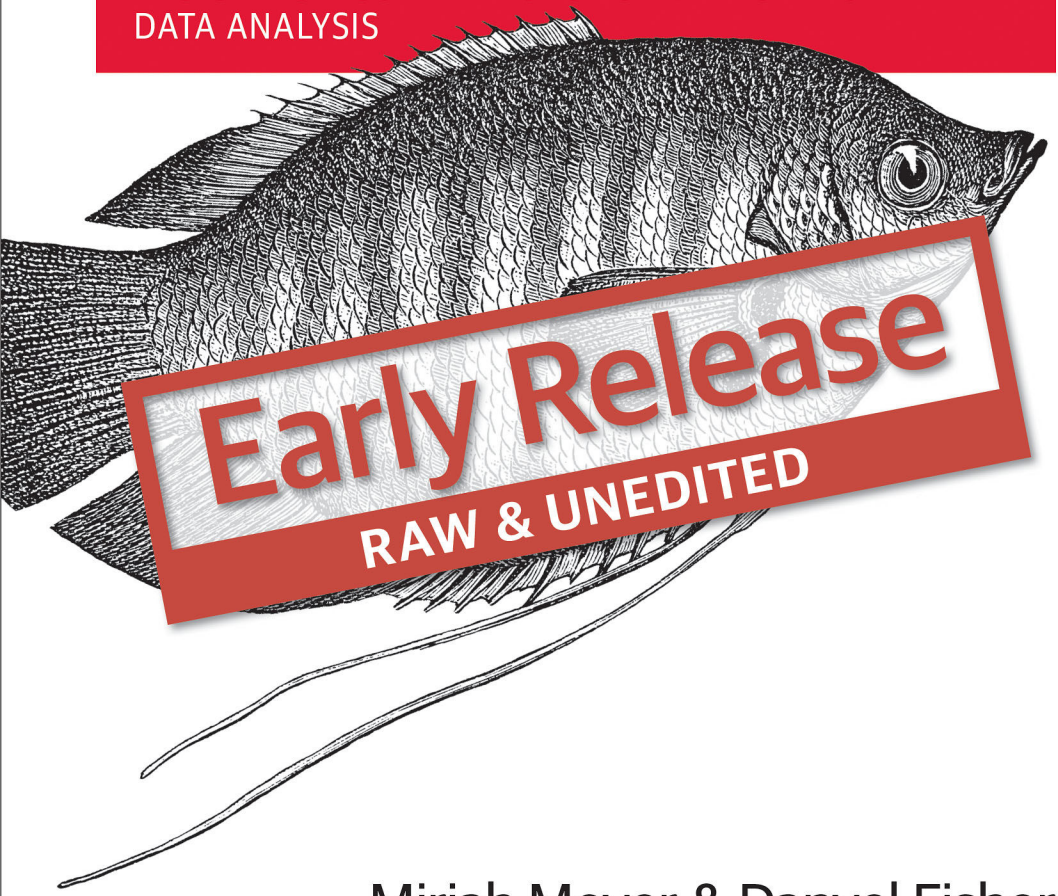
O'REILLY®

Making Sense of Data

DESIGNING VISUALIZATIONS FOR EXPLORATORY DATA ANALYSIS



Miriah Meyer & Danyel Fisher

Making Sense of Data

First Edition

Danyel Fisher & Miriah Meyer

Beijing • Boston • Farnham • Sebastopol • Tokyo O'REILLY®

Making Sense of Data
by Miriah Meyer and Danyel Fisher

Printed in the United States of America.

Published by O'Reilly Media, Inc. , 1005 Gravenstein Highway North, Sebastopol, CA 95472.

O'Reilly books may be purchased for educational, business, or sales promotional use. Online editions are also available for most titles (*http://safaribooksonline.com*). For more information, contact our corporate/institutional sales department: 800-998-9938 or corporate@oreilly.com .

Editors: Laurel Ruma and Shannon Cutt
Production Editor: FILL IN PRODUCTION EDITOR
Copyeditor: FILL IN COPYEDITOR
Proofreader: FILL IN PROOFREADER
Indexer: FILL IN INDEXER
Interior Designer: David Futato
Cover Designer: Karen Montgomery
Illustrator: Rebecca Demarest

April 2016: First Edition

Revision History for the First Edition
2016-04-04: First Early Release

See *http://oreilly.com/catalog/errata.csp?isbn=9781491928400* for release details.

978-1-491-92840-0

[FILL IN]

Table of Contents

CHAPTER 1
Introduction

Visualization is a vital tool to understand and share insights around data. The right visualization can help express a core idea, or open a space to examination; it can get the world talking about a dataset, or sharing an insight.

As an example of how visualization can help people change minds, and help an organization make decisions, we can look back to 2006 when Microsoft was rolling out their new mapping tool, Virtual Earth, a zoomable world map. At that time the team behind Virtual Earth had lots questions about how users were making use of this new tool, and so they collected usage data in order to answer these questions.

The usage data was based on traditional telemetry: it had great information on what cities were most looked at; how many viewers were in "street" mode vs "photograph" mode; and even information about viewers' displays. And because the Virtual Earth tool is built on top of a set of progressively higher resolution image tiles, the team also collected data on how often individual tiles were accessed. What this usage data didn't have, however, was specific information that addressed *how* users were using the system. Were they getting stuck anywhere? Did they have patterns of places they liked to look at? What places would be valuable for investing in future photography?

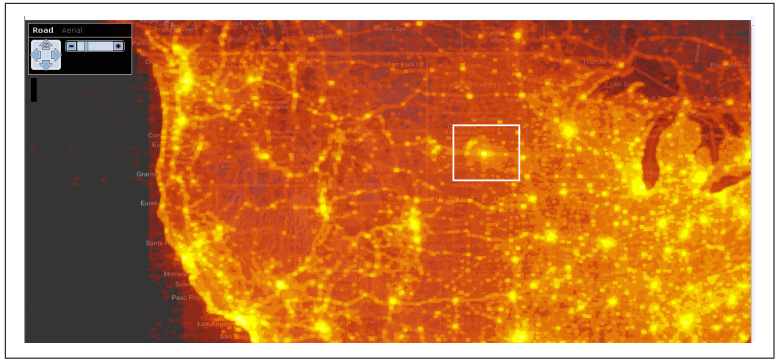


Figure 1-1. Hotmap, looking at the central United States. The white box surrounds the anomaly discussed below.

To unravel these questions, the team developed a visualization tool called Hotmap. Figure 1 shows a screen capture from the visualization tool, focusing on the central United States. Hotmap uses a heatmap encoding of the tile access values, using a colormap to encode the access values at the geospatial location of the tiles. Thus, bright spots on the map are places where more users have accessed image tiles. Note that the color map is a logarithmic color scale, so bright spots have many more accesses than dim ones.

Some of the brightest areas correspond to major population centers — Chicago and Minneapolis on the right, Denver and Salt Lake City on the left. In the center, though, is an anomalous shape: a bright spot where no big city exists. There's a star shape around the bright spot, and an arc of bright colors nearby. The spot is in a sparsely-populated bit of South Dakota. There's no obvious reason why users might zoom in there. It is, however, very close to the center of a map of the continental US. In fact, the team learned that the center of the star corresponds to the center of the default placement of the map in many browsers. Thus, the bright spot with the star most likely corresponds to users sliding around after inadvertently zooming in, trying to figure out where they had landed; the arc seems to correspond to variations in monitor proportions.

As a result of usability challenges like this one, many mapping tools — including Virtual Earth — longer offer a zoom slider, keeping users from accidentally zooming all the way in on a single click.

A second screen capture looks at a bright spot off the coast of Ghana. This spot exhibits the same cross pattern created by users

scrolling around to try to figure out what part of the map they were viewing. This spot is likely only bright because it is 0 degrees latitude, 0 degrees longitude — under this spot is only a large expanse of water. While computers might find (0,0) appealing, it is unlikely that there is much there for the typical Virtual Earth user to find interesting.

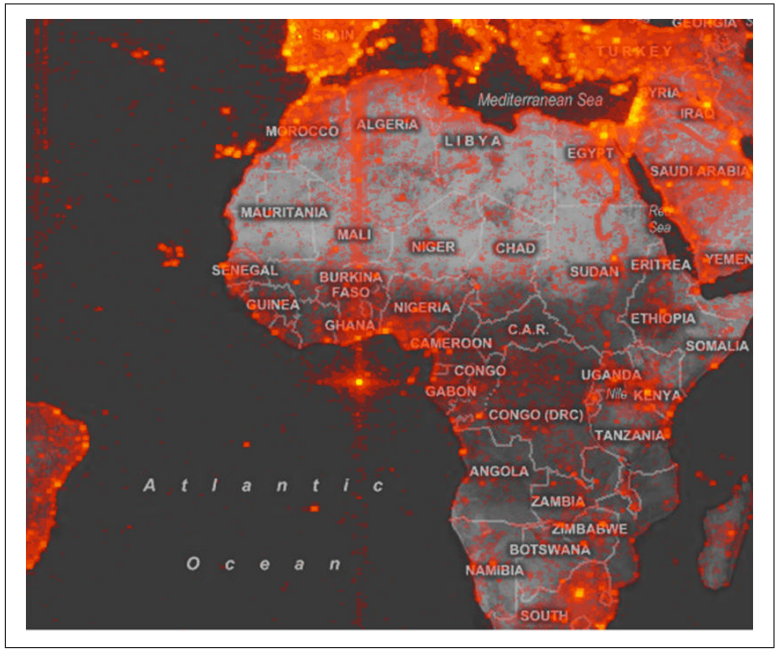


Figure 1-2. Hotmap, looking at the map origin (0,0).

This bright spot inspired a hunt for bugs; the team rapidly learned that Virtual Earth's search facility would sometimes fail: instead of returning an error message, typos and erroneous searches would sometimes redirect the user to (0,0). Interestingly, the bug had been on the backlog for some time, but the team had decided that it was not likely to influence users much. Seeing this image made it clear that some users really were being confused by the error; the team prioritized the bug.

Although the Virtual Earth team had started out using the Hotmap visualization expecting to find out about how users interacted with maps, they gleaned much more than just a characterization of usage patterns. Like many — dare we say most? — new visualizations, the

most interesting insights are those that the viewer was not anticipating to find.

Making Sense of Data

Visualization can give the viewer a rich and broad sense of a dataset. It can communicate data succinctly while exposing where more information is needed or where an assumption does not hold. Furthermore, visualization provides us a canvas to bring our own ideas, experiences, and knowledge to bear when we look at and analyze data, allowing for multiple interpretations. If a picture is worth a thousand words, a well-chosen interactive chart might well be worth a few hundred statistical tests.

Is visualization the silver bullet to help us make sense of data? It can support a case, but does not stand alone. There are two questions to consider to help you decide if your data analysis problem is a good candidate for a visualization solution.

First, **are the analysis tasks clearly defined**? A crisp task such as *"I want to know the total number of users who looked at Seattle"* suggests that an algorithm, statistical test, or even a table of numbers might be the best way to answer the question. On the other hand, *"How do users explore the map?"* is much fuzzier. These fuzzy tasks are great candidates for a visualization solution because they require you to look at the data from different angles and perspectives, and to be able to make decisions and inferences based on your own knowledge and understanding.

The second question to consider: **Is all the necessary information contained in the data set**? If there is information about the problem that is not in the data set, requiring an expert to interpret the data that is there, then visualization is a great solution. Going back to our fuzzy question about exploring a map, we can imagine that it is unlikely that there will be an explicit attribute in the data that classifies a user's exploration style. Instead, answering this question requires someone to interpret other aspects of the data, to bring knowledge to bear about what aspects of the data infer an exploration style. Again, visualization enables this sort of flexible and user-centric analysis.

In the figure below we illustrate the effects of considering the task and data questions on the space of problems that are amenable to a visualization solution.

