Project Voyager: Building an Internet Presence for People, Places, and Things

by

Wesley Chan

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Masters of Engineering in Computer Science and Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 2001

Author.....
Department of Electrical Engineering and Computer Science
May 18, 2001

Certified by.....
Michael Hawley
Dreyfoos Professor of Media Technology
Thesis Supervisor

Accepted by.....
Arthur C. Smith
Chairman, Department Committee on Graduate Students

Project Voyager: Building an Internet Presence for People, Places, and Things

by

Wesley Chan

Submitted to the Department of Electrical Engineering and Computer Science on May 18, 2001, in partial fulfillment of the requirements of the degree of Masters of Engineering in Computer Science and Engineering.

Abstract

Project Voyager builds applications that link virtual services to people, places, and things. This thesis describes the design and implementation of two such systems.

The MIT Campus Guide application provides services for specific points of interests. Campus visitors used the system to interact with different sites—for example, a restaurant, a lecture hall, or a sculpture.

The Voyager Personal Shopping Assistant (PSA) connects shoppers in a supermarket to virtual services, linked to specific products. Customers used handheld computers to scan products. The system retrieved relevant information from the Internet about each item, including recipes, nutritional information, and related product recommendations.

Project Voyager successfully deployed both systems and conducted informal user trials. User feedback confirmed our belief that people derive significant value from interacting with information that is relevant to them.

Acknowledgements

This thesis is for the following people, whom shared my vision for changing the way we interact with other people, places, and things.

Mike Hawley, for challenging me to aim for the sky. Brian Smith, Turner Whitted, and Mike Sinclair, for keeping my feet on the ground. Osh Momoh and Mike Smith, for introducing me to the technology needed to make Voyager happen.

L. Jonathan Brunsman, for being one awesome Pocket PC programmer. Neil Chungfat, for learning how to code in C# in less than two weeks. Manny Picciola, for teaching me the meaning of the phrase "value proposition."

Chris Newell, Amy Holden, Kristin Hall, and Charlotte Burgess for believing in Project Voyager when everyone else thought it was a crazy idea.

Curt Avallone, for allowing me to tear apart a Stop & Shop supermarket. Dan Fay, John SanGiovanni, and William Vabalis, for helping to make Project Voyager a reality. Doug Seven and Don Mack, for fixing bugs at three in the morning. Brandan Honan, for giving us more recipes than I could ever come up with on my own. John Britts and Noel Eberhardt, for giving us lots of fun new stuff to play with.

Matt Lee, Mike Goertz, Caroline Hon, and Jason Chan, for their unbounded support.

Contents

1.0 Introduction

1.1 Overview: The Usual, Mr. Bond?

Imagine how day-to-day interactions with people, places, and things will change when everything has a connected Internet presence. Cash registers will disappear. Credit cards or paper money becomes obsolete. Paying for groceries becomes a simple exchange of bits on the network: you pick items up, you are sensed as you walk out of the store, and your bank account is autobilled. Paying for cab fare could be as easy as pressing your thumb against a portable computer terminal.

Likewise, things and places will talk to you. Museum paintings sense you and tell you their story. A package of macaroni and cheese reminds you to buy the milk and butter needed to prepare it. Your camera teaches you how to take dazzling photographs. You'll even be able to know the top three entrées—and the *soupe du jour*—at every restaurant you walk by, without ever having to set foot in one.

Better yet, things will talk to each other. Your smart microwave oven identifies a box of frozen vegetables and automatically cooks it for two minutes. A carton of orange juice tells your fridge that it is empty and adds it to your electronic shopping list. Your car tells your handheld computer that it needs an oil change and schedules an appointment with your mechanic. Your cell phone can tell a computer projector to display a slide deck for an important sales presentation you are about to give.

The emerging technologies of automatic identification, locus and other linked sensing, wireless capillaries, and services on tap, will cause traditional interfaces to vanish. Radically different, simpler interfaces will result. Gone are the long lines at the supermarket checkout stand—you pay by pressing your thumbprint at a networked terminal. The knobs, dials, and buttons on a coffee maker will disappear—the appliance brews coffee when your alarm clock tells it to, and dispenses it just the way you like it (with two lumps of sugar and some cream) when it senses your cup. Even James Bond

will be able to order a martini—shaken, not stirred—at a bar he's never been to, without having to utter a word.

This transition is underway. W.I.S.E. technologies—Wireless capillary networks, Internet presence platforms, Sensing devices, and Electronic services—are a sign of the times. You can browse the web and send e-mail messages on a cell phone. You can even connect to a high-speed 802.11 wireless network while sipping a latté at a Starbuck's Café. New wireless data networks will provide connectivity everywhere and anywhere [BT, Deboo, IEEE, HRF, Rehoo, Whoi]. Also, you'll be able to pay for a TV by simply walking out of a department store—Motorola's BiStatix RFID tag readers [Mot] will automatically sense the television and autobill your bank account. Teams at HP Labs [KBoo, Kinoo, Praoo] and the MIT Auto-ID Center [AID, Schoi] are working diligently on building an infrastructure to create Internet presences for people, places, and things. Companies like Microsoft and Sun Microsystems are creating platforms (.NET and Java) to facilitate the implementation of e-services [NET, JINI].

With Project Voyager, I am examining new notions of people, places, and things online. To do this, I have built two applications—the MIT Campus Guide and the Voyager Personal Shopping Assistant—to explore a bit in the realm of what has just become possible.

1.2 Thesis Structure

This thesis consists of four sections. Chapter 2 discusses the MIT Campus Guide, an early Project Voyager application. Chapter 3 describes the Voyager Personal Shopping Assistant, which connects customers to information services linked to products. Finally, Chapter 4 discusses future directions for Project Voyager and concludes with a summary of our research and its significance.

2.0 The Voyager MIT Campus Guide

2.1 Motivation

An audio museum guide—the headphones and cassette player you might wear at a gallery—is just a sequence of narrative clips. Using it, however, restricts you to following a pre-programmed path. Suppose instead that you could wander freely at a museum—indoors or out—and hear clips keyed to things and places of interest. Each place or object would have a story to tell. By going near that object or place, you could hear its story.

Figure 2.1

The MIT Campus Guide system helps users locate and interact with points of interests on campus.

The MIT Campus is such a "museum." The campus is a cluster of computer clusters, statues, libraries, and restaurants, each with a story to tell, and services to deliver. A computer cluster may have information about how to use the workstations or print a document. Just as important, however, are the stories of students spending late nights working on projects or learning how to write the next killer application that takes the business world by storm. Currently, the only way to hear these stories is through a printed guide or a live person giving the tour. Many visitors to the campus do not

have access to a live tour guide. A multimedia, location-activated virtual guide would be an ideal substitute—hence the motivation for the MIT Campus Guide System (see Figure 2.1).

2.2 Design

2.2.1 Functional Description. The Voyager MIT Campus Guide connects visitors to virtual services based on their location. Various "points of interests" are tied to an array of information and services—for example, an audio narration describing a sculpture or the weekly menu of a dorm cafeteria. When the guide detects that the user has arrived at a point of interest, it notifies the user. The user then has the option to interact with the information or services associated with that location, and can do so by using the Campus Guide system.

2.2.2 Design Considerations. We designed the system with several considerations in mind. First, client devices are usually limited in screen size, battery life, and storage space. A Palm VIIx PDA, for example has a 240 x 160 pixel display, less than 10 hours of battery life, and 8 MB of memory. Second, MIT does not have a high-bandwidth local area network that covers the entire campus. The Campus Guide would have to rely on a slow cellular network (with a maximum bandwidth of 9.6 Kbps) as a means to store and retrieve remote data. Finally, the simplest way of sensing position was through using a GPS receiver. However, GPS receivers only work outdoors when the unit has a clear view of the sky. Thus, when the user is indoors, the system will be unable to track the user, unless another mechanism is available.

2.2.3 Design of the System. The MIT Campus Guide consists of three components: a mobile client device, a software application, and an optional remote server component. When the application starts, it loads and parses a data file containing a mapping of locations to services and data. The file can be cached locally or stored remotely on a server. Information in the

data file is stored in the following format, with the "/" character as the delimiter:

```
Longitude/Latitude/Location Name/Location Description/Image File/Photo File/
```

An example of an entry in the data file appears below:

```
2.3612159/71.0875458/MIT Media Laboratory/The Media lab is a world-class research institution that pioneers collaboration between academia and industry. The lab provides a unique environment to explore basic research and applications, without regard to traditional divisions among diciplines./medialab.wav/medialab.jpg/
```

Data files stored on the remote server can be updated. To ensure that updates propagate, the client application synchronizes the file periodically with the server. When synchronization occurs, the application loads the new file and creates a new database. Additionally, any new audio or image files will also be downloaded to the client device. Old audio files that are no longer referenced in the new data file are deleted to conserve space.

When the application detects that the user has arrived at a point of interest, it displays a list of services (e.g. a text description, an audio narration, or an photograph) linked to that location. Visitors then have the option to view the photograph, play back the audio file, or read the text description of that site. Also, audio or image files are not streamed off the server in real time, since network bandwidth is too slow.

When the user is not at a point of interest, the application displays a map of the MIT campus, centered at the user's current or last known position. The map is cached locally and the application loads the image when it is executed. The application also monitors the serial port for GPS data. When a new coordinate is received, the application performs a polar transformation on the latitude and longitude coordinates in order to convert them into linear Cartesian map coordinates.

2.3 Implementation

2.3.1 Hardware. The MIT Campus Guide consists of several off-the-shelf hardware components. The Campus Guide application software runs on a Compaq iPaq H3650 Pocket PC pre-loaded with Microsoft Windows CE 3.0 as the operating system. The Pocket PC has a 12-bit color display, the ability to play back full motion video and MP3 audio, and a built-in web browser [PPC]. A Garmin GPS 12-channel parallel receiver tracks the user's location. An extrenal GPS antenna helps to boost sensitivity so that the receiver can still triangulate the user's position among the many tall buildings that may otherwise cause the receiver to lose the signal. A custom-built cable connects the GPS to the serial port of the Compaq Pocket PC. (See Figure 2.2)

Figure 2.2

The MIT Campus Guide system consists of a GPS Receiver connected to a Pocket PC handheld computer.

