UK Atomic Energy Authority

Simvue - A generic and configurable real-time tracking and monitoring framework for any simulation or data processing tasks

Andrew Lahiff, Vignesh Gopakumar, Aby Abraham

IRIS – TWG – 13th Feb 2024

UKAEA mission

"To lead the delivery of sustainable fusion energy and maximise the scientific and economic benefit."

1. Be a world leader in fusion research and development

2. Enable the delivery of sustainable fusion power plants

3. Drive economic growth and high-tech jobs in the UK

4. Create places that **accelerate innovation** and develop skilled people for industry to thrive

Outline

A) Background, vision & scope

B) Simvue functionality

- Real time monitoring & comparisons
- Real time tracking & alerting
- Building a digital thread
- Handling multiple input & output files
- Green computing

C) Data processing application

D) Simvue integration examples

https://simvue.io/

https://docs.simvue.io/

Background (1)

The traditional view is that HTC is for many independent small tasks & HPC is for small numbers of very large tasks

- This is increasingly being blurred

Increasingly common to run larger and larger numbers of HPC simulations

- Parameter studies
- Uncertainty quantification
- Design of experiments
- Creating surrogate models using machine learning

Need to track increasingly large numbers of files, tasks & associated metadata

- Need both data management & experiment management

Background (2)

Automatically monitoring the performance of tasks in near real-time is also important

- Progress, performance, convergence, …

It is still common for people to manually monitor simulations

- This is becoming too time-consuming & no longer possible at larger and larger scales

In AI/ML experiment tracking platforms are becoming very common, but not yet for people running simulations

Simvue

Made in UKAEA

Generic framework for tracking and monitoring scalable tasks

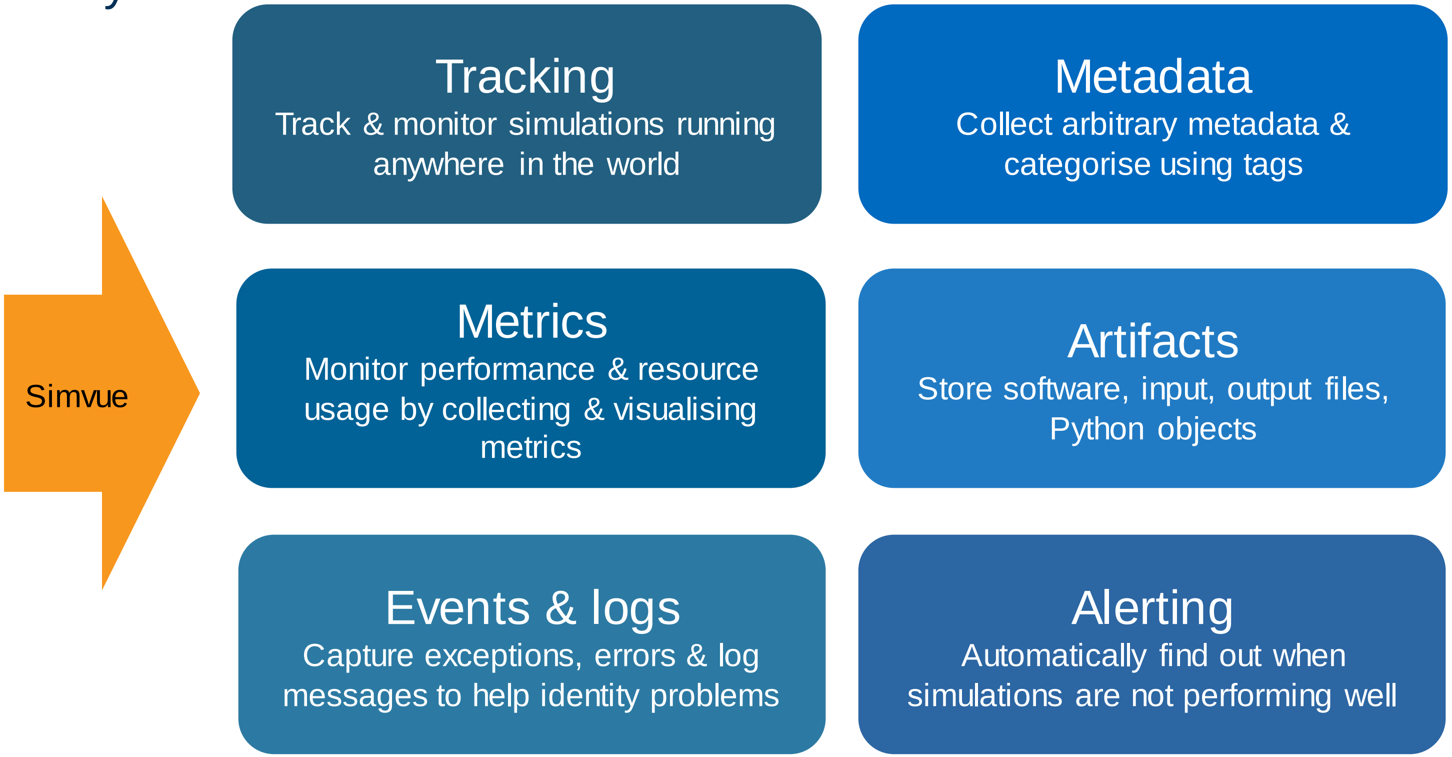
Modern and legacy applications integration

