Engineering

The Anatomy of the Perfect Technical Interview from a Former Amazon VP

Neil Roseman is downright tired of hearing Silicon Valley companies say they “hire only the best and the brightest.” No matter how many times they say it, most still make decisions based on gut feel, basic credentials, GPAs, ivy league educations, flashy company names — even SAT scores. Roseman objects. As the former Technology VP for both Amazon and Zynga, he’s interviewed hundreds of people and believes every phase of the process needs to be meticulously designed to drill deep into skill sets, actual accomplishments, culture fit and leadership potential.

For the first time, in this exclusive First Round Review interview, Roseman explains exactly how he designs interview processes from top to bottom to build an effective organization, regardless of size or resources.

At the 10,000-Foot Level

Starting out, there are a few key organizing principles to acknowledge in interviewing, even before you start collecting resumes or asking for referrals. These can be used to craft a clear protocol around interviewing and hiring.

> You should come out of every interview with a clear sense of whether the person could improve the probability of your company’s success.

> Great interviewing is work. It takes time to prepare, conduct the interview and then de-brief in an effective way. If you don’t want to do the work, don’t interview.

Once you form an initial impression of someone - which usually happens within the first 60 seconds - you should spend the rest of the interview trying to invalidate that impression.

You should always take copious notes during interviews so you can make a cogent argument for or against a candidate.

In most cases, the "best and the brightest" already have jobs, so you're really just on the lookout for the best available. Plus there's no way to prove that your hiring process resulted in the right people because you can't A/B test hiring decisions.

Always strive to hire superstars but realize that not all hires need to "walk on water." This person should be better along some dimensions than a majority of the current staff and have the potential to have a long-term impact.

You want to hire people who are smart, that get stuff done and have the functional set of skills you need for the role.

The Right Way to Read Resumes & Design Questions

Every job search starts with a resume or web profile — whether it comes in as referral or through a proactive sourcing strategy. While current wisdom suggests that resumes aren't all that important compared to what a candidate actually knows, Roseman argues there's a lot to be gained through careful review.

The things that most teams have the least experience in is how to read a resume, identify the areas worth probing, and determining how to probe into them.

While screening resumes, Roseman keeps an eye out for areas where he can push candidates. “I always look for things where they have a measure of their success, especially if they make comparisons or use percentages.” For example, "grew revenue by 50% or decreased downtime by 30%."

This is how he develops most of his questions. “You want to find out what somebody really did, as opposed to just being an observer or a participant,” Roseman says. “Even at the greatest companies, there’s a gap between those who get the most stuff done and those who don’t get much done. You need to try and figure that out during an interview.”

In many cases, this can be a litmus test for how clearly they think about themselves and their role as well. “They might think it sounds good to say 'I improved system availability by 50%', but if we’re hiring someone for a system engineering role, I need to know they actually did that. In most cases with high level statements like this — which appear on resumes all the time — the person actually hasn’t done it or was just a participant, and understands very little. They won’t have a very clear answer.” The good candidates will be able to explain and backup their claims no matter how far you drill down.

As an example, here is a resume snippet Roseman recently received, “Led a team of 3 engineers in creating massively scalable storage infrastructure used by various Google products.” Roseman took note of this and during the interview, had the candidate whiteboard the infrastructure, and then highlight his main contribution, where Roseman could dig down and see if he really knew his stuff.

To put this another way, great interview questions focus on specific examples of the candidate’s unique contributions, actions, decisions and impact. Ideally, you want to:

Probe: give me an example…

Dig: who, what, where, when, why and how on every accomplishment or project

> **Differentiate:** we vs. I, good vs. great, exposure vs. expertise, participant vs. owner/leader, 20 yard line vs. 80 yard line

Roseman goes on to share, "I look for past projects and accomplishments that seem to have enough weight and depth that I can apply STAR questions — STAR stands for situation, task, actions and results." Roseman subscribes heavily to an approach called Behavioral Interviewing, in which STAR questions are a staple. They include:

> What was the background of what you were working on?

> What tasks were you given?

> What actions did you take?

> What results did you measure?

In addition, when reading resumes prior to interviews, it's a good idea to suss out whether the projects or products listed were significant to the company, even if they didn't succeed. Roseman uses Microsoft as an apt example. "It's such a big company, you want the employees that worked on important teams," he says. "Companies like Microsoft move people around to get the best people on the highest profile products. Whether it's Google, Facebook, Apple, Amazon…you start to learn. You can try to read the company."

