i*CATch: A Scalable, Plug-n-Play Wearable Computing Framework for Novices and Children

Grace Ngai, Stephen C.F. Chan, Vincent T.Y. Ng,
Joey C.Y. Cheung, Sam S.S. Choy, Winnie W.Y. Lau and Jason T.P. Tse
Department of Computing,
Hong Kong Polytechnic University, Hong Kong
{csgngai,csschan,cstyng,cscycheung,cssschoy,cswylau,cstptse}@comp.polyu.edu.hk

ABSTRACT
There has been much recent work in wearable computing that is directed at democratization of the field, to make it more accessible to the general public and more easily used by the hobbyist user. As the field becomes more diversified, there has also been a shift away from the highly specialized functionality of earlier applications towards aesthetics, creativity, design and self-expression, as well as a push towards using wearable computing as an outreach tool to broaden interest and exposure in engineering and computing.

This paper presents the design and development of the i*CATch wearable computing framework, which was developed specifically for children and novices to the field. The i*CATch framework is based upon a bus-based architecture, and is more scalable than the current alternatives. It consists of a set of plug-and-play components, a construction platform with a standardized interface, and an easy-to-use hybrid text-graphical integrated development environment. We will also present results of the evaluation of the i*CATch framework in real teaching environments.

Author Keywords
i*CATch, wearable computing, electronic textiles, e-textiles, computational textiles, smart textiles, construction kits.

ACM Classification Keywords
K.3.0 [Computers and Education]: General; H.5.2 [Information Interfaces and Presentation]: User Interfaces; C.3 [Special Purpose and Application-Based Systems]—microprocessor/ microcomputer applications, real-time and embedded systems.

General Terms
Human Factors

INTRODUCTION AND BACKGROUND

One of the perennial tradeoffs in the field of wearable computing is the balance between providing functionality and power versus expressivity and flexibility. Early wearable computing constructions focused on body monitoring, military or pervasive computing applications, and required a lot of processing power. An example is the Georgia Tech Wearable Motherboard [20], which was developed for military use and demonstrated the concept of personalized mobile information processing (PMIP) in the context of pervasive/invisible computing. Appearance-wise, these systems tended to be awkward and aesthetically unattractive, but incorporated a lot of computing power and functionality as the wearer was essentially carrying a portable computer around.

In recent years, a shift in the field has brought wearable computing closer to the realm of aesthetics, interaction and personal expression [2]. For example, Orth et al [17] introduced fabric computing interfaces that use sewn fabric sensors and circuits to eliminate uncomfortable and heavy wires, connectors and electronics. The e-Tags [13] and Electric Suspenders [9] work developed innovative methods for introducing reconfigurability and embedding electronic communications and power supply into textiles.

Many of the recent developments in the field of wearable computing have been aimed at "democratizing" the field by providing a low-cost, low-threshold entry point that allows enthusiasts to experiment with technology that was previously available only in laboratories. The Arduino Lilypad [3, 4, 5], for example, provides a set of low-cost, easy-to-use controllers and sensors with a form factor specifically for wearable computing and e-textiles. This was complemented by related work and experience sharing from various sources [3, 7, 16, 19], which moved wearable computing further into the realm of tangible interactivity and artistry. The TeeBoard [16] took that one step further by eliminating the need for formerly indisposable low-level skills such as sewing and ironing, and facilitated the use of wearable computing as an educational and outreach tool for engineering and computing.

Even though these advances have significantly lowered the threshold to wearable computing, significant challenges still exist for the non-experienced user. While the economic and accessibility thresholds have been lowered for wearable computing, the usability thresholds have stayed at the level of the skilled hobbyist. Despite all the progress, it is still difficult to imagine an everyday person on the street buying a wearable computing construction kit and creating his or her own construction with it. Even for the semi-skilled hobbyist or the student in a wearable computing workshop, it is still difficult to create wearable computing fashions with more than a few sensors and actuators, as current state-of-the-art microcontrollers do not easily support simultaneous connections to more than a handful of sensors and actuators without some serious hacking.

Another challenge lies in the lack of a common standard for interfacing electronic components to textiles at the consumer level. A variety of different methods exist: for example, the Arduino Lilypad components are sewn onto the cloth with conductive thread, while the Elektex fabric controls [8] uses a plastic socket-and-pin connector, and the TeeBoard uses metal snap fasteners. This diversity of interfacing choices makes it very difficult to interchange components and ultimately makes it hard for the field to gain traction with the general public, whether in the consumer or the educational market.

This paper presents the i*CATch framework, which combines a scalable, extensible construction platform with standardized interfacing for a set of open-source, expandable electronic modules. Our objective is to encourage creativity and self-expression in wearable computing, with our target users being children or novices to the field.

METHODOLOGY

We postulate that many of the current problems in general purpose, low-entry threshold wearable computing could be eliminated by a change in the communications architecture between the peripheral electronic modules, such as the sensors and actuators, and the microcontroller main board. By and large, current state-of-the-art practices in low-entry-threshold wearable computing rely on point-to-point connections, in which individual input and output pins on the microcontroller and the peripheral modules are connected directly to each other. Programming for a wearable computing construction in such an architecture is then a matter of sending a signal along the corresponding connection. For example, to turn on an LED light with one of its “legs” connected to the power supply and the other “leg” connected to Pin 23, one would need to send a LOW signal to Pin 23 for the duration that we wish the LED to stay on.

