

# Pivot: Automatically Offering Information and Services to Real-World Shoppers

Nathan Nichols, Kristian J. Hammond, Larry Birnbaum, Lisa Gandy

Northwestern University

Intelligent Information Laboratory

{ndnichols, hammond, birnbaum}@cs.northwestern.edu, nwu-lmg215@northwestern.edu

## Abstract

Shoppers with an internet-enabled computer have a wealth of product information available to them. By browsing to a variety of websites, users can conduct searches and compare prices, read reviews, and learn more about a product. These sites are pivot points for a user; once they are at Amazon.com's landing page, for example, they can navigate outwards to a million different products. The idiom of browsing to a central page for a site and then navigating outwards is acceptable when browsing is convenient, with large displays and useful input devices. This process becomes inconvenient, however, when the user is out and about in the world. We have built a system, Pivot, that uses physical objects as pivot points for the user. Specifically, Pivot uses the 1-D barcodes present on every product to deliver powerful services and options to a user on his or her cellphone. These services are chosen to be most useful to a user in the moment and trying to make a purchase decision. This paper describes the motivations for the system, the system itself, its current real-world deployment, and our intended future work.

**Key Words-** M-Commerce, Mobile applications user interface and design, Mobile Search engines

## 1. Introduction

Shopping for products online is a popular activity [3] and only becoming more so. Online shopping typically involves two different tasks: choosing amongst a wide variety of items, and then acquiring the chosen item. A user at his or her computer can visit a variety of sites to help them choose amongst products. For example, they could visit NetFlix to get DVD recommendations, or Metacritic to learn about new movie releases. For book suggestions, they could browse Amazon's selection, or look at their friends' favorite books on Facebook. They can listen to thirty-second snippets of new songs on iTunes, or watch the video for a new music single on YouTube. This process typically involves going to the landing page for a major website and then navigating

within that website.

Once a user has chosen a particular item to buy, they often visit a new set of sites to actually do the purchasing. They may visit popular e-commerce sites like Amazon or Buy.com, or they may go to the website of traditional brick and mortar stores, like Best Buy or Target. They may also go to a physical store to buy the item they choose. Again, this interaction typically involves navigating to a major website's landing page, typing in the name of the item into a search box, and then processing the resulting information.

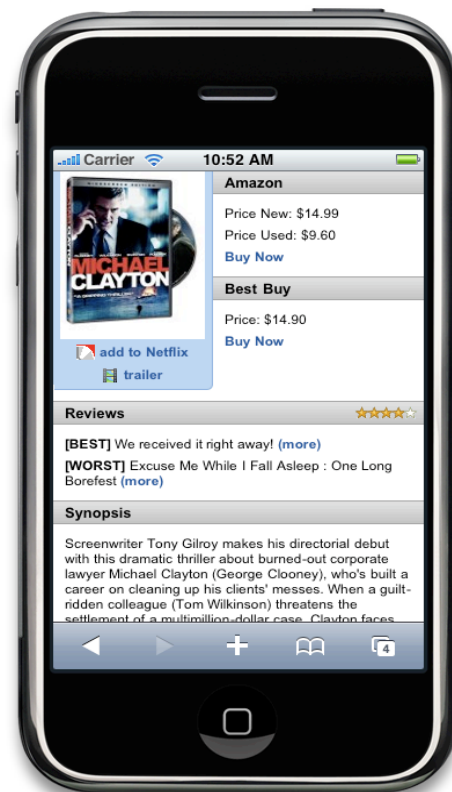


Figure 1: A screenshot of the current Pivot client

This shopping idiom--visiting a set of sites to choose an item, visiting a separate set to do the price comparisons and actual purchasing--works acceptably when a user is at a PC or laptop. Unfortunately, it fails badly when a user is out and about in the world. Although almost everyone carries a cellphone [2] and more of these phones are capable of fully using the web, browsing on the phone is still a more difficult process than it is on a PC. Phones have limited bandwidth, small screens, and small keyboards; it is difficult to imagine a shopper at Best Buy taking the time to pull out their cellphone, load the browser, go to Amazon.com, wait for it to load, peck in the name of the DVD in their hand, wait for the search page to load, and then finally compare prices. There is simply too much friction involved.

