FULL-SCALE BIG COMPUTE SYSTEM FOR PHARMACODYNAMICS

Summary

Our Customer had mathematical models of the in vivo behavior of their patented, FDA-approved molecule, and wanted a detailed understanding of the molecule's effects on individual patients to further increase clinical effectiveness and reduce side effects. Each run of the complex dynamical systems model required substantial CPU time to numerically solve the differential equations and simulate time-series results. They also needed hundreds or thousands of runs per patient using a Monte Carlo approach to account for intra-patient variability across a range of dimensions. Multiplied by thousands of patients, millions of CPU-intensive runs were needed. Finally, to satisfy regulators, our Customer wanted demonstrable correctness at the source-code level as well as in the clinical trial.

The Customer had exhausted its internal IT resources and had been frustrated by the lack of progress from its normal external vendors. They turned to FP Complete for a comprehensive solution.

Name: US-based Fortune 500 company

I**ndustry/Sector**: Pharmaceuticals

Project Type: Regulated software medical device

Technology Used: Amazon Cloud, Redis, Terraform, Docker, Kubernetes, Haskell

The Project Requirements

- The project would need large, but extremely variable compute capacity across many months.
- The Customer had concerns with clarity and maintainability of their Matlab and C++ code, and wanted to do better for regulators with their own productivity.
- Huge compute volumes put traditional slower tools like Python out of the question; a near-supercomputer level of performance was desired.
- The Customer had been let down by their large IT providers on scalability and maintainability, as well as on their scientific and mathematical knowledge level.

The Solution

Our solution was to deploy the model on a high-performance cloud of multiple 36-core virtual compute servers, automatically adding machines according to the workload requirements. We used our High-Performance Computing (HPC) system based on Redis to distribute computations transparently across all of the machines, creating very large computing power.

To get the most out of these big computers, we helped the Customer improve the model's concurrent performance using Haskell for high-performance parallel programming.

To accelerate experimentation, we configured container software including Docker and Kubernetes to automatically bring up whole clusters — so test, research, and production runs could each have their own systems on demand running different versions of the model. Each cluster was put into its own Amazon Private Cloud (VPC) and four logically distinct subnets, each replicated for scalability and fault tolerance:

- a Web API front end using Yesod,
- a message queue work allocation middle tier using Redis,
- a distributed compute tier running many instances of the parallel mathematical model,
- and a system management & monitoring tier capable of creating new compute machines on demand for auto-scaling, as well as logging and other control tasks.

New Challenges for FP Complete

During the project, it became clear that the Customer might need to bring in Personal Health Information (PHI) obtained from various existing databases. PHI is regulated under privacy laws such as HIPAA and use of such data—or even granting access to those databases—could greatly increase operational complexity and the need for regulatory oversight. To reduce this, we modeled the external web API to use depersonalized information. This allowed users to send only anonymized data into the online system, which returns computed results based on anonymous requests and does not retain identifiable patient data.

The system became so popular within the Customer's R&D group that soon there were competing demands for many versions of the system, trying out various new analysis techniques across various new datasets. This created a high DevOps workload, and the queue to deploy new systems began to slow potential new research work. To address this newfound need, we implemented containerization using Docker and automation using Kubernetes, allowing the creation of complete test systems on demand in the cloud.

Conclusion

The system massively increased the throughput of the Customer's pharmacodynamic model, scaling it from a desktop implementation to a 360-CPU virtual supercomputer available as a Web service. This delivered huge amounts of computation to the R&D team, completing ultra-detailed dynamical and probabilistic analysis of a large set of clinical data. Access to so many runs on demand enabled the team to identify important mediating factors that were never before known to clinicians. They used this knowledge to improve the model and used on-demand cloud deployment to test and retest improved models in a very short cycle time. The completed and scaled-up system achieved significant predictive power, far exceeding any previous predictive model of this molecule's effects on individual patients. The resulting software is so safe and effective, and developed with FP Complete under such strong engineering controls, that the Customer is preparing a regulatory submission to use it in a clinical medical device.