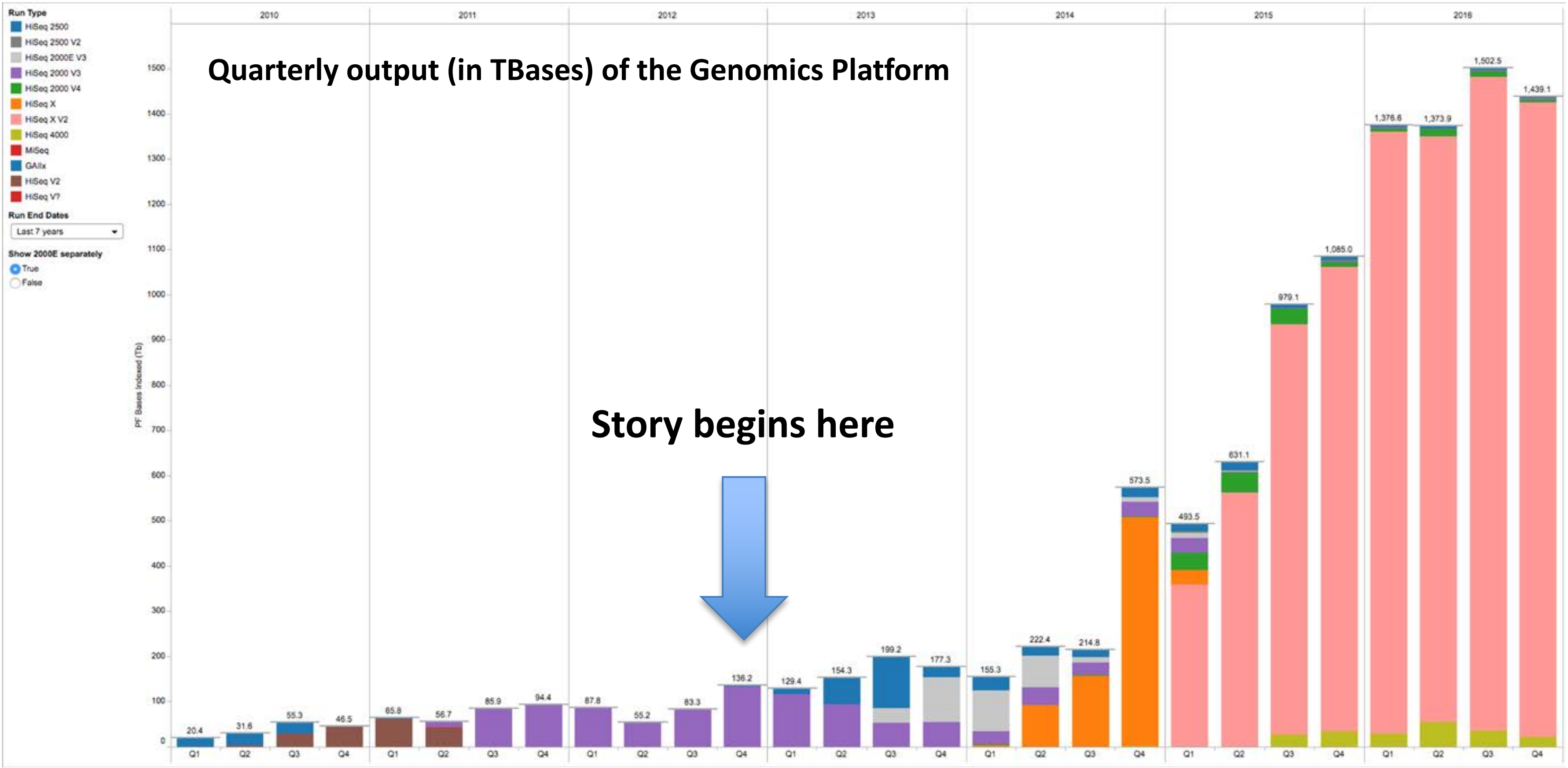
Getting started with WDL & Cromwell

Bioinformatics workflows at any scale

Ruchi Munshi
Data Sciences Platform
Broad Institute

The backdrop: data generation set to explode



Run End Dates: Last 7 years

Show 2000E separately: [✓ True] False

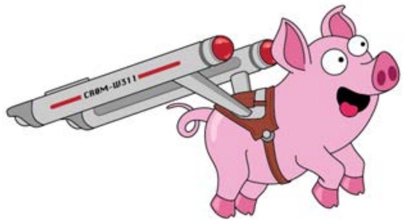
Quarterly output (in TBases) of the Genomics Platform