Technical Questions & the Interview

All too often Roseman hears hiring managers begin an interview by asking candidates to walk through their resume. "That is not a highly valuable question to ask unless you've already decided you're not hiring this person," he says. You can set a candidate at ease without this warmup softball. Roseman recommends introducing yourself so they know where you're coming from, and clearly stating what his goal of the interview is. “I ask the person to introduce themselves and give me a couple of minutes about what they’re interested in and most excited about,” he says. “This way we can relax, set the stage for the conversation and make sure we’re both comfortable with each other. Calm any nerves that the candidate might have.”

After a quick intro, Roseman begins an interview with the most hands-on technical questions. “For an engineering position, the reason most people don’t get hired is because they simply don’t have the skills — they don’t pass the technical bar,” he says. “I want to validate that first.”

Here, he pays attention to the candidate’s area of focus. If it’s coding, he asks them a coding question based on their experience. The important thing, he says, is to never ask a question that you haven’t asked ever before. “You want to have tried the question out on other people in a non-interview situation so you can understand how to guide someone through the question,” he says.

When asking a question, you have to know what to consider a very good, good, poor or very poor answer and why.

To Roseman, this is the most important tenet of crafting smart interview questions. “One of the things that really pisses me off is people asking questions that they don’t even have a good handle on themselves,” he says. “They don’t have a good distinction between a great answer and a crappy answer. They decide to try out a new question they heard from somebody or make something up themselves."

This is especially important today, when there are infinite resources on the web for interview questions. Candidates can look up past interview experiences on Glassdoor or Quora and have an inside track. Even so, Roseman says it’s okay to borrow questions from these resources as long as you make them your own. “Your whole team should sit down and discuss what’s a good coding question, what’s a good answer and why you should use this one.”

For example, Roseman often asks candidates to palindrome the words in a string, then digs a little deeper by asking them to imagine they’re on a limited memory machine and they have to do this in place. “You can always push them so they might have gotten one level of the answer from the web. If they’ve gotten all the levels then they’re pretty darn smart.” Again, the key is to start high-level and then begin to dig deeper. Even if a candidate has seen the question before, if you dig, you can learn something.

To create one-of-a-kind questions, he suggests asking candidates how they would solve the kinds of problems your company actually faces. “While at Amazon, I often asked a design question that’s based on the recommendation system — the ‘people who bought also bought feature.’ It’s always best if I cast this in terms of a product that people know. That way you see if they’re both product focused and solution focused.”

Roseman especially likes to press engineering candidates on product design. Great engineers should not simply be order takers, but actively part of product development. Design questions also allow you to better understand how someone thinks. To get to the heart of this ability, Roseman drills candidates on past products they have been involved in, and may ask them to write a small portfolio management program. Roseman also often asks the candidate to walk through a more generic design question like “design an ATM/Elevator for blind people” or something more technical, depending on the role.

“If I have enough time, I give them the laptop and have them write a small command line app, or I have them design it and or write it on the whiteboard where they have to ask me questions about what I expect the product to do,” he says. “We want employees to ask questions. I want co-workers who ask questions and don’t just sit in the corner and wait to be given orders.”

Leading with technical questions has one caveat: answers can run really long. This is where Roseman says it is critical to keep your eyes on the time and rope people in if they start to ramble. It’s easy to eat up 45 minutes of a 60-minute interview on a particularly complex query.

“I might only be able to ask one ‘write code on a whiteboard’ question because I want to move on to a more design or product-focused question,” he says. “Then I also want to get to the soft skills, company values and culture fit." This is also the part of the interview where Roseman asks his pre-engineered questions based on the resume review that were not covered in the technical portion of the interview.

When it comes to soft skills and culture fit, Roseman is a big fan of one question — he asks everyone, no matter the position: Do you consider yourself lucky?

"If you look at what you've done, would you put yourself in the camp of people who say they've been lucky in their career?" Roseman explains. "I find a lot of people who will say I would have gotten that promotion but my manager cancelled my product, or they find other reasons for failure. Those are the ones who say they don't think of themselves as lucky. I'm looking for the people who embody the phrase 'fortune favors the prepared.' It's the willingness to be ready and take advantage of every opportunity that presents itself. At a startup, this is particularly valuable."

Roseman often follows this up by asking what he calls his "three adjectives" question. Many interviewers ask candidates for their strengths and weaknesses, but Roseman puts a different spin on it. "I will ask a candidate to think of people who they worked with, professors, fellow students, managers, etc. and imagine that I am going to go to all these people and ask them for their top three adjectives to describe the candidate. Often, it causes people to think a little bit differently about themselves. They won't just say their worst feature is working too hard." Once a candidate shares their three adjectives, Roseman follows up and asks for examples — so if someone says, "creative," his standard follow up will be "what are some examples of when you were creative."

