

# People, Places, Things: Web Presence for the Real World

Tim Kindberg, John Barton, Jeff Morgan, Gene Becker, Debbie Caswell, Philippe Debaty, Gita Gopal, Marcos Frid, Venky Krishnan, Howard Morris, John Schettino, Bill Serra, Mirjana Spasojevic  
*Hewlett-Packard Laboratories*  
1501 Page Mill Road, Palo Alto, CA 94304, USA.  
*timothy@hpl.hp.com*

## Abstract

The convergence of Web technology, wireless networks, and portable client devices provides new design opportunities for computer/communications systems. In the HP Labs' "Cooltown" project we have been exploring these opportunities through an infrastructure to support "web presence" for people, places and things. We put web servers into things like printers and put information into web servers about things like artwork; we group physically related things into places embodied in web servers. Using URLs for addressing, physical URL beaconing and sensing of URLs for discovery, and localized web servers for directories, we can create a location-aware but ubiquitous system to support nomadic users. On top of this infrastructure we can leverage Internet connectivity to support communications services. Web presence bridges the World Wide Web and the physical world we inhabit, providing a model for supporting nomadic users without a central control point.

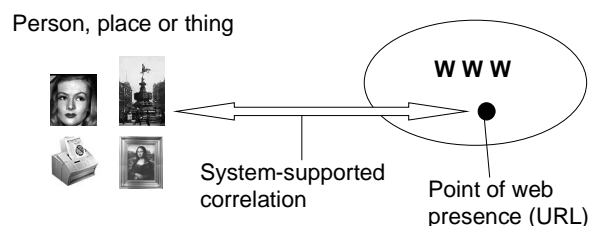


Figure 1. Points of web presence

## 1 Introduction

People are nomadic. They move around to work, to shop, or to play. Increasingly the places they enter are computerized. Their workplaces are filled with computers; the shops they browse are computerized; the toys they buy are computerized. And communications networks reach into every corner: web access continues to soar; cell phones abound, new wireless technologies are emerging. The convergence of these increasingly pervasive computers and communications networks offers new opportunities and challenges for systems designers, which are being addressed in the fields of "pervasive" [57], "ubiquitous", "nomadic" [27] and "context-aware" [50] computing.

At HP Labs we have been exploring these opportunities and taking on these challenges through adaptations of web infrastructure to support nomadic users in a project called "Cooltown" [10, 34]. We have been pushing web technology into digital "appliances" or "things" like printers, radios, and automobiles [14]; we have even been enabling users to automatically discover web resources associated with non-electronic things such as CDs, books and printed papers. We have been organizing physically related things into web "places" [8]. We have been exploring ways for people to use new digital communications devices to interact with these places and use the things they find there. Our early results, which we outline here, have been very promising. We believe that the web model of decentralized but standardized interaction will provide the basis for computing networks to support nomadic users.

Most of our work has focused on extending web technology, wireless networks, and portable devices to create a virtual bridge between physical entities (including users) and electronic services. We think the physical world and the virtual world would both be richer if they were more closely linked. Currently the Web is largely a *virtual* space: a space of web “sites”, online “malls”, and chat “rooms”. These virtual locations have little correspondence with physical spaces. While much of the information on the Web describes the world we physically inhabit, there are few systematic linkages to real world entities. This is unfortunate, because most of our activities concern physical objects other than computers.

In this paper we describe how we have systematically integrated web services to enhance communications with mobile people, to provide location-specific services in the places that they visit, and to provide interaction with the things that they encounter. To help users and developers formulate a common model of how these systems behave we describe them as supporting “web presence”. Web presence extends the “home page” concept to include all physical entities and to include automatic system supported correlation of the home page or *point of web presence* with the physical entity (see Figure 1).

We divide up physical entities into three categories: people, places, and things. This is the set of categories that the designers of Taligent followed, although to different ends[46]. As we go about our daily lives we move between *places* (the home, the office, shopping malls etc.), we meet with and talk to *people*, and we find and use *things*. People are, of course, the users of things and the occupants or visitors in places. Places have a special role as the venue or container for people and things. The simple catch-phrase “people, places, and things”, helps our system description focus on entities important to end users.

Our goal is to expand our access to people, places, and things by bridging them to the virtual world of web content. We want to make people, places, and things *web-present*. Things become web-present by embedding web-servers in them or by hosting their web-presence within a web server. Places become web present by organizing web things into collections under the management of a web service we call a “PlaceManager”. People become web present by offering global web home pages with “WebLink” services to facilitate communications between individuals and by offering information via location-specific PlaceManagers. Together these mechanisms and the overall model are important new steps towards bridging the physical and virtual worlds.

## 1.1 Building on the Web

Our nomadic computing infrastructure builds on web infrastructure. Therefore we need to establish why we think the Web is the most promising basis for nomadic systems and explain how we have extended it. We enumerate the key aspects of the Web’s design and practice that are important for us first and then outline our contributions:

*Low barrier to entry.* The diversity of the devices that mobile users carry and encounter argues against specific software based on device type or even application type. It also argues against assuming that all devices will run a form of middleware, such as Java (see Jini [30]) or CORBA [45], that is heavy on the commitments it imposes at the application, language, or resource level. The relatively simple web standards of HTTP and HTML with URLs [4] have proven themselves sufficient for transmitting information to users and allowing them to control systems and services. It is a simple matter to put HTTP *server* support in devices [14, 6] that nomadic users encounter, such as printers and projectors. Equally, web clients can be implemented in personal digital assistants (PDAs) and laptops as well as information “appliances” dedicated to a specific function, such as digital cameras.

*Device, language, and service independent protocol.* The form of interaction with particular devices and other entities should be encoded using HTML or XML [60] documents and MIME types, not application-specific binary formats and language-specific interface signatures. The communications protocol is text-based and described in human-readable documents, which simplifies such factors as standards evolution, implementation and debugging. Web standards are language-independent, and have been implemented on many OS and hardware platforms.

*Content evolution, not interface heterogeneity.* The basic web interface, consisting of GET and POST operations, allows interactions between client and server components that have no a priori knowledge of one another’s particular functionality. Components are programmed to process standard-format content that is moved between them, always using the same operations. That is to say, the Web’s architecture is extensible and robust through being “content-oriented”, not object-oriented. The Web’s API is simple and static; evolution occurs through changes in content. Clients and servers interpret content permissively, rendering what they can and ignoring the rest.

*Depth.* Web solutions exist for higher-level content-provision services, including security and payment.

*Ubiquity.* The rising quality and reach of the Web increases the probability of web access in mobile venues. People use the Web at work and at home today; increasingly they will find it available when they shop or travel; eventually they will find adequate access “everywhere” they want electronic connections. The Web supports mobile users by allowing them transparent access to resources inside or outside their current environment. That is, the Web runs over the Internet, the

Internet is widely accessible, and the Web runs the same way at all points of the Internet. We shall discuss ways of bridging devices that are *not* IP-capable to the Web.

