

InfoNet: the Networking Infrastructure of InfoPad

My T. Le, Fred Burghardt, Srinivasan Seshan, Jan Rabaey
Electrical Engineering & Computer Sciences Department,
University of California, Berkeley

Abstract

This paper provides an overview of InfoNet, the networking infrastructure for the InfoPad mobile computing system. First, the goals and architecture of InfoNet is defined. Next, the current implementation, performance measurements, and proxy connections are discussed. Finally, we will present the future directions for InfoNet.

1. Introduction

InfoPad [1] is comprised of three main components: portable terminals, *Pads*; multi-media applications that include motion video, audio, pen, text, and graphics [2]; and a communication system.

The communication system, *InfoNet*, includes wireless links, a wired backbone network, and software modules. The lack of a general purpose processor in the Pad distinguishes InfoPad from other Personal Computing Systems (PCS) in an important way: all processing is performed by computing nodes on the backbone network, and data transferred between network and Pad are displayed with minimal operations by the Pad. Because the Pad only serves as a displaying device with little local intelligence, large amounts of data will be transferred from the applications to the Pad. Thus, the primary goal for InfoNet is to provide an infrastructure in which these data transactions can occur quickly and efficiently.

The InfoNet architecture is composed of both hardware components and software modules (see Figure 1). Hardware components include wireless links and a backbone network:

- Wireless link:

Pad is connected to the backbone network via two wireless links, a downlink from the network to Pad and an uplink from the Pad to the network. Currently, the downlink is implemented using commercial Plessey radio components operating at a data rate of 700kbps. The uplink is implemented using a Proxim radio components operating at a data rate of 244kbps.

- Backbone network:

10Mbps Ethernet and 155Mbps Asynchronous Transfer Mode (ATM) networks are used as the backbone network.

Besides the wired and wireless physical links described above, all other components required to provide the communication link between applications and Pads are implemented in software. The software modules can reside on one or more workstations. Software components include the PadServer, CellServer, Gateway, and Network Controller. Their functions are as follows:

- PadServer:

For each Pad, there is a PadServer running on the backbone network. The PadServer is responsible for managing and controlling access to the Pad. It allocates the bandwidth and resources (speaker, microphone, etc.) among the different applications.

- CellServer:

There is a CellServer associated with each cell. This CellServer controls the allocation of resources (i.e. power and bandwidth) among the Pads within the cell.

- Gateway:

In addition to a CellServer, a Gateway is also associated with each cell. The Gateway connects a cell of the InfoPad's proprietary wireless network to a standard backbone network, such as Ethernet or ATM. The Gateway is responsible for converting protocols between the wired network and the wireless network.

- Network Controller:

The network controller's main responsibility is to enable the creation of connections between InfoNet entities. It must provide name service support for the Pads, PadServers, CellServers and Gateways and maintain information about the physical location of these elements. In some network technologies, such as ATM, the network controller may also route and create the connections between the different entities.

To allow development efforts in the InfoPad project to proceed concurrently, the current InfoNet prototype has been implemented to operate on the backbone network with no specific knowledge of the wireless links or the applications. To operate independently from the Pad and the wireless link, InfoNet software uses a software entity called the Emulator (Emu) that has been implemented to emulate the actions of a hardware Pad.

