The Benefit of Patterns

Linda Rising

In 1995, I was working in a company where folks kidded me about being the "Patterns Princess," a reflection of my enormous enthusiasm for patterns. I believed that the ideas in Design Patterns[1] would make a real difference in the way software was developed. These ideas were the fleshed-out experience of experts with names that gave the rest of us a handle on the ideas, a way to talk about architecture. We could say, for example, "We will have a Proxy at each interface, and a Factory Method that creates messages, then a Visitor can deliver the messages to each device." For those of us who knew those names, it meant conversations at a higher level of abstraction and a faster way of making design decisions. I thought having the input of experts would improve both our designs and our way of designing.

Of course, many of these patterns were not earth shaking. I remember looking through the book and seeing, for instance, the *Mediator* pattern and nodding. I had used this pattern many times. It had nothing to do with object-oriented (OO) development, even though this book was purported to be about OO approaches. You could write the *Mediator* in Fortran (and I had)! I concentrated on the new and interesting twists on OO design. The *Visitor pattern* was fascinating. I had never thought of this design solution and probably never would have, so I regarded this as the most important contribution of the book.

Clearly, these patterns tapping the minds of experts were beneficial because they were uncovering startling solutions—yes, that was the big benefit of patterns! Of course, you had to include patterns like *Mediator*, because novice designers would need that introduction. The more obvious patterns should be documented for completeness, if nothing else. And, of course, we would all need the names and the vocabulary, for consistency. We all were going to learn about these patterns, no matter our current state.

Shapes and Connections

Today, although we're surrounded by increasing numbers of patterns in just about every software development area, the initial excitement has faded. Most software engineers see a new publication on patterns and think, "Yikes! Yet another pattern book!" Unfortunately, most of these publications lack the blockbuster sales of the 1994 appearance at the Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA) conference, when the line stretched from the Addison-Wesley booth out the door of the exhibit hall.

In my mid-'90s role as the "princess," I was working with a small team on a complicated problem. It involved a lot of protocols for

0740-7459/10/$26.00 © 2010 IEEE

I wonder if the real power of patterns is not to hand us exotic solutions, but to give us a way to remember the simple, ordinary, basic solutions that we know but forget in the heat of battle.

connecting to devices, where each designer was responsible for some subset of those protocols. We began with a whiteboard and a deadline. Pretty soon that whiteboard was covered with shapes and connections. It resembled a giant web, implemented by a spider on Benzedrine. And it was getting worse. The room got warmer and the discussion reflected that. We generated lots of heat. I've heard someone call this process "design by shouting." We were becoming muddy as we dug ourselves deeper and deeper into the trenches. At one point, some insightful soul (and it wasn't me) said, "Why don't we use a *Mediator*?"

The room got quiet. We all looked at one another. Of course, it was so obvious! **Because our company** had been going through a week-long training session in *Design Patterns*, we were certainly familiar with the *Mediator* pattern.

The Need to Simplify

Figuring out the best way to learn about patterns has long been a subject of debate within the patterns community. Some feel there should be a giant, searchable repository, where a struggling developer could—in the depths of despair—reach out for a pattern to address some sticky problem. Others felt that without training or some kind of exposure before the encounter with the problem, there was no hope of matching up a solution to the problem statement. As far as I know, this debate still rages. I'm a fan of the training model, simply because I tried to start on the repository journey with *The Pattern Almanac 2000*[2] and it didn't seem to help. Perhaps it was the wrong first step.

To return to the scenario of our little team, someone went up to the whiteboard and started erasing. Pretty soon, instead of a complicated spider web, we had something that looked like a wheel with a central hub and spokes connecting the hub to the device boxes—which was much simpler and easier to understand and maintain.

I often wonder if we would have gone ahead with that web diagram. Starting with something that complex when we all know that software only gets worse as it ages would have led to an unmaintainable mess in no time. I suspect, though, that we still would have tried to move forward with that plan. It's never been my experience that a design team had a moment of reflection and said, "Wait. This is too complicated. Let's throw the whole thing out and start over. Let's take our time and do this right!" I could be wrong. Maybe other teams do this all the time.

At the end of the day as I was explaining the experience to my husband, I heard myself say, "This changes everything. I wonder if the real power of patterns is not to hand us exotic solutions, but to give us a way to remember the simple, ordinary, basic solutions that we know but forget in the heat of battle." Does that ever happen to you? You're listening to your conversation and suddenly you say something you didn't expect and it causes you to stop and want to write it down. What a turnaround that was for me!

Organizational Patterns

I **once** read that cognitive scientists say that our ability to make good decisions of any kind—not just about software design—is hampered by the stress of the moment, by the movement in any direction of our peers. We get caught up and are swept away. Our best intentions get lost and we forget even those simple, ordinary, basic solutions like *Mediator*.

