


A GUIDE TO NON-TECHNICAL MANAGEMENT SKILLS:
**Everything We've Learned About Hiring,
Creating A Career Ladder,
and Managing Distributed Teams**



**BASED ON WORK AND WRITING BY:
LENA REINHARD, JUSTIN COWPERTHWAIT, MAREK NOWAK, AND HANNAH HENDERSON**



At CircleCI, we believe iterative processes contribute to quality work. Nowhere is this more apparent than in our hiring. We've learned a lot in the past few years (and are still learning more). Based on our experience, we've gained new insights about the ideal breakdown of roles and responsibilities on a high-functioning engineering team, and moved to make a clearer separation between technical and non-technical roles. Along the way we've learned a few things about how to make sure everyone is working on what motivates them most. For example, we've learned that no matter how closely two roles are linked— like engineers and engineering managers—how we recruit and what we expect must be tailored to each role's specific responsibilities and place within our company.

This ebook will walk you through our hiring philosophy and processes, but it's really a compilation of the topics we find to be most important to any engineering organization's success. We share what we've learned about critical areas such as hiring, role structure, career growth, communication, and the types of day-to-day mentorship and support that enable engineers to thrive.

We'll take a deep dive into the following topics to help you think in a more focused way about your own organization and how you can support your team's success:

- 4** Why Hiring Managers Is Different Than Hiring Engineers
- 12** Hiring for — and nurturing — the right competencies
- 18** Establishing practices that help your engineering team thrive



Why Hiring Managers Is Different Than Hiring Engineers

Building a distributed team is both art and science. On the science side, you've spent hours (probably years) selecting things like programming languages, architectures, and tools you'll need to create your product. But assembling a group of people with complementary skills, and building them into a thriving team is another thing entirely.

Good managers are essential in this project. They're the glue that holds everyone together, overseeing initiatives, making sure everyone feels appropriately challenged, and facilitating collaboration across teams and time zones. That's why having the right hiring approach matters so deeply to your company's success. To produce great results, you need great people — and great leaders to guide them.

When hiring a new manager, what do you prioritize — management skills or technical ability? If you're like us circa early 2018, you emphasize the latter. After all, an engineering manager should be able to do everything his or her team does. Right?

Well, yes and no. While some engineering ability is important, managing is an entirely different skill set. We found that applying the same criteria to our engineering managers as we did our engineers actually did everyone a disservice. Engineers don't need managers who can code as well as they can. They need managers who can lead, coach, mentor, and strategize.

We want CircleCI to be a place where all engineers can learn and grow, and where they can be supported no matter what their ability level when they join us, or how they want to shape their careers. To support this goal, we use an [engineering competency matrix](#), a framework that outlines expectations and growth paths for engineers. This matrix is a tremendously useful tool for us: it informs the structure of our job descriptions and our interview processes. It also helps us set expectations with our engineers, and it serves as a basis for goal setting as well as conversations about learning and development.

Importantly, the matrix provides a jumping-off point for having more objective performance conversations that are less susceptible to the biases and skills of the engineer's manager. Overall, it clarifies the vision of our organization and helps us maintain consistency throughout all stages of hiring and professional development.

Time for a reboot

We developed our first competency matrix three years ago. Then, in early 2018, we realized it was time to refresh and rebuild our matrix to better reflect how our engineering org had matured and what we've learned along the way.

Revising our engineering matrix presented an opportunity to reconsider on our approach to hiring engineering managers as well. At the time, our job postings for engineering managers were heavily focused on alignment between their technical backgrounds and the technical work of their teams.

We typically included a coding problem because we felt it was important to find out if a candidate's approach and priorities aligned with those of the team they would be managing.

Unfortunately, we realized that we frequently got to the onsite stage, and only found out during pairing that the candidate's people management skills didn't match our expectations – far too late to find this out.

