Websign: hyperlinks from a physical location to the web

Salil Pradhan, Cyril Brignone, Jun-Hong Cui,
Alan McReynolds, Mark Smith
Internet and Mobile Systems Laboratory
HP Laboratories Palo Alto

HPL-2001-140
June 11th , 2001*

augmented reality, pervasive computing, location awareness, ubiquitous computing, CoolTown

In the CoolTown research program at Hewlett Packard Laboratories, we are building Ubiquitous Computing systems by embedding and embodying web technologies into physical environments. One of the newer components of this research is Websign. By using a simple form of augmented reality, the system allows users to visualize services related to physical objects of interest. The websign system provides infrastructure not just for detecting websigns but also for creating and deploying them. In this paper we present the concept, an overview of the prototype and the algorithms used in the implementation.

Websign: hyperlinks from a physical location to the web

Salil Pradhan, Cyril Brignone, Jun-Hong Cui, Alan McReynolds, Mark Smith

Hewlett-Packard Laboratories, 1501 Page Mill Road, Palo Alto, CA 94304

Abstract: *In the CoolTown research program at Hewlett Packard Laboratories, we are building Ubiquitous Computing systems by embedding and embodying web technologies into physical environments. One of the newer components of this research is Websign. By using a simple form of augmented reality, the system allows users to visualize services related to physical objects of interest. The websign system provides infrastructure not just for detecting websigns but also for creating and deploying them. In this paper we present the concept, an overview of the prototype and the algorithms used in the implementation.*

Keywords: Augmented reality, pervasive computing, location awareness, ubiquitous computing, CoolTown.

1 Websigns link physical points to the web

Websigns are hyperlinks from physical points in space and time to web resources. A person selecting such a hyperlink is directed to the associated web resource. Websign users carry internet-enabled wireless mobile devices such as PDAs or smart phones equipped with the client software, positioning systems such as GPS[8] and a digital compass. On request, the websign-enabled mobile device connects to a service and downloads and caches XML descriptions of websigns in a wide area. This description consists of locations of websigns, their associated URLs and other control information. Users then freely move in that wide area and point their mobile devices towards physical objects of

