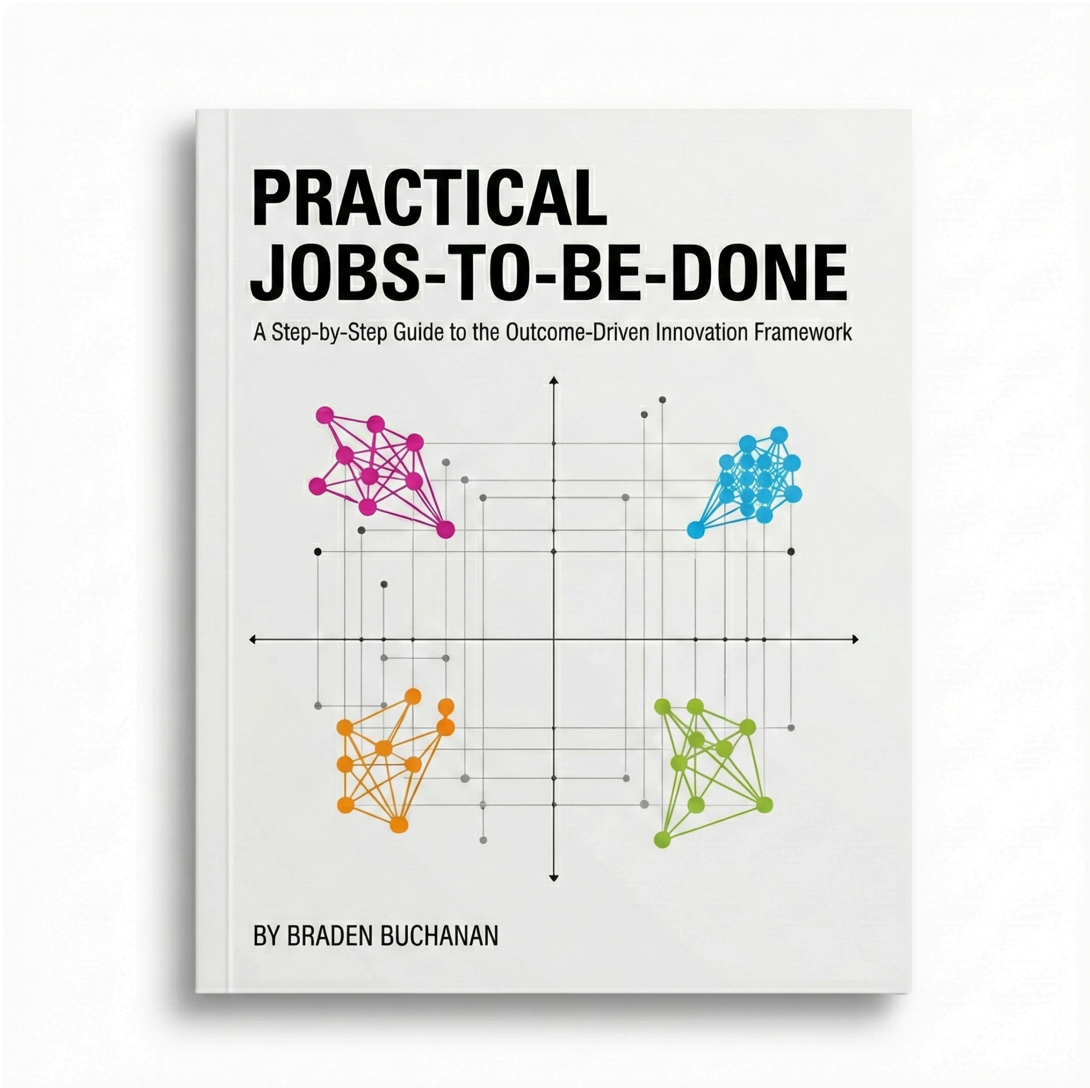
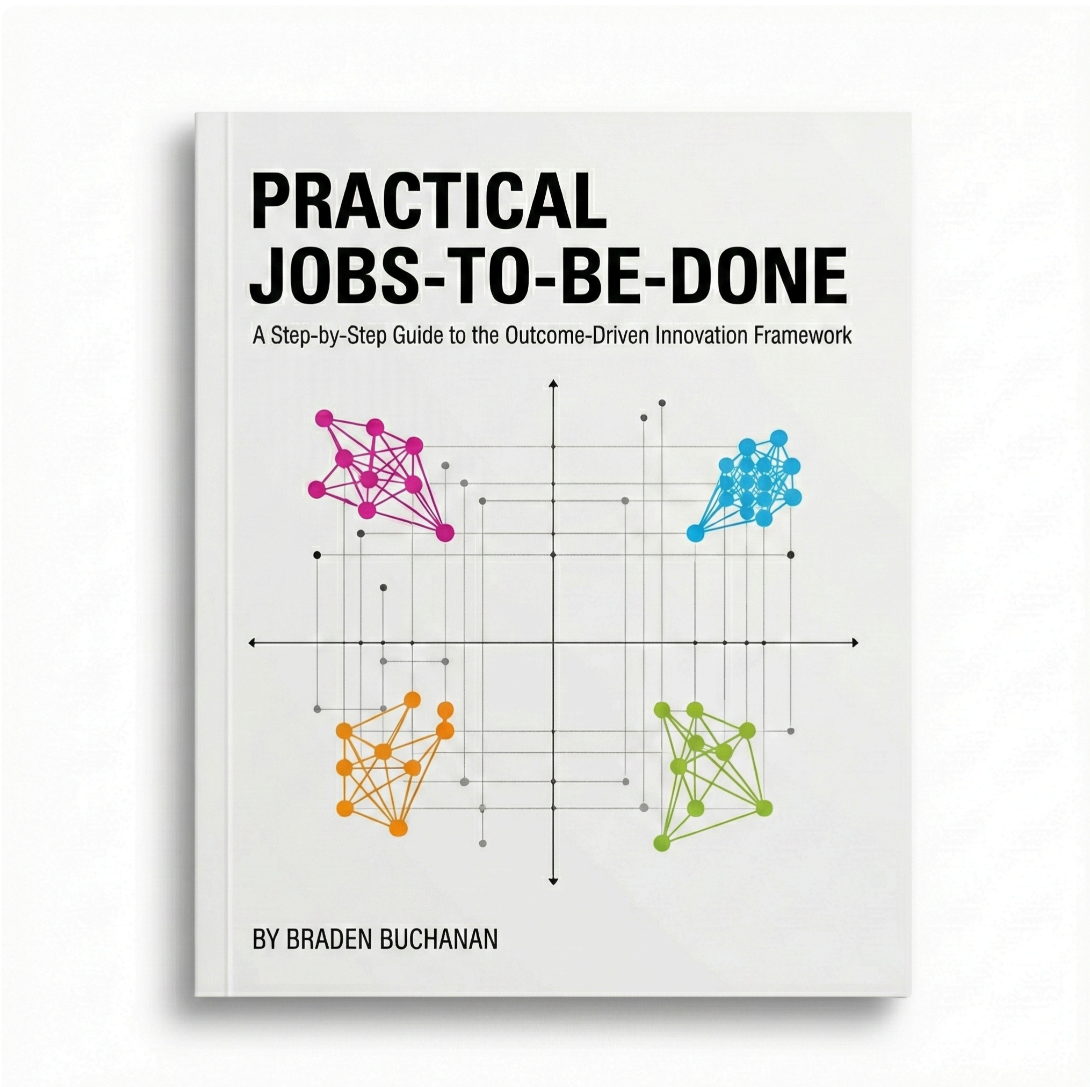

PRACTICAL
JOBS-TO-BE-DONE
A Step-by-Step Guide to the Outcome-Driven Innovation Framework
BY BRADEN BUCHANAN

Table of Contents

INTRODUCTION

Welcome the Practical Jobs-to-be-Done Book

This is the website for the first edition of Jobs-to-be-Done & Outcome-Driven Innovation



Bookcover

Many teams struggle incorporating Jobs-to-be-Done into their product and market research efforts. This books goal is to highlight practical ways to get more out of the theory, the potential unlocks of knowing how to use it properly, and the many pitfalls and things to look out for.

This book exists for one reason: to give you a complete, step-by-step playbook for running real JTBD research using the Outcome-Driven Innovation (ODI) process pioneered by Tony Ulwick, Strategyn, and knowing the limitations of the approach.

You'll learn how to:

- Uncover the exact functional, emotional, and consumption-related outcomes customers use to measure success
- Understand the JTBD and ODI quantification process
- The critiques with the methodology and alternatives teams may consider.

No methodology is perfect, and I'll openly discuss the real drawbacks of the ODI approach. I will highlight where it shines, where it can be rigid or overly quantitative, common pitfalls teams run into, and situations where other JTBD flavours (Switch interviewing, Forces of Progress, Jobs-as-Progress, etc.) or even completely different frameworks might serve you better.

Whether you're a product manager, designer, researcher, founder, marketer, or executive, this is the practical field guide I wish had existed when I started doing this work.

Acknowledgements

Jobs-to-be-Done is a community effort. Huge thanks to every contributor, reviewer, and practitioner who shared examples, feedback, and late-night debates that made this resource what it is.

Preface

Introduction to Tony Ulwick's systematic approach to JTBD implementation

Many researchers and product managers might have found themselves in this familiar situation. A new manager or researcher recommends something like, "We should try the Jobs-to-be-Done approach to help truly understand our customers' needs better. Have you read Clayton Christensen's book on Competing Against Luck? We just need to understand what job our customers are trying to get done."

Curious to implement this approach, you begin researching the literature and practical steps, only to encounter confusing and often conflicting pieces of advice. You go to Reddit, Google, and or even review internal documentation on JTBD research that has been ran in the past. But it still does not make sense.

Despite the growing popularity of Jobs-to-be-Done theory and Outcome Driven Innovation, there's a real gap between understanding these concepts in theory and applying them effectively in practice.

That gap is why I wrote this book. Fresh out of college, I landed my first job on Roche's Diabetes Care Division's Strategic Insights and Open Innovation global team who utilized this methodology. Eager to learn and grow, I read every resource I could find, reading books on Jobs-to-be-Done (JTBD), Outcome Driven Innovation (ODI), and various research methodologies.

Over five years, I spent my time conducting lean validation studies on early medical device and digital prototypes. The biggest thing I learned was how to really understand and use the Outcome Driven Innovation (ODI) approach to Jobs-to-be-Done (JTBD). Working directly with ODI researchers and consultants at Roche, plus getting involved with other JTBD practitioners, taught me not just the theory but when to actually use this methodology and, just as importantly, when not to.

As I learned more, I found myself wanting to share what I'd learned. I started having more conversations with colleagues, jumping into discussions on r/UXResearch and r/ProductManagement, and providing my perspective when I could. That's when I noticed something missing. There's tons of theory out there about JTBD and ODI, but not much that actually shows you how to do it. Most practitioners I talked to wanted the same thing I did when I was starting out: real examples, step-by-step walkthroughs, and tools you can actually use right away, not just more theory.

Overview

This book addresses that gap directly. It's written for researchers, product managers, and strategists working in organizations ranging from startups to established enterprises, across B2B and B2C contexts. Whether you're conducting research for SaaS platforms, medical devices, consumer products, or professional services, the principles and methods in this book can be adapted to your specific context.

Rather than just theory, this is a hands-on guide that provides you with concrete tools and techniques. For example, you'll find interview scripts that help you uncover the emotional and functional dimensions of customer jobs, step-by-step instructions for building outcome statements that actually drive product decisions, and R code templates for analyzing satisfaction and importance data to identify the highest-value opportunities. When a customer tells you they need to "manage their project timeline," you'll learn how to dig deeper to uncover the underlying job of "feeling confident that deliverables will meet stakeholder expectations without constantly monitoring every detail."

While Jim Kalbach's *The Jobs To Be Done Playbook* stands out as the most practical guide on JTBD to date, this online e-book complements his work with a critical lens on the Outcome Driven Innovation (ODI) approach.[26] Unlike other resources that treat ODI as gospel, I examine its limitations alongside its strengths. The methodology's rigid survey requirements, expensive implementation costs, and prescriptive outcome statement formats don't always align with real-world research constraints or organizational needs. More importantly, ODI's quantitative focus can sometimes obscure the nuanced, contextual insights that make JTBD powerful in the first place.

Recognizing that not every team can, or should, implement the full methodology, I also provide flexible approaches throughout. For instance, while traditional JTBD unmet need quantification surveys are valuable, teams with limited resources can use MaxDiff analysis to prioritize opportunities, or leverage lightweight observational methods combined with targeted interviews to generate actionable insights. When ODI's outcome statement format feels too constraining, I show how to adapt the underlying principles to create statements that better fit your product context.

How to use this book effectively

This book is structured to serve both newcomers and experienced practitioners effectively.

If you're new to Jobs-to-be-Done and Outcome Driven Innovation, I recommend reading this book chronologically from start to finish. Each chapter builds upon the knowledge gained from previous chapters, creating a solid foundation. The early chapters establish core concepts and frameworks, while later chapters dive into practical implementation details.

For experienced JTBD researchers and practitioners who are already familiar with the theoretical foundations, feel free to navigate directly to the sections most relevant to your current challenges. Each chapter is designed to stand alone while still connecting to the broader methodology. You might find particular value in the advanced interview techniques, data analysis methods, or stakeholder communication strategies covered in later chapters.

The appendices contain ready-to-use resources, including interview guides, analysis templates, and code samples that have been refined through real-world application and feedback from practitioners across different industries.

References

[26]: Kalbach, Jim. *The Jobs To Be Done Playbook: Align Your Markets, Organization, and Strategy Around Customer Needs*. 1st ed., Two Waves Books, 2020. ISBN-13: 978-1933820682. Available on Amazon.

Section 1 Overview

Introduction to Tony Ulwick's & Strategyn's approach to implementing JTBD

Section 1: Foundations of Jobs-to-be-Done

Jobs-to-be-Done theory has evolved considerably since its early conceptual origins, but the gap between theory and practice remains a challenge for most product teams. Section I establishes the historical background you need to understand both where JTBD came from and how it can be applied.

These two chapters trace the evolution of JTBD from Theodore Levitt's early insights through Clayton Christensen's theoretical framework to Tony Ulwick's Outcome Driven Innovation methodology. You'll understand the different schools of JTBD thinking, why they emerged, and how practitioners like Ulwick and his firm turned academic concepts into their processes they promote.

Chapter 1 covers the history and core principles of JTBD theory, addressing common misconceptions and questions that arise when teams first encounter these ideas. Chapter 2 introduces the Outcome Driven Innovation framework that serves as this book's primary methodology.

By the end of Section I, you'll have a solid grounding in JTBD fundamentals and understand why ODI provides the structured approach needed to move from theory to practice. This foundation sets the stage for Section II, where we'll dive deep into the specific methods, mental models, and implementation guidelines that make JTBD work in real-world product development contexts.

Chapter 1: History of JTBD

A deep dive into the history, principles, and practical applications of JTBD theory

The Jobs to be Done (JTBD) framework is built on a simple idea, customers don't buy products; they "hire" them to make progress in their lives. While this concept was formalized in the late 20th century, its intellectual roots stretch back much further than most practitioners realize.

The Prehistory of an Idea

Long before JTBD had a name, in 1954, Peter Drucker published *The Practice of Management*, a landmark work that emphasized customer focus and the importance of understanding what value customers actually derive from products rather than what companies think they are selling. [1]

Then came the quote that would be used in all JTBD literature. In 1960, Harvard Business School professor Theodore Levitt published his seminal article "Marketing Myopia" in the Harvard Business Review, arguing that businesses need to shift their focus from producing and selling goods to understanding and meeting customer needs.[2, 3] Levitt used to tell his students, "People don't want a quarter-inch drill. They want a quarter-inch hole!" This captured the essence of JTBD before the term existed, establishing the foundation that the customer's desired outcome is the true object of their desire.

But elegant aphorisms don't ship products. It would take an engineer's professional disappointment to turn this insight into a rigorous, repeatable methodology.

The Genesis of a Theory: From Engineering Failure to a Formal Process

The practical history of Jobs to be Done begins with Tony Ulwick. In 1983, Ulwick was an engineer on the IBM PCjr team, building what many expected to become the definitive home computer. The Wall Street Journal had other ideas, declaring it a flop the day after launch. The failure cost IBM over a billion dollars.

That high-profile failure helped shape Ulwick's career. By 1990, he had developed a key insight: instead of studying products, study the underlying process customers are trying to complete. He applied Six Sigma and process control thinking to innovation itself, creating a methodology that could identify exactly where customer needs were going unmet.

Ulwick left IBM in 1991 to found The Total Quality Group, now known as Strategyn. His approach, originally called CD-MAP, evolved into Outcome-Driven Innovation.

The methodology's first real test came in 1992 with Cordis Corporation, a medical device company struggling with its angioplasty balloon line. Ulwick's team focused on understanding what cardiologists were actually trying to accomplish when treating blocked arteries, then identified which outcomes mattered most but remained poorly served. Within two years, Cordis went from minimal market share to over 20% and eventually brought the first coronary stent to market. [4]

Around 1999, the company became Strategyn and the process became Outcome-Driven Innovation. In 2000, Ulwick had the distinct pleasure of introducing ODI and his research and segmentation techniques to Harvard Business School Professor Clayton Christensen in a series of meetings in Cambridge.

One of the highlights of Ulwick's career came in 2002, when Harvard Business Review published an article he wrote called "Turn Customer Input into Innovation," which described ODI and its successful application at Cordis. [5]

The Theorist Enters the Picture

Christensen immediately recognized the JTBD as the missing "demand-side" explanation for why customers choose to adopt new solutions. In his 2003 book *The Innovator's Solution*, co-authored with Michael Raynor, Christensen referenced Ulwick's work and used phrases like "circumstances-based categorization." [6] But the full "Jobs to be Done" terminology wouldn't become mainstream until later.

The iconic milkshake story first appeared publicly in an April 2007 MIT Sloan Management Review article titled "Finding the Right Job for Your Product," co-authored by Christensen, Scott D. Anthony, Gerald Berstell, and Denise Nitterhouse. When a fast-food restaurant resolved to improve sales of its milkshake, its marketers first defined the market segment by product and then segmented it further by profiling the customer most likely to buy a milkshake. The consequent improvements to the product had no impact on sales. [7]

A new researcher spent a day in a restaurant documenting when each milkshake was bought. He was surprised to find that 40% of all milkshakes were purchased in the early morning. These early-morning customers almost always were alone, they did not buy anything else and they consumed the milkshakes in their cars. When the researcher returned to interview these morning customers and essentially asked what job they were hiring the milkshake to do, most of them bought their shakes for similar reasons: They faced a long, boring commute and needed something to keep that extra hand busy and to make the commute more interesting. They wanted to consume something that would stave off hunger until noon. [7]

This illustrated that the customer's circumstance, not their demographic, is the key to understanding their motivation.

The Great Divide: Two Schools of JTBD Thought

Christensen's popularization of the idea also led to a philosophical split, resulting in two distinct schools of thought that persist today. Understanding this divide is important context to how different practitioners and philosophies go about applying JTBD.

Jobs-as-Activities (The Ulwick School): This school defines a "job" as a functional task or activity a person is trying to accomplish. The focus is on the *process* of getting the job done. Innovation comes from identifying the metrics customers use to measure success (the "desired outcomes") and helping them execute the job better: faster, more predictably, and with greater efficiency. This approach is highly analytical and quantitative, treating innovation like an engineering discipline. ODI is its primary methodology.

"The Customer-Centered Innovation Map" by Lance A. Bettencourt and Anthony W. Ulwick appeared in the May 2008 issue of Harvard Business Review.[8] The article introduced job mapping, a methodology that helps companies analyze the biggest drawbacks of the products and services customers currently use and discover opportunities for innovation. It involves breaking down the task the customer wants to accomplish into eight universal steps. This framework serves as a bridge that many practitioners use regardless of which school they lean toward philosophically.

Jobs-as-Progress (The Christensen/Moesta School): This school defines a "job" as the progress a person is trying to make in a particular circumstance. It's not about the task itself, but about resolving a struggle and transitioning to a better state. The focus here is on the *why* behind a customer's decision to change. This perspective explicitly includes the powerful emotional and social dimensions of a decision. Its research methods are qualitative and narrative-based.

Bob Moesta is the president and CEO of the Re-Wired Group, and one of the core figures in the Jobs-To-Be-Done methodology. Moesta and his team developed frameworks including "The Forces of Progress," which examines the push of the current situation, the pull of the new solution, the anxiety of the new, and the habit force that causes people to resist change.[9] If the push and pull are not greater than the anxiety and habit, people will never switch. The Re-Wired Group formalized switch interviews, a method of interviewing people who have recently made a purchase to understand how they actually overcame the forces that might have prevented them from switching.

Bob Moesta pioneered the Jobs-to-be-Done framework in the mid-90s, alongside Harvard Business School Professor Clayton Christensen. JTBD is a research process that helps uncover a customer's motivation for buying a product, the "job" the product is "hired" to complete.

The Purist: Alan Klement and the Rejection of "Tasks"

While Moesta and Christensen focused on the "Switch," author Alan Klement took the Jobs-as-Progress philosophy a step further, becoming the vocal critic of the activity-based approach. In his book *When Coffee and Kale Compete* (2016), Klement argued that viewing a Job as a process or a workflow is a fundamental error, essentially forcing "Task Analysis" into a JTBD wrapper. [19]

Klement defines a Job strictly as the desire for self-betterment: "*A Job to be Done is the process a consumer goes through whenever she aims to change her existing life-situation into a preferred one, but cannot because there are constraints that stop her.*" [18]

For Klement, a Job has no functional steps. The "Job" is not to "drill a hole" (an activity); the Job is the emotional struggle to feel proud of one's home. He argues that activities change constantly as technology evolves, but the human desire for progress is the only constant. This perspective created considerable friction in the JTBD community, drawing a hard line between those who view JTBD as an engineering discipline (Ulwick) and those who view it as a psychological investigation (Klement/Moesta). [20]

JTBD Reaches the Mainstream

In 2016, Christensen, Taddy Hall, Karen Dillon, and David S. Duncan released *Competing Against Luck: The Story of Innovation and Customer Choice*.[10] After years of research, the authors came to one important conclusion: the long-held maxim that the crux of innovation is knowing more and more about the customer is wrong. Customers don't simply buy products or services; they hire them to do a job. This book finally brought "Jobs to be Done" into mainstream management literature, though it draws almost entirely from the Jobs-as-Progress school and focuses less on Ulwick's ODI methodology.

The "Jobs to Be Done" approach can be seen in some of the world's most popular companies including Amazon, Intuit, Uber, Airbnb, Coinbase, Slack, Twitter (now X), and Chobani yogurt.

The Spread of JTBD in Tech and Startups

Corporate adoption followed a predictable pattern. Early adopters from the late 1990s through 2005 were mostly B2B and industrial companies like Cordis, Bosch, and Microsoft. Between 2007 and 2012, B2C and tech companies began adopting the framework.

The explosion came after 2015, particularly in startup and Lean circles. Des Traynor, co-founder at Intercom, helped shine a spotlight on Jobs to be Done when it wasn't yet popular in the tech startup scene. The interviews and content Intercom produced became some of their most popular material. Intercom's book *Intercom on Jobs-to-be-Done* combined ReWired's foundational concepts alongside fresh thoughts from Intercom employees and affiliates, largely reformatting their popular blog posts into an accessible ebook format. [11]

Another influence in the rise of JTBD in the startup scene is thanks to Ash Maurya, founder of LeanStack and creator of the Lean Business Model Canvas. Ash popularized the operationalization of the "Jobs-as-Progress" philosophy specifically for achieving Need-Solution Fit. [12, 13]

While Moesta and Christensen provided the theoretical foundation for the "Forces of Progress" (Push, Pull, Inertia, Friction), Maurya focused on "simplifying complexity" to make these concepts usable for agile teams. He translated the abstract interplay of forces into a visual "Hill-Climbing" metaphor. In this model, the customer is visualized at the bottom of a hill, needing enough "Push" and "Pull" to overcome the gravity of "Inertia" and "Friction" to reach the top. [12]

Maurya initially captured these insights in the Customer Forces Canvas. However, realizing that static canvases often "collapsed the timeline" and lost the nuance of the customer's journey, he evolved the methodology to focus on narrative structure. Drawing inspiration from Pixar’s storytelling rules, Maurya introduced the Customer Forces Story, a "Mad-lib" style framework. This 3-act structure (Inciting Incident → Progressive Complication → Resolution) provides strict guardrails that help teams effectively translate interview notes into a coherent "Global Story" of the customer’s journey, greatly increasing the success rate of capturing actionable insights. [12]

This Book's Focus

While both schools of thought offer value, this book will focus primarily on the Outcome-Driven Innovation (ODI) approach.

I chose this focus not because it is the simplest path, but because it is the most *complex*. Having spent years applying this methodology within a global strategic insights team, I have seen firsthand that ODI is not a "quick fix." In fact, compared to the refreshing, intuitive immediacy of the Moesta, Klements, and Maurya approaches, which are excellent for uncovering the narrative *why*, ODI can feel dauntingly mechanical.

ODI’s structured, approach is often a double-edged sword. Some are drawn to it as a "silver bullet," hoping its complexity guarantees success. Others are repelled by that same complexity, fearing it will bog down their teams in bureaucracy and data.

The truth lies somewhere in the middle, and that is what we will explore. We are going to peel back the layers of ODI, acknowledging both the clarity it provides and the friction it can create. My goal is to take the complexity that I have seen teams struggle with and break it down into a navigable, step-by-step process.

We will not ignore the other schools—the emotional "push and pull" of the Jobs-as-Progress school provides context that data alone cannot. However, by understanding the *most complex* approach to JTBD, you will be able to choose whichever implementation you see fit for your own efforts.

In the next chapter, we'll begin our deep dive into the five core steps of the ODI process, looking at exactly how to make this engine work for you.

> *There is **no** single right approach to JTBD. It all depends on the level of resolution your team needs. Think of the Jobs-as-Progress (Moesta/Klement) approach as a **Telescope**. It allows you to see the big picture: the market forces, the emotional trajectory of the customer, and the "why" behind the switch. It is perfect for positioning, marketing strategy, and understanding demand. Think of the Jobs-as-Activities (Ulwick/ODI) approach as a **Microscope**. It allows you to zoom in on the specific workflow to see the cracks, the friction, and the inefficiencies invisible to the naked eye. It is perfect for product roadmapping, feature prioritization, and engineering specifications. You do not have to choose between a telescope and a microscope. A good researcher uses both or other research methods.*

Key Takeaways

- **The intellectual foundation of JTBD predates its formal name by decades.** Motivation researchers in the 1940s and 1950s, management thinkers like Peter Drucker, and marketing pioneers like Theodore Levitt were all circling the same insight: customers buy outcomes, not products.
- **Tony Ulwick transformed this insight into a rigorous methodology.** His experience with the IBM PCjr failure in 1983, combined with exposure to TQM and Six Sigma process thinking, led him to develop what would become Outcome-Driven Innovation. The 1992 Cordis Corporation project provided the first major proof point, with market share jumping from 1% to over 20% by mid-1993.
- **Clayton Christensen popularized the "Jobs to be Done" terminology but did not invent the underlying concept.** He repeatedly credited Ulwick with

originating the functional job framework, while his own contribution was adding emotional and social dimensions. The famous milkshake study appeared in a 2007 MIT Sloan Management Review article.

- **Two distinct schools of thought—and occasionally conflict—exist today.** The **Jobs-as-Activities** school (Ulwick) treats innovation as an engineering discipline, focusing on functional steps and metrics. The **Jobs-as-Progress** school (Christensen/Moesta) focuses on the emotional "switch" and self-betterment. **Author Alan Klement further polarized this divide by arguing that "Jobs have no functional steps," strictly defining a Job as the desire to change one's life situation.**
- **There is no single "right" approach; it depends on the question you need to answer.** The Progress school helps answer *why* a customer buys (ideal for Marketing & Sales), while the Activities school answers *what* to build to satisfy them (ideal for Product & R&D).
- **JTBD adoption has followed a pattern from industrial B2B to mainstream SaaS.** Early adopters in the late 1990s and early 2000s were companies like Cordis, Bosch, and Microsoft. The framework reached mainstream startup culture after 2015, driven largely by content from companies like Intercom and practitioners in the Lean Startup movement.

Sources Used

[1] Drucker, P. F. (2006). The Practice of Management. HarperBusiness. Retrieved From https://www.amazon.com/Practice-Management-Peter-F-Drucker/dp/0060878975

[2] Levitt, T. (1960). *Marketing Myopia*. Harvard Business Review. Retrieved From https://seekscholar.com/sites/default/files/Myopia.pdf

[3] Levitt, T. (2004). Marketing Myopia. Harvard Business Review, July–August issue. Retrieved From https://hbr.org/2004/07/marketing-myopia

[4] Ulwick, T. (n.d.). *The Birth of Predictable Innovation*. Strategyn. Retrieved from https://strategyn.com/predictable-innovation-business-growth-strategy/

[5] Ulwick, A. W. (2002). Turn Customer Input into Innovation. Harvard Business Review, January issue. https://hbr.org/2002/01/turn-customer-input-into-innovation

[6] Christensen, C. M., & Raynor, M. E. (2013). The Innovator's Solution: Creating and Sustaining Successful Growth. Harvard Business Review Press. Retrieved From https://www.hbs.edu/faculty/Pages/item.aspx?num=15473

[7] Christensen, C. M., Anthony, S. D., Berstell, G., & Nitterhouse, D. (2007). Finding the Right Job For Your Product. MIT Sloan Management Review, April 1. Retrieved From https://sloanreview.mit.edu/article/finding-the-right-job-for-your-product/

[8] Bettencourt, L. A., & Ulwick, A. W. (2008). The Customer-Centered Innovation Map. Harvard Business Review, May issue. Retrieved From https://hbr.org/2008/05/the-customer-centered-innovation-map

[9] Bob Moesta. (n.d.). What Is the Jobs To Be Done Framework?. Playbooks. Retrieved From https://therewiredgroup.com/learn/complete-guide-jobs-to-be-done/

[10] Christensen, C. M., Dillon, K., Hall, T., & Duncan, D. S. (2016). *Competing Against Luck: The Story of Innovation and Customer Choice*. HarperBusiness. Retrieved From https://www.amazon.com/Competing-Against-Luck-Innovation-Customer/dp/0062435612/ref=sr_1_1?s=books&ie=UTF8&qid=1493055294&sr=1-1&keywords=competing+against+luck

[11] Moesta, B., & Traynor, D. (2016). *Bob Moesta on Jobs-to-be-Done*. The Re-Wired Group. (Published May 12, 2016). Retrieved From https://www.intercom.com/blog/podcasts/podcast-bob-moesta-on-jobs-to-be-done/#:~:text=%C3%97,%2Dsell%2Dmy%2Dinfo.

[12] Maurya, A. (n.d.). *The Backstory Behind Customer Forces Stories*. Retrieved From https://www.leanfoundry.com/articles/the-backstory-behind-customer-forces-stories

[17] Maurya, A. (2019). *What Is a Job-To-Be-Done (JTBD)*. Retrieved From https://medium.com/lean-stack/what-is-a-job-to-be-done-jtbd-21304ce7441b

[18] Jain, R. (2021). *Martech's Magicians: Microns, Micronbox and µniverse*. Published November 18–29, 2021. Retrieved from https://rajeshjain.com/martechs-magicians-microns-micronbox-and-%C2%B5niverse/

[19] Klement, A. (2018). *When Coffee and Kale Compete: Become Great at Making Products People Will Buy*. Paperback, March 21, 2018. Retrieved from https://www.amazon.com/When-Coffee-Kale-Compete-products/dp/1534873066

[20] Klement, A. (2018). *Know the Two — Very — Different Interpretations of Jobs to Be Done*. 19-minute read, January 15, 2018. Retrieved from https://jtbd.info/know-the-two-very-different-interpretations-of-jobs-to-be-done-5a18b748bd89

Chapter 2: The Outcome Driven Innovation Approach Introduction

Putting Jobs-to-be-Done theory to practice

Outcome Driven Innovation (ODI) is a structured framework that helps put Jobs-to-be-Done theory into practice.

Developed by Tony Ulwick and applied across numerous industries through his firm Strategyn, ODI focuses on identifying gaps between what customers want to accomplish and how well current solutions serve those needs. Strategyn claims high success rates for their approach, around 86% according to their internal studies [21].

> ***Author's Note:*** *It is worth noting the 86% success rate comes from Strategyn's internal research. I am generally skeptical of such "success rates" and how exactly this percentage was determined. It likely is just a catchy metric to help promote the methodology.*

Notably, ODI addresses only a few parts of innovating

1. identifying unmet customer needs
2. Quantifying the unmet needs found
3. Understanding the context around the needs (emotional and social jobs)
4. Segmentation based on needs

Success still depends on a company's ability to **design, develop, and market solutions that address those needs effectively**. What ODI *does* provide is a way to help teams turn the abstract theory of JTBD into tangible, practical steps.

Set realistic expectations. ODI is **not** a silver bullet. This framework provides a lens for understanding customer problems, their journeys, and what they're ultimately trying to achieve, but it won't answer every strategic question or guarantee product success. It is simply one tool in one's toolkit.

Anyone claiming that any single methodology solves all innovation challenges is likely either trying to sell you something or lacks sufficient real-world experience with the complexities of product development.

I am prioritizing and anchoring this book to the Strategyn approach to implementing JTBD because I have found it provides the most "linear" and step-by-step documentation. While other approaches—like Bob Moesta’s "Jobs-as-Progress"—are excellent for understanding the emotional "why" behind a purchase, ODI provides the engineering-grade "how" that product teams often require to build the solution.

The 5 step ODI approach

ODI follows a systematic five-phase approach that moves from understanding customer jobs to implementing winning strategies. Each phase builds on the previous one.

FIGURE 1: OUTCOME DRIVEN INNOVATION PROCESS

Phase 1: Define the Market Around the Job-to-be-Done

This step involves identifying the core functional job customers are trying to accomplish, along with related emotional and social jobs. In ODI, a market is defined specifically as **The Job Executor + The Job-to-be-Done**.

For example, when people "plan a vacation," the functional job might be "organize travel arrangements," but there are also emotional jobs like "create anticipation for a enjoyable experience" and social jobs like "demonstrate thoughtfulness to travel companions."

This phase also maps the complete ecosystem around the job, including **job executors** (who actually performs the job), **supporting cast members** (who help or are affected by the job), and **purchase decision makers** (who choose and buy solutions). In B2B contexts, these roles are often split across different people and departments, making this mapping essential for understanding the complete value proposition.

Rather than defining markets by product categories or customer demographics, ODI defines them by the fundamental progress customers seek to make. This perspective often shows that your competition isn't who you think it is, and your biggest opportunities is outside outside your usual market.

Phase 2: Uncover the customer's needs

Through qualitative interviews and secondary research, teams uncover the specific outcomes customers use to evaluate success when getting their job done.

For instance, when helping customers "plan a vacation," some outcomes might include: *minimize the time it takes to compare accommodation options*, *minimize the likelihood of booking hotels that don't match expectations*, *maximize confidence that the itinerary will be enjoyable for all travelers*, and *minimize the cost of changing plans if circumstances change*.

This phase typically reveals 100+ outcome statements that capture the full spectrum of customer needs (functional, emotional, and social). These outcomes are carefully crafted to be stable over time (they don't change with technology), solution-agnostic (they don't assume any particular way of solving the problem), and measurable in ways that customers can evaluate.

> ***Author's Note:*** *Yes, it says 100+ outcome or need statements. If that sounds overwhelming, you are not alone. I will discuss in later chapters why this approach to quantification does not make sense for 99% of teams and how to manage it without getting bogged down. Chapter 7 highlights these concerns in more detail.*

Phase 3: Quantify Unmet Needs

Qualitative interviews tell you *what* the needs are; Phase 3 tells you *which ones matter*. Through surveys, teams measure how **Important** each outcome is to customers and how **Satisfied** they are with current solutions.

ODI uses a specific opportunity scoring algorithm to process this data:

$$Opportunity = Importance + (Importance - Satisfaction)$$

This formula heavily weights features that are important but currently frustrating. This generates a quantitative score for every single need.

- **Underserved Needs:** High Importance, Low Satisfaction. These are your opportunities for innovation.
- **Overserved Needs:** Low Importance, High Satisfaction. These are opportunities for disruption (simpler, cheaper solutions).

The result is typically visualized as a scatter plot, often called the Opportunity Landscape, which instantly shows you where the market is broken.

Phase 4: Discover hidden segments of opportunity

This phase identifies groups of customers with similar sets of unmet needs, creating needs-based segments that often reveal opportunities competitors miss entirely. Unlike traditional demographic segmentation, these segments are based on what customers are trying to achieve rather than who they are.

For example, instead of segmenting business travelers by company size or industry, you might discover segments like **"efficiency optimizers"** (who prioritize minimizing travel time and maximizing productivity) and **"experience seekers"** (who value comfort and amenities even for business trips). These segments cut across traditional demographic boundaries but represent distinct opportunity spaces for innovation.

Phase 5: Formulate and deploy a winning strategy

Once you have needs-based segments and a quantified market map, the path forward becomes a calculation rather than a guess. Strategyn's framework categorizes the opportunities into specific strategic avenues based on the data:

- **Dominant Strategy:** If you can satisfy underserved needs better *and* cheaper, you target the whole market.
- **Differentiated Strategy:** If a specific segment is underserved, you build a premium solution for them (charging more for better performance).
- **Disruptive Strategy:** If the market is overserved (too much performance), you build a simpler, cheaper solution to capture the low end.

The goal of Phase 5 is to align your product roadmap, marketing messaging, and pricing to the specific opportunity landscape of your target segment.

> ***Author's Note:*** *I disagree with the framing that a methodology alone can "formulate" a strategy. A spreadsheet can give you coordinates, but it cannot drive the ship. In my experience, ODI provides inputs (sometimes confusing ones!) for strategy, but it must be paired with business context, technical feasibility, and competitive reality. We will explore how to blend these insights in* ***Chapter 11****.*

Reconciling the Tension

This distinction deserves a moment of clarification before we move on. The disagreement is not with the value of Phase 5, but with its naming convention. Calling this phase "formulate a strategy" implies that the methodology delivers a complete strategic plan. It does not.

What ODI actually delivers at this stage is an *input* to strategy: a recommendation about where market opportunities exist and which customer segments are most underserved. It provides a rationale for prioritization. It helps teams avoid building features nobody wants.

However, real strategy requires additional layers. You must consider what your organization can build (technical feasibility), what fits your business model (financial viability), and how competitors might respond (market dynamics). ODI is one way to tell you where the opportunity is. It does not tell you whether pursuing that opportunity makes sense for your specific company at this specific moment.

For now, understand that Phase 5 provides a clear direction. In later chapters, particularly Chapter 10 on the Growth Strategy Matrix and Chapter 11 on translating strategy into execution, we will examine how to combine this signal with the other inputs that true strategy requires.

Chapter 2 Key Takeaways

The ODI process follows five key steps:

1. **Define the market around the job-to-be-done:** Specifically, define the market as "The Executor + The Job."
2. **Uncover desired outcomes:** Gather the metrics customers use to measure success (typically 100+ outcomes).
3. **Quantify unmet outcomes:** Use the Opportunity Score formula to identify underserved needs.
4. **Discover hidden opportunity segments:** Group customers by their needs, not their demographics.
5. **Formulate and deploy strategy:** Build features that target the high-opportunity scores.

ODI differs from traditional approaches by:

- Looking beyond existing solutions.
- Segmenting by needs rather than demographics.
- Understanding problems before jumping to solutions.
- Using structured metrics instead of vague customer feedback.

A Final Note on Expectations

As we proceed through this book, keep the Phase 5 tension in mind. ODI is a powerful lens for understanding customer needs, but it is one input among many. The chapters ahead will teach you how to execute each phase rigorously. They will also teach you where the methodology has limitations and how to compensate for them. The goal is not to follow a process blindly, but to develop judgment about when and how to apply these tools effectively.

References

[21] Strategyn. (2019). Innovation Track Record Study. Retrieved from https://strategyn.com/wp-content/uploads/2019/10/Innovation-Track-Record-Study-Strategyn-1.pdf

Section 2 Overview

Learn how to frame your market by focusing on the jobs your customers need done.

Step 1: Define your market around the job-to-be-done

This section provides the foundationals for the entire Outcome-Driven Innovation (ODI) process. Before you can uncover unmet needs, you must first define the market correctly. These two chapters guide you through the critical first steps of establishing a stable and accurate target for your innovation efforts.

In **Chapter 3, "Defining your market around the Job-to-be-Done,"** you'll learn to shift your perspective away from product categories and technologies. Instead, you'll define your market based on the stable, underlying job customers are trying to accomplish. This chapter teaches you how to deconstruct the stakeholder ecosystem by clearly identifying the three critical roles: the **Job Executor**, the **Product Lifecycle Support Team**, and the **Purchase Decision Maker**. You'll learn why focusing on the Job Executor is paramount for core product innovation and how to strategically prioritize your efforts, especially in complex scenarios like platform businesses.

With the "who" clearly identified, **Chapter 4, "Identifying the Core Job,"** addresses the "what." Here, you'll master the art of articulating a precise, solution-agnostic, core functional job. This chapter introduces the concept of job hierarchies and teaches you how to select the right **level of abstraction** for your research—whether you're aiming for incremental improvements or exploring entirely new market opportunities. You'll also learn a practical, three-step process for narrowing down the primary job to focus on when your product serves multiple functions, ensuring your research is scoped for maximum impact and relevance to your key stakeholders.

By the end of this section, you will have a clearly defined market, a primary customer to target, and a well-articulated core job, setting the stage for the in-depth needs discovery that follows.

Chapter 3: Defining your market around the Job-to-be-Done

Describing your market around the JTBD changes the perspective in who you target

Step 1: Define your market around the job-to-be-done

Defining the market around the job-to-be-done is the first step in the ODI process, setting the foundation for all subsequent research efforts.

When using the ODI approach, we use a specific syntax to ensure we are looking at the market correctly:

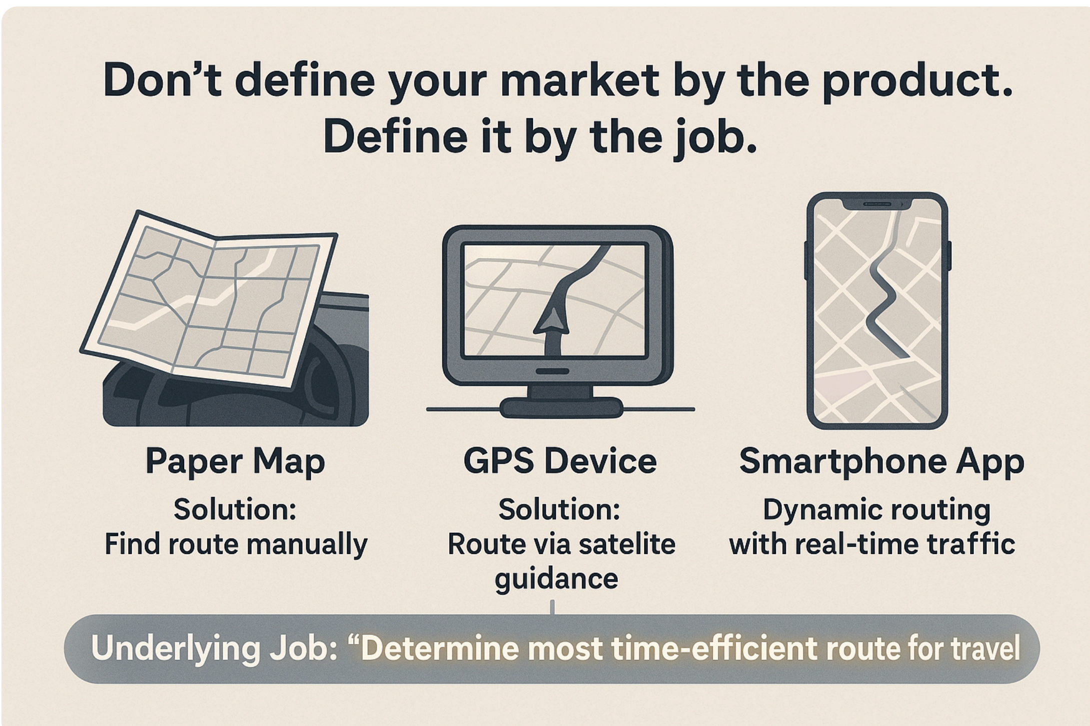
Market = [Job Executor] + [The Job-to-be-Done]

This implies that a "market" is not defined by a region (e.g., "The US Market") or a technology (e.g., "The SaaS Market"). Instead, a market is simply a group of people trying to get a specific job done.

Defining the market through a JTBD lens

Traditional market definitions often revolve around products or solutions, which limits our understanding of customer needs and opportunities.

Consider the evolution of navigation. If we had defined the market as "paper maps," we would have missed the fundamental job that people were trying to get done: "Determine the most time efficient route for travel." This job has remained constant even as solutions evolved from paper maps to GPS devices to smartphone navigation apps.

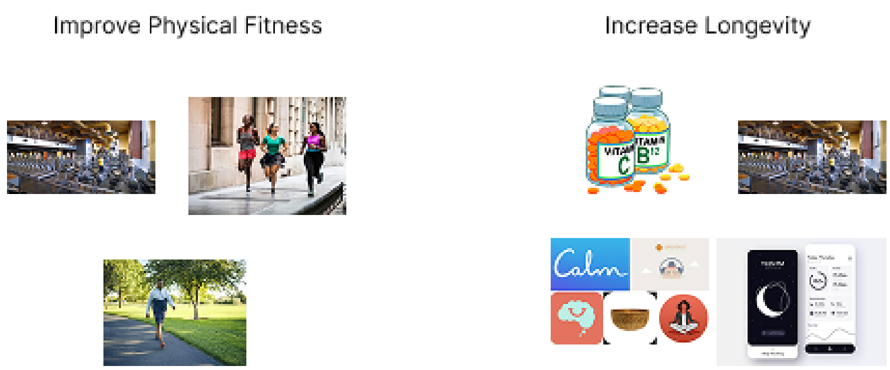


Solutions change over time, core job stays the same.

The key to defining markets through a JTBD lens is to shift focus from the solution to the underlying job.

Take the fitness industry as an example. Rather than defining separate markets for treadmills, fitness classes, and personal training, we need to consider the deeper core job that people are truly trying to accomplish. While it might seem like "improve physical fitness" is the job, for many people the true job might be "increase longevity," "feel confident in my appearance," or "maintain independence as I age." This shift in perspective which limits how we view the competitive landscape.

If someone's core job is "increase longevity," then Peloton, local gyms, and fitness apps aren't just competing with each other; they are competing with everything from meditation apps and dietary supplements to sleep tracking devices and preventive healthcare services. Similarly, if the core job is "feel confident in my appearance," the competition extends beyond traditional fitness solutions to include clothing brands, cosmetics, mental health apps, social media filters, and even drugs like GLP-1.



Solutions change over time, core job stays the same.

This broader understanding of the core job reveals that companies often define their competition too narrowly. A fitness equipment manufacturer focusing solely on competing with other equipment makers might miss that their real competition includes walking groups, gardening clubs, or even social dancing classes. All of which might serve the customer's true job just as effectively.

The transportation industry offers an illustration of this through Uber's approach. Travis Kalanick and the founding team didn't set out to build a "better taxi company." They focused on the job of "getting from point A to point B reliably and conveniently."[22, 23] This perspective helped them see that their competition included not just taxis, but car ownership, public transportation, and even decisions to stay home. Kalanick often spoke about "transportation as reliable as running water,"[23] demonstrating their understanding of the fundamental job customers were trying to accomplish.

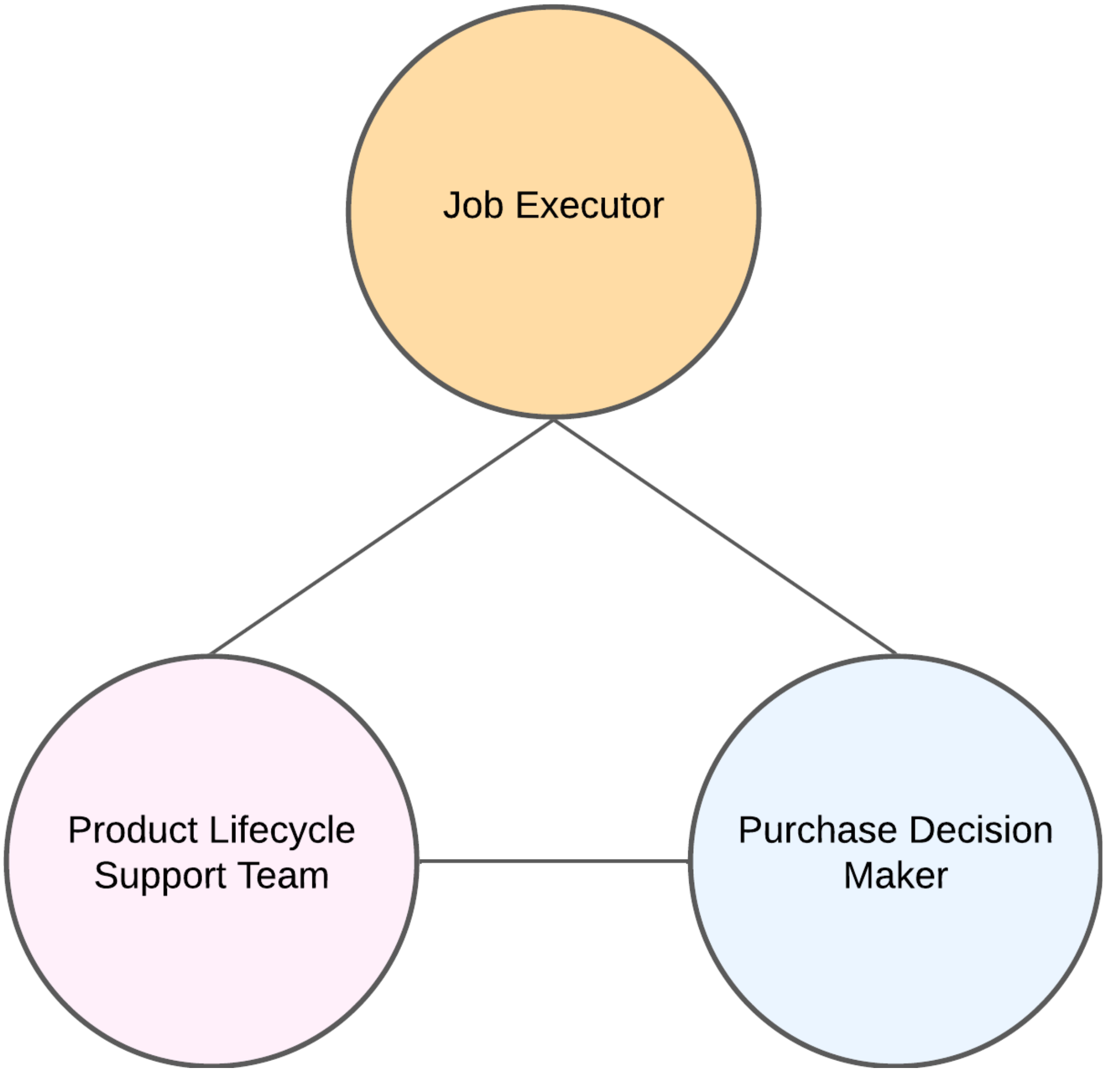
Airbnb similarly demonstrates this job-focused thinking. Brian Chesky has consistently articulated that Airbnb's job isn't about "short-term rentals" but about "belonging anywhere." [24] This job-focused definition led them to compete not just with hotels, but with the entire travel experience, eventually expanding into experiences and long-term stays.

When companies define their markets around jobs rather than products or technologies, they create a stable foundation regardless of how the technology changes.

The Job Executor, Product Support Team, and Purchase Decision Maker

One of the key steps in defining your market is identifying who actually executes the core functional job. While this might seem straightforward, many teams stumble by confusing the job executor with buyers or support personnel. It is vital to understand the distinction between three key roles:

- **The Job Executor:** The person using the product to get the job done.
- **The Product Lifecycle Support Team:** The people who install, maintain, or clean the product.
- **The Purchase Decision Maker:** The person who pays for the product.



The stakeholder ecosystem: The Job Executor, the Product Lifecycle Support Team, and the Purchase Decision Maker have distinct but interconnected roles.

The **Job Executor** is the person or role that performs the core functional job, the **fundamental reason why your market exists**. This person directly uses your product or service to accomplish their goal.

Consider the healthcare technology market: A hospital administrator might make purchasing decisions, and IT staff might handle installation and maintenance, but if you're developing surgical instruments, your **primary job executor** is the surgeon. Their needs, constraints, and desired outcomes should drive your core product efforts.

Product Lifecycle Support Teams play a different role. These individuals handle the **consumption chain jobs**, which are all the tasks associated with the product's lifecycle, encompassing everything from installation and maintenance to storage and disposal. They impact the overall customer experience but don't execute the core job. In a manufacturing setting, while machine operators are the job executors for production equipment, maintenance technicians form the support team, handling tasks like equipment calibration and repairs.

The **Purchase Decision Maker (or buyer)** holds the purse strings but may have limited interaction with the product itself. In B2B contexts, this role often belongs to procurement teams or senior management. While their needs must be addressed in your go-to-market strategy and business model, their requirements shouldn't drive core product innovation.

> ***Author's Note:*** *Be careful here. In B2B software, it is tempting to build features exclusively for the Purchase Decision Maker (e.g., fancy admin dashboards or compliance reports) because that is what "closes the deal." However, if you neglect the Job Executor, you end up creating "Shelfware"—software that is bought but never used. Long-term retention only comes from satisfying the Job Executor.*

While identifying these roles might seem straightforward, modern business models often present more complex scenarios. In platform businesses, there may be multiple distinct job executors, each with their own core jobs to be done. For instance, a marketplace platform needs to simultaneously serve and understand two different types of job executors: suppliers and consumers, each with distinct needs and success metrics.

Product/Service	Job Executor(s)	Product Support Team	Purchase Decision Maker
SaaS Tools (bottom-up SaaS approach)	Designers, PMs, Engineers	IT (for enterprise deployment)	Initial: Individual users/teams Later: IT/Procurement
Airbnb	Hosts (monetizing property) Travelers (finding accommodations)	Airbnb customer service/platform teams	Hosts and travelers
Hospital Medical Device	Healthcare Professionals	Nursing/administrative staff	Hospital administrators/payers
Consumer Packaged Goods (CPG)	Consumer	Retailer customer service, manufacturer customer service	Consumer

The key lies in properly **identifying** and **prioritizing** these roles during your research and development process.

Who to prioritize?

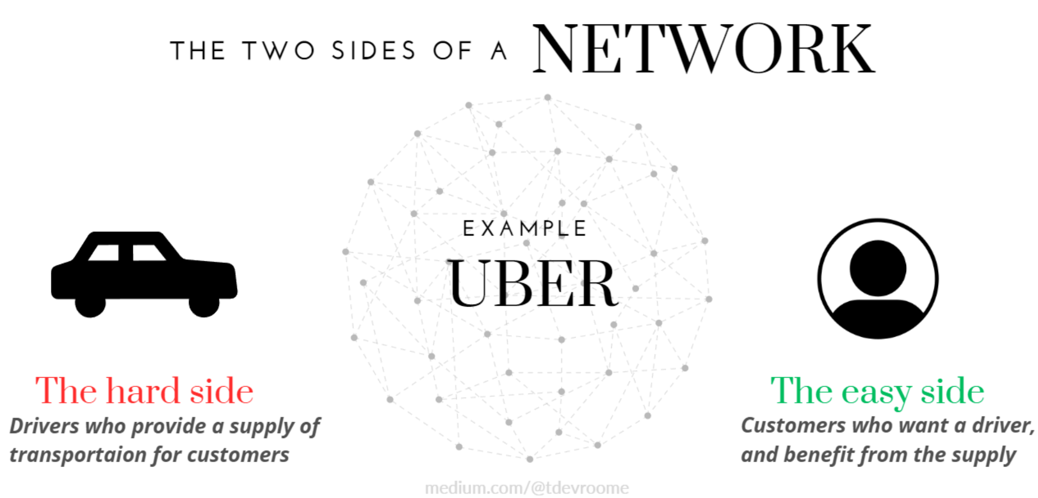
While identifying job executors and purchase decision makers is necessary, determining which group to target primarily for your go-to-market strategy presents a challenge for many teams.

Platform businesses face what's known as the "chicken and egg problem": the need to build both sides of a marketplace simultaneously to create value. However, successful platforms typically solve this by identifying which side is more crucial for building initial marketplace liquidity.

Important Note for JTBD: When dealing with a platform like Uber or Airbnb, you are not researching "one market." You are researching **two distinct markets** that happen to interact.

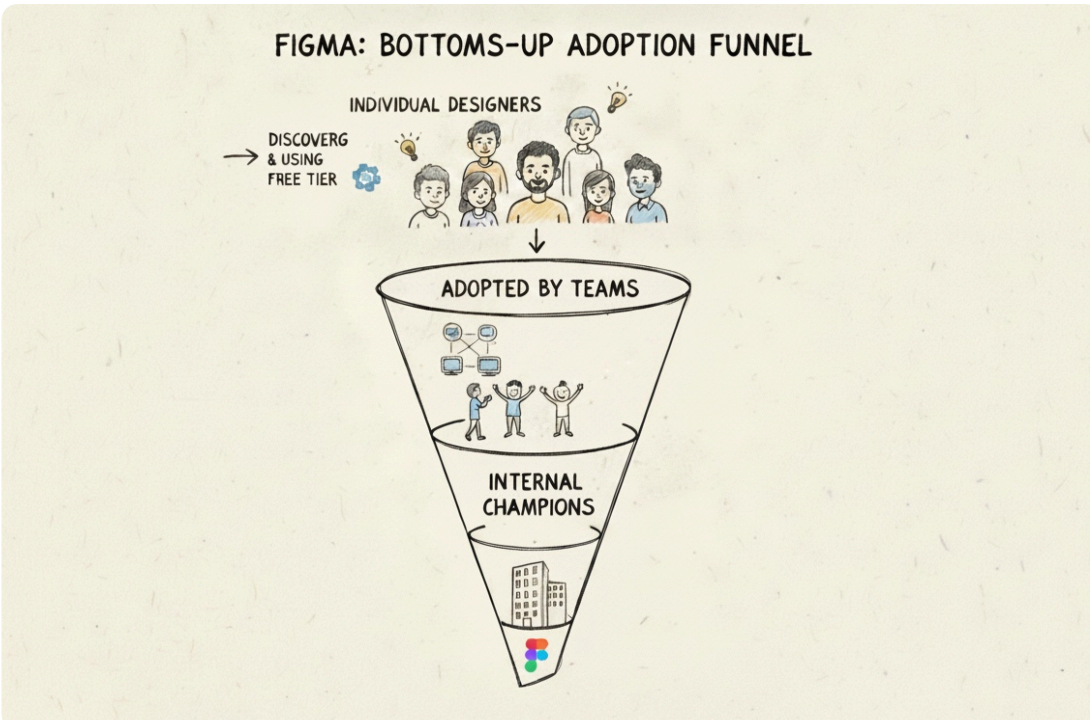
1. **Market A:** Drivers (Executor) + Earn Income (Job)
2. **Market B:** Passengers (Executor) + Get to Destination (Job)

You must make a strategic choice on which market to prioritize first. Andrew Chen describes this as the **hard side** of the network in his book "The Cold Start Problem" (2021)[25]. Chen observes that there is typically a minority of users who create disproportionate value but are harder to acquire and retain.



Two Sides of a Network Illustration by Trev de Vroome [25]

Consider Wikipedia's business model. While hundreds of millions use the platform, only a tiny fraction actively contribute content, yet these contributors create the core value that attracts all users. Similarly, for platforms like Steam (the gaming marketplace), individual game developers create content that might be downloaded millions of times.



Bottoms Up SaaS Funnel

For non-platform products, the targeting decision typically revolves around three key factors: pain point intensity, purchase decision influence, and accessibility. Modern SaaS tools like Figma illustrate this well. While both individual designers and IT teams are important stakeholders, Figma targeted individual designers first because they had the strongest pain point, could influence purchase decisions through bottom-up adoption, and were easily accessible through design communities.

For the majority of businesses, my recommendation is to focus initial targeting efforts on those who use your product **most frequently**, so typically the job executor or end consumer. This approach recognizes that sustainable product adoption and growth usually stem from solving real problems for actual users rather than just appeasing purchase decision makers.

Questions to help identify the job executor, Product Support Team, and Purchase Decision Maker

Identifying the Job Executor

The most reliable way to identify your true job executor is to ask **targeted questions** that reveal who actually performs the core functional job. Key questions include:

- Who physically interacts with and operates the product/service on a daily basis?
- Whose success metrics are directly tied to the product's core functionality?
- If the product stopped working, who would be immediately impacted in their ability to complete their work?
- In a typical workday, who spends the most time directly using the product?

Mapping the Product Lifecycle Support Team

Support teams can be identified by examining who handles the peripheral but essential tasks surrounding your product. Consider:

- Who handles product setup, configuration, and maintenance?
- When something goes wrong, who gets called first?
- Who manages user access, permissions, and system administration?

Understanding Purchase Decision Makers

To identify **purchase decision makers**, focus on financial authority and accountability:

- Who owns the budget that pays for this solution?
- Who evaluates competing solutions and creates vendor shortlists?
- Who is accountable for ROI on this purchase?

Common Pitfalls in Role Identification

Teams frequently encounter several challenges when identifying these roles:

1. **Overemphasizing organizational authority** instead of actual job execution.
2. **Mistaking frequent product interaction** for job execution.
3. **Assuming job executors are always lower-level employees**.
4. **Missing secondary job executors** in platform businesses.

By working through these questions and conducting careful observation, teams can build a clear picture of their key stakeholders and ensure their product development efforts are properly targeted.

Next Step: Uncovering the core job/need

At this point, you've mapped the stakeholder landscape for your market. You know the difference between the job executor (the person actually doing the work), the product lifecycle support team (the people who install, maintain, and troubleshoot), and the purchase decision maker (the person who controls the budget). You understand that in platform businesses, you're dealing with two distinct markets, each with its own executor and its own job.

You've also learned who to prioritize: for core product innovation, the job executor almost always takes precedence. Building for buyers instead of users creates shelfware. Building for the hard side of a network creates the foundation for marketplace liquidity. But knowing who to focus on is only half the equation. The next question is: what are they actually trying to accomplish?

This is where the core functional job comes in. The job executor you've identified doesn't just "use your product." They're trying to achieve something specific, something that exists independent of your solution. A surgeon using surgical instruments isn't just "operating equipment." They're trying to perform precise surgical procedures with minimal patient trauma. A marketing project manager using Asana isn't just "managing tasks." They might be trying to align stakeholders on project progress or translate project goals into an actionable plan. Defining this core functional job with precision is the foundation for everything that follows in your JTBD research. If you get this wrong, the rest of your research will be flawed.

In the next chapter, we'll explore how to articulate the core functional job clearly, how to navigate the hierarchy of abstraction (knowing when to zoom in and when to zoom out), and how to handle the common challenge of products that serve multiple jobs for different users. The stakeholder identification work you've done here ensures you're defining the right job for the right person.