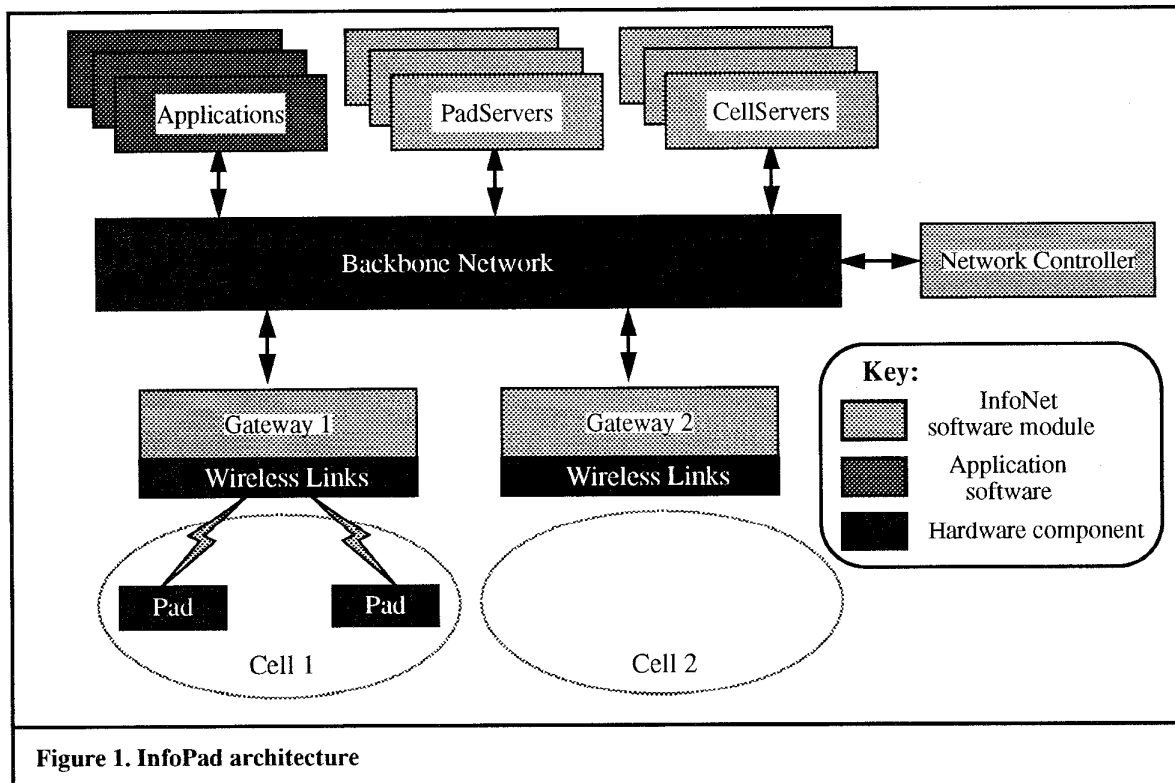
InfoNet software exchanges data with the applications using a programming interface called TypeServers. TypeServers serve as management agents for the four data types in InfoPad: video, audio, pen, and text/graphics. Thus other InfoNet software modules do not need to be concerned with type-dependent information. An overview of the current InfoNet implementation is shown in Figure 2.

2. Results

In this section we will present two results: one that describes the performance measurements of InfoNet and the other presents a new and improved way to establish connections between applications and the Pad.

2.1. Performance Measurements

This section is a brief summary of performance measurements of a non-threaded implementation of the InfoNet software architecture. In this case, non-threaded means that all InfoNet modules are implemented with a single Unix process using a state machine based algorithm to achieve effective concurrence. Independent state is assigned to each port, allowing a simulated *sleep* when a port is not in use. The current implementation has been run primarily under SunOS 4.1.3, which does not support efficient threads. The next implementation will be multi-threaded under Solaris 2. The InfoNet module architecture fits naturally into an environment that supports multiple threads of control, but for performance reasons the standard Unix approach of multiple heavy-weight processes communicating via inter-process communication channels was not used. The proposed multi-threaded implementation is expected to improve on the results reported here.



The architecture tested also does not support proxy connections. Proxy connections are data channels running directly to a Gateway, under the control of a PadServer but not through the PadServer. In this performance measurement, the PadServer multiplexes all data types into a single stream, which is sent to the Gateway. The PadServer also de-multiplexes an uplink stream into data types. Proxy connections are described in section 2.2 and will be added to the performance measurement later. Figure 3 shows the software topology used for the measurements. The network is a single segment Ethernet.