In our experience, creating wearable computing constructions from such point-to-point connections require care and foresight, as correcting mistakes is time-consuming and costly. For example, moving a device once it has been attached to the garment may involve changing the pins to which it is attached to avoid crossing connection lines, or lines that are too long and therefore contain too much internal resistance. Different modules also require varying numbers of input and output connections, which can also be intimidating to novice users without prior experience in electronics. Microcontrollers also have a limited number of input and output pins, and forward planning is required to ensure that the garment can accommodate all the desired modules. Finally, most people will first build the physical construction before they start programming it, which means that the programming will require tracing through the network of connections lines to find out which ports are connected to which sensors -- a process that is error-prone as the number of devices and the number of connection lines increase. This all hinders inexperienced novices from creating even marginally complex wearable computing constructions that involve more than one or two sensors and actuators.

We therefore propose a shift from the currently widely-used point-to-point communications architecture to a broadcast-based communications architecture. Broadcast-based architectures are widely used in microprocessor systems and on computer networks. In such a platform, all the devices sit on a common communication channel on which messages are broadcast. Since all devices will receive all messages, each device has a unique identifier address, and each message contains a header with information that identifies the sender and the recipient.

As a result of this added complexity, more sophisticated background support is required, and more overhead will be incurred. For example, some overhead will be involved in preventing message collisions, where multiple devices attempt to broadcast a message on the same channel at the same time. We believe, however, that broadcast-based architectures could simplify the construction of wearable computing constructions for the novice user:

- All the devices sit on a common communications channel, meaning that connecting a new module only involves attaching it to the communications channel either through pairing (in the case of wireless communication) or daisy-chaining (for wired communication). This simplifies the connection issue as there is no need to match input and output ports between the main controller and the peripheral device.
- Each device has its unique identifier address, which is independent of location and is constant. This makes programming for the construction easier as there is no need to trace connection lines to locate the pin that the device is connected to in order to control it.
- In general, point-to-point architectures are less scalable than broadcast-based architectures. Point-to-point architectures are necessarily limited by the number of input and output pins on the main board, which is directly

related to its size. Broadcast-based architectures can usually accommodate a much larger number of devices.

Design Objectives of the i*CATch framework

The i*CATch construction kit was designed for the two purposes of supporting creativity and facilitating standardization in wearable computing. In order to fulfill our objectives, it had to meet the following criteria:

- Given our target users, the system needed to be stable and reliable. Novice users tend to get alarmed easily and blame themselves for malfunctions in the system. We wanted our users to focus on the creation that they were designing, not on the system behind it.
- We did not want to assume any pre-knowledge or aptitude with low-level skills, such as sewing or ironing. This necessitated the use of a ready-to-go construction platform, with "plug-and-play" electronic modules that would be designed specifically to work with it.
- Since our target users are novices and children, this also meant that the system needed to be reasonably robust against user error and abuse. To a certain extent, it should even try to prevent typical errors from happening.
- For convenience and economic reasons, the electronic modules would have to be low-energy consuming and also cheap to manufacture. This ruled out the use of wireless personal area network architectures, which tend to be energy-hungry and expensive.
- Since one of our goals is to create a standard in wearable computing, we wish to facilitate other experts in the field to create modules that would work with the i*CATch kit. This requires the use of a widely-accepted, easily-acquired technology for the communications system and easily-obtained parts for the i*CATch interface.
- Another of our goals is to popularize wearable computing among the general public and the educational community. Therefore, the i*CATch framework also needs to be reliably manufacturable on a large scale and at low cost, in order to further reduce the entry barrier.

Communications Bus, Interface and Construction Platform

Under the i*CATch framework, we envision ready-made construction platforms that would essentially be garments or other accessories, with multiple "sockets" that would allow a user to "plug in" electronic modules to create their wearable computing construction.

To support this functionality, we chose to deploy the widely-accepted inter-integrated circuit (I2C) bus technology [10] as the communications channel between the electronic devices. The I2C technology transmits data with two connecting lines, the "clock" and the "signal" lines, and commands and messages are sent to and from the individual modules in the form of pulses.

It is worth noting that bus-based technologies have been previously used in wearable computing, especially for complex systems that perform monitoring and logging functions. As an example, the PadNET system [11] deployed a 3-wire bus as the communications backbone for a wearable physical activity detection network, the I2C bus technology has also been used in eTags [13], Gorlick's Electric Suspenders [9] and Nanda et al.'s bYOB system [15], among others. However, most of this work falls in the area of high-functionality, specialized applications, not in the user-customizable, creativity-oriented applications that we are interested in.

The use of I2C in previous work demonstrates that it is a well-tested, reliable technology, which satisfies one of our objectives. In addition, the choice of I2C also has the following advantages:

- As I2C uses only two wires for data communications, the i*CATch interface on the electronic modules will only require two data connection points. This helps to save on the physical space that the devices will take up, as compared with other physical broadcast-based technology such as Ethernet.
- The two-wire requirement also means that implementing the physical bus infrastructure will require only two data communication lines. This saves on the needed fabric and makes the bus less bulky and easier to integrate into a garment.
- As I2C has widespread use and acceptance in the community, it is also easy for other users to create their own modules that will work with the i*CATch kit. This is in contrast to the one-wire bus technology, which was also considered, but was deemed not suitable because of the lack of open-source modules.

