Did the Microsoft Stack Kill MySpace?

Friday, March 25, 2011 at 9:30AM

Todd Hoff in Example, failure

Robert Scoble wrote a fascinating case study, MySpace's death spiral: insiders say it's due to bets on Los Angeles and Microsoft, where he reports MySpace insiders blame the Microsoft stack on why they lost the great social network race to Facebook.

Does anyone know if this is true? What's the real story?

I was wondering because it doesn't seem to track with the MySpace Architecture post that I did in 2009, where they seem happy with their choices and had stats to back up their improvements. Why this matters is it's a fascinating model for startups to learn from. What does it really take to succeed? Is it the people or the stack? Is it the organization or the technology? Is it the process or the competition? Is the quality of the site or the love of the users? So much to consider and learn from.

Some conjectures from the article:

1. Myspace didn't have programming talent capable of scaling the site to compete with Facebook.
2. Choosing the Microsoft stack made it difficult to hire people capable of competing with Facebook. .Net programmers are largely Enterprise programmers who are not constitutionally constructed to create large scalable websites at a startup pace.
3. Their system had " “hundreds of hacks to make it scale that no one

wants to touch," which hamstrung their ability to really compete.
4. Because of their infrastructure MySpace can’t change their technology to make new features work or make dramatically new experiences.
5. Firing a lot of people nose dived morale and made hiring tough. (duh)
6. Los Angeles doesn't have startup talent capable of producing a scalable social network system.
7. Facebooks' choice of the LAMP stack allowed them to hire quicker and find people who knew how to scale.

Some very interesting points in the comments (from the article, email, Hacker News, Hacker News):

1. **Nick Kwiatkowski**: Having been in a position where I was able to work with some of the programmers who worked at MySpace, the issue wasn't the engine (whether it was on ColdFusion or .NET), it was the environment they choose to breed for their developers. Management would say "We need X feature NOW to remain competitive". They would then select a group of developers to implement that feature. The biggest problem was they didn't allow the developers to have staging or testing servers -- they deployed on the production servers on the first go-around. Sometimes these developers were given 5 or 10 projects that they had to deploy in very little time. And all this with no change management or control. No versioning either. MySpace management never wanted to go back and review code or make it more efficient. Their solution was "more servers". They ended up hiring a crew who's sole job was to install more servers. Meanwhile they had developers checking in buggy code and they were racking up technical debt at an alarming rate. At the time MySpace was

running two major versions of their application server behind what was recommended for use. When Microsoft & New Atlanta came around, they jumped at the idea to essentially sell off their technical debt (like a mortgage to a financial firm), and have somebody else take care of their problem. The problem then was Microsoft was not updating their old code, they simply were adding new features on .NET. This didn't solve their problems and left them in a situation where they still needed to fix the old stuff, all the while updating new code.

The issue with MySpace was this : they are a classic example of when you don't listen and you accumulate too much technical debt. Fixing old stuff should be a priority, and doing things like change management, version control, testing and development servers, etc. are all a must. This is why the facebookis able to deploy new changes with little impact -- they have everything tested and proofed out before they let their actual users play with it.

2. **u_fail**: All this sounds like is business people with little to no understanding of technology blaming technology , when all along it was lack of innovation and lack of technology people at the top making the decisions. MySpace was always run like an entertainment company.
3. **Eric**: It's not about technology, it's about the people implementing the technology.
4. **Robert Scoble**: MySpace's architecture made it very difficult to ship new features and "pivot" once it was clear they were getting their ass kicked.
5. MySpace's core competencies relied on others and never scaled in-house, and seemed to completely buckle under.
6. **Robert Scoble:** It's all about talent and getting the techie help to innovate. Those folks aren't using Microsoft's systems.
7. **codezed**: MySpace betting on Microsoft's stack was not the problem. The problem is they bet on a social media platform from

Telligent. Instead of developing their own platform and adding features to a platform that they owned and developed, MySpace decided to let another company's product provide innovation for their core feature set.

