

1 Introduction

Artists, industrial designers, and technical researchers are often interested in attaching various devices to computers for use with applications they are creating. These devices usually measure something or perform actions, and include devices such as temperature sensors, sonar distance rangers, lights or other displays, and motors.

Many sensors, motors, and other devices exist. They take a number of forms, and have very different interfaces. Most devices, however, are provided with an interface that is either too high-level or too low-level for potential users.

High-level interfaces often require a special interface card or provided software, or don't have any computer readable interface. Low-level interfaces require resistors, analog to digital converters, various small parts, or a knowledge of circuit design and construction techniques.

For example, one temperature sensor produces a certain voltage to represent the temperature, while another has eight digital outputs indicating the temperature within a range. Another plugs into the wall and displays the temperature on its own screen, while yet another is a part of an entire weather station, requiring a special computer card and the software provided by the manufacturer.

This diversity in an interface partially demonstrates the difficulty in connecting these devices to computers. Since each device is unique, even someone skilled in electronic design may have a tough time controlling them with or connecting them to a computer. If none of the methods provided is appropriate for the designer or end application, decisions must be made as to which method to use, and how to modify it for the designer's need. This may involve special IO boards or an attempt to modify the custom software. Very few devices and systems provide a solution where the hardware and software interface at the right point for someone desiring to integrate them into their application.

For many people, an interface that is too high-level or low-level prevents or delays them from accomplishing their end task. They may spend large amounts of time finding and collecting just the right interface and parts, reverse engineering an undocumented library, or building the device, when they're much more interested in using the device for its intended purpose. They often have creative, high-level concepts of what functions they'd like the device to perform. They want the device to interact with their existing programs, and are concerned with the information produced or actions performed by the end product.

Their skill generally lies in the use of the device (and its information or function), rather than its creation. Many people are interested in a solution that lies in the middle, between high-level and low-level. Sensors and active devices should be simple, dependable, and cost-effective [1], and have a uniform interface. This allows people to concentrate on building and using their entire project, rather than the detailed tasks required for each individual device.

1.1 Applications for computer control of hardware

Devices which perform actions or measure some element of the environment around them are used heavily in robotics. Common devices include motorized wheels, gripping hands, and light or distance sensors. When combined with a central controller, these devices allow robots to interact with the world around them.

An example of how these devices are combined is the Mars Pathfinder and Sojourner Rover combination. These contained several cameras and lasers for distance measurement, several temperature sensors, wind socks to measure wind characteristics, and pressure sensors and accelerometers to gather useful information during its descent [2]. This information was then relayed back to Earth, or used to help guide the rover on the surface of Mars. Robots such as Sojourner are specially designed by engineers for the functions they will perform [3].

While some robotic systems have specific constraints which require an individual engineering process, many others are purchased pre-built. Additional sensors and effectors may often be purchased and attached later [4]. This allows the robot to be used quickly in higher-level applications, including machine learning and control algorithms, such as teaching a robot to follow a line or avoid obstacles, or measuring environmental conditions.

Sensors and effectors are also used in a number of systems such as security systems, computer rooms, and other remote or environmental monitors. These systems often involve motion detectors and temperature sensors to detect problems, and alarms, sirens, warning lights, and other means of alerting the end user of problems. These systems may be purchased (as a unit or in pieces) or engineered.

In situations like Sojourner, NASA had the time, money, and expertise to design and build an entire robot system from the ground up. However, researchers usually do not want to spend the first year of a project collecting and assembling all the pieces before they can begin performing their higher-level studies. Buying pre-built products can be costly as well, and the product may not perform the correct range of functions.

For many of these applications, the end user would benefit greatly from a middle-of-the-road solution that allows device control at the right level. End users are generally concerned about the time and money involved, and usually have expertise in other fields. In these cases, a pre-built solution can be very costly, and reverse-engineering current products or designing something from the ground up requires more time, work, and in-depth electronic knowledge.

1.2 Previous work

Some work has been done in an attempt to simplify communications between a controller and various devices. There are several kits which contain various motors, gears, sensors, and connecting wires. These kits are often hobbyist kits, or are approached from a Human Computer Interaction (HCI) standpoint. Both approaches can be useful in different situations, but there are many cases where neither is implemented at the right level for the studies researchers are interested in.

Lego Mindstorms are Lego kits containing a computer module (the RCX) and a number of Technics parts [5]. Mindstorms have enjoyed a large following, and many people have designed sensors and other systems for use with it [6]. The RCX contains a small computer controller which can be programmed through infrared from a PC. RCX can only communicate directly with three sensors and three motors, limiting its capabilities.

Robotix [7] bills itself as “a motorized modular building system.” Motors, grippers, jaws, and winches are controlled using a small hand-held remote module. Commands are given on the fly, so the device does not apply higher-level ideas.

Phidgets [8, 9, 10] are devices and software designed to “abstract and package input and output devices” to hide implementation and construction details. They are similar to graphical user interface widgets and expose functionality through an API. Phidgets require a connection manager to manage devices which are online, and have a simulation mode, where the programming interface can be used without an attached device.