The i*CATch connection interface is used to attach the electronic modules to the i*CATch bus. To maximize surface area for self-expression, we envisioned garments such as jackets embedded with a ready-made underlying bus infrastructure. The mechanism of the bus and the physical structure of the human body that would be wearing the garment suggested an underlying network with a linear or a partially-connected mesh topology, with "sockets" at strategic locations that would accommodate the attachment of a module. That means that the i*CATch interface has to satisfy the following requirements:

- To facilitate the development of a standard set of modules that will work with each other, the i*CATch interface should be created from cheap, easily-obtained parts that work equally well with textiles and electronics;
- Since the users of the i*CATch platform will likely be novices, the interface needs to be easily usable and intuitive;

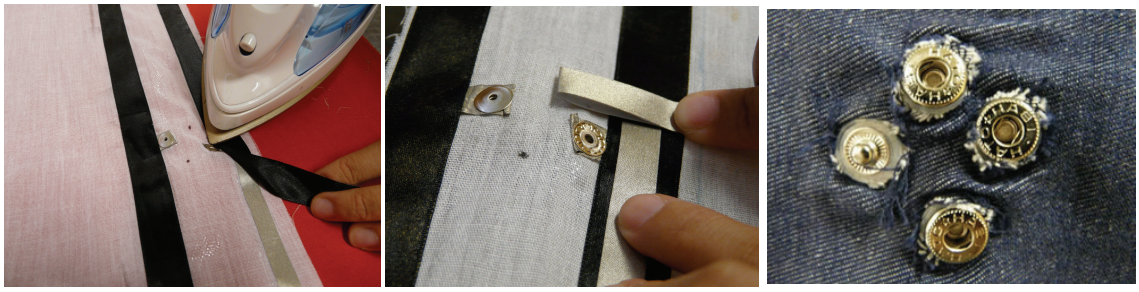


Figure 1. Making the i*CATch bus and interface. (Left) insulating two conductive strips from each other with close-weave polyester, (middle) folding the conductive strip to make a tab, two other tabs already have fasteners attached; (right) an i*CATch interface socket.

- For aesthetic reasons, the materials used in the i*CATch interface should be in keeping with the theme of textiles and intelligent, interactive garments.
- Since one of our goals is to popularize wearable computing, the i*CATch construction platform must be manufacturable on a medium to large scale, with a easily duplicated quality assurance process.

To satisfy these criteria, the i*CATch interface was created from 1-cm-diameter spring-loaded metal snap fasteners. These fasteners have the advantage of being very robust, as the in-built spring keeps the connection secure over multiple connect/disconnect cycles. They are easily attached to fabric using a button gun; and can also be easily soldered to an electronics PCB board.

The two-wire configuration of the I2C bus necessitated at least two connection points per socket. In addition, virtually all of the electronic devices needed to be connected to the power and the ground lines. The i*CATch interface sockets were therefore created from four metal snap fasteners to accommodate the power, ground, clock and data lines. The distances between the snap fasteners as well as their configuration was precisely measured to ensure a standardized socket for the interface. To make it less likely that the user would misuse the interface, a constraint to ensure proper usage was created by using a male snap fastener for the power line, this prevents the user from plugging in a device backwards and causing damage (Figure 1, right).

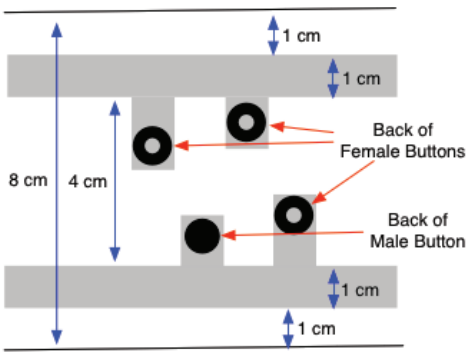


Figure 2. Diagrammatic representation of i*CATch bus.

The need for reliability and to prevent user mistakes was crucial in our considerations. We had considered the use of conductive hook-and-loop material, which would have the advantage of being lighter and blending in better with the substrate material. However, as the hooks and the loops attach to each other more freely than snap connectors, this also creates opportunities for user error with unsecure connections, or with connecting a wrong hook plug to a given loop socket.

Figure 2 is a diagram of how the the i*CATch interface sockets are incorporated onto the i*CATch bus. Since I2C requires two lines for data communication, and most (if not all) modules would need to be connected to the power supply and ground wires, the physical bus uses four 1-cm-wide strips of conductive fabric, each strip carrying one signal or power line. To reduce the physical width of the bus, the conductive strip for the power and ground lines were laid over each other, with the tabs extending in the same direction and an insulating layer of close-weave polyester fabric sandwiched between them (Figure 1, left); the same was done with two data communications lines. At pre-designated intervals, each strip was folded to create a tab that extended from the main strip at a right angle (Figure 1, center). The four strips were positioned 4 cm apart with the tabs extending towards each other and affixed to a cotton lining material using iron-on adhesive. The i*CATch construction platform was then created by using iron-on adhesive to affix the bus to a garment, and attaching the snap fasteners to the fabric tabs. By careful and precise placement of the conductive fabric strips and the positions of the tabs, we can create identical sockets with 4 connection points at designated locations on the garment substrate (Figure 1, right).

