Construx Web Site Link

Inc.com

Why I Never Let Employees Negotiate a Raise

At Fog Creek Software, every worker at the same level is paid the same salary. And when one gets a raise, they all do

From: Inc. Magazine, April 2009 | **By:** Joel Spolsky

What would happen if you got to work one day, went into the kitchen, and saw a list of your employees' salaries taped to the fridge? Would you freak out? Would you expect to find half of your staff weeping and the other half waiting with pitchforks outside your office door?

Because salary information is viewed as particularly sensitive, employers often go to great lengths to keep it under wraps. Some companies even make it a fireable offense for employees to compare salaries, or they write something into the standard employment contract prohibiting workers from disclosing their pay. (In the United States, this kind of rule is unenforceable, by the way, but some bosses hope their workers won't know that.) The trouble with keeping salaries a secret is that it's usually used as a way to avoid paying people fairly. And that's not good for employees -- or the company.

When my partner and I started Fog Creek Software, we knew that we wanted to create a pay scale that was objective and transparent. As I researched different systems, I found that a lot of employers tried to strike a balance between having a formulaic salary scale and one that was looser by setting a series of salary "ranges" for employees at every level of the organization. But this felt unfair to me. I wanted Fog Creek to have a salary scale that was as objective as possible. A manager would have absolutely no leeway when it came to setting a salary. And there would be only one salary per level.

After some digging, I found a Seattle-area software-consulting firm called Construx that had published on its website the outline of a decent professional ladder system (read about it at **construx.com/?nid=244**). It reminded me of the old pay system at Microsoft, which had worked pretty well when I was there. We used this model as a rough basis for our system, although we added some flourishes. I posted the first draft on my blog and got tons of great feedback, which I used to write up the second draft. The basic system has remained in place ever since.

In Fog Creek's system, every employee is assigned a level. Currently, these levels range from 8 (for a summer intern) to 16 (for me). Your level is calculated formulaically based on three factors: experience, scope of responsibility, and skill set. Once we determine your number, you make the same as every other employee at that level.

The experience part is pretty easy: It's based on the number of years of full-time experience you have in the field you're working in. No work done while you were still in school counts, and certain types of rote, menial work can never add up to more than a year of experience. If you worked as a receptionist for six years, for example, you aren't credited with six years of experience; I give you credit for one year.

Scope is pretty easy, too. Are you primarily helping someone else do his or her job? Do you have your own area of responsibility? Or are you running a whole product? We are able to define the scope of most jobs pretty objectively.

Quantifying skill is a little bit harder, but we still find it possible to define a fairly objective continuum from a newbie programmer ("Is learning the basic principles of software engineering; works under close supervision; not expected to write production code") to an expert programmer ("Has consistently had major success during participation in all aspects of small and large projects and has been essential to those projects' successes").

Once we defined our terms, we created a little chart that assigns a level based on an employee's experience, skill, and scope (a section of it appears on the previous page, and the whole thing is posted at **joelonsoftware.com/articles/ladder.html**). Then, we created another chart that lists the base salaries for each level, and that's how we figure out how much an employee makes.

Once a year, my management team sits down, reviews every employee's work, and recalculates every employee's level. Then we look at competitive market salaries using online tools such as Salary.com and Glassdoor.com, and we consider our own knowledge of the job market from the past year of recruiting and make sure that the salaries we have at each level are exactly where we want them to be.

Because everyone at the same level gets the same salary -- no fudging -- we sometimes run into difficulty. One problem with our system reveals itself when we're pursuing an employee who wants to negotiate for a higher salary. Sometimes this occurs when we find a great person who is currently being paid a salary that, in our view, is way above market. And sometimes this occurs when a potential hire just expects a reasonable amount of back-and-forth over salary because almost every other employer he has ever worked for maintains ambiguous salary ranges and there is always room to get paid better if you negotiate well. We usually address these situations by guaranteeing the recruit a larger first-year bonus than he would normally get. Here's the thing: Fog Creek is extremely profitable, and we have a generous profit-sharing plan, so the "guaranteed first-year bonus" is almost always less than the employee's profit-sharing bonus would have been anyway.

Our system was put to the test over the past eight years when the labor market was tight. It's easy to see why: Suppose you hire 100 yak drivers at $10 an hour, but then the Tibetan economy heats up, and you have trouble finding more yak drivers. The market rate might rise to $15 an hour. The weak-kneed thing to do is to hire new employees at $15 and hope that the senior people don't discover that the rookies are making more money than they are.

