How to come up with the right set of categories? I made "user_extensible_categorization" a settable option. If set, users of the classified service in question can type in arbitrary new categories that then become available as menu options for everyone else.

CLASSIFIEDS IDEA: AUTOMATIC MANAGEMENT

I don't want to spend the rest of my life managing my classified ad system. Users post ads using forms. Users edit and delete their obsolete ads by typing in a password. If the password isn't the same as what they typed when they posted the ad, my software offers to e-mail it to them.

How well does this work? My system has handled about 4,100 ads so far. I've gotten about ten e-mail messages asking me to edit ads. In all but one or two cases, the user was able to edit his own ad after I pointed him to the forms.

CLASSIFIEDS IDEA: E-MAIL ALERTS

With only a few hundred current ads in my database, I have to do something to distinguish myself from rec.photo.marketplace with its 1,000 current ads at any time. Starting a couple of years ago, I offered to mail out e-mail alerts to people who were interested in seeing all the ads, just ads in a particular category, or just ads that match a query string (according to PLS). There are 205 people currently requesting alerts (options are daily, Monday/Thursday, and weekly as with the RemindMe system).

CLASSIFIEDS IDEA: SHOW OFF NEW ADS

One-line summaries of the newest ads should be displayed on the very first page of the system. Users shouldn't have to click to see ads.

CLASSIFIEDS IDEA: AUCTION

One thing that Web classifieds can do that newspaper classifieds can't is turn into an auction. Users of my system can choose at posting time whether or not they'd like Philip's Classifieds to record bids. A bid is instantly e-mailed to the seller and is also displayed underneath the ad for subsequent displays (see Figure 13.16).

STEPS 1–4

My classified system is a bit too complicated to describe in a book. However, I think it is worth showing off a few points, starting with the cover page (see Figures 13.17 and 13.18).

Oracle Exhibit 1011
Page 140 of 171

Cannon EOS 300 2.8L Pristine Cond.$3800 B/O

advertised 12 days ago in the photo.net Classifieds by EricZ@EZarakov.XO.Com (Eric Zarakov)

Barely used EOS 300 2.8L lens. This is literally a flawless lens that I bought new and used just a few times. Rather than sit in my closet, it belongs in the hands of someone who has a need for it, wants a new lens and is looking for a bargin.

Place a bid | Reply privately to EricZ@EZarakov.XO.Com

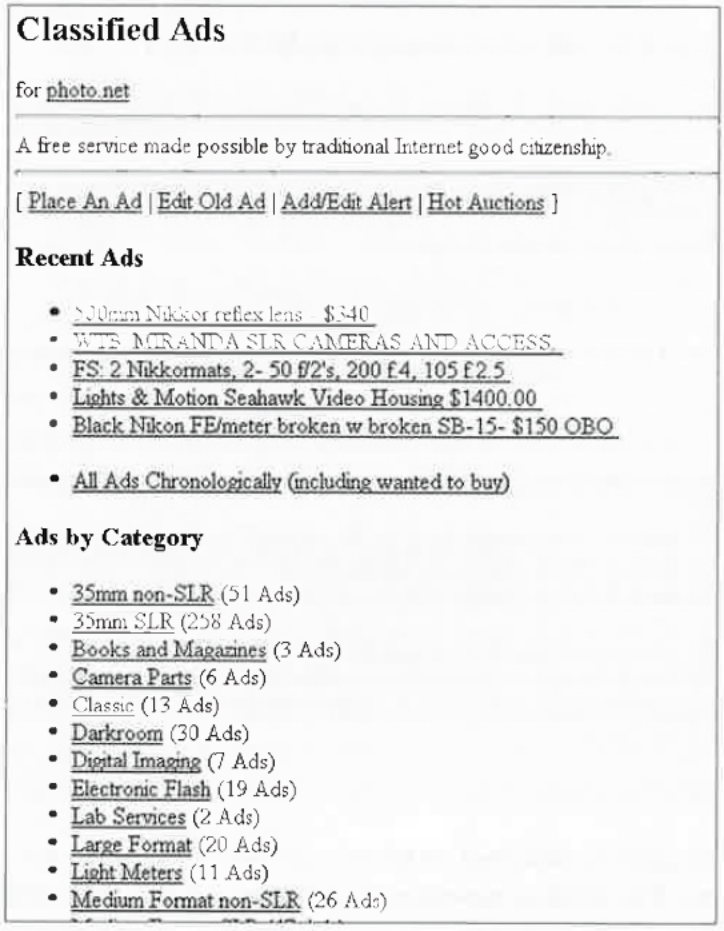
Note: if you place a bid using the whizzy auction software, email will be sent immediately to EricZ@EZarakov.XO.Com notifying him/her of the new bid.

Bids

- 2750.00 US dollars bid by Majik1187@aol.com (Mathew Joseph) at 1997-03-11 23:37:10 in Yonkers,NY
- 2600.00 US dollars bid by homefs@cei.net (Lee Moore) at 1997-03-11 22:48:01 in Fort Smith
- 2500.00 US dollars bid by billyy@worldnet.att.net (Bill McCloe) at 1997-03-10 10:40:16 in New York
- 2400.00 US dollars bid by mer@mail.odyssey.net (John Merriman) at 1997-03-08 23:19:51 in New York
- 2300.00 US dollars bid by billyy@worldnet.att.net (Bill McCloe) at 1997-03-07 15:47:50 in NY City
- 2000.00 US dollars bid by lscottmann@aol.com (L. Scott Mann) at 1997-03-03 20:56:16 in Texas
- 10.00 US dollars bid by EricZ@EZarakov.XO.Com (TestBid for Eric) at 1997-03-03 19:33:03 in San Francisco

Figure 13.16: My generic classified ad system can be configured to run auctions for advertised items. Eric Zarakov is auctioning a delicious Canon EOS 300/2.8L lens. They cost about $4,600 new. Mr. Zarakov is asking $3,800 or best offer. Over eight days, he has received six bids (the highest so far is $2,750). (Note: this is not the most expensive item sold in the http://photo.net/photo classifieds; that distinction belongs to a $20,000 Zeiss 300mm lens. The seller was so grateful that he wanted to send me some money. I told him that the classifieds were a free service. He insisted. I asked him to write a check for whatever amount he deemed appropriate to Angell Memorial Animal Hospital. I didn't hear from him for a few weeks. Eventually I and Angell got the check. It turned out that the seller had a millionaire relative who'd died and left all of her money to an animal charity, rather disappointing the family. My request was a painful reminder!)

Oracle Exhibit 1011
Page 141 of 171



Classified Ads

for photo.net

A free service made possible by traditional Internet good citizenship.

[Place An Ad | Edit Old Ad | Add/Edit Alert | Hot Auctions]

Recent Ads

- 300mm Nikkor reflex lens $340
- WTB: MIRANDA SLR CAMERAS AND ACCESS
- FS: 2 Nikkormats, 2- 50 f/2's, 200 f.4, 105 f.2.5
- Lights & Motion Seahawk Video Housing $1400.00
- Black Nikon FE/meter broken w broken SB-15- $150 OBO

- All Ads Chronologically (including wanted to buy)