By using web technology we increase our chances of having a ubiquitous, scalable system and web systems have proven to be simple enough for many users to configure. The central task of a web presence infrastructure then is to add new layers to support presence while preserving the desirable properties of the Web.

## 1.2 Contribution: Extending the Web for Nomadicity.

While the Web has properties necessary for a successful nomadic computing it is not sufficient. Even the much-discussed “wireless web”, supporting mobile users of web browsers, is not enough. Web presence extends the wireless web with new technologies specific for nomadic users.

If we examine the architectural requirements for a nomadicity enumerated by Kleinrock [39], we see that the Web provides several of them. The Web provides “integrated access to services” through the model of web pages, hypertext links and form-filling. The wireless web can support “bandwidth/processor/resolution adaptively” through adaptation services and proxies [18]. The Web demonstrates sufficient “independence between the network and applications” and it certainly scales along all the design parameters Kleinrock lists [39].

However, the Web—as it comes to us—fails to meet other requirements Kleinrock identified. The Web provides no *ad hoc* access to services for mobile users and no location awareness. We will describe new and more effective means for *ad hoc* access to services based on techniques of embedding and recovering URLs on objects and in places. We provide location-awareness for nomadic users—the requirement that we believe underlies Kleinrock’s requirement for “automatic sensing, searching, and/or tracking of users”. We provide location-awareness by providing location-dependent information resources to users, rather than providing user locations to some tracking system. We shall show how users can be in charge of where, when and to whom they make their presence known. We expect this to be a critical factor in users’ acceptance of systems that provide location-aware services.

In addition to these extensions, we also demonstrate how the Web’s built-in independence from network implementation helps in designing applications for nomadic users. We will demonstrate how to deliver web services to mobile users without requiring a global wireless connection like a cell-phone or mobile IP [43]. This has the advantage of minimizing how much of the infrastructure needs to be up and running in order for users to interact with local services. For example, a mobile user should be able to print a document at a nearby printer without necessarily having to contact any global services. A local web server should be enough, and sometimes even a simple peer-to-peer interaction will suffice. Thus we support a familiar service across a broad range of clients and connectivity options.

Our contribution comes in efforts to make the appropriate extensions to web technology to support nomadicity, through a unifying “people, places, and things” model. We provide an overview of our results, reviewing some previously published accounts of it and describing some work not described elsewhere.

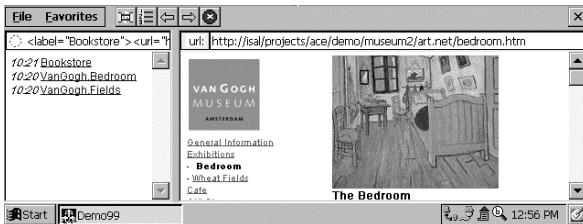


Figure 2. Museum visitor’s PDA screen.

## 2 Web Presence by Example

Since our approach to services for mobile users differs from others we begin with some examples. These examples are based on the demonstrations running in our “Cooltown” demonstration center at HP Labs [10]. After describing the examples we will explain the infrastructure that supports them.

### 2.1 Visiting the Cooltown Museum

The Cooltown Museum and Bookstore offers visitors a web-enhanced experience. As visitors tour the museum, their portable digital assistant (PDA) can receive URLs from wireless “beacons”. These beacons are small infrared transceivers located close to pictures or sculptures; the URLs link into a web of information about the items. Using the PDA’s web

browser, visitors can read or hear about the artist or the work and about related art works in the museum. The URLs can also be stored as bookmarks for further study or they can be used to select reproductions of the artwork from the museum's online store.

As visitors move into the Cooltown Museum bookstore they can use their PDAs to sense the URLs associate with books, calendars, and posters for sale (Figure 2). These URLs will give them book reviews, available inventory of calendars, and other colors of posters for examples. The museum bookstore's web portal provides services to assist in buying books, including a service to order books that are not available. In addition, the bookstore offers a printing service, for visitors to print out web pages that they collected during their visit or even order reproductions of paintings to be printed just in time for purchase at the exit.

These scenarios combine local wireless communications for location-specific addressing with wider range wireless Internet access to obtain the web pages addressed. Thus these PDAs have wireless connectivity (e.g. IEEE 802.11 [41]), but no location-sensing or tracking technologies are needed to create a location-aware application.

## 2.2 Working in a Cooltown Conference Room

Every Cooltown conference room supports mobile users. As you enter a Cooltown conference room you can collect the URL for the room from a beacon into your PDA or cell phone. The URL will lead to a web page for the room giving links to, for example, the room's projector, printer, and electronic whiteboard, as well links to web-based maps. The web page is served from a "PlaceManager", which implements a web portal service dedicated to the conference room. This portal is a physically-based local equivalent to web portals like *yahoo.com* and *excite.com*. This meta-service acts as a container for the services of the room and as a portal or gateway to web access for visitors. Its contents can be tailored to the conference room's attendees or the current activities.

Workers in the room can also "eSquirt" URLs at the room's "web appliances" such as web-enabled projectors or printers. An eSquirt is a wireless transfer of a URL over a short-range wireless link, like infrared [28]. Squirting a URL at the projector will project the corresponding web page, creating a shared web browser. Squirting a URL at the printer will fetch the web pages and print them. A worker armed with only a cell phone or even a wristwatch can use the conference room facilities by squirting URLs referencing content available on the Web.

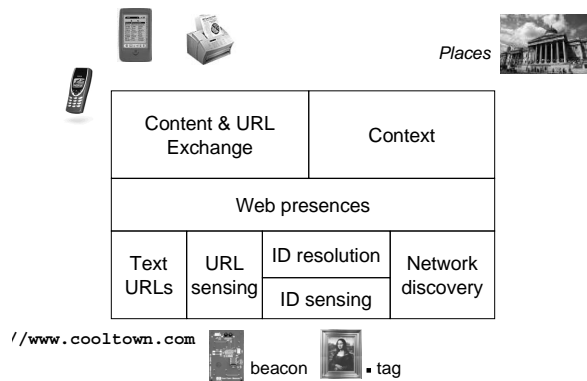
To support visiting guests of an organization the host organization provides a "reverse" proxy accessible only from the visitors' wireless network (outside the intranet). This allows visitors inside the conference room access to certain room facilities via the proxy without compromising the rest of the resources in the surrounding intranet (which is protected by a firewall). Visitors can also access web resources that are within their company's intranet from their host's wireless network. For this, they employ a secure "web tunnel" that encrypts traffic and authenticates users all over HTTP links.

Finally a visiting worker may need to pull in colleagues from remote sites to join the discussions in Cooltown. "WebLink" supports multi-modal web-facilitated communications in a way that is completely controlled by its users. The visiting workers will discover services for communicating with them in the meeting room, and their selection from among those services will automatically be made available through their WebLink homepages to those with sufficient privileges. For example, the local speakerphone number may be made available, or a Microsoft NetMeeting session.

This conference room scenario extends the URL addressing and discovery illustrated by the museum scenario to include directories (PlaceManager) and controllable services (projector). Note that the assumptions made by the infrastructure are quite low: these mechanisms are all straightforward extensions of web technologies and they are all based on resident services with standard clients. No centralized global service needs to be deployed for these nomadic users.

