

# Using a Web Service, Mobile Device and Low-Cost Robot To Teach Computer Science and Engineering

Abraham L. Howell and David W. Sersen  
{ [abe@aboitcs.com](mailto:abe@aboitcs.com), [dave@cts-c.com](mailto:dave@cts-c.com) }

Department of Computer Science  
Binghamton University (SUNY)

## Abstract

Currently, robots are being used as teaching tools and can be found in K-12 and college classrooms. The primary appeal associated with employing robots is that they tend to capture the attention of the target audience and thereby facilitate the learning process. While there are numerous robotic kits on the market today, few can be used to teach both high-level concepts in computer science and engineering as well as low-level concepts in a K-12 environment and those that are capable, are more expensive. In this paper we propose an affordable setup of a basic software and hardware framework to accomplish this interaction. Specifically, the overall system architecture, low-cost robot, Web Service and mobile device software will be the topics of interest in this paper.

## Introduction

Applied technology offers a significant catalyst to education. Robots can be a great way to stimulate and boost the interest of students. There are a plethora of highly capable robots available for use in education, but the majority of these are specifically designed for research and are therefore expensive. These prohibitive costs tend to keep teachers from involving

robots and other technologies in their curriculum. A collegiate level educator looking to incorporate a robot into one of their computer science courses will more than likely want to have students code fairly complex applications and also be able to involve cutting edge technologies. On the other hand, an educator at the K-12 level will have substantially lower requirements and for the most part will utilize the robot along with software to teach an elementary concept in science or math. In this paper we set forth the foundation for integrating a low-cost Bluetooth<sup>1</sup> capable robot, Pocket PC mobile device and a Web Service. We feel that this combination makes for a flexible and robust system that is able to satisfy the varying needs of educators at all levels.

### **Overall System Architecture**

At the heart of the system is a web service, communicating with a handheld Pocket PC, which in turn communicates with a robot. Microsoft Visual C# .Net was used to create the web service and Pocket PC software used in this application. C# was chosen because of the ease of development and deployment.<sup>2</sup> Several key functions are provided by the web service: compilation, database caching and searching of compiled programs. Code compilation is an issue that must be dealt with no matter which type of microcontroller is used on the robot.

Typically, the end-user will create a program for the microcontroller using a desktop or notebook and compile with either a proprietary/licensed or free compiler. For the most part, microcontroller compilers are designed specifically for the desktop/notebook computer architecture and operating system. However, in this application we want to leverage a handheld Pocket PC and extend the mobility of the entire system. Additionally, by integrating a mobile

device we are expanding the capabilities of the system. The Web Service will provide the link between the handheld and microcontroller compiler, thereby allowing for the creation of programs on the handheld and compilation by the web service. The Web Service will compute other intense tasks such as storing user created programs in a database. The user can retrieve previously stored programs in future sessions and then modify or implement as needed. All users will also be able to search for any compiled program in the database from their mobile device using the Web Service. After they download a program, the user can then upload it to their robot to execute. This is exceptionally useful for classroom demonstrations as well as many other applications aside from education. Other applications will be discussed later.

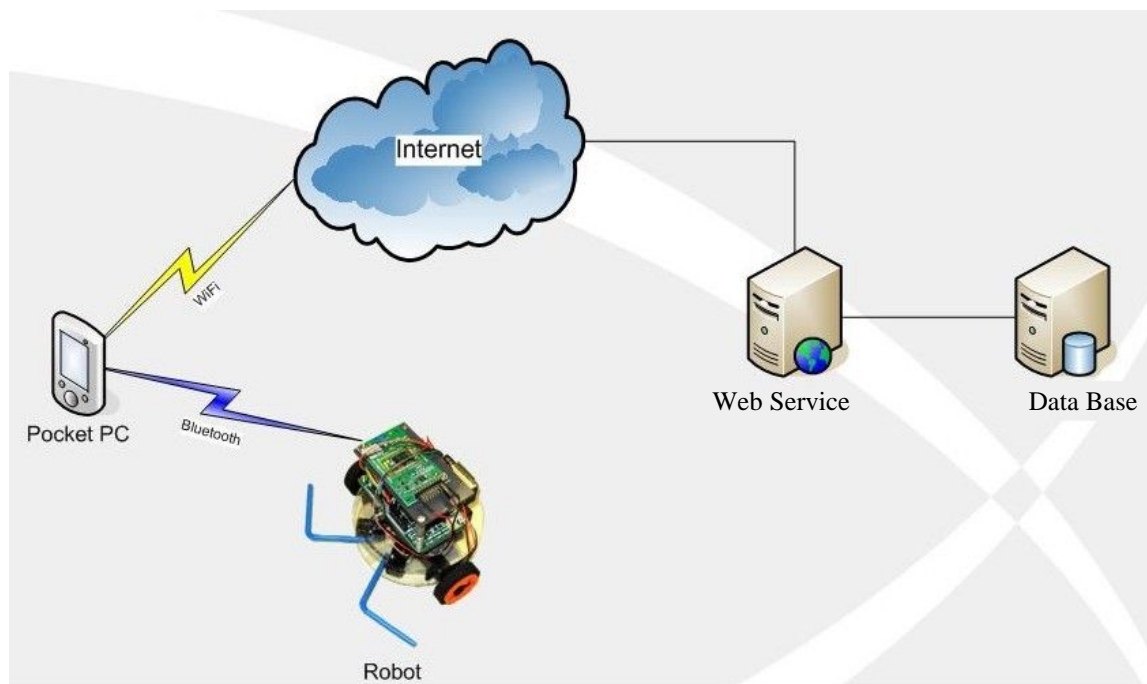


Figure 1. Overall System Architecture.