This is technically called salary inversion -- if you're the kind of person who likes to use self-important HR jargon. Salary inversion can lead to strife within an organization. It can also completely warp the relationships among managers, HR, and employees. This may seem ridiculous and sound apocryphal, but I actually once heard that managers at a major corporation told their key employees to quit and reapply for their old jobs, because the bureaucracy had made it nearly impossible to give them raises that reflected the competitive job market. At Fog Creek, we decided that the right thing to do when the labor market tightens is to give raises to everybody at the same level. This move can be painful and expensive, but the alternative is worse. I don't know about you, but I'm scared of pitchforks.

I can't guarantee that our system would hold up if margins were to erode, but I'm pretty sure that employees would be willing to accept slightly lower salaries as long as the system were transparent and fair, and it were clear what you needed to do to move up the ladder.

At the same time, if you hear a lot of griping about salaries, you shouldn't look just at your system for paying people. One thing I've learned from experience is that happy, motivated employees who are doing work they love and feel they are being treated as adults don't gripe about money unless their pay is egregiously unfair. If you hear a lot of complaints about salaries, I suspect that's probably a manifestation of a much bigger disease: Your employees aren't deriving enough personal satisfaction from their work, or they are miserable for other reasons.

It takes a lot of salary to make up for a cruel boss or a prisonlike workplace. And rather than adjusting pay, you might choose to focus on some nonmonetary ways to make employees happy. Happy employees make better products and provide better customer service and will make your company successful and profitable. And success allows you to pay workers better. It's a virtuous circle, and it has worked for Fog Creek. Let me know if it works for you.

Zack has four years' work experience, and the average of his scope (3) and skill (2) is 2.5, which Joel rounds up to 3. According to the pay scale, Zack would be a Level 11 employee (on a range of 8 to 16). Every other employee at Level 11 would be paid the same as Zack.

IMAGINARY EMPLOYEE: Zack

EXPERIENCE: 4
Years of work in the field

SCOPE: 3
Runs part of a project
Ranges from 0 to 6, with 0 representing an intern and 6 representing a worker who manages multiple projects

SKILL: 2
Can work independently but needs supervision
Ranges from 0 to 6, with 0 representing an intern and 6 representing someone whose contributions are unique

AN EXCERPT FROM FOG CREEK'S PAY SCALE:

YEARS OF EXPERIENCE	AVERAGE OF SCOPE AND SKILL					
	1	2	3	4	5	6
0—1	9	9	9	9	9	9
1—3	9	10	11	12	13	14
3—5	9.5	10	11	12	13	14
5—7	10	10	11	12	13	14

Joel Spolsky is the co-founder and CEO of Fog Creek Software and the host of the popular blog Joel on Software. For an archive of his columns, go to www.inc.com/keyword/spolsky.

- **Apply for the 2009 Inc. 500|5000**
- **This is How It's Done: 30 Smart Business Moves**
- **Wellness Programs Healthy Despite Other Cutbacks**
- **12 Cool Web Tools for Small Business**
- **Blog: 10 Tips to Avoid the Biggest SEO Mistake**

Joel on Software

Fog Creek Professional Ladder

by Joel Spolsky

Friday, February 13, 2009

The Fog Creek Professional Ladder determines your base salary. It is recalculated every August, and new base salaries go into effect September 1st (you'll see it on the September 15 paycheck).

Currently, the professional ladder is used for:

- Software developers
- Software QA/Testers
- System Administrators

Your career level at Fog Creek is determined as a function of three things: **experience**, the **scope** of your job, and your **skills**.

Experience

Definition: Years of full-time experience developing and testing software or administering computer systems.

This includes things like:

software development/programming
user interface design
managing software teams
software testing using scripting/programming tools
marketing software
selling software
system administration
1 year for completing a PhD

It does not include:

anything that happens in school, before school, or during school

technical positions that are not software development

Penalty:

Low level / highly repetitive tasks (rote tech support, manual black box testing) are collapsed into one year. (You don't have three years of experience, you have the same year experience three times over).

Scope

Definition: What your current job entails

Scope	Software Development	System Administration	Test/QA
0 Intern	Summer interns and Co-ops	Summer interns and Co-ops	Summer interns and Co-ops
1 Coder	Does development work but does not own any specific area of the code. Works on non-shipping code **or** on other people's areas, for example, fixing bugs, making small modifications, and implementing very small features.	Does small system administration projects as directed by someone else. Most work is on internal systems which are not customer-facing. Not much coding.	Executes test defined by someone else
3 AREA OWNER	Owns a major functionality area in a product, where they do or lead most of the development.	Owns a major area of system administration functionality, where they lead or do most development and work. Job must require coding, managing critical customer-facing systems, and responsibility for high-availability systems	Designs tests and test strategies for a major area of functionality in a product. Generally given an area to test and expected to come up with a test plan independently
4 PROJECT OWNER	Owns the development for a major project (an entire product), for example, FogBugz or Copilot.	Owns multiple major areas of system administration functionality, which they run with little supervision.	Responsible for all QA for a major project. Designs test plans, allocates resources, signs off on shipping versions.
6 MULTIPLE PROJECT OWNER	Owns or oversees multiple major projects	Overall responsibility for all Fog Creek systems, internal and external	Overall responsibility for all Fog Creek QA.

