Genomic Workflows

Jason Walker

BFX Workshop

April 22nd, 2024

Image by ElasticComputeFarm from Pixabay


30 Years of Our Best Recipes
ANNUAL RECIPES
ANNUAL RECIPES
ANNUAL RECIPES
NUAL RECIPES

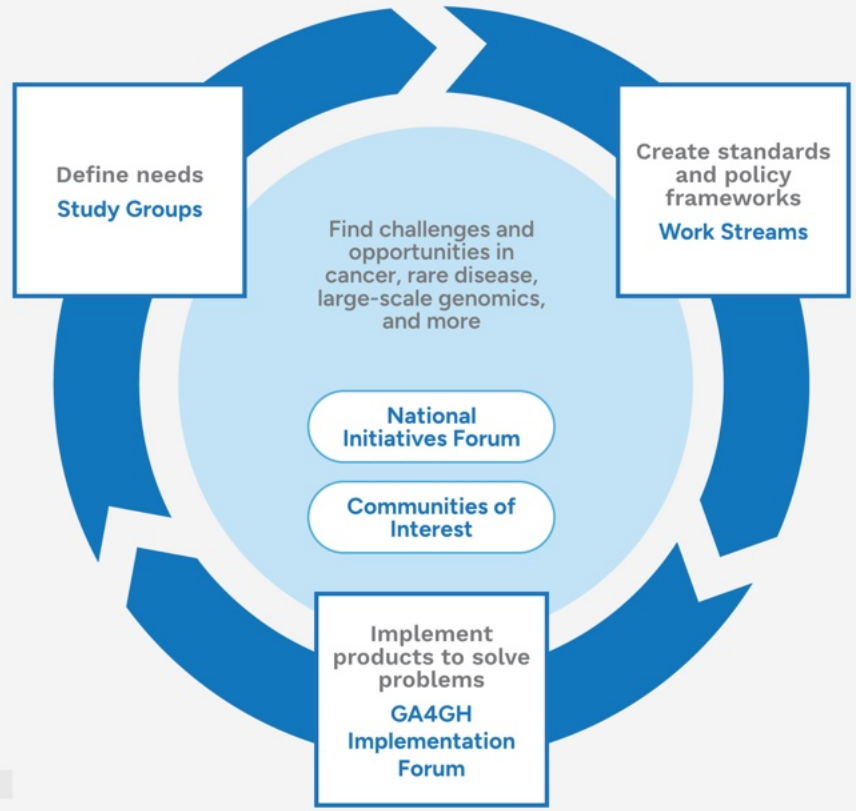
https://pixabay.com/

How we work

Here's our community...

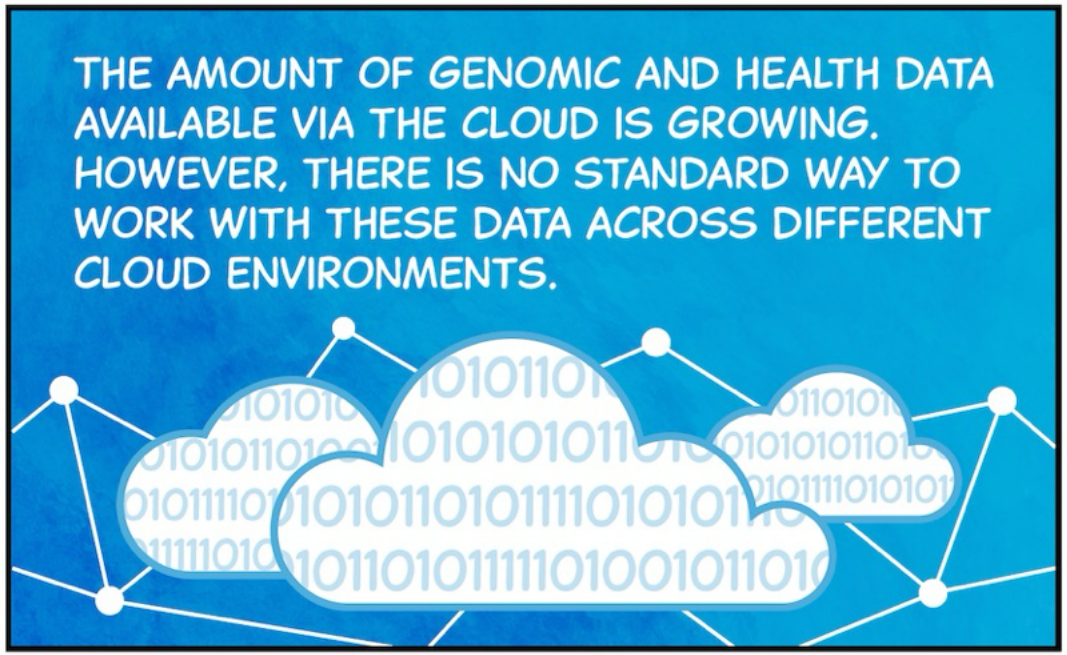
- Strategic Partners
- Driver Projects
- Organisational Members
- Assigned Experts
- Individual Contributors

