Preface

When we were first approached with the idea of a follow-up to *Beautiful Code*, this time about data, we found the idea exciting and very ambitious. Collecting, visualizing, and processing data now touches every professional field and so many aspects of daily life that a great collection would have to be almost unreasonably broad in scope. So we contacted a highly diverse group of people whose work we admired, and were thrilled that so many agreed to contribute.

This book is the result, and we hope it captures just how wide-ranging (and beautiful) working with data can be. In it you'll learn about everything from fighting with governments to working with the Mars lander; you'll learn how to use statistics programs, make visualizations, and remix a Radiohead video; you'll see maps, DNA, and something we can only really call "data philosophy."

The royalties for this book are being donated to Creative Commons and the Sunlight Foundation, two organizations dedicated to making the world better by freeing data. We hope you'll consider how your own encounters with data shape the world.

How This Book Is Organized

The chapters in this book follow a loose arc from data collection through data storage, organization, retrieval, visualization, and finally, analysis.

Chapter 1, *Seeing Your Life in Data*, by Nathan Yau, looks at the motivations and challenges behind two projects in the emerging field of personal data collection.

Chapter 2, *The Beautiful People: Keeping Users in Mind When Designing Data Collection Methods*, by Jonathan Follett and Matthew Holm, discusses the importance of trust, persuasion, and testing when collecting data from humans over the Web.

Chapter 3, *Embedded Image Data Processing on Mars*, by J. M. Hughes, discusses the challenges of designing a data processing system that has to work within the constraints of space travel.

Chapter 4, *Cloud Storage Design in a PNUTShell*, by Brian F. Cooper, Raghu Ramakrishnan, and Utkarsh Srivastava, describes the software Yahoo! has designed to turn its globally distributed data centers into a universal storage platform for powering modern web applications.

Chapter 5, *Information Platforms and the Rise of the Data Scientist*, by Jeff Hammerbacher, traces the evolution of tools for information processing and the humans who power them, using specific examples from the history of Facebook's data team.

Chapter 6, *The Geographic Beauty of a Photographic Archive*, by Jason Dykes and Jo Wood, draws attention to the ubiquity and power of colorfully visualized spatial data collected by a volunteer community.

Chapter 7, *Data Finds Data*, by Jeff Jonas and Lisa Sokol, explains a new approach to thinking about data that many may need to adopt in order to manage it all.

Chapter 8, *Portable Data in Real Time*, by Jud Valeski, dives into the current limitations of distributing social and location data in real time across the Web, and discusses one potential solution to the problem.

Chapter 9, *Surfacing the Deep Web*, by Alon Halevy and Jayant Madhavan, describes the tools developed by Google to make searchable the data currently trapped behind forms on the Web.

Chapter 10, *Building Radiohead's House of Cards*, by Aaron Koblin with Valdean Klump, is an adventure story about lasers, programming, and riding on the back of a bus, and ending with an award-winning music video.

Chapter 11, *Visualizing Urban Data*, by Michal Migurski, details the process of freeing and beautifying some of the most important data about the world around us.

Chapter 12, *The Design of Sense.us*, by Jeffrey Heer, recasts data visualizations as social spaces and uses this new perspective to explore 150 years of U.S. census data.

Chapter 13, *What Data Doesn't Do*, by Coco Krumme, looks at experimental work that demonstrates the many ways people misunderstand and misuse data.

Chapter 14, *Natural Language Corpus Data*, by Peter Norvig, takes the reader through some evocative exercises with a trillion-word corpus of natural language data pulled down from across the Web.

Chapter 15, *Life in Data: The Story of DNA*, by Matt Wood and Ben Blackburne, describes the beauty of the data that is DNA and the massive infrastructure required to create, capture, and process that data.

Chapter 16, *Beautifying Data in the Real World*, by Jean-Claude Bradley, Rajarshi Guha, Andrew Lang, Pierre Lindenbaum, Cameron Neylon, Antony Williams, and Egon Willighagen, shows how crowdsourcing and extreme transparency have combined to advance the state of drug discovery research.

Chapter 17, *Superficial Data Analysis: Exploring Millions of Social Stereotypes*, by Brendan O'Connor and Lukas Biewald, shows the correlations and patterns that emerge when people are asked to anonymously rate one another's pictures.

Chapter 18, *Bay Area Blues: The Effect of the Housing Crisis*, by Hadley Wickham, Deborah F. Swayne, and David Poole, guides the reader through a detailed examination of the recent housing crisis in the Bay Area using open source software and publicly available data.

Chapter 19, *Beautiful Political Data*, by Andrew Gelman, Jonathan P. Kastellec, and Yair Ghitza, shows how the tools of statistics and data visualization can help us gain insight into the political process used to organize society.

Chapter 20, *Connecting Data*, by Toby Segaran, explores the difficulty and possibilities of joining together the vast number of data sets the Web has made available.

Conventions Used in This Book

The following typographical conventions are used in this book:

Italic
: Indicates new terms, URLs, email addresses, filenames, and file extensions.

