http://www.geant4.org

Geant4 on Azure using Docker containers

Andrea Dotti (adotti@slac.stanford.edu) ; SD/EPP/Computing

Outlook

Motivation/overview

Docker + G4

Azure + G4

Conclusions

Motivation/overview

A valid alternative to our traditional systems (grid and local batches queue)

The use of containers can greatly improve portability (e.g. SuperComputers, batch queues, grid, cloud)

Motivation

Questions:

1. Can we use public clouds to integrate GRID and local batch queues for Geant4 validation workloads?
2. Can we develop a pipeline that does not lock-in us with a specific vendor?
3. Is it worth the effort?

Partial support from Microsoft Corporation, Azure4Research grant in Life Science track: Geant4 Medical Simulation Benchmarking Group

1

Develop a Geant4 Application

Write a Dockerfile and a `runme.sh` script

docker

Build and publish an image

2

Copy to Azure storage G4-DBs file (once)

Azure Batch

3

Write Azure batch configuration files

Submit jobs to Azure batch

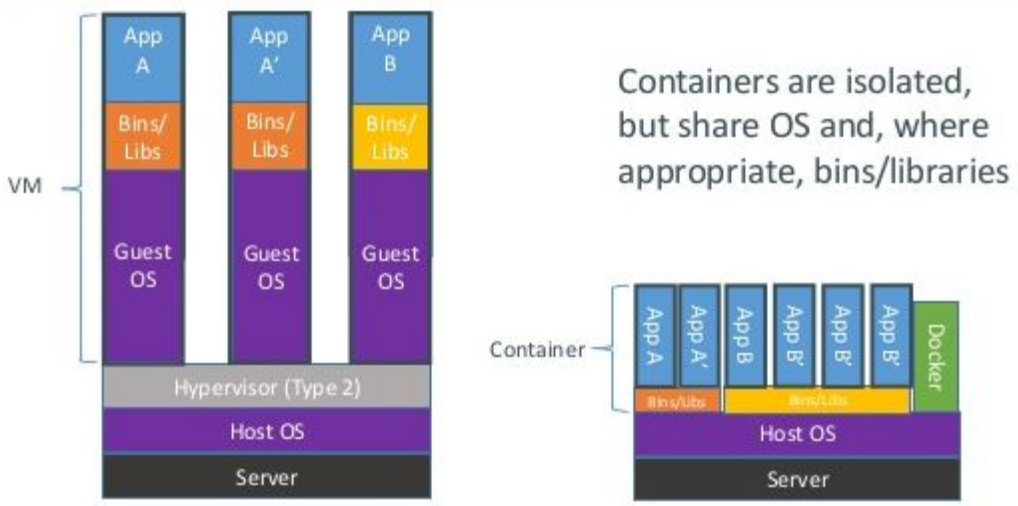
4

Get back output data

Docker + G4

Docker as a Geant4 deploy environment

Docker as a Geant4 deploy environment



Containers:
Lightweight VMs that share kernel with the host OS

Geant4 & Docker

Few design principles (based on ecosystem best experiences):

1. one application per image
2. provide G4 DataBases as docker volume
3. output stored on a docker volume (shared with host)
4. provide layering of images and use docker-tags to identify G4 versions (e.g. andreadotti/geant4:10.3.p01)
5. separate development images (G4-SDK) and Runtime. It's an optimization, keep image size small

Docker use in HEP:

@CERN for CI/CD on Gitlab
@Computing centers for local batch queues (e.g. LSF@SLAC)
@Super-Computers (e.g. NERSC@Berkeley)
Interest from LHC experiments for GRID

Interest from science community driven by:

Docker technology is open-source
Well integrated with many other services (git{hub,lab}, CI/CD, …)

Minimal Example

Try out simplified calorimeter (with a default macro):

```
docker run -v $PWD:/output \
       andreadotti/geant4-val-sc:10.3.p02-10.3.2-data
```

Minimal Example

Try out simplified calorimeter (with a default macro):

```
docker run -v $PWD:/output \
       andreadotti/geant4-val-sc:10.3.p02-10.3.2-data
```

Results in container directory `/output` will appear into current local directory

Minimal Example

Try out simplified calorimeter (with a default macro):

```
docker run -v $PWD:/output \
       andreadotti/geant4-val-sc:10.3.p02-10.3.2-data
```

Repository name (on docker hub) and application name

Minimal Example

Try out simplified calorimeter (with a default macro):

```
docker run -v $PWD:/output \
       andreadotti/geant4-val-sc:10.3.p02-10.3.2-data
```

tag: Geant4 version, application version. If `-data` is present image contains G4 databases