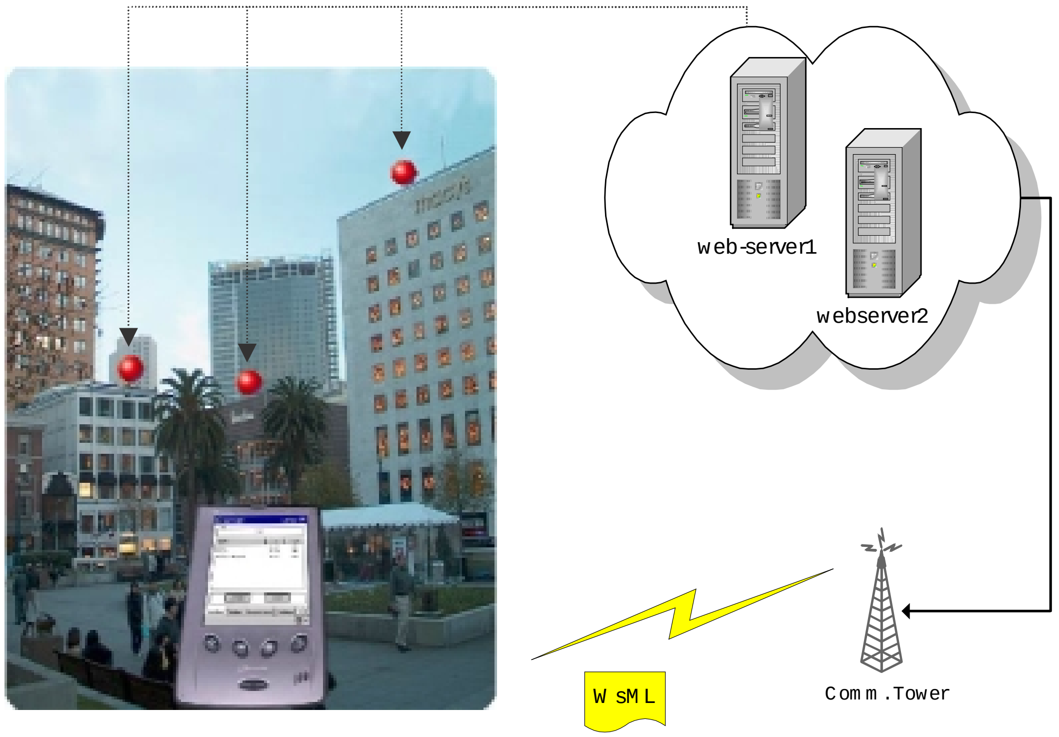


Figure 1. Detecting websigns with a wireless PDA in an urban environment: When users visit new locations, their devices can download and cache websigns for the locality. This information is then filtered based on users' precise location and heading, and the relevant websigns are displayed.

interest. Using the locally cached information, websign clients present relevant websigns from the user's vicinity. The system provides a simple form of augmented reality[2]. In its traditional form, augmented reality allows the user to see the real world superimposed with virtual objects. Users typically wear see-through Head Mounted Displays (HMD) and can see the physical environment as well as any computer-generated images. With websign, our approach is to use commonly available mobile devices, not HMDs, to help users visualize virtual objects associated with physical objects. Figure 1 shows an example of what the user's PDA will display when pointed towards the buildings in San Francisco's Union Square. The red spheres indicate positions where websigns can be detected.

To avoid overloading the user and their mobile device we need to limit the number of visible websigns. Users should only be able to experience the websigns in their *horizon*. We define a horizon as the set of all the websigns that a user "activates" by being in their range. While range is the property of individual websigns, the horizon is the temporal and spatial property of individual users.

Two types of websigns are supported, virtual beacons (v-beacons) and virtual tags (v-tags). To be linked to web resources associated with v-beacons, users are required to point their mobile device and explicitly select one v-beacon. (V-tags on the other hand are automatically triggered when the user gets in their proximity) Activated v-tags may perform an HTTP request informing an infrastructure service about a user's presence at a location. Users have control on all aspects of v-tags, including the ability to (selectively) turn them off.

Websigns may be personalized, group-targeted or universal. While personalized and group-targeted websigns are uniquely created for a receiving individual or a group of individuals, universal websigns are available for anybody who may choose to use them. Websigns may be further categorized by resource type, for example restaurants, theatres etc. This makes it easier for class specific filtering.

2 Websigns in Ubiquitous Computing

First generation mobile computing technologies typically consist of PDAs or phones integrated with web browsers and wireless networks. Examples of such systems include WAP and I-mode. These devices free users from the shackles of their desktops by allowing them untethered access to their web resources. We believe that in addition to mobility, users need a transparent linkage between the physical world around them and the web and services world it relates to.

In the CoolTown[5] research program at Hewlett Packard Laboratories, we are building Ubiquitous Computing[12] systems by embedding and embodying web technologies into physical environments. To support mobile users, we have extended web browsers and back-end infrastructure to sense physical entities in the environment and map them onto web resources. By attaching beacons, RFID tags or barcodes to physical places and by storing/associating an appropriate URI in them (and resolving the URI in the network if it is not already a URL) we create a binding between a physical place and a web resource. Users choosing to receive the signals are transparently linked to the web resource.

The current generation CoolTown beacons use infrared technology to broadcast URLs. This choice was made due to the built-in infrared transceivers in many handheld computing devices today, the stability of the IrDA protocol, and ease of implementation. On the mobile client side, the e-squirt[5] software handles the reception of the data over infrared. The beacons are directional and have an access range of about one meter. To receive a URL, users need to point their mobile device in the direction of a beacon and receive a periodic broadcast. Mechanically, CoolTown beacons are similar in design to Active Badges[11], however the application space is different. Active badges are worn by individuals and objects and are used to track their presence at a location. CoolTown beacons are used to connect mobile users to services associated with physical objects.

While we have achieved good results in our limited trials, the large-scale deployment of bridging devices, like beacons and tags, may raise scalability issues. Beacons run out of batteries, so they have to be periodically maintained. As web resources are updated, beacons have to be reconfigured. RFID tags and bar codes, while not having this problem, can only be read off very short distances. Wide-area deployment raises access-range issues. As an example: the local cafeteria serves the entire Hewlett Packard laboratories' Palo-Alto site. However, since their physical access range is limited, we would need to garnish the entire campus with beacons/barcodes directing users to the cafeteria. What is desired is a system where access range is not limited by technology but rather by context.

