40: Hire Less and Hire Later

Add slow to go fast

There's no need to get big early — or later. Even if you have access to 100 of the very best people, it's still a bad idea to try and hire them all at once. There's no way that you can immediately assimilate that many people into a coherent culture. You'll have training headaches, personality clashes, communication lapses, people going in different directions, and more.

So don't hire. Really. Don't hire people. Look for another way. Is the work that's burdening you really necessary? What if you just don't do it? Can you solve the problem with a slice of software or a change of practice instead?

Whenever Jack Welch, former CEO of GE, used to fire someone, he didn't immediately hire a replacement. He wanted to see how long he could get along without that person and that position. We're certainly not advocating firing people to test this theory, but we do think Jack is on to something: You don't need as many people as you think.

If there's no other way, then consider a hire. But you should know exactly who to get, how to introduce them to the work, and the exact pain you expect them to relieve.

Brooks' law

Adding people to a late software project makes it later.

—Fred Brooks

Programming and Mozart's Requiem

> *A single good programmer working on a single task has no coordination or communication overhead. Five programmers working on the same task must coordinate and communicate. That takes a lot of time... The real trouble with using a lot of mediocre programmers instead of a couple of good ones is that no matter how long they work, they never produce something as good as what the great programmers can produce. Five Antonio Salieris won't produce Mozart's Requiem. Ever. Not if they work for 100 years.*

—Joel Spolsky, software developer, Fog Creek Software (from Hitting the High Notes)

41: Kick the Tires

Work with prospective employees on a test-basis first

It's one thing to look at a portfolio, résumé, code example, or previous work. It's another thing to actually work with someone. Whenever possible, take potential new team members out for a "test drive."

Before we hire anyone we give them a small project to chew on first. We see how they handle the project, how they communicate, how they work, etc. Working with someone as they design or code a few screens will give you a ton of insight. You'll learn pretty quickly whether or not the right vibe is there.

Scheduling can be tough for this sort of thing but even if it's for just 20 or 40 hours, it's better than nothing. If it's a good or bad fit, it will be obvious. And if not, both sides save themselves a lot of trouble and risk by testing out the situation first.

Start small

> *Try a small test assignment to start. Don't leap in with all of your work at once. Give your new [virtual assistant] a test project or two to work on and see how the chemistry develops. In the beginning, it's too easy to gloss over potential problems with rose-colored glasses. Make it clear this is a test run.*

—Suzanne Falter-Barns, author/creativity expert (from How To Find And Keep The Perfect VA)

42: Actions, Not Words

Judge potential tech hires on open source contributions

The typical method of hiring for technical positions — based on degrees, resumés, etc. — is silly in a lot of ways. Does it really matter where someone's degree is from or their GPA? Can you really trust a résumé or a reference?

Open source is a gift to those who need to hire technical people. With open source, you can track someone's work and contributions — good and bad — over a lengthy period of time.

That means you can judge people by their actions instead of just their words. You can make a decision based on the things that really matter:

- **Quality of work**
  Many programmers can talk the talk but trip when it comes time to walk the walk. With open source, you get the nitty gritty specifics of a person's programming skills and practices.
- **Cultural perspective**
  Programing is all about decisions. Lots and lots of them. Decisions are guided by your cultural vantage point, values, and ideals. Look at the specific decisions made by a candidate in coding, testing, and community arguments to see whether you've got a cultural match. If there's no fit here, each decision will be a struggle.
- **Level of passion**
  By definition, involvement in open source requires at least some passion. Otherwise why would this person spend free time sitting in front of a screen? The amount of open source involvement often shows how much a candidate truly cares about programming.
- **Completion percentage**
  All the smarts, proper cultural leanings, and passion don't amount to valuable software if a person can't get stuff done. Unfortunately, lots of programmers can't. So look for that zeal to ship. Hire someone who needs to get it out the door and is willing to make the pragmatic trade-offs this may require.
- **Social match**
  Working with someone over a long period of time, during both stress/relaxation and highs/lows, will show you their real personality. If someone's lacking in manners or social skills, filter them out.