Note: tag naming scheme is a convention, not enforced by docker

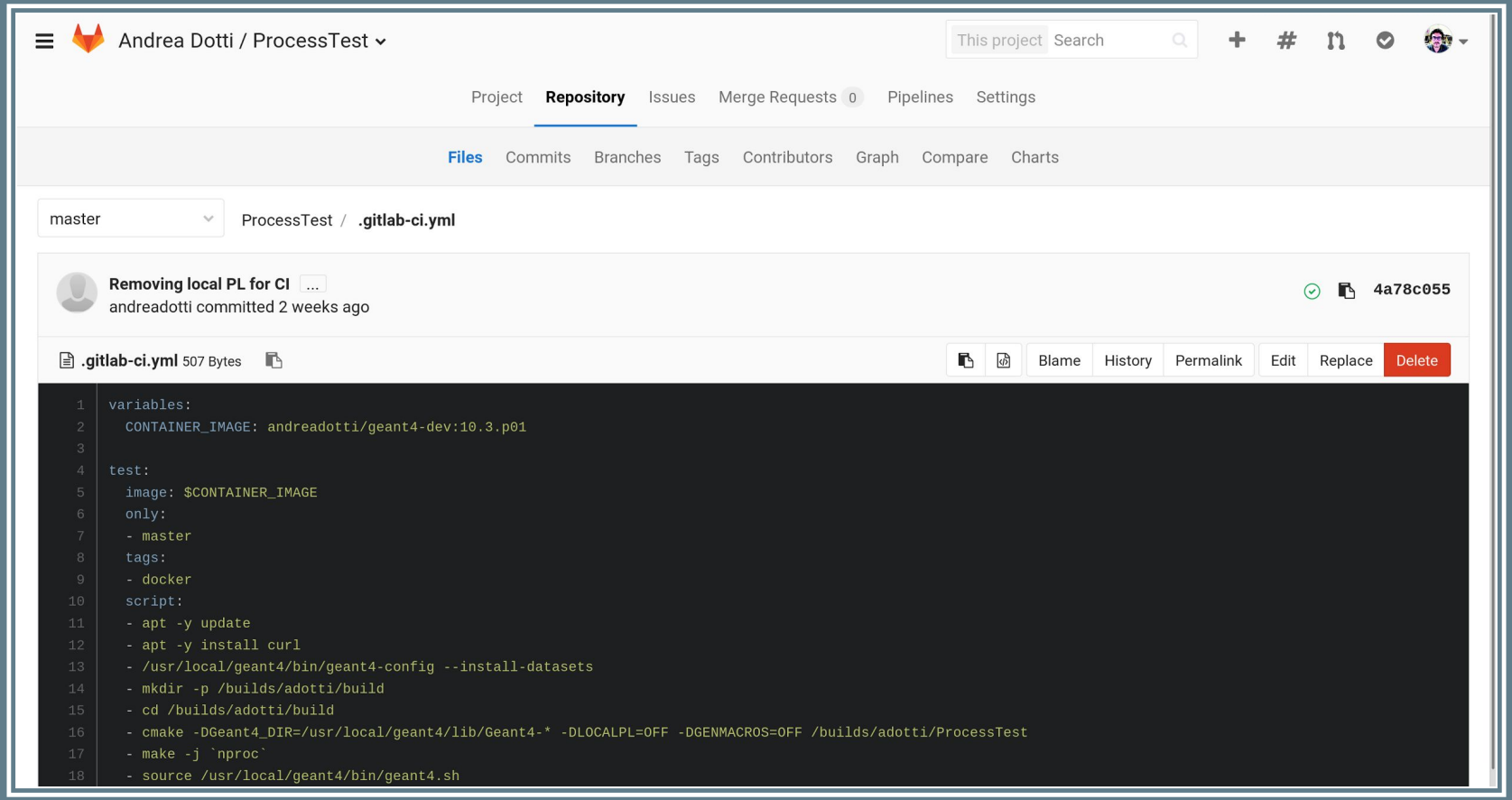
Experience so far

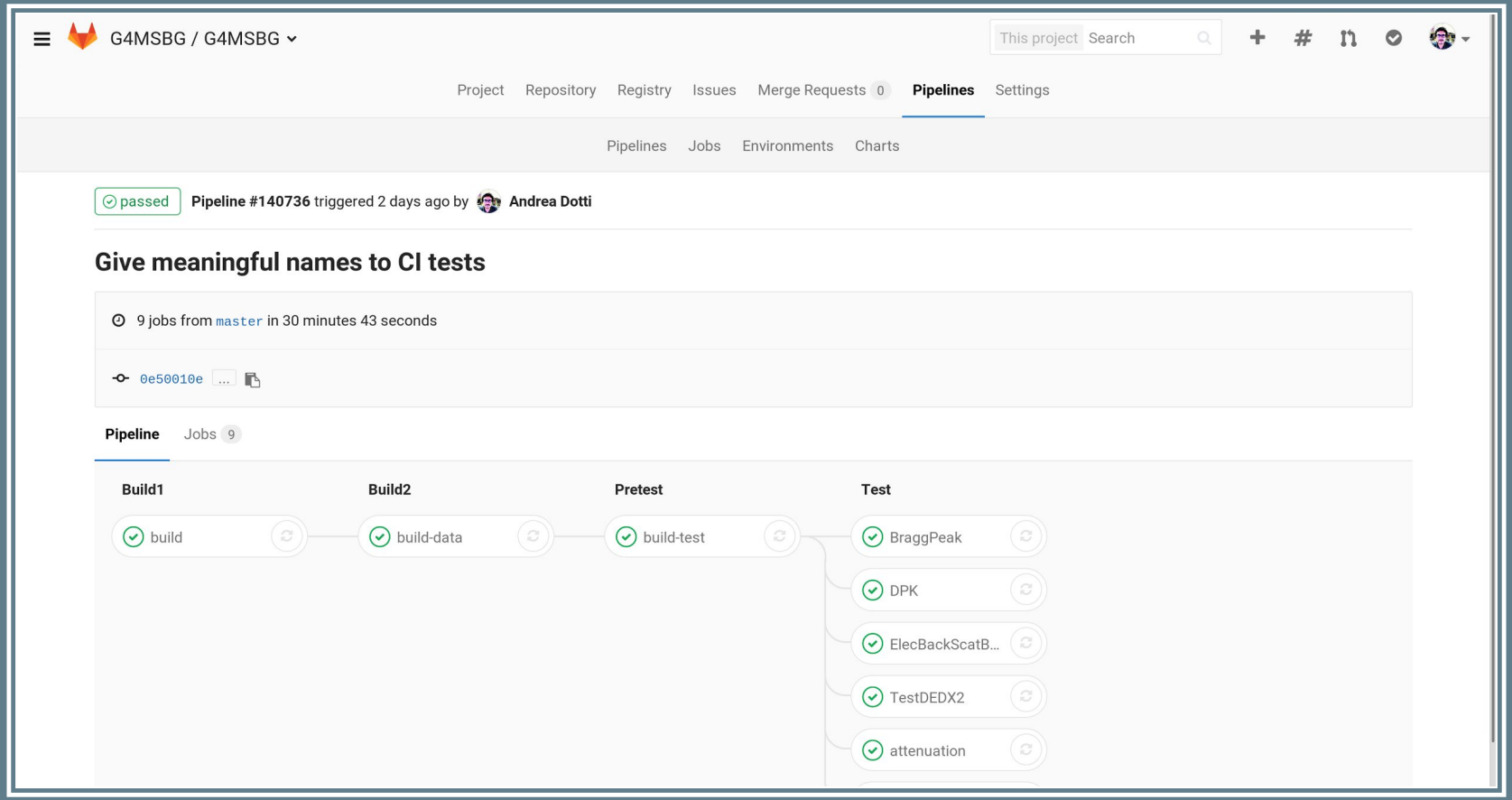
Successfully integrated three set of applications:

1. Medical Benchmarking group
2. ProcessLevel Testing
3. SimplifiedCalorimeter

1. and 2. are hosted at gitlab.cern.ch, the same infrastructure is used for CI/CD: at each `git push` to the repository docker images are built and tests are run

For Medical Benchmark Docker image for Azure testing is automatically created


Andrea Dotti / ProcessTest
This project Search
Project Repository Issues Merge Requests 0 Pipelines Settings
Files Commits Branches Tags Contributors Graph Compare Charts
master
ProcessTest / .gitlab-ci.yml
Removing local PL for CI ...
andreadotti committed 2 weeks ago
4a78c055
.gitlab-ci.yml 507 Bytes
Blame History Permalink Edit Replace Delete
variables:
CONTAINER_IMAGE: andreadotti/geant4-dev:10.3.p01
test:
image: $CONTAINER_IMAGE
only:
- master
tags:
- docker
script:
- apt -y update
- apt -y install curl
- /usr/local/geant4/bin/geant4-config --install-datasets
- mkdir -p /builds/adotti/build
- cd /builds/adotti/build
- cmake -DGeant4_DIR=/usr/local/geant4/lib/Geant4-* -DLOCALPL=OFF -DGENMACROS=OFF /builds/adotti/ProcessTest
- make -j `nproc`
- source /usr/local/geant4/bin/geant4.sh


G4MSBG / G4MSBG
This project
Search
Project
Repository
Registry
Issues
Merge Requests 0
Pipelines
Settings
Pipelines
Jobs
Environments
Charts
passed
Pipeline #140736 triggered 2 days ago by Andrea Dotti
Give meaningful names to CI tests
9 jobs from master in 30 minutes 43 seconds
0e50010e
Pipeline
Jobs 9
Build1
Build2
Pretest
Test
build
build-data
build-test
BraggPeak
DPK
ElecBackScatB...
TestDEDX2
attenuation