These results are important for InfoPad research and development, but should not be taken as a performance target. They are intended to serve as a baseline for comparison with later measurements taken at important project *checkpoints*, such as implementation of a high-speed Basestation (Gateway)

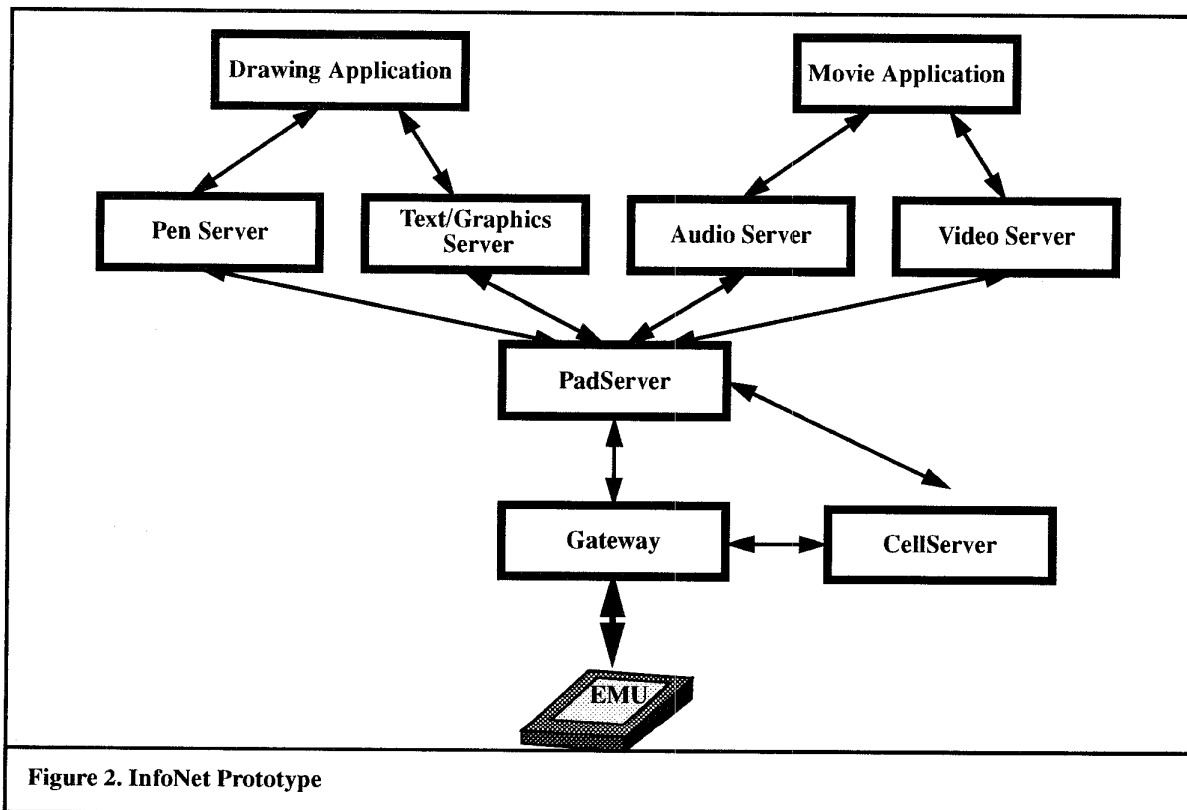
Throughput and delay were two of the most important parameters. Throughput is expressed as end-to-end bits per second, although due to the network topology each measured bit makes three network hops from production