When it comes to programmers, we only hire people we know through open source. We think doing anything else is irresponsible. We hired Jamis because we followed his releases and participation in the Ruby community. He excelled in all the areas mentioned above. It wasn't

necessary to rely on secondary factors since we could judge him based on what really matters: the quality of his work.

And don't worry that extra-curricular activities will take focus and passion away from a staffer's day job. It's like the old cliché says: If you want something done, ask the busiest person you know. Jamis and David are two of the heaviest contributors to Rails and still manage to drive Basecamp technically. People who love to program and get things done are exactly the kind of people you want on your team.

Open Source Passion

> *What you want the most from a new hire is passion for what he does, and there's no better way of showing it than a trace of commitment in open source projects.*

—Jarkko Laine, software developer
(from Reduce the risk, hire from open source)

43: Get Well Rounded Individuals

Go for quick learning generalists over ingrained specialists

We'll never hire someone who's an information architect. It's just too overly specific. With a small team like ours, it doesn't make sense to hire people with such a narrowly defined skill-set.

Small teams need people who can wear different hats. You need designers who can write. You need programmers who understand design. Everyone should have an idea about how to architect information (whatever that may mean). Everyone needs to have an organized mind. Everyone needs to be able to communicate with customers.

And everyone needs to be willing and able to shift gears down the road. Keep in mind that small teams often need to change direction and do it quickly. You want someone who can adjust and learn and flow as opposed to a stick-in-the-mud who can do only one thing.

44: You Can't Fake Enthusiasm

Go for happy and average over frustrated and great

Enthusiasm. It’s one attribute you just can’t fake. When it comes time to hire, don’t think you need a guru or a tech-celebrity. Often, they’re just primadonnas anyway. A happy yet average employee is better than a disgruntled expert.

Find someone who’s enthusiastic. Someone you can trust to get things done when left alone. Someone who’s suffered at a bigger, slower company and longs for a new environment. Someone who’s excited to build what you’re building. Someone who hates the same things you hate. Someone who’s thrilled to climb aboard your train.

Extra points for asking questions

> *Observe whether a potential hire asks a lot of questions about your project. Passionate programmers want to understand a problem as well as possible and will quickly propose potential solutions and improvements, which leads to a lot of questions. Clarifying questions also reveal an understanding that your project could be implemented thousands of different ways and it’s essential to nail down as explicitly as possible exactly how you imagine your web app working. As you dig into the details, you’ll develop a sense of whether the person is a good cultural match.*

—Eric Stephens, BuildV1.com

45: Wordsmiths

Hire good writers

If you are trying to decide between a few people to fill a position, always hire the better writer. It doesn't matter if that person is a designer, programmer, marketer, salesperson, or whatever, the writing skills will pay off. Effective, concise writing and editing leads to effective, concise code, design, emails, instant messages, and more.

That's because being a good writer is about more than words. Good writers know how to communicate. They make things easy to understand. They can put themselves in someone else's shoes. They know what to omit. They think clearly. And those are the qualities you need.

An Organized Mind

> *Good writing skills are an indicator of an organized mind which is capable of arranging information and argument in a systematic fashion and also helping (not making) other people understand things. It spills over into code, personal communications, instant messaging (for those long-distance collaborations), and even such esoteric concepts as professionalism and reliability.*

—Dustin J. Mitchell, developer (from Signal vs. Noise)

Clear Writing Leads To Clear Thinking

> *Clear writing leads to clear thinking. You don't know what you know until you try to express it. Good writing is partly a matter of character. Instead of doing what's easy for you, do what's easy for your reader.*

—Michael A. Covington, Professor of Computer Science at The University of Georgia (from How to Write More Clearly, Think More Clearly, and Learn Complex Material More Easily)