6 Lessons from Dropbox - One Million Files Saved Every 15 minutes

Monday, March 14, 2011 at 9:02AM

Todd Hoff in Python, Strategy

Dropbox saves one million files every 15 minutes, more tweets than even Twitterers tweet. That mind blowing statistic was revealed by Rian Hunter, a Dropbox Engineer, in his presentation How Dropbox Did It and How Python Helped at PyCon 2011.

The first part of the presentation is some Dropbox lore, origin stories and other foundational myths. We learn that *Dropbox is a startup company located in San Francisco that has probably one of the most popular file synchronization and sharing tools in the world, shipping Python on the desktop and supporting millions of users and growing every day*.

About half way through the talk turns technical. Not a lot of info on how Dropbox handles this massive scale was dropped, but there were a number of good lessons to ponder:

1. **Use Python**

- 99.9 % of their code is in Python. Used on the server backend; desktop client, website controller logic, API backend, and analytics.
- Can't use Python on the Android due to memory constraints.
- Runs on a single code base using Python. Dropbox runs on Windows, Mac, Linux using tools like PyObjs, WxPython, types, py2exe, py2app, PyWin32.
- Pros:
  - Developers talk to each other and express ideas in Python
  - Easy to learn, easy to read, easy to write, easy for new people to pick up.
- Cons:
  - Don't be silly.
  - OK, it can use too much memory and be too slow. Not a big deal on the server side, just buy bigger machines. On the client side you can't get an old Power PC user to upgrade.
  - Coding in a mixed environment of Python and C creates problems because it's hard to profile across the language boundaries like you want to do when fixing memory and CPU problems.
  - Memory fragmentation issues are reason why scripting languages may not be a good idea for long running processes.

2. **Just Work Baby**
   - Shouldn't matter what file system you are on, what OS you are using, what applications you are using. The product should always just work.
   - Python helped them iterate fast through all the different error cases they experienced on the wide variety of platforms they support.

3. **Release Early**
    - Code something in a day and release it. Python makes that easy.
4. **Use C for Inner Loops - Optimizing CPU is easy**
    - A way to handle the too slow problem.
    - Optimize inner loops to reduce CPU time.
    - 44% of overhead when looping in Python vs C (2.88s vs 1.61)
    - Python VM bytecode dispatches are really slow.
    - Many tools exist for profiling CPU.
    - CPU optimizations are usually limited to small code sections.
5. **Poll - Polling 30 Milion Clients All Over the World Doesn't Scale**
    - Created an HTTP notification structure to avoid polling the server on the client site.
6. **Custom Memory Allocator - Optimizing Memory is Hard**
    - This was there biggest problem for a while. Could use huge amounts of memory and the memory would never be freed. For large sync they could use up to 1.5GB, now they rarely use more than 100MB.
    - Hard because:
        - Few tools exist for profiling memory for Python and C
        - Memory bloat has so many causes: leaks in Python and C code; memory fragmentation; inefficient use of memory.
    - Fixing obvious memory inefficiencies didn't help. They thought there was a memory leak, but there wasn't.
    - Problem turned out to be **memory fragmentation**. Memory fragmentation is what happens when different sized memory blocks are continually being deleted and allocated. What happens is contiguous blocks of memory can no longer be allocated. CPython doesn't have a garbage collector, so all this memory simply wasn't able to be allocated and the heap

continually grew so memory requests could be satisfied.
    - Solution was to create a **custom allocator**. The file meta-data object grows a lot when doing transfers, so the obvious low hanging fruit was to create a custom allocator in C using mmap.

Future Directions

- Dropbox on toasters. File sharing on toasters will be really big.
- They see folders as a unifying metaphor for storing, organizing, and accessing data in the cloud and on any device, anywhere, anytime.

Related Articles

- Hackers News Thread
- Dropbox Blog
- Slidedeck for the Talk
- Dropbox - Startup Lessons Learned by Drew Houston.
