. THE CONTEXT-AWARE SERVICE PROTOCOL

### A. Components in CASP

CASP is a SIP-compatible protocol. A context-aware system of CASP consists of four types of component. They are: user agent, content server, proxy server, and redirect server. A user agent is an end device that acts as someone who wants to connect to a content server for getting context-aware services, for instance, a PDA with a receiver of Global Positioning System (GPS). A content server is someone that provides services such as analysis of traffic congestion, which helps the driver to find a faster way to his/her destination. In addition to user agent and content server, CASP also has two different types of network servers: proxy and redirect server. A CASP proxy acts like a SIP proxy, but not all the same; it receives requests, transforms them to certain formats if necessary, determines which server to route these request, and then forwards the requests possibly with some modified header fields. A redirect server works in a different way. It also receives requests, but instead of forwarding them to the next-hop node, it directly replies the next-hop node’s address to the user agent, and then the user agent connects to it without helping of the redirect server. Both proxy server and redirect server should have functions to provide name resolution and user location. Fig. 1 gives an example of four components connected by the Internet.

### B. CASP commands and messages

A CASP request consists of a request line, header fields, and a message body. The various header fields contain information of user addresses, requested service types, and protocol features, etc.

In CASP, we define several methods, including INVITE, REGISTER, ACK, BYE, and CANCEL. INVITE is originally used to invite a device for a session in SIP. In CASP, INVITE is used to invite a context-aware service delivery from a server; that is, we use INVITE to request for context-aware services. For instance, a user agent may send an INVITE request with some header fields and attributes such as user’s location, favorite food, and acceptable price to a content server, which provides a restaurant-finding service to find out the most desirable restaurant near to your current location. The header fields of an INVITE request contain the addresses of source and destination, subject of the connection, connection priority, and desired features of the response, etc. The body of the request contains descriptions of the media content of a session in SIP, and Session Description Protocol (SDP) describes these descriptions [5]. SDP uses textual syntax for describing information on codes, ports, and protocols, etc. In CASP, SDP is also used in the body, but we add some extra attributes about context information. To avoid confusion with original SIP, we use the header field “Content-Type: application/casp” in all CASP examples in this paper. The REGISTER request carries messages such as server address, content supplied, service ranges, and system info to a CASP proxy/redirect server for registering where the content server is and what features it supplies. Hence CASP proxy/redirect can locate proper server according to the context information in the client’s request. ACK is used to confirm reliable message exchanges. When a user agent receives a reply of INVITE from servers, the agent will send an ACK for telling the response sender that the reply is received successfully. BYE terminates a session between two ends, and CANCEL terminates a pending request. Fig. 2 and Fig. 3 shows an INVITE and REGISTER request of the CASP. Note that the SDP attributes ‘a=address’, ‘a=favorfood’, ‘a=price’ in INVITE and ‘a=servicerange_N’, ‘a=servicerange_E’, ‘a=servicerange_W’, ‘a=servicerange_S’ in REGISTER are additionally defined.
C. Introduction to Session Description Protocol

In our CASP protocol, we use the SDP to describe the body of request. SDP is a text-based protocol and could be handled by shell script languages. It is defined to describe a session, notify an existing session, let the user join a session, and give the essential information about the session. More detailed information could be found in RFC 2327. In addition to attributes that have already been included in SDP, we add some new attributes to extend its functions. Here an example shows how these attributes work. Attributes ‘a=longitude’, ‘a=latitude’, ‘a=favorfood’, and ‘a=price’ are those we extend.

\[
v = 0 \\
o = john 2890844526 2890842807 IN IP4 140.134.25.5 \\
s = FCU food-finding service \\
a = address: 100 Wenhwa Rd., Seatwen, Taichung, Taiwan \\
a = favorfood: pizza \\
a = price: 200NTD
\]

Indicating the longitude of the user location.

