GRANNIE: A SCALABLE, INTERACTIVE, ARTIFICIAL INTELLIGENCE SUPERVISORY SYSTEM FOR MEDICAL DEVICES

Paul Frenger MD
A Working Hypothesis, Inc., P.O. Box 820506, Houston, TX 77282-0506 USA
pfrenger@alumni.rice.edu

ABSTRACT

Complex medical devices are difficult for patients to use. The author suggests putting a biologically-inspired, intelligent, modular, open-systems robot control network into these devices for improved patient interaction and supervision. The author's system, GRANNIE (an acronym for "Generalized Robotic Accessory with Neural Network, Intellect and Emotions") emulates human sensory, motor, cognitive and emotional responses. GRANNIE has a processor-neutral hardware bus and software which is independent of CPU or programming language. Improved plug-and-play functions and transducer interoperability were recently added. Scalability for various hardware configurations is provided. GRANNIE technology will help to create user-friendly, safety-promoting interactive supervisors for many varieties of medical devices.

KEYWORDS

IEEE P996.1, IEEE 1275, IEEE 1451.4, Artificial Intelligence, Computer Network, Medical Device, Motor System, Sensory System, Robotics

INTRODUCTION

Devices for diagnostic, assistive, rehabilitative and eldercare services are increasingly complex and difficult for patients to use without extensive training and human supervision. For example, there are many brands and models of diabetic glucose monitors on the market, each having unique human interfaces, calibration methods, blood collection requirements, data storage and downloading techniques, and report printing procedures. To alleviate this problem, I suggest creation of an attached or embedded, intelligent, modular, open-systems, multiprocessor robotic control network on-or-in medical devices (as appropriate) to improve patient interaction and supervision. These "smart" systems would replace a medical device's original complex user interface with a more intuitive, patient-friendly one.

Some inspiration for this project came from "terminal emulation" methodology, typically where decades-old financial applications running on obsolete mainframe computers are controlled by graphical-windowed PCs which interface these programs in a more user-friendly fashion [1].

Encouragement also came from successful adaptations of my robot control system (RCS) "ANNIE" (an acronym for Android with Neural Network, Intellect and Emotions) [2] into artificial intelligence (AI) areas: emulating human emotions, cognition, brain growth and development [3], hormone functions and response to medications. My ANNIE AI design appeared to be flexible enough to be developed into a customizable, intelligent, anthropomorphic interface for medical devices and equipment.

MATERIALS AND METHODS

1. <u>The Basic ANNIE Design</u>:

My ANNIE RCS design includes biologically-inspired modalities: head and torso control, facial gesturing, a dexterous hand, speech recognition / synthesis, artificial vision with optical character recognition (OCR), sensation of pressure, humidity, temperature and proprioception; and several which are non-biological: navigation (magnetic, inertial and GPS), ranging (ultrasonic and laser), and communication (infrared / radio / cell phone). She does not walk but is wheelchair-mobile.

ANNIE utilizes the IEEE P996.1 (PC/104) processor-neutral hardware bus [4]. PC/104 cards are slightly larger than 3.5-inch diskettes and can be connector-stacked on 0.6-inch centers to add functions. ANNIE has PC/104 prototyping cards stacked on her CPU card to provide hardware co-processors for artificial neural network (ANN), OCR, emotion emulation, and content-addressable memory (CAM) text-indexing accelerators. The head and extremities are controlled by networked regional processors. All hardware coprocessors have software-only versions available.

ANNIE's command software and AI functions employ a machine-independent, programming language-independent byte-coded script language based on an extension of the IEEE 1275-1994 (Open Firmware) standard [5], which contains a Forth language compiler [6], but which may be implemented in ANSI C [7] or other computer languages. Open Firmware is employed by IBM, Apple Macintosh, and Sun Microsystems (under the trade name OpenBoot) for computer booting and plug-and-play initialization. A free portable version is available [8]. Normally after bootup Open Firmware loads another operating system over itself, but it can be used stand-alone [9].

ANNIE's data files contain a knowledge base (64,000 words per dictionary in multiple languages such as English and French), her personality parameters, and her experience history. Also stored there is the AI schema-script system (a schema is a memorized data structure which has features of a digital neuron for decision-making, and a rule-based expert system for action control) [10]. ANNIE's schema scripts are generally used in "subject-verb-object" format.

I recently improved this design by adding a novel IEEE 1451.4 subsystem with enhanced plug-and-play and transducer interoperability to the base IEEE 1275 RCS [11].

2. <u>GRANNIE Adds Features Onto ANNIE</u>:

Extending ANNIE's functions, I have designed a medical device interface proof-of-concept system I call GRANNIE (which stands for Generalized Robotic Accessory with Neural Network, Intellect and Emotions). GRANNIE technology may be employed in two ways: it may be placed in a separate chassis for interfacing with existing medical devices, or it can be embedded within future enhanced medical devices.