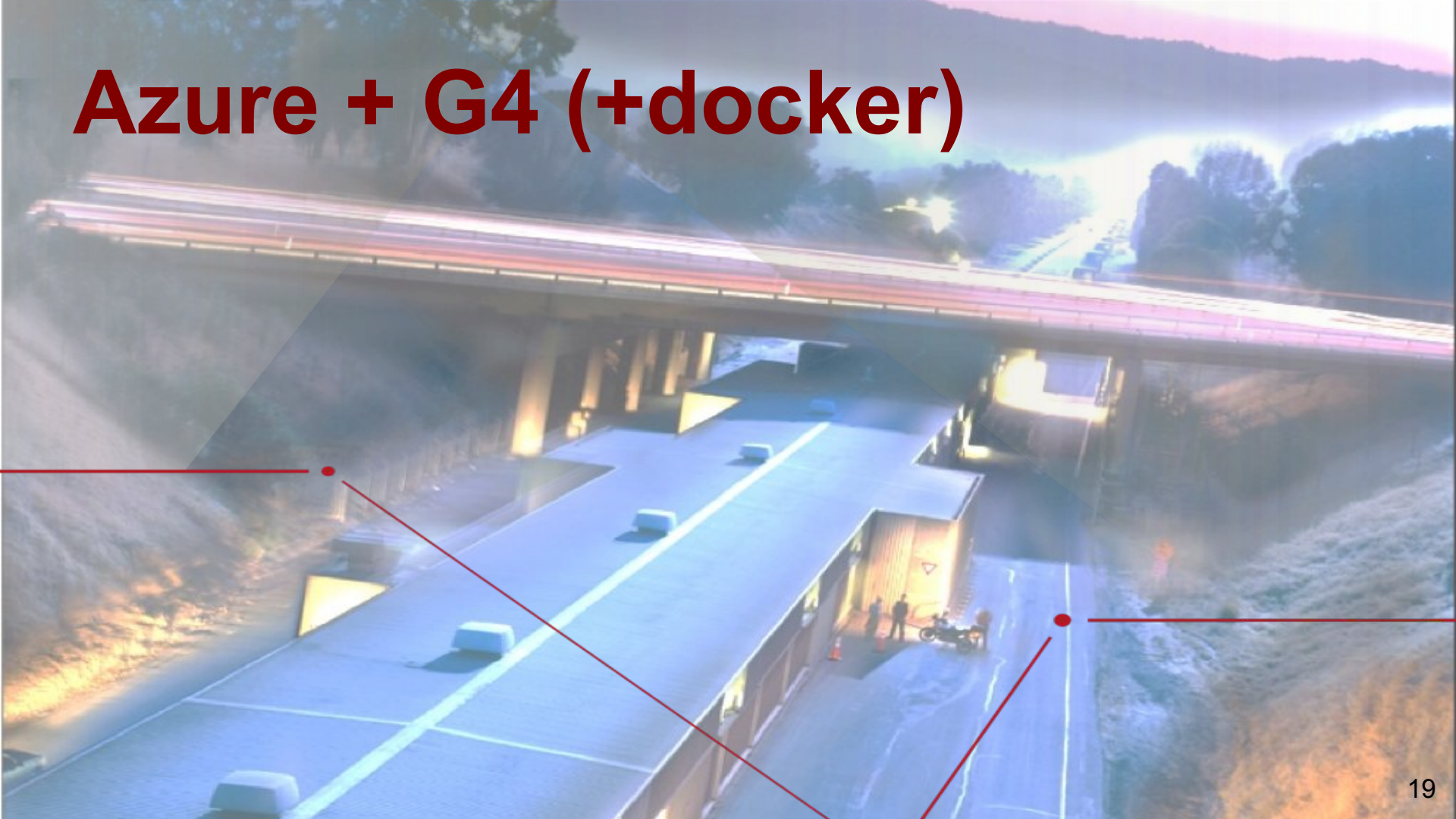
Docker conclusions

Very positive experience so far

Docker made very simple integrate different applications in same environment (never managed with GRID)

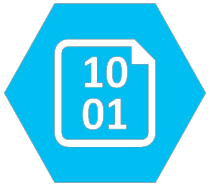
Docker compatible systems are gaining substantial traction even in HEP data-centers. We are “ready-to-go”

Azure + G4 (+docker)

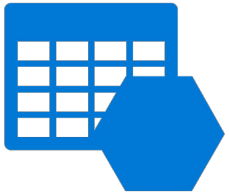


Components

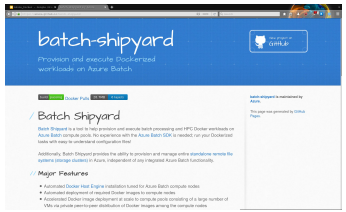
Provides VMs to run jobs: different sizes available



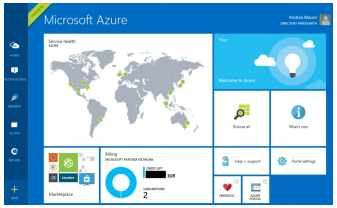
Azure Storage Blob: output of jobs stored here



Azure Storage File Share: G4Dbs are stored here



Azure batch-shipyard: configure and submit docker workloads



Azure portal: web-app to manage, monitor, request support, ...

Blobs: optimized storage for ~large files, accessed through API (similar to CASTOR/EOS/XROOTD)

File Share: read/write random access area, seen as NFS mount (similar to AFS/CVMFS)

Batch-shipyard: does the dirty work, in particular configures I/O from/to Azure (docker image does not depend on Azure)

Workflow

First you create one or more pools: a virtual farm composed of the desired number and type of virtual machines

Then you define one or more jobs. Each job is composed of tasks, they have very similar configuration but run a different application configuration

