

# Smart Tupperware<sup>1</sup>: An Example of Bluetooth Wireless Sensor Networks for Human Assistive Mechatronic Systems

Richard M. Voyles

Computer Engineering Department  
University of Denver, Denver, CO  
rvoyles@du.edu

Jaewook Bae

Electrical Engineering Department  
University of Minnesota, Minneapolis, MN  
baex0021@umn.edu

*Abstract*— This paper describes Smart Tupperware, a system for automatically maintaining inventory status of kitchen foodstuffs for the purpose of updating the user's shopping list on a PDA (personal digital assistant). Commercial kitchen containers were instrumented with microcontrollers and a variety of sensors to determine the volume, weight, and optical properties of dry goods commonly stored in the kitchen. The containers periodically “wake-up” and wirelessly communicate to a base station PC regarding their inventory status. The base station maintains a web page for each of the containers, creating a distributed database of foodstuff properties. The purpose of this distributed database is to eliminate the need for human annotation of each inventory item, making the inventory system totally transparent to the user. Once the container has determined its contents by matching with the distributed database, the fill status is automatically updated on the user's PDA via WiFi. A custom application on the PDA organizes the shopping list for both inventoried and non-inventoried items. An optional barcode scanner on the PDA facilitates greater information gathering from the UPC code at point of sale.

## I. INTRODUCTION

IMAGINE a trip to the store on the way home from work. You stop to pick up milk because you noticed at breakfast that it is low. While at the store, it dawns on you that you might be low on cereal and you'd like to avoid another trip tomorrow. But you don't remember...

Enter Smart Tupperware. Smart Tupperware (no endorsement by Tupperware Corp. is implied) is a system for automatically maintaining kitchen foodstuffs inventory and integrating it with a shopping list application on your PDA. Through sensorized kitchen containers, the system measures the fill status of the containers and maintains a distributed database that is uploaded to the PDA.

The key difficulty with this system is the user interface. It is unreasonable to expect users to plug a keyboard into their kitchen containers to enter descriptive information. In fact, the best user interface for the container contents is no interface at all; the containers should just “know” what they contain. Toward that end, the Smart Tupperware system includes a base station PC with an internet connection. The base station posts web pages for each

<sup>1</sup> Tupperware is a registered trademark of the Tupperware Corp. No endorsement is implied.

container with distinguishing sensor data including time-of-day usage. Based on this information, newly-filled smart containers can query one another to attempt to infer their own contents with no user input.



(a.)



(b.)

Figure 1a: Before: “Hmmm. I wonder if I'm out of Cheerios...” Figure 1b: After: “My Smart Tupperware tells me I only have two servings left. Time to stock up!”

The described application is a “toy” project initiated as an undergraduate senior design project that has grown into a larger effort in home and elderly assistance. The bulk of this paper describes the preliminary undergraduate projects including the mechatronic systems developed and the software applications designed to engage the user. The later sections describe extensions of the work into the definition of an architecture to include disparate mechatronic devices for sensing and actuation of the environment to assist users. These extensions include improved sensing within the containers themselves.

This is an example of symbiotic robotic systems [14] – systems that enable greater human assistance without individually achieving full-scale autonomy.

## II. PROJECT DESIGN

### A. Software

The project software consists of three parts: user application, embedded code, and base station code. The user application implements a shopping list that provides a way to select purchased and not-purchased foods on a list, both manually and by an automatic update. The shopping list was developed using embedded Visual Tools for the Pocket PC 2002. The application offered a simple five-button interface that allowed for easy customer operation.



Figure 2: Shopping list application. For user convenience, the shopping list can maintain both items stored in Smart Tupperware (of which it has data) and other items the user may add, about which it has no data and can make no recommendation.

The user can enter items to buy on the list either from a database of known items, or by typing in an unknown item. Once an item is typed in, it becomes part of the known database, even if it is not inventoried by the automated system. Items that are typed in manually, become part of the known database and can be retrieved later. Items that the containers determine are running low are automatically entered onto the shopping list.

In the current implementation, the base station software suite does most of the analysis and interpretation of the sensor data. This suite of applications includes both a “researcher’s interface” and the normal operational software for the system. The researcher’s interface provides a bootstrapping mechanism for inputting new container data directly into the database. No intelligence

is yet implemented that can infer unknown contents directly from sensor data. Only clustering is implemented, so the database (including distributed information on the web) has to be, at least, partially populated before any user operation. Perl was used to implement configurable screen scraping of the web database for easy implementation of future intelligent algorithms for inferring content.

