karat^

The Engineering Leader's Guide to Technical Interviewing

karat^

The Engineering Leader's Guide to Technical Interviewing

Why do we conduct job interviews? You might think there's a pretty obvious answer: to find the best person we possibly can for a given job.

But in reality, our motivations and approaches are a bit more nuanced. For example, modern technology has made it so easy for hundreds or even thousands of applicants to apply for a job that some of us actually construct our interviews to keep the "wrong" people **out** of the job. We build our interviews as a series of "gates" that people have to pass through, and we hope that those gates will "weed out" as many people as possible. But a danger in that approach is that we'll encourage the creation of a monoculture in our teams. By only "allowing" certain people in, we start to bias for a single type of person.

It's also easy to commingle some of the different things we care about into one interview. For example, is a given interview intended to evaluate someone's job skills, or is it intended to help us decide if we like the candidate or not? Both are valid things to consider when hiring someone, but when we combine them into a single interview, it makes it harder to make the interview consistent and measurable.

Let's go back to first principles and really define what we want from a series of job interviews. The most concise and meaningful explanation is this:

A good job interview teaches us something about a candidate, and it should be exclusively things that relate to how well that person is likely to perform in the job they've applied for.

With a statement like that as our grounding principle, we can start thinking about the important elements of an interview. For example:

Interviews should teach us about a candidate's ability to perform the job purely from a skills perspective. Skills can be learned and improved over time, and they can be objectively observed and measured.

Interviews might teach us something about a candidate's job-related aptitudes. Aptitude is the innate or natural ability to learn to do something. For example, someone might not know how to fix a car, but they might have a strong mechanical aptitude that makes it easier for them to learn how. Aptitudes can be difficult to quantify and measure though.

Interviews should teach us about how well a candidate is likely to perform in the social elements of our organization. Organizations often refer to this as a "culture fit" interview, although "culture" isn't necessarily a good word to use because it can be vague and difficult to pin down. Instead, we might focus on more specific terms, such as whether a candidate shares our organizational values or whether the candidate communicates in a way we're comfortable with. Like aptitudes, these social elements can be difficult to measure objectively — they often come down to "we know what we like when we see it."

With those points in mind, we can see that interviews will often contain a mix of things that we can measure objectively and things we cannot. For example, if we're concerned about a candidate's ability to type quickly and accurately, we can measure it objectively through a typing test. We might not even need to meet the candidate in order to test that skill. On the other hand, if we're concerned about a candidate's ability to effectively communicate complex topics, that's more difficult to objectively measure.

It's important to acknowledge that just because something is difficult to measure objectively doesn't mean it isn't important or relevant to the job. Many of the things that make someone incredibly successful in a job *are* difficult to objectively measure. But in order to make interviews more consistent and scientific, we should try to firmly separate those things which *can* be objectively measured from those things which *cannot.*

When we focus on objectively measurable skills, it can be easier to compare candidates to one another. For example, Candidate A was able to type 75 words per minute with a 10% error rate, while Candidate B was able to type 60 words a minute with a 1% error rate. By separating those interview elements, we can have a clearer idea of where candidates can be accurately compared to one another, versus more subjective areas where we're relying more on our intuition.

Even for more subjective considerations though, we can make cross-candidate comparisons more viable when we focus on consistency in the interview — both in the questions we ask and how we evaluate candidate responses.

Part of being scientific is controlling for as many variables as possible so that whatever you're studying can be studied as accurately as possible. If you think about an interview, we typically have two major variables: the interviewee or candidate, and the interviewer. If both of those things are highly variable, then it's very difficult for the output of the interview to be accurate. When the interviewer asks different questions each time, we can't always tell if the candidates are doing a good job or not. This is because we're not really looking at the candidate, we're looking at an intersection of the candidate and the interviewer. However, when the interviewer always asks the same questions every time, we can be reasonably sure that most of the differences we see between interviews are due to the candidates themselves. By "controlling" the variability of the interviewer, we ensure that any remaining variability comes from the candidates, and we can be pretty certain that what we're learning from the interview is things that the candidates themselves are teaching us.

As we move forward in how to build interviews that are predictive of job success, that are fair to the candidates and our organization, and that are possibly even enjoyable for the candidates, let's focus on two more refined core principles:

- Job interviews should exclusively teach us things about our candidates — not about our interviewers or our organization.

- To achieve the first principle, we must be as scientific as possible, controlling for as many variables as possible so that the maximum amount of remaining variability comes entirely from the differences between the candidates themselves.

It turns out that the best way to achieve these principles is to focus less on the interview questions we ask, and to focus more on how we evaluate the answers to our questions. So we'll begin our discussion with scoring, the fundamental element of a successful, predictable, fair, and enjoyable interview.