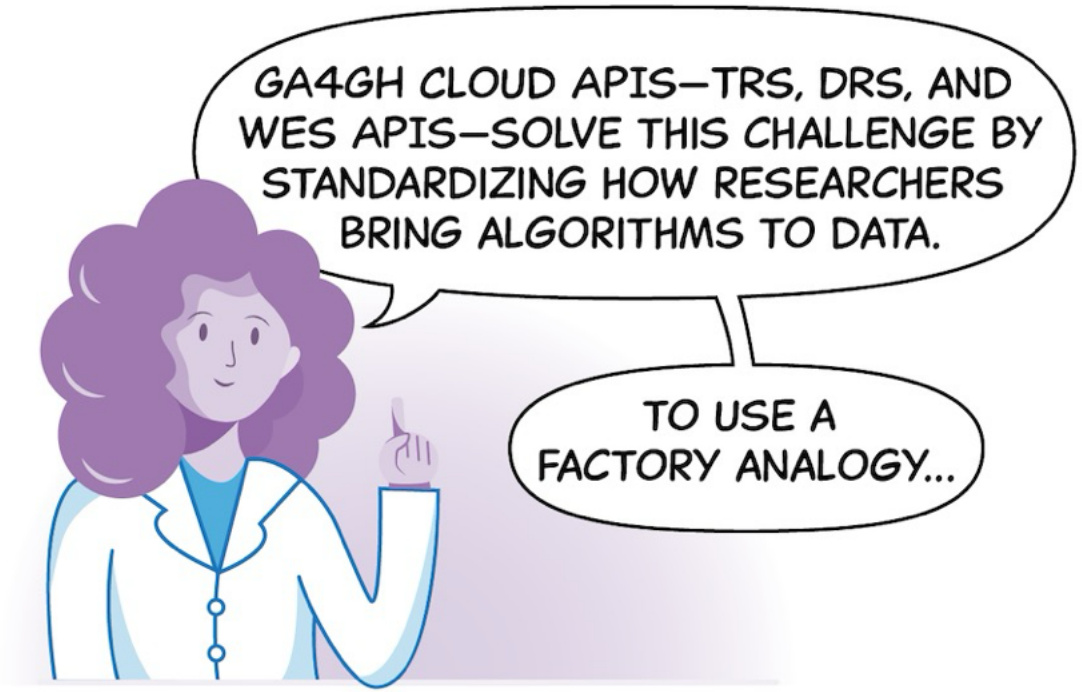
...and here's what we do.