## Low Cost Robot

Built from basic robot components such as modified hobby servos, foam wheels, a programmable microcontroller, infrared distance sensor and infrared temperature sensor the NewCDBot robot is cost efficient, capable and expandable<sup>3</sup>. A differential drive train is utilized and provides sufficient locomotion for the robot. Currently, the differential drive train on the robot does not provide any type of distance or velocity feedback. However, recent advances in electronic kits for the robotics enthusiasts have made available a relatively low cost wheel encoder kit that can be added to the modified hobby servos of the robot<sup>4</sup>. Wheel encoders can provide enough feedback such that the end user can perform complicated distance and velocity control via the onboard microcontroller. An OOPic II+ microcontroller provides the computational power and behavior for the robot<sup>5</sup>. The basis for the OOPic II+ is the PIC16F877, which is manufactured and distributed by Microchip Inc. While the PIC16F877 chip provides access to hardware like I2C, SPI, 10-bit ADC and UART, the task of programming such a chip in assembly may not be desirable depending upon the application.

One may however go the route of purchasing a C compiler and ease the aspect of programming, but may incur substantial cost associated with such compilers. The OOPic II+ provides user access to the chip's plentiful hardware and also allows for coding in C, Basic, or Java. A unique firmware is pre-programmed into the OOPic's PIC16F877 chip and allows users to create/compile programs in C, Basic, or Java and then finally download the compiled program to an onboard EEPROM. Once a program is loaded into EEPROM the firmware will begin interpreting and executing the code. While this method of code interpretation obviously slows down code execution, it is more than adequate for our application.

An EB500 Bluetooth transceiver module provides the NewCDBot with bi-directional communication capability between itself and the Pocket PC. Even though the focus of this paper is on Bluetooth communication between the robot and a Pocket PC, it can be expanded to include communication between the robot and a Bluetooth equipped mobile device including notebooks or classroom equipment such as a desktop or workstation computer. Figure 2 is a detailed picture of the robot and Pocket PC handheld computer along with individual component labels.

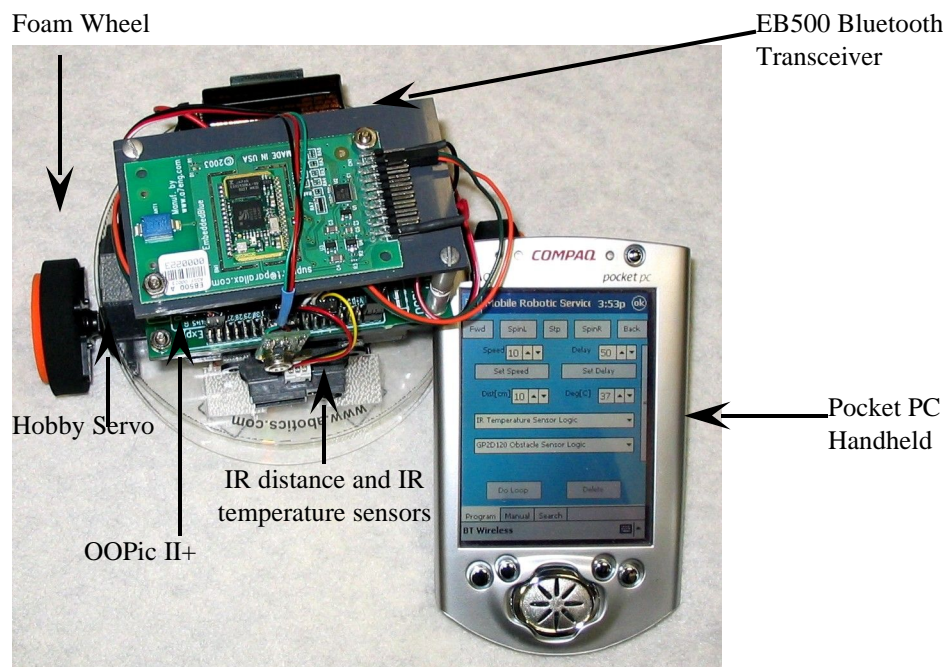


Figure 2. Low cost NewCDBot Robot with Pocket PC Handheld.