With the foundation of strong scoring in mind, we'll look at how to design interviews that are predictive of success, fair and equitable, and enjoyable for candidates, and use science, particularly some basic fundamentals of cognitive science, to guide us. We'll also examine some of the dangers we should explicitly try to avoid in interviews, including the dangers of interviews that unintentionally lead to "monoculture mentality." Finally, we'll close the loop on interviewing by looking at ways to hold ourselves accountable to our stated interviewing goals.

In the end, you'll have a framework for creating interviews that are repeatable, reliable, and relatable — interviews that are, in short, successful.

karat^

Table of Contents

Scoring: The Key to Success

One of the goals of designing interviews is to make them comparable. That means our evaluation, or score, for one candidate can be reliably and accurately compared to every other candidate for the same job role.

We also want our scoring to be consistent. In the world of assessment science, there's a measurement called inter-rater reliability, which is often expressed in shorthand as a p-value. This is an indicator of how consistent assessment scores are between multiple evaluators — in this case, the people conducting interviews. It works something like this: If you were to have two interviewers each interview the same 100 people and they both came up with exactly the same rating, then you'd have a p-value of 1 or 100% agreement. If they both came up with different scores for all 100 candidates, then you'd have a p-value of 0 or 0%.

The goal is to get 100% agreement between all of your interviewers, and if you think about it for a moment, you'll realize that the more objective and measurable your interview questions are, the easier it will be to hit that 100% goal. Conversely, when you interview on things that are inherently difficult to quantify — such as a candidate's verbal communications skills — it's going to be much more difficult to hit 100% inter-rater reliability. That doesn't mean you shouldn't interview for those more subjective skills, but it does mean you need to be aware that you'll have a level of interviewer variability that you can't control for.

So here's the elephant in the room: **anytime you can't fully control for the variability of your interviewers, you will inherently have a lower p-value, and you will run a risk of introducing conscious and unconscious bias into your interviews.**

One side effect of very high inter-rater reliability is that you truly minimize the opportunity for individual interviewer bias to enter the interview. If 50 interviewers can interview 100 candidates, and all 50 interviewers come up with the same score for each candidate, then it's pretty difficult to imagine where an individual interviewer might be introducing their own biases. On the other hand, if those 50 interviewers turn in markedly different scores for each candidate, then there's a good chance that at least *some* of that variability is from interviewer bias.

It's very rare for most organizations to monitor inter-rater reliability, which means they don't know if they're introducing bias into their interviews or not. One reason for the rarity is that it'd be cruel to ask a group of candidates to go through the same interview with multiple interviewers. This is a topic we'll come back to toward the end, when we discuss ways to hold ourselves accountable for our interview goals and outcomes. But the short answer is that it's difficult, with real candidates, to measure ourselves and hold ourselves accountable — and so in many cases, we don't know if we're doing a good job or not.

But let's get back to the topic of scoring interviews: how can we work toward a higher level of inter-rater reliability?

The answer is rubrics or scorecards.

How to Design Great Scoring Rubrics

The way to design a good interview is to start with the answers, and rubrics are the key. There are a few different types of rubrics though and each one affects inter-rater reliability — and therefore, our ability to be fair, consistent, and as free of bias as possible. For these examples, we're going to consider both a hard skill question, which we'll construct around a software developer role, and a soft skill question, targeted at the same role.

An example of a hard skill interview question is:

> *Ask the candidate to write a function that processes a directory tree to count the number of PNG and JPEG files contained in that directory tree.*

We might write a scorecard that says something like:

- Award 1 point if the candidate creates a working solution in under 20 minutes.
- Award 1 more point if the candidate's solution is optimal.

This is an okay rubric, but it's not great. Presumably, we can score the working solution part of this with a high degree of inter-rater reliability — after all, that's a pretty yes-or-no decision without a lot of need for interpretation on the part of the interviewer. But the optimal piece seems to leave a lot open to interpretation, and that's where an interviewer's unconscious biases can come into play. Because the interviewer is being asked to make a judgment call, their brains might unconsciously start considering things other than the actual code the candidate wrote. Maybe they don't like the way the candidate indents their code, or they don't think the candidate did a good job of asking clarifying questions. While it's unlikely that those concerns were part of the intent of the interview question, the vagueness of "optimal" provides the opportunity for the interviewer to consider those when deciding whether or not to award the additional point.

So we can rework that rubric to be more specific:

- Award 1 point if the candidate produces a working solution in under 20 minutes, which correctly produces the value of 10 PNG files and 18 JPEG files using the sample data provided.
- 0 additional points if the candidate uses a memory-intensive approach, such as loading all the files in the directory tree into an array and then filtering that array.
- 1 additional point if the candidate writes a recursive function that is called each time the function encounters a new sub-folder.
- 1 additional point if the candidate uses a directory tree access function that supports filtering, and enables filtering for just PNG and JPEG files.

