Decoupling Interaction Hardware Design Using Libraries of Reusable Electronics

Rajesh Sankaran, Brygg Ullmer, Jagannathan Ramanujam,
Karun Kallakuri, Srikanth Jandhyala, Cornelius Toole, and Christopher Laan
Louisiana State University: Dept. of Electrical Engineering,
Dept. of Computer Science, and Center for Computation and Technology (CCT)
rajesh@cct.lsu.edu, ullmer@lsu.edu

ABSTRACT

This paper presents our research toward the design and development of a library of electronic hardware modules called *Blades and Tiles*. Interaction hardware design with blades and tiles provides researchers with considerable flexibility in iterating their designs, decoupling between the domains of electronics, software, firmware and mechanical design. Our approach has been driven by design objectives including hardware reusability, reliability, scalability, and flexibility. We have created a library of blades and tiles, and used them to develop several interaction devices. We present both conceptual and applied aspects and discuss future directions.

Author Keywords

Hardware toolkit, reusable hardware, blades and tiles, decoupling TUI design, modularity.

INTRODUCTION

The integration of electronic, mechanical, software, and interaction design presents a challenging design space for tangible user interface (TUI) researchers. The physicality of such interfaces also makes them more complex and time consuming to prototype than pure software systems, often requiring expertise in various areas. Fortunately, the creation of hardware toolkits and prototyping systems (e.g. [1–4, 7–13]) is easing the process of realizing novel physical interfaces. These toolkits are evolving significantly in their ability to support a wide variety of interaction modalities.

The hardware toolkits community has made valuable progress toward bootstrapping the initial development of TUIs, especially from the perspective of software developers. TUI researchers are invested and use these toolkits in the development of various promising interaction devices. This is evident in the growing number of users embracing hardware prototyping platforms such as d.tools [5], Arduino [3] and Phidgets [4] (to name a few). However, we feel several conceptual and technical developments in the architecture of the hardware toolkits can yield a number of benefits.

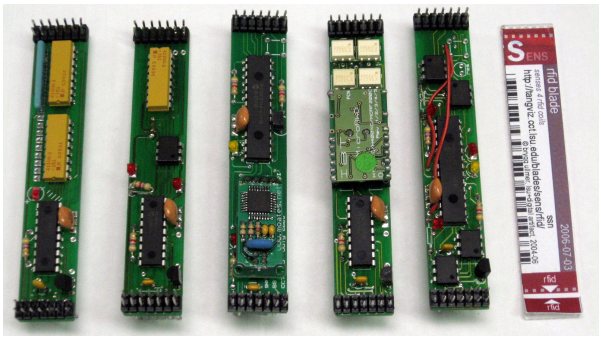

Figure 1. Close up of *Blades*: The figure shows from left, Switch-LED, Haptic Rotor, USB-Gateway, RFID, and Intracomm blades, and Blade label that sticks to the bottom of a blade. The barcoded and color-coded blade-labels provide blade details.

From a hardware standpoint, the creation of novel interaction devices involves advancements in at least two different realms. Fundamentally, new sensors and/or devices – i.e. (interaction modalities) [15] – are investigated and developed. Simultaneously, the design and composition of interaction devices, application integration, and interaction sequences are explored.

While many existing toolkits provide users with accessible avenues to program pre-existing toolkit hardware, they often lack straight-forward means to integrate user-developed interaction modalities (e.g., custom electronic modules and firmware e.g., for PWM of LEDs or haptic feedback) with existing toolkit elements. The interaction researcher is often restricted to using only the interaction modalities provided by the toolkits, as typically developed by a small set of hardware designers. We believe several factors contribute to this:

- existing toolkits have typically been designed to provide pre-built interaction electronics for use by researchers with predominantly software skills;
- the tightly-coupled electronics and firmware components of many toolkits do not grant easy access for developers outside the core hardware team; and
- there is often a lack of documented hardware schematics, protocols, and software architectures to foster the development of new interoperable toolkit elements by users.

Prior toolkits have simplified hardware design, and reduced the level of technical electronics expertise required to build and prototype TUIs in small numbers. However, we have not found any toolkits that support the construction of moderately larger numbers of devices (e.g., ∼10-100) for research purposes (e.g., colocated or distributed collaborative systems) or stable deployments. Likewise, it has been challenging to develop prototypes that integrate many (e.g., hundreds) interactors in a scalable fashion.