`Constant width`
: Used for program listings, as well as within paragraphs to refer to program elements such as variable or function names, databases, data types, environment variables, statements, and keywords.

`Constant width bold`
: Shows commands or other text that should be typed literally by the user.

`Constant width italic`
: Shows text that should be replaced with user-supplied values or by values determined by context.

Using Code Examples

This book is here to help you get your job done. In general, you may use the code in this book in your programs and documentation. You do not need to contact us for permission unless you're reproducing a significant portion of the code. For example, writing a program that uses several chunks of code from this book does not require permission. Selling or distributing a CD-ROM of examples from O'Reilly books does require permission. Answering a question by citing this book and quoting example code does not require permission. Incorporating a significant amount of example code from this book into your product's documentation does require permission.

We appreciate, but do not require, attribution. An attribution usually includes the title, author, publisher, and ISBN. For example: "*Beautiful Data*, edited by Toby Segaran and Jeff Hammerbacher. Copyright 2009 O'Reilly Media, Inc., 978-0-596-15711-1."

If you feel your use of code examples falls outside fair use or the permission given here, feel free to contact us at *permissions@oreilly.com*.

How to Contact Us

Please address comments and questions concerning this book to the publisher:

O'Reilly Media, Inc.
1005 Gravenstein Highway North
Sebastopol, CA 95472
800-998-9938 (in the United States or Canada)
707-829-0515 (international or local)
707-829-0104 (fax)

We have a web page for this book, where we list errata, examples, and any additional information. You can access this page at:

http://oreilly.com/catalog/9780596157111

To comment or ask technical questions about this book, send email to:

bookquestions@oreilly.com

For more information about our books, conferences, Resource Centers, and the O'Reilly Network, see our website at:

http://oreilly.com

CHAPTER ONE

Seeing Your Life in Data

Nathan Yau

IN THE NOT-TOO-DISTANT PAST, THE WEB WAS ABOUT SHARING, BROADCASTING, AND DISTRIBUTION. But the tide is turning: the Web is moving toward the individual. Applications spring up every month that let people track, monitor, and analyze their habits and behaviors in hopes of gaining a better understanding about themselves and their surroundings. People can track eating habits, exercise, time spent online, sexual activity, monthly cycles, sleep, mood, and finances online. If you are interested in a certain aspect of your life, chances are that an application exists to track it.

Personal data collection is of course nothing new. In the 1930s, Mass Observation, a social research group in Britain, collected data on various aspects of everyday life—such as beards and eyebrows, shouts and gestures of motorists, and behavior of people at war memorials—to gain a better understanding about the country. However, data collection methods have improved since 1930. It is no longer only a pencil and paper notepad or a manual counter. Data can be collected automatically with mobile phones and handheld computers such that constant flows of data and information upload to servers, databases, and so-called data warehouses at all times of the day.

With these advances in data collection technologies, the data streams have also developed into something much heftier than the tally counts reported by Mass Observation participants. Data can update in real-time, and as a result, people want up-to-date information.

It is not enough to simply supply people with gigabytes of data, though. Not everyone is a statistician or computer scientist, and not everyone wants to sift through large data sets. This is a challenge that we face frequently with personal data collection.

While the types of data collection and data returned might have changed over the years, individuals' needs have not. That is to say that individuals who collect data about themselves and their surroundings still do so to gain a better understanding of the information that lies within the flowing data. Most of the time we are not after the numbers themselves; we are interested in what the numbers mean. It is a subtle difference but an important one. This need calls for systems that can handle personal data streams, process them efficiently and accurately, and dispense information to nonprofessionals in a way that is understandable and useful. We want something that is more than a spreadsheet of numbers. We want the story in the data.

To construct such a system requires careful design considerations in both analysis and aesthetics. This was important when we implemented the Personal Environmental Impact Report (PEIR), a tool that allows people to see how they affect the environment and how the environment affects them on a micro-level; and your.flowingdata (YFD), an in-development project that enables users to collect data about themselves via Twitter, a microblogging service.

For PEIR, I am the frontend developer, and I mostly work on the user interface and data visualization. As for YFD, I am the only person who works on it, so my responsibilities are a bit different, but my focus is still on the visualization side of things. Although PEIR and YFD are fairly different in data type, collection, and processing, their goals are similar. PEIR and YFD are built to provide information to the individual. Neither is meant as an endpoint. Rather, they are meant to spur curiosity in how everyday decisions play a big role in how we live and to start conversations on personal data. After a brief background on PEIR and YFD, I discuss personal data collection, storage, and analysis with this idea in mind. I then go into depth on the design process behind PEIR and YFD data visualizations, which can be generalized to personal data visualization as a whole. Ultimately, we want to show individuals the beauty in their personal data.