Chapter 3 Key Takeaways

- **The Market Formula:** In ODI, a market is defined as `Market = [Job Executor] + [The Job-to-be-Done]`.
- **Define your market by the job, not the product:** This reveals the true competitive landscape (e.g., Netflix competes with sleep).
- **Distinguish the roles:** Clearly separate the Job Executor (user), the Product Lifecycle Support Team (installer/maintainer), and the Purchase Decision Maker (buyer).
- **Beware the Buyer Trap:** For core product innovation, always prioritize the needs of the Job Executor to avoid building "shelfware."
- **Two-Sided Markets:** In platform businesses, treat the supply side and demand side as two distinct markets with two different jobs.

Chapter 3: Practice Questions

These exercises are designed to help you apply the core concepts of defining a market through a Jobs-to-be-Done lens.

Exercise 1: Redefining the Market

Instructions: For each product listed below, first articulate the traditional, product-defined market. Then, redefine the market by identifying a higher-level core functional job.

1. **Product:** A high-end espresso machine for home use.
2. **Product:** A financial budgeting app (like Mint or YNAB).
3. **Product:** A project management software (like Asana or Trello).

Exercise 2: Identifying Key Stakeholder Roles

Instructions: For each scenario, identify the **Job Executor**, **Product Lifecycle Support Team**, and **Purchase Decision Maker**.

1. **Scenario:** A large law firm is purchasing a new document management system. Paralegals use it daily. IT installs it. Managing partners sign the check.
2. **Scenario:** A family buys a smart home security system. Parents monitor it. One parent installs it. Both parents pay for it. The teenager uses it to enter the house.
3. **Scenario:** A freelance marketplace connects writers with businesses.

Exercise 3: Strategic Prioritization

Scenario: You are launching a new platform "SkillSwap" connecting learners with experts. Based on the "hard side of the network" concept, which group should you prioritize in your initial go-to-market strategy: the **learners** or the **experts**? Why?

References

[22] Moonshots. (n.d.). Travis Kalanick–Episode 2. Retrieved from https://www.moonshots.io/episode-02-travis-kalanick

[23] TechGoondu. (2015, September 17). Uber: We want transportation to be as reliable as running water. Retrieved from https://www.techgoondu.com/2015/09/17/uber-we-want-transportation-be-as-reliable-as-running-water/

[24] Medium. (2014, July 16). Airbnb announces complete rebranding with a "symbol of belonging". Retrieved From https://medium.com/@bchesky/belong-anywhere-ccf42702d010

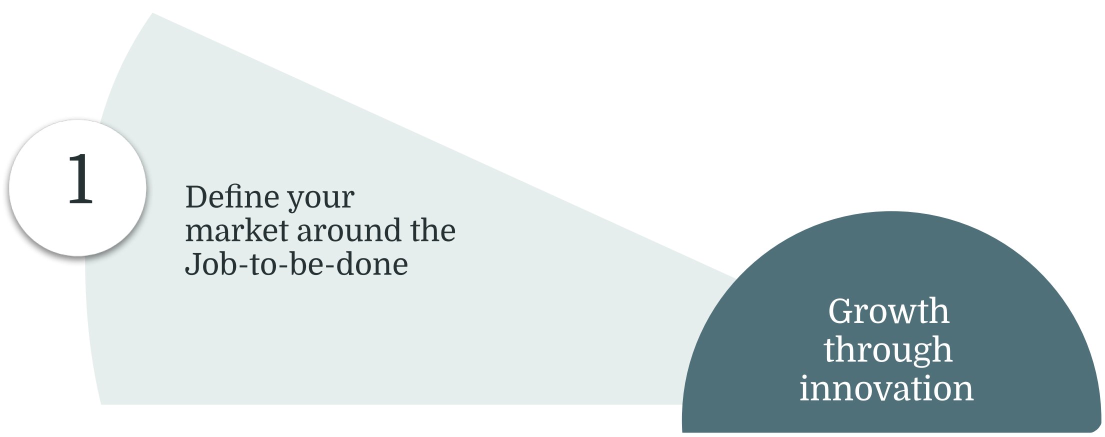
[25] Chen, A. (2021). The Cold Start Problem: How to Start and Scale Network Effects. Harper Business. Retrieved From https://www.amazon.com/Cold-Start-Problem-Andrew-Chen/dp/0062969749

[25] de Vroome, T. (2023). Book Summary: The Cold Start Problem. Medium. Retrieved From https://tdevroome.medium.com/book-summary-the-cold-start-problem-2306afb9bc9d

DEFINE YOUR MARKET AROUND THE JTBD

Chapter 4: Identifying the Core Job

Defining your core functional job



Step 1: Define your market around the job-to-be-done

With the job executor identified, the next step is defining their core functional job.

What Makes a Well-Defined Functional Job

We focus specifically on **functional** jobs because they represent the core utility customers seek from any solution. Unlike emotional jobs (how customers want to feel) or social jobs (how customers want to be perceived), functional jobs describe the practical outcomes customers need to accomplish. These functional jobs remain stable over time, making them reliable foundations for product strategy and innovation.

Whether you choose a broad or specific focus, effective job statements share several characteristics. They describe what customers want to accomplish, separate from any particular solution. Your product becomes simply a means to help them achieve this goal.

Effective core functional job statements must be:

- Formulated precisely without ambiguity
- Remain free from solutions or implementation details
- Stay stable over time as technologies and markets evolve
- Focus on functional outcomes rather than emotional or social needs

The syntax follows a structure that ensures clarity and consistency across your research:

> *Verb + object of the verb + contextual clarifier*

For example, Spotify users might have a core job to "listen to music while commuting." This statement captures the functional goal (listening to music), the object (music), and the relevant context (while commuting) without prescribing any particular solution.

Additional examples of clear functional jobs

Calculator app: "Perform mathematical calculations for daily tasks" The job isn't about having a digital tool or using any particular interface. It's about getting accurate mathematical results when needed for work, shopping, or personal finance decisions.

Weather app: "Check current weather conditions before leaving home" Users need reliable weather information to make clothing and activity decisions. The job remains the same whether fulfilled by an app, website, or looking outside.

Password manager: "Securely access online accounts across multiple devices" The functional need is seamless, secure authentication. Whether through passwords, biometrics, or future technologies, the job stays constant.

Ride-sharing app: "Get transportation from current location to desired destination" This captures the core transportation need without specifying cars, bikes, scooters, or any particular vehicle type.

Food delivery app: "Obtain prepared meals at home without cooking" The job focuses on getting ready-to-eat food conveniently, regardless of restaurant type or delivery method.

Banking app: "Transfer money between accounts during business hours" Users need to move funds securely and efficiently. The job doesn't specify mobile apps, websites, or physical locations.

Notice how each example avoids mentioning technologies, features, or implementation approaches. They describe stable functional needs that could be fulfilled through various solutions, both current and future.

These clear functional job statements become the foundation for understanding what customers truly value and where innovation opportunities exist. When you define the job properly, you can evaluate any solution based on how well it helps customers accomplish their core functional objective.

Multiple Core Jobs

What happens when your product does multiple core jobs? Take something like Slack or Facebook. These aren't simple solutions with one clear purpose. They help people accomplish several different jobs, and each one matters to different users.

This creates a real problem for product teams. Which job should you focus on? Which one drives people to use your product in the first place? You can't optimize for everything at once without ending up with a mediocre experience across the board.

To handle multiple jobs, first ask: What are you trying to learn? What is the primary research objective of this research?

The way you frame your core functional job depends entirely on what you're **trying to learn and what decisions you need to make afterward**. A team redesigning a single feature needs a different level of focus than executives planning a five-year product strategy.

Your research objectives, the people who will use these insights, and the business context you're operating in all shape how broadly or narrowly you should define the job. Getting this framing right upfront determines the scope for the entire JTBD project.

Starting with Your Research Purpose

Before diving into job identification, step back and clarify what you want to achieve. Different research objectives require different approaches to defining and analyzing jobs.

When your goal involves improving existing products, you're looking to make what you already do more effective or satisfying for current users. You'll examine current user experiences, identify friction points, and find opportunities to enhance your existing value proposition. This optimization focus keeps you within the boundaries of your current capabilities.

When you're exploring new opportunities, you're considering what you could do beyond current solution boundaries. This exploration looks for adjacent markets, new user segments, or entirely different approaches to serving customer needs that extend beyond what you currently offer.

Both approaches uncover similar foundational elements like user segments, pain points, and existing solutions. However, how you define your core job and apply your findings differs considerably based on which path you're taking.

Research Objective	Focus
Incremental product improvements	Optimizing what you already do
New innovation opportunities outside of the core product	Exploring what you could do

The question is how do we define a core functional job or jobs that are either closer to our core product for incremental improvements or define the jobs with a high enough *level of abstraction* to help uncover adjacent market opportunities.

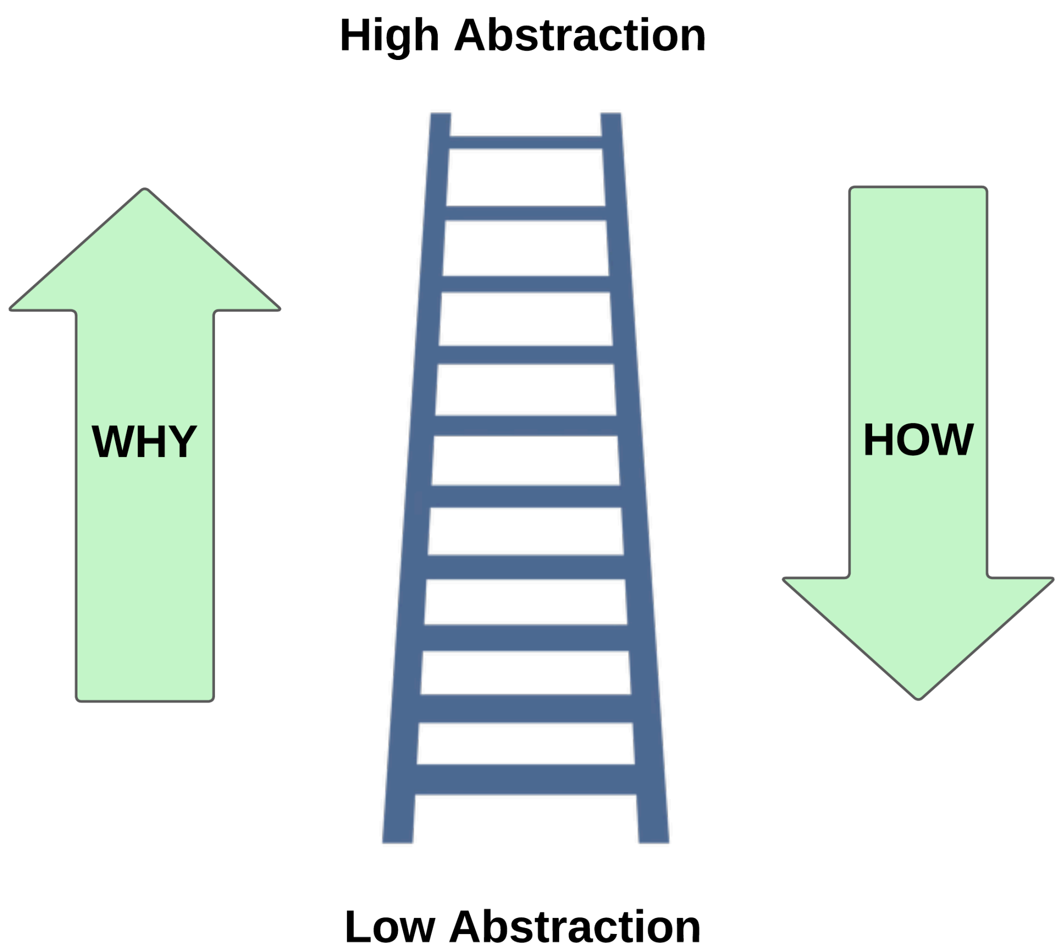
In the next section, we will look at how teams can answer this by looking at job hierarchies or what I call, determining the right "flight level".

Defining the right level of abstraction

Your research objective directly influences how broadly you should define the core job you're studying. Jobs exist in a natural hierarchy where higher levels become broader and more encompassing while lower levels become more specific and tactical.

Consider someone preparing a healthy meal at home. At a tactical level, their job might be "chop vegetables efficiently" or "follow recipe instructions accurately." Moving to a higher level reveals jobs like "prepare nutritious meals at home" or "maintain a healthy diet." At the highest level, they might be trying to "improve overall health and wellness."

You can navigate this hierarchy using two simple questions that Jim Kalbach explores in The Jobs To Be Done Playbook[26]. Asking **why** moves you toward higher abstraction and broader scope. Asking **how** moves you toward more specific, tactical details.



Determining the core functional job hierarchy

Navigating abstraction with "why" and "how"

Let's see how this works with a practical example. Starting with the core job "prepare nutritious meals at home":

Moving up with "why" (higher abstraction, broader scope):

- Why prepare nutritious meals at home? → To maintain a healthy diet
- Why maintain a healthy diet? → To improve overall health and wellness
- Why improve overall health and wellness? → To live a fulfilling, energetic life

Moving down with "how" (lower abstraction, more specific):

- How do you prepare nutritious meals at home? → Plan balanced recipes and source quality ingredients
- How do you plan balanced recipes? → Research nutritional requirements and select appropriate foods
- How do you select appropriate foods? → Read nutrition labels and compare ingredient quality

Notice how the "why" questions reveal increasingly broader jobs that open up entirely different solution spaces, while the "how" questions drive toward more tactical, implementation-focused jobs. Notably, even as we move down the hierarchy, we remain solution-agnostic. We're describing what needs to be accomplished, not prescribing products or methods.

Consider another example with a professional context. A marketing manager's job hierarchy might look like this:

Higher abstraction (asking "why"):

- Create compelling marketing campaigns → Drive business growth → Ensure company success and sustainability

Lower abstraction (asking "how"):

- Create compelling marketing campaigns → Develop targeted messaging for particular audiences → Research audience preferences and pain points

Choosing Your Research Approach: A Decision Framework

Before defining your core job and selecting an abstraction level, use this framework to ensure you're applying the right methodology to your business question.

Step 1: What is your primary research objective?

Start by identifying what question you're ultimately trying to answer:

If your question sounds like…	Your objective is…	Recommended approach
"What markets should we enter?"	**Strategic exploration**	ODI at high abstraction
"How do we position against competitors?"	**Strategic positioning**	ODI at mid-to-high abstraction
"What should our 3-year roadmap look like?"	**Strategic planning**	ODI at mid abstraction
"Which features should we build next quarter?"	**Tactical prioritization**	Consider alternatives to ODI
"How do we improve our checkout flow?"	**Tactical optimization**	Use usability testing, A/B testing
"What's causing users to churn?"	**Diagnostic research**	Combine methods; ODI may help with "why"

Step 2: Who will act on your findings?

Your audience determines the level of abstraction that will be most useful:

Primary audience	Abstraction level	Why
C-suite / Board	High	They need vision, market opportunities, portfolio decisions
Product leadership	Mid-to-High	They balance strategic direction with roadmap implications
Product teams / Engineers	Mid-to-Low	They need specificity for feature development
UX/Design teams	Mid-to-Low	They need concrete pain points for interface solutions
Marketing teams	Varies	High for positioning, lower for campaign targeting

Step 3: What's your timeline for action?

Timeline	Recommended approach
Weeks (need insights for immediate decisions)	ODI is likely too slow. Use rapid methods like user interviews, usability testing, or analytics review
Months (planning next quarter or two)	ODI can work if scoped carefully. Consider a focused study on a particular job area
6+ months (strategic planning horizon)	ODI is well-suited. Invest in comprehensive job mapping and quantification

Step 4: Decision checkpoint

Based on your answers above, determine your path:

→ If strategic objective + senior audience + longer timeline: Proceed with ODI. Use this chapter to define your core job at the appropriate abstraction level.

→ If tactical objective + product team audience + short timeline: ODI will likely create more complexity than clarity. Consider:

- **For feature prioritization**: MaxDiff studies, Kano analysis, or conjoint analysis
- **For usability improvements**: Usability testing, heuristic evaluation, or session recordings
- **For conversion optimization**: A/B testing, funnel analysis, or behavioral analytics
- **For understanding user needs qualitatively**: Switch interviews (Bob Moesta), demand-side research (Alan Klement), or continuous discovery interviews

→ **If mixed objectives or uncertain:** Start with qualitative JTBD interviews to understand the landscape, then decide whether full ODI quantification is warranted. You can always use the job mapping frameworks from this book without committing to the full ODI quantification process.

Aligning abstraction level with research objectives

This hierarchy matters when you consider how your research objectives should influence where you focus. The level you choose determines not only what you study but what solutions become visible.

For incremental improvements, match your product's current scope. If your meal kit service handles ingredient sourcing and recipe curation, focus on jobs like "efficiently prepare home-cooked meals with minimal planning" rather than the broader "improve overall health." This focused approach helps you identify pain points in the cooking process. Perhaps customers struggle with timing multiple dishes or dealing with unfamiliar cooking techniques.

For new markets or opportunities outside your core business, explore higher abstraction levels. The health improvement job reveals possibilities extending far beyond meal preparation: fitness tracking integration, personalized wellness coaching, nutritional deficiency monitoring, or even sleep and stress management solutions. A meal kit company operating at this level might discover opportunities in mental health support, since nutrition strongly impacts mood and cognitive function.

For tactical optimization, dive into lower levels. If you're improving an existing recipe app, focusing on jobs like "accurately measure ingredients" or "coordinate cooking timing" reveals usability improvements. These might include better measurement tools, timer integration, or step-by-step visual guidance.

> ***Author Note**: If teams are looking to be tactical in terms of product improvements, I would recommend not using the ODI approach. Potentially look at other applications of JTBD like Alan Klements, Bob Moesta, Ash Maurya, etc. Reason is because the ODI approach will have teams left with 50-60 outcome/need statements that make it incredibly difficult for tactical product teams to know what to deal with. More on this later in the chapter.*

Extended examples across industries

Financial services example:

- High level: Achieve financial security and peace of mind
- Mid level: Effectively manage personal finances
- Low level: Track monthly expenses → Categorize individual transactions

A traditional bank focusing on the mid-level job might develop better budgeting tools. But exploring the higher-level job could reveal opportunities in financial education, insurance products, or even career development services that help people earn more.

Fitness industry example:

- High level: Live a healthy, confident lifestyle
- Mid level: Maintain physical fitness
- Low level: Complete effective workouts → "Perform exercises with proper form

A gym focusing on the lower level might invest in better equipment or instructional videos. But the higher level reveals opportunities in nutrition counseling, mental health support, social community building, or even career coaching that builds confidence in other life areas.

Software development example:

- High level: Deliver valuable software that serves users
- Mid level: Build reliable, maintainable applications
- Low level: Write clean, efficient code" → Debug functionality

Development tool companies operating at different levels create vastly different solutions. Lower-level focus yields better debuggers and code editors. Higher-level focus might produce user research tools, product management platforms, or even business strategy software.

Matching Your Approach to Your Audience

The stakeholders who will use your research insights should heavily influence how you define and present jobs. A job definition that helps executives make portfolio decisions might be too abstract for product teams building features. Conversely, tactical job definitions that guide feature development might miss the strategic lens that inform market expansion decisions.

This audience consideration connects directly to the abstraction level choices we discussed earlier. Different organizational roles need different flight levels to make effective decisions and take meaningful action. **Your job definition should serve the people who will apply your research findings**.

Stakeholder	High-Level Abstraction Benefits	Low-Level Abstraction Benefits
C-suite	Vision, new market opportunities, portfolio decisions	Resource allocation for features, ROI on tactical improvements
Product Teams	Product roadmap expansion, adjacent product opportunities	Feature prioritization, user experience optimization
Marketing/Brand	Brand positioning, market expansion messaging	Campaign targeting, feature-specific messaging
Strategy/Consulting	Market opportunity assessment, competitive positioning	Process optimization, tactical recommendations
Growth Teams	New user acquisition channels, market expansion	Conversion optimization, retention improvements
UX/Design Teams	Holistic user experience design, journey mapping	Interface improvements, pain point solutions
Engineering	Architecture decisions for broader capabilities	Technical debt prioritization, feature development
Sales Teams	New market segments, expanded solution selling	Feature demonstrations, objection handling

C-suite executives need clarity for market positioning and portfolio decisions, benefiting from high-level abstraction that reveals new market opportunities and competitive positioning insights. Product teams need tactical specificity for feature development and user experience optimization, though they also benefit from understanding broader roadmap expansion possibilities.

Marketing and brand teams use high-level insights for positioning and market expansion messaging while applying low-level abstraction findings that are closer to the product for campaign targeting and feature-specific communications. UX and design teams use broad perspectives for holistic user experience design and journey mapping while using detailed insights for interface improvements and pain point solutions.

Best use of JTBD and ODI Research

In my experience, JTBD and specifically ODI research excel when addressing **strategic business questions** and **engaging senior stakeholders**, but struggle with **tactical product decisions**.

JTBD works best for strategic questions like "What markets should we enter?" or "How do we redefine our competitive landscape?" The methodology's strength according to Ulwick and others lies in **revealing unmet needs across broader solution spaces**, **uncovering adjacent market opportunities from a needs perspective**, and **providing frameworks that senior leaders can use to communicate a perspective of their strategy**. Directors and above value these insights because it connects customer needs to business strategy in ways that inform portfolio decisions and resource allocation.

For example, companies like Amazon have used job-based thinking to expand beyond their original scope. Understanding that customers wanted to "acquire needed products efficiently" rather than just "buy books online" opened pathways to everything from cloud computing services to grocery delivery. The job framework provided a common language for executives to evaluate seemingly unrelated opportunities against a consistent customer value proposition. Look at the quote below from Jeff Bezos, former CEO and founder of Amazon.

> *This is another really good and deep question because there are big things that are really important to manage, and then there are small things. Internally into Amazon, we call them paper cuts. So we're always working on the* ***big things, if you ask me. And most of the energy goes into the big things****, as it should, and you can identify the* ***big things****. And I would encourage anybody, if anybody listening to this is an entrepreneur, has a small business, whatever, think about the things that are* ***not going to change over 10 years****. And those are probably the big things. So I know in our retail business at Amazon, 10 years from now, customers are still going to want low prices. - Jeff Bezos, [Lex Fridman Podcast]*

However, this strength or positive becomes a weakness when teams need **immediate, actionable guidance** for product development. JTBD research typically requires many months to execute properly, involving customer interviews, job mapping, and competitive analysis. Also, the end output of ODI research is a list of unmet need statements in a rigid syntax that is hard for product teams to follow (more on this in chapter 5). Product teams working in agile environments need insights they can act on within days, not quarters.

For tactical product decisions like feature prioritization, interface design, or conversion optimization, JTBD often creates **more complexity** in my experience. Teams asking "What should we build next sprint?" or "Which features drive retention?" typically get faster, more actionable insights from methods like Conjoints, MaxDiff, user interviews, focus groups, usability testing, or behavioral analytics. JTBD's broad lens can actually mislead teams into over-analyzing decisions that benefit from rapid experimentation.

Consider a product team trying to improve their mobile app's checkout process. JTBD research might reveal that customers want to "complete purchases quickly during busy moments." While accurate, this insight doesn't tell the team whether to reduce form fields, add payment options, or improve error messaging. A/B testing different checkout flows provides clearer direction for immediate improvements.

The ODI framework also struggles with questions about user interface design and interaction patterns. Asking customers about their jobs rarely yields insights about button placement, navigation hierarchy, or visual design preferences. These tactical elements require direct observation of user behavior and rapid testing cycles that JTBD's comprehensive approach can't match.

Furthermore, ODI research can overwhelm product teams with too many possibilities. When research reveals multiple unmet needs across different job stages, teams often struggle to prioritize what to build first. The methodology's strength in revealing opportunities becomes a paralysis of choice for teams that need clear direction for their next development sprint.

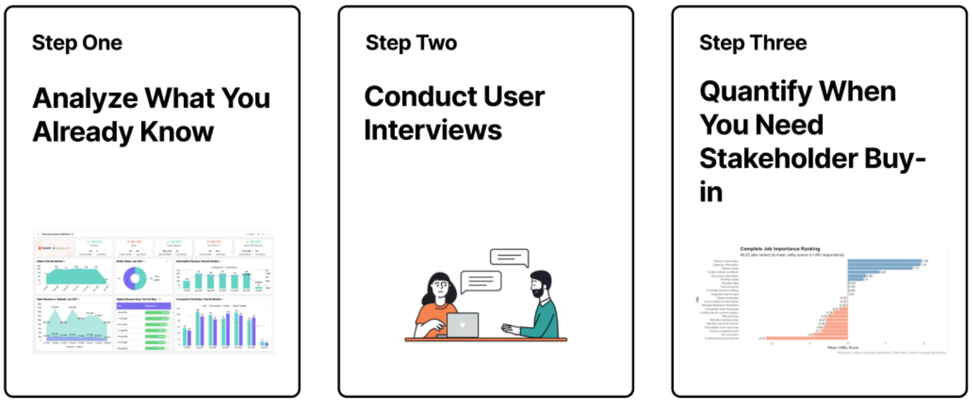
So my recommendation is to use ODI to establish strategic direction and market understanding, then transition to different research methods for execution. This approach gives senior stakeholders the strategic narrative they need while providing product teams the tactical insights that drive day-to-day decisions.

The most effective organizations I've worked with create a research pipeline that flows from strategic to tactical. They begin with JTBD using the ODI approach to establish market positioning and identify opportunity areas. Once they've committed to pursuing specific opportunities, they shift to other research methods that inform feature development and user experience optimization. This staged approach ensures that tactical decisions support strategic direction while maintaining the speed that product development requires.

The key is matching the research method to both the business question and the stakeholder who needs to act on the results. JTBD and ODI provides the strategic foundation that guides where to focus effort, while other methodologies provide the tactical insights that determine how to focus that effort most effectively.

Narrowing Down to Your Core Job

Once you recognize that your product likely serves multiple jobs, you need a way to identify which ones matter most. These three steps may help you determine what the core job of your platform is.



Three steps to narrow down to your core functional job

Step 1: Analyze What You Already Know

Begin by examining data you already have access to within your organization. Usage analytics reveal which features and workflows drive the most engagement, while support tickets and churn analysis illuminate where current solutions fail to serve user needs effectively. Customer feedback, reviews, and sales conversations often explicitly mention the primary goals users are trying to accomplish.

Look beyond surface level metrics to understand the why behind user behavior. High engagement with a particular feature might indicate it serves a core job, but it could also signal friction elsewhere in the experience. Similarly, features with low usage aren't necessarily unimportant they might serve highly important but infrequent jobs, or they might be poorly designed solutions to important problems.

Support data offers particularly rich insights into job priorities. Tickets that appear repeatedly across different user segments often point to gaps in serving a core job. Pay attention to the language customers use when describing their problems they'll often articulate the job they're trying to accomplish and why your current solution isn't helping them get it done.

Sales conversations and customer success interactions provide another valuable lens. Listen for phrases like "we bought this to help us" or "our main goal is" These conversations capture jobs at the moment of purchase decision and throughout the customer lifecycle, revealing both primary motivations and evolving needs.

Pay particular attention to segmentation patterns in this data. Different user segments often hire your product for different primary jobs, and understanding these patterns helps you make informed decisions about where to focus your research and development efforts. A feature that's critical for enterprise customers might be irrelevant for small businesses, suggesting different core jobs across segments.

Step 2: Conduct User Interviews

While you'll learn detailed interview techniques in Chapter 5, the key at this stage is conducting interviews focused on job prioritization rather than general discovery. Interview participants from your primary market segments and job executor roles to uncover what main goals they're trying to accomplish using your product.

Structure these conversations around understanding not just what jobs they're trying to get done, but which jobs are most important, most frequent, and most poorly served by current solutions. Ask participants to walk you through their typical usage patterns and describe what success looks like for their primary use cases.

Focus on comparative questions that force prioritization decisions. Rather than asking "What do you use our product for?" try "When you absolutely need to get something done quickly, what's the first thing you turn to our product for?" or "If you could only use three features, which would they be and why?" These questions reveal true priorities rather than comprehensive feature lists.

Pay close attention to the emotional language participants use when describing different jobs. Frustration, urgency, and relief are strong indicators of job importance. When someone says "I was so stressed until I figured out how to do this" or "This completely changed how we work," they're highlighting a core job your product serves.

Don't just focus on current usage patterns. Ask about workarounds, alternative solutions, and what they did before your product existed. Understanding the full context of how they approach these jobs helps you see where your product fits in their larger workflow and which aspects truly matter most to them.

> ***Authors Note****: I believe most of the time by analyzing previous data sources, speaking with customers, and even talking through with internal stakeholders, teams can have an 80%-90% confidence or strong enough point in knowing what the core functional job(s) of their product/service they should focus on* ***without*** *having to run any quantification research. In my opinion, the reason to run quantification research to understand the underlying needs to prioritize or hidden patterns in how users might cluster jobs is to make stakeholders happy with big samples.*

Interview and Analysis: Uncovering Multiple Core Jobs

Products, especially complex platforms like Asana, are often hired to do several different core jobs for the same user. A skilled interviewer can uncover these distinct jobs by probing different phases of the user's workflow.

Let's continue our conversation with Sarah, the marketing project manager, to see what else we can learn.

Extended Interview with Sarah

Interviewer: Thanks for speaking with me, Sarah. Can you start by walking me through the last time you used Asana?

Sarah: Sure. This morning, actually. I had our weekly check-in with the VP of Marketing. Before the meeting, I opened up the main project Portfolio in Asana to get a quick snapshot of where everything stood. I needed to see which tasks were overdue and what the team had completed since last week.

Interviewer: What was the most important thing you needed to accomplish with Asana at that moment?

Sarah: Definitely getting a clear status update for the VP. Before we had Asana, that was a nightmare. I'd spend an hour before each meeting pinging people on Slack, digging through email threads... it was chaos. Now, I can just look at the dashboard and have a confident answer when she asks, "Are we on track for the launch?" That single view saves me from so much stress. The **feeling of relief is huge**.

Interviewer: Thanks for that insight on reporting. It's clear that visibility for leadership is critical. I'd like to shift focus a bit. Can you walk me through the beginning of a project, before any tasks are even assigned?

Sarah: Oh, the beginning is the messiest part. I get a project brief from a director, and it's usually just a goal, like "Launch the new winter campaign." My job is to turn that one sentence into a full-blown project plan. Before Asana, I'd do this in a spreadsheet. It was terrible because I'd always miss dependencies. The copywriter couldn't start until the messaging was approved, but the designer needed copy for the mockups... it was a chicken-and-egg problem. It felt like I was building a puzzle in the dark.

Interviewer: How has Asana changed that for you?

Sarah: The Timeline view is a lifesaver. I can visually map out the phases and drag-and-drop dependencies. If I see that a design task depends on a copy task, I can link them. When I finish, I have a realistic schedule that I can actually commit to. It's the difference between guessing and **knowing we have a viable plan**.

Interviewer: That makes sense. Now let's talk about the middle of a project, when things are in full swing. What's a major challenge you face there?

Sarah: Definitely managing team workload. I have three designers, and one of them, Alex, is a superstar, so everyone wants him on their project. In the past, I had no real way of knowing how much was on his plate. I'd assign him a new task, and he'd say yes, but then he'd be working until 10 PM. I felt awful. **Preventing my team from burning out** is one of the things that keeps me up at night.

Interviewer: And how do you approach that challenge now?

Sarah: The Workload feature in Asana has been a game-changer. I can see at a glance that Alex is at 150% capacity while another designer is at 60%. It allows me to reassign tasks and have a real conversation about priorities. It's not about micromanaging; it's about **protecting my team's well-being** so we can maintain a sustainable pace.

Interviewer: One last question. What about your *own* work? How do you start your day?

Sarah: It's funny, we've talked all about the team, but my personal "My Tasks" page is my command center. I come in, and I have 30 tasks assigned to me from five different projects. It's overwhelming. My first action every morning is to sort that list by due date and manually add my top three priorities for the day. That simple act of sorting and choosing helps me **cut through the noise and focus** on what I need to accomplish.

Breaking Down the Interview to Find Multiple Core Jobs

This extended conversation reveals that Sarah hires Asana for at least four distinct functional jobs, each addressing a different pain point in her workflow.

Job 1: Aligning Stakeholders

- **Evidence:** The first part of the interview, where Sarah expressed "relief" at being able to answer her VP's questions confidently. Her pain was the "chaos" of manually chasing down status updates.
- **Synthesis:**
  - Verb: **Align**
  - Object: **stakeholders**
  - Clarifier: **on project progress**

- **Core Job: Align stakeholders on project progress.**

Job 2: Planning Projects

- **Evidence:** Sarah described the beginning of a project as "messy" and like "building a puzzle in the dark." The key outcome she needs is a "realistic schedule" and a "viable plan."
- **Synthesis:**
    - Verb: **Translate**
    - Object: **project goals**
    - Clarifier: **into an actionable plan**
- **Core Job: Translate project goals into an actionable plan.**

Job 3: Managing Team Capacity

- **Evidence:** Sarah's fear of "burning out" her team and the stress of not knowing individual workloads. The desired outcome is a "sustainable pace" and protecting her team's "well-being."
- **Synthesis:**
    - Verb: **Balance**
    - Object: **team workload**
    - Clarifier: **to ensure project sustainability**
- **Core Job: Balance team workload to ensure project sustainability.**

Job 4: Prioritizing Personal Tasks

- **Evidence:** Sarah described her own task list as "overwhelming." Her daily ritual is to "cut through the noise and focus." This is a more personal, execution-focused job.
- **Synthesis:**
    - Verb: **Prioritize**
    - Object: **daily tasks**
    - Clarifier: **to maintain personal focus**
- **Core Job: Prioritize daily tasks to maintain personal focus.**

This exercise shows how a single user can have multiple, equally valid core jobs. For a product team at Asana, recognizing these distinct jobs is crucial. They could decide to focus on improving the stakeholder reporting experience, the initial project planning tools, team management features, or the personal productivity workflow. Each path would lead to a different product strategy and roadmap.

Step 3: Validate Job Priorities with Your User Base

Once you have identified a list of functional jobs through interviews, you are often left with a long and unmanageable list of needs. To understand the broader strategy, you need to condense this list into manageable dimensions.

Quantifying Job Relationships with Exploratory Factor Analysis (EFA)

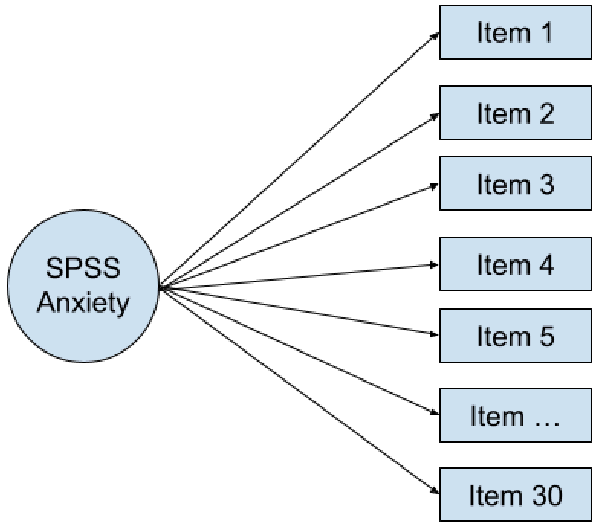


Image source: Wikipedia — "Exploratory factor analysis." Retrieved from https://en.wikipedia.org/wiki/Exploratory_factor_analysis

When you decide to quantify job importance, we turn to Exploratory Factor Analysis, or EFA. Joseph Hair and colleagues, authors of the standard text on multivariate analysis, define EFA as an "interdependence technique." [29]

Unlike regression analysis, which tries to predict an outcome using inputs, EFA looks at all variables simultaneously to define the underlying structure among them.

The Goal: Managing the Variate

As we add more variables (job statements) to our research, the likelihood of overlap or correlation between them increases. EFA allows us to manage this complexity by achieving two goals:

1. **Data Reduction:** We take a large number of variables, such as 22 Notion features, and reduce them to a smaller number of summary "Factors" with minimal loss of information.

2. **Structure Detection:** We identify which variables are highly intercorrelated. These groups of variables are assumed to represent a single underlying dimension or "Core Job" that cannot be adequately described by a single measure.

The Instrument: The Likert Scale

To prepare our data for analysis, we must capture it quantitatively. The standard practice in JTBD research is to use metric data. While nonmetric data (Yes/No) is possible to analyze, it is often problematic for this technique. Therefore, we use a standard Likert Scale (typically 1–5 or 1–7) to measure importance.

For each job identified in the qualitative phase, we ask the user:

> *"When trying to [Core Job], how important is it to you that you be able to [Specific Functional Job]?" (1 = Not at all important, 5 = Extremely important)*

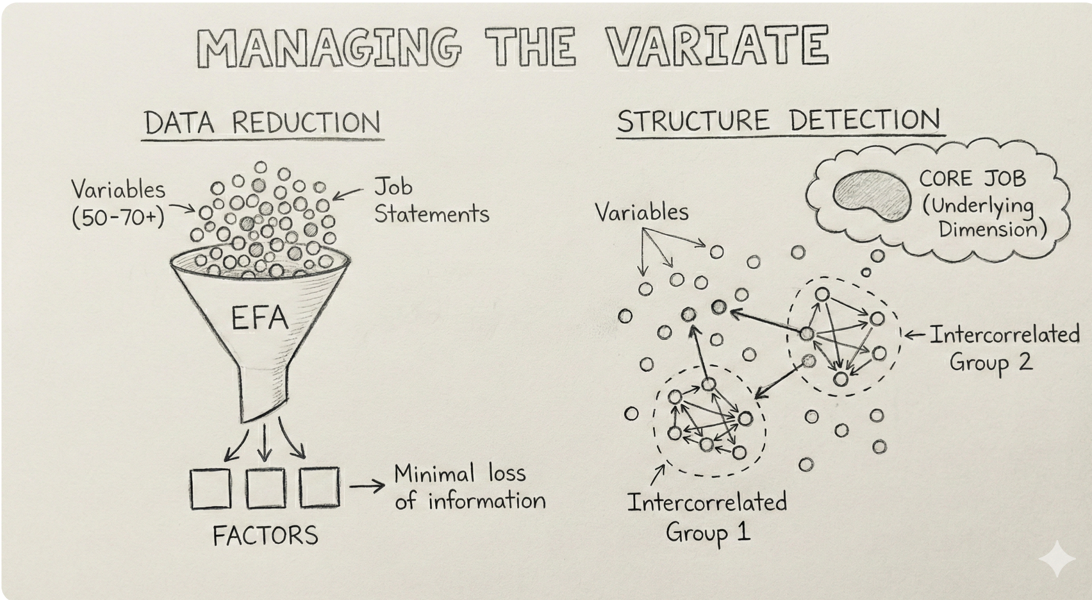
This creates the correlation matrix used for analysis.

How it Works: Finding Latent Dimensions

In your survey, you might ask users about "sharing documents," "commenting," and "assigning permissions." EFA analyzes the correlation matrix of these responses. If users who rate "sharing" highly also consistently rate "permissions" highly, EFA groups them together.

Since we have no preconceived hypothesis on how these jobs relate, we use an **Exploratory** perspective. We take what the data gives us. If we already had a theory we wanted to test, we would use a **Confirmatory** perspective. For this chapter, we are taking an exploratory approach to discover how Notion users naturally group these capabilities in their own minds.

The Objective: Summarization vs. Reduction



Managing The Variate Illustration

According to Hair, we are attempting to achieve two distinct outcomes with this process:

- **Data Summarization:** We view the variables collectively. Instead of tracking 22 separate behaviors, we look for the underlying structure that binds them together. This helps us describe the market in broad strokes. For example, we might find that five different features actually represent one desire: Efficiency.
- **Data Reduction:** We calculate a new, empirical value for each dimension, known as a Factor Score. This allows us to substitute the original 22 variables with just 4 Factor Scores in future analyses to simplify our decision making.

Analyzing Factor Analysis Results: The Notion Example

> ***Note:*** *For readers interested in the statistical methodology, the following section provides a step-by-step guide using R. If you prefer to focus on the outcomes, you can skip ahead to the "Factor Analysis Conclusion" section.*

Now let's examine real factor analysis results from 300 Notion users rating 22 different functional capabilities on a 5-point Likert scale.

Factor analysis using simulated data

Download Raw Survey Data (300 responses)

R

```r
# Load required libraries
library(psych)         # Main factor analysis package
library(corrplot)      # Correlation visualization
library(ggplot2)       # Advanced plotting
library(dplyr)         # Data manipulation

# Load your survey data
survey_data <- read.csv("jobs_survey_data.csv")
str(survey_data)
describe(survey_data)
```

str() & describe()

R OUTPUT

```
LOADINGS ===
>str(survey_data)
'data.frame':   300 obs. of  22 variables:
$ Organize_information        : int  5 4 4 3 4 4 3 4 5 3 ...
$ Document_information        : int  5 5 5 3 4 4 2 5 4 4 ...
$ Capture_ideas               : int  4 4 3 3 4 4 2 4 4 4 ...
$ Retrieve_information        : int  5 4 4 4 4 5 4 4 5 3 ...
$ Share_knowledge             : int  5 3 3 4 4 4 2 3 4 4 ...
$ Maintain_personal_records   : int  5 4 4 3 4 5 3 4 4 4 ...
$ Archive_completed_work      : int  5 4 4 3 3 4 4 4 5 4 ...
$ Plan_activities             : int  5 4 5 4 3 4 4 4 3 5 ...
$ Track_progress              : int  5 5 5 5 4 5 4 5 4 5 ...
$ Collaborate_content_creation: int  5 3 4 4 4 4 3 3 4 3 ...
$ Manage_database_information : int  4 3 3 4 4 3 3 4 3 3 ...
$ Create_custom_workflows     : int  5 3 4 3 4 3 3 3 3 2 ...
$ Coordinate_team_resources   : int  5 4 3 5 4 4 3 3 4 5 ...
$ Prioritize_tasks            : int  5 2 5 4 4 4 4 4 5 5 ...
$ Facilitate_decision_making  : int  5 4 5 3 4 5 3 5 4 5 ...
$ Consolidate_work_resources  : int  4 4 5 4 2 4 3 4 4 5 ...
$ Visualize_data              : int  5 4 3 4 3 3 3 3 3 4 ...
$ Maintain_meeting_notes      : int  4 2 4 5 5 4 3 3 4 4 ...
$ Integrate_external_data     : int  5 5 3 4 3 3 1 4 2 3 ...
$ Cross_reference_information : int  4 4 4 3 4 5 3 5 4 4 ...
$ Set_reminders               : int  5 3 5 4 3 3 3 3 4 4 ...
$ Control_access_permissions  : int  3 2 3 5 5 4 3 1 5 5 ...

> describe(survey_data)

                           vars   n mean   sd median trimmed  mad min max rang

Organize_information 1 300 3.94 0.85 4 3.98 1.48 2 5 3 -0.26 -0.84 0.05
Document_information 2 300 3.87 0.84 4 3.90 1.48 2 5 3 -0.26 -0.62 0.05
Capture_ideas 3 300 3.60 0.83 4 3.59 1.48 2 5 3 0.11 -0.64 0.05
Retrieve_information 4 300 3.86 0.88 4 3.90 1.48 1 5 4 -0.40 -0.28 0.05
Share_knowledge 5 300 3.43 1.01 3 3.43 1.48 1 5 4 -0.04 -0.82 0.06
Maintain_personal_records 6 300 3.70 0.86 4 3.73 1.48 2 5 3 -0.15 -0.66 0.05
Archive_completed_work 7 300 3.77 0.84 4 3.80 1.48 2 5 3 -0.27 -0.52 0.05
Plan_activities 8 300 4.03 0.80 4 4.08 1.48 2 5 3 -0.37 -0.65 0.05
Track_progress 9 300 3.96 0.84 4 4.01 1.48 2 5 3 -0.40 -0.54 0.05
Collaborate_content_creation 10 300 3.47 0.92 3 3.47 1.48 1 5 4 0.00 -0.72 0.0
Manage_database_information 11 300 3.13 0.92 3 3.10 1.48 1 5 4 0.10 -0.31 0.05
Create_custom_workflows 12 300 3.09 0.92 3 3.08 1.48 1 5 4 0.03 -0.47 0.05
Coordinate_team_resources 13 300 3.32 0.98 3 3.29 1.48 1 5 4 0.12 -0.61 0.06
```

```
Prioritize_tasks 14 300 3.99 0.83 4 4.01 1.48 1 5 4 -0.29 -0.68 0.05
Facilitate_decision_making 15 300 3.71 0.88 4 3.73 1.48 1 5 4 -0.17 -0.57 0.05
Consolidate_work_resources 16 300 3.78 0.86 4 3.80 1.48 2 5 3 -0.14 -0.77 0.05
Visualize_data 17 300 2.99 1.08 3 2.98 1.48 1 5 4 0.05 -0.62 0.06
Maintain_meeting_notes 18 300 3.41 1.01 3 3.40 1.48 1 5 4 0.02 -0.82 0.06
Integrate_external_data 19 300 2.95 1.03 3 2.90 1.48 1 5 4 0.25 -0.52 0.06
Cross_reference_information 20 300 3.76 0.86 4 3.78 1.48 2 5 3 -0.13 -0.75 0.0
Set_reminders 21 300 3.91 0.83 4 3.94 1.48 2 5 3 -0.27 -0.67 0.05
Control_access_permissions 22 300 3.23 1.05 3 3.20 1.48 1 5 4 0.07 -0.73 0.06

>
```

The describe() function from the psych package provides descriptive statistics for each job rating, including mean, standard deviation, median, skewness, and kurtosis. The str() function shows the data structure, confirming that your Likert scale responses are properly formatted as numeric variables rather than factors or characters.

This output confirms we have 300 complete responses across 22 functional jobs, with all variables properly coded as integers reflecting our 1-5 Likert scale. Notice there are no factor variables or missing data indicators, which is exactly what we want for factor analysis. If you ever see variables coded as factors or characters in your own data, you'll need to convert them to numeric before proceeding with any quantitative analysis.

The descriptive statistics reveal insights about how users perceive Notion's job portfolio. A clear job importance hierarchy emerges from the mean ratings, with core productivity jobs like planning activities (4.03), tracking progress (3.96), and prioritizing tasks (3.99) rating highest among users. At the other end of the spectrum, specialized technical jobs such as data visualization (2.99) and external data integration (2.95) receive notably lower ratings, suggesting natural user segments based on job complexity and technical sophistication. The scale usage patterns validate that our survey captured meaningful user preferences. All variables show full 1-5 range usage with reasonable standard deviations between 0.8 and 1.1, indicating users are thoughtfully differentiating between jobs rather than giving uniform ratings across all items. This variability is essential for factor analysis to identify meaningful patterns.

From a statistical perspective, our data appears ready for factor analysis. The skewness values ranging from -0.4 to +0.25 fall within acceptable ranges, and the similarity between trimmed means and regular means indicates no extreme outliers that could distort our factor extraction. The sufficient variability across all items provides the foundation needed for identifying underlying factor structures.

Testing Factor Analysis Readiness for Your JTBD Data Let's see how we can test that our Notion data is ready for factor analysis. Factor analysis makes several key assumptions about your data structure and quality. If these assumptions aren't met, your factor solution may be unstable, uninterpretable, or simply meaningless for product decisions.

Key assumptions that must hold for reliable factor analysis:

- Sample size adequacy requires enough respondents relative to the number of variables being analyzed. The general rule is at least 5 respondents per variable, with 10:1 being preferred for stable solutions.
- Sampling adequacy means your variables share enough common variance to form coherent factors. If variables are too independent of each other, there's no underlying structure to discover.
- Sufficient correlations between variables must exist for factor analysis to identify patterns. If all variables are uncorrelated, there are no factors to extract.
- Linearity assumes that relationships between variables follow linear patterns rather than complex curves or thresholds.

Let's test these assumptions with our Notion JTBD data:

R

```r
# Calculate correlation matrix for all subsequent tests
cor_matrix <- cor(survey_data, use = "complete.obs")

# 1. Sample size adequacy check
sample_ratio <- nrow(survey_data) / ncol(survey_data)
print(paste("Sample size ratio:", round(sample_ratio, 1), ":

# 2. Kaiser-Meyer-Olkin (KMO) test for sampling adequacy
kmo_result <- KMO(cor_matrix)
print(paste("Overall KMO value:", round(kmo_result$MSA, 3)))

# Check individual variable KMO values for problematic items
low_kmo <- which(kmo_result$MSAi < 0.6)
if(length(low_kmo) > 0) {
  print(paste("Variables with low individual KMO (<0.6):", n
} else {
  print("All individual variables have adequate KMO values")
}

# 3. Bartlett's test of sphericity
bartlett_result <- cortest.bartlett(cor_matrix, n = nrow(sur
print(paste("Bartlett's test p-value:", format(bartlett_resu

# 4. Examine correlation matrix characteristics
cor_values <- cor_matrix[upper.tri(cor_matrix)]
print(paste("Mean correlation:", round(mean(cor_values), 3))
print(paste("Correlation range:", round(min(cor_values), 3),

strong_corr <- sum(abs(cor_values) > 0.3)
total_corr <- length(cor_values)
print(paste("Strong correlations (|r| > 0.3):", strong_corr,
            paste0("(", round(100*strong_corr/total_corr), "%

cor_matrix <- cor(survey_data, use = "complete.obs")
```

Factor Analysis Assumptions

R OUTPUT

```
size ratio: 13.6 :1"
[1] "Overall KMO value: 0.933"
[1] "All individual variables have adequate KMO values"
[1] "Bartlett's test p-value: 0e+00"
[1] "Mean correlation: 0.351"
[1] "Correlation range: 0.03 to 0.641"
[1] "Strong correlations (|r| > 0.3): 165 of 231 (71%)"
```

The sample size ratio of 13.6:1 means we have nearly 14 respondents for every job we're analyzing, well above the minimum threshold that statisticians recommend for stable factor solutions. This excellent ratio gives us confidence that our factor structure will be robust and generalizable beyond this sample.

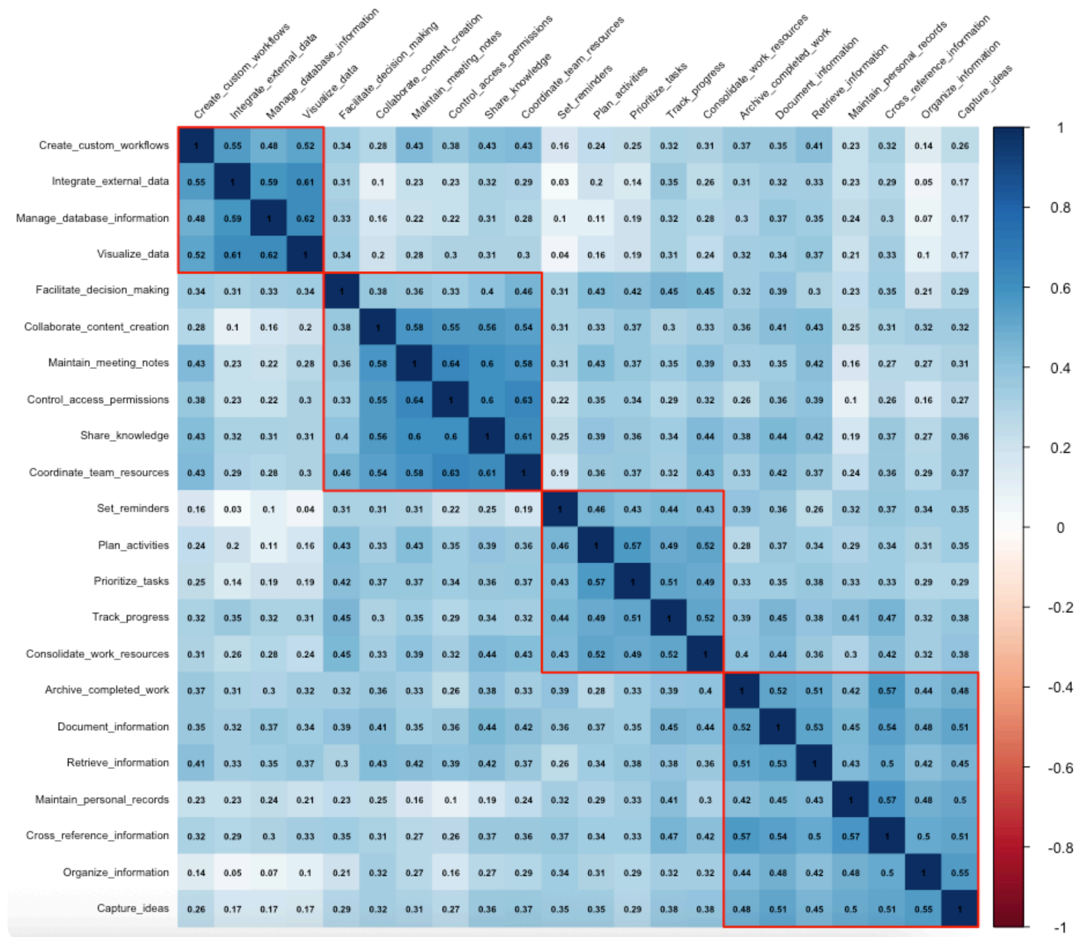
The Kaiser-Meyer-Olkin value of 0.933 indicates strong sampling adequacy, meaning our 22 JTBD variables share enough common variance to form meaningful factors. This high KMO value is particularly encouraging for product teams because it suggests that users have coherent mental models about how different platform capabilities relate to each other, rather than thinking about each job in complete isolation.

Bartlett's test of sphericity confirms that our correlation matrix contains sufficient relationships between variables to warrant factor extraction. The small p-value rejects the hypothesis that our variables are uncorrelated, meaning there are meaningful patterns in how users prioritize different capabilities that factor analysis can uncover.

Let's take a closer look at the correlation matrix. We can visualize the correlation matrix in different ways to help identify patterns to help us think about the potential factor structures that might affair during the factor analysis.

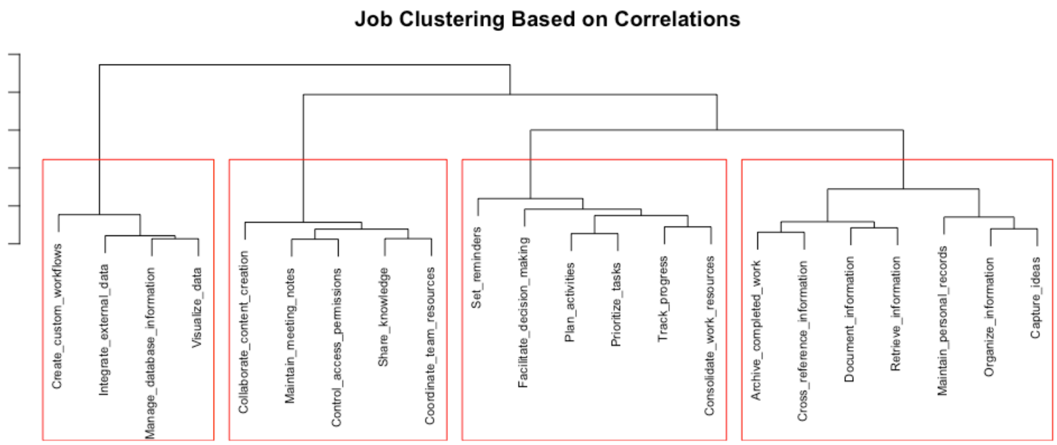
R

```r
# Load required libraries for visualization
library(corrplot)
library(ggplot2)
library(reshape2)
library(RColorBrewer)
library(igraph)

# 1. Clustered correlation heatmap
corrplot(cor_matrix,
         method = "color",
         type = "full",  # Changed from "upper"
         order = "hclust",
         addCoef.col = "black",
         tl.cex = 0.6,
         tl.col = "black",
         tl.srt = 45,
         number.cex = 0.5,
         addrect = 4,
         rect.col = "red",
         rect.lwd = 2,
         title = "Notion JTBD Correlations with Values and C

# 2. Hierarchical clustering dendrogram
job_dist <- dist(cor_matrix, method = "euclidean")
job_cluster <- hclust(job_dist, method = "ward.D2")

plot(job_cluster,
     main = "Job Clustering Based on Correlations",
     xlab = "Jobs",
     ylab = "Distance",
     cex = 0.7)
rect.hclust(job_cluster, k = 4, border = "red")
```



Correlation with 4 clusters outlined

The clustered correlation heatmap reveals four distinct blocks of highly correlated jobs, confirming our statistical assumptions about underlying structure. The information management jobs cluster together at the bottom right, showing strong intercorrelations around organizing, documenting, and cross-referencing information. The productivity workflow cluster emerges clearly around planning, tracking, and prioritizing tasks, while collaboration jobs form their own coherent block centered on sharing knowledge and coordinating team resources. Most isolated are the technical jobs, which show strong internal correlations but weaker connections to other job families.



Job Cluster Hierarchical Dendrogram based on correlations

The hierarchical clustering dendrogram provides the clearest preview of our likely factor structure. The four natural clusters shown by the red rectangles mirror our correlation analysis, with the technical jobs, collaboration jobs, productivity workflow jobs, and information management jobs each forming distinct branches. The height at which clusters merge indicates their relative similarity, with technical jobs merging at a much higher level, confirming their distinctiveness from other job families.

These visualizations validate both our statistical assumptions and our conceptual understanding of Notion's job architecture. The clear separation between job clusters, combined with moderate correlation strengths within clusters, suggests we'll extract interpretable factors corresponding to these natural groupings. The visual evidence gives us confidence that our factor analysis will reveal meaningful insights about how users mentally organize Notion's capabilities, providing a solid foundation for the factor extraction process that follows.

Let's start the factor analysis

First, we need to determine the optimal number of factors

R

```r
# Kaiser criterion (eigenvalue > 1)
eigenvalues <- eigen(cor_matrix)$values
sum(eigenvalues > 1)

# Scree plot
scree_data <- data.frame(Factor = 1:length(eigenvalues),
                         Eigenvalue = eigenvalues)

ggplot(scree_data, aes(x = Factor, y = Eigenvalue)) +
  geom_line(size = 1) +
  geom_point(size = 3) +
  geom_hline(yintercept = 1, linetype = "dashed", color = "r
  labs(title = "Scree Plot",
       subtitle = "Look for the 'elbow' where slope flattens
       x = "Factor Number", y = "Eigenvalue") +
  theme_minimal()

# Parallel analysis
pa_result <- fa.parallel(survey_data, fa = "fa", n.iter = 10
                         show.legend = FALSE, main = "Paralle
pa_result$nfact

# Variance explained
variance_explained <- cumsum(eigenvalues) / sum(eigenvalues)
variance_explained[1:6]
```

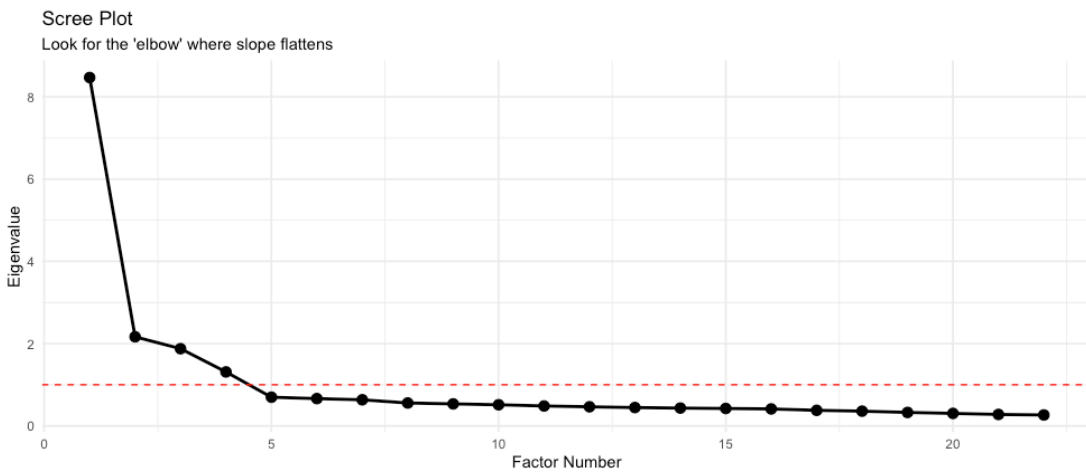
The above code highlights three different methods to determine the optimal number of factors, and all point to the same conclusion.

Eigenvalue output

R OUTPUT

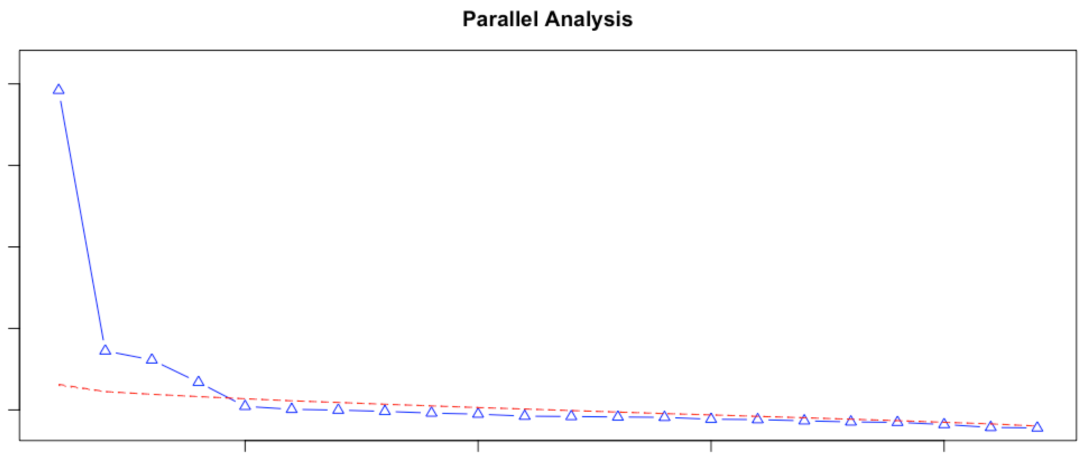
```
es <- eigen(cor_matrix)$values
> sum(eigenvalues > 1)
[1] 4
```

The Kaiser criterion examines eigenvalues, which represent **how much variance each factor explains in your dataset**. Any factor with an eigenvalue **greater than 1** is considered meaningful because it explains more variance than a single survey question would on its own. The data reveals 4 factors that meet this threshold, suggesting there are 4 distinct underlying dimensions in your survey responses.



Scree Plot

The scree plot provides a visual representation of these eigenvalues, plotting them against factor numbers to help identify the "elbow" point where the line flattens out. This elbow indicates where additional factors stop adding meaningful explanatory value to your analysis. The red dashed line at eigenvalue = 1 serves as a visual reference for the Kaiser criterion cutoff, making it easy to see which factors meet the threshold.



Parallel Analysis Plot

Parallel Analysis Output

R OUTPUT
```
<- fa.parallel(survey_data, fa = "fa", n.iter = 100,
+                          show.legend = FALSE, main = "Parallel Analysis")
Parallel analysis suggests that the number of factors =  4  and the number of
> pa_result$nfact
[1] 4
```

Parallel analysis offers another validation point by comparing the actual eigenvalues to those generated from random data with the same dimensions. This method confirms the presence of 4 meaningful factors, which strengthens confidence in this number since it agrees with the Kaiser criterion. The fact that both methods converge on the same answer suggests this is a robust finding rather than a statistical artifact.

Parallel Analysis Output

R OUTPUT
```
explained[1:6]
[1] 0.3849310 0.4833719 0.5686769 0.6283131 0.6600193 0.6901751
```

The variance explained results show the cumulative explanatory power of these factors. The first factor alone captures 38.5% of the total variance in your survey responses, while the first two factors together explain 48.3%. When you include the third factor, you're capturing 56.9% of the variance, and all four factors together account for 62.8% of the total variation in the data.