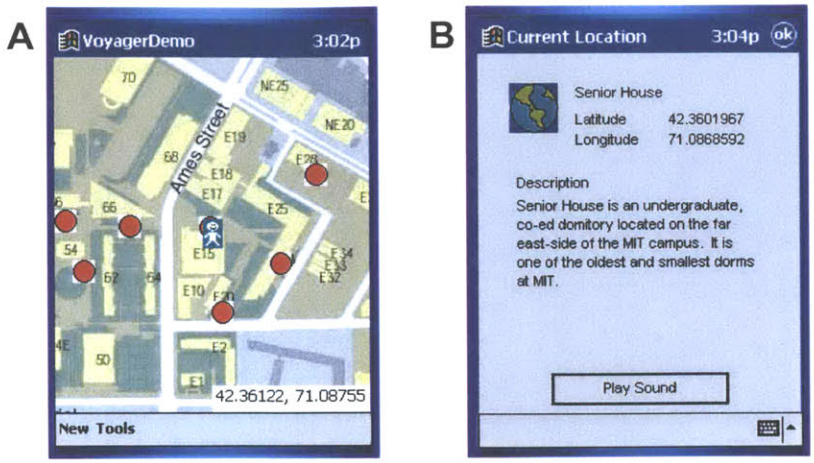
2.3.2 Connecting the GPS to the Pocket PC. The Garmin GPS receiver transmits its coordinates to the Pocket PC using a standard 4800 bps RS-232 serial connection. The GPS encodes coordinates using the National Marine Electronics Association (NMEA) 0183 standard [NM]. The MIT

Campus Guide application periodically monitors the serial port for any communication from the GPS receiver.

2.3.3 Application Software. We wrote the MIT Campus Guide application using Microsoft Embedded Visual C++ 3.0. The application runs on Pocket PC devices with a StrongArm processor, a Hitachi SH3 processor, a MIPS processor, or Intel x86-based processors.

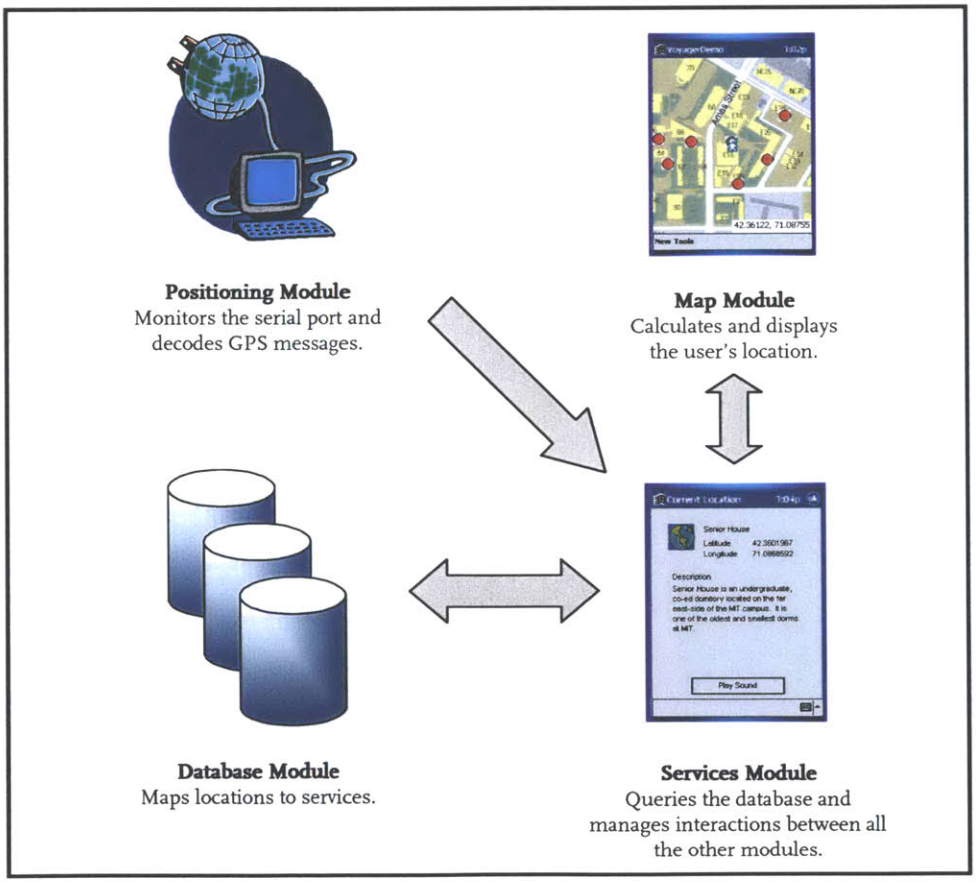
Each of the modules consists of one or more separate C++ classes in the application. For example, the *positioning module* consists of several classes that monitor the serial port, read in NMEA packets sent by the GPS receiver, and parses the data packets into a format that other modules can use. The *map display module,* on the other hand, consists of a class that paints the map image and converts polar GPS coordinates to Cartesian map coordinates. Furthermore, the *database module* consists of two classes: one that loads and parses the data file, and another that specifies the data structures for the internal database. Finally, the *services module* consists of several classes, each containing functions to display or play back multimedia files. Together, the modules abstract functionality of the program into discrete components that create a robust system.

Figure 2.3



(A) The main display screen of the MIT Campus Guide consists of a map of the campus, centered at the user's current location (green icon). The red circles represent various points of interests. (B) When the user reaches a point of interest, services appear that the user can take advantage of. This point of interest (Senior House) has a description service and an audio narration service.

Figure 2.4



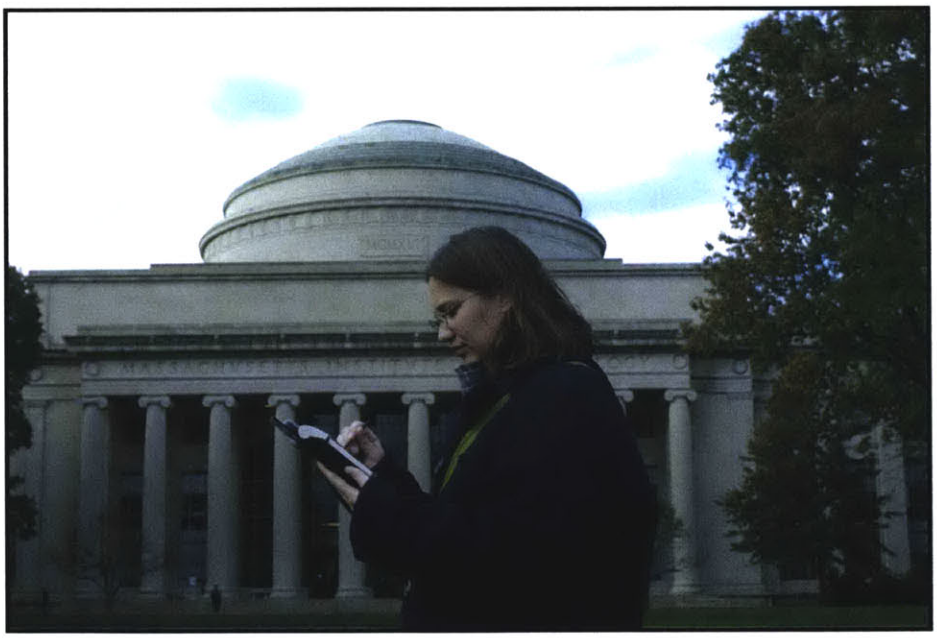
The four modules of the MIT Campus Guide system.

2.4 Evaluation

2.4.1 User Feedback. Over 40 users toured the MIT Campus with the system during the Media Laboratory Things That Think consortium meeting in October, 2000. Before using the system, we briefly described system and how it worked. After touring the campus, users met informally for half an hour to discuss their experiences while using the MIT Campus Guide.

Feedback was generally positive. Almost all of the users felt that the application exceeded their expectations. A majority said that the interface was intuitive and easy to use. Furthermore, many people were actually surprised to have something "pop up" on their screen when they reached a particular point of interest. Many users told us that they were initially skeptical that the MIT Campus Guide would actually work.

Figure 2.5



Charlotte Burgess Auburn testing the MIT Campus Guide system.

Some people, however, felt that the system would not be useful unless it also had an indoor positioning system, which the Campus Guide lacked. As several sponsors pointed out, most visitors often spend their time indoors because of inclement weather that Boston is known for. Thus, having an exclusively outdoors-only system would not be too useful.

Other users suggested that the Campus Guide include a "user-profile" system to automatically generate a points-of-interest listing that would be most appropriate for that person. For example, a first time visitor to the campus should have a different experience then a parent of a student who has been on campus before. Finally, people also suggested that the application should include an *instant messenger* or *communications* component. This module would allow users to communicate in real time with someone who they could ask questions about a particular location.

2.4.2 Future Work. In addition to building any of the new features suggested during the informal feedback session, much work still remains to be done on the MIT Campus Guide. First, the client hardware needs to be redesigned. Sensors that track a user while indoors need to be integrated into the system. Furthermore, the GPS receiver should be miniaturized. Embedding a GPS receiver inside the Compaq iPaq or the PC Card Sleeve make the system easier to handle. Additionally, due to time limitations, we were unable to finish the *communication module*—which enables the application to query a remote server and download new data files. Also, we were unable to build a server to store data files (again, due to time limitations), so the application had to cache everything locally on the unit. These two components would add significant functionality to the Campus Guide system.

3.0 The Voyager Personal Shopping Assistant

3.1 Motivation

Buying groceries today is rather mundane. You enter a supermarket. You pick up a shopping cart. You walk around the store and add items to your cart. While in the breakfast foods aisle, you grab add a box of pancake mix. You pick up milk and eggs afterwards. Then you spend 15 minutes waiting in line, right behind the mother with an overloaded cart and two screaming kids. As you leave the store, it instantly hits you: you forgot to pick up a bottle of maple syrup for the pancakes you plan to make tomorrow.

Figure 3.1

The Voyager Personal Shopping Assistant aspires to change the way you shop by connecting you to the virtual services tied to products you buy.

The Voyager Personal Shopping Assistant (PSA) can change the way you shop. The PSA entertains and educates you. It teaches you how to prepare new recipes and helps to count calories if you're on a diet. It will also help you save money by offering personalized discounts, attached to the products you buy. Better yet, the system eliminates much of the hassle of buying groceries—like waiting in line at the checkout stand or forgetting to purchase something that you need.

Stores also benefit from the system. Shoppers using the system will buy more, increasing the store's revenue. Royal Ahold, the parent company of the Stop & Shop supermarket chain, estimated in a marketing study that customers would spend on average $51 more each month if they had a "smarter" shopping cart. In fact, people that brought groceries at non-Stop & Shop supermarkets would spend on average $85 more per month at Stop & Shop if they had access to such an application.

By automating the checkout process, stores would also reap substantial savings by eliminating the need for checkout stands. Each stand, for example, costs over US $22,000 a year to maintain. If the store can reduce even a few of them, they will save a sizeable amount of cash. The additional savings could enable supermarkets to offer additional amenities. Store associates could roam around, offering tasty samples to customers. Other employees could give customers shopping tips, help them locate items, and take groceries out to their car. Stores offer a much more animated place to buy groceries. Costs go down. Service improves. As a result, revenues increase.

