

SNAP Computing: Wireless Location-based Plug and Play

Andrew Christian, Brian Avery, Steven Ayer, Frank Bomba, Jamey Hicks

Cambridge Research Laboratory
Hewlett-Packard Company
One Cambridge Center
Cambridge MA 02142 USA

{andrew.christian,b.avery,steven.ayer,frank.bomba,jamey.hicks}@hp.com

ABSTRACT

This paper outlines SNAP, a new research effort to create a wireless modular computing environment. SNAP computing eliminates the wires between components to give users the flexibility to mix, match, and share all available resources. Each component provides services as an intelligent, self-contained entity; e.g. keyboards and mice process user input, video screens and audio speakers give feedback, and disk drives provide storage. Device services can be shared, and persistent, peer-to-peer relationships are allowed. All components conduct authenticated, authorized and encrypted communications via IPv6 in a location-aware space. Our project goal is to develop, build, and test the architecture, protocols, and a representative set of components necessary to constitute the springboard for a new and novel approach to congregating computing elements.

Categories and Subject Descriptors

C.2.4 [Computer-Communication Networks]: Distributed Systems—*distributed applications*.

Keywords

Wireless networking, secure distributed computing, recombinant, modular component computing, location-awareness.

1. INTRODUCTION

The number and variety of communication and computing devices that people use every day has grown quite large. Because of the antiquated way in which these devices are connected and configured, setting them up and using them has become exceedingly frustrating. Why can't the laptop computer automatically play audio through those nice, expensive stereo speakers? Why can't the desktop computer's optical mouse automatically be used with the laptop? Why can't all of the computers in an office automatically share one flat panel monitor?

We seek a better world. We envision a world of plug-and-play, mix-and-match, wireless, shared resources that don't require you to be an expert to configure, use, and extend. We call this world *SNAP* computing.

Fundamentally, SNAP is about eliminating the wires between devices. Wires are not inherently bad things; on the contrary, they are often the most effective means to transfer power and signals between devices. However, wires become a problem when engineers build systems with hardware transceivers, low-level chipsets, and cables that enforce particular protocols and limitations, each specific to the type of low-level protocol used (USB, RS-232, RS-422, 1394, etc). This design principle limits the flexibility of the services presented by a new component and

frustrates the user, who must find the right combination of null modems, adapter cables/hubs, and other hardware flotsam in order to make all of the pieces that comprise a "computer" work together. Consider that a typical new PC comes with a video monitor, keyboard, mouse, microphone, headphones, audio speakers, color printer, and a beige box containing a floppy disk drive, hard drive, DVD or CD-RW drive, and network card. Once assembled, all of these components live in a fixed, subservient relationship to the processor in the beige box.

Wireless systems are not immune to this problem. For example, Bluetooth enforces the same master-slave relationship as USB, although each Bluetooth device can be either master or slave. A goal of SNAP is to remove the wires and make each component or device into a first-class service provider which may be shared among multiple applications or computers.

What are the elements necessary for SNAP?

First, eliminate all wires but the power cords. Eliminate power cords from devices that can use batteries.

Second, all components need to become intelligent, standalone entities with some form of network connectivity. They don't need to be full-blown appliances, which implies that they could be used in the absence of other components. They don't even need to provide user feedback; they may be as blank and as boring as the average mouse. However, they can't rely on other devices to do their thinking for them.

Third, *service sharing* needs to be intrinsic to all types of components, including ones not normally considered as sharable, such as video screens, audio speakers, and keyboards.

Fourth, we need mechanisms for setting up relationships between components and controlling their operation. For example, a person should be able to set up or modify the relationship between a DVD player, video screen, and set of speakers so that when playing a DVD, its video and audio signals automatically reach the right components. This person should be able to connect with a PDA or any other device to form a temporary relationship that will allow control over playback, volume, and display positioning.

The power of SNAP comes from allowing components to interact peer-to-peer without requiring exclusive relationships. For example, multiple audio streams to an audio speaker should be mixed and played simultaneously.

While it may be argued that SNAP will prohibitively add cost to each component, the cost of a built-in wireless transceiver has dropped to around \$2 in 2005 and is forecast to continue to decline. In contrast, cables and connectors are a surprisingly

expensive part of a computer system; a 17" color monitor may cost \$150, yet a new VGA cable can add \$25 to that figure.