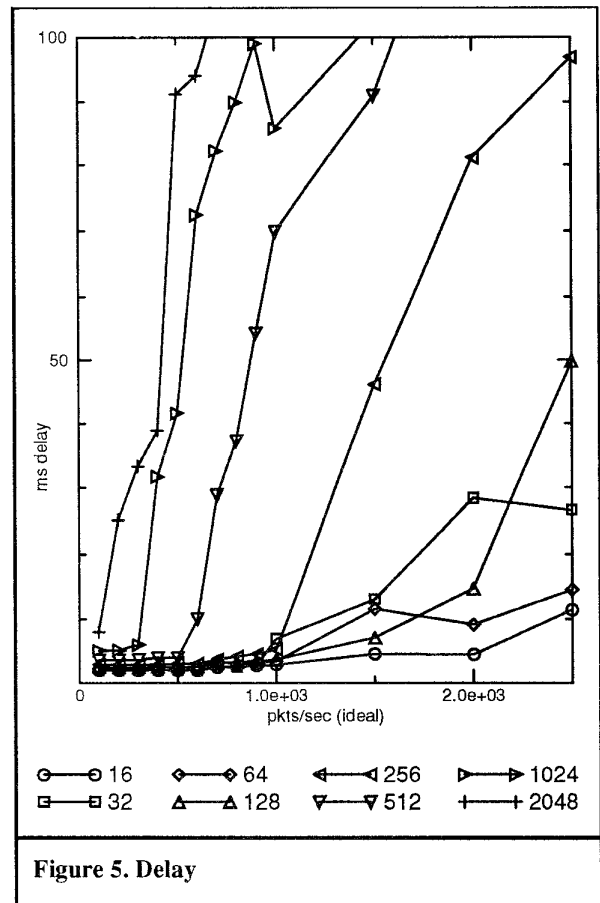
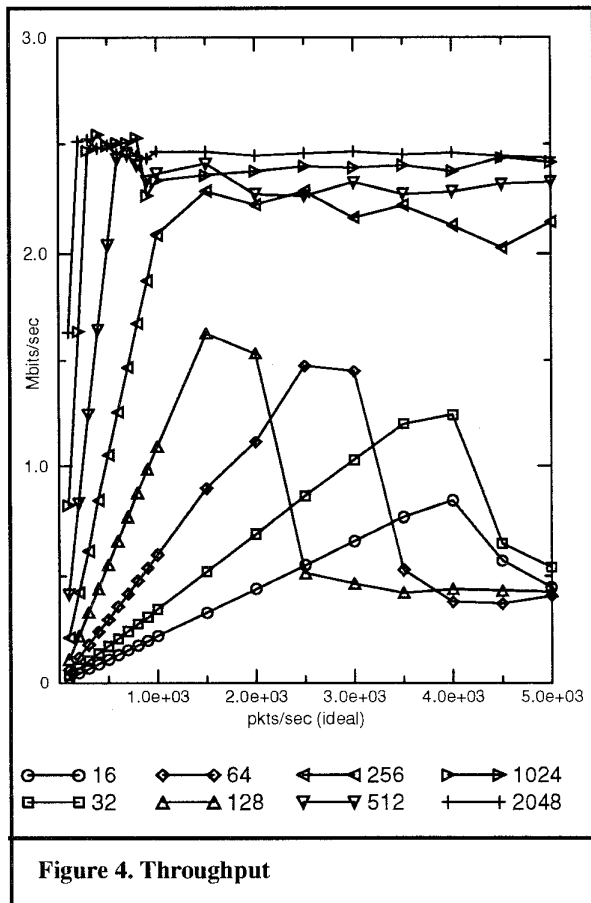
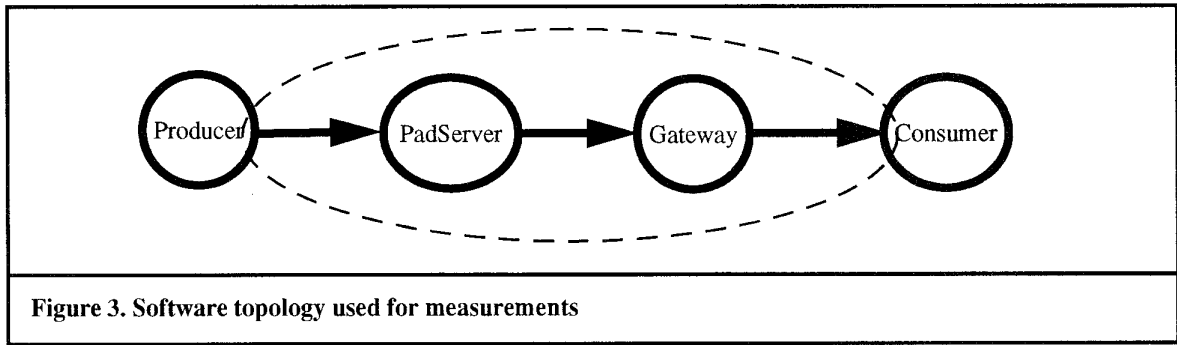
to consumption so aggregate network traffic is three times the net throughput. Delay is expressed as time elapsed between a timestamp taken in the producer immediately before a *Write()* call and a timestamp taken in the consumer immediately before a *Read()* call.