On the other hand, manufacturers like Kraft and Proctor & Gamble can use the PSA as a personalized marketing tool. With services on tap like recipes, promotions, nutritional advice, and entertainment, right there in the shopping cart, manufacturers can enhance the shopper's experience and also influence buying behavior. Kraft can offer personalized services like recipes and nutritional facts. Proctor and Gamble could offer promotional services, offering discounts for related items—scan a bottle of Tide laundry detergent

and get a coupon for a free box of Bounce dryer sheets. These value-added services differentiate a manufacturer's products from others. As a result, directing marketing costs go down. More people buy a manufacturer's products. Revenues therefore increase.

Figure 3.2

By providing customers information about their products at the point of decision making, manufacturers can use the Voyager Personal Shopping Assistant as a more effective marketing tool.

Looking beyond the supermarket, the Voyager Personal Shopping Assistant can also alter the way other retailers do business. Where Amazon.com brought stores into people's home, the PSA brings Amazon.com into brick-and-mortar stores. Amazon today offers a wide variety of services to consumers. You can read reviews while searching for a novel. You can also pay for items with the click of a button. Amazon.com even thoughtfully suggests that you buy extra batteries when you purchase a new camera. However, you can't hold or play with the product until *after* waiting a week or more for Amazon to ship the product to you.

The PSA brings these services, however, to brick-and-mortar stores. Items you choose are sensed. Product reviews for those items pop on the system's display. Coupons for related items appear when you ask for them.

Furthermore, checking out is as simple as pressing a button and entering your authorization code—no more waiting in line for a cashier. Most importantly, however, you can touch, smell, play with, and use the products in the store before buying them—something you can't do while shopping online from your home PC. As an added bonus, you are able to walk out of the store with the items you buy—rather than having to wait a week or more for them to arrive.

3.2 Design

3.2.1 Functional Description. The Voyager Personal Shopping Assistant enables customers to interact with electronic services linked to items in a supermarket. The system delivers personalized, product-specific information and services to shoppers via a wirelessly networked PDA with an attached product scanner. The PSA also provides additional services, including personalized marketing (for example, the customer may receive specific discounts on products based on their pervious shopping history), time-based electronic coupons (e.g., a bottle of milk may be nearing its expiration date so the store issues an electronic coupon for 50% off its regular price), self-checkout, mobile point of sales, a nutrition guide, and a shopping list planner. The Voyager PSA is a proof-of-concept prototype—so many of these services may not be available initially, or for all products. However, we designed the system to allow new services to be added once they become available.

3.2.2 User Scenarios. A customer enters the store and borrows a shopping basket or cart, mounted with a wirelessly networked handheld computer and a product scanner. The shopper logs onto the system (for example, by scanning their store loyalty card). The person then walks around the store and scans items into the system before placing them into their cart. After scanning an item, information about the product appears on the PSA's display, including any services associated with it. Some products may have many services connected to it, including a recipe service, a related products

suggestions service, and electronic coupons. Other items, however, may have only a single service associated with it—most likely a pricing service offered by the store.

As the user scans each item, it is added to the shopper's virtual shopping cart. The customer presses the "Done Shopping – Check Out" button when they have finished shopping. At that time, the customer's bank account or credit card is automatically debited. The shopper picks up a printed receipt at a kiosk right before they leave the store, in addition to a printout of recipes and other services they elected to "save" while using the system.

3.2.2 Design Considerations. We designed the PSA with several important technical issues in mind:

Open Standards. The system should be based, as much as possible, on current and emerging open standards—like HTTP and XML. This will encourage manufacturers and retailers to create additional services for products and customers, furthering adoption of the system.

Product Identifiers. The PSA application must be able to systematically identify every product that the customer scans into the system. This mechanism must be inexpensive, require little effort to install, and support the 55,000 different items that a supermarket typically stocks. Furthermore, the system should be able to support any new product identification mechanism that may be used at a future date.

Multiple Service Providers. The system should allow different organizations to easily connect services to products and customers. Manufacturers may want to create a recipe service for their products. Review sites may want to create product reviews for products. Advertising agencies may want to offer promotions or other discount services with the products in their portfolio.

Internet Presence Infrastructure. The system must support an infrastructure that systematically ties services to people, places, and things. Additionally, this infrastructure must be simple and easy to use, so that

providers can easily expose their previously incompatible legacy information systems.

Multiple Device Types. The system should run on multiple client devices, including Microsoft Pocket PC clients, Palm-OS based clients, WAP phones, and web tablets.

Scalability. The system should work for a convenience store with 500 different items, a mid-sized grocery store with 10,000 different products, or a supermarket that stocks 55,000 different items.

Additionally, there are several non-technical considerations that we factored into the design of the PSA:

Perceived Time Savings. Shoppers want to think that they are saving time. In a focus group study done by the Stop & Shop supermarket chain, customers indicated that convenience was the most important benefit they would get from a Personal Shopping Assistant application. Initial services for the prototype should provide some sort of perceived time savings benefit.

Ease of Use. Many customers that will use this system may not be computer-literate. Stop & Shop's focus group study also concluded that customers do NOT want to expend much effort to learn how to use a new system in the store. The interface for the system should be intuitive and easy the follow; the shopper should be able to learn how to use the system in less than two minutes.

3.2.2 Design Overview. The Voyager Personal Shopping Assistant application consists of the following components (see Figure 3.3):

Object Naming Service (ONS): The ONS connects an arbitrary Product ID to the name or address of the service that contains a schema for that particular product.

Product Schema Service (PSS): The PSS map an arbitrary Product ID to a directory of services that is associated with that product. For example, a box of cream cheese may be tied to a recipe service, a product suggestion

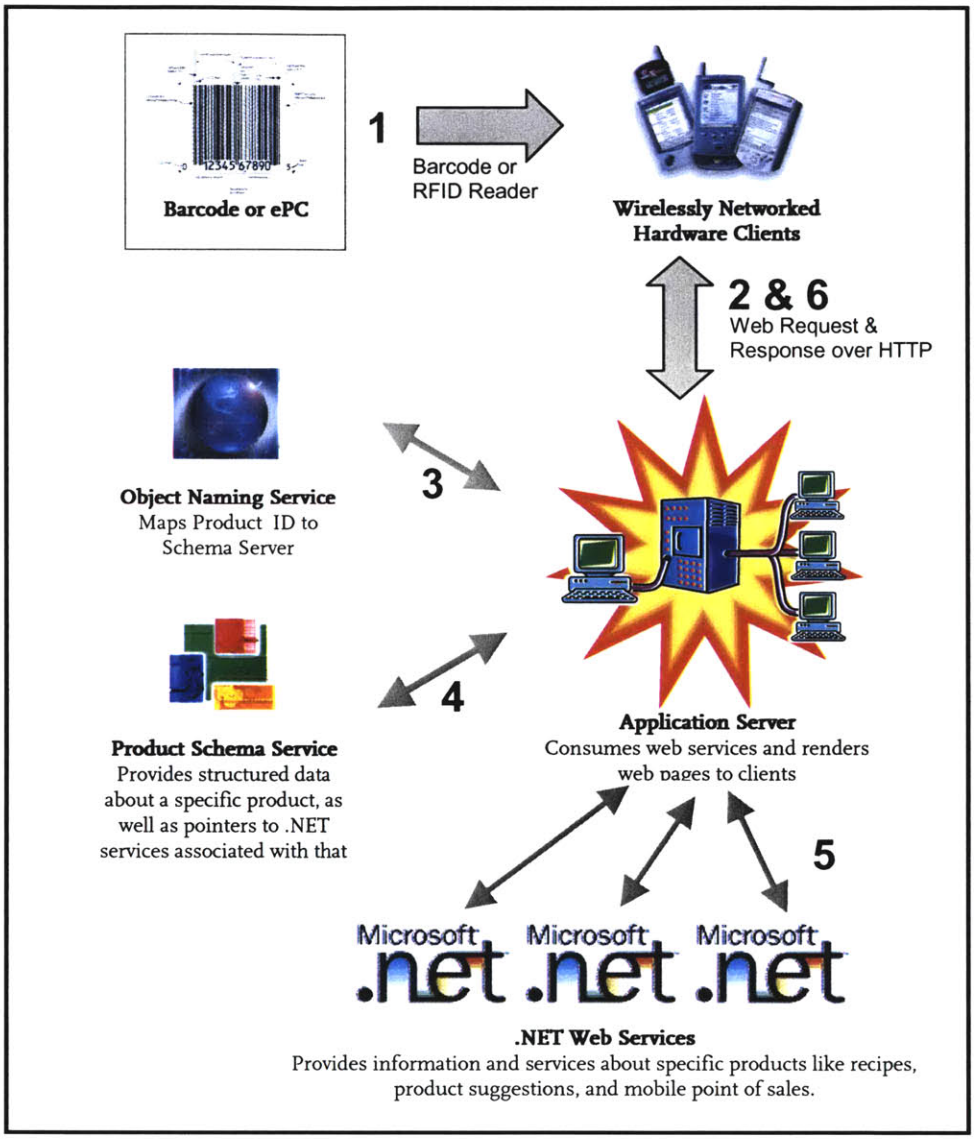
service, a nutritional facts service, and a pricing service. The PSS maps the Product ID for the box of cream cheese to pointers to these services.

Various Web Services: These are services that are connected to specific products, which provides a wide variety of information that the consumer can interact with. Examples of these services include a recipe suggestion service, a product recommendation service, and a pricing service.

Application Server: The Application Server is an interface between a user and the services that are connected to a product. The server contains the application logic for the entire system. Note that the application server can be replaced by a *compiled application* that runs directly on the client. However, because of the desire to support different client device, rendering a display to the client's browser via HTTP or WAP is the simplest design for this proof-of-concept system.

Mobile Client Device: This is the device that the customer interacts with, and uses to scan in Product IDs. The software on the device consists of a customized web browser that will send web requests to and display web pages created by the application server.

Figure 3.3



(1) The consumer scans a Product ID into the system. (2) The client transmits a web request with the Product ID to the application server. (3) The application server queries the Object Naming Service with the Product ID. The Object Naming Service returns a pointer to the address of the Product Schema Service mapped to that particular Product ID. (4) The application queries the Product Schema Service to retrieve a list of services associated with that Product ID. (5) The application server then retrieves information (e.g. recipes and pricing information) from those services. (6) The application server renders that information on a web page that is sent to the hardware client and displayed to the user.

3.2.3 Communications Protocols. Web requests and other messages should be sent over a standard Hypertext Transfer Protocol (HTTP/1.1) channel [Lee97]. Additionally, messages sent between the various backend web services and application server should be packaged using the Simple Object Access Protocol (SOAP) [Box00], which is based entirely off the eXtended Markup Language (XML). Both HTTP, SOAP, and XML are widely-used open standards that are backed by the World Wide Web Consortium, or W3C.