Fortunately, there is an alternative shopping idiom that works well when people are browsing in a physical store. First, for many products, there is no need to visit websites to choose one of ten million potential items to purchase; if a user is already in a store, they can acquire very basic information about a large number of products quickly. Although a shopper at Best Buy doesn't have access to the advanced search capabilities available at a site like Amazon.com, they can actually see the product, they can examine it, they can talk to the friends there with them, etc. Shoppers at a retail store can quickly narrow down the three-hundred video games in stock to just two or three that they may be interested in buying. This greatly simplifies the first shopping task of choosing an interesting item: people browsing Amazon.com's video game section have questions like "What game should I get?" while a shopper at a brick and mortar Best Buy has questions like "Should I buy the shooter game in my right hand or the one in my left?"

The second shopping task--choosing where and how to purchase an item--can also be made easier. Users don't need to visit an assortment of sites and type the same product name into every search box. Because every product has a unique ID, the barcode, a system can use that unambiguous attribute to provide the user with a variety of purchasing options and services. Rather than forcing the user to browse to an array of sites to find out information about the same item, we can use that item to present the user with information from an array of sites.

Pivot is designed to assist with both of these *item selection* and *item acquirement* tasks. Using the product's UPC code, it can quickly and painlessly present the user with a plethora of information and services to help the user decide if she wants to own a certain product; if she decides Yes, Pivot then has a variety of options to help the user acquire that item.

## 2. System

There are two core components to the Pivot system: the Pivot server and a Pivot client. The server is designed to support a wide variety of clients; we intend to have an

iPhone client, an Android client, a Symbian client, etc. This paper will mainly deal with the server component, but will also touch on the iPhone-capable browser-based client we have built so far. The Pivot server is designed to support a wide variety of input and output interfaces, so the core of the system takes a UPC value and optional user id, and returns XML. The server itself is agnostic as to whether the UPC value came from an image or a user typing it in manually, and the XML can be formatted to power an AJAX-based webpage or used to drive native UI elements on a client version.

When the server receives a UPC value (currently we support the major formats prevalent in the U.S., namely UPC-A, UPC-C, EAN, and ISBN), it first determines the object's title and type by consulting its internal UPC databases and by using Amazon's web service. Each type of object uses a subset of the *information providers* we have built. For example, a music CD might use the Amazon.com price, BestBuy.com price, and iTunes information providers. Each provider returns its data to the user asynchronously; as soon as a provider is ready with its information, it is sent back to the client as XML.

This architecture has a number of benefits. First, it is simple to add new information providers. Each provider uses only the item's basic information, and is a separate programmatic unit with no dependencies on other providers. For example, if we wanted Pivot to search eBay for an item, we would simply have to write an information provider that searched for the item and returned results from the eBay API, and then update the item's configuration to include the new provider.

Second, this architecture is flexible and highly-configurable. Although we currently use the same information providers for every object of a certain type, it would be straightforward to allow users to customize which information providers they are interested in; a particularly frugal user might be interested in always knowing what a used version of an item costs, for example. We could also alter the providers based on more specific attributes of the object; we could have more information about why an R-rated DVD got its rating, for example, but not bother showing that for G-rated movies.

Third, this architecture is able to quickly begin returning results to a user. Because each information provider returns its own data to the client, the user never has to wait for a slow provider to begin receiving information. The system is designed to be used in a store, with the user standing and people milling past, and needs to begin returning useful information as quickly as possible. The backend has also been designed to respond quickly and aggressively caches previous responses, etc.

## 3. Information providers

The Pivot system currently supports four types of items: books, CDs, DVDs, and video games. For these four types of objects, we have built 14 different

information providers, a subset of which is used by each type. All of these providers are implemented and running in the deployed version of Pivot, discussed in detail later.

**Table 1: A table relating item types and information providers**

	Image	Synopsis	Best Buy Price	Best Buy Location	Amazon New Price	Amazon Used Price	Amazon Reviews	Metacritic Reviews	B & N New Price	B & N Used Price	iTunes	Trailer	Social networks	Netflix
DVDs	✓	✓	✓	✓	✓	✓	✓	✓				✓	✓	✓
Books	✓	✓			✓	✓	✓		✓	✓			✓	
CDs	✓	✓	✓	✓	✓	✓	✓	✓			✓		✓	
Video games	✓	✓	✓	✓	✓	✓	✓	✓				✓		

### 3.1. Image provider

For each item type, Pivot always provides a picture of the item. This is mainly done to assure the user that the system is providing information about the correct product. It can also be used by Pivot clients to maintain a visual history of scanned items.

