Intent Driven Interfaces to Ubiquitous Computers

Neil G. Scott and Martha E. Crosby

University of Hawaii at Manoa

Abstract. An intent driven interface allows a person to control a computer by stating an intended outcome rather than entering the sequence of tasks required to achieve the same outcome. Techniques that were originally developed as part of a universal access accelerator for individuals with disabilities are now being applied as convenience and productivity tools for accessing any computer based device, appliance or system. An intelligent universal serial bus (USB) Hub, called the iTASK Module, determines user intent independently of the input source or the system that is accessed. iTASK Modules can be interconnected to support multi-user collaboration and sharing of system resources without requiring any hardware or software changes to the accessed system.

Keywords: IDEAL, iTASK Module, NIP, intent, ubiquitous, USB.

1 Introduction

For many people, the term ubiquitous computers conjures up visions of an environment in which there is always some type of personal computer within easy reach. While there are now several billion personal and business computers in use around the world, they are just a small fraction of the total number of computers that have been deployed. Appliances, toys, cell phones, industrial controllers and cars are examples of devices that contain one or more embedded computers. An economy car, contains twenty to forty processors, and a luxury car contains eighty to one-hundred.

Business and personal computers that almost always provide a screen, keyboard and mouse to support user interaction, this is rarely the case with embedded systems where low cost and small size makes conventional human interaction strategies impractical for most applications. When absolutely necessary, dedicated, function-specific buttons and knobs may be provided but most of the embedded systems have no directly accessible human/computer interface.

This paper describes an intent driven user interface that enables an operator to control any computer-based system by simply describing the intended outcome rather than the sequence of commands normally required to reach that outcome. It can be used equally well on conventional business and personal computers or on embedded systems. User input consists of text messages containing one or more words that describe the intent of the user. The interface maintains context information by keeping track of the location, device, application, and control function that was most recently accessed. This context enables the interface to recognize applicable synonyms, such as "tv," "television," and "idiot box;" and to filter out words that don't have meaning

D.D. Schmorrow, L.M. Reeves (Eds.): Augmented Cognition, HCII 2007, LNAI 4565, pp. 241–250, 2007.

within the current context. This makes it unnecessary for the operator to adhere to predefined formats or scripts when formulating input messages. For example, "turn on the tv," "turn the tv on," "I want the television turned on," or "turn on the idiot box" all reduce down to "tv on." The operator doesn't need to know the mechanics of how the intended outcome is achieved, in this case, how the tv is actually turned on.

The intent driven interface is implemented as a small standalone device called an iTASK Module that uses universal serial bus (USB) ports for all input and output connections. While similar in appearance to the USB hubs that are used to increase the number of USB host ports on a personal computer, the iTASK Module contains its own processor and an additional USB client port.

The iTASK Module operates independently of the device to which it is connected. It checks every incoming message to determine whether to pass it directly to another port, as in a standard USB hub, or whether to analysis the message to determine user intent and then send out the sequence of commands necessary to achieve the user's intended outcome.

2 Background

The concept of an intent driven interface grew out of research into how individuals with disabilities could use speech recognition and head tracking to simplify and improve their interaction with computers and environmental control systems.

Speech recognition became a practical access tool for individuals with physical disabilities in 1989. The speech recognition software stretched the early PCs to the limits of their capabilities, particularly when used in conjunction with large applications programs, such as MS Word. Speech recognition impacted the computer system in at least three areas: (i) overall performance of the system was often significantly degraded, (ii) the combined user interface was often overwhelming for the user, particularly when he or she had to correct recognition errors, and (iii) software bugs in the applications and operating system were more prevalent when the computer was operating at its upper limit.

Individuals with disabilities who worked with non-IBM personal computers, such as the Macintosh, Sun and SGI were unable to reap the benefits of speech recognition which was only available for the IBM PC.

