Endosymbiotic Computing: Enabling Surrogate GUI and Cyber-Physical Connectivity

Pai H. Chou
University of California, Irvine, USA, and
National Tsing Hua University, Hsinchu, Taiwan
phchou@uci.edu

ABSTRACT

Endosymbiotic Computing entails attaching an RF-enabled microcontroller module (endomodule) to an appliance such that it appears as a networked device in the cyber world. It enables a smart phone to work as not only a universal remote control but also a surrogate GUI for inspecting the attributes of these appliances, without modifications to legacy circuits. To minimize the cost and resource requirements of the endomodules, we propose a generalized active message programming method that executes dynamically-loaded threaded code on-demand without requiring parsing.

Categories and Subject Descriptors

J.7 [**Computers in Other Systems**]: Consumer products; C.3 [**Special-Purpose and Application-Based Systems**]: Real-time and embedded systems

General Terms

Design

Keywords

Endosymbiotic computing, wireless interface, surrogate GUI

1. INTRODUCTION

Nearly 30 years ago, graphical user interface (GUI) helped users make sense of computing by borrowing metaphors from real-life objects. Thirty years later, GUI operations such as cut-and-paste and drag-and-drop are as natural to modern users as any real-life ones. Unfortunately, many every-day appliances such as microwave ovens, TVs, air conditioners (A/C), and telephone/answering machines (TAM) still rely on simple buttons, LEDs, and remote controls as their primary user interface. Although PCs and cell phones may be programmed as universal remote controls [1], they merely turn hardware buttons into software ones. One trend with networked devices is *surrogate GUI*, where devices such as printers and routers contain embedded web servers that provide the GUI as HTML. The Surface Computer [2] provides a GUI for devices in proximity, but it assumes large displays with no mobility. We propose to generalize surrogate GUI with low-cost, short-range wireless technologies available as integrated, RF-enabled, 8-bit MCU (microcontroller unit) system-on-chips (SoCs).

DAC'09, July 26-31, 2009, San Francisco, California, USA

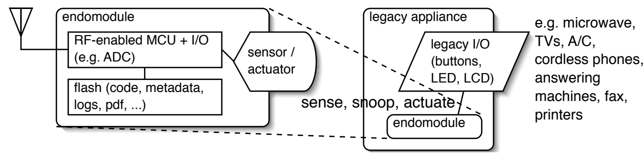

Figure 1: Endosymbiotic module.

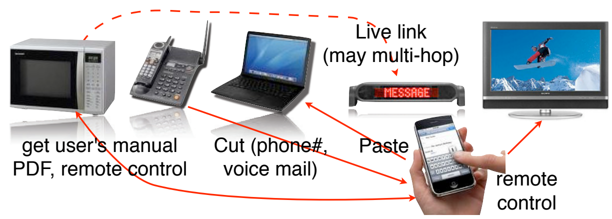

Figure 2: Wireless data grabbing and live linking.

Our vision is called *Endosymbiotic Computing*, where *endo-* is a Greek prefix for "internal, within," and *symbiosis* means a mutually beneficial relationship. An *endomodule* containing an RF-enabled MCU is to be embedded inside an appliance (Fig. 1), draws power from it and controls it programmatically.

2. ILLUSTRATIVE EXAMPLES

Scenario 1: Smart Phone as Universal GUI

A compatible phone discovers a list of endosymbiotically-enabled appliances in proximity, including a microwave, an A/C, a TV, a fax, a TAM, and a marquee. The user taps on the menu for the microwave, whose endomodule replies to offer more choices, including the serial number, model number, and a script for a software remote control to be run on the smart phone. Moreover, the endomodule keeps a usage log for the user to track energy usage for the past year. This particular endomodule contains flash memory with plenty of room to provide a PDF file for the user's manual, a file for the instruction video, and voice files in several languages to help the visually-impaired, all without having to decode these files.

Scenario 2: Inter-Device Cut-and-Paste

The user now inspects the TAM and wants to copy the caller-ID from this TAM to her computer. The smart phone would act as not only a remote control but more generally a *data grabber*. The endomodule may snoop the pins of the LCD to reconstruct the content

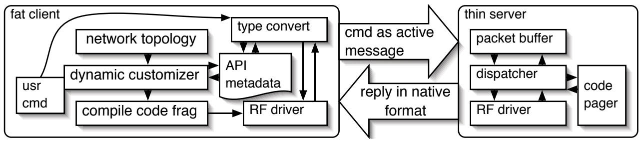

Figure 3: Software architecture with fat client and thin server.

being displayed on the LCD. The data grabber can first wirelessly grab the text from the endomodule, connect to a PC, and paste the text possibly by a gesture. The voice message can also be digitized by the endomodule's built-in ADC and grabbed similarly.

Scenario 3: Inter-Appliance Live Linking

The user discovers that she can not only grab a snapshot of the data but also establish a *live link* from the source to the destination. She uses her cell phone to grab the TAM's caller-ID as a virtual "outlet," walks outside the RF range of the TAM, and virtually connects it to the "inlet" on the marquee. Even though the TAM cannot connect directly to the marquee, a multi-hop route is discovered automatically to realize this end-to-end connection. Moreover, additional agent processes are synthesized and injected into the network to actively pull and push data on behalf of the two passive endpoints.