### B. Sensors

The role of the sensors was to examine the food stored in the containers. By using a variety of sensors the two main goals -- identifying the amount and type of food -- were accomplished based on a small benchmark set of food items. The benchmark set of food items consisted only of five different types of dry breakfast cereal: Fruit Loops, Corn Pops, Frosted Flakes, Cocoa Puffs, and Raisin Bran. This initial set was chosen to simplify the problem in order to focus on system architecture and because cereals are a common food item stored in third-party containers. Greater sophistication to achieve better food discrimination can always be added, as needed.

Numerous types of sensors were examined in order to determine what combination was best for achieving the two goals. These included sensors for determining weight, volume, appearance, color, optical transmission, and humidity. The sensors were tested against the benchmark cereals for discrimination ability.

The first sensor examined was a digital color camera since this was the easiest method of determining the type of cereal in the container. The MATLAB Image Processing Toolbox was used to analyze pictures of each cereal. MATLAB showed that each cereal had a different RGB histogram as seen in Figure 3. Therefore by using the camera it was possible to distinguish cereal simply by analyzing the photo. Miniature color cameras used in the toy industry were readily available for an affordable price so it seemed the solution of discriminating food was found, except that two key issues came up. The problems were focusing and an external light source. A strong light source was needed for the camera to perceive anything besides darkness inside the container. This meant power had to be allotted for this device in the already stringent power budget. The placement of the light was also a concern because it had an effect on focusing.

The solution to the above problems, and a reduction in cost, came in the form of a custom-designed, single-pixel camera. A single cadmium sulfide photocell was placed in the bottom of the container (the darkest area) with four different LEDs for illumination: red, green, blue, and infrared. By individually illuminating the LEDs, the single photocell can sequentially measure the reflectance of the cereal at the different wavelengths and with substantially lower power consumption.

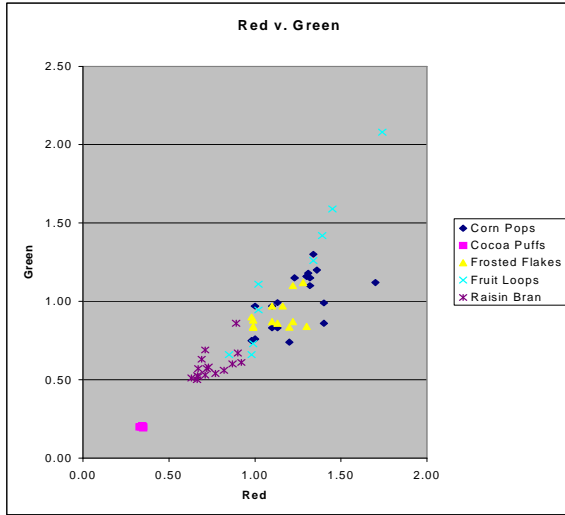


Figure 3: Comparison of the red and green histogram peaks for many samples of the five cereals.

To measure food density, both volume and weight are required. A matrix of IR LEDs were positioned on the large side of the containers, with a corresponding photodiode matrix on the other side to measure volume. This produced a matrix of light beam sensors indicating presence or absence of cereal. Since cereal has a tendency to shift to the sides after being poured, the majority of the components were oriented near the middle and sides of the container. A total of eight light beam pairs were used to quantize the volume.

Mass sensing was done by strain gages placed on the bottom of the containers to measure the deflection like a diaphragm. Strain gages are amplified to detect minute deflections of the floor. Larger forces applied to the floor of the container provide larger outputs from the amplifier. To determine where the gages would be placed, equations of diaphragm strain were consulted. For circular fixed diaphragms, a zero in strain occurs about half-way along the radius. Figure 4 indicates strain maxima occur at the edges and at the center of the diaphragm.

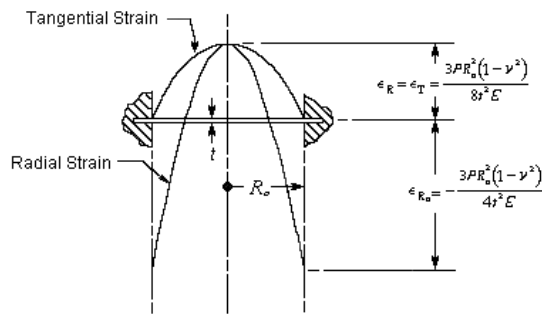


Figure 4: Membrane deflection (from efunda.com)

With all the sensors in place, a benchmark was created.