That rubric is far more specific and less open to interpretation, and it would be even better if it provided code examples of good, better, and best solutions. A candidate might still produce something that doesn't look exactly like any of the examples, but this is where the value of a human interviewer — as opposed to using an automated solution of some kind — comes into play. The interviewer can recognize which approach the candidate is taking, match it to the appropriate example, and award the correct number of points.

Some of the characteristics of a really good rubric include:

- A high level of specificity about what "good," "better," and "best" actually look like.
- Explicit instructions on how many points to award for each possible candidate answer.
- Sufficient specificity so that the interviewer doesn't need to engage in judgment calls when awarding points.
- Comprehensive enough that everything we might care about as an organization is represented in the scorecard — meaning anything we don't care about is explicitly left out.

On that last point, it's completely valid to expressly list things we don't care about. For example:

> When considering optimality, we do not consider stylistic choices such as variable or function naming, indentation style, brace positioning, or related concerns. We also do not care if candidates choose to use v1 framework components versus more modern v2 framework components.

These kinds of clarifications make it even more likely we'll have a high level of inter-rater reliability.

Now let’s move on to a softer-skill kind of question:

> Ask the candidate to explain the difference between a dictionary and an array, and to describe which one would be more appropriate for storing key-value pairs that represent configuration settings.

This is a good example of what can seem like a very objective interview question that can be scored with a high level of inter-rater reliability. However, the details of the rubric can tell a different story. Consider:

+ Award 1 point if the candidate’s explanation is clear and concise, and if the candidate suggests a dictionary as being more appropriate than an array.

There’s a lot of wiggle room there. What exactly does “clear” mean? What does “concise” mean? There’s nothing in the rubric that even suggests we’re looking for a correct explanation! The odds of this rubric giving us a high level of inter-rater reliability seems pretty low because there’s a lot of room for subjectivity on the part of the interviewer.

Rating things like communications skills is always going to be at least somewhat subjective. As human beings, we don’t all feel the same way about how other people speak to us. One person’s feelings about “concise” might be very different from another’s. That doesn’t mean we shouldn’t interview for those things, but we have to recognize that we’ve stepped into highly subjective, personal territory and that people’s opinions aren’t wrong.

However, we can tighten up scorecards a bit.

- Award 1 point if the candidate correctly describes the difference between arrays and dictionaries, touching primarily on “Array: An array stores a collection of elements of the same type. Elements in an array are typically accessed in a sequential manner. Dictionary: A dictionary stores key-value pairs, where each key must be unique within the dictionary. Keys can be of any hashable type, and values can be of any type.”
- Award 1 point if the candidate correctly identifies a dictionary as being the best choice for the scenario described.
- Award 1 point if the candidate’s explanation is concise — coming in under about 5 minutes — and the candidate speaks with no or only minimal hesitation or confusion about the topic.
- Award 1 point if the candidate uses an analogy to illustrate the purpose of an array, a dictionary, or both; or if the candidate provides a concrete example of either or both.

We haven’t completely removed subjectivity — ”concise” is still a little loosely defined — but we’ve been more explicit about what we care about. We’ve put a bit of a time-box on how long a “concise” explanation can be, although we’re still open to interpretation on words like “minimal.” Again, things like communicating are really tough to rate with complete objectivity. But we’ve listed some characteristics we find valuable, such as using analogies or examples in communications. This suggests the organization values someone with teaching skills, for example. In looking at this improved scorecard, which has kind of forced us to spend more time thinking about what we value, we might even go back and modify the question so that it’s driving more toward those things.

This is why we say that you often start with the rubric — not the question. Obviously, you have to have a question in mind when you build a rubric, but thinking about the rubric forces you to think about what really matters. To approach this same example from the other end, we might start by asking ourselves, “What matters?”

- We want candidates who can teach and explain, which means we value the use of analogies and concrete examples.
- We want candidates who can speak concisely, getting straight to the point.
- We obviously want candidates who can be accurate about things they know.

With those bullets, we've actually started constructing the rubric before we wrote the actual question. Now we can think about a question that precisely hits those bullets, rather than starting with a question and then trying to imagine how it hits on the points we care about.

Designing Interviews to Be Predictive

A job interview is all about figuring out if a candidate is right for the job, but we need to be thoughtful about how we define "right for the job." Specifically, it should mean the candidate is able to be successful in the role. But what does success look like?

In theory, we should be able to turn to the role's job description, which should clearly and unambiguously describe what "success" means. In reality, it's rare for job descriptions to be that revealing. So where can we turn?