3.2.4 Object Naming Service (ONS). The Object Naming Service maps an arbitrary Object ID to the Internet address of the computer that has a schema for that Object ID (see Figure 3.4). The ONS is based very loosely on the on a similar concept developed by the MIT Auto-ID Center. The Auto-ID Center's Object Naming Service, in turn, is based loosely on the Domain Naming System (DNS) standard that many Internet computers rely on to translate addresses (e.g. www.media.mit.edu) to numeric IP addresses (e.g. 18.85.2.80). [AID]

Figure 3.4

The Object Naming Service takes a Product ID as an input and returns the Internet address of the schema server associated with that Product ID.

Like the DNS, the Auto-ID Center's Object Naming Service is a distributed service. Resolving one Product ID to an Internet address often requires the use of several networked servers, as information about each *part* of the address are distributed to various computers on the network. This allows the service to be *scalable*. Thus, the Auto-ID Center's Object Naming Service can handle billions, or even trillions of Product ID translation requests. The trade-off, however, is that the service can only support the use of one standardized, Product ID format as opposed to an arbitrary Product

ID. In fact, the Auto-ID Center's Object Naming Service only resolves Product IDs that conform to their electronic Product Code (ePC) standard. [AID]

The PSA's ONS service, on the other hand, supports a wide variety of Object IDs, including ePC, Universal Product Code (UPC) barcodes, or even the fully qualified name of the object. This extra flexibility has a cost, however, in terms of scalability. Since arbitrary Object IDs have no hierarchy, the service cannot distribute lookup information across several servers. Thus, the Voyager ONS cannot handle billions or even trillions of lookups every second. On the other hand, for small systems like the Personal Shopping Application, such a system is appropriate.

3.2.5 Product Schema Service (PSS). The Product Schema Service maps a Product ID to a schema of structured data (see Figure 3.5).

Figure 3.5

The Product Schema Service maps a Product ID to a particular schema, which contains a listing of pointers to services associated with that particular Product ID.

The schema consists of a listing of pointers to available services for a specific product. An example schema for a box of Kraft Stove Top Stuffing (with the Object ID 04300028521):

```
<?xml version="1.0" ?>
<DataSet xmlns="http://explorer.media.mit.edu">
   <xsd:schema id="_x0030_43000285213ProductServicesDataSet"
         targetNamespace="" xmlns=""
         xmlns:xsd="http://www.w3.org/2000/10/XMLSchema"
         xmlns:msdata="urn:schemas-microsoft-com:xml-msdata">
      <xsd:element name="WebServs">
         <xsd:complexType>
            <xsd:all>
               <xsd:element name="upc" minOccurs="0"
                  type="xsd:string" />
               <xsd:element name="Service_Name" minOccurs="0"
                  type="xsd:string" />
```

```
            <xsd:element name="ServiceLink" minOccurs="0"
               type="xsd:string" />
          </xsd:all>
        </xsd:complexType>
      </xsd:element>
      <xsd:element
          name="_x0030_43000285213ProductServicesDataSet"
          msdata:IsDataSet="true">
        <xsd:complexType>
          <xsd:choice maxOccurs="unbounded">
              <xsd:element ref="WebServs" />
            </xsd:choice>
        </xsd:complexType>
      </xsd:element>
  </xsd:schema>
  <_x0030_43000285213ProductServicesDataSet xmlns="">
    <WebServs>
        <upc>043000285213</upc>
        <Service_Name>Nutritional Facts</Service_Name>
        <ServiceLink>
          http://18.85.9.15/psa-nutrition.aspx?upc=043000285213
        </ServiceLink>
    </WebServs>
    <WebServs>
        <upc>043000285213</upc>
        <Service_Name>Recipes</Service_Name>
        <ServiceLink>
          http://18.85.9.14/psarecipes.aspx?upc=043000285213
        </ServiceLink>
    </WebServs>
  </_x0030_43000285213ProductServicesDataSet>
</DataSet>
```

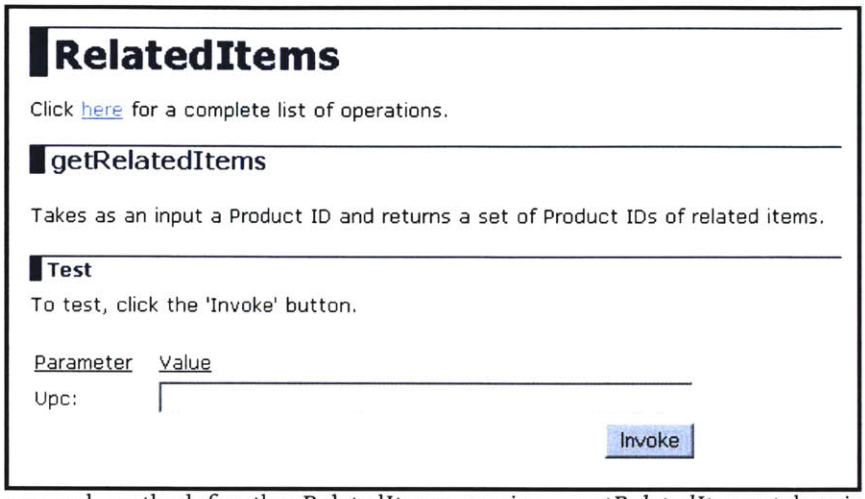
The box of stuffing has two services linked to it: a recipe suggestion service (pointer at `http://18.85.9.14/psarecipes.aspx?upc=043000285213`) and a nutritional facts service (pointer at `http://18.85.9.15/psa-nutrition.aspx?upc=043000285213`). The schema is encoded using the Simple Object Access Protocol (SOAP) [Box00], which is based on XML.

3.2.6 Backend Web Services. A product may be tethered to several backend web services. In a supermarket, products may be connected to recipe services, nutritional facts, pricing services, and a product identifier service, just to name a few. Different organizations may provide the services. For example, the manufacturer may provide recipes that use a particular product, while the store may provide a pricing service for that product. Additional services may be associated with a particular product by adding pointers to that service to the product's schema, maintained by the PSS.

A service contains an exposed set of methods that a particular application can call. For example, the related-items service has one method:

GetRelatedItems(String productID), which takes in a Product ID and returns a set of Product IDs of related items (See Figure 3.6).

Figure 3.6



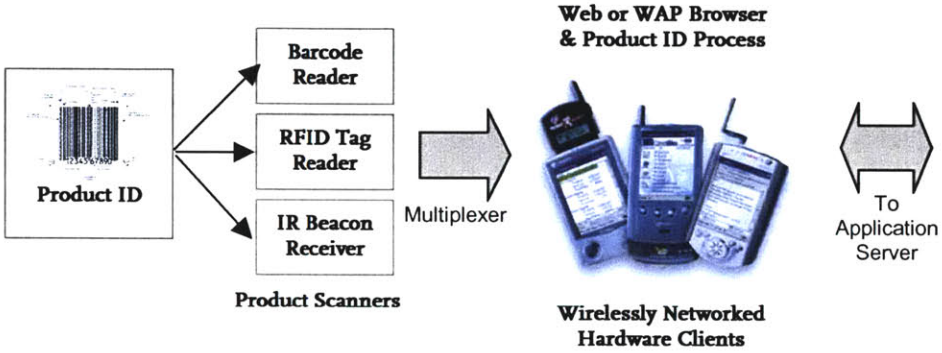
An exposed method for the RelatedItems service. *getRelatedItems* takes in a Product ID as an input and returns a set of Product IDs of related items.

3.2.7 Application Server. The application server consumes services provided by the ONS, the PSS, and the various web services for different products. The application server first receives a web request containing a Product ID from the mobile client device. The application server, after querying the ONS and the PSS, receives a list of pointers to services associated with that product. The server then renders a web page that lists available services for the user to interact with, or defaults to a particular service (e.g. pricing or recipes) that is of use to the user. The application server thus contains the logic needed to discover services associated with a particular product, to present those services to a user, and to enable the user to interact with those services when appropriate.

3.2.8 Mobile Client Devices. Customers interact with the PSA via the mobile client device. The device consists of a wirelessly-networked PDA with an attached product scanner (a barcode or RFID tag scanner, for example). The attached product scanner connects to the PDA's serial port). The PDA also has a custom web browser that displays web pages to the user. Additionally, a program will run on the client that will trigger the browser to

send a web request to the application server when the customer scans a Product ID. Once the web request is sent, the application server will render relevant web page for that product, and the browser on the client will display that page to the user. (See Figure 3.7).

Figure 3.7



The design of the mobile client device, consisting of a wirelessly networked PDA and an attached product scanner.

3.3 Implementation

3.3.1 Implementation Overview. In three months, we built and deployed a prototype of the PSA at Stop & Shop Store #35, located in Hingham, Massachusetts. The PSA consisted of several web services and an application server, all hosted on a server located inside the Stop & Shop supermarket. We connected the server to the store's internal 2 MB wireless local area network. We also provided shopping baskets mounted with a wirelessly networked PSA client to shoppers that signed up to test the system. When customers finished using the system, they could print out a receipt and any recipes they saved at a kiosk located at the front of the store.

Additionally, there are approximately 55,000 different products in a Stop & Shop supermarket. Each product has a unique 8- or 12-digit UPC code. Consequently, the PSA used a product's UPC code as the product identifier, since using another mechanism would have been too time

consuming, or too costly. Customers additionally signed onto the system using their Stop & Shop loyalty card, which had a globally unique 13-digit barcode. This "Customer ID" enabled the system to maintain session state after the shopper logged into the system (see Figure 3.8).

Figure 3.8

(A) Customers signed onto the system using their Stop & Shop card. (B) The PSA maps services to a product's UPC barcode. Every different product has a unique 8- or 12-digit UPC barcode number.

Several different organizations assisted in building the Voyager PSA prototype. Royal Ahold—the parent company of the Stop & Shop supermarket chain—provided the pricing data that Project Voyager used to create a pricing service and the use of one of their Stop & Shop supermarkets deploy and conduct user trials. Microsoft Corporation assisted in building the application server and backend services using their soon-to-be-released .NET Framework. Additionally, Microsoft supplied several Compaq iPaq H3600 series Pocket PCs that were used to construct the PSA shopping baskets. Symbol Technologies and Motorola Corporation provided barcode scanners and a BiStatix RFID Tag Reader that we also used in building the PSA baskets.

3.3.2 Using Microsoft's .NET Framework. We built the application server and the various backend web services using Microsoft's .NET Framework. While the .NET Framework supports numerous languages, we chose to write the application server and PSA services in C#,

an object-oriented language that was created especially for use with the .NET Framework.

The .NET Framework was an obvious choice for implementing our initial version of the PSA for a couple of reasons. First, the framework provides an easy-to-use platform to build *web services*. It also supports automatic encoding and decoding of data into XML SOAP packages. Finally, the .NET Framework supports easy development of web forms for a wide variety of devices, including Pocket PC PDAs, and notebook computers. Accordingly, the .NET Framework helped us reduce the time needed to implement the PSA by providing the infrastructure necessary to exchange information between different services.