In order to address the deployment, scalability and access range issues we have been exploring the notion of augmenting users' reality with virtual beacons and virtual tags. In this paper we present an overview of the websign system, focusing on v-beacons. In §3 we state our design objectives, followed by an overview of the prototype in §4. Before concluding we review related work and applications for websign.

3 Design objectives

Based on our experiences with physical beacons and tags and the desire to emulate them with virtual implementations, we came up with a set of requirements for the websign system:

Responsiveness. The websign client must quickly adapt to change of direction and somewhat quickly to user mobility. In order to reduce latency, at least some part of the storage and processing must be done on the mobile device.

Robustness. The system must be resilient to loss of connectivity. Even in the sporadically connected mode, it should permit user mobility especially for short distances and not restrict adaptability to change in direction.

Deployment & scalability. The system must scale well in the number of users that can be supported as well as the number of websigns.

Open and web based. The power of a websign-like system comes from its ability to operate in an environment where websign bindings can be fetched from ubiquitous web-servers. For this reason we choose to make the system open and based on the Web infrastructure.

Inaccuracy tolerance. The better the positioning accuracy of the underlying system the more realistic the experience for the user. Ordinary GPS receivers[1] even with selective availability [8] turned off, are accurate to within a few meters. Indoors, where GPS receivers usually do not work, location accuracy will depend on the local positioning technology and can be better or worse than the figures for GPS. The websign system must therefore tolerate positioning inaccuracies and, whenever possible, compensate.

Platform heterogeneity. Clients should have reasonable computation requirements so that they can be implemented to various smart phone and PDA platforms. The communication protocols should be open to allow these diverse websign-enabled devices (and servers) to create and send websigns.

4 The websign system

The system consists of server side infrastructure to map and deliver (deploy) websigns and clients to detect them in a wide-area physical context.

Before a websign can be delivered to a client, a binding needs to be created between the physical deployment location and a semantic location[5], such as a URL. The semantics of this binding are expressed in XML and can include parameters for temporal activation, access range etc. Bindings can

[1] Differential GPS receivers offer better accuracy.

be created by any number of means, including by typing a simple XML description in the Websign Mark-up Language (WsML). WsML is usually hosted on web-servers, but peer mobile devices can also serve it, to create interesting peer-to-peer applications. Later in this section we describe our service for creating websigns.

The clients can auto-discover WsML servers or they can be manually configured. Information transmitted from an infrared beacon can be used to auto-configure clients to communicate with known servers. Auto-configuration is particularly useful in enterprise scenarios, for example, visitors and Palo Alto employees of HP Laboratories, can on entering the main building, receive an infrared message and have their PDAs configured to communicate with the websign server `wspalo-alto.hpl.hp.com`. Their PDAs can download WsML, which contains pointers to physical resources such as printers and smart conference rooms along with the associated services. The manual configuration mode is similar to the address bar of the typical browser, however, unlike a typical browser, many addresses can be entered and the content can be aggregated. This mode is useful for applications such as tourist guides, where users might want to aggregate websigns from various web sources; some of these could be from standard web based yellow pages, while other could be from someone's personal web page.