This shone a light onto a bigger issue: it made us realize that our attempt to align all our hiring processes had led us to focusing on, and optimizing for, the wrong skill set. There was a time when we thought having very technical managers would be a good thing. We were going for an adaptation of the Spotify model, where we'd have delivery teams and managers across different departments aligned by discipline. As our take on this model evolved to align with our needs as a distributed engineering organization, we realized we wanted to distribute leadership more.

We introduced additional roles to support our teams in different areas: team leads who were responsible for their team's delivery, and tech leads who were responsible for facilitating technical work. With our new engineering competency matrix, we also codified our expectation of leadership from all engineers, across all levels. The early successes we witnessed from these changes encouraged us to continue evolving our understanding of what makes a good engineering manager.

The ideal manager

Engineering managers at CircleCI are now dedicated to people management: focused on the development of a set of engineers, tech leads, and team leads. They hold regular 1:1s and career growth conversations with the engineers who report to them. They're also responsible for goal setting, feedback, coaching, and mentoring.

Additionally, they work across a set of teams to ensure team health, knowledge sharing, business value delivery, and alignment. This means that our engineers have managers who have great interest and investment in their personal and professional growth, and teams have someone to coach them through the product delivery process.

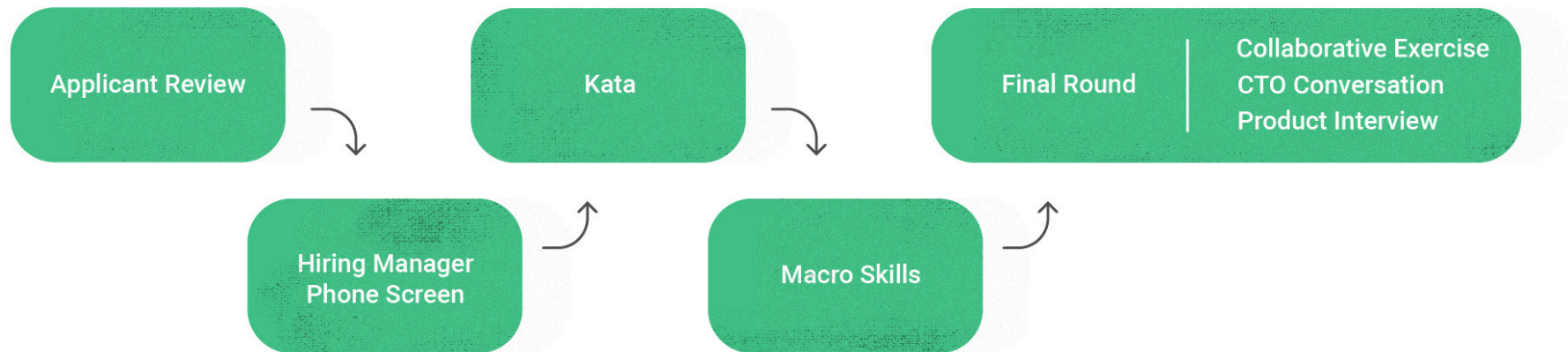
Whatever your product or service, hiring for non-technical roles is as important as recruiting for any other position. Managers set the tone in the company, and they have the opportunity to really drive growth and innovation. A great manager understands how to work with different personalities and work styles and identify each person's strengths.

More importantly, they know how to nurture and draw out those strengths for the good of the organization. When engineers are continually being challenged and offered growth opportunities, their career satisfaction and output soar. And all of that begins with your non-technical leaders.