The intent of this paper is to describe the broad vision and core technical issues of SNAP, not to describe a specific implementation. This paper is divided into the following sections: (a) characteristics of the SNAP environment, (b) a series of sample setup and usage scenarios, and (c) technical problems that need to be addressed. There are many good and useful technologies that may play a part in building a complete solution (e.g. Bluetooth, 802.11, 802.15.4/Zigbee, Universal Plug and Play, Jini, CoolTown, ZigBee, SIP, RDF, WSDL); however, this paper does not address the advantages and shortcoming of the various technologies and how they might be integrated into the overall architecture.

2. SNAP CHARACTERISTICS

SNAP components are self-contained wireless-enabled individual units that can be mixed, matched, and shared between users. To implement a solution, we see the need for the following key characteristics:

- Wireless self-configuring mobile communication
- Shareable self-describing services
- Location-aware persistent relationships
- Authorized, authenticated, secure communication

2.1 Wireless Communication

Each SNAP component is capable of communicating over a wireless link using internet protocols. Control information and low speed communications can be carried over 802.15.4 or infrared where line-of-site constraints are helpful. High speed communications may be carried over 802.11 or UWB links.

All basic wireless networking will need to be able to automatically configure itself without reference to a central repository of knowledge or assistance from the user; this requirement strongly suggests the IPv6 stateless autoconfiguration mechanism.

Over the course of time the available wireless network may change as components come and go, the things move around, and power comes on and off. The SNAP component must continue to locate and acquire suitable connectivity and fix routing information.

Wired links should always be optional and parallel to the existing wireless link, used only because it lends some speed or power advantage. For example, your video screen may suggest a 1394 cable between it and a DVD player to enhance playback speed.

2.2 Services

All components provide at least this minimal list of shared services, potentially accessed or subscribed to by multiple components:

- Discovery protocol: the component must be discoverable by other components.
- Service descriptor: a meta-service that describes the services provided by the component.
- Configuration interface: specifies security settings, relationships with other components, and standard behaviors.

- Diagnostic interface: access run-time statistics, error logs, and firmware updates.

Any two SNAP components with compatible services should be able to set up a basic relationship with no human assistance.

Services operate in either "pull" or "push" mode, meaning that a service can be interrogated by or send data to an outside device. For example, a keyboard can provide a *KeyPress* service that sends individual key presses (good for games/editors) or processed Unicode (good for text editors) to interested components. Because the keyboard is sharable there can be multiple simultaneous clients: e.g. normal keystrokes sent to the local video screen displaying an e-mail client, special multimedia keyboard keystrokes sent directly to the local CD player for audio control, and all keystrokes forwarded to a universal logging agent.

Sample services:

- Video display: using a video screen to display a running application. The video display service receives keystroke/cursor movement information and passes it to the appropriate client application. Video display services can work at the low "show these bits" level, or at higher levels such as the X window protocol or Microsoft's terminal server.
- Audio playback: sending an audio stream to a set of audio speakers to be mixed and played. The audio service may include different audio codecs for decoding the audio stream and may or may not have real-time guarantees of service.
- Location tracking: an object's physical location. An interesting challenge is preventing components from spoofing their location and claiming they are where they are not.
- Network-attached storage, print, fax, and personal video: devices already available in a form that should make integrating them into a SNAP environment straightforward.
- Human-interface translation for components that only present machine-readable interfaces: for example, a video screen may provide a "configuration" viewer that understands the underlying machine protocols.
- Capability augmentation proxies: for example, a dumb printer may not be able to print arbitrary documents. An application program running on a CPU may serve as an intelligent proxy that formats and queues the more complicated documents for the printer.

2.3 Location-Aware Relationships

Wireless components are not an improvement over wired components unless they intelligently set up relationships with other components in the vicinity, facilitating an automatic "mix and match" capability.

Furthermore, mobile components must be able to determine where they are in relation to other components. At any time a SNAP component has one of four relationships with a fixed reference location. Consider a home office: it may be physically present in the office (co-located), close to the home office, but not in the same room (proximate), remotely located but still accessible (remote), or inaccessible (offline).

There are two key aspects to location awareness: (1) finding other components in the vicinity and (2) establishing service relationships based on configuration settings, authorization group membership, and default behaviors.

