Cost-Aware Architectures

As engineers we are remarkably badly trained in thinking about cost. I don't mean some big O order of magnitude cost but real dollar cost. You could given me five fault-tolerant algorithms and almost with my eyes closed I could pick the best one. But if you would then ask me what the difference would be in cost between the best and second-best when the system would start to scale I would falter.

I learned how to incorporate cost into architecture the hard way. At Amazon.com the margins are, as in most retail, razor thin and as such any fluctuation in the cost of infrastructure and systems immediately eats into the bottom line. And Amazon.com has many moving pieces, with hundreds and hundreds of internal services, so keeping control over cost, architecturally, is just plain hard. Especially in the days before we had AWS, our cloud services.

The most important concept is that, when you are growing, your cost should grow over the same dimension your revenue is coming in over. For Amazon.com that dimension is number of orders. If orders go up your cost should be allowed to rise as well. Although if you are architected well, you will be able to exploit economies of scale and your cost will rise less than the rise of your revenue. If you are architected correctly for cost-awareness scale becomes your friend.

I work with many young businesses exactly on this concept: how to find the dimension you are going to make money over and then make sure that the architecture follows the money. You decompose into smaller building blocks, and make sure the management of those blocks is automated through APIs and/or tools like chef & puppet. You make sure the blocks are fine grained enough for you to have tight control over the growth of your cost. And then you use business rules to drive the way that you auto-scale your systems up and down when you start to grow. You make explicit decision about for example what you want the customer experience to be in terms of latency and auto-scale you systems to meet these goals and not better. You can then make a trade-off of xxx msec better latency is going to cost us $0.0y per customer, but conversion will improve by z%. You will then also start to use the different pricing mechanisms to fine tweak it: on-demand, reserved and spot. I have seen some companies do very innovate things with spot to drive their cost down, for example pinterest and vimeo both use Spot in their online web server fleet; they bid on the spot market before the allocate an on-demand instance.

Problem is of course that early on it is sometimes not obvious over which dimension you money is going to come in. In those cases you assume it is going to come in over the same dimension as your customer traffic is going to grow as a surrogate for the revenue dimension. This works of course well with pay-as-you-go systems, it is not be a good surrogate in the case of a subscription model. For that you need a different data driven approach.