The RGB reflectance of the cereal was tested by pouring the cereal into the instrumented container and reading its red, green, and blue value individually, with the lights off to simulate a cupboard location. The container was then emptied so the above process could be repeated for 20 readings. The red and green LEDs were powered at 5mA, while the blue used 4mA, due to differences in luminosity. The result was each cereal having different RGB values as predicted by MATLAB.

Determining density turned out to be more difficult. Testing of the weight sensor revealed non-monotonic behavior of the sensor output with increasing cereal weight. The reason for this is that dry cereal does not lay uniformly on the container floor. Instead, it distributes in a mound, rendering the weight sensor useless, initially.

### C. Microcontroller

The data gathering and analysis is handled by the microcontroller. The requirements of the microcontroller were to gather sensor data and store it until it was ready for communication to the base station. Possible microcontrollers were examined based on several factors: power consumption, price, size, abundance of I/O ports, multiple analog-to-digital (A/D) conversion and availability. Another vital feature was the ability to place the microcontroller in sleep mode in order to save power.

The search for a microcontroller that fit those criteria led to three primary choices: PIC, Motorola MC68HC12, and the Atmel ATmega. The main issue confronting the microcontroller was that it needed to be flexible for the research phase of the project. Based on all the restraints the Atmel ATmega128L was chosen. The Atmel ATmega128L was a fairly low power microcontroller that affords large amounts of flexibility for I/O and A/D.

The primary tasks of the microcontroller is powering on and off sensors, performing analog to digital conversions, signal conditioning, and data transmission. The microcontroller will activate the sensor and communication system several times per day during its active mode and then go back to sleep. The microcontroller consumes around 1.3mWhr in sleep mode over a 24-hour period. In active mode the peak current jumps to 17mA, but this time is very limited.

### D. Wireless Communications

The information from the Smart Tupperware is transmitted using wireless communication. In choosing a wireless communication protocol there were several possibilities explored: AM single-channel, FM single-channel, Bluetooth, and 802.11b. The criteria for choosing a wireless option included: power, cost and size, with sufficient range to transmit over a kitchen distance without interference.

802.11b was quickly rejected due to power and complexity concerns. Bluetooth seemed an ideal fit, but

both Bluetooth and AM transceivers were included in the circuit design assuming the AM transceiver would act as a simpler fallback medium if the Bluetooth protocol was too complex. The alpha prototype used the AM link while subsequent implementations used Bluetooth. A radio transceiver chip from RF Monolithics was used to implement the on-off-keyed AM link. It operates essentially like a serial port without wires. Infineon supplied the Bluetooth chips.

Data packet transmissions are time multiplexed and initiated by the container. The packet contains a unique container identifier followed by the red, green, blue, and infrared values, weight, volume, and a checksum. With the AM transceiver, data was transmitted three times to correct errors.

### E. Power

Power management is a key aspect of this system as it can have a profound impact on user acceptance. In a battery-powered system, the power draw determines how often batteries must be changed. An alternative is to scavenge power from the environment to run the container, and total power consumption determines the feasibility of this option. Scenarios were evaluated based on number of polling cycles per day. (A polling cycle includes wake-up, reading the sensors, transmitting the data, and going back to sleep.) Ideally, the system must be able to distinguish at what times of the day its contents are accessed. This provides additional data for discrimination between food type. For example, if Cheerios and macaroni had similar color and density properties, a notion of what time of day the product is used could provide a likely discriminator. However, this requires polling several times per day. Therefore, the scenarios examined included polling once a day, twice a day and three times a day. Figure 5 shows one day's energy distribution with only one polling cycle. One polling cycle per day yields a total power figure of 1.58 mWhr for a 24-hour period. Extrapolating to 30 days yields power consumption of 47.5 mWhr per month.

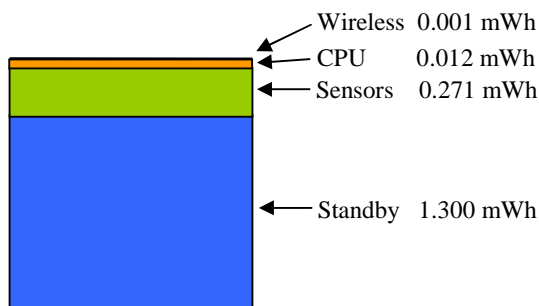


Figure 5: Energy distribution of one polling cycle per day

Polling two times per day yields the following energy distribution as shown in Figure 6.