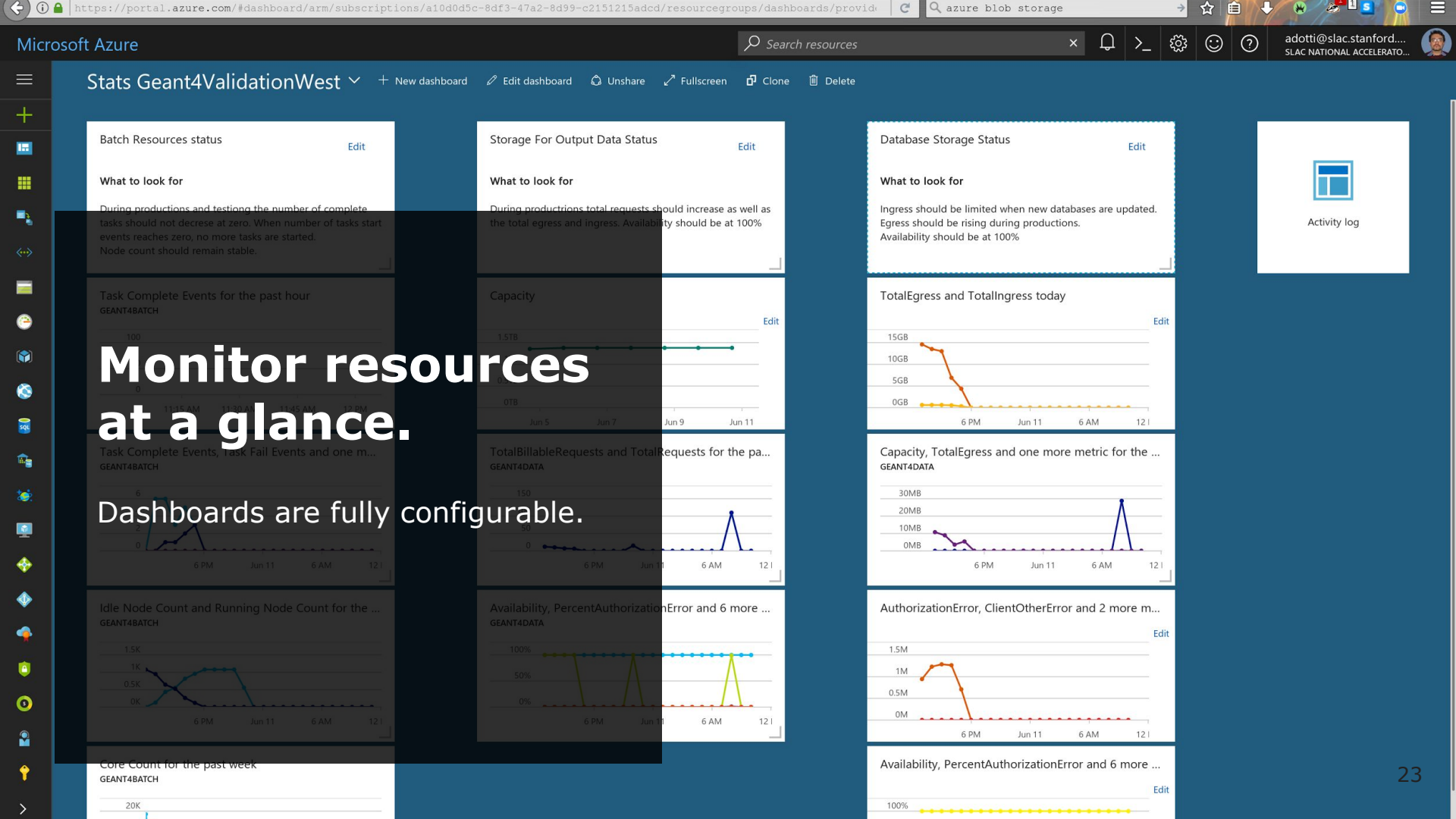
Example: Job 1 is SimplifiedCalorimeter for FTFP_BERT with pi- as primary. Each combination of {calorimeter, beam-energy} is a different task

System is configured via a set of text files in JSON format

az-batch script

Creating the JSON files for Geant4 is tedious and error prone. `az-batch` script can be used to simplify the creation of this file.