Cost-effective task management

Democratisation and standardisation of data processing or simulations

Interoperability

- Simulation
- Processing
- Training
- Analysis



Simvue

Tracking
Track & monitor simulations running anywhere in the world

Metadata
Collect arbitrary metadata & categorise using tags

Metrics
Monitor performance & resource usage by collecting & visualising metrics

Artifacts
Store software, input, output files, Python objects

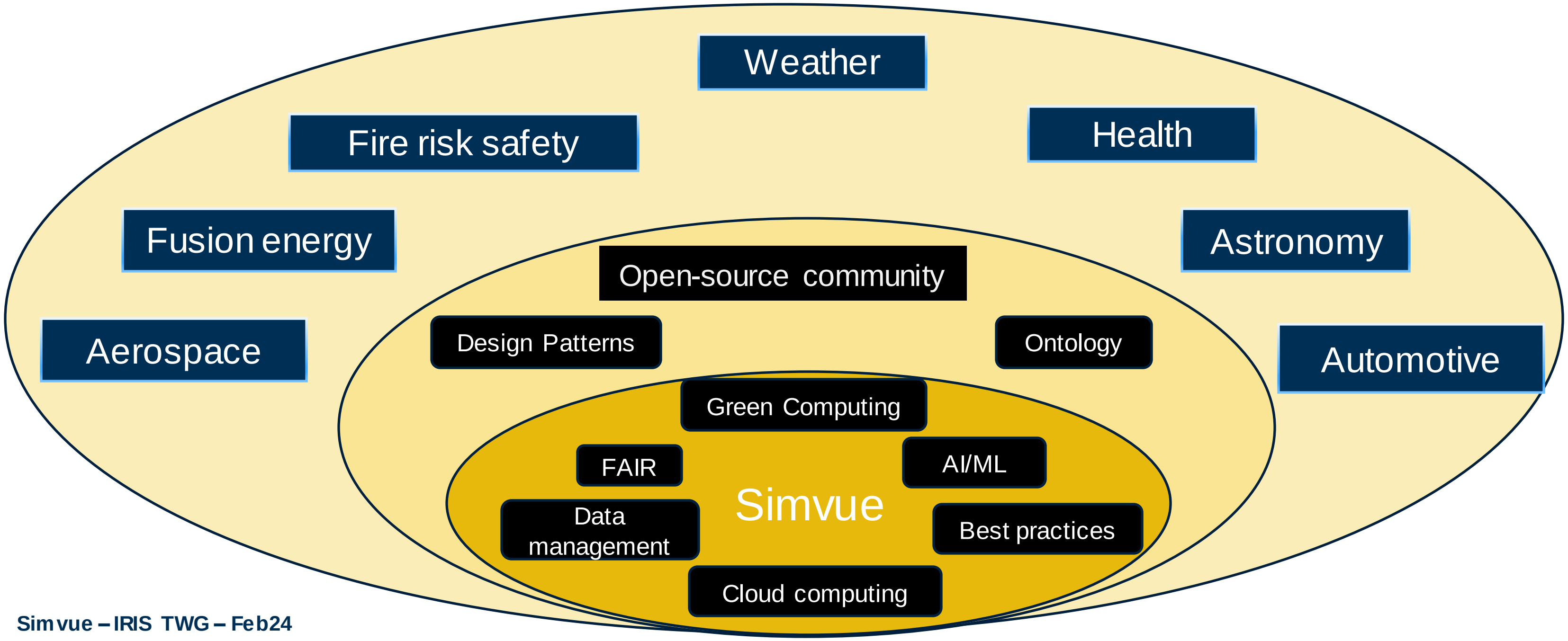
Events & logs
Capture exceptions, errors & log messages to help identity problems