Revamping our process with refocused values

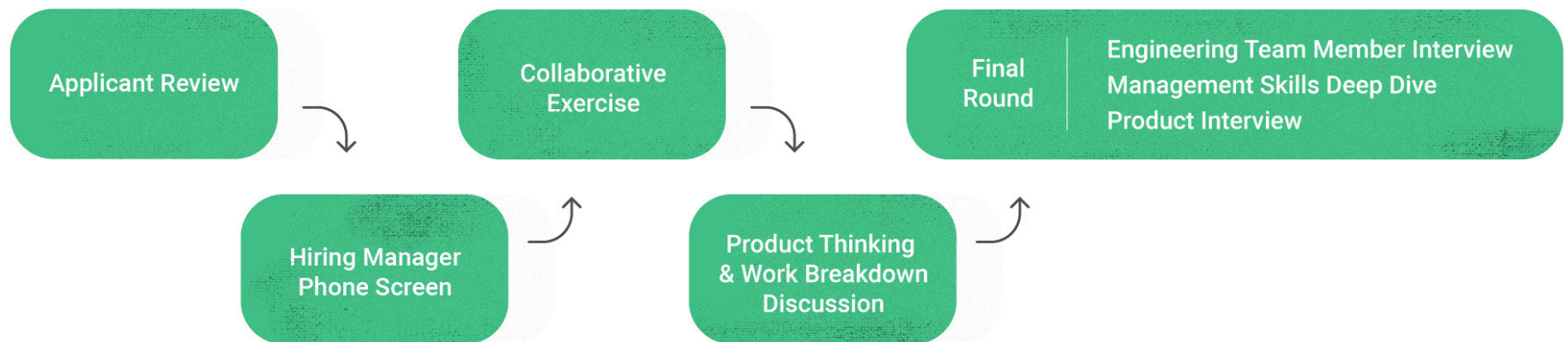
As a result of learning what really matters to us in engineering management, we adapted our open management positions to roles focused on both people and business value delivery. We also changed our hiring process and the design of each interview stage to explore these areas earlier in the interview process. At the same time, we moved to a more structured, behavior-based interview for all applicants.

To give you a sense of how we've shifted our processes, here's a look at what our hiring flow looked like before:



**Kata: technical exercise or problem to test engineering skills*

And this is what our hiring process for engineering managers looks like now, after we delved into how we could more effectively recruit people who would support our team's specific needs:



Note: We're always looking for ways to improve our processes, so these stages are subject to change


1. **Hiring manager phone screen:** The hiring manager for this role talks with the candidate about their management experience across different areas.
2. **Collaborative exercise:** This is a collaborative interview, in which the candidate is paired with one of our current engineering managers. Together, they work through two exercises based on challenges our organization has faced in the past and which could occur again today. As with our technical pairing exercises, we want to make sure to reflect the actual work we're doing. Talking through challenges and developing solutions together as an engineering management team is very important to our daily work. This interview helps us understand a bit more about the candidate's approach to organizational challenges and how they'd collaborate with a peer.
3. **Product thinking and work breakdown discussion:** In this interview, the candidate and another member of our engineering management team discuss questions related to their ability to understand work and delivery. They also discuss different elements from a customer value perspective.
4. **Engineering team member interview:** This is a conversation with a senior engineer with whom the candidate would be working. Together, they discuss collaboration challenges that our teams have faced in the past. We look for candidates' ability to mentor and add value to technical discussions while understanding their own limitations, thereby showing whether they can support a technical decision without acting as a decision-maker.

5. **Management skills deep dive:** This conversation is usually with a senior member of the engineering management team, and it involves a deeper discussion of important leadership skills and the candidate's management experience.
6. **Product interview:** This last interview is with a member of our product team, and focuses on the candidate's perspective on product, process, and building healthy engineering teams.

Changing our process this way has helped us screen for the right values and skills and determine which candidates align with them early on. Ultimately, it has enabled us to hire great managers who care, whose values align with ours as a company, and who can help us take our engineering organization through its next growth phases.

Distinguishing between management skills and technical skills can be a game-changer for your organization. Being an excellent engineer does not automatically qualify someone to be a manager. Leaders must be able to connect with a number of different personalities and understand how to bring unique individuals together to achieve shared goals.

The mindset you need to succeed as an engineer does not always overlap with management, and vice versa. That's why it's important to tease out the differences and recruit people who deeply understand engineering but who know even more about being superior managers.



