Next Century Challenges: Data-Centric Networking for Invisible Computing

*The Portolano*Project at the University of Washington*

Mike Esler, Jeffrey Hightower, Tom Anderson, and Gaetano Borriello

Department of Computer Science and Engineering
University of Washington, Seattle

{esler, jeffro, tom, gaetano}@cs.washington.edu

Abstract

Computing and telecommunications are maturing, and the next century promises a shift away from technology-driven general-purpose devices. Instead, we will focus on the needs of consumers: easy-to-use, low-maintenance, portable, ubiquitous, and ultra-reliable task-specific devices. Such devices, although not as limited by computational speed or communication bandwidth, will instead be constrained by new limits on size, form-factor, and power consumption. Data that they generate will need to be injected into the Internet and find its way to the services to which the user has subscribed. This is not simply a problem of ad-hoc networking, but one that requires re-thinking our basic assumptions regarding network transactions and challenges us to develop entirely new models for distributed services. Network topologies will be intermittent and services will have to be discovered independently of user guidance. In fact, data transfers from user interfaces to services and back, will need to become invisible to the user and guided by the task rather than explicit commands. This paper outlines a vision of this future and identifies research problems that will require our attention in the areas of user interfaces, distributed services, and networking infrastructure.

1 Introduction

As computing and communication technology advances, we are challenged to build applications where the user interface is not *on* a computer, but *is* the computer; where users do not *connect* to a network, but have their data *travel* on the network; where users are not *commanding* operations to be performed, but agents are operating autonomously on their *intentions*. This will be a major aspect of the coming transition of computing into a consumer mass-market for a wide range of interconnected task-specific devices motivated by function and convenience rather than by technological prowess.

A simple analogy inspired by Donald Norman illustrates our vision [22]. Today's desktops and palmtops are multi-purpose tools – electronic Swiss Army knives. But how many of us would use a Swiss Army knife for preparing a dinner at home? It may be fine on a camping trip, but impractical for more routine activities where efficiency and quality are more highly valued. Instead, we use specialized tools such as paring knives, tongs, stirring implements, and a variety of food processing appliances. Moreover, we have a very easy time borrowing a carving knife from a neighbor or using one in a kitchen other than our own.

This is not the case for computers today. They are too complex because they try to be all things to all people. Although we may encounter them everywhere, we cannot easily use them for our purposes; they often operate in an unfamiliar way, attempt to do too much, or use services we may not (or should not) be able to access. Consumer computing devices should be just as easy and flexible to use as kitchen utensils. The focus needs to shift from the innards of devices to the uses and services that make sense in our daily lives.

Such a conceptual shift becomes possible as technology ceases to be a limiting factor. We can expect in the very near future to pay $10 for a GB of RAM, a GFLOP/sec of computation, a megapixel of display area, 1Mbs of wireless communication, or 100GB of disk space. This will enable us to take computing capability for granted in everyday devices. New computing capabilities will also be common-place: speech and handwriting recognition are rapidly improving and disseminating, vision-based gesture recognition is coming out of the laboratory, and wireless technologies are proliferating at ever increasing bandwidth.

But before these changes radically alter the consumer landscape, crucial research must be carried out in at least three areas:

1. **User Interfaces**: New modes of interaction such as user movement, proximity of devices, and embodied information presentation will augment the keyboard, pen, audio and video interfaces we see today. The challenge is maintaining task-oriented consistency across physical devices while managing the multiple interfaces in a coherent manner. Also, the focus must shift to user intent and expectation and away from the execution of explicit commands. Data gathered from a variety of location and motion sensors, identification tags, and on-line services, will augment or replace many user directives common today.

2. **Distributed Services**: Rather than abstract capabilities and specific infrastructure items, emphasis must

*Portolano charts are the seacoast charts created by Portuguese sailors of the 14th and 15th centuries. They were fundamental to the Age of Discovery initiated by Prince Henry the Navigator that led to the European discovery of the African coast and the New World.

To appear in the Proceedings of MOBICOM '99, Seattle, Washington, August 1999.

be placed on applications to which users can easily relate. Different user interfaces, appropriate to their contexts, should be able to interact with the same services. A scheduling service, for example, should be uniformly available via home display, PDA, auto PC, and a phone with voice recognition and synthesis. These services must be more openly organized - into extensible horizontal layers rather than the vertically integrated monolithic services of today - to facilitate service deployment and consumer choice.