### 3.2 Synopsis provider

Every item also gets a synopsis. Unfortunately, there isn't one good resource for descriptions of the four current types of items, so the system goes to different sources based on the type of the item to find the synopses. For DVDs, the system goes to Netflix and gets information like a short summary of the movie, the rating, and the actors and director. For video games, Pivot goes to Barnes & Noble and gets a summary, the ESRB rating, and the genre; the system also goes to Barnes & Noble for book synopses. For CDs, the system goes to Wikipedia and returns the first paragraph for the album's page; if the album itself doesn't have a page, the system returns the page for the artist. **Use case:** Mary is at a Blockbuster and considering renting the film Blue State; she sees it is rated R and wonders if it's appropriate for her kids. She enters the barcode and reads in the synopsis that it is rated R for language, and feels comfortable renting it.

### 3.3 Best Buy price provider

For DVDs, CDs, and video games, Pivot goes to BestBuy.com and searches in the appropriate category for the item. If the item is available online, the provider returns XML with the price and a URL to purchase the item. If the user follows the link, they are taken to the same Best Buy page as if they had clicked the Add to Cart

from that item's BestBuy.com page. From that page, they can checkout and order the item; Pivot itself doesn't handle any of the transaction, it just makes it much simpler for the user to order it through BestBuy.com. **Use case:** Mark is at a Wal\*Mart and considering purchasing the new Rihanna CD. He scans the barcode, and sees that it is on sale this week at BestBuy.com; he follows Pivot's BestBuy.com link and orders the item from his phone.

### 3.4 Best Buy location provider

For DVDs, CDs, and video games Pivot also tries to find a nearby Best Buy that has that item in stock. Using the user's zip code (either entered by hand, or automatically calculated and posted by an Pivot client) the system scrapes BestBuy.com and returns the addresses of nearby Best Buys that have the item in stock. These addresses can either be simply displayed to the user or a client can place them on a map. **Use case:** Mike is at a Circuit City and is about to buy Call of Duty 4. He scans the barcode, and sees that it is five dollars cheaper at the Best Buy just around the corner. He decides to walk over there and get it instead.

### 3.5 Amazon new and used price provider

For all items, Pivot returns a link to buy the item new on Amazon (assuming the item is in stock) and the item's price. If a user follows the link, the item is added to their Amazon shopping cart and they are redirected to a checkout. Because we are an Amazon Associate, we receive a small percentage of the price for any item purchased through Pivot. This information provider also gives the price of the item and a link to buy it used. **Use case:** Pam is browsing at a Borders bookstore. She sees a new vegetarian cookbook, and thinks a lot of the recipes look good. She scans the book's barcode, and sees that it's eight dollars cheaper on Amazon.com. She decides she can wait to own the book, and orders it through Amazon; it arrives at her home two days later.

### 3.6 Amazon reviews

For all items, Pivot returns two customer reviews from Amazon, along with a link to read all of the reviews. To ensure good coverage of the reviews (and not just show two similar positive or negative reviews) the system shows the most positive and most negative helpful reviews (helpful reviews have been explicitly marked as such by other Amazon shoppers.) **Use case:** Jan is interested in Alexander the Great, but doesn't have much of a background in Greek history. She sees a book about Alexander that looks interesting, but is worried that it might be too historical. She loads the Pivot page for the book, and sees that one of the Amazon reviewers said that it was a "great introduction to the life of Alexander." She is reassured and decides to purchase the book.

### 3.7 Metacritic reviews

For DVDs, CDs, and video games, Pivot goes to Metacritic, an aggregation site for reviews. While Amazon's reviews give a good sense of what normal customers think about an item, Metacritic has professional reviewers' opinions. Like the Amazon customer reviews, Pivot returns the most positive and most negative reviews to help provide good review coverage. **Use case:** Tom is trying to rent a DVD. He enjoyed the first Elizabeth movie, but hadn't heard much about the sequel, Elizabeth: The Golden Age. He scans its barcode, and sees that film critics gave it much lower scores than the first one. He decides to pass in lieu of a different movie.

