;login:
APRIL 2013 VOL. 38, NO. 2
ELECTRONIC SUPPLEMENT

Conference Reports

LISA '12: Advanced Topics Workshop

San Diego, CA

December 11, 2012

Advanced Topics

Summarized by Josh Simon (jss@clock.org)

The Advanced Topics Workshop began on Tuesday morning; once again, Adam Moskowitz was our host, moderator, and referee. We started with our usual administrative announcements and the overview of the moderation software for the one new and several long-absent participants. Then, we went around the room and did introductions. In representation, businesses (including consultants) outnumbered universities by more than 3 to 1 (down from 4.5 to 1 last year); over the course of the day, the room included 11 LISA program chairs (past, present, and future, up from 6 last year). The Workshop is now old enough to vote (this is the 18th ATW).

For the first topic we discussed collaboration. One attendee works in a widely distributed environment and wondered how others were doing environment-wide collaboration. Some of the technical answers included a site-wide cross-team chat service (both textual and video), mailing lists, a wiki, discussion forums, Google Docs, and a centralized place to store scripts with revision control, as well as the telephone for remote users. Social answers included using Agile methodologies (especially the standing meeting where everyone quickly reports what they did yesterday, what they're working on, and what's blocked on someone or something else). Other techniques included going to lunch as a group (if you're sufficiently close-by geographically), and recognizing success so people feel appreciated for their contributions. Several found that providing these tools and services centrally with some kind of branding was helpful. Some of the challenges included getting sysadmins to work with the security and other teams due to team siloing, showing management that there's actually value in collaboration, and motivation; getting individuals to want to collaborate can be hard, and seeing collaboration as work as opposed to socializing can be tricky. Some environments require the silos, such as governmental or military sites where, by policy, the classified and non-classified sides cannot easily talk to each other. It was also noted that collaboration, especially in environments where it's new, is a cultural shift. Changes should be small and incremental, with the benefits clearly visible, to break down the perceived barriers between teams. Getting management buy-in is essential for collaboration, especially across teams or geographies, to work.

After the morning break we had our first lightning round, asking how people stay positive given the stresses of the job. Answers included being selective about which battles to fight, biking to work or otherwise exercising, doing approved non-work things during the day (such as attending a physics lecture), eating regularly (don't skip lunch), finding things to be satisfied with or about, job hunting, keeping work at work, liking who you work with, making small changes to foster excellence, playing video games, putting things in perspective (one said he's paid "a foolish amount of money to code interesting things from home"), realizing what may be causing the problem is not within one's control or responsibility, remembering the long-term company direction, spending time on non-work activities (such as grandchildren, hobbies), talking to users about their research areas, volunteering to use one's work skills for social justice organizations, working from home, working on cool or exciting or new things, working on multiple projects, and working on projects that are rewarding. One noted that happiness is contagious; another noted that in almost all cases one's coworkers and customers aren't malicious or dumb, but that they, like you, are trying to accomplish something for the good of the company or institution.

Configuration Management

Our next topic was configuration management (CM) in general. CM is effectively a solved problem for systems, but what about for applications? One site is working on controlling complexity, describing complex setups in a human-readable, human-manageable way. Several aren't in this space yet but realize they need to be. One noted that CM is good for hardware but not necessarily for applications or services; we have to deal with computational ontologies. The game is harder when you're dealing with internal guts of applications, and managing failure states on clusters makes it an immensely hard problem. There's no good answer yet. One person noted that the current tools work for most things if you'll run those tools as root or the Administrator, but they have issues running something untrusted as the superuser. A lot of the CM tools are good skeletons but cross-component complexity of simple components is where it starts to break down (things become too complex and too fragile); you need to avoid making simple concepts complex. Unfortunately, many practical problems aren't solved in the CM space (and are barely solved in the programming language space). One noted that none of the tools track who does what where; there's no audit trail to speak of.