We propose *Blades and Tiles* as a step toward realizing the above-described architectural features. Through blades and tiles, we aim to empower the interaction design community with the freedom to develop and integrate their implementations of interaction modalities, and benefit from standards for the sake of easier adoption and interoperability. Blades and tiles are a library of extensible, reusable hardware entities designed with the intent of supporting community collaboration in the vein of open source software development. Here we describe the design and technical aspects of blades and tiles, example interface implementations using them, and discuss future directions.

EXAMPLE INTERFACES

Blades and tiles can be used to build a variety of interfaces. To illustrate their use, we give several concrete application examples.

Core tangibles: interaction trays

We have developed several *interaction trays*, as one embodiment of the core tangibles concept [20]. Figure 2 shows one

Figure 2. *Parameter Tray*: This tray has been implemented using two tiles and ten blades.

embodiment of a three-wheeled parameter tray. To implement this device, we have used two radio-frequency identification (RFID) blades; two Switch-LED blades; and three haptic rotor blades plugged into two tiles. Each tile also contains an "intracomm" blade, which serves as a message router. These tiles plug into a *spine* module that hosts one of several gateway blades – e.g., USB and Bluetooth – for linkage with an external PC or embedded gateway computer.

LONI interaction device

Where the parameter tray of figure 2 is a relatively general-purpose interaction device, we have also used blades as core elements of a class of special-purpose interaction devices we call tangible visualizations [20]. For example, Figure 3 shows an interaction device specific to a statewide network infrastructure named LONI. Here, a Louisiana-shaped printed circuit board (PCB) hosts an array of RFID readers at the eight key city/nodes of the network; and a series of linear LED arrays representing the networks connecting these nodes. These RFID and LED elements are supported by 9 blades including 4 LED and 2 RFID blades, which plug into the back of the PCB. The PCB itself contains only RFID antenna coils; LEDs; and connectors to the blades. Here,

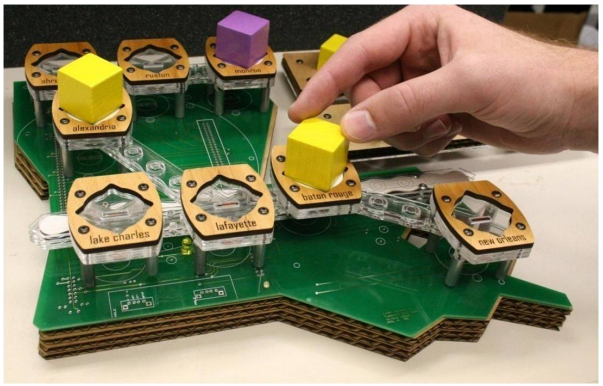

Figure 3. *LONI Interface*: The PCB is shaped like the state of Louisiana and incorporates the tile circuits.

the whole LONI PCB serves as a special-purpose "tile." (A larger-scale LONI interface could also be built from a series of tiled subelements.) As the actual LONI network evolves in times, with city/nodes and network links added and removed, this bladed approach significantly reduces the complexity of implementing, debugging, and refining the interface.

Fan-based interfaces

The previous two examples are interfaces we have implemented. As a first illustration for how they might be used to support interfaces designed by others, leveraging some of the scaling properties of our blade-based approach, we briefly describe how the Blow Displays by Minakuchi et al. [14], and Murmur by Rydarowski et al. [17] could be implemented using blades and tiles. As built by Minakuchi et al., the Blow Display uses two fans driven by a Phidgets Motor Controller to divert user attention towards a PC monitor. This prototype can be built using a *power blade*. A power blade has four solid-state switches each supporting a current of 1 ampere. These switches can be turned on/off or pulsed, and can also be used to control larger relays (Figure. 4).