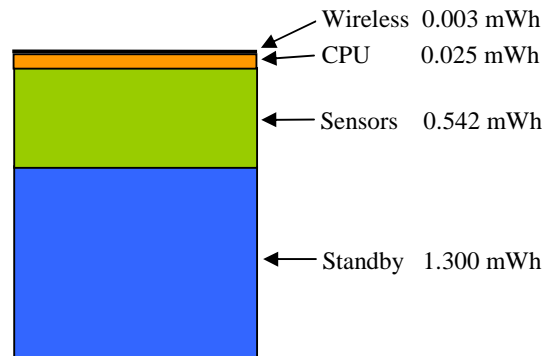


Figure 6: Energy distribution of two cycles per day.

Total power consumption with two readings a day is 1.87 mWhr. Again, extrapolating to 30 days yields 56.1 mWhr per month. Polling three times per day we would yield the distribution shown in Figure 7. At this rate, the power consumption in one day is 2.15 mWhr. One month equates to 64.6 mWhr.

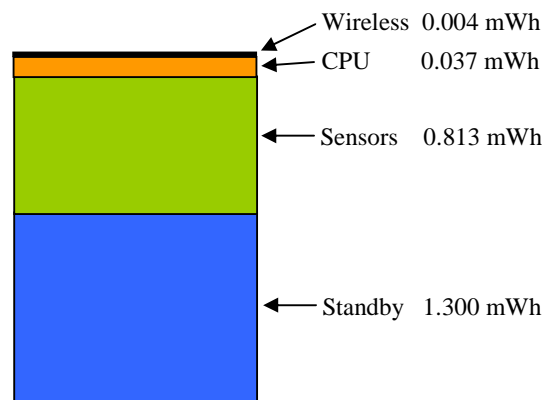


Figure 7: Energy distribution of three cycles per day.

With these figures one can estimate life from various power configurations. A Panasonic CR2477 button cell battery has a nominal voltage of 3.0 volts and a capacity of 950mAh. It is also quite small with a diameter of 24mm and a thickness of 7mm. Since button cells have negligible self-discharge rates, this cell could theoretically operate a single container polling three times per day for nearly four years! Realistically, the battery must be derated for the minimum voltage of the CPU and radio chips and also for the bursty draw of the circuit. The battery specifications are based on a uniform current draw of 0.4 mA. While the average current draw is far less than that over a 24-hour period, it bursts up to tens of milliamps for a few seconds per polling cycle. This will likely further reduce the available ampacity, but lifetimes of three years

are not unimaginable. In fact, the use of a parallel power capacitor could completely remove the bursts seen by the battery.

Other power options that were explored were a Lithium Ion Rechargeable, AAA household batteries, and an inductive charging method. The inductive charger was ruled out due to its complexity and the fact the regular users do not have outlets in their cabinets.

Three AAA battery provide 4.5 volts with a capacity of 900 mWhr. Considering a 90 percent efficient voltage regulator with three polling cycles per day yields an estimate of about one year of operation. An added benefit of this power source is availability. Lithium-ion rechargeables were considered, but the cost and the logistics of recharging a large number of containers seemed prohibitive.

### III. RESEARCH EXTENSIONS

#### A. Weight Sensors

During testing, a critical flaw was discovered in the weight sensor design, as mentioned. The sensor design was modeled after diaphragm pressure sensors. But these sensors are subjected to uniform pressure distributions and do not work given varying loads. Liquids would provide a good uniform pressure distribution on the floor of the container, but dry goods do not. Finite element analysis was employed to study the effects of various physical distributions of dry goods on strain in the floor of the container.

To study the diaphragm, a canonical set of dry good distributions was developed and simulated. In Figure 8, the results of a triangular mass distribution (top) and trapezoidal mass distribution (bottom) are illustrated. Rectangular distributions were also examined. From this set of strain data, repeated for each distribution with different masses, an optimal distribution of  $n$  strain gages was determined for small numbers,  $n$ , to minimize the error across all canonical measurements. We examined values of  $n$  from 1 to 6. As expected, as  $n$  increased, the accuracy of the measurement increased, but with diminishing gain. We chose  $n = 2$  to keep the amount of physical post-processing of the containers themselves to a minimum. Adhering strain gages to polypropylene can be problematic.