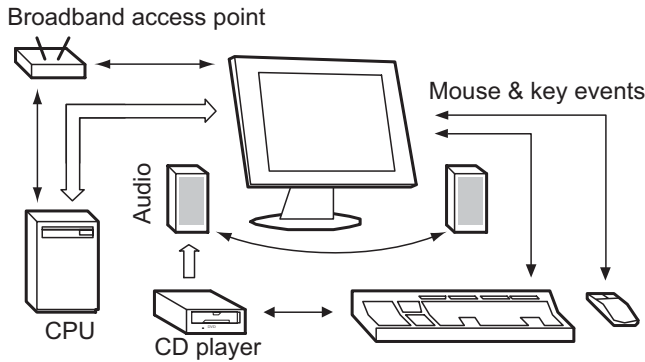


Figure 1: Home office components

Locating other components in the immediate vicinity is a technical challenge that may be addressed by using wireless signal strength combined with location services, a relationship that may be established along with the core wireless links. For example, an access point may also provide a default location service. Differentiating between co-located and proximate is trickier. A line-of-sight communication method, such as an IR beacon, may prove a reasonable higher-resolution location finder.

Once other devices in the vicinity have been located, a component will set up relationships based on desired configuration, prior history, available security authorization, and default behaviors.

Because of a potentially large quantity of configuration information, components may rely on a home configuration service to make many of the choices, and internally carry around only pointers to the configuration services and some information on default behavior. For example, a keyboard may have a default set of relationships set up for the home office. When the keyboard is carried into a public conference room, it will establish local communications, determine that it has no previously established relationships, and then request the local authoritative configuration service (the conference room itself) to suggest a default behavior.

2.4 Security, Integrity, and Privacy

SNAP components need to overcome the many potential security holes associated with any device communicate wirelessly using standard internet protocols. Hence all SNAP communications must be authenticated, authorized, encrypted, secured, and have guaranteed integrity. Moreover, there should be provisions that isolate attacks, so if one component is compromised, related components aren't automatically exposed.

An important part of the component configuration is a "network of trust," which includes persistent device relationships, proxy relationships, and device access rights. Persistent device relationships provide convenient default targets for things like audio and video output. Proxy relationships allow for redirection of specific classes of traffic, such as routing an undecoded video stream to a decoding proxy on a local CPU.

Access rights hinge upon the concept of ticket authorization by trusted location and identification services. Tickets will be issued taking into account connection method classification, a device's membership in a known category, or a specific user's ability to establish identity. For example, a keyboard may provide a

biometric identification service and in turn rely on a local wireless access point for establishing location.

3. SAMPLE SCENARIOS

In thinking about SNAP we've found it useful to set up sample usage scenarios. Here is a series of scenarios arranged in order of increasing complexity.

3.1 Home Office

Our first scenario is a home office, with a single user and a set of known components (figure 1).

3.1.1 Minimal set

We start with a minimal set of components: a video screen, keyboard, mouse, CD player, and audio speakers. Ignoring details of the underlying wireless infrastructure, we'll assume that all devices can communicate with each other and that each device is already configured for the local environment.

The mouse and most of the keyboard keys are bonded¹ to the video screen. The CD player is bonded with the audio speakers and is playing music. We bind a few multimedia keys on the keyboard to the CD player and speakers to control playback and volume. The audio speakers also pick up occasional beeps and other useful tones from the video screen.

What can the video screen do without a traditional computer? The video screen has an embedded operating system that provides a minimal web-browsing interface and a component locator service. This configuration tool locates SNAP components and displays their configuration/diagnostic interfaces.

The removal of any component will not affect the others or their interconnections. For example, remove the video screen. The CD player still works; the CD-bonded keys on the keyboard still control playback and volume level and the audio still plays.

3.1.2 Broadband access point

Putting a broadband access point in the home office changes the minimal set of components in the prior example from an expensive CD player into a network computer. The video screen's built-in web browser now has access to the Internet. The audio speakers now have the ability to play streamed audio directly by using the "play a live stream from a URL" functionality.

The broadband access point adds firewall and location services to the home office. The ability of the access point to disambiguate the whole of the web from the components in the local room provides benefits in both security and location awareness, in that a new component needs to prove its proximity before being granted much license to affect other components: you may not want someone in Taiwan re-configuring your mouse or reading your keystrokes.

3.1.3 CPU and disk drive

Adding a CPU module and a network-attached disk drive to the mix upgrades it from a network computer to a full-fledged home computer. The disk drive provides a persistent store and file system accessible by all devices. The CPU module runs a full operating system and gives the user the power to run more

¹ Two components are bonded if they maintain a dedicated link for sending information—it's a subscription to a service, not just the access of the service.

interesting applications, such as games, a full-featured web browser, or productivity tools.