Story begins here

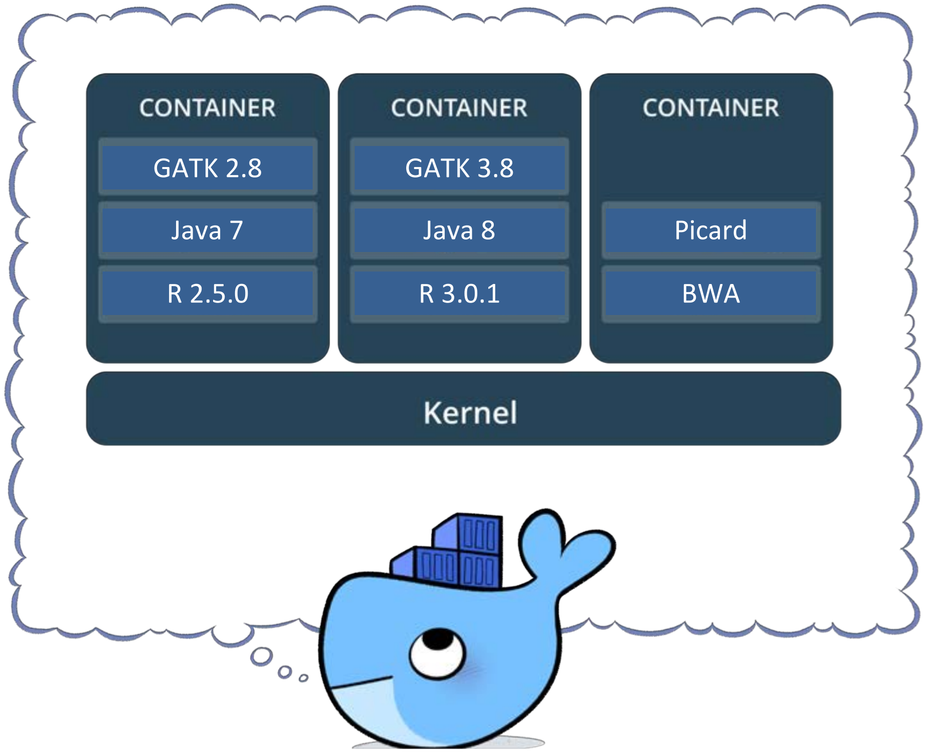
Meet Cromwell & WDL

- Execution engine that can
  - Run on any platform (HPC *and* on Cloud)
  - Seamlessly scale based on workflow needs
  - Provide maximal flexibility for all use cases
  - https://github.com/broadinstitute/cromwell

- Workflow language that humans can read/write
  - Methods developers and biomedical scientists at large
  - https://github.com/openwdl/wdl/



Use containers for portability & reproducibility

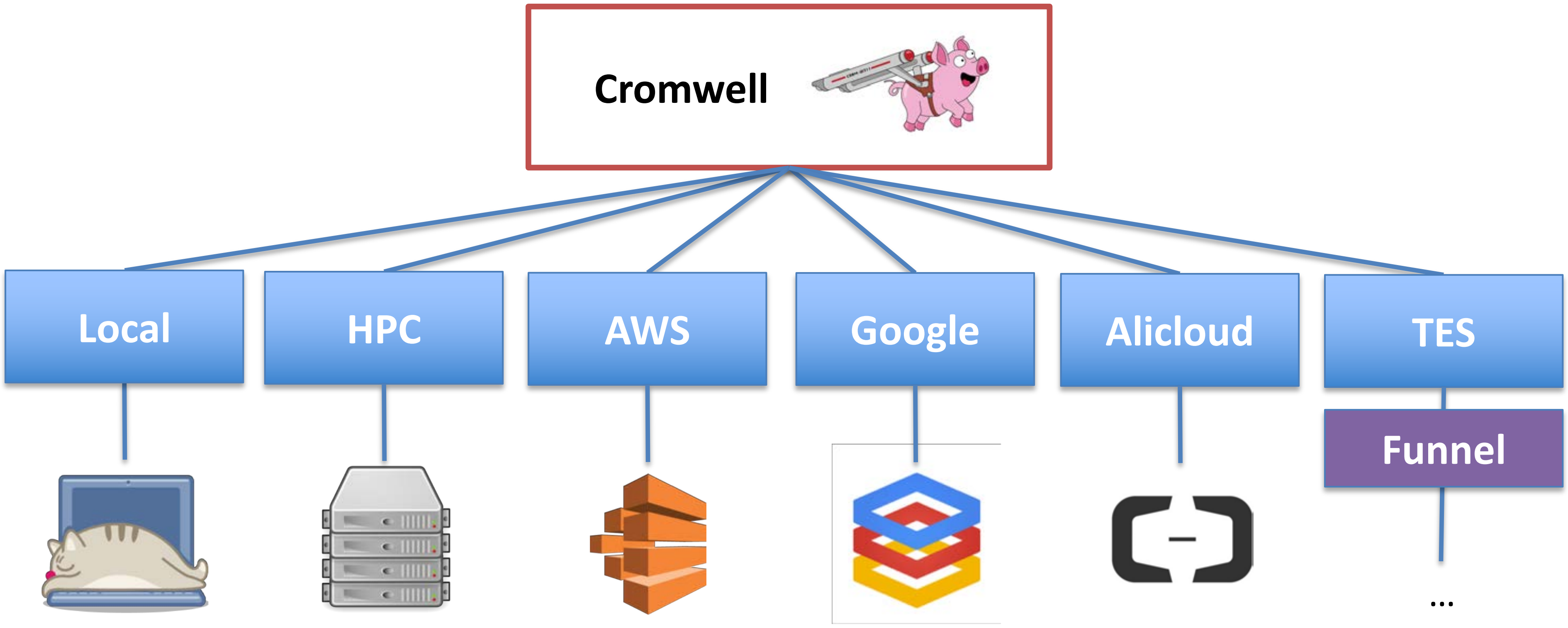


A container encapsulates all the software dependencies associated with running a program

Takes the guesswork out of running workflows on different platforms!