Under a grant funded by the Fund for Improved Post Secondary Education (FIPSE), one of the authors (Scott) developed a Universal Access System that made it possible to use the speech recognition with any type of computer. It achieved this by splitting the user interface into two separate components: A speech recognizer running in its own dedicated IBM PC, and a Universal Access Port that emulated the keyboard and mouse on whatever type of PC the user was required to work with. The Universal Access System proved to be extremely effective for at least three reasons: the speech recognizer always runs at full speed in a dedicated computer, it connects to any computer that has a Universal Access Port (Universal Access Ports were developed for IBM, Macintosh, and Sun, SGI, and HP workstations) and separating the speech interface from the application reduced the cognitive load on the user.

Conceptually, dictating text into a computer is very simple; you speak, it types. And it really is this simple until the recognizer misrecognizes what you said. Once

again, it is conceptually simple to say what is required to make the necessary corrections. It is just a matter of remembering the appropriate commands and speaking them into the computer in the correct order. With practice, you become proficient at making corrections but then, you need to be able to edit what you have written and save your work to a file. Yet again, it is conceptually easy to use predefined spoken commands to access the necessary menu items in your word processor. It is at this point, however, that many people become overwhelmed by the complexity of the system. Remembering which window should have the focus, the correct word or phrase to say, the right time to say it, and the order in which to say each word or phrase rapidly becomes a significant cognitive load.

Prewritten verbal commands, called macros, are used to reduce the number of utterances required to perform a particular task. A macro consists of a short phrase that describes a task linked to the sequence of commands required to perform it. The task is performed automatically each time the phrase is spoken. Properly designed macros significantly increase the productivity of speech recognition in real world working environments but they also introduce significant problems. The first problem is that the user must remember precisely what to say to trigger each macro. This gets increasingly difficult as the number of macros on a system grows. The second problem is part of a good news bad news situation in that the more comprehensive the macros, the more work that can be achieved with each utterance but accidentally triggering one of these macros at the wrong time can be devastating.

When speech recognition first became a viable option for people with disabilities in 1989, it took about two days of intensive training to enable a disabled person to perform non-trivial work tasks. A decade later, when computers were more than fifty times faster and the recognition software was much more advanced, it still took the same amount of time to train a new user. After questioning many disabled users of speech recognition, the author (Scott) concluded that the problem was not caused by the technology but rather, by user uncertainty about what to say to the computer in different scenarios. This uncertainty was exacerbated by a fear of accidentally triggering a macro that would damage or lose what had already been written. Almost without exception, the first utterance a person made when first introduced to speech recognition was “what do I say?”

3 Developing an Intent Driven Interface

The Archimedes Project at Stanford University undertook a project during the summer of 2002 to design a new Universal Access System that would eliminate the “what do I say?” problem. A team of twenty researchers and students worked together to develop a strategy for solving the problem and them broke into small groups to design the hardware and software components. Initial brainstorming sessions led to the development of a scenario in which a user describes the intended outcome to a virtual operator who knows the correct way to perform any required task on the device that is being accessed. In this scenario, users could use cryptic commands like “tv on” or they could be more tentative, using commands such as “I would like to watch the television please.” If the user says something that is confusing or unclear, the virtual operator should ask for clarification and remember the response to prevent

confusion should the user say the same thing again. A living room was chosen as the primary scenario for the test bed. The goal of the project was to enable a user to turn individual lamps on and off, open and close doors, windows and drapes, control all user functions in an audio visual system and control the operation of a computer.

3.1 The iTASK Module

The overall design of the iTASK, as it became called, is similar to the original Universal Access System except that that all input and output connections are now made through USB ports as shown in Figure 1, and a Natural Interaction Processor (NIP) is incorporated into the internal software to determine user intent based on one or more text messages received from accessors as depicted in Figure 2.

