# A Framework for Universal Service Access using Device Ensemble

Z. Morley Mao, Randy H. Katz
{zmao, randy}@cs.berkeley.edu
Computer Science Division, EECS Department, University of California at Berkeley, CA, USA

*Abstract*— **As more wireless devices become available and new applications emerge for these diverse gadgets, there is an increasing need to intelligently adapt existing services (e.g., Web browsing, video streaming) to them. Adaptation is needed due to heterogeneous and vastly differing capabilities in hardware, software, and network connectivity. We argue that existing services should remain unmodified and a automatically created, adaptive, distributed service proxy architecture with properties of fault-tolerance, composability is necessary for creating truly mobile services using a collection of devices. These devices form a *device ensemble* or a virtual device. We demonstrate feasibility of such an architecture based on our previous work in wide-area service composition [1].**

## I. Introduction

Today, there exists a spectrum of devices with different networking, hardware, and software capability. At one end of the spectrum are small mobile devices with relatively poor network connectivity, low computational power, limited memory and battery supply. Examples are various PDAs (e.g., Palm and iPAQ Pocket PC), pagers, smartdust motes [2], and cellular phones. At the other end of the spectrum exist more powerful devices such as desktop machines, laptop computers, and public kiosks. There is an inevitable mismatch between existing services and devices in the former group. To access a service using such a device, the current model of service adaptation neither scales nor enables rapid service deployment due to the model of vertical integration. WAP [3]-enabled cell phones, for example, requires Web pages to be encoded in WML before they can be displayed for small devices. Reencoding all existing Web pages using WML is an impossible task. Furthermore, such a solution is not general because future hand-held devices may have completely different display and rendering capabilities, thus requiring different levels of transcoding. We propose a distributed proxy architecture (Figure 1) that reconfigures itself on demand to adapt existing services to any access device.

Our work is also motivated by the availability of a large number of heterogeneous devices using potentially different networks. One class of devices is personal devices like cell phones and pagers. Another class is infrastructure devices, e.g., public kiosk displays, terminals at coffee shops. Each device has strength in certain types of applications. For example, cell phones are good for low latency, reliable voice channel. Traditionally a given service is targetted at one device at a time. However, users may receive better service if a number of devices are *simultaneously* used with the content split among them appropriately depending capabilities of individual devices. Based on the properties of each device, users may also wish to use multiple devices as the I/O channels. For example, the display screen of a cell phone is rather limited in size and resolution. A user in the vicinity of a large projector screen may decide to handoff the video stream to the larger screen while continuing using the cellphone for the audio output. Similarly, users may wish to use their PDAs as a input device rather than the touch screen of a crowded public kiosk. Ubiquitous network connectivity should enable a service to be controlled by a collection of devices at any given time. Such interaction should enable service mobility across devices as users choose to receive service content on different devices as they become available.

Even though wireless devices may not have continuous network connectivity due to the limited coverage area, some wireless devices may have multiple network interfaces. For example, a laptop computer may use a Ricochet modem at one time and a 802.11 inbuilding network at another time. Thus, to hide temporary network disconnectivity and potential changes in IP addresses, the service infrastructure needs to provide support for service session handoff. Our proposed architecture aims at solving this problem to provide true service mobility across different networks. Our architecture differs from previous solutions in providing multi-device service access, mobility support, and handoffs across devices.

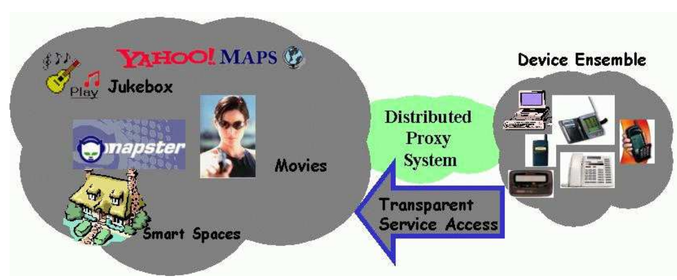


Fig. 1. **Universal Service Access:** This figure demonstrates our vision of ubiquitous service access.

