Cloud Programming Directly Feeds Cost Allocation Back into Software Design

Friday, March 6, 2009 at 2:09AM

Todd Hoff in GAE, Strategy, algorithm, cloud

Update 6: CARS = Cost Aware Runtimes and Services by William Louth.
Update 5: Damn You Google, Damn You Yahoo! Why D'Ya Do This to Us? Free accounts on a cloud platform are a constant drain of money.
Update 4: Caching becomes even more important in CPU based billing environments. Avoiding the CPU means saving money.
Update 3: An interesting simple example of this idea showed up on the Google AppEngine list. With one paging algorithm and one use of AJAX the yearly cost of the site was $1000. By changing those algorithms the site went under quota and became free again. This will make life a lot more interesting for developers.
Update 2: Business Model Influencing Software Architecture by Brandon Watson. *The profitability of your project could disappear overnight on account of code behaving badly*.
Update: Amazon adds Elastic Block Store at $0.10 per 1 million I/O requests. Now I need some cost minimization storage algorithms!

In the GAE Meetup yesterday a very interesting design rule came up: Design By Explicit Cost Model. A clumsy name I know, but it is explained like this:

> *If you are going to be charged for an operation GAE wants you to explicitly ask for it. This is why some*

automatic navigation between objects isn't provided because that will force an explicit query to be written. Writing an explicit query is a sort of EULA for being charged. Click OK in the form of a query and you've indicated that you are prepared to pay for a database operation.

Usually in programming the costs we talk about are time, space, latency, bandwidth, storage, person hours, etc. Listening to the Google folks talk about how one of their explicit design goals was to require programmers to be mindful of operations that will cost money made me realize in cloud programming cost will be another aspect of design we'll have to factor in.

Instead of asking for the Big O complexity of an algorithm we'll also have to ask for the **Big $** (or Big Euro) notation so we can judge an algorithm by its cost against a particular cloud profile. Maybe something like **$(CPU=1.3,DISK=3,IN-BANDWIDTH=2,OUT=BANDWIDTH=3, DB=10)**. You could look at the Big $ notation for algorithm and shake your head saying that approach will never work for GAE, but it could work for Amazon. Can we find a cheaper Big $?

Typically infrastructure costs are part of the capital budget. Someone ponies up for the hardware and software is then "free" until more infrastructure is needed. The dollar cost of software design isn't usually an explicit factor considered.

Now software design decisions are part of the operations budget. Every algorithm decision you make will have dollar cost associated with it and it may become more important to craft algorithms that minimize operations cost across a large number of resources (CPU, disk, bandwidth, etc) than it is to trade off our old friends space and time.

Different cloud architecture will force very different design decisions.

Under Amazon CPU is cheap whereas under GAE CPU is a scarce commodity. Applications between the two niches will not be easily ported.

Don't be surprised if soon you go into an interview and they quiz you on Big $ notation and skip the dusty old relic that is Big O notation :-)
