Scale-Free Extreme Programming

Kent Beck, Three Rivers Institute

A question that has vexed Extreme Programming from its inception is, “How does it scale?”

Sometimes the scaling question is asked out of legitimate concern. “If I work on a 450-person project, does XP have anything to offer me?” Sometimes the scaling question seems more like a defense. “XP doesn’t scale? Then I can safely ignore it.”

The Extreme Programming community has never had a satisfying answer to the scaling question, whatever its motivation. XP was designed with teams of up to 20 in mind. Larger XP teams have tried a variety of strategies. Some folks have single teams of 40-50. Some folks have multiple teams, with either some teams acting as customers for others or all teams working on a common code base. Some folks scale XP by absolutely fixing the team size and improving productivity year on year. None of these strategies has been successful enough to get the adoption ball rolling on larger projects.

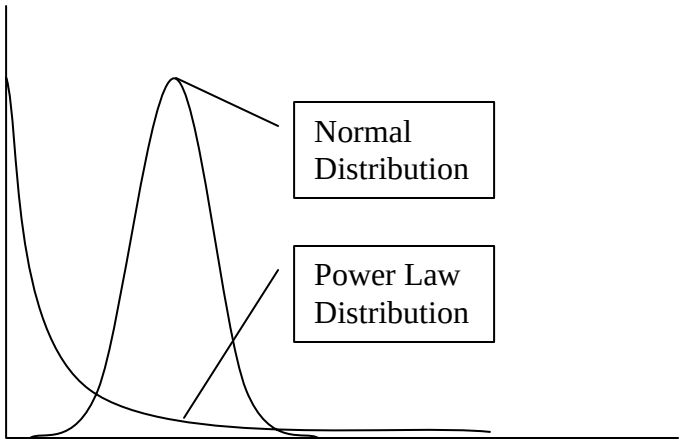
Scale-free networks provide a framework for thinking about problems of scale. What does the theory suggest about scaling XP?

Warning. I’ve recently been introduced to the idea of scale-free networks. In what follows I’ve tried to clearly indicate what is utter speculation and what I think I can defend with evidence. However, it might all be bollocks.

Power Law and Scale-Free Networks

The discussion in this section is stolen from *Linked*, by Barabasi. If you want to understand scale-free networks more deeply, I highly recommend you get a copy. It’s an engaging read and full of cool examples.

The power law is a distribution of random numbers significantly different than the “normal” distribution. In a normal distribution, you have lots of samples around a central peak, and very few samples very far from the peak. The power law produces lots of little samples and a few very big samples, far more big samples than predicted by the normal distribution.

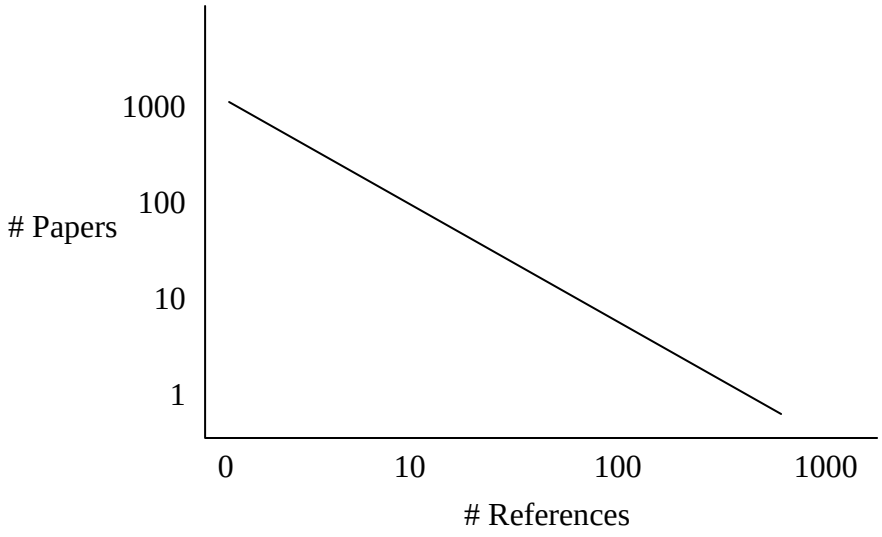


Imagine that we wanted to understand the patterns of reference between scientific papers. We could turn a corpus into a graph where each paper was a node and a reference in one paper to another is a link from one node to another.