## 2.3 The common user experience in Cooltown

The three examples above explore various aspects of web presence but they also show a common theme. The typical experience with web presence we seek for users is that of collecting links to points of web presence as they encounter them in the physical world. One could think of the physical world as having web hyperlinks at certain physical points. The links are captured and presented on the user's client device, and the result of clicking on such a link will be a web page, delivered to the user's screen. The user can then travel through the virtual space of the Web with normal web techniques or the user can travel through the physical space to find a different point of web presence. The user understands URLs as pointers to information and other services associated with physical objects at the same level that users of web browsers now understand URLs as links to web resources.



**Figure 3. Infrastructure layers**

### 3 A Web Presence Infrastructure

With the advantages of a web basis in mind and some examples to see where we are headed, we next describe the additional infrastructure we have developed to support web presence. Figure 3 shows the layers of a web-presence infrastructure discussed in the following subsections. At the bottom layer are mechanisms for obtaining the points of web presence of people, places and things, through discovery systems and by sensing URLs and identifiers. In the middle layer are web presences. Some of those are legacy web resources. Others are resources specifically designed for capturing the functions of and relationships between people, places and things. In the top layer are services for exchanging content over HTTP with web-present entities and coordinating the processing of that content to provide application-level functionality. Also at the top layer is infrastructure to provide services to nomadic users according to their context; that is, parameters such as the user's location.

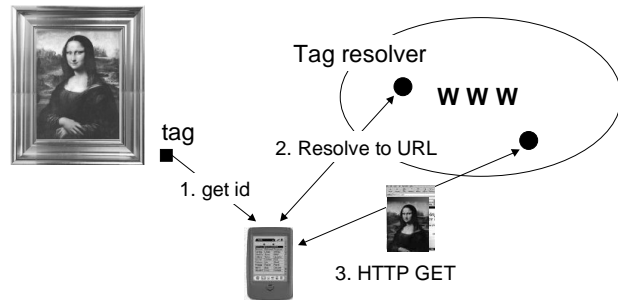
In the remaining discussion we will personalize the descriptions by replacing "a visitor" with "Veronica" and we'll call Veronica's host "Harry". Veronica travels with a portable web browser, visits places that may or may not know her, and she is able to use things she finds there when appropriate without prior arrangements.

#### 3.1 Finding Points of Web Presence

We start with Veronica entering an art museum carrying her PDA. Figure 2 shows a screenshot, from our demonstrator implementation of the Cooltown Museum described in Section 2.1, of the PDA Veronica carries with her as she tours the museum. The PDA is equipped with a sensor for beacons (see below), wireless LAN connectivity, and a slightly modified web browser. With the wireless web link, Veronica will be able to traverse hyperlinks to explore the Web. Our aim is to start her exploration on a point of the Web appropriate for the museum or even a particular work of art she finds interesting. Using the sensor on the PDA, she can obtain URLs for points of web presence associated with objects in the museum. On the left of the screenshot are links to three web-present entities that Veronica has recently sensed. The first is a place, the bookstore in the museum. The next two (less recently visited) are paintings. The device has remembered the links to these web-present entities (and will do so as long as Veronica desires). As she visits places and senses things, the PDA records all such links by default, creating a series of bookmarks corresponding to Veronica's interests. Using the web browser, Veronica can follow hyperlinks from the web pages corresponding to these points of web presence to web pages provided by the museum and out to the Web in general. The PDA connected to the Web over a wireless network becomes connected to the physical world through its sensor. When the sensor leads to a web presence we call the process "physical discovery" of the corresponding online resource.

There are two main approaches to discovering the URLs of entities via a sensor: either to sense that URL directly (as Veronica does in sensing URLs directly from beacons) or to sense an identifier from the entity, which is then looked up to obtain a URL. To consider those possibilities more fully, we first outline identification technologies.

*Beacons, tags and other identification technologies.* Web presence identifiers, whether they are full URLs or identifiers for looking up URLs, can be correlated with physical objects using a variety of technologies. The *beacons* we mentioned earlier are active emitters of identifiers over, for example, infrared links. Alternatively, *tags* are passive devices or labels that provide identifiers to corresponding active sensors. For example, a barcode scanner actively scans a passive barcode



**Figure 4. Indirect sensing: the web presence for a painting.**

tag. Some small electronic devices, like passive RFID tags [47] and iButtons [27] are also tags that supply an identifier to a sensor placed, respectively, near them or in contact with them. Sensors, whether they are beacon receivers or tag readers, are increasingly being integrated with PDAs and other portable devices.

Other suitable types of sensor include a GPS sensor for obtaining the entity's latitude and longitude coordinates, or a camera for performing image recognition on the entity. The choice of identifying technology depends on many factors, including cost, the size of the sensor and the identifying device, the need for batteries and their lifetime, and whether it is preferable for the user to be active or passive in obtaining the identifier.

*Direct sensing.* In our "art museum" implementation, we laid out a room with pictures on the walls and situated the "bookstore" nearby. We implemented web presence for these entities using active, URL-emitting devices (beacons). We call this form of discovery, where the sensor obtains the URL directly, *direct sensing*. Next to each painting and in the bookstore we placed infra-red beacons that supply PDAs with the URL of the corresponding point of web presence. The paintings' URLs are those of pages supplied by the Van Gogh museum. The URL of the bookstore resolves to a "PlaceManager" server (discussed in Section 3.3) operating as the bookstore's web portal.

*Indirect sensing.* *Indirect sensing* is where the user's client device obtains an identifier that it looks up to obtain a URL. This level of indirection allows more convenient identifiers and more flexible system design as we discuss below. Figure 4 shows a different art museum demonstrator in which a PDA client senses a painting's tag. The sensing device knows the URL of a *resolver*—a service that maintains a collection of bindings from identifiers to URLs and which returns the URLs bound to a given identifier. When the client senses a tag, it sends the identifier to the currently selected resolver. If the resolver has an entry for the identifier, it sends back the corresponding URL(s).

Our implementation of resolution follows the web service architecture so as to minimize the mechanisms that need to be added to browsing as it stands. The client software is a browser augmented by a simple plug-in. Resolvers provide web forms that contain a new type of field that is filled in as a side-effect of sensing, rather than by the user typing or using a pointing device. When the user, say, scans a barcode, the resultant identifier is automatically filled into the form and the form is posted to the resolver. The resolver replies with a page containing the corresponding URL.

Furthermore, we have devised a new type of URI so that everyone with a domain name or email address may easily mint unique identifiers for the physical entities that they manage [51]. Service providers and users may independently bind those and other identifiers to web resources that suit the purposes of their clients and communities.