A test is defined as a 30 second burst of packets of a specific size at a specific constant rate. Figure 4 and figure 5 show throughput and delay for each test. The notation *ideal* in the x-axis label means that the packet rates shown are what the producer is attempting to send. The actual packet rate degrades beyond a certain point, as implied by the non-linearity of the throughput curves at higher packet rates.

The packet rate for approximate real-time behavior is under 800 packets per second for the smallest packets.

As packet rates increase further, the Gateway host CPU is seldom idle and spends most of its time in the kernel processing read and write system calls. In addition, the system call rate is approaching maximum for the CPU (other tests have verified that system call rate is directly related to CPU clock speed). The main reason for packet rate limits is the overhead of kernel crossings; an earlier profile of the Gateway executable shows that about 90%





of CPU time is spent in system calls, almost all of which are reads and writes.

Throughput tops out at about 2.5Mbits per second. This appears to be directly related to Ethernet saturation (the aggregate network throughput is three times the measured end-to-end throughput, giving an aggregate maximum of about 7.5 Mbits per second). The remainder of the 10Mbits per second Ethernet bandwidth is

consumed by protocol headers and incidental traffic, and wasted by a fairly large number of collisions.

As the graph shows, delay is under 10ms until the point at which throughput becomes non-linear with linear increase in ideal packet rate.

These measurements produce or verify three major findings:

1) Packet rate is limited primarily by the maximum system call rate of a host, closely associated with CPU speed. Proxy connections will not require that data pass through the PadServer, so any packet rate bottleneck will be in the Gateway. For this reason, we are actively researching alternative architectures for the Gateway.

2) Throughput is limited, at least in these tests, by the network. It is expected that with ATM, throughput will be limited by the same factors that limit packet rate with additional penalties for large payload byte copies.

3) The system under test was remarkably well-behaved below specific packet rates for each packet size, as indicated by the linear increase in throughput under 1000 packets/sec and correspondingly low delay in the same region.

2.2. Proxy Connections

The current mechanism to route data from an application to the Pad is shown in Figure 2. Each application transmits its data to the appropriate TypeServers. The TypeServers

transcode the information into a format the Pad can use. For example, the Video Server may convert standard video streams (MPEG) into the Infopad compressed video format (VQ). The TypeServers transmit the transcoded output to the PadServer. The PadServer must then route this data to the Gateway connected to the Pad.

The main advantage of this scheme is simplicity. The PadServer is the only software entity that must maintain information about the Pad's current location. When a Pad moves between radio cells of the network, data destined for the Pad must be routed through a different Gateway. This is called handoff. In our current scheme, only the connection between the PadServer and Gateway must be modified during handoff. However, this simple design results in a significant performance penalty. Since the PadServer does not modify the data in any way, the transmission of data through the PadServer is unnecessary. To eliminate this extra copy of data, we use a technique called *proxy connections*.

Proxy connections allow a proxy entity to control the route and characteristics of a network connection. In our system the proxy entity is the PadServer and the connections are