Personal Environmental Impact Report (PEIR)

PEIR is developed by the Center for Embedded Networked Sensing at the University of California at Los Angeles, or more specifically, the Urban Sensing group. We focus on using everyday mobile technologies (e.g., cell phones) to collect data about our surroundings and ourselves so that people can gain a better understanding of how they interact with what is around them. For example, DietSense is an online service that allows people to self-monitor their food choices and further request comments from dietary specialists; Family Dynamics helps families and life coaches document key features of a family's daily interactions, such as colocation and family meals; and Walkability helps residents and pedestrian advocates make observations and voice their concerns about neighborhood

walkability and connections to public transit.* All of these projects let people get involved in their communities with just their mobile phones. We use a phone's built-in sensors, such as its camera, GPS, and accelerometer, to collect data, which we use to provide information.

PEIR applies similar principles. A person downloads a small piece of software called Campaignr onto his phone, and it runs in the background. As he goes about his daily activities—jogging around the track, driving to and from work, or making a trip to the grocery store, for example—the phone uploads GPS data to PEIR's central servers every two minutes. This includes latitude, longitude, altitude, velocity, and time. We use this data to estimate an individual's impact on and exposure to the environment. Environmental pollution sensors are not required. Instead, we use what is already available on many mobile phones—GPS—and then pass this data with context, such as weather, into established environmental models. Finally, we visualize the environmental impact and exposure data. The challenge at this stage is to communicate meaning in data that is unfamiliar to most. What does it mean to emit 1,000 kilograms of carbon in a week? Is that a lot or is that a little? We have to keep the user and purpose in mind, as they drive the system design from the visualization down to the data collection and storage.

your.flowingdata (YFD)

While PEIR uses a piece of custom software that runs in the background, YFD requires that users actively enter data via Twitter. Twitter is a microblogging service that asks a very simple question: *what are you doing right now?* People can post, or more appropriately, *tweet*, what they are doing via desktop applications, email, instant messaging, and most importantly (as far as YFD is concerned), SMS, which means people can tweet with their mobile phones.

YFD uses Twitter's ubiquity so that people can tweet personal data from anywhere they can send SMS messages. Users can currently track eating habits, weight, sleep, mood, and when they go to the bathroom by simply posting tweets in a specific format. Like PEIR, YFD shows users that it is the little things that can have a profound effect on our way of life. During the design process, again, we keep the user in mind. What will keep users motivated to manually enter data on a regular basis? How can we make data collection as painless as possible? What should we communicate to the user once the data has been logged? To this end, I start at the beginning with data collection.

Personal Data Collection

Personal data collection is somewhat different from scientific data gathering. Personal data collection is usually less formal and does not happen in a laboratory under controlled conditions. People collect data in the real world where there can be interruptions, bad network connectivity, or limited access to a computer. Users are not necessarily data experts, so when something goes wrong (as it inevitably will), they might not know how to adjust.

* CENS Urban Sensing, *http://urban.cens.ucla.edu/*

Therefore, we have to make data collection as simple as possible for the user. It should be unobtrusive, intuitive, and easy to access so that it is more likely that data collection becomes a part of the daily routine.

Working Data Collection into Routine

This is one of the main reasons I chose Twitter as YFD's data proxy from phone or computer to the database. Twitter allows users to post tweets via several outlets. The ability to post tweets via mobile phone lets users log data from anywhere their phones can send SMS messages, which means they can document something as it happens and do not have to wait until they have access to a computer. A person will most likely forget if she has to wait. Accessibility is key.

One could accomplish something similar with email instead of Twitter since most mobile phones let people send SMS to an email address, and this was in fact the original implementation of YFD. However, we go back to data collection as a natural part of daily routine. Millions of people already use Twitter regularly, so part of the challenge is already relieved. People do use email frequently as well, and it is possible they are more comfortable with it than Twitter, but the nature of the two is quite different. On Twitter, people update several times a day to post what they are doing. Twitter was created for this single purpose. Maybe a person is eating a sandwich, going out for a walk, or watching a movie. Hundreds of thousands tweet this type of information every day. Email, on the other hand, lends itself to messages that are more substantial. Most people would not email a friend to tell them they are watching a television program—especially not every day or every hour.

By using Twitter, we get this posting regularity that hopefully transfers to data collection. I tried to make data logging on YFD feel the same as using Twitter. For instance, if someone eats a salami sandwich, he sends a message: "ate salami sandwich." Data collection becomes conversational in this way. Users do not have to learn a new language like SQL. Instead, they only have to remember keywords followed by the value. In the previous example, the keyword is *ate* and the value is *salami sandwich*. To track sleep, a user simply sends a keyword: *goodnight* when going to sleep and *gmorning* when waking.

In some ways, posting regularity with PEIR was less challenging than with YFD. Because PEIR collects data automatically in the background, the user just has to start the software on his phone with a few presses of a button. Development of that software came with its own difficulties, but that story is really for a different article.

Asynchronous data collection

For both PEIR and YFD, we found that asynchronous data collection was actually necessary. People wanted to enter and upload data after the event(s) of interest had occurred. On YFD, people wanted to be able to add a timestamp to their tweets, and PEIR users wanted to upload GPS data manually.