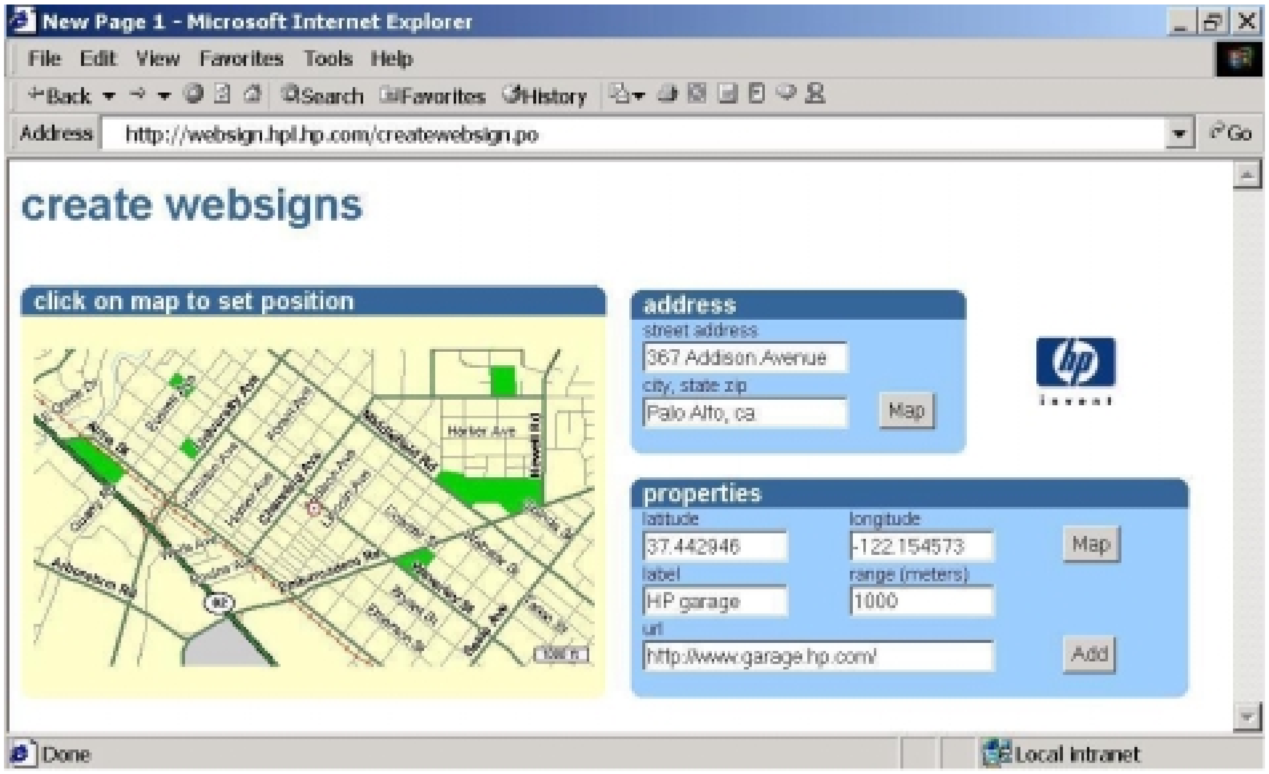
4.1 Websign Mark-up Language

Unlike beacons, tags or barcode-readers, websign-enabled devices do not actually "sense" URLs.

```
<?xml version="1.0"?>
<websign>
    <vbeacon id="ID" href="url" label="title">
                <location>
                      <lat>lat</lat>
                      <lon>lon</lon>
                      <range>range</range>
                </location>
                <time>temporal activation parameters</time>
    </vbeacon>
    <vbeacon …>
          …
    </vbeacon>
    <vtag …>
          …
    </vtag>
</websign>
```

Figure 3. A snippet using the Websign Markup Language.

Instead they present the appearance of having sensed the URLs associated with physical objects, by looking up the device's local cache. The Websign Markup Language (See the snippet in Figure 3) is our experimental XML language for describing the binding between a physical location and a semantic location[9] (represented by a URL). WsML is intended to be a compact format for transmitting this binding information over potentially low-bandwidth wireless networks. The Geographic Markup Language (GML) is another XML language for binding complex geo-spatial areas to URLs. This is a


New Page 1 - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites History
Address
Go
click on map to set position
Done
Local intranet

http://websign.hpl.hp.com/createwebsign.po

create websigns

address

street address: 367 Addison Avenue

city, state zip: Palo Alto, ca

Map

properties

latitude: 37.442946	longitude: -122.154573	Map
label: HP garage	range (meters): 1000	
url: http://www.garage.hp.com/		Add

Figure 4. Websign service.

rich language with many attributes and complex descriptions, however most of them are unnecessary for the websign system. In addition websign has additional requirements that are not in the attribute set of GML. With the websign system we desired to keep the mathematical computation in the PDA manageable and therefore we bind URLs to a simple point, with a square area around that point as the access zone. Websigns also have temporal activation parameters, which are not normally described in GML. For all these reasons we chose to define a new, simple markup language instead of using GML.