8. **Hack**: I remember going on someones page with 100 .gifs and having a page load after 25 seconds. Facebook definitely takes the crown.
9. **Marcelo Calbucci**: This is a bad analysis. They would have suffered the exact same faith if they've gone RoR or PHP. Actually, their destiny was pretty much set once Facebook changed the bar for what a social network should offer and look like.
10. **Sriram Krishnan**: Your usage of any stack is proportional to the expertise you have in it.
11. **S Jain**: Putting the blame on Microsoft is totally wrong. I think it was a culture at MySpace. I work at a start-up in LA and have many friends at Myspace. It was so often I'd be working late and they'd say apply to a bigger company why work so late.
12. **Gregg Le Blanc**: When MySpace gave their keynote as to why they switched to Microsoft at MIX06, the cited the fact that they could handle the same user load on basically 2/3 the servers (246 -> 150) with a decrease of average CPU load from 85% to 27% to serve pages at their 65 million user load. So, I think it's more about people, architecture, and business plan. Their strategic decision was forgetting to innovate. That's technology independent.
13. **Robert Scoble**: Mark Zuckerberg said it would be a lot harder elsewhere to build a webscale company, which is why he moved from Boston to Silicon Valley.
14. **Jebb Dykstra**: MySpace was lost the very moment it sold to News Corp. The true death spiral started at that very moment. If the talent had stayed, such as Richard Rosenblatt and they hadn't sold themselves off for $670 mm, then many of these issues could have been the excuse.

15. **Omar Shahine**: Lets not forget Twitter could not scale at all, and was basically broken. It wasn't build on Microsoft technology, but they managed to fix it.
16. **Sean Scott**: Even if technology and a lack of talent were part of the issue and MySpace's inability to respond to Facebook, ultimately the user experience is what cost them.
17. **David Noël**: When they said this would cost a few clicks and page reloads but increase user experience and longer-term engagement, NewsCorp would say no because of the fear of loosing monetizable metrics.
18. **Steve Smith**: MySpace's code apparently had a lot of technical debt and from the sounds of things, its architecture was textbook Big Ball of Mud. They made bets on third party software for their core business.
19. **Scott Stocker**: The availability of developers willing to work for MySpace was the biggest issue.
20. **Linda Nicolai**: Users weren't vortexing away from the site because of technology complaints, it was because of the new kid on the block that had a shinier bike...Lack of innovation, dismissiveness of the competition, limited product roll-outs, too much turmoil at the top, technical issues, but don't blame it on the lack of technical talent here in L.A.
21. **Dan Bartow**: My initial impression after reading the article is that I pretty much agree with everything Scoble said related to LA and the talent problem. MySpace is in a unique situation because they are entertainment industry focused... primarily music industry and related content. Because of this focus, LA is really 'the' place to be for what they are doing. To be plugged in to the scene and doing artist events, networking with record companies... what better place to be? If you're a financial company... New England. Entertainment... LA. Technology startup, Silicon Valley... etc. On the flip side, they are also one of the highest traffic web sites out

there so they really need to be on the forefront of technology. It's true that LA isn't necessarily known for having the type of engineering talent that the SF Bay Area has in terms of cutting edge, web scale architectures. However, think that MySpace's problems are much less about the technology platform choices they have made (Microsoft vs. anything else) and more about engineering leadership, market positioning and time to react, etc. I was personally very very impressed with their performance and scalability. Their quick auto-provisioning and re-provisioning stuff was amazing. They were able to take a 1 million concurrent user test like it wasn't even happening so that's pretty awesome. I actually had no idea it was Microsoft technology and probably would have never guessed based on what they were doing in their architecture. As engineers you and I both know that the right mindset along with strong leadership can take pretty much any technology to the highest levels out there. PHP was one of the least scalable, worst performing languages out there until Facebook took it all the way to the top. Leadership. There are plenty of high traffic Microsoft sites out there so I don't think the technology CHOICES are as much of an issue as Scoble makes it out to sound, and based on what I saw engineering leadership at MySpace was actually pretty bad ass. Talent... sure, LA sin't Silicon Valley but there are plenty of reasons to move to SoCal, believe me. Personally I think MySpace's problems have all been in business leadership and slow reaction time to their competition.