In the world of assessment science, we'd begin by performing a Job Task Analysis (JTA). As the name implies, that study would reveal the key job tasks needed on the job. A good JTA will look at a cross-section of criticality and commonality. In other words, it'll show us which tasks are most commonly performed on the job — the ones that incumbents in the role perform day in and day out — as well as the ones that are most critical to the role. Those latter tasks are the ones that, if performed wrong, will cause the highest level of negative impact on the organization. With a good JTA in hand, you can select a group of tasks that are both common and critical and focus on those in the job interview. That intersection between commonality and criticality shows what "success" truly looks like in the role.

Without a JTA, or something substantially like it, you may have no idea what success actually looks like. The sad reality is that most job interviews start from a "gut feeling" of what's required to be successful in a role. Even more sadly, many job interviews are not built around what "success" looks like, but instead around topics that weren't well-understood by people that the organization perceived as being unsuccessful in the role.

Read that a few times and let it sink in. Many times, organizations build their job interviews without a strong understanding of what success looks like. Instead, they consider past incumbents who they feel were unsuccessful and build an interview that those past employees wouldn't be able to pass.

At the outset of this report, we mentioned that some interviews are constructed as gates designed to keep people out, while other interviews are designed to include those who are qualified. Many organizations instinctively do the former — they consider people they wish they hadn't hired in the past and construct interviews that would keep those people out. They interview from a place of fear, rather than from a place of hope, and that behavior is driven primarily by the fact that the organization hasn't actually defined what success looks like.

How to Conduct a Job Task Analysis

Constructing a decent JTA doesn't need to be hard. You can follow these six steps to create a JTA for any role and turn your interviews from exclusionary to inclusionary.

1. Start by polling your incumbents — all of them, but especially those who routinely meet expectations in their performance reviews. Ask them to create bullet lists of the tasks they perform. Don't worry about commonality or criticality at this stage, but do worry about specificity. For example, software engineers might be tempted to just put "write code," but ask them to dig a little deeper. What kind of code? What does the code actually do? Do they write code that maintains state,

that scales in a concurrent environment, that accesses databases, or that secures data? Sometimes putting a group into a conference room with some pizza for a couple of hours can be a useful way to come up with this list.

Notably, soft skill tasks can and should be included. That might include things like "presenting at stand-up meetings," "constructing status slide decks," and so on.

2. Continue by asking one or two team members to de-duplicate the combined list. In many cases, you'll find you have the same tasks listed multiple times but worded a bit differently, and it's a quick exercise to consolidate them.

3. Send the consolidated list around to *everyone* who currently sits in the role. For each task, ask them to rate the commonality of each task on a scale of 1 (I almost never do this) to 5 (I do this multiple times a day). Also ask them and their managers to rate the criticality of these tasks, again on a scale of 1 (it's not a huge deal if this is done wrong, perhaps because it's so easy to catch or fix) to 5 (the world will explode if this is done wrong).

4. For each task, average out the commonality and criticality responses.

5. For each task, multiply criticality by commonality. You'll now have rankings from 1 to 25. Sort these in descending order.

6. Your top 10 tasks are the ones to consider for an interview. Some of these may not be feasible to actually evaluate in an interview, and that's fine — just keep moving down the list until you have 10. In reality, 10 will be too many to properly evaluate, but it gives you a little flexibility in constructing the final interview.

With your JTA in hand, you now have a much more scientific and data-driven idea of what "success" looks like in a role. If a candidate can successfully demonstrate their ability to perform those tasks in an interview, then there's a good chance they'll be able to perform those tasks well on the job too. Perhaps more importantly, you've taken an important first step in removing bias from the interview because you're now focused solely on job-related tasks.

Identifying Correlations to Improve Interviews

Let's return to one of the first principles of interviewing: each thing we examine in an interview, meaning each question we ask or task we consider, should teach us something new about a candidate. For example, let's suppose we have three questions, identified as A, B, and C. Over time, we might notice that candidates who always answer A correctly also answer C correctly, and candidates who get A wrong always get C wrong. In that case, A is said to correlate strongly to C. We would modify the interview so that we asked one or the other, but not both. After all, if a candidate always gets them both or misses them both, then asking both doesn't teach us something new. We only need to ask one of them to learn what we're going to learn — and we free up time to ask a different question that *does* teach us something new.

That correlation principle can actually be measured over time as you interview candidates, which is why it's important to keep track of interview results. Like the p-value for inter-rater reliability, correlation is measured on a scale of 0 to 1. In our above example, A would be described as having a correlation of 1 with C. Now consider a case where half the candidates who get A right also get B right and vice-versa; A would be said to have a .5 correlation with B. Typically, you would drop any questions that have a correlation of .8 or more with a question you're already asking. That is, if 80% or more of candidates always get or miss them both, then don't ask both.