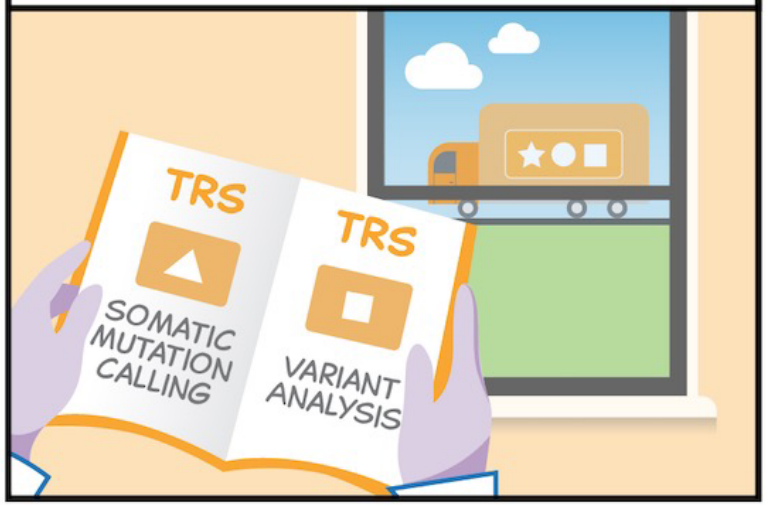
DISCOVER WHAT WE DO

GA4GH CLOUD APIS:
BRINGING ANALYSES TO DATA

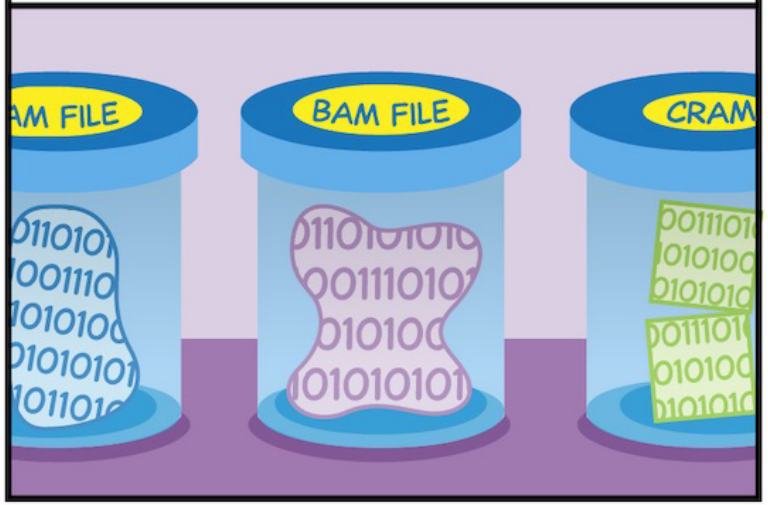




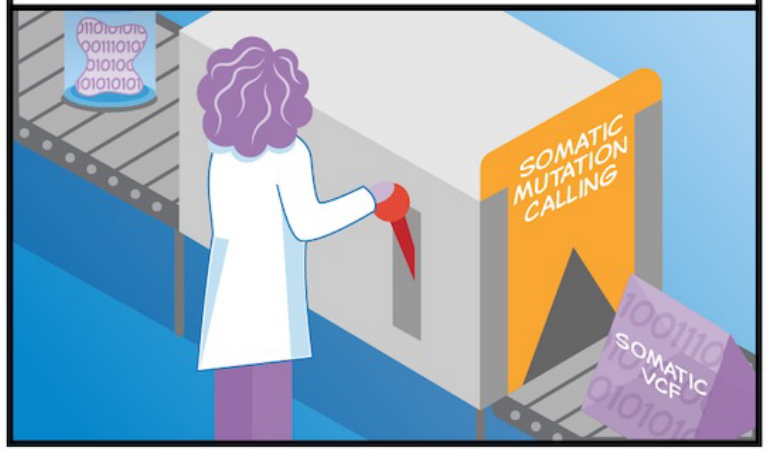
WITH TRS (TOOL REGISTRY SERVICE) API, RESEARCHERS CAN SHARE AND DISCOVER TOOLS AND WORKFLOWS FOR WORKING WITH DATA.