#### B. Bluetooth Networking

As mentioned, Bluetooth wireless communications were thought to be appropriate for the kitchen container network, but the protocol does not allow for multi-hop communications. For the Smart Tupperware application, the configuration of devices can be constrained to single-hop links, but the broader goals of human assistance across disparate mechatronic systems requires a more agile networking scheme. ZigBee networks have become

popular in wireless sensor networks, but they do not have sufficient bandwidth to support real-time image transfer required by some assistive applications.

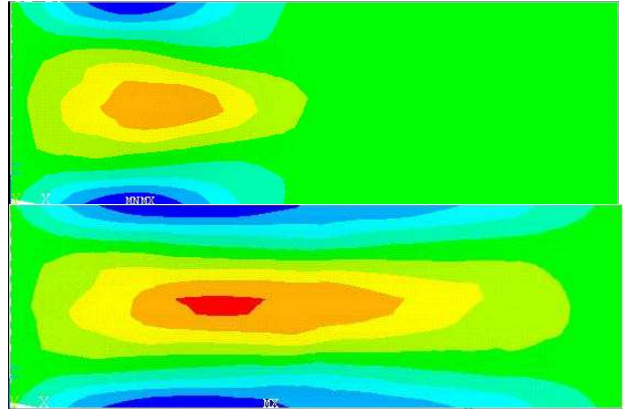


Figure 8: Example finite element modeling of diaphragm strain resulting from uneven mass distributions.

We have augmented the Smart Tupperware microcontrollers with Bluetooth radios and a novel multi-hop hybrid routing scheme described in [4]. This routing protocol combines the low-overhead benefits of proactive cluster-head gateway switch routing [6][7] with the latency benefits of reactive route discovery [8][9] in the case of node failure.

This networking provides the backbone for an architecture incorporating sensor nodes, robotic nodes, and data nodes into a benign home environment. Sensor nodes can include embedded devices such as the Smart Tupperware described here, or more sophisticated sensing and processing nodes such as for tracking human activities [5][10][11].

As far as robotic devices are concerned, we are investigating Gesture Based Programming [12][13] to provide robotic kitchen assistants for the elderly.

The architecture we employ is an extension of the Port-Based Object model of the Chimera real-time operating system [15]. Port-Based Objects are port automata (software entities that only operate on input ports, output ports, and resource ports) with methods for real-time instantiation, synchronization, and control. They provide an excellent framework for building real-time mechatronic entities such as Smart Tupperware containers or robots [3]. However, the framework does not lend itself as well to the fluid communication conduits and data associations of distributed networks, particularly those that involve human interaction.

The port-based objects implemented for Smart Tupperware are rather trivial, as illustrated in Figure 9. The wake-up module manages sleep mode and also triggers the data processing module through resource ports. More complex examples of port-based objects for

robot control can be found in [12].

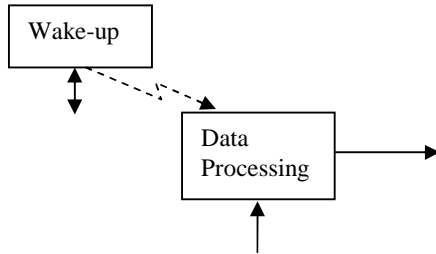


Figure 9: The basic port-based object driving a container. Input ports are on the left, output ports on the right, and resource ports on the bottom.

To implement the event-based networking manager, a special software agent was developed to implement finite state machines and manage the network protocol stacks. This separation of event flow and data flow is similar to real-time data flow diagrams. Illustrated in Figure 10, we use the convention of real-time data flow diagrams to indicate data flow with solid lines and control flow with dotted lines. For the control element, data flow now becomes a resource, while control actions represent inputs and outputs.

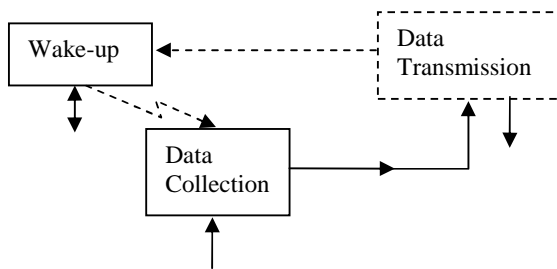


Figure 10: The basic port-based object model augmented with an event-based control object for networking and synchronization.

#### IV. DISCUSSION

This paper describes the implementation of Smart Tupperware using Bluetooth wireless sensor networks. As an application example, no specific user test results are presented. The only user tests were done by the students that designed the interface, so those results are biased. Furthermore, the benchmark set of cereal was chosen to make the undergraduate component of the project simple.