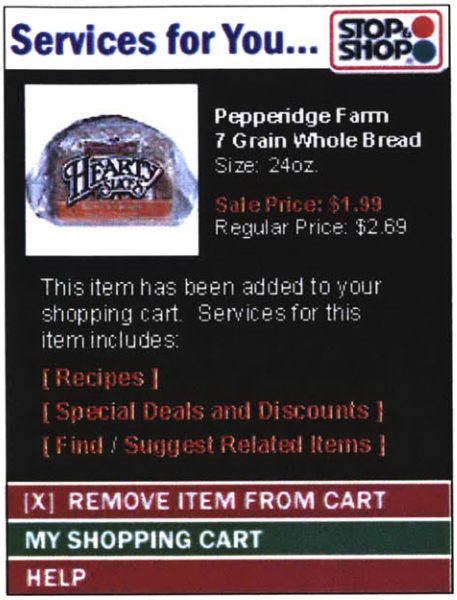
3.3.3 Building the System. The application server and web services were hosted on a single Pentium III-based computer running Microsoft Windows 2000 Server, Microsoft SQL 2000 Enterprise Server, Microsoft Internet Information Server, and the .NET Framework SDK (pre-release build 2615). We built the PSA clients using wirelessly networked Pocket PC PDAs with attached barcode scanners, mounted to plastic shopping baskets. Customers would "login" to the system by scanning the barcode on their Stop & Shop Card. They would then be able to move through the store and scan in the UPC codes of products they were interested in. The PSA generated a listing of services for that particular product when the customer scanned in a UPC code.

We built two default services for each product: a pricing service and an information service. The pricing service returned the real-time price of the product. The information service returned a cryptic one-line description of the product, from Stop & Shop's legacy product database (e.g. `KFT STV TOP STFN 20 OZ` for a 20 oz. Box of Kraft Stove Top Stuffing). We also built additional services for various products throughout the store. We implemented a "nutritional facts" service for a couple of products. Other service that we constructed include an extended information service about the product, which included the manufacturer, the product name, the size or weight of the product, and a product photograph, and a "related items"

service which recommended additional products that were "related" to the one that the customer scanned in (e.g. the customer scans in a steak and the system suggests a marinade sauce). Furthermore, Kraft Foods worked with Microsoft to create a recipe service for some their products. This service would return a list of recipes that used that Kraft product. Screen shots of how the application server renders these services as web pages to the user are available in Appendix C.

3.3.4 Product Information and Pricing Services. Royal Ahold provided a data file of the 55,000 items they stock in their Hingham Stop & Shop store, mapped to a price and a cryptic description of the product. Using this data, Project Voyager created a simple product information and pricing service that, when given a Product ID, returns a cryptic description of the product or the price of the product, respectively. One interesting thing to note—Royal Ahold encoded their UPC numbers in the file in a rather bizarre format. For most standard 12-digit codes, Royal Ahold striped off the first and the last digit of the code, and then added 2 leading zeros. For 8-digit codes, Royal Ahold striped off the first digit and the last two digits, and then added a leading zero and 5 additional zeros after the second digit (e.g. `12345678` becomes `02300000456`). Additional, UPC codes that begin with a "2" indicate that the item has a "variable price" (for unpackaged fruit and fresh meat products, for example). These items have their price encoded in their UPC code, so the service had to decode the price and return this information as appropriate. (See Figure 3.9)

Figure 3.9

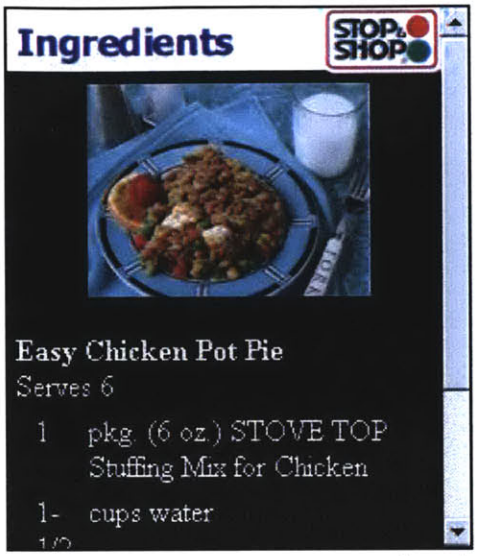


The pricing and information service delivers information about the product (manufacturer, size or weight, real-time pricing, and a photograph of the item) to the user.

3.3.5 Nutritional Facts and Related Items Services. The Nutritional Facts service maps a Product ID to a XML dataset of nutritional facts. The related items service maps a Product ID to a list of Product IDs of related items. Both services do not connect to another provider, and we entered the information for each product into the service table by hand.

3.3.6 Kraft Recipe Service. Kraft Foods, working in conjunction with Microsoft, built a .NET Recipe Service that maps the Product IDs of Kraft Products to various recipes that use those products. Kraft provided 500 Kraft items mapped to 818 different recipes. With Kraft's permission, we entered several hundred more non-Kraft products into the recipe service, so that non-Kraft items would also be mapped to recipes in the database. (See Figure 3.10).

Figure 3.10



The Kraft recipe service delivers recipes to customers when they scan products into the system. Customers can also save their recipes and print them out as they leave the store.

3.3.7 Check-Out and Recipe Printing Services. Neil Chungfat, an undergraduate researcher working on Project Voyager, built a check-out and recipe printing kiosk. The kiosk enables shoppers, after scanning their Stop & Shop Cards, to print out their receipt and the recipes that they saved while using the system. The kiosk consumes services that map a customer's Stop & Shop Card ID to the content of their virtual shopping cart, as well as a list of recipes that they have saved.

3.3.8 Wirelessly-Networked Pocket PCs Clients. The hardware clients, which the supermarket provided to customers during user testing, consists of a Compaq iPaq H3650 Pocket PC with an attached 802.11b wireless Ethernet PC Card Sleeve and a Symbol CS-1504 Keyfob Barcode Scanner, mounted to a plastic shopping basket (see Figure 3.11 & 3.12). The supermarket where the PSA was deployed already had a Symbol Spectrum 24 802.11 2 MB frequency hopping wireless network installed in the store. Thus, in order to connect to that network (and avoid having to install a separate one), the clients used a Spectrum 24 LA-3020 wireless Ethernet PC Card that was compatible with the in-store wireless network. A

bill of materials used to build the hardware clients, as well as schematics and photographs, can be found in Appendix B.

Figure 3.11

The Voyager PSA client consists of a Pocket PC PDA with an attached barcode scanner mounted to a plastic shopping basket.

Symbol Technologies provided ten CS-1504 Keyfob memory barcode scanners that we used to build several client prototypes [Sym]. The CS-1504 scanners attach to the RS-232 serial port on the Pocket PC PDA. However, the CS-1504 scanner does not simply send an ASCII numeric stream to the Pocket PC when an item is scanned; the contents of the scanner's memory had to be downloaded via a SDK that Symbol provided with the scanner. However, the SDK only worked on Windows-based computers. L. Jonathan Brunsman, another undergraduate researcher working on Project Voyager, ported the Symbol CS-1504 SDK code to the Pocket PC platform. Additionally, Jonathan wrote a simple program that sent a web request, using the Pocket PC's built in Pocket Internet Explorer web browser, with the Product ID to the Application Server when the customer scanned an item with the CS-1504 scanner. The application server then takes that

information, constructs a web page in HTML, and sends the page back to the Pocket PC. The Pocket PC displays the web page using Pocket Internet Explorer, allowing the customer to interact with that item's services.

Figure 3.12

A close up view of the Voyager PSA Client—a wirelessly networked Pocket PC mounted to a shopping basket.

3.3.9 Deploying at Stop & Shop. After testing the system at the Media Lab, we moved it to Stop & Shop Supermarket #35, located in Hingham, Massachusetts. The server hardware was physically stored in the computer room at the supermarket, and connected to the store's wireless IEEE 802.11 local area network. We provided shoppers who volunteered to test out the PSA with shopping baskets while in the store. Furthermore, the store already had a Symbol Spectrum 24 802.11 2 Mbps wireless local area network previously installed to support inventory operations.

Additionally, we also worked frantically to enter product information into the Product Information Service, the Nutritional Facts Service, and the Related Items Service before user testing. We took items off of the supermarket's shelves, entered in the relevant information into the various service databases, and photographed the items before returning them to the store shelves (see Figure 3.13). We successfully added approximately 850 items to the product information service before user testing began.

Figure 3.13

(Left to Right): Neil Chungfat, Manny Picciola, and Jonathan Brunsman entered product data into the Product Information, Nutritional Facts, and Related Items service databases.

Of note, many of the items in the fresh produce section did not have pre-printed UPC barcodes. Stop & Shop installed scales where shoppers could weigh their product items, enter a 4-digit "PLU" item code, and print out a UPC barcode label for their produce (see Figure 3.14). After printing the label out, shoppers could use the PSA to scan in the item.

Figure 3.14

Stop & Shop installed scales that shoppers can use to print out their own UPC barcode labels for produce items.

3.3.10 The BiStatix RFID Basket. We also built an additional PSA basket using a Motorola BiStatix RFID tag reader, instead of a Symbol barcode scanner. While at the Media Lab, we affixed BiStatix RFID tags to several grocery items and to a Stop & Shop card. Shoppers could use the basket by signing in with their RFID Stop & Shop card. They could then scan items with affixed RFID tags. The system was built for demonstration purposes at the lab, and we did not deploy at the Stop & Shop store. (See figure 3.15)

Figure 3.15

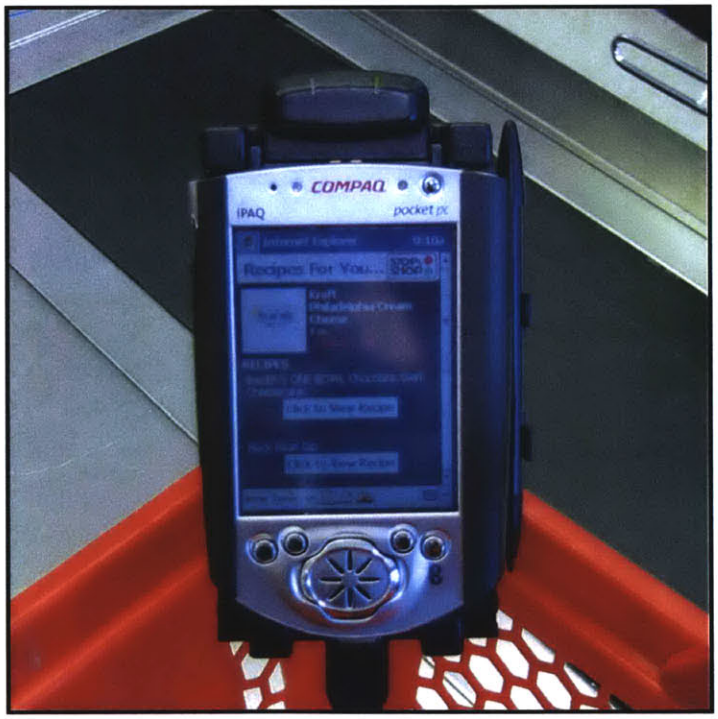
Another version of the PSA uses a Motorola BiStatix RFID Reader to scan in Product IDs. Items that work with this basket must have a BiStatix RFID tag (like the one on the back of the laundry detergent bottle) affixed to the label.