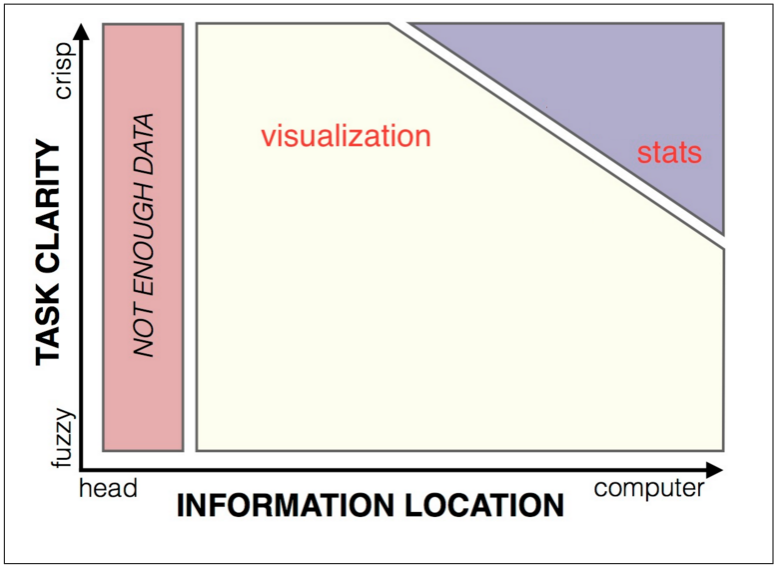


Figure 1-3. The best visualizations combine information in the user's head with system-accessible data

Fairly regularly, someone shows up at one of our offices with a dataset; they want us to help them make sense of their data. Our first step is to consider the fuzziness of the tasks and extent of the digital data in order to determine whether we should begin the process of designing a visualization, or instead throw the data into some statistical software. More often than not, the problems we see benefit in some way from an interactive visualization system.

We've learned over the years that designing effective visualizations to make sense of data is not an art --- it is a systematic and repeatable process. This book is an attempt to articulate the general set of techniques we use to create insightful visualizations.

Creating a Good Visualization

Choosing or designing a good visualization is rarely a straightforward process. It is tempting to believe that there is one, beautiful visualization which will show all the critical aspects of a dataset, that

the *right* visual representation will open the secrets and reveal all. This is often the impression that we, at least, are left with after reading case studies in data science books. A perfect, simple, and elegant visualization — perhaps just a bar chart, or a well-chosen scatterplot — shows precisely what the important variable was, and how it varied in precisely the way that taught a critical lesson.

In our experience, this does not really match reality. It takes hard work, and trial and error, to get to an insightful visualization. We break apart fuzzy questions into actionable, concrete tasks, and we have to reshape and restructure the data into a form that can be worked into the visualization. We have to work around limitations in the data, and we need to try to understand just what the user wants to learn. We have to consider which visual representations to use and what interaction mechanisms to support. And no single visualization is ever quite able to show all of the important aspects of our data at once --- there just are not enough visual encoding channels.

We suspect that your situation looks something like this too.

Designing effective visualizations presents a paradox. On the one hand, visualizations are intended to help a user learn about parts of their data that they don't know about. On the other hand, the more we know about the user's needs, and about the context of their data, the better a visualization can serve the user. In this book, we embrace this paradox: we attempt to weave through the knowledge users do have of their datasets, of the context that the data lives in and the ways it was collected — including its likely flaws, challenges, and errors — in order to figure out the aspects of it that matter.

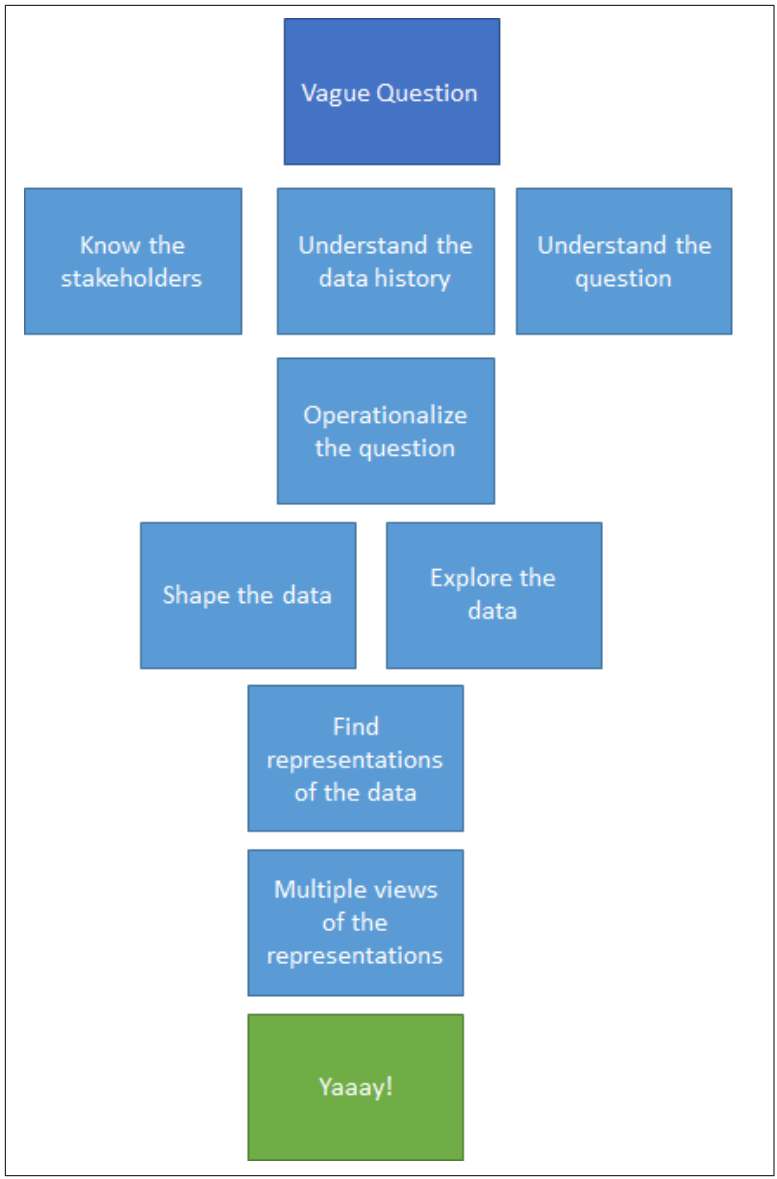


Figure 1-4. The path from ill-formed problem & dataset to successful visualization

Put another way, this book is about the path from *"I have some data…"* to *"Look at my clear, concise, and insightful visualization."* We believe that creating effective visualizations is, itself, a process of

exploration and discovery. A good visualization design requires a deep understanding of your problem, data, and users. In this book, we lay out a process for acquiring this knowledge and using it to design effective visualization tools.

Who are we?

The authors of this book have a combined three decades of experience in making sense of data through designing and using visualizations. We've worked with data from a broad range of fields: biology and urban transportation, business intelligence and scientific visualization, debugging code and building maps. We've worked with teams of analysts spanning small, academic science labs to teams of data analysts embedded in large companies. Some of the projects we've worked on result in sophisticated, bespoke visualization systems designed collaboratively with other analysts, and other times we've pointed people to off-the-shelf visualization tools after a few conversations. All in all, we've thought about how to visualize hundreds of data sets.

We've found that our knowledge about visualization techniques, solutions, and systems shapes the way that we think and reason about data. Visualization, fundamentally, is about presenting data in a way that elicits human-reasoning, that makes room for individual interpretations, and supports exploration. Because of this, we work with our collaborators to operationalize their questions and data in a way that reflects these characteristics. The process we lay out in this book describes our thinking and inquiry in these terms.

Who is this book for?

This book is for people with access to data and, perhaps, a suite of computational tools, but are less than sure how to turn that data into visual insight. If you've found that data science books too-casually assume that you can figure out what to do with the data once collected, and that visualization books too-casually assume that you can figure out what dimensions of the data you need to explore, then this book is for you.

We're not going to teach you in detail how to clean data, manage data, or write visualization code: there are already great books written about these topics, and we'll point you to some of them. (We will

talk about why those processes are important, though.) You will not come out of this book being able to choose a beautiful colormap or select a typeface — again, we will point to resources as appropriate. Instead, we will lay out a framework for how to think about data given the possibilities, and constraints, of visual exploration.

We'll walk through a process that we call *data counseling*, a set of iterative steps that are meant to elicit a wide range of perspectives on, and information about, a data problem. The goal of data counseling is to get to an *operationalization* of the data that is amenable to a visualization solution. This solution may be a series of charts created during the process as you explore the data, or it could be an off-the-shelf, interactive visualization tool that you use after you've operationalized your data. And it some cases, the solution will be a *bespoke* visualization tool that you'll create because your unique problem requires a unique solution.

There are four components to a good operationalization:

Regardless of the visualization outcome, a person going through the data counseling process will make new discoveries and gain new insights along the way. We believe that effective visualization design is about a deep investigation into sense making.

A Note on the History of Data Counseling

Miriah and Danyel jointly, and independently, described this process; we're sure that many other researchers carry out similar processes. One of us jokingly calls it "data psychotherapy." (The other, more reasonably, named it "data counseling.") It starts, not uncommonly, when people walk into our office:

```
CLIENT: I have some data that I'd like to visualize
Q: What about the data would you like to visualize?
CLIENT: I think the data would show me how profitable our stores are.
Q: What does it mean for a store to be profitable?
CLIENT: It means that the store has lots of sales of high-profit items.
Q: Why does profit vary by store?
…
```

And so on. By the end of this process, we would often find that the user had described the dimensions they found most important—the outcome measure (profit); the relevant dimensions upon which it might vary (which store, which item); and so on. They key step, however, was stepping away from the data to ask what end the user truly wanted to accomplish — "to persuade my boss to increase my

department's funding", or "to find out whether our users are happy", or "to change the mix of products we sell". Once we'd articulated these questions, finding an appropriate visualization became much easier.

In this book, we systematize this process into what we hope are reproducible and clear steps.

The rest of this book

In Chapter 2, we describe the Operationalization Tree. The Tree is the core technique that gets us from high-level user needs down to specific, actionable questions. We'll discuss how to narrow a question from a broad task into something that can be addressed with a sequence of visualizations. For example, the broad question *"how do users use our maps?"* does not necessarily suggest a specific visualization – but *"what places do users look at on our maps?"* can leads very clearly to a visualization like Hotmap.

In the next chapter, Chapter 4, we'll translate from the high-level concepts to low-level visualization components. We will discuss concepts like dimensions and measures, and how to identify them in your data. We'll talk about the broad set of tasks that can be carried out with a visualization, and we'll connect those to tasks that we identified in Chapter 2.

An operationalization is not born, fully formed, from the skull of a visualization expert: the data has a history and pedigree; people have created and collected it. In Chapter 3, we lay out an iterative set of steps for getting to an operationalization, which we call "data counseling". This process is about working with data owners and other stakeholders to delve deep into an analysis problem, uncovering relationships and insights that are difficult to articulate, and then using that knowledge to build an effective operationalization. The process describes the kinds of questions to ask, who to ask them of, and how to rapidly explore the data through increasingly sophisticated prototypes.

With this technique for operationalizing, and for collecting information from interviewees, in mind, we turn to the visualizations themselves. In ???, we'll discuss the core types of visualizations. We'll start with the familiar, such as bar charts, scatter plots, and timelines, and move on to some less well known variants. For each class, we will

describe the types of data that fit on them, what sorts of tasks they can address, and how they can be enhanced with additional dimensions.

Often, more than one visualization may be necessary to examine a complex, real-world dataset. Infographics and dashboards commonly show several different visualizations; we can apply interactive techniques to build richer connections. ??? talks about multiple linked view visualizations. These linked views employ individual visualizations, tied together through user interaction, to support a very rich and complex set of tasks and data constraints.

For example, overview+detail can be a good solution to visualize lots of data, but requires a good way to meaningfully summarize and aggregate the data. A complex data set with many different attributes might suggest a multiform visualization, which allows the users to examine the attributes contrasted against each other in pairs or triads, linked across different views. Chapters four and five together form the core knowledge that is necessary to have in order to know what kinds of visualization solutions are possible.

With this understanding of creating a visualization—from data to visualization—we might consider declaring victory and going home. The remainder of the book gives us tools for carrying out these steps.

In ???, we present two case studies that focus on the how we applied the data counseling process to real-world problems. These problems illustrate the flexibility of the process, as well as the diverse types of outcomes that are possible.

??? addresses the design process. We discuss design iteration and rapid prototyping; and we discuss some of the tools we use for deciding how well a visualization suits user needs. We discuss considerations that we've found meaningful for creating effective tools: the role of aesthetics; the difference between exploratory and explanatory visualizations; and value of bespoke visualizations.