3. MECHANISMS

Hardware

The endomodule hardware must be simple and low cost such that appliance manufacturers will be willing to incorporate them and add sufficient value at a very small cost. In the extreme case, the endomodule may be as simple as a passive RFID, and the smart phone would look up all attributes from the Internet or a database. RFID may certainly co-exist with active RF devices, though here we focus the discussion on active RF interfaces.

Short-range wireless protocols include Bluetooth[3], Wibree[4], ZigBee[5], Z-Wave[6], and others. Bluetooth is popular for computer peripherals and cell phones. ZigBee is positioned for wireless sensor networks, though it is common to use just the 802.15.4 MAC layer and replace the ZigBee stack with a simpler networking layer. Z-Wave supports home control and is simpler than ZigBee but at a lower data rate. Wibree, or Bluetooth Low Energy Technology, will be a subset of Bluetooth and requires only a few KB of memory, compared to 30-100 KB with others. Wibree, 802.15.4, and Z-Wave are available in integrated 8051 MCU+RF chips, making them cheaper (around US$1 in large quantities) and simpler to implement. Power consumption is less of an issue here than cost and complexity. We believe that, similar to the Internet, the physical and MAC layers do not matter so much, as they can be easily integrated by wireless routers that can bridge these standards. What is important is how to support the execution of software on highly resource-constrained hardware.

Software Architecture

Endosoftware, namely that running on the endomodule to be attached to the appliance, should be kept as simple as possible. Java's hardware demand may be too high [7], as we assume something closer to an 8-bit, 8051-class MCU. Our approach is to move most of the complexity to the *fat clients*, a collective term for general-purpose computers, smart phones, gateways, and routers. Each endomodule works as a *thin server* that passively responds to commands. The software architecture is shown in Fig. 3.

To eliminate the parsing overhead, we borrow the idea from *active messages* [8], where a packet contains the address of the handling routine (in the receiver's address space) followed by the parameters. This way, the receiving endomodule just needs to jump to the address stored at the beginning of the packet without parsing. Active messages can be viewed as a special form of *threaded code* [9], a classical technique for efficiently implementing interpreters by representing virtual instructions as the addresses of the corresponding routines. Such a "command interpreter" does not understand the meaning of the command, as it obliviously jumps to the target address. With this active message layer, an endomodule can bootstrap wirelessly and execute any function that it loads. Security may be a concern, and our solution is to exploit hardware-supported encryption and decryption engines that are found as part of most transceivers of this class.

For "standard I/O" with minimal overhead, an endomodule would read and write data in the most convenient, native format. For instance, for date and time, general-purpose computers commonly use the "seconds since the epoch" representation, but it is not convenient for most 8-bit MCUs. The ordering, size, and endian of the day, month, year, hour, minute, and second are described as metadata in a form such as XML. The fat client is responsible for looking up the metadata associated with each command and formatting the data in the packets accordingly. This way, the replying endomodule does not have to know anything about the data types on the fat client side. More generally, to support the live linking in Scenario 3 above, the fat client would actively poll the source, perform type conversion, and actively push the data to the sink, or delegate any of these tasks to a thin server by dynamic compilation.

We propose to optimize the code for endomodules by dynamic compilation on the host side and wireless code update. The fat client may synthesize and dynamically compile a customized version of a function by exploiting run-time constants. For instance, the fat client may use its knowledge about the network topology to create a specialized version of the multi-hop routing function for each thin client with its neighbors' IDs hardwired.

4. CONCLUSIONS

Users today spend possibly more time in the cyber world than in the physical world. To truly bridge the gap, we propose the idea of Endosymbiotic Computing. It enables the user to (1) grab snapshots of data from appliances that previously had no connectivity to general-purpose computers, (2) query and control these appliances including accessing their manuals stored on the endomodules, and (3) customize the way these appliances should work with each other by inferring the necessary conversions and helper processes to inject into the network. It does not just dumb down the interface for the lowest common denominator users; it enables them to perform powerful distributed programming at their fingertips.

5. REFERENCES

[1] Brian X. Chen. Update turns iPhone into an Apple TV remote. `http://blog.wired.com/gadgets/2008/07/update-turns-ip.html`, 2008.

[2] Microsoft Surface. `http://www.microsoft.com/surface/`.

[3] The official Bluetooth® technology info site. `http://www.bluetooth.com/`.

[4] Bluetooth low energy technology. `http://www.bluetooth.com/Bluetooth/Products/low_energy.htm`.

[5] ZigBee Alliance – Home Page. `http://www.zigbee.org/`.

[6] Z-Wave. `http://www.z-wave.com/`.

[7] CASCADAS Consortium. CASCADAS Project. `http://www.cascadas-project.org/`, 2009.

[8] D.E. Culler, J. Hill, P. Buonadonna, R. Szewczyk, and A. Woo. Active message communication for tiny networked sensors. In *EMSOFT*, 2001.

[9] James R. Bell. Threaded code. *Communications of the ACM*, 16(6):370–372, 1973.