### 3.8 Barnes & Noble new and used price provider

For books, the system returns the new and used price when purchased at barnesandnoble.com. There are also links to purchase either version; like the Best Buy price provider, when these links are followed users are redirected to the Barnes & Noble site where the item has just been added to their cart. **Use case:** Susan is at a Barnes & Noble's bookstore and sees a new paperback novel she is interested in. She scans it, and sees there is a used version of the book available at barnesandnoble.com for half the price. She decides she doesn't need it to be brand new, and orders a used copy through her phone.

### 3.9 iTunes

For CDs, this information provider returns a link to the iTunes page for the album. For a client running on the iPhone, following this link will open the iTunes store to that album; from within the iTunes store, the user can listen to samples from the album or buy and download it immediately. **Use case:** Steve is at a Best Buy and sees that Willie Nelson has a new album out; Steve likes Willie Nelson's older albums, but hasn't heard much about the new one. He takes a picture of the barcode with his iPhone's camera and hits the iTunes link. His phone launches the iTunes mobile application, and he listens to snippets of a few of the tracks. He likes what he hears, and the album is significantly cheaper on iTunes (\$9.99) than it is at the Best Buy (\$15.99). He buys the album on his phone, and it's downloaded while he continues shopping; he plugs it into his car's stereo and enjoys the new album as he drives home.

### 3.10 Trailers

For DVDs and video games, this provider returns a link to a YouTube trailer for the item. The system uses the YouTube API to search for the name of the item and "trailer," and then returns the most-watched video that contains those terms. Often there will be multiple versions of the same trailer with varying quality uploaded

to YouTube and choosing the most-watched lets the system use the wisdom of YouTube's crowd to help us choose the best. **Use case:** Shelby is browsing the DVD section at Target and sees a copy of Princess Mononoke. She doesn't normally like anime, but she remembered her friends saying that she would love Mononoke's art style. She gets out her iPhone, enters the UPC code, and follows the YouTube link. She watches the first half of the trailer there in the store, and likes the style of the movie so much that she decides to buy it.

### 3.11 Social networks

One of the best indicators of whether or not you are interested in an item is if you have friends that like that item. However, knowing what your friends like can be difficult to determine. Unless they have explicitly mentioned liking something in the past, you're certainly not going to call everyone of your friends to see if they happen to have an opinion on an object you are considering purchasing. Fortunately, social networks are now beginning to make this kind of information programmatically accessible.

Our social network information provider currently uses Facebook, although we hope to also support Last.FM and other social networks soon. The idea here is that when a user enters a UPC code, this provider checks to see any of the user's friends like that item. For Facebook, the first thing the user does is sign up for the Pivot Facebook application. This is a straightforward (it simply requires using your phone's browser to enter your Facebook account and password) and only needs to be done once; once the user is authenticated through Facebook, we drop a cookie on the phone to allow us access in the future. (Pivot respects our user's privacy, so if the user uninstalls the application we can no longer access their friends' profiles.)

Once this one-time process is complete, every time a user searches for an item, the provider checks to see if any of the user's friends like that item. This check is a two-step process. The first step is generating different possible versions of the item's title. Because Facebook's Favorite Movies, Favorite TV, Favorite Books, and Favorite Music sections are all free-text, users don't often enter an item's full title. For example, many users have "Star Wars" as a favorite movie, but few have "Star Wars: Episode 4: A New Hope". Furthermore, the Favorite Music section typically contains both artist and album names, and the Favorite Books section has both writers and books. So the first step generates five or so different variations of a title by stripping stop words, splitting on colons, using the artist or title, etc. The second step is seeing which of a user's friends have any of those titles listed in the appropriate Favorites section. The system generates a large FQL (Facebook Query Language) query and sends it to the Facebook API. This query returns a list of friend-name/matched-favorite pairs. The social network information provider returns a list of friends who

like that album, for the Pivot client to display how it wishes. **Use case:** Amy is browsing the CD selection at a Best Buy, and sees that the new Kanye West album is on sale. She knows that West is a popular artist and she likes the single on the album, but she doesn't know if she would enjoy the whole CD enough to buy it. She enters the UPC code onto her phone, and sees that three of her friends from high school have listed that album as one of their favorites. She takes her friends' recommendation and purchases the album.