3.4 Results and Evaluation

3.3.1 Deployment. The PSA prototype system was built at the Media Lab, where it was first tested using approximately 50 different grocery products. Moving the PSA prototype from the lab into the supermarket, where there are roughly 55,000 different products, was a huge challenge.

What Went Right. For the most part, the PSA worked as intended we fixed several bugs. When a customer scanned in their Stop & Shop card, the system correctly displayed a page with five recipe suggestions for the user. When a shopper scanned items—including barcodes for produce items printed by a scale—the PSA appropriately displayed a web page with a description and the price of the product, in addition to a listing of any services associated to that item. Furthermore, when users choose to interact with one of the item's services (for example, viewing recipes), the system correctly displayed a web page with the service's features as well (see Figure 3.16). The PSA also correctly added scanned-in items to the user's shopping cart service. Equally as important, when the shopper scanned in an item not stocked by the store, the PSA appropriately displayed an error page.

Figure 3.16



The PSA correctly displays the recipe page for a box of Kraft Philadelphia Cream Cheese after the shopper scanned in the item.

While somewhat slow, the wirelessly networked clients successfully sent requests to and received responses from the application server when shoppers scanned in items. In fact, the supermarket's wireless local area network worked surprisingly well in almost every location within the store. We did not encounter any loss of connectivity while testing the system in the store; customers who used the PSA while shopping did not report any loss of network connectivity either.

What Went Wrong. Moving the system from the lab to the supermarket caused several problems. First, the system in the lab had a direct connection to the Internet. When we chose to have their saved recipes e-mailed to us, it worked while we were at the lab. However, the store had no easily accessible connection to the Internet, and the feature no longer worked. Consequently, the e-mail feature had to be disabled after moving the system to the store.

Figure 3.17

The Voyager PSA server hardware (right-most computer) was stored in the store's computer room, along with the store's pricing servers. The room had no direct connection to the Internet, so the "e-mail recipe" feature did not work. Furthermore, the pricing servers were on an entirely different physical network. Consequently, the product information service could not query the store's pricing servers.

The supermarket also changed the prices of products quite frequently. Every week, the store had a new set of advertised products, resulting in the temporary price reduction of several hundred products. Store managers also had the discretion to change prices, and they altered the prices of numerous items daily as well. The pricing service did not have a real-time connection into the Stop & Shop in-store pricing database (they were on entirely different physical networks). Instead, Royal Ahold's Information Technology group sent Project Voyager, via e-mail, a new items data file containing the product prices for that week. However, this file did not contain any of the daily price changes that occurred in the store. Consequently, many customers encountered prices that were not accurate, causing them a bit of confusion. This issue remains unresolved.

Additionally, Product IDs for several store-packaged fresh meat and cheese products were not in the items file that was sent by Royal Ahold. Customers who scanned in these items initially saw an error screen on the system, informing them that the store did not stock the item. We fixed the problem with a quick update to the pricing service. When the Product Information Service detected a variable price item (12-digit UPC barcode beginning with a "2") that was not in the Stop & Shop Items file, it generated a default product schema and returned the price that was decoded from the UPC code. Customers would then see a screen that displayed the description "Stop & Shop Grocery Item" and the price decoded from the UPC number.

Finally, we were unable to integrate our checkout service to the store's point of sale (POS) system. The store's POS system was somewhat complex, and integrating the two systems together would have taken several additional months. Thus, given the scope of this research, we "simulated" the checkout service by showing customers that they could print out a receipt, as well as any recipes they saved, at a kiosk located in the front of the store.

3.4.2 In-Store User Testing at Stop & Shop. Twelve Stop & Shop customers informally tested out the system. We issued each test subject a PSA client mounted to a plastic shopping basket. We gave each

person a two-minute briefing on how to use several key features of the PSA (recipes, the shopping cart, and how self-checkout would have worked if it were operational). Customers were then asked to go into the store, scan in items that they placed into their basket, and then check-out using the cashier, instead of the PSA system (since the check-out feature was not working at that time). Some testers chose not to purchase any items, and used the PSA only to get information about products and to view the contents of their virtual shopping cart.

Figure 3.18

Manny, a Stop & Shop customer, uses the PSA to scan in an item while shopping.

Since the system was a proof of concept, rather than a production-quality system, we chose to do informal ethnographic user testing rather than quantitative statistical testing. We followed customers around the store as they used the PSA and scanned items into the system. We observed how they scanned in items, as well as how much they used the system. Additionally, after they were finished using the PSA, we spent around 5-10 minutes with

each tester asking them for feedback about the system. Specifically, we asked users the following questions:

> *What benefits do you feel you get from using the Personal Shopping Assistant? What is the single most important benefit to you?*
>
> *How should we improve the Personal Shopping Assistant? What features would you add? What should we change?*
>
> *Would you use the Personal Shopping Assistant if it were offered in the store? (If the user answers in the affirmative) Under what circumstances would you use the Personal Shopping Assistant? (If the user answers in the negative) Why wouldn't you use the Personal Shopping Assistant? What should we do to improve it so that you would use it?*

From watching shoppers use the Personal Shopping Assistant, and from the responses to the questions we asked them, we learned three key points:

Customers find significant value from interacting with information that is relevant to them. More than half of the people we interviewed claimed that the single most important benefit to them was being able to view the total cost of all of the items that they scanned into the system. As one shopper stated, "I've always wanted to know how much money my groceries cost before I get in the checkout line. There have been a couple of times where I've gotten in the checkout stand only to find out that the total [of all of my grocery items] is more than the amount of money I have in my purse."

Furthermore, a majority of shoppers liked the recipe service, and said that they would find it useful for their weekly grocery trips where they would spend more time buying items. These users also said that if the system could "know" what items they brought in the past, and suggest recipes that used these items, that they would find the service even more useful. However, a few users said that the recipe service was not personally relevant to them (e.g.

they don't cook very much). As expected, they said that they would not find the recipe service useful. Shoppers found value in services they personally could relate to. Their perception of the system was heavily influenced by how much they could relate to the services that they interacted with.

Shoppers would use the system if they felt that it saved them time. Only one person mentioned saving time as one of the perceived benefits from using the system. In fact, four testers mentioned that the time it took to scan items in, coupled with the time they needed to read what the what was displayed, would be reasons why they might choose to *not* use the PSA. However, when we asked these four customers whether self-checkout would save them time, they answered that it would. When we pointed out that this was one key benefit of the system, all but one changed their mind about using the PSA.

Furthermore, almost everyone who said that they would use the PSA if it were available to them said that it was inappropriate for "short" or "quick" shopping trips. These people also said that they would only use the PSA for their longer shopping trips—if the PSA were mounted on a shopping cart instead of a basket. Again, these customers felt that the system would require too much time to use, and did not see a time benefit to using the system. However, when we described the self-checkout feature to them, all except one shopper changed their mind about using the system for short shopping trips.

The user interface layout and hardware design are as important than the services and information that customers interact with. While observing shoppers using the system, we noticed that many people had trouble trying to hold the shopping basket in one hand, and scanning in items with the other. One person set the basket on the ground in order to scan products. She could not find a way to hold the basket, as she needed one hand to operate the barcode scanner and the other to orient the item. Two customers actually put the PSA basket in a shopping cart, and wheeled the cart around (see Figure 3.20). Thus, they were able to free up both hands so they could scan items into the PSA. Furthermore, these two users gave a

much more favorable review of the system then the shoppers who carried the basket around with them.

Figure 3.19

Dan, a Stop & Shop customer, uses the PSA to scan in a package of chicken before putting it in his shopping basket.

Figure 3.20

Caroline, a shopper testing the PSA, found that holding the basket, orienting a product, and using the barcode scanner was too awkward. Her solution was to put the PSA basket into a shopping cart.

Of the users who said that they would not use the system, everyone pointed out that their difficulty in scanning items while holding the shopping basket was a key reason why they would choose not to use the it. Furthermore, every person who tested the system actually preferred to have the PSA client mounted to a shopping cart instead of a basket. More than half of the users who tested the system cited that the design of the basket as a suggestion for improving the PSA.

All but two of the shoppers who used the system claimed that the text on the screen was too small, and thus too difficult to read. One shopper said that while she could read the information on the Pocket PC screen, her mom and probably any other elderly person would be unable to use the system because the text was too small. Several shoppers suggested that using a device with a larger screen as a potential improvement to the PSA. Additionally, nearly every tester told us that the user interface was difficult to navigate around, and was not inherently obvious. Even after explaining how to use the shopping cart feature to customers, several people asked us how to view the contents in their shopping cart because the links were not inherently obvious. While many users eventually figured out the system, several users cited that they would only use the system in the future if the "buttons were bigger" and if "it wasn't so difficult to figure out how to see what is in my cart and to take items out of it."

3.4.3 Future Work. The Voyager Personal Shopping Assistant was a proof-of-concept system; creating a production solution requires much more work. While creating a production system was beyond the scope of our research, there are still many critical and interesting issues that must be addressed before the PSA is usable by the general public.

On a systems design level, the problem of security and customer privacy has not yet been solved. This, perhaps, was the primary reason why we didn't even attempt to integrate the PSA with the store's point of sale system or create our own mobile payment solution. We did not have the time or resources to design a system that properly ensures the integrity of

customer financial data. Furthermore, beyond posting a privacy policy, the system had no safeguards to ensure a customer's privacy. The system logged in perpetuity each item that the customer scanned. While the supermarket's use of that personal information may be benign, other organizations may be less honorable. Imagine, for example, a manufacturer requiring a customer to release some subset of their personal information before allowing the customer to access any services that it may provide. That manufacturer could potentially amass a huge database of what products the customer has scanned into the system, and as a result, send obtrusive advertisements to the user, either electronically or otherwise. Extrapolate this scenario to customers having to release sensitive information like social security numbers, phone numbers, and home addresses, and the entire concept of an Internet presence could fail simply because users perceive it to be insecure and an egregious violation of their personal privacy.

On the other hand, integrating the system with the different backend information systems was also a huge challenge for us. Many organizations run antiquated, proprietary information systems that were not meant for others to use. For example, Royal Ahold's pricing system in the Stop & Shop supermarket was originally implemented for use by the store's checkout stand only. Attempting to expose the system as a service that the PSA could query was a difficult and rather time-consuming task. The same was also true for Kraft's recipe database—integrating it into the PSA also was not easy. However, the benefits of exposing these services outweigh the effort required to do so. Kraft, on providing their recipe service, could encourage the development of other useful applications, including a home recipe suggestion system and a product guide that consumers could download onto their mobile devices for stores that did not have a PSA-like system available.