WsML is still an evolving format with optional features such as *websign-categories*; the overriding concern of keeping the format compact requires us to restrict the core language to a bare minimum.

4.2 Prototype websign creation service

The obvious choice of using the Web for the underlying backend allows us to keep the backend infrastructure both simple and open.

Our prototypical backend includes a web interface for creating outdoor websigns. Figure 4 shows the web interface that provides a geographical map as a reference for placing websigns. All one needs to do, is to enter a postal address of the general deployment location for the websign, view and select the exact location on the displayed reference map and specify rudimentary binding properties, such as a label, an access range and a URL to associate with that location. This information is then entered into a database.

Websign clients optionally post the coarse granular location of the user to a web-server. Using this location information, the information in the database and optionally the user's profile, the web server customizes, generates and serves the client WsML.

WsML could also be statically created and hosted on a web-server, but this would mean a loss of all the customization.

4.3 Websign Client

Figure 5 shows the overall architecture of the client. The positioning hardware provides the kernel with the current location as well as the direction. The horizon filter uses this location information to determine the user's exact horizon, which includes all the active[2] websigns. The horizon set is

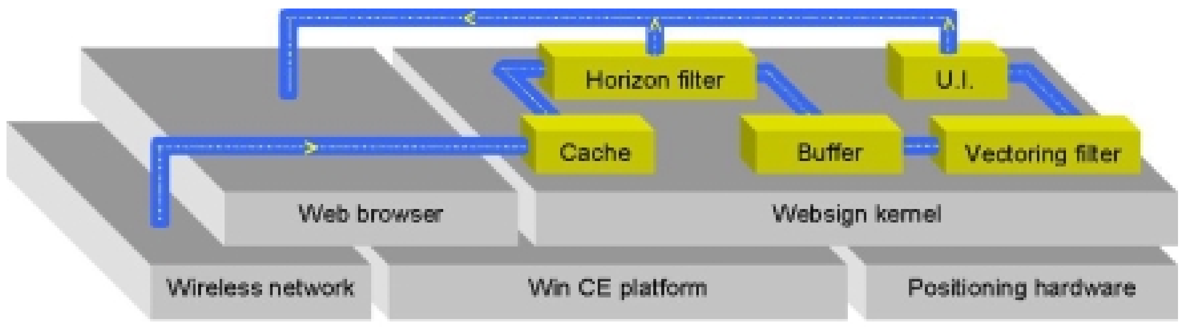


Figure 5. Client architecture.

forwarded to the vectoring filter for filtering out websigns, which are not in the users direction vector. Finally, the User Interface displays the labels of the relevant v-beacons for the user to select.

The current prototype is implemented in C++ on an HP Jornada 548 running the Pocket PC operating system, but the websign client could be implemented on any reasonable hardware platform such as the Palm or Epoch. For wireless connectivity we use 802.11B indoors and CDPD wireless modems outdoors. The prototype positioning hardware, shown in Figure 6, includes a magnetometer (sensor for digital compass), a GPS receiver and a PIC microprocessor to control the sensors and to multiplex the sensor data into a single stream. It communicates with the PDA over a standard RS232C serial link.

4.3.1 Websign kernel

The websign kernel is a client-side software component that communicates, performs power management & compass calibration and controls other features of the hardware. It also acts as a hardware abstraction layer. This is important as different hardware configuration could be used for

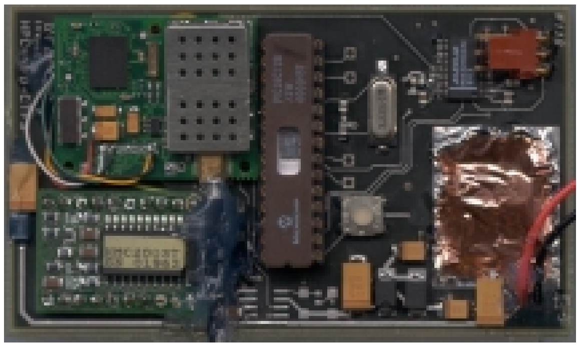


Figure 6. Prototype hardware.