## II. Motivating Scenario

We describe our motivating scenario that drives the design of our system. Alice wakes up in the morning and is reminded by her Palm that today is her mother's birthday. She fires up a video conference session to wish her Mom happy birthday. She sits down in front her desktop machine with an attached Web camera and picks up her home PSTN phone which rings as soon as her Mom answers. The video stream is automatically directed to the large desktop PC monitor. During the video conferencing call, she is reminded again by her Palm that she is running late for an appointment, thus she pushes the button on her home phone to transfer the call session to her cell phone. She also transfers the video streams to her Palm with CDPD network connectivity.

Since CDPD has a much lower bandwidth, the image size and quality is automatically reduced. The video input is received by the camera attached to the Palm. After completely transferring the video conference session, Alice walks into her garage and gets into her car equipped with a car cell phone and a CDPD network interface. She puts on the headphone set and docks her cell phone to transfer the call session to the car cell phone. The video stream is then automatically redirected to the dashboard display screen.

Later in the day, Alice goes to the shopping mall to buy a watch as a birthday gift. She first walks up to the information kiosk consisting of a large LCD flat panel display connected to a PC which is in turn connected to the server backend of the shopping center via an IP network. The shopping center has various indoor cell phone base stations for different cellular phone standards such as AMPS, CDMA, GSM, etc. As soon as Alice steps into the shopping center, her cell phone registers with the indoor base station. Now, Alice is standing in front of a kiosk which displays a simple number – *136. The number 136 indicates the actual location of kiosk. Alice chooses to dial *136 from the cell phone rather than waiting for the courtesy phone to be available. Then, the main menu of the help system pops up on the display of the kiosk. Alice can either use the keypad on her cell phone or speak directly into the phone to navigate the help system and find a list of stores and their locations within the mall. Alice then has a choice to download these data onto her cell phone or PDA through an IR connection, obtain a hard copy of the data using a printer attached to the kiosk, or simply store the voice instructions on the server of the mall.

## III. Design Goals

To support the above scenario, we propose a distributed service proxy architecture with the following design goals.

- **Seamless Handoff across Devices:** To provide support for user mobility, network variations, users frequently need to handoff a service session from one or more devices to another set of devices. Our service proxy system must support policy-based, user preference-driven handoffs across devices seamlessly – with minimal service disruption. The handoff decision is mostly initiated by the user who may decide to receive service on a different set of devices. However, the proxy system also automatically redirects the data stream to a different set of devices when needed. This occurs in two scenarios: Any device currently receiving service content becomes disconnected due to network congestion or lack of network coverage, or another device with better capability as specified by user's preference and policy becomes available to receive service content. Such handoffs should occur with little noticeable overhead.
- **Device Ensemble:** Rather than using a single device to receive the service content, users may wish to use a collection of devices (i.e. a *Device Ensemble*) to access the service based on device properties such as power consumption, display size, network connection speed. Similarly, the device ensemble also enable users to use different modes (e.g., speech, text menu) from potentially different networks to access services. Our proxy architecture must achieve multi-modal service access.
- **Automation:** The creation of data path for the adaptation of service content to heterogeneous devices should be automated. The capability negotiation between devices and services for the purposes of selecting the device ensemble to receive the service content and for device handoffs also occurs without user intervention.
- **Transparency:** The proxy system must be completely transparent to the legacy services and incur minimal changes to the client applications.
- **Quality of Service Guarantees:** For a large class of latency-sensitive, real-time applications, the infrastructure must deploy various mechanisms to reduce latency, jitter and optimize quality of service.
- **Fault-tolerant Service Composition:** *Service Composition*, similar to Unix pipeline-like chaining means that a service's output data feeds as input to another service, which in turn may get input from more than one service's output data flow. As a result, the functionality of the services are *composed*. When multiple services are composed, the composed service should be robust against process, machine, and network failures.

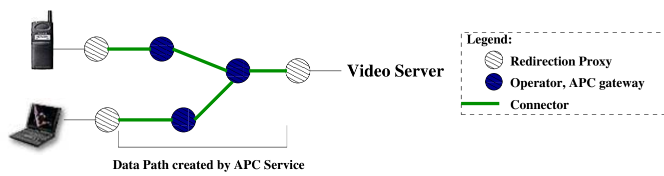


Fig. 2. **APC Proxy System:** This figure shows the APC Proxy System's main architectural components.

## IV. Design

### A. APC Proxy System

We propose a distributed service proxy architecture shown in Figure 2 to achieve the above design goals. We name our proxy architecture *APC (Automatic Path Creation) Proxy System*, for it automatically creates the data flow between a service and a device ensemble. Below we describe various architectural components and important design decisions.