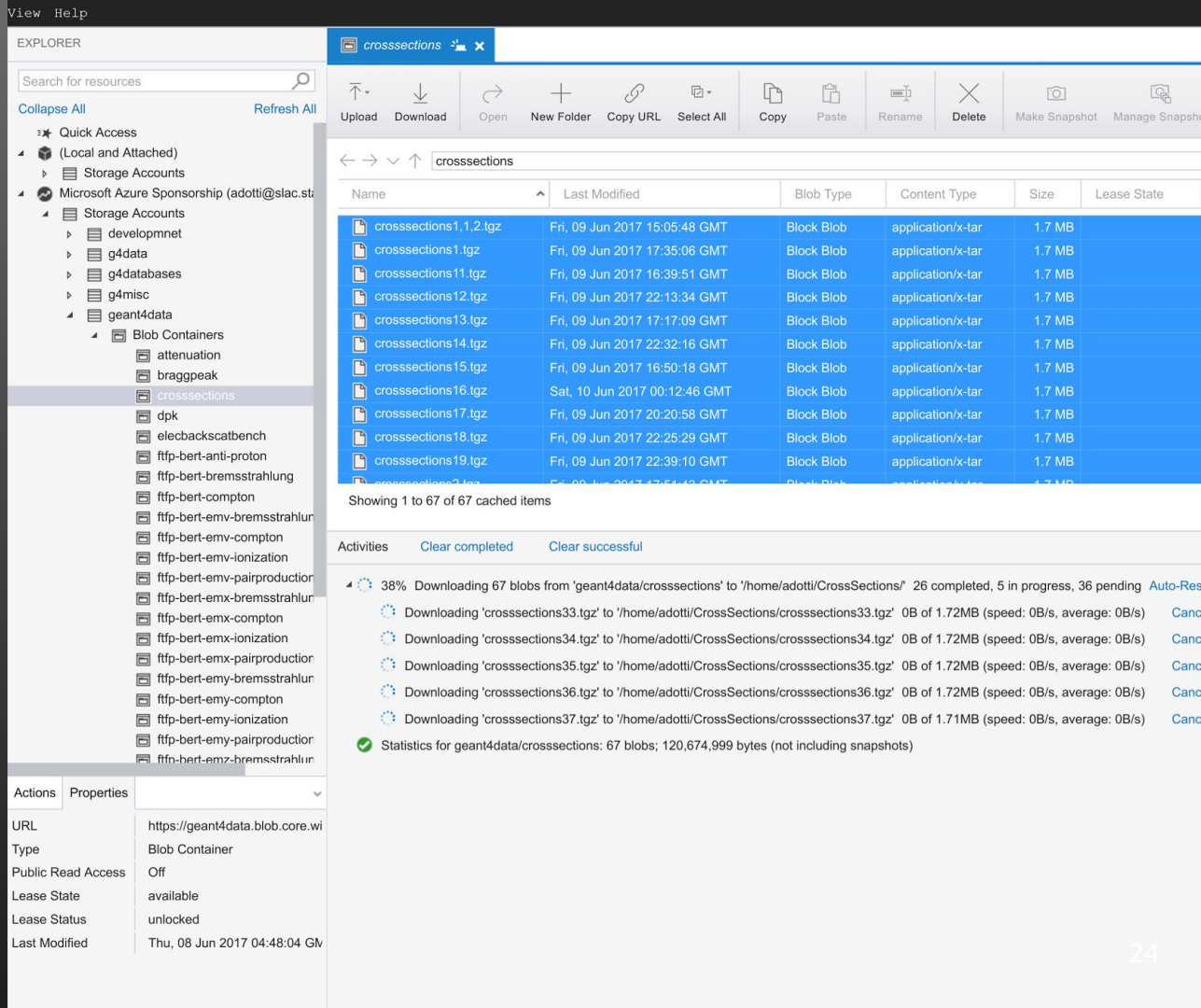
This is the only additional element specific to Geant4 that I had to develop. It’s a bash script that manipulates the `jobs.json` file content based on user input.


https://portal.azure.com/#dashboard/arm/subscriptions/a10d0d5c-8df3-47a2-8d99-c2151215adcd/resourcegroups/dashboards/provid
azure blob storage
Microsoft Azure
Search resources
adotti@slac.stanford....
SLAC NATIONAL ACCELERATO...
Stats Geant4ValidationWest
New dashboard
Edit dashboard
Unshare
Fullscreen
Clone
Delete
Batch Resources status
Edit
What to look for
During productions and testiong the number of complete tasks should not decrese at zero. When number of tasks start events reaches zero, no more tasks are started.
Node count should remain stable.
Storage For Output Data Status
Edit
What to look for
During productions total requests should increase as well as the total egress and ingress. Availability should be at 100%
Database Storage Status
Edit
What to look for
Ingress should be limited when new databases are updated.
Egress should be rising during productions.
Availability should be at 100%
Activity log
Task Complete Events for the past hour
GEANT4BATCH
Capacity
Edit
TotalEgress and TotalIngress today
Edit
15GB
10GB
5GB
0GB
6 PM
Jun 11
6 AM
Task Complete Events, Task Fail Events and one m...
GEANT4BATCH
TotalBillableRequests and TotalRequests for the pa...
GEANT4DATA
Capacity, TotalEgress and one more metric for the ...
GEANT4DATA
30MB
20MB
10MB
0MB
Idle Node Count and Running Node Count for the ...
GEANT4BATCH
Availability, PercentAuthorizationError and 6 more ...
GEANT4DATA
AuthorizationError, ClientOtherError and 2 more m...
Edit
1.5M
1M
0.5M
0M
Core Count for the past week
GEANT4BATCH
Availability, PercentAuthorizationError and 6 more ...
Edit
100%
Monitor resources at a glance.
Dashboards are fully configurable.

File Explorer

Standalone application (for Linux, Mac and Windows) to explore and interact with Azure storage

Alternative to CLI



Jobs and resource utilization

Performed validation of latest 3 Geant4 versions

For each version: ~400k jobs of different CPU-intensity (from 1 minute to 1 hour)