22. **Scott Seely**: The reasons for the death spiral at MySpace has nothing to do with technology. Is has everything to do with human issues. Once Chris DeWolfe and Tom Anderson left, a lot of the key leadership followed. Also, a lot of the creatives (product managers, devs, artists, etc.) saw executives leaving and decided it was a good time to try something else. Too much institutional knowledge left too quickly, and that sabotaged MySpace.

23. **jdavid**: What MySpace had was Cultural Debt, and a fear of destroying what they had. Facebook won because they were Socratic to the core. Fox was a closed source company, and so when we were working on Open Tech like Oauth and OpenSocial gadget servers, we had to do it as closed source software. We had no peer review. It made it really hard to ship and test something when you don't have linkedin, google, bebo, and twitter reviewing your code. On top of that when those companies found a bug, we had to re-implement that code in .Net. On top of that MySpace and .Net were well designed for strong typing and those types work a bit different than Java. It didn't take a lot of time to port so, we kept doing it, but you have to think about this, you are at a race with a company like Facebook who had zero profit motive at the time, and billion in funding and a ground up stack. Meanwhile MySpace was just clearing out cold-fusion and we had really old blogging platforms that could not just get ripped out. We also had management that didn't want to piss off users and so we would have 2 or 3 versions of the site out there, and we required users to upgrade to new versions.
24. **jdavid**: The deal soured with Google because the terms were around pageviews and clicks. MySpace and Fox decided to target those terms to maximize revenue. The result was that MySpace added unneeded page flow for just about every user action. It destroyed UX and pissed off Google. Our team kept joking with management that we were going to create a MySpace-Lite as a side project from our REST APIs and to get rid of all of the crap. WE SHOULD HAVE DONE IT. WE SHOULD HAVE CREATED MYSPACE-LITE. The Deal with Google was worth $300 million a year on total revenue of $750 million a year. MySpace Music was loosing the company mad money, it was and is a flawed model. Our team wanted to create an API where games and sites could access the music, and to create open playlists. We wanted to make the music

open, and then work to license it. We were told that it was not possible.

25. **lemmsjid**: The web tier has very little to do with scalability (don't get me wrong, it has a lot to do with cost, just not scalability, except in subtler ways like database connection pooling)--it's all about the data. When MySpace hit its exponential growth curve, there were few solutions, OSS or non OSS for scaling a Web 2.0 stype company (heavy reads, heavy writes, large amount of hot data exceeding memory of commodity caching hardware, which was 32 bit at the time, with extraordinarily expensive memory). No hadoop, no redis, memcached was just getting released and had extant issues. It's funny because today people ask me, "Why didn't you use, Technology X?" and I answer, "Well, it hadn't been conceived of then :)". At the time, the only places that had grown to that scale were places like Yahoo, Google, EBay, Amazon, etc., and because they were on proprietary stacks, we read as many white papers as we could and went to as many get-togethers as we could to glean information. In the end, we wrote a distributed data tier, messaging system, etc. that handled a huge amount of load across multiple data centers. We partitioned the databases and wrote an etl tier to ship data from point A to point B and target the indices to the required workload. All of this was done under a massive load of hundreds of thousands of hits per second, most of which required access to many-to-many data structures. Many startups we worked with, Silicon Valley or not Silicon Valley, could not imagine scaling their stuff to that load--many vendors of data systems required many patches to their stuff before we could use it (if at all). Times have changed--imagining scaling to MySpace's initial load is much easier now (almost pat). Key partitioned database tier, distributed asynchronous queues, big 64-bit servers for chat session, etc. But then you factor in that the system never goes offline--you need constant 24 hour access. When the whole system goes down, you