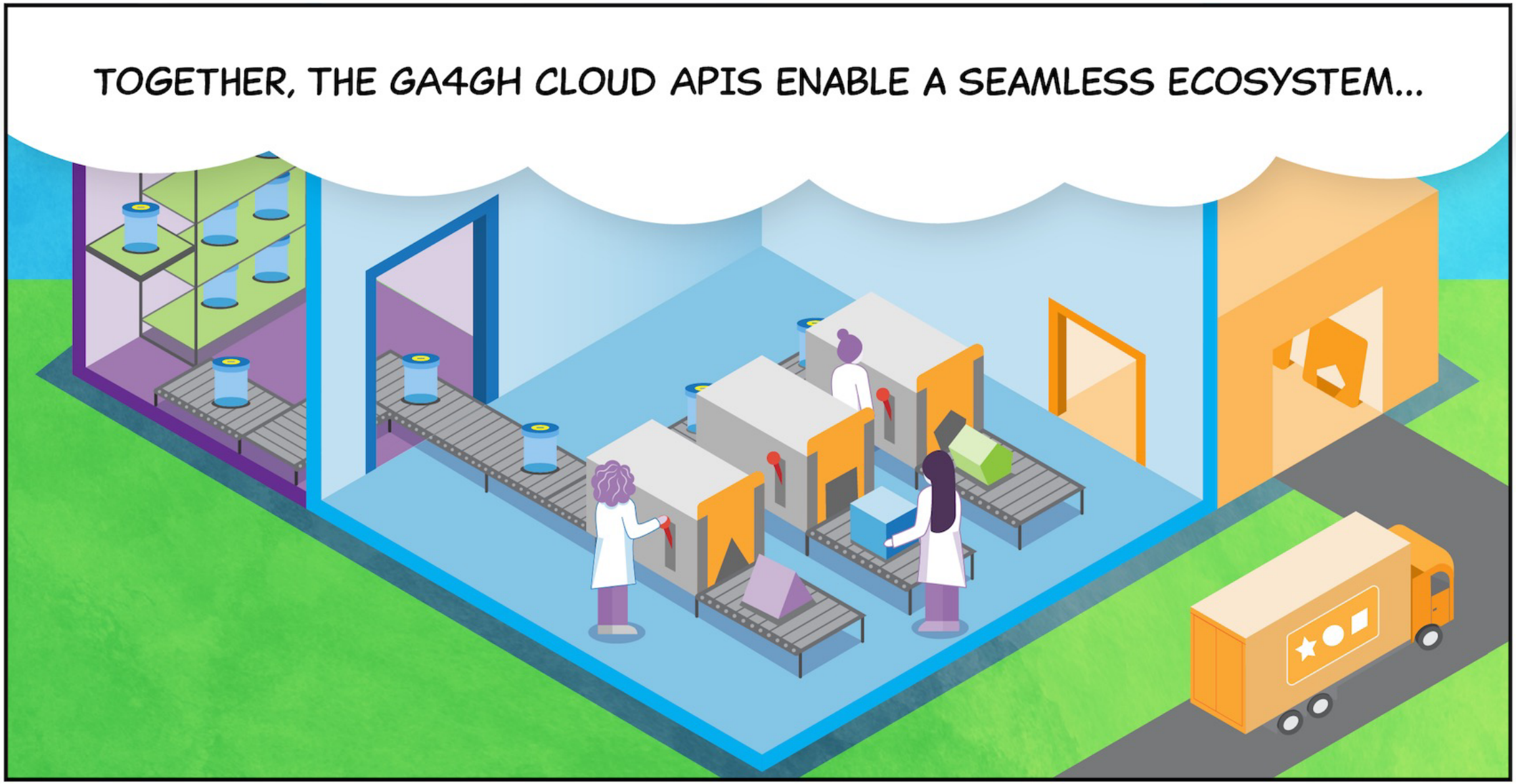


DRS (DATA REPOSITORY SERVICE) API PACKAGES DATA IN A UNIVERSAL WAY, DESPITE HOW THE DATA WAS STORED OR MANAGED ORIGINALLY.



WES (WORKFLOW EXECUTION SERVICE) API ALLOWS RESEARCHERS TO EXECUTE ANALYSES ACROSS DIFFERENT ENVIRONMENTS AND ACHIEVE CONSISTENT RESULTS.