The Murmur system [17] illustrates how bladed implementations can naturally scale, as well as provide convenient paths for progressively adding or modifying interaction modalities. Where the Blow display contains two fans, Murmur is described as including one hundred. With the scalable blades and tiles design, using numerous unmodified power blades, the Mumur hardware can be realized. Tiles, each accommodating four power-blades can be chained to energize the fan mesh, with the blades controlled via a USB and Bluetooth interface from a supporting PC. Alternately, tiles could be

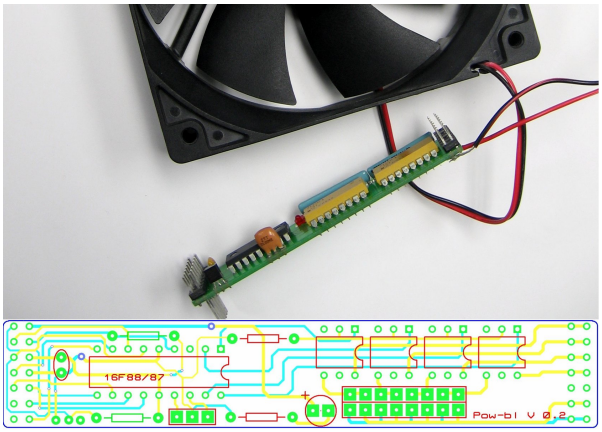

Figure 4. ***Power Blade and Fan*****: Top: A CPU fan can be connected to a power blade or can be switched using a relay in case of large power requirements. Bottom: Power blade schematic.**

left partially populated (with free blade slots). This retains prospects for subsequent integration of additional interaction modalities (e.g., promixity sensors, RFID sensors, LEDs, etc.) with no modifications to the underlying infrastructure. When thus implemented, it is also relatively straightforward to scale up the system to hundreds or thousands of nodes.

To be clear, in these systems, the electronics, firmware, and protocols represent a quite partial subset of the larger conceptual, software, mechanical, and user-centric implementational efforts. At the same time, we believe the bladed approach enables a highly valuable form of design and engineering decoupling. Subteams are enabled to work in parallel, while retaining greater prospects for implementational iteration of system subcomponents.

PREVIOUS AND RELATED WORK:

Our work has been inspired in part by hardware prototyping toolkits such as Phidgets, Teleo, and Ardiuno. Saul Greenberg et al. developed the Phidgets "plug and play" [4], building blocks that provide easy access to hardware sensors, actuation, and control via PCs. The Phidgets platform made device creation sufficiently simple that developers could concentrate on other issues such as form, use and design.

The Arduino [3] hardware and software environment has been popular amongst artists and interaction designers alike. Arduinos are designed to be stand-alone or to communicate with software on a computer (e.g. Flash, Processing, MaxMSP). We see strong potential for integrating blades as supporting functional libraries in Arduino-based systems.

Van Laerhoven et al. designed and presented a true physically distributed system of interaction elements in their "Pin and Play" concept [12]. Their system, with a multi-drop communication network, that uses surface as the medium for communication, demonstrates the flexibility and advantages of a decentralized approach.

In this paper we discuss the use of Inter-Integrated Circuit bus (I^2C) as the primary communication protocol in the blade tile system. The Smart-Its [2] were developed collaboratively by Lancaster University, ETH Zurich, University of Karlsruhe, Interactive Institute and VTT and also used I^2C interface for sensor/actuation boards. Similarly, Hartmann et al., use I^2C bus for communication between their controller board and hot-pluggable input and output components in their d.tools system [5].

Open-ended systems with generic connectors can present certain challenging safety and fault tolerance issues. In [19], J. Scott et al. deal with generic real-time configurable interconnects and discuss their mechanisms for fault tolerance. Finally, the form factor of our tiles has been motivated by the DataTiles [16] created by Rekimoto et al.

IMPLEMENTATION

Some of the basic concepts underlying blades and tiles have been introduced in [18]. In the following, we briefly introduce some of the mechanical, electronic, firmware, software, protocol, and architectural implementations and design decisions underlying blades and tiles.

Mechanical

Blades and tiles systems center around the integration of multiple interaction modalities as self-contained network nodes that can be added and removed from the underlying communication network. We have created and followed several system-wide standards regarding mechanical design.

Blade and tile dimensions

Drawing inspiration from DataTiles [16], tiles are prototypically 10x10cm. This is a a modal size corresponding to the power hand grasp, to many other tangibles; and one which affords composition of multiple tiles. Blades (Figure 1) are prototypically 10 cm long; 2 cm wide; and 1 cm deep. For implementing interaction devices at the scale of our tiles, we have found this dimension a promising tradeoff of physical real estate with ability to accomodate standard components.