Hiring for – and nurturing – the right competencies

Every engineer deserves a clear growth path so they can understand, plan, and execute on meaningful career development. Providing a framework for this growth (we call ours a competency matrix; it's also known as a career or professional development ladder) is important work. We believe it's the responsibility of any organization that wants to nurture and grow its employees to create a detailed workflow for helping people flourish. Back at the beginning of 2018, we had 32 developers and a plan to double that number throughout the year. We already had a competency matrix, but by this point it was woefully outdated. It focused on our more junior levels, maxing out at a level which some developers had already reached. We also realized the matrix was also misaligned with the skills our organization had grown to value. This meant that in practice, we often ignored it. Our competency matrix needed a re-design.

Building a new competency matrix was a learning process, and a lengthy one, taking about eight months to complete. Along the way we not only clarified and codified our deeply held values, but also discovered what the key steps to building a career ladder are (and which ones will waste a lot of time).

While every matrix is different, and will reflect the values of the organization that wrote it, the process of producing it is fairly consistent. Here, we'll go over our tips for building your own matrix and getting your team onboard.

STEP 1

Assign ownership.

Whether you are designing a new competency matrix or fine-tuning your existing process, designate a point person for the duration of the project. Building a competency matrix should not be a side project or afterthought. It's integral to developing a strong team, so make this a priority.

STEP 2

Get aligned.

Until all stakeholders agree on what your goals are, every attempt at implementation will stall. A competency matrix is a powerful tool to set a cultural tone and direction, so in designing your matrix, you'll have many impactful choices to make. Each one has consequences, and you'll only get through them if the team is aligned. One of the questions that came up repeatedly for us was: is this a codification of the status quo or is this aspirational? (We were halfway through the process when we finally explicitly agreed that it was a blend of both.)

Another key point of agreement is: who is going to be affected by this? In our case, this question hinged on whether or not our new matrix would affect our Site Reliability Engineers (we decided it would). Maybe you are building a matrix for your whole company, or maybe you are building a matrix for your frontend development team. The breadth of

roles affected will greatly change the competencies you choose to codify and the level of abstraction you need to use. So ask these questions, and get explicit agreement.

Finally, we needed to agree on the goals of the matrix. We decided the primary goal was performance evaluation and growth planning with individual engineers at CircleCI. Secondly, we wanted to use it to influence our hiring process and communicate externally what being an engineer means at various levels. Knowing the potential uses, and the priority of those, guided certain decisions.

STEP 3

Guide by values.

This is the part where you sit down and debate “what matters to us?” We had some help from our excellent Head of HR, David Mann, who came in with note cards detailing roughly 100 behavior traits that are valuable to have as a professional. These were not engineering-specific, and they ranged from Communication to Political Awareness. We also utilized other publicized competency matrices to seed ideas of what could be in the running.

The best part of this step for us was that we enjoyed coming up with our own values as a team. The one that sticks out most in my mind is Economic Thinking, something the management team continually discussed as being one of the key skills that differentiates good developers from great ones. We didn’t borrow that competency from any HR cliff notes or other matrices we consulted, but from the get-go this was a key competency we knew would be in the final cut.

STEP 4

Define levels.

Before you start to actually fill out the content, it's good to define the x-axis of the matrix: the growth in terms of title and responsibility. At CircleCI, we already had the existing set of titles we wanted to use: E1 / Associate Software Engineer to E6 / Principal Software Engineer. Then we explicitly agreed upon generalized responsibilities for each title.

We broke it down into two categories. E1, E2, and E3 would focus on mastering the skill of software engineering and becoming a highly effective IC. E4, E5, and E6 would focus on utilizing those skills to scale impact and create leverage across larger and larger groups of people. Once we had this high-level guidance, we broke it down further, assigning scope to each of those execution levels.

STEP 5

Create content.

This step is definitely the most laborious. However, if you have followed steps 1-4, it should make this step much easier for you than it was for us. Once the framework was in place, filling out the content became much more straightforward.

Given the meatiness of this step, let's break it down further into sub steps. Following this sequencing should minimize any wasted work.

- Define a single level for all competencies.
- Look for opportunities to merge competencies.
- Wordsmith.

STEP 5.1

Define a single level for all competencies.