As said before, the original concept of YFD was that people would enter data only when something occurred. That was the benefit and purpose of using Twitter. However, many people did not use Twitter via their mobile phone, so they would have to wait until a computer was available. Even those who did send SMS messages to Twitter often forgot to log data; some people just wanted to enter all of their data at the end of the day.

Needless to say, YFD now supports timestamps. It was still important that data entry syntax was as close to conversational as possible. To accommodate this, users can append the time to any of their tweets. For example, "ate roast chicken and potatoes at 6:00pm" or "goodnight at 23:00." The timestamp syntax is to simply append "at hh:mm" to the end of a tweet. I also found it useful to support both standard and military time formats. Finally, when a user enters a timestamp, YFD will record the most recent occurrence of the time, so in the previous "goodnight" example, YFD would enter the data point for the previous night.

PEIR was also originally designed only for "in the moment" data collection. As mentioned before, Campaignr runs on a user's mobile phone and uploads GPS data periodically (up to every 20 seconds) to our central server. This adds up to hundreds of thousands of data points for a single user who runs PEIR every day with very little effort from the user's side. Once the PEIR application is installed on a phone, a user simply starts the application with a couple of button presses. However, almost right from the beginning, we found we could not rely on having a network connection 100% of the time, since there are almost always areas where there is no signal from the service carrier. The simplest, albeit naive, approach would be to collect and upload data only when the phone has a connection, but we might lose large chunks of data. Instead, we use a cache to store data on a phone's local memory until connectivity resumes. We also provide a second option to collect data without any synchronous uploading at all.

The takeaway point is that it is unreasonable to expect people to collect data for events at the time they happen. People forget or it is inconvenient at the time. In any case, it is important that users are able to enter data later on, which in turn affects the design of the next steps in the data flow.

Data Storage

For both YFD and PEIR, it was important to keep in mind what we were going to do with the data once it was stored. Oftentimes, database mechanisms and schemas are decided on a whim, and the researchers regret it further down the road, either because their choice makes it hard to process the data or because the database is not extensible. The choice for YFD was not particularly difficult. We use MySQL for other projects, and YFD involves mostly uncomplicated `insert` and `select` statements, so it was easy to set up. Also, data is manually entered—not continuously uploaded like PEIR—so the size of database tables is not an issue in these early stages of development. The main concern was that I wanted to be able to extend the schema when I added new trackers, so I created the schema with that in mind.

PEIR, on the other hand, required more careful database development. We perform thousands of geography-based computations every few minutes, so we used PostGIS to add support for geographic objects to a PostgreSQL database. Although MySQL offers GIS and spatial extensions, we decided that PostGIS with PostgreSQL was more robust for PEIR's needs.

This is perhaps oversimplifying our database design process, however. I should back up a bit. We are a group of 10 or so graduate students with our own research interests, and as expected, work on individual components of PEIR. This affected how we work a great deal. PEIR data was very scattered to begin with. We did not use a unified database schema; we created multiple databases as we needed them, and did not follow any specific design patterns. If anyone joined PEIR during this mid-early stage, he would have been confused by where and what all the data was and who to contact to find out. I say this because I joined the PEIR project midway. To alleviate this scattered problem, we eventually froze all development, and one person who had his hand in all parts of PEIR skillfully pieced everyone's code and database tables together. It became quite clear that this consolidation of code and schemas was necessary once user experience development began. In retrospect, it would have been worth the extra effort to take a more calculated approach to data storage in the early goings, but such is the nature of graduate studies.

Coordination and code consolidation are not an issue with YFD, since there is only one developer. I can change the database schema, user interface, and data collection mechanism with little fuss. I also use Django, a Python web framework, which uses a model-view-control approach and allows for rapid and efficient development. I do, however, have to do everything myself. Because of the group's diversity in statistics, computer science, engineering, GIS, and environmental science, PEIR is able to accomplish more—most notably in the area of data processing, as discussed in the next section. So there are certainly advantages and disadvantages to developing with a large group.

Data Processing

Data processing is the important underpinning of the personal data collection system that users almost never see and usually are not interested in. They tend to be more interested in the results of the processing. This is the case for YFD. PEIR users, on the other hand, benefit from seeing how their data is processed, and it in turn affects the way they interpret impact and exposure.

The analytical component of PEIR consists of a series of server-side processing steps that start with GPS data to estimate impact and exposure. To be precise, we can divide the processing into four separate phases:*

* PEIR, *http://peir.cens.ucla.edu*

1. **Trace correction and annotation:** Where possible, the error-prone, undersampled location traces are corrected and annotated using estimation techniques such as map matching with road network and building parcel data. Because these corrections and annotations are estimates, they do carry along uncertainties.
2. **Activity and location classification:** The corrected and annotated data is automatically classified as *traveling* or *stationary* using web services to provide a first level of refinement to the model output for a given person on a given day. The data is also split into *trips* based on dwell time.
3. **Context estimation:** The corrected and classified location data is used as input to web-based information sources on weather, road conditions, and aggregated driver behaviors.
4. **Exposure and impact calculation:** Finally, the fine-grained, classified data and derived data is used as input to geospatial data sets and microenvironment models that are in turn used to provide an individual's personalized estimates.