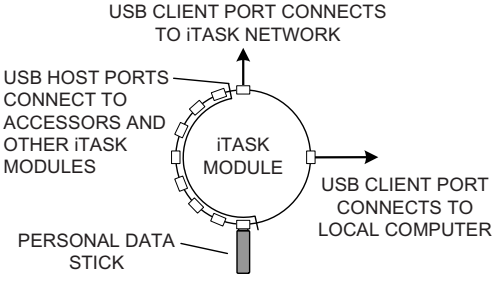

Fig. 1. USB Ports on an iTASK Module

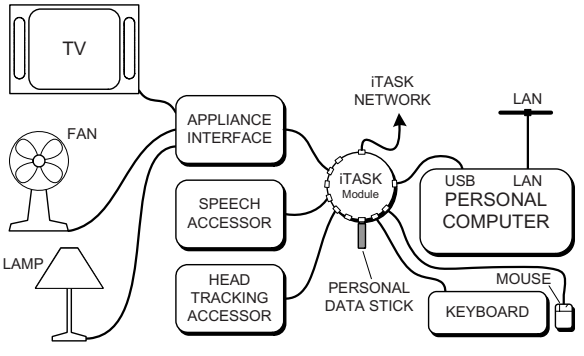

Fig. 2. iTASK module providing access to appliances and a personal computer using speech recognition and head tracking

Note the personal data stick connected to one of the USB host ports in Figure 1. This is a standard flash drive containing vocabulary trees, command definitions and user preferences that personalize the iTASK Module. In some situations, the personal data is stored in an accessor if it is always used by the particular user.

The keyboard and mouse depicted in Figure 2 are the original devices that were attached to the personal computer except that they now pass through the iTASK Module so that their data can be monitored to look for embedded text commands or gestures. The USB standards define devices like the keyboard and mouse as Human Interface Devices (HID) devices. All of the standard operating systems (Windows, Linux, Mac, and so on) incorporate software to automatically recognize and process HID data packets. In most situations, the iTASK Module automatically sends HID packets directly to the local personal computer regardless of the originating USB port. Therefore, sending an HID packet or plugging a USB keyboard or mouse into any port on the iTASK Module will automatically enter keyboard or mouse data into the local personal computer. HID data can also be sent to any computer on the iTASK network by wrapping it in an iTASK packet addressed to the iTASK Module that

connects to the target computer. When the iTASK packet is reaches its destination, the iTASK wrapper is removed and the HID packet is delivered to the local computer.

The speech accessor depicted in Figure 2 is a small computer running speech recognition software. It translates spoken messages into plain text that is placed in an iTASK packet and sent to the iTASK Module. The iTASK Module makes all of the decisions about whether the text is to be interpreted as a command or as dictated text, and the destination for the resulting information.

The head tracking accessor translates head movements into 2D direction vectors that mimic the operation of a mouse, 3D direction vectors that point to an object in space, movement dynamics that can be interpreted as gestures, or text messages that specify the name of the object currently being pointed to by the tracker. For example the text message might say "pointing at the tv" or "pointing at the bed lamp." Representing the output of the tracker as text allows the iTASK Module to interpret spoken or typed commands such as "turn that off" that are coincident with the head tracker identifying a particular lamp or appliance.

The Appliance interface depicted in Figure 2 translates generic device commands generated by the iTASK into specific commands and transmission formats expected by the targeted appliance. Much thought went into how the interface to appliances should be handled. The first version of the iTASK Module included infrared (IR) and radio frequency (rf) and X10 power line components that could directly control many different devices. It was later decided to place these components in separate appliance modules because they represent moving targets that are constantly changing as new standards and improved transmission protocols are introduced. For instance, there are more than one-hundred thousand different IR codes currently being used for audio visual equipment. The iTASK Module handles this by generating "iTASK" text commands, such as "volume up," that are translated into specific IR codes by a small natural interaction processor inside the appliance module. The IR codes that match the particular TV, for example, are down loaded from the manufacturer's website and stored in the appliance module when the TV is first installed and set up.

One of the most important characteristics of the iTASK Module is that the personal computer depicted in Figure 2 is absolutely standard and contains no added software or hardware. Applications function exactly as they would if the iTASK Module was not connected, i.e., there is no performance degradation. Furthermore, there is no requirement for the personal computer to be of a particular type; the only stipulation is that it has a USB port. This is particularly liberating for individuals with disabilities who have traditionally had to put up with slow and compromised computer systems containing additional layers of software on top of the operating system and applications.