Correlation theory is a great example of how interviews should evolve and improve over time. Out of the gate with a brand new interview, it's tough to guess where correlation might occur. It's not until you have data that you can start improving. And of course, your interviews need to be consistent, so that the data you're collecting can be used to make these kinds of improvements.

In the end and over time, your JTA-based interview will do a better and better job of identifying high-potential candidates. Of course, you do need to hold yourself accountable. We'll discuss that in more detail later, but for now that looks like tracking the performance of candidates who pass your interview and are hired. At their next performance review, does everyone agree that they're doing well? If so, then you can be pretty confident that your interview was, in fact, predictive of success.

Designing Interviews to Be Fair

We want interviews to be fair, but what do we actually do to ensure the fairness of an interview?

We should probably start by defining "fair." One dictionary definition is:

*adjective,*fair·er, fair·est.

1. free from bias, dishonesty, or injustice:
2. *a fair decision;*
3. *a fair judge.*

"Free from bias" is probably a good, concise definition of the term for our purposes. So how do we ensure that our interviews are free from bias?

Structure. Consistency. A laser-like focus on job tasks, to the exclusion of as much else as possible. Objective scoring rubrics that leave as little room as possible for interpretation by individual interviewers. A relentless focus on what actually matters on the job, and a willingness to rethink what *truly* matters for job success.

You're probably already aware of the concept of unconscious bias, and the role it can play in processes like job interviews. Having unconscious biases doesn't make someone a bad person, it simply makes them a human being. We're all hardwired to feel more comfortable around people like ourselves — people who look similar, speak similarly, and exhibit the same values that we prefer in ourselves. Having these unconscious biases doesn't mean we harbor any ill will toward people who look, speak, or think differently — it simply means we're human. That said, we can acknowledge those unconscious biases and work to keep them from coming into play during what should be a fair, equitable, and bias-free interview process.

- **Structure**: Having an interview that is structured is a good starting point. Structure means that our interviews have a fixed format. We might spend 10 minutes discussing one topic, and then move on to another topic. We might start the interview with 5 minutes of introductions to help a candidate feel more at ease. Whatever our structure is, the point is to have one. Ideally, we even share that basic structure with candidates up front, because doing so will help them feel more confident and enable them to better bring their best selves to the interview.

- **Consistency**: A critical element of fairness is treating every candidate the same, and that means conducting every interview the same. This doesn't necessarily mean we have only one set of interview questions — we do need to be aware of the fact that, especially in large, well-known organizations, the questions we ask during interviews will wind up on the Internet somewhere. It's okay to have multiple different "interview sets," so long as they are equivalent — meaning, each interview set targets the same job competencies and the same level of thinking on Bloom's taxonomy, and follows the same structure. Consistency is how we ensure that the scores we give to one candidate are directly comparable to the scores given to another candidate, allowing "apples to apples" comparisons.

- **A Focus on Job Tasks**: While it can seem more personable to ask a candidate about their hobbies, their families, or the neighborhood where they live, those topics have no relation to a candidate's ability to perform a job and therefore shouldn't be included in a job interview. Once you hire someone you can start to get to know them as a person. During the interview, focus solely on their ability to perform the most common and critical job tasks — ideally, tasks you've identified in a valid Job Task Analysis.

- **Objectivity**: To the maximum degree possible, score candidates using a rubric or scorecard that provides highly detailed guidance and absolutely minimizes interpretation by individual interviewers. This is truly the key to removing bias. Here's a good exercise: After you've come up with interview questions, have two or three existing team members answer those questions in writing. Then, share those answers — without the team members' names attached — with several prospective interviewers, and ask those interviewers to use your proposed rubric to score those answers. If you don't get the same scores from each interviewer — if you have poor inter-rater reliability — then your interviews will almost inevitably contain at least traces of unconscious bias. Consider reworking your rubrics and trying again until you can get a better p-value across interviewers.

Also consider the context of your interview questions. For example, if you work at a large bank, it can be tempting to frame interview questions in the context of banking. After all, doing so can feel very "real world." But when you do that, you're automatically adding a bias toward candidates who have already worked in that industry. If that bias is intentional — for example, perhaps your job description actually specifies a requirement for prior banking experience — then it's fine to acknowledge it and continue. If your job description doesn't specify a requirement for that prior experience, then you're creating a strong bias against candidates who had no idea to expect that kind of specificity in their interview.