Someone noted that today's tools don't solve all of today's problems. If you need to solve the problem today you can use the existing tools and processes to get close enough. If you don't need

to solve the problem today, you can spend more time writing the tools to move closer to the ideal. The question becomes if CM is the right model for application management. We have to achieve a balance; people need to understand they're making a choice to reach that balance. Tradeoffs are what we do all of the time; the problem is increased complexity, as we're trying to do more stuff with the existing tools. Can we simplify? Are we doing too much? Making assumptions? Taking a step back to figure out how (and why) we do things can be useful. Revision control can be done outside the tool. "Run As" is a missing feature, and even if it exists, the communications reporting piece fails.

Several people wondered whether using CM tools to manage applications is the wrong problem. In fact, one argued that the discussion is ill-aimed and ill-conceived, conflating several topics. People want control; the tools are more for the relatively static bits of CM, and it's being conflated with application deployment, application control, and application rollout, as well as SLA, monitoring, and health at the level above that. The bottom line is that trying to use the same tool and mind-set to control vastly different aspects of the system is not necessarily wise. We need to agree that there are three types of tools and not conflate them; the complexity comes from trying to mash this stuff all together: You don't control complexity, you manage it.

Women in Computing

Next we discussed women in technology. One of the problems with the Women in Advanced Computing (WiAC) summit is that not many men show up. What do the men present think? Some are nervous to show up and speak in that environment because they're worried that what they say might be misperceived despite what they try to champion and do. One question is how does someone talk about and suggest solutions to the problem without coming across as sexist (or, when relevant, racist, etc.)? A politically incorrect observation: of the female engineers one speaker has worked with, many weren't particularly good; another had the opposite experience.

Another person noted that it varies by discipline; he's seen women make inroads in the developer space, but engineering and chip fabrication is primarily male. Someone asked whether "diversity" instead of "women" would make it different. How do we encourage people outside the audience to come?

Several people noted they aren't getting (even unqualified) women applicants. In 11 years, one has had all of three female candidates to interview, and another has similarly had very few female applicants. There's a deep-seated cultural problem, especially with how girls are pushed away from math and science in general. There are conversations in progress; it makes sense and is important to observe before entering that discussion. There are institutions that reached out to train more women. Don't just wait for women to apply, but go out and encourage them to. Recruiting for all sorts is important; diverse groups tend to come up with better, broader, and more useful ideas. If we valued it, we'd go out and get it. Someone went to a Grace Hopper conference where he was one of two or three men in a crowd of several hundred women, which was a visceral wake-up call as to how people who aren't white men can feel every day. Respect the norms of behavior and treat everyone professionally and courteously.

We also need to make sure everyone gets a chance to speak; not everyone is aggressive enough to fight to do so. Others agreed; one applicant left IT for a while to start a catering business, but because of the gap she's having trouble getting back into the field.

"Make it happen" has a high risk. Some women think it's a higher risk to be the only woman (or one of very few) in the team, group, or company. One woman present has been told that she should negotiate a higher salary because of the higher risk. There are studies all over that show—in business, engineering, grants, etc.—given identical resumes with gender-relevant or culturally identifiable names, the man's is rated higher. Consider doing away with names in the candidate process.

Software-Defined Networks

Our next topic was software-defined networks (SDN). OpenFlow is a more-sophisticated-than-Infiniband way to take a switch and put whatever routing intelligence is on it by letting the switch talk to an off-system controller. OpenFlow gives you a way to write network control/routing software as a service on a UNIX box instead of adhering to on-board rules. This allows you to put a cheaper switch in the rack and "smear" traffic across multiple sites. For example, one person has a multiple terabit pipe in his WAN, made up of hundreds of 10 GB pipes, and he can run long-haul links for IP traffic at 95% utilization 24x7 instead of at 50% utilization. Others have begun looking into this, and find it interesting in and useful for building private cloud systems. It's actually not new; HPC interconnects have been doing this kind of thing for ages. SDN is broader than OpenFlow; the concept is "treat the network as building blocks" with more direct and more granular control than routing protocols. The protocols influence traffic routing and shaping; SDN comprises rules, not just influence. It was noted that the standards are incomplete; Juniper and Brocade gear can do it, and it's possible to do it with white-boxes and write-your-own firmware; QoS is in some versions but not from all vendors. For some, latency is very, very important and they need to know the latency penalty for OpenFlow and deep packet inspection (which can be hard and expensive).