The arrow at the top of Figure 2 identifies a USB client port that provided for connecting the iTASK Module to a hierarchical network as depicted in Figure 3. Each iTASK Module in the network knows about all devices and other iTASK Modules that are directly connected to it. A discovery mechanism built in to each iTASK Module enables any iTASK Module to automatically locate and interact with devices connected to any other iTASK Module in the network without requiring any stored master lists of available resources. This allows iTASK Modules to be added or removed on the fly without disrupting the network or leaving behind stale data.

3.2 An Example of an Intent Driven Application

The Intent Driven Environment for Active Learning (IDEAL) classroom depicted in Figure 4 shows how iTASK Modules can be configured as an iTASK network to support learning activities in a classroom. The apparent simplicity of the diagram belies the revolutionary features made possible by the iTASK Module.

- The iTASK Modules handle all of the conventional and special user interfaces for each computer. While each keyboard, for example, is normally connected to its local computer, the iTASK Module can easily redirect it to any other computer in the iTASK network. Typing or saying a simple command like "connect my keyboard to the class computer" or "connect my mouse and keyboard to Mary's computer" will automatically redirect the specified peripheral devices.
- Personal data sticks configure each iTASK Module to handle the special needs and personal preferences of individual students.
- The special access requirements of students with disabilities are easily and quickly accommodated by connecting one or more accessors to an iTASK Module, as depicted on the leftmost student computer in Figure 3. If this were a speech accessor, for instance, it could also be used to access any computer in the system simply by saying something like "connect this to the Mary's computer"
- Three different networks are shown in Figure 4. Each network handles a different aspect of the networking requirements.
    - The iTASK network handles all time-dependent communications between the iTASK Modules and the Class Resource Server.
    - A conventional cable or wireless local area network (LAN) delivers bulky material from the class resource server to the other computers.
    - A wide area network (WAN) connects the Class resource server to the school or district server and the internet. The class resource server manages and screens all individual student access to the internet based on lesson requirements and individual permissions.
- The natural interaction capabilities of the iTASK Modules make interaction with the computers more transparent thereby reducing the intimidation of students who are not technically savvy.
- A hands-on activity module is shown connected to each iTASK Module that connects to a student computer. The hands-on activity module operates in conjunction with a set of virtual instruments in the class resource server to incorporate real time measurements into lessons that are managed and delivered by a learning management system also included in the class resource server.
- Apart from the class resource server, every computer in the classroom has minimal performance requirements. Basically, they can be any computer that supports a LAN connection, a USB connection and a standard browser. Disc-less Linux computers are an ideal choice since they eliminate all licensing costs and require almost zero maintenance and technical support. It is not necessary for all of the computers in a class room to be of the same make, type or model, or be running the same operating system, since the only requirement is that they can run a standard browser such as Firefox. Being able to mix and match any

computers in a class set liberates schools from the high costs and major disruptions currently caused by built-in obsolescence. Every purchased or donated computer can be used in a classroom until the hardware actually fails.

- Fewer IT staff will be required to maintain the computers in a school.
- The long term stability of the classroom computer environment will encourage teachers to become more engaged in using computer-based learning because the materials they prepare will continue to run from one year to the next.