Ads by Category

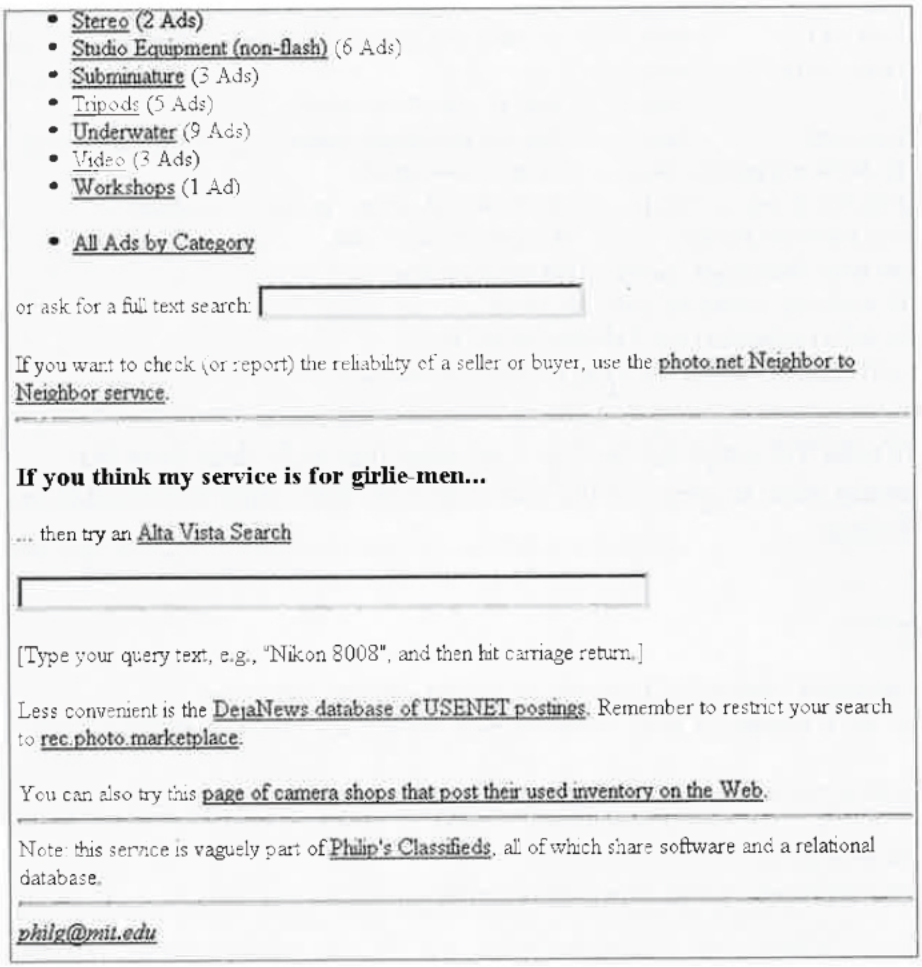
- 35mm non-SLR (51 Ads)
- 35mm SLR (258 Ads)
- Books and Magazines (3 Ads)
- Camera Parts (6 Ads)
- Classic (13 Ads)
- Darkroom (30 Ads)
- Digital Imaging (7 Ads)
- Electronic Flash (19 Ads)
- Lab Services (2 Ads)
- Large Format (20 Ads)
- Light Meters (11 Ads)
- Medium Format non-SLR (26 Ads)

Figure 13.17: Top-level page for a domain in my generic classifieds system. Note that the user gets a tremendous amount of information right on the first page: one-line summaries of recent ads, a listing of the available categories, a count of the number of ads in each category. See Figure 13.18 for the bottom of the page.

The Cover Page First, here's a table that defines an ad *domain*. I run classifieds for photo.net, Web Tools Review, and Web Travel Review all from the same server (http://classifieds.photo.net/gc/) and from the same tables. Each of these publications is a domain as far as my generic classified system is concerned. Here's what I represent for each domain:

```
create table ad_domains (
  domain               text not null primary key,
  maintainer_name      text,
  maintainer_email     text,
  maintainer_telephone text,
```

Oracle Exhibit 1011
Page 142 of 171



- Stereo (2 Ads)
- Studio Equipment (non-flash) (6 Ads)
- Subminiature (3 Ads)
- Tripods (5 Ads)
- Underwater (9 Ads)
- Video (3 Ads)
- Workshops (1 Ad)

- All Ads by Category

or ask for a full text search:

If you want to check (or report) the reliability of a seller or buyer, use the photo.net Neighbor to Neighbor service.

If you think my service is for girlie-men...

... then try an Alta Vista Search

[Type your query text, e.g., "Nikon 8008", and then hit carriage return.]

Less convenient is the DejaNews database of USENET postings. Remember to restrict your search to rec.photo.marketplace.

You can also try this page of camera shops that post their used inventory on the Web.

Note: this service is vaguely part of Philip's Classifieds, all of which share software and a relational database.

philg@mit.edu

Figure 13.18: The bottom portion of the first page for a domain in my generic classified system (see Figure 13.17 for the top). Note that only here is the full-text search engine made available. Users can browse just by clicking the mouse on the links above. Note also that I provide a link to the AltaVista search engine. My form targets a little script of mine that tacks on additional syntax to restrict AltaVista's search to the rec.photo.marketplace newsgroup. So a user typing "Canon macro lens" on my page results in a 302 redirect to http://www.altavista.digital.com/cgi-bin/query?pg=aq&what=news&fmt=.&q=newsgroups:rec.photo.marketplace+AND+Canon+macro+lens

Oracle Exhibit 1011
Page 143 of 171

```
  display_telephone    text,  -- maybe a customer service 800 number
  backlink_url         text,  -- if this is part of a bigger service
  backlink_title       text,
  blurb                text,  -- top of the front page
  blurb_bottom         text,  -- top of the front page
  insert_form_fragments text,  -- HTML form INPUTs
  ad_deletion_blurb    text,  -- printed out after an ad is deleted
  default_expiration_days       integer default 100,
  -- how many levels of categorization (if they just
  -- use primary_category then it is 1)
  levels_of_categorization integer default 1,
  user_extensible_categorization_p  boolean default 'f'
);
```

Here's the Tcl script for the top-level page that pulls data from the ad_domain table to generate the user interface, then pulls information from the ads table.

```
#
# parameters

# this probably should be a column in the ad_domains table but it
# isn't; so a change in the .tcl file will affect all the domains

set how_many_recent_ads_to_display 5

set form variables
set_form_variables_string_trim_DoubleAposQQ

# domain

set db [ns_db gethandle]
set selection [ns_db 1row $db "select * from ad_domains where domain =
'$QQdomain'"]
set_variables_after_query

# now all the properties of the domain are available as Tcl local vars

# we're doing some expensive database queries; we don't want users
# staring at a blank window, so return headers and some text

ReturnHeaders $conn


ns_write $conn "<html>
<head>
<title>$backlink_title Classified Ads</title>
```

Oracle Exhibit 1011
Page 144 of 171

```
</head>
<body bgcolor=#ffffff text=#000000>
<h2>Classified Ads</h2>

for <a href=\"$backlink_url\">$backlink_title</a>

<hr>

$blurb

<hr>

\[ <a href=\"place-ad.tcl?domain=$domain\">Place An Ad</a> |

<a href=\"edit-ad.tcl?domain=$domain\">Edit Old Ad</a> |

<a href=\"add-alert.tcl?domain=$domain\">Add/Edit Alert</a>

\]


<h3>Recent Ads</h3>

<ul>

"

# we only want the first 5 ads, but there is no clean way to specify
# that in SQL. So we just ask for all the unexpired ads in this domain
# orcered by descending classified_ad_id

set selection [ns_db select $db "select classified_ad_id,one_line
from classified_ads
where domain = '$QQdomain'
and (current_date <= expires or expires is null)
order by classified_ad_id desc"]

set ad_counter 0
while {[ns_db getrow $db $selection]} {
    incr ad_counter
    if { $ad_counter > $how_many_recent_ads_to_display } {
        # we've already displayed enough recent ads, so we shouldn't
        # read anymore rows from the preceding SELECT.  However, we
        # can't just walk away from an "open cursor". That can leave
        # all kinds of garbage around in the RDBMS client and server.
        # We call the AOLserver function "ns_db flush" which will in
        # turn inform the RDBMS server that no more rows are wanted.
```