You can perform a kind of "bias check" on your interview questions and scoring rubrics. Sit down as a team and ask yourselves, "Is there anything in here that might require some background, experience, or other information that is unique to us, and not specifically demanded in the role's job description?"

karat^

Designing Interviews to Be Enjoyable

Can an interview be enjoyable for a candidate? Rather than blithely answering “yes” or “no,” consider this: *They’d better be.*

A job interview is your first encounter with what will hopefully be a future colleague. A coworker, and possibly even a friend. Certainly a member of your professional network. With all of that on the line, don’t you want to make a good first impression? You do, as the saying goes, only get one chance to make that first impression, and you’re going to make it long before that person’s first day on the job.

A job interview also says a lot about your organization, what it values, and how it treats people — and please believe that candidates talk. Numerous websites — beyond the obvious ones like Indeed and Glassdoor — offer candidates a chance to anonymously share their impressions about organizations’ interview approaches, and candidates are rarely shy about clearly stating their feelings. Organizations with a bad interviewing reputation tarnish their employer brand, and they make it harder for themselves to recruit qualified talent.

First, let’s keep in mind that interviews are inherently stressful. To you and your team, they can be an interruption in the day, an unwanted burden, and an unwelcome social encounter. But to the candidate, interviews can be very nearly life or death. To the candidate, a good interview may mean the difference between putting food on the table or not, and that creates a lot of pressure. Nobody performs their best under that kind of pressure.

We often say that, “Candidates lose 30 IQ points the minute the interview starts.” It’s a bit of a quip, obviously, but it acknowledges an underlying truth that none of us are at our best in that kind of situation.

Your interview needs to acknowledge and accommodate the existence of that stress.

For example, you can't reliably "test" an interview on existing employees — they'll never feel the same pressure because the stakes are practically nonexistent for them. Employees will always do better and will almost always feel an interview was "too easy," simply because they're performing in a much better headspace than a candidate will.

Many of the tactics for creating an enjoyable interview, therefore, involve minimizing that stress while firmly realizing that we can never, ever fully eliminate it and see candidates performing at their peak. Here are some ideas on how to do so.

- **Job Descriptions:** Most organizations' job descriptions do not set candidates — or, frankly, interviewers — up for success. Your job description should clearly state the minimum required skills that the candidate needs for the job, with as much specificity as possible.

If you're writing your job descriptions well, then they should essentially be a clear, concrete list of things the candidate should expect to discuss during the job interviews. If you don't know enough about the job to write descriptions to that level of detail, then you also don't know enough to write an interview that is truly fair, predictive, and enjoyable.

Poor	**Good**
Strong back-end development experience	Strong skill in writing Java 11 code in a highly concurrent environment
Experience with Cisco routers	Demonstrated skill of configuring Cisco routers in both internal and edge scenarios
Experience with SQL	Skill in writing multi-table joins in SQL, creating and using user-defined functions, and analyzing index performance for query optimization

- **Pre-Interview Communications:** Be transparent about what the candidate should expect, and offer any tips for preparation that you feel would be helpful. For example:

During our interview, we'll be asking you to look at code written in Java 11 and SQL. We'll be asking you to comment on that code from the perspective of security, scalability in a highly concurrent environment, and performance. You won't be required to write code from scratch, although you're welcome to write a line or two to demonstrate your suggestions for improvement. You won't be asked to run code. We'll also be asking some questions to check your understanding of Java and SQL concepts that are especially important to us, with a focus on concepts related to high scalability and concurrency.

- **Interview With Compassion:** During the interview, don't let candidates "hang." If they're stuck, offer a push in the right direction — a push that's been designed and documented into the interview script at the outset. Track those pushes, because doing so will make it easier to compare candidates to each other. When a candidate does well, tell them so right away: "That's great, thank you. You got that one on the nose." If a candidate doesn't do well, try to reassure them: "Hey, we don't expect everyone to know everything. If you encountered this on the job, how would you go about finding the answer? Do you want to share your screen and walk me through your process?" Remember that failure is how we learn and we're often happier to hire someone who can self-teach, so these interview "failures" can be an opportunity to learn something valuable about a candidate. Again, build that opportunity into your interview script, track it, and handle it the same way for every candidate.

- **Connect to the Job:** With every question, try to provide a little bit of context about why that question matters to you. For example:

  The role you're interviewing for works on both back-end code and middleware code. We run in an extremely high-scale, very concurrent environment, and so dealing with concurrency, locks, and other issues like that is part of our daily existence. I'm going to ask you some questions related to that kind of work.

The idea is to give the candidate every chance to shine and to help them know what to expect — with as much detail as possible — at every step. Keep the interview focused on those things that matter most to the job, and make it clear why those things matter to the job.

Beware the Monoculture

As stated previously, an intrinsic part of human psychology is that we tend to prefer people who are like us. Humans have an immense capacity for othering people. "I'm safer with my tribe," our brains are constantly telling us. But intellectually, we know that having a diversity of backgrounds, perspectives, opinions, and approaches are better for business. Those differences help us identify new opportunities and new ways of working, and they're how we "disrupt ourselves," learn, and grow.