??? discusses shaping and reshaping data, data cleaning, and tools: those that are intended for reshaping data into the shape we need; and then tools for visualizing data. The latter will range from tools oriented toward programmers (those implemented over Java, Javascript, and Python) through those oriented toward data scientists and data users, such as R, Tableau, and even Excel. As we will see,

there are many tradeoffs to these different tools; some are excellent in one context, but cannot fulfill needs in another.

??? touches on some challenges of encountering data in the real world: collecting, shaping, and manipulating it.

There is a lot that will not be covered in this book, such as the perceptual aspects of visualization, human factors components of interfaces, or how to use a variety of visualization toolkits. We do, however, included references to these types of issues along the way. We also provide a github site, *http://shouldhaveaname.github.com*, where a reader can download the code to regenerate many of the book's figures. We're not claiming these are the right implementations — or even particularly good code — but we feel the reader should be able to use this as an opportunity to see what it takes to carry out these operations.

CHAPTER 2

Operationalization, from questions to data

In this chapter we look at how to turn data, and a question, into more meaningful tasks. More specifically, we discuss the notion of *operationalization*, the process of refining high-level, problem-specific questions into specific tasks over the data. The operationalization of a problem provides concise design requirements for a visualization tool that can support finding answers to those questions

The concept of operationalization appears across data science: the idea of transforming user questions into data-driven results can be found in dozens of references. Most commonly, we hear only about the successful design and data choices: a chart of the perfect dimensions that throws a phenomenon into perfect focus. This is also common in popular press retellings of data science --- "when the data scientists analyzed shoppers' check out data, they realized that people who bought soda often bought nothing else." This is, however, only half the story. What inspired the analysts to look at soda sales? Were they looking at the shopping cards of people who bought just one thing, or the co-purchasing behavior around soda, or did they look at a dozen other products before noticing this?

Data analysts often begin with different questions and goals than where they end up, and these questions are often underspecified or highly abstract. "Do our users find this game fun to play?" "Which articles on our site are selling ads?" "Which donors should we keep

track of for outreach in the next year?" The process of breaking down these questions into something that can actually be computed over the data is iterative, exploratory, and sometimes surprising. Operationalization is the process of making decisions that lead to an answer.

What makes operationalization for data visualization different? In many fields, operationalization is a process of reducing a process to a single, or small number of metrics, and attempting to optimize that metric. Here, we read operationalization more broadly: we are not trying to merely identify a *single* metric, but to instead choose the techniques that allow the analyst to get a usable answer. Visualization, though, is not an inevitable outcome: as we explore the data, we might realize that our goal is best answered with a machine learning algorithm, or a statistical analysis.

Visualization has the unique feature of allowing users to explore many interrelated questions, and to get to know how data looks from different perspectives. Complex, vague tasks require looking at a number of different dimensions to understand what they mean, and how they interact. We can ask how a variety of metrics relate to different parts of the data. Visualization allows us to explore data in an open-ended way.

In this chapter we outline a framework for articulating a set of analysis tasks over the data, tasks that can then be explored, in unison, in a visualization system. We detail a specific set of considerations for breaking down a high-level, abstract question into these tasks. But before we get into the details of the operationalization, we are going to start with a motivating problem which we will use to ground our discussion throughout the rest of the chapter.

Example: Understanding the Design of a Transit Systems

Our example looks at the design of a public transit system. Or more specifically, the questions that a geography colleague has about the effects of the system design on the local community of residents. We'll follow this example in order to better understand how to operationalize a complex question, and look at several different paths toward making sense of it.

We collaborated with Dr. Steve Farber, a geographer interested in this characterization who is studying the Utah Transit Authority (UTA), the public transit system that services Salt Lake City and the surrounding areas. [1] Steve's research focuses on a core concern of transit design: "how do the tradeoffs between removing cars, versus servicing those who rely on transit, play out?"

This is a well-known trade-off within public transit design: there are very different priorities to taking cars off the road versus servicing economically-disadvantaged residents who rely on transit to get around, and they require very different implementations. If a system is designed to take cars off the road it would be as efficient as possible when going between the most-popular points at the busiest times, making the transit system competitive with cars. If the goal is to serve people without cars, however, it would need to adequately — if never highly efficiently — serve low-income neighborhoods for a broad set of times throughout the day. Furthermore, not only do transit designers need to optimize against these competing needs, but they also have to design around legacy routes, road designs, and political influences.

Due to the challenges inherent in designing a public transit system it is important to be able to characterize how design decisions affect the core efficacy of the system in order to steer improvements and refinements for the better.

This question is, as phrased, poorly defined. There is no single source of data that labels the tradeoffs explicitly; the planners themselves most likely never addressed the question directly. Furthermore, there are many different ways we might imagine trying to answer it. Can we look at ridership data? Can we survey residents? Could we just interview the transit designers themselves?

Our goal of operationalization is to refine and clarify this question until we can forge an explicit link between the data that we **can** find and the questions we'd like to answer. As a first step, we asked our collaborator **what data is actually available**?

Steve computed a **time cube** based on the UTA time tables that stores, for every pair of locations in the Salt Lake Valley and for each

1 Our thanks to graduate students Josh Dawson and Sean McKenna, who have been working with us on this collaboration.

minute of the day, the time it takes to travel on existing transit routes [2]. The cube was generated using a sophisticated algorithm that considers not only the fastest transit option between two locations, but also walking and waiting times at pick-up stops. Thus, the cube can tell us that it takes 28 minutes to get from A location to B at 5:01 am, but at 4:03 pm it takes 35 minutes. There is one cube for the weekdays, and one for weekend schedules.

Additionally, he collected a number of government census datasets that characterize the neighborhoods in and around Salt Lake City, and the people who live there. The travel cube shows how long it takes to go between places, the census data helps us understand how many people go between these pairs of places, and how often --- and perhaps what sorts of places they are. It allows us to ask which districts have the most wealthy or poor people; it allows us to ask what places tend to be the origins or destinations of trips, and so to characterize areas as job hubs or residential areas. Along with demographic information of the people in each neighborhood, the census data also tracks, [3] for pairs of neighborhoods, the income distribution for the people who commute between them for work.

Our collaborator computed the travel times for each block in the region. The travel cube allows us to ask questions like "how long does it take to get between block A and block B at a given time?".

The census data provides a much richer analysis. Specifically,. While the data is at different granularities, but combining them might allow us to ask questions like "for each district, how long does it take the people in the highest income bracket to get to work by transit?"

Now that we have data and a high-level question, our visualization work begins. Data alone is not enough to dictate a set of design requirements for constructing a visualization. What is missing here is a translation of the high-level question — understanding the trade-offs in the transit system — into a set of concrete task that we can perform over the data. And that's where operationalization comes in. We'll dig further into this example after describing a construct for guiding the translation: the operationalization tree.

2 cite paper

3 *http://www.census.gov/hhes/commuting/*

Before continuing, though, it is worth noting that the data and the operationalization are fundamentally a *specific* perspective on a problem: they are proxies for what we are trying to understand. In this UTA example there are other ways that our collaborator could have framed his inquiry, and other types of data he could have collected. This is a large part of why visualization is so important for answering questions like these as it allows an analyst's experience and knowledge to layer directly on top of the actual data that is ultimately shown.

The Operationalization Tree

The core process of operationalization is the route from a general goal or a broad question, to specific questions, and to visualizations based on concrete data. We begin with a broad question that describes a research interest, or a business goal, or that orients a data exploration. We go through a series of stages meant to refine the question, based on the knowledge of the problem, needs of stakeholders, what data is available (or can be collected), and the way the final audience will consume it.

Carrying out this transformation requires collaboration with stakeholders: to learn what data is available, and how the results will be used. Interviews help us identify the questions and goals of the stakeholders with respect to the data, and understanding what data is available, or can be made available. Throughout the transformation we use operationalization to translate those questions and goals into a description of the problem that is amenable to a data solution. We'll talk more specifically collaboration techniques --- specifically interviewing and prototyping --- in Chapter 3.

The operationalization tree is a construct that represents a process of refining a broad question into a set of specific tasks that can be performed over the data. The root of the tree is the high-level question that the stakeholder wishes to answer; the internal levels represent mid-level tasks that describe goals using the language of the problem space; and the leaves represent specific tasks that can be performed over specific data measures, often utilizing a visualization.

A data analyst constructs the tree from the root, exploring both depth and breadth. The construction of the tree represents the continual refinement of tasks into computable chunks. Once leaf nodes

are defined and tasks resolved, the solutions are propagated back up the tree to support answering higher level tasks.

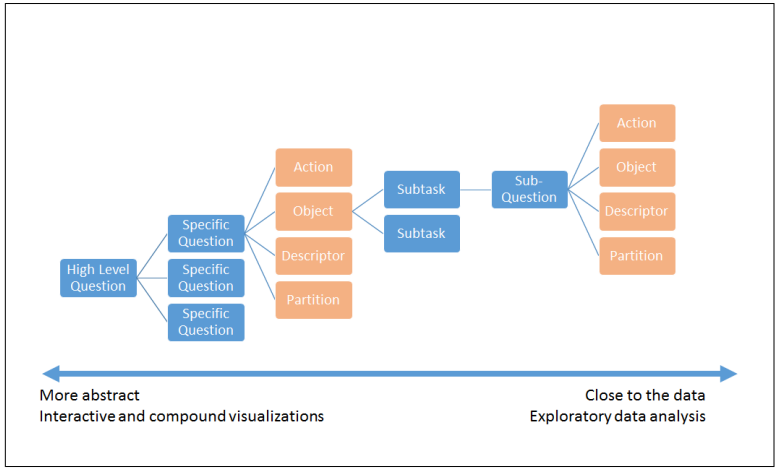


Figure 2-1. Recursive representation of the operationalization tree. The question is rephrased as one or more tasks; each task in turn is separated into an action, several objects of the action, and a descriptor.

Building an operationalization tree begins with a high-level question or goal for the project. The general question might be a research goal, or a general question about user behavior, or a specific aspect we wish to improve or change. In the UTA scenario, the question we begin with is "How do the tradeoffs between removing cars, versus servicing those who rely on transit, play out in the UTA system?" From there, we go through the following steps to build the tree:

1. Refine the question into one or more **tasks** that, individually or together, address the general question.
   - If the task is unambiguous and we can figure out what visualization, background knowledge, or computation will address it, we do so.
   - If the task **is** ambiguous, break it down into four components **actions, objects, descriptors, and partitions**--- looking for undefined terms and ambiguous phrases.
2. Define the objects, descriptors, and partitions by creating a new **question** that addresses each one, and return to step 1 with those questions.

3. Lastly, once tasks have been addressed, propagate the results back up to support higher level tasks.

The root question is the most difficult one to translate into a task. This translation in particular relies on the data counseling process, as well as from a detailed understanding of what data exists and is available. We discuss this further in Chapter 3.

After selecting a task, particularly one that is abstract, fuzzy, or ambiguous, the next step is to identify the *four components* of that task. We use these components as a guide to finding the more specific questions, and then tasks, that will provide the next step downard in the tree:

- **Actions**: Actions are the words that articulate the specific thing being done with the data, such as *compare*, *identify*, or *characterize*, etc. Actions are helpful for identifying the other components, and can help choose visualizations.
- **Objects**: Objects are things that exist in the world; these are the items which respond to the action. "A neighborhood," or "a store," or "a user" are all objects.
- **Descriptors**: The value that will be measured for the objects. "Effectiveness of the transit system," or "happiness of a user", or "sales of a store" are all descriptors.
- **Partition**: Logical groups of the objects. "The western vs eastern region of stores," or "Players, divided by the day they started playing," or "players, partitioned by whether they have bought an upgrade."

Every task will have an action, and this verb is useful for identifying the other components. Take this task, **"Compare the amount of money spent in-game by players who play more hours to those who play fewer hours".** Here, the action is *compare*, which is useful for determining the object. The objects in this task are the things we want to compare; we want to compare *players*. But, what is it about players we want to compare? That is the descriptor, which in this example is *money spent*. Finally, there is a specific partitioning of the objects. We don't just want to compare **all** players, we specifically want to compare two groups --- those that play many hours and those that play few hours.

Example 2-1. Exemplar Task for a Game

Task: Compare the amount of money spent in-game by players who play more hours to those who play fewer hours.