[2] Range spans the user's location

different applications. In a corporate environment, there may only be a need for an indoor positioning system and as a result the GPS receiver might be safely omitted.

The other main functionality of the kernel is to keep the onboard cache refreshed with websigns. When requested by the user, the kernel queries the positioning hardware for the current location, fuzzes it (to protect user privacy) and queries web-servers. The response – WsML – is stored in the internal cache.

4.3.2 Horizon

The kernel invokes the Horizon Filter in response to changes in the user's location. The algorithm queries websigns in the cache with the current location, aggregating websigns whose range spans the current location (active websigns), into the horizon set.

4.3.3 Vectoring filter

In response to changes in the compass readings, the kernel invokes the Vectoring Filter (VF). The VF is one of the more complex client-side components but its functionality is quite simple – to filter the websigns in the direction pointed by the device and forward them to the User Interface. Given that the set of websigns it has to iterate through has already been reduced by the HC, the algorithm is quite efficient.

Before we go into further details it would be useful to define two terms:

Aperture. (a°). The aperture is the angular range through which the mobile device may be rotated while continuing to "view" a v-beacon. This is not a global value and varies per v-beacon.

Vision. (ν). This is the set of all the active websigns that fall within the cone shaped polygon formed by the aperture angle (Figure 7a).

The VF algorithm addresses three main issues:

Jitter. In the early trials of the prototype we noticed a strong screen jitter. This was mainly due to websigns on the edge of the aperture. Minor changes in the direction, typically caused by a user's unsteady hand, caused these borderline websigns to vacillate in and out of the display list. The VF corrects this aberration by artificially introducing *hysteresis*. The idea is to make it a little harder to drop a websign from the aperture once it is captured. The aperture for a websign is initially computed from a number of parameters. Once the v-beacon is displayed, a small hysteresis angle is added to the aperture. This makes the websign appear in a wider angular range. Once the websign falls out of the angular range the aperture is reset.

Depth and size perception Humans perceive depth in a seemingly two-dimensional world by interpreting binocular visual cues from the environment. Further, to a human eye, the size of an object appears to be inversely proportional to the distance. It is important for the websign client to preserve this perception of depth and size. To achieve this goal, the aperture is computed as a property of each websign and is inversely proportion to the distance between the user and that websign. As a result, websigns that are nearer to the user have a wider aperture while those further out have a smaller aperture.

Inaccuracy tolerance. The accuracy of the underlying GPS receiver is usually a few meters. While this accuracy is adequate for websigns that are further out from the user, it can be a problem for those in the user's immediate vicinity. In the early states of our prototype, we placed a websign at a bus stop, just outside HP Laboratories. We walked near the bus stop and pointed the jornada in its general direction – *no websign*. Scanning around with the device, we found the websign in the exact opposite direction. Due to positioning inaccuracies, the device's location was off by a few meters. While the

GPS claimed that it was at a given location, it could have been at any location ρ anywhere in a *circle of inaccuracy* Γ around that point. As a result, the PDA's vision v was shifted and the websign was no longer visible. Figure 7 shows the possible vision(s) that might arise due to positioning inaccuracies.

To counter this inaccuracy, the PDA needs an *extended vision* V_E. This extended vision is the union of the visions computed for the each individual point in the circle of inaccuracy.

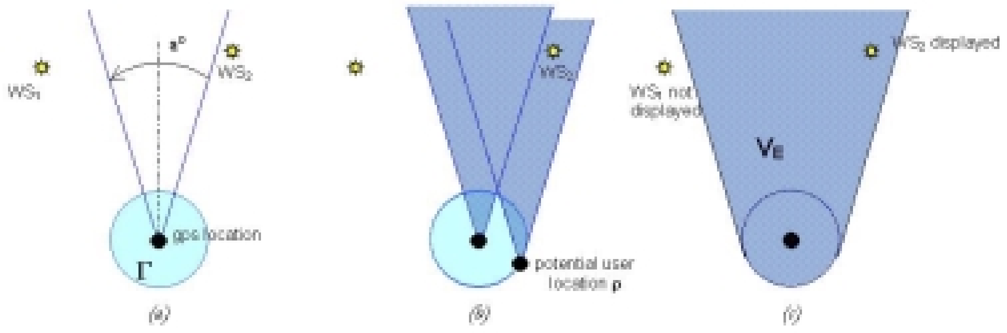


Figure 7. Extended vision. (a) Location and vision of a user as reported by gps (b) user could be anywhere in the circle of inaccuracy (c) extended vision is the union of the visions at all locations inside the circle of inaccuracy