3. **Infrastructure**: The networking fabric must provide robust data transfer with replication and discovery as well as the ability to marshal computing resources at internal network nodes. Users must be able to count on their data arriving where it needs to go without their direct intervention. Thus, the network must be data-centric; transmission, routing, authentication, and resource reservation should be handled independently of the location of the user who injected the data. Intermittent wireless connectivity and transmission cost optimizations are examples of decisions for which the user should simply provide guidance once rather than for each use.

The remainder of the paper revolves around these three areas. In Section 2, we elaborate on our vision of mobile computing in the next century with an example that illustrates our ideas. Section 3 describes in detail the specific obstacles to realizing our vision as well as potential partial solutions derived from ongoing research. Finally, we conclude in Section 4 with recommendations for the mobile computing community and outline our own research efforts.

2 The Vision

For too long our computing tasks have been driven by infrastructure and technology. Instead, tasks should be accomplished easily, ubiquitously, and worry-free. The landscape of the 21st century we envision is one where computing devices will be highly specialized to particular tasks, will be ubiquitous consumer items, and their user interfaces will be invisible to all but the most sophisticated users. Users will not be concerned with such esoteric issues as file formats, configurations, and connectivity. They will expect tools to match tasks and not be dependent on ownership or access rights. They will exploit a vibrant, robust, and highly differentiated market of information services to get their objectives accomplished. Moreover, they will trust the communication infrastructure to safeguard their data and deliver it to the services to which they have subscribed or requested.

This terrain is illustrated by the following scenario:

Alice begins the day with a cup of coffee and her personalized newspaper. When her carpool arrives, she switches to reading the news on her handheld display, where she notices an advertisement for a new 3-D digital camera. It looks like something that would interest her shutterbug-friend Bob, so Alice asks her address book to place the call.

Bob's home entertainment system softens the volume of his custom music file as his phone rings. Alice begins telling Bob about the camera, and forwards him a copy of the advertisement which pops up on his home display. Bob is sold on the product, and after hanging up with her, he asks his electronic shopping agent to check his favorite photography stores for the lowest price and make the purchase.

When the camera arrives, Bob snaps some photos of his neighbor's collection of antique Portuguese navigation instruments. After reviewing the photo album generated automatically by a web-based service, Bob directs a copy of his favorite image to the art display in his foyer. He also sends a pointer to the photo album to Alice and instructs his scheduling agent to set up a lunch date so that he can thank her for the suggestion.

In section three, we will break apart this scenario into its constituent parts and highlight the major issues that it raises, but first, let us briefly speculate about what sort of environment would support this sequence of events.

Alice loses her news connection as she exits her apartment, but it is automatically reestablished via a wide area wireless connection when she turns the next page in her (now PDA-based) personal newspaper. When Bob decides to purchase his new camera, his voice-print and his biometric identification earring authenticate the transaction. His digital camera comes equipped with a short-range radio transceiver as well as a removable cartridge. The datawatch he wears can communicate with the camera as well. When he snaps a picture, the photo data is communicated to the watch where it is replicated and held until a connection can be made to the rest of the world.

While the camera may remain isolated from the network (in the trunk of his car, for example) Bob will eventually enter his home or bring his watch close to his mobile network/telephone terminal. At this point, the phone or, alternatively, the home network portal, can upload the data from the wrist-watch and send it along on the next step of its journey. The pictures find their way through various proxies to the photo album service (As a shutterbug, Bob decided to purchase the album service instead of paying on a per use basis). The costs for all the communications listed above are borne by Bob. When the camera takes the snapshot it included his personal ID (from the earring with biometric safeguards) in the data packets. Lower-cost options are always considered. For example, if Bob is near his home (and likely headed there), the phone will not place a call to transfer the data, but rather defer the transfer to happen at home where a cheaper Internet connection is available.

3 Realizing The Vision

Why is such a storyline not possible now? Each of the areas identified in the introduction (user interfaces, distributed services, and infrastructure) has numerous obstacles which currently prevent the vision from being realized. We will now consider each of these areas in some detail.

3.1 User Interfaces

User interface (UI) issues are of major importance in our vision. Alice and Bob are surrounded by computing appliances, but the interactions occur smoothly and easily. While there are certainly many issues in UI evolution raised by our vision, in this paper we consider two that we feel are especially important.