Action: compare

Object: players

Descriptor: money spent

Partition: players who play many hours; players who play few hours

Given this breakdown of the task we can now figure out where we need to further refine our descriptions. We do this by considering the question "Are the object, descriptor, and partition each directly linked to the data?" For each of the three, do we know specifically which aspect of the data it represents, or how to derive it from the data? If not, we formulate a subquestion in order to derive a more specific answer. In Example 2-1, the partition divides between "many" and "few" hours. We will need to divide further, so we ask a new question: "In our game, how many is 'many' hours for a player?"

Not all tasks will have a partition. Sometimes the task is meant to occur over the full set of data specified by the objects and descriptors. We'll see some examples of this when we discuss the operationalization tree for the UTA example below.

The Leaves of the Tree

Just how far does the tree go down? There's no specific answer to this question: the answer really is, "far enough." We know we've made it all the way down one branch of the tree when the task is directly actionable, using data at hand. We know how to describe the objects, descriptors, and (optional) partitions in terms of the data --- where to find it, how to compute, and how to aggregate it. At the leaf of the tree, we finally hit a questions like: "What ARE the top ten most-populated census blocks?" "What products DO sell the most across our stores?" We know what the question will look like, and we know what we can do to get the answer.

Low-level **objects** can be interpreted from the data. It may be that we can read the data directly off the table, but it may be more indirect: we may need to carry out transformations on it – whether

mathematical transformations or database joins. However, by the time we get to a leaf, the definitions should be unambiguous. The leaf-level objects directly describe items in the dataset. Similarly, partitions at the leaf level will specifically describe a dimension of the data over which a logical set of objects can be created.

Descriptors will turn into measures and metrics: a good descriptor is a concrete number. On the way to the leaf, we have used proxies: " we can't directly quantify a "convenient transit schedule," but we can estimate "a bus comes at least every fifteen minutes". In the leaf note, the descriptor is a precise, measurable dimension.

To solve a leaf, then, is to answer the question—whether as a number, or as a visualization, or even as an interaction. We might decide that the right answer to "many hours" of gameplay is "six hours"--a number—or "the hours played by the top 10% of players"--a formula —or "above the logical breakpoint", which might be represented by a distribution. We can now propagate the leaf's results back up.

Flowing Results Back Upwards

When a low-level task is completed, we have a refined, specific result. We can list the top most-populous counties, or the distribution of hours that players have spent on the game, or the number of minutes that it takes to get between two places, by hour of the day.

We now propagate that task upwards. We had collected this highly refined data for a reason: we were fulfilling a more abstractly-defined task. This low-level fact is one piece of the task above. We may have defined what an object will be, or confirmed that a descriptor would be a measurable and meaningful choice on this dataset, or refined a task verb.

Sometimes, though, we don't have an exact answer and instead we have a visualization that helps an analyst in making a decision, a decision that might be different under different conditions in the the context of other data. As such, we will sometimes propagate up a visualization, or even interactive tool, that allows the decision-making to happen at a higher level of the tree.

Through refinement of the objects, descriptors, and partitions we eventually propagate up answers to the task at hand: and thus, we build a higher-level visualization. The visualization design requirements are built from the propagated results from the lower levels of

the tree. For example, we might decide to allow a high-level user to select between several different descriptors.

Applying the Tree to the UTA Scenario

Going back the UTA example, we left off with the question of "How do the tradeoffs between removing cars, versus servicing those who rely on transit, play out in the UTA system?" While there are several different ways to address this question, such as surveying riders or interviewing the system designers — Steve's approach to this question entails analyzing the travel times inherent to the transit system.

The high level question cannot be addressed directly with the data, and so needs to be refined. We asked Steve what would answer this question. Steve clarified that he can see the choice as a tradeoff: comparing the effectiveness of the transit system for removing cars versus supporting people who rely on transit for their transportation. This is the root of our operationalization tree.

- **Question:** How do the tradeoffs between removing cars, versus servicing those who rely on transit, play out in the UTA system?
- **Task:** Compare the effectiveness of the transit system for removing cars versus supporting people who rely on transit for their transportation.

From this task we identify the four task components in order to help guide our operationalization refinement.

- **Action:** compare
- **Objects:** removing cars; people who rely on transit
- **Descriptor:** effectiveness of the transit system
- **Partition:** (none)

Now we can use these components to help refine the ambiguity in this task. We start with the objects, and ask "Are the objects directly linked to the data?" The answer is no, we do nothave anything in our data that specifies people's decisions about cars and driving.

Instead, what we do have from the census data is information about the salaries of people living in the different census blocks, as well as the blocks' populations and some of the most popular commutes. We want to find a proxy that will get us, at one end of the scale, cars

that can be replaced by transit; at the other end, people who are dependent on transit. One reasonable choice is to decide that high-income earners are more likely to be able to afford a car, while low-income earners are less likely to. We decide to use income as a reasonable proxy for car ownership.

This leads us to a new, lower level node in the operationalization tree where we refine the objects.

- **Question:** How do we define removing cars and people without cars?
- **Task:** Identify census blocks as high- or low-income.
- **Action:** identify
- **Objects:** census blocks
- **Descriptor:** income
- **Partition:** high-income; low-income

Again, we look at the object and ask whether it is directly linked to the data — here, this object IS clearly defined. Census blocks are the items contained in the census datasets. We move on to the descriptor and ask if it is directly linked to the data. The answer is no, we don't have any direct measure of income for census blocks. This leads us to create yet another, even lower level node in the operationalization tree.

- **Question:** What does it mean for a census block to be high- or low-income?

What we do have from census data is the number of people in each census block whose salaries fall within 1 of 3 brackets: people making less than $1250 a month, people making between $1250 and $3333 a month, and people making more than $3333 a month. We can try different choices here: we can do is compute, for each census block, the ratio of the number of workers in the highest bracket to the total number of workers, along with the ratio for the lowest bracket.

- **Task:** Characterize census blocks by the ratio of the number of people at different salary levels to the total population.
- **Action:** characterize
- **Objects:** census blocks

- **Descriptor:** proportion of of people to the population
- **Partition:** salary bracket

The objects are, once again, directly linked to the data; the descriptor and partition are directly computable from the data. Thus, we have reached a leaf node and can solve the task. In this case, we compute, for each census block, the ratios of the numbers of workers in each salary bracket to the total number of workers in the block. We have created a new dimension in the data set.

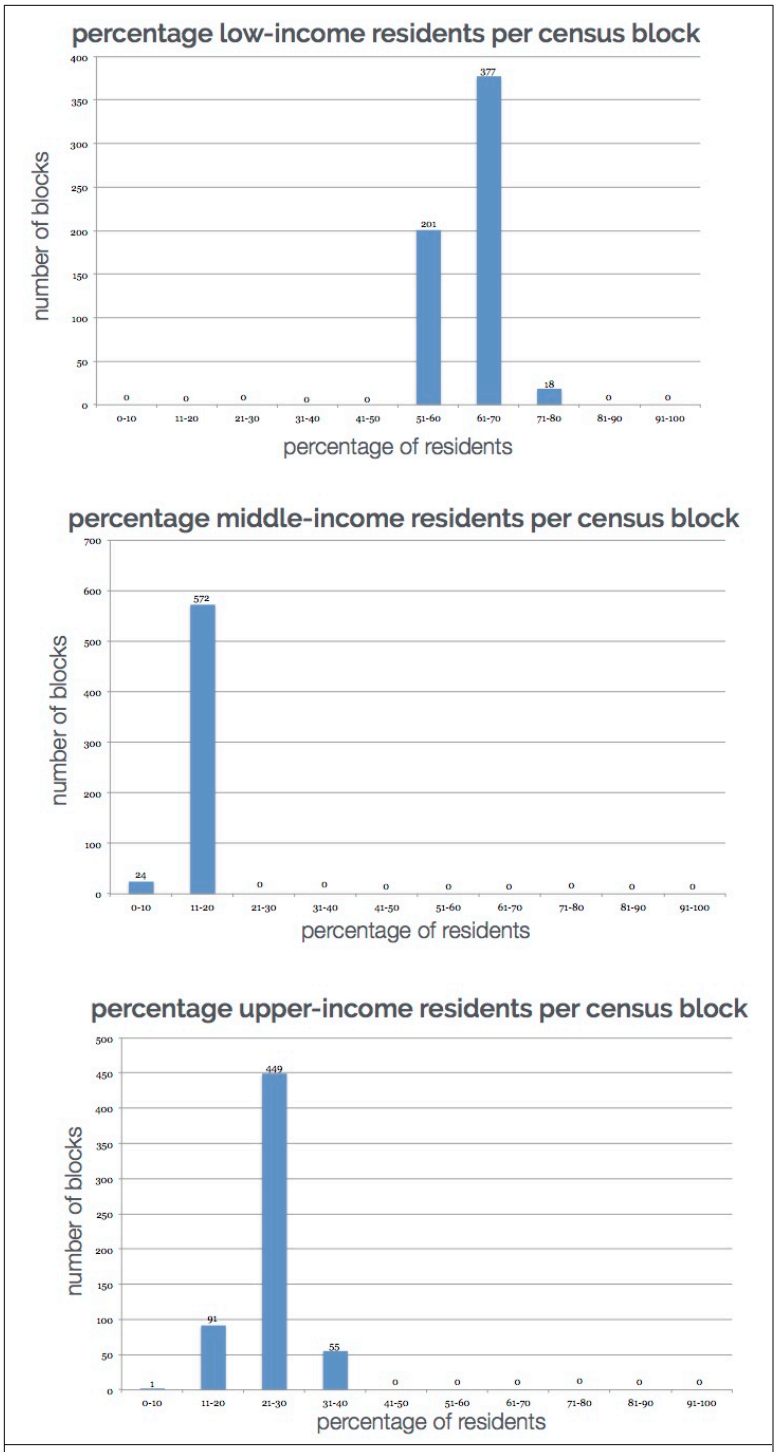


Figure 2-2. Distributions of the number of residents within the three salary brackets. (a) People making less than $1250 per month. (b) People making between $1250 and $3333 a month. (c) People making more than $3333 a month.

Having solved the leaf node we can now flow the result back up the tree. That solution substitutes for the object "taking cars off the road". In this manner, the tree helps structure the operationalization as both a guide for where to refine, as well as a bookkeeping mechanism for aggregating proxy solutions to support a high-level question.

As we bubble up, we can check on these assumptions. We might have mistakenly chosen blocks that have low population—and so do not take many cars off the road. Or we might have chosen blocks that have very few commuters from them. We can generate maps to check these.

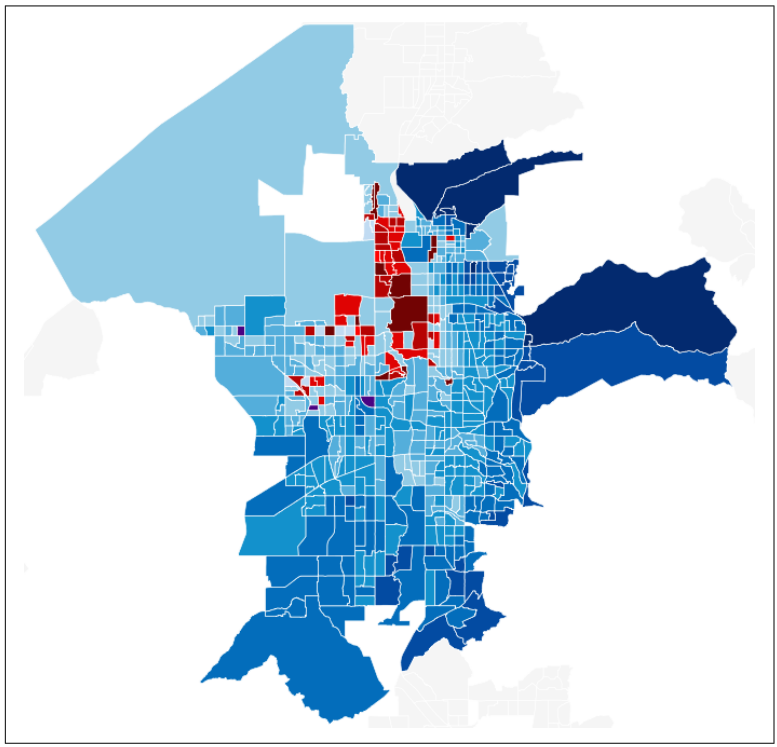


Figure 2-3. An example choropleth showing the census blocks with a larger percentage of higher income residents in blue, and larger percentage of lower income residents in red. Several purple blocks indicate an equal number of high and low income residents.

