THOMAS A. LIMONCELLI

a system administration parable: the waitress and the water glass

Thomas A. Limoncelli has written or co-written four books, including *Time Management for System Administrators* (O'Reilly) and *The Practice of System and Network Administration* (Addison-Wesley). He is a system administrator at Google in NYC and blogs at http://EverythingSysadmin.com.

tal+usenix@everythingsysadmin.com

I PRESENT TO YOU, DEAR READER, THIS parable about how the different ways we organize our work result in different levels of customer satisfaction.

It was the summer of 2008. July. I ate dinner in three different restaurants on three consecutive nights. I noticed something about the way that different restaurants have different schemes for how they manage to refill my water glass.

Interrupt Driven

Tuesday night I had dinner at Friendly's. In case you are not familiar with Friendly's: it is a mid-priced restaurant chain that markets to families with children. There are three within easy driving distance of where I live in the suburbs of New Jersey. The ones near me are usually staffed by high school kids, probably their first job.

As we ate dinner, the waitress refilled our water glasses every time we asked. She would hear and acknowledge our request, take our glasses to the kitchen, and return with new glasses full of water. She felt she was giving us excellent service. We asked, she provided promptly.

While she felt she was prompt, we saw it differently. It took a long time to get her attention. If she was taking orders from a large table with many children, or indecisive adults, it would be a while before we would be able to talk to her. She felt she was being immediately responsive to our requests; we saw her as being difficult to get service from.

The result:

- Best case: She was available immediately and we got our water quickly.
- Worst case: It took 5–10 minutes to reach her, and we got our water quickly, though we were without our half-full glasses while she was refilling them in the kitchen.
- Average case: not so good.

Batching with Interrupts

Wednesday night I had dinner at a fancy restaurant in the Hell's Kitchen neighborhood of New York City. While the name Hell's Kitchen comes from the seedy history of this part of town, lately it has gentrified and is the home of many excellent restaurants. New York City's waitress and waiter subculture has very high standards. These people work hard, are paid little, and live in a place where the cost of living is huge. Tipping starts at 20 percent in this city for good reason. Since I started working at Google's NYC office I have gotten to know many fine restaurants here.

At this restaurant our waitress refilled our water glasses differently. Every now and then she would pause from taking and fulfilling orders and sweep through her section with a pitcher of water. Every glass that wasn't full would be topped off.

If someone requested water sooner, she would bring out the pitcher, refill his or her glass, and then sweep through the rest of her section.

In other words, she was performing periodic batch processes and sometimes the batch process was kicked off due to an interrupt being triggered.

The result:

- Best case: Refilled before you asked.
- Worst case: Similar to the waitress at Friendly's but we always had glasses at our table, and fewer glasses were being rewashed.
- Average case: Sometimes we wait, but usually we don't. On average, pretty good.

Delegation

Thursday night I had dinner at the Skylight Diner in the Clinton neighborhood of New York City. Skylight is a Greek diner. If you are unfamiliar with the concept of the Greek diners that are typical of the New York and New Jersey area, there are a few things you should know. First, they are not called "Greek" because the food is Greek. They are usually owned and run by people of Greek descent. NYC is, very much, a city of immigrants and it is one of the things that makes this city so wonderful. Second, their menus are huge. Page after page of items from burgers to pasta to seafood to sautés. Some even serve Greek dishes, too. Third, the food is usually excellent and the portions are amazingly huge. Finally, if you hear the term "diner" and think "truck stop," you are 180 degrees wrong. To redeem yourself, come visit and be hungry. Very hungry.

At the Skylight our waitress never refilled our glasses. That was the responsibility of the busboys. The busboys were continually roving, taking away our empty dishes and refilling our water glasses. If your water glass is empty and you ask your waitress for more water, they turn to the nearest busboy, point at you, and say, "Más agua aquí."

This is the power of delegation or, one might say, automation.

The result:

- Best case: Refilled before you asked.
- Worst case: Refilled when you ask.
- Average case: Nearly always the same as the best case.

If you have a large party at a diner, they will simply leave a pitcher of water at the table. The entire process becomes self-service.

The result:

- Best case: Water exactly when you need it.
- Worst case: Waiting for a pitcher to be refilled.
- Average case: Usually close to the best case.

Organizing Work

We system administrators organize our work as differently as these three waitresses manage water refills.

When we are new, we are interrupt-driven. Someone asks, we respond. We feel we are doing a great job, because we are being responsive. We are, however, ignorant of the time our customers wait to get our attention.

On a busy day we are unreachable. We pride ourselves on having an excellent best case, but our average case is terrible. Worst of all, we are running ourselves ragged; interrupts manage our time, and we have little control.