This analysis suggests the survey data reveals 4 distinct underlying "jobs" or categories of needs that customers are trying to fulfill. Each factor likely represents a different dimension of customer motivation or desired outcome. The relatively high variance explained (62.8% with 4 factors) indicates these factors capture most of the meaningful patterns in your survey responses, giving you a solid foundation for understanding the different types of jobs your customers are hiring your product or service to do.

To continue with our analysis, next we will extract and rotate the factors

```r
# Extract 4 factors using maximum likelihood with varimax r
fa_result <- fa(survey_data, nfactors = 4, rotate = "varimax
                fm = "ml", max.iter = 100)
```

Now that we know there are 4 underlying dimensions in the JTBD survey data, this code extracts those factors and makes them easier to interpret.

The fa() function performs the factor extraction using several key specifications. We are telling it to extract exactly 4 factors based on the previous analysis, ensuring consistency between the exploratory work and the final factor solution. The method uses maximum likelihood estimation (fm = "ml"), which is considered one of the most robust approaches for factor extraction because it provides better statistical properties and allows for significance testing of the factor loadings.

The varimax rotation (rotate = "varimax") is helpful for interpretation. Without rotation, factors can be difficult to interpret because variables often load substantially on multiple factors. Varimax rotation mathematically transforms the factors to maximize the variance of the squared loadings, which essentially means it tries to make each variable load highly on one factor and minimally on others. This creates cleaner, more interpretable factor patterns where each survey question is primarily associated with one underlying job or need.

The max.iter = 100 parameter sets the maximum number of iterations the algorithm can use to find the optimal solution, ensuring the process has enough computational cycles to converge on stable factor loadings.

Essentially this step transforms the survey data into 4 distinct, interpretable factors that represent different categories of customer jobs. Each factor will show which survey questions cluster together, helping you understand what needs or motivations define each job category. The varimax rotation ensures these job categories are as distinct and non-overlapping as possible, making it easier to develop targeted solutions for each type of customer need.

Now let's examine the factors

R

```
print(fa_result$loadings, cutoff = 0.3, sort = TRUE)
```

Factor Analysis Results

R OUTPUT

```
LOADINGS ===
Loadings:
                           ML1    ML3    ML2    ML4
Organize_information          0.693
Document_information          0.573
Capture_ideas                 0.655
Retrieve_information          0.514  0.329
Maintain_personal_records     0.655
Archive_completed_work        0.584
Cross_reference_information   0.672
Share_knowledge                      0.674
Collaborate_content_creation         0.653
Coordinate_team_resources            0.676
Maintain_meeting_notes               0.718
Control_access_permissions           0.768
Manage_database_information                 0.729
Create_custom_workflows              0.343  0.591
Visualize_data                              0.748
Integrate_external_data                     0.780
Plan_activities                                    0.679
Track_progress                0.323                0.598
Prioritize_tasks                                   0.639
Consolidate_work_resources                         0.571
Set_reminders                 0.345                0.540
Facilitate_decision_making                         0.453

               ML1   ML3   ML2   ML4

SS loadings 3.391 3.253 2.763 2.646
Proportion Var 0.154 0.148 0.126 0.120
Cumulative Var 0.154 0.302 0.428 0.548

>
```

This output shows the factor loadings matrix, which is the heart of the factor analysis results. The loadings represent correlation coefficients between each job (survey question) and each of the four factors, with only values above 0.3 displayed to focus on the strongest relationships.

The structure reveals four distinct job clusters. Factor ML1 captures personal information management jobs, with strong loadings from tasks like organizing information (0.693), capturing ideas (0.655), and maintaining personal records (0.655). These jobs all relate to individual knowledge management and personal productivity needs.

Factor ML3 represents team collaboration and coordination, showing high loadings for sharing knowledge (0.674), coordinating team resources (0.676), maintaining meeting notes (0.718), and controlling access permissions (0.768). This factor clearly captures jobs related to working with others and managing shared resources.

Factor ML2 focuses on technical data management, with the strongest loadings coming from managing database information (0.729), visualizing data (0.748), and integrating external data (0.780). These jobs involve more sophisticated data handling and technical integration tasks.

Factor ML4 encompasses planning and task management activities, including planning activities (0.679), prioritizing tasks (0.639), and tracking progress (0.598). This represents project management and workflow optimization needs.

The summary statistics at the bottom show each factor's explanatory power. Factor ML1 has the highest sum of squared loadings (3.391) and explains 15.4% of the total variance. The factors collectively explain 54.8% of the variance in your survey data, which represents a solid factor solution that captures most of the meaningful patterns in customer job preferences.

R

```r
# Create comprehensive loadings table
loadings_df <- data.frame(
  Job = rownames(fa_result$loadings),
  Factor1 = round(fa_result$loadings[,1], 3),
  Factor2 = round(fa_result$loadings[,2], 3),
  Factor3 = round(fa_result$loadings[,3], 3),
  Factor4 = round(fa_result$loadings[,4], 3),
  Communality = round(fa_result$communalities, 3)
)

# Identify primary factor for each job
loadings_df$Primary_Factor <- apply(abs(fa_result$loadings),
loadings_df$Strength <- apply(abs(fa_result$loadings), 1, ma

# Show complete loadings table
loadings_df
```

Factor Loadings Table

R OUTPUT

```
                    Factor2 Factor3 Factor4 Communality Primary_Factor  Strength
Organize_information                    Organize_information   0.693   0.162  -0.
Document_information                    Document_information   0.573   0.278   0.
Capture_ideas                                  Capture_ideas   0.655   0.223   0.
Retrieve_information                    Retrieve_information   0.514   0.329   0.
Share_knowledge                              Share_knowledge   0.212   0.674   0.
Maintain_personal_records          Maintain_personal_records   0.655  -0.018   0.
Archive_completed_work                Archive_completed_work   0.584   0.205   0.
Plan_activities                              Plan_activities   0.190   0.267   0.
Track_progress                                Track_progress   0.323   0.124   0.
Collaborate_content_creation Collaborate_content_creation   0.278   0.653   0.
Manage_database_information      Manage_database_information   0.146   0.106   0.
Create_custom_workflows              Create_custom_workflows   0.157   0.343   0.
Coordinate_team_resources          Coordinate_team_resources   0.203   0.676   0.
Prioritize_tasks                            Prioritize_tasks   0.205   0.255   0.
Facilitate_decision_making        Facilitate_decision_making   0.161   0.294   0.
Consolidate_work_resources        Consolidate_work_resources   0.280   0.255   0.
Visualize_data                                Visualize_data   0.137   0.171   0.
Maintain_meeting_notes                Maintain_meeting_notes   0.135   0.718   0.
Integrate_external_data              Integrate_external_data   0.100   0.091   0.
Cross_reference_information      Cross_reference_information   0.672   0.141   0.
Set_reminders                                  Set_reminders   0.345   0.127  -0.
Control_access_permissions        Control_access_permissions   0.072   0.768   0.

>
```

This code creates an analysis table that transforms the raw factor loadings into a more practical format for understanding customer jobs. The data frame structures all the key information in one place, making it easier to interpret and act upon the factor analysis results.

The table reveals some different insights about how well each job is captured by the four-factor model. The Factor columns show the exact loading values for each job across all four factors, while the Communality column indicates how much of each job's variance is explained by the four-factor solution. Most communalities range from 0.4 to 0.7, suggesting the model captures a substantial portion of what drives customer preferences for these jobs. Values closer to 1.0 indicate better representation, while values below 0.3 might suggest the job doesn't fit well into any of the four categories.

The Primary_Factor and Strength columns provide immediate clarity about which factor dominates each job and how strong that relationship is. The code uses the `which.max()` function to automatically identify the factor with the highest absolute loading for each job, eliminating guesswork. For instance, "Integrate_external_data" has a strength of 0.780 and belongs primarily to Factor 3, making it a core defining job for the technical data management category. Similarly, "Control_access_permissions" shows a strength of 0.768 and belongs to Factor 2, indicating it's central to the collaboration factor.

Some jobs show interesting cross-factor relationships that merit attention. "Track_progress" loads on Factor 4 (0.598) but also has a notable loading on Factor 1 (0.323), suggesting it bridges personal productivity and project management needs. This kind of overlap indicates potential integration opportunities or explains why some customers might struggle to categorize certain needs cleanly. Jobs with secondary loadings (above 0.3) on multiple factors may represent complex needs that span categories.

The strength values also reveal hierarchy within each factor. In Factor 1 (personal information management), jobs like "Organize_information" (0.693) and "Cross_reference_information" (0.672) are much stronger than "Retrieve_information" (0.514), suggesting core versus supporting jobs within each category. This hierarchy can inform feature prioritization, with higher-strength jobs potentially deserving more development resources or prominence in the user interface.

The negative loadings, such as the -0.068 for "Organize_information" on Factor 3, indicate weak negative correlations. While not practically significant at such low values, larger negative loadings would suggest jobs that work in opposition to a particular factor. This comprehensive view helps analysts understand not just what belongs together, but also what distinctly separates different customer job categories.

For business applications, this analysis provides understanding customer job complexity. The communality scores help identify which jobs are well-understood by the current framework and which might need additional research. Jobs with lower communalities like "Facilitate_decision_making" (0.406) suggest there may be additional factors or nuances not captured in the four-factor model, pointing toward areas where customer needs might be more complex or context-dependent than initially apparent.

Next, let's evaluate the models fit.

Evaluating Model Fit

R OUTPUT

```
$RMSEA[1]

  RMSEA

0.0126198

> fa_result$TLI
> [1] 0.996014

> # Variance explained by the 4-factor solution
>
> if(!is.null(fa_result$Vaccounted)) {

- fa_result$Vaccounted[3, 4] # Cumulative proportion
- }
[1] 0.5478221
>
```

This code evaluates how well the four-factor model fits the actual survey data using several statistical measures that assess model quality. Model fit evaluation is crucial because it tells analysts whether the factor solution accurately represents the underlying structure in the data or if adjustments are needed.

The Root Mean Square Error of Approximation (RMSEA) value of 0.0127 indicates excellent model fit. RMSEA measures how well the model approximates the population, with values below 0.05 considered good fit and values below 0.08 acceptable. The extremely low RMSEA here suggests the four-factor model closely matches the actual correlation patterns in the data, with minimal approximation error.

The Tucker-Lewis Index (TLI) of 0.996 also demonstrates high model fit in this case. TLI compares the proposed model to a baseline model where all variables are uncorrelated, with values above 0.95 indicating excellent fit and values above 0.90 considered acceptable. A TLI of nearly 1.0 means the four-factor model explains the correlations among variables almost perfectly compared to assuming no relationships exist.

The cumulative proportion of variance explained is 0.548, meaning the four factors together account for approximately 54.8% of the total variance in the survey responses. This aligns with the earlier eigenvalue analysis and represents a substantial portion of customer job preferences being captured by these four underlying dimensions. While this leaves about 45% unexplained, this is typical for behavioral data where individual differences and measurement error contribute to variance.

Together, these fit indices provide strong statistical evidence that the four-factor solution is appropriate for the data. The low RMSEA and high TLI suggest the model structure is sound, while the variance explained indicates practical significance. For business applications, this means confidence that the four job categories identified genuinely reflect how customers think about and prioritize different types of work, rather than being statistical artifacts.

Factor Analysis Conclusion

To conclude, factor analysis from the simulated data reveals that Notion users naturally organize the platform's 22 functional capabilities into four distinct job categories, each representing a different approach to knowledge work. The clear factor structure suggests users have coherent mental models about how different capabilities relate to their workflows, rather than viewing each feature in isolation.

The analysis provides several implications for product teams. The hierarchy within factors indicates which jobs are core versus supporting within each category. For instance, organizing information and cross-referencing emerge as central to personal productivity, while data visualization and external integration define technical users. The moderate correlations between factors suggest opportunities for cross-selling users from basic personal productivity tools into more advanced collaboration or technical features.

The four-factor structure confirms that users prioritize jobs in patterns that match how they actually work.

This type of analysis becomes particularly valuable for platform products like Notion, where understanding user segments based on job complexity rather than demographics enables more targeted feature development and user experience optimization across different levels of technical sophistication and collaboration needs.

Key Takeaways

Start with what you already know. Before launching elaborate research studies, examine your existing data sources and talk directly with customers. Most teams can reach a high level of confidence about their core functional jobs through usage analytics, support tickets, customer feedback, and targeted stakeholder conversations. Quantitative validation often serves more to convince stakeholders than to uncover genuinely surprising insights.

Choose your research approach based on what you're trying to accomplish. If you're optimizing existing products, define jobs at a level matching your current capabilities and focus on friction points within that scope. If you're exploring new opportunities, consider broader job definitions that reveal adjacent markets and different approaches to serving customer needs.

Match your abstraction level to your audience. Different stakeholders need different flight levels to make decisions and take action. Executives benefit from higher-level perspectives that reveal market opportunities, while product teams need tactical specificity for feature development. Your job definition should serve the people who will use your research insights.

Use quantification methods strategically. MaxDiff analysis answers "which jobs matter most" and works well for roadmap prioritization. Factor analysis answers "how do jobs relate to each other" and helps identify user segments and natural product bundles. Choose the method that addresses your research questions rather than defaulting to what seems most sophisticated.

Keep the customer's perspective central. Your internal product categories and organizational structure shouldn't drive how you define jobs. Ground your job definitions in how customers actually think about and experience the work they're trying to accomplish. Their mental model of the complete job should guide your abstraction level and research boundaries.

Expect complexity and plan for it. Multi-purpose platforms, ecosystems, and products with broad capabilities will naturally serve multiple legitimate jobs for different user segments. This complexity requires systematic prioritization rather than oversimplification. Use the three-step approach of analyzing existing data, conducting targeted interviews, and selectively quantifying when you need stakeholder buy-in or want to understand user segments.

With a clearly defined core functional job that aligns with your research objectives and stakeholder needs, you're ready to move into the discovery phase of JTBD research. The job definition you've established will guide everything from participant recruitment to interview questions to analysis frameworks. This upfront investment in clarity pays dividends throughout the rest of your research process by ensuring every subsequent decision serves your ultimate goal of understanding what customers are really trying to accomplish.

Chapter Four Exercises

Linked here is a transcript of an interview with a YouTube Creator. Read through the interview transcript and see what different core functional jobs you notice throughout the interview.

References

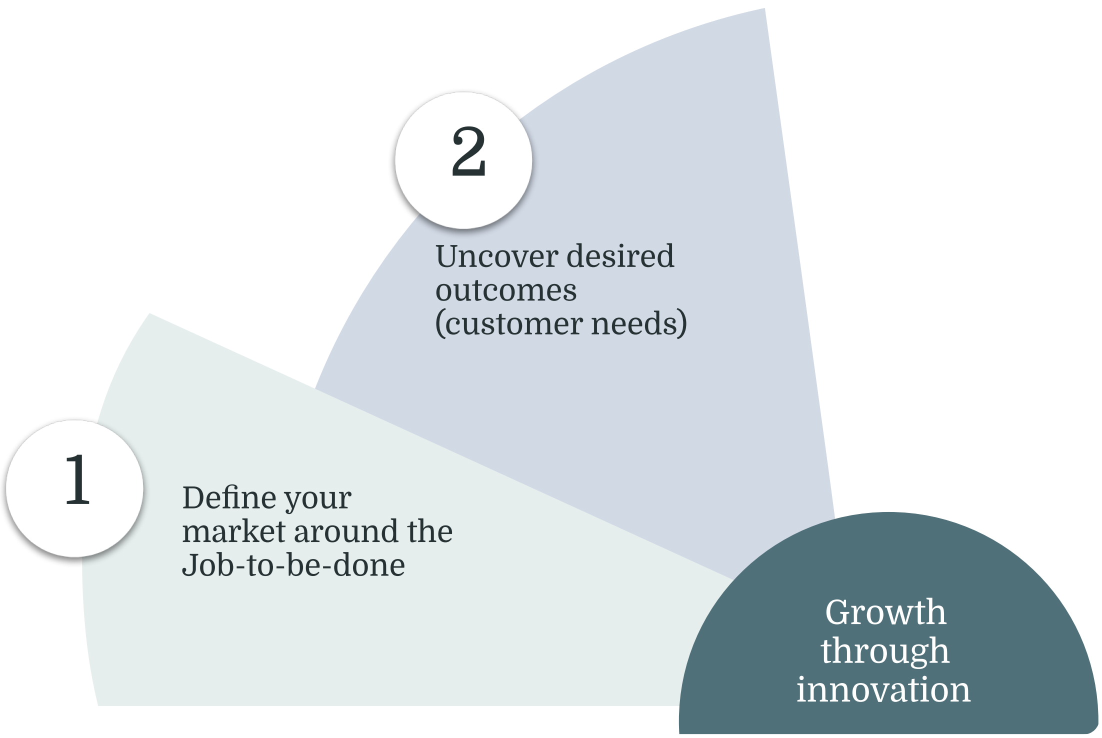
[26]: Kalbach, Jim. *The Jobs To Be Done Playbook: Align Your Markets, Organization, and Strategy Around Customer Needs*. 1st ed., Two Waves Books, 2020. ISBN-13: 978-1933820682. Available on Amazon.

[27] Bezos, J. (2022). Jeff Bezos: Amazon and Blue Origin | Lex Fridman Podcast #405 [Transcript]. Lex Fridman. Retrieved from https://lexfridman.com/jeff-bezos-transcript/

[28] Ulwick, A. W. (2016). Jobs to Be Done: Theory to Practice. Paperback, October 28, 2016. Retrieved from https://www.amazon.com/Jobs-be-Done-Theory-Practice/dp/B0CCZV8JJH/

[29] Hair, J. F. Jr., Black, W. C., Babin, B. J., & Anderson, R. E. (2010). Multivariate Data Analysis (7th ed.). Retrieved from https://www.amazon.com/Multivariate-Data-Analysis-Joseph-Hair/dp/0138132631

Section 3 Overview



Step 2: Uncover the desired outcomes

You have already defined who your customer is and what main goal they are trying to accomplish. Now, this section breaks that goal down into smaller pieces. Instead of looking at the job as one big task, you will learn to see the specific steps customers take and how they measure success. This process helps you find exactly where current solutions fail so you can build something better.

Here is what these two chapters cover:

Chapter 5: The Job Map

This chapter introduces a tool called the Job Map. This is a step-by-step guide that lists everything a customer must do to finish a job. It does not focus on your specific product. Instead, it focuses on the customer's process.

- **The Framework:** You will learn an eight-step standard to ensure you do not miss any part of the customer's workflow.
- **The Application:** You will see how to use this map to find spots where competitors are weak or where you can enter new markets.

Chapter 6: Uncovering Outcomes and Needs

Once you have the map, this chapter teaches you how to define what "success" looks like for the customer at each step. You will learn to write specific statements that describe exactly what the customer needs. This chapter looks at the full picture of customer satisfaction, including:

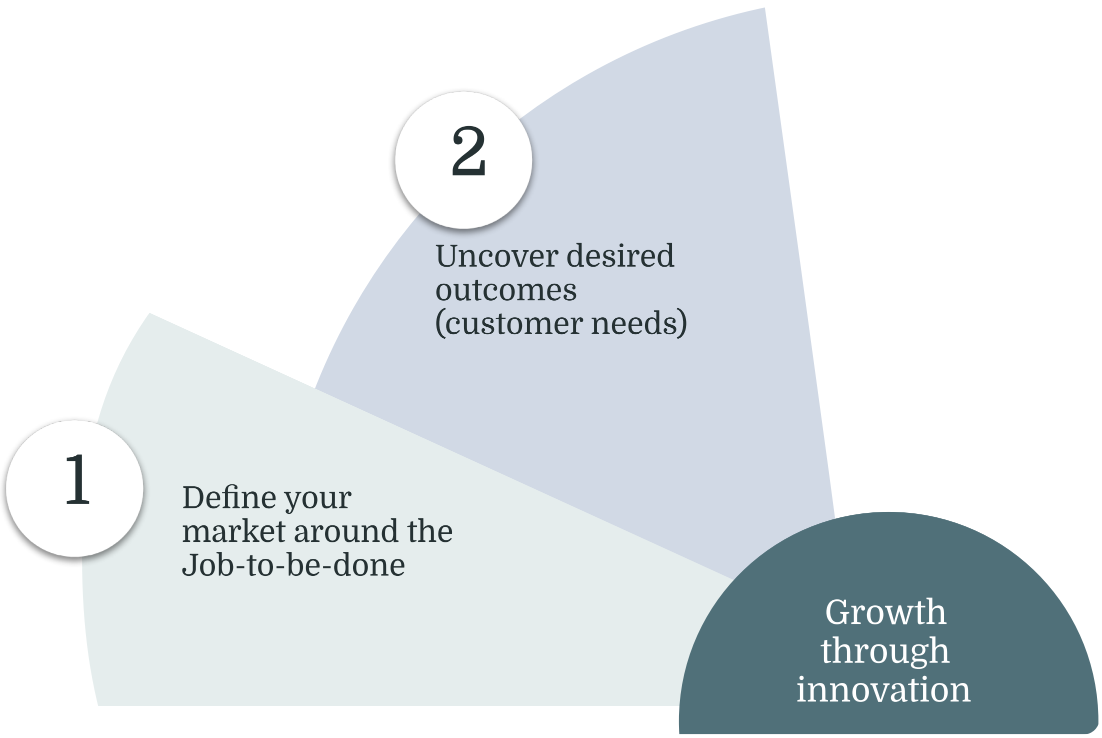
- **Emotional Needs:** How the customer wants to feel.
- **Consumption Chain Jobs:** The effort required to use a product (like buying, cleaning, or fixing it).
- **Financial Needs:** The desire to save money or time.
- **Complexity Factors:** The specific situations or barriers that make the job hard for certain people.

By the end of this section, you will have a complete list of specific customer needs. This gives you the data you need to pinpoint where your customers are struggling and where your business can identify opportunities to grow.

UNCOVER DESIRED OUTCOMES

Chapter 5: The Job Map

The solution agnostic journey for your core functional job



Step 2: Deconstruct the Job before creating a list of outcomes/needs

With the core job defined, the next step is deconstructing it. The Job Map helps you understand the different steps the main job executor is taking in order to get the job done.

Job Map Introduction

Generic Job Map Template

After you have determined the core functional job that will serve as the foundation for your JTBD research, the next part of step two in the ODI process is to build out the job map.

The Job Map breaks the core functional job into a series of solution-agnostic steps that represent the ideal path to success. I like to think of it as a solution agnostic customer journey map. The major difference is that a Customer Journey Map outlines the current process in how customers are currently using your product or service. It outlines transaction moments, pain points within the journey, and what the customer is actively doing.

Job Map Fundamentals

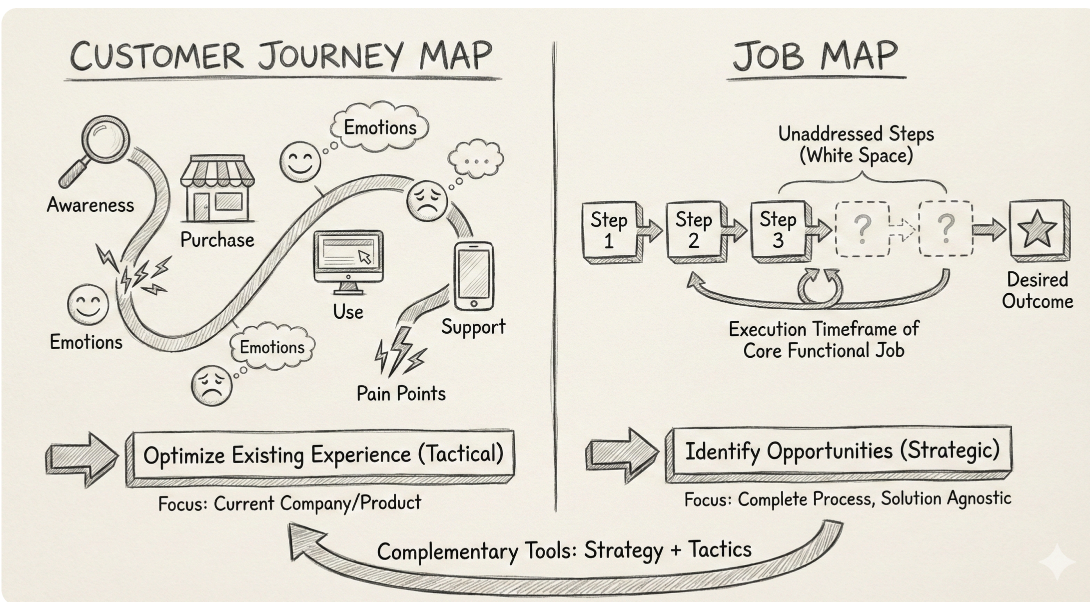
The Job Map differs by focusing more on the goals a user is trying to achieve in order to get their core functional job done. The job map follows a few key principles.

- Generalized – the job steps must be relevant to anyone doing the job.
- Ideal – all job steps must be in the optimal order for execution.
- Functional – all job steps of the core job must be functional, not emotional.

- Format follows the rules for core functional jobs (solution agnostic, format, no jargon, etc.).
- Active – all job steps state what the customer tries to accomplish in each job step.
- Completeness– the job map should cover all steps of the Universal job map; typically 10-20 steps

Tony Ulwick says, "A job map does not show what the customer is doing (a solution view); rather, it describes what the customer is trying to get done (a needs view)". [30]

Job Map vs. Customer Journey Map



Difference Between a Job Map and Customer Journey Map

The distinction between Job Maps and customer journey maps reflects a distinct difference in perspective and purpose that impacts focus on product improvements or overall strategy. Customer journey maps typically focus on documenting the current experience customers have with a specific company, product, or service, mapping touch-points, emotions, and pain points within the existing relationship. While valuable for improving current customer experience, this approach limits the scope of insight to incremental improvements within the current solution paradigm.

Job Maps start differently by focusing on the job itself rather than any particular solution. This job-centric perspective reveals the complete process customers need to execute, including steps that current solutions **may not address at all or may address poorly**. By maintaining solution agnosticism, Job Maps expose white space opportunities and highlight fundamental job steps that competitors may be ignoring or underserving.

The scope also differs notably between these two tools. Customer journey maps typically capture the experience from initial awareness through purchase and ongoing use of a specific solution, reflecting the linear progression through a company's sales and service processes. Job Maps, in contrast, focus on the execution timeframe of the core functional job itself, which may be much shorter or longer than any individual customer journey and may involve multiple solutions or no solutions at all.

This difference in scope and perspective makes Job Maps helpful for identifying a different perspective on opportunities, while customer journey maps excel at optimizing existing customer relationships and experiences. Organizations benefit most from using both tools in **complementary ways**, with Job Maps informing strategic decisions and customer journey maps guiding tactical experience improvements.

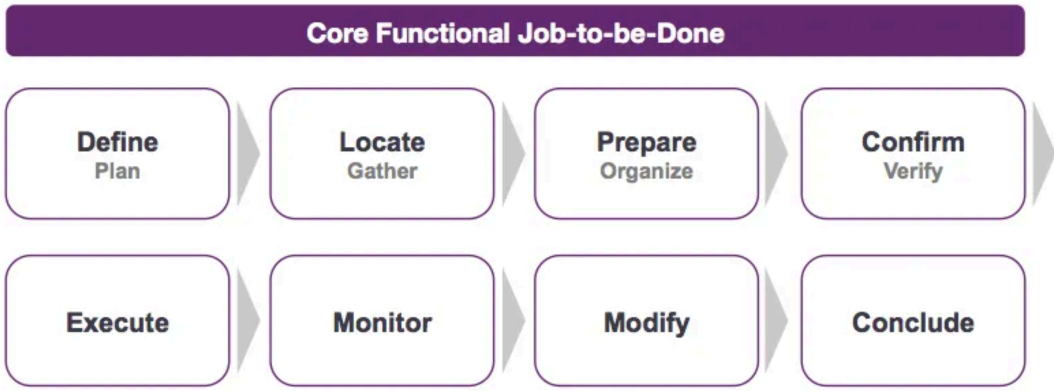
How to Build a Job Map

Step 1: Starting with your Core Functional Job

See Chapter 4, Identifying the Core Job, to help determine what the core functional job of your product or service is. Once you have that, you can start building out your job map.

To provide an example of creating a job map throughout this chapter, we will use the core job, "Communicate information clearly to an audience for understanding."

Step 2: Identifying the Job Steps



Generic Job Map Template

The next step is to identify the job steps for a core functional job. It is recommended to follow the generic 8 steps job map template like the above image highlights of the 8 steps for a core functional job.[28]

The 8 steps are listed below, along with questions you can ask participants during interviews:

1. **Define**

- Where/when does the job begin? The job begins when there's a need to share specific information with others for a purpose.
- What must be defined up front? The communication objective, target audience characteristics, key message, desired outcome, and success criteria.

2. **Locate**

- What must be located, gathered or retrieved? Relevant data, supporting evidence, credible sources, audience contact information, appropriate communication channels, and any existing materials or templates.

3. **Prepare**

- What must be organized, examined, or set up? Structure the information logically, choose the right format/medium, prepare visual aids if needed, schedule timing, and set up the delivery environment or platform.

4. **Confirm**

- What must be validated before execution? Verify information accuracy, confirm audience availability, test technical setup, review message clarity, and ensure all materials are ready.

5. **Execute**

- What's at the heart of getting the job done? Deliver the core message using chosen communication method, engage the audience appropriately, and present information in a clear, structured manner.

> ***Authors Note****: The Execute step is not the entire job. It is the moment the user engages in the core action. For example, in 'dental surgery,' the job includes preparation and recovery, but the 'Execute' step is the surgery itself.*

6. **Monitor**

- What must be monitored during execution? Audience comprehension signals (questions, body language, engagement), technical performance, time management, and message reception.

7. **Modify**

- What adjustments might be needed? Clarify confusing points, adjust pace or complexity, address unexpected questions, resolve technical issues, or adapt to audience feedback in real-time.

8. **Conclude**

- What must be done to successfully conclude? Summarize key points, confirm understanding, provide follow-up resources, establish next steps, and document outcomes or lessons learned.

These are the recommended steps to start with when creating a job map. From here you can branch off into more granular or less granular steps depending on how detailed you want your job map to be.

Ok, let's imagine again we are using the core functional job of Communicating information clearly to an audience for understanding. Let's review the transcript below of asking the questions for each job map step and see how we can construct a job map based on the responses from the participant.

Example Interview

Interviewer: Let's start with the heart of what you're trying to accomplish. When you need to communicate important information to your team, **what's the most central task or the core of what you are trying to get done?**

Marketing Manager: Well, it's really about **delivering the message in a way they can act on it** [Execute]. I'm usually presenting findings from our customer research or explaining a new campaign strategy.

Interviewer: Got it. Now let's back up, **where does this job actually begin for you? What triggers the need to communicate?**

> ***Authors Note****: Notice in the interview that we don't start the conversation by asking about the beginning of the job (the Define step). Instead, we start by asking about the Execute step—the heart of the action. It is often easier for customers to describe the main action first (e.g., 'delivering the presentation'). Once they are grounded in that moment, it becomes easier for them to work backward and answer questions like, 'What did you have to plan or define before you reached that point?*

Marketing Manager: It usually starts when I have **new insights that will change how the team approaches their work** [Define]. Like last week, our user research revealed customers were confused about our pricing tiers.

Interviewer: What do you need to figure out or define before you can move forward?

Marketing Manager: I need to be clear on **what action I want them to take afterward** [Define]. Am I asking them to change existing campaigns? Create new materials? Just be aware for future work? That shapes everything else.

Interviewer: That's the "Define" step emerging. **What do you need to gather or locate to make this work?**

Marketing Manager: I pull together **the actual research data, maybe some customer quotes, competitive examples** [Locate]. I also need to **check everyone's calendars** [Locate] and figure out if this needs to be email, Slack, or a proper meeting.

Interviewer: Before you actually deliver the information, **what do you need to prepare or set up?**

Marketing Manager: I usually **create an outline or slides if it's complex** [Prepare]. For the pricing thing, I made a simple visual showing the customer confusion points. I also **book a conference room** [Prepare] and make sure the projector works -

learned that the hard way!

Interviewer: What do you double-check or confirm before you actually start communicating?

Marketing Manager: I **rehearse key points to myself** [Confirm], especially if there might be pushback. I also **verify my facts** [Confirm] - nothing kills credibility like getting basic numbers wrong.

Interviewer: During the actual communication, **what are you monitoring or watching for?**

Marketing Manager: Facial expressions are huge [Monitor] - I can see when people are confused or skeptical. I watch for **when people start checking phones** [Monitor] - that means I'm losing them. I also listen for **the types of questions** [Monitor] they ask.

Interviewer: When do you need to modify or adjust your approach?

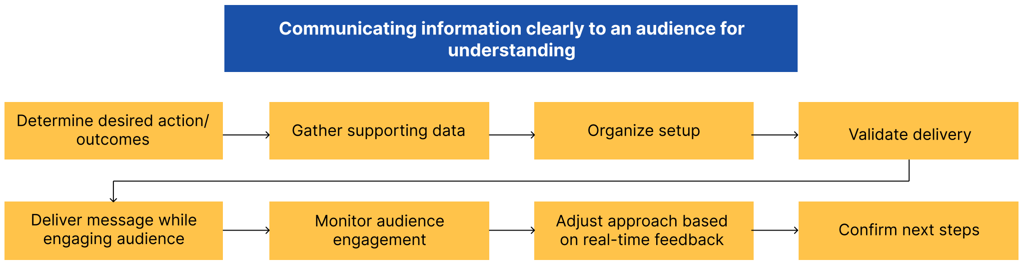
Marketing Manager: If I see confusion, I **slow down and give more examples** [Modify]. If they seem skeptical, I **pull out additional data** [Modify] or ask them to share their concerns. Sometimes I realize mid-presentation that **I should have started with the business impact** [Modify] rather than the research details.

Interviewer: How do you know when the job is successfully completed?

Marketing Manager: When **people start talking about next steps without me prompting them** [Conclude]. Like, "So should we pause the current email campaign?" or "I can update those landing pages by Friday." That tells me they understood and are ready to act.

Interviewer: What do you do to wrap things up?

Marketing Manager: I always **send a follow-up email with the key points and any agreed actions** [Conclude]. I also **set calendar reminders to check in** [Conclude] on whether changes actually happened.



Communicating information clearly to an audience for understanding job map example

Key Job Steps Revealed:

1. **Define:** Determine desired action/outcome
2. **Locate:** Gather supporting data
3. **Prepare:** Organize setup
4. **Confirm:** Validate delivery
5. **Execute:** Deliver message while engaging audience
6. **Monitor:** Monitor audience engagement
7. **Modify:** Adjust approach based on real-time feedback
8. **Conclude:** Confirm next steps

Notice how the generic 8-step template guided the questioning, but the job steps emerged naturally from the conversation rather than being imposed.

The 8 Steps Are Your Starting Point, Not Your Limit

The 8 generic steps serve as guardrails, not rigid requirements. Depending on your core job's complexity, you might end up with **8 - 15 steps**. A simple consumer task might compress into fewer steps, while complex B2B processes or manufacturing operations often require 12-15 detailed steps to capture the full reality.

Use the 8 steps as a starting point to ensure you don't miss major phases, then expand where your customer interviews reveal additional complexity.

Taking Your Interview Insights and Creating Order

Let's return to our communication example and expand it beyond the basic 8 steps. After conducting several interviews, you might have gathered insights like:

- I need to know what action I want them to take
- I have to figure out if this is urgent or can wait
- I check everyone's calendars
- I need to gather the right supporting evidence
- I create different materials for different audiences
- I rehearse key points to myself
- I watch facial expressions during the presentation
- I adjust my approach if people seem confused
- I send a follow-up email afterward
- I check back later to see if action was actually taken

Your first task is to **map these insights using the 8 core steps as your foundation**, then expand where the complexity demands it:

Define Phase:

1. **Assess Urgency** → Determine timing and priority level
2. **Define Outcome** → Clarify desired action and success criteria

3. **Identify Audience** → Understand who needs this information

Locate Phase:

4. **Gather Evidence** → Collect supporting data and examples
5. **Check Availability** → Confirm audience schedules and accessibility
6. **Select Channels** → Choose appropriate communication methods

Prepare Phase:

7. **Create Materials** → Develop audience-appropriate content
8. **Setup Environment** → Book space and test technology

Confirm Phase:

9. **Validate Content** → Verify facts and rehearse delivery

Execute Phase:

10. **Deliver Message** → Present information while engaging audience

Monitor Phase:

11. **Track Reception** → Watch for comprehension and engagement signals

Modify Phase:

12. **Adjust Approach** → Adapt based on real-time feedback

Conclude Phase:

13. **Secure Commitment** → Confirm understanding and next steps
14. **Follow Up** → Check that promised actions actually occur

Ensuring Complete Coverage

Look for gaps in your expanded framework. If you have detailed steps for preparation but nothing about follow-up, go back to your interviewees and ask: *"After the communication is over, what do you do to ensure it was successful?"*

Watch for phases that seem too sparse. Complex jobs often have multiple sub-steps within each major phase. A manufacturing process might have 3-4 steps just within the "Confirm" phase.

Sequencing the Steps Properly

While the 8 generic steps provide a natural flow, **pay attention to the actual sequence your customers follow.** Sometimes they'll say things like:

"Actually, I usually start gathering data before I'm even clear on what I want to accomplish."

This suggests their job might actually start with "Gather Evidence" activities that help them "Define Outcome." **Don't force their reality into the template—adapt the template to reflect how the job really gets done.**

Creating a Usable Format

Transform your organized insights into a clear job map format:

Job: Communicate information clearly to an audience for understanding

1. **Assess Urgency** → Determine timing and priority level
2. **Define Outcome** → Clarify desired action and success criteria
3. **Identify Audience** → Understand who needs this information
4. **Gather Evidence** → Collect supporting data and examples
5. **Check Availability** → Confirm audience schedules and accessibility
6. **Select Channels** → Choose appropriate communication methods
7. **Create Materials** → Develop audience-appropriate content

8. **Setup Environment** → Book space and test technology
9. **Validate Content** → Verify facts and rehearse delivery
10. **Deliver Message** → Present information while engaging audience
11. **Track Reception** → Watch for comprehension and engagement signals
12. **Adjust Approach** → Adapt based on real-time feedback
13. **Secure Commitment** → Confirm understanding and next steps
14. **Follow Up** → Check that promised actions actually occur

Each step should have a clear trigger (what prompts this step) **and outcome** (what's accomplished when it's complete). This creates a logical flow that others can follow and validate.

Remember: The right number of steps is however many it takes to accurately capture your customers' reality. Simple jobs might compress to 8 steps, complex industrial processes might expand to 18. The 8-step framework ensures you don't miss any major phases along the way.

Identifying the Right Granularity

One of the most common challenges in job mapping is finding the right level of detail. Too granular, and your job map becomes unwieldy and loses strategic value. Too high-level, and you miss opportunities. The key is finding the "Goldilocks zone" where each step reveals actionable insights.

The Granularity Test: Can You Innovate Here?

Ask yourself whether a product, service, or solution could make this step noticeably better, faster, or easier. If the answer is yes, you're at the right level of granularity. If not, you may need to zoom in or zoom out.

"Click the send button" represents too much granularity because this micro-action offers little opportunity for meaningful innovation. "Communicate the message" is too broad because it encompasses too much and obscures pain points. "Deliver message while engaging audience" captures the right level because it represents a meaningful job step with clear opportunities for improvement through presentation tools, engagement techniques, and feedback mechanisms.

Consider Your Scope

The right granularity depends on what you're trying to innovate. If you're building communication software, you might need more granular steps around content creation and delivery. If you're designing organizational processes, you might focus on higher-level coordination steps.

From a software company perspective, you might focus on steps like creating slides, formatting content for mobile, and tracking message opens because you can build better tools for these activities. From an organizational consultant perspective, you might emphasize aligning stakeholders on messages and cascading information through hierarchy because you can improve these processes.

Test with the "So What?" Question

For each step, ask what could go wrong and what would make this noticeability better. If you can't generate compelling answers, the step might be too granular, too broad, or not part of the core job.

Practical Granularity Guidelines

Right-sized steps typically share several characteristics:

- Have a clear beginning and end
- Involve a decision or create an output
- Can fail or succeed independently
- Present opportunities for measurement
- Could benefit from tools, services, or improvements

Warning signs of wrong granularity include steps that are sub-tasks of other steps, steps with no clear success criteria, steps that never fail or cause problems, steps that require no decisions or judgment, and steps that offer no room for innovation.

Remember that you can always adjust granularity later. Start with what feels natural from your interviews, then refine based on where you see the most potential and customer frustration.

Method of Validation

There are several methods for validating your job map, and the approach you choose depends on your familiarity with the core functional job and the domain expertise required.

For jobs you've personally experienced many times, you might feel confident validating your job map with just **2-4 interviews**. Take our communication example around presenting information to an audience. If this represents a core job you've executed repeatedly throughout your career, you can often validate the steps based on your own experience combined with minimal external input.

However, when dealing with core jobs that require industry knowledge or domain expertise in fields like manufacturing, medicine, or law, expert feedback becomes essential. These specialized domains have nuances and steps that only subject matter experts (SMEs) would recognize. These might require more than just 2-4 interviews.

The Secondary Research Approach

I always recommend teams conduct secondary research on a core job and create a hypothesized job map before validating with users. If you're working in healthcare and evaluating a core job around using medical devices, start by reading medical reports, research articles, and reviewing transcripts related to that job. This preparation creates a stronger foundation for your validation interviews.

The Validation Interview Process

After developing your hypothesized job map, conduct 3-5 follow-up interviews with the main job executors. Present your job map and ask targeted questions about its accuracy. Are there missing steps they would add? Do they perceive any steps as confusing or unnecessary? Would they sequence anything differently?

These validation interviews serve a dual purpose. Beyond confirming your job map's accuracy, they provide deeper insights into the different outcomes within each job step and the complexity factors that influence execution. See chapter 6 for more details on outcomes and complexity factors This additional layer of understanding matters for identifying the most impactful opportunities.

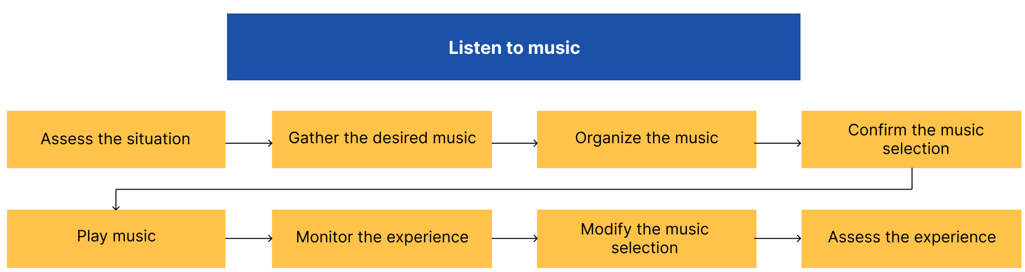
The goal isn't perfection but confidence that your job map reflects the real experience of people executing this job in the field.

Using the Job Map for competitive positioning

One of my favorite ways to use a job map is to evaluate where your company's portfolio of solutions or a specific solution lies vs. your competitors.

To illustrate this, let's look at a crowded market like the music streaming service market. Companies like Spotify, Apple Music, YouTube Music, Tidal, and many more all exist.

Let's imagine we are part of the consumer strategy group looking at how we could expand the core Spotify solution into other opportunities. To get a sense of where things currently are in the market, let's create the job map for Spotify's core functional job, "Listen to Music".



Listen to Music Job Map Example

As we can see with the above job map, it's quite clear that Spotify, and really any music streaming service for that matter adequately addresses each job step. This means the current core job offers few opportunities for major improvements.

Like we discussed in chapter 4 about flight levels, let's evaluate what a job map might look like for a higher flight level core job. A trick I like to do for this is to go back and look at the core mission of a company to get a sense of what their ultimate goal is. Let's look at Spotify's mission statement.

> *Our mission is to unlock the potential of human creativity—by giving a million creative artists the opportunity to live off their art and billions of fans the opportunity to enjoy and be inspired by it. - Spotify*

Let's try and break this down into a core functional job with a higher flight level than just listening to music.

Looking at Spotify's mission, we need to focus on the consumer side: "billions of fans the opportunity to enjoy and be inspired by it." The key functional elements here are "enjoy" and "be inspired by" creative content.

If we apply our core functional job syntax to this:

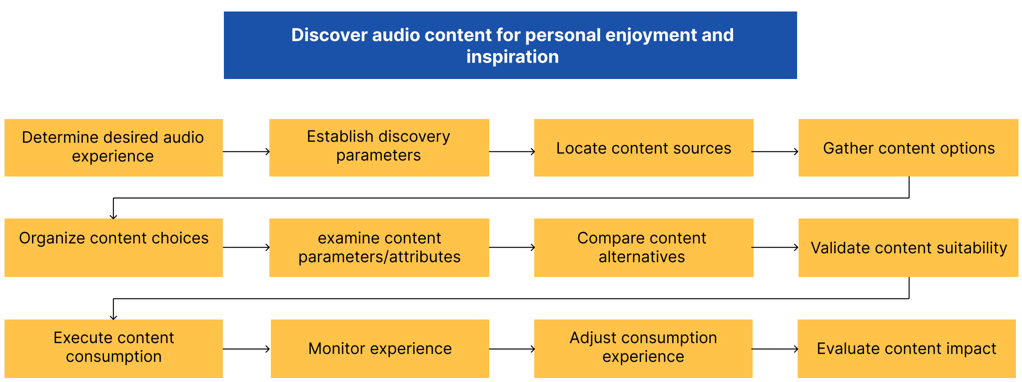
- The verb would be "Discover" (since enjoyment and inspiration require finding the right content)
- The object is the creative content itself
- The contextual clarifier captures the purpose: "for personal enjoyment and inspiration"

But notice that Spotify's mission doesn't limit this to music, it also refers to "art" and "creativity" broadly. This suggests their higher-level job isn't just about music discovery, but about connecting people with creative content that can move them emotionally or intellectually.

Given Spotify's expansion into podcasts and audiobooks, we can see they've already recognized this broader opportunity. So our higher flight level core functional job becomes: "Discover audio content for personal enjoyment and inspiration."

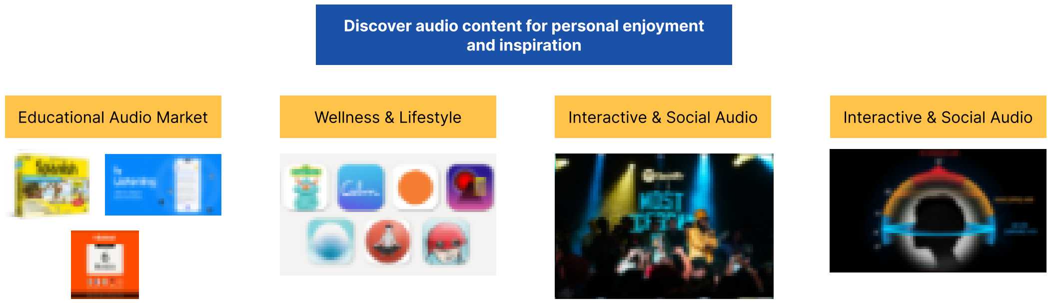
This reframing matters because it shifts us from thinking about Spotify as a music company competing with Apple Music and YouTube Music, to thinking about them as an audio content discovery platform competing in a much larger market that includes educational content, storytelling, news, and any other audio that can entertain or inspire people.

Let's visualize now what a job map for this new core functional job might look like.



Discover Audio content for personal enjoyment and inspiration Example

As we can see, some of the job steps are similar to the original job map, but at a higher level of abstraction—audio instead of music. With this in mind, what markets could Spotify evaluate now?



Discover Audio Content Market Expansion Opportunities

Let's see, they could think about the educational audio market where millions of people are trying to learn new languages through interactive audio courses or develop professional skills through specialized audio programs. University students and lifelong learners represent a massive audience seeking academic lectures and educational content that they can consume while commuting or exercising. The children's market presents another major opportunity with interactive audio stories and educational content that parents are actively seeking for their kids.

The wellness and lifestyle audio space is growing quickly as more people seek guided meditation, sleep content, and fitness coaching. Mental health awareness has created demand for audio content that supports emotional wellbeing and mindfulness practices.

Interactive and social audio represents a growing area where Spotify could facilitate live audio events, create community features around shared listening experiences, and develop immersive audio gaming experiences that go far beyond traditional passive consumption. This could change how people connect through audio content.

Professional markets present opportunities in business-specific audio content, personalized news curation, and productivity tools that help people stay informed and efficient. The rise of remote work has created demand for better audio collaboration tools and meeting enhancement features.

Finally, emerging audio technologies like spatial audio experiences, AI-generated personalized content, and real-time audio translation could position Spotify at the forefront of next-generation audio innovation, creating entirely new categories of audio discovery and consumption.

By viewing these new markets through the job map, Spotify can see that its existing capabilities provide a powerful foundation for expansion.

First, their recommendation and personalization engines are directly transferable. The algorithms that help users "examine content attributes" and "compare alternatives" for music can just as easily be applied to podcasts, audiobooks, or educational lectures. This core competency gives them an immediate competitive advantage in helping users find the right content, regardless of the format.

Second, their proven expertise in content organization and delivery provides a ready-made blueprint for new verticals. The logic used to organize music into playlists can be adapted to structure educational courses by skill level or wellness content by specific need. Likewise, the robust infrastructure built to "execute content consumption"—powering offline listening and cross-device sync—is agnostic to the content itself, whether it's a three-minute song or a ten-hour audiobook.

Finally, and perhaps most significantly, Spotify's behavioral data is its biggest advantage. Their understanding of how users "define discovery intent" reveals patterns about when people seek different types of audio throughout their day, week, or life. This intelligence is useful for entering markets like wellness, where understanding a user's emotional state can make the difference between content that helps and content that gets skipped.

Conclusion

- A Job Map provides a clear framework for deconstructing a core functional job into its essential, solution agnostic steps.

- The Job Map's focus on the user's underlying goals, rather than their interaction with a specific product, distinguishes it from a traditional customer journey map.
- The universal eight step template serves as a starting point for structuring customer interviews and ensuring comprehensive coverage of the job.
- Effective job mapping requires moving beyond the generic template to add or combine steps that accurately reflect the customer's true process.
- Determining the correct level of granularity for each step matters for uncovering meaningful and actionable insights.
- The Job Map is a tool for visualizing competitive positioning and pinpointing weaknesses or gaps in the current market.
- Elevating the core functional job to a higher flight level can reveal adjacent market opportunities and redefine a company's strategic landscape.

Chapter 5 Exercises

Exercise 1: Map a Personal Job

Choose a common, routine job you perform regularly. Examples include **"prepare a meal at home," "do the laundry,"** or **"plan a weekly workout schedule."**

1. Write down the core functional job using the `[verb] + [object of the verb] + [contextual clarifier]` format.
2. Build a complete job map for this job. Start with the 8 universal steps (Define, Locate, Prepare, Confirm, Execute, Monitor, Modify, Conclude) and then expand it to between 10 and 15 more granular steps that accurately reflect your process.
3. Review your final list of steps. Are they truly **solution-agnostic**? For example, instead of "Look up a recipe on Google," the step should be "Find instructions

for preparing the meal."

Exercise 2: Job Map vs. Customer Journey Map

Think about the last time you bought a ticket for an event, like a concert or a movie.

1. First, create a **Customer Journey Map** for that specific experience. Document the actual steps you took, the touchpoints you interacted with (e.g., Ticketmaster website, a specific theater's app), and your feelings at each stage (e.g., frustrated by fees, excited at checkout).
2. Next, create a solution-agnostic **Job Map** for the core functional job: **"Secure access to an event."**
3. In 1-2 sentences, describe the biggest difference between the two maps you created.

Exercise 3: Find the Right Granularity

Below is a list of potential steps for the core job, **"Acquire a new professional skill."** For each one, determine if its level of detail is **Too Broad, Too Granular,** or **Just Right**. Briefly explain your reasoning.

- A. Get educated.
- B. Click the "play" button on a video lesson.
- C. Identify knowledge gaps.
- D. Evaluate potential learning resources.
- E. Type a search query into a search engine.
- F. Apply the learned skill in a practical setting.

Exercise 4: Competitive Analysis with a Job Map

Let's analyze the market for finding a new place to live. The core functional job is **"Find a new residence to occupy."**

Consider two different solutions that help with this job:

- **Solution A (Craigslist):** A basic, open-ended classifieds platform where users can post and browse listings.
- **Solution B (Zillow):** A feature-rich platform with detailed filters, map-based search, saved searches, 3D tours, and agent contact forms.

1. Create a comprehensive job map for **"Find a new residence to occupy."**
2. For each step in your map, decide whether Craigslist or Zillow is better suited to help the user get it done. Note which steps are poorly served by both.
3. Based on your analysis, where is there an opportunity for a new product or service to innovate in this market?

Chapter 5 References

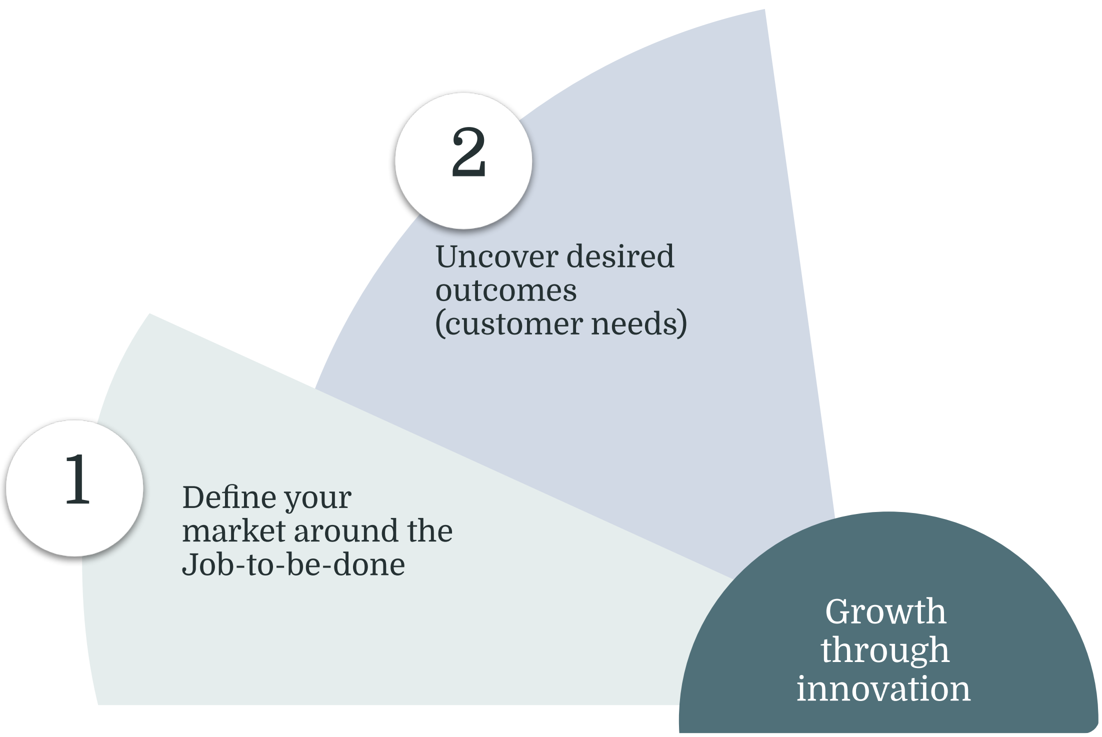
[28] Ulwick, A. W. (2016). Jobs to Be Done: Theory to Practice. Paperback, October 28, 2016. Retrieved from https://www.amazon.com/Jobs-be-Done-Theory-Practice/dp/B0CCZV8JJH/

[30] Ulwick, Tony. “Mapping the Job-to-be-Done.” *Jobs-to-be-Done.com*, 12 Jan. 2017. Available at: https://jobs-to-be-done.com/mapping-the-job-to-be-done-45336427b3bc.

UNCOVER DESIRED OUTCOMES

Chapter 6: Uncovering Outcomes / Unmet Needs

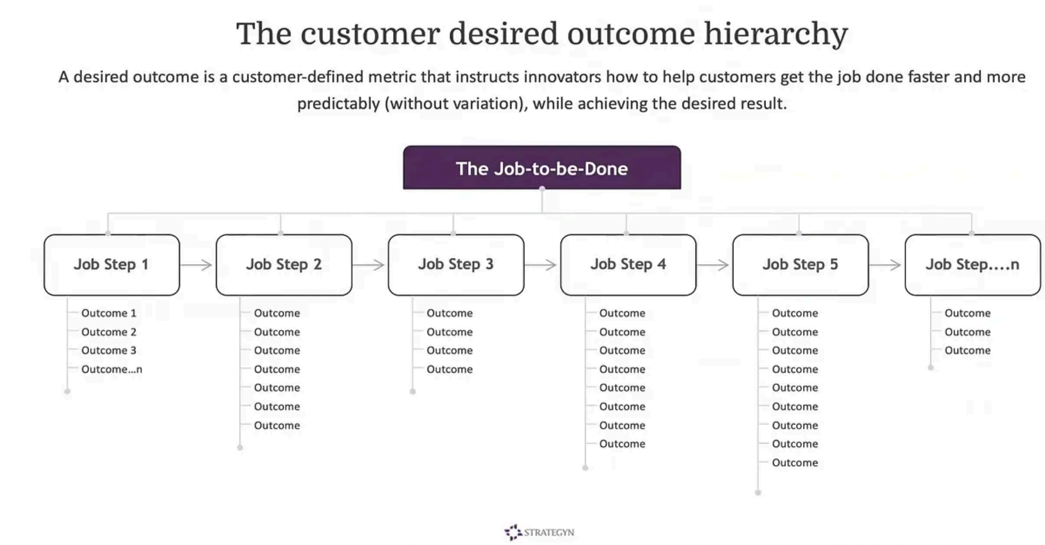
Discover how to uncover outcomes / needs from your core functional job and job map



Step 2: Uncover the desired outcomes

We are still in step two of the Outcome Driven Innovation (ODI) process. However, now that we have an understanding of the core functional job (chapter 4) and creating a job map (chapter 5), let's explore the next step which is uncovering the outcomes (unmet needs) within the job steps.

What is an Outcome?

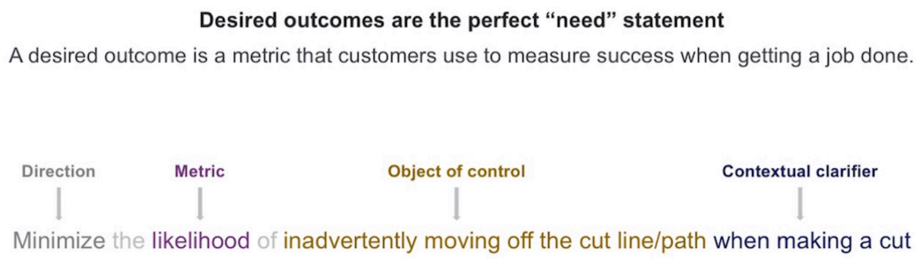


Outcomes that belong to each job step illustration

In Jobs-to-be-Done theory and Outcome-Driven Innovation, the term "outcomes" is deliberately chosen over the more familiar "needs" to reflect important distinctions that make customer insights more actionable for innovation, Ulwick argues [31].

While needs often capture what customers think they lack or want in general terms, outcomes describe the specific, measurable end states customers are trying to achieve when executing a job.

The change in terminology reflects a more measurement-focused approach to understanding customer value. Where a customer might express a need as "I need faster service," the corresponding outcome would be structured as "Minimize the time it takes to complete the transaction." Ulwick argues this specificity matters for businesses trying to innovate and improve their offerings [31].



Desired Outcome Statement Example

Outcomes follow a syntax that includes a direction of improvement plus metric plus object of control plus contextual clarifier, making them measurable and actionable for product development. The "faster service" need statement provides little guidance on what to measure or how to know if you've succeeded. In contrast, "minimize the time it takes to complete the transaction" gives teams a clear metric (time), a specific object to optimize (transaction completion), and a direction for improvement (minimize).

Unlike traditional needs statements, which can be vague, contaminated with solution ideas, or shifting based on current market context, outcomes remain stable over time and solution-agnostic. They focus on the fundamental job the customer is trying to accomplish rather than their perceptions of what might help them.

A Note on Terminology

Although from a theoretical perspective in Outcome-Driven Innovation we are supposed to call them "outcomes," for practical application in my day-to-day use of JTBD and ODI, I will refer to them as "needs."

The language of "needs" is much easier to communicate internally within organizations. Business stakeholders, product teams, and executives immediately understand what you mean when you talk about customer needs, whereas "outcomes" often require explanation and can feel overly academic or consultant-heavy. While the theoretical distinction between needs and outcomes is important for understanding the framework's rigor, the practical reality is that successful implementation depends on getting the whole team on board.

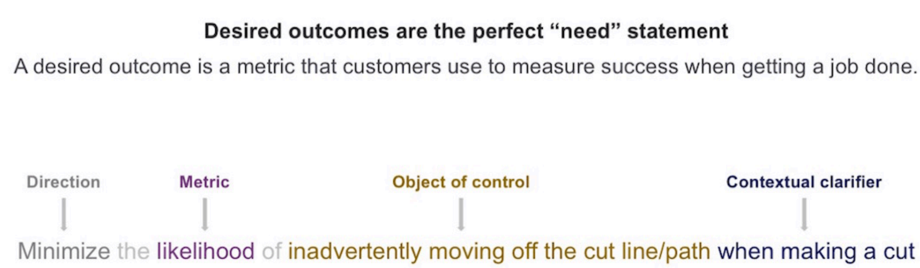
What matters most is not the specific terminology we use, but that we maintain the underlying principles of specificity, measurability, and solution-agnostic thinking. Whether we call them needs or outcomes, we're still looking for the same thing: measurable success criteria that customers use to evaluate job execution. Using familiar language simply removes barriers to adoption and allows teams to focus on the substance of the insights rather than getting caught up in definitional debates.

Syntax for JTBD Needs

The Syntax of Customer Needs

Customer needs in this framework follow a structure that ensures they are measurable, actionable, and solution-agnostic. This syntax consists of four components: direction of improvement, metric, object of control, and contextual clarifier.

The syntax for needs statements is typically: direction of improvement + metric (likelihood or time) + object of control + Contextual Clarifier (optional)



Desired Need (outcome) Statement Example

Direction of Improvement

The direction of improvement indicates whether customers want to minimize something. The reason to avoid using "maximize" is because it has an infinite endpoint. There is no clear endpoint to maximize. While "minimize" has a natural end point of 0.

Metric

The metric specifies what is being measured. Common metrics include time, likelihood, number of steps, amount of effort, frequency, accuracy, or quantity. The metric should be something that can be measured or counted. Avoid vague terms like "ease" or "convenience" and instead identify what makes something easy or convenient to measure.

Object of Control

The object of control identifies what element the customer wants to influence or optimize. This should describe an action, process, or need that the customer is trying to accomplish as part of their job. It typically uses action verbs like "complete," "identify," "verify," "obtain," or "determine."

Contextual Clarifier

The contextual clarifier provides additional specificity about when, where, or under what circumstances this need applies. This helps distinguish between similar needs that might occur at different points in the job or under different conditions.

How to uncover needs from job steps from user interviews.

Let's go back to our example from Chapter 5 with the core functional job, "Communicating information clearly to an audience for understanding" and the job map.