**IP-based Core:** We use IP network as the unifying core network for integrating various networks to which services and devices are connected to (e.g., Pager, cellular, and Satellite networks). The reason for Internet-based integration is the following. Internet has an open service model allowing rapid creation and deployment of proxies. Internet has a well-developed client-proxy service model. Components can be incrementally deployed and scaled, because computational resources are relatively inexpensive given incrementally scalable compute clusters of commodity PCs.

**APC Redirection Proxy:** To achieve seamless handoffs across devices and service mobility across networks, we create a light-weight redirection proxy for each service and device ensemble. It receives all outgoing data from the service or device. All incoming data to the service or device also comes from the redirection proxy. If a device is powerful enough, the device redirection proxy can live directly on the device handling all its incoming and outgoing connections. The primary purpose of APC redirection proxy is to enable transparent data stream redirection for service session handoffs across devices and device network interface changes.

Another goal of APC redirection proxy is to provide transport level adaptation. For instance, data may be encoded using forward error correction (FEC) codes to achieve error resilience on wireless links. FEC decoding is done on the device redirection proxy before delivering data to the client. Similarly, data buffering, prefetching, caching, and retransmission are also used to optimize performance. To determine which optimization mechanism is needed, the proxy monitors the packet loss rate, jitter, and delay. It then selects a set of adaptations for optimizing QoS.

**APC Gateways:** APC gateways are machines in compute clusters that provide interface between IP network and another network. For example, GSM-IP gateway is a gateway that interfaces a GSM network to the Internet. Any non-IP network requires a gateway for integrating devices from heterogeneous networks.

**Transformation Operators and Connectors:** To handle data format and capability mismatch between a service and a device, transformation of data and protocols is performed between the device and server redirection proxy. We define two abstractions: operators and connectors. An *Operator* has clearly defined input and output types and is only responsible for computation. It is agnostic of the actual transport mechanism used by a connector for delivery to the user. A *Connector* is an abstraction of the ADU transport mechanism between two operators. This abstraction encapsulates various properties of the transport such as reliability, order of delivery, and drop policy. The key advantage of this abstraction is that a connector hides the potential differences in network protocols from the operators and allows them to communicate as long as the output data type of the downstream operator matches the input data type of the upstream operator.

APC Proxy System supports four types of operators classified according to its functionality. Various operator properties such as input, output type, classification, etc. are encapsulated using XML to facilitate automated path discovery.

- **Data Format Translation Operators.** This type of operator performs conversions between different data formats (e.g., GIF to JPEG, PCM to GSM, powerpoint to HTML).
- **Protocol Conversion Operators.** These include translation between different security protocols. For example, an operator that converts between the heavy-weight security handshake protocol such as SSL to simple shared-key encryption and decryption falls into this category.
- **Content Transformation Operators.** This type of operator filters or aggregates the information produced by the previous operator. Another example would be a language translation operator from English to Chinese. or an operator giving a summary of a paragraph.
- **Optimization Operators.** This category of operators does not change the semantic content of the data. Instead, they modify the properties such as data size and security guarantees. These can be lossy or lossless compression operators to optimize bandwidth usage, encryption operators, and FEC operators to lower data error rate. The tradeoff of adding these operators is improved resource utilization but more operation complexity and potentially increased latency.

To handle the I/O mismatches between legacy operators (any existing software), we create a high performance connector library to convert different I/O methods (e.g., file, network, disk I/O). Such a library alters the behavior of communication protocol of legacy operators transparently.

**APC Service:**

**Data Path Creation:** APC service is a wide-area cluster service that automatically constructs a the data path (a chain of operators joined by connectors). Given application-specific optimization metrics, APC finds an optimal sequence of transformation path by conducting the shortest path search on a graph modeling the space of operators. Optimality is determined by the application metric. Any vertex of the graph denotes a data or protocol format. Each edge represents an operator performing the translation between the two formats with a value denoting the cost of operation. Using Dijkstra's shortest path search algorithm, the running time is $O(E \log V)$, where V is the number of vertices in the graph, and E is the number of edges.