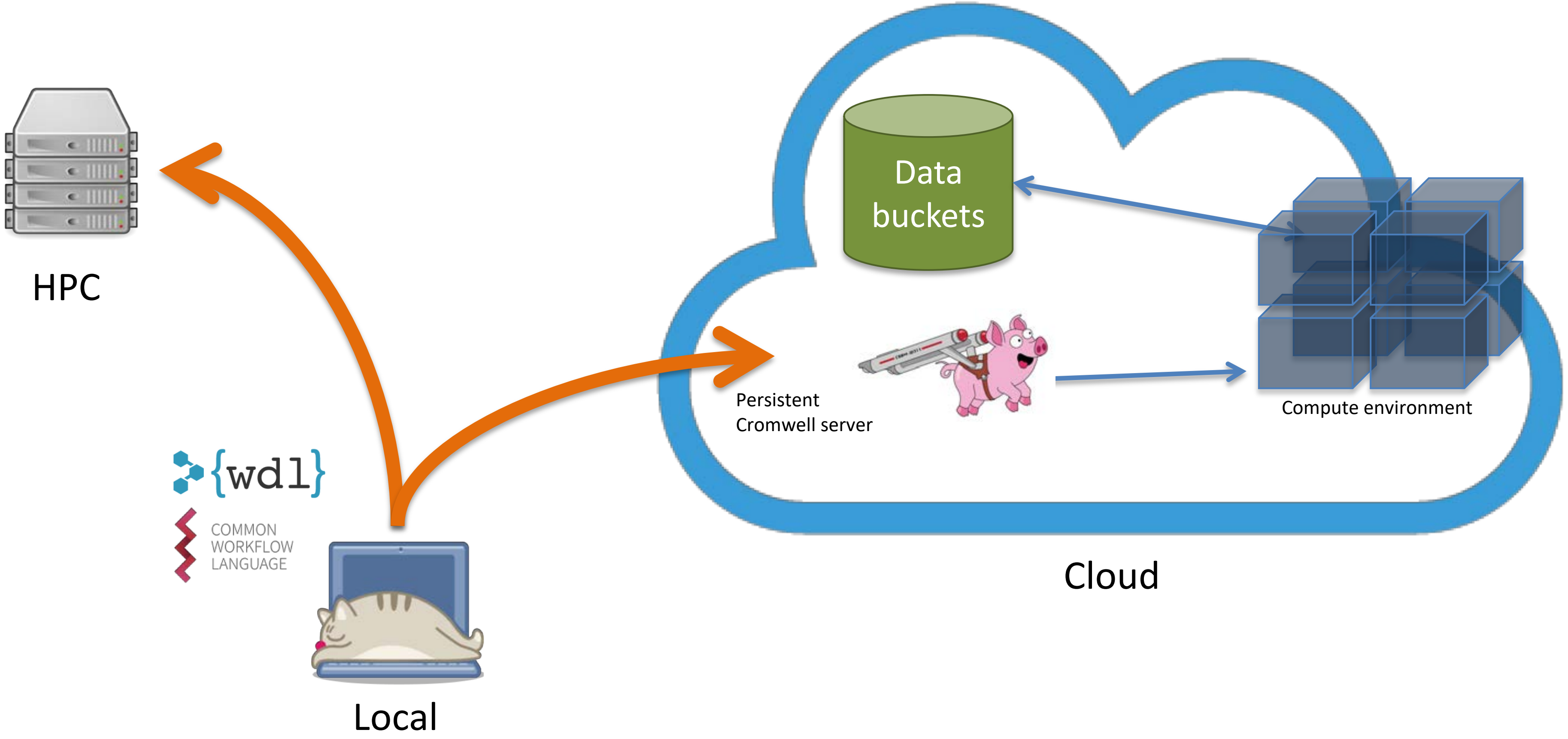
Modified from https://www.docker.com/what-container

Use a workflow execution engine that runs anywhere



https://github.com/broadinstitute/cromwell

Run using HPC **and** Cloud resources!