These questions force us to dive deeper. Looking at these images side by side, it is less clear what we mean by "rich" or "poor" neigh-

borhoods. Is it more important to look on neighborhoods with the most commuting, or the ones with the most people? Is a rich neighborhood with a high population of rich people, or a high percentage?

We know, for example, that we can determine rich neighborhoods by thresholding the ratio of the number of residents with salaries in the highest bracket to the total number of residents. This threshold is fuzzy, however, so visualization is important to help us characterize an appropriate value.

Now we begin to look at the second branch of the tree, based on the question of effectiveness of the transit system. We know that we're going to want to **compare** "effectiveness of the transit system" between two classes of uses.

"Effectiveness of the transit system" might mean different things. It could mean that we can get from one place to another quickly; or that we can get between two places at all. It might mean that we can get between places if we're willing to synchronize our watches — with infrequent express busses—or it might mean that we can wander out of our apartment and know that a bus will come shortly. It might mean a fast fastest trip of the day, or it might mean a low variance (taking the bus always takes at least 45 minutes, but never more than an hour).

For step 5, lets drop further to explore "What does effectiveness mean?" We will need to take this on in stages, but a first concrete task would be, "what is the distribution of travel time between some pair of locations?"

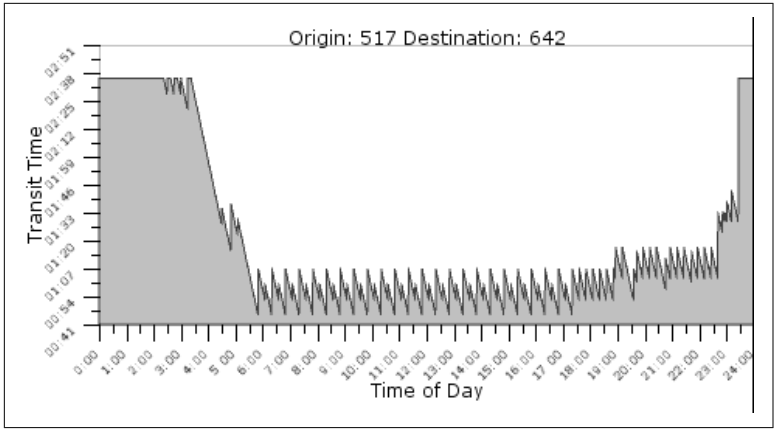


Figure 2-4. An different origin/destination pairs between two points, by time of day. The sawtooth pattern occurs because being one minute early for a bus means catching the it a minute later, while being one minute late for a bus means waiting for the next. Narrower teeth, closer together indicate high frequency; wider teeth indicate lower frequency. Note shifts between daytime schedule and evening; and between commute hours and midday schedules. Around midnight, when the bus service doesn't run, the horizontal line suggests it is just as effective to walk.

We learn a few insights from this chart. There are some optimal times to get a bus. If the rider is a minute early, the may have to wait an extra minute. If they are a minute late, however, they may have a long wait. Perhaps a very long wait. We learn that there are different times—daytimes, and evenings, and commute hours.

To choose a definition of "effectiveness" means picking a specific task that we are trying to measure. We would tend to converge on this using, again, a data counseling and interviewing process, talking to customer groups.

This reflects somewhat on the next phrase: "removing cars" presumably means "letting people with cars leave them at home." Walking through that further, we don't expect them to abandon vehicles for errands, or ski trips — but we might expect them to commute to work without a car. "Servicing non-car owners," is a different question. Presumably, those users need transit to get them to lots of locations, at variable times, but might be less concerned about whether a trip is competitive with driving.

These two groups clearly have different senses of what "effective" means to them.

Note that we've avoided picking a single metric here. Because we see this as a complex and multifaceted problem that we'll really use for later investigation, we'd really most like to develop a sense of what the major issues involved here are. Visualization allows us to explore a richer story. For this project, we'll choose several different times—to examine both rush hour and off-commute times; we'll look at several different measures.

"Compare the *speed, and variance the transit system* for *wealthy people* versus *less wealthy*, at both *commute and non-commute* hours."

This process of laying out the choices can seem somewhat painstaking. It is far too easy to skip these steps, and not bother to record what operationalizations were chosen — the visualizer might not think of them, and might make arbitrary decisions around the data; the decider might not realize that these decisions can make for important differences in the data.

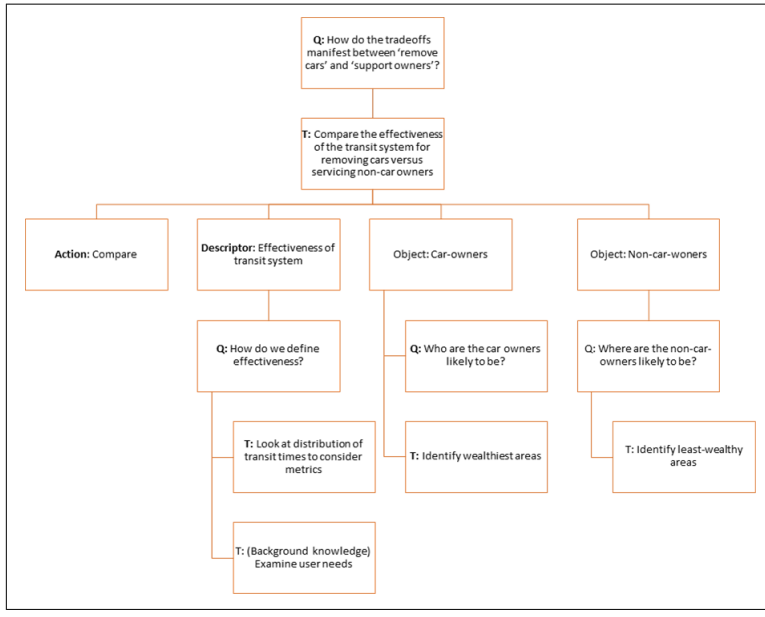


Figure 2-5. Schematic of the tree that describes the UTA example.

This is, of course, just the first pieces of the entire tree.

Visualization, from Top to Bottom

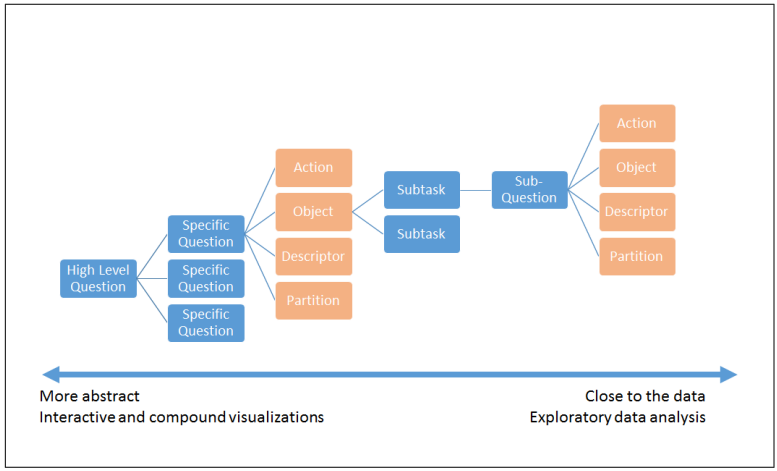


Figure 2-6. Schematic of the tree in general. Compound, interactive visualizations are more likely to address abstract questions, while specific subquestions are answered by smaller, specific visualizations and exploratory analysis.

One interesting aspect of the tree is that as the questions get more refined, and closer to the data, the visualizations become less complex. We may need several visualizations, multiply compounded and interconnected, to work our way through the proxy measures and detailed interactions. On the other hand, the subquestions tend to be simpler, and closer to the data. They require single views, and the questions are less complex.

The base of the tree, then, is grounded in small, coherent questions about the data, and ways to visualize it; the apex of the tree is about bigger, more abstract questions and proxies.

Conclusion: A Well-Operationalized Task

We can now define a well-operationalized task, relative to the underlying data.

- it can be computed based on the data
- It makes specific reference to which fields of the data it's looking at

- It follows down the tree to have specific definitions for the terms it uses

A well-operationalized task, at any level, can be a specification for a visualization. In Chapter 4, we will begin to describe the ways we translate these components into aspects of a visualization; in ??? and ???, we'll actually construct visualizations.

Written out, this process seems tedious. In practice, we do not carry out every step, each time. However, we find two important uses to this formulation. First, the process of explicitly looking at these steps can help untangle knotty problems, decomposing places where we make assumptions about the data. Explaining precisely *why* "income" is a proxy for "car ownership" allows us to explore whether it is a good choice — and, perhaps, to revise that choice later.

The second use is in guiding our questions and interviews. As we will discuss in Chapter 3, the process of carrying out this operationalization entails working with domain experts. Climbing down the tree can help guide conversations; it can also help keep them on track. Every dataset has subtleties; we find that it can be easy to slip down rabbit-holes of complications. We find that walking through the tree can help focus our conversations with experts, only introducing the complications when needed.

This chapter describes the operationalization tree, a framework for refining questions into tasks that a visualization can support. We walked through a task, from the general question to the specific tasks at the base, and showed how we'll use this to understand a broad question about the Utah Transit Authority. In Chapter 4, we're going to take a closer look at the components of an operationalization, discussing how to translate these pieces into visualization decisions. After that, in Chapter 3, we'll talk about techniques for working with stakeholders to help in the construction of an operationalization tree, including interviewing and prototyping techniques.

For Further Reading

This process is similar—and in many ways parallel—to the GQM ("Goal, Question, Metric") process found in the software engineering space. The goal of GQM is to refine from a genereal goal to a specific metric; usually oriented around process improvement so

that the consumer can have a single number that helps know whether they are succeeding in improving that process. Our process is more exploratory, and often comes earlier in the cycle. If in GQM, the goal were to "improve user retention," our operationalization might start instead with "how is user retention doing?" with the awareness that the problem is multifaceted and complex.

GQM

- [BasiliGQM] Basili, Victor; Gianluigi Caldiera; H. Dieter Rombach (1994). "The Goal Question Metric Approach." Encyclopedia of software engineering, 1994. Wiley.

CHAPTER 3
Data Counseling

In the previous chapter, we discussed the Operationalization Tree. We discussed the many decisions that are used to populate the tree: the broad question must be turned into more specific tasks; the tasks must be decomposed into specific actions, objects, and descriptors; visualizations must be built that address these tasks. This process requires sophisticated domain expertise, knowledge of the data and the problem space, and a sense of what would make the right answer to the question.

In this chapter we discuss a process, which we call **data counseling**, that provides guidance about how to work with stakeholders to both develop the operationalization of their problem and to design visualizations that support it. We chose this name because working with stakeholders is a back-and-forth series of interviews, of diving deeply into a user's intents around data, and of understanding the stories of where the data comes from, what problems are associated with it, and what it can mean [1].

In this chapter we lay out steps for creating an effective operationalization and the resulting visualization tool. Theses steps are interweaved with exploring data, developing visualization prototypes, and collecting feedback on these preliminary results. For a major visualization project this can be an elaborate and complex process

1 This process is closely related to 'task analysis', in interface design. The distinction is that task analysis is typically oriented toward creating interfaces; we, instead, are working with data, which warrants a unique set of considerations on the part of the designer.

with multiple interviews and prototypes; for a smaller project, however, this might entail just one or two informal interviews and throwing the data into a graphing program, like Tableau.

Why is this hard?

Turning a dataset into insight is nontrivial --- in Chapter 2 we saw just how many different perspectives we could take on a single dataset, and how these perspectives can range from interesting and insightful, to not. It is straightforward to select a particular perspective when we address a particular question. As we have seen in the operationalization process, the goals of a project are not usually phrased as detailed, particular questions --- they start off very high-level, and are almost always ill-defined.

Furthermore, not all the information necessary to answer a high level question will exist in the data set --- indeed, the information may not exist at all. We saw in the UTA scenario, for example, that we did not have a direct way to measure how many cars the transit system took off the road. Analysts often try to get access to elusive insights that they suspect can be gleaned from data without access to a direct measure. Visualization can be extremely effective for helping someone bring their own knowledge to bear in the context of the data that *is* available. The challenge is in finding an effective way to describe and present it.