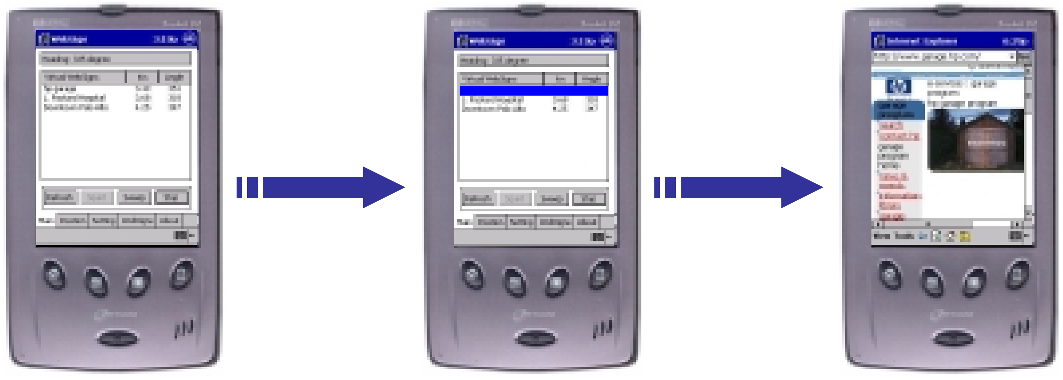
$$V_E = \bigcup \{ v(\rho) | \forall \rho \in \Gamma \}$$

4.3.4 User interface

The Vectoring filter forwards the final set of v-beacons to the User Interface (UI), which displays them on the screen. To avoid screen flicker, the result of the current iteration is discarded if it is unchanged from the previous iteration.

The UI supports two modes:

- The 'physical browsing' mode is the default mode of operation. While the button is



(a) *(b)* *(c)*

Figure 8. Display transition. (a) User point in a direction, sees websigns, (b) selects one websign (c) and the associated service is displayed.

pressed the UI continues to refresh, and display the labels of appropriate websigns. Users can scroll through the list and select the websign of interest. Clicking the label invokes the web browser with its URL (figure 8).

- In the 'casual capture' mode, the device aggregates all the websigns as they are displayed on the screen. This mode is useful for collecting websigns for future reference.

5 Related work

Cyberguide[1] was one of the first mobile context-aware tour guides. Developed at the GVU Center at Georgia Tech, the initial prototype received its positioning information from a collection of TV remote control beacons (later versions used GPS for outdoor use). The user's device used this location id to determine orientation and display the map on the display of the PDA. Having tested the Cyberguide in an enclosed environment of the GVU Open House, Abowd et al. found that information on the user's orientation was more useful than finer-grained location information. Similar systems have also been proposed and built by many others including Lancaster University's GUIDE[6] project. While websign can certainly be used for "tourist guide" kind of applications, the intended use is more general: as a digital viewing glass for seeing and interacting with services associated with physical objects pointed by the user. The websign system does not display maps but rather labels of services. Of course if desired, websigns could be deployed at popular tourist spots to show maps of the area. Websign can also enable companion applications like "virtual post-its"[3]. Such systems allow users to annotate physical locations with virtual notes, much like placing post-it notes on physical objects.

Research on smart spaces or intelligent environments has become popular at many universities and corporate research facilities. Interactive space management software such as the Web presence manager[4], aims to connect computers and computation resources to the real world and to the people who inhabit them. However, due to the nature of sensors typically used in these environments (such as beacons, tags, tracking cameras etc.), their use is usually limited to room size environments. By adding websign to the gamut of sensors, it is possible to create large-scale smart environments like "smart-theaters". Websign users pointing in the direction of a smart-theater would be connected to its service, even if they were sitting in a nearby restaurant. We note that the websign system is intended to augment the palette of sensors, not as a replacement for beacons, rfid tags or barcodes. The use of these sensors to bridge physical objects to services has been successfully demonstrated[5][10].