**Computation Placement and Migration:** APC provides a wide-area *Computation Distribution Network* (ComDN) for purposes of making intelligent operator placement and migration decisions. Analogous to *Content Distribution Network* (CDN) like Akamai, a ComDN distributes the computation rather than content for better performance, access latency, and robustness to failures. Moreover, it also dynamically *relocates* the computation during a service session when performance becomes unacceptable. For example, when the network becomes congested between two operators, APC relocates one of them so that data path can avoid such network bottleneck.

A ComDN consists of compute servers where operators execute. It keeps track of existing running operator instances and monitors the load of compute servers. For load and network monitoring, it deploys a *wide-area monitoring service*, combining techniques of both passive and active monitoring. It collects both application-specific and application-independent measurements in a soft-state based distributed database.

ComDN decides which operator instances to use for a particular data path and on which compute servers to place dynamically created operators. This decision is based on the network and server load, locations of input data sources and output data sinks, and overhead of path creation. Application-specific QoS metrics (e.g., latency, throughput, image resolution, etc.) is used to determine the best location.

Not only is ComDN responsible for computation placement, it also makes operator migration decisions. In a transcoding data path, computations are often easily relocatable due to soft state nature of the operation. For this class of stateless operators, their locations are not fixed in the network and can be changed for the purpose of performance enhancement. For example, during a real-time service session, the network and server load may increase, resulting in unacceptable latency. This can also occurs due to service handoff to a device using a different network, or due to roaming of high-speed mobile users.

During runtime, the monitoring service of ComDN notifies the APC Service when the data path performance goes below the acceptable threshold. APC constructs a new data path by migrating computation state of existing operators to other newly selected computation sites by ComDN. Each operator has application specific service session state which are to be encapsulated and sent over to the new location (e.g., cookies for Web browsing, transcoding parameters for data transformation). In addition to operator migration, APC also performs automatic insertion and deletion of computation in the data path to adapt to changes in server and network load.

**Proactive Adaptation to Resource Variations:** Here we generalize the techniques APC uses to adapt to dynamic resource variations during a path execution. Frequently end users may experience degraded performance of the service due to dynamic changes in network conditions, e.g., sudden drop in bandwidth due to network congestion, or other resources such as computational cycles and memory space. There is a need for applications to adapt to dynamic changes in available resources to optimize user's perceived quality of service. We define three ways to adapt to changes in resources. These adaptations are performed only after the effect has been observed to persist beyond a threshold amount of time, since reacting to transient resource changes result in unjustified overhead and instability of the system.

- **Application-intelligent adaptation**

The application is powerful enough to do its own adaptation to resource changes. For instance, RealAudio combines multiple streams encoded for different bit rates into a single clip, and RealVideo uses a single codec to encode data for all bandwidths. During runtime, the audio and video streams dynamically adapt to changes in bandwidths [4]. In this case, APC should directly take advantage the application adaptation mechanism in the composed path.

- **Application-specific adaptation**

The application provides mechanisms for dynamic adjustment, but does not do so automatically. For instance, for bandwidth adaptation, there are different instances of the same codec intended for different bit rate. A codec may be error-resilient, but needs to be notified of the current error rate through a control channel. In this case, APC is responsible for monitoring the resource changes and providing the feedback to the applications to enable dynamic adaptation.

- **Application-independent adaptation**

If there is a lack of knowledge of the underlying implementation of the application, APC treats it as a black box and does application-independent adaptation. For instance, to adapt to high packet loss rate, forward error correction (FEC) and compression operators are inserted for better data throughput. FEC can also vary the amount of redundancy based on the packet loss rate.

Combinations of the above approaches are used. Given the application-specific optimization criteria, APC strives to create service compositions that best utilize network and computational resources to achieve the optimal desired QoS. Optimized resource utilization and differentiated QoS are enabled by our iterative data path construction process with continuous feedback and clear specification of optimization metrics.

### B. APC Proxy System Control Signaling

To realize our motivating scenario in Section II, we now describe the control signaling to enable seamless service access using a device ensemble.

**Device Discovery:** As part of the APC Proxy System, there is a local-area device registry which keeps track of available devices, their properties such as ownership, compute power, battery life, memory, etc. It uses a IP multicast channel to discover any new devices in the local area network. If devices are bluetooth-enabled or have a common network interface, they can discover each other and directly report their neighbors to the device registry.