Rather than trying to wordsmith the nuanced differences between what it means to exhibit Delivering Feedback as an E2 vs an E3, identifying what you mean with each competency by defining a single level is a very enlightening and aligning process.

STEP 5.2

Look for opportunities to merge competencies.

An amazing thing will happen while you are defining that single level for every competency: you'll realize some of the values point at the same behaviors. This is an opportunity to merge!

After we defined single levels, we noticed that we had two competencies that were eerily similar: Self Starting and Delivery Accountability. Although these competencies had appeared to be different initially, the behaviors we had codified were similar enough that we merged them under the heading of Reliability and Delivery Accountability.

Early on, we made the explicit agreement that we wanted our matrix to be a collectively exhaustive definition of what growth looked like for a developer at CircleCI while also being as simple as possible. Ultimately, we ended up with 27 competencies. However, when we started Step 5.1 we had almost 50. Defining the behaviors for each showed us where we had opportunities to consolidate.

STEP 5.3

Fill out the rest of the levels & wordsmith.

During this step would be the best time to start wordsmithing. While you are in the throes of defining the nitty gritty details of what it means to scale Economic Thinking to one team vs. many, for instance, is the best time to be critical about whether or not the language you are using properly conveys the intent. I would strongly advise against wordsmithing before this, but this would be a great time to revisit the content you've written in your first pass.

STEP 6

Involve the team.

Once we completed the first pass of our competency matrix, we orchestrated a feedback session in the format of a focus group, made up of individual contributors of all levels, managers, and HR. There are many ways to gather feedback, such as having a diverse sampling of individual contributors complete mock self evaluations with their manager. Regardless of how you do it, the important thing is ensuring you gather feedback from the people this will affect.

The feedback we received was extremely valuable for two reasons. It surfaced good questions, which led us to tweak the matrix so it conveyed what we intended. But the process also ensured we had codified the values of the organization, not just those of a few managers, and that the value progression was representative of how our engineers saw their careers growing. This step was extremely important in generating confidence that what we had developed represented what we wanted, and that it would be well received.

STEP 7

Ship it!

The best step of all - ship it! You have a finished product that has been wordsmithed, stress-tested, and has buy-in from your organization. However, even after this thorough process, it will never be perfect. As people start to use and test what you have created they will find slight inconsistencies and opportunities for improvement. Be ready for feedback now and on an ongoing basis.

There are two mechanisms you should put in place to that end: a way to gather feedback and an agreed-upon timeline for addressing it. Constantly iterating on your competency matrix would be a bad idea, because it means constantly moving the goal posts for career development at your organization. However, pretending that it is perfect and will never change is foolhardy and unrealistic. Ensure your team knows you're listening to their feedback and have a plan for addressing it.

Establishing Intentional Practices to Help Your Engineering Team Thrive

Congrats! You've assembled a competency matrix and a stellar engineering team. But your work isn't done. Your engineers can only succeed if they're working in a supportive environment that allows them to learn, succeed, and grow professionally.

To provide such an environment, you need to think both philosophically and logistically about your company's culture. We find it helpful to think about this from three key perspectives:

Why we chose to be a distributed team

CircleCI's team is distributed around the world, and we've spent a lot of time fine-tuning our approach to this structure. Being a remote-first organization provides our engineers a degree of flexibility they wouldn't have if we required everyone to be in a central location, and we've seen that autonomy lend itself to greater team satisfaction.

The fact is, you get the best from your employees when they feel supported to balance their personal and career priorities.

There are several ways to do this, but we've opted for distributed teams as our chosen model to empower people to work on their own terms — where they want, when they want. This allows them to structure their lives in a way that allows for maximum fulfillment in their careers, relationships, and other areas that matter to them.

Running a successful distributed teams requires forethought and cooperation from across the team, to create and enforce practices that keep everyone on the same page, and prevent common problems that pop up in distributed organizations.

An example of a challenge we faced with our distributed team model was how to avoid pairs of engineers self-segregating along time zones. We didn't want our engineers in Europe to pair-program exclusively with one another or those in the U.S. to do the same. So, we encouraged "ping-pong pairing," an asynchronous version of pair-programming, in which engineers in different parts of the world could get into a rhythm of programming, commenting, and handing off work-in-progress to one another.