We improve our situation when we become batch driven. We improve the average case. Sometimes we fear we are reducing the probability that someone will receive best-case service, since people won't get service on demand. The reality is that this case is very rare and actually isn't accounting for the wait time before they can make the request. The average case is greatly improved. The average case is pretty important.

There are some ways to turn interrupts into batching.

When we are interrupted, rather than jumping to that task, we have a choice. Listen to the request. Pause. Take time to consider: should I record this, delegate it, or do it?

I can record this request in my to-do list or open a "ticket" in my "request tracking system."

I can delegate it if we have set up a division of labor, where each sysadmin specializes in various things.

If it is truly urgent or if it is my job to take requests of this stripe, I can do it. However, this is the last option. I would rather record it, so that I can schedule it after higher-priority items or batch up similar requests.

Ask Questions

In the past I assumed all requests were urgent. Now I always ask, "How soon do you need this?" It is surprising how many urgent-sounding requests aren't urgent when we take the time to ask. "Oh, I'm about to go to Boston. Can this be done before I get back?" or "The new employee starts on October 18. Anytime before that is fine."

It is funny how often we forget to ask that question.

Our ability to record a task is dependent on having a place to record it. Keeping a to-do list is one way. (See my recommendations in [1].) However, it is important to have a request tracking system that lets our customers make requests without interrupting us. These "helpdesk automation" products or "request tracking systems" come in many shapes and sizes. There are open source ones such as Request Tracker [2] or ORTS [3], and commercial products too many to list.

Keeping requests in such a system permits us to record all communication related to the request in one place, permits customers to check on the status without bothering us, permits us to work as a team better (hand off tasks to co-workers), and lets management both observe our work without being annoying and produce metrics to help them do their own jobs better.

A request tracking system also lets us abide by priorities better. When there is a backlog, we can find the high priority tasks easily and work on them first. We can sort the list of tickets by due date.

Emergencies

Emergency requests are the one thing requiring an interrupt-driven response. Sadly, we find some customers who claim everything is an emergency. We can fix this by having a written definition of what constitutes an emergency. This policy must be written and agreed to at the management level. At a newspaper an "emergency" might be something that would directly prevent tomorrow's edition from getting out on time, but not something that is one degree away. At a school, an emergency might be something that prevents a scheduled class activity from happening on the date listed in the teacher's lesson plan or syllabus (but only if the instructor gave prior notice).

Just as stoves, pots, and pans are tools that a chef has in a kitchen, a request tracking database and a written definition of "emergency" are tools needed by a system administrator.

Better Batching

We can batch our work in other ways.

We can do all related tickets as a batch. For example, sometimes doing a DNS update requires certain tools to be open: a control panel, a certain command line, or maybe a text editor open to a particular configuration file. Once we've done one DNS update, we can search for all the other requests related to DNS and do them too. Or we can also ignore all non-urgent DNS-related requests until 4 p.m. each day and do them then.

We can batch by location: gather all the tickets that require physically visiting Building 47 and do them in one trip.

We can batch up all the requests from a particular user. When we are feeling overloaded it can be very satisfying that, while there are dozens or hundreds of tickets sitting idle, at least we've made one user very happy.

If working on the requests requires communicating with the user, it can be faster to get the person on the phone and walk though each request, completing them in real time. Even better, do all the tickets that don't require talking with the user, get them on the phone, and work through the remaining. The user sees a flurry of emails related to status updates on those tickets and then suddenly receives a phone call. They feel like they've won the lottery.

It's more efficient, too. It takes a certain amount of time to open a communication channel with a person (tracking them down, setting up an appointment, walking to their office, or opening an Instant Message window). Once you've suffered that overhead, you might as well do all their open tickets or at least report on their status.

I encourage batching by doing certain requests on certain days. Requests for activating network jacks might be done on Tuesdays and Thursdays. Rather than changing backup tapes every day, use a tape jukebox large enough that they only need be changed every Monday and Friday; a larger jukebox permits them to be changed monthly, an even larger one can practically eliminate human interaction.

Delegation and Specialization

Sometimes we delegate or specialize like the Skylight Diner. We can delegate more easily when things are documented, but we don't need fancy documentation. A bullet list of steps on a wiki can be "just enough."

Specialization is only possible as our organization grows. Often an IT team begins as a single individual who carries the entire burden for the small company.

Then another person is hired and the two share the load. Fast-forward a few years and we find a 10-person IT team all struggling to do all tasks. At what point should they have specialized? Probably after the second or third person. It depends on each organization. Different organizations need different specializations. Typically people specialize where specific knowledge is needed. If the environment requires a particularly large and ever-growing storage system, some may specialize in storage. Even small teams have one person who knows networking better than others, and is responsible for the Internet gateways and firewalls.