3.1.1 Multiple Interfaces

The first challenge is handling multiple access points to distributed services. For example, how do we present Alice with a usable newspaper as she leaves her home display and switches to her PDA to ride with her carpool? How do we provide the scheduling service to a nomadic user like Bob as he potentially wanders from home to office to car? He should be able to complete a task such as scheduling the lunch date with Alice as naturally and easily from his car phone or as he can using his home display terminal. His interface should be intuitive on each physical device and yet still clearly derived from the same scheduler service.

Let us first step back for a moment. In the WIMP (Window, Icon, Menu, Pointing) arena, the notion of separating interface issues from semantics is well studied. Window managers, in all their flavors, are quite good at abstracting and managing screen real estate. Also HTML, at least in its intended form, is excellent for specifying content and leaving presentation to the discretion of the clients. Similar techniques might therefore be employed as clients "go mobile" and use new and novel interactions methods such as vocal, handwritten, video, gestural, or olfactory interfaces. One possible quick solution is to force the WIMP approach on mobile devices with the advantage that familiar desktop applications can migrate directly to the mobile devices. While this migration seems appealing, it is flawed since the new constraints and benefits levied by mobile devices are ignored. A better solution is a method to allow mobile clients to discover the semantics of any service's UI and present an interface suited to the client's size, shape, abilities, or resource limitations.

Research toward this goal is already underway. A first attempt was Interface Description Languages (IDLs) as described by Hodes et al [16]. IDLs describe abstract UI semantics via a hierarchical set of types. More recently, IDLs have been superseded by a scheme built on top of the extensible markup language (XML) [15]. XML is a format for using SGML, the Standard Generalized Markup Language (of which HTML is a subset), for document interchange. XML is not a language as much as it is a standard that permits the creation of custom markup syntax to suit the needs of a particular document format [4]. Another interesting research effort is the VoxML markup language from Motorola. VoxML allows the integration of speech interfaces for interaction with web content through simulated dialogues [21].

Markup languages are undoubtedly an enabling technology – the web's rapid growth can largely be attributed to the ease and simplicity they afford to publishing. But as good as XML is at describing the semantics and content of a document, it will never be enough. The abundance of HTML extensions and plugins illustrate that application designers want more control over the precise "look and feel" layout of the interfaces to their systems. There may be an initial embrace of XML-based markup as in the heady days of HTML, but there is no reason to believe that the desire for more precise layout control will not resurface in a mobile multi-interface environment once the novelty of newly enabled technologies such as speech browsing has passed. The challenge is to provide a robust UI architecture to balance the users and developers' needs with the continual growth of the network fabric.

This line of argument suggests a need for advanced development environments that allow developers to create new user interfaces (on specific devices) while re-using much of the code of the back-end of the application (the network communication code and the ties to network services).

3.1.2 Invisible Interfaces

Another interface issue is the movement away from explicit commands and toward interfaces that implicitly take their direction from people's behavior. The challenge is to make the UI fit so well into the environment that it becomes invisible insomuch as the user is aware she is interacting with a computing device. Such design is enabled by involving users in the design process. This idea is the thrust of much recent work in the HCI community including that of Fishkin et.al.[10] and Weiser [39].

To achieve this invisible design, we must first conduct user studies, create prototypes of the new mobile world, and deploy applications and services that a wide range of users will find useful. This process is apt to be a multidisciplinary effort involving talent from new fields traditionally not directly related to computing such as anthropology, human factors, and ethnography. Along the way we will develop the networking protocols, distributed services, location sensing infrastructure, and a collection of devices that demonstrate the effectiveness and utility of our approach.

One potential dividend of invisible computing research is context aware computing (CAC). CAC attempts to coalesce knowledge of the user's task, emotions, location, and attention with other available data such as the time and knowledge about other users. The CAC field is in its infancy, but already groups are exploring this design space with projects such as Georgia Tech 's CyberDesk [1] and the spatial location work at AT&T Laboratories Cambridge [37] and Xerox PARC [36] [35]. The likely result is that the fusion of data from a variety of sensors and databases will be crucial to inferring intention. A major issue will be in how to get the required information in a timely an efficient manner.