lose a huge amount of money, as your database cache is gone, your middle tier cache is gone, etc. That's where the operations story comes in, wherein I could devote another bunch of paragraphs to the systems for monitoring, debugging, and imaging servers. Of course there's the data story and the web code story. MySpace was an extraordinarily difficult platform to evolve on the web side. Part of that was a fragmentation of the user experience across the site, and a huge part of that was user-provided HTML. It was very difficult to do things without breaking peoples' experiences in subtle or not subtle ways. A lot of profile themes had images layed on top of images, with CSS that read, "table table table table...". Try changing the experience when you had to deal with millions of html variations. In that respect, we dug our own grave when it came to flexibility :). Don't get me wrong, there were more flaws to the system than I can count. There was always something to do. But as someone who enjoys spending time on the Microsoft and OSS stacks, I can tell you it wasn't MS tech that was the problem, nor was it a lack of engineering talent. I am amazed and humbled at the quality of the people I worked next to to build out those systems.

26. **by n_are_q**: The reason for MySpace's downfall is crystal clear to anyone who worked at the company and cared to look around and make sense of what was happening - it was catastrophic lack of leadership and vision in management and product, paralyzing political infighting, and general lack of competence at the top levels of the company's management. The people making the decisions would change their mind and requirements constantly because of this. There were numerous entire features that were simply not launched and abandoned AFTER they were completed because the management couldn't agree on how they wanted to "position them" (and they were great features). The top management level was in a constant state of political infighting, and that most likely came from fox and the way they ran shit. There was no one to judge

and reward competence at that level, it was simply about who could cover their ass better or come out looking better. MySpace was huge, and everyone just wanted a piece of the pie. One of the issues that stemmed from this was lack of respect for technology in the sense that no one at the higher levels saw the company as a technology company. They saw it as an entertainment or media company. That created the cultural problems on down that eventually contributed to bad products and shoddy implementation. Now, the core technical part of the organization was actually extremely competent. MySpace was pushing more traffic than Google at one point in its heyday, and the site scaled just fine then. That wasn't an accident, I have worked with some of the smartest people in the industry there. But because tech wasn't the point for executives, those people were tightly controlled by non-technical management, and so products suffered. MySpace could (and still can) scale anything, to say that they had a scaling problems by the time they got to their peak is complete gibberish. Over the years they have developed a very mature technology stack. No one knows about it because it's entirely proprietary. The problem was management and product that was basically... incompetent, and lacked anyone at the proper levels who would care to see and fix it.

27. **mhewett**: I'm not sure these technology-based analyses are correct. I had four teenagers at the time and they all switched from MySpace to Facebook because the MySpace pages got cluttered with glaring ads. The Facebook layout was cleaner and had no ads (at the time). There was no problem with site speed.
28. **strlen**: This indicates that the technology and location choice aren't the cause, they're symptoms of company management that doesn't understand building non-trivial Internet applications (what the optimal technology stack for one is; where, when and how to hire developers to build one) and yet maintains authority to make all technical decisions. Contrast this with Facebook, Google et al--

where every process (from corporate IT to production operations to HR and recruiting) is designed with needs of the engineering organization in mind: "Want two 24" monitors? No problem. Want Linux on your laptop? No problem. Want ssh access to production? No problem. Want to fly in a candidate to interview? No problem."

29. **sriramk**: I have a bunch of friends from MySpace and one common theme I do hear is that they feel that a better architecture (not connected to the stack) would have let them ship stuff quicker. It seems like there is a limit to technical debt you can endure. If you rewrite too quickly, you'll be made fun of as the company which killed their product by trying for a massive rewrite/re-engineering effort. Take too long and somebody like FB gets to ship features very quickly and leap ahead of you.
30. **ChuckMcM**: I think the insight here for entrepreneurs is that 'web' is not 'enterprise'. Microsoft has spent billions targeting 'enterprise' (to compete with the big software players like SAP, Oracle, Salesforce, Etc) which is structured and deployed very differently than 'web' (where big users are Google, Facebook, Etc.). When I worked at NetApp we had customers in both sides of the world, what we discovered was that the **'enterprise' guys often had a cadence, it went something like grow, identify challenges, coast, upgrade, repeat**. That cycle was built around taking the latest release of 'big player' software, qualifying it, then putting it into production, growing a bit, figuring out where the problems were, talking with their vendor, waiting, getting a new release and then repeating the cycle. But the cadence at 'web' players was sporadic at best and often frenetic, feature request, test, iterate, feature, iterate, test, coast, coast, coast, feature, feature, feature, test, coast, Etc. As ideas struck the 'web' infrastructure folks they would would immediately implement some sort of prototype or test case, then rapid fire come up with features the infrastructure needed to support to maximize this feature. But sometimes they would go long