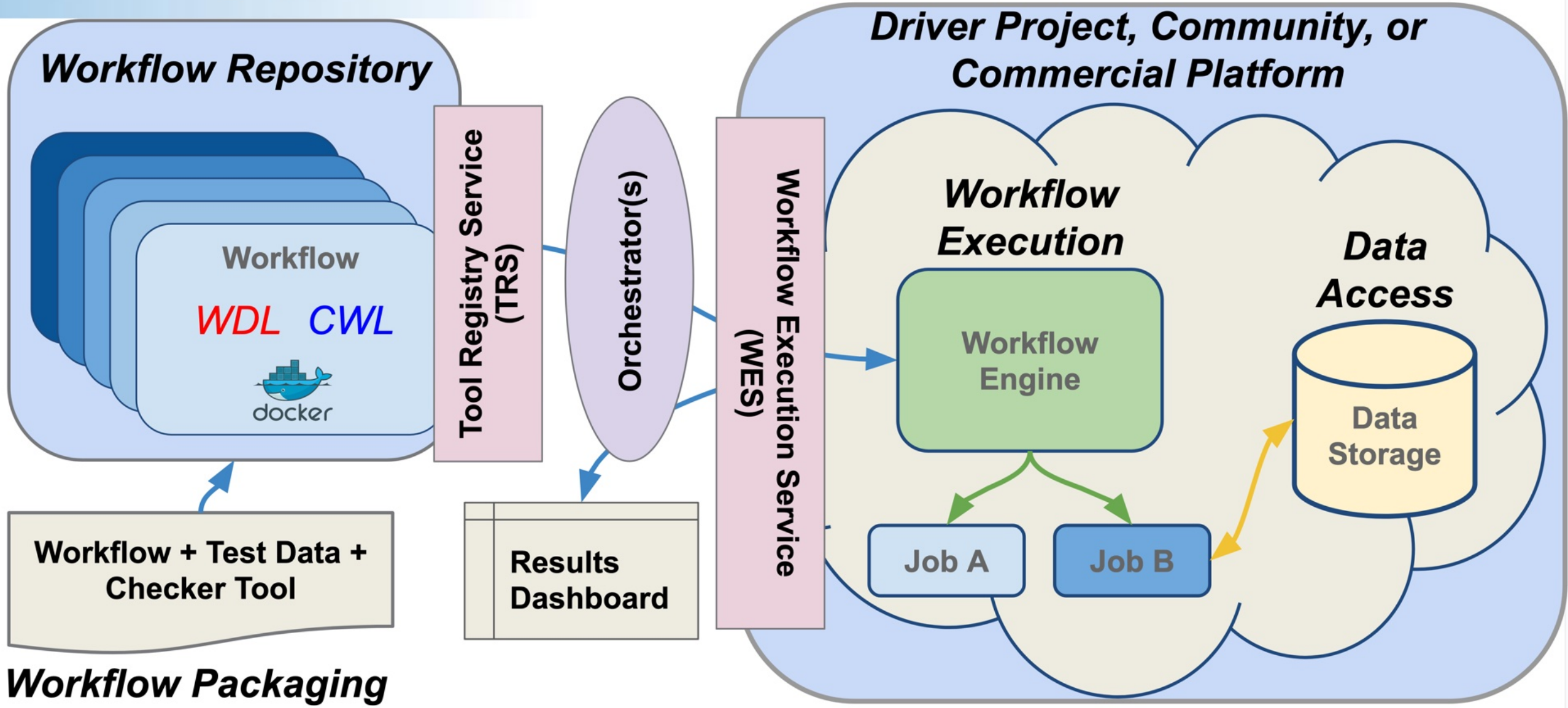



TOGETHER, THE GA4GH CLOUD APIS ENABLE A SEAMLESS ECOSYSTEM...


...ALLOWING RESEARCHERS TO ACCESS, ANALYZE, AND WORK WITH DATASETS IN THE CLOUD.

Leveraging Our Existing Cloud Work Stream Standards

Global Alliance for Genomics & Health



https://www.commonwl.org/

https://openwdl.org/

https://www.nextflow.io/

https://snakemake.github.io/

https://galaxyproject.github.io/

Common Workflow Definition Languages

- Defining a process, including the software tools, settings, and configuration, as a workflow results in portable and sharable best practices.
- CWL, WDL, Nextflow, Galaxy, and Snakemake are the most *common* ways to represent workflow definitions in Genomics.
- Standard ways to represent inputs, outputs and tools used in workflows.
- Combined with Docker these definitions are portable to many High Performance Computing environments including public Clouds.
- The definitions themselves adhere to FAIR Principles for Research Software (FAIR4RS), making the workflows digital data objects as well as the results.

FAIR4RS - Findable

F: Software, and its associated metadata, is easy for both humans and machines to find.
F1. Software is assigned a globally unique and persistent identifier.
F1.1. Components of the software representing levels of granularity are assigned distinct identifiers.
F1.2. Different versions of the software are assigned distinct identifiers.
F2. Software is described with rich metadata.
F3. Metadata clearly and explicitly include the identifier of the software they describe.
F4. Metadata are FAIR, searchable and indexable.