This decision-making theory made sense to me in the context of our team's brainstorming scenario. Sometime later as patterns took hold in our organization, I wrote a team management pattern called *No More than 10*—the result of hearing numerous retrospectives where team members said things like

- I try to estimate the amount of work needed to keep this team of 10 busy. I think the maximum size is 10. More than 10 is too much overhead.
- The project had about 10 people on it, on average, including System Test and Marketing.
- The team had six to eight developers and two system testers, a small team that interacted daily. They sat close together and worked together.
- The team was originally six or seven people. Now there are 30 to 35. When you have a small group, it's okay to be self-directed. With a larger group, there are too many given the amount of change. It can't be managed. It requires a team of managers.
- Team chemistry was very good. There were 5 to 10 people who worked well together, even those with diverse or clashing personalities.
- We kept the team size small, around 9—always less than 10. There might have been pressure from management to add more people to try to get it done faster, but we didn't want to add any more people. The tenth person would have made it difficult to divide the work.

That number 10 kept popping up in comments at the retrospectives. It seemed like a "magic number" to me. Now I have learned, again from the cognitive scientists, that this might be a "hardwired" number (whatever that means); about 10 to 12 people is usually the team-size limit that we're able to effectively collaborate with.[3]

Regardless of what the psychologists say, the retrospectives showed that our small teams that escaped "management attention" were able to make the best progress toward the delivery date. Unfortunately, adding more people seems to be the first thing that management wants to do in a crisis—not just at that company! (We're still learning the lesson from Fred Brooks,[4] who observed that adding more women to the job won't produce a baby in less than nine months.)

I wrote the pattern and presented it at our next Tech Forum. I thought it was good to document what we were learning, but I wasn't especially hopeful that it would make a difference in how teams were managed. I thought if Fred Brooks can't do it, I know I sure can't.

Using the Magic Number

As it turned out, my pattern was helpful. And as trivial as it might sound, I think it caught on with our company in part because of its catchy name. I believe that words, especially names, are important. I was reluctant to incorporate that magic number until a manager I respected told me something important. He said, "Linda, managers like numbers. If you call this pattern something like *Keep the Team Small*, it just won't have impact. People will keep asking how big is 'small,' and you'll generate a lot of not-so-helpful discussion. Just say 10!" So I did.

A few weeks later, I participated in a management version of the *Mediator* design pattern story. One of our "new ventures" teams was in trouble. Someone called one of those big meetings with lots of table-pounding attendees who wanted to see "something done about this." The room heated up, with lots of discussion but no real progress until someone said (and again, it wasn't me), "Wait, guys. Remember we have this pattern now. It's called *No More than 10*. It's got lots of our history in it and I know I can speak to the X experience. I say let's hold off and not jump on this right away." Again, the room got quiet. I said nothing. Everyone looked at me and then at each other. What a moment!

Finally, someone said, "Linda, you think we should leave them alone, right?" I reluctantly said, "Look who's on this team: John, Sam, Anne, Oliver … They're really good people. And you trust them to do their best. So, yes, the pattern says this, and I agree. Hold off on adding more people." And that was it. Similar scenarios played out many times afterwards in other big meetings. The pattern *No More Than 10* gave us a little point of light in the darkness. It led us out of the cave and back into the sunlight, and we looked around and saw how good it was. It saved us from ourselves.

So that's what I have learned about patterns. It's not the startling solutions that we need. It's a way of hanging on to the simple, ordinary, basic principles that call out to our best selves and remind us of the things we already know. Is this true for all patterns? I'm not sure, but I've seen this many times, so maybe it's a meta-pattern—a pattern about patterns or a pattern about some patterns.

Have you ever had an experience like this? A moment of insight that changed the way you see the world of software development? I'd really like to hear it and I know our readers would like to hear it, too. Thanks for listening.

About the Author

Linda Rising is an independent consultant living in Phoenix, Arizona. Her research interests include patterns, retrospectives, influence strategies, agile development, and the change process. She's authored four books, including Fearless Change: Patterns for Introducing New Ideas and The Patterns Handbook. Rising has a PhD from Arizona State University in object-based design metrics. Contact her at linda@lindarising.org or visit www.lindarising.org.

References

1. E. Gamma et al., *Design Patterns*, Addison-Wesley, 1995.
2. L. Rising, *The Pattern Almanac 2000*, Addison-Wesley, 2000.
3. R. Dunbar, *Grooming, Gossip, and the Evolution of Language*, Harvard Univ. Press, 1998.
4. F.P. Brooks, *The Mythical Man-Month*, Addison-Wesley, 1995.