Though Phidgets are implemented from the HCI standpoint, their interface is bulky and specific to Visual Basic, COM, and ActiveX in Windows. Their API can be useful to some researchers, but they can be complex devices to work with.

Xerox PARC [11] has performed a number of experiments related to modular robots. These robots consist of a number of similar parts connected together, programmed to work together as a single robot.

In [12], students designed and built a small robot from various parts. The pieces were connected with a 2400 baud serial communication bus, and contained microcontrollers. One part acted as a master, sending commands to all the other modules.

OOPic [13] is a programmable PIC microprocessor that can be programmed to function as a number of objects, including hardware, processing, variable, and system objects. These objects simplify connection of the OOPic with hardware devices, and allow several OOPics to communicate with each other.

1.3 Proposed work

In my thesis, I will create a new sort of device, which I call a POD (or PEL Object Device). These devices will be smart devices, and will be capable of reporting information requested or performing tasks. To simplify their connection and computer control, I will define an environment for building and communicating with the PODs. They will contain microcontrollers, and will communicate directly with a user program or with a connection manager over a simplified USB link.

Within a POD-based system, the microcontrollers allow the devices to contain methods through which they can be controlled from the host side. A connection manager allows programs to communicate with all connected PODs over a single port. This closes the connection between the device and the programmer, allowing a simple and uniform programming interface.

All PODs will respond to several required commands. These commands include:

- Requests to identify the device type (and uniquely define it in some cases)
- Query the device for the location (such as a web URL) of drivers and API information
- Reset the device

In addition, each device type will also respond to a number of specific commands for that type, such as a temperature sensor reporting temperature, measurement type (for example F or C or K), and useful range (such as 0°-120°).

Since PODs are essentially objects, they can take advantage of the properties of object-oriented design. For example, a motor is a simple device which can understand things such as direction of spin and rotational velocity, and a powered wheel would inherit all the properties of a motor and add more, such as wheel radius and actual linear speed. A thermostat could contain all the methods of a thermometer, and add the ability to set alarms or warn when the temperature leaves a specified comfort zone.

The POD model is flexible enough that a POD could be designed to act as both a device and a controller for other PODs. This allows a hierarchy of PODs, and could simplify the control of a cluster of devices. This could be used, for example, to create a control device which manages 4 motorized wheels. The control device would abstract the lower devices, and appear as a single motorized 4-wheel movement platform. This hierarchy could be implemented in hardware or in software, and allows for a physical hierarchy similar to a logical hierarchy of software objects.

Since PODs will communicate over USB, up to 127 devices could be connected to a single USB port (using expansion hubs), removing the need for additional free ports. Low-power devices will be able to draw power from the bus, requiring no additional wiring. This also eliminates the need for special communication cards in the host computer, and allows all the devices to share a common, well-defined bus, which many computers and embedded controllers now have and can take advantage of.

USB will allow all PODs to share the same hardware interface, allowing simple attachment and communication. The software interface for all devices will be common among PODs. The POD environment will not replace the need for monolithic robots when there exist certain constraints, such as weight or size. However, in many cases, PODs can be minimized to just a few small parts, making them very small, simple, and lightweight. PODs can then be added to a robot later if necessary, allowing post-production or in-the-field flexibility.

For my project, I will build a number of PODs. The completed PODs will interact with a connection manager on a controlling computer. I will build a sample connection manager, implementing device addition and removal, enumeration, and message queuing. My connection manager will multiplex the various connected PODs over a single port.

I will use the PODs in building two very different end products, demonstrating their usefulness and simplicity in diverse applications. These end products will be a robot and a server-room monitoring station. The PODs I intend to build are:

- Temperature sensor

- Audible alarm
- Motorized wheels
- Digital compass
- Light sensor
- Small lights
- Power sensor
- Distance sensor

2 Thesis Statement

By building intelligence into devices using an object-oriented communication and software interface, a number of devices can be attached to a computer with little effort. This solution will allow people to integrate these smart devices into current or new programs without requiring a detailed knowledge of electronics.

3 Methods

Design and construct a robot and server room monitoring station using the PODs mentioned in section 1.3. The 2 systems will have the following basic characteristics:

The Robot:

- Two individually controllable motors (left & right) for speed and direction
- A digital compass
- A front distance sensor
- Several light sensors to search out light or dark areas, or to determine light levels
- Headlights which could be used in the dark
- A siren may be used to alert people of the robot's presence or when the robot is backing up

The server-room monitor:

- Temperature sensor that can warn of temperature problems
- A power sensor that can cause alerts when the power goes out
- A light sensor and distance sensor that can detect whether someone is in the room or at the console
- An alarm or siren that can be used as another reminder that a problem is occurring
- Small lights can turn on when the power goes out

Design the POD system, identifying methods all devices must respond to, and device-specific methods. Once the high level interfaces are identified, the low level message passing protocol will be defined. This will include the USB interface details for both the PODs and the connection manager, as well as the interface seen by the user.