Two main ways to run Cromwell

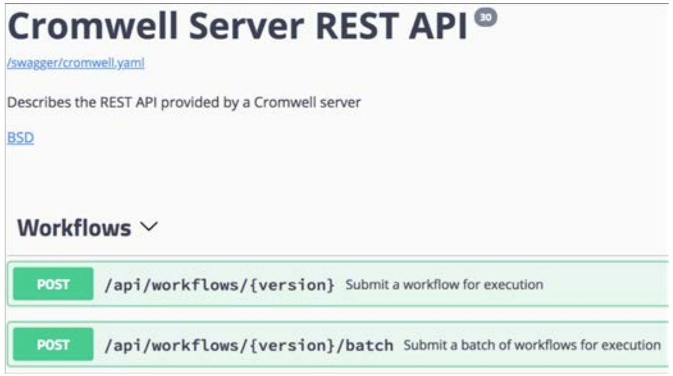
Command Line

- Simple, self-contained command
- Appropriate for independent analysts
- Call Caching

```
java -jar cromwell.jar \
     run hello.wdl \
     hello_inputs.json
```

Server

- API endpoints
- More scalable, appropriate for production environments
- Call caching



Cromwell Server REST API 30

/swagger/cromwell.yaml

Describes the REST API provided by a Cromwell server

BSD

Workflows

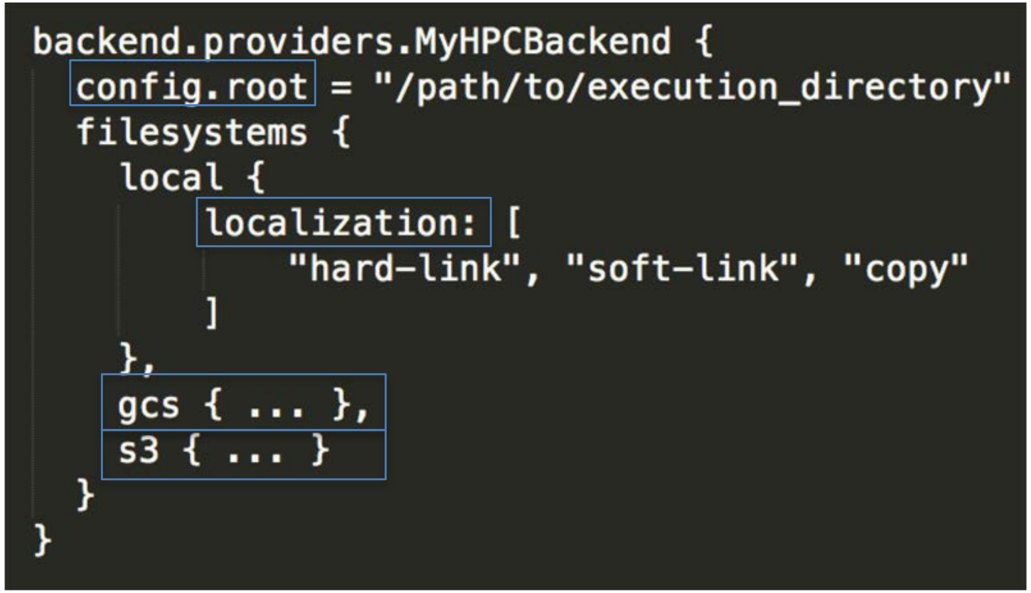
POST /api/workflows/{version} Submit a workflow for execution

POST /api/workflows/{version}/batch Submit a batch of workflows for execution

HPC Backend Configuration

Managing data

- Root directory
- Data handling strategies
- Support for object stores



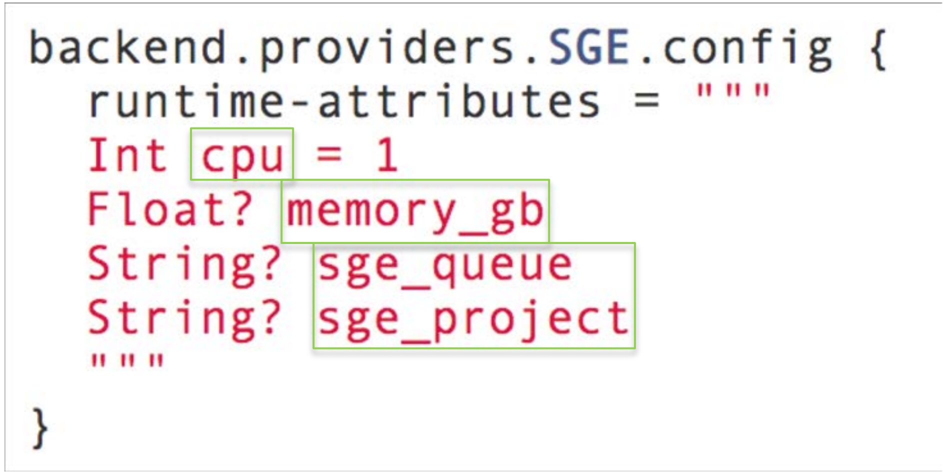
```
backend.providers.MyHPCBackend {
  config.root = "/path/to/execution_directory"
  filesystems {
    local {
        localization: [
          "hard-link", "soft-link", "copy"
        ]
    },
    gcs { ... },
    s3 { ... }
  }
}
```