Functional testing was performed and the individual capabilities were demonstrated adequately, after further study of the weight sensor. The system gathered container data, relayed it to the base station PC, and updated both the PDA application and container web pages.

#### V. ACKNOWLEDGEMENTS

The authors would like to thank Tupperware Corp for

the donation of kitchen containers in support of the senior design projects. Initial programming and design work was carried out by Ryan Goss, Sitha Chhum, Hendry Gouwanda, George Kaniamos, Andrew Engebretson, Angela Brockman, Chris Sanny, Hung Le, and Mike Wenger. Bluetooth network development was funded by the NSF Safety, Security and Rescue Research Center.

#### REFERENCES

- [1] R. A. Gonçalves, P.A. Moraes, J. M. P. Cardoso, D. F. Wolf, M. M. Fernandes, R. A. F. Romero, E. Marques, "ARCHITECT-R: A System for Reconfigurable Robots Design", in ACM Symposium on Applied Computing (SAC 2003), March 9-12, Melbourne, Florida, pp. 679-683.
- [2] B.H. Kim, C. D'Souza, R.M. Voyles, J. Hesch, S. Roumeliotis, "A Reconfigurable Computing Platform for Plume Tracking with Mobile Sensor Networks", Proceedings of the 2006 SPIE Defense and Security Symposium, Orlando, FL, April, 2006.
- [3] Richard M. Voyles, "TerminatorBot: A Robot with Dual-Use Arms for Manipulation and Locomotion", in *Proceedings of the 2000 IEEE International Conference on Robotics and Automation*, San Francisco, CA, April 2000, pp.61-66.
- [4] J. Bae and R. Voyles, "Wireless Video Sensor Networks Over Bluetooth for a Team of Urban Search and Rescue Robots," in *Proceedings of the 2006 International Conference on Wireless Networks*, Las Vegas, NV.
- [5] E. Ribnick, S. Atev, O. Masoud, N. Papanikolopoulos, and R. Voyles, "Real Time Detection of Camera Tampering," in *Proc. of the 2006 Intl. Conf. on Advanced Video and Signal Based Surveillance*, Sydney, NSW, Australia.
- [6] E. M. Royer and C. K. Toh, "A review of current routing protocols for ad hoc mobile wireless networks," pp. 46-55, Apr. 1999.
- [7] C. C. Chiang, "Routing in clustered multihop, mobile wireless networks with fading channel," in *Proceedings of IEEE SICON*, Apr. 1997, pp. 197-211.
- [8] C. E. Perkins and E. M. Royer, "Ad-hoc on-demand distance vector routing," in *Second IEEE Workshop on Mobile Computer Systems and Applications*, 1999, pp. 99-109.
- [9] D. Johnson and D. Maltz, "Dynamic source routing in ad hoc wireless networks," in *Mobile Computing*, 1996, pp. 153-181.
- [10] O. Masoud, N.P. Papanikolopoulos, "A novel method for tracking and counting pedestrians in real-time using a single camera", *IEEE Trans. on Vehicular Technology*, vol. 50, no. 5, pp. 1267-1278, Sep. 2001.
- [11] Masoud, O., and Papanikolopoulos, N.P., "A Method for Human Action Recognition", *Image and Vision Computing*, Volume 21, No. 8, 2003, pp. 729-743.
- [12] R.M. Voyles and P.K. Khosla, "A Multi-Agent System for Programming Robots by Human Demonstration," in *Integrated Computer-Aided Engineering*, v. 8, n. 1, pp. 59-67, 2001.
- [13] S. Watters, T. Miller, P. Balachandran, W. Schuler, R. Voyles, "Exploiting a Sensed Environment to Improve Human-Agent Communication," in *Proceedings of the 4th International Joint Conference on Autonomous Agents and Multi-Agent Systems*,
- [14] Silvia Coradeschi, Alessandro Saffiotti, "Symbiotic Robotic Systems: Humans, Robots, and Smart Environments," *IEEE Intelligent Systems*, vol. 21, no. 3, pp. 82-84, May/June, 2006.
- [15] D. B. Stewart and P. K. Khosla, "The Chimera Methodology: Designing Dynamically Reconfigurable and Reusable Real-Time Software Using Port-Based Objects," *International Journal of Software Engineering and Knowledge Engineering*, vol. 6, no. 2, pp. 249-277, June 1996.
- [16] J.E. Cooling, "The Real-Time Data Flow Diagram: Control Plus Data," chapter 7 of *Real-Time Software Systems*, International Thompson Computer Press, Boston, MA, 1997.