Note that the CPU module may contain its own persistent storage; the equivalent of the `C:\WINNT` directory in a Microsoft OS. The CPU module may also expose part of its persistent storage as a network-attached storage device.

The CPU needs a high-bandwidth connection to the video screen to run games and other graphics-intensive applications. The keyboard and mouse are not necessarily bonded to the CPU; in this case the video screen serves as an intelligent proxy for the user's input. This proxy is necessary because not all of the keyboard or mouse input goes to the CPU. It may be directed to applets running on the video screen itself or to control the screen configuration of the video screen (the CPU doesn't automatically control the entire video screen).

The CPU may also serve to proxy some of the less capable components. For example, inexpensive audio speakers may benefit from allowing the CPU to handle streaming media, audio stream mixing, and echo cancellation if a microphone is added to the system.

3.1.4 Other equipment

Most other types of equipment added to the home office require little explanation. Printers, scanners, and fax machines show up as logical services that can be accessed either by sending data directly, or by sending references to data to be downloaded and processed. Tablets, trackballs, and other input devices bond logically to the video screen. Telephones, TV tuners, and internet radios participate in the local network as equal partners.

A second video screen is a more interesting problem. There are two ways to add a second video screen to the home office. The two screens may work completely independently or bond together to share user input devices (for example, when dragging a mouse cursor from one device to another). Components using the video services may tie to a specific screen or to the bonded screen service.

3.2 Component Configuration

In the prior examples we have ignored configuration and security issues. These arise in two contexts: when adding or removing a equipment from a static configuration (such as an office) and with inherently mobile equipment (such as handheld computers).

Locations awareness plays a critical role in configuration, as it enables the user to specify location-based behaviors and permissions. For example, when co-located in the office a laptop computer should play audio through the local sound system and use a local mouse. While traveling, a laptop computer may default to using proximate printers and fax machines, yet still use the home office's network-attached disk drive and watch a movie from the home video recorder.

3.2.1 Example: Configuring a new DVD+RW drive

Consider what happens when the user brings home a brand new DVD+RW drive. On power-up, the DVD unit announces its presence and searches for local registration services. A registration service specifies what policies are to be followed when adding new components and provides suitable notifications to other components and the user.

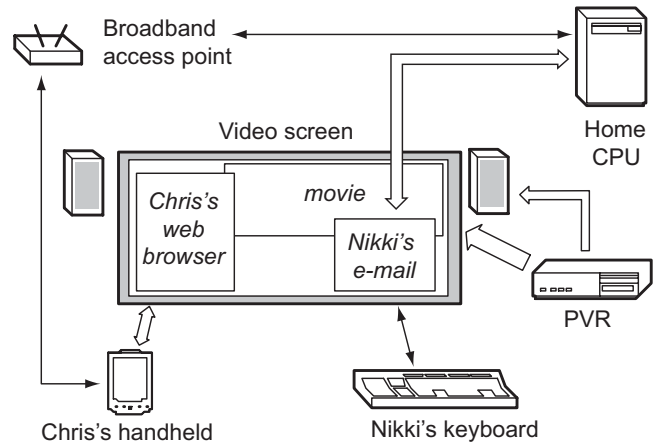


Figure 2: Living Room

In the presence of a registration service, adding the DVD is as easy as clicking a mouse button and requesting the new component be given appropriate group rights and authentications. Without registration services, the default behavior of the new unit can be set by accessing its configuration service (for example, by using the video screen's minimalist web interface).

Determining permissions to configure the DVD player will require some initial authentication of proximity and ownership. For example, the user may have to type in the serial number printed on the bottom of the unit, physically touch ("kiss") the unit to a one-wire interface, scan a barcode, or use an IrDA transceiver to authenticate proximity.

Once the initial authentication of ownership is made, configuration data and authentication tokens can be exchanged and placed in either in the DVD's non-volatile storage, or on the local registration server.

3.2.2 How does a laptop or handheld computer fit?

In the SNAP world, a laptop or handheld computer comprises a tightly integrated bundle of the services provided collectively by separate home office components, but the independent functionality of this bundle's individual pieces is preserved: a laptop's display screen can be shared by other components, its disk drive provides network-attached storage, and its keyboard can be directed to arbitrary components.