Barker, M., Chue Hong, N.P., Katz, D.S. *et al.* Introducing the FAIR Principles for research software. *Sci Data* **9**, 622 (2022). https://doi.org/10.1038/s41597-022-01710-x

FAIR4RS - Accessible

A: Software, and its metadata, is retrievable via standardised protocols.
A1. Software is retrievable by its identifier using a standardised communications protocol.
A1.1. The protocol is open, free, and universally implementable.
A1.2. The protocol allows for an authentication and authorization procedure, where necessary.
A2. Metadata are accessible, even when the software is no longer available.

Barker, M., Chue Hong, N.P., Katz, D.S. *et al.* Introducing the FAIR Principles for research software. *Sci Data* **9**, 622 (2022). https://doi.org/10.1038/s41597-022-01710-x

FAIR4RS - Interoperable

I: Software interoperates with other software by exchanging data and/or metadata, and/or through interaction via application programming interfaces (APIs), described through standards.
I1. Software reads, writes and exchanges data in a way that meets domain-relevant community standards.
I2. Software includes qualified references to other objects.

Barker, M., Chue Hong, N.P., Katz, D.S. *et al.* Introducing the FAIR Principles for research software. *Sci Data* **9**, 622 (2022). https://doi.org/10.1038/s41597-022-01710-x

FAIR4RS - Reuseable

R: Software is both usable (can be executed) and reusable (can be understood, modified, built upon, or incorporated into other software).
R1. Software is described with a plurality of accurate and relevant attributes.
R1.1. Software is given a clear and accessible license.
R1.2. Software is associated with detailed provenance.
R2. Software includes qualified references to other software.
R3. Software meets domain-relevant community standards.

Barker, M., Chue Hong, N.P., Katz, D.S. *et al.* Introducing the FAIR Principles for research software. *Sci Data* **9**, 622 (2022). https://doi.org/10.1038/s41597-022-01710-x

Language-level Comparison

Aspect	Nextflow	CWL	WDL
Parent language	Ruby and Groovy	N/A	N/A
Compilation	Interpreted	Compiled	Compiled
GUIs	NextflowWorkbench[27], DolphinNext[28]	Rabix composer	Pipeline Builder[29]
DSL features	complete, extensible in Groovy and Java	Limited standard library, extensible via `javascript`	Limited standard library
Variables	Qualified, unique within scope	Typed, unique identifiers	Typed, fully qualified names
Loops	Parallel `queue channels`	Parallel `scatter` via `ScatterFeatureRequirement`	Parallel `scatter`
Conditionals	Via `when` declaration within a process	`When` and `pickValue` fields proposed in CWLv1.2	`If` blocks producing optional output types
Enforcing good practices	`nf-core` (https://nf-co.re/)	CWL guide (https://www.commonwl.org/user_guide/rec-practices/)	–