There is an inherent problem with interviews that seek to focus on "someone's approach to the problem," which is that, unless we're very, very careful, we will bias toward people who use approaches we like and are already comfortable with. That's the main ingredient in building a monoculture, meaning a team that tends to all think alike, approach problems the same way, and so forth. It is, in other words, the opposite of diversity.

Essentially, almost anything you can think of that's really difficult to create an objective scoring rubric for is also something that can admit conscious and unconscious bias, and those are how you create and sustain a monoculture.

Think about it like this — it's not hard to give someone an analysis-level exercise that can be objectively scored. For example:

> *Take a look at this router configuration file. Do you see anything here that would concern you from a security perspective? What would you do to fix it?*

- 1 pt: Candidate identifies the insecure configuration of EIGRP.
- 1 pt: Candidate explains the risks associated with this configuration (ask the candidate to explain the risks if they don't do so on their own). *Include example answer that lists the key points.*
- 1pt: Candidate suggests enabling message authentication mode.
- 1pt: Candidate explains how message authentication mode mitigates the risks (ask the candidate to explain if they don't do so). *Include example answer that lists the key points.*

This kind of question and rubric doesn't leave a lot of room for interpretation. Now, this question also isn't telling us much about how a candidate mentally works through the problem — how they break down the configuration file in their mind, how they prioritize different areas to consider, and so forth. While "how someone works the problem" can be predictive of job success, it can also be an area where our biases come into play and lead us to support a monoculture.

It comes down to this: when you focus on observable job skills, by using an objective rubric, you will make it hardest for bias to creep in.

So how can you still include the admittedly valuable "seeing how they think" without also inadvertently creating a monoculture?

Consider "pulling back" a little. For example, make sure your interview includes analysis-level, skill-based questions that can be graded with an objective scoring rubric. Obtain the majority of your interview scores from those. Then, go ahead and include a problem where you ask the candidate to work through a problem, but don't grade them on how they work through the problem. Instead, focus on whether they work through the problem. In other words, it isn't the results that count and it isn't even the "journey" that counts. What counts is the fact that the candidate is indeed on a journey. You can see them working through the problem, even if you don't love the way they're working through the problem. For example:

- 0 pt: Candidate struggles to articulate the essential problem, even when prompted.
- 1 pt: Candidate articulates the problem, but has no suggestions for how they'd proceed next.
- 2 pt: Candidate articulates the problem, and has a suggestion on where they'd start trying to solve it.
- +1 pt: Candidate asks follow-up questions to better understand the scope/nature of the problem.
- +1 pt: Candidate asks follow-up questions related to the business logic they're seeing.

This kind of rubric can help you objectively score how a candidate engages. You're looking at whether they can restate a problem and whether they see areas that should prompt additional questions. You're not judging them on the quality of their approach, but rather on the fact that they are taking an approach. These rubrics can be quite complex, and it's important that they not place a value judgment on the candidate's exact journey. For example, the following would be poor rubric items:

- 1 pt: Candidate asks about the method signature's construction (which is incorrect).
- 1 pt: Candidate suggests refactoring function A into two more tightly-scoped functions.

Those kinds of rubric items reward the candidate for taking the same approach that the existing team values, which is what leads to a monoculture. You might rephrase that last one, for example, to something like:

- 1 pt: Candidate notes that function A is functionally overloaded.

Here, you're now valuing the candidate being able to identify a problem, rather than valuing a specific, preordained answer from them.

With all rubrics, simply ask yourself, “How much of this answer is 100% objective and could be graded the same way even by someone who doesn’t work for us, and how much of this answer simply reflects our way of doing things, even though other organizations might legitimately choose a different way?” The more of the latter you can do away with, the more open your interviewing process will be to candidates with unique perspectives and approaches.

Remember: there’s plenty of time later to coach a new hire into your way of doing things. Ideally, you’d give them the space to challenge existing assumptions and help create positive change, but the interview isn’t the place to restrict them.

Creating Accountability

The last bit — and this is where we truly create science from our interviews — is to hold yourself accountable for your goals. To measure yourself, and to do so with rigor and honesty. The exact metrics you choose will obviously depend on your specific goals, but some good metrics to consider are:

- **Passthrough Ratios:** Each stage of your interview should be passing along a reliable portion of your candidates and should have a meaningful and desirable effect on your overall hiring pipeline. For example, an interview that routinely passes along 80% of candidates is probably not a useful interview. It isn’t narrowing your pipeline much. On the other hand, an interview that passes only 20% of candidates is also probably not useful unless it’s a very late-stage interview intended to whittle down just a few remaining candidates.