### 3.12 NetFlix

NetFlix is a popular website that allows people to rent movies online. When a user enters the barcode for a DVD, this provider does a search for it on NetFlix, and returns a link to add the movie to their queue. If the user clicks that link, one of two things happen. If the user has used their phone to sign into NetFlix before, then the existing NetFlix cookie is used and the DVD is added to their queue; after scanning a DVD's barcode, it's literally a single click for a user to add it to their NetFlix queue. If the user has not signed onto NetFlix from their phone before, they are simply redirected to the NetFlix sign-in page. Once entering their name and password, the item is added to the queue. **Use case:** Kent is visiting an out-of-state friend's apartment. The friend owns "2001: A Space Odyssey," a movie that Kent has never seen before. The friend is willing to let Kent borrow it, but Kent doesn't want to hassle with having to mail the DVD back to his friend. Instead, he enters the barcode into his phone, and adds the DVD to his NetFlix queue. It's waiting in his mailbox by the time he gets back home from his trip.

## 4. Deployment

The Pivot server is currently up and running, and we are in the process of finding and fixing bugs, ensuring the relevancy of the reviews and trailers, etc. On the Pivot client side, we have a browser-based client available at <http://www.PivotApp.com>. Although this client is usable from any normal web browser, it is specially designed to be used with an Apple iPhone. All of the UI is based around a 320x240 screen, and the Best Buy location, iTunes store, and YouTube trailers all launch the native iPhone applications.

We developed a web-based client first because we could do it quickly and it is convenient to demonstrate. Users don't have to install a whole application to their phone to see what Pivot can do; they can simply try it in a browser with no download necessary. However, because this client runs in a browser it has limitations that a native client would not. First, it doesn't have actual barcode recognition; instead, the user has to type the UPC number into a text box. Second, we can't access information on the phone, like its GPS coordinates. If the user enters their zip code, we can show nearby stores, but they have

to take that additional step. Finally, the system can only be used when online; it would be nice to allow the user to capture a barcode even if they are in an area without service, and then interact with it later when they have service again.

Because of these limitations, we are currently in the process of building a native client for the iPhone, with other clients to come in the future. This version could use the iPhone camera to capture and decode barcode images, exploit the GPS to know the user's zip code without them explicitly entering it, use native iPhone UI elements like buttons and sliders, and allow for easy offline use. Unfortunately, early experiments with decoding barcode images taken with the iPhone have not been encouraging. Because the camera lacks a macro lens, the barcodes themselves are often out of focus and blurry. We are still hopeful about getting decoding to work on the phone, however, and we intend on having a Pivot native iPhone app ready to launch when the iPhone App Store becomes available in June.

We are also further developing the Pivot server in three different ways. The first is the architecture of the server itself; for example, we hope to support user-customization of information providers. If the user doesn't have a NetFlix account, then giving them a link to add a DVD to their queue doesn't help them. The second direction is expanding the list of information providers we have for our currently supported items. We'd like to use your Last.FM friends and neighbors to recommend CDs for example, or allow people to see prices on items from Half.com. Finally, we hope to increase the number of supported items; for example, over-the-counter medication and wine.

We have performed a small user-study of the current deployment of Pivot, both to gauge users' reactions and to get a quantitative sense of how convenient and useful it really is. The study had four users; all who used the internet on a daily basis, and two who owned iPhones. Each participant did the same task twice, once on a desktop computer and once on an iPhone; the task was to find all of the information that Pivot finds for a certain music CD. The participants took an average of four minutes to find all the information--prices on Amazon, BestBuy.com, and iTunes, and reviews from Amazon and Metacritic--on a desktop PC. It took the experienced iPhone users about five and a half minutes to find the same information on the iPhone; the two iPhone novices took seven and ten minutes to find the same information. Using Pivot, gathering all of this information only takes between ten and fifteen seconds. We also asked the four participants a few questions. All agreed that this information could be useful when shopping in a store, and said they would be much more likely to use this information if it were easier to access (as it is in the Pivot application.) Finally, all four said they would be interested in using the Pivot application, especially if it were free (which it is.)