Alerting
Automatically find out when simulations are not performing well

Vision & scope

Increasingly common to run more **computationally intensive** simulations and data processing tasks

- Setting up bespoke **real-time monitoring** is non-trivial and error prone
- **<u>Imagine</u>** if there was a **generic framework** for real-time monitoring and tracking
  - Actively developed by an **open-source community**
  - **Agnostic** to hardware and simulation software
  - **Cost-effective and scalable** for complex simulation configuration



Simvue functionality

Basic overview

Real time monitoring & comparisons

Tracking & alerting

Building a digital thread

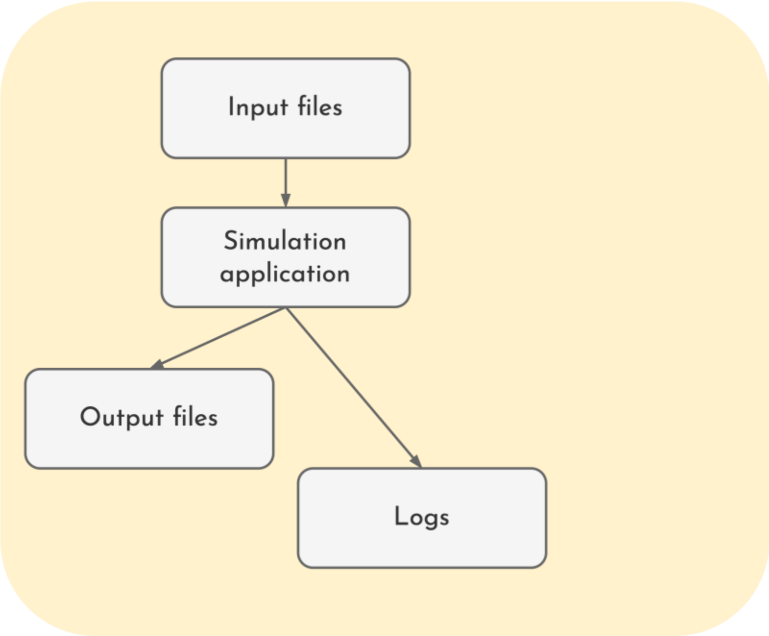
Handling multiple input & output files

API access

Green computing

Simvue overview

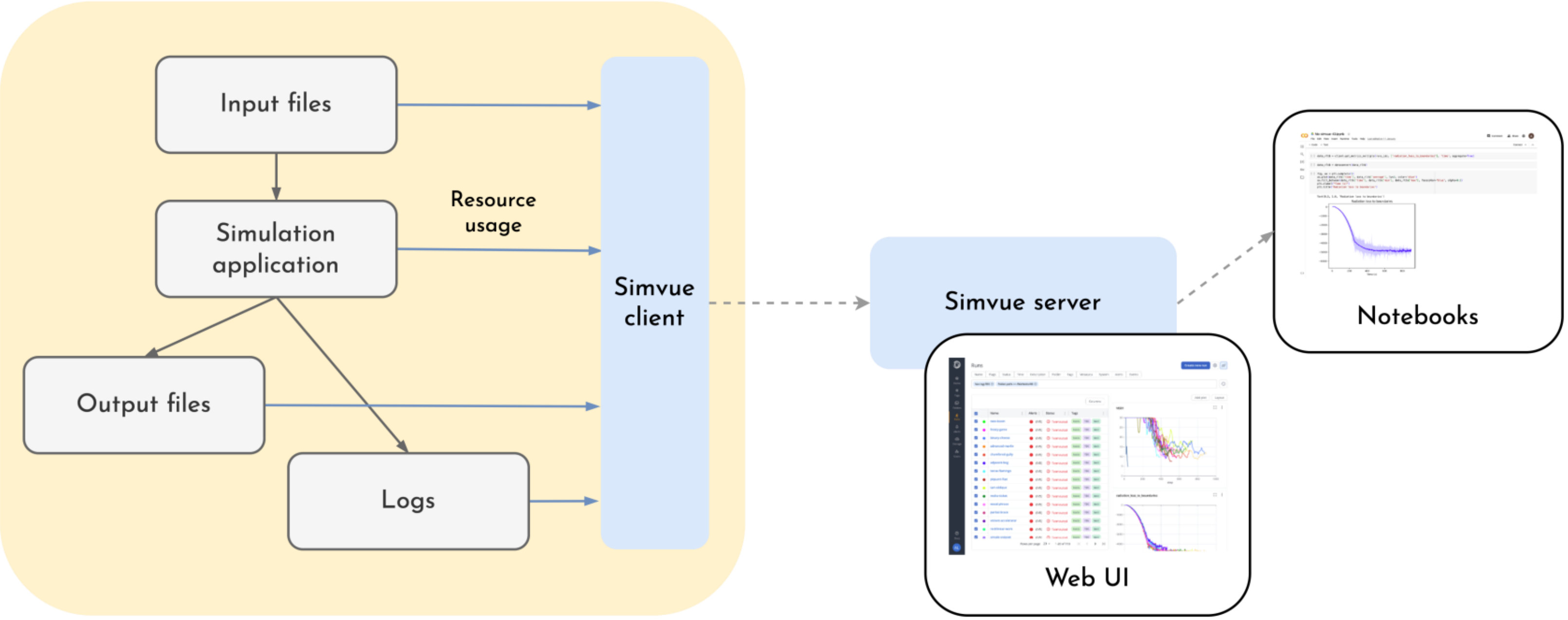
Typical simulation – reads input files, writes output files & logs



Simvue overview

Simvue client monitors user application

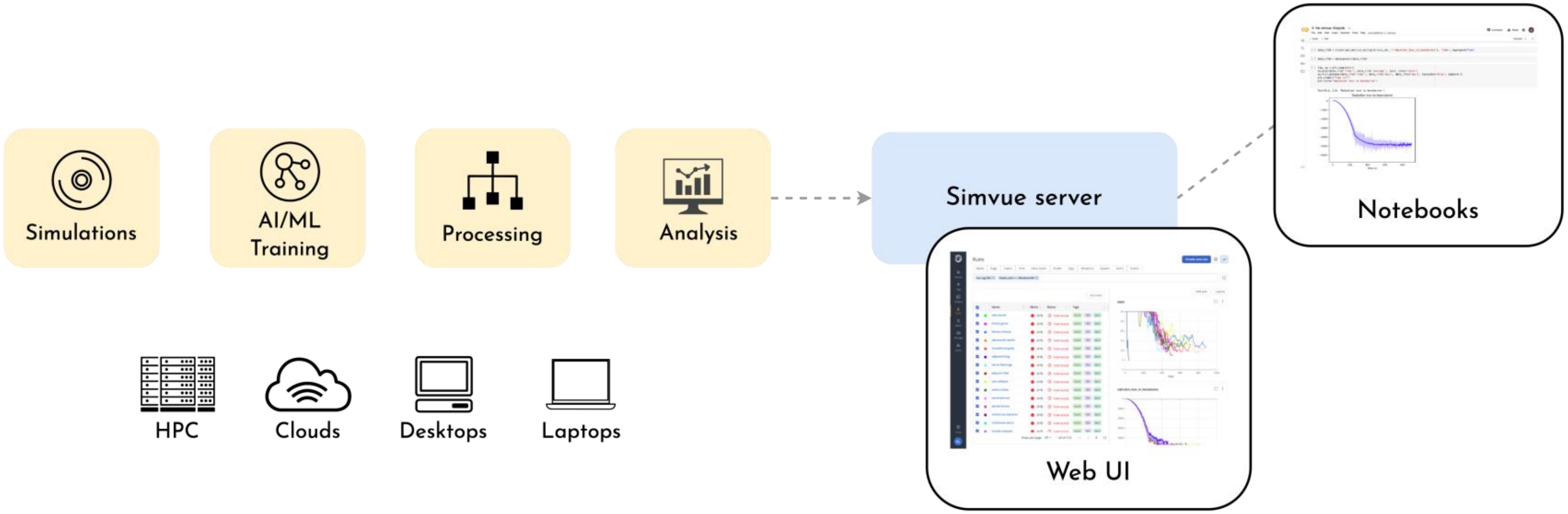
- Registers input & output files, copy to object storage (if needed)
- Gathers metadata, metrics, logs, etc, sends to remote Simvue server