https://cromwell.readthedocs.io/en/stable/tutorials/HPCIntro/

HPC Backend Configuration

Managing resources

- CPU
- Memory
- Custom attributes



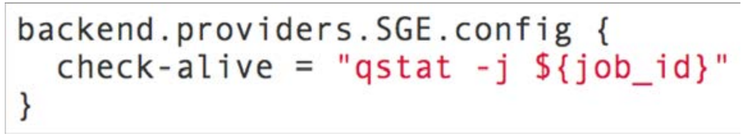
```
backend.providers.SGE.config {
  runtime-attributes = """
  Int cpu = 1
  Float? memory_gb
  String? sge_queue
  String? sge_project
  """
}
```

https://cromwell.readthedocs.io/en/stable/tutorials/HPCIntro/

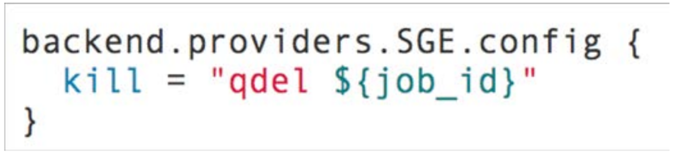
HPC Backend Configuration

Run command

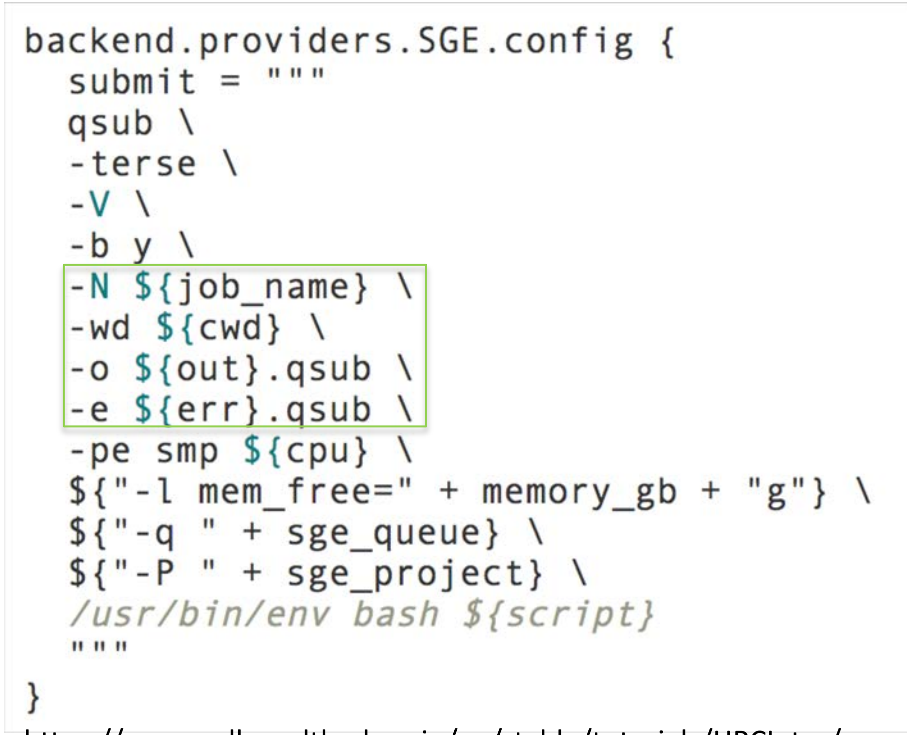
- Built-in variables
- Full flexibility



```
backend.providers.SGE.config {
  check-alive = "qstat -j ${job_id}"
}
```



```
backend.providers.SGE.config {
  kill = "qdel ${job_id}"
}
```



```
backend.providers.SGE.config {
  submit = """
  qsub \
  -terse \
  -V \
  -b y \
  -N ${job_name} \
  -wd ${cwd} \
  -o ${out}.qsub \
  -e ${err}.qsub \
  -pe smp ${cpu} \
  ${"-l mem_free=" + memory_gb + "g"} \
  ${"-q " + sge_queue} \
  ${"-P " + sge_project} \
  /usr/bin/env bash ${script}
  """
}
```

https://cromwell.readthedocs.io/en/stable/tutorials/HPCIntro/

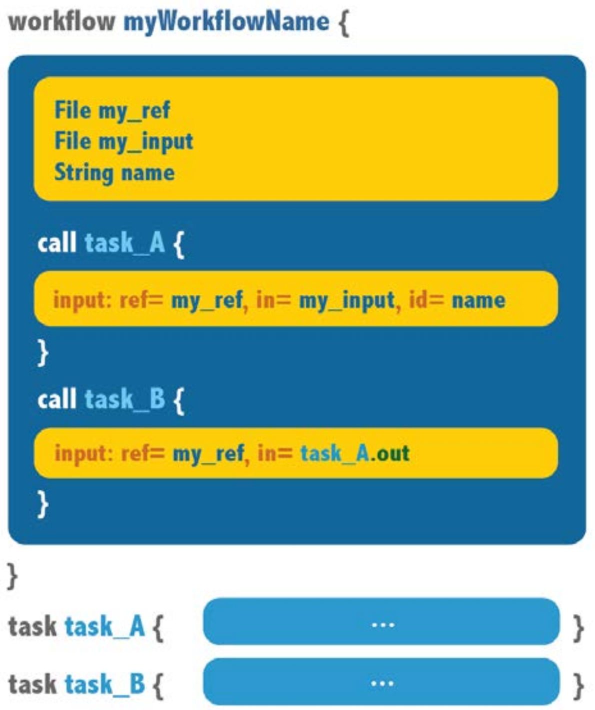
Plenty of workflow solutions to go around