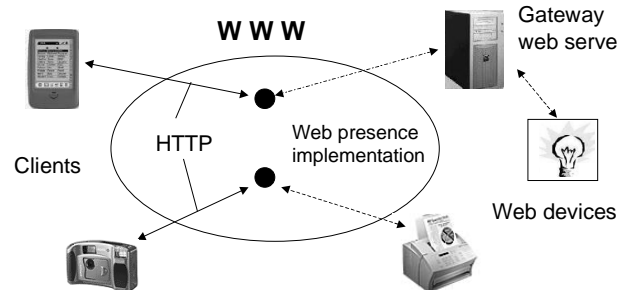
Resolvers are themselves web resources, and the user navigates to a resolver as they would any other web page, or they may pick up the URL of a local resolver using direct sensing. The user may choose a resolver that reflects his or her personal requirements, or those of a group of users with interests or tasks in common. For example, visitors to the art museum could utilize local resolvers in the museum that provide the URLs of pages in their native language, pages maintained in a global catalogue of paintings around the world, or pages provided by the visitor's own art school. Details of the resolution design are available in [33].

In comparing direct and indirect sensing, the obvious advantage of direct sensing is its independence of additional network services. One reason for using indirect sensing is to work within the constraints of some identification technologies. For example, latitude and longitude coordinates could be translated into a zip code and hence into the URL of a place's web portal provided by Yahoo; the result of image recognition is some type of signature that must be looked up to obtain a URL, e.g. against a database of tourist sites to provide the URL of a corresponding web page. But there is a positive reason to utilize the level of indirection implied by a mapping from an identifier to a URL. That is to be able to associate physical entities with points of web presence in a context-dependent or application-dependent way. Since any number of mappings may exist, a given physical entity can be associated with multiple points of web presence. Services and applications can maintain and rely on mappings from identifiers to resources that are appropriate for a given domain,

user group or other contextual factor. These technologies can also differ in their human factors, a subject of an ongoing collaboration between the Cooltown group and San Francisco’s Exploratorium [15].

Beacons and especially tags have been used to establish a correlation between physical objects and electronic resources. Mostly these correlations have been in closed systems like supermarket checkouts. Our focus has been on open systems where the user of the beacon or tag may not be organizationally related to the service provider. The work at Xerox PARC [55] was similar, using electronic tags and beacons to enhance physical things by associating their presence with virtual actions, such as the invocation of a program. We leverage web systems and wireless local area networking to improve the practicality of these correlations. Yanin and Ishii show how an action, such as removing or placing a container on a shelf, can be used to automatically update a web page about that container [59]. This correlation is more automatic than we have attempted, but also more narrow in scope.

*Comparison to Network discovery.* Physical discover can be compared to network discovery. When Veronica enters an area of the Cooltown Museum with a wireless network compatible with her PDA or when she attaches her computer to a public access point on a wired network, her device may send network packets to a “service discovery” multicast address [2,11,20,21,49]. This well-known, preconfigured address routes the packets to all network services that have joined the group. If Veronica requests printing services, printers in the multicast group can respond. Special servers that have grouped printer services can also respond. In both cases, the response is a URL pointing to the printer’s web presence. In our example, the museum bookstore’s publicly available printer will respond.



**Figure 5. Devices (printer, light), with and without embedded web servers.**

This multicast network-based discovery works well in some scenarios but it suffers from several problems based on its reliance on network topology. First, the range of the discovery may include too few resources, preventing users from locating electronic services they need and that are near by, but that don’t happen to be within range of the multicast packets. While newer discovery schemes seek to avoid this problem (see [11]), they can’t address the second problem: the range of discovery may include too many resources. The underlying issue is that real places don’t always match the network’s topology. They can physically overlap, they may have fuzzy boundaries, and they may contain devices that are inappropriate for discovery on human-centered grounds such as administrative or territorial control. For example, we want to be sure that we access the printer in our hotel room, and not the one in the next room even if both are accessible on the same 802.11 network. Moreover, places may contain non-electronic entities such as paintings, whose web presences are eligible for discovery but which may be hosted anywhere on the Internet.

For the above reasons, we consider that, while network discovery has its uses, the “physical discovery” mechanisms of direct and indirect URL sensing are set to prevail in many practical circumstances. We shall return to this issue in Section 3.3.

### 3.2 Infrastructure for Things: eSquirt and Form-exchange

Web devices provide services to users based on their electronically accessible functionality. For example, Veronica sends a document to a web-present printer; Veronica switches on the lights in her home when it gets dark, by accessing the web-present lighting controller. These operations take the form of the exchange of content with points of web presence. Sometimes content transfer is the primary aim, as when we print. Other times the content transfer is secondary, as when we control things by using content to change their state. We describe the eSquirt protocol, an approach to content-transfer that sends the URL of the content to the recipient. Then we outline a forms-based protocol that provides more control over the

transfer of content from a source to a sink device. Finally, we outline some of those protocols' implications for client and server software.

*eSquirt.* The eSquirt technique described in Section 2.2 provides a simple approach to printing. Suppose Veronica is in the art museum bookstore and decides to print a reminder of the Van Gogh whose web page she collected from a beacon. With an eSquirt-capable PDA, Veronica can walk up to the printer and “squirt” the URL for the page. The PDA sends the URL of the page to the printer—not the page content that it has stored. On receipt of the URL, the printing service obtains the web page, renders it and sends the result to the printer.

Earlier Veronica used the eSquirt protocol to receive the URL from the beacon by the painting. What is transmitted, in general, is an XML string that specifies the URL of the resource, the title of the resource and, optionally, a digital signature that recipients may use to verify the provenance of the URL. The string is transmitted over infrared. The Ultra protocol [29] is well-suited to such short, one-shot transmissions. Ultra is unreliable but, as with a TV remote control, if transmission fails then the user presses the “squirt” button again until feedback at the recipient confirms success.

One advantage for Veronica of the eSquirt protocol over printing directly via a driver in her PDA is that the PDA, with its limited resources, has only a reduced fidelity version of the page, including, for example, images in relatively low resolution. In eSquirt, by contrast, the printer obtains a full-fidelity version of the page and prints that.

But the most important advantage of the eSquirt paradigm is its device-independence. In her office, Veronica could beacon a URL from her workstation into her PDA. The URL could refer to a presentation she has prepared. Veronica may then squirt that URL into a printer or a projector or a picture frame or any other device that is capable of rendering the page. The protocol always works the same way. What differs is the processing of the URL by the recipient.

This design implies that Veronica needs to have a reasonable idea of what combinations of squirted URLs and receiving devices make sense. The receiving device designer must expect some mistakes: Veronica may mistakenly squirt a URL of an audio file into a printer. However, we specifically do not want to prevent such mistakes in advance. The prevention measures, such as type-checking, create costly and confusing dependences in the interoperation of devices and stifle innovations.

The device-independence of eSquirt and the portability of Veronica's PDA allows it to act as a physical clipboard for web resources. In the same sense that a desktop user-interface supports user-controlled *ad hoc* inter-application communication by cutting and pasting to a mouse-controlled global temporary storage point called a clipboard, we use the PDA as a temporary storage point. Veronica copies a URL from one physical device and pastes it to another device. The result is a human-controlled integration of the devices.