Careers and Life

After the lunch break, we had another lightning round, asking about new-to-us tools we've discovered in the past year. Answers included Agile methodologies, Go, Google applications, having a team of minions to do one's billing, iPython Notebook for

research labs, jenkins, logstash, Mobi battery pack for the iPhone, mosh (ssh over slow or inconsistent connections), multibeast, S3, SoundHound, tablets, using Evernote for collaboration, using git for network management, using spreadsheets for small group project management, and vagrant (to automate setting up virtual machines).

Next we discussed the aptly-named "career-type stuff." One person mentioned his marriage failing in part due to the stress and focus of being a sysadmin and tipping his work/life balance too far onto the work end of the spectrum. He asked what do you do to leave work at work? Answers included allowing things to fail, changing one's reporting chain ("firing the boss"), delegating tasks to others (at both home and work such as TAs and graduate students in the educational space and a cleaning company at home), empowering one's employees to make decisions (and backing them up), having alternate work schedules (such as nine hours per day but every other Friday off), keeping Mondays and Fridays as meeting-free days, learning to say No (or at least asking how critical something is so you can level-set priorities and expectations), letting the awful processes tie management up in their own red tape, listening to audio books, meditation as part of a martial arts regimen, not logging into work email from home at all, not trying to save everyone, practicing for choir, putting recurring meetings on your calendar (named "Sleep" or "Lunch"), realizing that "good enough" often is, replying to email only if there's a reason to do so, reserving certain times of day for family, taking public transportation that requires keeping a schedule, the discipline and focus inherent in exercise in general, using the security policies as an excuse not to VPN in, and working from home (and having a dedicated space for doing so). Whatever you do, take small steps in the process to increase the chances of success.One person noted that the vacation culture in the US is "a day at a time and stay reachable" as opposed to Europe where it's "a couple of weeks or more and not necessarily reachable."

Someone lets failures occur by telling the appropriate audience, "This is going to fail for these specific reasons" and then shutting up. Do it in writing so you can refer back to it after the fact. This approach does have pitfalls; if you let too much fail then you're the incompetent one. Another reminded us that there's a difference between "letting things fail because it's not your job" and "letting things fail because you're overworked." He lets the different work-providers fight it out and decide what gets priority.

That segued to a question about career paths. One person has no promotion path other than management (which he's hated in the past) or technical fellow (which requires him to stop being good at his current job). He asked how to deal with a pinnacle career, when all the paperwork asks, "What's your next job?" and he has no answer for that. Advice included becoming a consultant at a different company, creating a new job to exhibit progress and using it as a goal, leaving the employer when there's nowhere else to go, and picking things you want to do and making work provide time to build those skills.

IPv6

Our next discussion was on IPv6. One place is running out of their assigned IPv4 space. Their management doesn't see the issue but does see the cost of moving to IPv6 so they don't want to do it. Tens of thousands of homegrown applications (mostly related to money) mean they don't want to mess with it. Getting the impetus to move forward (even on small things or dual-stack) is tricky. Someone asked if anyone has experience doing this in a commercial environment. One person suspects that there's a middle step between "working in IPv4" and "working in IPv6" since IPv6 isn't entirely ready. Using 6-to-4 NAT may be required and it may suck for the people using it until they get around to the recoding. Someone else had a discussion at the bar last night; you can run IPv6 in parallel with your IPv4 environment, and he's learning about IPv6 by using it at home. Another thinks of it as another Y2K problem: there will be a crisis point down the line and people will panic and deal with it then. Yet another is seeing that interest in IPv6 is finally growing beyond a small core community that's been looking at it for a while; he can't go to it yet due to missing vendor support. It was suggested that instead of planning a whole-company conversion, get started by just picking something and doing it, such as building a new datacenter. It was noted that this isn't IPv6 specific; you always need a business case as to why spending the effort now is worth it.