## 5. Related work

Other systems, broadly classified under the “m-commerce” umbrella, have explored the role of cellphones and barcodes in the realm of on- and off-line commerce. However although there has been an encouraging amount of interest from potential users [7], none of these systems have caught on nor are used by an appreciable amount of people in the United States. Outside the U.S., especially Japan and parts of Europe, 2-D QR codes have more mainstream acceptance and are often used to direct shoppers to webpages or similar basic functionality; even in these places, though, there is no system that is regularly used to bridge the on- and off-line shopping divide. With Pivot, we have approached the problem from a novel perspective that we hope will allow it to gain more widespread adoption.

The first difference in Pivot is that our emphasis is on the server and not on the client. Other research teams have made great strides in both 1-D and 2-D barcode recognition[1][4]; with their advances in image processing and decoding, and corresponding increases in cellphone power and camera quality, it seems clear that in a year or two, millions of Americans will have cellphones capable of quickly and accurately decoding the ubiquitous 1-D barcode. Unfortunately, there is currently absolutely no reason for the average user to do so. We consider the barcode decoding problem as an essentially solved one, and instead began building the computational and information-processing backend that would motivate the decoding in the first place.

Another common issue with m-commerce systems is that they use only custom-created content; for example, they may rely on hand-annotated information encoded in QR codes [5][6]. This creates a classic “chicken and egg” problem; shoppers don’t use the QR codes because they’re unfamiliar or don’t often contain useful information, and manufacturers and publishers don’t take the time to encode useful information in the QR codes because shoppers don’t use them. In contrast, Pivot is able to automatically populate a universe of information and services for a variety of products, by collecting and transforming already existing, freely available online content.

## 6. Conclusions

Shopping online is a common activity, and there is a correspondingly vast amount of resources available to an online shopper. Many of these resources are designed to help a user choose amongst millions of possible purchases, and almost all begin with a user typing an item into a search box. In this way, users can read professional and customer reviews, see recommendations, watch trailers or listen to samples. Once the user has selected one of the millions of items, they begin the process of deciding where and how to purchase the item. This

typically involves another round of visiting sites and another round of typing an item’s title into search boxes.

Done like this, neither of these two tasks--item selection and item acquirement--work well on a cellphone. Although cellphones are quickly advancing in bandwidth, input abilities and screen-size, it still takes much more time to acquire and process this kind of information on a phone than it does on a PC.

To make the cellphone useful in real-world shopping, we have to rethink how these two tasks can work when users are out and about in the world. First, users don’t need to choose amongst millions of items when they are already at a store. They can use their five senses to quickly winnow down the 1,000 CDs that are for sale to the one or two they may be interested in purchasing. Because the user has the item--and its unique identifier--in their hand, we can move the focus off of the website and its search boxes and onto the item itself, bringing a constellation of useful services and information to the user quickly and easily. This same approach of moving the locus of the experience off of the search box and onto the item makes acquiring the item equally easy. Pivot can let you rent an item, buy it new, buy it used, buy a digital version, buy it online, or buy it in a brick and mortar store without you ever typing anything into a search box.

We are very excited about the future of this technology, and we think that this approach could eventually make the cellphone an indispensable part of shopping. As we continue to improve the robustness and capabilities of the Pivot server, and release a variety of powerful Pivot native clients, we hope for further adoption in the future.

## 7. References

- [1] Adelman, R. Mobile Phone Based Interaction with Everyday Products--On the Go, NGMast 2007.
- [2] CTIA - The Wireless Association. <http://www.ctia.org>, 2008.
- [3] Cyber Monday Online Retail Spending Hits Record \$733 Million, Up 21 Percent Versus Last Year, <http://www.comscore.com/press/release.asp?press=1921>, 2008.
- [4] Ohbuchi, E., Hanaizumi, H. Hack, L. A. Barcode Readers using the Camera Device in Mobile Phones, 2004 International Conference on Cyberworlds, 10.1109/CW.2004.23
- [5] Pohjanheimo, L. Keranen, H., Ailisto, H. Implementing TouchMe Paradigm with a Mobile Phone. Joint sOc-EUSAI, 2005.
- [6] Scornavacca, E., Barnes, S. J. Barcode Applications for M-Business, *Unwired Businesses: Cases in Mobile Business*, 2005.
- [7] Vartikovski, L., Kao, H. eBay Mobile: Mobile Shopping Application for Comparison on eBay. <http://hci.stanford.edu/srk/cs377a-mobile/project/final/kao-vartikovski.doc>, 2008