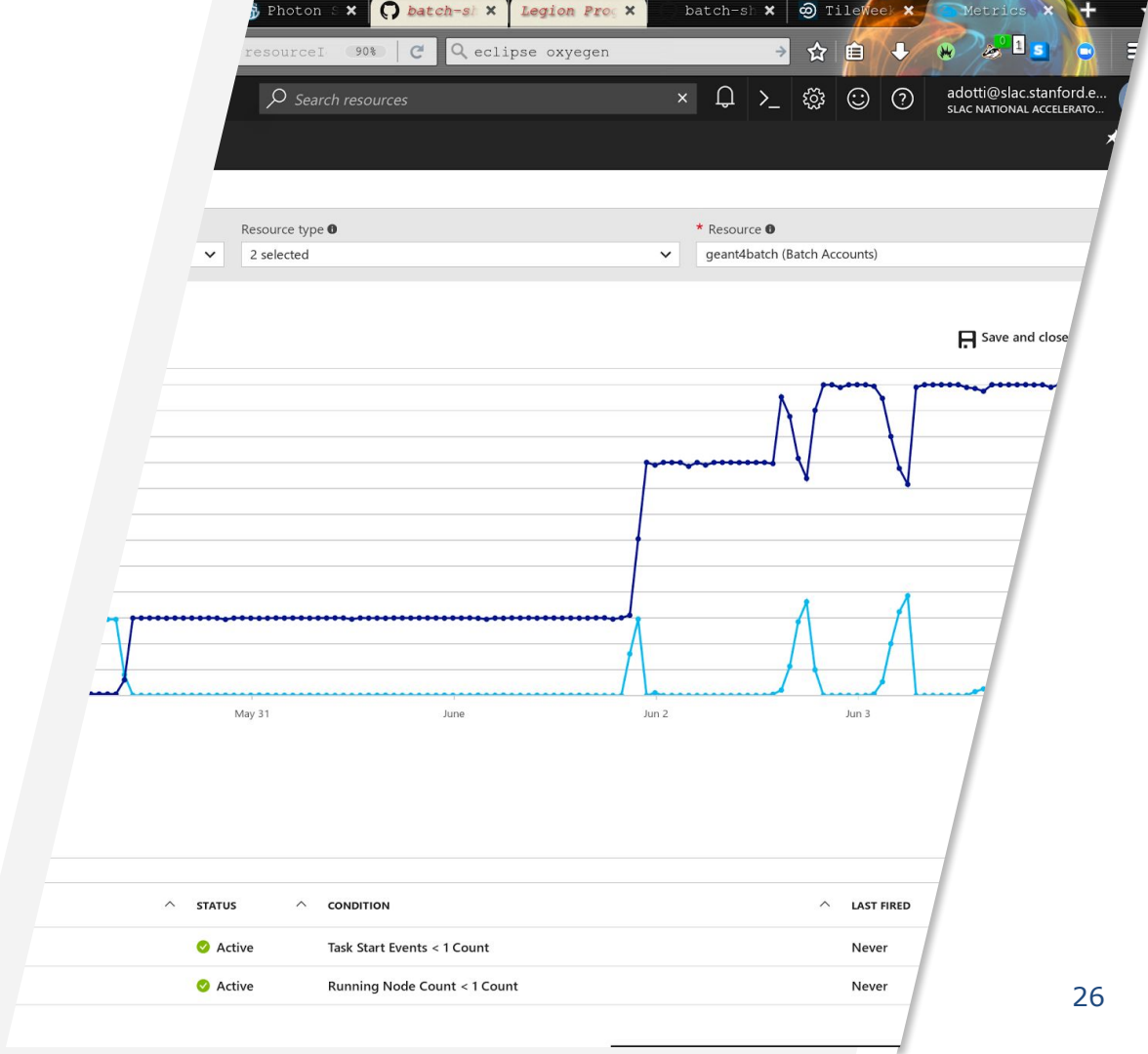
Jobs organized in about 200 jobs with different number of tasks

CPU

200 Standard F4 compute nodes (800 cpus)

Jobs are parallelized via multi-threading, but not via MPI: embarrassingly parallel problem