3.2 Distributed Services

The core of our research should be in distributed services. Instead of computing infrastructure with abstract capabilities, we must instead provide the user with services to which they can easily relate. For example, when Bob creates his new photo album, he may call on several different services to operate on his photos. One of these stores the photos on a reliable server in his home or a rental site. Web pages are then prepared by the service and stored as part of the subscription arrangement. The request to forward his favorite photo to the foyer display is handled by the photo album service. Because satisfying the request may involve the purchase of rights to images, the photo album service checks with Bob's financial management service to see if he is on budget and the purchase meets his expected usage profile. Once the purchase is made, the agent may then negotiate with a home decorator service to determine in which situations the image should be displayed in the foyer (for example, especially when Alice pays him a visit). Bob does not concern himself with the technical specifications of the photo server nor did he muddle through file format conversions. Such issues are the concern of the service provider Bob employs. To Bob, the infrastructure is simply a means to an end.

3.2.1 Horizontal Integration

The current infrastructure-centric focus has led to system architectures that are vertically integrated, not horizontally layered. By vertical we mean systems that they attempt to provide entire solutions to a problem. Traditionally, these suffer from high-cost and inflexibility (e.g. the inability to get information to and from users who do not subscribe to the same service). Although administration and regulation can be centralized, vertical systems often make it difficult or even impossible for a user to obtain exactly the subset of services he requires (the "take it or leave it" dilemma). Furthermore, vertical systems tend to stifle competition and make it difficult to quickly deploy new or alternative services.

We argue that horizontal layering is much more appropriate for the mobile networks of the future. If Bob desires to migrate to a new photo album service or even use the same service with different mass storage arrangements or more exciting graphic design it should be possible for him to easily do so.

3.2.2 Agent Technology

Many of the tasks that Alice and Bob performed in the scenario require the unintentional use of a variety of distributed services. When Bob scheduled lunch with Alice, he did not explicitly (intentionally) access Alice's calendar and find an available date. A scheduling *agent* performed this task on his behalf - taking into account both Alice and Bob's schedules, eating habits and preferences, and convenient travel and meeting times. Yet another agent assisted Bob in publishing a photo album by making layout and presentation decisions based on his preferences.

The term *agent* is difficult to define. Nwana describes an agent as a "component of software and/or hardware which is capable of acting exactingly in order to accomplish tasks on behalf of its user." [23] The technology and protocols used to implement agents are becoming better understood [9] [8], but how they can be applied to mobile applications in environments with widely distributed data sources and intermittent connectivity is research that must be further explored. Of particular importance will be an active networking structure allowing the flexible integration of applets and servlets.

3.2.3 Service Deployment

Smooth integration of new services is essential. If Bob were to purchase a new photo service such as face recognition for automatic portrait labeling, its installation should occur seamlessly. The service should be able to discover Bob's resources (rented or owned) such as storage, processing cycles, etc. necessary for its operation. Similarly, a new hardware component must be able to setup itself and its connection without the explicit involvement of a network provider or other such middlemen. In the current model, significant configuration is required for new devices such as wireless phones.

Discovering available resources, both in general and at any specific point in time, is crucial to effective service deployment and should be an automatic process from the user's perspective. Resource discovery is an infrastructure issue and thus will be addressed in some detail in section 3.3.1

Deploying services effectively also necessitates new distribution and maintenance models. In current systems, users often feel overwhelmed from perpetually having to upgrade their systems with new enhancements, bug fixes, and security patches. Although the user must certainly be kept in the loop about major issues, the day to day maintenance chores should be the responsibility of the service or subscription provider. This model implies the creation of an architecture supporting dynamic upgrading and hot-swapping of system components. Indeed, many vendors are already addressing this issue. For example, web browsers and multimedia playback tools often come equipped with these autoupdate abilities in order to simplify supporting new codecs and data formats. The challenge will be to implement similar techniques in the mobile domain while spanning heterogeneous hardware and connectivity situations.

3.3 Infrastructure

While task-oriented applications will be the motivation of our research, we expect to encounter many interesting infrastructure challenges in their implementation. The nature of these challenges will be applying existing technologies to the mobile and invisible domains.

3.3.1 Resource Discovery

In our scenario, Alice and Bob's devices were able to discover the network services available and communicate with each other without the end-user's assistance. When Alice asked to call Bob, her address book discovered the phone service available in the car, downloaded an interface to that phone, and invoked a remote method using the interface. Similarly, Bob's phone was able to discover the stereo system, retrieve its interface, and pause the music.