- **Candidate Impact:** Ideally, each stage of your interview process should pass along the same percentages of men as women, of people of color as white people, and so forth. You should watch this metric carefully, because less-than-equal passthrough ratios may be a sign that you've designed interviews that will lead to an undesirable monoculture.

- **Candidate Sentiment:** Does it matter what candidates think? Absolutely yes. Remember, your interview process is the first time you're meeting a new colleague — hopefully, you want them to come away with a positive impression. If nothing else, word about your organization's interview process will be on the Internet. A poor overall level of candidate sentiment *will* damage your employer brand and make it difficult to attract new candidates. Some sentiment questions to consider:

    - Were the questions we asked clear and easy to understand?
    - Were you confident that you understood how you were being scored?
    - Were the questions we asked clearly related to the job role you applied for?
    - Did the questions we asked seem aligned to the job level you expected?

- **Job Performance:** This is a measurement many organizations fail to track, often because once a candidate is hired they pass out of the Talent Acquisition organization and into a production team — breaking the "link" between interview and performance. But this is the most important metric because it tells you how predictive your interviews are. Candidates who score well throughout should perform well after their first 90 days, six months, and year on the job. If you're not seeing a close correlation between interview scores and ultimate performance, then your interviews need some redesigning.

Now, there's an important caveat about all of this that keeps interviews from being truly valid science — and it's a caveat you can't do anything about. But it's important to keep in mind, because without this caveat firmly in mind you can become overconfident in your interview approach.

When you conduct an interview and make a decision about a candidate, you are implicitly stating a scientific hypothesis: This interview does a good job of identifying qualified candidates, and a good job of identifying unqualified candidates.

In science, a good, rigorous hypothesis exhibit's Karl Popper's basic scientific principle: falsifiability. Falsifiability is the assertion that for any hypothesis to have credence, it must be inherently disprovable before it can become accepted as a scientific hypothesis or theory.

- A candidate scored well, and we hired them, and they performed well on the job. This proves the positive side of the hypothesis.

- A candidate did not score well, and we hired them, and they performed poorly on the job. This is the falsifiable part of the hypothesis.

Real science has to do *both,* but obviously as a hiring organization you're never going to even consider the second one. That's where it becomes easy to get overconfident about your interviewing approach. To put it another way, you'll never know if you left qualified candidates behind, because once your interview filtered them out, you never performed additional tests to see if the interview was correct in doing so or not.

In other words, our hypothesis that our interviews do a good job is inherently unfalsifiable*,* which means it isn't truly "good science."

As stated, you're never going to be able to control for that, but you do have to keep it in mind. Never assume that just because your interview only passes on good candidates that it isn't also denying some good candidates a further opportunity. This is where monoculture comes strongly into play: a good interview — meaning, one that only passes through qualified candidates — can still create a monoculture, which shouldn't be desirable. That's why we always have to approach our interview metrics with a healthy awareness that we'll often never know if we're doing a less-than-excellent job.

Conclusion

Creating successful technical interviews isn't necessarily difficult, but it does require a clear understanding of the goal of interviews and a deliberate approach and methodology. Now that you're armed with the following framework for crafting predictive, fair, and enjoyable interviews, you're well on your way to hiring candidates who are most likely to thrive in your organization while providing a great candidate experience that boosts your employer brand:

- Scoring rubrics are the key to consistently evaluating candidates and preventing bias, but it's important to design rubrics that are specific and objective.
- To design predictive interviews, start with a Job Task Analysis and then build your interview questions around the tasks that you identify as being common and critical.
- Interviews that are free from bias are ones that have a fixed format, have consistent interview questions, are focused on the job tasks, and objectively evaluate candidates using a scoring rubric.
- Creating enjoyable interviews means having a clear job description, transparently communicating with the candidate about what to expect before the interview, interviewing with compassion, and connecting the interview questions to why it's important for the job.
- Interviews that test how a candidate approaches a problem are inherently biased, but you can prevent this by evaluating candidates on *whether* they work through the problem and not *how* they work through it.
- Measuring your interviews helps you stay accountable to your goals and indicates how predictive, fair, and enjoyable your interviews truly are.

karat^

Work With Karat to Make Your Technical Interviews More Effective

As the leader in technical interviewing, Karat has conducted over 350,000 technical interviews for companies across the world. When companies partner with Karat, they get access to Karat's innovative, equitable top-of-funnel technical assessment; 24/7, on-demand technical interviews; and the company's expertise and data-driven processes. Whether you're looking to increase your hiring yield, reach more diverse candidates, or rapidly scale your hiring, Karat can help you transform your technical interviews to meet your goals.

Request a demo today ↗