While PEIR's focus is still on the results of this four-step process, we eventually found that users wanted to know more about how impact and exposure were estimated. So for each chunk of data we provide details of the process, such as what percentage of time was spent on a freeway and what the weather was like around where the user was traveling. We also include a detailed explanation for every provided metric. In this case, transparency in the estimation process allows users to see how their actions have an effect on impact and exposure rather than just knowing how much or how little they are polluting their neighborhood. There is, of course, such a thing as information overload, so we are careful in how much (and how little) we show. We address much of these issues in the next section.

Data Visualization

Once data is collected, uploaded, and processed, users need to be able to access, evaluate, and explore their data. The main design goal behind YFD and PEIR was to make personal data understandable to nonprofessionals. Data has to be presented in a way that is relatable; it has to be humanized. Oftentimes we get caught up in statistical charts and graphs, which are extremely useful, but at the same time we want to engage users so that they stay interested, continue collecting data, and keep coming back to the site to gauge their progress in whatever they are tracking. Users should understand that the data is about them and reflect the choices they make in their daily lives.

I like to think of data visualization as a story. The main character is the user, and we can go two ways. A story of charts and graphs might read a lot like a textbook; however, a story with context, relationships, interactions, patterns, and explanations reads like a novel. This is not to say that one or the other is better. There are plenty of interesting textbooks, and probably just as many—if not more—boring novels. We want something in between the textbook and novel when we visualize personal data. We want to present the facts, but we also want to provide context, like the who, what, when, where, and why of the numbers. We are after emotion. Data often can be sterile, but only if we present it that way.

PEIR

In the case of PEIR, we were met with the challenge of presenting scientific data—carbon impact, exposure to high levels of particulate matter, and impact to sensitive sites such as hospitals and schools. Impact and exposure are not a part of everyday conversation. Most people do not know whether 1,000 kilograms of carbon emissions in a day is a lot or a little. Is one hour of exposure to high levels of particulate matter normal? These types of questions factor into PEIR's visualization design. It is important to remember, however, that even though the resulting data is not immediately understandable, it is all derived from location data, which is extremely intuitive. There are perhaps few types of data that are so immediately understandable as one's place in physical space. Therefore, we use maps as the visualization anchor point and work from there.

Mapping location-based data

Location-based data drives the PEIR system, so an interactive map is the core of the user interface. We initially used the Google Maps API, but quickly nixed it in the interest of flexibility. Instead, we use Modest Maps. It is a display and interaction library for tile-based maps in Flash and implemented in ActionScript 3.0. Modest Maps provides a core set of features, such as panning and zooming, but allows designers and developers to easily customize displays. Modest Maps implementations can easily switch map tiles, whether the choice is to use Microsoft's map tiles, Google's, custom-built ones, or all of the above. We are free to adjust color, layout, and overall style, which lend themselves to good design practice and useful visualization, and the flexibility allows us to incorporate our own visualizations on the map or as a supplement. In the end, we do not want to limit ourselves to just maps, and Modest Maps provides the flexibility we need to do this.

Experimenting with visual cues

We experimented with a number of different ways to represent PEIR data before deciding on the final mapping scheme. During the design process, we considered several parameters:

- How can users interact with a lot of traces at once without cluttering the map?
- How can we represent both stationary (user is idle) and traveling (user is moving) data chunks at the same time?
- How do we display values from all four microenvironment models?
- What colors should we use to represent GPS trace, impact, and exposure?
- How do we shift focus toward the actual data and away from the underlying map tiles?

Mapping multivariate location traces

In the early stages of the design process, we mapped GPS traces the way that users typically see location tracks: simply a line that goes from point to point. This was before taking values from the microenvironment models into account, so the map was a basic implementation

using Modest Maps and tiles from OpenStreetMap. GPS traces were mono-colored and represented nothing but location; there was a circle at the end so that the user would know where the trip began and ended.

This worked to a certain extent, but we soon had to visualize more data, so we changed the format. We colored traces based on impact and exposure values. The color scheme used five shades of red. Higher levels of, say, carbon impact were darker shades of red. Similarly, trips that had lower carbon impact were lighter shades of red.

The running metaphor is that the more impact the user has on the environment, the more the trip should stand out on the map. The problem with this implementation was that the traces on the map did not stand out (Figure 1-1). We tried using brighter colors, but the brightly colored trips clashed with the existing colors on the map. Although we want traces to stand out, we do not want to strain the user's eyes. To solve this problem we tried a different mapping scheme that again made all trips on the map mono-color, but used circles to encode impact and exposure. All traces were colored white, and the model values were visually represented with circles that varied in size at the end of each trip. Greater values were displayed as circles larger in area while lesser values were smaller in area. This design scheme was short-lived.