Resource discovery is the subject of numerous research efforts including the RDP [27] and SLP [25] protocols, Berkeley's Service Discovery Service [5], Sun Microsystems' JavaSpaces and Jini [34], T-Spaces from IBM [29], Universal Plug and Play from Microsoft [28], and the HAVi consumer electronics consortium [14]. Each takes a slightly different approach based on the application domain they were intended for. None were designed specifically with mobile networks in mind, so a host of questions should be re-considered in this new context: Should resource discovery depend on a local service database, or are ARP-style requests more appropriate? Should a lookup service provide clients with a resource name, or an object written in Java or object-TCL? How powerful should the lookup query model be, and how should it be implemented: as Java objects, XML, or something else? In addition to answering these questions, it is important that any discovery systems we implement be self-managing since both clients and resources (including the lookup registry) are likely to change as devices are disconnected and reconnected.

Due to intermittent connections and ad hoc networks, data may have to find services on its own without the assistance of the application that injected them it the network. This necessitates the ability of the networks to execute code in the data packet. This code can call on discovery functions provided on major nodes or, if it finds itself on minor nodes that only route, select the best route to follow to get to those services. To guarantee data safety, this will also require controlled replication of data packets and finite lifetimes. Acknowledgments from the services that receive the data packets back to the original generators of the data are also problematic as the source may be disconnected or may have moved to a new location. Thus, replies also need to be able to call upon distributed location services (that may in turn call upon schedule and profile services) to help them find their routes or cache data in anticipation of a future connection.

While it is important for devices to be able to discover services, they should only be able to use those services for which they have permission. Bob's camera, for example, did not publish the photos in his neighbor's album. Likewise, Alice's address book can only place calls from phones that she is allowed to use. Kerberos [30] is currently a popular system for authenticating network requests, but its statically configured centralized servers make it a poor choice for mobile networks. The IPsec protocol [17] is a newer scheme for private data transfer built on public-key infrastructure, which may be better suited to the needs of mobile computing. But IPsec does not solve the problems of privacy and authentication central to our vision. The SDS protocol also provides security, in the form of DSA signatures, and RSA encrypted broadcasts. Significant challenges remain, however: What does a device do if the network authentication server is not available? How do you revoke certificates of devices that are only intermittently connected? For example, can Alice's daughter have a private video diary that is kept secret from others in her family as well as protected from outside access?

3.3.2 Data-Centric Networking

Another key issue is the need for data-centric networks. Active data bundles should be able to marshal (and pay for) the resources they need to make progress in the network. Bob's camera, for example, is capable of transferring photos to several locations without the aid of cables or a base station. Its RF transmitter can communicate with nearby devices including Bob's wristwatch, cellular phone, and appliances. Data moves from device to device until it reaches the service it is intended for. Though the ideas of ad-hoc networking are valuable, we need to re-think our basic assumptions about network operations and construct a data-centric network architecture as a means for distilling, naming, and locating the data objects that travel within the network.

When Alice leaves home, her personalized newspaper is still available to her. It is certain that the network characteristics of her home and car are very different, so the quality of the newspaper adapts. A high-bandwidth service may be essentially free at home, but costly to use elsewhere. Network infrastructure must be able to inform devices about the network they are using, as well as be able to provide admission control. Unlike the Internet, Bluetooth [32], HomeRF [11], USB [33], and IEEE 1394 [18] networks all reserve bandwidth for synchronous data channels. In these cases, the network infrastructure should be able to offer guarantees about the quality of service (QoS) to those users willing to pay a premium, provided the network has sufficient free capacity. Protocols such as RSVP [41] and QEX [6] have only laid the initial groundwork for effective QoS management in mobile applications. Those services which do not have specific QoS requirements may still benefit from knowledge about the current network state.