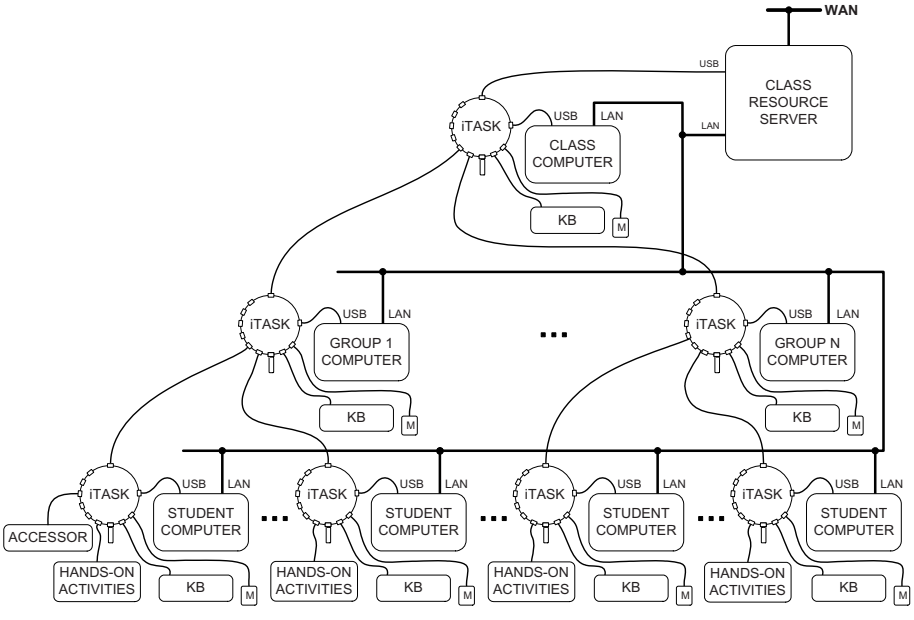

Fig. 3. Example of an iTASK Network supporting computer based learning in a classroom

4 Implementation of the iTASK Module

The proof of concept prototype constructed in 2002 combined conventional linguistics analysis with distributed intelligent agents. The resulting program occupied forty-six Mbytes of memory. A one Gigahertz computer with one Gigabyte of random access memory (RAM) correctly decoded natural language commands for controlling devices in a living room scenario but even with a simple command like "turn on the lamp," there would be a delay of up to five seconds before the lamp would turn on. While the results were functionally correct, the project was considered impractical due to the size and slowness of the software and the cost of the hardware.

Armed with the knowledge that the basic concepts worked, one of the authors (Scott) developed a second prototype based on control engineering algorithms instead of the statistical methods used by linguists. The key to using this more direct approach resulted from continually tracking the context of the interaction between the user and the iTASK Module. Whereas the linguistic approach considers every word in the

vocabulary each time a new test word is received, the control engineering approach rejects all words that have no meaning within the current context. For most practical commands, this results in a search space of less than ten words that can be easily handled by a small microprocessor. The second prototype of the iTASK Module was implemented on an eight bit microprocessor that had only four hundred bytes of random access memory (RAM). This system correctly interpreted commands for the living room scenario in less than ten milliseconds.

Two new prototypes are currently being designed. The first system uses a thirty-two bit ARM processor and a recently introduced USB processor chip that performs all of the USB processing in firmware. This system will support multiple, large vocabulary applications with almost imperceptible processing delays. The second system uses an extremely low power sixteen-bit processor. It will handle a medium size vocabulary and make it feasible to incorporate fully functional iTASK Modules into small devices such as infrared remote controls, electronic medical devices, electronic toys, lamp controls, door locks or control panels for smart environments.

4.1 Accessors

As mentioned earlier, an accessor is a portable input/output (I/O) device that provides a personalized bridge between a user and an iTASK Module. Originally developed for individuals with disabilities, accessors translate alternative user inputs such as speech, or Morse code into text messages that are processed by an iTASK Module. Outputs from the accessed system are sent from the iTASK Module to an accessor where they are translated into an alternative modality matched the capabilities of the user, e.g., spoken text or Braille for a blind person or graphically generated sign language for a deaf person.

Accessors and iTASK Modules are not restricted to disability-related scenarios. They provide an ideal way to provide on-demand access to embedded processors that normally function without any direct interaction with a human. Including a small iTASK Module in an embedded industrial control processor, for example, allows any accessor to be used to perform tests, make adjustments or load new software. In this situation, the iTASK Module eliminates the need to use a different proprietary remote terminal for each different controller since a technician can use a single accessor to interact with any type of embedded controller.

4.1.1 Input Accessors

Input accessors fall into two basic categories: Accessors that generate messages to be interpreted as text functions in an iTASK Module, and accessors that generate messages to be interpreted as pointing functions.