Creating visualizations is a collaborative process

Creating a visualization to understand data is a fundamentally collaborative process. In a project team, the data scientists are experts in data analytics, but must be comfortable in a number of other technical areas, from tooling to database design to machine learning. Experts in fields like game design, cancer biology, and finance who have a deep understanding of their specific domains rely on the computational tools and insights that data scientists bring to their problems. Executives, managers, and principal investigators must make high-level decisions based on the results. In this chapter we discuss strategies for how to expose underlying data analysis needs, assumptions, and interpretations across a team in order to design effective visualization tools.

Even when designing a visualization for yourself, there is still a need to uncover core data analysis needs. While the operationalization pyramid discussed in Chapter 2 provides some guidance to break down high level questions, this chapter provides additional strategies for prototyping and refining visualization solutions.

The Goal of Data Counseling

The goal of data counseling is to fill in the gaps in the operationalization tree. It uses feedback from stakeholders, and their domain expertise, to: * Reduce general question to specific tasks * Select good operationalizations of objects and descriptors * Explore data to verify these operationalizations * Construct and verify designs for high-level visualizations

Each of these requires asking lots and lots of questions. The hardest part of data counseling is figuring out *what* questions to ask.

We seek answers by interviewing people that are involved with data and the project, called *stakeholders*. At early stages of the process, we interview them empty-handed. As we proceed through the operationalization, we bring raw data, or initial data explorations, even with prototypes.

At the start of a project, we want to understand the high-level questions; we want to choose a reasonable operationalization of that question into a task; and we want to learn about domain-specific aspects of the problem. The first two points are explicitly about determining the root question and task in the operationalization tree. The third, fairly ambiguous, question is about getting a sense of the domain and building an intuition about the stakeholders, what they care about, and how they think. This intuition can be important of designing effective tools, as well as for helping to facilitate your own data exploration through the data counseling process.

The data counseling process

In the data counseling process, we begin by identifying relevant stakeholders. As we find them, we carry out interviews with these stakeholders, and then bring their data back to them in visualizations and prototypes.

This is an iterative process, interlinked with our climb through the tree: the results of data exploration can be shown to stakeholders to get their feedback on whether the exploration is going along the right track. In obtaining data and exploring it, new questions, and even new stakeholders, often come to light.

In this section, we outline these steps. The remainder of this book is dedicated to a discussion of visualization solutions and what types of data and tasks they are good for, knowledge that is important for shaping your approach in the data counseling process.

Identify stakeholders

When it comes to tackling a problem, who is invested in the result? Who will use the results, and who will they present those results to? If the visualization produces a valuable insight, who will act on those insights, and what will they do with them? There is likely a whole ecosystem of people that have been, or would like to be, involved with the data and the problem. There are people who produce and store the data, people who want to consume it, and those who make decisions based on it.

All the people that are invested in the problem in some way, shape, or form are the **stakeholders**. Identifying these stakeholders is crucial for data counseling. The different stakeholders can give different perspectives on the data, the problem, and potentially unanticipated paths to insight.

We have encountered a number of recurring types of stakeholders. This list is by no means complete, but we provide it here as guidance for identifying some of the important people in the ecosystem of a problem. A single person could embody one, some, or all of these roles.

Analyst: The person who actually sits with the data, searching and exploring to make discoveries. These stakeholders are the people most likely to use the visualization tool designed for the problem.

Data producer: A person that collects, creates, and curates the data. Data producers can often shed light on the nuances and quirks of how the data was attained, and can be invaluable during the data cleaning process.

Gatekeeper: The gatekeeper is a person with the power to approve, or block, the project, including authorizing people to spend time

talking about the data and problem. Their perspective can be good for understanding high-level goals and potential impact of the project. In some settings, a gatekeeper may require a proposal to carry out an analysis.

Decision maker: This is a person that wants to use the insights gleaned from the your data to execute on a decision. Decision makers are often one-step removed from the analysts, and could be thought of as the analysts' customers. Thus, they often have a different interpretation of goals and questions.

Connector: A connector is great at identifying good people to talk with. They are able to understand aspects of both the data analysis side of things, as well as the problem, specifics the data, and the analysis that needs to happen. Connectors are worth their weight in gold.

Guiding Interviews

Across all interview types, and across all stakeholders, we begin with a series of open-ended questions about the problem, data, and context that will help us understand where the interviewee sits: what their perspective on the problem is, what the scope of the problem as they see it is, and how they expect to interact with it.

Some of the types of interview questions we begin with include:

- What are the goals of the project? How do those goals fit with the organizational needs?
- Who would act on the insights and results of this analysis? What decisions are they looking to make?
- What are the questions that we can answer with this data?
- What do you already know from the data, and what else do you expect to find?
- What do you want to do with the data that isn't currently possible? If you could do this, what would you want to do next?

These general questions are meant to get a conversation going, and to help in establishing the start of the operationalization. These questions will lead to lower-level questions that are necessary to clarify your understanding, to confront assumptions on the part of

the stakeholder, and to shape the description of the problem into something that a visualization to solve.

Low-level questions arise when trying to make general tasks more concrete, such as defining what a specific descriptor is meant to convey, or how to relate some object to actual aspects of the data. We listen carefully for fuzzy descriptors and objects in a conversation, and ask many questions in order to clarify what those concepts mean more concretely. Sometimes we will go so far as to ask the person we are talking with to show us what they mean within their workflow, or to indicate more specifically what aspect of the data they are talking about by pulling it up in a spreadsheet.

During the operationalization process, we are often trying to chase down particular meanings of unclear words. A useful question that we often use is "What would it look like in the data if …" For example, if we're trying to operationalize "high quality" bus service, we might ask "What would the data show as high quality bus service? What would low-quality service look like?" This can help interviewees to nail down specifics. (Fairly often, the interviewee won't be sure what 'high quality' bus service would look like. That's fine; the process of articulating out a list of possibilities—as we did in the UTA example-- can be highly informative in itself.)

As we noted in the previous chapters, particular actions in tasks will help identify potential visualizations. During interviews, we listen for those words, sometimes going so far as to guide our interviewees: "Would you say you'd like to compare one item, or group many items together?". Words such as *"select, compare, group"* can translate directly into tasks that can be supported with a visualization. Other words like *"shape, structure, size"* can help in deciding what kinds of visual encodings to use, or what the characteristics of the data the stakeholder is most interested in seeing. The concepts we discuss further in the second part of the book will provide a rich palette of visualization concepts that can help in recognizing visualization keywords during interviews.

Conduct interviews

We operationalize questions through information gleaned from interviews; we also use interviews to get feedback on our intermediate results and our final designs. The role of the interviewer is to ask questions that will guide the stakeholders towards elucidating the

information necessary for designing an operationalization and visualization.

We find interviewing very challenging, but also one of the best parts of what we do —-- how many jobs let you, no, *require you* to talk to experts about the deepest, most interesting parts of their problems? Interviewing is not easy, though, and requires practice and experience. The necessary skills include how to keep a conversation moving along and on track, how to elicit meaningful responses, how to revise questions based on responses, and how to interpret both subtle cues and detailed responses. While we provide several strategies to help with these tasks, gaining competency of these skills is truly a matter of practice.

Regardless of what questions are asked, there are a number of different ways to conduct interviews. We have carried out interviews as casual, hallway conversations around a whiteboard, as well as conducted formal interviews with a fixed set of predetermined questions. There is a continuum of interview types, characterized by the amount of control exerted by the interviewer on the direction the discussion goes. We can roughly divide this continuum into four categories:

1. **Informal:** Very little structure or control of the discussion. These can feel more like a conversation than an interview. The interviewer engages with the stakeholders in casual conversation without seeking specific types of information, although with slight control on the direction of the conversation to keep it on the general topic of data needs. Informal interviews are important for getting to know a group of people and understanding their work culture. It is also important for gaining rapport and finding out about new topics of interest that may not have been known about.
2. **Unstructured:** Loosely structured with very little control. Unlike an informal interview, an unstructured interview is very clearly an interview. The interviewer will have a plan in mind for the topics to cover, but the discussion is mostly led by the stakeholders' responses. This type of interview is typically based on an interview guide, which is a list of topics or questions that guide the conversation. The interview is started by asking a question from the guide, and usually followed up with probing questions to elicit more details about interesting comments.

When the conversation slows or stalls, another question from the guide steers the conversation. Unstructured interviews often lead to new, unanticipated questions or topics, allowing for a deeper understanding of the problem space. These interviews, however, require skill to conduct in order to understand when to probe deeper, when to change topics, and how to keep the stakeholder engaged and informative. They also require significant time to conduct, for both the interviewer and the stakeholder, and often require follow-up interviews.

3. **Semistructured:** Mostly structured with moderate control. In a semistructured interview the interviewer covers all the questions in the interview guide, but they are free to follow-up on responses from the stakeholder to elicit more details. This type of interview is particularly useful when there is only one opportunity to interview someone. Like unstructured interviews, semistructured require both skill and time, but they are easier to conduct due to the scripted nature of the interview. These interviews are particularly useful after the interviewer has a sense of the needs of the group, gained through a series of unstructured or informal interviews.
4. **Structured:** Completely structured with total control. For a structured interview the interviewer asks (usually multiple) stakeholders the exact same set of questions, and only those questions, in a scripted order. This is like doing a survey in person except that the stakeholder is likely to give more detailed responses. Structured interviews make it easy to compare responses between interviews, but do not allow follow-up on interesting responses with new questions. Because of their rigid constraints, structured interviews are usually a poor match for the data counseling process.

Figure Figure 3-1 illustrates the trade-offs between these different interview types.

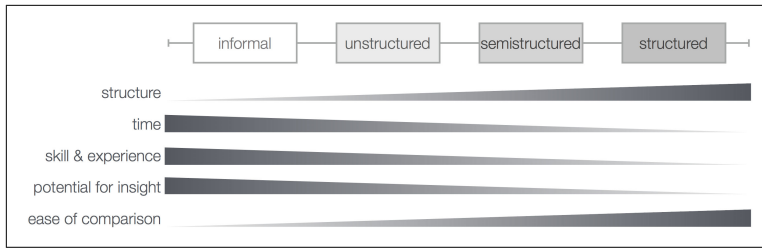


Figure 3-1. The range of interview types along with their relative trade-offs.

After, or as part of, an unstructured interview with a stakeholder a **contextual interview** can be a valuable step. These interviews take place in the stakeholder's work environment and consist of demonstrations of the tools and data inspection methods that a stakeholder currently uses. These types of interviews have the potential of bringing to light many aspects and challenges of a problem that the stakeholder may not have communicated in a strictly verbal interview. Conducting a contextual interview can be done as unstructured or semistructured.

These can therefore be invaluable to identify how the operationalization is played out in practice: what happens with the data today? How do users handle and understand the data they see?

We often start a contextual interview by asking the stakeholder to either walk us through a specific analysis task they have already performed, or, to have them conduct their work for that day with us there. Either way, we ask the stakeholder to talk about each step they are taking and we interrupt with questions whenever something is not clear. Here are some interview guide questions that are meant to bring to light what works, what doesn't work, and what doesn't yet exist:

- What is the work process you currently use? What tools are involved?
- What challenges do you have in analyzing the data?
- Are there limitations with the system? If so, how do you work around them?
- Do you understand what the system is doing to the data and any algorithms that are being applied, or is this a black box?

Many times we will get to a point in an interview that feels like a dead-end: the stakeholder has answered our questions, and yet we still don't have a good understanding of the problem. In these situations, there are a couple of strategies we employ. The first is rephrase the stakeholder's response in our own words. This strategy allows the stakeholder to correct any misinterpretations, and it also can prompt the stakeholder to explain their ideas in your language versus their own.

The second strategy we employ is to try to ask the same or similar questions in different ways. Often a specific phrase or choice of words will click with the stakeholder and cause them to respond in a way that makes more sense. And when we run out of ideas for gaining clarification on a topic we just steer the conversation along a different path. We have found that one of the most important things we can do in an interview is to keep the stakeholder talking. The more they talk, the more likely we are to hear a response that clarifies a topic or opens up a new avenue of inquiry. When we reach an absolute dead-end in an interview, that is when we resort to our interview guide and ask the next question in the list.

There are several tools that can be invaluable to have during an interview. Commonly used interviewing tools include pen and paper, voice recorder, camera, and video recorder. We advocate for voice recording of interviews in large part because it is difficult to take detailed notes while also trying to think of follow-up questions based on what is being said. We try to transcribe an interview shortly after it is conducted to ensure the context is fresh in our minds. We rarely transcribe an interview word for word, but instead transcribe the most important or complex details. Transcribing is useful for analyzing the interview results, as well as for making it easy to refer back to the conversation later in the design process.