Quality Control and Testing

In order for our i*CATch system to be effectively usable by novices, it needs to be stable and reliable. In our experience, the reliability of the construction platform is the difficult part of the system to ensure, as there are multiple places during the construction process in which mistakes could be made. For example, some of the most common

problems happen when the snap buttons are not attached securely enough to the conductive fabric, thereby causing broken circuits when the fabric is folded or crushed. Another common problem occurs when the insulating fabric is not secured around the edges of the conductive fabric, thereby allowing short circuits to occur when the fabric bus is crumpled and the conductive strips slide around and contact each other.

Currently, there exists no conventional testing procedure for textile-based conductive strips integrated into garments. The most straightforward method of testing for conductivity using a multimeter is not thorough enough for bus-based communications, as even a momentary interruption in the conductivity of the strip would cause the bus signal to be disrupted, which may result in the board resetting or the program stalling.

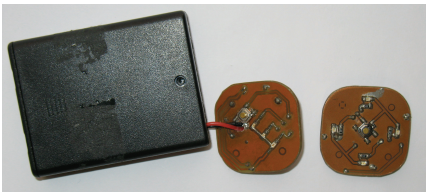


Figure 3. A prototype of the i*CATch test kit. Two modules plug into the sockets of the i*CATch bus; LEDs on the back light up to indicate discovered problems with the connection.

The i*CATch test kit (Figure 3) was designed and developed to address the problem of quality control. The purpose of the kit is to quickly determine whether the connection between each socket is functional. That means that it can:

- Indicate whether the snap fasteners are well attached to the conductive cloth;
- Detect any short-circuits and/or open circuits;
- Indicate whether the I2C signal has been too attenuated over long distances due to the resistance of the conductive cloth.

Our kit includes two electronic modules, one of which is attached to a battery pack. The modules have LED indicator lights on them, which light up in different patterns corresponding to different situations:

- The connection between the two modules is secure and well-established;
- The connection between the two modules was interrupted at some point;
- The connection between the two modules cannot be established.

The design and the construction process of the i*CATch construction platform creates consistently-sized interface sockets with a repeated, reliable and documented process,

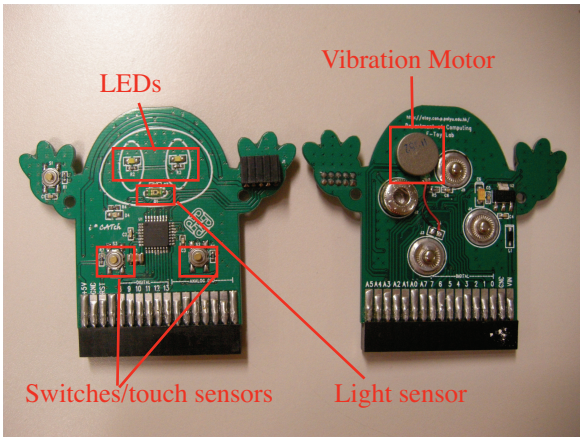


Figure 4. The i*CATch main board

and the i*CATch test kit allows us to reliably control the quality of the i*CATch construction platform as it is manufactured. Furthermore, as our test kit can easily be used by untrained individuals, this allows the manufacturing of our platform on a larger scale. In our laboratory, we manufactured 30 such platforms with the help of 3 first-year undergraduate students, working over a period of 5 half-days.

i*CATch Electronic Modules

In order to work with the new platform, we needed a set of electronic modules that would be compatible with the i*CATch bus. As we wished our platform to be usable by other designers, this precluded the use of proprietary hardware or software.

The i*CATch main board (Figure 4) was designed on the Arduino platform, and used an ATMega 168 chip as the microcontroller, with pins 27 and 28 for I2C communications. From our experience, we noted that most of the introductory tasks that were used to teach programming on a wearable computing platform would invariably revolve around using the same sets of sensors and actuators: turning lights on and off, turning a motor on and off, getting a reading from the light sensor, waiting for a switch to be pressed. To facilitate the use of the i*CATch platform for learning, we therefore designed the main board to include a small set of sensors and actuators on-board -- two LED lights, a light sensor, two switches and a vibration motor. This integration allowed beginners to write simple but functional programs using the main board alone.

On the i*CATch bus, the main board acts as the master device and is responsible for initiating data communications with the peripheral sensors and actuator modules. These modules were also developed based upon the Arduino open-source platform. Each module contained an ATMega 168 microcontroller chip, with Pins 27 and 28 serving as the I2C channel, connected to one or more sensors and/or actuators. Each of these sensors and actuators is identified with its own unique address, and the