Oracle Exhibit 1011
Page 145 of 171

```
        ns_db flush $db

        # call a Tcl function to break out of the while loop
        break
    }
    # we still have some recent ads to display
    set_variables_after_query
    ns_write $conn "
<li><a href=\"view-one.tcl?classified_ad_id=$classified_ad_id\">
$one_line
</a>
"

}

# we're done displaying the recent ads

ns_write $conn "<p>
<li><a href=\"domain-all.tcl?domain=$domain&by_category_p=f&wtb_p=f\">
All Ads Chronologically</a>
(<a href=\"domain-all.tcl?domain=$domain&by_category_p=f&wtb_p=t\">
including wanted to buy</a>)

</ul>

<h3>Ads by Category</h3>

<ul>"

# this is a very expensive database query using the GROUP BY feature
# of SQL.  We are asking the RDBMS to gather up the ads, group them
# by PRIMARY_CATEGORY, count the ads in each category, then return
# the count and the category name, ordering the returned rows by
# UPPER( PRIMARY_CATEGORY ) so that capitalization doesn't affect
# sorting. This query necessarily involves the RDBMS scanning the
# entire table.  It can still be very fast if you have a smart RDBMS
# on a computer with a lot of RAM. It is even tolerably fast with
# Illustra (a very stupid RDBMS when it comes to caching) because my
# table is pretty small and my RDBMS server box has 4 CPUs

# SQL nerd note 1: if you don't have OR EXPIRES IS NULL then you won't
# get any rows where EXPIRES is NULL.  A profound concept, to be sure,
# but one that escaped me at first.

# SQL nerd note 2: if you don't have a good name for something in the
# FROM list, you can just refer to it by position.  Here ORDER BY 3
# says "order by the 3rd thing in the FROM list" which is to say
```

```
# ORDER BY UPPER(PRIMARY_CATEGORY)

set selection [ns_db select $db "select count(*) as count,
primary_category as category,
upper(primary_category)
from classified_ads
where domain = '$QQdomain'
and (current_date <= expires or expires is null)
group by primary_category
order by 3"]

while {[ns_db getrow $db $selection]} {
    set_variables_after_query
    if { $count == 1 } {
        set pretty_count "1 Ad"
    } else {
        set pretty_count "$count Ads"
    }
    ns_write $conn "<li>
<a href=\"view-category.tcl?domain=[ns_urlencode
$domain]&primary_category=[ns_urlencode $category]\">
$category
</a>
($pretty_count)
"
}

ns_write $conn "<p>
<li>
<a href=\"domain-all.tcl?domain=$domain&by_category_p=t&wtb_p=t\">
All Ads by Category
</a>

</ul>

<form method=post action=search.tcl>
<input type=hidden name=domain value=\"$domain\">
or ask for a full text search:
<input type=text size=30 name=query_string>
</form>

$blurb_bottom

<hr>
```

Oracle Exhibit 1011
Page 147 of 171

```
Note: this service is vaguely part of
<a href=index.tcl>Philip's Classifieds</a>,
all of which share software and a relational database.

<hr>

<a href=\"mailto:$maintainer_email\">
<address>$maintainer_email</address>
</a>
</body>
</html>"
```

Note how the returned document will adopt the personality of the ad domain. The signature at the bottom is $maintainer_email (from the database). The links up at the top are to $backlink_url (from the database). There are blurbs at the top and bottom that come straight from the RDBMS. Go over to http://photo.net/photo/ right now and see how responsive the pages are. It is surprisingly fast to pull the ad_domain row out of the RDBMS. I've not yet been tempted to do any clever caching in Tcl global variables.

The Auction System Recording bids is straightforward:

```
create table classified_auction_bids (
  classified_ad_id      integer not null references classified_ads,
  bid                   numeric(9,2),
  currency              text default 'US dollars',
  bid_time              timestamp(0),
  location              text, -- e.g., 'New York City'
  email                 text,
  name                  text
);
```

Note the integrity constraint that this table can't contain bids for ads that don't exist in the classified_ads table. Assuming the RDBMS is behaving itself, we won't be able to delete an ad without first deleting bids for that ad. Here is a legal transaction in the augmented data model:

```
begin transaction;
delete from classified_auction_bids where classified_ad_id = 17;
delete from classified_ads where classified_ad_id = 17;
end transaction;
```

Oracle Exhibit 1011
Page 148 of 171

SUMMARY

From reading any of these cases, I hope you've come away thinking that interaction design is the heart and soul of good service. I also hope that the straightforward nature of the Tcl code in all of these examples convinces you that it is indeed tough to hang yourself with the software if you start out with a decent data model. Finally, I hope that the RemindMe system example has convinced you of the virtues of the management philosophy laid out in Chapter 10: Programmers do a text-only site; the graphic designer comes in after it is working and makes it pretty.

If you didn't like staring at my software, you'll be pleased to learn that the next two chapters take us back to the tone and level of the beginning of the book.

Oracle Exhibit 1011
Page 149 of 171

A Future So Bright You'll Need to Wear Sunglasses

Should Software Really Be Sold Like Tables and Chairs?

Your User's Browser: a GE Range

Personalization

Collaboratively Exchanged Data Models

Collaboratively Evolved Data Models

Grand Conclusion

Chapter

15

Eric Rabkin, Professor of English at University of Michigan, surveyed science fiction and found only one case in which a science fiction writer had accurately predicted an invention: Arthur C. Clarke's 1945 proposal for geostationary communication satellites, the first of which was launched in 1965. All the other writers credited with scientific invention were merely extrapolating implications of technologies that had already been invented.

The most successful Internet punditry is a lot like Rabkin's survey of science fiction. University labs got corporate money in the 1980s to invent virtual reality. Magazines and newspapers got advertising dollars in the 1990s to tell the public about the amazing new development of virtual reality. All of this was greatly facilitated by Ivan Sutherland and Bob Sproull. They placed an array of sensors in a ceiling to track the user's head postion and attitude. With a head-mounted display and real-time information about the user's position, Sutherland and Sproull were able to place synthetic chairs in the room (Sutherland joked that the ultimate computer display would let the user sit down in the chair). They built a completely functioning virtual reality system, using government research funds, and published the results to the world with papers, photographs, and movie demonstrations. The year? 1966.

Note: You can read more about Ivan Sutherland at http://www.eu.sun.com/960710/feature3/.

I've made a careful study of book and magazine Internet punditry. I've graphed the author's wealth and fame versus the novelty of the ideas presented. Based on this research, here are my predictions for the future:

- Have you ever picked up a small plastic device and heard a voice from the other side of the planet? You will. (Alexander Graham Bell, 1876)

Oracle Exhibit 1011
Page 151 of 171

- Have you ever had a friend in the office next door type a message and watched an image of the typed letters form seconds later on a cathode-ray tube in front of your eyes? You will. (Various e-mail systems of the 1960s)
- Have you ever cooperated with a friend in another city, each of you typing on a computer keyboard, drawing with a mouse, talking to each other, and looking at a little inset video picture of your collaborator on the screen? You will. (Douglas Engelbart at the December 1968 Fall Joint Computer Conference in San Francisco; demonstrated to a room of 2,000)
- Have you ever clicked a mouse on a hypertext document or graphic? You will. (Engelbart, same 1968 demo)
- Have you ever imagined that the mouse click could grab a file from halfway across the Internet? You will. (Engelbart, same 1968 demo)

Can we learn anything general from my results? Absolutely. Armies of hardware engineers will work anonymously in cubicles like slaves for 30 years so that the powerful computers used by pioneers in the 1960s will be affordable to everyone. Then in the 1990s rich people and companies will use their PR staffs to take credit for the innovations of the pioneers in the 1960s, without even having the grace to thank the hardware geeks who made it possible for them to steal credit in the first place. Finally, the media will elect a few official pundits who are (a) familiar enough with the 1960s innovations to predict next year's Cyberlandscape for the AOL crowd, but (b) not *so* familiar with the history of these innovations that they sound unconvincing when crediting them to the rich people and companies of the 1990s.

Where does that leave me? I'm not one of the pioneers of the 1960s—I was born in 1963. I'm not a rich person of the 1990s—I forgot to get rich during the Great Internet Boom. I'm not an official pundit, except once for an Italian newsweekly (see http://photo.net/philg/narcissism/narcissism.html for the full story)—I guess I must have done a bad job for those Italians.