A data-centric network must also be able to manage ubiquitous persistent storage. Bob's photo album may reside on a specialized storage device in his home that his camera, art display, and friends can all access simultaneously or, alternatively, Bob's photos may reside in several different locations – redundantly or in different forms. Sun Microsystem's NFS [20] offers transparent and authenticated access to a global set of files residing on a central server. Unfortunately, this system requires static configuration and has a single point of failure, making it undesirable for mobile applications. Reliability and availability can be increased by using a distributed file system such as the Open Software Foundation's DFS [24]. At the same time, DFS provides users with access security via Kerberos, and a *uniform name space* for accessing files. The Coda filesystem [31] is a descendant of AFS that is designed specifically for mobile clients. The Bayou project [7] provides mobile clients access to data by using a distributed database approach. Using these projects as starting points, a combination of storage services will need to be designed and integrated into mobile environments. What is really needed to make our vision a reality is ubiquitous storage made available to distributed applets running across an ad hoc network. We still have much to study about the consistency semantics in this type of environment in the presence of failures and intermittent connections. None of the existing systems are adequate for these purposes.

3.3.3 Distributed Computing

The infrastructure and technology of distributed computing also plays an important role in building the services illustrated in the scenario. When Bob sent his photos to the photo album service, they needed to be in a specific compression format. Rather than waste bandwidth and process the photos when they arrive at the service, the camera downloads software to run locally. How did the camera know what methods this object exports? Bob's phone cannot store interfaces for every device in his home, so it must request the correct interface whenever it needs to execute a remote method. How is this accomplished?

Several emerging solutions to this problem, including Jini, Liquid Software [13], and the Active Networks Toolkit [40] are based on the Java language. In these models, bytecode is downloaded and executed on the client. Using the Java RMI, clients can then use the services of other devices. Other solutions are based on CORBA [12] and Microsoft's COM [19]. Both allow clients to execute code located elsewhere, and provide mechanisms that allow clients to discover an object's interface at runtime. Unlike the Java-based solutions, CORBA and COM place restrictions on the interface rather than the implementation language. The challenge will be in coming to a consensus and developing an open standard that incorporates the positive aspects of each.

In section 3.1.1 we suggested that application development environments should automate multiple user interface creation, but ideally development environments could also be built to synthesize all the detailed communication and coordination code and optimize it for the needs of the application. These needs could include restoration of hard and soft state after network partitions and transfer of data in appropriate bundles for service discovery, authentication, and data migration. As it is likely that the code will need to targeted to very different architectures and operating systems, mitigating the development burden - synthesizing communication and coordination code and deploying it automatically onto a distributed fabric - will be crucial to ensuring that applications are written by as large a segment of the population as possible. This automatic code synthesis is similar in spirit to CAD tools for distributed embedded systems such as Chinook [2].

3.3.4 Intermittent Connectivity

In order to achieve invisible, trouble-free connections and disconnections from networks, mobility must be built into our protocols. Indeed, we find it likely that intermittent connectivity will be the norm for the foreseeable future due to power, cost, bandwidth, latency, and congestion limitations. By disconnecting during idle periods, devices will consume less power, and lengthen the time between recharging batteries. Not surprisingly, new mobile protocols are already appearing for the intermittent connection environment. Bluetooth [32] and HomeRF [11] are standards for small area RF networks in which devices can join and leave ad-hoc networks of devices as necessary. While this technology will allow mobile devices to join new local networks to take advantage of local resources, it does not address more complex issues such as hand-off and signal strength analysis found in cellular protocols [38]. Mobile devices also need to be able to ascertain information about the networks that they join. Some solutions leverage off of the DHCP and DNS protocols [3], but their use is limited to IP networks. Other important issues, including IP tunneling and mobile host registration, are contained in the mobile IP specification [26].

Each technology is important to the future of mobile computing, but no single protocol is completely satisfactory. The cellular model works well, but is it appropriate for the desired services and applications? More likely we will need a range of wireless technologies with different transmission ranges and power requirements to support devices like key-chains and earrings that function indefinitely without recharging but have very limited range to more traditional PDAs that need to connect to the Internet and can be more easily recharged. Can the existing infrastructure support a large increase in the number of devices connected? RF is promising, but it suffers from a limited bandwidth per volume defined by its range - another reason for using a wide range of wireless technologies and overlaying their ranges. Many devices will want to operate in multiple overlays and act as routers for others. Irrespective of the medium, security must be integrated into the protocols to satisfy application requirements.

4 Recommendations

We have presented a vision that we believe will require fundamental new work in several disciplines if it is to become a reality. Key features of our vision include:

- multiple user interfaces that rely upon user intent,
- horizontally-layered network-based services, and
- novel routing and active networks infrastructure.