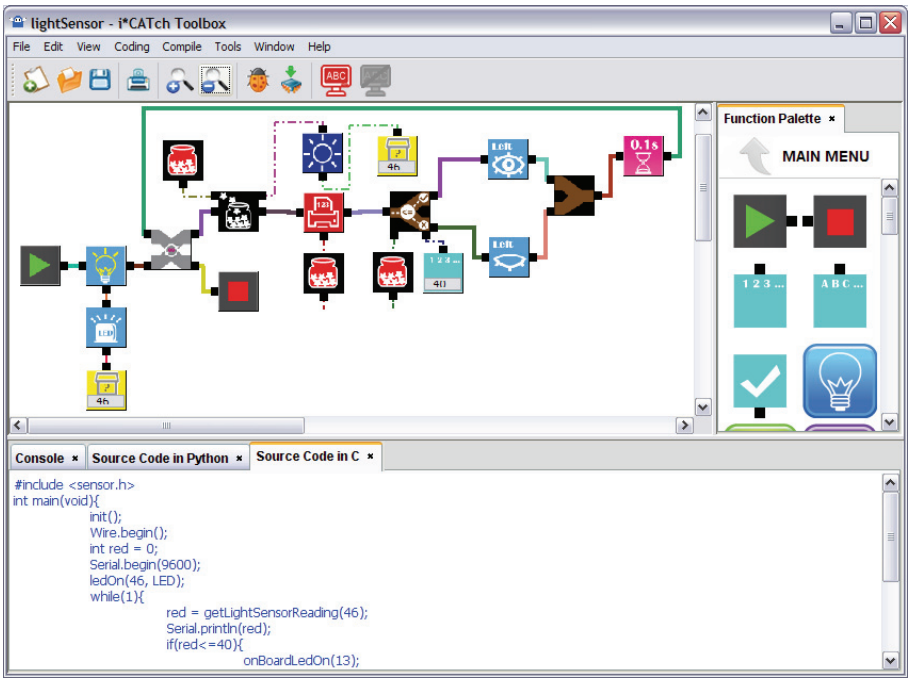


Figure 5. The i*CATch IDE, with a program in the hybrid text-graphical programming language shown. The user has the option of viewing his code in C or in Python, and can switch to textual programming as desired.

microcontroller chip ignores all messages broadcast on the bus except for those that are directed towards its own sensors or actuators. The microcontroller chip therefore functions as a serial peripheral interface (SPI) for the sensors and/or actuators on board.

As the Arduino platform is open-source, other users can develop their own slave programs for the modules, or they can develop their own modules using their own sensors, which is a move towards creating an open standard for the field of wearable computing. The only drawbacks are that the ATMega 168 is somewhat overpowered for this task, and the inclusion of the chip on every peripheral module raises the cost of the modules (by less than US$2 per module at the time of writing), but we felt that it was a small price to pay for open-source user extensibility and a standardizable interface.

To allow the electronic modules to interface with the i*CATch platform, we repurposed the same spring-loaded metal snap fasteners as our connectors. The snap fasteners are soldered to the PCB in the i*CATch interface configuration. This created semi-rigid "plugs" that could be plugged into the "sockets" on the i*CATch bus.

i*CATch Programming Environment

As the i*CATch modules are based upon the Arduino platform, the i*CATch programs are transferable to the Lilypad and other Arduino-based boards. The Arduino integrated development environment [1] can also be used to program our main board.

To make it easier for children to learn programming, we created a hybrid text-graphical programming language that is inspired by Bricklayer [6] and Robolab [18], and developed our own integrated development environment (IDE) for it. Figure 5 is a screenshot of our IDE, which allows programming by dragging and dropping graphical blocks that represent programming constructs and joining them together to denote program flow. As the user constructs his program, the actual source code is generated in real time, thus allowing the user to actually see what is being generated and sent to the i*CATch main board, thus facilitating a later switch to pure text-based programming. At any point, the user may also stop programming in graphical blocks and switch to typing in text programming statements, which allows for more flexibility and power.

Overview of i*CATch Components

We have described the design and development of i*CATch, a scalable, extensible wearable computing kit that is easily manufacturable and can be tested for quality control. Our kit is specially designed for use by novices, and we hope that it will also play a role in the standardization of an interface for wearable computing. The first version of the i*CATch system consisted of the following components:

- A main board, developed upon the Arduino platform;
- A construction platform, with an integrated textile-based bus and a standardized interface for electronic modules;

i*CATch	Teeboard/Lilypad
nil	Electricity and Circuitry
Output (Actuators) and Sequential Logic	
Input (Sensors) and Conditional constructs	
Repetitive constructs	
Putting things together (Project)	

Table 1. Comparison of Syllabus Topics between i*CATch and Teeboard/Lilypad Workshops.

- A set of sensors, including a light sensor, an ultrasonic proximity sensor, and a touch sensor;
- A set of actuators, including seven LED lights, one buzzer and one vibration motor.
- An integrated development environment featuring a text-graphical programming language.

To our knowledge, there is no similar work on the design and development of an entire wearable computing user kit. The Lilypad system is the most similar in nature, but for the reasons mentioned before, it is more appropriate for more experienced users or for small-sized workshops.

EVALUATION RESULTS

Using the i*CATch system

To evaluate our i*CATch framework and to verify that it does indeed fulfill our goals, we organized a 5-day wearable computing workshop. The course was attended by 9 girls of ages 9-12, none of whom had any prior experience with programming or electronics.

The overall objective of the course was to introduce them to programming and electronics through wearable computing. Hence, the earlier elements of the course used small tasks involving just one or two modules to give the participants the necessary programming knowledge, such as conditionals or looping programming constructs. The students were asked to create a demonstrable project at the end of the workshop that would combine these elements and showcase what they had learned.