I may be a failure, but they can't take away my aspirations. There isn't much point in aspiring to be a pioneer of the 1960s. The '60s are over, even if some folks in my hometown (the People's Republic of Cambridge, Massachusetts) haven't admitted it. There isn't much point in my aspiring to be an official real dead-trees media pundit. My friends would only laugh at me if I started writing for *Wired* magazine. However, being a rich person of the 1990s has a certain indefinable appeal for me. Perhaps it is this comment I made on one of my Web pages: "Not being a materialist in the U.S. is kind of like not appreciating opera if you live in Milan or art if you live in Paris. We support materialism better than any other culture. Because retailing and distribution are so efficient

Oracle Exhibit 1011
Page 152 of 171

here, stuff is cheaper than anywhere else in the world. And then we have huge houses in which to archive our stuff."

Materialism is definitely more fun when one is rich. How to get there, though. Conventional wisdom in Italy has it that "There are three ways to make money. You can inherit it. You can marry it. You can steal it." Based on my survey of the computer industry, the third strategy seems to be the most successful. With that in mind, here are some ideas that I've stolen from smarter people.

SHOULD SOFTWARE REALLY BE SOLD LIKE TABLES AND CHAIRS?

Steve Ward is probably the only MIT computer science professor that any undergraduate would ever want to be like. If you walk into the average MIT CS lecture, you'll see what looks like a troll in used clothing mumbling out equations to a sea of somnambulant nerds. Ward, on the other hand, stands over six feet tall, is crisply dressed, and speaks coherently. He looks like he would know what wines to order with a seven-course French dinner.

Talking to Ward about computer science reminds me of what it must have been like to be an Irish monk when the Roman Empire was disintegrating and Europe descended into barbarism. It doesn't matter what your idea is. Steve has a more elegant one, and he had it in the 1980s. Here's one thing I've learned from Professor Ward.

THE NUB

We software developers live in a pre-industrial age. We don't build on each other's work, we reinvent the wheel over and over again—and the bumps in the wheel. Ultimately, it is the user who gets the stuffing beaten out of him.

It is the way that software is sold that keeps software development mired in the 1950s. Software is put into packages and sold like tables or chairs. That's great because we have a highly efficient distribution and retail system for tables and chairs and because we've been buying things like tables and chairs for centuries. It would all work out beautifully for everyone if only tables and chairs needed periodic upgrades, if tables and chairs required documentation and support, if tables and chairs could be downloaded over networks, if users developed significant investments in the interfaces of tables and chairs, and if it cost $30 million to develop a slightly better table or chair from scratch.

Look at the choices that current software pricing forces people to make.

Johnny the User Johnny the user is a university student. He wants to use Adobe PhotoShop for a class project and has a Macintosh on the Internet in his dorm room. He can buy PhotoShop for $500, he can steal it from a friend, or he can drive to Kinko's to rent a Macintosh for a few hours.

Suppose that Johnny buys PhotoShop. Adobe gets $500 and is happy. Johnny gets manuals and support and he's working efficiently. Johnny doesn't have to drive anywhere so society doesn't suffer from increased pollution and traffic congestion. Unfortunately, probably not too many students are happy about paying $500 for software that they're only going to use for a day or two. Also, when Johnny next wants to use the software, he'll probably find that the version he has no longer runs with Apple's new operating system, or that Apple has gone belly-up and his version doesn't run on his new Windows NT machine, or that the instructor wants him to use a program feature that is only in the latest version of PhotoShop.

Let's be realistic. Johnny probably isn't going to buy PhotoShop. He's going to steal it from Adobe by borrowing the CD-ROM from his friend's friend. He'll spend his $500 on a spring break trip to Florida. Unfortunately for Johnny, PhotoShop is almost impossible to use without the manuals. Johnny drives to the bookstore and spends $30 on an "I stole the program and now I need a book on how to use it" book. Johnny wastes some time; Adobe gets no money; society has to breathe Johnny's exhaust fumes and wait behind his jalopy at intersections.

If Johnny is remarkably honest, he may go to Kinko's and rent a Macintosh running PhotoShop. This is great except that the network was supposed to free users from having to physically move themselves around. Johnny is inconvenienced and society is inconvenienced by the externalities of his driving.

Amanda the User Interface Programmer Amanda is writing some user interface code for an innovative new spreadsheet program. She wants it to appeal to the users of Microsoft Excel and Lotus 1-2-3 but knows that they have spent years learning the user interface quirks of those programs. Amanda has to choose between copying the user interface and spending ten years in federal court or making her new program work in a gratuitously different manner (in which case each user has to spend several days relearning commands that they already knew in their old programs).

Joey the Image Editor Programmer Joey wants to make a nice program for quickly converting all the images on a PhotoCD. Adobe PhotoShop does 99 percent of what his program needs to do. Unfortunately, lacking that last one percent, PhotoShop is useless for the task at hand. Adobe had no incentive to make the pieces of PhotoShop callable by other programs, so Joey has to start

Oracle Exhibit 1011
Page 154 of 171

from scratch or abandon his project. Should Joey succeed, his program will contain duplicates of code in PhotoShop. Joey's software, though, will have bugs that Adobe stamped out in 1991. It will ultimately be the user who is pushing the Macintosh restart button and losing work.

Adobe the Software Publisher Adobe wants to maximize its revenue under the "tables and chairs" software vending model. It will do this by keeping manuals and documentation out of the hands of users who don't pay, by not putting full documentation up on the Web, for example. Adobe will withhold support from users who stole the binary. Adobe will sue companies who copy the PhotoShop user interface. Adobe will not share its internal program design with anyone.

CHOICES SUMMARY

Selling software like tables and chairs forces users to make a buy/steal choice with a threshold of $500. It forces thousands of confusing new user interfaces into the marketplace every year. It forces programmers to start from scratch if they are to market anything at all.

A BETTER WAY

Suppose that Jane's Software Consortium (JaneSoft) negotiated deals with a bunch of software authors. Users would pay a fixed $x per year to JaneSoft for the right to use any software that they wished. Each user's computer would keep track of which company's programs were actually executed and file a report once a month with JaneSoft. Based on those usage reports, JaneSoft would apportion its revenues to software publishers and authors.

Let's revisit the same people under the new model . . .

Johnny the User Johnny can decide whether to (a) pay his $x a year and get everything, or (b) buy a few important packages under the tables-and-chairs model and steal/rent everything else. Assuming he pays the $x a year, Johnny may legally run any software that he finds useful. It gets delivered via the Internet to his machine along with any documentation and support that he may need.

Amanda the User Interface Programmer Amanda still wants her users to be able to employ the familiar Lotus user interface. It is now in Lotus's interest to tell other programmers how to call their user interface code. Because licensing consortium revenues are apportioned according to usage, every time a Lotus menu is displayed or command is run, Lotus is going to get some extra money from the consortium. Amanda's company will get paid when her new spreadsheet core program is executing. Lotus and Amanda's company are sharing revenue and it is in both of their interests to make the user productive.

Joey the Image Editor Programmer Adobe now has an incentive to document the internal workings of PhotoShop. Joey can tap into these and add his one percent. Because PhotoShop is an old and well-debugged program, the user gets a much more reliable product. Joey only gets 1 percent of the revenue derived from any user's session with "his" software, but he only did one percent of the work so he can move on to other projects. Furthermore, because he doesn't have to come up with an attractive physical package, his cost of entering the software market is considerably reduced.

Adobe the Software Publisher Adobe's main goal now is to get as many people as possible to run PhotoShop and for as long as possible. Remember that a user won't pay extra to run PhotoShop more frequently, but if a user spends a greater percentage of his time in PhotoShop then Adobe will get a greater percentage of the licensing consortium's revenues. Adobe's first likely action would be to put the PhotoShop manuals on the Web, possibly open only to people who are licensing consortium subscribers. Making telephone and e-mail support fast and effective becomes a priority because Adobe doesn't want any user to give up on PhotoShop and run Doom instead. Hardcopy manuals are mailed out free or at nominal cost.

Adobe sponsors conferences to help other software developers call PhotoShop's internals. Adobe will not file any look-and-feel lawsuits because they're getting paid every time someone uses their user interface code.