Text-related accessors include:

- Keyboards may have larger or smaller dimensions than standard keyboards, specialized keys, customized layouts, and different languages.
- Chordic keyboards that translate combinations of one to seven keys into keystrokes. These can be designed for left or right handed operation, and can support faster text entry than a conventional qwerty keyboard.

- Scanning inputs make it possible to generate full text messages with a single input switch. Statistically organized options are displayed sequentially to the user who operates the switch when the desired option is displayed. Options include: groups of letters, individual letters, icons, spoken words or phrases, musical tones or arbitrary sounds.
- Morse code inputs can be generated with one, two, or three switches, depending on the range of movement and timing/response capabilities of the user. Dash, dot and enter switches are suitable for people who have poor timing control or are subject to spasms. A single two-way bug switch can generate Morse code faster than anyone can type. Morse code characters have been defined for all characters available on a computer keyboard.
- Handwriting recognition on an HP iPAQ PDA is sufficiently fast and accurate for it to be used as a reliable handwriting accessor. Some tablet computers also work well but they are still expensive and clumsy to handle.
- Speech recognition is available with many different vocabulary sizes and processor requirements. The vocabulary of low-cost, chip-based systems ranges from ten words through to a few hundred words. PDA based systems have vocabularies of several thousand words and PC based systems have vocabularies ranging from ten to more that one-hundred thousand words.
- Direct selection accessors use pointing devices such as a mouse, finger, stylus, head trackers or eye gaze trackers to select objects displayed on some type of display screen. The objects can be letters, words, phrases, or icons that are translated to text when selected.

Pointing related accessors translate some form of pointing action into: 2D vectors, 3D vectors, movement dynamics, or messages that specify the object at which the person is pointing. Devices that detect pointing movements include:

- Various types of mouse.
- Tablets or touch pads operated by a finger or stylus
- Gyroscopes for rotational movement
- Accelerometers for linear movement or tilt sensing
- Head tracking using ultrasonics, infrared, gyroscopes, accelerometers, or video cameras
- Eye gaze tracking based on specialized video cameras
- Gestures based on any moving part of the body detected by: video cameras, ultra wideband radar, or electric-field-disturbance detectors,
- EMG, EKG or EOG signals detected by electrodes attached to the skull

4.1.2 Output Accessors

Output accessors were designed to translate information derived from the screen or speaker of the target computer into an alternative modality and/or format that matches the special needs of a person with a disability. They are also useful for non-disabled users who need to access embedded processors that are not normally equipped with a human/computer interface. An output accessor designed specifically to access embedded processors may simply duplicate the functions of a standard computer

screen on the screen of a PDA or cellular phone. Examples of output accessors that could be adapted for use with ubiquitous computers include:

- Devices with alternative screen properties optimized particular locations, size constraints, formats and application specific requirements.
- Screens with magnified screen images for visually impaired users.
- Text-to-speech synthesizers are used in screen readers for blind users or communication devices for people with severe speech disorders. Speech output has many uses beyond disability such as talking instruments and interactive controls in smart buildings,
- Text-to-sign language for deaf users. Signing avatars are now supported on PCs, PDAs, and Cellular phones.
- Text-to-flashing lights for telling deaf people to look at a screen message or to inform then that someone is ringing the door bell, etc.
- Braille displays for blind users.
- Graphics-to-haptic displays that represent lines, images, icons and text locations as forces that can be felt by a blind user.
- Graphics-to-sound (sonification) displays for blind users represent screen objects, colors and textures by modifying various sound properties.
- Vibration output devices (tactors) communicate directional information to people who are blind, deaf or performing tasks that heavily load other senses.

5 Summary

While originally designed to provide universal access for individuals with disabilities, the iTASK has the potential to improve and simplify human/computer interaction with any device or system that uses information and computer technology. iTASK Modules will be made available with support for small, medium and large vocabularies. Small vocabulary iTASK Modules will be suitable for embedding into low cost devices that currently use cryptic push button interfaces. Every iTASK Module is compatible with any type of input accessor. At this stage of development, most of the output accessors require special additions to a standard iTASK Module. Examples of specific applications are described in the paper but it is easy to see how these can be extrapolated into almost any area of human/computer interaction.