Cloud

Clouds were up next. One person has rolled out two internal clouds. The main issue is for things that need stable naming of some kind; DNS and Kerberos are both in the cloud in his environment now. Naming and external configuration were concerns. Another is finding unexpected resistance to people using clouds; in particular, when you buy a computing resource there's something to touch ("kick the tires"). Capital funding requires something tangible. There are serious security implications with the data entailed in the systems; you have to be careful with the appropriate controls (in the US, these include FERPA, FISMA, and HIPAA). Someone noted "cloud computing" is the latest CIO "gotta-do" buzzword, and it's not the right tool for all jobs. For putting up public-facing Web sites, it's great; for encumbered data, it's not so great. It's also expensive; it's cheaper to build your own DC if you know what you want and need. Data locality is the single biggest issue facing some people. People are virtualizing because their configuration management systems don't work.

One environment is using the cloud to compartmentalize to keep student personally identifiable information (PII) hidden from

their corporate overlords, and spinning up infrastructure is easy. Another has been doing proof-of-concepts of new large infrastructure before buying the hardware. DevOps in the cloud is a win; do it virtual first and then do it on real hardware; however, some environments can't put anything classified in the public cloud. Bring your own device (BYOD) and having access to all the data they want can be problematic. Their executives are demanding the ability to do this despite security policies.

One person argued that GRID computing failed because you couldn't handle different parallel versions of the same software. Configuration isolation is a feature, not a bug. They're trying to adapt an HPC machine for a workload and are seeing little overhead and better I/O than expected. Another noted that leveraging technology is good; outsourcing and the subcontractors and so on puts data in interesting problems: once the data's out of house, it's not private regardless of the SLA and contract. Virtualization is a valid way to compartmentalize applications. Someone at a software development house noted that cloud computing lets them test at scale without paying for the hardware.

Next we had a few quick polls: Of the 28 of us in the room, only 1 does not carry a cell phone; 4 have dumb phones, 12 have Android devices, 9 have iPhones, and 5 have BlackBerrys. (Yes, some people carry more than one device.) Fourteen people in the room have some form of eating plan (and 4 blame work for the reason they go off it). Twelve of us are in new jobs since last year's workshop, about half of those to different companies as opposed to different roles within the same company. One even has a new job since the start of the session! Five people in the room are actively looking for a new job.

We had a brief meta-discussion about the workshop. Some people think some topics drag a bit (and someone jokingly suggested we bring in a gong). Some want to blacklist some specific topics. Most like the soft-versus-hard topic balance but would like to alternate between them more as opposed to a long string of only one type. It was reiterated that there's intentionally no attempt for us to accomplish something specific, and the participants agree that's still desired.

During the afternoon break, we took a quick unofficial poll: Of the 29 systems in use, there was 1 tablet, 5 PCs running either Windows or Linux, and 23 Macs.

Wishful Thinking

We kicked off the final block of the workshop with a quick topic: "If I were king, I'd wave my magic wand and...." Leaving aside that wizards or witches have magic wands and kings have scepters, answers included bringing all of Human Resources into a room and setting it on fire, explaining to the user support team that their job is to support users, firing the business unit whose profit model is effectively fraud, fixing attitudes about documentation so that getting it done is more important, fixing the research funding model so principal investigators can share resources and funding, forcing everyone to and everything into source control, getting rid of half of one's leadership chain, getting rid of legacy garbage (both things and processes), having management better shelter workers from administrative garbage, hiring more people and training them with a deep understanding of the product, hiring quality consultants (not just the lowest bidder), killing off incompetents who refuse to learn, level-setting processes to the right amount, moving away from a forest of symlinks, putting policies and procedures together and moving away from "talk to so-and-so," rebuilding the team from the ground up, replacing all of internal IT management, staffing projects adequately, stopping others from using my team to keep their team from failing, taking more time off, tracking bugs and issues better using a good single sign-on and source-of-truth for credentials (one place has four such systems), using the inventory system correctly, and working closer to home.