If we plotted a histogram of the number of times a paper is referenced, what would the distribution look like? My first naïve thought when presented with this question was that every paper is referenced a few times, some a few more, some a few less. That would be a normal distribution.

Score one against common sense. Lots of papers aren't referenced at all, ever. A few, however, the "hubs" of our paper network, are referenced many times, far more than would be predicted by even the widest normal distribution.

What if we were to plot the number of paper versus the number of references? With linear axes we'd get a picture like the power law distribution above. If the axes were log scaled, though, we'd get a straight line (the data below is faked—we'll see real data later in the paper, once we get to stuff that matters, like programs):



Most anything can be turned into a network. You just have to decide what are the nodes and what are the links. If actors are the nodes in your graph and two actors are linked if they worked on the same film, you get a power law distribution shaped exactly like the one above. If chemical within the cell are the nodes in your graph and two chemicals are linked if they participate in the same chemical reaction, you get (you guessed it) a power law distribution. That little airline map in the back of the in-flight magazine is already a graph. Make a histogram of the number of links per node. Another power law.

Graphs where the number of links per node follows the power are called “scale free”. As long as the hubs can accept more connections, the graphs can grow without limit. (It would be a bit extreme to say that the airline network stopped growing quickly *because* Heathrow Airport couldn’t add more flights, but the two are definitely linked.)

Not all networks follow a power law distribution of links. The power law is an emergent property of two simple rules:

- Piecemeal growth—the nodes in the graph must be added one at a time.
- Preferential attachment—it must be more advantageous for a new node to be attached to an existing hub than to any random node.

Take a road map. Cities and towns are the nodes in the networks. Roads are the links. If you plot a histogram of the number of links versus the number of nodes, you get something that looks much more like a normal distribution:
Insert example

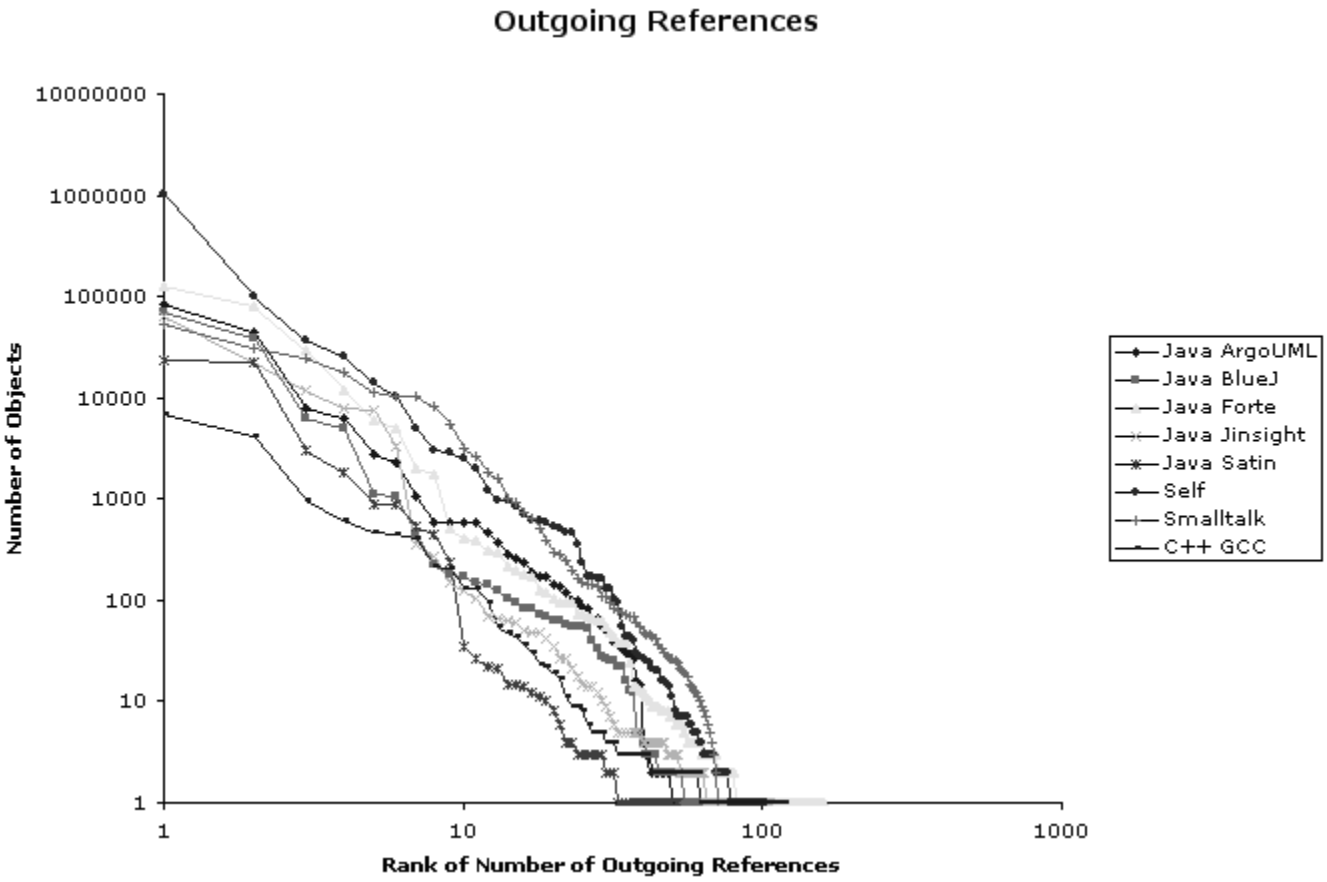
Why? Piecemeal growth is definitely present, but preferential attachment isn’t. It might be convenient for truckers to have a highway straight from the Oakland container terminal to Denver, but there is this little problem of distance in the way. We can have a