Even if you know a candidate is a no-go 15 minutes into an interaction, it's important to get through all these phases of the interview. "You want to do close to a full interview because it's a small world out there, and even if someone doesn't get hired, my theory is it's a good thing if the person believes they just had a great interview, even if they don't get the offer."

At the other end of the spectrum, if someone is a clear winner, selling them on the position at the end of the session is critical. "You better do a good job both answering their questions and communicating your enthusiasm about the place and the opportunity," says Roseman. "I've been very clear with the people on my teams that even if they don't have an excitable personality, they have to be very positive about the company. If you can't be, then don't interview people."

In the actual interview, Roseman tries to anticipate what concerns a candidate might have, or what might convince them not to accept the job. "Sometimes, since I'm old now, I can get away with asking 'Have you thought about where you want to be in two, three, five years?'" he says. "I want to make sure I understand potential sticking points and what's driving the candidate."

The Hiring Team

The employees you hire will only be as good as the hiring team you assemble. Too many companies do not invest time in training current employees to interview their prospective peers. But according to Roseman, this is critical. "I've required anybody that even does a phone screen to first do it with other people who are more experienced, just sitting in."

Once you have a qualified squad of interviewers, it's time to take a closer look at their decision-making processes. To ensure that everyone can articulate their thinking on a candidate, Roseman prefers everyone to take exhaustive notes on their conversations. After everyone on the hiring team has met the candidate, the group deliberates in person. Roseman explains what happens when everyone gets in the room, "What we do is put our votes in and then list in detail the questions we asked and the candidate's answers. If someone is not inclined to hire them because of a pure technical question, then they should list the exact question they asked and the candidates code."

This feedback can't be too long or drawn out, according to Roseman. Every decision is critical, and critical decisions require careful deliberation. He tells his hiring teams two things: 1) if they can't provide comprehensive feedback, then they've wasted their own time, the company's time and the candidate's time; and 2) "If you get to the end of an interview and all you can say is 'Yeah, I kind of liked them, I think they'd be good,'" then you've also wasted everyone's time. You have failed to do your job, and you should either learn how to do it or please just tell me and you won't have to interview people anymore. I don't do pressure interviewing of candidates, but I do pressure the people I work with to do good interviews."

Being a quality interviewer doesn't have to be a selfless act either, Roseman says. In fact, it can reflect very well on existing employees. "It can be a big deal to show that you're someone who should get more responsibility. It shows you're thinking, and that's what we're trying to find."

Despite his team-oriented approach to hiring decisions, Roseman is wary of hiring managers being too involved in the process. Placing emphasis on the opinions of people who will work with the new employee day in and day out is critical. He says, "no matter what, you never want to let a hiring manager override the group's decisions. They have to be able to convince everyone, and not by fear."

Given all the talk about sky-high Silicon Valley standards, Roseman acknowledges it's not rare for hiring teams to pass on candidates because they don't display superhuman skills. But not everyone should be required to walk on water in an interview. To keep this tendency front-of-mind, he's set a more explicit bar: "The expectation is that the people you hire are better than when you were hired. So that in fact if you left and came back, you might not be hired again in that position. You want to improve your overall bar up with each hire. Another way to put it is every new hire should be better than your average current team member."

Roseman uses this same logic to encourage quality referrals from current employees. "You better be good with referrals, and it's always nice to give referral bonuses," he says. "It's a lot cheaper than any other network you're going to use to hire people, and it's the best way to get some of that already hired talent that has some experience."

Here's a distilled list of Roseman's Rules:

Don't forget to introduce yourself to help work out everyone's nerves.

"Tell me about your background" is not a useful question for a tech interview.

Pick specifics out of a resume to determine what the candidate actually did. Remember, you want people who get stuff done. Period.

Probe when you see a resume with a long list of skills. Separate the truth from filler.

Don't "try out" new questions on candidates. Know what a good answer sounds like.

Make sure you have them write code! This is too often skipped.

Dig into algorithms, data structures, code organization, simplicity.

Use some questions that are vague and open-ended. See if they ask you questions to find out more.

Ask a design question. See how people think about a bigger picture problem.

Create core competences for your company. Make sure candidates measure up well.

Make it tough but fun. Good developers want to know they're talking to smart folks.