“ I am thankful for the team's ability to continue progressing on work while some of us are asleep. Our team's timezone distribution means I have on-call cover for my evenings and early mornings.”

**DAN CARLEY,
SITE RELIABILITY ENGINEER (SRE)**

“ Flexible hours mean that I can spend time with my family that I would otherwise spend commuting, and that my kids would be in daycare for longer.”

**MARC O'MORAIN,
CIRCLECI SENIOR STAFF ENGINEER**

This allowed for the knowledge sharing that makes pair programming so valuable while also encouraging team members around the globe to collaborate with one another. If we insisted that pairs work in tandem on the same schedules, the overall team would have felt more fractured. We had to create a system that allowed for the type of collaborative environment we wanted.

Another area in which we had to be quite deliberate was communication. When you have employees in different time zones, it's easy for information to fall through the cracks. So, we adopted a bias toward overcommunication. Even if we assume our engineers understood why we made a particular decision, we make it explicit anyway, and we encourage them to do the same.

We've also implemented standard behaviors so it's easy to get in touch with one another and to flag important messages and chats. For instance, we use Slack for ongoing team communication, and we realized that often a decision that was made in one discussion thread might not be read by everyone.

“ I enjoy that we are all able to have flexible lives! Our team is passionate about our work and our mission, and I love seeing what everyone is doing outside of work (traveling, working out, cute babies, etc.). I feel extremely supported and cared for by everyone.”

JACQUE GARCIA,
SOFTWARE ENGINEER

Therefore, we created a policy to tag the entire channel when there's something the whole team needs to know.

We encourage everyone to use certain emojis to catch people's attention as well. A question mark emoji calls out a question so someone can respond quickly; an exclamation mark emoji signals a warning; a megaphone highlights a PSA; and a red arrow says "Hey, I'm signing off for now." Since Slack is our go-to communication platform, a lot of messages get exchanged there every day. The emojis help ensure that we stay on the same page, even when there are lots of threads being created and issues being discussed.

Another, less measurable, benefit of Slack is that its informal nature encourages casual personal conversations among team members as well. Our engineers work hard, and they all have unique hobbies and interests outside of work, too. Our team enjoy sharing anecdotes about their lives, along with photos of their pets and families. Despite the fact that we all work in different places, we still get to enjoy a sense of camaraderie and time around

the virtual water cooler. And that's what we want, a team of people who bring their whole selves to work, wherever they are working from.

These are examples of how we work, but we know it will be different for your organization. Whether you choose to work as distributed teams as well depends on your company culture and the structure that works best for your organization. But however you work, you want your managers and your engineers to speak the same language—even if that language is emoji.

Dynamic mentorship and growth opportunities

Opportunities for growth play a critical role in employee engagement and success, and they go beyond opportunities provided directly by management. While managers can nurture growth by recommending team members for promotion or offering to mentor promising engineers, we believe empowering engineers to help each other makes for an especially dynamic work environment.

Here are some strategies that have worked for us:

Model empowering behaviors. Managers can model empowering behaviors such as giving positive feedback, or inviting new team members to be part of projects so they can learn. Something as simple as an engineer receiving a thumbs up emoji on a pull request from a more experienced team member, or a group Slack message highlighting someone's efforts, fosters an environment in which good work is recognized by everyone in the organization. Regularly encouraging and publicly appreciating people's successes validates their

efforts and sets the tone for everyone else's interactions.

Set a standard for going above and beyond to help people learn. One of our engineering team leads shared that when she was new on the team, she learned a great deal from her more experienced colleagues who created opportunities for her to grow her skills. They walked her through problem-solving deep dives, and when she asked questions, they provided not only responses but context as well, referencing codebase histories and the evolution of different programming languages.