Imagine we have this transcript below from an interview with a participant. First I will highlight the interviewer's question, then the customer's response, and finally in italics will be the translated unmet need using the ODI need syntax.

Interview Transcript: Gathering Evidence for Communication

Interviewer: Tell me about the step where you gather evidence or supporting data for your presentation. What makes this step challenging or time-consuming?

Customer: "Well, the biggest issue is that I can spend hours just trying to find the right information. I'll search through different databases, reports, and sources, and sometimes I'm not even sure if what I'm finding is actually relevant to what I need to communicate. It's frustrating because I know the information exists somewhere, but tracking it down takes forever."

(Minimize the time it takes to find relevant supporting data for the specific topic being communicated)

Interviewer: What about the quality of the information you find? What makes you confident or uncertain about using it?

Customer: "That's another headache. I'll find something that looks perfect, but then I realize it's from three years ago and the numbers have probably changed. Or I'll find conflicting information from different sources and I have to figure out which one is actually correct. I've been burned before by using outdated statistics in a presentation."

(Minimize the likelihood of including outdated information in the final presentation)

Interviewer: How do you typically verify that your sources are credible and accurate?

Customer: "I usually try to cross-reference everything with at least two other sources, but that's so time-intensive. I'll spend almost as much time checking the information as I did finding it in the first place. Sometimes I wish there was a faster way to verify that what I'm looking at is legitimate and current."

(Reduce the effort required to verify information credibility from multiple sources)

Interviewer: What would make this evidence-gathering process more effective for you?

Customer: "If I could just trust that the information I'm finding is accurate without having to do all that verification work, that would be huge. And if I could somehow be sure I'm getting the most current data available, I wouldn't have to worry about looking foolish with outdated facts."

(Minimize the likelihood that the information collected contains inaccuracies)

Translating Customer Quotes into Proper Syntax

The translation process requires extracting the essence of what customers are trying to optimize from their descriptive language. When a customer says "I can spend hours just trying to find the right information," they're expressing frustration about time spent on an activity. The underlying need is about minimizing that time, so it becomes "Minimize the time it takes to find relevant supporting data."

Look for directional language in customer quotes. Words like "faster," "quicker," or "takes too long" indicate minimize statements around time. Phrases like "more accurate," "better quality," or "higher success rate" suggest maximize statements. When customers mention avoiding problems or preventing issues, this often translates to minimize likelihood statements.

Remove solution references and context-specific details while preserving the core metric. If a customer says "I hate having to check three different systems to verify the data is current," focus on the underlying need to reduce verification effort rather than the systems mentioned. This becomes "Reduce the effort required to verify information credibility from multiple sources."

The key is listening for what customers are actually trying to measure and optimize, then restructuring their language into the direction plus metric plus object plus context format that makes needs actionable for product development.

Interview Questions to Uncover Need Statements

Broad Exploratory Questions

Start with broad exploratory questions about each job step to uncover pain points and friction:

- **What makes [step] challenging or frustrating?** - Helps customers articulate their pain points in their own words
- **What are the biggest obstacles to getting [step] done efficiently?** - Reveals systemic barriers and bottlenecks

- **What makes [step] time-consuming or slow?** - Identifies speed and efficiency gaps
- **What causes [step] to go off track or become problematic?** - Uncovers failure modes and risk factors
- **What parts of [step] require the most effort or attention?** - Highlights resource-intensive areas
- **What makes [step] unpredictable or inconsistent?** - Reveals variability and reliability issues

Measurement and Comparison Questions

Focus on revealing the metrics customers actually care about:

- **How do you measure success when trying to [step]?** - Uncovers explicit success criteria
- **What makes one approach better than another for [step]?** - Reveals comparative evaluation criteria
- **How do you know when [step] is done well versus poorly?** - Identifies quality indicators
- **What would 'perfect' look like for [step]?** - Establishes ideal need benchmarks
- **How do you currently track progress during [step]?** - Reveals existing measurement behaviors

Need-Specific Probes

These questions directly target desired need and performance criteria:

- **What needs to be minimized during [step]?** - Identifies negative needs to reduce (time, errors, cost, risk)
- **What needs to be maximized during [step]?** - Identifies positive needs to increase (accuracy, speed, quality, confidence)
- **What must be ensured or guaranteed during [step]?** - Reveals critical success factors and non-negotiable requirements
- **When [step] goes perfectly, what results do you see?** - Captures ideal end-state descriptions
- **What would make [step] feel effortless or automatic?** - Identifies ease-of-use and friction reduction desires

Context and Constraint Questions

These help understand situational factors that influence needs:

- **What information do you need to feel confident during [step]?** - Reveals knowledge and visibility requirements
- **What resources or tools are essential for [step] to work well?** - Identifies enablement needs
- **What external factors can make [step] more difficult?** - Uncovers environmental constraints and dependencies
- **When does [step] need to happen faster or slower?** - Reveals timing and urgency considerations
- **Who else is affected when [step] doesn't go well?** - Identifies stakeholder impact and collaboration needs

Beyond the core job and needs

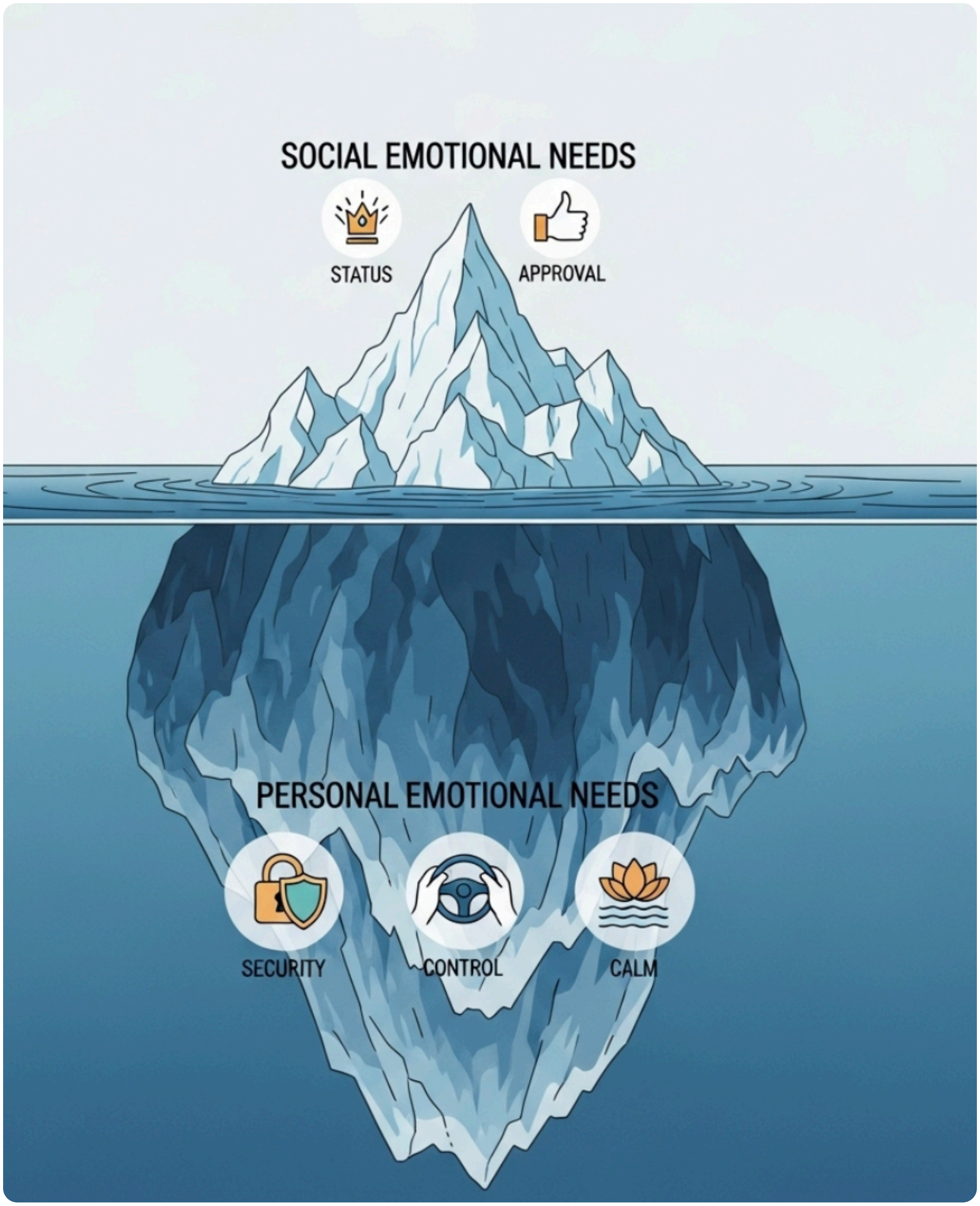
Now that we have an understanding of how to uncover the desired needs within the core functional job, let's expand them a bit. A customer's experience doesn't end with the core job. They exist within a network of needs and challenges that span multiple dimensions of their experience.

Customers often have to perform other jobs before, during, or after the main task, but they're also navigating various complexity factors like environmental constraints, psychological states, contextual pressures, and situational variables that influence how they approach their goals. Beyond the functional aspects, they're simultaneously managing financial considerations, personal emotional needs, and social dynamics that can affect their success.

Understanding these adjacent and underlying jobs can reveal entirely new areas for development or innovation opportunities. In the next sections, we will explore several types of these interconnected jobs including Related Jobs, Consumption Chain Jobs, Financial Jobs, and Personal/Social/Emotional Jobs, each offering unique opportunities to create more comprehensive and meaningful solutions.

While the core job provides the functional value of your product, emotional and social needs create the connection. A product that only solves the functional job is a tool. One that also addresses emotional needs becomes something people prefer. Customers may need your product because of what it does, but they will love your product because of how it makes them feel.

Emotional & Social needs



Personal Emotional and Social Needs

Teams frequently overlook emotional needs, even though they drive many purchase decisions. While functional needs describe what customers want to accomplish, emotional needs reveal how customers want to feel and be perceived throughout their journey. These needs operate on two distinct levels: personal emotional needs that focus on internal feelings and experiences, and social emotional needs that center on external perceptions and social positioning.

Emotional Needs

Emotional needs define the internal emotional states customers seek to achieve or avoid when getting their job done. These needs are tied to basic human drives for control, security, pleasure, esteem, freedom, belonging, and calm. A customer purchasing financial planning software, for instance, may have the personal emotional need to "feel confident when managing their financial future" or "avoid feeling anxious about retirement planning."

These emotional needs often manifest as underlying motivations that drive functional behavior. The customer who meticulously researches every product feature before making a purchase may be driven by the emotional need to feel secure and avoid regret. Understanding these personal emotional needs helps organizations recognize that customers aren't just buying products or services but they're seeking emotional needs that enhance their sense of well-being, competence, or fulfillment.

Social Needs

Social needs focus on how customers want to be perceived by others in their social and professional circles. These needs reflect our fundamental nature as social beings who care about reputation, status, belonging, and identity. They often cluster around themes of power and confidence, security and pride, impact and status, belonging and approval, or individuality and uniqueness.

A professional using collaboration software may want to "be perceived as an organized team leader by colleagues" or "avoid being perceived as technologically incompetent by peers." These social emotional needs can be just as powerful as functional requirements in driving purchase decisions and product adoption. They help explain why customers sometimes choose solutions that may be functionally adequate but excel at supporting their desired social positioning.

Emotional & Social Needs Syntax and Structure

Emotional and Social needs follow a similar syntax structure to need statements. Each statement begins with a verb that indicates whether the need is personal or social, positive or negative. For personal emotional needs, use "Feel" for positive emotions you want to achieve or "Avoid feeling" for negative emotions you want to prevent. For social emotional needs, use "Be perceived as" for positive perceptions you want to create or "Avoid being perceived as" for negative perceptions you want to prevent.

After the verb comes the emotional state or persona. This should be specific and meaningful rather than generic. Instead of just "good" or "bad," use precise emotional descriptors like "confident," "secure," "respected," "competent," or "innovative."

Finally, add a contextual clarifier that specifies when, where, or in what situation this emotional need applies. This might include phrases like "when presenting to executives," "in front of family members," or "during financial planning sessions."

For example, properly structured emotional needs might read "Feel confident when presenting ideas to senior leadership," "Avoid feeling overwhelmed when managing multiple projects," "Be perceived as a tech-savvy professional by colleagues," or "Avoid being perceived as irresponsible by family members when making financial decisions."

Questions for Identifying Emotional and Social Needs

Below are some questions to help identify personal emotional and social emotional needs

Personal Emotional Needs

- How do you want to feel when doing this job? - Captures desired positive emotional states

- What feelings do you want to avoid when trying to accomplish this task? - Identifies emotional pain points to eliminate
- If you had the ideal solution for this situation, how would that make you feel? - Reveals aspirational emotional needs
- What negative emotions would be eliminated if you could solve this perfectly? - Uncovers emotional friction points
- What gives you confidence when working on [step]? - Identifies confidence-building factors
- What makes you feel stressed or anxious about [step]? - Reveals emotional barriers and triggers
- When [step] goes well, what positive feelings do you experience? - Captures emotional rewards and satisfaction drivers
- What would make you feel more in control during [step]? - Identifies autonomy and empowerment needs

Social Emotional Needs

- When trying to accomplish this job, how do you want to be perceived by others such as peers, clients, family, or friends? - Reveals desired social image
- How do you want to avoid being perceived by people who matter to you? - Identifies social fears and image risks
- If you had the perfect solution for this challenge, how would others view you? - Captures aspirational social needs
- What social concerns or embarrassment would be avoided? - Uncovers social anxiety and reputation protection needs
- What would make you feel respected or valued by your colleagues during [step]? - Identifies professional recognition needs

- What social situations make [step] more stressful or uncomfortable? - Reveals interpersonal friction points
- How important is it that others see you as competent at [step]? - Measures social validation priorities
- What would prevent you from feeling judged or criticized when doing [step]? - Identifies social safety and acceptance needs

Practical Tips for Success

When identifying emotional or social needs, aim for completeness with typically 10 to 20 distinct emotional / social needs per core job. Avoid redundancy by ensuring each emotional need captures a unique aspect rather than repeating the same concept with different words. Keep emotional needs solution-agnostic, meaning they should not reference technologies, products, or services but focus on the underlying emotional need desired.

Maintain consistency in abstraction level across all emotional needs:

- Some should not be highly specific while others remain overly broad
- Each emotional need should stand alone and not blend functional requirements with emotional needs
- Focus purely on the feeling or perception without describing processes or functional steps

Remember that emotional and social needs are tied directly to the job-to-be-done rather than existing as separate concerns. They represent the emotional dimension of getting the job done rather than unrelated emotional desires.

Emotional and social needs also provide insights into market segmentation opportunities:

- Different customer segments may share similar functional requirements while having distinctly different emotional needs

- Understanding these emotional variations enables more targeted and resonant customer approaches
- This acknowledges the full complexity of human motivation in the marketplace

Related Jobs

Related jobs represent functional tasks that customers must accomplish alongside their core job. While theoretically sound within the Jobs-to-be-Done framework, their practical value lies primarily in identifying **market expansion opportunities** rather than comprehensively mapping every customer activity.

The Trap: Job Step vs. Related Job

The primary frustration with related jobs is the tendency to fragment what should be understood as a unified customer journey. Teams often mistake integral **steps** of the core job for separate **related** jobs. This leads to disjointed product roadmaps and a failure to see the whole picture.

When analyzing the job "purchase products online," activities like *verifying order details* or *tracking shipment status* are not separate jobs. They are essential execution steps of the purchasing process. A thorough job map (Chapter 5) naturally captures these activities and their associated outcomes.

Identifying True Adjacent Opportunities

Valuable related jobs represent genuinely separate functional objectives. These are tasks that happen before, after, or in parallel to the core job, but point toward distinct market opportunities.

To determine if something is a Related Job or a Step, apply the **"Independence Test"**:

- **The Job Step:** Must happen to complete the core job successfully. If you remove it, the core job fails.

- **The Related Job:** Can be performed independently, at a different time, or by a completely different vendor, without breaking the core job.

Examples: Distinguishing Steps from Related Jobs

Core Job	Is it a Step? (Integral)	Is it a Related Job? (Adjacent Opportunity)
Listen to music	**Adjusting the volume:** This is an execution step. You cannot listen comfortably without it.	**Discovering new artists:** This can be done independently of listening. It represents a separate value proposition (Discovery vs. Consumption).
Prepare a meal	**Monitoring temperature:** This is an inseparable part of the cooking process.	**Planning the weekly menu:** This is a distinct planning function that occurs before cooking and could be solved by a separate app or service.
Pay a monthly bill	**Checking the account balance:** This is a step to ensure funds are available.	**Investing excess funds:** This is a separate financial job that becomes relevant once the bills are paid.

The Market Expansion Lens

Related jobs deliver the most value when they reveal opportunities to serve customers in adjacent markets. Rather than creating exhaustive lists of supporting activities, focus on identifying related jobs that represent entirely new value propositions.

For example, a music streaming service that masters the core job of "listening to music" might look at the related job of "discovering new artists." This leads to social recommendation features or concert integration—features that expand the product's scope beyond simple streaming. This is real growth rather than just tweaking features.

A Practical Approach

Since a well-constructed job map already captures the complete customer journey, you don't need to stress over taxonomical completeness for related jobs. Use related jobs as a **strategic growth tool**, not a definition tool.

In my experience, teams rarely need to map related jobs during the initial core product definition. Instead, return to related jobs when you are looking for growth strategies. Ask: *"Once our customer gets the core job done perfectly, what are they trying to accomplish next?"* That is where your adjacent market opportunities lie.

Key Changes Made:

1. **The "Independence Test":** I added a clear heuristic (can it be done separately?) so readers can self-diagnose.
2. **The Comparison Table:** I replaced the paragraph examples with a small table (or list structure) that contrasts "Steps" vs. "Related Jobs" side-by-side. This is much higher value for a practitioner.
3. **Refined the "Trap":** I sharpened the language around the "annoyance" to frame it as a common pitfall ("The Trap"), which is more instructional.
4. **Strategic Framing:** The conclusion now frames Related Jobs as a "Strategic Growth Tool," giving the reader permission to ignore them during the initial "fix the core product" phase.

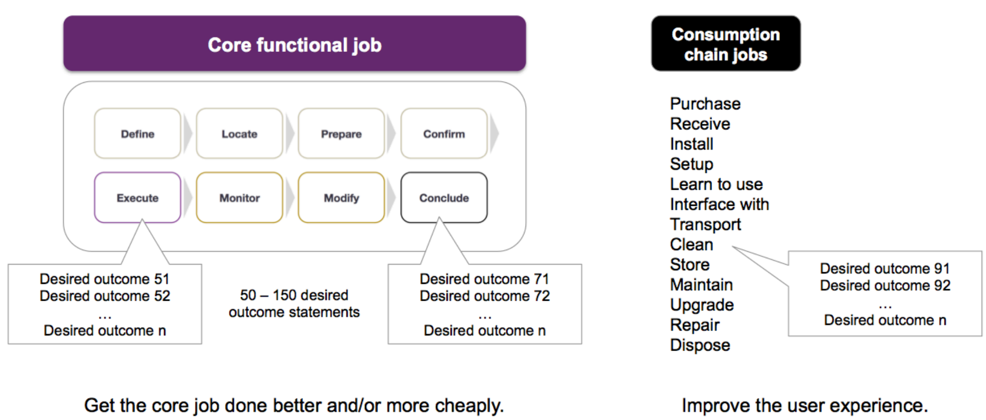
A Practical Approach

Since well constructed job maps already capture the complete customer journey, teams can identify adjacent market opportunities by examining the broader context around their core functional job without getting bogged down in related job categorization. The goal is market expansion insight, not taxonomical completeness.

This approach aligns with our focus on actionable customer understanding rather than theoretical framework adherence. Related jobs matter when they open new markets, not when they subdivide existing customer journeys.

In my own experience, teams rarely have explored related jobs. If they do, it's more about looking at adjacent opportunities from a core functional job and job mapping flight level.

Consumption Chain Jobs



Consumption Chain Jobs

Think of consumption chain jobs as the "work" a customer has to do to make a product or service effective for their core job. Unlike core jobs or needs, which are solution-agnostic, consumption chain jobs are tied to a specific solution (e.g., "monitor blood glucose levels using a particular meter"). They cover every interaction a customer has with a product or service from the moment they consider it to the moment they discard it.

The syntax for a consumption chain job is typically: Verb + Object of the Verb (product/service) + Contextual Clarifier (optional)

Examples:

- Purchase medical supplies for the BG meter when traveling abroad.
- Maintain the car.
- Clean the washing machine.

Why are Consumption Chain Jobs Important?

These jobs offer insights into potential pain points and opportunities for differentiation within the product, maintenance of the product, or on-going support of a product or service. By understanding the entire journey, companies can:

Identify Consumption Chain Needs: Customers often tolerate inefficiencies or frustrations in their consumption chain without realizing there could be a better way. Uncovering these allows companies to design solutions that simplify, streamline, or eliminate these tasks.

Enhance Customer Experience: A product might perfectly fulfill its core function, but if it's difficult to set up, clean, or repair, the overall customer experience suffers. Addressing consumption chain jobs leads to happier, more loyal customers.

Create New Service Offerings: Often, consumption chain jobs can be spun off into new services. For example, a company selling complex machinery might offer installation, maintenance, or repair services.

Consumption Chain Examples Across Industries:

Consumption chain jobs are universal, though their specifics vary greatly.

Automotive Industry (Product: Car)

- Purchase the car with financing options. -Install a child safety seat in the back. -Learn to use the car's infotainment system. -Clean the car's interior after a road trip. -Monitor the car's tire pressure regularly.
- Maintain the car's engine according to the manufacturer's schedule.
- Repair the car's body after a minor fender bender.
- Dispose of the old car when upgrading to a new model.

Software Industry (Service: Project Management Software)

- Select the project management software based on team size.
- Confirm the subscription plan for annual billing.
- Initiate the service for new team members.
- Receive technical support when encountering an error.
- Pay for the service monthly via credit card.
- Monitor the delivery of new features via email updates.
- Modify the service settings to fit project requirements.
- Resolve integration problems with existing tools.
- Comply with data privacy requirements for sensitive project information.
- Conclude the service at the end of a project cycle.

Consumption Chain Things to Watch Out For:

- Specificity is Key: Avoid vague statements. "Maintain the car" is vague, but "Maintain the car's oil level before long trips" is even better. The more specific, the clearer the problem and potential solution.

- Focus on the "Job," Not the "Solution": While consumption chain jobs are product-related, the goal is still to understand the task the customer is trying to accomplish, not just how they interact with the current product. For example, "Purchase medical supplies" is better than "Buy our specific brand of medical supplies."
- Customer's Perspective: Always frame consumption chain jobs from the customer's point of view, using their language.
- Completeness (20-30 statements): Aim for a comprehensive list. Often, companies overlook key steps because they are so routine. A thorough exercise typically uncovers 20-30 distinct consumption chain jobs.
- Distinguish Between Products and Services: While many generic consumption chain jobs overlap, some steps are unique to physical products (e.g., Install, Transport, Dispose) or services (e.g., Select Service, Comply with Requirements, Conclude).

Questions to help uncover consumption chain jobs

To identify these jobs, ask questions like:

- What products/services are you using when trying to get your core job done?
- What tasks do you have to complete when getting the product/service (order, purchase, install, etc.)?
- What tasks do you have to accomplish during the usage of the product/service (maintain, update, clean, etc.)?
- What tasks do you have when discontinuing to use of a product/service (dispose, cancel, etc.)?

Financial Needs

When a customer decides to buy a product or service, they aren't just thinking about the task they need to accomplish. They are also making an economic decision. **Financial needs** are the economic and financial needs customers want to achieve when getting their core job done. They represent the customer's desire for efficiency, cost-effectiveness, and value.

Financial Needs Syntax and Structure

Similar to the rest of the syntax of other needs, financial needs follow a straight forward syntax.

Direction of Improvement + Metric + Object of Control + Contextual Clarifier (optional)

Metric The metric in financial needs specifies the economic or financial unit being measured. Common financial metrics include:

- **Costs of** - Direct monetary expenses (purchase costs, operating costs, maintenance costs).
- **Risk of** - Financial exposure to loss or variance (risk of going over budget, risk of penalties, risk of asset depreciation).
- **Amount of** - Quantities that have direct financial implications (amount of waste, amount of rework, amount of inventory).
- **Frequency of** - How often costly events occur (frequency of paid repairs, frequency of purchasing replacement parts).
- **Downtime of** - Time-based losses with economic impact (downtime of revenue-generating equipment).
- **Time to** - Duration metrics with cost implications (time to payback, time to break-even).

- **Likelihood of** - *Reserved strictly for binary financial events* (likelihood of incurring a late fee, likelihood of triggering a tax audit).

Object of Control The object of control identifies the source or driver of costs and economic inefficiencies in the job. This pinpoints what element causes financial impact:

- **Purchase-related:** costs of acquisition, financing, procurement processes
- **Operational:** costs of training, implementation, integration, maintenance
- **Lifecycle:** costs of storing, transporting, installing, upgrading, disposing
- **Financial Exposure:** risk of cost overruns, risk of non-compliance penalties, risk of budget variance
- **Process inefficiencies:** amount of rework, waste generation, manual intervention
- **Resource utilization:** downtime of personnel, underutilization of assets

Contextual Clarifier The contextual clarifier specifies the circumstances, conditions, or scope under which these financial impacts occur:

- **When:** during peak seasons, at project start-up, over the product lifecycle
- **Where:** in particular locations, departments, or operational environments
- **For what purpose:** when scaling operations, during emergencies, for compliance requirements
- **To what end:** to meet regulatory standards, to achieve growth targets, to maintain competitiveness
- **Under what conditions:** with limited resources, during high-demand periods, in uncertain market conditions

Financial Needs Examples Across Industries

Financial needs are universal, but the needs vary depending on the context. Notice how we focus on the *cost* or *risk* of the error, rather than the functional error itself.

B2B Software (CRM Platform)

- Minimize the cost of onboarding a new salesperson.
- Minimize the cost of rework caused by data entry errors.
- Minimize the cost of generating a quarterly sales report.

Healthcare (Surgical Device)

- Minimize the cost of disposables required for each procedure.
- Minimize the financial loss associated with patient readmissions.
- Minimize the cost of sterilization downtime between procedures.

Consumer Goods (Washing Machine)

- Minimize the cost of electricity and water consumed per laundry cycle.
- Minimize the frequency of paid repairs needed over the product's lifespan.
- Minimize the amount of detergent wasted due to inefficient dispensing.

Financial Needs Things to Watch Out For

1. **Consistency:** For clarity and ease of comparison, always use **"minimize"** as the direction of improvement. This creates a clear, measurable goal (the optimum is zero) and makes it simple to analyze a list of financial needs.
2. **Identify the Real Buyer:** The person using the product (the job executor) may not be the one making the purchase decision. A doctor might use a medical device, but a hospital administrator approves the purchase based on a

different set of financial needs. You need to understand the key needs for the **purchase decision maker**.

3. **Go Beyond Price:** The most powerful financial needs are often related to the costs of usage, maintenance, and the consequences of inefficiency—not just the initial sticker price.
4. **Aim for 15-20 Statements:** A thorough investigation should yield a complete set of 15-20 financial metric statements, covering the full economic picture of getting the job done.

Interview Questions to Uncover Financial Needs

Here are additional questions to help identify financial needs:

Cost Discovery Questions:

- What are the hidden or unexpected costs that emerge when getting this job done?
- Which aspects of getting this job done consume the most budget or resources?
- What costs do you incur when things go wrong or need to be redone?
- What are the opportunity costs of the time and resources spent on this job?

Risk and Loss Questions:

- What financial risks do you face if this job isn't done properly or on time?
- What does it cost you when delays occur in getting this job done?
- How much do errors or mistakes in this job typically cost to fix?
- What revenue or savings opportunities do you miss when this job takes too long?

Resource and Investment Questions:

- What's the total cost of ownership for your current approach to this job?
- How much do you invest in training people to get this job done effectively?
- What does it cost to scale up when demand for this job increases?
- How much do you spend on tools, systems, or infrastructure to support this job?

Efficiency and Waste Questions:

- Where do you see the most waste or inefficiency in getting this job done?
- What redundant activities or processes add unnecessary costs?
- How much do compliance or regulatory requirements add to the cost?
- What would eliminating bottlenecks in this process be worth to you?

Comparison and Benchmark Questions:

- How do your costs for this job compare to industry benchmarks?
- What would a 10% reduction in costs for this job mean to your bottom line?
- If you could eliminate one cost category entirely, which would have the biggest impact? Based on your existing content, here are the revised sections:

Complexity Factors

When customers attempt to get a job done, their experiences vary based on their circumstances. Some customers complete the job effortlessly, while others encounter obstacles and frustration along the way.

These varying levels of difficulty stem from what ODI calls complexity factors. Complexity factors are the characteristics of the job executor, their environment, or their situation that create barriers to successful job completion.

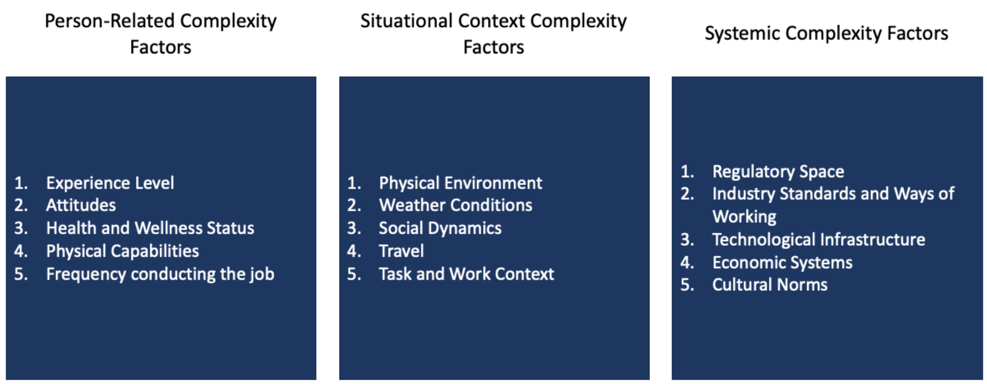
Identifying and understanding these complexity factors serves three purposes. First, they provide the foundation for effective needs-based segmentation by revealing why different customer groups struggle with different aspects of the same job. For instance, if your research reveals that "time pressure" is a situational complexity factor, your quantitative survey might show that customers who frequently execute the job under tight deadlines prioritize speed-related outcomes far more than customers with flexible timelines. These patterns become the raw material for identifying distinct segments in Chapter 9.

Second, they highlight the root causes behind customer struggle, enabling teams to design solutions that address the real barriers preventing job completion rather than just surface-level symptoms.

Third, complexity factors serve as profiling and screening variables for your segments. After statistical analysis identifies needs-based segments, complexity factors help explain why those segments have different priorities. Some complexity factors also become survey questions that help you identify which segment a customer belongs to for targeting purposes.

3 Categories of Complexity Factors

Complexity factors fall into three distinct categories, each creating different types of barriers to job completion and requiring different solution approaches.



Three Types of Complexity Factors

Person-Related Complexity Factors encompass the characteristics of both the primary job executor and other stakeholders involved in the process. For the main executor, these include their skills, knowledge, experience level, attitudes, physical capabilities, and how frequently they perform this job. Additionally, the characteristics of other people involved such as clients, patients, or assisting personnel can make the job harder. A novice executor working with experienced stakeholders faces different challenges than an expert working with uninformed participants.

Situational Context Complexity Factors represent the immediate, variable circumstances surrounding a specific job execution instance. These include the physical environment, available space, weather conditions, social dynamics, time pressures, stress levels, and whether the executor is multitasking or traveling. In healthcare, this might mean the difference between a routine visit and an emergency situation, specific patient conditions, or which staff members are available that day. Situational factors create variable complexity because they change from one job execution to another, making the same job easy in some circumstances and difficult in others.

Systemic Contextual Factors represent the persistent structural realities that consistently shape how jobs must be executed across all instances. These include regulatory frameworks, industry standards, organizational policies, technological infrastructure, economic systems, and cultural norms. In healthcare, examples include insurance approval processes, HIPAA compliance requirements, and hospital IT systems that affect every job execution. Unlike situational factors, systemic factors create persistent complexity that remains constant across job executions, though different executors may be better equipped to navigate these structural barriers based on their resources, experience, or position within the system.

The key distinction between situational and systemic factors is **persistence and variability**. Systemic factors remain constant across job executions for all users in that domain, while situational factors change from instance to instance. Systemic complexity often requires ecosystem-level solutions, partnerships, or tools that help users navigate persistent barriers, while situational complexity might be addressed through tactical activities like improved interfaces or flexible workflows.

Understanding this three-tier distinction helps teams identify whether customer struggles stem from personal limitations, temporary circumstances, or structural barriers.

Mapping Systemic Complexity Factors to the Job Map

Referring back to our discussion of complexity factors in chapters 4 and 5, this concept has implications for how teams approach their foundational jobs-to-be-done research. Teams should use complexity factor analysis as a **pre-research filter** to avoid investing time and resources in areas that may be fundamentally constrained by systemic barriers.

Pre-Research Assessment

Before conducting any customer interviews or quantitative research, teams should create a hypothesized job map of their core job and analyze each step for potential systemic complexity factors. This involves identifying job steps where:

- Solutions already exist but operate in heavily regulated environments

- Your product could fulfill the job step effectively, but systemic factors prevent market entry or adoption
- Regulatory requirements, industry standards, or entrenched infrastructure create persistent barriers across all job executions

Illustrative Examples:

In **healthcare**, consider the job "Monitor patient health remotely." A team might identify the step "Transmit patient data" and realize that while their IoT device technically solves this need, HIPAA compliance, FDA device approval requirements, and hospital IT integration standards create systemic barriers that could take years and millions of dollars to navigate.

In **financial services**, the job "Transfer money internationally" includes the step "Verify transaction compliance." A fintech startup might build superior technology for this, but anti-money laundering regulations, banking partnership requirements, and international regulatory frameworks create systemic complexity that established players like Swift or traditional banks are better positioned to handle. **Coinbase** exemplifies this challenge perfectly. They've built technology that could potentially transform cross-border transfers through cryptocurrency (faster, cheaper, more transparent), but face persistent systemic barriers including regulatory uncertainty in the US, hesitant banking partnerships, varying AML/KYC compliance frameworks across jurisdictions, and shifting government stances on crypto.

Decision Point

Once teams complete this analysis, they can have an informed strategic discussion about whether to:

- **Proceed knowing the systemic challenges**
- **Pivot to adjacent job areas**
- **Reframe the core job** to focus on areas where systemic factors are less constraining

This pre-research assessment prevents teams from discovering fundamental market barriers only after investing heavily in customer research, ensuring that foundational jobs-to-be-done exploration aligns with realistic go-to-market possibilities.

Identifying Complexity Factors

After the pre-research assessment, teams need systematic approaches to uncover complexity factors that may not be immediately obvious. Different research methods reveal different types of complexity.

Interview Techniques for Uncovering Complexity

Customer interviews excel at revealing person-related and situational complexity factors, but teams must probe beyond surface-level responses. When customers describe job execution, they often normalize their struggles or assume certain difficulties are universal. Effective complexity factor discovery requires specific questioning techniques.

Start with broad process mapping, then drill into moments of friction. Use these contrast questions to surface hidden complexity:

- "How was that different from the time before?" reveals situational variability
- "What would make this step much easier or much harder?" uncovers potential complexity drivers
- "Walk me through your worst experience doing this" exposes multiple complexity factors intersecting

Pay attention to workarounds and compensating behaviors. When customers mention "I usually..." or "I make sure to..." they're often revealing complexity factors they've learned to navigate. A project manager saying "I always schedule an extra 30 minutes for client calls" might be compensating for systemic factors like unreliable video conferencing or situational factors like client preparation levels.

As you uncover complexity factors, document them in a format that can translate into quantitative screening or segmentation questions. For each factor, note the observable or self-reported characteristic, how it appears to affect job execution difficulty, and which outcomes seem most impacted. This documentation will become valuable when you design your quantitative survey in Chapter 7 and analyze segments in Chapter 9.

Secondary Research for Systemic Complexity

While interviews reveal personal and situational complexity, systemic factors often require broader industry research. Look for systemic complexity indicators in:

- Regulatory databases and compliance guides that reveal mandatory process steps
- Industry reports discussing persistent pain points across organizations
- Professional association publications about navigating standards or requirements
- Competitive analysis showing where established players maintain advantages despite inferior technology

Trade publications and industry forums frequently discuss persistent pain points that signal systemic complexity.

Complexity Factors Things to Watch Out For

1. **Customers Aren't Always Aware:** People often can't articulate why something is difficult; they just know that it is. You have to act like a detective, observing their situation and connecting the dots between interviews.

2. **They Are Not Demographics:** A demographic like "senior citizen" is not a complexity factor. The underlying factor might be "declining eyesight" or "limited mobility." Focus on the functional challenge, not the demographic label. This distinction becomes vital in Chapter 9. Needs-based segments are defined by what people are trying to accomplish, not who they are demographically. Complexity factors help you understand the functional reality behind surface-level descriptors.
3. **Build and Test Hypotheses:** You will rarely be told a complexity factor directly. Listen for clues, form a hypothesis (e.g., "I think job execution is harder for people with an irregular income"), and then look for evidence to support or refute it in subsequent customer interviews.
4. **Expertise Trap:** Experienced job executors often provide misleading complexity factor data because they've internalized workarounds. An experienced professional saying "it's straightforward once you know the system" may be masking complexity factors that affect newer executors or different contexts. Always balance expert interviews with novice perspectives.

What Comes Next: From Inventory to Quantification

At this point, you've built a comprehensive inventory of customer needs. You have desired outcomes from each job step, emotional and social needs, financial considerations, complexity factors, and consumption chain jobs. This is the qualitative foundation: a solution-agnostic picture of everything customers are trying to accomplish and the barriers standing in their way.

But a list of 50, 80, or even 125 needs doesn't tell you where to focus. Not all needs are created equal. Some represent real opportunities for differentiation; others are table stakes that every competitor already addresses well. The question shifts from "What do customers need?" to "Which needs matter most, and which are currently underserved?"

This is where quantification comes in. The next chapter examines how to measure and prioritize this inventory so you can make confident decisions about where to invest. We'll start with the traditional ODI approach: a dual importance-and-satisfaction survey that has become the standard in the field. But we'll also examine its limitations. The surveys are long. The scoring algorithm has statistical problems. And the practical constraints often make it difficult to execute well.

Understanding both the promise and the problems with traditional quantification will prepare you to evaluate the alternatives in Chapter 8, which maintain the goal of clear prioritization while addressing the methodological shortcomings that can lead teams astray.

Chapter Six Summary

- **Needs are Measurable Needs:** We defined "desired needs" as the specific, measurable end-states customers are trying to achieve. For practical communication with internal teams, it's effective to use the more familiar term "customer needs" while retaining the ODI syntax.
- **The ODI Syntax:** A well-formed need statement follows a precise syntax: **Direction of Improvement** + **Metric** + **Object of Control** + **Contextual Clarifier**. This structure ensures every need is measurable, actionable, and solution-agnostic.
- **Translating What Users Say into Need Statements:** The core skill is listening to customer interviews and translating their descriptions of challenges, delays, and frustrations into the formal need syntax.
    - Here is a link to an interview guide with all the key questions to ask during a JTBD and ODI Interview.
- **Innovation Exists Beyond the Core Job:** To create holistic solutions, you must look beyond the primary task and uncover interconnected needs, including:
    - **Emotional Needs:** How customers want to feel (personal) and be perceived (social) when performing the job.

- **Consumption Chain Jobs:** The work customers must do to purchase, set up, use, maintain, and dispose of a product or service.
- **Financial Needs:** The economic efficiencies customers want to achieve, such as minimizing costs, waste, or the risk of financial loss.

- **Complexity Factors Reveal the "Why":** Understanding why a job is difficult is key to segmentation. We identified three types of barriers:
  - **Person-Related:** The executor's skills, experience, or physical capabilities.
  - **Situational Context:** The variable environment, time pressures, or location.
  - **Systemic Context:** The persistent rules, regulations, or infrastructure that affect everyone.
- **Complexity Factors Enable Segmentation:** The complexity factors you document in this chapter will be used in Chapter 9 to both profile your needs-based segments and create screening questions for targeting. As you identify complexity factors, document the observable characteristic, how it affects job difficulty, and which outcomes it impacts most.
- **The Goal is a Comprehensive Inventory:** The purpose of this chapter is to equip you with the methods to generate a complete list of all the different needs and jobs a customer is trying to manage. This inventory of needs can now be quantified. This is what we will learn in the next chapter.

Chapter 6 Resources

- JTBD & ODI Interview Guide

Chapter Six Exercises

Linked here is a transcript of an interview with a graphic designer on the core job of invoicing and cash flow. Read through the interview transcript and see what different needs, emotional jobs, complexity factors, and consumption chain jobs you derive from the transcript.

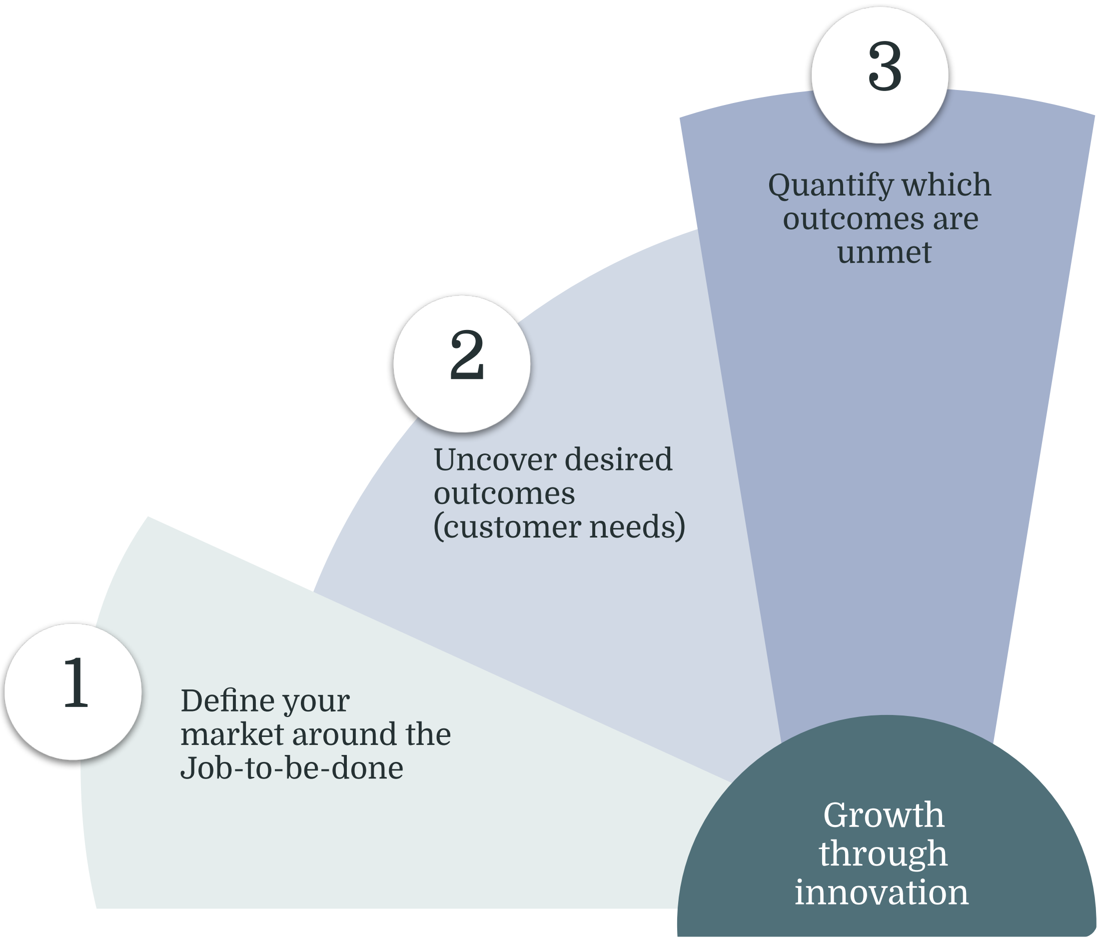
References

[28] Ulwick, A. W. (2016). Jobs to Be Done: Theory to Practice. Paperback, October 28, 2016. Retrieved from https://www.amazon.com/Jobs-be-Done-Theory-Practice/dp/B0CCZV8JJH/

[31] Ulwick, Tony. “Defining Customer Needs: The Root of the Problem.” AnthonyUlwick.com, 7 Jan. 2017. Available at: https://anthonyulwick.com/2017/01/07/defining-customer-needs-root-problem/

QUANTIFY WHICH OUTCOMES ARE UNMET

Section 4 Overview



Quantify which outcomes are unmet

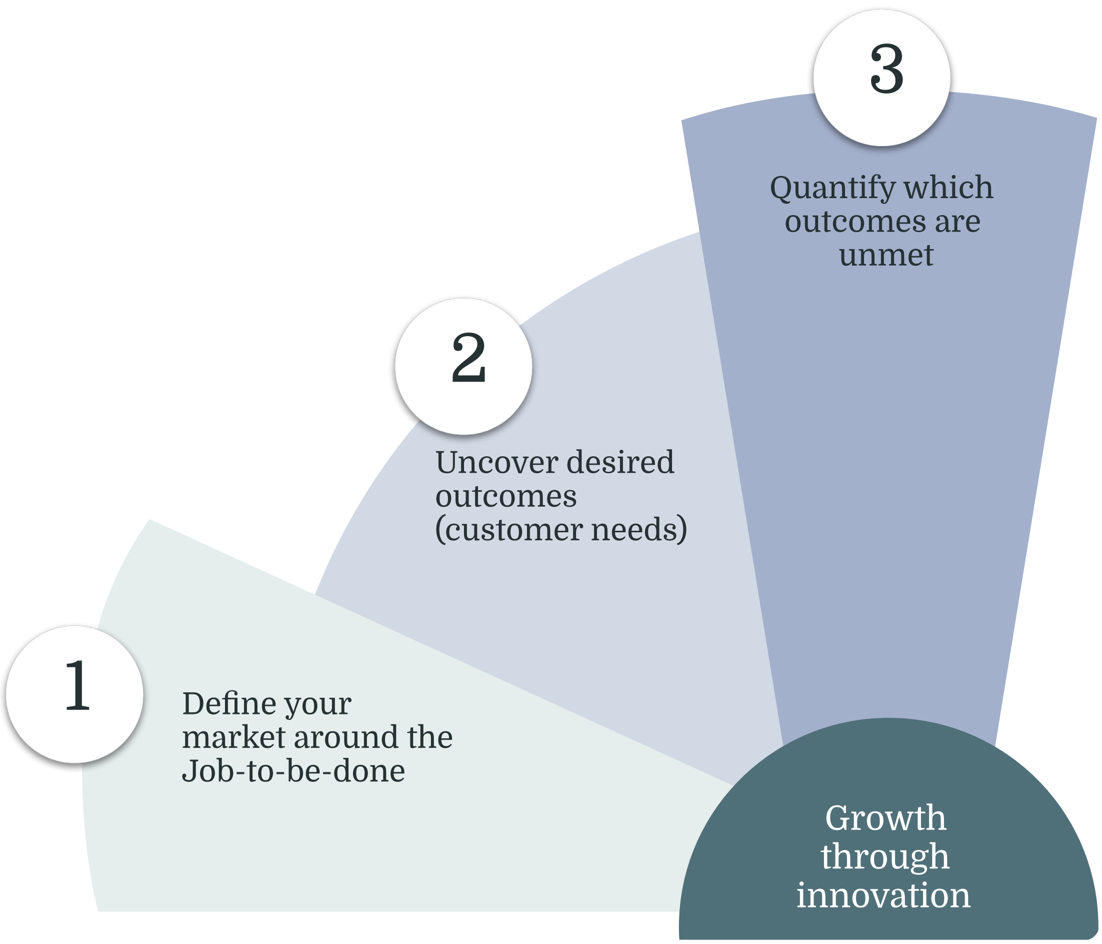
Section Four covers step 3 in the ODI process called, Quantify which needs/outcomes are unmet.

This section has arguably the most controversial chapter in the book, chapter 7, where I look to critique the ODI method of quantification. In chapter 7, I highlight some concerns I have with the opportunity scoring algorithm used by Strategyn, surveys that can potentially 50-100+ questions, and handling the different biases use to the underlying methodology.

Chapter 8 offers one possible alternative approach to the ODI quantification methods by utilizing Maximum difference scaling (best-worst scaling) or MaxDiff. This approach as several positive aspects that make it quite appealing to product and researchers looking to implement the jobs-to-be-done and outcome-driven approach without all the underlying concerns that are outlined in chapter 7.

By the end of this section, you will have all the knowledge to know the practical limitations of the quant methods used by Strategyn and the original ODI approach, plus know an alternative that you might find practical to try out yourself.

Chapter 7: The Problems with Traditional JTBD Quantification



Quantify which outcomes are unmet

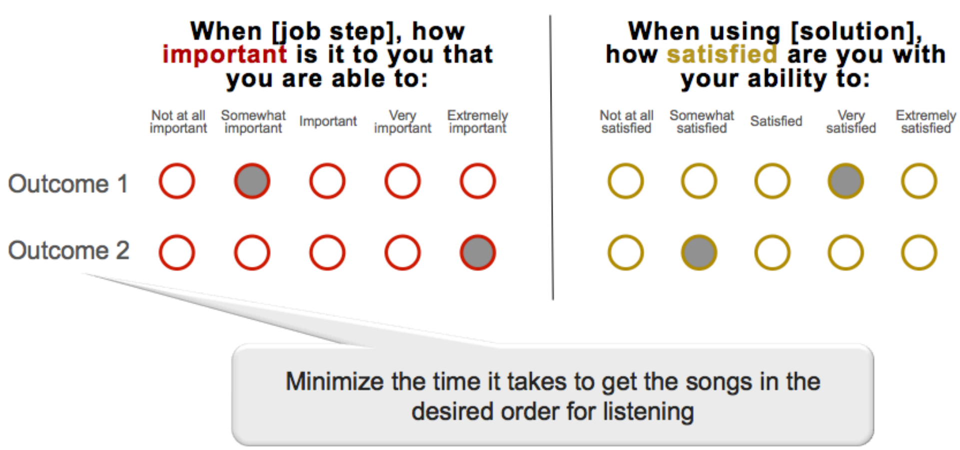
This chapter will potentially be the most controversial part of the entire book. After you have identified through interviews, secondary research, internal organizational studies, or other data sources, you will have a long list of needs. Ranging from needs from job steps, emotional or social needs, financial needs, complexity factors, and consumption chain jobs that need to be quantified. The next logical question is how do you quantify all of this?

The traditional ODI approach has an answer for this through survey research and statistical analysis. While the established ODI approach offers one approach for quantification, it comes with drawbacks that teams need to understand before trying it themselves.

The original ODI quant methods often fall short when they meet the real world of survey fatigue and the practical constraints most teams face. This chapter will walk you through the established ODI approach, examine the limitations, and prepare you to evaluate alternatives that may be more advantageous.

The Outcome-Driven Innovation Survey Approach

The ODI survey approach that was started by Tony Ulwick uses a 5-point likert rating scale questions with importance and satisfaction being the main focus. Let's examine this approach to understand both its underlying logic.



Importance and Satisfaction Likert Scale

The ODI approach uses Likert scales because it's designed to measure **two distinct dimensions simultaneously**: **importance and satisfaction**. This dual measurement is the foundation for the **opportunity scoring algorithm** that calculates where the Strategyn argues are the biggest gaps exist between what customers want and what they currently get.

Likert scales also allow ODI to treat each need as an independent variable. Every need gets its own importance and satisfaction rating, which means you can theoretically identify opportunities across your entire list without forcing customers to choose between needs that might not compete with each other in their minds.

The approach **assumes customers can accurately self-report both how much they care about something and how well their current solutions perform**. This assumption works well for functional needs where customers have direct experience, but becomes more questionable for emotional or social needs where self-awareness may be limited.

The downside is that this approach often leads to surveys that, thanks to the comprehensive needs list plus necessary screener and demographic questions, can easily end up being **50-125+ questions** (which obviously has downsides and is not possible for many organizations to field effectively).

Tony has said, "We create a survey. It might have 100-150 different need statements in it, and we've created a way to get all those inputs from customers in a pretty quick period of time. We've been using similar techniques for 25 years, so we make it better and better over time.[11]

We try to make sure the questionnaires are done in 25-30 minutes, which is fairly lengthy, but we often pay people to take the surveys and have good quality control checks to make sure that people aren't finishing a 25-minute survey in five minutes.

What we've discovered is that in most markets, maybe 10 to 15 percent of those taking surveys are fudging their way through the data sets, but they are eliminated, so we don't worry about that. From that good set of data we can figure out which of the needs are important and unsatisfied " [2]

Before critiquing the approach further, we need to understand the mathematical approach that serves as the foundation for the entire ODI methodology. The opportunity scoring algorithm is what turns your raw importance and satisfaction ratings into what Strateygn argues as clear priorities. it's the reason ODI surveys are structured the way they are.

Understanding how this algorithm works, and more critically, what assumptions it makes is necessary before evaluating alternatives. Let's look at the scoring algorithm in the next section.

Understanding the Opportunity Scoring Algorithm

In Tony's own words, "This formula reveals which customer needs are most important and least satisfied the ones that represent the best opportunities for growth."[1]

> *It's worth noting that, there is **never** one single metric that answers all of your business questions. There is no silver bullet. Chris Chapman, former Principal Quant UXR @ Google, Amazon, Microsoft, and Director of the Quant Conference wrote a great blog post titled, North Star... a path to being lost on this topic.*

The actual calculation involves a conversion step that often confuses newcomers. While an individual customer rates a need on a scale of 1 to 5, the algorithm converts these individual responses into a standardized aggregate score out of 10.

This is why you will see final Opportunity Scores that go up to 20, even though the survey scale only went up to 5.

- Importance Score = (respondents rating 4 or 5) ÷ (total respondents) × 10
- Satisfaction Score = (respondents rating 4 or 5) ÷ (total respondents) × 10

- Opportunity Score = Importance + max(0, Importance - Satisfaction)

To understand how this works in practice, let's imagine we're researching the customer "job" of **planning a family vacation**. A key part of this job is booking flights. We're not focused on any specific solution like a website or an app; we're focused on the customer's underlying goal.

Step 1: Creating the Initial Scores

First, the algorithm converts raw survey responses into standardized 10-point scores for both **Importance** and **Satisfaction**. It uses a "top-2-box" method, counting the percentage of people who rated an need a 4 or 5 on the survey's 5-point scale.

During our research, we find one of the key needs customers want is to **"minimize the time it takes to find flight options that fit my budget and schedule."** This is a solution-agnostic need because it describes the desired result without mentioning any specific tool or feature.

- **Importance**: Let's say 270 out of 300 travelers rated this need as highly important (a 4 or 5). The math to create the standardized score would be: `(270 ÷ 300) × 10 = 0.9 × 10`, which gives you an **Importance Score of 9**.
- **Satisfaction**: However, when asked how satisfied they are with their ability to do this quickly using current tools, only 90 of those 300 travelers were highly satisfied. The Satisfaction Score would be: `(90 ÷ 300) × 10 = 0.3 × 10`, which gives you a **Satisfaction Score of 3**.

Step 2: Calculating the Final Opportunity Score

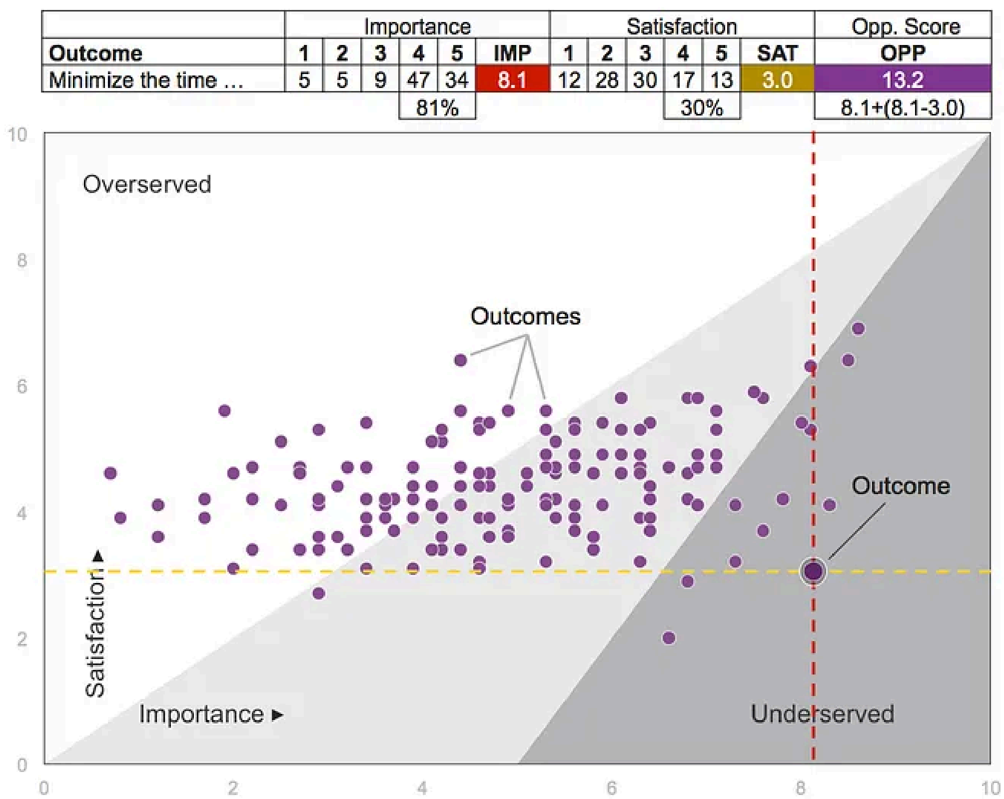
Once you have these standardized scores, they get plugged into the final formula: `Opportunity Score = Importance + max(0, Importance − Satisfaction)`.

In simple terms, the final score is the **Importance score plus any gap** where importance is higher than satisfaction. For our vacation planning example, with an Importance of 9 and a Satisfaction of 3, the calculation is `9 + (9 − 3)`, which results in a final **Opportunity Score of 15**.

Ok, let's see how Strategyn uses these importance, satisfaction, and opportunity scores in their underlying methodology they promote.

The Opportunity Landscape

According to Strategyn, the real insight of this quantification and scoring is when you plot these importance and satisfaction scores on what Strategyn calls the Opportunity Landscape.



Opportunity Landscape

This scatter plot puts importance on the horizontal axis and satisfaction on the vertical axis, creating a visual map that reveals where teams should focus.

Needs/outcomes that fall in the bottom-right are "underserved" which are highly important to customers but poorly satisfied by current solutions. These are your "innovation" opportunities, the needs that should drive your product roadmap and receive the bulk of your innovation investment.

Needs in the top-left are "overserved". These are well-satisfied but not particularly important to customers, which might indicate feature bloat or resources being allocated to things customers don't value.

The diagonal line running from bottom-left to top-right represents the boundary where importance equals satisfaction. Needs below this line have satisfaction gaps and represent potential opportunities, while those above it are performing better than their importance level would suggest. The further an need sits from this line toward the underserved quadrant, the higher its opportunity score and the more compelling the business case for addressing it.

This visualization helps turn abstract JTBD and ODI survey data into something *concrete* that product teams can act on. Crucially, the data collected for the Opportunity Landscape serves as the direct foundation for Strategyn's "Needs-Based Segmentation." The methodology typically utilizes cluster analysis on the opportunity scores to identify groups of customers who value different needs.

However, this creates a downstream risk regarding data quality. Effective segmentation relies on capturing nuance, specifically the variance in how different people rate different needs. If the input data is compromised by the survey fatigue described earlier where respondents straight-line their answers, or if the nuance is flattened by a "Top-Two-Box" approach that treats distinct preferences as identical, the clustering algorithm will fail to detect real behaviors.

When you feed low-fidelity or fatigue-biased data into a clustering algorithm, the math will still force a result. It will create segments based on statistical noise rather than actual market differences. Teams risk allocating resources to "phantom segments" which are groups that look distinct on a spreadsheet but do not exist in the real world.

A Critique of the Opportunity Algorithm

The ODI opportunity scoring algorithm presents itself as a systematic, objective method for identifying unmet needs and innovation opportunities, but from a research methodology perspective, it contains several fundamental flaws that undermine its reliability and validity. While the approach offers businesses the clean, potentially actionable outputs they need, these come at the cost of statistical rigor that may lead to misguided insights.

The Double-Weighting Problem

The biggest issue lies in the algorithm's core formula. By definition, the formula counts the Importance score twice: once as the base, and again as part of the gap calculation.

$$Opportunity = \mathbf{Importance} + (\mathbf{Importance} - Satisfaction)$$

This isn't just a quirk of arithmetic; it is a structural bias that ensures **Importance will always dominate the Satisfaction Gap.** The algorithm effectively decides that a minor annoyance in a "very important" task is more essential to solve than a complete failure in a "moderately important" task.

Let's look at how this distorts prioritization using our vacation planning example:

- **Need A (High Importance, Moderate Problem):** "Minimize time to find flights."
  - Importance: 9 | Satisfaction: 6
  - *The Gap is 3.*
  - **Opportunity Score: 12**
- **Need B (Medium Importance, Total Failure):** "Minimize risk of hidden fees."
  - Importance: 6 | Satisfaction: 0
  - *The Gap is 6.*

- **Opportunity Score: 12**

The Critique: Look closely at the results. Customers are **twice as frustrated** with Need B (a 6-point gap) compared to Need A (a distinct but smaller 3-point gap). Yet, the algorithm rates them as identical opportunities.

By double-counting Importance, the ODI method systematically suppresses "low-hanging fruit" problems that are extremely annoying to customers but related to slightly less important tasks. It forces teams to chase marginal improvements in high-traffic areas while ignoring broken experiences elsewhere, simply because the math says importance matters more than frustration.

Top-Two-Box Problem

The second major flaw is the methodology's reliance on **"top-two-box" analysis**, where scores are calculated by converting the 5-point scale into a simple "high/not high" binary. This approach treats a respondent who feels a need is absolutely critical (a rating of '5') the same as someone who feels it's just neutral (a rating of '3'), discarding information in the process.

This method has been largely abandoned in other fields for this reason. As Gerry Katz points out, consumer goods researchers found that customers who "definitely" intended to buy a product (a '5') were often **five times more likely** to actually make a purchase than those who "probably" intended to buy (a '4'). Lumping these distinct levels of intent together, as ODI's scoring does, masks the true urgency and passion customers feel. This loss of nuance, combined with an arbitrary cutoff point for what counts as "high importance," can hide the very opportunities teams are trying to find.