3. <u>The Proof-of-Concept Experiments</u>:

The initial GRANNIE proof-of-concept test involved simulating an interface between a blood glucose meter and the GRANNIE system. The emulated glucometer is the Roche Diagnostics ACCU-CHEK® *Advantage*® blood glucose meter. Instructions for using this unit and its specifications are given in the owner's booklet [12]. This device has three buttons, a multipurpose LCD display and a tone generator as the principal user interface. A serial TTL UART jack is provided for data transfer to a personal computer, as is a slot for test-strip-specific electronic code keys. The glucometer stores up to 480 test results (time and date stamped) for downloading to a PC.

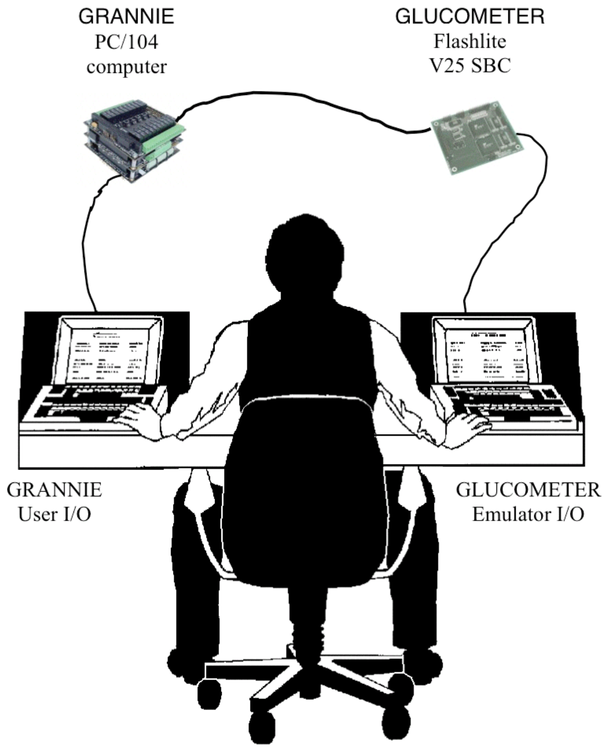

Figure 1. GRANNIE – Glucometer Test Setup

The hardware used in this test included a 500 MHz PC/104 x86-type computer, to perform the

GRANNIE functions. The glucometer emulator was embodied in a "Flashlite" 4.2 by 3.6 inch single board computer (SBC) containing an 8 MHz NEC V25 (8086-like) controller [13]. The SBC has two flash ROM disk drives and an MS/DOS 3.3 file system, a TTY serial port (for programming and control), a secondary serial data port and A/D input channels. The glucometer-emulator SBC conversed with GRANNIE over the SBC's second serial data port. As depicted in Figure 1, one PC was attached to GRANNIE's TTY port, and another was attached to the SBC TTY port, for observation and control of each system.

A Forth-83 interpreter (called "Flash-Forth") was installed on the SBC, in which the glucometer simulator program was written. GRANNIE's glucometer control and reading functions were created as schema scripts specifically written to support this particular SBC emulator.

To simplify testing, only the following glucometer functions were included: run a level 1 control test; take a blood glucose reading; download meter memory to computer; and clear memory. To further simplify tests, GRANNIE's visual, OCR, auditory, speech recognition / synthesis and dexterous hand control systems were disabled; instead, direct UART-to-UART data transfers emulated these functions.

Different simulated blood glucose values were produced by turning a potentiometer connected to the SBC's A/D input. Time and date values were obtained from the SBC. Test readings were downloaded as ASCII comma-separated-values (CSV).

RESULTS

Multiple tests were run using the above experimental setup, conducted as follows.

A communications check verified that the SBC glucometer and GRANNIE were talking to each other and to their respective PCs.

I asked GRANNIE to request an initial level 1 control test from the SBC. The SBC reported "55 mg/dl" (the strength of the actual glucometer testing solution) which GRANNIE displayed on the PC attached to her TTY port.

I then asked GRANNIE to request several blood glucose tests from the SBC. Values between "0 mg/dl" and "255 mg/dl" (depending on the potentiometer setting on the SBC's A/D port) were displayed on GRANNIE's PC display.

GRANNIE was next asked to request a download of the meter values in the SBC's memory; these were also displayed on GRANNIE's PC.

Following this, GRANNIE was asked to tell the SBC to clear its meter memory.

These test runs simulated the operations an actual glucometer user would routinely perform. All were accomplished successfully, albeit with some bug-eliminating adjustments to the SBC's Forth program and GRANNIE's schema scripts.