With proper documentation everyone can do all the basic "everyday" tasks related to provisioning ("adds, changes, and moves"). Leave the uncommon tasks to the specialist (grow the service, optimize, add entirely new features).

In other words every system administrator on the team should be able to connect a new machine to the network, update DNS, and so on. The specialist might be the one who has the knowledge required to create a new subnet and DNS zone or to modify firewall rules.

When tasks are documented it is easier to optimize and automate them. We can strategically select specific parts to automate (which bullet items on the wiki page), or we can automate the entire process.

Automation

Giving a table of customers at a restaurant their own pitcher of water turns a burdensome request into a self-service feature. In system administration it is often easiest to create self-service versions of provisioning requests. Take, for example, providing VPN service to your users. Setting up the VPN server is something done once; there is no benefit to automating. However, adding new accounts should be a repeatable process, and therefore easy to automate.

At first one might automate the process so that a system administrator can enable or disable service for a single user very easily. That gives them a tool that frees up their time for other things. The next step would be to make this a self-service operation: a Web page that users can click on to make the request, the request is approved by a human, and the system does the right thing. Some people might be pre-approved; for example, users in the LDAP Group "engineers" might be pre-approved for VPN access. Or only people in the LDAP group "visitors" require approval. Now more than freeing up your time, you have a tool that empowers users to help themselves.

A restaurant doesn't have as many opportunities for automation as system administrators do. Yes, they could build robots to take our orders and deliver food. That would be awesome. However, what we love about restaurants is the human aspect. It is a luxury to be served. While restaurants lack opportunities to automate, they can improve workflow through batching and better organization.

As system administrators we have many choices about how we do our work: interrupt-driven, batching, delegating, automating, self-service.

What do you do?

Things to Think About

1. In the past week at work, were you interrupt-driven, batching, delegating, automating, or creating self-service systems?

2. What are three tasks you do at work that could be batched?

3. When someone makes a request, how do you know how soon he or she needs it?

4. What specializations are in your team? Are they recognized formally?

5. How would you reorganize how you do your own work? How would this make things better and worse for you? For your customers?

6. How would you reorganize your team or IT organization? How would this be better or worse, for you and your customers?

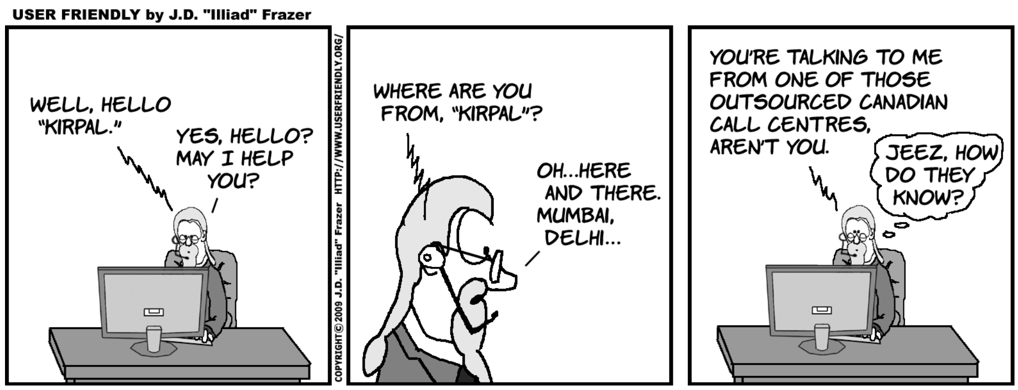
7. In answering question 5 and 6, many changes were proposed. Which specific changes would have the biggest impact? Which one change would have the most immediate benefit?

REFERENCES

[1] Thomas A. Limoncelli, *Time Management for System Administrators* (O'Reilly Media, 2005).

[2] RT: Request Tracker: http://www.bestpractical.com.

[3] ORTS: http://www.orts.org.


USER FRIENDLY by J.D. "Illiad" Frazer
WELL, HELLO "KIRPAL."
YES, HELLO? MAY I HELP YOU?
COPYRIGHT © 2009 J.D. "Illiad" Frazer HTTP://WWW.USERFRIENDLY.ORG/
WHERE ARE YOU FROM, "KIRPAL"?
OH...HERE AND THERE. MUMBAI, DELHI...
YOU'RE TALKING TO ME FROM ONE OF THOSE OUTSOURCED CANADIAN CALL CENTRES, AREN'T YOU.
JEEZ, HOW DO THEY KNOW?