NEW WORLD ORDER

Five years after software licensing consortia are in place, the world looks very different. Fewer programs are written from the ground up, fewer users stab away cluelessly at stolen programs for which they lack documentation, fewer look-and-feel lawsuits are filed, fewer bugs are created. Roughly the same amount of money is flowing to the software publishing industry, but the industry has better information about who its customers are and how useful they find their products.

My personal prediction is that two kinds of consortia would emerge. One kind would cater to business. Users would pay $x per year and get the old familiar software. Consortia catering to home users, however, would offer a $0 per year deal: You can use any software you want, but we're going to replace those startup screens and hourglasses with ads for McDonald's and Coke. Ask for a spell check in your word processor? While it is loading, an ad for Rolaids will ask you how you spell relief. Ask PhotoShop to sharpen a big PhotoCD scan? That

Oracle Exhibit 1011
Page 156 of 171

thermometer progress bar will be buried underneath an ad reminding you how sharp you'd feel if you were dressed from head to toe in L.L. Bean clothing.

A LESS RADICAL APPROACH

Renting *software* rather than the physical machines on which it is installed would achieve some of the same goals as blanket licensing and metering. Certainly a casual user would prefer to spend $1 an hour trying out PhotoShop than $500 for "the package" and then $100 a year for updates. Adobe would then have many of the same incentives to make documentation and support readily available.

However, renting software would not solve the deeper problem created by software developers standing on each other's toes rather than each other's shoulders.

PRIVACY

I probably wouldn't want my employer to know that I spent 95 percent of my time running Netscape and Doom when I was supposed to be using Word and Excel. So I want to make sure that a public-key encryption system can be designed so that nobody can figure out which programs were run on my machine. Anonymity is good, but it opens the door to fraud by software publishers. Suppose that I write a text editing program. It isn't nearly as good as Emacs, so nobody uses it. But if I can figure out a way to file false usage reports that fool the consortia into thinking that 100,000 people ran my text editor for 2,000 hours each, I'll get a much larger than deserved share of license revenue. Again, public-key encryption and digital signatures can be used to fraud-proof the system.

WE HAVE A NETWORK; WE CAN DO BETTER

Selling software like tables and chairs is a fairly new idea. In the mainframe decades, customers rented software so that they could be sure of getting support and updates. The idea of selling software like tables and chairs was an innovation that came with the personal computer and it worked pretty well for a while. However, it doesn't make sense in a networked world.

> **Note:** If you want to see how absurd the current system has gotten, visit the IBM Patent Server (http://patent.womplex.ibm.com/) and look at the patents assigned to your favorite software vendor.

YOUR USER'S BROWSER: A GE RANGE

Your reader's house will be a Class C subnet. Every device in the typical American home will have an IP address. The washing machine, the microwave oven, the VCR, the stove, the clock radio, the thermostat. Any device with an IP address is a potential Web client. As a Web publisher, you have to think about how your content can be used by browsers that aren't keyboard, mouse, and monitor.

Do I believe in this explosion of Internetworking because I'm a technology optimist? Have I decided to write for *Wired* magazine after all? No. I believe this because I've become a technology pessimist.

PRODUCT ENGINEERING: THEORY VERSUS REALITY

When I graduated from MIT in 1982, I was a technology optimist. I was a genius doing brilliant engineering. My work would go out the door into the arms of an adoring public whose lives would be enriched by my creations. Experience taught me that I had at least the first part of this right: New products indeed go out the doors of companies. As to the rest, well, sometimes those products work. Sometimes the documentation is adequate. Sometimes the consumer can figure out how to make it work. But mostly every time consumers buy a new gizmo they are in for a few days of misery and waiting in tech support phone queues. Our society can engineer lots of things that it can't support.

An engineer's age is thus determinative of his or her attitude toward home networking. Young engineers think that we'll have home appliance networking because it will make life easier for consumers. Gerry Sussman, my old advisor at MIT, is getting a little bit grizzled and probably wouldn't argue with my characterization of him as an old engineer. Gerry loves to pull a huge N ("Navy") RF connector out of his desk drawer to show students how it can be mated with the small BNC ("Bayonet Navy Connector") for expediency. "These were both designed during World War II," Gerry will say. "You don't get strain relief but it makes a perfectly good contact in an emergency. The guys who designed these connectors were brilliant. On the other hand, there has been a commission meeting in Europe for 15 years trying to come up with a common power-plug standard."

The problems of home appliance networking are human and business problems, not technical problems. There is no reason why a Sony CD player shouldn't have been able to communicate intelligently with a Pioneer receiver ten years ago. Both machines contain computers. How come when you hit "Play" on the CD player, the receiver doesn't turn itself on and switch its input to CD?

Why can't a Nikon camera talk to Minolta's wireless flash system? Or, for that matter, why can't this year's Nikon camera talk intelligently to last year's Nikon flash?

Computer engineers are confused into thinking that companies care about interoperability. In fact, the inherently monopolistic computer industry was dragged kicking and screaming toward interoperability by the United States federal government, the one buyer large enough to insist on it. Many of the standards in the computer industry are due to federal funding or conditions in government purchasing contracts. Buyers of home appliances are too disorganized to insist on standards. General Electric's appliance division, the market leader in the U.S., isn't even a sponsor of the Consumer Electronics Bus consortium. IBM is. AT&T Bell Labs is. Hewlett-Packard is.

Does this mean you have to figure out how to fry an egg on your PC or telephone before you'll have a really smart house? No. As I hinted up top, I think that companies such as GE will start to put Internet interfaces into their appliances as soon as about 20 percent of American households are wired for full-time Internet, for example, with cable modems (see Chapter 6). But they won't do it because they think it is cool for your GE fridge to talk to your Whirlpool dishwasher. They'll do it because it will cut the cost of tech support for them. Instead of paying someone to wait on the 800 line while you poke around with your head underneath the fridge looking for the serial number, they'll want to ping your fridge across the Internet and find out the model, its current temperature, and whether there are any compressor failures.

WHAT KINDS OF THINGS CAN HAPPEN IN A NETWORKED HOUSE?

My GE Profile range (see http://money.rules-the.net/philg/materialism/kitchen.html) already has a tall backsplash with an LED display. If GE had put a 10base-T outlet on the back to provide technical support then the next logical step would be to replace the LED display with a color LCD screen. Then I would be able to browse recipe Web sites from my stove top. Once I'd found the desired recipe, I would press "start cooking." A dialog box would appear: "JavaScript Alert: Preheat oven to 375?" After I'd confirmed that, the recipe steps would unfold before me on the LCD.

WHAT DOES THIS MEAN TO ME AS A WEB PUBLISHER?

Ubiquitous Internet and therefore ubiquitous Web browsers imply that publishers will have to adhere to Tim Berners-Lee's original vision of the Web: The browser renders the content appropriately for the display. This idea seemed laughable when the "weirdo displays" were VT100 terminals in the hands of

Oracle Exhibit 1011
Page 159 of 171

physics post-docs. Who cares about those pathetic losers? They don't have enough money to buy any of the stuff we advertise on our site anyway.

So I watched as the sites I'd built for big publishers got tarted up with imagemaps and tables and frames and flashing GIFs and applets. If it looks OK in Netscape Navigator on a Mac or a PC, then ship it. Don't even worry whether it is legal HTML or not. Then one day WebTV came out. Suddenly there was a flurry of e-mail on the group mailing lists. How to redesign their sites to be compatible with WebTV? I had to fight the urge to reply, "I looked at my personal site on a WebTV the other day; it looked fine."

WebTV was a big shock to a lot of publishers. Yet WebTV is much more computer-like than any of the other household appliances that consumers will be connecting to the Web. Be ready: Focus on content. Standard HTML plus semantic tags can make your content useful to household devices with very primitive user interface capabilities.

PERSONALIZATION

Though I love to diss the bloated MIT administration and the hubris of computer science academics, I can say sincerely that one of the greatest privileges life can offer is teaching a section of MIT undergraduates.