SITUATION: THERE ARE 14 COMPETING STANDARDS.
14?! RIDICULOUS! WE NEED TO DEVELOP ONE UNIVERSAL STANDARD THAT COVERS EVERYONE'S USE CASES.
YEAH!
SOON:
SITUATION: THERE ARE 15 COMPETING STANDARDS.

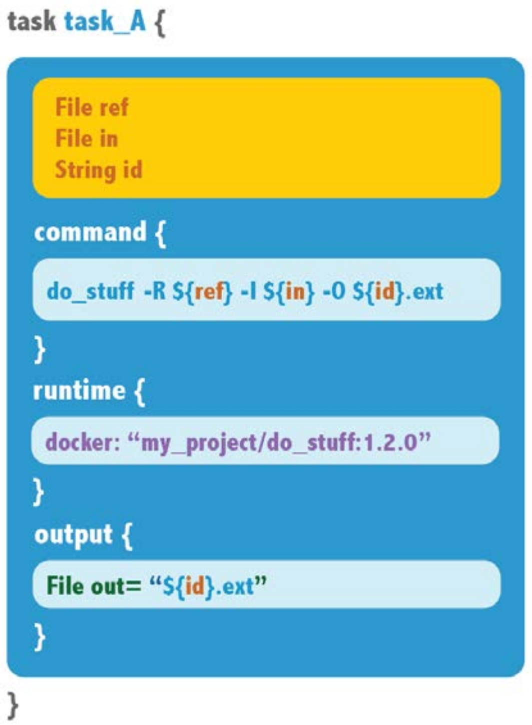
So of course we decided to create a new one.

Randall Munroe, XKCD
https://www.xkcd.com/927/

Workflow description Language (WDL)

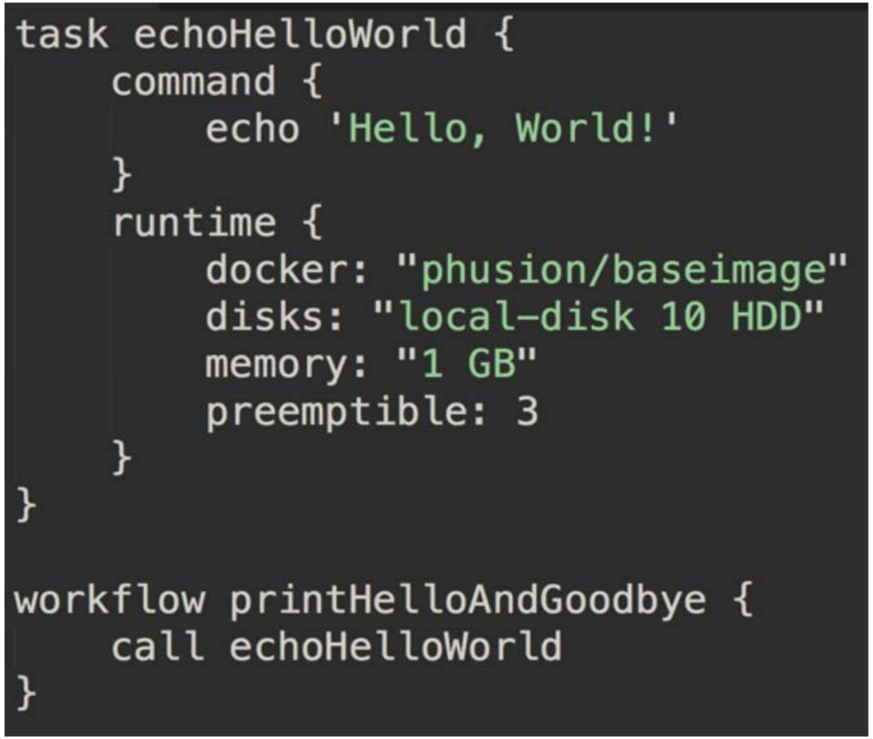


```
workflow myWorkflowName {
    File my_ref
    File my_input
    String name

    call task_A {
        input: ref= my_ref, in= my_input, id= name
    }
    call task_B {
        input: ref= my_ref, in= task_A.out
    }
}
task task_A {          ...          }
task task_B {          ...          }
```