As a specific example underlying blade width, 2 cm width will exactly accomodate a 28-pin standard-width through-hole integrated circuit. This is the maximum width common among through-hole components, and the size of (e.g.) the popular IB Technology family of RFID reader hybrid modules. In cases where 2cm width is insufficient, we suggest integer multiples of width (4cm, 6cm, etc.). E.g., our stepper motor blade is 4 cm wide, on account of wider driver and power components.

Connectors

The blades and tiles system is designed to facilitate addition and removal of interaction components, both at the blade and tile levels of granularity. We presently use two sets of pluggable USB connectors as tile-to-tile interconnects; and two sets of 2x7 headers for attaching blades to tiles. We employ a pair of 70 pin female headers to both electrically connect and mechanically fixture blades to tiles.

Electronic components

We currently build all blades and tiles with through-hole electronic components [21]. Through-hole components can be easily attached to a circuit boards (PCB) by developers with minimal soldering experience. While through-hole components have limitations in terms of size and density, we have found that they provide considerable flexibility for prototyping and small fabrication runs. Simultaneously, while through-hole components are preferred for human assembly, surface-mount components are better suited for machine (mass) assembly, and lead gracefully to smaller future standard dimensions for blades and tiles.

Electronics

Tiles

Tiles (Figure 5) form the meeting ground between the electronics underlying specific interaction elements (e.g. switches, LEDs, rotary encoders) and the blades supporting their underlying implementations. Additionally, tiles can provide mechanical structure for electronic components and mounting external housings.

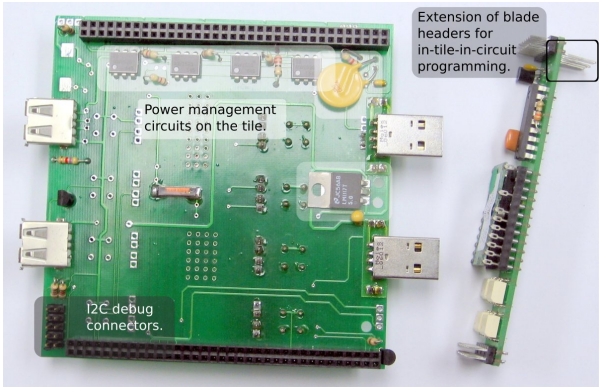

Figure 5. Bottom of a *Tile*: The underside of a tile houses the tile specific circuits consisting of power management, communication and power circuits. This tile has been designed with some perforated PCB space for rapid prototyping.

Tiles incorporate three kinds of electronic elements: circuits for blade management and control (e.g. blade connection headers, power control & distribution), circuits that implement tile specific electronics (e.g. tile id, tile interconnects, power converter, debug circuits), and application specific components (typically interactors like RFID coils and LEDs).

Blades

Aside from power, communications, and external interactors, blades are designed to be self-sufficient, eliminating strong linkages between electronic realization of different modalities. They are comprised of two types of components, shown in Figure 6: a microcontroller for local intelligence, and electronic components and circuits required to realize the interaction modality. Blades currently use Microchip's PIC series of microcontrollers, but are fundamentally designed to be processor independent. Blades are uniquely identified (electronically) by a six byte ID hard-coded in a DSN chip (Dallas DS2401).

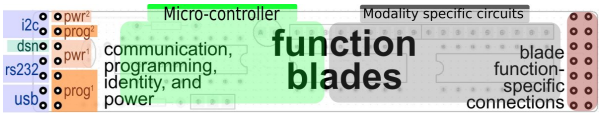

Figure 6. *Function Blades*: The figure shows the logical parts, and pin-outs of a function blade. Blades contain two sets of interconnects, one generic across all blades with power, programming, and communication pins, and the other blade specific. The circuits on a blade comprise of a intelligent processor and interaction specific circuits.

Blades fall into three main classes: *core*, *function*, and *resource* blades. *Core* blades mediate the communication between blades. They also provide the communication interface to connect the hardware with external entities. Additionally, core blades ensure safe operation of the system: interrogating blade IDs before soft power-up; detecting faulty blades; and supporting active power management.