*Form-exchange.* In addition to moving URLs around in the physical world, we also need to move content and to control devices. For these cases we have been exploring the Web's model of fill-in forms that are submitted via HTTP POST. That is, we first GET an HTML or XML form, fill in values at the client, and submit it back to the server [34]. At the architecture level the only differences from conventional web practice are the transport—we might be using OBEX over IRDA for example—and the server—it will may run locally in a device we are using rather than out on the Internet somewhere. Using web forms provides greater facilities for interaction between devices while retaining eSquirt's crucial property of device-independence through the content-oriented nature of HTTP. Our client acts as a remote control with a user interface and the command set determined by the form obtained from the server device.

Figure 5 shows interactions between two clients and their respective web devices. It shows a digital camera that interacts with a printer via HTTP operations on the printer's point of web presence. The URL for the printer resolves to its network address. The printer responds to requests from clients with a form containing fields for specifying the content to be printed and its settings, for example, paper size. The printer may supply the form either as an HTML page, to clients such as PDAs, or as an XML (Xform) document, to clients such as cameras that present the form according to their own special interface constraints. In our example, the camera sends the image as, for example, JPEG-encoded data in a form field. While human users typically interpret conventional web forms, a simple form might be interpreted directly by some devices. For example, a camera might interpret a form requiring an image as a request for an image upload.

Figure 5 also shows the case of controlling devices that are not themselves capable of HTTP: lights connected to a home PC by a controller unit. The home PC acts as a gateway to the lighting unit, which controls the lights using a non-HTTP protocol such as X10 [38]. A remote user fills in the controller's form and issues an HTTP POST operation via a web browser on a PDA; the programs on the PC that handle HTTP operations directed to the light's URL manipulate the lighting unit according to the values in the filled-in form.

*Implications for client-server software.* Our implementations of Esquirt work over IRDA. Beacons emit URLs every 3 seconds. PDAs have a small application for accepting, storing, and resquirting URLs. If the PDA has a network connection, then we interface the eSquirt application with the PDA's web browser so that the user can load the browser with URLs by clicking on them. Form-based exchange is more demanding but its complexity must be minimized if it is to be implemented widely on devices. We have prototype implementations of HTTP client software for devices that have

specialized user interfaces and devices equipped with sensors to supply values for forms. These include cameras to upload images to the Web and picture frames to download images from the Web. The latter device may not even have a physical user interface, and so requires a server on board to supply an interface for presentation on other devices.

Small, embedded web servers [6] are needed to run on devices that have sufficient capabilities to support them, such as a printer; associated HTTP operation handlers are required to read and manipulate the devices' physical state. Gateway web servers running on conventional server machines are needed for devices that cannot run a web server, such as a simple domestic lighting unit. These utilize HTTP operation handlers that read and manipulate the physical state of remote things. To explore the potential for web-present devices, we developed the ChaiServer web server [9]. ChaiServer was engineered to have a small footprint (200 kbyte). A ChaiServer executes objects called *chaillets*, which are similar to servlets (see for example [26]). More recently, we have re-implemented ChaiServer in C, achieving an even smaller footprint around 40 kbyte [40]. In addition to the web server, an eSquirt receiver such as a printer or projector must fetch and render content. That requires much of the software that makes up a browser but without a user interface.

The Satchel project at Xerox's European Research Centre [17] has produced tools to support the mobile worker with services to fetch remote documents, and local document services such as printing, faxing and scanning. Their infrastructure is also web-oriented, and they incorporate a "beaming" service for transferring URL-based document "tokens" between devices. Their beaming protocol uses a two-way form exchange; by contrast, eSquirt's one-shot simplicity allows URL transfer between simpler devices—beacons, in particular. Earlier work at HP produces a device-independent, peer-to-peer, content-transfer protocol called "Jetsend" [58]; it has appeared in several HP products. Our work differs by extending open web standards, transferring URLs as well as images and other content, and by working on the control of devices through web forms as well as transfer.

### 3.3 Infrastructure for Places: PlaceManager

A place is a context for service provision, based on an underlying physical domain permeated by one or more networks [8,22]. Examples are a railway station covered by WaveLAN connectivity [56], a café covered by Bluetooth [5], and a home covered by HomeRF [24]. In this subsection, we outline the functionality of a PlaceManager [8], our implementation of web portals for places. We describe our approach to registering the services that the PlaceManager makes available through that web portal. Then we outline "WebBus", an exercise in context-aware computing in which the "place"—a bus—is mobile.

Just as physical places contain people and things (and sometimes other places), web-present places are hyperlinked collections of the web presences of those people, places and things. that creates a physical and contextual organization of place information. For example, if a printer is in a certain meeting room then the printer's web presence can show its location (to people who might view it remotely) and the meeting room can present a link to that printer. We store these links and present web pages based on them from a service we call PlaceManager. It is responsible for providing secure views of the set of resources present in the place and the services based around them. In general, the content that the PlaceManager provides to a particular user depends on the client's security principal, the user; on that principal's preferences; and on the client device's functional capabilities. For example, an employee might have access to more services in a place than a visitor; a user with a PDA might be presented with different content than a user at a PC.

By placing beacons or tags in a physical place (sec. 3.2) that lead, through physical discovery, to a PlaceManager with links to nearby web-enabled appliances and place-relevant services, we create a location-aware system that can be administered locally or globally. Users need only a slightly augmented wireless web browser on anyone of a wide variety of mobile devices to discover and use location-specific services. They need not be tracked or registered and the system can be administered as a self-contained local operation or as part of an arbitrarily large physical complex.

Early on in our work, we realized that the PlaceManager is a special case of a "web presence manager" [13]: a database of web-present entities and their relationships—relationships that do not have to be physical proximity but might, for example, be "Joe is the system administrator of printer 6297". Our web presence manager stores descriptions of entities—their URLs and attributes, including the services they provide—in an XML database. To represent relationships such as co-location or ownership, it aggregates the entities into directories. For example, Harry's personal devices could be aggregated in the web presence for Harry; Harry's web presence and those of other users could, in turn, be aggregated in the web presence of her current location. A PlaceManager just becomes a Web Presence Manager configured to represent a place.

*Registration.* In Section 3.1 we discussed ways clients can discover local services. The reciprocal problem is the registration of services to create a "place" or any other directory of related web presences. Several automatic discovery and registration services have been designed [2,11,20,21,49]. One of the issues with those protocols is their plurality—we cannot expect all devices to run all or one of them. Our PlaceManager, acting as a higher-level discovery and registration

agent, can combine the results of lower level protocols. For example, it can run both UpnP [53] and SLP and make the results available as web pages to visiting users whose devices run none or only one of the protocols.

As we pointed out in Section 3.1, the other problem with automatic service discovery protocols is their inaccuracy. The same is true of them as registration protocols. Those services work well only when the area of service provisioning matches the area covered by a sub-network and where there are no issues of administrative control or semantic appropriateness. Just as we gave alternative, “physical” models of discovery, we provide an alternative, physical model of registration [3]. In that model, the place’s administrator walks up to and senses the tags or beacons on just those entities that are to be included as providing services in the place. Similarly, if Harry wants to be contacted via one of his devices while in the place, he physically adds that device to his web presence by sensing its identifier. Registration thus becomes volitional (Harry might not want his device to be discovered automatically) and it becomes highly accurate. The downside is that it becomes effortful but we believe that human control of registration and discovery, supported by hints from automatic discovery systems, is the best balance.