route from Oakland to Denver, but not a direct link. The expense of building longer roads is a stronger influence than the desire for direct connections.

I tell you this last to warn you. When I "got" scale-free networks I started seeing them everywhere. Some of the networks really were scale-free, and some weren't.

Software Is Scale-Free

Alex Potanin, a student of James Noble's at the University of ??? in New Zealand, found that networks of objects follow the power law http://www.mcs.vuw.ac.nz/~alex/powerlaws/index.shtml. If you plot the number of objects versus the number of outgoing references on a log-log graph, you get—ta da—a straight line. Similarly, if you plot the number of objects with a given number of incoming references, you also get a straight line. This property holds regardless of programming language or size of program.



The same is true of incoming references and static dependencies between classes. Why? Because the two conditions for a scale-free network are present:

- Piecemeal growth—design, even if done quickly and without feedback from implementation, still happens one or at most a few objects at a time.
- Preferential attachment—an object in the system that already performs a widely useful function is more likely to be useful as you add a new object than some

backwater object that is only there to capture the obscure difference between formatting sheep and goat weights.

Software Development is Scale-Free

Is software development itself scale free? We can turn a software development team into a graph by turning people into nodes and forming links between people if they need to communicate on a regular basis.

I don't know of any direct observation of teams to see if they are scale-free, but I can argue indirectly that they are. Conway's Law states that the organization of a system reflects the social organization that created it. The famous example is if you have a seven teams collaborate to write a compiler, you'll end up with a seven pass compiler.

If Conway's Law holds, and the organization of the system is scale-free, then the organization that created the software must also be scale-free. While this is far from a "proof" the software organizations are scale-free, it is suggestive enough to me to try to figure out the consequences of scale-free software development assuming software organization are scale-free.

I was coaching a team at a bank recently. They use lots of different technologies, and on our team we had some people who had no experience with the technology we were using. The barriers to them becoming effective had nothing to do with technology, though. The problem was that they didn't know who to call to get questions answered.

The two preconditions for scale-free networks certainly seem to be present:

- Piecemeal growth—teams grow a few people at a time. Even if 40 people are "added to" a project, they are assimilated a few at a time.
- Preferential attachment—you certainly ask questions of folks who have answers. The more questions a "hub" answers, the more questions they are able to answer.

You can experiment with this yourself. Have everyone write down the names of everyone they've collaborated with in the last week (or day or month, it should be self-similar). Plot this as a histogram with log-log axes. Fit a line to the data points.

This is good enough for me. We can begin thinkiing, "Assuming software development is scale-free, what does that mean for XP?"