Function/interaction blades implement specific interaction modalities. They are broadly classified as *sense, actuate, and display* blades. Most blades fall in this category. *Resource* blades implement supporting resources to augment function and core blades. Examples include battery power, additional memory, supporting computational capabilities, and data encryption.

Blades connect to tiles through two 14 pin interconnects. One is common across all blades; the other, blade-specific. The common interconnect provides linkage to power, in-tile + in-circuit blade programming, blade ID interrogation, and blade communication. The configuration of this interconnect has been refined over many iterations to support in-tile-in-circuit programming of different microcontrollers with off-the-shelf programmers; separate power sources for low and high power circuits on-board a blade; and add various communication protocols to blades.

Firmware, communication architecture, and protocols

Blade firmware has been written in C using CCS's proprietary We believe that with minimal modifications (primarily I^2C and RS232 calls), the code can be made compatible with other C compilers, both for PICs and other architectures.

In contrast to systems like Arduino, which typically employ high-ly integrated hardware implementations of interaction modalities, our architecture is deeply distributed, with dependence on communication and hardware level APIs at several levels of abstraction.

Device-level communication and protocols

Most bladed interaction devices are built with more than one blade. We have found our medium-complexity interaction

devices typically integrated 3 to 10 blades, although also envision complex systems where this number may be significantly higher (e.g., hundreds of blades). Communication within a device is channeled at two levels; globally between tiles, and locally between a tile's blades. At a low level, a packet-based protocol is used. Our protocol supports both global and hierarchical communication buses.

Our current blades communicate over a 100 kHz I^2C network for inter-blade data transfer. I^2C is a multi-master serial computer bus invented by Philips that is largely used to attach low-speed peripherals. We also use a 1-Wire bus for communicating with Dallas Semiconductors' 1-Wire DSNs for blade and tile identification.

The I^2C protocol only defines primitive methods for sending and receiving network messages. To realize our goal of a scalable, hot-pluggable, dynamically reconfigurable system, we have developed a suite of supporting procedures, protocols, and algorithms. Our transport protocols have been modeled on TCP/IP and the CANbus embedded protocol. Some of these include:

- Dynamic blade discovery and I^2C address assignment;
- Standardized I^2C message packet structure adapted to and taking advantage of distributed intelligence;
- Dynamic branching and merging of I^2C busses for various methods of communication; and
- On-the-fly I^2C address decoding and message routing.

Communication between blades is carried out in a time division multiple access (TDMA) fashion. The communication blades switch between bus-master and bus-slave configurations per our protocol, wherefunction and resource blades function purely as slave blades. All blades currently uniformly communicate with each other and an external system ten times a second. Our communication protocol has also been designed to support different blade update rates, and we believe substantially accelerated update rates are implementationally near at hand.

External Communication and Protocols

Most interaction systems we have implemented and deployed involve more than one interaction device built using blades-tiles. These devices are being used in environments involving multiple operating systems and high-level software packages (e.g. the amira and VisIt visualization environments). Correspondingly, software and protocol implementations at this level are designed to provide higher level abstractions, with emphasis on language independence. After completing early implementations using XML-RPC, we are successively migrating to ICE [6] to take advantage of improved update rates and a variety of transport mechanisms.

Software

Software for integrating blade+tile-based interaction devices with external systems has undergone several iterations. These have supported interaction device research; four semesters of class use; and applications to several end-user contexts, including co-located and distributed collaborative visualization and microscopy imaging. For portability, TUI integration software has been written in Python, Tcl, and Java.

We have observed our software converging toward two distinct modes of usage. The first implementational mode couples bladed hardware as mixed hardware/software library elements of medium- to high-end computational systems. The second combines blades as library elements for small computational systems – e.g., as function-specific co-processors for Arduino, Rabbit core modules, or other microcontroller-based systems.

Thus far, our efforts have concentrated on the first mode of use. This typically takes the form of integration with a supporting PC or gateway computer. Here, we are converging toward a layered architecture in which data is successively refined and transformed into high-level interaction events.

At a lower level, we have implemented a device driver tasked with collecting raw data from a blades+tiles tangible. The next level – the service layer – involves a tangibles interaction server which collects and redistributes events to the appropriate target. We are also implementing additional services including session handling, and management + coordination models for interaction devices and the target applications to which they are bound (e.g., at the application layer).