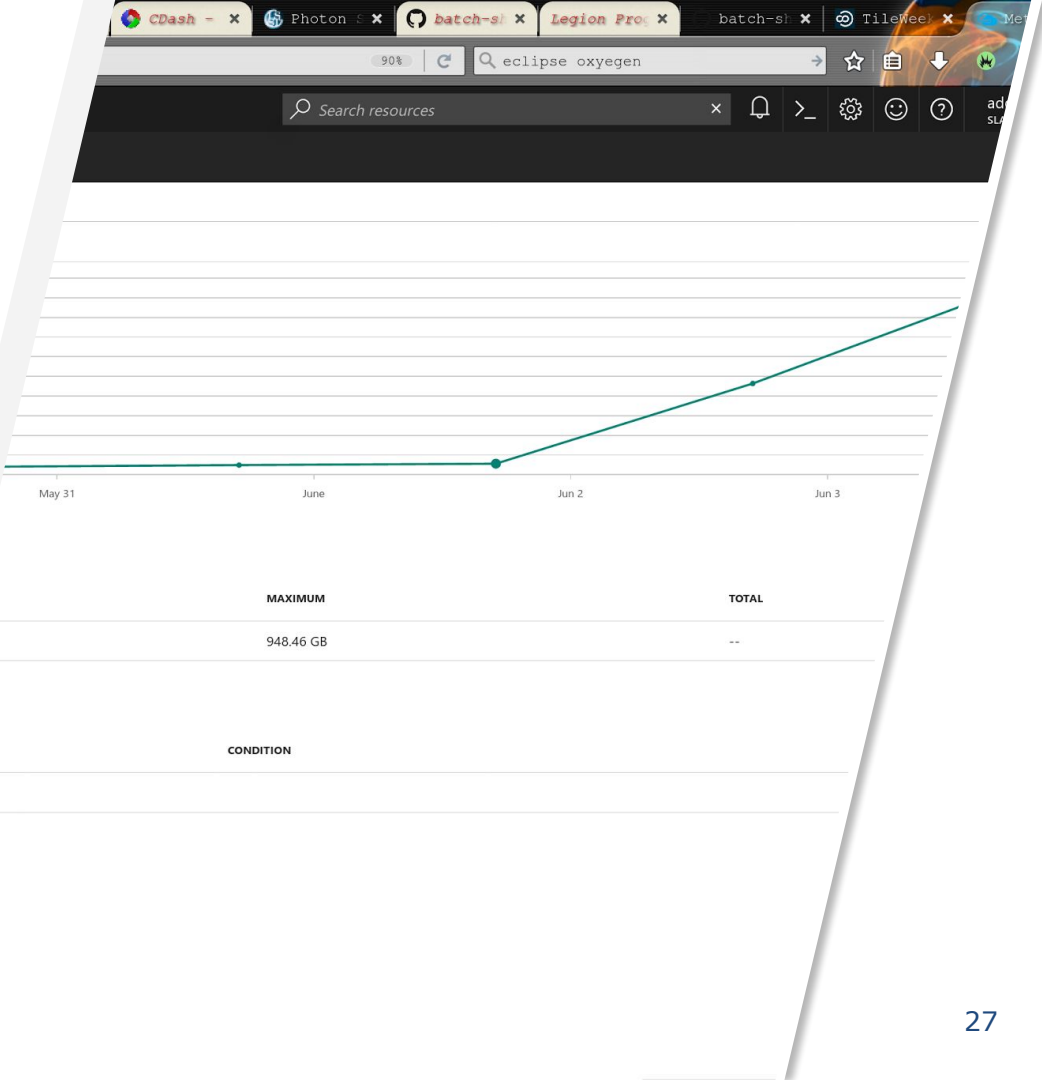
Relatively low memory requirement <8GB



Disk

Each validation produces about 3TB of data

Data will be further reduced



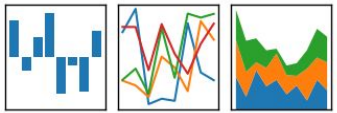
Next Steps

Post-processing of validation results done on Azure:

Fire up VMs with specific ROOT/python code to analyze ntuples

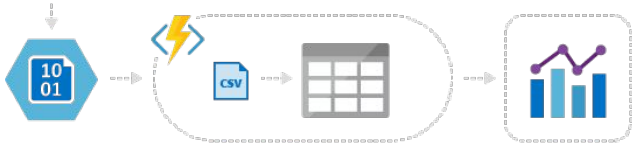
pandas

$$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$$



Investigate use of Azure Functions for part of post-processing:

Conversion of format for final upload to DataBase



Opportunistic computing, jobs can be killed, but cost is very small. Use of checkpointing?

Conclusions: positive aspects

Very positive experience with Microsoft Azure

Docker integration makes portability possible. Azure specific layers are well isolated and can be replaced easily

Azure tools and web-portal are very well designed. CLIs are powerful and extensible

Wonderful support: guaranteed 8 hours response time to tickets 0-24;7/7, they even call you on the phone!

The “new Microsoft” is really impressive: open source code on github, collaborative with developers, interest in scientific research beyond CS, Linux as a first-class citizen

Conclusions: challenges

batch-shipyard JSON based configuration is very verbose. Geant4 typical jobs cannot be manually created

I had to create yet-another-script to manipulate these files. It is not ideal, but it is a small script and simple enough
https://github.com/andreadotti/geant4-val-azure

Differently from GRID, you pay for all these goodies :-) check out
https://azure.microsoft.com/en-us/pricing/calculator/

Back to our initial question

Can clouds replace/extend our GRID/local batch queues jobs?

Definitely easier to use than GRID: I had to develop a single bash script, all the rest is comes from MS with great support
Probably more complex than local batch queue for simple tasks

When docker will be used on GRID and batch systems, it will become even simpler to inter-operate local/GRID/cloud. This was the promise of GRID to be honest…

Geant4 Specific Considerations

We are a very small GRID VO, with ~no manpower for operating it

SimplifiedCalo on the GRID: 70 configurations in few days.
SimplifiedCalo on Azure: 240 combinations in ~48 hours
We have ~100 cores on the GRID. A single ~newish server has typically 20

How many times in the last few years did we rewrite from ~scratch our GRID tools? How many times we failed in adding new applications to the GRID?
Probably the cost of supporting of our GRID system is larger than what we would pay using public clouds.

Geant4 Specific Considerations

We are a very

SimplifiedCal
SimplifiedCal
We have ~10

How many tir D tools? How
many times w
Probably the would pay
using public clouds?

Before we end:

This slide is clearly provocative and an over-simplification of the problem, I know… I want to trigger a discussion

What I really think would be ideal for us is the possibility to integrate transparently different production systems (docker seems the first step). So long live the G4 VO GRID

Said that, the hidden cost of our GRID system should be considered and a serious discussion on its future use could be useful