*WebBus.* WebBus is an example of dynamic location-aware computing. We constructed the WebBus by embedding PlaceManager with a GPS transponder and a wireless Internet link in a bus. Clients contacting the web server get different experiences depending upon their location. For Veronica in the bus, the WebBus provides a web portal view that depends upon the bus’ current location. As the bus approaches Coit Tower, the portal begins to offer information about that attraction. For Harry at the bus stop waiting for Veronica, the WebBus server gives the bus’s current location and expected arrival time. Note that the location dependence works twice here. First the bus stop’s PlaceManager gives Harry’s PDA the link to the appropriate WebBus. Then the WebBus provides up to date information based upon its current GPS reading. In this way Harry can easily decide if waiting for Veronica is prudent or whether he should browse the bus stop’s web portal for a local bookstore.

In one way our PlaceManager is a directory and hence should be compared to service discovery directories as we did above. In another way our directory, being focused on supporting physical areas, can be compared to work on “smart environments”. A smart environment is based on a physical environment equipped with sensors and actuators. It may span a room (e.g. the iRoom [19]), or a set of rooms or house (e.g. the EasyLiving project [7], the Aware Home [32]). What makes those environments smart is that services and applications in the infrastructure process the sensor readings, in order to trigger events, to adapt their behavior or to control the physical state of the environment. At present, our infrastructure is general-purpose and it is strong on the users’ co-navigation of the virtual and physical domains. We do not have a well-integrated event model and we avoid automatic tracking of users—characteristics of smart environments. Our work is closer to projects that have investigated the bridging of services to points in geographic areas at the instigation of the user or as automatically instigated from the user’s device. For example, the “SpaceTags” system [52] maps the user’s GPS coordinates to virtual services from their handheld device; and the GUIDE project at Lancaster University [12] has developed a system for presenting guide information to tourists as they visit sites.

### **3.4 Infrastructure for People: WebLink**

WebLink uses a person’s point of web presence as a level of indirection for electronic communications. A user such as Harry, with whom Veronica needs to communicate, places a link to a WebLink redirector service in his conventional personal web page. The WebLink redirector service runs on a globally accessible web server. When Veronica clicks on the link in Harry’s page, an invocation is made to the WebLink redirector service to request a resource for communicating with Harry. If the web redirection service possesses a URL for communicating with Harry in the place where he currently resides, it returns that URL to Veronica’s PDA. However, if Harry is “offline” then the redirector returns a web page that includes a service to leave a message for Harry. This information will be delivered to Harry when he comes on-line.

The web redirection service is updated with the URLs of suitable communication services as Harry moves around, as long as he wishes to be reachable. There are various ways of implementing the update mechanism, reflecting different choices of responsibility between the user’s PDA and web servers in the places that host the user. We have implemented an update service that runs in each web-present place. When Harry enters a place, he may identify himself to the PlaceManager through his PDA; if he wishes to be contacted there, his PDA also supplies the URL of his redirection service. The place-level service identifies the URL of a suitable communication service for Harry in the place—unless, as in our example in the previous subsection, Harry nominates his own personal device. It registers this choice of communication channel with Harry’s redirection service. The service will then offer that channel via the link in Harry’s web page.

With WebLink, the communications channel that Harry and Veronica use can depend upon their locations and preferences without the usual trial-and-error used in electronic communications currently. There are other efforts similar to WebLink, including the mobile people architecture [42] and various commercial “universal inbox” systems. Weblink’s

distinguishing features include its integration with other aspects of web presence. Since the location of the user can be sensed electronically the appropriate communication channel can automatically be offered. Standard web security techniques can be used to control access to Harry's entry in the WebLink redirection service. Harry can password-protect his entry, so that only those he trusts can communicate with him through it.

### 3.5 Security infrastructure for nomadicity

The typical nomadic scenario raises security issues for visitors, such as Veronica, and for hosts, such as Harry. Harry wants to offer certain of his place's resources to Veronica over a wireless network while protecting the rest of his resources. He may want to offer some of his resources only while Veronica is on his premises. Veronica wants to be sure that she is picking up URLs that originate with Harry, and not URLs spoofed by another visitor with an infrared-capable device or a set of barcode stickers. Veronica also may wish to access documents that she has not brought with her and which are behind the firewall of her own intranet.

We now outline some of the security mechanisms that we have constructed so far to address those issues. This is ongoing work and there are several other security issues raised by nomadicity, such as user authentication and privacy that we shall not discuss here.

*Restricted access points and reverse proxies.* We integrated and adapted the existing notions of wireless access points and reverse proxies so that host organizations can provide protected services to visitors. The visitors-only wireless local area network that Veronica uses is connected to other networks only by a restricted access point that Harry provides. The access point allows traffic only to two machines: an external proxy, which enables traffic to the Internet, outside Harry's intranet; and a "reverse proxy" machine [44], outside Harry's intranet, which allows traffic to a restricted set of Harry's services. The reverse proxy handles URLs containing capabilities—opaque strings that the reverse proxy looks up to obtain a record about the client's ability to access one of Harry's services. That access record may be set up to restrict use of the URL according to local policies. For example, it may restrict the number of uses or the time span of use. Harry may allow Veronica to access certain resources even after she has gone home but he may wish to prevent access after a specific time, such as the expiration of a work-for-hire contract.

*Location authentication.* If Harry wishes to allow only visitors who are currently on his premises to access one of his resources, then the visitors' capabilities may be qualified to be location-specific. We have devised protocols for authenticating the location of a client [36]. The basic idea is to use a network whose range is practically constrained so that it covers only the physical domain of interest. A service issues a challenge that requires the client to use the physically constrained network to give a satisfactory response. A reverse proxy can issue the challenge on the behalf of a number of services it protects. Suitable network technologies include Bluetooth, infrared and even, in some cases, IEEE 802.11.

*Beacon and tag authentication.* When Veronica picks up a URL via infrared reception or, via a resolver, from a barcode or another type of tag, she wants to be sure that the result is really a URL that Harry has set up—rather than a mischievous URL giving her misleading information. For that reason, beacons may emit digitally signed XML strings [54] and the results of resolvers may be digitally signed XML strings. Harry gives Veronica the public key she needs to verify those strings. This key is placed into her device, along with other security-relevant parameters such as a WEP key [41], when she enters Harry's organization. The key entry takes place over short-range infrared at a point that is physically monitored against tampering, such as the reception desk of his organization.