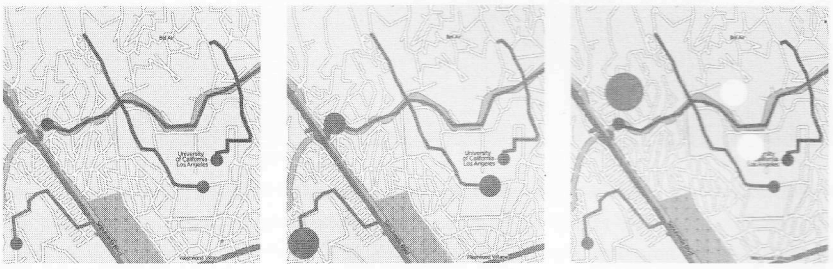


FIGURE 1-1. We experimented with different visual cues on a map to best display location data with impact and exposure values. The above shows three iterations during our preliminary design. The left map shows GPS traces color-coded by carbon impact; in the center map, we encoded impact with uni-color area circles; on the right, we incorporated GPS data showing when the user was idle and went back to using color-coding. (See Color Plate 1.)

One problem with representing values only at the end of a trace was that users thought the circles indicated that something happened at the very end of each trip. However, this is not the case. The map should show that something is happening during the entirety of a trip. Carbon is emitted everywhere you travel, not collected and then released at a destination.

We switched back to color-coding trips and removed the scaled area circles representing our models' values. At this point in the design process, we now had two types of GPS data: traveling and stationary. Traveling trips meant that the user was moving, whether on foot or in a vehicle; stationary chunks are times when the user is not moving. She might be sitting at a desk or stuck in traffic. To display stationary chunks, we did not completely abandon the idea of using area circles on the map. Larger circles mean longer duration, and smaller circles mean shorter duration. Similar to traveling trips, which are represented by

lines, area circles are color-coded appropriately. For example, if the user chooses to color-code by particulate matter exposure, a stationary chunk that was spent idle on the freeway is shown as a brightly colored circle.

However, we are again faced with same problem as before: trying to make traces stand out on the map without clashing with the map's existing colors. We already tried different color schemes for the traces, but had not yet tried changing the shades of the actual map. Inspired by Trulia Snapshot, which maps real estate properties, we grayscaled map tiles and inverted the color filters so that map items that were originally lightly colored turned dark and vice versa. To be more specific, the terrain was originally lightly colored, so now it is dark gray, and roads that were originally dark are now light gray. This darkened map lets lightly colored traces stand out, and because the map is grayscale, there is less clashing (Figure 1-2). Users do not have to try hard to distinguish their data from roads and terrain. Modest Maps provided this flexibility.

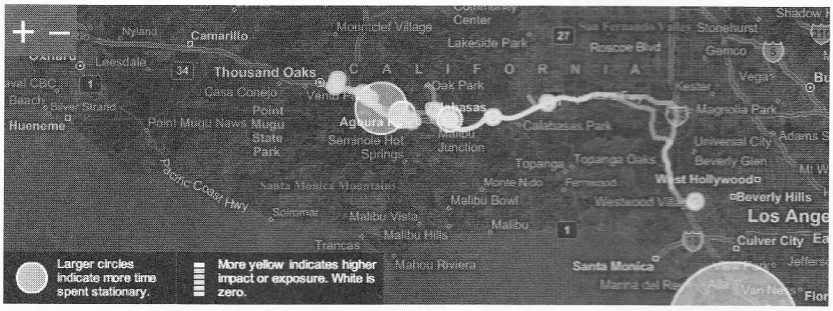


FIGURE 1-2. In the current mapping scheme, we use color filters to highlight the data. The map serves solely as context. Linked histograms show impact and exposure distributions of mapped data. When the user scrolls over a histogram bar, the corresponding GPS data is highlighted on the map. (See Color Plate 2.)

Choosing a color scheme

Once we established map tiles as the dark background and represented trips in the light foreground, we decided what colors to use. This is important because users recognize some colors as specific types of events. For example, red often means to stop or that there is danger ahead, whereas green means progress or growth, especially from an environmental standpoint.

It is also important to not use too many contrasting colors. Using dissimilar colors without any progression indicates categorical data. Model values, however, are on a continuous scale. Therefore, we use colors with a subtle gradient. In the earlier versions we tried a color scale that contained different shades of green. Users commented that because green usually means good or environmentally friendly, it was strange to see high levels of impact and exposure encoded with that color. Instead, we still use shades of green but also incorporate yellows. From low to high values, we incrementally shift from green to yellow, respectively. Trips that have impact or exposure values of zero are white.

Making trips interactive

Users can potentially map hundreds of trips at one time, providing an overview of traveling habits, impact, and exposure, but the user also needs to read individual trip details. Mapping a trip is not enough. Users have to be able to interact with trips so that they know the context of their travels.

When the user scrolls over a trip on the PEIR map, that trip is highlighted, while all other trips are made less prominent and blend in with the background without completely disappearing. To be more specific, transparency of the trip of interest is decreased while the other trips are blurred by a factor of five. Cabspotting, a visualization that maps cab activities in San Francisco, inspired this effect. When the user clicks on a trip on the map, the trip log automatically scrolls to the trip of interest. Again, the goal is to provide users with as much context as possible without confusing them or cluttering the screen.