## Mobile Device Software

The current version of our Pocket PC software is designed for the K-12 audience, but can and will be expanded to service the needs of educators at all levels of instruction. As previously mentioned, the software has been specifically developed for the Pocket PC version of handheld computers and is not compatible with PDAs running the PalmOS. Once properly installed on the handheld, users will be able to interact with several screens to perform the following necessary tasks: creating, compiling, uploading code, storing programs locally, retrieving programs from the web service's database and manually controlling the robot.



Figure 3: Pocket PC software screen shot of Main



Figure 4: Code slide screen.

From the Main screen users can create their program by simply tapping on buttons via drop-down menus and numeric up/down controls (Figure 3). To review the progress of their generated code, users can tap the long button on the right side of the Main screen and view the Code Panel (Figure 4). The Code Panel displays the current program and also allows the user to add a "Do Loop" or delete unwanted code. Since our initial target is the K-12 student we made the task of generating code as easy as possible and tried to minimize the generation of errors. To reduce syntax errors, our software will find and delete the corresponding "If/End If" or "Do Loop" statement when the user deletes a line of code.

After creating a program, students will need to compile their code using the Web Service. Compiling the code is simply a matter of tapping the "BT Wireless" menu in the lower left hand part of the screen and then selecting "Compile Program". Next the user will be presented with a screen that will give them the option to name and describe their program, as well as the choice to store it in the Web Service's database or locally on the handheld. Once the proper selections are made the program is sent to the Web Service where it will be compiled. If compilation is successful the corresponding hex code will be returned to the handheld (and if selected, it will simultaneously be sent to the database to be cached). At which time, the user can choose to upload their new program to the robot using Bluetooth. The robot will begin executing the new program once the upload is complete.



Figure 5. Manual Screen shot.

The Manual screen can be accessed by tapping the appropriate tab (Figure 5). It allows the user to place the robot into manual mode whereby they can control all functions using the handheld. Before attempting to control the robot manually the necessary program must be uploaded to the robot by tapping the “Place Robot into Manual Mode” button. Once in manual mode the user can drive the robot around by using the four motion buttons: forward, reverse, spin right and spin left. Each motion command will be executed for one second and after that the robot will be stopped. Users can also request the current sensor reading for available sensors. In this case we have a single infrared distance sensor and an infrared temperature sensor. Additionally, selections can be made for either English or Metric units with regards to the sensor readings.

Users can retrieve previously created programs by using the “Search” screen (Figure 6). Within the search screen there are several methods for retrieving programs. First, the user can request the most accessed programs and then select a program from this list. If the program name is already known then a search can be performed for that specific program. Finally, users have the option of browsing through previous programs saved locally on the handheld. This method is beneficial for when the Web Service may be down or off-line program modification is desirable. Once the appropriate program is selected users must decide whether they simply want the source-code or the compiled hex code. By selecting the source-code users can modify their programs, but by selecting (compiled) hex code they will only be able to upload the program directly to the robot for execution.



Figure 6. Local Search screen shot.

## **Applications**

A scaleable architecture is inherent by the concept of having a simple user interface with the vast power of a remote server to do CPU intensive processing and data caching. Our project uses the Web Service on the server to compile our program instead of consuming the limited Pocket PC battery to power the extra CPU cycles. Power conservation is yet another benefit of our model. Likewise, our model is beneficial to other applications other than education.

Home appliances, such as a robotic vacuum cleaner with programmable functions for specific areas (floors, stairs & under tables), would allow a user to either write or download a popular program from the online database to perform specialized tasks. Like the educational aides, the home user would have the option of using an inherent program, or creating their own patterns of work.

Furthermore, any product of this nature that would need to be kept up to date can benefit by having a central source (in our case the compiler). Maintaining a core location reduces redundancy and system requirements for users. Thus, alleviating having to issue patches, upgrades or reinstalls to all clients.

Another engaging application, especially to the military is the capability to deploy to multiple devices. Teams can manage their programs through their own client and share their features with other teams as add-on or upgrade modules. Utilizing the database to store compiled programs, the breadth of available programs and program maturity will foster teamwork and superior outcomes.

## **Conclusion**

In this paper we have presented the hardware and software foundation for an educational robot system that can be utilized to teach concepts of varying difficulty throughout science and engineering education. While our primary focus was on K-12 education it is obvious that the system can be expanded to incorporate high-level concepts found in college level computer science and engineering courses. We believe that incorporating math and science by including a real-life problem and allowing students to design a solution is one that encourages knowledge and imagination. Our system can be a means to teach concepts including software and hardware integration whilst providing students with a basic foundation upon which to build and expand. Students will also gain tremendous experience in working with such technologies as Bluetooth and mobile device computers. Although not mentioned previously, the total cost associated with the robot used was approximately \$250, which excludes the price of a Pocket PC handheld. In the future all source code associated with this project will be made available.

## Bibliography

1. <http://www.bluetooth.org>
2. Williams, Mickey, Microsoft Visual C# .NET, Microsoft Press, 2002.
3. [http://www.abotics.com/buildit\\_yourself.htm](http://www.abotics.com/buildit_yourself.htm)
4. <http://www.nubotics.com>
5. <http://www.oopic.com>