6 Conclusion and future work

The scalability of the websign system depends on the underlying positioning system. For an outdoor system, the GPS is the standard way to obtain positional information. GPS systems are deployed and maintained by the US military and require no maintenance from the users perspective. Few would disagree that for outdoor applications, the system is very scalable in terms of the number of users as well as websigns. For indoor applications -- where GPS systems often do not work – the system relies on a locally deployed positioning system. Appropriate choice of a local positioning technology will ensure that the system also scales well for indoor applications.

It is clear that privacy issues are important to users. In order to protect user privacy, websign clients do not divulge users' location without their authorization. Unscrupulous services may attempt to use unique (to a user as well as a location) URLs to try and associate its use with presence at the location of a v-beacon. But they cannot be sure, as the URL could be book marked and revisited without physical presence.

Bridging technologies like Websign are of value to users if they are convenient and transparent to use.

In order to support both, indoor as well as outdoor use, websign relies on a diverse set of positioning technologies. One of the usability challenges is to make this location diversity, transparent.

We believe that the current websign prototype satisfies most of the design objectives highlighted in §3. But to verify this and to test the true value and usability of the system we need to do further user studies.

7 Acknowledgements

As always, the development of the prototype turned out to be more complex than originally estimated. Fortunately, we have a broad set of engineering and research talent available at HPL. In particular, we would like to thank Bill Serra for explaining the intricacies of the Pocket PC operating system. The concept and demonstration of the URL beacons, which had a strong influence on Websign, is due to Jeff Morgan Marcos Frid, Bill Serra, and John Schettino. We also thank Patrick Scaglia, Gary Herman, Gita Gopal, Fred Kitson, Venky Krishnan, Jean Tourrilhes, Glenn Steiner, Patrick Goddi, Tim Kindberg, John Barton & John Grundbäck for helpful discussions and suggestions.

References

[1] Abowd, Gregory D., Christopher G. Atkeson, Jason Hong, Sue Long, Rob Kooper and Mike Pinkerton. Cyberguide: A Mobile Context-Aware Tour Guide. *Wireless Networks*, 3(5): 421-433, October 1997.

[2] Azuma, R. A survey of Augmented Reality. Presence: Teleoperators and Virtual Environments, pp.355-385, MIT Press, 1997.

[3] Brown, P.J. The Electronic Post-it note: A metaphor for mobile computing applications. IEEE Colloquium on Mobile Computing and Its Applications, 1995

[4] Caswell, D. and Debaty, P. Creating Web Representations for Places. Proc. of Second International Handheld and Ubiquitous Computing Symposium, HUC'2000, England, 2000.

[5] CoolTown home page. http://www.cooltown.hp.com/

[6] Davies, N., Mitchell, K., Cheverst, K., Blair, G. Developing a Context Sensitive Tourist Guide. Technical Report Computing Department, Lancaster University. March 1998.

[7] T. Kindberg, J. Barton, J. Morgan, G. Becker, D. Caswell, P. Debaty, G. Gopal, M. Frid, V. Krishnan, H. Morris, J. Schettino, B. Serra, and M. Spasojevic. "People, Places, Things: Web Presence for the Real World". Proc. 3rd Annual Wireless and Mobile Computer Systems and Applications, Monterey CA, USA, Dec. 2000. p 19.

[8] Interagency GPS Executive board. http://www.igeb.gov

[9] Pradhan, S. Semantic Location. Personal Technologies, vol. 4(4), pp.213-216, Springer, 2000.

[10]Want, R., Fishkin, K. P., Gujar, A., and Harrison, B. L. Bridging physical and virtual worlds with electronic tags. In Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI'99) (May 1999), Addison-Wesley, pp. 370--377.

[11]Want, R., Hopper, A., Falcao, V., Gibbons J. The Active Badge Location System. ACM TOIS 10(1), Jan 1992 pp.91-102.

[12]Weiser, M. The Computer for the 21st Century. Scientific America, 265(3), 1991, pp. 94-104.