Skills

Definition: your skill level, regardless of actual responsibility.

Skills	Software Development	System Administration	Test/QA
0	Summer interns and Co-ops	Summer interns and Co-ops	Summer interns and Co-ops
1	Learning the basic principles of software engineering; works under close supervision; not expected to write production code	Learning the basic principles of system administration; works under close supervision; not expected to work on customer-facing or	Learning the basic principles of software testing; works under some supervision and occasionally responsible for developing tests.

		mission-critical systems independently	
2	Works under some supervision and occasionally writes production code	Works under some supervision and occasionally works on customer-facing or mission-critical systems	Extensive background in non-automated QA and test, qualified to develop test plans independently for most software. Has learned the practices, methods, conventions, and standards of software QA and test.
3	Some background in software engineering, qualified to write production code without much supervision, although they probably aren't designing anything. Will be expected to learn the software development lifecycle practices, methods, conventions, and standards of the computer industry. Understands and practices the skills of The Joel Test.	Some background in system administration, qualified to administer most types of systems we have in production without much supervision, although they probably aren't designing anything. Extensive experience with both Windows and Unix systems and most major Internetworking technologies. Will be expected to learn the practices, methods, conventions, and standards of system administration.	Extensive experience with automated and code-based testing. Writes scripts and creates unit tests. Mostly works on fully-automated tests. Will be expected to learn the software development lifecycle practices, methods, conventions, and standards of the computer industry. Understands and practices the skills of The Joel Test.
4	Familiar with industry practices and therefore can work independently as necessary. Proposes design approaches for review and agreement from peers and his or her supervisor. Has worked on one or more shipping projects, and has experience in each of the basic software development lifecycle steps needed to ship a product. Very competent in nearly all code-centered, detailed-design centered, and task-centered areas, and demonstrates additional competencies in other software lifecycle areas. Teamwork skills are excellent.		Software development skills equivalent to a programmer at Level 4. At this level no distinction is made between someone who primarily writes code used in testing vs. code used in production.
5	Has consistently had major success during their participation in all aspects of small and large projects and has been essential to those projects' successes. Has a track record of consistently rendering clear technical judgment and routinely considers architecture-level and project-planning issues. They ensure that projects are conducted in ways that benefit the project objectives, the people participating in the project, and Fog Creek's long-term interests. Innovative, consistent, and contributes beyond the assigned tasks. Mentors others. Actively seeks accountability. Has achieved mastery of The Joel Test. Competence extends to architecture, user interface design, project planning, documentation, fit and finish, and other project-level issues. Teamwork skills are excellent. Committed to a self-study program, reading books and journals. A developer at skill level 5 delivers **complete**, fully-baked products, from specs and prototypes, complete with documentation, management interfaces, polished user interfaces, automatic build routines, marketing collaterals, etc., which have been tested internally and externally.		
6	Has been critical to shipping a world-class product. Takes total ownership for all aspects of their project and makes many unique contributions. Decisions have a significant impact on Fog Creek's profitability and overall well-being. Routinely provides technical direction to other groups and people. Their competence extends well beyond project-level issues to company-level issues.		

DETERMINING YOUR LEVEL

Level is determined as a function of (a) experience (b) the average of scope and skills (rounded using normal math rules, so 3.5 becomes 4) using this chart:

Experience		Average of scope and skills						
Min	Max	0	1	2	3	4	5	6
0	1	INTERN	9	9	9	9	9	9
1	3	INTERN	9	10	11	12	13	14
3	5	INTERN	9.5	10	11	12	13	14
5	7	INTERN	10	10	11	12	13	14
7	9	INTERN	10	11	12	13	14	15
9	11	INTERN	11	11	12	13	14	15
11	15	INTERN	11	11	12	13	14	15
15	20	INTERN	11	12	13	14	15	16
20	25	INTERN	11	12	13	14	15	16
25		INTERN	11	12	13	14	15	16

Want to know more? You're reading Joel on Software, stuffed with years and years of completely raving mad articles about software development, managing software teams, designing user interfaces, running successful software companies, and rubber duckies.

About the author. I'm Joel Spolsky, founder of Fog Creek Software, a New York company that proves that you can treat programmers well and still be highly profitable. Programmers get private offices, free lunch, and work 40 hours a week. Customers only pay for software if they're delighted. We make FogBugz, an enlightened project management system designed to help great teams develop brilliant software, and Fog Creek Copilot, which makes remote desktop access easy.

joel@joelonsoftware.com