\[
a = longitude: N25°03'41.5" \\
Indicating the latitude of the user location.
\]

\[
a = latitude: E121°31'44.3" \\
Indicating the latitude of the user location.
\]

\[
a = favorfood: pizza \\
Showing the food that user preferred.
\]

\[
a = price: 200NTD \\
Showing the price that user preferred.
\]

III. SOME SCENARIOS OF CASP

In this section, we give some examples of finding the most desirable restaurant that you want, to explain how our CASP protocol works with/without proxy and redirect server. Imagine that you are walking on a street filled with all kinds of restaurants. You are hungry enough but still could not decide on which restaurant to eat in. Now, you may turn on your

![User Agent](http://140.134.25.5/foodfinding/Wenhwa/800x600.htm)

![Content Server](http://140.134.25.5)

\[
INVITE casp: food@fcumarket.com.tw CASP/1.0 \\
From: john <casp:johnlee@fcu.edu.tw>; tag=124 \\
To: food <casp:food@fcumarket.com.tw> \\
Supported: CPU PIII 933 Screen 800x600 \\
a=address: 100 Wenhwa Rd., Seatwen, Taichung, Taiwan \\
a=favorfood: pizza \\
a=price: 200NTD \\
c=IN IP4 140.134.25.5
\]

\[
CASP/1.0 302 Move temporarily \\
Contact: http://140.134.25.25/foodfinding/Wenhwa/800x600.htm \\
From: john <casp:johnlee@fcu.edu.tw>; tag=124 \\
To: food <casp:food@fcumarket.com.tw> \\
Supported: CPU AthlonXP 2000+ Screen 1024x768 \\
c=IN IP4 140.134.25.25
\]

\[
ACK casp: food@fcumarket.com.tw CASP/1.0 \\
From: john <casp:johnlee@fcu.edu.tw>; tag=124 \\
To: food <casp:food@fcumarket.com.tw>
\]
notebook or PDA with a wireless network connected, login a content server that provides a food-finding service, and try to dig out which restaurant is your favorite and nearest to you. Fig.4 shows the protocol flow without proxy or redirect server.

At first, the user agent sends an INVITE request to the content server with certain header fields and SDP attributes. The INVITE request describes that the request is sent from john and ongoing to food; the user agent’s CPU is PentiumIII® running on 933MHz with 800x600 screen resolution supported; its current address is 100 Wenhwa Rd., Seatwen, Taichung, Taiwan; and the user wants to eat pizzas with acceptable price NT$200. When the content server receives such a request, it will select an appropriate restaurant for you according to your current location and favorite food with price. Then the content server reply a 302 Move temporarily request with a HTTP address for telling the user agent to connect to that web page for detailed information. Then the user agent replies an ACK to content server for confirmation and passes the HTTP URL to a web browser to view the page.

In the scenario of Fig. 4, the user knows the local content server and the location representation (road address) used by the server. Assume the user knows neither what the road address is, nor where the local content server is, but rather the longitude/latitude and global content server information. In such case, we need a proxy or redirect server to translate the longitude/latitude information to a road address and redirect the user from a global content server to a local one. Fig.5 shows the protocol flow with a proxy server.

In Fig.5, the user agent sends an INVITE request with the same header fields and SDP attributes as in Fig.4 except the longitude/latitude attributes and the invitee is changed to the global content server ‘food@market.com.tw’. When a proxy server receives the request, it will translate the longitude/latitude information to a road address and resend an INVITE request to the local content server ‘food@fcumarket.com.tw’ according to where you are. With the information conveyed in the INVITE request, the local content server chooses an appropriate web page and sends a “302 Move temporarily” response through the proxy server to tell the user agent the web page’s URL and the server info such as CPU frequency and screen resolution for further interactions.

Fig.6 is the protocol flow with a redirect server. Instead of resending an INVITE request to the content server, the redirect server translates the messages and replies a “302 Move temporarily” response with the necessary information to the user agent, then the user agent connects to the local content server directly without the involvement of redirect server.