*SecureWebTunnel.* Harry exported certain relatively low-security resources such as printers to visitors outside his firewall, using a reverse proxy. The URLs-as-capabilities handed out to users are convenient in situations where they may remain anonymous. HP's SecureWebTunnel work addresses the following example or resource export, where the security requirements are more demanding. Suppose Veronica is working in Harry's conference room and she needs to fetch a business plan from her home office. Unfortunately, she is of course outside the firewall of her company's intranet. SecureWebTunnel enables Veronica to fetch documents via the Web from her company intranet securely, even though she is outside the firewall. Moreover, apart from having to prove her identity, the fact that she is outside the firewall is transparent. She can continue to use the standard browser on her PDA without reconfiguring it when she travels outside the organization. And she continues to use the same URLs outside the organization, as she would use inside it.

SecureWebTunnel provides a secure communication channel through a pair of web proxies, one in Veronica's PDA, and one in the firewall of Veronica's company intranet. The proxy in her PDA is location-aware. When the PDA's proxy is given a URL for a location that is inside Veronica's home intranet, it behaves differently according to whether it is currently inside or outside that intranet's firewall. In the latter case, it contacts the SecureWebTunnel proxy in the firewall rather than the proxy normally used inside the intranet. The PDA proxy makes a secure SSL connection [16] to the firewall proxy. Veronica is challenged, and has to type in a password (which is transmitted securely over the SSL connection). The SecureWebTunnel proxy in the firewall verifies the password. For a limited time, Veronica can now issue requests to any

web resource that has been cleared for secure export in this way. The secrecy and integrity of the data is guaranteed by the SSL connection.

The SecureWebTunnel work is similar to that of Abadi et al. [1]. SecureWebTunnel has a significant advantage: the use of a proxy on the client device means that the firewall proxy does not have to re-write web pages on the fly in order to provide URL transparency outside the firewall. The Satchel project also addresses document access from outside firewalls [17], but access is not transparent between standard web browsers and Satchel browsers. An outstanding research problem is how to implement policies and manage more selective access control for this type of export of services from firewall-protected places. Aspects of this problem have been tackled for virtual workspaces for collaborative activities [35]. The ability to export web resources selectively is a potential advantage of SecureWebTunnel over conventional “virtual private network” (VPN) solutions, which use secure IP [31] or a proprietary protocol. VPNs typically provide all-or-nothing access to the intranet.

## 4 Conclusion

We have described web presence as a basis for bridging the physical world with the World Wide Web. Through our examples, many of them based on demonstrators that we have implemented, we have illustrated how web presence for people, places, and things can support users as they go about their everyday tasks.

The notion of web presence and our demonstrations are built upon earlier work in our laboratory. This includes work on web-enabled devices [25], work on embedded web server technology [14], and Java implementations of web servers augmented with object models, event models and service discovery [9]. This exploration of the interaction between the Web and the physical world continues with the projects outlined here.

In presenting our work we have made two central arguments: that users can benefit from greater connection between the physical and the virtual worlds, and that the Web is the best technology for building that connection.

The first argument is based upon the observation that users’ everyday activities mostly concern physical objects, but most of today’s computers are not linked to physical objects. Relatively few physical objects have computers inside them or information systems correlated with them. Portable computers with wireless communications and sensors allow us to systematically increase the connection.

Our second argument is that the Web (rather than, say, JINI or CORBA) is the best middleware for building the connection. The Web is widely deployed because of the robustness of its standards. It supports evolving standards to embrace new content; it can be implemented on devices with relatively little memory and other resources; and its user interface, the web page, enables users to perform a wide variety of tasks. It supports both human-to-computer and computer-to-computer interactions.

Many similar projects exist, whose aim is to address some aspect of building the connection between the virtual and the physical. We compared our work to these projects. Overall we differ in the comprehensiveness of our approach and our commitment to the Web as the key component of the connection. However, we made it clear that the Web provides only part of the infrastructure for web presence. We need sensing and service discovery technologies to feed the Web with URLs. The Web provides HTTP for content exchange with points of web presence, but the Web as it stands does not support nomadic working: that is where PlaceManager and the other technologies we have described make their contribution.

More information about our demonstration applications is available at our “CoolTown” web site, <http://www.cooltown.com/>.

## References

- [1] Abadi, M., Birrell, A., Stata, R., and Wobber, E. *Secure Web tunnelling*. [http://www.research.digital.com/SRC/personal/Martin\\_Abadi/Papers/tunnel/206.html](http://www.research.digital.com/SRC/personal/Martin_Abadi/Papers/tunnel/206.html).
- [2] Arnold, K., Wollrath, A., O’Sullivan, B., Scheifler, R., and Waldo, J. *The Jini Specification (The Jini Technology Series)*. Addison-Wesley Pub Co; ISBN: 0201616343.
- [3] Barton, J., Kindberg, T., Sadalgi, S. *Physical Registration: Configuring Electronic Directories using Handheld Devices* Hewlett Packard Labs Tech Report HPL-2001-119 and submitted to IEEE Personal Communications.
- [4] Berners-Lee, T., Masinter, L., and McCahill, M. *Uniform Resource Locators (URL)*. December 1994. Internet RFC 1738. Available at URL <ftp://ftp.nordu.net/rfc/rfc1738.txt>.
- [5] *Bluetooth home page*. <http://www.bluetooth.com/>.
- [6] Borriello, G., Want, R.: *Embedded Computation Meets the World Wide Web*. CACM 43(5) pp. 59–66.
- [7] Brumitt, B., Meyers B., Krumm, J., Kern, A., and Shafer, S.. *EasyLiving: Technologies for Intelligent Environments*. *Proc. of the 2nd Int’l Symp. on Handheld and Ubiquitous Computing*. Bristol, UK 25-27 September 2000, pp. 12-29.