Figure 7. A project created by one of the TeeBoard/ Lilypad workshop participants. Note the external connecting wires (encased in ribbons for insulation)

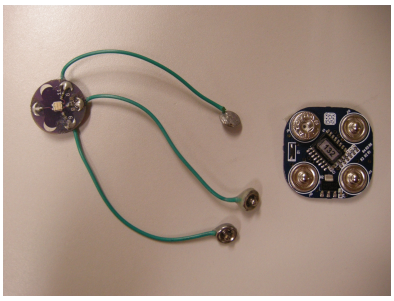


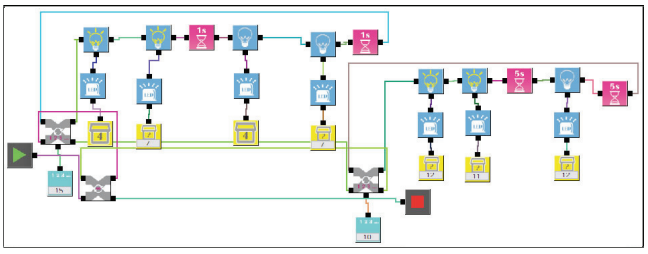
Figure 6. A TeeBoard/Lilypad module (left) and a i*CATch module (right). The i*CATch module does not have loose wires and is less prone to breakage.

Using traditional point-to-point communications

As a comparison, we organized a second workshop that would use the traditional point-to-point architecture. This workshop used the Teeboard construction kit and the Arduino Lilypad modules, which were modified to work with the Teeboard. To ensure a fair comparison, the differences between the workshops were kept as minimal as possible:

- The participants were of similar age and academic level. The TeeBoard/Lilypad workshop was attended by 14 girls and 6 boys, also of ages from 9-12. They also had similar prior experience (as in: none) with programming and electronics.
- A similar structure and participant-instructor ratio were used in both of the workshops. The workshops were also of similar duration; 16 hours for the i*CATch workshop versus 18 hours for the TeeBoard/Lilypad workshop.
- The i*CATch IDE was used in both of the workshops. This was made possible by the fact that both the i*CATch and the Lilypad were based on the Arduino platform; we modified the IDE to handle either a bus-based or a point-to-point-based architecture.

There were two unavoidable differences. First, the use of point-to-point communications in the TeeBoard and the Lilypad modules meant that participants had to learn about basic electricity and circuitry concepts -- for example, understanding the difference between open and closed circuits, learning to avoid short circuits and knowing how to connect devices in parallel and serial configurations -- before they could start creating their constructions. Hence, the syllabus for that workshop started off with two extra hours on electricity and circuits before moving on to programming concepts (See Table 1 for a comparison). Second, there was a slight difference in the available components for the workshops: the TeeBoard/Lilypad workshop did not have buzzers, vibration motors or proximity sensors, which were available in the i*CATch workshop; the i*CATch kit did not have an accelerometer, which was available to the TeeBoard/Lilypad participants.



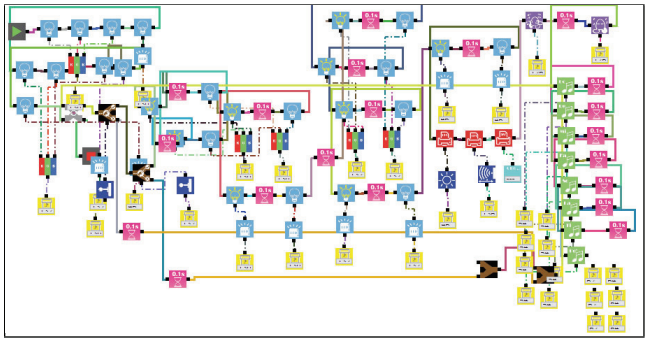


Figure 8. A typical program by the TeeBoard/Lilypad children (top) and by the i*CATch children (right). The i*CATch programs tend to be longer and more elaborate, and use more functions and programming constructs.

Usability Evaluation

To compare the usability of the i*CATch system with the conventional point-to-point architecture, we made observations during the workshop along the several aspects: durability, stability, errors reduced and flexibility. We expected that the bus-based framework of the i*CATch platform would be more flexible and result in less errors from the users; but we expected the durability of the two platforms to be about the same, as they utilized the same technology, and we expected the bus-based platform to be slightly less stable than the point-to-point architecture, since we were transmitting high-frequency signals over unshielded conductive lines.

On the durability aspect, we were surprised to find that the i*CATch modules and construction platform were less easily damaged than the Teeboard/Lilypad system. This can be attributed to the fact that the standardized i*CATch interface allowed us to eliminate the loose wires that were required for the Lilypad modules to connect to the Teeboard (Figure 6), and replace them with semi-rigid plugs and sockets, which were much more durable and could stand up to rougher handling from the children.

On the stability front, we were also surprised that the stability of the i*CATch bus and interface outperformed that of the point-to-point architecture. This can be attributed to the structure of the i*CATch platform and the interface plugs and sockets, which limits the number of connections between the main board and any module to two per line (one each for the main board and peripheral's connection to the bus), as opposed to the extra connection points that were needed in the point-to-point architecture to bridge connection wires. This reduces the probability of loose connections caused by ill-fitting connection points -- one of the most problematic aspects of the construction platform. The more durable standardized i*CATch plugs and sockets also helped to ensure secure connections, as it lowered the probability of loose connections caused by breakage. As a result, we noticed that the i*CATch main board suffered fewer resets and stalled programs than the Teeboard/ Lilypad architecture.

The observations on the error reduction front were as we expected. We noticed during the TeeBoard/Lilypad workshop that many of the children had trouble tracing the circuitry of the conductive strips on the TeeBoard. As a result, many of them resorted to bypassing the internal conductive strips and chose to use external connection wires to construct their circuits instead. Figure 7 shows such an example of external wires (ribbon-encased conductive fabric) being used to complete circuits rather than the internal conductive strips. Several participants complained that the differing number of connections required of the sensors and actuators were confusing. As an example, even at the last workshop session, we had to help debug non-working circuits, many of which had very typical problems such as open circuits, LEDs plugged in backwards, etc. In contrast, the participants of the i*CATch workshop did not encounter these problems, since the i*CATch interface prevents incorrect attachment of the electronic modules.