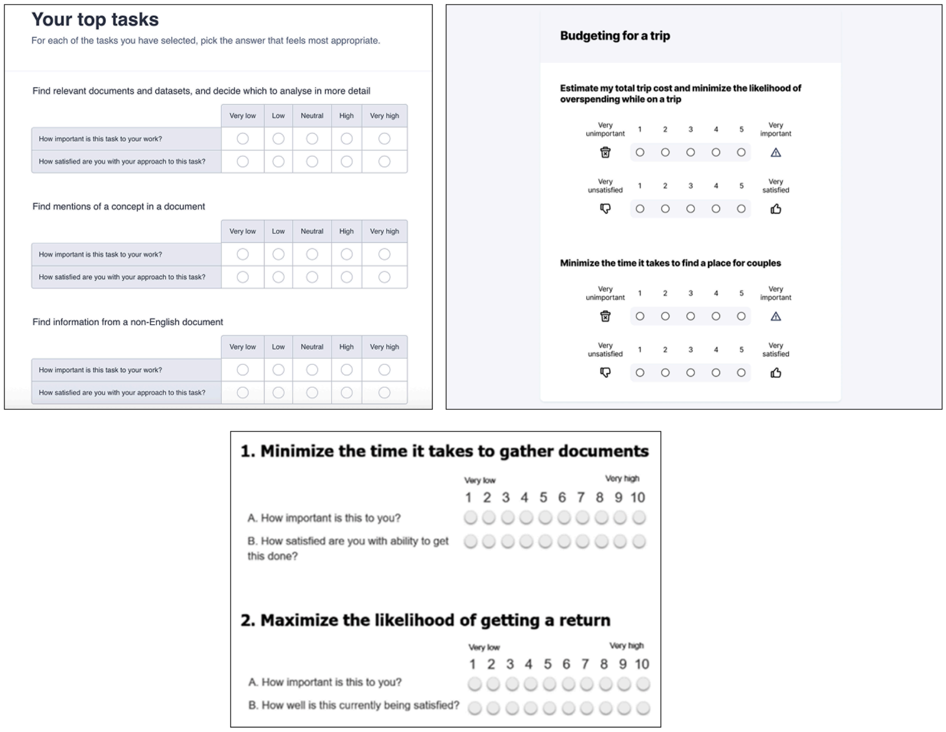
Based on their writings, authors Jeff Sauro and Jim Lewis would strongly agree with this assessment and add the following statistical concerns:

- **It "Dilutes" the Predictive Signal.** They argue that the single **top-box** (only the '5's) is the more predictive metric for predicting behavior because it isolates the most passionate customers. By including the '4's, the ODI method dilutes signal with more moderate, less predictive feelings, a point that directly supports the example from Gerry Katz. They state, "Because measurements of extreme responses tend to be better predictors of future behavior than tepid responses, we prefer top-box to top-two-box measurements".[33]
- **It Offers Little Advantage Over the Mean.** A point they would add is that top-two-box scores are often **highly correlated with the simple mean**. In one analysis, they found a correlation of .97 between the mean and top-two-box scores, meaning the shared variance is 94% (Sauro & Lewis, 2024). This indicates the top-two-box score provides virtually the same information as the mean, while being statistically less precise and losing the unique predictive power found in the less-correlated single top-box score.
- **It Causes a Major Loss of Information and Precision.** This is the core problem with this approach. They would emphasize that by converting the 1-5 scale to a binary metric, the algorithm treats a '1' the same as a '3', discarding vast amounts of information. This isn't just a theoretical problem—it leads to tangible negative consequences like **wider margins of error** and the need for **larger sample sizes** to achieve statistical confidence.[32]

The Survey Fatigue Problem

A major challenge in the ODI survey approach is the survey burden it places on respondents. Strategyn and Tony Ulwick advocate for surveys that include "100 or more desired need statements," as stated directly on their website. They claim that "knowing which of the 100 or more desired needs are most important and least satisfied pinpoints the opportunities for value creation" and recommend surveying "anywhere from 120 to 1200 customers, asking them to tell us the importance of each need and their current level of satisfaction." [2]

From a survey science perspective, this approach presents validity challenges. Asking respondents to evaluate 100 or more needs, where each need requires both an importance and a satisfaction rating, creates a high cognitive burden. When combined with other demographic and screener questions, the length of these surveys often leads to survey fatigue and reduced response quality.



Examples of JTBD and ODI Surveys

The volume of questions forces the survey design into a format known as "Matrix Grids" where rows of needs intersect with columns of ratings. While this looks organized on a researcher's screen, it creates a difficult user experience for the respondent. On mobile devices, where roughly 50% of survey traffic now originates, these grids often require pinching and horizontal scrolling to view. This friction frequently leads to a behavior called "straight-lining" where a fatigued respondent simply clicks the same column, such as voting "4" for every single item, all the way down the page just to reach the next section. This creates data that looks complete but lacks distinct signal.

To mitigate these data quality risks, Firms like Strategyn often move away from standard online panels and utilize high-touch methods like CATI (Computer-Assisted Telephone Interviewing) or managed CAVI (Computer-Assisted Web Interviewing). In a CATI approach, a human interviewer calls the respondent and verbally guides them through the survey to record the answers for them. In managed CAVI, respondents might be recruited into a live session or a supervised environment to ensure they are paying attention.

While these methods effectively reduce straight-lining and robotic behavior, they introduce substantial barriers for modern product teams.

High Cost: Utilizing human interviewers and managed panels increases the cost per response. While a standard online survey might cost a few dollars per respondent, CATI and managed CAVI approaches can raise budgets into the tens of thousands of dollars for a single study. This places this type of research out of reach for many organizations.

Persisting Cognitive Load: Having a human interviewer read the questions does not solve the underlying design flaw. Even if the respondent is engaged, asking them to cognitively evaluate 100 separate items is mentally exhausting. By the 80th question, the respondent's ability to discern nuances between options like "Somewhat Important" and "Very Important" has likely degraded, regardless of who is asking the question.

Velocity: Most product teams operate in agile environments requiring continuous discovery and quick feedback loops. Setting up a formal CATI study with recruited experts and phone banks often takes weeks or months. This slower process is rarely practical for teams that need to validate hypotheses and move forward in days rather than quarters.

> ***Author's Note***: *I often ask teams who are adamant about using this methodology if they have ever taken a 50+ question survey themselves. If they say yes, I ask about their honest experience: Did they maintain focus, or did they start "straight-lining" answers to finish faster? No matter how relevant the topic, there is a limit to how much cognitive load a respondent can handle. If they say no, I offer a challenge: Build a draft ODI survey and spend the full 25 minutes taking it yourself. Before asking customers to endure that experience, you should verify if it is an experience you are willing to endure.*

Bias Amplification:

Good research methodology attempts to account for systematic biases in self-reported data, but the ODI algorithm amplifies these distortions rather than correcting for them. Rating scale bias represents a fundamental challenge for any survey-based approach, and the ODI methodology is particularly vulnerable to these systematic measurement errors.

Several biases are especially problematic for importance and satisfaction ratings in the ODI context. **Acquiescence bias** leads respondents to systematically rate needs as more important than they actually are, particularly when need statements are framed positively (as they typically are in ODI surveys). This inflates importance scores across the board, making it harder to distinguish truly underserved needs from merely desirable ones.

Social desirability bias particularly affects importance ratings when customers feel pressure to appear rational or knowledgeable. For example, customers might overstate the importance of "data security" or "environmental sustainability" because these sound like things a responsible person should care about, even if they don't actually influence their purchase decisions. This systematic inflation of certain types of importance ratings skews the entire opportunity landscape.

Negativity bias in satisfaction ratings compounds the problem from the other direction. Research shows that customers are naturally more likely to remember and report negative experiences than positive ones, leading to systematically deflated satisfaction scores. When combined with inflated importance ratings, this creates artificially large satisfaction gaps that don't reflect actual market opportunities.

Cultural response style differences create additional systematic distortions that the algorithm treats as valid signal rather than measurement noise. Some cultural groups exhibit extreme response tendencies (gravitating toward 1s and 5s), while others show central tendency bias (clustering around 3s). When these different response patterns get mixed in the same dataset and processed through the opportunity algorithm, cultural differences in rating behavior can masquerade as meaningful differences in customer needs.

The ODI approach's reliance on importance and satisfaction ratings makes it especially susceptible to these systematic biases.

Validation Gap Section:

A fundamental issue underlying all these concerns is the lack of empirical validation for the algorithm itself, though this limitation exists within a broader context of business framework validation that deserves acknowledgment.

To be fair, the absence of rigorous validation is not unique to ODI. Many widely-adopted business frameworks operate without comprehensive empirical validation of their core assumptions. Net Promoter Score, despite extensive criticism from statisticians, continues to provide organizational *value* through its simplicity and ability to focus teams on customer advocacy. The Boston Consulting Group's Growth-Share Matrix lacks empirical validation for its strategic recommendations, yet remains a useful strategic thinking tool. Even fundamental approaches like market segmentation rarely undergo rigorous validation of their predictive power for business needs.

The difference with ODI, however, lies in both the specificity of its mathematical claims and the magnitude of investment decisions it drives. While NPS functions primarily as a tracking metric and the Growth-Share Matrix serves as a strategic thinking framework, ODI explicitly positions itself as a precise method for identifying innovation opportunities that warrant major resource allocation. Strategyn's marketing claims, such as their assertion of "an 86 percent success rate, a five-fold improvement over the industry average," present the methodology as scientifically validated when no such validation has been provided.

More problematically, the mathematical specificity of the opportunity algorithm creates an illusion of precision that can be misleading. When a framework produces scores like 14.7 versus 12.3 for different needs, it implies a level of measurement accuracy that the underlying methodology simply cannot support. This false precision becomes particularly dangerous when teams use small differences in opportunity scores to make major strategic decisions about where to invest resources.

The lack of validation also means there's no evidence that the formula predicts successful opportunities better than simpler alternatives. Would a formula that weighted satisfaction gaps more heavily perform better? Would treating importance and satisfaction equally produce more reliable results? Without empirical testing, these remain open questions.

What's needed isn't necessarily the same level of statistical rigor required for academic research, but rather **transparency** about the methodology's limitations and some form of cross-validation against business outcomes. Even simple retrospective analyses comparing ODI-driven innovation decisions against their market performance would provide valuable insight into where the approach works well and where it might lead teams astray.

The Actionability Problem: What Are Teams Supposed to Do With This?

Set aside the statistical concerns for a moment. A practical question remains: what are product teams actually supposed to do with these opportunity scores?

Consider the hypothetical analysis below, which shows realistic outputs from an ODI study for a music streaming application.

Product Opportunity Analysis Report

Definitions

Formula
Opportunity Score = Importance + Max(Importance - Satisfaction, 0)

Scoring Legend
> 15: Extreme Opportunity (Critical to solve immediately)
12 – 15: High Opportunity (Low hanging fruit)
10 – 12: Worthy of Consideration (Monitor closely)
< 10: Appropriately / Overserved (Do not invest / Maintain only)

Data Clusters

Theme: Discovery & Curation (Underserved)

Desired Outcome	Importance	Satisfaction	Calculation	Opportunity Score	Status
Minimize the time it takes to find new music that fits a specific mood	9.4	3.2	9.4 + (6.2)	15.6	Extreme Opportunity
Minimize the likelihood of a playlist containing a song that disrupts the vibe	8.8	4.1	8.8 + (4.7)	13.5	High Opportunity
Minimize the time it takes to organize songs into a coherent playlist	8.1	3.0	8.1 + (5.1)	13.2	High Opportunity
Minimize the likelihood of hearing the same 'recommended' track repeatedly	8.5	4.5	8.5 + (4.0)	12.5	High Opportunity
Minimize the steps required to rediscover a song heard in passing	7.9	4.0	7.9 + (3.9)	11.8	Worthy of Consideration

Theme: Playback & Technical Performance (Overserved)

Desired Outcome	Importance	Satisfaction	Calculation	Opportunity Score	Status
Minimize the likelihood of the app crashing during playback	9.8	9.7	9.8 + (0.1)	9.9	Appropriately Served
Minimize the time it takes for a selected song to begin playing	9.5	9.1	9.5 + (0.4)	9.9	Appropriately Served
Minimize the time it takes to find a specific song by exact title	9.2	8.8	9.2 + (0.4)	9.6	Appropriately Served
Minimize the steps required to pause or skip a track	7.5	8.5	7.5 + (0) [Sat > Imp]	7.5	Overserved
Minimize the likelihood of sound distortion at high volume	5.0	9.0	5.0 + (0) [Sat > Imp]	5.0	Overserved

Illustrative: Spotify Needs Scoring Example

Looking at the "Discovery & Curation" theme, teams might feel optimistic at first. The methodology has identified a clear winner: "Minimize the time it takes to find new music that fits a specific mood" scores 15.6, classified as an "Extreme Opportunity." But what happens when you try to turn this into a roadmap?

The clustering problem. Three needs—avoiding playlist disruption (13.5), organizing songs into coherent playlists (13.2), and reducing repeated recommendations (12.5), all fall within the "High Opportunity" band. Their scores differ by only 1.0 to 1.3 points. Given the measurement error in survey data, can anyone confidently claim these represent different priorities? Is a 13.5 meaningfully distinct from a 12.5 when both numbers come from biased ratings processed through a formula that double-weights importance?

The "now what?" meeting. Imagine presenting this to a product team. Someone will ask: "Should we tackle the 15.6 first, or would we get more value from addressing the cluster of 12-13 point needs together?" The methodology doesn't answer this. The algorithm produces ranked numbers, but nothing in the framework helps teams understand whether a 2-point difference justifies different investment levels or whether adjacent needs should be bundled into a single initiative.

The false precision trap. The scores create a false sense of precision. Product managers (or other stakeholders) look at "15.6 versus 13.5" and instinctively treat these as exact measurements, like comparing prices or conversion rates. But these numbers emerged from self-reported ratings, converted through top-two-box analysis, and processed through an algorithm with known biases. The precision of the output overstates the precision of the underlying data.

The overserved paradox. Look at the "Playback & Technical Performance" theme. These needs all score below 10, deemed "Appropriately Served" or "Overserved." The methodology says: don't invest here, maintain. But if competitors are equally strong in these areas, there may be no differentiation opportunity. And if the market shifted tomorrow—if a new entrant introduced reliability problems or usage patterns changed—these "overserved" areas could become important retention factors. ODI scoring is a snapshot. It provides no insight into competitive dynamics or future risk.

The interpretation burden. Notice how much work falls on whoever presents this data. They must explain the scoring methodology, defend the cutoff thresholds, justify why a 15.6 warrants immediate action while a 12.5 merely requires monitoring, and somehow translate "minimize the likelihood of a playlist containing a song that disrupts the vibe" into actual product features. The framework identifies problems but offers no bridge to solutions.

The result: teams invest substantial resources into quantitative research hoping for clear direction, only to find themselves in the same prioritization debates they would have had without the data.

The deeper issue is that ODI quantification tries to reduce complex product decisions to a single ranked list. Real product strategy requires understanding relationships between needs, evaluating technical feasibility, considering competitive positioning, and assessing organizational capabilities—none of which appear in an opportunity score. Teams need frameworks that inform these conversations, not numbers that claim to resolve them.

Summary of Concerns with the ODI Opportunity Algorithm Approach

Mathematical and Statistical Issues

- Double-weighting of importance in the formula creates an untested assumption that importance should dominate over satisfaction gaps
- Top-two-box analysis discards valuable information by converting continuous scales into binary categories
- Arbitrary cutoff points (4+ considered "high") affect results with no principled justification
- Percentage-based scoring creates volatility and instability, especially with smaller sample sizes
- No empirical validation that this formula predicts innovation success better than alternatives

Methodological Biases

- Algorithm amplifies systematic response biases rather than correcting for them
- Social desirability bias gets magnified when customers overstate importance of things they think they should care about
- Natural negativity bias in satisfaction ratings gets amplified in opportunity calculations
- No statistical adjustments for known patterns in self-reported data

Structural Design Problems

- Independence assumption ignores interconnected nature of customer experiences
- Treats needs as separate variables when many are causally related or represent trade-offs

- No mechanism to identify when improving one need might negatively impact another
- Missing factor analysis or relationship modeling between related needs

Survey Design and Quality Issues

- Surveys with 100+ need statements create severe respondent burden
- Survey fatigue virtually guaranteed with such lengthy evaluations
- Matrix questions rating 100+ items likely take 30-45 minutes to complete thoughtfully
- Poor response quality from satisficing behaviors, straight-lining, or survey abandonment
- Compromised data quality undermines entire analytical framework regardless of mathematical sophistication

Transparency and Validation Gaps

- No evidence presented that the approach actually predicts successful innovation opportunities
- Methodology appears designed to produce clean outputs rather than accurate modeling
- Presents statistically problematic methods as scientifically rigorous
- Creates false confidence in potentially flawed conclusions
- Lacks cross-validation against actual market performance or business needs

Fundamental Conceptual Issues

- Represents "looking scientific" rather than "being scientific"

- Systematic bias that looks rigorous on the surface
- Doesn't acknowledge trade-offs between business utility and statistical validity
- May lead to misguided innovation investments based on methodologically flawed prioritization

Actionability and Interpretation Challenges

- Opportunity scores cluster together, making differentiation difficult
- Small point differences (1-2 points) drive major strategic decisions despite measurement uncertainty
- Framework identifies problems but provides no path to solutions
- Teams end up in the same prioritization debates they would have had without the data

Given these fundamental methodological concerns, organizations need practical alternatives that address these concerns while still providing actionable insights for teams. The next chapter explores a different approach that maintains practical utility while avoiding the statistical pitfalls and response quality issues that plague the traditional ODI methodology.

Chapter 7 Conclusion

The ODI opportunity scoring algorithm represents a well-intentioned attempt to bring systematic rigor to innovation and needs prioritization, but my analysis reveals fundamental methodological flaws that can lead teams toward misguided investment decisions. The double-weighting of importance, information loss through top-two-box analysis, amplification of systematic biases, and heavy survey burden creates a mix of reliability issues.

This doesn't mean quantification of customer needs is impossible or undesirable. Businesses absolutely need systematic methods for prioritizing innovation opportunities, and the intuitive appeal of measuring both importance and satisfaction gaps points toward genuine customer insight needs. The problem isn't with the goal of quantification, but with this particular approach to achieving it.

The challenge moving forward is developing methods that maintain the business utility that makes ODI attractive while addressing its statistical and methodological shortcomings. We need approaches that can handle the practical constraints real organizations face. Such as limited survey response rates, budget restrictions, time pressures, and the cognitive limits of actual customers, without sacrificing the reliability needed for sound decisions.

Fortunately, several alternative approaches exist that can provide actionable prioritization insights without falling into ODI's methodological traps. Some focus on more sophisticated statistical techniques that account for response biases and need relationships. Others take entirely different approaches to quantification that reduce survey burden while improving data quality. Still others combine quantitative and qualitative methods to create more robust insight frameworks.

The next chapter explores these practical alternatives, examining approaches that teams are actually using successfully to quantify customer needs and prioritize innovation opportunities. Rather than throwing out quantification entirely, we'll look at methods that acknowledge the complexity of customer needs while still producing the clear, actionable outputs that innovation teams require. The goal isn't perfect measurement, it's reliable enough measurement that leads to better decisions than intuition alone.

Chapter 7 References

[11] Traynor, Des. “Strategyn’s Tony Ulwick on Jobs-to-be-Done.” Intercom Blog (Podcast), 10 Dec. 2015. Available at: https://www.intercom.com/blog/podcasts/podcast-tony-ulwick-on-jobs-to-be-done/

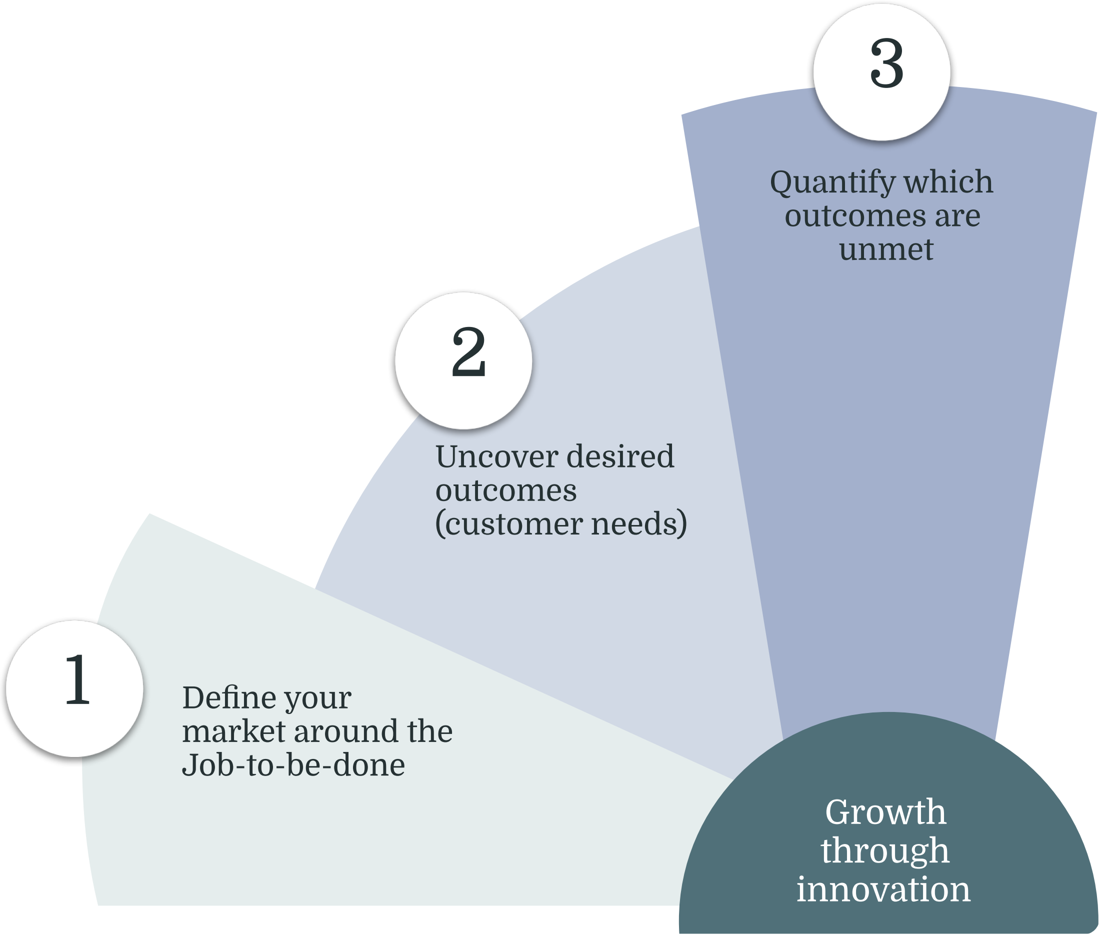
[32] Sauro, Jeff. “Are Top Box Scores a Better Predictor of Behavior?” MeasuringU.com, 2 May 2018. Available at: https://measuringu.com/top-box-behavior/

[33] Sauro, Jeff, and Jim Lewis. “Top Box, Top-Two Box, Bottom Box, or Net Box?” MeasuringU.com, 4 June 2024. Available at: https://measuringu.com/top-top-two-bottom-net-box/

QUANTIFY WHICH OUTCOMES ARE UNMET

Chapter 8: A Practical Alternative - MaxDiff

This chapter introduces Maximum Difference Scaling, or MaxDiff, as a reliable and user-friendly method for figuring out what your customers' unmet needs are. We'll show you how it directly solves the problems with the old opportunity algorithm we discussed in Chapter 7 and give you a complete hands-on guide to using it for your own research.



Quantify which outcomes are unmet

Review of Chapter 7

In the last chapter, we saw how the traditional opportunity scoring algorithm, while well-intentioned, has some potential concerns. Double weighting importance, losing information by grouping ratings, and asking people too many questions all lead to unreliable results. At best, this can hide what your customers really care about. At worst, it can send your team chasing after phantom opportunities based on survey noise. This leaves us with a key question: **how can we figure out customer needs in a way that is both statistically sound and practical**?

The answer is to stop asking customers for abstract ratings and start asking them to make realistic trade offs. This is the simple idea behind a method called **Maximum Difference Scaling**, or **MaxDiff**. Instead of asking a customer to rate the importance of 100 different need/outcome statements on a five point scale, a mentally draining task, MaxDiff presents a much simpler request. It shows people small sets of items and asks them to make a straightforward choice: "Of this list, which is the most important and which is the least important?"

This one change fixes the major flaws of the old approach. By forcing a choice, MaxDiff avoids the biases that come with rating scales and gives you a true hierarchy of priorities. It mimics how people make decisions in the real world by comparing options and deciding what matters more. The result is a cleaner, more reliable, and more precise picture of what your customers truly value.

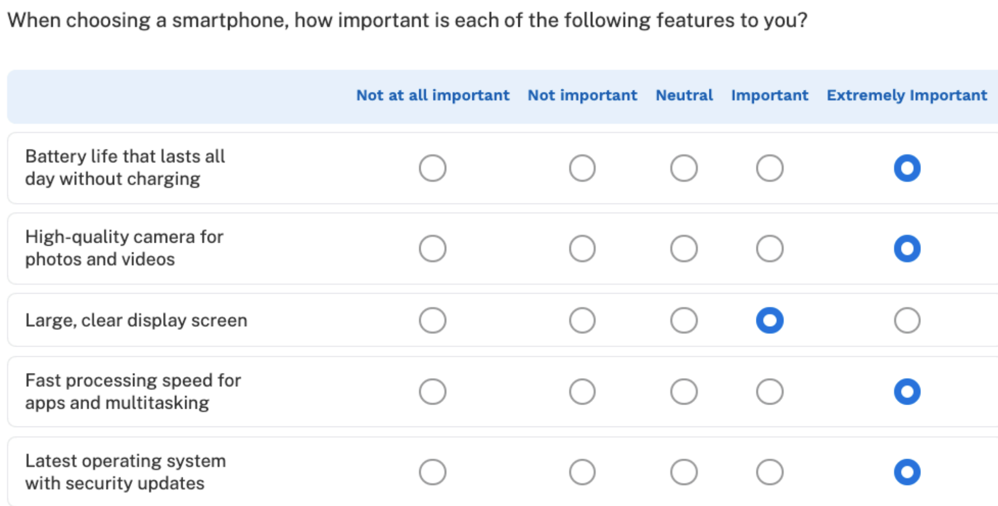
This chapter is your hands on guide to MaxDiff. We will walk step by step through how to design, run, and analyze a MaxDiff study for your JTBD and ODI research.

What is MaxDiff? A Simple Explanation

At its heart, MaxDiff is a way to understand what really matters to people by asking them to make simple, repeated choices. It breaks down the overwhelming task of rating many items into something much more manageable and human.[37]

The Basic Idea

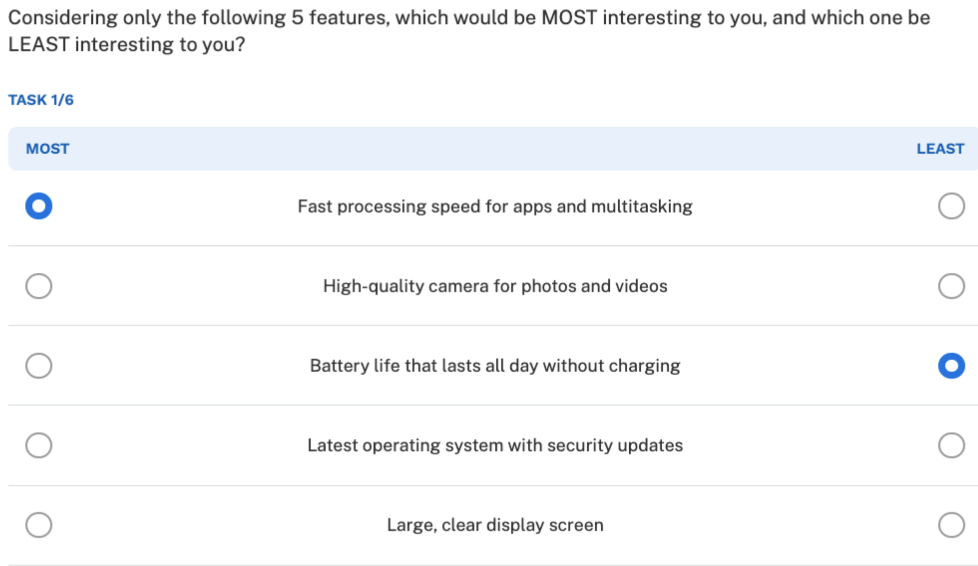
Imagine you want to know which iPhone features a group of friends thinks are most important. The old survey approach would be to list several features and ask your friends to rate each one on a scale of 1 to 5.



Example of how someone might prioritize features with likert scales

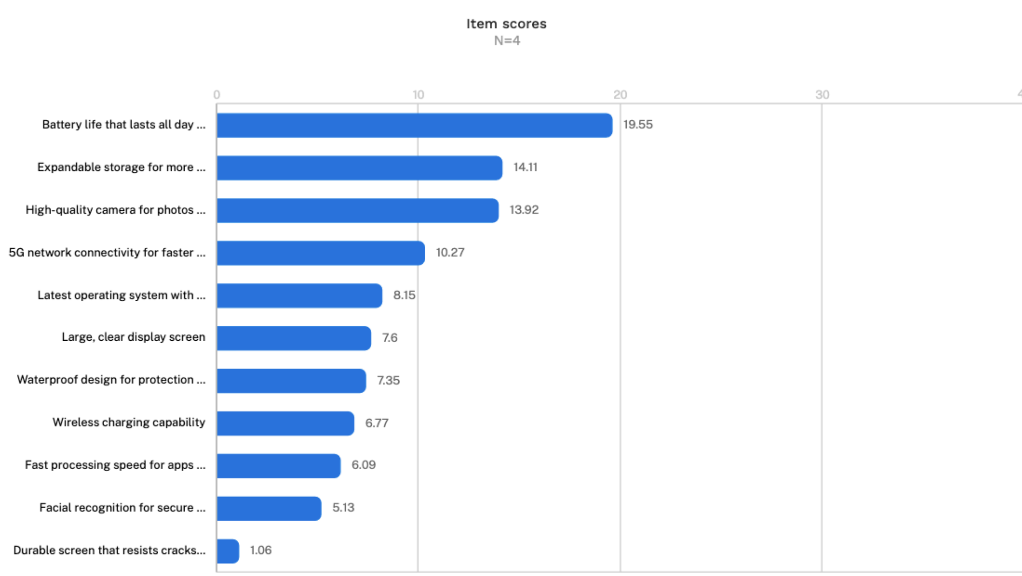
You would almost certainly get a lot of 4s and 5s for popular items like "battery life," "high quality camera," and "faster processor," but you wouldn't know which of those is the *most* important.

Formally known as Best-Worst Scaling and developed by Jordan Louviere, MaxDiff works differently. [24] Instead of that long list, you would show your friends just four or five features at a time and ask a simple question: "Of these options, which is the MOST interesting to you, and which is the LEAST interesting to you?"



iPhone MaxDiff Example

You would then show them a few more sets with different combinations of features. After a dozen or so of these simple choices, a Bayesian analysis running in the background can figure out a complete, ranked preference list for every single feature for each person. The final result is a chart that clearly shows what people value most.



Smartphone Features MaxDiff Scores

This chart shows what are called "item scores," which are basically preference rankings that come from all those best and worst choices. Here, "Battery life that lasts all day" is the clear winner with a score of 19.55. This immediately tells product managers where to focus. The data also shows smaller differences. For example, "Expandable storage" (14.11) and "High quality camera" (13.92) are close, suggesting they are competing for a similar level of customer interest. This level of detail helps teams make much smarter trade off decisions.

Why MaxDiff

So why is this simple method of choosing the best and worst so much better than rating scales? It turns out that MaxDiff directly solves the problems we identified in the last chapter.

- **It Avoids Unreliable Rating Scales.** People use rating scales differently. Some people are optimists who rate everything highly, while others avoid the extremes. MaxDiff gets rid of this problem because it relies on comparison, not abstract ratings. It doesn't matter what a "4" means to someone. What matters is what they pick as most and least important, which is a much more consistent and natural way for people to think.
- **It Reduces Survey Fatigue.** Rating 100 different items is boring and exhausting. As people get tired, the quality of their answers drops. MaxDiff turns this into a more engaging, puzzle like task. Each question is a new decision, which keeps people more focused. This leads to higher completion rates and better data from start to finish.
- **It Keeps the Math Simple and Clean.** The old opportunity formula mixed importance and satisfaction scores in statistically questionable ways. MaxDiff analysis, on the other hand, produces a single set of utility scores. These scores are on a relative, interval scale, meaning the distance between them is meaningful. For easier interpretation, these raw scores are often rescaled to a common scale, such as 0 to 100. When rescaled this way, an need with a score of 20 is twice as preferred as one with a score of 10, creating a clear and intuitive ranking of customer needs without any complex formulas.

- **It Uses All the Information.** Grouping ratings of 4 and 5 together, known as "top two box" analysis, discards valuable information. A passionate "5" gets treated the same as a lukewarm "4." MaxDiff uses every single choice a person makes to build its model. This allows it to make fine distinctions between needs that are close in importance.
- **It's More Efficient.** MaxDiff gives you more reliable results with fewer people. Because each person makes many choices, a 200 person MaxDiff study can generate as much useful data as a traditional rating study with 400 or 500 people. This makes your research faster and more affordable.

The Measurement Decision: What Should Your MaxDiff Actually Measure?

Before we dive into building a MaxDiff survey, we have to address a key decision. It depends on what dimension you ask people to evaluate.

Traditional ODI requires measuring both importance and satisfaction for every need, then combining them through the opportunity algorithm. We have already discussed the problems with that approach: survey fatigue from rating 100+ items twice, the double-weighting of importance in the formula, and the false precision of the resulting scores.

MaxDiff solves the fatigue problem by using forced choices instead of ratings. But you still face a decision about what dimension to measure, and this decision matters more than you might think.

The "Importance" Trap

When trying to understand customer needs, our first instinct is often to ask about importance: "Which of these needs is most important to you?" Unfortunately, this usually leads to a "ceiling effect," where almost everything is rated as highly important, and you cannot tell what to focus on.

For example, imagine a hospital trying to improve the patient experience. If they ask patients to prioritize needs like these:

- Having medical staff take my symptoms seriously
- Getting diagnostic test results quickly
- Understanding my treatment plan clearly
- Having nurses respond promptly to requests

Almost every patient would say all of these are "important." They are all fundamental to good healthcare. The results would be a flat, undifferentiated list of priorities, giving the hospital no direction on where to improve.

Undifferentiated needs from maxdiff

Undifferentiated needs maxdiff barchart example

In professional research, we call this the problem of "Stated Importance." When you ask people what they want, they say everything. This is why many researchers advocate moving toward "Derived Importance," where we uncover what matters by analyzing the choices people make rather than the ratings they give.

MaxDiff helps with this by forcing trade-offs, but the framing of your question still matters enormously. Ask people which needs are "most important" and you may still get clustering at the top. Ask them which problems are "most frustrating" or which improvements would "make the biggest difference" and you often get much cleaner separation.

To avoid the importance trap and get actionable data, you have four realistic approaches to consider.

Option 1: MaxDiff on Importance, Then Targeted Satisfaction

You run your MaxDiff study asking customers to identify which needs matter most to getting their job done. This gives you a clear hierarchy of importance. Then you add a short follow-up section (not a second MaxDiff, just 10-15 simple satisfaction rating questions) covering only the needs that ranked in your top tier.

This approach preserves the gap analysis logic of ODI while dramatically reducing survey burden. Instead of rating satisfaction on 100 items, customers only evaluate the 15-20 that the importance ranking identified as priorities. You get both dimensions without the exhaustion.

The downside is added survey length and complexity. Even a short satisfaction section adds time. You also need to design conditional logic so the satisfaction questions reflect each respondent's importance rankings, which requires more sophisticated survey programming.

Choose this approach if: You have a mature product with established value propositions and need to know not just where to innovate but what to protect. The extra survey complexity is worth it because a misstep (deprioritizing something that turns out to be table stakes) is costly.

Option 2: MaxDiff on Satisfaction Only

You run your MaxDiff asking customers to identify which needs are most poorly served by current solutions, or which problems are biggest. This captures dissatisfaction directly through forced choice.

Customers will not identify something as a major problem unless it matters to them. If someone does not care about a capability, they are unlikely to flag it as their biggest pain point even if current solutions handle it poorly. Dissatisfaction, the argument goes, implicitly signals importance.

This is often true, but not always. There are scenarios where satisfaction alone can mislead you.

A customer might express dissatisfaction with something they rarely use and do not actually value. They tried your dark mode once, thought it looked terrible, and now report dissatisfaction. But they never use dark mode and would not care if it improved. If you only measure satisfaction, this noise can look like signal.

More significantly, satisfaction-only measurement makes it difficult to identify your **table stakes**: the needs that are currently well-served. These are things you must not regress on. High satisfaction might tempt you to deprioritize maintenance or take quality for granted, not realizing that any degradation would cause damage. Without some importance signal, you lose visibility into what you need to protect, not just what you need to fix.

Choose this approach if: You are focused purely on identifying pain points and have other signals (support tickets, churn analysis, customer health scores) to help you understand what existing value you need to protect.

Option 3: Combined Framing (Recommended for Most Teams)

The most practical approach for most teams is to frame your MaxDiff question in a way that captures both dimensions simultaneously.

Instead of asking "Which of these is most important?" or "Which of these are you least satisfied with?", you ask something like:

- "Which of these unmet needs would make the biggest difference if solved?"
- "Which of these is the most important problem you currently face?"
- "Which of these improvements would have the greatest impact on your ability to get your job done?"

This framing implies both importance (it would "make a difference" or have "impact") and dissatisfaction (it is an "unmet need" or "problem"). You are asking customers to prioritize based on the gap between what matters and what is working, which is exactly what the opportunity concept tries to capture, but in a single, natural question.

Hospital MaxDiff Survey Example with Combined Framing

Hospital MaxDiff Survey Example with Combined Framing

The trade-off is that you lose the ability to cleanly separate the two dimensions. You cannot say with precision "this need is highly important but already satisfied" versus "this need is moderately important but completely unmet." The combined framing blends these into a single priority signal.

For most product decisions, this blended signal is sufficient. You want to know where to focus. The combined framing tells you: focus here, where importance and dissatisfaction intersect. The cases where you would make a different decision with separated data are relatively rare.

Choose this approach if: You are an early-stage product still searching for product-market fit, or you need to identify the most important unmet needs quickly. The nuance of separating importance from satisfaction matters less than speed and clarity.

Option 4: Relevant Items MaxDiff

When your need list is large and relevance varies across respondents, Relevant Items MaxDiff offers an elegant solution. Before the MaxDiff exercise begins, respondents complete a quick screener indicating which needs are actually relevant to their situation. The MaxDiff tasks are then built only from those relevant items.

For example, if you are researching the job of "plan a vacation," a business traveler might indicate that needs related to "entertaining children during travel" or "finding family-friendly accommodations" are not relevant. Those needs would be excluded from their MaxDiff exercise entirely, reducing cognitive load and eliminating noise from forced evaluations of irrelevant items.

This approach offers several advantages:

- **Reduced respondent burden.** Instead of evaluating 80 needs, a respondent might only see the 25-30 that apply to them.
- **Cleaner data.** You avoid the noise that comes from respondents guessing or satisficing on needs they have never experienced.
- **Relevance as signal.** The selection of relevant items itself becomes valuable data. You can analyze which needs different user types consider relevant before even looking at the utility scores.

Handling the analysis. When running HB estimation on Relevant Items MaxDiff data, you need to decide how to treat items that were not shown. If respondents explicitly marked items as "not relevant," you would typically use the "Missing Inferior" setting, which treats those items as systematically less preferred than the items they did select. This prevents the model from imputing neutral or positive utilities for needs the respondent indicated do not apply to them.

The segmentation trade-off. Traditional MaxDiff segmentation clusters respondents based on their utility scores across identical item sets. With Relevant Items MaxDiff, segmentation works differently:

- You can segment based on **relevance patterns**: which needs did different user types select as relevant? This can reveal fundamentally different job contexts or user segments before you even analyze preferences.
- You can segment based on **utilities within shared items**: for respondents who selected overlapping relevant items, you can still cluster based on how they prioritized those shared needs.

- You may need to **combine approaches**: use relevance patterns for initial segmentation, then analyze utility differences within each segment.

This is genuinely messier than traditional MaxDiff segmentation. If two respondents have no overlapping relevant items, you cannot directly compare their preferences. You would need to rely on the relevance selection patterns or external variables (demographics, behaviors) to group them.

Choose this approach if: You have a large needs list (50+) where relevance genuinely varies by user type, you are comfortable with segmentation based on relevance patterns rather than pure utility comparisons, and you suspect your market contains fundamentally different user segments with different job contexts. It is particularly powerful when the relevance selection itself is strategically interesting (for example, discovering that enterprise buyers consider an entirely different set of needs relevant than SMB buyers).

Be cautious if: You need clean utility comparisons across your entire sample, your segmentation strategy depends on clustering everyone based on the same items, or your needs are universally relevant to anyone doing the job.

Option 5: Feature-Based MaxDiff

If your team has already moved past discovery and has a concrete list of feature concepts, you can skip the needs layer entirely and ask customers to prioritize features directly.

The question becomes straightforward: "Which of these features would you most want us to build?" or "Which of these improvements would be most valuable to you?" This approach offers practical advantages:

Immediately actionable. The output is a prioritized feature list that can directly inform your roadmap without additional translation.

Speaks the language of stakeholders. Engineers, executives, and product managers naturally think in features. Research framed this way is easier to communicate and act on. Reduces abstraction. You avoid the sometimes awkward process of translating ranked needs into features after the research.

However, this approach has clear limitations:

You are testing solutions, not problems. If a feature ranks low, you cannot tell whether the underlying need is unimportant or whether your feature concept simply failed to resonate. You might abandon a valuable opportunity space because your first solution idea was weak.

You constrain your innovation space. Needs-based research can reveal opportunities you have not considered. Feature-based research only validates or invalidates ideas you have already generated. Features bundle multiple needs. A single feature often addresses several needs, making it hard to interpret what the ranking actually tells you about underlying priorities.

Choose this approach if: You are in execution mode rather than discovery mode, you have high confidence that your feature concepts are good solutions to real needs, and you need to prioritize a backlog quickly. This works best as a complement to earlier needs-based research, not a replacement for it.

Be cautious if: You are still in early discovery, you want to understand the problem space before committing to solutions, or you suspect your current feature ideas might not be the best ways to address customer needs.

> *I would be cautious about positioning it as equivalent to the other four. The other options are all variations on measuring needs (with different framings around importance, satisfaction, or relevance). Feature-based MaxDiff is measuring something fundamentally different: preference for proposed solutions. Another way to think about it: Options 1-4 help you figure out where to focus. Feature-based MaxDiff helps you figure out how to execute once you have already decided where to focus. They answer different questions at different stages of the product development process.*

Choosing Your Approach: A Summary

Your Situation	Recommended Approach	Why
Early-stage product, searching for fit	Combined framing	Speed and clarity matter most
Mature product with established value	Importance + targeted satisfaction	You need to know what to protect, not just what to build
Resource-constrained team	Combined framing + operational data	Supplement with support tickets, churn analysis to catch blind spots
Large outcome list (50+) with varying relevance	Relevant Items MaxDiff	Reduces burden, eliminates noise, relevance patterns become segmentation input
Need clean utility comparisons across all respondents	Standard MaxDiff (any framing)	Everyone evaluates same items, enabling direct comparison and clustering

A note on feature-based MaxDiff: If you have already completed needs-based discovery and want to prioritize a backlog of feature concepts, you can run MaxDiff on features directly. This gives you immediately actionable output but answers a different question: "which solution should we build?" rather than "which problem should we solve?" Use this as a follow-up to needs research, not a replacement for it.

Why This Changes Prioritization Conversations

Regardless of which option you choose, MaxDiff outputs create different conversations than traditional ODI scores.

Traditional ODI outputs create a specific problem in prioritization meetings. When you present opportunity scores like 14.7 versus 12.3, stakeholders inevitably ask whether that difference is meaningful. Is a 2.4-point gap worth reorganizing the roadmap? The math looks precise, but as we discussed in Chapter 7, that precision is largely illusory. The honest answer to "does this difference matter?" is usually "it depends," which undermines confidence in the data and opens the door for whoever argues loudest.

MaxDiff outputs sidestep this problem. Instead of debating point differences, you can make a cleaner statement: when forced to choose, customers consistently ranked "data export reliability" above "collaborative editing features." The hierarchy is the insight, not the precise numerical distance between items. You are not claiming that export reliability is exactly 1.4 times more important. You are claiming it wins head-to-head matchups more often, which is a more defensible and intuitive statement.

This changes how prioritization debates unfold. With traditional ODI scores, teams often get stuck arguing about methodology. Is the algorithm right? Is 14.7 really different from 12.3? Should we trust the survey? With ranked preferences from forced choices, the conversation shifts to strategic questions that matter:

- Should we address the top-ranked need first, or is there a cluster of related needs in positions two through five that we could solve together more efficiently?
- The third-ranked need is technically lower priority, but it is much easier to build. Could we capture a quick win while we plan the larger effort?
- Our enterprise segment shows different rankings than our SMB segment. Should we build different solutions or find the common thread?

These are productive debates about strategy and resources, not arguments about survey statistics. The methodology becomes invisible, which is what you want. The research should inform decisions, not become the subject of decisions.

The Statement Syntax Decision: Solution-Focused vs. Traditional JTBD Needs Statements

Once you have decided what dimension to measure, you face a second decision: how to write the actual statements. This brings us to a tension between the rigorous principles of Jobs-to-be-Done theory and the practical realities of survey design.

A core tenet of JTBD is to remain solution-agnostic, focusing exclusively on the customer's desired need/outcome. However, to get clear, quantifiable data from a survey, we sometimes need to bend this rule for the sake of clarity (for the researcher and the respondent).

This is a practical trade-off you often have to make when quantifying needs. JTBD focused need statements are the standard for discovery research, where you are mapping out the job for the first time. But in a quantitative survey, these abstract statements can be hard for people to evaluate, often leading to that ceiling effect where everything seems equally important. To get a clear signal on priorities, you sometimes need to frame the needs in a more concrete way that grounds the respondent in their actual experience.

Option A: Use Solution-Focused Statements

This approach sacrifices theory for practical clarity. It makes the survey much easier for people to answer and gives you clear, actionable insights about performance gaps, even if the statements hint at a solution.

Let us look back at our hospital example. The traditional JTBD need statement might be:

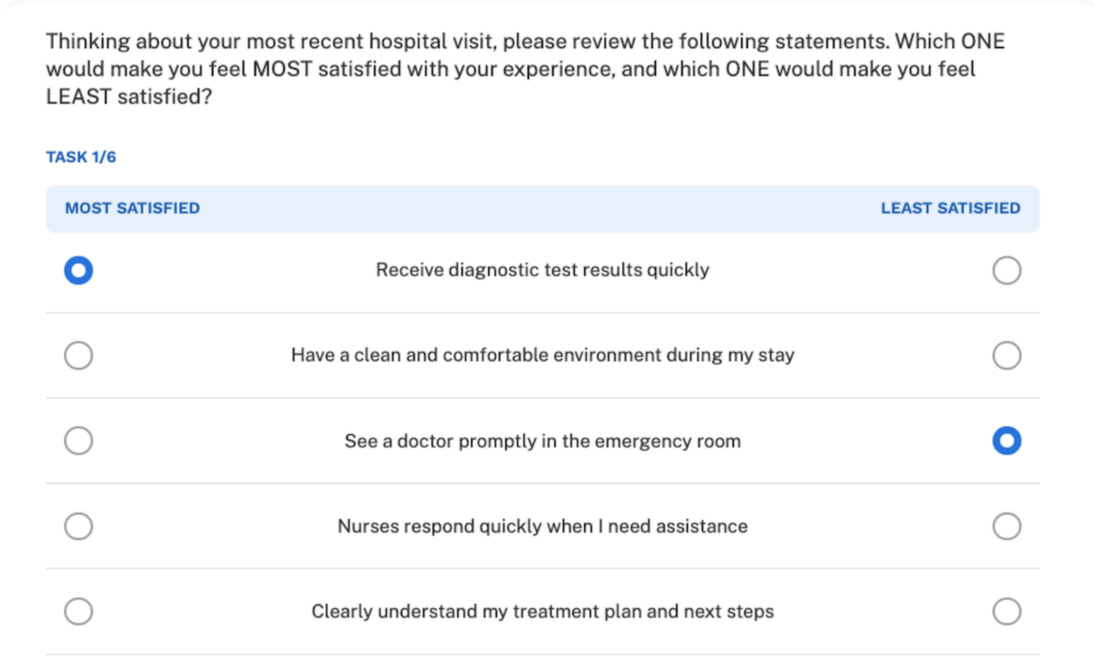
> *Minimize the time spent waiting for the results of a diagnostic test.*

We could change it to the more solution-focused statement:

> *I receive the results of diagnostic tests in a timely manner.*

The second version is far easier for a patient to evaluate. They can think back to their actual experience and quickly decide if they got their blood test results when they expected them. This grounds the question in reality and reduces the mental effort required to answer.

You might notice that "receiving results" sounds like an activity rather than a need. Strict practitioners may argue this violates the rules of defining a need. While true, the statement remains neutral regarding the solution. It does not mention an app, a phone call, or a paper letter. It simply describes the successful completion of the step. In a survey context, the clarity for the respondent is worth this minor shift in language.



Hospital MaxDiff Survey with Solution-Focused Statements

Choose this approach if: Your goal is to identify and prioritize improvements for an existing product or service. It gives you a clear roadmap for optimization.

Option B: Use Traditional Need Statements in a Narrow Context

This approach sticks much closer to JTBD principles. To make abstract needs comparable, you narrow your focus to a theme or step within the larger job, such as "managing a treatment plan." Within this tighter context, you can quantify more granular, tactical needs that are still solution-agnostic. When all the statements relate to the same focused activity, respondents can make more meaningful trade-offs.

For example, if the theme was "understanding the treatment plan," your MaxDiff statements might look like this using the classic, direction-based JTBD syntax:

- Minimize the time it takes to get my questions about the plan answered by a doctor.
- Minimize the confusion caused by medical jargon used by staff.

- Minimize the difficulty of remembering all the steps in my treatment plan.
- Minimize the likelihood of feeling rushed when discussing the plan.

As we discussed in Chapter 6 with JTBD and ODI syntax, if this rigid syntax feels too restrictive or unnatural for a survey, you can rephrase these statements using more conversational language. The goal is to remain focused on the need, not the solution. Here is how the same needs could be written in a more flexible style using words like quickly, easily, or avoid:

- Quickly get my questions about the plan answered by a doctor.
- Avoid confusion from the medical jargon used by staff.
- Easily remember all the steps in my treatment plan.
- Avoid feeling rushed when discussing the plan.

Notice that in both formats, all of these statements are traditional need statements. They describe what the patient wants to achieve without mentioning a solution, but they are all related to a particular part of the patient's journey or a job step in the job map. This narrow focus makes the trade-off ("What is more frustrating: the medical jargon or feeling rushed?") a realistic choice for the respondent.

Choose this approach if: You are doing foundational research to deeply understand a part of the customer's job. It is ideal for uncovering opportunities for breakthrough innovation rather than just incremental improvements.

Combining the Two Decisions

To summarize, you are making two independent decisions when designing your MaxDiff:

1. **What dimension to measure:** Importance only, satisfaction only, or combined framing

2. **How to write statements:** Solution-focused for clarity, or traditional JTBD need syntax for methodological rigor

These choices are orthogonal. You can use combined framing with solution-focused statements (practical and clear) or combined framing with traditional JTBD need statements (rigorous but requires narrow scope). The right combination depends on your research goals, your product's maturity, and the cognitive load you are willing to place on respondents.

The broader point is that there is no single correct approach. The original ODI methodology presents itself as a precise system, but as we have seen throughout this book, that precision often obscures judgment calls and trade-offs. MaxDiff is a better tool, but it is still a tool. You have to decide how to wield it based on your context, resources, and risk tolerance.

What matters is that you are measuring customer needs through forced trade-offs rather than inflated ratings, that you are producing a clear hierarchy rather than a spreadsheet of similar-looking scores, and that you are designing your research to answer the strategic questions your team actually faces, not just following a methodology because someone said it works.

A Practical Guide to Building Your MaxDiff Study

With that decision made, we can now walk through the mechanics of creating an effective MaxDiff study. Getting this right comes down to three things: the study design, the quality of your statements, and the experience you create for the person taking the survey.

A quick note on tools: You'll find many great platforms out there to run a MaxDiff study, including tools like Qualtrics, Sawtooth Software, Conjointly, and others. Because the specific buttons you click and the exact setup menus change over time and differ between platforms, this guide won't be a detailed tutorial for any single piece of software. Instead, we will focus on the universal principles and platform-agnostic steps that are essential for a successful study, no matter which tool you choose.

The fundamentals of research design are what truly drive good results, and mastering them will allow you to confidently set up your study on any platform.

Step 1: Write and Test Your Statements

The quality of your data is entirely dependent on the quality of the statements you test.

- **Write Clear Statements:** Each statement should be short, clear, and contain only one idea. Instead of a complex statement like "A15 Bionic chip with 6 core CPU for faster machine learning," break it down into benefits like "Fast performance for demanding apps" or "Quickly switches between apps."
- **Ensure They Can Be Compared:** All statements in your list must make sense when compared with each other. A person should be able to make a meaningful trade off between any two statements on your list.
- **Pilot Test Everything:** Before you launch your study, test your statements. First, have your internal team take the survey. This will catch obvious clarity issues. If you find yourself rereading a statement, your customers will too. After an internal review, test the survey with 20 to 30 people from your target audience. Ask them what they think each statement means and if they found any choices difficult or confusing. Use their feedback to refine your list before the full launch.

Step 2: Design the Study

The statistical setup for MaxDiff is more forgiving than many other methods, but you need to get a few parameters right.

Of course. Here is a more detailed expansion on the three key parameters of MaxDiff study design.

Sample Size: How Many People Do You Really Need?

The goal with sample size is to reach a point of **stability**, where adding more respondents doesn't meaningfully change the overall ranking of your items.

- **The Baseline (200 Respondents):** A sample of 200 is considered a strong baseline because it typically provides enough data to create narrow **confidence intervals** around your scores. A narrow confidence interval means you can be more certain of the precise score for each item. This makes it easier to declare a "winner" when two items are ranked closely together. With 200 respondents, you can be more confident that a 5-point difference between items is statistically real and not just random noise.
- **The Practical Minimum (50-75 Respondents):** Why can MaxDiff work with smaller samples? The answer lies in the **Hierarchical Bayes (HB)** analysis used to calculate the scores. HB is a sophisticated model that estimates scores for each individual while simultaneously learning from the patterns of the entire group. In simple terms, it "borrows strength" across respondents. If one person's answers are a bit inconsistent, the model uses data from other, similar people to improve its estimate for that individual. This makes the data from each person more powerful, allowing you to get good **directional insights** (knowing the top 5 items, for instance) even with a smaller group. The trade-off is that your confidence intervals will be wider, so you'll have less precision in the final scores.
- **Segmentation (50+ Per Group):** When you want to compare different groups of customers (e.g., new vs. loyal, US vs. Europe), you should treat each group as a mini-study. Aiming for at least 50 people *per segment* ensures you have enough data to get a reliable read on that group's priorities. If you plan to analyze four segments, a total sample of 200 (50 for each) would be your minimum starting point.

Note on Calculations: The sample sizes listed here are practical rules of thumb that work for the vast majority of commercial projects. If you need to calculate exact power requirements for a complex academic study, I recommend reviewing the technical papers provided by Sawtooth Software, the creators of the standard algorithms used in this field. [39]

Choice Set Configuration: Designing for the Human Brain

This is about managing the **cognitive load** on your respondents to ensure you get high-quality data from beginning to end.

- **Items Per Set (4 to 5):** This range is the sweet spot for human decision-making. When presented with 4 or 5 options, a person can reasonably hold them all in their working memory to make a comparative judgment. If you show 7 or 8 items at once, people get overwhelmed. They can't effectively compare all the options, so they often resort to mental shortcuts, and the quality of their choices declines. The task becomes a chore, and the data suffers.
- **Sets Per Respondent (6 to 8):** The biggest threat to data quality in any survey is **respondent fatigue**. While the MaxDiff task is more engaging than rating scales, it's still repetitive. Experience shows that after about 8 sets, many people start answering on autopilot to get through it. Their response times get shorter, and their choices become less thoughtful. Keeping the task to 6-8 sets ensures you capture high-quality, considered choices from each person. The HB analysis method is efficient enough to build a solid model from this amount of data without needing to push respondents to their limits.

Statistical Coverage: Ensuring a Fair and Accurate Test

Your survey software doesn't just show random items; it follows a precise **experimental design** to ensure the results are accurate and unbiased.

- **The "Round Robin" Principle:** The core goal of the design is to generate enough direct comparisons to build a reliable model. To do this, the design ensures that, across all respondents, every single statement appears in the same set with every other statement multiple times. Think of it like a sports tournament: to get a true ranking, you want every team to play every other team. Since one person can't do all those comparisons, the experimental design spreads these "matchups" intelligently across the entire sample.

- **Balance and Orthogonality:** A good design follows two key principles. First is **balance**, which means every item is shown a roughly equal number of times overall. This ensures that no item gets an advantage by appearing more frequently. Second is **orthogonality**, which is a technical term for ensuring the items are shown together in a way that lets the model tell their individual preferences apart. It prevents items from always appearing with the same "partner," which would make it hard to know which of the two is driving the choice.

You don't need to build this design yourself. Modern survey platforms handle it automatically. But knowing these principles helps you understand that the background process is a structured, scientific approach designed to give you the cleanest possible read on what your customers value.

Step 3: Build the Survey Experience

How you present the survey to a respondent can dramatically affect the quality of the data you get back. A clear, thoughtful experience encourages focus and honesty, while a confusing one leads to frustration and rushed answers. Here's a more detailed look at the three key elements for creating a great respondent experience.

Set the Context: Grounding Your Respondent

This is the first thing your respondent should see. Its job is to activate the right memories and put them in the correct frame of mind for your questions. Without proper context, people will answer based on abstract feelings rather than specific, relevant experiences, which makes the data less reliable.

Why this matters: You want them thinking like a "customer who just used your app," not like a "person taking a random survey." A strong context setter acts as a mental warm up.

Practical Examples: The key is to be specific enough to trigger a memory but general enough that most people can easily recall an experience.

- **For B2B Software:** "Please think about your typical process for completing [Job to be Done, e.g., your monthly expense report] using [Software Name]. The following questions will be about that experience."
- **For a Retail Store:** "We'd like you to think about your most recent shopping trip to [Store Name]. Please keep that visit in mind as you answer the next few questions."
- **For a Travel Website:** "Please reflect on the last time you booked a personal trip online. We're interested in understanding what was most and least important to you during that booking process."
- **For a Healthcare Experience:** "Thinking about your last check-up with your primary care doctor, please consider all aspects of that visit, from scheduling to the appointment itself."

Helpful Tip: Use recency to your advantage. Asking about "your last visit" or "your experience in the past month" is usually more effective than asking about their experience in general, as it prompts a more vivid and accurate memory.

Give Simple Instructions: Clarity Over Complexity

Your respondents don't need to know the name of the methodology or the statistics behind it. In fact, mentioning "MaxDiff" or explaining the experimental design will only cause confusion and make the task seem more intimidating than it is. The goal is to make the task feel effortless.

Why this matters: Simple instructions build confidence and let the respondent focus all their mental energy on making thoughtful choices, not on trying to figure out what you're asking them to do.

Examples of Effective Wording:

- **Standard & Clear:** "On each of the next few screens, you will see a group of statements. From each group, please choose the **one** that is **MOST** important to you and the **one** that is **LEAST** important to you. There are no right or wrong answers; we want to understand what matters most to you."

- **Short & Casual:** "In each set, just pick your top choice and your bottom choice. That's it!"
- **Benefit-Focused:** "To help us improve your experience, please review each set of options below. In each one, simply select which aspect you find most appealing and which you find least appealing."

Helpful Tip: Explicitly tell respondents to **only consider the items on the screen**. This prevents them from trying to compare an item in the current set to one they saw two sets ago, which is not how the exercise is designed to work. You could add a sentence like, "Please make your choices based only on the options currently shown in each group."

Step 4: Choose Your Platform

Several tools can help you run a MaxDiff study, each with its own pros and cons.

- **Qualtrics:** This is one of the most accessible options, especially if you already use it. It has a built in MaxDiff question type that handles the design for you.[35] The analysis tools are basic, so you may need to export your data, but it's a great choice for straightforward studies.
- **Sawtooth Software:** This is the gold standard for choice based research, offering sophisticated design and analysis tools.[34] It has a steeper learning curve and costs more, but it's the right choice for complex or large scale projects.
- **Other Platforms:** Tools like Conjointly offer a good middle ground, with more advanced analysis than Qualtrics but a more user friendly interface than Sawtooth.

No matter which platform you choose, make sure to test your survey on both desktop and mobile devices to ensure it works smoothly for everyone.

Step 5: Fielding your survey

Launching your survey is a major milestone, but your work isn't done yet. How you find, motivate, and manage your respondents will directly impact your timeline, budget, and the trustworthiness of your data. This phase requires active planning and diligent monitoring to ensure the data you collect is clean, reliable, and comes from exactly the right people.

Planning Your Recruitment and Fielding Strategy

Before you can even think about a soft launch, you need a solid plan. This involves not just finding respondents, but also managing the timeline, budget, and all communication associated with the study.

1. Define Your Audience and Design an Effective Screener First, be crystal clear about who you need to hear from. Are they customers who have used a particular feature in the last 90 days? Are they people in a certain industry who do not use your product? The criteria you set will be turned into a short **screener questionnaire** at the beginning of your survey to ensure only qualified people participate.

- **Best Practice:** Design your screener questions carefully to avoid giving away the "right" answer. For example, instead of asking "Do you use our advanced reporting feature?" (a yes/or-no question that signals what you're looking for), ask "Which of the following features have you used in the past 90 days?" and include your target feature among a list of other plausible options. This ensures you get more honest and accurate qualifications.

2. Estimate Incidence Rate (IR) and Set a Budget Your screener criteria will determine your **Incidence Rate (IR)**, which is the percentage of a general population that will qualify for your study. This is one of the biggest factors driving the cost and timeline of your research.

- A **high IR** (e.g., 50% or more) means your audience is broad (e.g., "adults who have shopped online in the past year"). This makes recruitment much easier and more affordable.

- A **low IR** (e.g., 5% or less) means your audience is niche (e.g., "anesthesiologists in the Pacific Northwest who use a specific brand of monitoring equipment"). This makes recruitment more difficult and more expensive because you have to screen through many people to find one qualified respondent.

Your total budget will be a function of your target sample size, your IR, and the incentive you offer.

3. Choose Your Recruitment Method There are several ways to get your survey in front of people. Each has its own distinct benefits and drawbacks.

- **Your Own Customer Lists (Email or In-Product):** This involves inviting your existing customers to participate, either through an email campaign or a pop up or banner inside your application.
    - **Pros:** It's often the most affordable method. The audience is highly relevant, and you may have behavioral data you can use to target them.
    - **Cons:** You risk "sampling bias," meaning you might only hear from your most engaged or happiest customers, not a true cross section. You also risk "survey fatigue" if you are constantly asking your customers for feedback.
- **Third Party Research Vendors (Panel Companies):** These are firms that maintain large databases ("panels") of people who have pre-profiled themselves and agreed to take surveys for compensation. You provide your screening criteria, and they deliver the qualified respondents.
    - **Pros:** This is often the fastest way to get a large, diverse sample. They can reach specific demographic and professional groups that you can't access yourself, and they handle all the logistics of quotas and incentives.

- **Cons:** It can be expensive, especially for low IR audiences. You must be diligent about data quality, as some panelists are "professional survey takers" who may rush through to maximize their earnings.