We strongly believe that any exploration should begin with user studies, analysis of current practices, and the building of complete usable systems. It is imperative that we shift from a technology to an application focus. We must determine how our prototypes will meet the challenges of user-centric tasks, and how these motivate the new devices and services.

Simple applications motivated by needs of the office and home environments can serve as a starting point. An example that demonstrates the extreme interwoven nature of our vision is a calendar application that ties together a variety of input devices (including PDAs, desktops, telephones, location sensors, and wall-mounted displays) with a variety of actuators (including alarms, e-mail notification, and automobile navigation systems) with a variety of services (including individual and group diaries, negotiation agents for scheduling meetings, and photo album services). For example, data fusion should make it possible for the calendar service to adjust the morning wake-up alarm based on traffic conditions or the knowledge that the car is almost out of gas. Building these devices will expose interesting issues in all three of the thrust areas.

We are beginning our research work by looking at both applications and infrastructure issues in parallel. In the applications area, we are experimenting with the practical uses of in-building location sensing based on RF tags. Our hope is not to track people but objects of a wide range of sizes - from folders to books to appliances - via triangulation over time with connection to physical layout and the expected uses of objects and locations. Another application area involves the dissemination of small embedded web servers (2 sq.in. boards with 10-baseT Ethernet and serial connections) into the instructional spaces in and around our building and campus acting as portals to our department's network. On this infrastructure, we initially plan to deploy a simple messaging service and architecture robust enough to support future applications such as distributed calendars and scheduling.

In the infrastructure area, we are evaluating different approaches to service discovery and more specifically the architectures for proxies to handle computationally-limited mobile devices. In networking, we are focusing on creating a general-purpose API for collecting data from varied sensors, operating on the composite data, and controlling actuators with the results of those computations. Our hope is to develop an open-source library to enable others to work in this space.

We expect the constraints that arise due to:

- intermittent connectivity,
- power consumption,
- application development and deployment,
- service architectures and discovery, and
- active networking

to dominate our research agenda in making invisible computing a reality.

5 Acknowledgments

We acknowledge stimulating collaborations, discussions, and comments from Roy Want of Xerox PARC, David Wetherall of the University of Washington, the DARPA Expeditions program and David Tennenhouse director of DARPA/ITO, and the the students of CS-590ES - particularly Kurt Partridge.

References

[1] G. Abowd and C. Atkeson. Future computing environments: Cyberdesk. Technical report, Georgia Institute of Technology, 1998. http://www.cc.gatech.edu/fce/cyberdesk/.

[2] P. Chou, R. Ortega, K. Hines, K. Partridge, and G. Borriello. ipChinook: An integrated IP-based design framework for distributed embedded systems. In *DAC-99*, 1999.

[3] D. Comer. *Internetworking with TCP/IP*, volume 1. Prentice Hall, Upper Saddle River, NJ, third edition, 1995.

[4] World Wide Web Consortium. Extensible markup language (XML), 1998. http://www.wc3.org/XML/.

[5] S. Czerwinski, T. Hodes, B. Zhao, and A. Joseph. The service discovery service: Searching for computation power, 1999. http://www.cs.berkeley.edu/ czerwin/sds-project.html.

[6] N. Davies, A. Friday, G. S. Blair, and K. Cheverst. Distributed systems support for adaptive mobile applications. *Mobile Networks and Applications*, 1(4):399–408, 1996.

[7] W. K. Edwards, E. D. Mynatt, K. Petersen, M. J. Spreitzer, D. B. Terry, and M. M. Theimer. Designing and implementing asynchronous collaborative applications with bayou. In *UIST '97*, Banff, Alberta, Canada, 1997. ACM.

[8] O. Etzioni, S. Hanks, and D. Weld et al. Internet softbot research, 1998. http://www.cs.washington.edu/research/projects/softbots/www/softbots.html.

[9] Tim Finin. UMBC agent web, 1998. http://www.cs.umbc.edu/agents/technology/.

[10] K. P. Fishkin, T. P. Moran, and B. L. Harrison. Embodied user interfaces: Towards invisible user interfaces. Technical report, Xerox Palo Alto Research Center, Palo Alto, CA, 1998.

[11] HomeRF Working Group. HomeRF, 1999. http://www.homerf.org/.

[12] Object Management Group. Corba news, 1998. http://www.omg.org/corba/.