My favorite course to TA is 6.041. Yes, all the courses at MIT are just numbers (the "6" refers to the Department of Electrical Engineering and Computer Science so it really isn't that much more dehumanizing than the "EECS 041" that you might find at another university). One of the reasons that I love 6.041 is the professor, Al Drake. He is one of the fully-human human beings that never seem to get past tenure committees these days. He's been teaching 6.041 for decades and he wrote the text: *Fundamentals of Applied Probability Theory.*

> **Note:** Despite the name, Drake's book (McGraw-Hill, 1967) is the world's clearest statistics text. I tried to learn statistics about four times and gave up. Statistics books and courses cater to two audiences: People who are presumed unable to think and/or learn probability theory, and people who are mathematics graduate students. Drake only devotes one chapter to statistics but a few hours spent with it is much more illuminating than any of the MIT stats courses.

Each week in 6.041, I would meet with students in small groups. I'd make them go up to the blackboard and work through problems that they hadn't seen before. Partly the idea was to see how they were thinking and offer corrections.

Partly the idea was to prepare them to give engineering presentations and communicate their ideas. The student at the board wasn't really supposed to solve the problem, just coordinate hints from other students at the conference table.

One day I gave a problem to a quiet Midwestern girl named Anne. She studied it for a moment, walked over to the board, and gave a five minute presentation on how to solve it, mentioning all of the interesting pedagogical points of the problem, writing down every step of the solution in neat handwriting. Her impromptu talk was better prepared than any lecture I'd ever given in the class.

Anne and I were chatting one day before class.

"What did you do on Sunday?" she asked.

"Oh, I don't know. Ate. Brushed the dog. Watched *The Simpsons*. And you?" I replied.

"Me and my housemates decided to have a hacking party. We do this every month or so. Since we have a network of PCs running Unix at home, it is easy to get lots of people programming together. We couldn't decide what to build so I said 'Well, we all like science fiction novels. So let's build a system where we type in the names of the books that we like and a rating. Then the system can grind over the database and figure out what books to suggest.'" She said.

And?

"It took us the whole afternoon, but we got it to the point where it would notice that I liked Books A, B, and C but hadn't read Book D which other people who liked A, B, and C had liked. So that was suggested for me. We also got it to notice if you and I had opposite tastes and suppress your recommendations."

This was back in 1994. Anne and her friends had, in one afternoon, completed virtually the entire annual research agenda of at least two professors whom I knew at MIT (neither in my department, I'm relieved to note).

The first lesson to be drawn from this example is that Anne is a genius. The second is that an afternoon hack, even by a genius, isn't enough to solve the personalization problem. Yet if you cut through the crust of hype that surrounds any of the expensive Web server personalization software "solutions" available in 1997, all that you find underneath is Anne's afternoon hack. Nor am I aware of any publisher who has done better with software developed in-house (though I know Pathfinder is trying).

What's wrong with Anne's system? First, it imposes a heavy burden of logging in and rating on users. Given that we're going to lose our privacy and have an unfeeling computer system know everything about our innermost thoughts and tastes, can't it at least be a painless process?

Suppose we did get everyone in the world to subscribe to Anne's system and tirelessly rate every Usenet posting, every Web site, every musical composition, every movie, every book. Does this help me make the choices that matter? If

I've typed in that I like the *Waldstein* sonata, probably Anne's software can tell me that I wouldn't like the Pat Boone cover of AC/DC's *It's a Long Way to the Top (If You Wanna Rock and Roll)*. But will it help me pick among Beethoven's other 31 piano sonatas? Is it meaningful to rate Beethoven's sonatas on a linear scale: *Pastoral* good, *Appassionata* great, *Moonlight*, somewhere in between?

Suppose my tastes change over time? Consider that old French saying that "If you're not a liberal when you're young, then you have no heart; if you're not a conservative when you're old, then you have no mind." Perhaps I liked Guy de Maupassant and Dickens when I was foolish and young but now that I'm old, I've come to see the supreme truth of Ayn Rand. I don't want Anne's system recommending a bunch of sissy books about people helping each other when I could be reading about a perfect society where rich people rent rather than loan their cars to friends.

That's no big deal. We'll just expire the ratings after ten years. But what if my tastes change over the course of a few days? Last week I was content to sit through four hours of *Hamlet*. This week the InterNIC, with that mix of greed and incompetence peculiar to unregulated monopolies, ripped my (fully paid up) domain WEBTRAVEL.ORG out of their database. I need a comedy.

READER RATINGS: A BIG MISTAKE?

Why do we ask readers to explicitly rate content? Each American is being watched by so many computers so much of the time that if we have to ask a person what he or she likes, then that only reveals the weakness of our imagination and technology.

Ken Phillips, a professor at New York University, has been thinking about these issues since the late 1970s when he set up a massive computer network for Citibank. He asked me what I thought was AT&T's most valuable asset. I tried to estimate the cost of undersea cables versus the fiber links the crisscross the continent. Ken laughed.

"AT&T gives you long distance service so they know which companies you call and how long you spend on the phone with each one. AT&T gives you a credit card so they know what you buy. AT&T owns Cellular One so, if you have a cell phone, they know where you drive and where you walk. By combining these data, AT&T can go to a travel agency and say 'For $100 each, we can give you the names of people who drive by your office every day, who've called airline 800 numbers more than three times in the last month, who have not called any other travel agencies, and who have spent more than $10,000 on travel in the last year.'"

Ken is a lot smarter than I am.

As discussed in Chapter 7, Web publishers and marketers are trying to do some of this with persistent magic cookies issued by central ad delivery/tracking services. However, these systems are extremely crude compared to traditional direct marketing databases. Judging from last week's harvest of junk snail mail, I'd say that the world's IBM mainframes know that I recently bought a condo, that I won't go to a store to buy anything, that I will buy stuff from a catalog, and that I shave with a blade. Judging from last week's harvest of junk e-mail, I'd say that the world's Unix and NT servers have decided that I'm a regular reader of Web sites that in fact I haven't visited for two years, that I enter contests (I don't), that I buy junkware/middleware and Web authoring software (I don't), and that I'm in the market for a Russian or Ukranian bride (one out of four isn't bad, I guess).

My behavior on the Web is much more consistently logged than my behavior in real life. Why then is my Web profile so much less accurate? Partly because Web data is fragmented. Information about which files I've downloaded is scattered among many different sites' server logs. But mostly because people don't know what to do with their data. Server-side junkware and Web site marketers are invariably expert at telling a story about all the wonderful data that they can collect. Occasionally they actually do collect and store this data. However, once the data goes into the big Oracle table, it seldom comes back out.

Why not? Partly because of technology. Web sites are generally implemented in a stateless fashion, as per the spirit of the original protocols. Each request from a user is handled in isolation. Producing an up-to-date profile on User X requires sifting through all the available data for User X. A typical implementation would use a bunch of RDBMS tables to store this data. These will grow to hundreds of megabytes in size. Sifting through these tables and JOINing them with each other is not going to be quick no matter how smart your RDBMS software. Certainly it is not something that you can afford to do on every page request.

Switching to an object database for storing user profiles is potentially beneficial. I talk about this a bit in Chapter 11 and in fact am planning some experiments myself in this area, using Franz Common Lisp to drive ObjectStore. Certainly Lisp is a huge improvement in software development technology over the mishmash of Tcl, Perl, Java, and C that sit behind the average Web site. And certainly an object database could be orders of magnitude faster for certain kinds of queries. But the main barriers to working personalization are inadequate data models, inadequate user models, inadequate thinking about obtaining data from off-the-Web sources, and inadequate characterization of Web site content. The last barrier on the list ought to be easy to surmount. If my Web pages are in the Unix file system, nothing stops me from creating a database table with one row per Web page. The row would contain the Unix

file name and some kind of description of its content. It sounds easy but if you think about it a bit, it is tough to imagine how to do a better job than just dumping all the words into a full-text indexer such as Excite for Web Servers. Anyway, even if we solve the content characterization problem, that still leaves all the hard user and data modeling problems.