The flexibility of the i*CATch platform was also observed to surpass that of the TeeBoard/Lilypad platform. One of the advantages of having a bus-based architecture was that the electronic modules could be plugged in anywhere -- the programming would still remain the same. In the i*CATch workshop, we noticed that the participants would often refer to the simple programs they wrote when they were working on the early tasks, and copy over the entire chunk into their final project program. As the addresses of the modules did not change, their program would still work, even though the location of the module had changed. This was more difficult for the participants in the TeeBoard/ Lilypad workshop, as moving the locations of the devices would necessitate a change in the pin numbers before the program would work. The upshot was that the i*CATch participants were more willing to experiment with changes in their design, since their program was more portable; they were also willing to write more elaborate programs for their project. This will be further discussed in the Pedagogical Evaluation section.

Another way in which the i*CATch platform enhances flexibility is its ability to incorporate more modules. On the Lilypad platform, the number of modules that can be supported is limited to the number of pins on the board -- 14 digital pins and 6 analog pins. Given how some modules, such as the accelerometer and the multi-colored LED lights, require multiple input channels, all input ports

are very quickly fully occupied. As an example, each multi-colored LED requires three input channels for the red, green and blue lights -- hence the number of such lights that could be supported by the Lilypad would be limited to 5 at a maximum unless serious hacking was applied. The i*CATch platform does have a maximum number of devices, but this number is much higher -- theoretically, we could connect up to 128 different devices onto the platform!

Overall, the i*CATch platform demonstrates superior usability in terms of durability, stability, error reduction and flexibility, as compared to the TeeBoard/Lilypad platform.

Pedagogical Evaluation

In addition to the usability evaluation of the i*CATch platform, we also wished to evaluate our platform pedagogically. Simply put: does the increased durability, stability, error reduction and flexibility of the i*CATch platform result in added pedagogical value for the users?

We evaluate the pedagogical value of the i*CATch platform along two lines: the level of creativity that it encourages, and the knowledge and skills that were acquired by the participants.

On the knowledge and skills aspect, one major contribution of the i*CATch platform was that it allows more time to be spent on programming, and it also encourages good programming practices of code reuse, divide-and-conquer coding strategies and modular design. For example, since the programming code for a given sensor or actuator is the same regardless of location, we observed many instances in which the participants reused code from the earlier, simpler tasks into their final project. We also observed many instances in which the participants would work on developing the functionality for one or two sensors at a time (the speaker was a common choice, they loved composing their own tunes on it!), saving the program in a separate file, working on another couple of sensors, and then finally integrating everything into a final product. In comparison, the participants in the TeeBoard/Lilypad workshop, tended to use a "tear it down and start from scratch" mode of programming. We attribute it to the fact that programs for a bus-based architecture are inherently more portable and reusable than programs for point-to-point architectures. As a result, the programs written by the i*CATch children tended to be longer and more elaborate (on average, about 100 icons) than those written by the TeeBoard/Lilypad children (about 40 icons on average). There was no noticeable difference in the variance of the programs generated by the children. Figure 8 shows two typical programs as examples.

In addition to program length, the complexity of the projects is another way to ascertain the knowledge and skills acquired by the participants. In general, the i*CATch workshop participants used more modules (many of them filled up all available sockets) than the TeeBoard/Lilypad participants, many of whom did not fully utilize all the pins on the Lilypad. (The project in Figure 7 only uses six of the Lilypad's pins.) The reason is because circuit construction on the TeeBoard/Lilypad platform is more complex and involves more dependencies (for example, different modules "fighting" over the same conductive strip), which discourages children from making overly complex circuits and to stick to things that they already know instead. Confirming this point, many of the participants stated that once they had something that worked, they were reluctant to modify it in case they broke something.

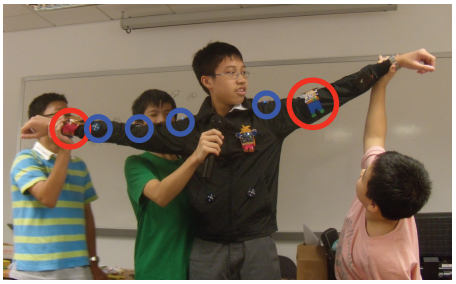


Figure 9. An advanced project demonstrating the flexibility of the i*CATch platform: two touch-sensor-embedded gunmen (circled in red) fire at each other; LEDs along the arms (circled in blue) flash on to indicate the trajectory of the bullet.

The creativity aspect is harder to evaluate, as it does not make sense to compare "levels of creativity" between projects that were constructed over such a short period of time. However, an overview of the constructed projects confirms that the i*CATch platform does indeed support creativity in wearable computing construction. For example, Figure 9 demonstrates the extra power that the i*CATch platform affords the user. This project, which was created by a group of boys aged 12-15, has two felt cut-outs of gunmen, embedded with touch sensors, are positioned at the arms of the garment; a series of LEDs along the shoulders flash in order, one after the other, at the trigger of the embedded touch sensors to indicate the trajectory of the bullet when one of the gunmen fires his gun. We believe that the flexibility of module placement that this project requires would have been very difficult to achieve with point-to-point architectures. This also validates that the i*CATch framework can be used to support creativity and design in users of varying levels.