Data Exploration

While talking with stakeholders can be very informative, there's no substitute for reaching deep into the data. We start playing with the data as early as possible for a couple of reasons: we want to understand how it is structured, and what is available; we want to test whether an operationalization makes sense; and we want to collect answers to the fine-grained tasks, like *"what is the distribution of workers' salaries within this neighborhood?"*.

Thus, as we build an operationalization tree we use a variety of visualization tools to explore the data. Tools like Excel, Tableau, and R allow us to rapidly generate visualizations --- mostly static --- that can teach us about the distributions of data, its major dimensions, and the values within. We also use tools like this to check whether a dataset makes sense --- for example, to confirm that a hierarchy really is layered appropriately, or to ensure that there are only a small number of categories for a specific dimension of the data. The transit time charts and choropleths in the previous chapter are examples of this.

Other times we need to build a bespoke visualization tool from scratch because the data we need to explore is not amenable to an off-the-shelf tool. In the transit dataset we saw an example of this when it was necessary to explore *many* transit time curves in order to characterize what effectiveness of transit means. Here the problem was not only that we needed to see time curves in relation to their geospatial position, but also that the data itself was large, requiring both software optimizations as well as interaction mechanisms to support efficient exploration. We create these types of bespoke tools using visualization-specific languages like D3 and Processing, and code in such a way as to get ideas up and going as fast possible, as opposed to carefully considering software architecture for long-term use and reusability.

Making the decision to create a custom data exploration tool requires weighing the development time against the significance of the analysis --- if you can get 80% of the way to a good decision using Excel, then it may not be worth spending three months to develop a custom solution. We find that these bespoke exploration visualization tools often come into play when we are trying to refine complex datasets with complex interlocking dimensions. .

One thing to be aware of is whether or not the high level goals of a project can be answered with off-the-shelf tools; sometimes the answer is yes and you are a hero for quickly solving the problem! We always try to explore data initially with existing tools to avoid reinventing the wheel. These tools, though, can also be great ways to better understand what is lacking from the current ecosystem of visualizations and what is unique and different about your problem. These insights are invaluable when designing a custom visualization tool.

Rapid Prototyping

Throughout the process, we build prototypes of the final tool we intend to deploy to our stakeholders --- we refer to these tools as prototypes. The intention behind prototypes is to explore the visualization *design space*, as opposed to the *data space*. We usually create a series of prototypes, using each one to gather feedback from our stakeholders to refine our ideas on how to most effectively support the higher level tasks in the operationalization tree; we build prototypes on the way to the leaves of the tree; these help think about how the decisions we make at the leaves will affect the root. We build more prototypes on the way back to the root, as we try to work out hte final design.

Rapid prototyping is a process of trying out many visualization ideas as quickly as possible and getting feedback from our stakeholders on their efficacy. In our data counseling sessions we do **a lot** of rapid prototyping. Early on low-fidelity sketches on the whiteboard give us the ability to better understand what types of visualizations to use and how our stakeholders might interact with them; later on we use increasingly higher fidelity techniques to explore the space of possible visualization designs. Failing fast is well-known concept in design; it's incredibly valuable to explore multiple ideas rapidly.

We create prototypes using a broad range of techniques and tools, from paper to programming. The fidelity of these prototypes, as well as the time and energy required to create them, lives on a spectrum:

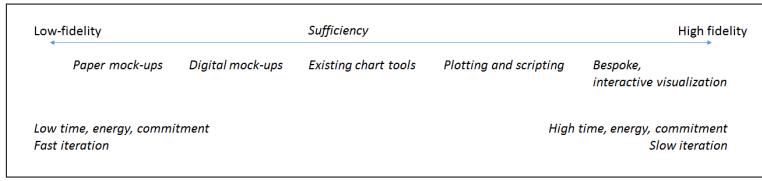


Figure 3-2. Prototypes range from low-fidelity sketches to high-fidelity, working models.

On one end we have *low-fi* prototypes. These include quickly sketched mock-ups on paper or a whiteboard with impressions of what the data might look like, or fast digital mock-ups that may include a some controls for explaining interaction ideas such as slide jumps in PowerPoint or Keynote --- the image below is an example of a quick interface mock-up made during the design process of the project we discuss in more detail in ???. Lo-fi digital mock-ups could

also incorporate charts generated in a tool like Excel or Tableau with fake or sampled data to explore possible visualization representation ideas. These low-fi prototypes are great for communicating the gist of an idea in an interview, or for recording high-level ideas when planning out how you might want to explore the data yourself. Low-fi prototypes are, by nature, fast and easy to produce.

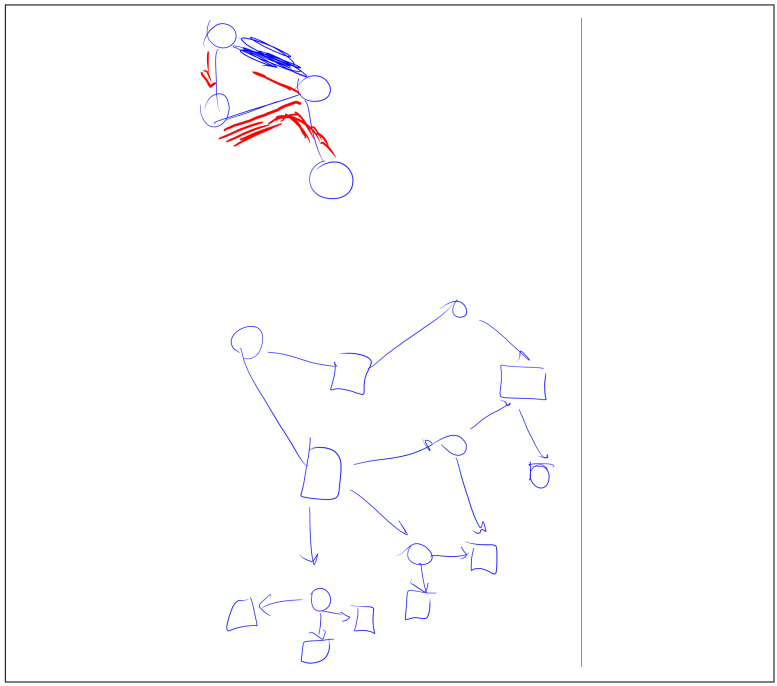


Figure 3-3. Recent low-fi prototype exploring the idea of a weighted, directed graph layout. Hand-drawn during interview session, and based on sample data by manually looking at the spreadsheet and drawing out the relationships.

Low-fi sketches, like Figure 3-3, are often parts of our interview process. Creating these images can help understand what the implications of the data are. If a diagram is confusing to explain and design on a white board, it may require too much detail to fit on a screen.

We have often found that communicating ideas with low-fi prototypes can rapidly help establish whether we are on the same page as our stakeholders: by drawing pictures of possible interfaces, we often learn more about the problem and its constraints. Figure 3-3 shows one instance: our stakeholder was discussing relational data,

and we wanted to start talking about what it would feel like to build a network interface. The multiple colored lines allowed them to start thinking about how to view multiple modalities on the data; the directed edges were actually built from a sample of the data.

Drawing this prototype helped the client realize that there was more structure to their data then they had been communicating: every node in the graph represented by a box actually occurred at a specific time, and it was important in the analysis to expose the temporal dimension of the data.

Later in the process, low-fi "slideware" can help ensure our designs will make sense to users. The slideware in Figure 3-4 shows one step in the feedback cycle. This image was manually assembled in a variety of different tools; the prototype sketch is meant to help the user understand how the final interaction will work.

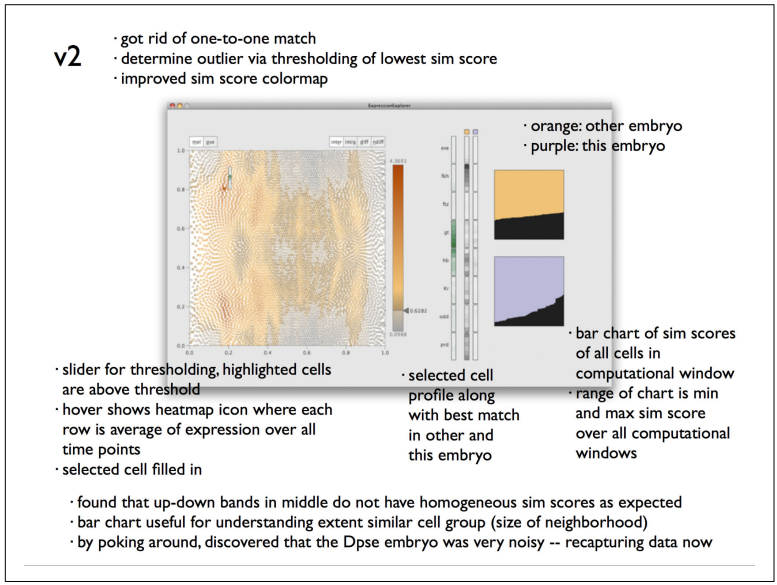


Figure 3-4. Slideware image of a design stage shows iteration from previous version; the images were created in a variety of different tools.

On the other end of the prototype spectrum are high fidelity, custom visualizations which must be created from scratch. These high-fi prototypes are meant to largely contain the core functionality of an envisioned visualization tool, including all necessary visualizations of the data and interaction mechanisms. They will often, however,

gloss over many back-end issues such as smooth integration with existing workflows or fully fleshed out features for I/O. Just as for bespoke visualizations created for our own data exploration, we most often use languages like D3 or Processing for high-fi prototypes. These prototypes are meant to be thrown away footnote[In our experience, however, high-fi prototypes are often the tool that is deployed and adopted by some users, particularly frantic to get into their data. The point, though, is not to worry about the code other than to get your ideas working.].

Iteration

We have found that it is very difficult to get a good (or even adequate) understanding of the problem the first time around, particularly as we are defining the root of the operationalization tree. Getting this right often requires multiple interviews with stakeholders, interspersed with some data exploration.

The Sad Reality of Data Cleaning

Looking at data almost always inspires a round of data cleaning. It's often the case that the data was not collected for the tasks you've defined in the operationalization tree; even if it was, data often has errors and misunderstandings within it. In our own data counseling sessions we've found ourselves saying each of these:

- What are these strange spikes scattered throughout the data? They don't seem to make sense for what you are measuring.
- Column E is always zero. Why?
- You have a column called "Sum of Sum of Sum." What does it mean?
- Wow, this data takes forever to load. How big did you say it was again?
- Half the temperatures you have are around 37; the other half is around 99. Is this in Fahrenheit or Celsius?
- Our prototype looked great when we thought there would be 5 categories; but it turns out there are instead 500.

In more extreme cases, we may experience cases where data is missing, or blank, or mis-entered. For example, in our origin/destination data, it turns out that a small percentage of records have

obviously-invalid travel times. We need to find a policy—do we regenerate the data, or drop the single datapoint, or the entire record?

Depending on the task, any of these might be most appropriate. Other ORA books explore data cleaning in far more detail.

A key component of a good operationalization is having explicit policies for handling quirks found in the data. For example, if an analyst wants to eliminate the count of automated bots before looking at the number of users, the analyst needs a working definition of what a bot looks like in the dataset, and what is the procedure for removing them. Making cleaning steps explicit, like all other parts of walking through the operationalization tree, makes the process both reproducible, and allows the process to be tweaked more clearly.

One challenge is knowing when, and how, to just start digging into the data. Often times the stakeholders we work with will already have some way of analyzing or visualizing the data that they find to be insufficient for their question. This is usually good place to start. For example, are they looking at many static visualizations? Add interactivity to support exploration. Are they using only one kind of visualization? Take a different perspective on the data and visualize it in a different way. Use these early data explorations for a deeper conversation about what works and what doesn't, and why. It also provides a chance to better understand the analysts' perspectives on the data.

Thus, the data counseling process is often a very iterative one. Talk with some stakeholders, try some ideas with the data, share those ideas back with the stakeholders. And, repeat.

Conclusion

In this chapter we looked at the core components of the data counseling process: identifying stakeholders, conducting interviews, data exploration, and rapid prototyping. Data counseling allows you to gain different perspectives on the problem and the data in order to build, refine, and support an operationalization tree.