Accordingly, there are countless other services that could be added to the PSA system. Project Voyager has only scratched the surface. After testing the system, one person stated that he would like to use a service that provided electronic coupons, personalized discounts, and product promotions for a majority of products stocked in the supermarket. As a future service,

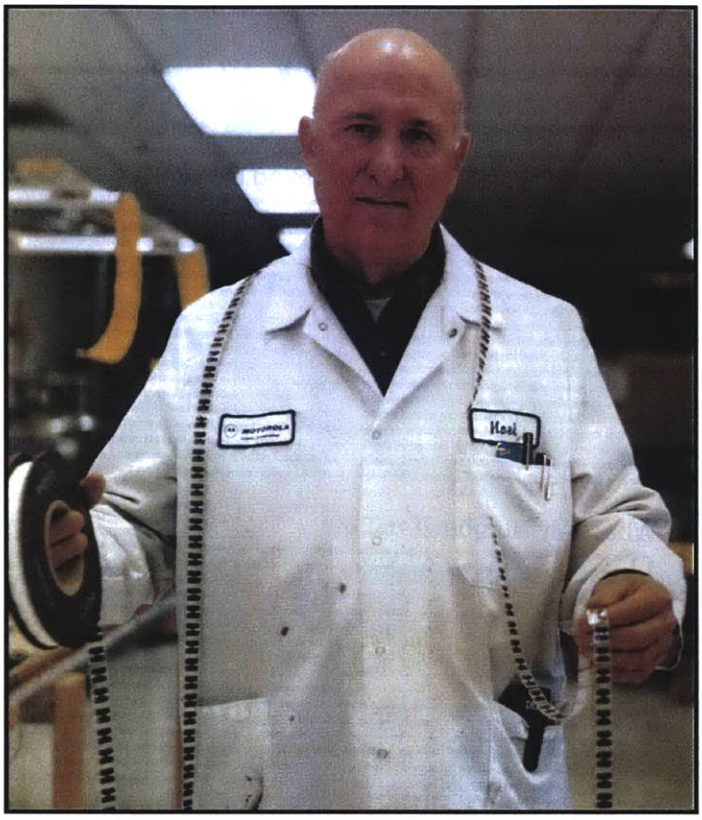
both the store and product manufacturers could provide a service that gave customers monetary discounts off certain items, or even tie two related items together—for example, if the user buys three boxes of macaroni and cheese, the manufacturer would offer the customer a free half gallon of milk.

However, these services need not exist solely to encourage customers to buy more products. Several subsets of the population would find different services useful. The American Heart Association, in conjunction with several large food manufacturers, for example could create a service or application that monitors the cholesterol content of all the items in the shopping cart. People with diabetes, for example, would find great value in a service or application that calculates the total amount of fat or sugar of the products that they have scanned in thus far.

Furthermore, as we discovered during user testing, we need to improve the design—both in terms of the hardware interface and graphical user interface—that the customer interacts with. From using more graphics to enlarging buttons, people who tested out the system unequivocally demanded a more friendly, easy to navigate system. We used off-the-shelf parts of our prototype and mounted them to a flimsy shopping basket. Customers gave us quite a bit of negative feedback because they found the system hard to use. Thus, a production system would require an entire redesign of the hardware and a graphical user interface to ensure that customers can learn and use the system with ease.

Finally, the PSA should be able to support new product identification standards and technologies as the industry begins adopting them. The MIT Auto-ID center, for example, is pushing their electronic Product Code standard. Motorola is aggressively marketing their BiStatix RFID technology to retailers and product manufacturers (see Figure 3.21). These technologies will become the de facto installation in stores and factories in the very near future. Consequently, the PSA should be able support and integrate these new technologies and standards as people and organizations begin adopting them.

Figure 3.20



Noel Eberhardt holds up a spool of Motorola's BiStatix RFID tags, which he helped to invent. BiStatix RFID tags have the potential to replace UPC barcodes as the *de facto* product identification mechanism. [Scho1]

4.0 Synthesis

4.1 Future Directions: Building an Enchanted Land

Tying services to people, places, and things is not confined only to a campus tour guide and a personal shopping assistant. This technology has the potential to radically change how we interact with other physical objects in the world we live in. Countless additional applications have yet to be built. The result will be simpler interfaces that create a vastly more animate world.

My MIT Campus Guide and Personal Shopping Assistant explore how people interact with services tied to locations and products. However, there are many more services I have yet to investigate that will change how people interact with other people, or how things interact with other things.

4.1.1 P2P: People to People Interactions. People-to-people interactions could be augmented by virtual services as well. Doctors will be instantly able to view your medical records online if you wind up in the hospital. Their ability to identify allergies to certain medications could potentially save your life. Carrying passports could become a thing of the past—a terminal senses you and digitally verifies your identity as you walk past an immigration officer at the airport. Voyager applications could even forever change the way people find companions: personal ads become digital services linked to your online identity. The concepts and technologies used to build the Personal Shopping Assistant and the Campus Guide could also enable these scenarios.

4.1.2 T2T: Things to Things Interactions. Things have Internet presences. Smart, networked appliances can interact with each other—and with the objects they encounter. Coffee makers, televisions, DVD players, and microwaves will all become "smarter." Your coffee maker senses your cup and dispenses coffee—the way you like it, with two lumps of sugar and a touch of cream—after retrieving your online profile. Your microwave detects a box of frozen vegetables and automatically cooks it for two minutes.

Your DVD player discovers that you're about to play *Terminator 2* and automatically dims the lights, lowers the volume on your sound system so that you don't disturb the neighbors, and turns on the TV. The concepts and technologies used to build the Campus Guide and the Personal Shopping Assistant could also enable these scenarios as well.

4.1.3 Security and Privacy. Security and privacy are still thorny issues. As briefly discussed in the previous chapter, mechanisms are needed to safeguard users and their personal information. People must be able to manage, or at least oversee, their public exposure. Whether to trust someone request access to your personal services is a matter for you to decide. These issues, however, are currently topics that other research efforts are addressing. Researchers from the Cooltown project at HP have proposed one solution [Kinoo, KBoo]. The .NET Framework is currently developing a security mechanism to digitally verify the identity and the trust category of any service that requests resources or sensitive information from another source. The MIT Auto-ID Center has listed privacy and security as one of its chief concerns and research topics on its web site [AID]. Thus, the challenge a project like Voyager is not to develop a separate privacy and security mechanism, but to identify and integrate the most appropriate mechanism for the applications one plans to build.

4.2 Conclusions: And They Lived Happily Ever After

Putting people, places, and things online will lead to exciting new interfaces. My MIT Campus Guide and Voyager Personal Shopping Assistant explore a bit of what has just become possible. Products taught users how to cook in a supermarket. Customers paid for their groceries with the click of a button. Artwork on the MIT Campus spoke to tourists. Even campus dining spots told you what they were serving that day. People who used our applications verified that interactions with their surroundings were enriched in new and meaningful ways.

W.I.S.E. (Wireless capillary nets, Internet presence platforms, Sensing devices, and Electronic services) technologies will enable these new interfaces. Customers retrieved virtual services using handheld computers connected to wireless capillary networks. Shoppers scanned in items using portable barcode scanners and RFID tag readers. Internet infrastructures like the .NET Framework enable the rapid development of electronic services. Finally, to build the PSA, we devised a method to tie services to physical objects. Other new technologies are on the horizon that will no doubt create an even more animate world.

Many of the scenarios I presented sounds like something out of a fairy tale. Allen Newell thought so, in a talk he gave in 1976:

> "Computer technology differs from all other technologies precisely in providing the capability for an enchanted world:
>
> For brakes that know how to stop on wet pavements.
>
> For instruments that can converse with their users.
>
> For bridges that watch out for the safety of those who cross them.
>
> For streetlights that care about those who stand under them—who know the way, so no one gets lost.
>
> For little boxes that make out your income tax for you.
>
> In short, computer technology offers the possibility of incorporating intelligent behavior in all the nooks and crannies of our world. With it we could build an enchanted land." [New]

Project Voyager brought Newell's enchanted land onto the MIT Campus and into the supermarket. Dormitories told you about their residents. Shopping carts counted calories and helped you eat healthier. But we have barely scratched the surface—we're just a bit past "Once upon a time." There are still so many more applications that can make our world more enchanted. "In fairy stories," Newell pondered, "there are great trials to be performed before the happy ending. Great dangers must be encountered and overcome." Indeed, we still have much work ahead of us. Challenges have yet to be overcome. People, places, and things still have much more to say. But I am confident that Project Voyager, along with the many other

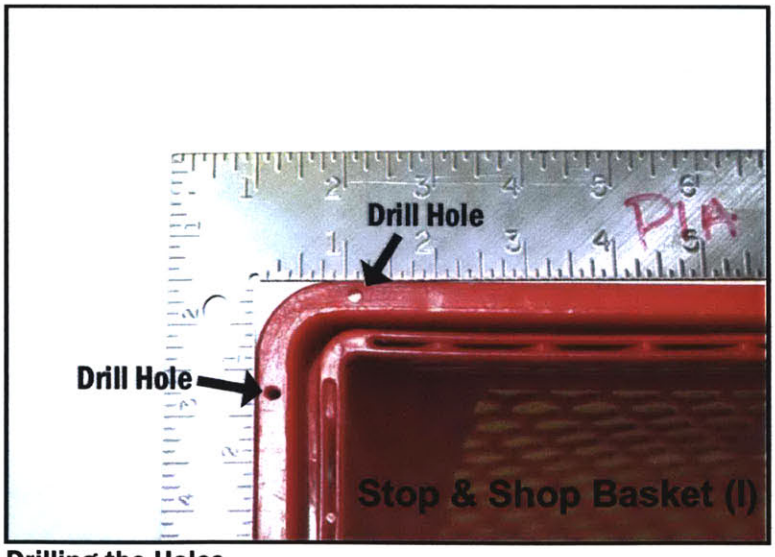
endeavors that are creating Internet presences for coffee makers, travelers, and automobiles, will bring us one step closer to the happy ending—the enchanted world that Newell spoke about over two decades ago.