What are the biggest, most sophisticated Web publishers doing right now to address these problems? Most of them are still trying to figure out how to add WIDTH and HEIGHT tags to their IMGs. Does the incompetence of publishers mean that hope is lost for personalization? Absolutely not. In fact, the best place for most "quiet personalization research" is at the client end.

CLIENT-SIDE PERSONALIZATION

My desktop machine knows that it is running Windows NT. If publishers added semantic tags to their sites (see Chapter 3), my Web browser could warn me that the software whose blurbs I was investigating wasn't available for NT. My desktop machine knows not only which Web pages I've downloaded, but also how long I've spent viewing each one. It knows which Web pages I've saved to my local disk. My desktop machine knows that I've sent a bunch of e-mail today to friends telling them how the InterNIC took my money and then shut down my domain. It can listen to my phone line and figure out that my autodialer has called the InterNIC 50 times and gotten a busy signal. It has watched me program one of my AOLservers to send InterNIC e-mail every 15 minutes. You'd think that my desktop machine could put all this together to say, "Philip, you should probably check out http://www.internicsucks.com. It also looks like you're going a little non-linear on this InterNIC thing. You ought to relax tonight. I notice from your calendar program that you don't have any appointments. I notice from your Quicken database that you don't have any money so you probably shouldn't be going to the theater. I notice that *Naked Gun* is on cable tonight. I don't see any payments in your Quicken database to a cable TV vendor so I assume you aren't a Cable Achiever. I remember seeing some e-mail from your friend David two months ago containing the words "invite" and "cable TV" so I assume that David has cable. I see from watching your phone line's incoming caller line ID that he has called you twice in the last week from his home phone so I assume he is in town. Call him up and invite yourself over."

I trust my desktop computer with my e-mail. I trust it with my credit card numbers. I trust it to monitor my phone calls. I trust it with my financial and tax data. I can program it to release or withhold this information. I don't have to rely on publishers' privacy notices. If publishers would stop trying to be clever behind my back, I would be happy to give them personal information of

my choosing. Publishers could spend a few weeks sitting down to come up with a standard for the exchange of personalization information. Netscape would add a Profile Upload feature to Navigator 6.0. Then a magazine wouldn't have to go out and join an ad banner network to find out what I like; they could just provide a button on their site and I'd push it to upload my profile. This would be useful for more mundane transactions as well. For example, instead of each publisher spending $150,000 developing a shopping basket system and order form, they could just put an "upload purchase authorization and shipping address" button on their site. I'd type my credit card and mailing address just once into the browser's Options menu rather than 1,000 times into various publishers' forms.

Note: If I didn't tend to always use the same browser/computer to surf and/or was heavily dependent on mobile computing, I would probably want to designate a single hard-wired computer as my personalization proxy, more or less like the Internet Fish that Brian LaMacchia built back in 1995 (see http://www-swiss.ai.mit.edu/~bal). These are "semi-autonomous, persistent information brokers; users deploy individual IFish to gather and refine information related to a particular topic. An IFish will initiate research, continue to discover new sources of information, and keep tabs on new developments in that topic. As part of the information-gathering process the user interacts with his IFish to find out what it has learned, answer questions it has posed, and make suggestions for guidance." As far as a Web publisher is concerned, a proxy such as an Internet Fish looks exactly the same as a client.

WHAT DOES THIS MEAN TO ME AS A WEB PUBLISHER?

Take two tacks. First, count on the client-side (and proxy) personalization systems getting smart and pervasive. This is where the most useful systems are going to be built. You can help client-side systems by adding semantic tags to your content. As discussed in Chapter 3 and in http://photo.net/philg/research/shame-and-war-revisited.html, computers can't understand natural language and aren't likely to learn how any time soon. Thus you need to express, in a formal language, "This is a list of features for a commercial computer program; this program is only available for Intel processors running Windows 95; this program costs $95."

Sadly, even if you wanted to do the right thing as a publisher, it isn't possible today. There is no agreed-upon language for tagging the semantics of Web documents. People who set the Web standards have instead invested the past five years in devising ways to support more garish advertising. You can

Oracle Exhibit 1011
Page 165 of 171

publish colored text. You can publish blinking text. You can publish blinking pictures. You can publish blinking pictures that make noise. You can publish moving blinking pictures that make noise. You just can't publish anything that is meaningful to another computer and that therefore might save a human being some time.

One day users will get tired of this. You can be ready for that day by keeping your content in a more structured, more semantically meaningful form than HTML.

In the meantime, you can take the second tack: banking on server-side personalization being done better and on a larger scale. Record user click streams. Record user click-throughs (see Chapter 7). Then sell the information! Remember the example of AT&T. Even if they made no money delivering cellular phone service, long distance service, and credit card transactions, they could still get quite fat by selling information about their users. With a sufficiently evil and refined system pervading the Internet, you might be able to make a living from a popular site without ever putting in a single banner ad. Just tell Honda that users A, B, and C downloaded several large JPEGs of the Acura NSX from http://money.rules-the.net/philg/cars/nsx.html, Canon that users D and E were studying the Nikon F4 review in http://photo.net/photo/, and American Airlines that users F, G, and H were reading the full story on Costa Rica in http://webtravel.org/cr/.

COLLABORATIVELY EXCHANGED DATA MODELS

As discussed briefly in Chapter 2, corporations have been squandering money on computers for years and don't have too much to show for their investment. Suppose that Spacely Sprockets wants to buy widgets from Acme Widgets. Spacely Sprockets has an advanced computerized purchasing system. An employee in purchasing works through some online forms to specify that Spacely needs 2,500 widgets, Spacely part number W147, Acme model number A491, to be delivered on June 1. The order is stored in a relational database.

Acme is also a modern company. They have an integrated order entry, inventory, and billing system backed by an RDBMS. As soon as the order goes into the system, it sets into motion a synchronized chain of events in the factory.

How does the data for the 2,500-widget order get from Spacely to Acme? Each decade had its defining technology:

- In the 1970s, a Spacely employee printed out the order from the advanced Spacely system, stuck it into an envelope and mailed it to Acme.

An Acme employee opened the envelope and keyed the order into the Acme system, typing "25,000" instead of "2,500."

- In the 1980s, a Spacely employee printed out the order from the advanced Spacely system and faxed it to Acme. An Acme employee grabbed the order from the fax machine output bin and keyed the order into the Acme system, typing "25,000" instead of "2,500."
- In the 1990s, a Spacely employee pulls the order out of the advanced Spacely system and e-mails it to Acme. An Acme employee grabs the order from his or her inbox and rekeys the order into the Acme system, typing "25,000" instead of "2,500."

If this all sounds a little more efficient than the business world with which you're familiar, keep in mind that the whole process is repeated in the opposite direction when Acme wants to invoice Spacely for the 25,000 widgets.

What stops Spacely's computer from talking directly to Acme's? Pre-Internet, one would give up when faced with the difficulty of getting bits back and forth. Post-Internet, one *could* give up when faced with the difficulties of security. Can we be sure that Spacely's computer won't attempt any naughty transactions on Acme's computer? For example, if Spacely had full access to Acme's RDBMS, it could mark lots of invoices as having been paid. The issue of security is an anthill, however, compared to the craggy mountain of data model incompatibility.

Column names may be different. Acme's programmers choose "part_number" and Spacely's use "partnum." To us they look the same, but to the computer they might as well be completely different. Worse yet are differences in the meaning of what is in that column. Acme has a different part number for the same widget than does Spacely. Nor need there be a one-to-one mapping between columns. Suppose Spacely's data model uses a single text field for shipping address and Acme's breaks up the address into line_1, line_2, city, state, postal_code, and country_code columns? Nor finally need there be a one-to-one mapping between tables. Spacely could spread an order across multiple tables. An order wouldn't contain an address at all, just a factory ID. You'd have to JOIN with the factories table if you wanted to print out one order with a meaningful shipping address. Acme might just have one wide table with some duplication of data. Multiple orders to the same factory would just contain duplicate copies of the factory address.