Knowing how to articulate concise tasks over the data we can now begin to look at visualizations that support these tasks. In the next chapter we start from the leaves of the tree, looking at the core visu-

alizations for basic data types. We'll look at the different combinations of dimensions we saw in the previous chapter, and explore how to choose an appropriate chart type. In the following chapter we'll look at higher level compound chart types. Combining simple visualizations into a compound or coordinated view system allows us to address higher level tasks in the operationalization tree.

CHAPTER 4

Components of a Visualization

In the previous chapter, we outlined the process of refining a question into tasks; we described an operationalization tree, in which each task is broken into four components: "actions," "descriptors," "objects," and "partitions." We used these terms to help describe the process of exploring the task, and working through the tree down to fine-grained subtasks.

Now that we know something about the process of working our way down the tree is, it is valuable to take a step back. After the process in Chapter 2, we have a well-operationalized task, which we promised would lead to a visualizartion. However, that chapter avoided the question of *how* to translate data into visualizations. One wonderful virtue of a well-operationalized task is that it translates well into a visualization: when we say "do players spend more hours playing level 2 then level 3," or "do people who buy more coffee also buy more eggs," these can be used to describe and generate visualizations.

In this chapter, we take the first step to translating these descriptions to visualizations by discussing measures and dimensions. At the leaves of the tree, the descriptors and objects are close to the data, and we translate them into the terms more familiar from data analysis: "measures" and "dimensions."

Data Abstraction

We begin with the "data abstraction." We borrow this term from computer science; for our purposes, the data abstraction is the ways we understand what *meaningful operations* we can carry out on the data.

For example, "time of day" is an important value in the Utah transit example in the previous section: buses have different frequencies at different times of day. It is very meaningful, then, to talk about "rush hour" or "evening." On the other hand, even though time is technically a number, it is less meaningful to talk about "times divisible by four." As a result, an analysis is likely to carry out aggregate operations on the morning commute, or to aggregate time by hour.

The fully-operationalized object, which we saw in the last chapter, is a meaningful entity of the abstraction. It might—and often does—refer to a single row of the data; however, it can also refer to relationships and partitions within the data. The object might be aggregated into all rush-hour bus rides from a certain location, for example.

The data abstraction helps think about the semantics of the data, and its relationships: knowing that geography in the Utah Transit time cube, for example, comes in pairs: every "start" comes with an "end"--means that most visualizations will want to take into account the two-ended nature of this data.

The abstraction also allows us to think about how to partition the data: there are a variety of visualization techniques that can reflect a partition, from multiple series on a single chart, to hierarchy and trellis views, to small multiples. We will discuss these different ways of partitioning in ???

Direct and Indirect Measures

Having linked the object to the data, we turn to the descriptor. The descriptor is a quantifiable notion; an operationalized descriptor helps us find a measure. The measure is perhaps the most-discussed aspect within the data science community.

The first component of addressing a question is choosing what measure will allow us to answer the question. Sometimes, we're lucky: the dataset already has a relevant measure, which we can read

off the set directly, or with a little computation. For example, we may wish to know precisely how many dollars a shop has made; summing up those transactions gets us what we need to know.

More often, we need to choose a proxy value: our measure will stand in for something else. We want to figure out which product is "worth the shelf space"; we can't compute "worthwhile", but we can compute how much profit it makes and how much space it takes, and compute that indirect measure. The term *metric* is sometimes used to describe a measure that stands for a desired value [1].

In many fields, metrics are used to measure the success of a project; comparing the metric over time is a proxy for the overall project. One common example is the Dow Jones Industrial Average, which is the average value of a selected basket of stocks; it is used as a commonplace proxy for how the stock market as a whole is doing. It's worth noting that the Dow is actually a very poor metric: it measures the current value of stocks (as opposed to the change in market caps) for a very small number of stocks; as such, small fluctuations in those prices can send false signals about the rest of the market.

Indeed, the IEEE has a standard [footnote: 1061, where it is called a "quality factor"] to define the attributes of a good metric. It should be correlated with the underlying value: when the underlying value grows or shrinks, the metric should change in the same direction. The reverse should be true, too: an increase in the metric should indicate that the underlying factor has increased. The change should be proportionate: a bigger change in the underlying factor change should cause a change significant change in the metric.

Critically, it should be difficult to increase the metric in other ways. When a business focuses entirely on a single measure, it provides impressive incentive to find ways to maximize the metric — whether it's student test scores, or number of bugs fixed.

In other words, a system will tend to optimize on the metrics that it is measured on. Not long ago, advertisers often charged per impression — with the result that websites would optimize on "number of ads shown", which in turn incentivized them to put up slideshows, or break articles into multiple pages. (As the metric has begun to

1 While the distinction between a "metric" and a "measure" makes for entertaining online battles, in this volume, we treat the two as effectively synonymous.

change to "cost per click-through," websites instead begin to pop-up ads that are difficult to remove, and easy to accidentally click on.) We know of one organization that tweaks its metrics every couple of years, just to make sure that they don't sink too deeply into optimizing on a single suite.

We often refer to "gaming the system" when people attempt to manipulate metrics intentionally, but it can happen by accident just as easily.

Despite these challenges, metrics are necessary and inevitable. We cannot measure "effectivenes" of a bus route directly, but we can find proxies that we think will stand in for effectiveness: variance is one choice; speed of the fastest route is another; the ratio of route speed to driving time (or distance) is a third. Of course, any of these introduces possible bias.

One virtue to a visualization approach is the awareness that we can handle multiple metrics at once. Rather than trying to reduce our world to a single number, we can look at several different measures: it's reasonable to say "the fastest route is getting faster, and that's good, but the variance is really brutal."

In both the chapters on single and multiple visualizations, we'll talk about ways to visualize multiple metrics at once .

Dimensions

Where metrics are what is being measured—the output—the dimensions of the data are the ways in which the data varies: the independent variables. The object from the operationalization process is the thing in the world that remains independent of our measurement. The "partitions" from that process, then, are the attributes of the thing that we can divide or group them on—what we will be describing here as a dimension.

We've discussed several different variables in the UTA scenario.

- Time of day (and whether that time is a commute time or not)
- Location (census distrcts; as origin or destination)
- Routes (from an origin to a destination)
- Income of a commuter, or group of commuters

Just looking at these, it's clear that there are some different types of data here. A visualization that works well for time of day is unlikely to be useful for location.

In ???, we will to through a variety of charts, based on the types of dimensions and data. Conventionally , we'll discuss four different types of data, plus three specialized forms.

- **Nominal Data** has many different possible values which are not comparable to each other. Nominal data is often an identity or a name for a data point. They are good for join keys — but lousy on an axis. In business intelligence, fields like customer names, phone numbers, and addresses, are good examples. In the UTA example, the name of each census tract is nominal: there's no particular meaning to the 12 digit numbers; a lower or higher number isn't a sequential value.
- **Categorical data** has few discrete values which items fall into—as categories. Categories are used to cluster data into groups. Categorization comes in no particular ordering—North does not logically come "before" or after "West"; there are few enough categories that it makes sense to group the data into these. It's not uncommon to be forced to impose categories onto more fluid data—for example, in the UTA data, we might categorize time into "morning commute," "evening commute," plus other times—but this does impose a hard line on smooth data. (Is someone really commuting at 6:59 am, but IS commuting at 7:00?)
- **Ordinal data** consists of discrete values that should be ordered. Rankings are a good example of ordinal data: if a runner comes in first in one race and ninth in another, they didn't come in a total of tenth, and it's unclear how to compare them to the runner who came in fifth twice. Census data doesn't provide direct income of participants; instead, it provides number of people in an income range. As a result, the income range is an ordinal value.
- **Interval and Ratio Data** consist of values that are ordered and equally spaced. Interval values can't be logically added together directly—such as dates or oven temperatures. You can do math on ratio data; it can be added or subtracted meaningfully. In the

UTA dataset, the actual length of a trip, or the number of commuters, is ratio data.

In addition,

- **Temporal Data** is a form of interval data that has a time component. Temporal data is actually very complex, as it can be seen from multiple perspectives. Times may refer to specific moments ("November 20, 2010, 8:01 am"), or may take advantage of cycles ("every day at 8:00 am", or "weekdays at 8:00 am") . Temporal data may mean a range ("the month of November, 2010"). The time that a bus leaves is one form of temporal data; however, the duration of a trip is just a ratio data, measured in minutes or seconds.
- **Geographical Data** refers to places; it is inherently two-dimensional (or three-dimensional, in some cases); it may come in the form of positions, or outlines, or names of places.
- **Relational Data** is data that connects two other points: this might be from a *hierarchy* or a *network*. For example, the fact that some number of commuters go from one place to another is relational data; so is the fact that one person reports to another. When data points area categorized, they often wind up in a hierarchy; the relation is between the point and its category.

Fortunately, we can transform data between these different forms. Sporting events assign points for different ordinal ranks in order to get a ratio scale, so that they can compare athletes; the results of the ratio scale is then transformed back into a ranking. We can group ratio ages into groups to get interval age ranges. We will often assign an order to categorical data in order to place it in a meaningful order on screen.

A Suite of Actions

We've spent some time on the questions of data. In this section, we take on the question of actions. Above, we actually kind of let this go[2] : we claimed that "compare" was enough to know, and called it a day. Compare is a very broad term: it might mean many different things. In this case, we're referring to comparing two parallel cases.

We'll use the action to identify candidate visualizations and encodings. Of course a single visualization can address multiple actions: a

humble bar chart can allow a user to find a specific value, to identify the largest or smallest value, to roughly guess an average, and, yes, to compare two or more bars to each other. However, some visualizations are more effective for given actions than others, and knowing the dimensions available to us, plus the actions we wish to carry out, will help a great deal.

In operationalization, we're hoping the pyramid will provide us with a selection of specific actions. The visualization research community has spent a great deal of time[3] working out different tasks that can be done in visualizations

Some of the operations that will come up:

- Reading individual numbers off the data
- Distribution of a column: minimum, maximum, outliers, central tendency, sort order
- Trend of a metric over time (or another dimension)

There are more complex operations:

- Comparing a value across a category ("dollars from store A vs store B")
- Comparison of a metric to another metric ("Height and weight of subjects"); compare distributions across a category ("salary distribution of men vs women")
- Contrast a metric against many others (e.g. "Seattle vs other cities")
- Cluster values (eg. divide players by their play style into "damage-dealers, defenders, healers")

Now, lots of these look like statistical tasks. Indeed, if you're only doing any one or two tasks — "I want to know if men or women spend an average of more money at our store" — then a visualization isn't necessary. However, it's often the case that tasks are linked: I don't only want to know the mean value of a set, but also to know the median. And the variance.

Visualization gets very good at showing all of these at once. It also becomes reasonable to start with a simple task and dive in deeper:

- … upgrade from average to distribtion

- … split across dimensions (“at which store? split by product? split by aisle?”)
- … change comparisons (“older women vs younger women”. “older women vs older men”)

As a result, we try to match up keywords with visualization tasks. Terms like “compare one object to another” are a cue that we are looking at multiple series; while “how is this item different” perhaps suggests that we want to pull out a single item to compare to averages. “Are any items different” is a cue to look for visualizations that help show outliers.

Choosing an Appropriate Visualization.

In ??? and ???, we will discuss different types of visualizations, and the ways that they can be used to answer different sorts of questions. As part of the process of operationalization, we will need to figure out the types of visualizations that will be available to us, and start producing them. We will discuss the visualizations by the dimensions that they use, and by the types of tasks they support.

We now have a goal for the interviews we will discuss in Chapter 3: we want to understand the task, the data available (or collectable), and to establish that the operationalization is a valid one. We will discover cleaning tasks on the way (“our diagnostics app is producing half of the website traffic”) and decide how to score and weight metrics appropriately (“are we measuring people who see our ad, or people who look at our site?”)

Because operationalization involves making many decisions about the data and its manipulations, it’s important to understand how the results will be used, so that metrics will be appropriate to those results.

The process of interviews then goes through a series of steps: discussing possible refinements of the tasks; seeking out actions, objects, and descriptors; and trying to work through the operationalization tree.

Working through the tree is an iterative process: it can involve going back and forth between the data and the people who will use using it. Sometimes, presenting a possible visualization ends up teaching us mostly that we have the wrong task: “that visualization would

answer the question I asked, but not the thing I want to know." Other times, phrasing a task teaches us that we don't have appropriate data.