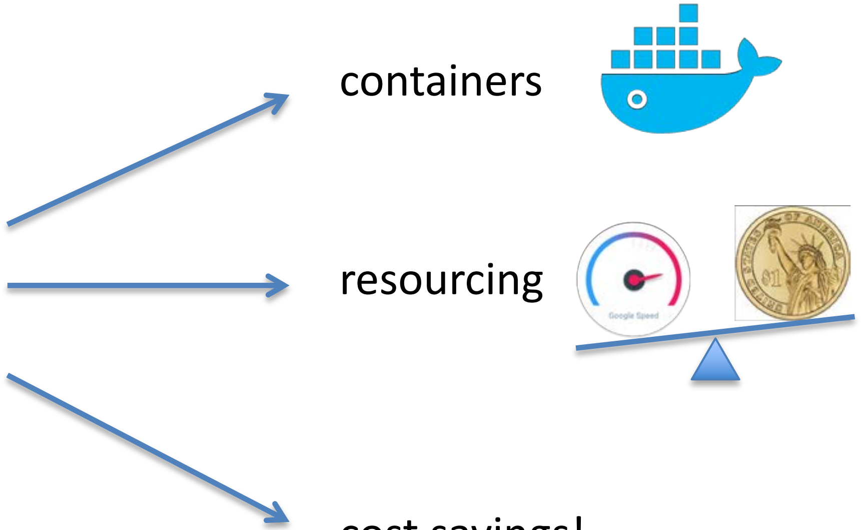


```
task task_A {
    File ref
    File in
    String id

    command {
        do_stuff -R ${ref} -I ${in} -O ${id}.ext
    }
    runtime {
        docker: "my_project/do_stuff:1.2.0"
    }
    output {
        File out= "${id}.ext"
    }
}
```

WDL runtime parameters

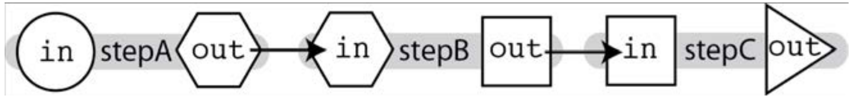


```
task echoHelloWorld {
    command {
        echo 'Hello, World!'
    }
    runtime {
        docker: "phusion/baseimage"
        disks: "local-disk 10 HDD"
        memory: "1 GB"
        preemptible: 3
    }
}

workflow printHelloAndGoodbye {
    call echoHelloWorld
}
```



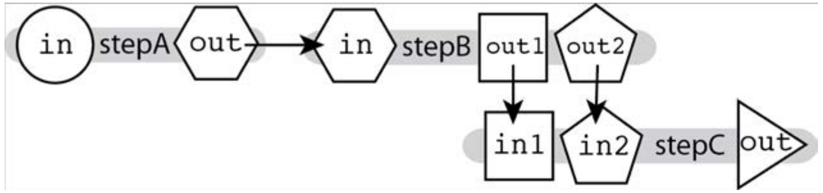
Basic WDL plumbing options

LINEAR CHAINING



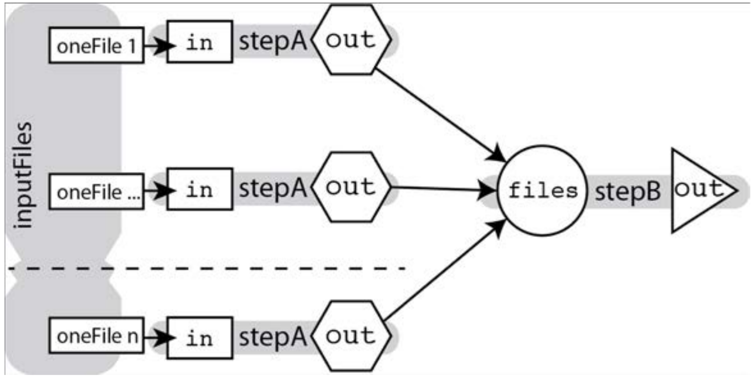
```
call stepA
call stepB { input: in=stepA.out }
call stepC { input: in=stepB.out }
```

MULTI-IN/OUT



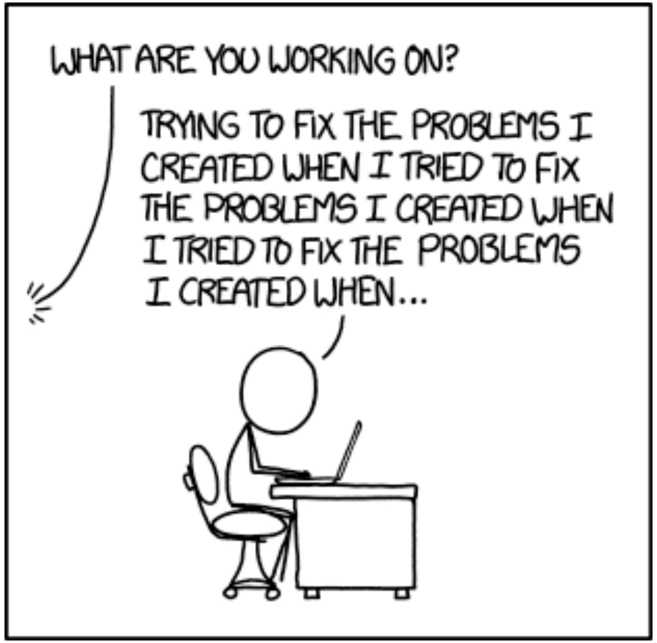
```
call stepC { input :
               in1=stepB.out1,
               in2=stepB.out2 }
```