periods, months, where nothing would change (in the infrastructure side, web pages, UX, etc sure but same servers and storage assets). What I learned from that was that while it was great to have some other company own the burden of the 'base infrastructure' in terms of operational expense (hey you don't need Linux hackers if your not changing stuff at that level, so its a savings in staff etc etc) it imposes a hard limit on the change derivative. What happens if you try to change faster than the infrastructure can change, is that you end up hacking around the limits, and that builds up technical scar tissue that over time slows your mobility still further. So bottom line, you can't continue to pivot faster than your infrastructure, if you're hacking your way around the infrastructure to change, then your ability to change will die by a thousand hacks. If you find your self thinking you need to hack around your infrastructure, listen to that warning and start planning for a more agile base to work from now rather than when we're struggling to keep an old system working while developing a completely new one.

31. **phlux**: There is one infrastructure choice, policy really, that can be made that will provide you with the best available path throughout the life of your companies infrastructure: VIRTUALIZE EVERYTHING
32. **akshat**: Silicon Valley might be awesome for its talent, but I would bet that there are enough awesome developers in any metropolis to make any web company successful. You need to be a place which needs to be able to attract such people.
33. **Lukeas14**: You can try to blame Myspace's "death spiral" is due to their tech stack, moving to L.A., but the root cause is their inability to innovate. It was virtually the same site in 2010 as it was in 2002. Think of how many new features Facebook has launched and relaunched in that time. There was a time when many people enjoyed their custom designed profile pages. But then they grew up and MySpace didn't.

Some of my observations:

- The stack isn't the real problem. That's almost silly. Look at Windows, .Net, etc and they are all quite capable of scaling if used correctly. Facebook started with LAMP, but along the way they changed everything about it such that it would be hard to say they were still using LAMP in the end. Twitter went through a similar phase with RoR. You can't exist at this scale without transforming everything you touch to meet your specific needs. Facbook also uses very small teams of people, so you don't really need great hoards of talented people. You need some talented people backed up by the right vision, support, and a management who sets them free to solve problems.
- The question is why wasn't the technology used correctly? The comments point to a lot of different reasons, but they all eventually get down to people. Were they bought by a management team that didn't value technology? That seems likely if some of the development, release, and design decisions are to be believed. Core competencies seemed to be farmed out to third parties. Technoligies like SANs were brought into to solve problems instead of dealing with the problems directly. Both are deadly. Whatever makes you a success you must own completely. Maybe being an "entertainment" company rather than a technology company fosters that sort of approach. But that again would get back to ownership and management. People seem to forget that talent can't work in a straight jacket, unless they are magicians.
- The innovation question is interesting. We see companies like Facebook, Amazon, Google, and Apple constantly innovating. How important was that in MySpace's fate? And why wasn't innovation important to them?

So, what happened? More importantly, how can new startups and groups within companies avoid this fate? We see this all the time. It's depressing for everyone involved. Shouldn't we be past this sort of thing by now?

Related Articles

- StackOverflow and PlentyOfFish are famous examples of successful companies using Microsoft technology. But notice they make use of it, which is different than handing the keys to your kingdom to a company that doesn't have the same aligned goals as you. Only bet the farm on you. The LAMP stack are separate components, easily replaced and modified. When you buy into a complete stack you are stuck when there are problems. You never want to be stuck.
- Good comments on Hacker News Thread
- How MySpace Tested Their Live Site With 1 Million Concurrent Users

---