We could fix this problem the way GM did. Go over to Germany and buy some data models from SAP (http://www.sap.com/). Then make every division of the company use these data models and the same part numbers for the same

screws. Total cost? About one billion dollars. A smart investment? How can you doubt GM? This is the company that spent $5 billion on robots at a time when they could have purchased all of Toyota for about the same sum. Anyway, the bureaucrats at MIT were so fattened by undergrads paying $23,000 a year and so impressed by GM's smart move that they bought SAP data models, too. My advisor was skeptical that data models designed for a factory would work at a university. "Sure they will," I said, "You just have to think of each major as an assembly line. You're probably being modeled as a painting robot."

Was my faith in SAP shaken when, two calendar years and 40 person-years into the installation process, MIT still wasn't really up and running? Absolutely not. SAP is the best thing that ever happened to computer people. It appeals to businesses that are too stupid to understand and model their own processes but too rich to simply continue relying on secretaries and file cabinets. So they want to buy SAP or a set of data models from one of SAP's competitors. But since they can't understand their business processes well enough to model them themselves, they aren't able to figure out which product is the best match for those processes. So they hire consultants to tell them which product to buy. A friend of mine is one of these consultants. If I score a $1,250 a day Web consulting gig, I don't bother to gloat in front of David. His time is worth $6,500 a day. And he doesn't even know SQL! He doesn't have to do any programming. He doesn't have to do any database administration. He doesn't have to do any systems administration. David just has to fly first class around the world and sit at conference tables with big executives and opine that perhaps Oracle Financials would be better for their company than SAP.

There are plenty of rich stupid companies on the Web. Is it therefore true that the same "convert everyone to one data model" approach will achieve our objective of streamlined intercompany communication? No. There is no central authority that can force everyone to spend big bucks converting to a common data model. Companies probably won't spend much voluntarily either. Company X might have no objection to wasting billions internally but management is usually reluctant to spend money in ways that might benefit Company Y.

What does that leave us with? n companies on the Web technically able to share data but having n separate data models. Each time two companies want to share data, their programmers have to cooperate on a conversion system. Before everyone can talk to anyone, we'll have to build $n^*(n\text{-}1)$ unidirectional converters (for each of n companies we need a link to n-1 other companies, thus the $n^*(n\text{-}1)$). With just 200 companies, this turns out to be 39,800 converters.

If we could get those 200 companies to agree on a canonical format for data exchange then we'd only need to build 400 unidirectional converters. That is a

much more manageable number than 39,800, particularly when it is obvious that each company should bear the burden of writing two converters (one into and one out of its proprietary format).

The fly in the ointment here is that developing canonical data models can be extremely difficult. For something like hotel room booking, it can probably be achieved by a committee of volunteer programmers. For manufacturing, it apparently is tough enough that a company like SAP can charge tens of millions of dollars for one copy of its system (and even then they haven't really solved the problem because they and the customers heavily customize their systems). For medical records, it is a research problem (see http://www.emrs.org/).

That's why the next section is so interesting.

Collaboratively Evolved Data Models

When I was 14 years old, I was the smartest person in the world. I therefore did not need assistance or suggestions from other people. Now that I've reached the age of 33, my mind has deteriorated to the point that I welcome ideas from other minds with different perspectives and experience.

Suppose I wanted to build a database for indexing photographs. When I was 14, I would have sat down and created a table with precisely the correct number of columns and then used it forever. Today, though, I would build a Web front-end to my database and let other photographers use my software. I'd give them the capability of extending the data model just for their images. After a few months, I'd look at the extensions that they'd found necessary and use those to try to figure out new features that ought to be common in the next release of the software.

Note: If this example sounds insufficiently contrived, it is because it is one of my actual back burner projects; check http://photo.net/photo to see if I've actually done it.

Ditto for my SPAM mailing list manager system (http://www.greenspun.com/spam/), described *ad nauseum* in Chapter 13. The interesting thing to do with it would be to let each publisher add extra columns to his or her private data model and then see what people really wanted to do with the system.

A much more challenging problem is building a computer system that can find commonality among the extensions that users have made to a data model and automatically spit out a new release of the canonical data model that subsumes 85 percent of the custom modifications (you want as much

capability in the canonical data model as possible because off-the-shelf software will only be convenient when working with the standard portions of the model).

Why this obsession with data modeling? Computers can't do much to help us if we can't boil our problems down to formal models. The more things that we can formally model, the more that computers can help us. The Web is the most powerful tool that we've ever had for developing models. We don't need focus groups. We don't need marketing staff reading auguries. We don't need seven versions of a product before we get it right. We have a system that lets users tell us directly and in a formal language exactly what they need our data model to do.

GRAND CONCLUSION

The Internet is going to be big. You heard it here first.

Did you expect something more profound? Perhaps we should listen to an MIT professor who was asked the following:

Q: "Do you see the new technologies, by helping to increase the flow of information, to be a force toward decentralization of power or toward more democracy?"

A: "Certainly not in the rich countries. . . . It's not a big secret that the economy is moving very fast, in fact, from what used to be mainly national economies to an increasingly internationalized economy. So take the United States: Thirty years ago the question of international trade was not a big issue because the national economy was so huge in comparison with trade that it didn't matter all that much. You didn't have big debates about trade policy. Now that's changed. The international economy is enormous. In fact, it's not really trade, so about 40 percent of U.S. trade, as it's called, is actually internal to big transnational corporations. It means like one branch of the Ford Motor Company moving things to another branch which happens to be across a border. Forty is not a small amount and it's the same worldwide. But, in any event, the economy's becoming much more internationalized. It's much easier to move capital abroad. The effect of that is that production can be shifted much more easily to low-wage/high-repression areas elsewhere. And the effect of that is to bring the third-world model home to the United States and other rich countries. It means that these countries themselves are drifting toward a kind of a third-world model in which there is a sector of great wealth and privilege and a growing mass of people who are basically superfluous. They're not necessary for a profit either as producers or consumers. You can produce more cheaply

elsewhere and the market can easily become the international wealthy sectors. You end up with south-central Los Angeles . . ."

I do apologize for that potentially unsettling bit. It seems that all the official Internet pundits who happen to be MIT professors were giving interviews to reporters from dead-trees magazines. So I turned to MIT's most cited professor: Noam Chomsky. The quote above comes from a 1993 interview printed in *Chomsky for Beginners* (David Cogswell, 1996; Writers and Readers; see http://www.worldmedia.com/archive/ for an extensive on-line collection of Chomsky's ideas).

I talked to Chomsky a bit about the above quote and it turns out that it doesn't really represent his thinking today on the subject of Internet:

"The answers depend on whose hands will be at the controls. Advanced technology, more integrated world economy (NB: relative to GNP, it's not so different now than early in this century), Internet/Web, etc., are in themselves neutral with regard to the rich/poor. They can liberate or oppress, like—say—a hammer. In the hands of a carpenter, it can help build a house for someone. In the hands of a torturer, it can bash in the person's skull. These are questions for action, not speculation, which is idle."

I had an MIT kid over to my house a few weeks ago. He said that he'd been working as a consultant to Netscape writing software to stream video. It turned out that Netscape was itself doing a consulting project for Hustler Magazine and that the ultimate application was streaming pornography.

"How do you feel about that morally?" asked one of my sincere liberally educated neighbors.

"Well, they paid me a lot of money," was Stuart's reply.

This gets us back to Noam Chomsky's answer in *Secrets, Lies and Democracy* (David Barsamian 1994; Odonian) to "What do you think about the Internet?"

> "I think that there are good things about it, but there are also aspects of it that concern and worry me. This is an intuitive response—I can't prove it—but my feeling is that, since people aren't Martians or robots, direct face-to-face contact is an extremely important part of human life. It helps develop self-understanding and the growth of a healthy personality.
>
> "You just have a different relationship to somebody when you're looking at them than you do when you're punching away at a keyboard and some symbols come back. I suspect that extending that form of abstract and remote relationship, instead of direct, personal contact, is going to have unpleasant effects on what people are like. It will diminish their humanity, I think."

Oracle Exhibit 1011
Page 171 of 171