In general, the difficulties and dependencies involved in modifying point-to-point constructions encourages careful layout design before implementation, which is reminiscent of the design process that is required before traditional structured design programming. In contrast, the flexibility of the i*CATch platform encourages more iterative design, experimentation, and development.

The i*CATch platform has one drawback, which would be true for most plug-n-play systems. Myers et al [14] argue

that a good toolkit should have a high ceiling, where much can be designed, and a low entry threshold. As the Lilypad platform is free-form, it allows users to place their modules anywhere on the garment, while the i*CATch platform requires that the modules be plugged into pre-specified socket locations. Therefore, it may be argued that the Lilypad system has a higher ceiling at the price of a higher entry barrier, while the i*CATch platform has a low entry threshold but a lower design ceiling.

CONCLUSIONS

We have presented the design and development of the i*CATch wearable computing framework, which is designed and developed to support plug-and-play construction of elaborate wearable computing creations by novices and children. We achieved this through the adoption of a scalable bus-based architecture for the communications network, a standardized interface for the modules and a hybrid text/graphical programming language. Our framework was verified to fulfill all its objectives through evaluation in real multi-day, multi-participant teaching environments.

REFERENCES

1. Arduino. Available: www.arduino.cc.
2. Berzowska, J. Electronic textiles: Wearable computers, reactive fashion, and soft computation. Textile, 3(1):2–19, 2005.
3. Buechley, L. Lilypad Arduino | build something. http://web.media.mit.edu/~leah/LilyPad/build.html
4. Buechley, L., and Eisenberg, M. Fabric PCBs, electronic sequins, and socket buttons: Techniques for e-textile craft. Journal of Personal and Ubiquitous Computing (2007).
5. Buechley, L., Eisenberg, M., Catchen, J., and Crockett, A. 2008. The LilyPad Arduino: using computational textiles to investigate engagement, aesthetics, and diversity in computer science education. In Procs. of CHI'08 (Florence, Italy, April 05 - 10, 2008), 423-432.
6. Cheung, J. C., Ngai, G., Chan, S. C., and Lau, W. W. 2009. Filling the gap in programming instruction: a text-enhanced graphical programming environment for junior high students. SIGCSE Bull. 41, 1 (Mar. 2009), 276-280.
7. Eduwear project. http://dimeb.informatik.uni-bremen.de/eduwear/.
8. Eleksen, "Belkin Selects ElekTex to Power New-to the-Market SportCommand Product," Sep. 19, 2006, retrieved from the internet at http://www.eleksen.com/?page=news/index.asp&newsID=61, on Dec. 11, 2006, pp. 1-2.
9. Gorlick, M. M. Electric suspenders: A fabric power bus and data network for wearable digital devices. In ISWC '99: Proceedings of the 3rd IEEE International Symposium on Wearable Computers (Washington, DC, USA, 1999), IEEE Computer Society, 114.
10. I2C, A networking solution for integrated circuits, http://www-us2.semiconductors.philips.com/i2c/I2C
11. Junker, H., Lukowicz, P., Tröster, G., "PadNET: Wearable Physical Activity Detection Network," iswc, pp.244, 7th IEEE International Symposium on Wearable Computers (ISWC'03), 2003
12. Lau, W. W. Y., Ngai, G., Chan, S. C. F., and Cheung, J. C. Y. Learning programming through fashion and design: A pilot summer course in wearable computing for middle school students. In Proc. of SIGCSE '09 (Chattanooga, TN, Mar 2009).
13. Lehn, D., Neely, C., Schoonover, K., Martin, T., and Jones, M. e-tags: e-textile attached gadgets. In Communication Networks and Distributed Systems Modeling and Simulation Conference (January 2004).
14. Myers, B., Hudson, S. E., and Pausch, R. 2000. Past, present, and future of user interface software tools. ACM Trans. Comput.-Hum. Interact. 7,1 (Mar. 2000), 3-28.
15. Nanda, G., Cable, A., Bove, V. M., Ho, M., and Hoang, H. 2004. bYOB [Build Your Own Bag]: a computationally-enhanced modular textile system. In Procs of the 3rd Int'l Conf on Mobile and Ubiquitous Multimedia (College Park, MD, Oct 27 - 29, 2004). MUM '04, vol. 83. ACM, New York, NY, 1-4.
16. Ngai, G., Chan, S. C., Cheung, J. C., and Lau, W. W. 2009. The TeeBoard: an education-friendly construction platform for e-textiles and wearable computing. In Procs of CHI'09. (Boston, MA, USA, April 04 - 09, 2009). 249-258.
17. Orth, M., Post, R. and Cooper, E.. Fabric computing interfaces, SIGCHI: ACM Special Interest Group on Computer-Human Interaction 1998
18. Portsmore, M., ROBOLAB: Intuitive Robotic Programming Software to Support Life Long Learning, APPLE Learning Technology Review, Spring/Summer 1999
19. Post, E., Orth, M., Russo, P., and Gershenfeld, N. Ebroidery: Design and fabrication of textile-based computing. IBM Systems Journal 39, 3-4 (2000), 840-860.
20. Park, S., K. Mackenzie and S. Jayaraman. The wearable motherboard: a framework for personalized mobile information processing (PMIP) 2002