SCATTER-GATHER

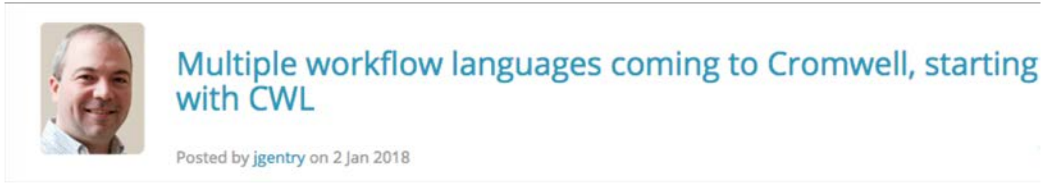


```
Array[File] inputFiles

scatter(oneFile in inputFiles) {
    call stepA { input: in=oneFile }
}

call stepB { input: files=stepA.out }
```

But what about CWL?


WHAT ARE YOU WORKING ON?
TRYING TO FIX THE PROBLEMS I CREATED WHEN I TRIED TO FIX THE PROBLEMS I CREATED WHEN I TRIED TO FIX THE PROBLEMS I CREATED WHEN...

Randall Munroe, XKCD
https://www.xkcd.com/1739/



Multiple workflow languages coming to Cromwell, starting with CWL

Posted by jgentry on 2 Jan 2018

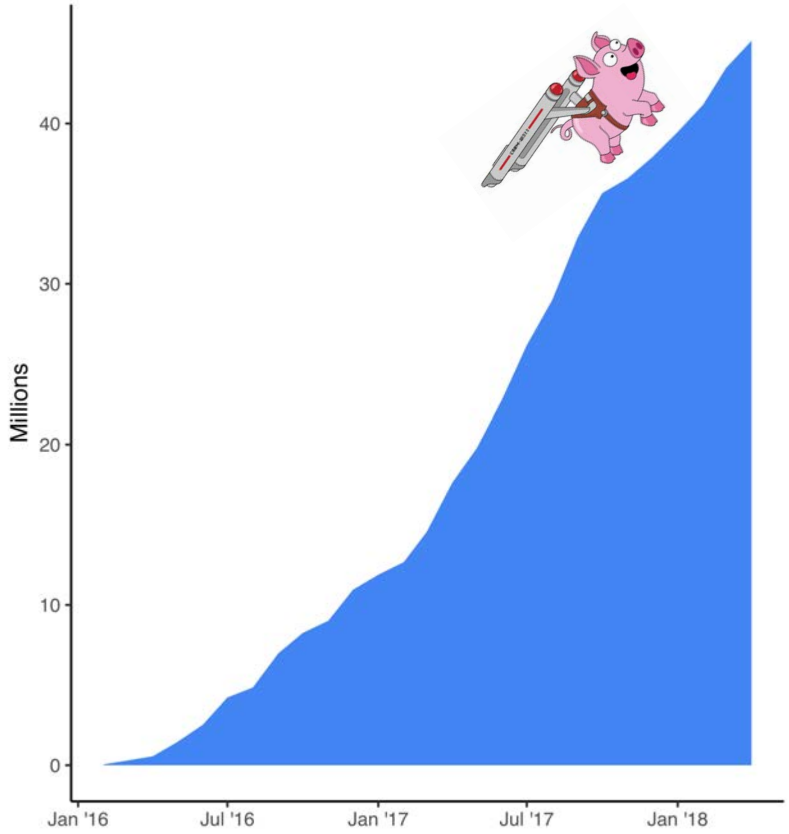
Thanks to our **Workflow Object Model** (**WOM**), Cromwell now supports multiple versions of WDL as well as CWL 1.0!

Cromwell has been busy

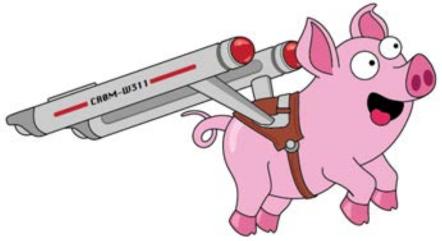
Cromwell in production at Broad:

Processed **47.5 million jobs** over the last two years

And this is just the tip of the iceberg!



Want to discuss further?



My Email:
rmunshi@broadinstitute.org

More Information:

Docs: http://cromwell.readthedocs.io/en/develop/

Github: https://www.github.com/broadinstitute/cromwell

WDL: http://www.openwdl.org

Example Pipelines: https://github.com/gatk-workflows