**Capability Negotiation:** When a user requests a service using a device ensemble, the redirection proxy of each device is contacted which performs a capability negotiation between the requested service and the device ensemble. Using the information from the device registry, it determines which set of devices are best suited for receiving the service content and how the service should be best delivered. It then makes a data path creation request to the APC Service. This decision is performed by examining the properties of each device. For instance, a cell phone provides a relatively low-latency, reliable voice channel. It is easy for punching in menu keys. Its coverage is almost ubiquitous. However, it has very limited screen display size and has relatively low battery life. Thus, a cell phone is useful as a voice-based or key-based control channel and audio output channel. In contrast, a desktop PC has much more capability in memory, compute power, and a higher fidelity graphical display. Nevertheless, it is not mobile. Thus, it is useful for video display only when a user intends to stay in a fixed location.

**Interaction with a Device Ensemble:** One potential problem of utilizing a collection of devices simultaneously is that synchronization may be necessary among the devices. For instance, if audio is delivered to the cell phone and video is streamed to the desktop PC, it is easy for data to get out of sync. We propose that the redirection proxies periodically send each other sequencing messages to resynchronize. These messages can be application-level sequence numbers of simply RTP timestamps if RTP is used as the transport.

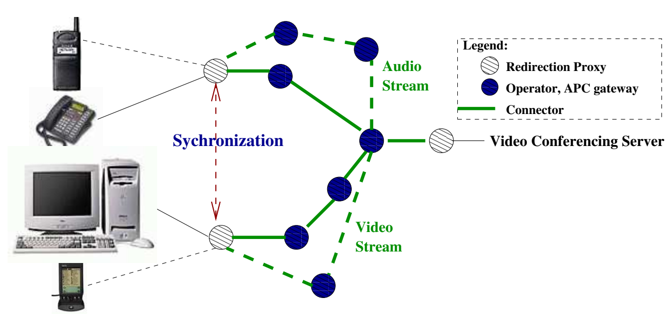


Fig. 3. **Device handoff:** This figure demonstrates our scenario of handing a video conference session from a PSTN phone and a PC to a GSM and Palm Pilot. Dashed lines represent data streams after the handoff.

**Policy-based Handoff:** A handoff can be either initiated by the user or suggested by the APC Service. The latter case occurs when device receiving the service becomes disconnected or overloaded. We now step through the scenario (Figure 3) of Alice handing off the video conferencing session with her Mom from her home PSTN phone and PC to her GSM cell phone and Palm Pilot. Alice first dials *#C* to indicate her wish to handoff her conversation to her cell phone. The redirection proxy receives the control signal for handoff and looks up the Alice's cell phone number in the device registry. It contacts the APC Service to create a new data path to a different end point – cell phone. It then dials the number through the GSM-IP gateway. As soon as Alice picks up her cell phone, the redirection proxy immediately redirects the voice data stream of the path created by APC to the cell phone. During the handoff, the proxy buffers some audio data and replays them to minimize the disruption. To handoff the video stream of session, Alice selects the corresponding menu on her desktop PC to indicate her wish to handoff to her Palm Pilot with CDPD connectivity. The device redirection proxy requests APC to build a new data path to the Palm Pilot. APC automatically inserts a distillation proxy to reduce video image resolution and size. After receiving the data stream, the proxy redirects the data to the Palm Pilot.

### *C. Modifications to Client and Server Applications*

The APC Proxy System is entirely transparent to services. There is no need for modification on the server side. To use the APC proxy service, devices need to provide a handoff registry user interface which allows users to select to which other devices to handoff the service session. The list of available devices is available from the local area device registry. The handoff registry directly contacts the device's redirection proxy for performing handoffs. Client applications need to be wrapped by a simple wrapper service that redirects all communication to the outside world to the client redirection proxy. There are no other changes needed.

## V. Development Status

As part of the ICEBERG project [5] we have already implemented handoffs of telephone conversation across PSTN, GSM, desktop IP phones. We have also built GSM-IP gateway and have created simple services such as reading news headlines from the New York Time on a GSM cell phone and use the cell phone to send voice commands to control A/V equipment of a smart room. We have built a prototype cluster-based APC Service for wide-area service composition. Existing composed service applications include listening to MP3 songs on a cell phone, watching a real-time MPEG-1 transcoded data stream on a Real Player. We have an extensive collection of operators doing conversion of various audio and video formats and between text and speech. As part of the APC Proxy System, we have experience in providing mobility support using adaptive FEC and data stream redirection for wireless clients who may change IP addresses during a real-time service session.