FUTURE WORK

Blades and tiles have undergone many revisions of design and testing, and have been successfully used in four mixed graduate/undergraduate project courses (two with $\sim$35 students). Based on these experiences, we plan several areas of enhancement in the near future, while keeping the system backward compatible.

Bandwidth: The current design of blades and tiles is constrained by bandwidth restrictions due to its low datarate USB serial and Bluetooth serial interfaces to external systems. Unfortunately, besides the 100 kbit/s standard mode I^2C [1], we were unable to identify a suitable scalable-frequency multi-drop bus for internal communication. High-speed I^2C supports a bandwidth of 3.4 Mbit/s, but is not yet implemented in most micro-controllers. Fortunately, the available overall bandwidth seems sufficient for most of our current needs.

With a view toward the requirements of future systems, we are working to alleviate this problem by developing gateway comm blades with significantly more powerful processors and high-bandwidth USB and WiFi support. Our efforts thus far include integration of embedded Linux systems like the Gumstix modules. Gumstix have a nearly identical form factor as blades; we are working on several implementations of bladified Gumstix.

Interoperability with other hardware systems is one of our driving design objectives. We have not yet implemented a simple way to interface blades and tiles to other existing

[1] More info on I^2C data rates can be obtained from: `http://en.wikipedia.org/wiki/I2C`

toolkits and prototyping hardware. These existing toolkits could interoperate with blades+tiles at both software and hardware levels. We are actively investigating both prospects. Both forms of interoperability would provide developers with a higher level of hardware abstraction and a wider selection of hardware implementations.

New blades and processor architectures: Our current blade implementations primarily use Microchip's PIC16F87 and PIC16F876A. We have focused our hardware development efforts predominantly towards refining the design of the total system and implementing needed functional submodules on demand. Having produced a scalable and stable design, we have begun developing a new set of blades including a battery blade, Gumstix blade, slider blade, and a/d (analogy/digital) blade. Additionally, building on observations of other toolkits, we are considering blades based on AVR microcontrollers, ARM processors, and embedded Linux systems such as the Gumstix family of modules.

DISCUSSION

Scalability: In principle, blades and tiles have been designed to be highly scalable. In practice, the effects of communication bus capacitance, power issues (both current and noise), and latency (esp. due to multiple levels of hierarchy) restrict the size and growth of our system. We have successfully tested networks of $\sim$40 blades and a dozen tiles with our current hardware. For broader scaling, we believe the use of multiple power sources and I^2C bus isolation+extension using repeaters could provide possible paths forward.

Size and shape: We have interests in experimental scientific apparatus, interaction devices, and environments employing bladed implementations in diverse form factors. For some, regular shapes such as tiles contribute toward prospects for composability. In other cases, such as our LONI interaction device, irregular. Furthermore, we envision blades and tiles could be produced in a variety of standard sizes (perhaps analogous to the ISO paper families of A0, A1, etc.). Here, complex blades could be realized as aggregates of smaller blades, potentially blurring the line between blades and tiles.

Beyond user interfaces: Our development of blades and tiles has grown from our research on tangible interaction design. In addition, we believe the current and evolving architecture holds relevance and strong potential well beyond the domain of user interfaces. We also hold strong interest in open-source and multi-source prospects for both blades and bladed interaction devices,

CONCLUSION

Building on a relatively simple set of combined hardware and software design principles introduced in [18], we have described their realization in the form of Blades and Tiles. Building on prior hardware toolkits and our experiences with implementing tangible interfaces, we have designed and developed a robust, scalable system. We have also described several concrete example applications and their implementation using blades and tiles.

While some of these principles are not individually novel, we believe our combined approach provides blades and tiles with powerful properties as library elements for decoupling the electronic, firmware, and software elements of complex, evolving physical/digital systems. We plan to release our implementation as open-source hardware and software in hopes of fostering wider collaborative efforts towards hardware interoperability.

ACKNOWLEDGEMENTS

This research has been supported by NSF MRI 0521559. Special thanks to Jerry Trahan, Christopher Branton, Alex Reeser, Alvin Wallace, Cole Wiley, Jennifer Claudet, Kexi Liu, Michael Carroll, Mohamed Diabi, Mariel Losso, Susie Poskonka, Shining Sun, and Zachary Dever.