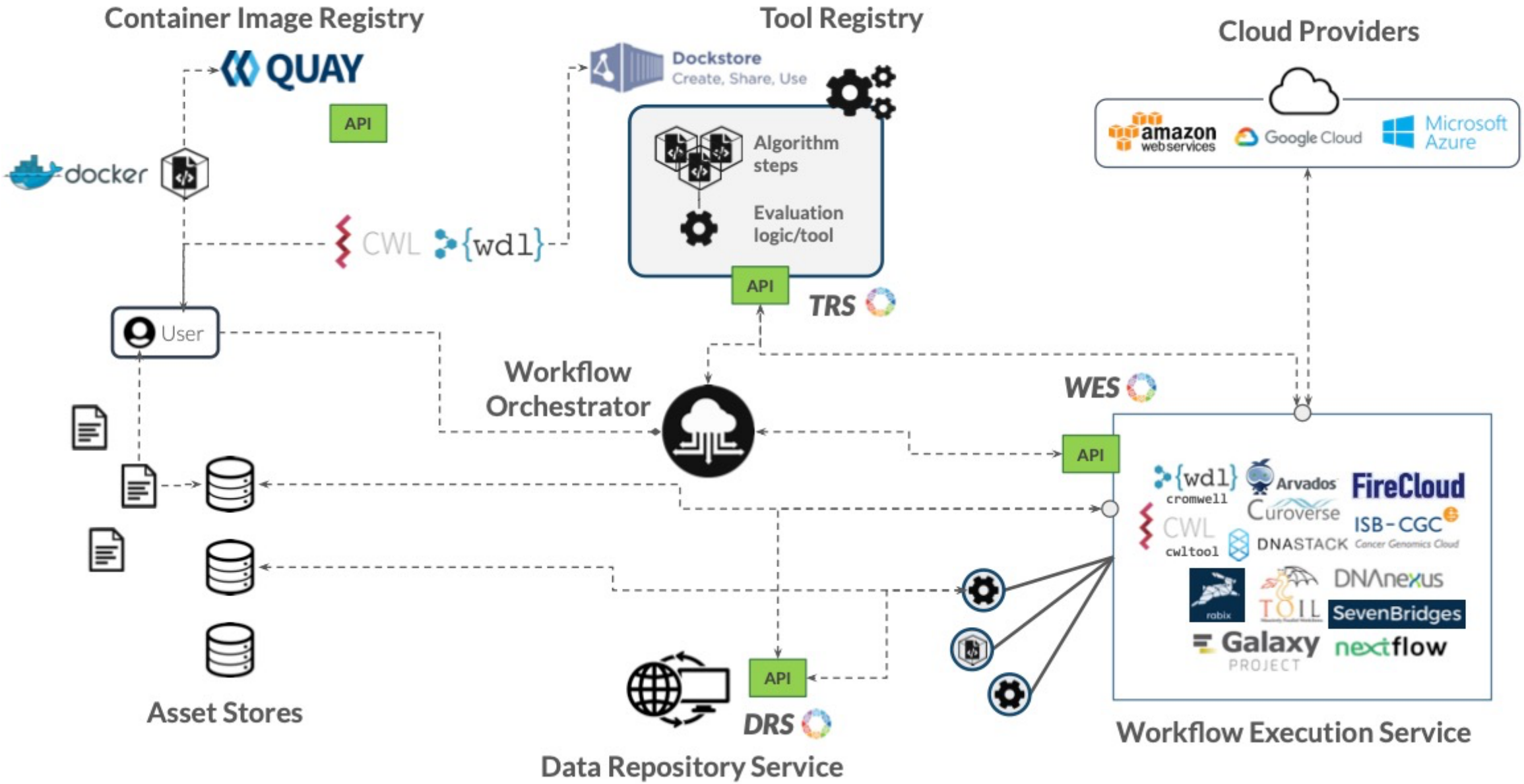
Ahmed, A.E., Allen, J.M., Bhat, T. *et al.* Design considerations for workflow management systems use in production genomics research and the clinic. *Sci Rep* **11**, 21680 (2021). https://doi.org/10.1038/s41598-021-99288-8

Executor-level Comparison

WfMS		Remarks
Language	Execution engine	
Nextflow (DSL-1, DSL-2)		Complete WfMS, supporting conditionals, loops and nested logic
CWL	*cwltool*†	The official reference implementation of an execution engine for the complete CWL standard[33]; no cluster or cloud support
	arvados†(`1.0`, `1.1`, `1.2`)	Most feature-rich CWL runner, albiet with tedious setup
	toil-cwl-runner†(`1.0.1`)	Optimized for cloud environments, less stable in batch environments (Section Scalability)
	cwl airflow†(`1.1`)	Works with `celery` and `Kubernetes` clusters, not readily with HPC CRMs
	REANA†(Documentation missing)	Cloud-optimized platform. For HPC, only CERN Slurm and HTcondor are supported
	Cromwell (`1.0`)	Supports CWL workflows via WOM, with comparable performance in both languages (Section Scalability)
	cwl-tes (`1.0`)	Partial implementation at present, with tedious setup. GA4GH TES API compatible
	rabix executor (`sbg:draft-2`, `1.0`)	Single node local executor is no longer supported by the original developer team at Seven Bridges
WDL	*Cromwell* (`draft-2`, `1.0`)	De facto standard for executing WDL workflows. Support for nested loops is version-dependent
	toil-wdl-runner (`draft-2`)	No support for modularity or nesting of loops and conditionals. Support for batch systems is also rudimentary
	miniWDL (`draft-2`, `1.0`)	No cluster or cloud support. Includes Cromwell wrapper

Ahmed, A.E., Allen, J.M., Bhat, T. *et al.* Design considerations for workflow management systems use in production genomics research and the clinic. *Sci Rep* **11**, 21680 (2021). https://doi.org/10.1038/s41598-021-99288-8

WES in the ecosystem of GA4GH APIs



A Workflow Example:

https://github.com/genome/bfx-workshop/tree/master/lectures/week_23/SingleSampleQc