

Abstract

The Broad Firehose pipeline was born of the desire to systematize analyses from

The Cancer Genome Atlas (TCGA) pilot phase and scale their execution to the

dozens of remaining diseases to be studied. To realize this goal, while retaining

flexibility to incorporate emerging algorithmic advances, it was necessary to

rapidly evolve research codes, data processing scripts, and infrastructure into

mature components that reliably process terabytes of data on a monthly basis; all

operation. It has led to our view of Firehose as a virtual data factory, subject to

the same production constraints of timeliness and reliability under which

physical factories function. We review the substantial progress made in several

key areas, including semantic clarity, robustness, transparency, pinpoint control

and easy extensibility through scripting, and scalability through automation and

We discuss how these improvements have not only allowed us to fulfill the

original aim of providing a monthly package of standard analyses results, but also to create additional value for the TCGA. Three examples are our twicemonthly standardized data packages, opportunistic runs for analysis working

groups, and our positive influence on other key processes such as mutation data

Here we detail how that process unfolded in the first year of Firehose

towards the additional aims of high transparency and scientific credibility.

parallelism

flow

Engineering Firehose: The First Year



Daniel DiCara, Gordon Saksena, Raktim Sinha, Rui Jing, Douglas Voet, Michael Noble

Broad Institute of MIT and Harvard, Cambridge

Http://GDAC.BroadInstitute.org

Robustness: Testing via Bamboo Continuous Integration

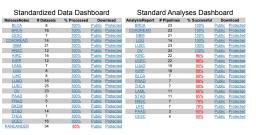
Plan	Build	Completed	Tests	Reason	
GDAC Ingestor	260	3 hours ago	26 passed	Manual build by Daniel DiCara	۵ 😭 化
Module Bam Realign BWA	2 #75	5 months ago	10 passed	Dependant of CGA-NB-81	۱۶
Module Create Merge Data Files SDRF	2 #20	1 day ago	5 passed	Updated by Daniel DiCara	۱۶ 😥 🕥
Module Iterative Scatter Gather Test	3 #57	5 months ago	3 passed	Dependant of CGA-NB-81	۱۶ 🛠
Module PVCA Aggregator	2 #14	4 months ago	3 passed	Updated by Daniel DiCara	۱ 🖌
Pipeline GISTIC 2	366 #366	19 hours ago	21 passed	Updated by Chip Stewart	۱۶ 🛠
Pipeline Mutation Significance	() #374	19 hours ago	2 of 6 failed	Updated by Chip Stewart	۱۶
Pipeline PARADIGM	2 #172	1 month ago	19 passed	Manual build by Daniel DiCara	۱۶ 🛠
Pipeline PVCA	2 #101	1 month ago	12 passed	Manual build by Daniel DiCara	۱

Scalability: Parallelized Mirroring and Dicing

Several process improvements have been made to improve the robustness of our work. First and foremost, a testing infrastructure has been developed. This infrastructure consists of a Bamboo Continuous Integration server and utilities for efficiently producing tests. It ensures that code under development is functioning properly and producing accurate results on a regular basis. And it allows our GDAC to quickly deploy algorithmic updates to our Analysis Pipeline in a dependable fashion. Second, our test coverage has increased significantly. As a consequence, it is simple to gauge the health of our GDAC pipelines via a dashboard in Bamboo that displays testing status. Third, logging has been significantly improved to facilitate reporting, diagnosing, and addressing bugs when they occur. Finally, an extensive effort to manually examine results to ensure their accuracy has been engrained in our week-toweek activities

Robustness

Transparency: Website & Dashboards



Pinpoint Control and Extensibility

Pinpoint control has been enabled via improved Firehose APIs. This has allowed the quick and easy generation of scripts to modify, invoke, and monitor Firehose workspaces. It has also enabled the creation of dashboards on a regular basis to inform both internal and external users about the state of the Broad GDAC standardized data and analyses runs. In terms of extensibility, improved Firehose APIs have allowed us to extend the functionality of Firehose and rapidly prototype new features without modifying the Firehose source code. Our FIrhose Service Selector (FISS) tool is a good example. Repetitive operations and combinations of common API calls can be easily invoked via FISS. These use cases can ultimately inform and guide the development of new Firehose features

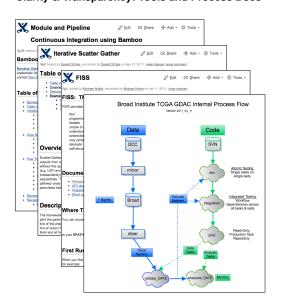
Scalability

In terms of scalability, multiprocessing has been added in a number of places to make maximal use of compute resources. Furthermore, use of cron has enabled the scheduling of jobs at times when compute resources are typically under lighter load (i.e. nighttime and weekends). Finally, improving LSF usage by designing software that maximizes the efficiency of our farm has been a key priority.

Conclusions

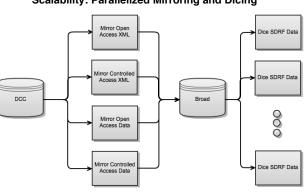
The Broad GDAC has made substantial progress in improving the efficiency and reliability of our Firehose data ingestion, standardized data, and standard analyses activities. This was achieved by applying solid software engineering principals to not only our codes, but also our processes. Our efforts focused on five key areas: clarity, transparency, robustness, pinpoint control and easy extensibility, and scalability. These improvements have facilitated the ingestion and processing of terabytes of data on a regular basis with minimal human interaction. Furthermore, they have improved our ability to add new data types and algorithms to our workflows in a quick and reliable fashion. Finally, these enhancements have allowed us to keep pace with the constantly increasing volume of data

Clarity & Transparency: Tools and Process Docs



Clarity & Transparency

In terms of clarity, a number of code enhancements have been made to abstract business logic out of procedural code spread across multiple files into concise classes and methods. This has facilitated greater code reuse, exposed bugs, and increased transparency. Furthermore, hard-coded features pervasive throughout some of our code have been placed in configuration files. This permits the overriding of default behavior in a non-invasive fashion for testing purposes or otherwise. Finally, and most importantly, we have significantly increased documentation of not only our software, but also our processes as well



Scalability: Workflow Automation