BYOD

Our next topic was Bring Your Own Device (BYOD) and the security and policy issues thereof. One environment has strict policies about work versus private space and that never the twain shall meet. Executives are moving away from that; the CEO wants an iPhone not a BlackBerry, and now they want personal devices accessing the work space. They have a pilot program coming up, with stipulations that the company can zap the device and use the users' own data plans. There will be no new BlackBerrys assigned or purchased after 1/1/2013. A specific college has no security restrictions like them, so BYOD has happened forever there. They treat U-purchased devices as their own. They explain to the important people that they need more than one-size-fits-all enterprise solutions for people who don't act like they're in an enterprise. Another environment approached it organically. BlackBerry allows remote-wipe (as does Android now), but iPhone didn't. There's a guest wireless network for personal devices, and they have firewalls that can do policies for threat protection, antispam, antivirus, and so on. They're experimenting with VPNs.

One concern with BYOD is data leakage protection; client confidential information can't go on a personal device. Someone noted that Financial shied away from BYOD because of the policies, but then they moved away from that position because the risk assessment said it wasn't adding anything to avoid BYOD. Someone else noted BYOD is not keeping up with users' needs. You don't want to demotivate people to stop doing their job. Perimeters don't work; you have to secure the machines, data, services, and so on, and decide what's important to you. One environment implemented an iPad network (not on the internal network); the corporate IT people type in the password for you. Another used to have a network drop per device and hardened hosts,

but they're moving away from "everything as a bastion" toward "Infrastructure as a Service." There are liability issues with BYOD (such as hardware damage). Their policy is to give people the tools they need to do the job. Users want support of company-purchased devices at home.

It was noted that it costs a horrific amount for big corporations to support phones, such that two-year support means they can give you a new phone every year or two. Some vendors support multiple personalities on smartphones (e.g., John Smith the Person versus John Smith the Employee).

"Give them what they want" is hard to do because marketing firms tell users what they want. This happens for applications as well as devices. It's all about the data that needs to be protected and secured. Some advocate for their users, but you can't unconditionally give them what they think they want, such as applications that aren't licensed for commercial use. Users don't understand what is or isn't supported.

Incidents

Next we discussed incident management. A few days before the workshop, a major public service was down for 18 minutes due to a load balancer problem. The problem was fixed quickly, but the people fixing it didn't communicate well with the rest of the company. This brought up the concept of incident response: how can they better organize and prepare to respond to incidents (outages, security events, and so on), and how do you test for readiness?

Someone brought up the Game Day exercises. One place had a datacenter person who was too good, so they arranged to have him arrested and taken to jail so others could have the joy of dealing with the issue. Another referenced Chaos Monkey: Do something chaotic during off-peak business hours to see what happens and how to fix it. Some places do postmortem writeups very well; they're blameless and they focus on what happened to prevent it from recurring.

As part of this discussion we had another lightning round: if you have outages and aren't doing anything like this, why not? Answers included being in an environment that won't test and has no failback, doing so ad hoc, having enough real outages not to need to test fake ones, having planned outage testing for new but not legacy systems, having server-level but not datacenter-level disaster recovery planning and testing, limiting testing to nonproduction or dark datacenters, not being involved in a planning or response role, not being pressured to get back up fast and/or being sufficiently resilient for the SLA, small outages are frequently unnoticable, and the SLA requiring only three 9s or five 8s or even three 6s of uptime. Some places do incident management postmortems, fail traffic to a secondary datacenter while they push code to the primary datacenter, or have business continuity weekends where they take down a regional datacenter. Others simply have no plans and no staff or capability to test. One person had an instance where an application died every so often and restarted itself, for four months, without user-visible notification, so it's possible to do this kind of work too well.

Finally, we closed out again with a lightning round asking what's coming up over the next year. Answers included allocating laptops instead of desktops, building out a home theater system, building the latest version of their supercomputer to handle 10 times the data, chairing the LISA '13 conference, consolidating DNS and DHCP services, coping with the next round of bigger disks, developing a standard API for an internal cloud, doing more business analytics, finding a new job, growing technical communities, implementing and migrating to a new CM system, implementing company-wide incident management, implementing more cross-silo services, implementing services for a zero-trust network, integrating the network with their new corporate parents, isolating stuff, launching new products, migrating to new versions of vendor software as old ones go to end of life, moving into the cloud, moving to a new location, relocating a lab, renovating houses, retiring at the end of 2013, revising books, selling a house, splitting a lab into two labs, and updating monitoring and visualization systems.