- **Other Digital Channels (Social Media, Online Ads):** You can use targeted ads on platforms like LinkedIn or Facebook to find niche professional or interest based groups.
    - **Pros:** Can be effective for reaching specific, hard-to-find audiences who may not be on traditional panels.
    - **Cons:** It can be difficult to predict the cost and time required. This method requires more hands-on management and careful data quality screening.

4. Determine the Right Incentives An incentive is a small token of appreciation to compensate respondents for their time and thoughtful feedback. A fair incentive signals that you value their input and encourages higher quality responses.

- **What to Offer:** Cash equivalent rewards like gift cards (e.g., Amazon, Visa) are usually the most effective and broadly appealing. Depending on your audience, product discounts or donations to charity can also be compelling options.
- **How Much to Offer:** The amount depends on the survey's length and your audience's profile. A common rule of thumb for general consumer audiences is to offer $1 to$**2 for every five minutes** of survey time. For highly paid professionals like doctors, lawyers, or C-level executives, you will need to offer a higher amount to make it worth their while. Your research vendor can provide guidance on appropriate rates. Be careful not to over-incentivize, as an unusually high reward can attract fraudulent respondents.

5. Craft Your Invitation and Set Expectations Your survey invitation, whether it's an email, an in-app message, or a description on a panel site, is your one chance to make a good impression. It should clearly and concisely state:

- The purpose of the study (e.g., "to help improve our product").
- The estimated time to complete the survey. Being honest here is crucial for reducing drop-outs.
- The incentive being offered for their participation.
- A statement on confidentiality, assuring respondents that their individual answers will be kept private and reported only in aggregate.
- A contact or support link for anyone who runs into technical trouble.

Executing the Launch and Monitoring Quality

Once your recruitment plan is set, you can move forward with the launch.

> *PRE-Test again before launching!!*

- **Start with a Soft Launch:** Before sending your survey to your entire sample, launch it to a small fraction (around 5–10%) of your target audience. This is your final real-world check. Review these initial responses carefully to catch any technical glitches, confusing wording that you missed in the pilot, or problems with how the survey displays on different devices. It's much easier to fix a problem after 20 responses than after 200.
- **Actively Check for Bad Respondents:** Not all survey takers are diligent. It is standard practice to identify and remove responses from people who are not giving thoughtful answers, as they can corrupt your results. Common culprits include:

  - **Speeders:** People who complete the survey so fast they couldn't have possibly read the questions. You should set a realistic minimum completion time based on your pilot tests and remove anyone who finishes faster.
  - **Inattentive Responders:** People who fail simple attention check questions (a "trap question"), such as, "For this question, please select 'Most' for the third item to show you are paying attention."
  - **Patterned Responders:** People who select options in a suspicious pattern, like always choosing the first and last items in every set.
- **Manage Your Quotas:** If your study requires input from customer segments (e.g., 50% new users, 50% experienced users), you need to monitor your incoming data to ensure you are meeting these targets. Most survey platforms and vendors allow you to track these quotas in real time and close the survey for a group once its target is met.
- **Clean Your Final Dataset:** The process of removing speeders, inattentive respondents, and patterned responders is called **data cleaning**. This is a non-negotiable final step before analysis. Ensuring your final dataset only includes responses from engaged, qualified people is what makes your insights trustworthy and defensible.

Making Sense of the Results

Before we dive into the analysis, let's clarify how to interpret the results. You might be wondering: "If we asked respondents about what they were most and least satisfied with, why are we looking at scores that represent importance or utility?"

Your analysis software will likely call the final numbers "utility scores." This is a generic statistical term. It does not mean the numbers represent economic value or importance. The meaning of the score depends entirely on the question you asked.

Since we asked respondents which items they were most and least satisfied with, these numbers are actually Relative Satisfaction Scores. A high score means that item is working well for the customer. A low score signals a problem area or a pain point.

You must be precise about how you label these charts. If you label a chart "Importance" when you actually measured satisfaction, you will confuse your stakeholders. A low score on this chart does not mean the item is unimportant. It means the customer is currently unhappy with it. To be accurate, we will label our charts as "Relative Satisfaction" to match the data we collected. Once the data is in, the final step is to turn the numbers into actionable insights.

Download MaxDiff Chapter 8 Survey Data

Loading Your Data

First, you need to load your MaxDiff results into R. Assuming you've saved your processed data as a CSV file, here's how to get started:

```r
# Load the dataset
maxdiff_chapter8_example <- read.csv("maxdiff_chapter8_examp
```

Exploring Your Data Structure

Before diving into analysis, you need to understand what you're working with. The `str()` function in R gives you a quick overview of your dataset structure:

```r
str(maxdiff_chapter8_example)
```

str() output

```
R OUTPUT ff_chapter8_example)
'data.frame':   400 obs. of  24 variables:
$ respondent_id                                 : int  1 2 3 4 5 6 7 8 9 10
$ age                                           : num  31 33 47 36 36 49 39
$ income                                        : num  69000 88000 66000 63
$ visits_per_week                               : num  4 4 6 7 4 4 10 3 1 4
$ gender                                        : chr  "Male" "Male" "Femal
$ work_status                                   : chr  "Full-time" "Part-ti
$ Consume high-quality coffee for satisfaction     : num  18.6 15.1 13.7 16.8
$ Obtain coffee quickly during time constraints    : num  0.625 1.294 1.574 0.
$ Secure comfortable space for extended stays      : num  1.41 1.67 1.44 1.45
$ Access internet connectivity while away from home: num  3.42 3.3 5.48 3.47 5
$ Accumulate rewards through repeat purchases      : num  13.9 16.15 9.27 10.4
$ Place orders remotely to avoid waiting           : num  0.921 1.406 0.729 1.
$ Acquire fresh food alongside coffee              : num  7.34 5.35 5.71 10.05
$ Access coffee within daily travel patterns       : num  12.7 16.7 21.3 16.6
$ Purchase coffee during off-peak hours            : num  3.29 3.11 4.52 5.29
$ Support businesses aligned with personal values  : num  2.57 1.84 2.44 1.87
$ Receive guidance for optimal coffee selection    : num  13 13.1 15 16.4 13.8
$ Purchase coffee within financial limits          : num  4.83 7.5 4.08 6.81 7
$ Find quiet space for focused activities          : num  5.56 1.85 1.82 2.34
$ Choose from options matching current preferences : num  2.7 3.21 2.1 2.47 6.
$ Experience service in hygienic conditions        : num  9.2 8.35 10.79 4.7 7
$ prediction_accuracy                           : num  0.812 0.938 0.938 0.
$ rlh                                           : num  0.588 0.662 0.662 0.
$ quality_rating                                : chr  "Excellent" "Excelle
>
```

This output shows we have 400 respondents and 24 variables. The dataset includes demographic information (respondent_id, segment, age, income, visits_per_week, gender, work_status) and 15 utility scores for different coffee shop attributes. Each utility variable represents one of our coffee shop attributes with descriptive names like "Consume high-quality coffee for satisfaction" and "Obtain coffee quickly during time constraints."

The dataset also includes model quality metrics (prediction_accuracy, rlh, quality_rating) that help assess how well the MaxDiff model performed for each respondent.RetryClaude does not have the ability to run the code it generates yet.

The `describe()` function from the psych package provides detailed statistics for each variable:

```r
library(psych)
describe(maxdiff_chapter8_example)
```

describe() output

R OUTPUT

```
maxdiff_chapter8_example)
                                           vars   n     mean       sd   m
respondent_id                                 1 400   200.50   115.61
age                                           2 400    36.39     8.97
income                                        3 400 73850.00 24947.61 7
visits_per_week                               4 400     4.28     2.58
gender*                                       5 400     1.49     0.50
work_status*                                  6 400     2.00     1.20
Consume high-quality coffee for satisfaction  7 400     8.51     5.45
Obtain coffee quickly during time constraints 8 400     5.96     8.12
Secure comfortable space for extended stays   9 400     6.87     6.27
Access internet connectivity while away from home 10 400 6.57    4.71
Accumulate rewards through repeat purchases  11 400     8.83     4.57
Place orders remotely to avoid waiting       12 400     4.41     4.56
Acquire fresh food alongside coffee          13 400     5.63     2.62
Access coffee within daily travel patterns   14 400     7.36     5.57
Purchase coffee during off-peak hours        15 400     7.15     3.11
Support businesses aligned with personal values 16 400  6.80     5.63
Receive guidance for optimal coffee selection 17 400    6.79     5.28
Purchase coffee within financial limits      18 400     8.01     6.07
Find quiet space for focused activities      19 400     5.58     4.46
Choose from options matching current preferences 20 400 6.54     4.22
Experience service in hygienic conditions    21 400     4.99     3.97
prediction_accuracy                          22 400     0.86     0.09
rlh                                          23 400     0.61     0.05
quality_rating*                              24 400     1.02     0.14
                                               range  skew kurtosis
respondent_id                                 399.00  0.00    -1.21
age                                            53.00  0.64     0.43
income                                     123000.00  0.25    -0.53 124
visits_per_week                                12.00  0.65    -0.04
gender*                                         1.00  0.03    -2.00
work_status*                                    3.00  0.57    -1.36
Consume high-quality coffee for satisfaction   23.33  0.76    -0.53
Obtain coffee quickly during time constraints  28.47  1.35     0.18
Secure comfortable space for extended stays    27.23  0.85    -0.51
Access internet connectivity while away from home 23.05 1.61   2.11
Accumulate rewards through repeat purchases    20.21  0.37    -0.76
Place orders remotely to avoid waiting         18.99  1.48     1.32
Acquire fresh food alongside coffee            14.13  0.42    -0.39
Access coffee within daily travel patterns     23.04  1.06    -0.08
Purchase coffee during off-peak hours          17.46  0.39    -0.31
Support businesses aligned with personal values 22.38 0.76    -0.73
```

```
Receive guidance for optimal coffee selection          21.68  0.62    -0.87
Purchase coffee within financial limits                25.62  1.20     0.18
Find quiet space for focused activities                22.90  1.05     0.52
Choose from options matching current preferences       17.91  0.74    -0.50
Experience service in hygienic conditions              15.24  0.47    -0.99
prediction_accuracy                                     0.44 -0.48     0.19
rlh                                                     0.26 -0.48     0.19
quality_rating*                                         1.00  6.83    44.78
```

For the utility scores, pay attention to the mean values. These represent the average importance each attribute holds across all respondents. Notice how "Accumulate rewards through repeat purchases" has the highest mean at 8.83, followed closely by "Consume high-quality coffee for satisfaction" at 8.51 and "Purchase coffee within financial limits" at 8.01. At the other end, "Experience service in hygienic conditions" has the lowest mean at 4.99, while "Place orders remotely to avoid waiting" scores 4.41.

Ranking Your Attributes

The most straightforward way to interpret MaxDiff results is to rank attributes by their mean utility scores. We can extract the utility scores and create a results dataframe:

```r
# Get basic summary statistics (updated column range)
summary(maxdiff_chapter8_example[,7:21])

# Calculate mean utilities for each attribute (updated colum
mean_utilities <- sapply(maxdiff_chapter8_example[,7:21], me

# Create results dataframe
results <- data.frame(Attribute = names(mean_utilities), Uti
results <- results[order(results$Utility, decreasing = TRUE)

# Display top results
print(results[1:10,])
```

mean utility scores output

R OUTPUT

```
ults[1:10,])



Accumulate rewards through repeat purchases Accumulate rewards through repeat
Consume high–quality coffee for satisfaction Consume high–quality coffee for s
Purchase coffee within financial limits Purchase coffee within financial limit
Access coffee within daily travel patterns Access coffee within daily travel p
Purchase coffee during off–peak hours Purchase coffee during off–peak hours 7.
Secure comfortable space for extended stays Secure comfortable space for exten
Support businesses aligned with personal values Support businesses aligned wit
Receive guidance for optimal coffee selection Receive guidance for optimal cof
Access internet connectivity while away from home Access internet connectivity
Choose from options matching current preferences Choose from options matching

>
```

This ranking reveals what matters most to your customers. The results show coffee shop attributes ranked by their mean utility scores, with clear customer priorities emerging. Rewards programs top the list at 8.83, followed by high-quality coffee at 8.51. The drop to price considerations at 8.01 shows a meaningful gap between top priorities and cost concerns.

The gap_to_next column shows the difference between each attribute and the next-ranked item. This helps identify where the largest preference gaps occur. There's a notable 0.65-point gap between price and location convenience (7.36), indicating distinct tiers of customer priorities rather than gradual decline.

The output shows that loyalty programs and coffee quality drive customer choice, while practical factors like pricing, location, and timing remain key secondary considerations.

Visualizing the Results

Let's look to visualize the prior printed list to better see our results.

```r
library(ggplot2)

ggplot(results, aes(x = reorder(Attribute, Utility), y = Uti
  geom_col(fill = "steelblue", alpha = 0.7) +
  geom_point(size = 3, color = "darkred", alpha = 0.8) +
  geom_text(aes(label = round(Utility, 2)),
            hjust = -0.2, size = 3.5, color = "black") +
  coord_flip() +
  labs(title = "Coffee Shop Attribute Importance",
       subtitle = "Based on MaxDiff Analysis (n=400)",
       x = "Attributes",
       y = "Mean Utility Score") +
  theme_minimal() +
  theme(axis.text.y = element_text(size = 10),
        plot.title = element_text(size = 14, face = "bold"))
  expand_limits(y = max(results$Utility) * 1.1)
```

Understanding the Code

The `coord_flip()` function rotates the chart to make attribute names readable. The `reorder()` function automatically sorts attributes by their utility scores, placing the most important at the top.

Visual Elements Working Together

The `geom_col()` function creates the blue bars that show relative importance. The `fill = "steelblue"` sets the color while `alpha = 0.7` makes them slightly transparent for a professional appearance.

The `geom_point()` layer adds dark red dots at the end of each bar. These points serve as visual anchors that make it easier to read exact values, especially when bars have similar lengths.

The `geom_text()` function displays the actual utility scores next to each bar. The `round(Utility, 2)` ensures numbers show with two decimal places for consistency. The `hjust = -0.2` parameter positions text slightly beyond the bar ends, preventing overlap with the bars themselves.

Layout and Spacing Details

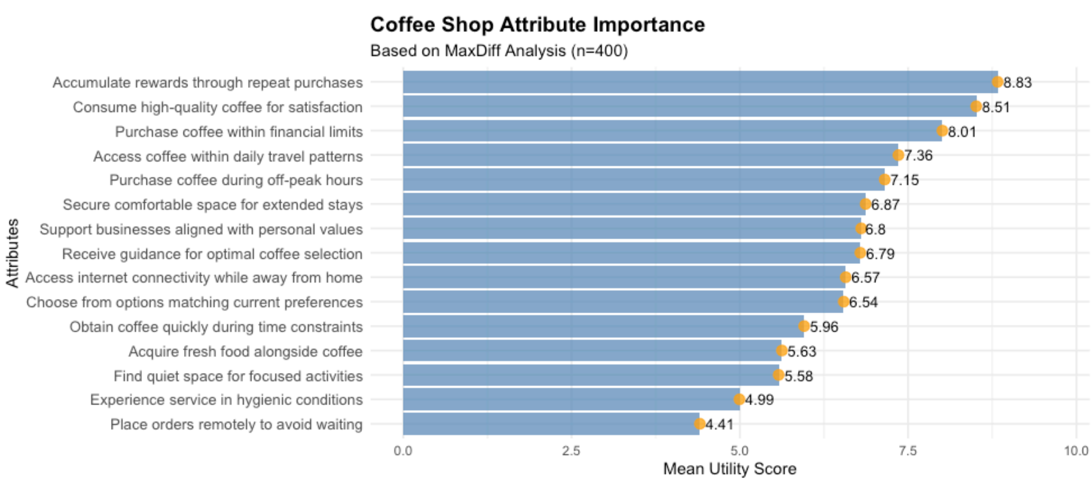
The `expand_limits(y = max(results$Utility) * 1.1)` line creates extra space on the right side of the chart. This prevents the text labels from getting cut off at the chart edges. The multiplication by 1.1 adds 10% additional space beyond the highest value.

Without this expansion, your text labels might disappear or appear cramped against the plot boundary. This small detail makes the difference between a professional-looking chart and one that appears unfinished.

Color and Text Choices

The combination of steelblue bars with dark red points creates good contrast without being overwhelming. The text labels use black color with `size = 3.5` to ensure they remain readable across different display sizes.

The `theme_minimal()` removes unnecessary chart elements like gray backgrounds, creating a clean appearance that focuses attention on your data.

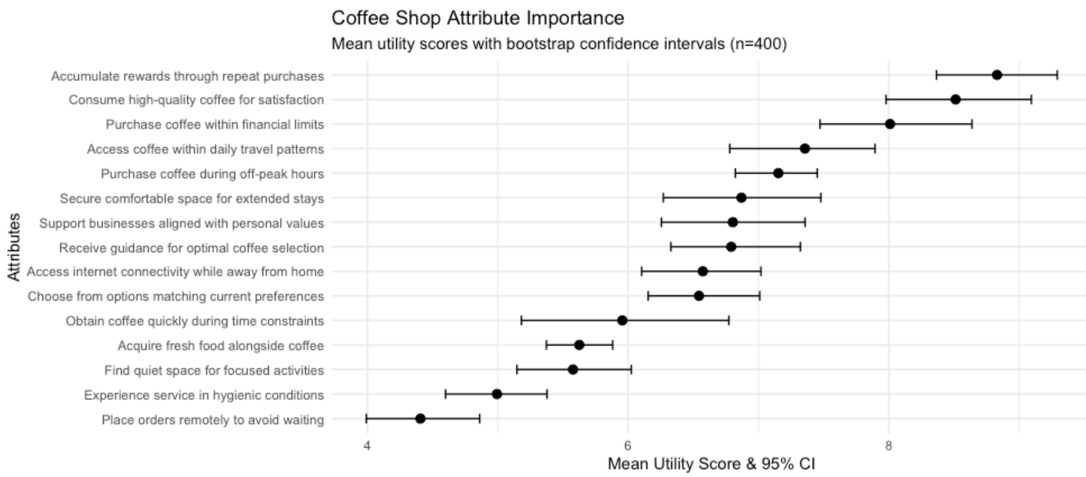


Coffee Shop Attribute Importance

Another way to visualize this information is to use estimates with error bars. This approach was inspired by Chris Chapman's blog post "[Individual Scores in Choice Models Part 1: Data & Averages]," where he demonstrates how to create more informative visualizations of MaxDiff and choice model results.[36]

Chapman highlights a key point about standard bar charts showing only averages. We can see the averages but have no insight into the distribution. Are the averages strongly different? Or are they close in comparison to the underlying distributions? By adding error bars, we can better assess whether observed differences between attributes are meaningful or simply due to random variation.

```r
library(ggplot2)
library(reshape2)
library(forcats)

# Melt only the utility columns (8:21), without respondent_i
utility_melted <- melt(maxdiff_chapter8_example[, 8:21])

# Reorder by mean
utility_melted$variable <- fct_reorder(utility_melted$variab

# Create the plot
ggplot(data = utility_melted, aes(x = value, y = variable))
  geom_errorbar(stat = "summary", fun.data = mean_cl_boot, w
  geom_point(size = 4, stat = "summary", fun = mean, shape =
  theme_minimal() +
  xlab("Mean Utility Score & 95% CI") +
  ylab("Attributes") +
  labs(title = "Coffee Shop Attribute Importance",
       subtitle = "Mean utility scores with bootstrap confid
```



Mean utility scores with bootstrap confidence intervals (n=400)

Following Chapman's approach, the chart above uses bootstrap confidence intervals to show the uncertainty around each mean utility score. The error bars help us distinguish between attributes that are truly different in importance versus those that may appear different but have overlapping confidence intervals. This allows us to determine whether the top attribute is really much stronger than other options or only slightly better.

Let's break down the key parts of the code:

The `melt()` function reshapes the data from wide format to long format. Instead of having separate columns for each attribute, it creates one column for attribute names and another for their utility values. This structure works better with ggplot2's layered approach to building charts.

```r
utility_melted <- melt(maxdiff_chapter8_example[, 8:22])
```

The `fct_reorder()` function from the forcats package sorts the attributes by their mean utility scores. This puts the most important attributes at the top of the chart and the least important at the bottom, making patterns easier to spot.

```r
utility_melted$variable <- fct_reorder(utility_melted$varia
```

The `geom_errorbar()` layer adds the confidence intervals. The `mean_cl_boot` function calculates bootstrap confidence intervals, which provide a robust way to estimate uncertainty around the mean values. The bootstrap method resamples the data many times to estimate the variability of the mean.

```r
geom_errorbar(stat = "summary", fun.data = mean_cl_boot, wi
```

The `geom_point()` layer adds the actual mean values as solid circles on top of the error bars. The `size = 4` parameter makes the points large enough to see clearly, while `shape = 20` creates filled circles.

```r
geom_point(size = 4, stat = "summary", fun = mean, shape =
```

This creates a clearer picture of which coffee shop attributes are genuinely more important to customers versus those that are statistically similar in their utility scores. Looking at our chart, we can see that "Accumulate rewards through repeat purchases" stands out as clearly more important than other attributes, while several attributes in the middle have overlapping confidence intervals.

However, as Chapman also points out, while charts with error bars provide valuable statistical insight, they still focus on averages rather than individual customer preferences. We do not reach any "average" customer. We reach individuals. This limitation leads to his preference for distribution plots that show the full range of individual responses. If you are interested in distribution plots check out his blog post for more details.

8.7 Chapter Conclusion

Figuring out what customers want is essential to building great products and services, and the method you use to get those answers matters. A flawed method can lead to misleading results, while a solid one gives you the clarity to act with confidence.

We've seen how MaxDiff, combined with a thoughtful approach to framing your questions around satisfaction, provides a complete and way to prioritize customer needs. You now have a practical workflow to get a reliable, ranked list of what matters most to your customers.

But what do you do with this list? Now that you have these priorities, how do you use them to come up with new ideas and build a product roadmap? That is exactly where we are headed in the next chapter.

Chapter 8 Summary

- The traditional **opportunity algorithm** presented in the previous chapter is methodologically flawed, often leading to unreliable priorities. **Maximum Difference Scaling (MaxDiff)** is a statistically sound and practical alternative.
- Instead of using abstract rating scales, MaxDiff works by showing respondents small sets of items and asking them to choose the **most** and **least important** (or appealing, satisfying, etc.). This forces realistic trade-offs and reveals a true hierarchy of needs.
- Framing the MaxDiff question around **satisfaction** ("Which were you most/least satisfied with?") is an approach to avoid users rating all needs as important. It is often more actionable than asking about abstract importance because it directly highlights performance gaps and unmet needs.
- Designing a successful MaxDiff study involves several key steps: writing clear, comparable statements; setting the right parameters (**sample size, items per set, sets per respondent**); carefully planning the survey **fielding** (screeners, incentives); and ensuring data quality through active monitoring and cleaning.
- The output of a MaxDiff analysis is a set of **utility scores** for each item. These scores allow you to create a clear, ranked list of customer priorities, confidently identifying what matters most and where to focus your efforts.

Chapter 8 Exercises

1. Choose Your Own Scenario & Plan Your Study: Think of a product, service, or experience you're familiar with and would like to improve. This could be related to your job, a hobby, or even a daily app you use. The key is to pick a topic where you can realistically brainstorm a list of customer needs or priorities.

For inspiration, you could focus on a topic like:

- Prioritizing new features for a **fitness app**.
- Improving the online checkout process for a **retail website**.
- Understanding what remote employees value most in a **work-from-home setup**.
- Enhancing the visitor experience at a **local museum**.

Once you've chosen your topic, create a brief research plan that defines:

- Your single most important **research objective**.
- A final list of statements you will test.
- Your chosen **design parameters** (sample size, items per set, sets per respondent).
- The **introductory text and instructions** for your respondents.

2. Select a Platform and Build the Survey: Many survey platforms offer MaxDiff question types. Find one you can access. Many offer free trials or have free plans with limited features.

- **Common Platforms: Qualtrics**, **Sawtooth Software**, and **Conjointly** are industry standards.

- **Action:** Sign up for a trial or use an existing account to build your survey. Input your statements and set up the MaxDiff exercise according to the design parameters you defined in step 1.

3. Pilot Test Your Survey: Once your survey is built, don't launch it to hundreds of people. Instead, **run a pilot test** as described in the chapter.

- **Action:** Find 2-3 friends or colleagues and send them the survey link.
- Ask them for feedback: Was anything confusing? Did any of the choices feel impossible to make? How long did it take them?
- Use their feedback to refine your statements or instructions before a full launch.

References

[34] *Sawtooth Software.* "Creating a MaxDiff." *SawtoothSoftware.com.* Available at: https://sawtoothsoftware.com/help/discover/survey-elements/maxdiff/creating-a-maxdiff.

[35] *Qualtrics.* "MaxDiff Analysis Technical Overview." *Qualtrics.com.* Available at: https://www.qualtrics.com/support/conjoint-project/getting-started-conjoints/getting-started-maxdiff/maxdiff-analysis-white-paper/.

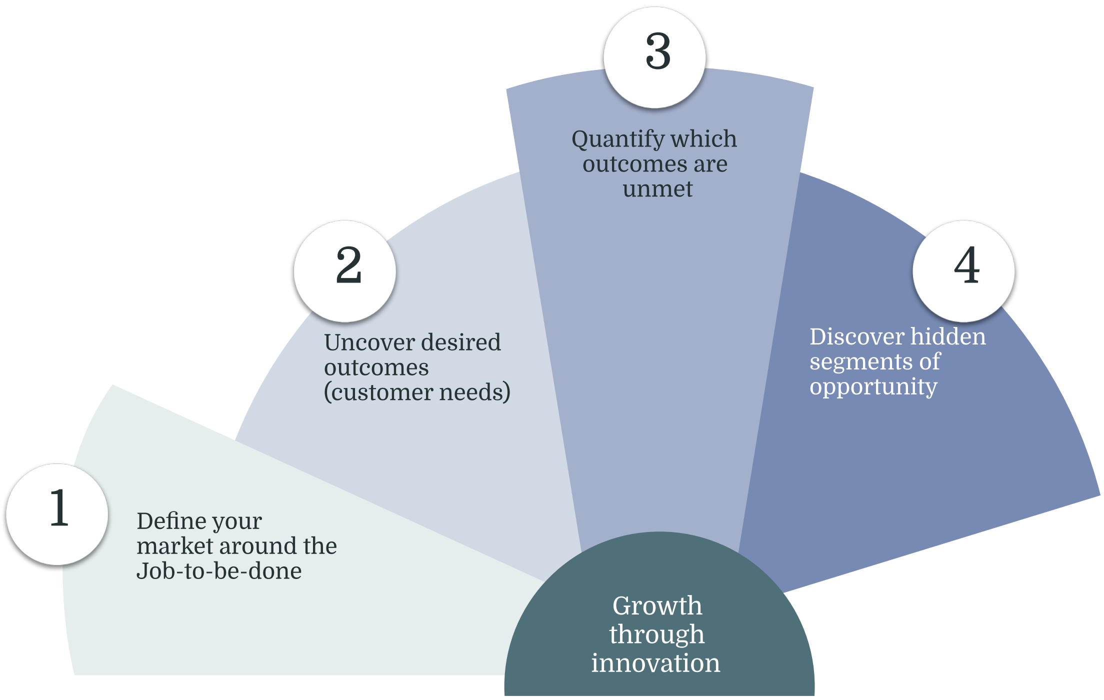
[36] *QuantUX Blog.* "Individual Scores in Choice Models, Part 1: Data & Averages." *QuantUXBlog.com.* Available at: https://quantuxblog.com/individual-scores-in-choice-models-part-1-data-averages.

[37] What Is MaxDiff? Available at: https://sawtoothsoftware.com/help/lighthouse-studio/manual/what-is-maxdiff.html

[38] Guinn, A. (n.d.). Stated vs. Derived Importance: What's the Difference? Decision Analyst. Available at: https://www.decisionanalyst.com/blog/stated-vs-derived-importance/

[39] Sawtooth Software. (n.d.). Sample Size Calculator. Available at:
https://sawtoothsoftware.com/resources/sample-size-calculator

Section 5 Overview



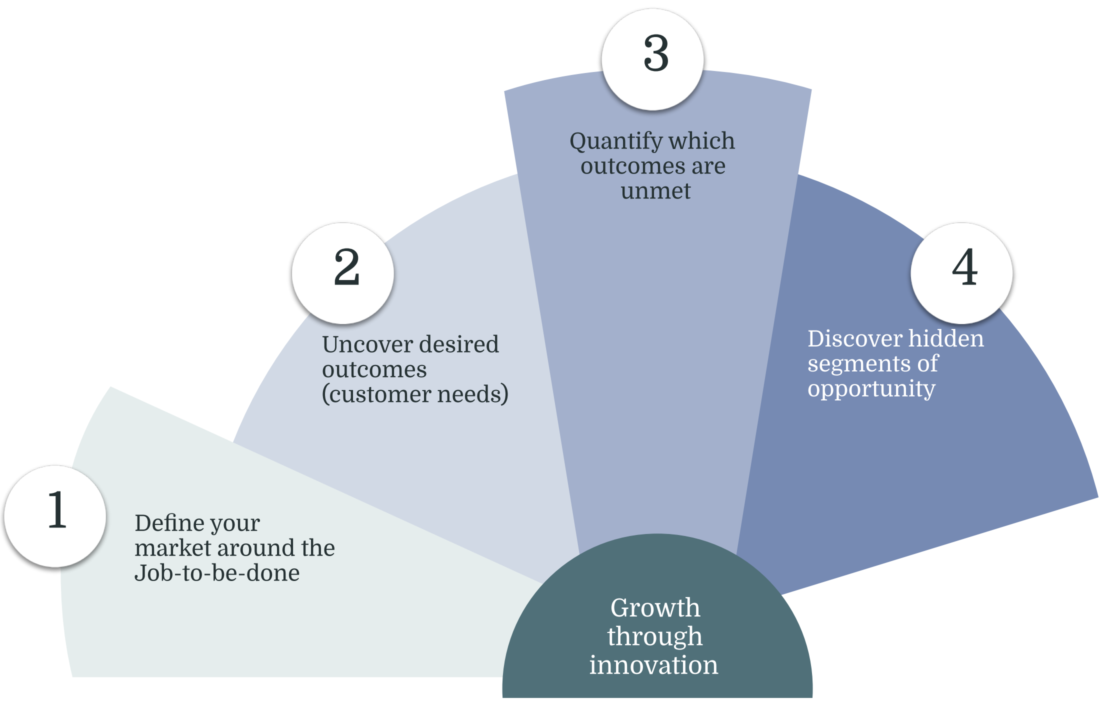
Discover Hidden Segments of Opportunity

This is the fourth step in the ODI process called "Discover hidden segments of opportunity." This is where we can start identifying distinct groups with distinctly different priorities based on their underlying needs and complexity factors. While Strategyn's traditional approach relies on opportunity scores from Likert scale data (with the limitations we explored in Chapters 7 and 8), I will explore how MaxDiff data can be used for segmentation as well. Using our Coffee MaxDiff example, we'll walk through clustering techniques that can help segment your data.

DISCOVER HIDDEN SEGMENTS OF OPPORTUNITY

Chapter 9: Needs-Based Segmentation

This chapter focuses on the next step after prioritizing needs: understanding that different groups of customers have distinctly different priorities. We will explore how to use the MaxDiff data from Chapter 8 to find these segments.



Outcome-needs based segmentation approach by Strategyn

Before we talk about segmentation

Before we look at the methods, remember: there is **no single right approach**. Anyone trying to sell you some perfect, universal method is likely misleading you. Thousands of segmentation techniques exist, each with different strengths and applications. The key is choosing the method that best answers your specific business questions, works well with your available data, and helps you make decisions that **work**.

Some segmentation approaches focus on demographics (age, income, location). Others look at behaviors (how people shop, what they buy, when they engage). Still others examine attitudes and values. Needs-based segmentation, which we'll focus on here, groups people by what they're trying to accomplish and what matters most to them in that process.

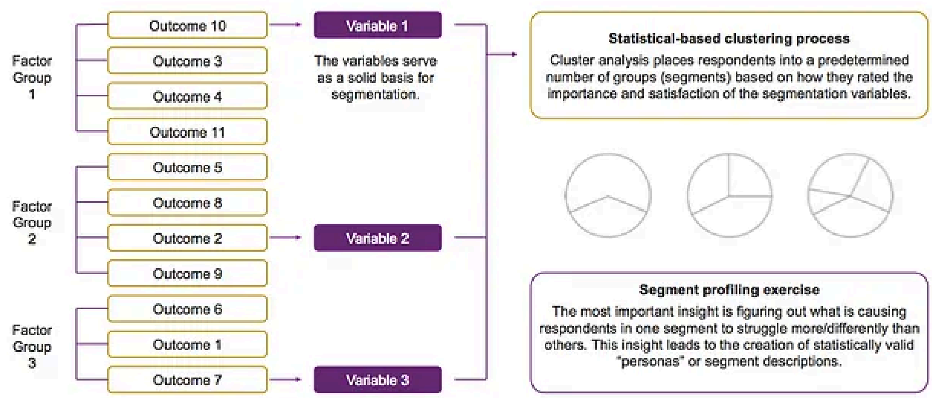
The right segmentation method depends entirely on your situation. If you're launching a new product, you might segment by unmet needs to find the biggest opportunities. If you're optimizing marketing spend, behavioral segmentation might work better. If you're expanding internationally, geographic and cultural segments could be most relevant.

What makes segmentation valuable isn't the sophistication of the statistical technique or the number of variables you include. It's whether the segments you create **help you understand your customers better** and **make more effective business decisions**.

The best segmentation is often the simplest one that still captures the differences that matter for your specific challenges.

This chapter will show you how to use MaxDiff data to create needs-based segments. But remember that this is just one tool in a much larger toolkit. The goal is finding groups of customers whose needs are different enough that they warrant different approaches, products, or messages.

The Original ODI Segmentation Method Approach



Outcome-needs based segmentation approach by Strategyn

Before we explore how to use MaxDiff data for segmentation, we should understand the traditional approach that established needs-based segmentation in the first place. The Outcome-Driven Innovation (ODI) methodology by Tony Ulwick and Strategyn created a process for segmenting customers based on their unmet needs. [40]

The classic ODI segmentation process follows three main steps:

1. Data Collection Researchers gather data on both the **importance** and **satisfaction** of dozens of need statements using traditional Likert scales. Customers rate statements like "minimize the time it takes to resolve an issue" on importance (typically 1-5 scale) and then rate how satisfied they are with current solutions on the same scale. This dual measurement allows researchers to identify gaps where something is important but current solutions fall short.

2. Factor Analysis The first step uses a statistical technique called **factor analysis** to manage the complexity of dozens (or more) of individual need statements. This identifies which needs tend to correlate with each other and groups them into a smaller number of underlying themes or "factors."

For example, needs like "minimize the time it takes to get support," "quickly get answers to my questions," and "resolve issues on the first contact" might all load together into a single factor that researchers would label "Responsiveness." Similarly, needs related to data security, privacy protection, and system reliability might group into a "Trust and Security" factor.

Instead of trying to segment customers based on their ratings of 50-125+ individual needs, you can work with 5-8 meaningful factors that capture the key themes. Distinguish between these two steps. Factor analysis groups the needs to simplify the list of questions. Cluster analysis groups the people based on those simplified factors.

3. Cluster Analysis The final step applies **cluster analysis** algorithms to the factor scores from the previous step. This statistical process groups individual respondents into segments based on how they rated the importance and satisfaction of these underlying factors. The algorithm identifies natural groupings where people within each segment have similar patterns of unmet needs.

The result is typically 3-5 segments, each with distinct profiles. One segment might show high unmet needs around speed and convenience, while another prioritizes quality and reliability over everything else.

4. Segment Profiling Once the statistical clustering is complete, researchers analyze what makes each segment unique. This involves analyzing not just the needs that define each group, but also their demographics, behaviors, and complexity factors that can help explain the data.

This traditional approach established the foundation for needs-based segmentation and proved that customers with different priority patterns often require different solutions. However, the method inherits some of the potential biases we discussed in Chapter 7 around Likert scale data, particularly issues with response bias and the challenge of comparing importance ratings across different people.

The MaxDiff-based approach we'll explore next builds on these same principles while addressing some of the measurement challenges inherent in traditional rating scales.

Segmenting with MaxDiff Data

With MaxDiff utility scores in hand, you have several options for identifying customer segments. Each method has its strengths and appropriate use cases. Here are the three main approaches researchers use to segment MaxDiff data.

Method 1: Latent Class Analysis (LCA)

Latent Class Analysis is often considered the most robust method for segmenting choice-based data like MaxDiff. This model-based approach works differently from traditional clustering methods because it doesn't just group people after the fact. As noted by Wedel and Kamakura (2000), this model-based approach assumes your sample contains hidden subgroups with different preference patterns, and it simultaneously identifies these groups while estimating what each group's preferences look like. [13]

Think of LCA as working backwards from the patterns in your data. Rather than starting with individual utility scores and then clustering them, LCA asks "what if there are actually three distinct types of customers in this data, each with their own preference pattern?" It then tests whether this assumption explains the observed MaxDiff choices better than assuming two groups, or four groups, or treating everyone as homogeneous.

The method provides clear statistical measures to help you decide on the optimal number of segments. Fit statistics like the Bayesian Information Criterion (BIC) and Akaike Information Criterion (AIC) give you objective ways to compare different segment solutions. Generally, lower values indicate better model fit, helping remove some of the guesswork from deciding whether you have two segments or five.

LCA also handles uncertainty well. Instead of definitively assigning each person to a single segment, it calculates the probability that each respondent belongs to each segment. This probabilistic assignment can be valuable for understanding borderline cases and the stability of your segmentation.

The main drawback is complexity. LCA requires specialized software and can feel like a black box if you're not comfortable with statistical modeling. The results also require more interpretation than simpler clustering methods.

Method 2: K-Means Clustering

K-Means offers a more intuitive, algorithm-based approach that many researchers find easier to understand and implement. The method works by treating each person's MaxDiff utility scores as coordinates in multi-dimensional space. If you have ten attributes in your MaxDiff study, each respondent becomes a point in ten-dimensional space based on their utility scores for those attributes.

The K-Means algorithm then searches for the best way to place cluster "centers" in this space and assigns each person to their nearest center. The algorithm iteratively moves these centers around until it finds the configuration that minimizes the total distance between all points and their assigned centers.

This approach is simple and fast. You can visualize what's happening even if you can't easily draw ten-dimensional space. K-Means is also widely available in most statistical software packages and even Excel plugins.

However, K-Means requires you to specify the number of clusters upfront. You need to decide whether you want three segments or five segments before running the analysis. This often means running multiple analyses with different numbers of clusters and comparing the results. The method can also be sensitive to outliers, since a few people with unusual preference patterns can pull cluster centers away from more typical respondents.

Additionally, K-Means assumes clusters are roughly spherical and similarly sized, which may not match the actual structure in your data. If one segment represents 60% of your market while another represents 10%, K-Means might not identify this naturally.

Method 3: Two-Step or Hybrid Approaches

Many experienced researchers use hybrid approaches that combine the exploratory power of one method with the stability of another. The most common version starts with Hierarchical Clustering to explore the data structure, then uses those insights to inform a K-Means analysis for the final segmentation.

Hierarchical Clustering works like building a family tree in reverse. It starts by treating each person as their own cluster, then iteratively combines the two most similar clusters until everyone is grouped together. This creates a tree-like structure (called a dendrogram) that shows how clusters combine at each step.

The advantage of starting with Hierarchical Clustering is that it doesn't require you to specify the number of clusters beforehand. You can examine the dendrogram to see where natural breaks occur and identify the most meaningful number of segments. This exploratory step gives you insight into the data structure that pure K-Means clustering might miss.

Once you've identified the optimal number of clusters from the hierarchical analysis, you can use that number as input for K-Means clustering. This gives you the stability and interpretability of K-Means while removing the guesswork about how many segments to create.

Some researchers extend this approach even further, using the hierarchical results to inform starting points (called seeds) for the K-Means algorithm. This can help ensure that K-Means finds the global optimum rather than getting stuck in a local minimum.

The main drawback of hybrid approaches is complexity. You're running multiple analyses and making decisions at each step, which requires more time and expertise. The process can feel more like art than science, especially when interpreting hierarchical clustering results.

Each of these methods can produce valuable segmentations, and the best choice often depends on your specific situation, data characteristics, and comfort level with different analytical approaches. The key is choosing a method that gives you segments you can understand, act upon, and defend to stakeholders.

Segmenting our MaxDiff Data: Step-by-step approach

Download MaxDiff data from Chapter 8

Before we begin segmenting, let's review a bit about the data. The coffee preference data contains responses from 400 consumers across 15 MaxDiff attributes. Each attribute received a preference score representing its relative importance to each individual customer. These scores show substantial variation across respondents, suggesting that meaningful segments exist but aren't immediately obvious.

The challenge is to group customers with similar preference patterns while ensuring that the resulting segments are both statistically sound and practically useful a business. As we'll see, we have to balance statistical fit with business reality.

We'll start by examining how different clustering approaches handle the same underlying customer preference data, starting with the most common method in marketing research: k-means clustering.

First Attempt: Understanding K-Means Clustering

K-means clustering is the most widely used segmentation method in marketing research, and for good reason. It's computationally efficient, conceptually straightforward, and often produces interpretable results. However, as we will learn, its simplicity can also be a limitation when dealing with complex customer preference data.

How K-Means Works

Before diving into the analysis, it's worth understanding what k-means does. The algorithm follows a simple process:

1. **Initialize**: Place k cluster centers randomly in the data space
2. **Assign**: Assign each customer to the nearest cluster center
3. **Update**: Move each cluster center to the average position of its assigned customers
4. **Repeat**: Continue assigning and updating until cluster centers stop moving

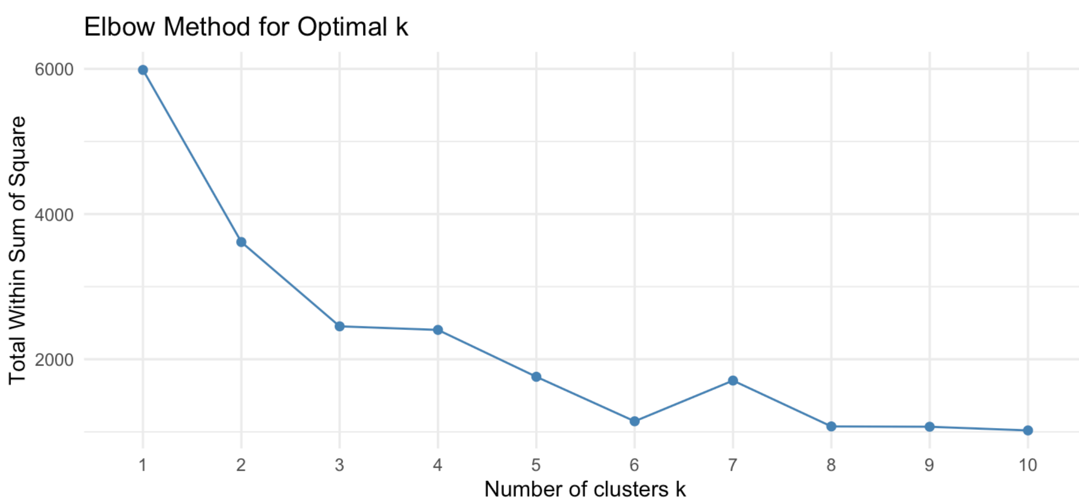
This process guarantees that customers within each cluster are as similar as possible to their cluster center, while being as different as possible from other cluster centers. However, as detailed in clustering reviews like Jain (2010), k-means makes several assumptions that may not hold for all datasets: [14]

- Clusters should be roughly spherical (circular in 2D, ball-shaped in higher dimensions)
- Clusters should be of similar sizes
- All variables should be equally important
- The optimal number of clusters should be specified in advance

Finding the Right Number of Clusters

The biggest challenge with k-means is determining how many clusters to create. I tested three statistical methods to help guide this decision:

```r
library(factoextra)

# Elbow method - looks for the "elbow" in within-cluster sum
fviz_nbclust(scaled_data, kmeans, method = "wss", k.max = 10
  ggtitle("Elbow Method for Optimal k")
```

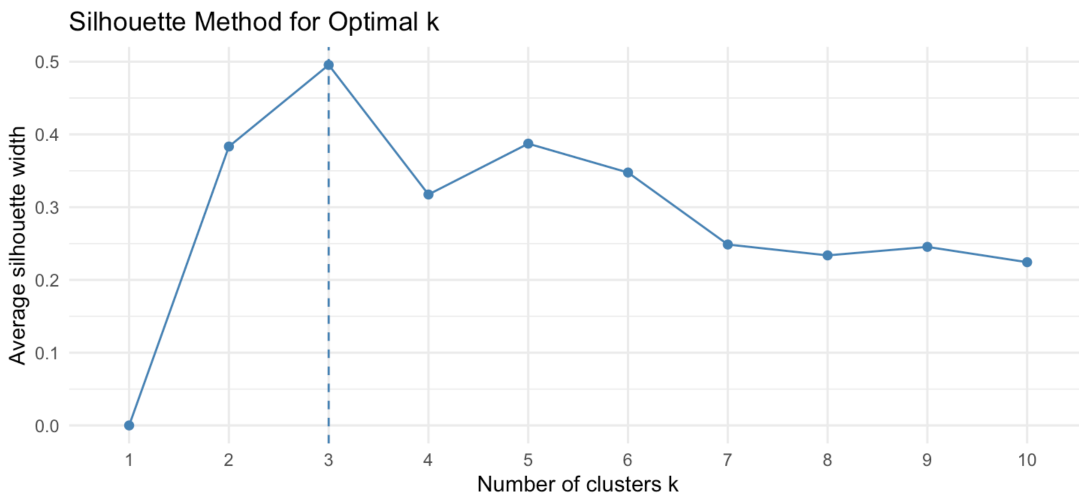


4 Cluster Elbow Plot

The elbow method plots the within-cluster sum of squares (WSS) for different numbers of clusters. WSS measures how tightly customers cluster around their assigned centers. As you add more clusters, WSS always decreases because customers get closer to their cluster centers. The "elbow" occurs where adding another cluster provides diminishing returns.

In my results, WSS dropped dramatically from k=1 to k=2 (from around 6000 to 3500), which makes sense because forcing all customers into one group creates high variation. The decline continued more gradually afterward, with potential elbows at k=3 (WSS around 2800) and k=6 (WSS around 2200). This gradual decline without a clear elbow suggested that the natural cluster structure might not be obvious.

```r
# Silhouette method – finds k that maximizes average silhou
fviz_nbclust(scaled_data, kmeans, method = "silhouette", k.m
  ggtitle("Silhouette Method for Optimal k")
```



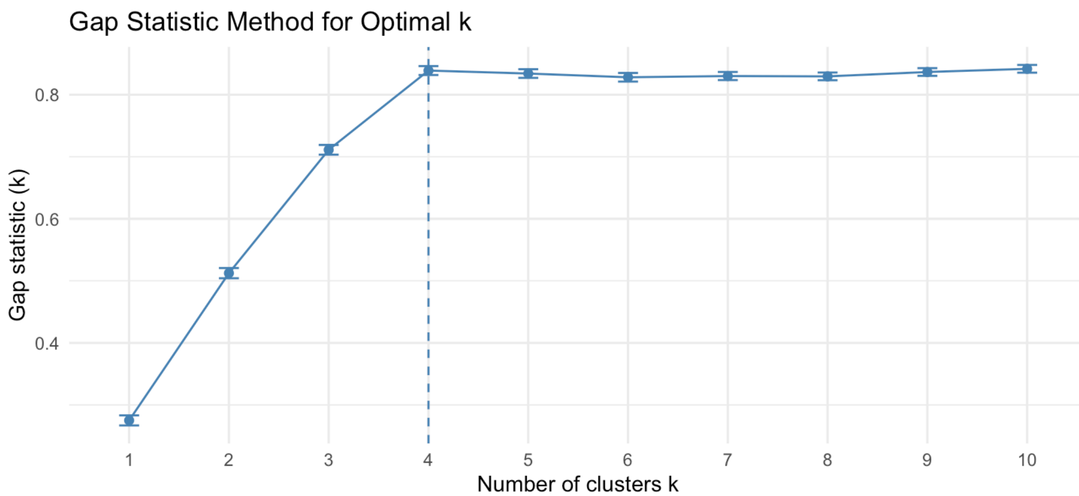
Silhouette Method for Optimal k

The silhouette method evaluates cluster quality rather than just within-cluster tightness. For each possible number of clusters, it calculates how well customers fit in their assigned clusters compared to alternative clusters. Higher average silhouette scores indicate better separation between clusters.

My results showed a clear peak at k=3 with a silhouette score around 0.50. This peak suggests that three is a point of strong, natural separation in the data. While other solutions might produce slightly higher average scores, the silhouette plot highlights k=3 as a structurally sound and interpretable option. Scores declined for k=4 and k=5 relative to this peak, suggesting that adding more clusters beyond three was creating weaker or more artificial divisions.

R

```r
# Gap statistic - compares clustering structure to random d
set.seed(123)
gap_stat <- clusGap(scaled_data, FUN = kmeans, nstart = 25,
fviz_gap_stat(gap_stat)
```



Gap statistic 4 Segment K-means

The gap statistic uses a more sophisticated approach. It compares the within-cluster dispersion of your actual data to what you'd expect from randomly distributed data. The optimal k occurs where the gap between your data's structure and random structure is largest.

The gap statistic showed its steepest increase from k=3 to k=4, then plateaued. This pattern suggested that k=4 captured meaningful structure that wouldn't appear in random data. However, the relatively modest gap values (around 0.15) indicated that while structure existed, it wasn't extremely strong.

Interpreting Conflicting Signals

These three methods provided conflicting recommendations, which is common in real segmentation work:

- **Elbow method**: Ambiguous, potential stopping points at k=3 or k=6
- **Silhouette method**: Strong recommendation for k=3
- **Gap statistic**: Suggestion for k=4

Faced with these mixed signals, I made a business judgment to start with k=4. My reasoning was that coffee consumers might naturally fall into four, or more, behavioral types based on different priorities: quality seekers, convenience seekers, budget-conscious consumers, and experience seekers.

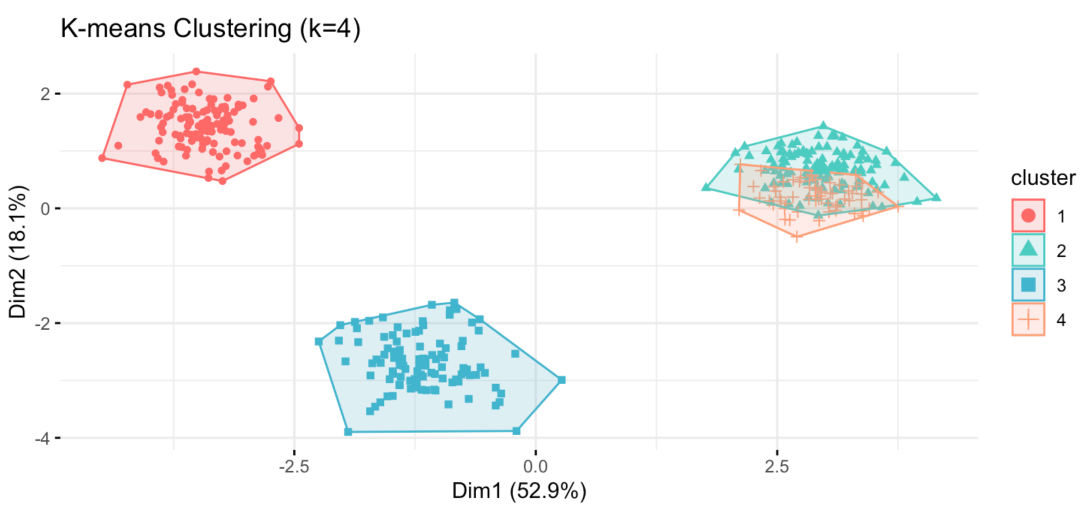
Implementing K-Means with Four Clusters

```r
# Perform k-means clustering
set.seed(123)
k4_clusters <- kmeans(scaled_data, centers = 4, nstart = 25,
```

Let me explain each parameter in this code:

- `set.seed(123)`: K-means starts with random cluster centers, so setting a seed ensures reproducible results. Without this, you might get slightly different clusters each time you run the analysis.
- `centers = 4`: Specifies that we want four clusters.
- `nstart = 25`: Runs the k-means algorithm 25 times with different random starting points and keeps the best result. This matters because k-means can get stuck in local optima (good solutions that aren't the best possible solution).
- `iter.max = 300`: Maximum number of iterations allowed. K-means usually converges quickly, but this ensures the algorithm has enough time to find stable cluster centers.

```r
# Add cluster assignments to original data
maxdiff_segmented <- maxdiff_chapter8_example
maxdiff_segmented$cluster <- as.factor(k4_clusters$cluster)

# Visualize clusters using PCA
fviz_cluster(k4_clusters, data = scaled_data,
             palette = c("#FF6B6B", "#4ECDC4", "#45B7D1", "#
             geom = "point",
             ellipse.type = "convex",
             ggtheme = theme_minimal()) +
  ggtitle("K-means Clustering (k=4)")
```



K-means clustering k=4

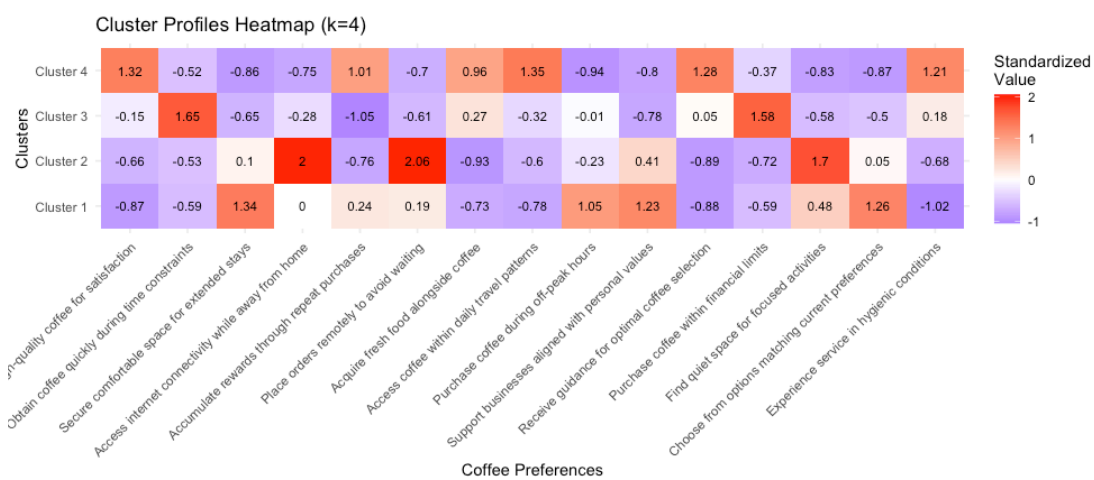
The visualization revealed problems. The `fviz_cluster()` function uses Principal Component Analysis (PCA) to project the 15-dimensional preference data into two dimensions for plotting. While this projection inevitably loses some information, it provides a useful overview of cluster separation.

The plot showed that while Cluster 1 (red circles) and Cluster 3 (blue squares) appeared well-separated, Clusters 2 (teal triangles) and 4 (orange squares) exhibited substantial overlap in the center-right area. This visual overlap was concerning because well-separated clusters should appear as distinct groups with minimal boundary overlap.

Examining Cluster Profiles

To understand what these clusters represented, let's create a heatmap showing each cluster's preferences:

```r
# Create heatmap of cluster profiles
cluster_centers <- as.data.frame(k4_clusters$centers)
cluster_centers$cluster <- paste("Cluster", 1:4)

heatmap_data <- melt(cluster_centers, id.vars = "cluster")

ggplot(heatmap_data, aes(x = variable, y = cluster, fill = v
  geom_tile() +
  geom_text(aes(label = round(value, 2)), color = "black", s
  scale_fill_gradient2(low = "blue", mid = "white", high = "
                       midpoint = 0, name = "Standardized\nV
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  labs(title = "Cluster Profiles Heatmap (k=4)",
       x = "Coffee Preferences",
       y = "Clusters")
```



K-means Cluster Heatmap k=4

The heatmap displays standardized cluster centers, where:

- **Red colors** indicate above-average preference for an attribute
- **Blue colors** indicate below-average preference
- **White colors** indicate average preference
- **Numbers** show the exact standardized values

Reading the heatmap, we can identify each cluster's characteristics:

- **Cluster 1**: High values for quick service (1.65) and budget constraints (1.68), negative for quality and atmosphere
- **Cluster 2**: Positive for convenience features like rewards (1.01) and remote ordering (1.28)
- **Cluster 3**: Strong preference for quality coffee (0.93) and supporting aligned businesses (0.96)
- **Cluster 4**: Similar pattern to Cluster 2, with positive values for convenience and negative for quality

The similarity between Clusters 2 and 4 seemed to overlap. Both showed nearly identical patterns across multiple variables, differing mainly in magnitude rather than direction of preferences. This suggested they might represent variations within a single customer type rather than truly distinct segments.

Quality Assessment: Silhouette Analysis

To quantify the clustering quality, lets also calculate silhouette scores:

```r
# Validation metrics
sil <- silhouette(k4_clusters$cluster, dist(scaled_data))
print(paste("Average silhouette width:", round(mean(sil[, 3]

[1] "Average silhouette width: 0.509"
```

> *need to explain what this means and provide the output above*

The silhouette analysis provides both overall and cluster-specific quality measures. For each customer, it compares how well they fit in their assigned cluster versus the next best alternative. The calculation involves:

1. **a(i)**: Average distance from customer i to other customers in the same cluster
2. **b(i)**: Average distance from customer i to customers in the nearest different cluster
3. **Silhouette score = (b(i) - a(i)) / max(a(i), b(i))**

Interpreting silhouette scores:

- **0.7 to 1.0**: Excellent clustering, customers clearly belong in their assigned cluster
- **0.5 to 0.7**: Good clustering, reasonable separation between clusters
- **0.3 to 0.5**: Weak clustering, some customers might fit better elsewhere
- **Below 0.3**: Poor clustering, artificial or forced groupings likely
- **Negative**: Customer fits better in a different cluster than their assigned one

My results showed:

- **Overall average**: 0.509 (just above the acceptable threshold)
- **Cluster 1**: 0.53 (good separation)
- **Cluster 2**: 0.48 (borderline quality)
- **Cluster 3**: 0.55 (good separation)

- **Cluster 4**: 0.44 (weak separation, below recommended threshold)

These scores revealed the core problem with my four-cluster solution. While two clusters achieved good separation, the other two fell into the borderline to weak range. Cluster 4's score of 0.44 particularly concerned me because it suggested forced grouping of customers who might naturally belong elsewhere.

Understanding the Cluster Size Distribution

```r
# Examine cluster sizes
table(k4_clusters$cluster)

  1   2   3   4
121  59 100 120
```

The cluster sizes also revealed potential issues:


```r
# Examine cluster sizes
table(k4_clusters$cluster)

  1   2   3   4
121  59 100 120
```


- Cluster 1: 121 customers (30%)
- Cluster 2: 59 customers (15%)
- Cluster 3: 100 customers (25%)
- Cluster 4: 120 customers (30%)

Cluster 2 was notably smaller than the others, which can be a warning sign. Small clusters sometimes emerge when an algorithm tries to separate a handful of outliers or when it artificially splits a larger, more natural group. In this case, its small size, combined with the poor silhouette score and visual overlap, strongly suggested that this four-cluster solution was not stable.

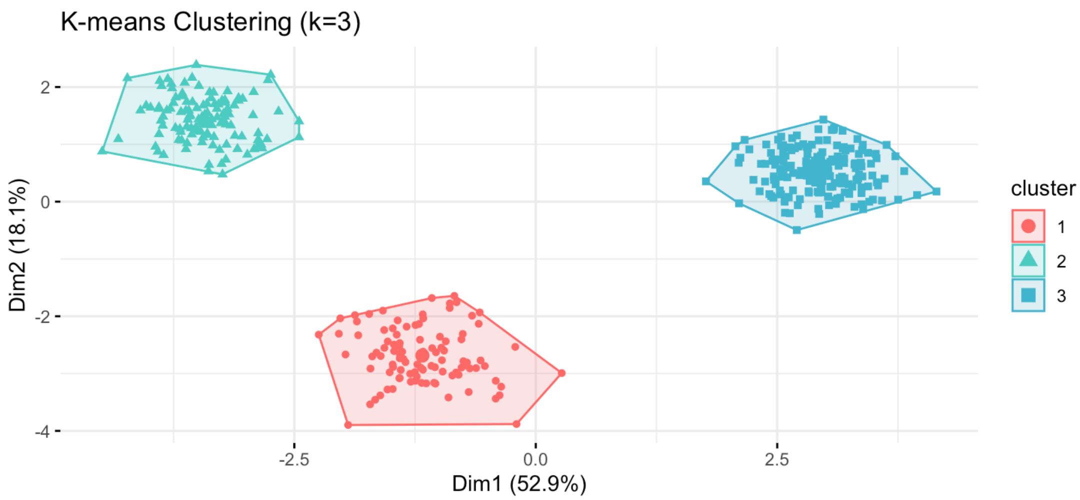
Trying Three K-Means Clusters

The problems with my four cluster solution forced me to reconsider the silhouette method's strong recommendation for three clusters. Sometimes stepping back from initial business assumptions leads to better results, and this proved to be one of those moments.

Implementing the Three-Cluster Solution

R

```r
# Perform k-means clustering with k=3
set.seed(123)
k3_clusters <- kmeans(scaled_data, centers = 3, nstart = 25,

# Add cluster assignments to original data
maxdiff_segmented$cluster <- as.factor(k3_clusters$cluster)

# Visualize clusters using PCA
fviz_cluster(k3_clusters, data = scaled_data,
             palette = c("#FF6B6B", "#4ECDC4", "#45B7D1"),
             geom = "point",
             ellipse.type = "convex",
             ggtheme = theme_minimal()) +
  ggtitle("K-means Clustering (k=3)")
```



K-means Cluster k=3

The improvement was clear. The PCA visualization showed much cleaner separation between segments, with minimal overlap between the convex hull boundaries. The problematic overlapping clusters from my four-segment solution had been consolidated into more coherent groupings.

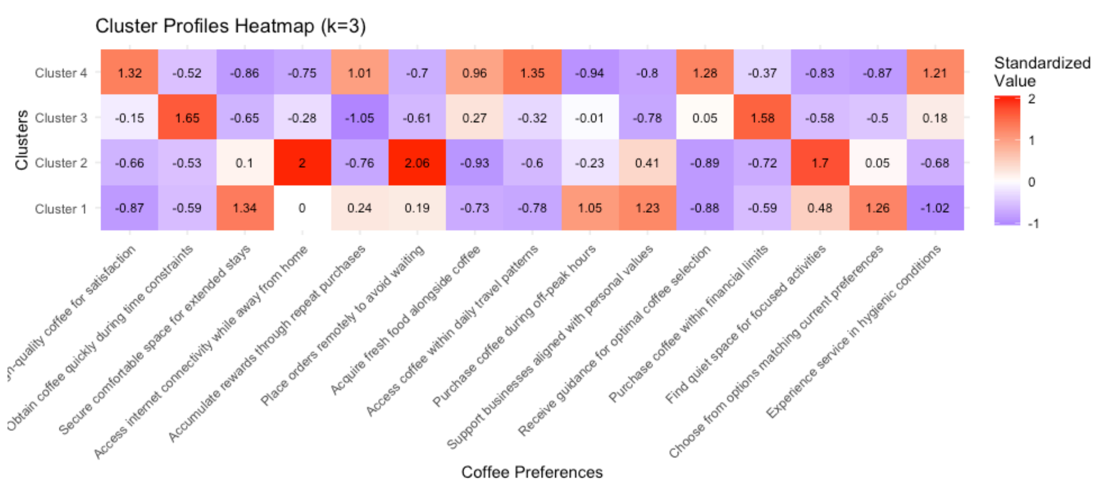
Where the four-cluster solution showed Clusters 2 and 4 bleeding into each other in the center-right area of the plot, the three-cluster solution created clear boundaries. Each cluster occupied its own distinct region of the preference space, with Cluster 1 (red circles) positioned in the lower portion, Cluster 2 (teal triangles) in the upper left, and Cluster 3 (blue squares) on the right side.