DISCUSSION

These first tests of GRANNIE's capabilities may not seem very challenging (except for its natural language AI input processing), but they suggest her future potential. Subsequent testing will be conducted with GRANNIE's visual, OCR, auditory, speech recognition / synthesis and dexterous hand control systems re-enabled. This will allow users to verbally ask GRANNIE to perform the various functions on an actual glucometer and hear GRANNIE recite the results. GRANNIE will be able to read the glucometer's LCD, hear its buzzer and press its buttons to navigate the control menus in place of the user.

An important future issue is providing scalability for GRANNIE over a variety of hardware configurations to accommodate 8 / 16 / 32 / 64-bit CPUs with widely divergent memory capacities. Scalability is the obverse of code transportability. Currently, use of IEEE 1275 byte-code processing software virtual machines (VM) created in the Forth and C languages makes this possible. Brad Eckert's version of IEEE 1275 written in C runs on 8-bit and 32-bit CPUs, requiring just 14 Kbytes combined ROM-RAM on the former and 24 Kbytes ROM-RAM on the latter [14]. Using a FIG-Forth software VM byte-code processor would allow use on any 8-bit to 64-bit CPU with a minimum of conversion effort [15].

Adding a Java byte-code processing capability along with IEEE 1275 byte-codes would enable GRANNIE to use existing libraries of Java and Linux-based code. Hardware Java processors exist, such as the 200 MIPS ATMEL ARM926EJ-S [16], which can process 32-bit ARM instructions, 16-bit Thumb instructions and 8-bit Java Jazelle instructions. This CPU also has a memory management unit, DMA controllers, and a graphics drawing toolkit. It has interfaces for cameras, AC97 audio, 10/100 Ethernet MAC, SPI, USB, CAN, CompactFlash cards, and other devices.

ARM's Jazelle DBX (Direct Bytecode eXecution) technology allows accelerated execution of Java programs [17]. The multitasking Jazelle IP core could be added onto FPGAs alongside a Forth byte-code CPU for IEEE 1275 processing [18], to produce an outstanding GRANNIE dual-core CPU. ANNIE already has provisions for dealing with mixed embedded executable and data byte-code streams [19].

SavaJe, employed in cellular phones, is a Java-compliant operating system [20] which could be used with GRANNIE technology in an external interface device, a smart phone, or in embedded medical devices.

GRANNIE's AI interface has to be customized for each new product, but only those components essential for a given application have to be installed initially; a "smart" auto-resize utility, remote installation and online upgrade utility can be added later. The development of tools to automate embedding GRANNIE's functionality into medical devices should logically come next.

CONCLUSION

GRANNIE AI-robotic technology can help to create user-friendly, safety-promoting interactive supervisors for many varieties of computerized health care-related devices and equipment.

ACKNOWLEDGMENTS

ACCU-CHEK® Advantage® is a trademark of Roche Diagnostics Corporation, Indianapolis IN.

REFERENCES

1. http://en.wikipedia.org/wiki/Terminal_emulator.
2. Frenger, Paul, "Meet ANNIE: an Open Systems Android Robot", *ANNIE 2000 Conf.,* 2000, 33-38.
3. Frenger, Paul, "Forth and AI Revisited: BRAIN-FORTH", *ACM Sigplan Notices,* **39** (12), Dec. 2004, 11-16.
4. http://www.pc104.org.
5. http://www.openfirmware.org.
6. http://www.firmworks.com.
7. http://www.codegen.com.
8. http://www.openbios.info/index.php.
9. Frenger, Paul, "OpenBoot: Java's Overlooked Sibling", unpublished paper available from author.
10. Drescher, G., *Made-Up Minds*, MIT Press, Cambridge MA, 1991.
11. Frenger, Paul, "IEEE 1451 with IEEE 1275: Synergy for Sensing and Control", *Proc IEEE Region 5 Tech, Prof and Student Conf,* 2006, 318-323.
12. Publication RDADV0_MNL1.0_121.pdf. http://www.accu-chek.com.
13. Frenger, Paul, "Using MS/DOS-Based Single Board Computers for Embedded Medical Systems", *Proc Southern Biomed Conf,* 1998, 2.
14. Eckert, Brad, "Firmware Factory", *ACM Sigplan Notices,* **34** (12), Dec. 1999, 30-33.
15. http://theforthsource.com/catalog.html.
16. http://www.atmel.com.
17. http://www.arm.com/products/esd/jazelle_home.html.
18. Thomas, Reuben, "An introduction to the Beetle Forth virtual processor", *ACM Sigplan Notices,* **32** (2), Feb. 1997, 22-25.
19. Frenger, Paul, "A Reduced Ambiguity Lexical System for an Artificial Cortex", *Biomed Sci Instrum,* **40**, 2004, 424-428.
20. http://www.savaje.com.