Build and program the PODs, as well as write the simple host-side connection manager.

Build the end products, and write some programs demonstrating their use. This will include showing that the robot can provide useful sensor information, and can be controlled either by the program or remotely. The server room monitor will be controlled by software which will record the temperature, create graphs of temperature over time, and send an email and sound the alarm if the temperature gets too high or there is a power failure.

4 Contribution to Computer Science

This thesis defines a system for attaching hardware devices to a computer and allows people to write programs to control these devices. It includes both hardware and software, and simplifies the interface between the two, since physical devices appear as software objects. The attached devices can be small, smart, and cost-effective.

5 Delimitations of the Thesis

- I will not build a virtual construction interface or simulation environment
- I will not be performing a usability survey

6 Thesis Outline

Chapter 1	Introduction and previous work	8 pages
Chapter 2	Thesis statement and POD environment	10 pages
Chapter 3	One subsection for each of POD created, including: Block diagrams POD-specific methods	~20 pages
Chapter 4	Case study 1 - The robot	5 pages
Chapter 5	Case study 2 - The server-room monitor	5 pages
Chapter 6	Conclusions and future work	5 pages
Appendices	Schematics, circuit board artwork, etc.	20 pages

7 Thesis Schedule

- Continue to collect additional information, sources, and parts (April-June, ongoing)
- Design OO interfaces for chosen devices (April)
- Design schematics for chosen devices (May-June)
- Build devices and program microcontrollers (June-July)
- Write connection manager (July)
- Attach devices to several systems (August)

- Writing the thesis (August-September)

8 Annotated Bibliography

1. Sensing Problems Solved. Industrial Technology. September 2001 - <http://www.industrialtechnology.co.uk/2001/sep/coverstory.html> - This article describes a new product, and discusses the need for inexpensive, simple devices.
2. Mars Pathfinder Instrument Descriptions - http://mars.jpl.nasa.gov/MPF/mpf/sci_desc.html. This page describes some of the instruments that were located on the Mars Pathfinder. It describes how they were used to gather useful information about Mars and the landing site, and how some of this information was used to facilitate maneuvering the rover.
3. Foster-Miller Robotics Homepage - <http://www.foster-miller.com/robotfr.htm>. Foster-Miller makes some robots which perform highly specialized functions, such as bomb disposal or high-voltage power-line maintenance.
4. ActivMedia Robotics - <http://www.activrobots.com/ACCESSORIES/index.html>. ActivMedia makes a variety of general purpose robots, as well as accessories which can be attached to them.
5. Lego Mindstorms Internals - <http://www.crynwr.com/lego-robotics/>. This page describes many of the low-level details for Lego Mindstorms.
6. <http://www.plazaeearth.com/usr/gasper/lego.htm> - This page describes and gives detailed plans on many devices which can be created and interfaced with the Lego RCX.
7. Homepage for Robotix - <http://www.roboticsandthings.com/robotix/robotix.html>. This page shows the various collections of motors and other arms that can be used with Robotix systems.
8. Phidgets: Incorporating Physical Devices into the Interface - <http://www.cpsc.ucalgary.ca/group/lab/papers/2001/01-Phidgets-UbicomWorkshop/phidgets-ubicomworkshop.pdf>. This paper describes how phidgets can implement physical objects.
9. Phidgets: Easy Development of Physical Interfaces through Physical Widgets - <http://www.cpsc.ucalgary.ca/group/lab/papers/2001/01-Phidgets.UIST/phidgets-uist-2001.pdf>. This paper introduces phidgets, and describes the the implementation.
10. The Phidget Architecture: Rapid Development of Physical User Interfaces. <http://www.cpsc.ucalgary.ca/group/lab/papers/2001/01-Phidgets.Workshop/phidgets.report2001-681-04.pdf>. This paper briefly describes the phidgets concept and general architecture.

11. Modular robotics: Publications - <http://www.parc.xerox.com/spl/projects/modrobots/publications/>. This page has links to a number of papers from Xerox related to modular robotics.
12. Kapil Kedia, Vincent Marshall, James Ostrowski, et al: An Architecture for a Modular Robot. NSF Summer Undergraduate Fellowship in Sensor Technologies. <http://www.ee.upenn.edu/~sunfest/pastProjects/Papers99/kapil.PDF>. This paper describes a project performed by a number of undergraduate students relating to a modular robot.
13. Object Oriented PIC - <http://www.oopic.com>. The OOPic is a board that contains a PIC microcontroller and can be programmed to act as one of a number of different devices.

9 Artifacts

- List of 20 devices which could be built/designed as PODs, as well as some possible uses
- General POD methods defined
- Schematics and pictures of the PODs actually designed and built
- Specific POD methods for the devices built
- The working PODs (the end-devices may be disassembled)
- C code for the connection manager, and code programmed into the POD microcontrollers
- Web-based documentation

10 Signatures

J. Kelly Flanagan, Committee Chair

#####, Committee Member

Michael Goodrich, Committee Member

David Embley, Graduate Coordinator