- [8] Caswell, D. And Debaty, P. *Creating a Web Representation for Places*. In the proceedings of the 2<sup>nd</sup> International Symposium on Handheld and Ubiquitous Computing 2000 (HUC 2000) Lecture Notes in Computer Science v 1927, pg 114-126.
- [9] *ChaiServer home page*. [http://www.chai.hp.com/chai\\_server.html](http://www.chai.hp.com/chai_server.html).
- [10] *CoolTown home page*. <http://www.cooltown.hp.com/>.
- [11] Czerwinski, S.E., Zhao, B.Y., Hodes, T.D., Joseph, A.D., and Katz, R.H. *An Architecture for a Secure Service Discovery Service*. Fifth Annual International Conference on Mobile Computing and Networks (MobiCom '99), Seattle, WA, August 1999, pp. 24-35.
- [12] Davies, N., Cheverst, K., Mitchell, K., and Friday, A. *Caches in the Air: Disseminating Information in the Guide System*. Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications (WMCSA '99), New Orleans, Louisiana, U.S., 25-26 February 1999.
- [13] Debaty, P. and Caswell D. *Uniform web presence architecture for people, places and things*. IEEE Personal Communications, vol. 8, no. 4, August 2001, pp. 6-11.
- [14] *Embedding Web Access Mechanism in an Appliance for User Interface Functions Including a Web Server and Web Browser*. United States Patent no. 5956487, Sep 21, 1999. Inventors: Venkatraman, C., and Morgan, J. Assignee: Hewlett-Packard Company, Palo Alto, Calif
- [15] *Electronic Guidebook Research Project* <http://www.exploratorium.edu/guidebook/>
- [16] Freier, A.O., Karlton, P., and Kocher, C. *The SSL Protocol Version 3.0*. <http://search.ietf.org/internet-drafts/draft-ietf-tls-ssl-version3-00.txt>.
- [17] Flynn, M., Pendlebury, D., Jones, C., Eldridge, M., and Lamming, M. *The Satchel System Architecture: Mobile Access to Documents and Services*. In Mobile Networks and Applications, vol. 5, no. 4, ACM, pp. 243-258.
- [18] Fox, A., Gribble, S., Brewer, E., and Amir, E. *Adapting to Network and Client Variability via On-Demand Dynamic Distillation*. ASPLOS 1996: 160-170
- [19] Fox, A., Johanson, B., Hanrahan, P., Winograd, T. "Integrating Information Appliances into an Interactive Workspace," IEEE Computer Graphics and Applications, May/June 2000, pp. 54-65.
- [20] Goland, Y.Y., Cai, T., Gu, Y., and Albright, S. *Simple Service Discovery Protocol/1.0 Operating without an Arbiter*. <http://search.ietf.org/internet-drafts/draft-cai-ssdp-v1-03.txt>.
- [21] Guttman, E. *Service Location Protocol: Automatic Discovery of IP Network Services*. IEEE Internet Computing, Jul.-Aug. 1999. pp. 71-80.
- [22] Harrison, S., and Dourish, P. (1996). *Re-placing space: The roles of place and space in collaborative systems*. In Proceedings Computer Supported Cooperative Work '96, Cambridge, MA. New York: ACM.
- [23] *HTTP - Hypertext Transfer Protocol*. <http://www.w3.org/Protocols/>.
- [24] *HomeRF home page*. <http://www.homerf.org/>.
- [25] *HP device could make Web a device controller*. In Netscape World, Dec. 1996. <http://www.net-dev.com/netscapeworld/nw-12-1996/nw-12-newsbriefs2.html#HP>.
- [26] Hunter, J. and Crawford, W. *Java Servlet Programming* O'Reilly & Associates, Inc. 2001
- [27] *iButton home page*. <http://www.ibutton.com/>.
- [28] *IrDA home page*. <http://www.irda.org/>.
- [29] *IrDA Infrared Mobile Communications* <http://www.irda.org/standards/specifications.asp>.
- [30] *Jini home page*. <http://www.jini.org/>.
- [31] Kent, S. and Atkinson, R. *Security architecture for the Internet Protocol*. Internet Request for Comment RFC 2401, Internet Engineering Task Force, November.
- [32] Kidd, C., Abowd, G., Atkeson, C., Essa, I., MacIntyre, B., Mynatt, E., Starner, T. *The Aware Home: A Living Laboratory for Ubiquitous Computing Research*. In the Proceedings of the Second International Workshop on Cooperative Buildings - CoBuild'99.
- [33] Kindberg, T. *Ubiquitous and contextual identifier resolution for the real-world wide web* HP Labs Tech. report HPL-2001-95.
- [34] Kindberg, T and Barton, J. *A Web-based Nomadic Computing System*, Computer Networks, v35 (2001) 443-456
- [35] Kindberg, T. *Security for Network Places*. In proceedings Distributed Systems Security Workshop, ECOOP '98, Belgium.
- [36] Kindberg, T. & Zhang, K. *Context authentication using constrained channels*. HP Labs Tech. report HPL-2001-84.
- [37] Kindberg, T. & Barton, J. *The challenges and opportunities of integrating the physical world and networked systems*. HP Labs TR HPL-2001-18.
- [38] P. Kingery. "Digital X10". <http://www.act-solutions.com/kingery13.htm>

- [39] Kleinrock, L. *Nomadcity: anytime, anywhere in a disconnected world*. Mobile networks and applications vol 1 no 4, (1997) pp. 351-357.
- [40] Krishnan, Venky. Hewlett Packard Labs, Private communication
- [41] LAN MAN Standards Committee of the IEEE Computer Society. *Wireless LAN medium access control (MAC) and physical layer (PHY) specifications*. IEEE Standard 802.11, 1999 Edition, 1999.
- [42] Maniatis, P., Roussopoulos, M., Swierk, E., Lai, K., Appenzeller, G., Zhao, X., and Baker, M. *The Mobile People Architecture*. In the ACM Mobile Computing and Communications Review, July 1999.
- [43] *Mobile IP Web Resources*, <http://computer.org/internet/v2n1/mobile.htm>
- [44] Netscape Reverse Proxy <http://developer.netscape.com/docs/manuals/proxy/adminux/revpxy.htm>
- [45] *The Object Management Group home page*. <http://www.omg.com/>.
- [46] Potel, M., and Cotter, S. *Inside Taligent Technology* (Addison-Wesley, 1995).
- [47] *RFID home page*. <http://www.aimglobal.org/technologies/rfid/>.
- [48] Salber, D., Dey, A.K., and Abowd, G.D. *The Context Toolkit: Aiding the Development of Context-Enabled Applications*. In Proceedings of the 1999 Conference on Human Factors in Computing Systems (CHI '99), Pittsburgh, PA, May 15-20, 1999. pp. 434-441.
- [49] *Salutation Consortium home page*. <http://www.salutation.org/>.
- [50] Schilit, B., Adams, N., and Want, R. *Context-Aware Computing Applications*. IEEE Workshop on Mobile Computing Systems and Applications (1994)
- [51] *The Tag URI Homepage*, <http://www.taguri.org>.
- [52] Tarumi, H., Morishita, K., Ito, Y., and Kambayashi, Y.: *Communication through Virtual Active Objects Overlaid onto the Real World*, Proc. of The Third International Conference on Collaborative Virtual Environments (CVE 2000), ACM, pp.155-164, 2000 (Sep. 2000).
- [53] *Universal Plug and Play Forum*. <http://www.upnp.org/>.
- [54] *W3C XML Digital Signature*. <http://www.w3.org/Signature/Drafts/WD-xmldsig-core-991008.html>
- [55] Want, R., Fishkin, K.P., Gujar, A., and Harrison, B.L. *Bridging Physical and Virtual Worlds with Electronic Tags*. In Proceedings of the 1999 Conference on Human Factors in Computing Systems (CHI '99), Pittsburgh, PA, May 15-20, 1999.
- [56] *WaveLAN home page*. <http://www.wavelan.com/>.
- [57] M. Weiser, *Some Computer Science Issues in Ubiquitous Computing*. Communications of the ACM, 36(7), 1993, pp. 74-84.
- [58] *JetSend Appliance Architecture*, <http://www.cswl.com/hpjetsend/white.html>.
- [59] Yarin, P., and Ishii, H. *TouchCounters: Designing Interactive Electronic Labels for Physical Containers*. In Proceedings of the 1999 Conference on Human Factors in Computing Systems (CHI '99), Pittsburgh, PA, May 15-20, 1999. pp. 362-369.
- [60] Extensible Markup Language (XML) 1.0, <http://www.w3.org/TR/REC-xml.html>.