REFERENCES

1. Rafael Ballagas, Meredith Ringel, Maureen Stone, and Jan Borchers. istuff: a physical user interface toolkit for ubiquitous computing environments. In *CHI '03*, pages 537–544, 2003.

2. Michael Beigl, Tobias Zimmer, Albert Krohn, Christian Decker, and Philip Robinson. Smart-its - communication and sensing technology for ubicomp environments. Technical report, The Telecooperation Office (TecO), 2003.

3. Joliffe D. Arduino fever. *MAKE V7*, pages 52–53, 2006.

4. Saul Greenberg and Chester Fitchett. Phidgets: easy development of physical interfaces through physical widgets. In *UIST '01*, pages 209–218, 2001.

5. Björn Hartmann, Scott R. Klemmer, Michael Bernstein, Leith Abdulla, Brandon Burr, Avi Robinson-Mosher, and Jennifer Gee. Reflective physical prototyping through integrated design, test, and analysis. In *UIST '06*, pages 299–308. ACM, 2006.

6. Michi Henning. A new approach to object-oriented middleware. *IEEE Internet Computing*, 8(1):66–75, 2004.

7. Scott E. Hudson and Jennifer Mankoff. Rapid construction of functioning physical interfaces from cardboard, thumbtacks, tin foil and masking tape. In *UIST '06*, pages 289–298, 2006.

8. Hiroaki Kimura, Eiji Tokunaga, Yohei Okuda, and Tatsuo Nakajima. Cookieflavors: easy building blocks for wireless tangible input. In *CHI '06*, pages 965–970. ACM, 2006.

9. Scott R. Klemmer, Jack Li, James Lin, and James A. Landay. Papier-mache: toolkit support for tangible input. In *CHI '04*, pages 399–406, 2004.

10. Shigeru Kobayashi and Masayuki Akamatsu. Spinner: a simple approach to reconfigurable user interfaces. In *NIME '05*, pages 208–211, 2004.

11. Shigeru Kobayashi, Takanori Endo, Katsuhiko Harada, and Shosei Oishi. Gainer: a reconfigurable i/o module and software libraries for education. In *NIME '06*, pages 346–351, 2006.

12. Kristof Van Laerhoven, Albrecht Schmidt, and Hans-Werner Gellersen. Pin&play: Networking objects through pins. In *UbiComp '02*, pages 219–228, 2002.

13. Johnny C. Lee, Daniel Avrahami, Scott E. Hudson, Jodi Forlizzi, Paul H. Dietz, and Darren Leigh. The calder toolkit: wired and wireless components for rapidly prototyping interactive devices. In *DIS '04*, pages 167–175, 2004.

14. Mitsuru Minakuchi and Satoshi Nakamura. Collaborative ambient systems by blow displays. In *TEI '07*, pages 105–108, 2007.

15. Laurence Nigay. *Design Space for Multimodal Interaction*. 2004.

16. Jun Rekimoto, Brygg Ullmer, and Haruo Oba. Datatiles: a modular platform for mixed physical and graphical interactions. In *CHI '01: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 269–276. ACM, 2001.

17. Aimee Rydarowski, Ozge Samanci, and Ali Mazalek. Murmur: kinetic relief sculpture, multi-sensory display, listening machine. In *TEI '08*, pages 231–238, 2008.

18. Rajesh Sankaran, Brygg Ullmer, and Srikanth Jandhyala. Blades and tiles: An extensible hardware architecture approach for ubiquitous interaction devices. In *UbiComp '07 Adjunct Proceedings*, 2007.

19. James Scott, Frank Hoffmann, Mike Addlesee, Glenford Mapp, and Andy Hopper. Networked surfaces: a new concept in mobile networking. *Mob. Netw. Appl.*, 7:353–364, 2002.

20. Brygg Ullmer, Rajesh Sankaran, and et al. Tangible menus and interaction trays: core tangibles for common physical/digital activities. In *TEI '08*, pages 209–212. ACM, 2008.

21. Wikipedia. Through-hole technology. `http://en.wikipedia.org/wiki/Through-hole_technology`.