By not simply giving short responses to the immediate problem and instead sharing their knowledge with her, they helped her build her own knowledge base on which she can draw to strengthen her own programming and problem-solving skills. They also set a standard for how she will likely lead and respond to other new hires. Now that she is a team lead, she can refer back to what helped her to feel quickly like a valuable part of her team.

Don't wait for official promotion opportunities to let people advance. Encourage technical mentorship among colleagues. Engineers often enjoy and take pride in sharing what they know and collaborating to write better code. Managers should encourage team members to share tips and technical knowledge regularly, not just when they're asked a question. They can also let engineers know that it's OK to take the initiative on offering help to new hires. The more collaboration and support your managers can nurture, the stronger a team you'll have company-wide. When managers and colleagues create learning opportunities for new hires, you give people a chance to grow and become more integral to the organization even outside of official promotion openings.

Tips for managing distributed teams

CircleCI has been a distributed organization since the very beginning. As we've grown, we've picked up quite a few lessons along the way (for example, on [communicating asynchronously](#), [pairing remotely](#), and [distributed onboarding](#)). For those considering joining our team, as well as those growing their own distributed teams, we'd like to share some of our favorite distributed team tips:

- **Use asynchronous communication where possible when working across timezones, and make the best use of any synchronous communication time you have.** Establish a go-to communication platform, such as Slack. Standardizing communication lets everyone know exactly where they can go to get in touch with each other and catch up on what happened during their off-hours. If your work hours overlap for a period each day, make a point to prioritize conversations or issues that require cross time-zone collaboration. That reduces unnecessary

downtime and it gives people a chance to touch base even if their schedules don't align perfectly.

- **Converse in common team channels rather than Direct Messages.** DMs can be useful if you need to get in the weeds about something with one other person, but generally speaking, common team channels work best for collaboration. When you work through issues in a common channel, everyone has a chance to see what's going on and to weigh in on challenges. If two programmers only hash out issues in a private chat, they don't get to benefit from the knowledge of their other colleagues, and likewise no one else can learn from their discussion. Common channels also nurture conversation and camaraderie among people who are located around the world.
- **Record meetings.** Ideally, you'll be able to hold semi-regular meetings that everyone can attend, regardless of where they're working. But that won't always be the case, so be sure to record meetings so that everyone can watch them and

know what was discussed. That's a great way to make sure everyone is on the same page. Someone who missed the live meeting but catches the recording can then come into Slack to give their thoughts or ask any questions about the issues that were raised.

- **Say hello and goodbye to your team each day to indicate when you're in/out.** Greeting people sends a friendly message and lets them know when you're available for questions and conversation. Making a point to sign on and off also formalizes your work hours, which can be helpful for you and your team.
- **Make your working hours visible on your calendar, so it's clear when you're available and working.** When you're working on a distributed team, it's really important that people know when they can expect to reach you. Not only does it set clear expectations for when you're around to answer questions and jump into a project, it also prevents you from being pinged at all hours when you're not on the clock.

- **Leave summaries in Slack threads after team discussions or huddles.** It can be tough to remember everything that was decided on or announced during discussions and huddles. Posting summaries reiterates key points and solidifies the information in people's minds. It also gives them something to refer back to if they need a refresh on the meeting.
- **Pair often.** As we discussed earlier, pair programming is a great way to share knowledge. Encourage your engineers to collaborate often so they can challenge one another and learn from each other's skills.

The best methods for working on a distributed team, whether at CircleCI or elsewhere, will vary by the team, the individuals on it, and the work you're trying to do. We encourage you to communicate with your teammates, test things out together, and overall be patient. Building a happy and successful team, especially a geographically distributed one, takes time, but having processes that work is worth the investment.



Conclusion

Great managers are the backbones of great engineering orgs. While technical companies tend to emphasize technical skills—and they should, especially in their earliest stages—non-technical attributes are equally important when building and scaling your management teams. The right managers will coach and lead your engineers to deliver better work, as well as help each other learn and grow. They'll bring out the best in your employees, which means you'll be producing a higher quality product. Most importantly, you'll be leading a company of people who are committed to supporting one another and fostering a positive and productive work environment.