Simvue overview

Not just simulations

- **Any** type of application on **any** type of resource



Data model

Run

- Execution of an application, such as simulation, data processing, training

Folder

- Hierarchical structure of folders allowing users to group runs, representing an experiment/study

Artifacts

- Codes, input files, output files, Python objects, …

Metrics

- Time-series metrics

Events

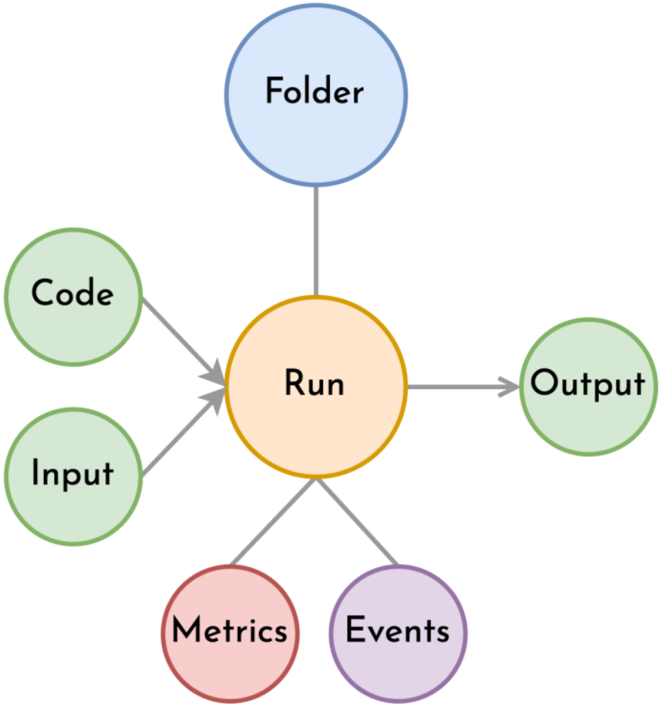
- Time series log messages

Metadata (runs, folders)

- Key-value pairs (string, int, float)

Tags (runs, folders)

- List of short strings
- Used for categorisation



Fire Dynamics Simulator (FDS)

Open-source software developed by NIST

- Computational Fluid Dynamics (CFD) model of a fire-driven fluid flow

Simulate the behaviour of fires in complex multi-room environments

- Models the interaction between fire, smoke, & airflow

Example use cases

- Design of smoke handling systems
- Sprinkler/detection studies
- Residential & industrial fire reconstructions

Our initial use case:

- Pallets in a garage
- Investigating the effect of different ceiling vent sizes & locations

This use case demonstrates quite a common situation:

- Need to run many simulations with different parameters
- Need to track metadata & data associated with each
- Some simulations may fail or may not meet requirements