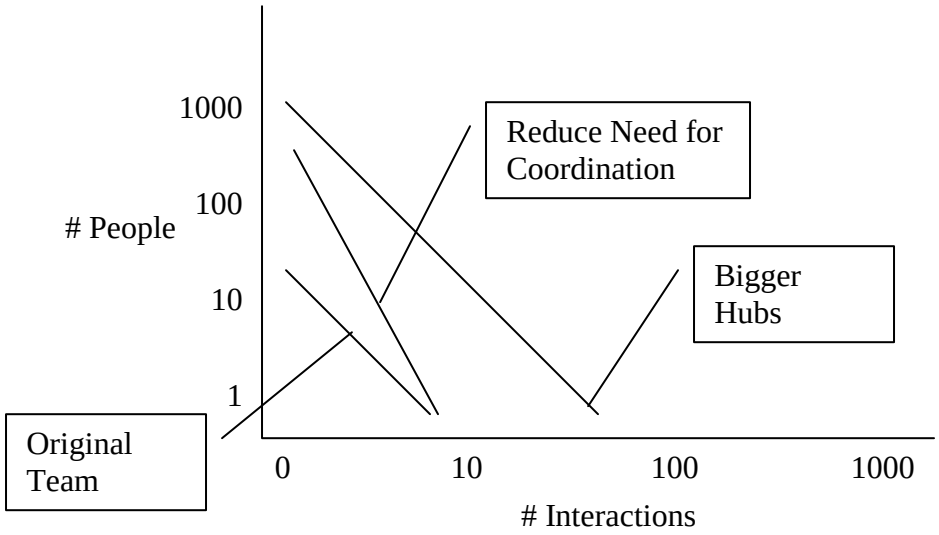
So What?

Metrics—What would happen if you measured an object graph and found significant deviation from the power law? It wouldn't necessarily mean there was a problem or mistake, but you would most certainly learn something if you figure out why there was a deviation.

Project size—[rank speculation ->] What if project size followed the power law? We would have lots of a little projects, and a few, but significant number of, very large

projects. This isn't a scale-free network, just a distribution of size. If this is true, XP's response that, "In every large project is a small project screaming to get and be successful," goes contrary to natural law. Instead, we should focus on providing real answers to projects budgeted for >10M USD.

Team size—This is the payoff pitch for scale-free networks. There are two ways of increasing the size of a scale-free network. Either you find hubs with more capacity, or you find a way to increase the slope of the line.



Who acts as a hub in current software development? Who are the folks who interact with the most other people on a regular basis? It depends on the team. Sometimes it is the project manager, wandering around to get schedule updates, who acts as a hub. Sometimes it is the architect, reviewing lots of code and design. Sometimes it is a particularly social QA person, making sure nothing falls through the cracks. Sometimes it is the buildmeister, gathering new code from all over.

From this perspective, XP has shot itself in the foot with respect to scaling. There is no single architect. There is certainly no buildmeister. QA, until recently, had a vague and peripheral role. As for project management, just look at the stories on the wall. From a micro perspective, eliminating these roles is valid. Teams work perfectly well without dedicating people to these roles. Eliminating these roles, however, has the secondary effect of limiting the potential size of XP teams. (The closest role to a hub on most teams is the coach, and we'd like to make the team self-coaching as quickly as possible.)

We didn't eliminate The Architect because we didn't like the clothes of people with The Architect on their business card. We eliminated central control of architecture because it didn't work well. It is difficult to align authority and responsibility when someone can make architectural decisions and they don't have to live with the detailed consequences.

There's the conundrum—we need high-capacity hubs, but we want them to align with the values and principles of the rest of Extreme Programming. I won't presume to suggest an answer here, but we'll know we've found one when XP teams highly value highly connected people on the team, but those people still have carefully aligned authority and responsibility.

The first place to look will be the traditional roles. Project managers make perfect sense on larger XP teams, or teams working in larger organizations. Teams need to coordinate with each other, and to project their activities in a form the rest of the organization understands. QA is coming to play a central role as more teams are figuring out how to write acceptance tests in advance of implementation. Architects as harmonizers of decisions made in separate teams seem to make good sense.

The second strategy for scaling XP is the one we've already been pursuing—reduce the need for explicit coordination between people. This has the effect on the chart of increasing the slope of the line, so we can have more people on the team without increasing the capacity of the hubs. Some activities that have this effect are collective code ownership, continuous integration, automated testing, and sitting together. Perhaps we can do more, looking at the team explicitly as a scale-free network.

Conclusion

The entire discussion above is preliminary and subject to experimental invalidation. Should it spark sufficient discussion to be annoying elsewhere, please use http://groups.yahoo.com/group/scalefreexp as a forum.