[13] J. Hartman, U. Manber, L. Peterson, and T. Proebsting. Liquid software: A new paradigm for networked systems. Technical report, University of Arizona, Tuscon, AZ, 1996.

[14] The HAVi specification. Technical report, Home Audio/Visual Interoperability Consortium, 1998. http://www.havi.org/.

[15] T. D. Hodes and R. H. Katz. Enabling "smart spaces:" entity description and user interface generation for a heterogeneous component-based distributed system. In *DARPA/NIST Smart Spaces Workshop*, Gaithersburg, Maryland, 1998. DARPA/NIST.

[16] T. D. Hodes, R. H. Katz, E. Servan-Screiber, and L. Rowe. Composable ad-hoc mobile services for universal interaction. In *Mobicom '97*, pages 1–12, New York, NY, 1997. ACM.

[17] S. Kent and R. Atkinson. Security architecture for the internet protocol, 1998. RFC 2401.

[18] A. J. Kunzman and A. T. Wetzel. 1394 high performance serial bus: The digital interface for ATV. *IEEE Transactions on Consumer Electronics*, 14(13):893–900, 1995.

[19] Microsoft. Microsoft COM technologies, 1998. http://www.microsoft.com/com/.

[20] Sun Microsystems. The NFS distributed file system. Technical report, Palo Alto, CA, 1995.

[21] Motorola. Motorola VoxML, 1998. http://voxml.motorola.com/.

[22] D. Norman. *The Invisible Computer*. MIT Press, Cambridge, MA, 1998.

[23] H. S. Nwana. Software agents: an overview. *Knowledge Engineering Review*, 1(3):205–244, 1996.

[24] File systems in a distributed computing environment. Technical report, Open Software Foundation, 1991.

[25] C. Perkins. SLP white paper. Technical report, Sun Microsystems, 1998. http://playground.sun.com/srvloc/.

[26] C. E. Perkins. IP mobility support, 1996. RFC 2002.

[27] C. E. Perkins and H. Harjono. Resource discovery protocol for mobile computing. *Mobile Networks and Applications*, 1(4):447–455, 1996.

[28] Universal Plug and Play Forum. Universal plug and play. http://www.upnp.org/.

[29] IBM Research. IBM T-Spaces project. http://www.almaden.ibm.com/cs/TSpaces/.

[30] Frequently asked questions about today's cryptography. Technical report, RSA Data Security, Inc., San Mateo, CA, 1998.

[31] Carnegie Mellon University School of Computer Science. Coda file system, 1999. http://www.coda.cs.cmu.edu/.

[32] Bluetooth SIG. Bluetooth technology, 1999. http://www.bluetooth.com/technology/.

[33] Universal serial bus specification, revision 1.1. Technical report, Compaq, Intel, Microsoft, and NEC, 1998. http://www.usb.org/developers/data/usb11.pdf.

[34] J. Waldo. Jini architecture overview. Technical report, Sun Microsystems, Inc., Palo Alto, CA, 1998.

[35] R. Want, K. Fishkin, B. Harrison, and A. Gujar. Bridging real and virtual worlds with electronic tags. Technical report, Xerox Palo Alto Research Center, Palo Alto, CA, 1999. Also in CHI 99 (supporting video available).

[36] R. Want, B. Schilit, N. Adams, R. Gold, D. Goldberg, K. Petersen, J. Ellis, and M. Weiser. The PARCTAB ubiquitous computing experiment. In T. Imielinski, editor, *Mobile Computing*, chapter 2, pages 45–101. Kluwer Publishing, February 1997.

[37] A. Ward, A. Jones, and A. Hopper. A new location technique for the active office. *IEEE Personal Communications*, 4(5):42–47, 1997.

[38] Wireless data primer. Technical report, Wireless Data Forum, Washington, D.C., 1998.

[39] M. Weiser. Open house. *Review*, 2, 1996.

[40] D. J. Wetherall, J. V. Guttag, and D. L. Tennenhouse. ANTS: A toolkit for building and dynamically deploying network protocols. In *OPENARCH '98*, San Francisco, CA, 1998. IEEE.

[41] L. Zhang, S. Deering, D. Estrin, S. Shenker, and D. Zappala. RSVP: A new resource reservation protocol. *IEEE-Network*, 7(5):8–18, 1993.