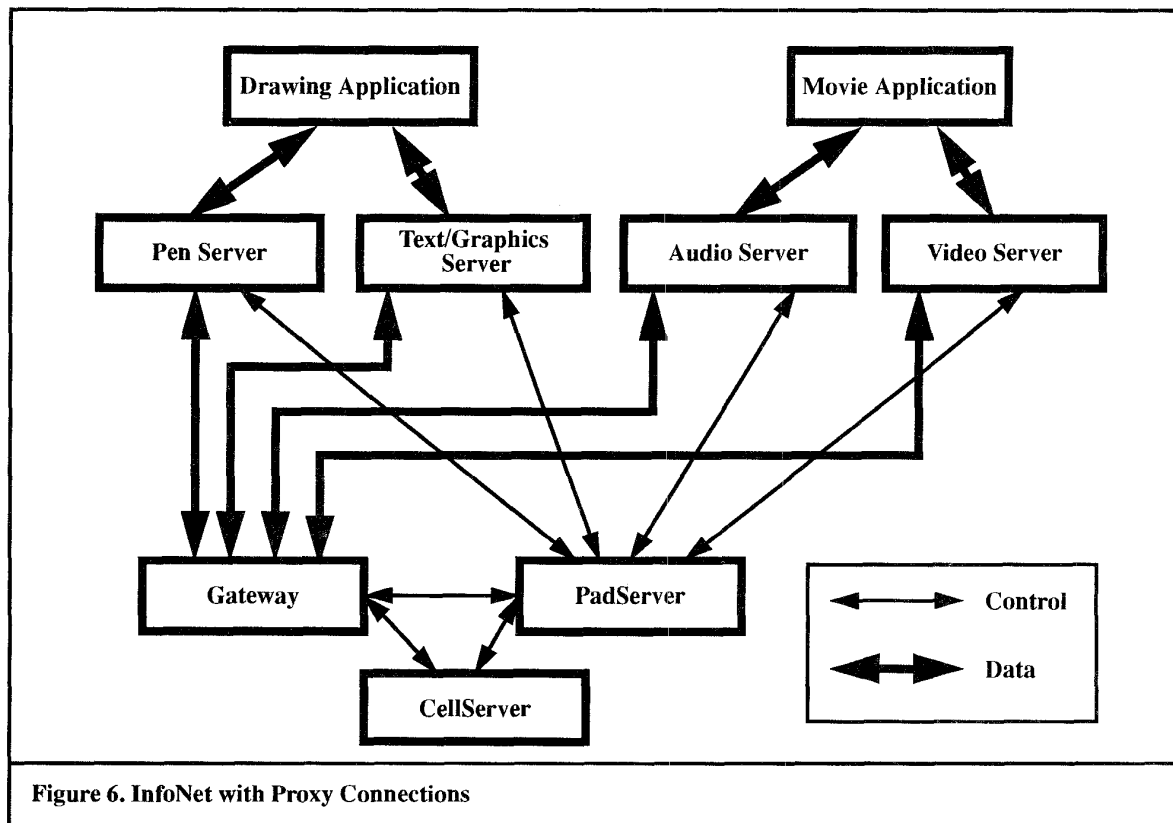


Figure 6. InfoNet with Proxy Connections

between the TypeServers and the Gateway. The TypeServers must still communicate with the PadServer to request bandwidth and resources on the Pad; however, data traffic from the TypeServers may be transmitted directly to the Gateway. This communication is shown in Figure 6. As before, only the PadServer keeps track of the Pad's physical location. During handoff, the PadServer modifies the route of each of the proxy connections to transmit to the correct Gateway.

3. Future Directions

Since the goal of the project is to provide multimedia services in a ubiquitous environment, future efforts will be focused in four areas:

1) Quality of Service:

The applications' QOS requirements (bandwidth, delay, reliability) need to be translated into backbone network and wireless link parameters (power level, error-correction, number of Pads per cell, etc.). These network parameters will then be used in the protocols that implement bandwidth control, error-correction, and power control.

2) Mobility:

A simple protocol for mobility is being implemented in an environment with two Pads and two cells. Further improvements will support mobility for large number of Pads over multiple cells.

3) Performance:

Ethernet is the limiting factor for throughput when there are multiple pads in the system. An ATM network operating at 155Mbps will be used in the next implementation to ensure that the Backbone network is not the bottleneck of the system.

4) System Integration:

Many of the topics discussed here involve complex modules. Each of the software module must operate properly with each other, and InfoNet as a whole must integrate well with the applications, radios, and Pads. Several efforts are underway in this area, including incorporation of proxy connections into the architectural model and building a hardware interface between a Gateway workstation and radio hardware.

4. Acknowledgment

We would like to thank all the members of the project for their tireless efforts in implementing Infopad. The architecture was developed by the InfoNet group whose members include: Fred Burghardt, Richard Han, My Le, Ken Lutz, Shankar Narayanaswamy, Jan Rabaey, Brian Rich-

ards, and Srinivasan Seshan. The Infopad project is supported by ARPA and numerous industrial corporations.

5. References

- [1] S. Sheng, A. Chandrakasan, R.W. Brodersen, "A portable multimedia terminal," IEEE Communications Magazine, December 1992, pp. 64-7.
- [2] Burstein, A., Long, A.C., Narayanaswamy, S., Han, R., Brodersen, R.W., "The InfoPad User Interface," COMPCON 1995.