Appendix A PSA Hardware Prototype Schematics

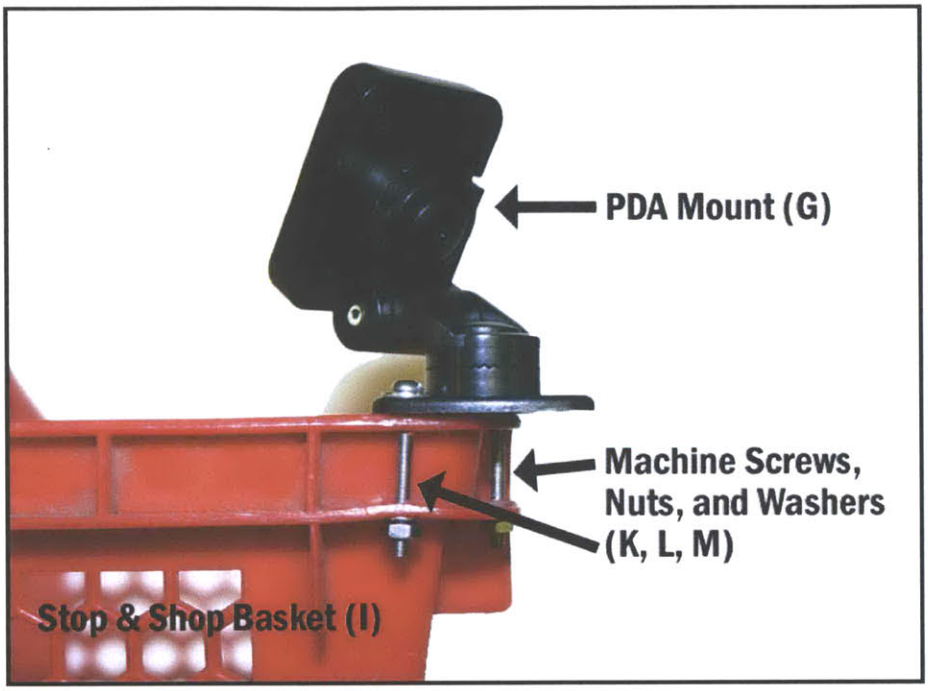
Bill of Materials

Ref	Description	Manufacturer Part Number	Quantity
A	Compaq iPaq H3650 Pocket PC StrongArm 206 Mhz Processor 32 MB RAM 240x320 Reflective LCD Color Display	170294-001	1
B	Compaq iPaq PC Card Sleeve	173396-001	1
C	Symbol Spectrum24 Wireless LAN PC Card. IEEE 802.11 Frequency Hopping, 2 Mbps	LA-3020-501-US	1
D	Symbol Technologies Key Fob Scanner with included Serial RS-232 Cable	CS1504	1
E	Compaq iPaq Serial Autosync Cable	191008-B21	1
F	Fellowes WriteRIGHT Microthin Screen Overlays for Compaq iPaq Pocket PC	842858	1
G	Arkon Industries Palm-Size Computer Multi-Angle Mount Kit	CM-228	1
H	The CLIP Company Swivel Mount	N/A	1
I	Stop & Shop Plastic Shopping Basket	N/A	1
K	1 1/2 Inch Machine Screws	N/A	2
L	Machine Nuts	N/A	2
M	Lock Washers	N/A	2

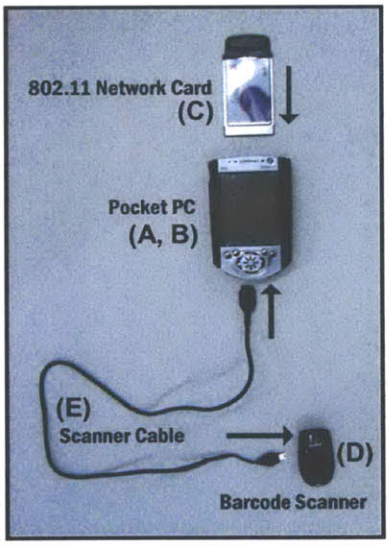
Schematics and Photographs



Drilling the Holes



Attaching the PDA Mount to the Basket

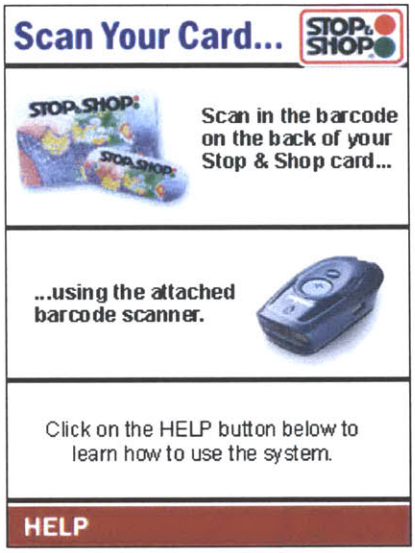


Assembling the Pocket PC Scanner

The Finished Prototype

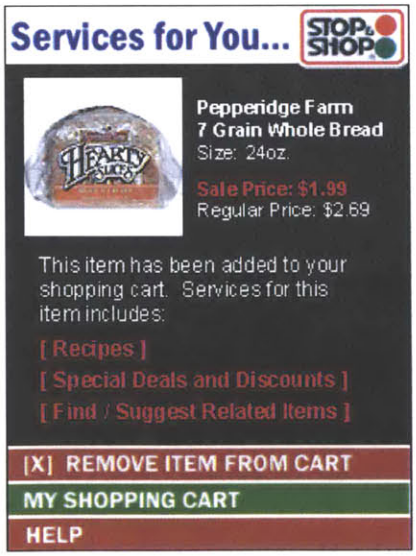
Appendix B Voyager PSA Application Screenshots

The figures below include several screenshots of the first version of the Voyager PSA Application. Several supermarket customers tested the system while shopping at the Stop & Shop Supermarket in Hingham, MA between May 3 – 14, 2001.

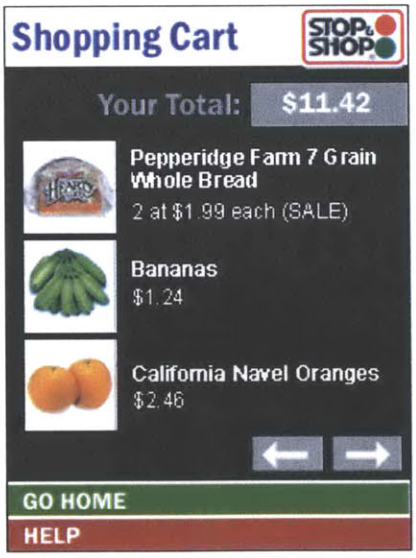


Login / Startup Screen

Home Screen

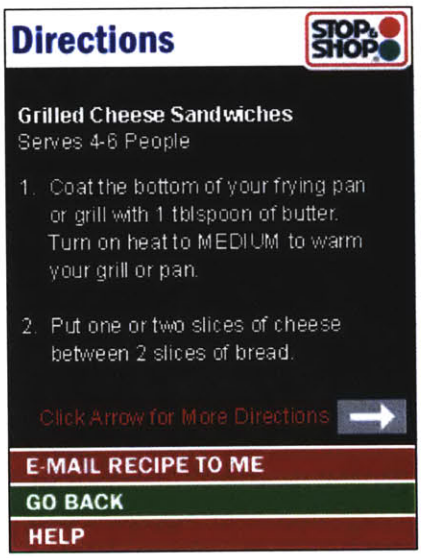


Product Services Screen



Shopping Cart Screen

Recipes – Ingredients Screen



Recipes – Directions Screen

References

[Box00] D. Box, et. al. Simple Object Access Protocol specifications. 2000. http://msdn.microsoft.com/xml/general/soapspec.asp

[BT] Bluetooth SIG Website. http://www.bluetooth.com

[Cat] Catalina Marketing Corporation Website. http://www.catmkt.com

[iPaq] Compaq Computer Corporation. Compaq iPaq H3600 Series Pocket PC Datasheet. http://www.compaq.com/products/handhelds/pocketpc/

[Sym] CS 1504 Barcode Scanner Datasheet, Symbol Technologies. http://www.symbol.com/products/consumer_systems/consumer_cs1504.html

[Deb00] M. Debski. "The SNAP! Toolkit for Developing Sensor Networks and Its Application to Cross Country Skiing." Master's Thesis, Department of Electrical Engineering and Computer Science. Massachusetts Institute of Technology. May 2000.

[ES00] D. Engels and S. Sarma. "Automatic Identification using RFID." MIT Auto-ID Center Draft Paper. September 8, 2000.

[HRF] Home RF Website. http://www.homerf.org

[IEEE] IEEE 802.11b Wireless Ethernet Tutorial. Wireless Ethernet Compatability Alliance. http://www.wi-fi.net/downloads/IEEE_80211_Primer.pdf

[NET] An Introduction to Microsoft .NET, Microsoft Corporation, 2001 http://www.microsoft.com/net/intro.asp

[JINI] JINI Architectural Overview. Sun Microsystems, Inc. 1999.

[Yos00] Y. Joshi. "Information Visibility and Its Effect On Supply Chain Dynamics." Master's Thesis, Department of Mechanical Engineering, Massachusetts Institute of Technology. June 2000.

[KB00] T. Kindberg and J. Barton. "A Web-Based Nomadic Computing System." Hewlett-Packard Laboratories White Paper. 2000.

[Kin00] T. Kindberg, et al. "People, Places, and Things: Web Presence for the Real World". Hewlett-Packard Laboratories White Paper. 2000.

[Kft] Kraft Corporation Website. http://www.kraft.com

[Lee97] T. Lee, et. all. RFC 2068: Hypertext Transfer Protocol – HTTP/1.1. January 1997.

[AID] MIT Auto-ID Center. "The Networked Physical World." White Paper (MIT-AUTOID-WH-001). December 2000

[Mot] Motorola Corporation, RFID and SmartCard Division. http://www.motorola.com/smartcard/

[New] A. Newell. "Fairy Tales" AI Magazine. 13(4). Winter 1992. pp. 46-48. This is a published text of a talk that Newell gave in 1976.

[NM] NMEA 0183 v3.0 Interface Document. National Electronics Marine Association, 2000.

[RA] Personal Shopping Assistant Concept Test: Consumer & Competitive Insights. Royal Ahold Marketing Report. December 22, 2000.

[PPC] Pocket PC Platform Features. Microsoft Corporation, 2000. http://www.microsoft.com/mobile/pocketpc/bguide/features/default.asp

[Pra00] S. Pradhan. "Semantic Location." Hewlett-Packard Laboratories White Paper. 2000.

[Reh00] M. Rehaut, et al. "Wireless Internet: The Path to Ubiquity." Solomon Smith Barney Internet Strategy Report. August 23, 2000.

[Sch01] C. Schmidt. "Beyond the Bar Code." *Technology Review.* Vol. 104, No. 2. March 2001. pp. 80-85

[UCC] Universal Product Code Guidelines, Uniform Code Council. http://www.uc-council.org/reflib/00810/index.html

[UPnP] Universal Plug and Play Device Architecture. Microsoft Corporation, June 8, 2000.

[WH01] Z. D. Wigder, E. Horowitz, D. Brooks, and S. McAteer. "Preparing for the Year in Wireless: Outlook for the Year Ahead." Jupiter Communications Concept Report. January 2001.