## VI. Related Work

There are a few areas of related work. Here we briefly examine them and discuss how our work distinguishes from them.

- **Transformational proxies:** Our work is heavily influenced by the TACC [6] programming model; however, TACC focuses on the last hop issue and is limited to a single proxy architecture. Conductor [7] remedies in this regard, but does not address issues of dynamic insertion, deletion, and migration of proxies to adapt to server and network load variations. Furthermore, Conductor does not provide mobility support or handoffs across devices.
- **Handoffs:** Our work has benefitted from ideas of projects that provide policy-based handoffs between networks (i.e. Vertical Handoffs [8]) and between cells in a single network (i.e. Horizontal Handoff [9]). We extend this idea to handoff across devices which encompass both of these types as well as handoff of service session.
- **Mobility support:** Mobile IP [10] provides a solution for dealing with IP address changes at the network layer level where the Home Agent keeps track of the current location of the mobile user. This approach, similar to IP Multicast, requires changes to the IP network and hence can be very difficult to deploy. Snoeren and Balakrishnan [11] proposes the addition of a TCP migrate option to migrate open TCP connections. Their approach requires adding a TCP option which requires the installation of a new TCP stack on both the client and server. Instead, we propose a solution at the application forwarding proxy level where the IP addresses are allowed to change. Our approach requires neither changing the IP layer nor the TCP layer. This is possible because APC has complete knowledge of the entire network path.

## VII. Conclusion

In summary, the contributions of our APC Proxy framework are the concept of device ensemble allowing mult-device service access, automated data path creation for content adaptation, runtime resource adaptation using computation relocation, insertion, and deletion, and seamless policy-driven handoffs across devices to support user mobility. Most importantly, all of the above goals are achieved transparent to existing services and imposing minimal overhead.

## References


[1] Z. Morley Mao, Eric A. Brewer, and Randy H. Katz, "Fault-tolerant, scalable, wide-area internet service composition," Tech. Rep. UCB//CSD-01-1129, U.C. Berkeley, January 2001, Available at `http://www.cs.berkeley.edu/~zmao/Papers/techreport.ps.gz`.

[2] Jason Hill, Robert Szewczyk, Alec Woo, David Culler, Seth Hollar, and Kristofer Pister, "System architecture directions for networked sensors," in *Proceedings of ASPLOS 2000*.

[3] WAP Forum, *Wireless Application Protocol (WAP) Forum*, `http://www.wapforum.org`.

[4] Real.com, `http://service.real.com/help/library/blueprints/8codecs/producer8codecs%.html`, *Working with RealProducer 8 codecs*, June 28, 2000.

[5] H. J. Wang, B. Raman, C-N Chuah, R. Biswas, R. Gummadi, B. Hohlt, X. Hong, E. Kiciman, Z. Mao, J. S. Shih, L. Subramanian, B. Y. Zhao, A. D. Joseph, , and . R. H. Katz, "Iceberg: An internet-core network architecture for integrated communications," *IEEE Personal Communications*, August 2000.

[6] Armando Fox, Ian Goldberg, Steven D. Gribble, David C. Lee, Anthony Polito, and Eric A. Brewer, "Experience with top gun wingman: A proxy-based graphical web browser for the 3com palm pilot," in *Proceedings of Middleware '98, Lake District, England*.

[7] Mark Yarvis, Peter Reiher, , and Gerald J. Popek., "Conductor: A framework for distributed adaptation.," in *Proc. Seventh Workshop on Hot Topics in Operating Systems (HotOS VII), Rio Rico, AZ, March 1999.*

[8] Mark Stemm and Randy H. Katz, "Vertical handoffs in wireless overlay networks," *ACM Mobile Networking (MONET), Special Issue on Mobile Networking in the Internet*, winter 1998.

[9] S. Seshan, *Low-Latency Handoff for Cellular Data Networks*, Ph.D. thesis, U.C. Berkeley, December 1995.

[10] C. Perkins A. Myles, D. B. Johnson, "A mobile host protocol supporting route optimization and authentication," *IEEE Journal of Selected Areas in Communication*, vol. 13, no. 5, pp. 839–849, June 1995.

[11] Snoren and Balakrishnan, "An end-to-end approach to host mobility," in *Mobicom 2000*, `http://www.research.att.com/conf/mobicom2000/papers.html`.