3.3 Multiple users

The final set of example scenarios considers what happens when multiple users share resources. For these scenarios we move out of the home office and into more public spaces.

3.3.1 Living room

In this example Nikki and her friend Chris are sitting in Nikki's living room watching television on a big, high-resolution video screen, but also doing a little work and web browsing (figure 2). The living room's personal video recorder (PVR) is playing a movie on the video screen and sending audio to the living room audio system. Nikki has pulled out a portable keyboard, connected to the home office CPU, and pulled up her e-mail on a corner of the living room video screen. As she browses her remote mail store, audio attachments are routed and mixed in the local audio system and played through the living room speakers so that they

appear on her side of the room (spatially located so as to not distract Chris).

Meanwhile, Chris has pulled out a handheld computer. Nikki has previously granted Chris some access rights for using the home's broadband connection and living room equipment, so Chris grabs a section of the big video screen and displays output from a web browser running on the handheld computer. Audio output from Chris's web browser is spatially located to help disambiguate it from Nikki's e-mail. Without a keyboard Chris must use the handheld computer for handwriting recognition and cursor control. To speed things up, Chris borrows the keyboard from Nikki's home office. The keyboard detects it is in the living room and bonds automatically to the big screen. Through the handheld computer, Chris assigns the keyboard to work with the web browser and goes back to surfing.

Most of the time Chris and Nikki are working within the confines of the big video screen. For example, both may be driving their own private pointing cursor on the screen. Security policies prevent them from controlling each others' applications; Nikki typing at her e-mail is kept separate from Chris's web browsing. However, the big screen also provides high level services that both can request and access. For example, a screen window manager service positions the individual windows and a screen cut-and-paste service allows data to be shared across users. Should Chris or Nikki wish to change channels or control audio volume in the room, either can ask for video screen control and use the shared, built-in video browser to access the audio volume control or bind it to their local device (Chris' handheld or Nikki's keyboard).

3.3.2 Conference room

Functionally, a corporate conference room is not greatly dissimilar from Nikki's living room. The conference room provides shared video screens that multiple users can access from their laptop/handheld computers, or via broadband connections back to their desktop machines.

The conference room provides several business-specific services. First, the room itself can provide scheduling and journaling functions. Because the conference room display screens are intelligent—rather than simple projectors—it is straightforward to allow them to record and store information about what was done in the room. Each user provides authentication before accessing services, so a clean record of users and activities can be journaled and made available to participants later.

Adding video conferencing introduces a second interesting feature: virtual proximity. A video conference establishes a virtual location relationship between people and devices. For example, the remote user may wish to print a file in the conference room, display and control a presentation on the video screen, and play audio through the local speakers.

4. TECHNICAL CHALLENGES

This section contains the most important and difficult research/engineering challenges to implementing a SNAP environment.

Wireless video screens

Will wireless bandwidth be adequate to support a large number of video screens in close proximity? A number of commercial products have demonstrated that 802.11 can support NTSC-quality video streaming, but the real challenge comes in a crowded environment with high bit rate video. Short-range ultra-wide band (UWB) radios have great promise for video.

Shared video screens

SNAP will require, at its core, video devices that can be shared at application, user, input device, and even working group levels, often simultaneously.

Locating components

We need reliable mechanisms to dynamically locate components both spatially and in relation to one another, at multiple levels of granularity (room, floor, office building). A variety of methods will likely be used to gather this information, including wireless signal strength and line-of-sight information (such as IrDA or IR beacons). Touching or user-assisted authentication (such as a bar code or serial number) may also help establish location.

Distributed relationship information

Connection, authentication information, and permissions will need to be stored and managed in a scalable distributed fashion, rather than in a common location. Real challenges lie in building tools that allow the user to specify the intent of permission sets, compiling those intents into specific permissions, and then distributing those permissions to the correct devices. Making user-friendly tools—most people have difficulty setting permissions correctly in the simple Unix security model, let alone the ACLs of Windows—looms as a major concern.

Providing secure, reliable communications

All communications between components need to be authorized, authenticated, approved, safe, and secure. This needs to be designed into the system, not tacked on afterwards.

5. CONCLUSIONS

SNAP promises great improvements to the existing model of computing through the simplification of user-device interaction. The technical challenges of creating a SNAP environment are non-trivial, but appear to be tractable.

Our long-term project goal is to build an interesting set of SNAP components, develop the underlying architecture and protocols, and test them in a shared environment.