These features, of course, handle multiple trips only to a certain extent. For example, if there are hundreds of long trips in a condensed area, they can be difficult to navigate due to clutter. This is an area we plan to improve as we incorporate user-contributed metadata such as tags and classification.

Displaying distributions

PEIR provides histograms on the right side of the map to show distributions of impact and exposure for selected trips. There are four histograms, one for each microenvironment model. The histograms automatically update whenever the user selects a trip from the trip log. If trips are mostly high in impact or exposure, the histograms are skewed to the right; similarly, if trips are mostly low in impact or exposure, the histograms are skewed to the left.

We originally thought the histograms would be useful since they are so widely used in statistics, but that proved not to be the case. The histograms actually confused more than they provided insight. Although a small portion of the test group thought they were useful, most expected the horizontal axis to be time and the vertical axis to be the amount of impact or exposure. People seemed more interested in patterns over time than overall distributions. Therefore, we switched to time-based bar charts (Figure 1-3). Users are able to see their impact and exposure over time and browse by week.

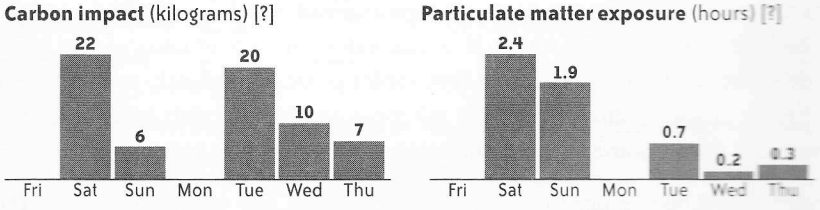


FIGURE 1-3. Time series bar charts proved to be more effective than value-based histograms.

Sharing personal data

PEIR lets users share their impact and exposure with Facebook friends as another way to compare values. It is through sharing that we get around the absolute scale interpretation of axes and shift emphasis onto relative numbers, which better helps users make inferences. Although 1,000 kilograms of carbon might seem like a lot, a comparison against other users could change that misconception. Our Facebook application shows aggregated values in users' Facebook profiles compared against other Facebook friends who have installed the PEIR Facebook application (Figure 1-4).

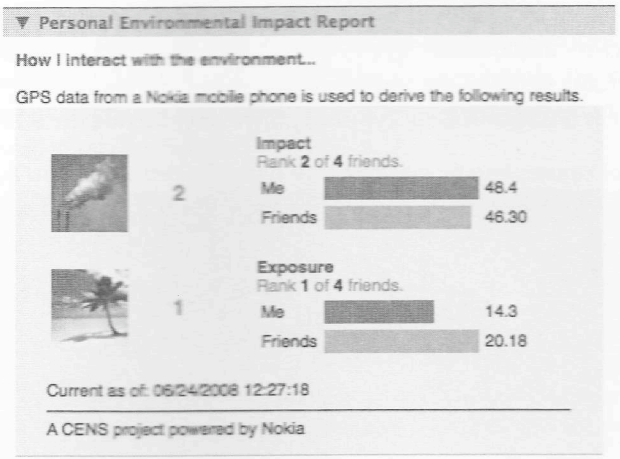


FIGURE 1-4. PEIR's Facebook application lets users share their impact and exposure findings as well as compare their values with friends. (See Color Plate 3.)

The PEIR Facebook application shows bar graphs for the user's impact and exposure and the average of impact and exposure for his or her friends. The application also shows overall rank. Those who have less impact or exposure are higher in rank. Icons also provide more context. If impact is high, an icon with a chimney spouting a lot of smoke appears. If impact is low, a beach with clear skies appears.

Shifting attention back to the PEIR interface, users also have a network page in addition to their personal profile. The network page again shows rankings for the last week of impact and exposure, but also shows how the user's friends rank. The goal is for users to try to climb in the rankings for least impact and exposure while at the same time encouraging their friends to try to improve. Although actual values in units of kilograms or hours for impact or exposure might be unclear at first, rankings are immediately useful. When users pursue higher ranking, values from PEIR microenvironment models mean more in the same way that a score starts to mean something while playing a video game.

The reader should take notice that no GPS data is shared. We take data privacy very seriously and make many efforts to keep certain data private, which is why only impact and exposure aggregates are shown in the network pages.

YFD

Whereas PEIR deals with data that is not immediately relatable, YFD is on the opposite side of the spectrum. YFD helps users track data that is a part of everyday conversation. Like PEIR, though, YFD aims to make the little things in our lives more visible. It is the aggregate of small choices that have a great effect. The visualization had to show this.