---

**Fig. 5. A protocol flow of CASP with a proxy server**
An important thing that must be mentioned is, both in the proxy and redirect mode, each local content server shall use the REGISTER request to register its information like service ranges, content supplied, and system information before starting services.

IV. RELATED WORK AND FUTURE WORK

About this protocol, we have two projects in progress. One is the Feng-Chia night market guiding system, which has been implemented partially. The other is the advisory system of jogging route map, which is our future work, and needs to cooperate with other departments, such as Global Information System center in Feng-Chia University.

The Feng-Chia night market guiding system is a web site that introduces every stores in the night market. This web site can serve a user, based on the location information that his/her mobile device gives. This web site also has an online discussion forum, which contains not only text, but also images or video clips that users provide.

In the initial stage, we have implemented a context-aware web site based on the mechanism of CASP, and a context-aware mobile client with CASP and global position system (GPS) inside. In this stage, the web site only provides the introduction of street vendors selling local cuisine. When users
want to know about information of food nearby, the site will shows a map of surrounding area, with street vendors’ marks on it, and users can browse what they want. Users sometimes do not really know about where they are. Even they do, the location awareness of web site helps a lot to filter out unnecessary information, especially when the mobile device has a small display only.

Because of the feature of service redirection in CASP, we don’t need to modify current program of web server and web browser. We just need to implement the entities of CASP at both client and server sides. In client side, the entity needs to collect the location information from GPS, and sends it to the server side. After the server receives the information, the entity of the server side transfers the location information to the web page request of food information corresponding to the area that client resides. Then, the client entity knows about the HTTP request, and executes the browser to browse the web page. After that, the server responds the page to the client. It is noticeable that the HTTP, web server, and web client do not need to modify, and a lot of effort can be reduced.

Following stages of this project in future, we will implement a location translation proxy server, which integrates the information from GPS, the area ID of cellular system, and the location of access point, into the uniform representation. This will provide a common representation of users’ location, and hence simplify cognition of location information at various servers. We will also implement a redirect server, which can locate a proper server for each client according to client’s location. As a result, we can provide an open and extensible framework. This framework allows numerous web sites to provide food information without knowing which networking technology users employ.

The second one is the advisory system of jogging route map and under construction. It provides a proper jogging route map for users who have smart devices and want to jog. The smart device can know about information of user context in advance, such as age, gender, height, weight, fat percentage, chest/stomach/thigh measurement, and jogging history. The device also can obtain context information in a real-time manner, such as heartbeat rate, body temperature, weather temperature, current location/orientation, spent calories, jogging distance, current time, lightness, and so on. The smart device tells the server about the context information mentioned above, and continuously transmits those measurements in real-time by CASP when the user is still jogging. The jogging advisory system determines the jogging route based on not only the information that the smart device provides, but also the information of weather, landform and roadmap of where the user resides. The system also monitors the real-time measurements, e.g. heartbeat rate, and decides if the jogging speed should be faster, slower, or the advisory system suggests suspending this jogging. When the user is jogging, the server will enable smart devices to play some music whose tempo matches the jogging speed. By the way, when the system suggests the user should make a turn, it can notify the user by speaking. In addition, the smart device can show the current position and orientation on the display as well as the right way to jog.

V. CONCLUSIONS

Pervasive computing environment is expected to be the future trend of computing and communication. All the technologies in such an environment are expected to invisible. Hence, a pervasive computing system that strives to be minimally intrusive has to be context-aware. The context information should be propagated and manipulated in a common protocol, so that every component in a context-aware service can be easily developed, integrated and deployed. This paper demonstrates a framework to provide context-aware services, and this framework uses a SIP-compatible protocol that is widely used in Internet telephony.

This paper describes the context-aware services and the detail of CASP. Some scenarios of CASP are illustrated, including the proxy and redirect servers. Then, this paper talks about the related implementations for the night market guiding near the Feng-Chia University and the advisory system of jogging route map. This study also demonstrates a referable process to propose and implement a context-aware concept, which can promote the technology of future Internet.

REFERENCES