The ellipses around each cluster also appeared more natural. In clustering visualizations, these ellipses represent the approximate boundaries within which most cluster members fall. Tighter, more circular ellipses indicate cohesive segments, while elongated or overlapping ellipses suggest internal heterogeneity or unclear boundaries. The three-cluster ellipses were more compact and showed minimal overlap compared to the four-cluster attempt.

Examining the New Cluster Profiles

The real test of improvement came when I examined what these three clusters represented in terms of coffee preferences:

```r
# Create updated heatmap for 3 clusters
cluster_centers <- as.data.frame(k3_clusters$centers)
cluster_centers$cluster <- paste("Cluster", 1:3)

heatmap_data <- melt(cluster_centers, id.vars = "cluster")

ggplot(heatmap_data, aes(x = variable, y = cluster, fill = v
  geom_tile() +
  geom_text(aes(label = round(value, 2)), color = "black", s
  scale_fill_gradient2(low = "blue", mid = "white", high = "
                        midpoint = 0, name = "Standardized\nV
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  labs(title = "Cluster Profiles Heatmap (k=3)",
       x = "Coffee Preferences",
       y = "Clusters")
```



K-means Cluster Heatmap k=3

The three-cluster heatmap revealed much more distinct and interpretable preference patterns compared to the overlap between Clusters 2 and 4 in my previous attempt. Each cluster now displayed clear, differentiated preferences that made business sense:

Cluster 1: Quick & Budget-Conscious Consumers This segment showed the highest positive values for speed-related attributes, with "obtaining coffee quickly during time constraints" scoring 1.65 and "purchasing within financial limits" at 1.68. These customers prioritized efficiency and value. Correspondingly, they showed negative values for quality-focused attributes like "consuming high-quality coffee for satisfaction" (-0.98) and comfort features like "accessing comfortable spaces away from home" (-0.45). This profile painted a clear picture of customers who view coffee primarily as a functional necessity rather than an experience.

Cluster 2: Convenience-Focused Customers This group displayed strong positive values for modern convenience features. They scored highly on "accumulating rewards through loyalty programs" (1.01), "accessing convenient locations" (1.35), and "placing remote orders ahead of arrival" (1.28). However, like Cluster 1, they showed negative values for traditional quality attributes such as "consuming high-quality coffee" (-0.72) and "supporting businesses aligned with personal values" (-0.55). This suggested customers who wanted coffee to fit seamlessly into their busy lifestyles through digital integration and location convenience, but weren't willing to pay premium prices for quality.

Cluster 3: Quality & Experience Seekers The largest cluster showed an entirely different preference pattern. They demonstrated high positive values for "consuming high-quality coffee for satisfaction" (0.93), "supporting businesses aligned with personal values" (0.96), and "experiencing hygienic conditions" (0.86). Conversely, they showed negative values for quick service needs (-0.71) and budget constraints (-0.62). This profile suggested customers who viewed coffee as an experience worth investing in, both financially and temporally.

Understanding Cluster Sizes and Market Implications

```r
# Examine cluster sizes and proportions
table(k3_clusters$cluster)
prop.table(table(k3_clusters$cluster))

  1   2   3
100 120 180

   1    2    3
0.25 0.30 0.45
```

The cluster distribution revealed interesting market dynamics:

- **Cluster 1 (Quick & Budget-Conscious)**: 100 customers (25%)
- **Cluster 2 (Convenience-Focused)**: 120 customers (30%)
- **Cluster 3 (Quality & Experience Seekers)**: 180 customers (45%)

The fact that Cluster 3 contained nearly half of all respondents suggested that quality and experience orientation might be the dominant preference pattern among coffee consumers in this sample. However, this large segment size also raised a flag that we'd need to investigate further. Sometimes when one cluster becomes too large, it indicates that heterogeneous customers are being forced together because the algorithm can't find enough distinct patterns to separate them properly.

Quality Assessment: Improved But Not Perfect

The quality metrics showed marked improvement over the four-cluster solution:

```r
# Validation metrics for k=3
sil_3 <- silhouette(k3_clusters$cluster, dist(scaled_data))
print(paste("Average silhouette width:", round(mean(sil_3[,

# Detailed cluster breakdown
summary(sil_3)
```

summary(sil_3) output

R OUTPUT

```
il_3)

> Silhouette of 400 units in 3 clusters from silhouette.default(x = k3_cluster
> Cluster sizes and average silhouette widths:

    100       120       180

0.5537768 0.5331681 0.4379362
Individual silhouette widths:
Min. 1st Qu. Median Mean 3rd Qu. Max.
0.2924 0.4397 0.5028 0.4955 0.5565 0.6708
```

The results were encouraging:

- **Overall average silhouette width**: 0.495 (approaching the 0.5 threshold for good clustering)
- **Cluster 1**: 0.55 (good separation)
- **Cluster 2**: 0.53 (good separation)
- **Cluster 3**: 0.44 (weak separation, but the only problematic cluster)

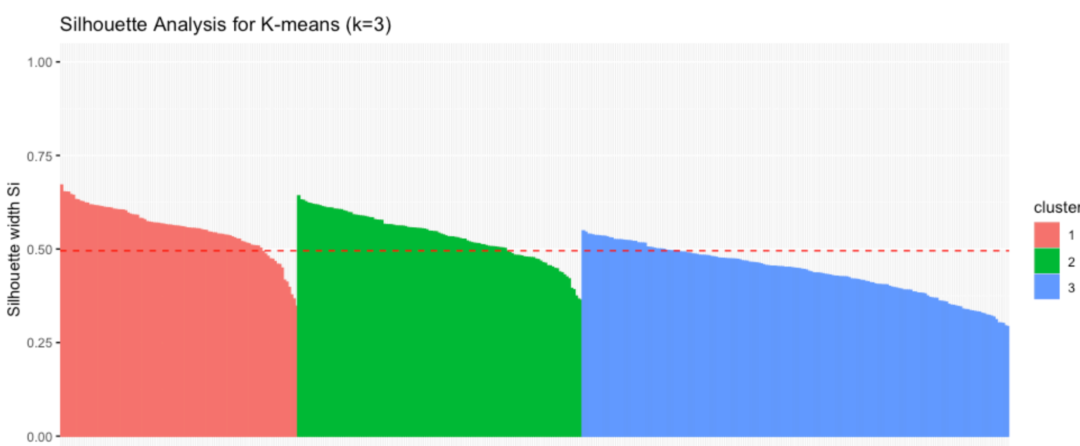
The improvement was clear when comparing to the four-cluster attempt. Instead of having two problematic clusters with scores below 0.48, I now had only one cluster with weak separation. More significantly, the two smaller clusters achieved good separation scores above 0.53, indicating that they represented genuine, well-defined customer segments.

However, the weak score for Cluster 3 was concerning, especially given that it contained 180 customers (45% of the sample). A silhouette score of 0.44 suggested that many customers in this cluster were nearly as similar to members of other clusters as they were to their cluster mates. This could indicate that Cluster 3 contained multiple sub-groups that the three-cluster solution couldn't differentiate.

Visualizing Individual Customer Fit

To better understand the quality issues, I examined the silhouette plot:

```r
# Create silhouette plot
fviz_silhouette(sil_3) +
  ggtitle("Silhouette Analysis for K-means (k=3)")

  cluster size ave.sil.width
1       1  100          0.55
2       2  120          0.53
3       3  180          0.44
```



Silhouette Plot 3 Segment

The silhouette plot displays individual customer scores sorted by cluster and score magnitude. Each bar represents one customer, with longer bars indicating better fit within their assigned cluster. Negative bars indicate customers who might fit better in a different cluster.

The plot revealed several insights:

Cluster 1 and 2: Most customers showed positive silhouette scores between 0.4 and 0.7, with few negative values. This confirmed that these segments contained customers with genuinely similar preferences who were well-separated from other groups.

Cluster 3: While most customers had positive scores, there was more variation and a longer tail of customers with scores near zero. Some customers even showed negative scores, suggesting they might be misassigned. The heterogeneity within this large cluster supported my suspicion that it might contain multiple sub-segments.

Exploring Cluster 3's Internal Structure

Given the size and quality concerns with Cluster 3, I investigated its internal composition:

```r
# Examine Cluster 3 customers in detail
cluster3_data <- scaled_data[k3_clusters$cluster == 3, ]

# Look at variation within Cluster 3
apply(cluster3_data, 2, sd)

# Compare to variation in other clusters
cluster1_data <- scaled_data[k3_clusters$cluster == 1, ]
cluster2_data <- scaled_data[k3_clusters$cluster == 2, ]

mean(apply(cluster1_data, 2, sd))  # Average variation in Cl
mean(apply(cluster2_data, 2, sd))  # Average variation in Cl
mean(apply(cluster3_data, 2, sd))  # Average variation in Cl
```

summary(sil_3) output

R OUTPUT

```
ster3_data, 2, sd)

   Consume high-quality coffee for satisfaction     Obtain coffee quickly duri
                                    0.3097366
    Secure comfortable space for extended stays Access internet connectivity w
                                    0.7651981
    Accumulate rewards through repeat purchases            Place orders remote
                                    0.6835872
            Acquire fresh food alongside coffee        Access coffee within da
                                    0.5099785
          Purchase coffee during off-peak hours   Support businesses aligned w
                                    0.8979413
  Receive guidance for optimal coffee selection          Purchase coffee with
                                    0.2073337
         Find quiet space for focused activities  Choose from options matching
                                    0.8389023
       Experience service in hygienic conditions
                                    0.2732193

> mean(apply(cluster1_data, 2, sd)) # Average variation in Cluster 1
> [1] 0.407853
> mean(apply(cluster2_data, 2, sd)) # Average variation in Cluster 2
> [1] 0.4076522
> mean(apply(cluster3_data, 2, sd)) # Average variation in Cluster 3
> [1] 0.582721
```

The analysis confirmed my suspicions. Cluster 3 showed higher average variation across preference attributes compared to Clusters 1 and 2. While the other clusters had relatively tight distributions around their center points, Cluster 3 contained customers with more diverse preference patterns that happened to be grouped together because they didn't fit clearly into the other two segments.

The Promise and Limitations of Three Clusters

The three-cluster solution represented an improvement over my four-cluster attempt in several ways:

Clearer Differentiation: Each cluster now had a distinct and interpretable preference profile. The confusion between similar clusters was eliminated, and each segment suggested different marketing approaches.

Better Statistical Quality: Two of the three clusters achieved good separation, and the overall silhouette score improved. The visual separation was much cleaner, with minimal overlap between cluster boundaries.

Actionable Insights: The three segments suggested clear marketing strategies. Quick & Budget-Conscious customers could be targeted with efficiency and value messaging. Convenience-Focused customers would respond to digital features and location strategies. Quality & Experience Seekers would appreciate premium positioning and values-based marketing.

However, key limitations remained:

Large Heterogeneous Segment: Cluster 3's size and internal variation suggested it might benefit from further subdivision. Nearly half of all customers fell into this category, which could limit the precision of targeted marketing efforts.

Moderate Quality Scores: While improved, the silhouette scores still fell short of the 0.7 threshold that indicates strong, well-separated clusters. This suggested that natural customer groupings in coffee preferences might be more subtle than extreme.

Potential for Refinement: The internal heterogeneity in Cluster 3 raised questions about whether a different approach might reveal additional meaningful segments within this large group.

This shows that segmentation quality involves more than just overall statistical measures. Even when average metrics improve, examining individual cluster performance can reveal opportunities for further refinement. The three-cluster solution was clearly better than my four-cluster attempt, but it wasn't necessarily the final answer to understanding customer segments in this coffee preference data.

Exploring a Different Approach: Hierarchical Clustering

After mixed results with k-means, I decided to test a different approach. K-means assumes spherical clusters of similar sizes, but perhaps my coffee preference data had different underlying structure that required a more flexible method.

Hierarchical clustering works differently from k-means in several key ways. Instead of starting with a predetermined number of clusters, it builds a tree-like structure called a dendrogram that shows how customers group together at different levels of similarity. Think of it like a family tree, but instead of showing genealogical relationships, it reveals preference relationships among customers.

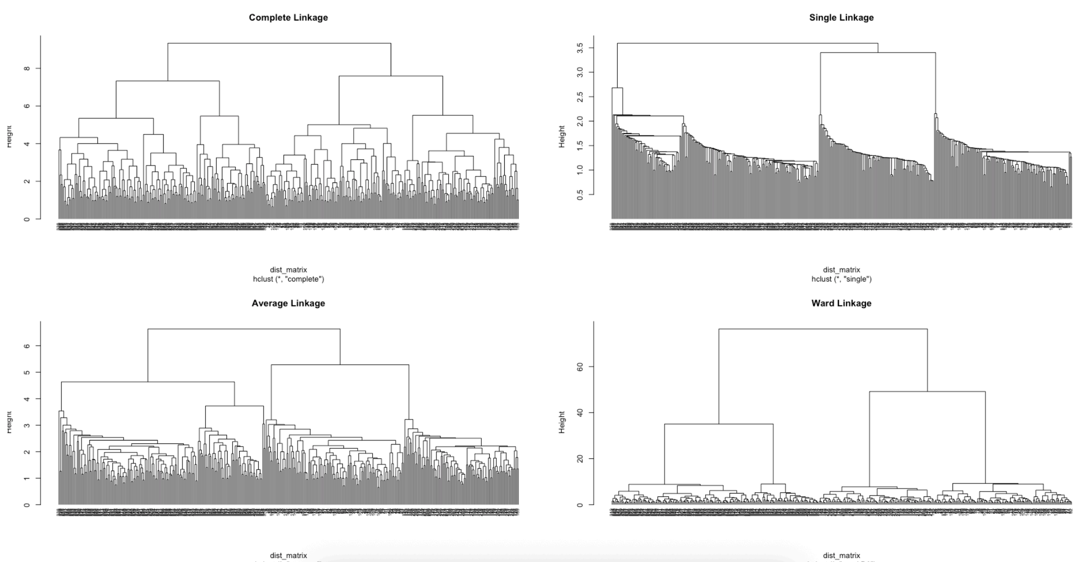
Understanding Linkage Methods

The first decision in hierarchical clustering involves choosing how to measure the distance between groups of customers. Different linkage methods can produce markedly different results:

```r
library(cluster)
library(dendextend)

# Calculate distance matrix
dist_matrix <- dist(scaled_data, method = "euclidean")
```

The distance matrix calculates how different each customer is from every other customer based on their coffee preferences. With 400 customers, this creates a 400x400 matrix containing 79,800 unique pairwise distances. Euclidean distance treats each preference like a coordinate in 15-dimensional space and calculates the straight-line distance between customers.

```r
# Test different linkage methods
hc_complete <- hclust(dist_matrix, method = "complete")
hc_single <- hclust(dist_matrix, method = "single")
hc_average <- hclust(dist_matrix, method = "average")
hc_ward <- hclust(dist_matrix, method = "ward.D2")

# Visualize dendrograms
par(mfrow = c(2, 2))
plot(hc_complete, main = "Complete Linkage", cex = 0.6, hang
plot(hc_single, main = "Single Linkage", cex = 0.6, hang = -
plot(hc_average, main = "Average Linkage", cex = 0.6, hang =
plot(hc_ward, main = "Ward Linkage", cex = 0.6, hang = -1)
par(mfrow = c(1, 1))
```



Testing Different Linkage Methods

Each linkage method defines distance between clusters differently:

- **Single linkage** uses the shortest distance between any two points in different clusters. This often creates long, chain-like clusters that may not reflect natural groupings.

- **Complete linkage** uses the maximum distance between points in different clusters. This tends to create compact, spherical clusters of similar sizes.
- **Average linkage** uses the average distance between all pairs of points in different clusters. This provides a middle ground between single and complete linkage.
- **Ward linkage** minimizes the within-cluster sum of squares when merging clusters. This method tends to create clusters of roughly equal size and is often preferred for customer segmentation because it produces the most interpretable results.

The dendrograms revealed clear differences between methods. Single linkage produced a few large clusters with many small outlier groups. Complete and average linkage showed more balanced structures, but Ward linkage produced the clearest branching pattern with distinct separation points that suggested natural customer groupings.

Reading the Dendrogram

A dendrogram shows the hierarchical relationship between all customers, with height on the y-axis representing the distance at which clusters merge. Lower heights indicate more similar customers, while higher heights show where dissimilar groups come together.

The Ward dendrogram revealed several interesting patterns:

- **Clear primary split**: The tree showed a major division around height 15, suggesting two fundamentally different customer types exist in the data.
- **Secondary branches**: Each major branch showed further subdivision around heights 8-10, indicating more nuanced differences within the broader customer types.
- **Stable clusters**: Some groups of customers clustered together at low heights (2-4), suggesting these individuals have nearly identical preferences.

To convert the hierarchical structure into discrete segments, I needed to "cut" the tree at a specific height. Cutting lower creates more clusters with smaller differences, while cutting higher produces fewer clusters with larger differences.

```r
# Create both k=3 and k=4 solutions by cutting at different
hc3_clusters <- cutree(hc_ward, k = 3)
hc4_clusters <- cutree(hc_ward, k = 4)

# Add to dataset for analysis
maxdiff_hc3 <- maxdiff_chapter8_example
maxdiff_hc3$cluster <- as.factor(hc3_clusters)

maxdiff_hc4 <- maxdiff_chapter8_example
maxdiff_hc4$cluster <- as.factor(hc4_clusters)
```

The `cutree()` function automatically finds the height that produces exactly k clusters. By specifying both k=3 and k=4, I could compare how the same underlying structure looked when divided into different numbers of segments.

Comparing Hierarchical Results

The hierarchical approach yielded interesting differences from k-means, and the results helped explain why my earlier attempts had struggled:

Three Cluster Hierarchical Results:

- Average silhouette width: 0.495
- Cluster sizes: 120, 100, 180 customers
- Same basic segment structure as k-means k=3

Four Cluster Hierarchical Results:

- Average silhouette width: 0.508

- Cluster sizes: 120, 100, 120, 60 customers
- Different cluster composition than k-means k=4

The hierarchical four cluster solution showed one key advantage over my earlier k-means attempt. It maintained the same strong clusters (later identified as "Frequent Premium Seekers" and "Value-Conscious Users") in both three and four cluster solutions. When moving from three to four clusters, it cleanly split the large mainstream segment rather than creating artificial divisions among the well-defined groups.

This stability suggested that some customer segments were more "real" than others. The hierarchical method was detecting natural groupings that persisted regardless of how I divided the remaining customers.

Understanding Silhouette Scores

```r
# Hierarchical 3-cluster validation metrics
sil_hc3 <- silhouette(hc3_clusters, dist(scaled_data))
print(paste("Average silhouette width (3-cluster):", round(m

# Hierarchical 4-cluster validation metrics
sil_hc4 <- silhouette(hc4_clusters, dist(scaled_data))
print(paste("Average silhouette width (4-cluster):", round(m

# Detailed breakdown by cluster
aggregate(sil_hc4[, 3], by = list(cluster = sil_hc4[, 1]), F
```

Silhouette Scores

R OUTPUT

```
te("Average silhouette width (3-cluster):", round(mean(sil_hc3[, 3]
[1] "Average silhouette width (3-cluster): 0.495"
> print(paste("Average silhouette width (4-cluster):", round(mean(sil_hc4[, 3]
[1] "Average silhouette width (4-cluster): 0.508"
> # Detailed breakdown by cluster
> aggregate(sil_hc4[, 3], by = list(cluster = sil_hc4[, 1]), FUN = mean)
cluster         x
1       1 0.5331681
2       2 0.5527411
3       3 0.4837314
4       4 0.4305644
```

The results showed interesting patterns:

Three Cluster Solution (Average: 0.495):

- Cluster 1: 0.53 (good separation)
- Cluster 2: 0.55 (good separation)
- Cluster 3: 0.44 (weak separation)

Four Cluster Solution (Average: 0.508):

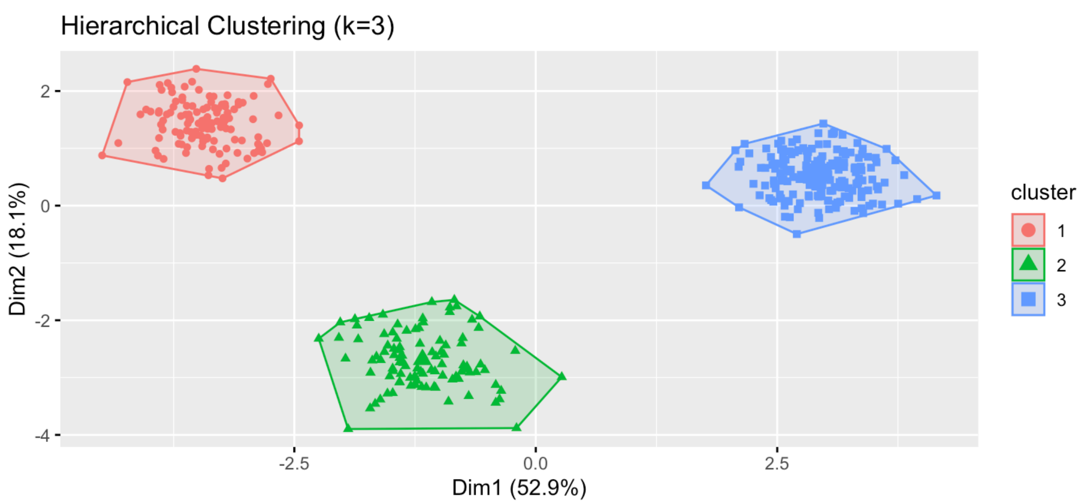
- Cluster 1: 0.53 (good separation)
- Cluster 2: 0.55 (good separation)
- Cluster 3: 0.48 (borderline separation)
- Cluster 4: 0.43 (weak separation)

The four cluster solution showed more balanced performance overall. While it still contained one weak cluster, the problematic large segment from the three cluster solution had been divided into two more manageable groups. This suggested that the hierarchical method was successfully identifying natural subdivisions within the heterogeneous mainstream segment rather than creating artificial splits among the well-defined clusters.

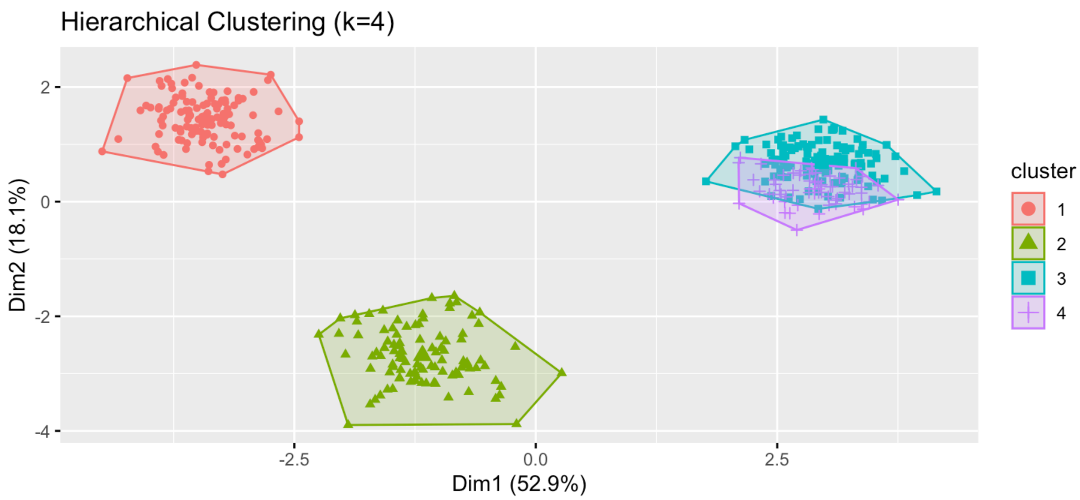
Visualizing Hierarchical Results

The PCA visualizations revealed notable differences from my k-means attempts:

```r
# Visualize 3-cluster solution
fviz_cluster(list(data = scaled_data, cluster = hc3_clusters
             geom = "point",
             palette = c("#FF6B6B", "#4ECDC4", "#45B7D1")) +
  ggtitle("Hierarchical Clustering (k=3)")

# Visualize 4-cluster solution
fviz_cluster(list(data = scaled_data, cluster = hc4_clusters
             geom = "point",
             palette = c("#FF6B6B", "#4ECDC4", "#45B7D1", "#
  ggtitle("Hierarchical Clustering (k=4)")
```



Hierarchical Clustering 3 Segment



Hierarchical Clustering 4 Segment

The `fviz_cluster()` function creates a two-dimensional representation of the multidimensional customer data using Principal Component Analysis (PCA). Think of this as creating a map where similar customers appear close together and different customers appear far apart. The ellipses around each cluster show the approximate boundaries, with larger ellipses indicating more internal variation within the segment.

The hierarchical four cluster visualization showed cleaner separation than my k-means attempt. While some overlap remained between adjacent clusters, the boundaries appeared more natural rather than artificially imposed. Critically, the clearly distinct cluster remained well-separated from all others, and the division of the mainstream segment looked meaningful rather than random.

Why Hierarchical Clustering Worked Better

Several factors explained why the hierarchical approach produced superior results for this coffee preference data. Unlike k-means, which assumes all clusters should be roughly spherical and similar in size, hierarchical clustering can detect clusters of different shapes and densities. My coffee preference data apparently contained some tight, well-defined groups and some looser, more dispersed groups that didn't fit k-means' geometric assumptions.

The dendrogram revealed which customer groupings were most stable across different solutions. The same core segments appeared in both three and four cluster solutions, suggesting these represented genuine customer types rather than statistical artifacts created by the algorithm. This stability provided confidence that the hierarchical method was identifying real patterns in customer preferences rather than imposing artificial structure.

Ward linkage specifically optimizes for creating clusters with minimal internal variation, which aligns well with segmentation goals. This method tends to produce segments where customers within each group are as similar as possible, making the resulting clusters more coherent and actionable for marketing purposes. The dendrogram also provided clear visual guidance about where natural divisions exist in the data, rather than forcing me to guess the optimal number of clusters based solely on statistical measures.

The convergence between hierarchical and k-means results strengthened my confidence in the underlying patterns. Both methods produced nearly identical silhouette scores of 0.508 and 0.509 respectively for four clusters, suggesting that genuine customer structure existed rather than method-specific artifacts. When different algorithms detect similar patterns, it indicates that the clustering reflects real customer groupings rather than algorithmic bias.

However, I wouldn't say I definitively prioritized hierarchical clustering over k-means based on this analysis alone. The statistical quality was virtually identical between the two approaches, with both four-cluster solutions producing silhouette scores around 0.508. The key advantage of the hierarchical method was its ability to reveal the data's natural structure, which led to a more interpretable and stable result. It cleanly subdivided the large mainstream segment along meaningful lines rather than creating artificial breaks. This highlights a crucial lesson in segmentation supported by Dolnicar et al. (2018): success is not merely a hunt for the highest statistical score, but a search for the most stable, understandable, and ultimately actionable customer groupings. [15] In this case, the hierarchical approach delivered a solution that was not just statistically sound, but also made more business sense.

The Missing Piece: Demographic Validation

Note that in real-world segmentation projects, researchers would conduct further validation beyond the statistical and preference-based analysis we've focused on here. A step involves examining how each clustering solution performs across demographic variables, behavioral data, and other customer characteristics available in the dataset.

For example, researchers would typically analyze whether the "Quality & Experience Seekers" segment shows higher income levels, different age distributions, or distinct geographic patterns compared to the "Quick & Budget-Conscious" group. They might examine whether convenience-focused customers are more likely to be working professionals with busy schedules, or whether quality seekers tend to live in urban areas with more coffee shop options.

This demographic profiling often reveals which statistical solution translates into the most meaningful and actionable business segments. Sometimes a clustering solution that appears statistically sound based on preference data alone falls apart when you discover that the resulting segments show no meaningful differences in age, income, lifestyle, or other relevant characteristics. Conversely, a solution with moderate statistical scores might prove highly valuable if it creates segments with distinct demographic profiles that align with existing customer data or market research insights.

We simplified this analysis by focusing primarily on preference-based clustering quality to illustrate the core methodological differences between approaches. However, the interplay between preference-based clusters and real-world customer characteristics frequently drives the final segmentation decision in ways that pure statistical measures cannot capture.

This exploration taught me that segmentation success often depends on finding the method that best matches the underlying structure of your specific data, rather than defaulting to the most commonly used approach. In this case, the hierarchical structure of coffee preferences made hierarchical clustering slightly more suitable, but the decision was based on interpretability and business actionability rather than purely statistical superiority.

From Segmentation to Action

At this point, you've identified distinct customer segments. Groups like the "Quick & Budget-Conscious," "Convenience-Focused," and "Quality & Experience Seekers" from our coffee shop example, each with different priority patterns. But a list of segments, no matter how statistically sound, doesn't tell you what to do next.

This is where many practitioners turn to strategic frameworks.

The next chapter examines Tony Ulwick's Jobs-to-Be-Done Growth Strategy Matrix. This framework attempts to translate segmentation insights into strategic choices (differentiated, disruptive, dominant, discrete, or sustaining) based on where customer segments fall on the importance-satisfaction landscape. The segments you've identified become inputs to this thinking. Are the "Quality & Experience Seekers" underserved? The matrix would suggest differentiated positioning. Are the "Quick & Budget-Conscious" overserved? That might point toward disruption.

That said, Chapter 10 isn't a sales pitch for the framework.

The matrix has real limitations, and understanding them matters as much as understanding the framework itself. Chapter 10 will walk through the five strategies, show how they connect to the opportunity landscape, and then explain why treating this as a complete strategy would be a mistake. The goal is to give you a useful tool while being honest about what it can and can't do.

Chapter 9 Summary

- **No Perfect Method for Segmentation:** There is no single best way to segment customers. The most effective method depends on your business goals and data. The ultimate goal of segmentation is to find actionable groups of customers by balancing the need for practical insights with the preservation of the underlying data's integrity and statistical validity.

- **Traditional vs. Modern Approaches:** The chapter contrasts the classic Outcome-Driven Innovation (ODI) method, which uses factor and cluster analysis on importance/satisfaction Likert scales, with newer methods that leverage MaxDiff data to avoid scale-related biases.
- **Key Segmentation Methods for MaxDiff Data:** Three primary statistical methods for segmenting MaxDiff utility scores are introduced:
    - **Latent Class Analysis (LCA):** A sophisticated, model-based approach that simultaneously finds hidden segments and estimates their preferences.
    - **K-Means Clustering:** A popular and intuitive algorithm that groups customers by minimizing the distance to cluster "centers." Its main challenge is that you must specify the number of clusters in advance.
    - **Hierarchical Clustering:** A method that builds a tree-like structure (dendrogram) to show how customers naturally group together, providing visual guidance on the optimal number of segments.
- **Determining the Number of Clusters is Challenging:** The practical analysis shows that statistical tools meant to find the optimal number of clusters (like the elbow method, silhouette method, and gap statistic) often provide conflicting recommendations, requiring researcher judgment.
- **Iterative Process Leads to Better Results:** The initial K-Means attempt with four clusters was statistically weak, with overlapping segments and poor quality scores. Revisiting the analysis and choosing three clusters resulted in a much cleaner, more interpretable, and statistically sounder solution, identifying three core segments: **Quick & Budget-Conscious**, **Convenience-Focused**, and **Quality & Experience Seekers**.

- **Even Good Solutions Have Limitations:** The three-cluster solution, while an improvement, created one large and internally diverse (heterogeneous) segment of "Quality & Experience Seekers," suggesting it might contain multiple distinct sub-groups.
- **Hierarchical Clustering Can Offer Deeper Insight:** By using hierarchical clustering, the analysis revealed the underlying structure of the data more clearly. It produced a superior four-cluster solution by cleanly splitting the large, heterogeneous segment identified in the K-Means analysis, resulting in a more balanced and stable segmentation.
- **Actionability Over Statistical Purity:** The key takeaway is that segmentation success is measured by the clarity, stability, and business utility of the resulting segments, not just by achieving the highest possible statistical score. The best method is the one that best matches the data's natural structure and produces groups that can be targeted effectively.

Recommended Readings

If you are interested in learning more about segmenting maxdiff data and other approaches to go about segmenting data in general. I strongly recommend reading through Chris Chapman's blog post titled, Individual Scores in Choice Models, Part 1: Data & Averages or checking out his book R For Marketing Research and Analytics (Use R!). [36, 41] For those dealing specifically with MaxDiff data, Chrzan and Orme (2019) explore the nuances of clustering utility scores versus using latent class methods directly. [16]

In his book, he goes through a bit more detail in different segmentation methods for general marketing research and analytics use cases. They are beyond the scope of my expertise and this online book I am writing but it's a great starting point.

Other resources include

- Market Segmentation Analysis: Understanding It, Doing It, and Making It Useful Book by Bettina Grün, Friedrich Leisch, and Sara Dolnicar

- Market Segmentation: How to Do It and How to Profit from It
- Market Segmentation

References

[13] Wedel, Michel, and Wagner A. Kamakura. *Market Segmentation: Conceptual and Methodological Foundations*. Kluwer Academic Publishers, 2000.

[14] Jain, Anil K. "Data Clustering: 50 Years Beyond K-Means." *Pattern Recognition Letters*, vol. 31, no. 8, 2010, pp. 651-666.

[15] Dolnicar, Sara, Bettina Grün, and Friedrich Leisch. *Market Segmentation Analysis: Understanding It, Doing It, and Making It Useful*. Springer, 2018.

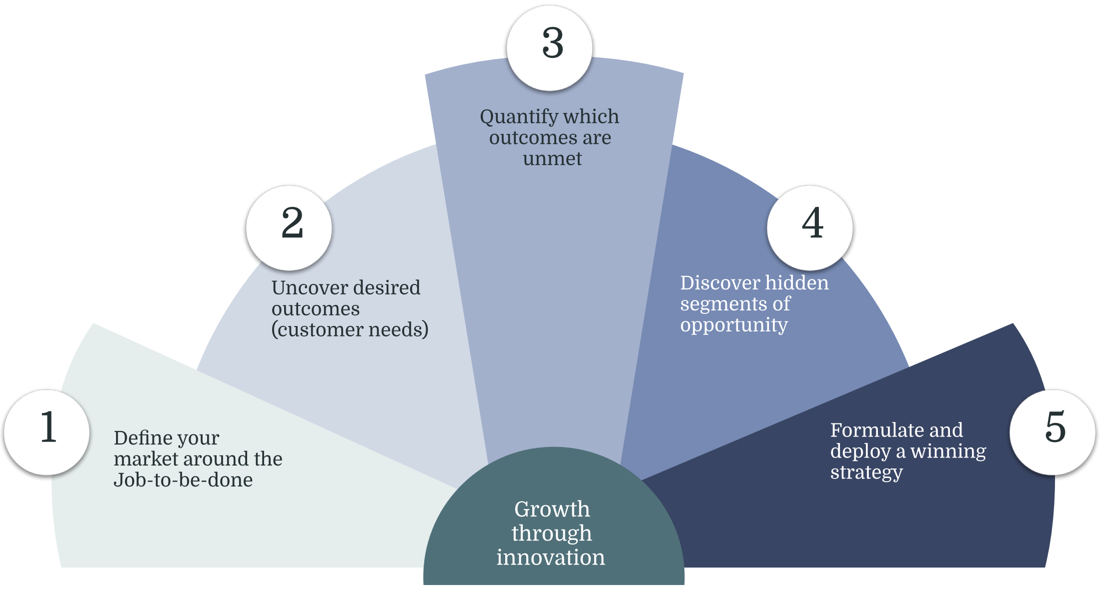
[16] Chrzan, Keith, and Bryan Orme. *Applied MaxDiff: A Practitioner's Guide to Best-Worst Scaling*. Sawtooth Software, 2019.

[36] Chapman, Chris. "Individual Scores in Choice Models, Part 1: Data & Averages." *QuantUXBlog.com*, 23 Oct. 2024. Available at: https://quantuxblog.com/individual-scores-in-choice-models-part-1-data-averages

[40] Ulwick, Tony. "Market Segmentation Through a Jobs-to-be-Done Lens." *Jobs-to-be-Done.com*, 18 Nov. 2017. Available at: https://jobs-to-be-done.com/market-segmentation-through-a-jobs-to-be-done-lens-5ef9242de65

[41] Chapman, Chris, and Elea McDonnell Feit. *R for Marketing Research and Analytics (Use R!)* 2nd ed., Springer, 2019. ISBN-13: 978-3030143152. Available on Amazon: https://www.amazon.com/Marketing-Research-Analytics-Use/dp/3030143155

Section 6 Overview



Formulate and deploy a winning strategy

Section 6 Overview: Formulate and Deploy a Winning Strategy

We have arrived at the final step of the process. You have defined the market, mapped the job, uncovered needs, quantified them with MaxDiff, and identified distinct segments. Now, the question remains: **What do we do with this data?**

In this final section, we will examine the traditional "Growth Strategy Matrix" often associated with Outcome-Driven Innovation. We will look at how it attempts to map underserved and overserved needs to specific business strategies.

However, we will also take a critical look at the limitations of "matrix-based" strategy. Real-world product strategy is messy. It involves regulatory environments, competitive pressure, organizational capabilities, and technological disruption, factors that a simple 2x2 grid cannot capture.

A Note on Scope: This section will not provide templates for product roadmaps or executive pitch decks. Those artifacts depend entirely on your specific organizational context. Instead, this chapter focuses on the *interface* between data and decision-making. We will explore how to use your MaxDiff segments not as a rigid instruction manual, but as a powerful input to **de-risk your strategic bets**.

Let's explore how to turn your data into evidence.

Chapter 10: Jobs-to-be-Done Growth "Strategy" Matrix

Introduction to Ulwick's Strategic Matrix

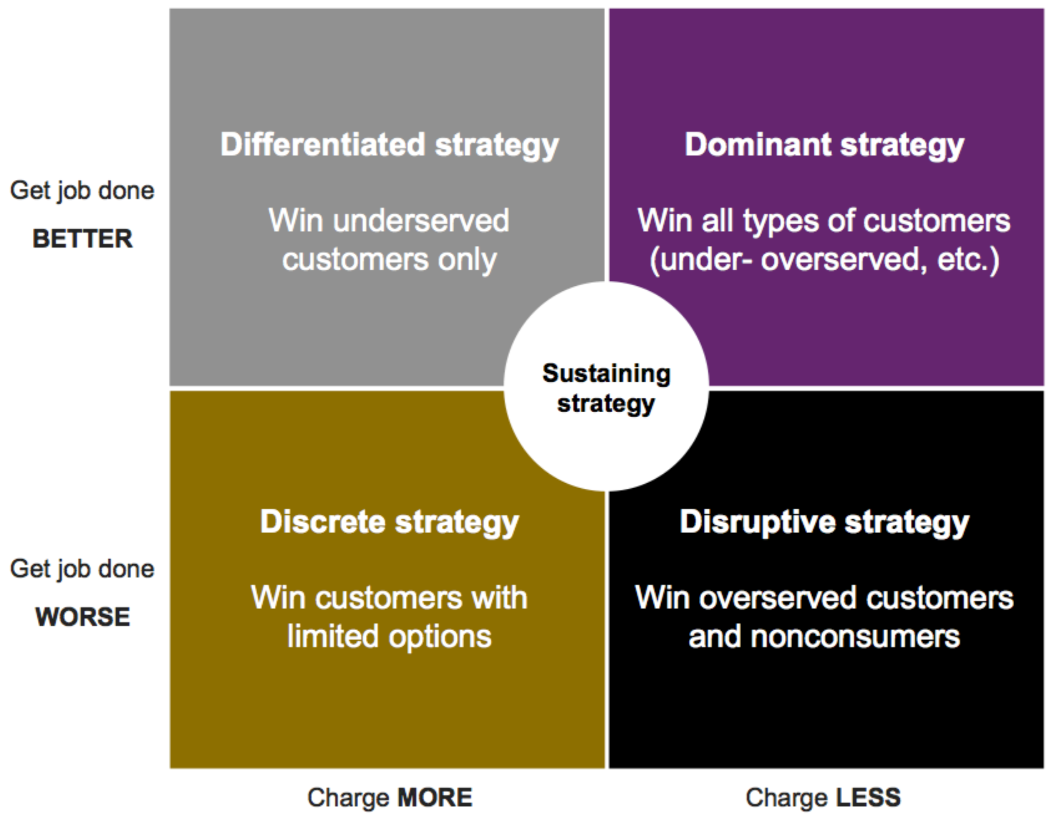
Tony Ulwick's Jobs-to-Be-Done Growth Strategy Matrix offers an approach to needs strategy. Built on 25 years of research across hundreds of markets with Fortune 500 companies, this framework addresses a challenge that most businesses face when planning their next move [42].

As Ulwick observes, "new products and services win in the marketplace if they help customers get a job done better and/or more cheaply" [42]. This explains why some products succeed while others fail, and more importantly, how to predict which strategy will work in a given market situation.

This chapter will go through the 5 different strategies Strategyn promotes based on their ODI approach.

> *Keep in mind that much of this perspective on strategy is based on the opportunity landscape and where outcomes/needs fall. For example if they are over served, underserved, appropriately served, etc. Given the critiques we have outlined in chapters 7 and 8, this chapter will focus on the theoretical approach of Strategyn's needs based strategy.*

The Five Growth Strategies Defined



Jobs-to-be-Done Growth Strategy Matrix [43]

Differentiated Strategy

A differentiated strategy targets underserved customers who have unmet needs and are willing to pay more for better performance. These customers value getting the job done better over cost considerations.

The strategy works when you can identify a segment that finds existing solutions inadequate. Tesla's early electric vehicles are a clear example. While traditional automakers focused on fuel efficiency improvements, Tesla recognized that environmentally conscious car buyers wanted zero emissions without sacrificing performance. They were willing to pay premium prices for a car that delivered superior acceleration, advanced technology, and environmental benefits.

Similarly, Peloton identified fitness enthusiasts who wanted the energy of group classes combined with the convenience of home workouts. Traditional home exercise equipment felt boring and isolated, while gym memberships required travel time and schedule coordination. Peloton's premium-priced solution delivered the best of both worlds for customers who valued this specific combination highly enough to justify the cost.

Dominant Strategy

The dominant strategy represents the ideal scenario: targeting all customers with a solution that performs much better while costing much less. This approach appeals to most segments because it delivers superior value across both dimensions.

Amazon Web Services changed business computing by offering better reliability, scalability, and functionality than traditional IT infrastructure while cutting costs. Companies no longer needed massive upfront investments in servers and data centers. They could access enterprise-grade computing resources on demand at a fraction of the traditional cost.

Spotify achieved similar dominance in music consumption. Compared to buying individual albums or songs, Spotify offered access to millions of tracks for less than the cost of a single CD per month. The service was more convenient, more comprehensive, and more affordable than existing alternatives, making it attractive to virtually every music listener.

Disruptive Strategy

A disruptive strategy targets overserved customers or nonconsumers with a solution that costs less but performs worse than existing alternatives.

Southwest Airlines built an entire business model around this concept. While major airlines competed on amenities, meal service, and seat comfort, Southwest recognized that many travelers simply wanted reliable, affordable transportation. They removed extras that many customers didn't value, focusing instead on frequent flights, low prices, and dependable service.

Zoom's rise during the early days of video conferencing follows a similar pattern. Enterprise video solutions from companies like Cisco offered extensive features and enterprise-grade security but required technical expertise and investment. Zoom provided adequate video quality with a far simpler setup, making video calls accessible to millions of users who had been priced out of the market.

Discrete Strategy

The discrete strategy operates in situations where customers have limited alternatives due to legal, physical, emotional, or other restrictions. These constrained environments allow companies to charge higher prices for inferior solutions.

Movie theater concessions represent the classic example. Customers pay premium prices for average-quality snacks because theaters prohibit outside food and beverages. The captive audience situation enables pricing that would be impossible in competitive environments.

Wedding vendors often employ discrete strategies as well. Couples planning weddings face emotional and social pressure to create perfect experiences, making them less price-sensitive and more willing to accept premium pricing for services that might cost less in other contexts. The unique, high-stakes nature of weddings creates emotional restrictions that vendors can leverage.

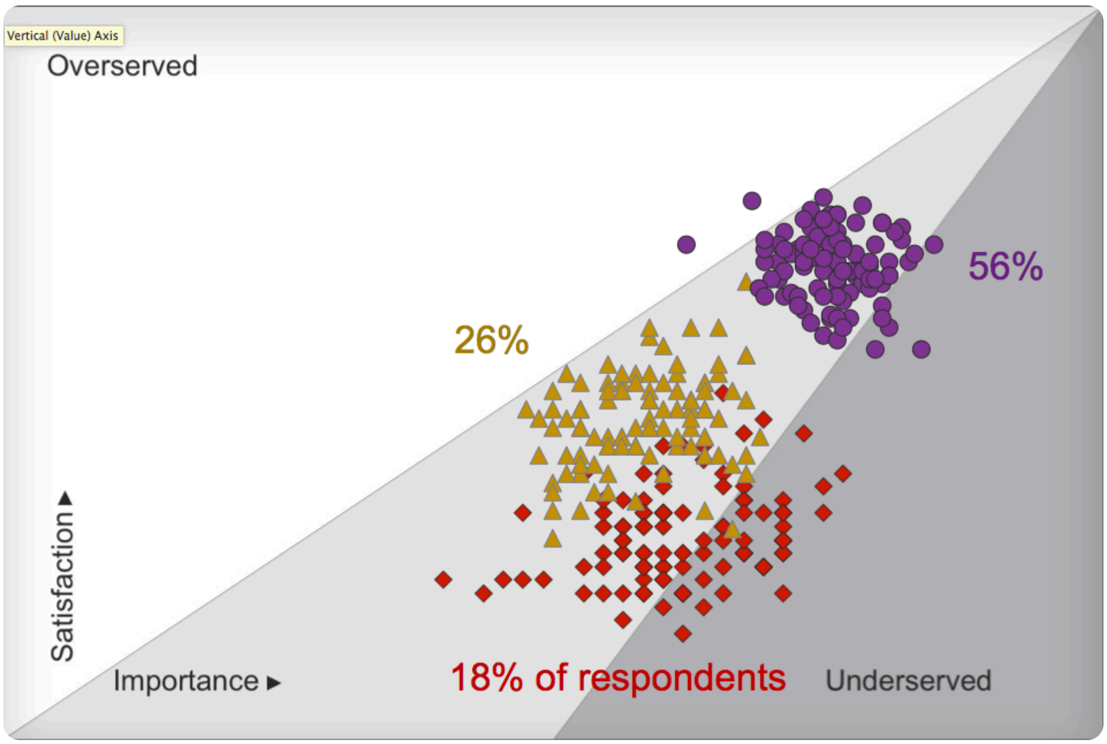
Sustaining Strategy

A sustaining strategy involves incremental improvements that make products slightly better or slightly cheaper. While these improvements may help retain existing customers, they rarely attract new ones or create competitive advantages.

Annual smartphone releases typically follow sustaining strategies. Each new model offers marginally better cameras, slightly faster processors, or modest design improvements. These changes help manufacturers maintain customer loyalty and justify regular upgrade cycles, but they rarely create breakthrough growth or attract customers from competing platforms.

Most software updates fall into this category as well. Adding new features, improving user interfaces, or enhancing performance helps retain existing customers but doesn't fundamentally change competitive dynamics or create new market opportunities.

Determining the right "strategy" based on the opportunity landscape



Jobs-to-be-Done Opportunity Landscape

Looking at this satisfaction-importance landscape, we can map where each strategic approach finds its optimal target customers.

Disruptive strategies work best in the bottom right quadrant, where customer needs are highly important but current satisfaction levels are low. This "underserved" territory represents the 18% of respondents who need better solutions but aren't getting them from existing offerings. These customers become prime targets for disruptive innovations that may sacrifice some advanced features in exchange for much better accessibility, affordability, or simplicity in addressing their core unmet needs.

Sustaining strategies operate most effectively in the middle diagonal band, where satisfaction roughly aligns with importance. Here, customers are reasonably well-served by existing solutions, creating opportunities for incremental improvements that enhance performance or reduce costs without fundamentally changing the value proposition. The 26% of customers in this zone typically respond to incremental rather than breakthrough changes.

Overserved customers cluster in the top left quadrant, where satisfaction exceeds importance. This represents the 56% of respondents who are getting more than they actually need or value from current solutions. These customers are open to disruption from competitors who can strip away excess features and complexity while focusing on what truly matters to them at a lower price point.

Dominant strategies Dominant strategies sit outside this framework entirely. When a company achieves dominance, they effectively shift the entire satisfaction axis upward while reducing costs, making their solution attractive to customers across all quadrants simultaneously. Amazon Web Services and Spotify succeeded by delivering superior performance at lower costs, appealing to underserved customers who needed better solutions, overserved customers who wanted simpler and cheaper alternatives, and everyone in between.

Differentiated strategies target specific pockets where importance is high but satisfaction varies widely, allowing companies to command premium prices by delivering superior performance on the dimensions that matter most to these particular customer segments. Meanwhile, **discrete strategies** can operate across various quadrants when external constraints limit customer choice, regardless of the satisfaction-importance relationship.

Limitations and Critiques of the Matrix Approach

While the Jobs-to-Be-Done Growth Strategy Matrix provides a useful starting framework, it oversimplifies the reality of strategic decision-making. Real strategy involves far more complex factors than customer satisfaction and performance trade-offs.

Markets don't operate in neat categories. Customer segments overlap, needs evolve constantly, and competitive dynamics shift unpredictably. Apple simultaneously pursues different strategies across market segments and geographies, while companies like Tesla have evolved from differentiated to dominant positioning as market conditions changed. The framework provides snapshots but misses the dynamic, multi-faceted nature of actual markets.

The AI disruption paradox reveals limitations. AI tools are disrupting industries where customers have highly underserved and potentially over served needs. ChatGPT doesn't write better than professional copywriters, yet it succeeds with disruptive positioning by offering "good enough" solutions at near-zero cost. This happens because AI represents capability substitution rather than enhancement, replacing entire approaches rather than improving existing ones.

Strategy is complex. Successful strategies must account for regulatory environments, organizational capabilities, competitive responses, technology evolution, capital requirements, network effects, and countless other variables. Tesla's success wasn't just about identifying underserved customers - it required breakthrough battery technology, vertical integration, charging infrastructure development, and regulatory navigation.

The framework offers a helpful lens for thinking about customer needs, but treating it as the only view for a "strategy" would be quite concerning. Strategy requires synthesizing complex, interdependent factors that extend far beyond any single matrix can capture.

A final note on Strategy

ODI is an input, **not a strategy**. In practice, researchers, product managers, and strategists use JTBD and ODI to inform product decisions or de-risk assumptions about customer needs. The framework helps answer questions like "Are customers satisfied with current solutions?" or "What performance gaps exist?" But strategy requires synthesizing these insights with competitive analysis, organizational capabilities, market timing, regulatory environments, capital requirements, and countless other factors.

The real value is in risk reduction. ODI helps teams avoid building products nobody wants by identifying genuine customer needs and satisfaction gaps.

Strategy is fundamentally more complex. Calling customer satisfaction analysis a "strategy" is like calling market research a business plan. Both provide essential inputs, but neither constitutes the full picture.

The framework offers customer insights that should inform strategic thinking, but practitioners should resist the temptation to treat customer satisfaction mapping as strategy itself. Its strength lies in helping teams ask better questions about customer needs, not in providing comprehensive strategic direction.

Chapter Conclusion – The Value of the Matrix

Communicating the Strategic Story: This brings us back to what I said in Chapter 4. We established that JTBD excels at answering high-level questions like "What markets should we enter?" or "How do we redefine our competitive landscape?"

If the Growth Strategy Matrix has limitations, does that negate the strategic value of the methodology?

Not at all. The conflict is not in the data but in how we define "strategy."

Real strategy is a blend of *desirability* (what customers want), *feasibility* (what we can build), and *viability* (what the business can sustain). The critique in this chapter highlights that the Matrix focuses almost exclusively on **desirability**. It assumes that if you identify the right customer need, the business model will follow. In the messy real world, that is not always true.

However, this does not diminish its value to senior leadership. Directors and executives value research teams conduct because it provides the **strategic story**. It anchors complex portfolio decisions in customer feedback rather than internal opinion. It allows a leader to say, "We are pivoting to a Differentiated Strategy because the data proves the market is underserved."

Looking Ahead: From Strategy to Execution

Identifying where customers are underserved is only useful if teams can act on it. The output of JTBD research is typically a spreadsheet of prioritized needs or a matrix showing market positions. But product teams need user stories, design briefs, and backlog priorities. This is the Execution Gap.

The next chapter addresses this gap through three principles:

Context shows you how to overlay your needs-based insights onto your organization's existing personas and frameworks rather than replacing them.

Verification demonstrates how to combine your quantitative priorities with behavioral data and qualitative context to build cases that withstand scrutiny.

Execution provides the tactical translation. It converts abstract needs into user stories, success metrics, and definitions of done that teams can build against.

Together, these principles transform ODI from a strategic input into executable product decisions.

Chapter 10 Summary

- **Core Principle:** The chapter introduces Tony Ulwick's Jobs-to-Be-Done Growth Strategy Matrix, which posits that products succeed when they help customers get a job done **better** and/or **more cheaply**.
- **Five Growth Strategies:** The framework outlines five distinct strategies based on how a product performs relative to existing solutions and what it costs:
    - **Differentiated Strategy:** Targets **underserved** customers by offering a **better** solution at a **higher** price (e.g., Tesla's early models, Peloton).
    - **Dominant Strategy:** The ideal scenario, targeting **all customers** with a solution that is both **better** and **cheaper** (e.g., Amazon Web Services, Spotify).

    - **Disruptive Strategy:** Targets **overserved** customers or **non-consumers** with a solution that performs **worse** but is **cheaper** (e.g., Southwest Airlines, Zoom).
    - **Discrete Strategy:** Targets customers in **constrained environments** (legal, physical, etc.) by charging a **higher** price for an **inferior** solution (e.g., movie theater snacks).
    - **Sustaining Strategy:** Involves making **incremental improvements** (slightly better or cheaper) to retain existing customers but rarely creates growth (e.g., annual smartphone updates).
- **The Opportunity Landscape:** This tool maps customer needs by **importance** versus **satisfaction** to identify strategic opportunities:
    - **Underserved** customers (high importance, low satisfaction) are prime targets for **differentiated** strategies.
    - **Overserved** customers (low importance, high satisfaction) are vulnerable to **disruptive** strategies.
    - **Appropriately served** customers are targets for **sustaining** strategies.
- **Limitations and Critiques:** The matrix is a useful starting point but has limitations:
    - It **oversimplifies** real-world markets, which are dynamic and complex, not neat categories.
    - It fails to adequately explain modern disruptions like **AI**, which substitute capabilities rather than just competing on performance and cost.
    - Real strategy must account for numerous other factors beyond this matrix, including regulations, technology, organizational capabilities, and competitive responses.

Chapter References

[42] Ulwick, Tony. "The Jobs-to-be-Done Growth Strategy Matrix." *Jobs-to-be-Done.com*, 6 Jan. 2017. Available at: https://jobs-to-be-done.com/the-jobs-to-be-done-growth-strategy-matrix-426e3d5ff86e

[43] Ulwick, Tony. "The Job-to-Be-Done Growth Strategy Framework." *Strategyn.com.* Available at: https://strategyn.com/5-business-growth-strategies-jobs-to-be-done/

FORMULATE AND DEPLOY A WINNING STRATEGY

Chapter 11: Translating Strategy into Execution

In the previous chapter, we looked at the limitations of the Growth Strategy Matrix. We established that while ODI offers one approach to identifying and quantifying customer needs, it is just one input. A matrix on a slide might tell you which market segment is underserved, but it does not tell you how to navigate technical debt, internal politics, or the conflicting priorities of a product roadmap.

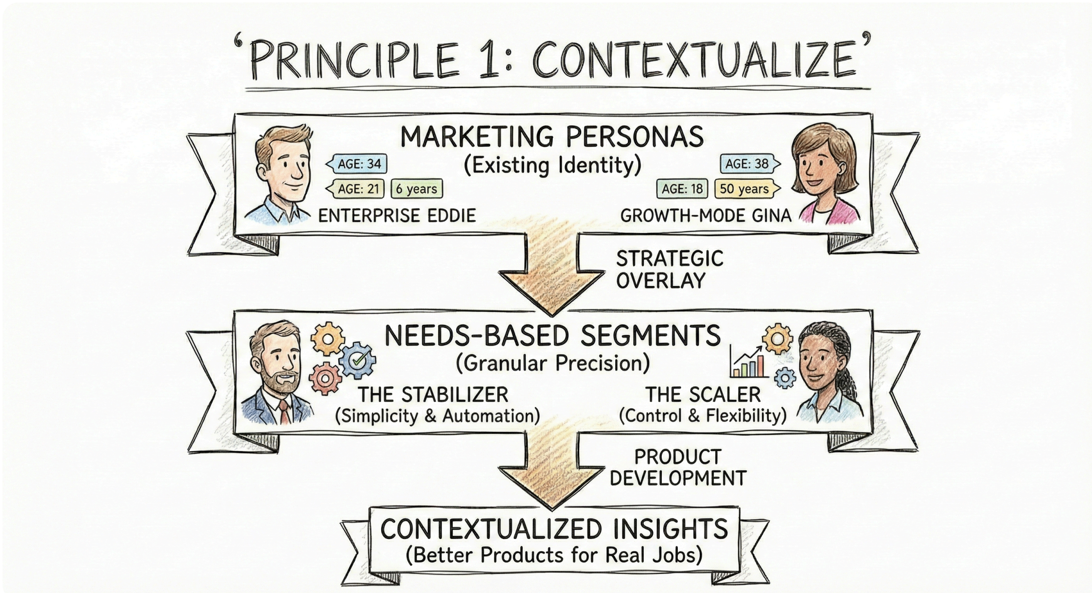
This brings us to a common challenge in the innovation process.

The problem is rarely having enough data. The problem is the artifact gap. The output of a JTBD research project is usually a complex spreadsheet or a dense report. However, the input required by a product team is a backlog of user stories, a set of technical constraints, a design brief, and alignment with senior stakeholders.

If you simply hand the spreadsheet to a product manager or technical team, it will likely be ignored. It is not in a format they can use. To make the data useful, you have to translate the abstract needs/outcomes into the tactical artifacts that drive daily work.

We do this by applying three core principles: Contextualization, Triangulation, and Operationalization.

Principle 1: Contextualize



Principle One: Contextualize

The first barrier to execution is usually the organization's existing mental models. Most companies have already invested in customer segmentation. They have personas, verticals, buyer types, or account tiers. These frameworks are not necessarily wrong or poorly constructed. Many organizations have thoughtful, well-researched personas that serve their intended purpose effectively.

The problem is not quality. The problem is that these artifacts have become part of the organization's shared language. Business leaders reference them in meetings. Sales teams use them to structure their territories. Marketing builds campaigns around them. When people talk about "the customer," they are often picturing a specific persona they have internalized over years of use.

When you finish a full JTBD research project, you will have a set of needs-based segments. A common error is to walk into a meeting and present these as the new, correct way to understand your customers, implying that the existing frameworks should be replaced. This creates immediate resistance.

The resistance is not usually about the validity of your research. It is about introducing new terminology into a system that already has a working vocabulary. You are asking people to unlearn how they talk about customers and adopt a different framework. Even if your segments are more precise, the organizational cost of this switch can be prohibitive. Instead, apply the Principle of Contextualization. Do not replace their existing frameworks. Overlay your insights onto them.

Mapping Segments to Personas

The key insight is that different frameworks serve different purposes. Demographics and personas help teams find and communicate with customers. Needs-based segments help teams build products that serve those customers well. These are complementary lenses, not competing truths. Your job is to map your new insights onto the existing vocabulary, adding resolution where it is needed without discarding what already works.

Consider a team that has a "Small Business Owner" persona. They treat this group as a single segment. But your research reveals that half this group wants advanced features and customization while the other half wants simplicity and guidance. The product team is currently building a compromise product that is too complex for one group and too simple for the other. Rather than arguing about personas, you explain that "Small Business Owner" remains the correct marketing bucket. But inside the product, there are two distinct modes. You introduce "The Stabilizer" (who wants automation and simplicity) and "The Scaler" (who wants control and flexibility). The marketing team keeps their targeting. The product team creates a "Simple Mode" and an "Advanced Mode" within the software. Both frameworks coexist because they serve different purposes.

The overlay can also work in the opposite direction, revealing similarities beneath apparent differences. Consider a sales team that treats "Healthcare Administrators" and "Financial Auditors" as completely different verticals. They have different sales decks and different feature requests. Engineering is being asked to build two separate reporting tools, which splits resources.

Your research shows that both groups share the exact same underlying needs around risk management. They both need audit trails, permissioning, and rollback capabilities. You validate a platform strategy. You build one Compliance Engine that serves both verticals, with only the front-end terminology changing. The sales team keeps their vertical positioning. Engineering builds one solution instead of two.

In both cases, you have not asked anyone to abandon their existing framework. You have added a layer of insight that makes their framework more effective.

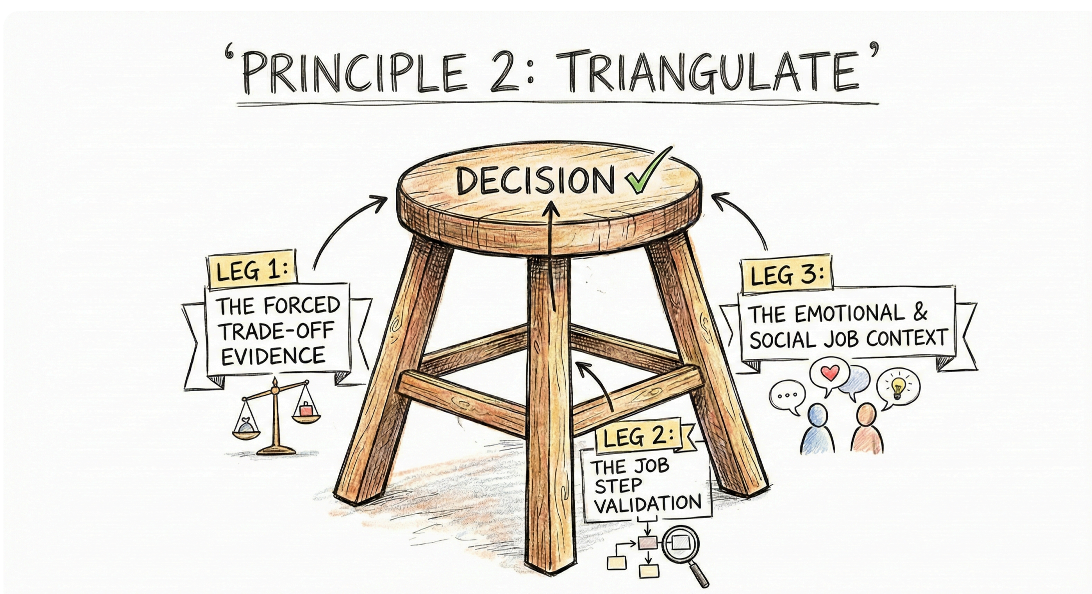
A Practical Tip for Data Analysis

As we discussed in Chapter 9, there are many ways to slice segmentation data. If the statistical validity holds up and the cluster analysis allows for it, try to align the number of your needs-based segments with the organization's existing structure.