To begin, we go back to one of the challenges mentioned earlier. We want users to tweet frequently and work personal data collection into their daily Twitter routine. What are the motivations behind data collection? Why does a user track what he eats or his sleep habits? Maybe someone wants to lose weight so that he feels more confident around the opposite sex, or he wants to get more sleep so that he does not fall asleep at his desk. Another user, however, might want to gain weight, because she lost weight when she was sick, or maybe she sleeps too much and always feels groggy when she gets up. Others just might be curious. Whatever the motivation, it is clear that everyone has his or her own reasons for personal data collection. YFD highlights that motivation as a reminder to the user, because no matter what diet system someone is on or sleep program he is trying, people will not change unless they really want to. Notice the personal words of motivation in large print in the middle of the screen in Figure 1-5.

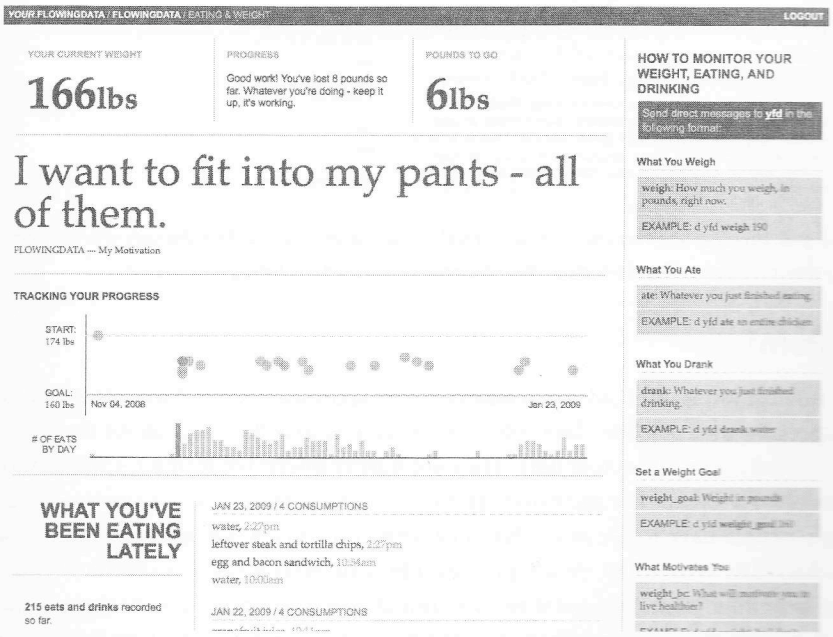


FIGURE 1-5. People track their weight and what they eat for different reasons. YFD places motivation front and center. (See Color Plate 4.)

It is also worth noting that each tracker's page shows what has happened most recently at the top. This serves a few purposes. First, it will update whenever the user tweets a data point, so that the user can see his status whenever he logs in to YFD. Second, we do not want to stray too far from the feel of Twitter, again to reinforce working YFD tweets into

the Twitter routine. Finally, the design choice largely came out of the experience with PEIR. Users seem to expect time-based visualization, so most YFD visualization is just that.

There is one exception, though—the feelings and emotions tracker (Figure 1-6). As anyone can tell you, emotions are incredibly complicated. How do you quantify happiness or sadness or nervousness? It did not seem right to break emotions down into graphs and numbers, so a sorted tag cloud is used instead. It somehow feels more organic. Emotions of higher frequency are larger than those that occur rarely. The YFD trackers are all modular at these early stages of development, but I do plan to eventually integrate all trackers as if YFD were a dashboard into a user's life. The feelings tracker will be in the center of it all. In the end, everything we do is driven by how we feel or how we want to feel.

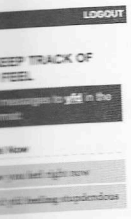
FIGURE 1-6. Users can also keep track of how they feel. Unlike the other YFD trackers, the page of emotions does not have any charts or graphs. A word cloud was chosen to provide more organic-feeling visualization.

The Point

Data visualization is often all about analytics and technical results, but it does not have to be—especially with personal data collection. People who collect data about themselves are not necessarily after the actual data. They are mostly interested in the resulting information and how they can use their own data to improve themselves. For that to come through, people have to see more than just data in the visualization. They have to see themselves. Life is complex, data represents life, and users want to understand that complexity somehow. That does not mean we should dumb down the data or the information. Instead, we use the data visualization to teach and to draw interest. Once there is that interest, we can provide users with a way to dig deeper and explore their data, or more accurately, explore and understand their lives in that data. It is up to the statistician, computer scientist, and designer to tell the stories properly.

perience with
ation is just that.

re 1-6). As any-
tify happiness or
nto graphs and
ganic. Emotions
kers are all mod-
egrate all trackers
in the center of it
ant to feel.



ege of emotions does not
tion.

t it does not have to
bout themselves are
e resulting informa-
r that to come
They have to see
nderstand that com-
ta or the information.
Once there is that
their data, or more
the statistician, com-

How to Participate

PEIR and YFD are currently by invitation only, but if you would like to participate, please feel free to visit our sites at *http://peir.cens.ucla.edu* or *http://your.flowingdata.com*, respectively. Also, if you are interested in collaborating with the PEIR research group to incorporate new models, strategies, or visualization, or if you have ideas on how to improve YFD, we would love to hear from you.