For example, if your sales team is already organized into four vertical industries, and your data shows four distinct clusters of needs that roughly align with those verticals, use that four-cluster solution. It lowers the friction of adoption.

However, do not force this if the math does not work. Needs-based segments are solution-agnostic. They cover the entire market, not just the people currently buying your product. Because of this, it is common to uncover more complexity than the organization currently recognizes. You might find six distinct needs clusters even if the marketing team only uses three personas. Do not oversimplify the data to make it fit, but do seek alignment where the statistics allow.

Principle 2: Triangulate



Principle Two: Triangulate

Stakeholders sometimes naturally skeptical. If you rely on a single chart to justify a roadmap change, especially one that contradicts leadership's intuition, you are likely to hit resistance. Quantitative data alone feels abstract. It tells you what is winning without explaining why it matters to real people trying to get real jobs done.

To turn a data point into a decision, you need to triangulate. This means cross-referencing your findings with other evidence sources to build a reliable case. Think of it as constructing a three-legged stool where each leg draws from a different aspect of your research.

Leg 1: The Forced Trade-Off Evidence

Start with your MaxDiff data. This is your strongest quantitative foundation because it reflects genuine prioritization, not inflated ratings.

Frame it explicitly as a trade-off finding. You might explain to stakeholders that you surveyed a large sample of users and forced them to make hard choices between competing needs. You did not ask them to rate everything highly. You made them choose. When forced to decide, customers consistently prioritized certain needs over others. This was not a mild preference. It was a clear hierarchy.

This framing matters because it addresses the "everyone says everything is important" objection. You have evidence of what customers chose when they could not have everything.

Leg 2: The Job Step Validation

Next, validate that the stated preference matches actual behavior during job execution. This is where you return to the job map you created during qualitative research.

If a particular need ranked highly, look at what happens during that job step. Pull product analytics for the relevant workflows. Check usage patterns, error rates, and time-on-task data. Review support tickets tagged to that area.

You can also validate against the job steps you mapped in qualitative research. If your interview participants described a struggle during a particular step, and your quantitative data shows that same step contains the highest-ranked unmet needs, you have convergent evidence. The qualitative told you where the pain was. The quantitative told you how widespread it is. The behavioral data confirms it is real.

Leg 3: The Emotional and Social Job Context

Finally, provide the human context that makes the numbers meaningful. This is where you draw from the emotional and social jobs you uncovered during discovery.

Remember that functional jobs rarely exist in isolation. In Chapter 2, we discussed how functional jobs are accompanied by emotional jobs (how customers want to feel) and social jobs (how customers want to be perceived). Your triangulation should reconnect the prioritized functional needs to these emotional and social dimensions.

Pull quotes from your qualitative interviews that reveal the emotional stakes. A user describing anxiety, frustration, or fear is not just reporting a functional problem. They are revealing the emotional weight behind the numbers. This context explains why certain needs rank so highly and shapes how you should address them.

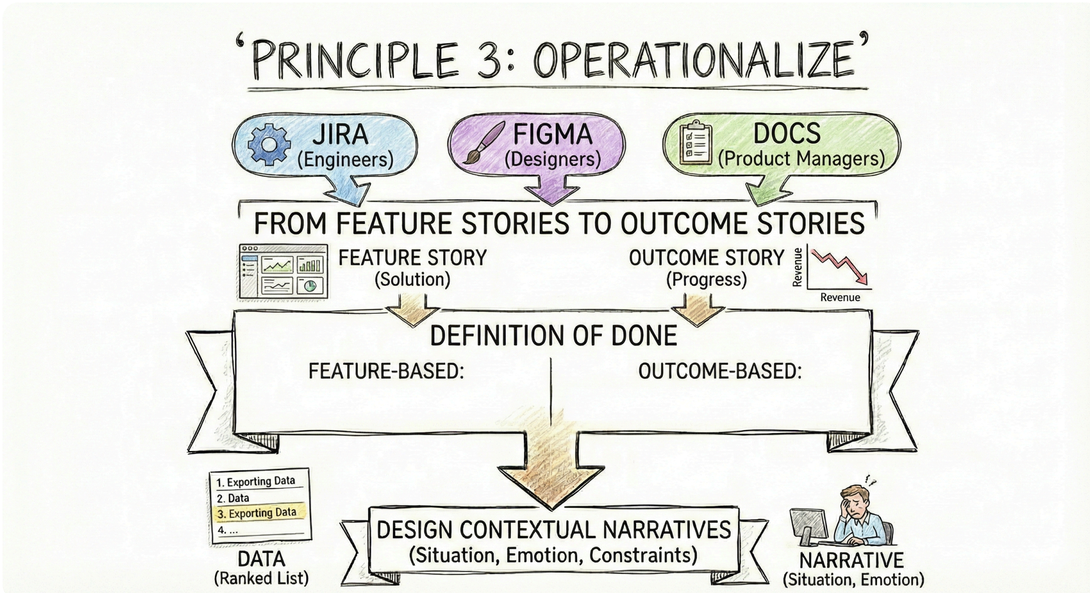
You can also layer in market context. If competitors are launching campaigns around the same themes, or if industry analysts are highlighting similar priorities, it validates that your findings reflect a broader market shift rather than just your sample's idiosyncrasies.

The Combined Effect

When you present all three legs together, you transform the conversation. You are no longer saying "I think we should prioritize this." You are presenting a factual case: a large sample ranked it first when forced to choose, product logs confirm the behavior is real, qualitative interviews reveal the emotional stakes, and the competitive landscape is moving in the same direction. The decision becomes easier to support.

This triangulation approach also protects you from the limitations of any single method. If your quantitative methodology has weaknesses, the behavioral and qualitative evidence provides a check. If your qualitative sample was small, the quantitative rankings provide scale. Each leg compensates for the others' blind spots.

Principle 3: Operationalize



Principle Three: Operationalize

The final principle is tactical. A major reason many initiatives fail is that they live in slide decks while the actual work happens elsewhere. Engineers live in Jira. Designers live in Figma. Product managers live in requirement documents. If your strategy does not translate into those tools, it will not get built.

You need to embed your strategy into the artifacts your teams already use. This means adapting your language to fit their existing workflows rather than forcing them to learn a new methodology.

From Feature Stories to Outcome Stories

The key translation is converting JTBD insights into user stories that engineering teams can act on. User stories are the standard format in most agile organizations, so learning to express JTBD findings in this format ensures your research actually influences what gets built.

User stories in agile development follow a standard format [44]:

> ***As a** [user type], **I want** [goal/desire] **so that** [benefit/outcome]*

The opportunity is in how you fill in this template. A feature-focused story puts a specific solution in the "I want" slot: "As a user, I want a search bar so that I can find content." This assumes a search bar is the right approach and frames success as the feature's existence.

An outcome-focused story puts the customer's desired situation in the "I want" slot and a measurable result in the "so that" slot. This leaves room for the team to determine the best solution while keeping everyone aligned on what success looks like.

A Five-Step Transformation Process

Here is how to translate a need statement into an outcome-focused user story.

Example 1: DevOps SRE Monitoring

Step 1: Start with the need statement. Using the syntax from Chapter 6, the underlying need might be: *Minimize the time it takes to determine whether a third-party API degradation is affecting customer transactions.*

Step 2: Identify the user and their context. The job executor is an SRE (Site Reliability Engineer). Their context is responding to potential incidents during on-call rotations.

Step 3: Reframe the need as a desired situation. What situation does the SRE want to be in? They want to *immediately understand impact*, not just *see data*. They want to *make confident decisions*, not just *monitor dashboards*.

Step 4: Connect to the business outcome. Why does this matter? Triggering unnecessary incident responses wastes team resources and creates alert fatigue. Missing real incidents harms customers and revenue.

Step 5: Write the outcome-focused user story.

> ***As an*** *SRE on call,* ***I want to*** *see a direct correlation between external API latency spikes and our checkout conversion rate* ***so that*** *I only trigger an incident response when revenue is actually at risk.*

Compare this to the feature-focused version: "As an SRE, I want a dashboard showing third-party API status so that I can monitor external dependencies." The feature-focused story prescribes a solution (a dashboard) and defines success as visibility (monitoring). The outcome-focused story describes the *insight* the SRE needs and defines success as *decision quality*—triggering responses only when appropriate.

Example 2: Professional Services Firm

Let's apply the same process to the Clarify example.

Step 1: Start with the need statement. *Minimize the time it takes to identify at-risk engagements before they become client-reported issues.*

Step 2: Identify the user and their context. The job executor is a Partner at a professional services firm. Their context is managing a portfolio of client engagements while handling business development responsibilities.

Step 3: Reframe the need as a desired situation. Partners do not want to *check dashboards*. They want to *be alerted proactively*. They want to intervene *before clients notice problems*, not after.

Step 4: Connect to the business outcome. Client-reported issues damage relationships, hurt referrals, and create reactive firefighting that consumes partner time.

Step 5: Write the outcome-focused user story.

> ***As a*** *partner,* ***I want to*** *be automatically alerted when an engagement shows early warning signs of trouble, before the client notices anything is wrong,* ***so that*** *I can intervene proactively rather than manage crises.*

Side-by-Side Comparison

Component	Feature-Focused Story	Outcome-Focused Story
User	As an SRE	As an SRE on call
Want	I want a dashboard showing third-party API status	I want to see a direct correlation between external API latency and checkout conversion
So that	so that I can monitor external dependencies	so that I only trigger incident response when revenue is actually at risk
Implied solution	Dashboard with API status indicators	Open—could be dashboard, alert system, or correlation engine
Success criteria	Dashboard exists and shows data	Incident responses correlate with actual revenue impact

Redefining "Done"

Outcome stories also change how you define success [46]. Instead of "the feature exists and works," the definition of done becomes a measurable outcome tied to the original need statement.

For the DevOps example, the definition of done might be: "The system alerts the on-call engineer specifically when a third-party error rate causes a greater than 5% drop in completed transactions."

For the Clarify example: "Partners receive alerts for at-risk engagements at least 48 hours before any client-reported issue, with a false positive rate below 20%."

These definitions are measurable, tied to business outcomes, and focused on the job rather than the feature. The team can now evaluate different technical approaches against these success criteria rather than debating implementation details in the abstract.

Working with Design Teams

For design teams, the challenge is slightly different. Designers need context about the user's environment, emotional state, and constraints during the moment of struggle. A ranked list of needs does not provide this.

The solution is to move beyond handing off data tables and instead provide contextual narratives. Use your qualitative research to paint a picture of the situation. When does this job step happen? What is the user's mental state? What has just happened before, and what needs to happen after? What are they afraid of?

For example, do not just tell the design team that "Exporting Data" is a high-priority unmet need. Use your interview notes to explain the context. Users typically perform this action at the end of a long work session when they are tired and anxious to finish. They have spent hours on the document and are terrified of losing their work. The export often happens right before a deadline.

This narrative changes how the team designs the solution. If they know the user is fatigued and time-pressured, they will not bury the export function inside multiple menu layers. They will make it prominent and foolproof. The context shapes the solution in ways the ranked data alone cannot.

Putting It Together: A Worked Example

Let me walk through how these principles combine in practice. This example uses a fictional company but draws on patterns common across real engagements.

The Company and Their Challenge

Clarify is a B2B SaaS platform that helps professional services firms manage client engagements. Their core product handles project tracking, time capture, document management, and client communication. Their customers include accounting firms, consultants, and financial advisors.

The product team was stuck in a familiar pattern. Every quarter, the roadmap discussion devolved into competing priorities. The sales team pushed for deeper integrations with accounting software because prospects kept asking about it. Customer success advocated for better onboarding flows because new users struggled in their first thirty days. Engineering wanted to rebuild the notification system because the current architecture created technical debt. The CEO had just returned from a conference convinced that AI-powered insights were the future of the industry.

Everyone had examples. The sales team could point to three lost deals where integration gaps were cited. Customer success had churn data showing first-month drop-off. Engineering had incident reports tied to notification failures. The CEO had competitor announcements to reference.

What nobody had was a systematic understanding of what customers actually prioritized when forced to choose. The team decided to run a JTBD study to break through the stalled progress.

Defining the Scope

Before designing the MaxDiff study, the team needed to define what they were researching. Following the principles from earlier chapters, they started by articulating the core functional job.

After reviewing support tickets, sales call recordings, and conducting eight preliminary interviews, they landed on: "Manage client engagements from initiation through completion while maintaining profitability and client satisfaction."

This job was broad enough to capture the full scope of what customers hired Clarify to do, but specific enough to exclude adjacent jobs like "win new clients" or "manage internal firm operations" that were not central to the product's value proposition.

They then mapped the key job steps: scope the engagement and set expectations, assign team members and allocate resources, track progress against milestones and budget, capture time and expenses accurately, communicate status to clients and internal stakeholders, identify and resolve issues before they escalate, complete deliverables and hand off to the client, and invoice and close the engagement.

From qualitative interviews and internal data review, they generated 43 initial need statements across these job steps. After consolidating duplicates and removing statements that were too solution-specific, they narrowed the list to 20 needs for the MaxDiff study.

The 20 Needs Tested

Here are the need statements they included:

1. Quickly identify scope changes before they impact profitability
2. Ensure all team members understand their responsibilities from day one
3. Minimize time spent figuring out who is available for new assignments
4. Avoid assigning team members who lack required expertise
5. Know immediately when a project falls behind schedule
6. See accurate profitability status without manual calculations
7. Identify which tasks are blocking overall progress
8. Reduce time spent chasing team members for missing time entries

9. Catch billing errors before they reach the client
10. Update clients on status without manually compiling reports
11. Ensure internal stakeholders see issues before clients raise them
12. Reduce time spent in status meetings
13. Identify at-risk engagements before they become emergencies
14. Quickly determine the root cause when something goes wrong
15. Ensure nothing falls through the cracks during final delivery
16. Avoid last-minute scrambles to locate documents for the client
17. Generate accurate invoices without reconciliation delays
18. Quickly identify which completed work has not been billed
19. Access engagement information from anywhere without VPN hassles
20. Trust that client data remains secure and compliant

Study Design

The team chose the combined framing approach for their MaxDiff question. Rather than asking about importance or satisfaction separately, they asked:

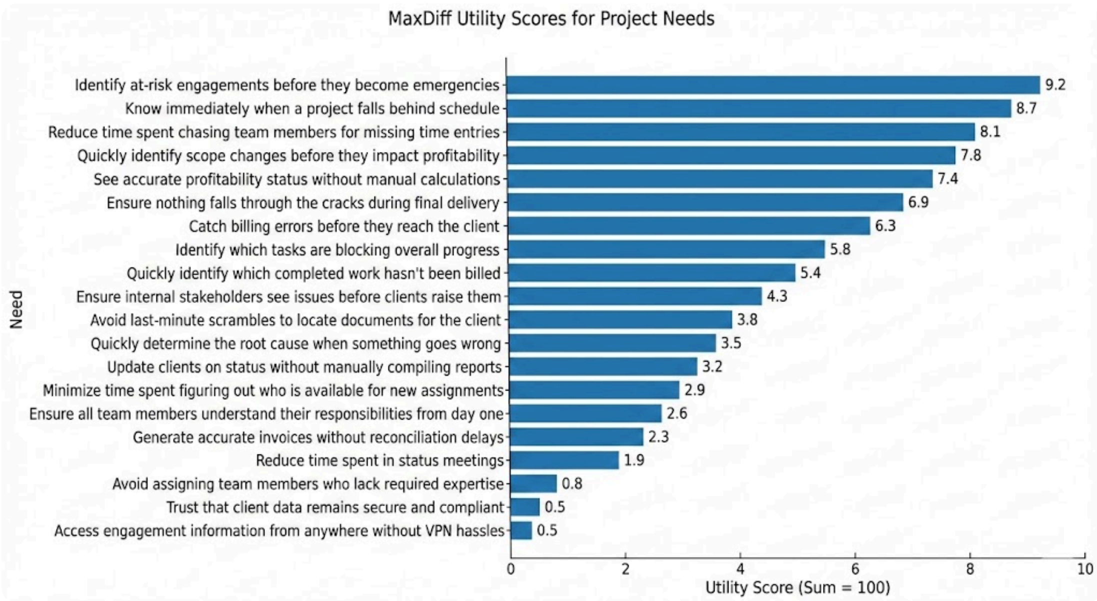
"When managing client engagements, which of these unmet needs would make the biggest difference to your firm if solved?"

This framing captured both dimensions in a single question. The need had to matter (importance) and not already be solved (satisfaction gap) to rank highly.

They configured the study with 5 items per choice set and 12 sets per respondent. With 20 total items, this design ensured each need appeared multiple times for each respondent and generated sufficient data for reliable estimation.

They recruited 340 respondents through their customer database, targeting engagement managers and partners at firms with 10 to 200 employees. They offered a $50 gift card incentive and achieved a 72% completion rate among those who started the survey.

The MaxDiff Results



Maxdiff Utility Scores for Project Needs

After running hierarchical Bayes estimation, they produced utility scores rescaled to sum to 100 across all items. Here is what they found:

Rank	Need	Utility Score
1	Identify at-risk engagements before they become emergencies	9.2
2	Know immediately when a project falls behind schedule	8.7
3	Reduce time spent chasing team members for missing time entries	8.1
4	Quickly identify scope changes before they impact profitability	7.8
5	See accurate profitability status without manual calculations	7.4
6	Ensure nothing falls through the cracks during final delivery	6.9
7	Catch billing errors before they reach the client	6.3
8	Identify which tasks are blocking overall progress	5.8
9	Quickly identify which completed work has not been billed	5.4
10	Ensure internal stakeholders see issues before clients raise them	4.9
11	Avoid last-minute scrambles to locate documents for the client	4.3
12	Quickly determine the root cause when something goes wrong	3.8
13	Update clients on status without manually compiling reports	3.5

Rank	Need	Utility Score
14	Minimize time spent figuring out who is available for new assignments	3.2
15	Ensure all team members understand their responsibilities from day one	2.9
16	Generate accurate invoices without reconciliation delays	2.6
17	Reduce time spent in status meetings	2.3
18	Avoid assigning team members who lack required expertise	1.9
19	Trust that client data remains secure and compliant	0.8
20	Access engagement information from anywhere without VPN hassles	0.5

Reading the Results: What the Hierarchy Reveals

The first insight was what rose to the top. The highest-ranked needs clustered around a single theme: early warning and visibility into problems. Customers wanted to know when engagements were going off track before the situation became critical. "Identify at-risk engagements," "know immediately when a project falls behind," and "identify scope changes before they impact profitability" all ranked in the top four.

The second insight was what did not rank highly. Notice where integrations landed. They did not appear in the top half. The sales team had been pushing for deeper accounting software integrations, but "access engagement information from anywhere" ranked last. "Generate accurate invoices without reconciliation delays" ranked sixteenth.

This did not mean integrations were worthless. It meant that when forced to choose, customers prioritized early warning systems over data connectivity. They would rather know about problems sooner than have smoother data flows.

The third insight was the natural break points. There was a meaningful gap between the top cluster (ranks 1 through 5, all above 7.4) and the middle tier (ranks 6 through 10, between 4.9 and 6.9). These natural breaks suggested where to draw priority lines.

Security and remote access ranked at the bottom. This initially surprised the team. Were those not table stakes? But remember the question framing: "unmet needs that would make the biggest difference if solved." Low scores here likely meant these needs were already adequately served, not that they were unimportant. This is the trade-off of combined framing. You cannot distinguish "unimportant" from "already satisfied" without additional data.

A Common Pitfall: Stopping at the MaxDiff Results

At this point, it would be tempting to simply take the top five needs and start writing user stories. The ranking seems clear. The data looks definitive. Why not execute?

This is the most common mistake teams make with quantitative needs research. MaxDiff tells you what customers chose when forced to prioritize. It does not tell you why they made those choices, whether their stated priorities match their actual behavior, or how these needs manifest differently across customer segments.

If Clarify had stopped here, they would have missed critical context that shaped how they ultimately addressed these needs. They needed to triangulate.

Triangulating with Behavioral Data

The team pulled product analytics to see whether the MaxDiff rankings aligned with actual user behavior.

The top-ranked need was "identify at-risk engagements before they become emergencies." Product logs showed that users who had access to Clarify's basic health scoring feature checked it an average of 4.2 times per week, more than any other dashboard view. But the same logs showed that 67% of users who checked the health score then navigated to three or more other screens, suggesting the score alone was not giving them what they needed. They were hunting for more information.

Support tickets corroborated this. The team tagged and reviewed six months of tickets and found that "engagement health" or "project status" appeared in 23% of all support conversations, the highest concentration for any topic.

The third-ranked need was "reduce time spent chasing team members for missing time entries." The team surveyed a subset of customers about their workflows and found that engagement managers spent an average of 3.2 hours per week on time entry follow-up. This was not just an annoyance. It was a measurable productivity drain.

Product logs showed that the "missing time entries" report was the second most frequently accessed report in the entire system. Users were already trying to solve this problem with existing tools. The tools were not solving it well enough.

Security and remote access ranked last in the MaxDiff. But were they unimportant, or already solved? Customer health scores showed that users who had experienced a security incident, even a minor one like a password reset issue, were 3.4 times more likely to churn within six months. Security was not unimportant. It was table stakes. The low MaxDiff ranking reflected satisfaction with current performance, not indifference to the need.

The team made a note: do not deprioritize security maintenance because it did not rank as an "unmet need." The combined framing revealed opportunities, not the full picture of what to protect.

Triangulating with Qualitative Context

Numbers told part of the story. But to translate needs into solutions, the team needed to understand the emotional and contextual dimensions.

They returned to their qualitative interview transcripts and pulled quotes that illuminated the top-ranked needs.

On identifying at-risk engagements, one partner at a mid-sized accounting firm had said: "I lie awake at night wondering which engagements are about to blow up. By the time I find out there is a problem, it is already a crisis. The client is upset, the team is stressed, and I am doing damage control instead of prevention."

Another engagement manager described it differently: "I know something is wrong when I start getting more emails from the client. But by then, the relationship is already strained. I wish I could see the warning signs before the client feels them."

These quotes revealed that the need was not just functional. It was deeply emotional. Any solution would need to address both the visibility gap and the anxiety it created.

On chasing time entries, a senior consultant explained: "I hate being the bad guy. Every week I am sending nagging emails to my team about time entries. It makes me feel like a babysitter, and it creates tension. They are professionals. They should not need reminders. But if I do not chase them, we cannot bill accurately."

This quote reframed the need. It was not just about efficiency. It was about role identity and team dynamics. A solution that simply automated the nagging might not solve the underlying problem. It might shift who was doing the nagging.

On scope changes, a partner at a consulting firm said: "Scope creep is how we lose money. The client asks for one more thing and my team says yes because they want to be helpful. By the time I find out, we have already done the work. I cannot bill for it without looking like I am nickel-and-diming, but I cannot eat the cost either."

This revealed that the scope visibility problem was not just about tracking. It was about the moment of decision. The partner needed to know about scope changes before the team committed, not after the work was done.

Triangulating with Market Context

Finally, the team looked outside their own data to validate that these priorities reflected broader market trends.

They found that two competitors had recently launched "engagement health" features with prominent marketing campaigns. Industry analysts had published reports highlighting "proactive risk management" as a top trend in professional services technology. A major accounting industry conference had added a track on "client relationship early warning systems."

This convergent evidence suggested Clarify was not just seeing patterns in their own customers. They were identifying a market-wide shift in priorities.

Contextualization: Mapping to Existing Segments

Clarify's marketing team had three existing personas. "Growth-Mode Gina" represented partners at firms actively expanding, focused on winning new clients and scaling operations. "Efficiency-Focused Eduardo" represented engagement managers at established firms, focused on profitability and utilization. "Compliance-Conscious Carla" represented firms in regulated industries with heavy documentation and audit requirements.

Rather than replacing these personas, the team analyzed the MaxDiff data by segment to see how priorities differed.

The finding was that the top five needs were consistent across all three personas. Every segment prioritized early warning and visibility into problems. The difference was in the why and the consequences.

For Growth-Mode Gina, an at-risk engagement meant reputational damage that could hurt new business development. She worried about word-of-mouth in her market.

For Efficiency-Focused Eduardo, an at-risk engagement meant margin erosion and utilization problems. He worried about the financial impact and resource allocation.

For Compliance-Conscious Carla, an at-risk engagement meant potential regulatory exposure and documentation gaps. She worried about audit trails and liability.

This insight shaped how the team would build and message the solution. The core functionality could be shared. But the specific signals that indicated "at risk," the dashboards that displayed status, and the messaging that promoted the feature could all be tailored to each persona's concerns.

What They Decided Not to Build

Just as significant was what the research told them to deprioritize.

The sales team's push for deeper integrations was tabled. The MaxDiff data showed integration-adjacent needs ranking in the bottom quartile. The triangulation did not surface any behavioral or qualitative evidence that contradicted this. The team acknowledged that integrations might matter for new customer acquisition, but for existing customers trying to do their jobs, it was not a priority. They decided to revisit integrations after addressing the top-tier needs.

The CEO's AI initiative was reframed rather than abandoned. None of the top-ranked needs explicitly called for AI. But the team realized that AI could potentially serve several of the top needs, such as predicting at-risk engagements, detecting scope creep patterns, or automating time entry reminders intelligently. Rather than building "AI features" as a category, they would evaluate AI as a potential solution approach for the needs customers prioritized.

The notification system rebuild that engineering wanted was approved, but reframed. Engineering had pitched it as technical debt reduction. The research revealed it was also a customer need. Several of the top-ranked needs required better notification infrastructure to solve. The rebuild was approved not as a maintenance project but as a foundation for the highest-priority customer outcomes.

Lessons from This Example

The MaxDiff ranking is the starting point, not the answer. The ranking told Clarify where to focus. It did not tell them how to solve the problems or what solutions would work. Triangulation with behavioral data and qualitative context was essential for translating priorities into effective solutions.

Combined framing reveals opportunities but hides table stakes. Security ranked last in the MaxDiff, but churn data showed it was critical. The low ranking reflected satisfaction, not indifference. Teams using combined framing need supplementary data to identify what they must protect, not just what they should build.

Contextualization beats replacement. Rather than telling marketing their personas were wrong, the team showed how the research added resolution to existing frameworks. The personas remained useful for targeting. The needs data made them useful for product development.

User stories should describe outcomes, not features. Every story the team wrote focused on the progress customers wanted to make, not the specific solution. This opened up solution possibilities and kept teams focused on the job rather than their first implementation idea.

Even good research requires iteration. The time entry solution underperformed despite being based on solid research. The research correctly identified the need. The first solution did not address it effectively. This is not a failure of the methodology. It is a reminder that research reduces risk rather than eliminating it.

Chapter 11 Conclusion

We began this book with a core value proposition, to give you a practical playbook for JTBD research that acknowledges both the benefits and the limitations of the methodology. This final chapter has focused on the gap that determines whether research influences decisions or gets ignored.

The Outcome Driven Innovation framework, as originally conceived, offers a systematic approach to understanding customer needs. But as we explored in Chapters 7 and 8, that system has real problems. Survey fatigue reduces data quality. The opportunity algorithm double-weights importance in ways that may not reflect actual priorities. The quantification creates false precision that can mislead strategic decisions.

These critiques do not mean you should abandon it to understand and quantify customer needs. They mean you should do it more carefully. The MaxDiff approach addresses the methodological weaknesses while preserving the core insight: customers have jobs they are trying to accomplish, and your product either helps them make progress or it does not.

But even rigorous research fails without translation. The three principles in this chapter are how you bridge the gap between insight and impact.

Contextualization means meeting your organization where it is. You will rarely have the luxury of replacing existing frameworks entirely. The skill is in overlaying new insights onto existing mental models in ways that add resolution without creating resistance.

Triangulation means building cases that withstand scrutiny. A single data source, no matter how rigorous, rarely survives contact with organizational politics. When three different data sources point the same direction, stakeholders stop arguing methodology and start debating what to do about it.

Operationalization means injecting your insights into the artifacts that drive work. Slide decks do not ship products. User stories, design briefs, and backlog priorities do. If your research does not translate into those formats, it will not influence what gets built.

Throughout this book, I have tried to be honest about what JTBD and ODI can and cannot do. This methodology will not guarantee product success. It will not tell you how to design the solution, how to price it, or how to market it. It will not account for competitive responses, regulatory changes, or shifts in technology.

What it will do, when applied thoughtfully, is reduce the risk that you build something nobody wants. It will give you a language for discussing customer needs that goes beyond feature requests and demographic assumptions. It will provide a framework for prioritization that is grounded in evidence rather than opinion.

That is not everything. But for teams stuck between conflicting stakeholder demands, unclear priorities, and the pressure to ship features that may not matter, it is substantial.

The research is one input. The strategy requires synthesis across many inputs. The execution requires translating strategy into the daily work of building products. If this book has helped you navigate any part of that journey with more confidence and rigor, it has accomplished its purpose.

Now go build something that helps customers make progress on the jobs that matter to them.

Chapter 11 References

[44] Cohn, M. (2004). *User Stories Applied: For Agile Software Development*. Addison-Wesley Professional. Retrieved from https://www.amazon.com/User-Stories-Applied-Software-Development/dp/0321205685

[45] Patton, J. (2014). *User Story Mapping: Discover the Whole Story, Build the Right Product*. O'Reilly Media. Retrieved from https://www.amazon.com/User-Story-Mapping-Discover-Product/dp/1491904909

[46] Schwaber, K., & Sutherland, J. (2020). *The Scrum Guide*. Scrum.org. Retrieved from https://scrumguides.org/

CONCLUSION

Final Thoughts

Throughout this book the main theme has been using a methodology in which it promises to help uncover unmet needs and provide alternative approaches to quantify, validate customer needs and alternative views at looking at customer problems to build better solutions.

However, in reality, many organizations already do a good job at exploring and validating customer needs. The biggest problem especially for large organizations is their inability to act on that knowledge and iterate.

Most companies have enough research. Actually, they have too much research. They have customer interviews, transcripts, quant results, analytics dashboards, all sitting in folders or different intranet websites. They have customer interviews sitting in forgotten folders. They have survey data that no one analyzed. They have support tickets that reveal the same pain points month after month. They have sales teams who know exactly why deals fall through. The information exists. It just never reaches the people who make decisions. Or it reaches them and gets ignored. Or it gets acknowledged and then deprioritized when the next quarter's targets loom.

This is the paradox. The better you get at uncovering customer needs or unique insights, the more clearly you see that the bottleneck was never the research. The bottleneck was actually the organizations willingness to change.

I have watched teams conduct rigorous foundational studies, identify clear opportunities, present compelling data, and then build the exact same roadmap they would have built anyway. The research became a box to check rather than a lens to see through. "Oh we did customer research with n=600 users", this is "validated". The methodology was followed. The insights were delivered. Nothing changed.

This is not a failure of JTBD. It is not a failure of ODI or MaxDiff or any particular framework. It is a failure of translation. Of timing. Of politics. Of incentive structures that reward shipping features over solving problems. Of cultures that treat customer research as a validation exercise rather than a discovery process.

If you take one thing from this book, let it be this. The frameworks are helpful but they are not enoguh. Learning to write a proper JTBD outcome statement is helpful. Understanding the flaws in the opportunity algorithm matters. Knowing how to triangulate quantitative rankings with behavioral data matters. But none of it matters if you cannot navigate the organizational reality that determines whether insights become products or slide decks.

Chapter 11 focused on contextualization, triangulation, and operationalization precisely because these are the skills that separate research that ships from research that sits. The methodology gets you to insight. The translation gets you to impact.

I wrote this book to demystify JTBD and provide practical alternatives to the rigid ODI approach. But I would be doing you a disservice if I let you believe that knowing all the methodology is the finish line. It is the starting line. The harder work comes after. It happens in the meetings where priorities get set. In the conversations where budgets get allocated. In the moments where someone with authority decides whether customer evidence outweighs internal opinion.

The best Researchers and Product Managers I know are not just skilled at research. They are skilled at reading organizational dynamics. They know when to push and when to wait. They know how to frame findings in language that resonates with different stakeholders. They know that being right is not the same as being effective.

This is the work that no methodology can teach you. It requires judgment, patience, and a willingness to play the long game. Sometimes the most important research skill is knowing that your findings will be ignored today but remembered six months from now when the product fails for exactly the reasons you predicted.

So yes, learn the frameworks and methodologies. Practice the techniques. Run the studies. But remember that the goal was never to become a better researcher. The goal was to help your organization build things that matter to customers. That requires more than methodology. It requires influence.

The research is one input. The strategy requires synthesis. The execution requires translation. And the impact requires an organization willing to act on what it learns.

This is the paradox. The better you understand your customers, the more you realize that understanding was never the hard part.

Ways to Support This Work

If this book has helped you or your team, here is a meaningful way to say thank you - buy me a coffee ☕

Your support is greatly appreciated!

RESOURCES

Chapter Exercise Solutions

Getting started with research methodologies for JTBD

Chapter 3 Answers

Exercise 1: Redefining the Market

This exercise focuses on shifting perspective from a product-defined market to a job-defined market.

1. Product: A high-end espresso machine for home use.

- **Traditional Market:** The home coffee and espresso appliance market.
- **Core Functional Job (JTBD):** "Feel energized and productive to start the day" or "Craft a premium coffee shop experience at home."
- **Unexpected Competitors:**
    - Energy drinks or dietary supplements (e.g., Celsius, Nootropics)
    - A subscription to a high-end local coffee shop
    - Sleep-tracking devices or apps (e.g., Oura Ring, WHOOP) that aim to increase natural energy levels

2. Product: A financial budgeting app (like Mint or YNAB).

- **Traditional Market:** The personal finance and budgeting software market.

- **Core Functional Job (JTBD):** "Gain financial peace of mind" or "Reduce anxiety about money."
- **Unexpected Competitors:**
    - A financial advisor
    - Mental health services or therapy
    - A higher-paying job or a side hustle
    - Automatic savings tools offered by a bank

3. Product: A project management software (like Asana or Trello).

- **Traditional Market:** The team productivity and project management software market.
- **Core Functional Job (JTBD):** "Ensure the team is aligned on key priorities" or "Provide clear visibility into project progress for stakeholders."
- **Unexpected Competitors:**
    - Daily in-person stand-up meetings
    - A shared physical whiteboard
    - A well-structured weekly email update
    - Shared documents in Google Docs or Notion

Exercise 2: Identifying Key Stakeholder Roles

This exercise tests the ability to distinguish between the Job Executor, Product Lifecycle Support Team, and Purchase Decision Maker.

1. Scenario: Law firm document management system.

- **Job Executor:** The **paralegals**.
    - *Justification:* They are the primary users who interact with the system daily to perform the core job of searching for, organizing, and sharing case files.
- **Product Lifecycle Support Team:** The firm's **IT department**.
    - *Justification:* They handle the consumption chain jobs—installation, managing permissions, and troubleshooting—but do not use the software for the core legal work.
- **Purchase Decision Maker:** The **managing partners** of the firm.
    - *Justification:* They hold the ultimate financial authority and sign off on the purchase, even though the Head of IT influences the decision.

2. Scenario: Family smart home security system.

- **Job Executor(s):** The **parents** and the **teenage child**.
    - *Justification:* All of them perform the core job of arming, disarming, and/or monitoring the system as part of their daily lives.
- **Product Lifecycle Support Team:** The **tech-savvy parent**.
    - *Justification:* This individual handles the installation and configuration updates, supporting the product's lifecycle.
- **Purchase Decision Maker: Both parents**.

    - *Justification:* The text states they are both involved in the final decision and control the family budget for this purchase.

3. Scenario: Online marketplace for freelance writers.

- **Job Executor(s):** The **freelance writers** AND the **businesses**.
    - *Justification:* This is a two-sided platform. Writers execute the job of "finding projects to earn income," and businesses execute the job of "sourcing talent for content creation."
- **Product Lifecycle Support Team:** The **marketplace's internal customer service and platform teams**.
    - *Justification:* They support both sides of the market by handling disputes, managing payments, and ensuring the platform runs smoothly.
- **Purchase Decision Maker:** The **freelance writers** and the **businesses**.
    - *Justification:* Both roles make a purchase decision. Businesses decide to pay for content, and writers often pay a platform fee or a percentage of their earnings to access the work.

Exercise 3: Strategic Prioritization

This exercise applies the concepts of the "chicken and egg problem" and the "hard side of the network" to a real-world scenario.

Scenario: The "SkillSwap" two-sided platform connecting learners and experts.

Question: Which group should you prioritize in your initial go-to-market and acquisition strategy: the learners or the experts? Why?

Answer: You should prioritize acquiring the **experts** (the teachers).

Justification:

Based on the principles in the chapter, the experts represent the "**hard side of the network**."

1. **They Create the Core Value:** The platform's value for learners is entirely dependent on the quality, quantity, and variety of available experts. Without compelling teachers, there is no reason for learners to join. This is analogous to Airbnb needing hosts before it could attract travelers.
2. **They Are Harder to Acquire:** It is much more difficult to convince skilled experts to invest their time to create profiles and offer teaching sessions than it is to find people who want to learn something new. The experts are the scarce resource.
3. **Solving the "Cold Start Problem":** To solve the "chicken and egg" or "cold start problem," you must first build the supply side of the network. By focusing on acquiring a critical mass of experts, you create the inventory that makes the platform attractive and functional for the "easy side" (the learners). Once a strong base of experts is established, attracting learners becomes a much easier task.

Chapter 5 Answers

Solutions

Solution to Exercise 1: Map a Personal Job

Let's use the example job, **"do the laundry."**

1. **Core Functional Job:** Get clothes clean for wearing.
2. **Job Map:**
    - **Define:**
        1. Determine if enough dirty laundry exists to warrant a load.

2. Assess the type of laundry to be done (e.g., colors, whites, delicates).

- **Locate:** 3. Gather dirty laundry from various locations. 4. Gather necessary cleaning supplies (detergent, fabric softener).
- **Prepare:** 5. Transport laundry to the washing machine. 6. Sort laundry into appropriate loads. 7. Set up the washing machine with the correct settings.
- **Confirm:** 8. Verify that no inappropriate items are in the load (e.g., pens, electronics).
- **Execute:** 9. Run the washing cycle. 10. Run the drying cycle.
- **Monitor:** 11. Check if the clothes are sufficiently dry.
- **Modify:** 12. Run an additional drying cycle if clothes are still damp.
- **Conclude:** 13. Fold and organize the clean laundry. 14. Put the clean laundry away in its proper place.

3. **Solution-Agnostic Check:** The steps above are solution-agnostic. They don't mention a specific brand of washing machine, a type of detergent, or a specific method for sorting. They apply whether you are using a modern machine, a laundromat, or washing by hand.

Solution to Exercise 2: Job Map vs. Customer Journey Map

1. **Customer Journey Map (Example: Buying an AMC movie ticket via the app):**

- **Action:** Hear about a new movie from a friend.
- **Touchpoint:** Open the AMC Theatres app on my phone.
- **Action:** Search for the movie title and select a showtime.
- **Action:** Choose my seats on the seat map.

- **Feeling:** Annoyed when a **$2.50 convenience fee** is added.
- **Touchpoint:** Pay using Apple Pay integrated into the app.
- **Feeling:** Relieved that checkout is fast.
- **Action:** Receive a QR code ticket in my email and in the app.
- **Touchpoint:** Get the QR code scanned at the theater entrance.

2. **Job Map (Core Job: "Secure access to an event"):**

    1. **Define:** Determine the type of event to attend.
    2. **Locate:** Identify available event options and showtimes.
    3. **Prepare:** Organize attendees and select specific seats/tickets.
    4. **Confirm:** Verify event details, time, and cost.
    5. **Execute:** Complete the transaction to acquire the ticket(s).
    6. **Monitor:** Confirm receipt of proof of access (the ticket).
    7. **Modify:** Make changes to the booking if necessary (e.g., cancel, rebook).
    8. **Conclude:** Store the ticket for easy retrieval at the event.

3. **Key Difference:** The Customer Journey Map documents my personal experience with a **specific solution (the AMC app)** and includes my emotions and solution-specific pain points (the fee), while the Job Map describes the **universal, functional process** anyone goes through to get the job done, regardless of the tool they use.

Solution to Exercise 3: Find the Right Granularity

- A. **Get educated: Too Broad.** This is the core job itself, not a step within it. It encompasses the entire process and isn't actionable as a single step.

- B. **Click the "play" button on a video lesson: Too Granular.** This is a micro-action. The more meaningful job step is "Consume educational content," which this action is a small part of.
- C. **Identify knowledge gaps: Just Right.** This is a distinct, critical thinking step at the beginning of the process. You can innovate here with assessment tools or skill-mapping software.
- D. **Evaluate potential learning resources: Just Right.** This is a key decision-making phase where a person compares options (e.g., courses, books, mentors). Innovation could focus on better review systems or recommendation engines.
- E. **Type a search query into a search engine: Too Granular.** This is a sub-task of a larger step, "Locate potential learning resources."
- F. **Apply the learned skill in a practical setting: Just Right.** This is a crucial step in the "Conclude" phase of learning, where the user validates that the job was done successfully.

Solution to Exercise 4: Competitive Analysis

1. **Job Map (Core Job: "Find a new residence to occupy"):**
    1. **Define:** Determine housing needs (budget, size, location, amenities).
    2. **Locate:** Find available properties that match the criteria.
    3. **Prepare:** Organize and schedule property viewings (virtual or in-person).
    4. **Confirm:** Evaluate the property and surrounding neighborhood to verify fit.
    5. **Execute:** Submit an application and necessary documentation.
    6. **Monitor:** Track the status of the application.
    7. **Modify:** Negotiate lease terms or offer details.

8. **Conclude:** Sign the lease agreement and secure the keys.

2. **Competitive Coverage:**

   - **Step 1 (Define):** Zillow is much better with its detailed filters. Craigslist is poor.
   - **Step 2 (Locate):** Zillow is better due to its map search and alerts. Craigslist is functional but basic.
   - **Step 3 (Prepare):** Zillow is better with integrated scheduling and 3D tours. Craigslist requires manual, offline coordination.
   - **Step 4 (Confirm):** Zillow is better with tools like Walk Score, school ratings, and price history. Craigslist is poor.
   - **Step 5 (Execute):** Both are poor. Zillow has some application features, but this step mostly happens offline.
   - **Step 6 (Monitor):** Both are very poor. There is no integrated way to track application status.
   - **Step 7 (Modify):** Both are very poor. Negotiation happens entirely offline.
   - **Step 8 (Conclude):** Both are very poor. This is almost always an offline, manual process.

3. **Innovation Opportunity:** There is a clear opportunity to create a solution that helps users with the final, high-stakes steps of the job: **Execute, Monitor, Modify, and Conclude.** A new product could focus on standardizing the application process, providing a secure platform for submitting documents, tracking application status in real-time, and facilitating digital lease negotiation and signing. This would address the most underserved and often most stressful part of the entire job.

Chapter 6 Answers

Of course. Based on the detailed interview transcript with Sarah, here is a comprehensive analysis of the different needs, jobs, and factors derived from her experience, structured according to the Jobs-to-be-Done and Outcome-Driven Innovation frameworks.

Desired Outcomes

These are the core, measurable metrics the customer uses to judge success when getting the job done. They are solution-agnostic and stable over time.

- **Derived Outcome:** Minimize the **time** it takes to transfer information from a proposal or timesheet to an invoice.

  > ***Quote:*** *"I'm always switching between three different applications: my spreadsheet, my proposal files, and then QuickBooks."*

- **Derived Outcome:** Minimize the **likelihood** of introducing errors when creating an invoice.

  > ***Quote:*** *"I'm always so paranoid about typos. I double, triple-check everything because I once sent an invoice to a big client and accidentally billed them for* $15,000 insteadof$ *1,500."*

- **Derived Outcome:** Minimize the **time** it takes to confirm a client has received and viewed an invoice.

> ***Quote:*** *"It's a black box. I'm never quite sure if the person I sent it to is the right person, if they got it, if it went to their spam folder..."*

- **Derived Outcome:** Minimize the **time** it takes to determine if an overdue invoice has been paid.

> ***Quote:*** *"I find myself compulsively checking my banking app every morning, which is a total waste of time and mental energy."*

- **Derived Outcome:** Minimize the **number of steps** required to resubmit a rejected invoice.

> ***Quote:*** *"I had to get the code, re-create the invoice, re-submit it to the portal, which reset the 60-day clock."*

Emotional Needs (Personal & Social Jobs)

These describe how the customer wants to feel (personal) or be perceived by others (social) while executing the core job.

- **Social Need: Be perceived as** a polished and professional business owner by clients.

> ***Quote:*** *"I want them to see me as a polished, professional business, not some amateur working from her kitchen table who can't type numbers correctly."*

- **Personal Need: Avoid feeling** anxious about the status of outstanding payments.

> ***Quote:*** *"This constant, low-level financial anxiety is the enemy of deep work."*

- **Personal Need: Avoid feeling** confrontational when reminding clients about payments.

> ***Quote:*** *"Ugh. The awkward follow-up. I absolutely dread it. It feels so... confrontational, like I'm a debt collector."*

- **Personal Need: Feel** confident that business finances are under control without constant monitoring.

> ***Quote:*** *"The dream is to feel totally confident that the business finances are running smoothly in the background, without me having to constantly poke and prod them. That would be real peace of mind."*

Complexity Factors

These are the situational or contextual variables that make it harder for some customers to get the job done successfully.

- **Client Type:** The job is more difficult when dealing with **large corporate clients** who have rigid procurement systems compared to smaller startups.

  > ***Quote:*** *"But my big corporate clients... that's a whole different beast. They have these rigid procurement systems."*

- **Location/Situation:** The job is more difficult when **traveling or on vacation**, away from a normal work setup.

  > ***Quote:*** *"When I'm traveling for a conference or trying to take a rare vacation, keeping up with invoicing is a complete nightmare."*

- **Project Staffing:** The job is more complex when **using subcontractors** who must be paid before the client's payment is received.

> ***Quote:*** *"When I bring in a freelance developer... I have to pay them out of my own pocket, and then I have to float that cost for 30, 60, or... 150 days."*

- **Client's Tools:** The job is harder when required to use a **client's specific, inefficient tools** (e.g., a "clunky" online portal).

> ***Quote:*** *"I have to log in to their clunky, ancient-looking online portal, manually enter all the same information again..."*

Consumption Chain Jobs

These are the tasks a customer must perform throughout the lifecycle of owning and using a specific product or service (in this case, primarily QuickBooks).

- **Consumption Job: Configure** the software for a new client.

> ***Quote:*** *"Also, setting up a new client is a pain; there are so many fields to fill out that I don't even use."*

- **Consumption Job: Pay for** the software subscription.

> ***Quote:*** *"And don't get me started on their subscription fee, which just went up again last quarter."*

- **Consumption Job: Generate** specialized financial reports from the software for an accountant.

> ***Quote:*** *"I have to export a bunch of reports from QuickBooks, but they're never quite right."*

- **Consumption Job: Manually match** bank deposits to the correct invoices within the software.

> ***Quote:*** *"It can pull in transactions, but I still have to manually categorize everything and match deposits to the right invoices."*

Financial Outcomes

These are the economic metrics the customer is trying to optimize related to the cost and financial impact of getting the job done.

- **Financial Outcome:** Minimize the **amount of non-billable time** spent on financial administration.

> ***Quote:*** *"Every hour I spend chasing invoices is a non-billable hour. It's pure overhead."*

- **Financial Outcome:** Minimize the **percentage of revenue** lost to payment processing and bank transfer fees.

  > ***Quote:*** *"And if a client wants to pay by credit card... that's $300 that just vanishes. It's a significant chunk of my profit..."*

- **Financial Outcome:** Minimize the **risk of incurring financial penalties** for incorrect tax filings.

  > ***Quote:*** *"If I mess it up and underpay my taxes, the penalties can be really steep."*

Related Jobs

These are separate functional jobs the customer is trying to get done before, during, or after the core job of obtaining payment.

- **Related Job: Create** project proposals for prospective clients.

  > ***Quote:*** *"I have a template, but I spend hours customizing it, trying to get the scope description just right so there are no surprises later."*

- **Related Job: Track** billable hours for client projects.

> ***Quote:*** *"For my hourly projects... I track my hours in a dedicated spreadsheet."*

- **Related Job: Prepare** financial records for tax filing.

> ***Quote:*** *"I n January, my accountant sends me this long list of things he needs. I probably spend two full weekends just gathering, organizing, and sending documents to him."*

RESOURCES

Interview Guides

Download our comprehensive interview guides to help you conduct effective JTBD interviews:

Defining the Market (JTBD) Interview Guide

Market Definition Interview Guide (PDF)

Description: One of the most common mistakes product teams make is defining their market by the technology they use (e.g., "The AR/VR Market") rather than the value they provide. This leads to fragile strategies that break when technology shifts.

This interview guide is a tool designed to help Product Managers and Consultants facilitate internal workshops or interviews with users. Its goal is to move your team away from a product-centric worldview and toward a stable, "Job-based" market definition.

Use this guide to:

- **Align your team:** Move stakeholders from a "Solution Mindset" to a "Problem Mindset."
- **Identify the true user:** Distinguish between the "Job Executor" (who uses it) and the "Purchase Decision Maker" (who buys it) to avoid building shelfware.
- **Future-proof your strategy:** Apply the "Time Travel Test" to ensure your market definition holds true regardless of technological trends.

The Master Job Deconstruction Interview Guide

[Master Interview Guide: Core Job, Job Map, Needs (PDF)]

Description: This guide is a guide designed to help Product Managers, Researchers, and Founders conduct Jobs-to-Be-Done (JTBD) interviews. Unlike standard user interviews that focus on opinions or product preferences, this script focuses on **process**.

Using the "Universal Job Map" methodology, this guide provides a step-by-step script to deconstruct the user's core workflow. It helps you move beyond vague customer statements to uncover the granular steps, hidden friction points, and needs for each step in the job map.

What's Inside:

- **The "Middle-Out" Technique:** A method to anchor the user's memory in the execution phase to prevent vague answers.
- **Universal Job Map Script:** Questions specifically designed to uncover the 8 distinct steps of any job (Define, Locate, Prepare, Confirm, Execute, Monitor, Modify, Conclude).
- **Needs Translation Cheat Sheet:** How to turn user complaints ("It's slow") into need statements ("Determine the time it takes to...").

RESOURCES

Further Readings and Sources

Top Book Recommendations of further readings on JTBD

Books

- Klement, Alan. When Coffee and Kale Compete: Become Great at Making Products People Will Buy. 1st ed., CreateSpace Independent Publishing Platform, 2018. ISBN-13: 978-1980611600. Available on Amazon.
- Kalbach, Jim. The Jobs To Be Done Playbook: Align Your Markets, Organization, and Strategy Around Customer Needs. 1st ed., Two Waves Books, 2020. ISBN-13: 978-1933820682. Available on Amazon.
- Ulwick, Anthony W. Jobs to be Done: Theory to Practice. CreateSpace Independent Publishing Platform, 2016. ISBN-13: 978-0990576740. Available on Amazon.
- Wunker, Stephen. Jobs to Be Done. 1st ed., IdeaPress Publishing, 2023. ISBN-13: 978-1646871087. Available on Amazon
- Christensen, Clayton M., Dillon, Karen, Hall, Taddy, & Duncan, David S. Competing Against Luck: The Story of Innovation and Customer Choice. Illustrated ed., Harper Business, 2016. ISBN-13: 978-0062435613. Available on Amazon.
- Spiek, Chris, & Moesta, Bob. The Jobs-to-be-Done Handbook: Practical Techniques for Improving Your Application of Jobs-to-be-Done. 1st ed., Re-Wired Group, 2014. ISBN-13: 978-1493683187. Available on Amazon.

Full Reference List

Chapter 1 Sources

[1] Drucker, P. F. (2006). *The Practice of Management*. Harper Business. ISBN: 978-0060878979. Available on Amazon.

[2] Levitt, T. (1960). Marketing Myopia. *Harvard Business Review*, 38(4), 45-56.

[3] Levitt, T. (2004). Marketing Myopia. *Harvard Business Review* (reprint). Available at: https://hbr.org/2004/07/marketing-myopia

[4] Ulwick, T. (n.d.). The Birth of Predictable Innovation. *Strategyn.com*. Available at: https://strategyn.com/jobs-to-be-done/history-of-jtbd/

[5] Ulwick, A. W. (2002). Turn Customer Input into Innovation. *Harvard Business Review*, 80(1), 91-97. Available at: https://hbr.org/2002/01/turn-customer-input-into-innovation

[6] Christensen, C. M., & Raynor, M. E. (2013). *The Innovator's Solution: Creating and Sustaining Successful Growth*. Harvard Business Review Press. ISBN: 978-1422196571.

[7] Christensen, C. M., Anthony, S. D., Berstell, G., & Nitterhouse, D. (2007). Finding the Right Job For Your Product. *MIT Sloan Management Review*, 48(3), 38-47. Available at: https://sloanreview.mit.edu/article/finding-the-right-job-for-your-product/

[8] Bettencourt, L. A., & Ulwick, A. W. (2008). The Customer-Centered Innovation Map. *Harvard Business Review*, 86(5), 109-114. Available at: https://hbr.org/2008/05/the-customer-centered-innovation-map

[9] Moesta, B. (n.d.). What Is the Jobs To Be Done Framework? *The Re-Wired Group*. Available at: https://therewiredgroup.com/learn/complete-guide-jobs-to-be-done/

[10] Christensen, C. M., Dillon, K., Hall, T., & Duncan, D. S. (2016). *Competing Against Luck: The Story of Innovation and Customer Choice*. Harper Business. ISBN: 978-0062435613.

[11] Moesta, B., & Traynor, D. (2016). Bob Moesta on Jobs-to-be-Done. *Intercom Podcast*. Available at: https://www.intercom.com/blog/podcasts/bob-moesta-on-jobs-to-be-done/

[12] Maurya, A. (n.d.). The Backstory Behind Customer Forces Stories. *LeanStack Blog*. Available at: https://blog.leanstack.com/the-backstory-behind-customer-forces-stories/

[17] Maurya, A. (2019). What Is a Job-To-Be-Done (JTBD)? *Medium/LeanStack*. Available at: https://medium.com/lean-stack/what-is-a-job-to-be-done-jtbd-21304ce7441b

[18] Jain, R. (2021). Martech's Magicians: How Marketers Turn Technology to the Service of Consumers. *Available on Amazon*.

[19] Klement, A. (2018). *When Coffee and Kale Compete: Become Great at Making Products People Will Buy*. CreateSpace Independent Publishing Platform. ISBN: 978-1534873063. Available on Amazon.

[20] Klement, A. (2018). Know the Two — Very — Different Interpretations of Jobs to Be Done. *JTBD.info*. Available at: https://jtbd.info/know-the-two-very-different-interpretations-of-jobs-to-be-done-5a18b748bd89

Chapter 2 Sources

[21] Strategyn. (2019). *Innovation Track Record Study*. Strategyn.com. Available at: https://strategyn.com/

Chapter 3 Sources

[22] Moonshots. (n.d.). Travis Kalanick – Episode 2. *Moonshots Podcast*. Available at: https://www.moonshots.io/episode-02-travis-kalanick

[23] TechGoondu. (2015, September 17). Uber: We want transportation to be as reliable as running water. *TechGoondu*. Available at: https://www.techgoondu.com/2015/09/17/uber-we-want-transportation-be-as-reliable-as-running-water/

[24] Airbnb. (2014, July 16). Airbnb announces complete rebranding with a "symbol of belonging". *Airbnb Newsroom*. Available at: https://news.airbnb.com/airbnb-announces-complete-rebranding-with-a-symbol-of-belonging/

[25] Chen, A. (2021). *The Cold Start Problem: How to Start and Scale Network Effects*. Harper Business. ISBN: 978-0062969743. Available on Amazon. See also: de Vroome, T. (2023). Book Summary: The Cold Start Problem. *Medium*. Available at: https://tdevroome.medium.com/book-summary-the-cold-start-problem-2306afb9bc9d

Chapter 4 Sources

[26] Kalbach, J. (2020). *The Jobs To Be Done Playbook: Align Your Markets, Organization, and Strategy Around Customer Needs*. Two Waves Books. ISBN: 978-1933820682. Available on Amazon.

[27] Bezos, J. (2022). Jeff Bezos: Amazon and Blue Origin. *Lex Fridman Podcast #405*. Available at: https://www.youtube.com/watch?v=DcWqzZ3I2cY

[28] Ulwick, A. W. (2016). *Jobs to Be Done: Theory to Practice*. CreateSpace Independent Publishing Platform. ISBN: 978-0990576747. Available on Amazon.

[29] Hair, J. F. Jr., Black, W. C., Babin, B. J., & Anderson, R. E. (2010). *Multivariate Data Analysis* (7th ed.). Pearson. ISBN: 978-0138132637.

Chapter 5 Sources

[28] Ulwick, A. W. (2016). *Jobs to Be Done: Theory to Practice*. (See Chapter 4 Sources)

[30] Ulwick, T. (2017). Mapping the Job-to-be-Done. *Jobs-to-be-Done.com*, 12 Jan. 2017. Available at: https://jobs-to-be-done.com/mapping-the-job-to-be-done-45336427b3bc

Chapter 6 Sources

[28] Ulwick, A. W. (2016). *Jobs to Be Done: Theory to Practice*. (See Chapter 4 Sources)

[31] Ulwick, T. (2017). Defining Customer Needs: The Root of the Problem. *AnthonyUlwick.com*, 7 Jan. 2017. Available at: https://anthonyulwick.com/2017/01/07/defining-customer-needs-root-problem/

Chapter 7 Sources

[11] Traynor, D. (2015). Strategyn's Tony Ulwick on Jobs-to-be-Done. *Intercom Podcast*, 10 Dec. 2015. Available at: https://www.intercom.com/blog/podcasts/podcast-tony-ulwick-on-jobs-to-be-done/

[32] Sauro, J. (2018). Are Top Box Scores a Better Predictor of Behavior? *MeasuringU.com*, 2 May 2018. Available at: https://measuringu.com/top-box-behavior/

[33] Sauro, J., & Lewis, J. R. (2024). Top Box, Top-Two Box, Bottom Box, or Net Box? *MeasuringU.com*, 4 June 2024. Available at: https://measuringu.com/top-top-two-bottom-net-box/

Chapter 8 Sources

[34] Sawtooth Software. (n.d.). Creating a MaxDiff. *SawtoothSoftware.com*. Available at: https://sawtoothsoftware.com/help/discover/survey-elements/maxdiff/creating-a-maxdiff

[35] Qualtrics. (n.d.). MaxDiff Analysis Technical Overview. *Qualtrics.com*. Available at: https://www.qualtrics.com/support/conjoint-project/getting-started-conjoints/getting-started-maxdiff/maxdiff-analysis-white-paper/

[36] Chapman, C. (2024). Individual Scores in Choice Models, Part 1: Data & Averages. *QuantUXBlog.com*, 23 Oct. 2024. Available at: https://quantuxblog.com/individual-scores-in-choice-models-part-1-data-averages

[37] Sawtooth Software. (n.d.). What Is MaxDiff? *SawtoothSoftware.com*. Available at: https://sawtoothsoftware.com/resources/blog/what-is-maxdiff

[38] Guinn, A. (n.d.). Stated vs. Derived Importance. *SawtoothSoftware.com*. Available at: https://sawtoothsoftware.com/resources/blog/stated-vs-derived-importance

[39] Sawtooth Software. (n.d.). Sample Size Calculator. *SawtoothSoftware.com*. Available at: https://sawtoothsoftware.com/resources/sample-size-calculator

Chapter 9 Sources

[13] Wedel, M., & Kamakura, W. A. (2000). *Market Segmentation: Conceptual and Methodological Foundations*. Kluwer Academic Publishers.

[14] Jain, A. K. (2010). Data Clustering: 50 Years Beyond K-Means. *Pattern Recognition Letters*, 31(8), 651-666.

[15] Dolnicar, S., Grün, B., & Leisch, F. (2018). *Market Segmentation Analysis: Understanding It, Doing It, and Making It Useful*. Springer.

[16] Chrzan, K., & Orme, B. (2019). *Applied MaxDiff: A Practitioner's Guide to Best-Worst Scaling*. Sawtooth Software.

[36] Chapman, C. (2024). Individual Scores in Choice Models, Part 1: Data & Averages. (See Chapter 8 Sources)

[40] Ulwick, T. (2017). Market Segmentation Through a Jobs-to-be-Done Lens. *Jobs-to-be-Done.com*, 18 Nov. 2017. Available at: https://jobs-to-be-done.com/market-segmentation-through-a-jobs-to-be-done-lens-5ef9242de65

[41] Chapman, C., & Feit, E. M. (2019). *R for Marketing Research and Analytics* (2nd ed.). Springer. ISBN: 978-3030143152. Available on Amazon.

Chapter 10 Sources

[42] Ulwick, T. (2017). The Jobs-to-be-Done Growth Strategy Matrix. *Jobs-to-be-Done.com*, 6 Jan. 2017. Available at: https://jobs-to-be-done.com/the-jobs-to-be-done-growth-strategy-matrix-426e3d5ff86e

[43] Ulwick, T. (n.d.). The Job-to-Be-Done Growth Strategy Framework. *Strategyn.com*. Available at: https://strategyn.com/5-business-growth-strategies-jobs-to-be-done/

Chapter 11 Sources

[44] Cohn, M. (2004). *User Stories Applied: For Agile Software Development*. Addison-Wesley Professional. ISBN: 978-0321205681. Available on Amazon.

[45] Patton, J. (2014). *User Story Mapping: Discover the Whole Story, Build the Right Product*. O'Reilly Media. ISBN: 978-1491904909. Available on Amazon.

[46] Schwaber, K., & Sutherland, J. (2020). *The Scrum Guide*. Scrum.org. Available at: https://scrumguides.org/

Instructions for Claude Code: Adding Chapter 11 References

Step 1: Locate the end of Chapter 11

Find the chapter conclusion that ends with:

```text
Now go build something that helps customers make progress o
```

Step 2: Find and REPLACE the empty references section

There should be an existing placeholder:

```text
## Chapter 11 References
```

REPLACE that line (and any empty content after it) with the following:

```text
## Chapter 11 References

[1] Cohn, M. (2004). _User Stories Applied: For Agile Softwa

[2] Patton, J. (2014). _User Story Mapping: Discover the Who

[3] Schwaber, K., & Sutherland, J. (2020). _The Scrum Guide_
```

Step 3 (Optional): Add in-text citations

If you want in-text citations, make these additional changes:

Change 1: In the section **"Why Most User Stories Fail"**, find this paragraph:

```text
User stories in agile development follow a standard format:
```

REPLACE with:

```text
User stories in agile development follow a standard format
```

Change 2: In the section **"Redefining 'Done'"**, find this sentence:

```text
Outcome stories also change how you define success.
```

REPLACE with:

```text
Outcome stories also change how you define success [3].
```

Summary of changes

1. **REPLACE** the empty `## Chapter 11 References` section with the populated references list (3 references)
2. **(Optional)** ADD two in-text citation markers `[1]` and `[3]` in the relevant sections

End of Book

Practical Jobs-to-be-Done by Braden Buchanan