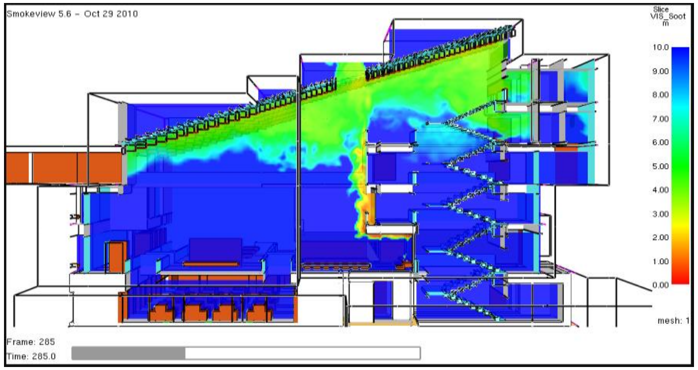
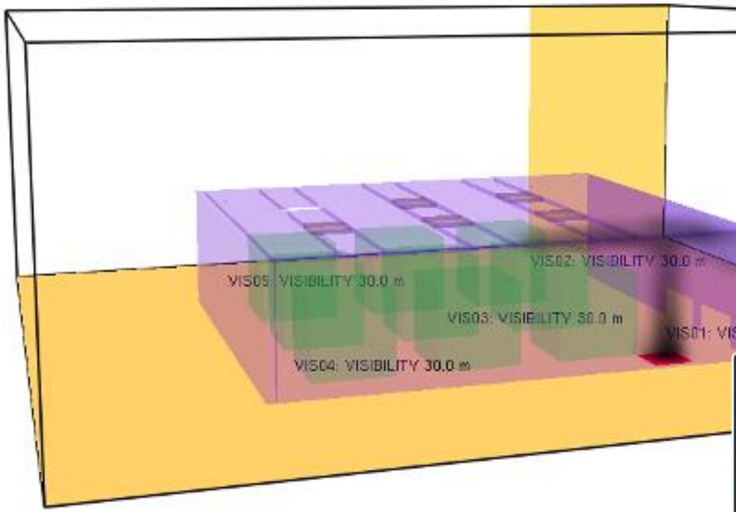
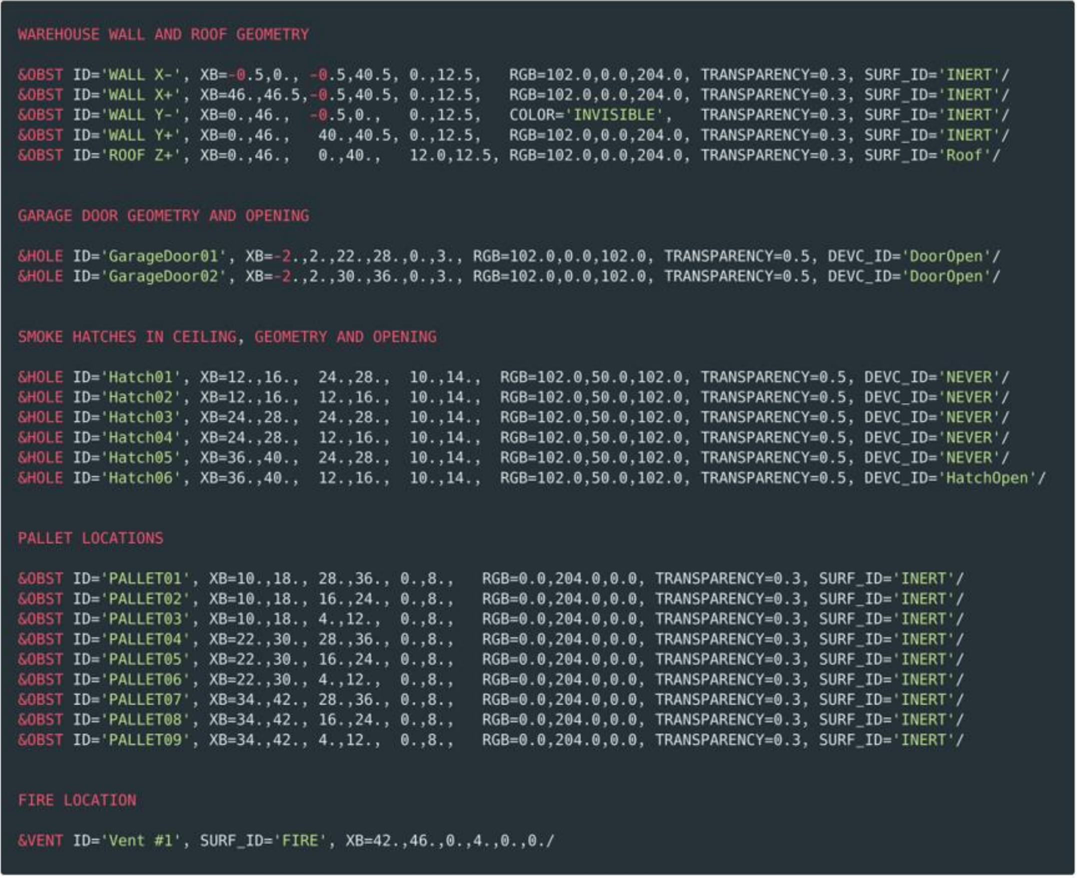


Image from https://fdstutorial.com/what-is-fds/

Fire Dynamics Simulator (FDS)



Example extract from input file



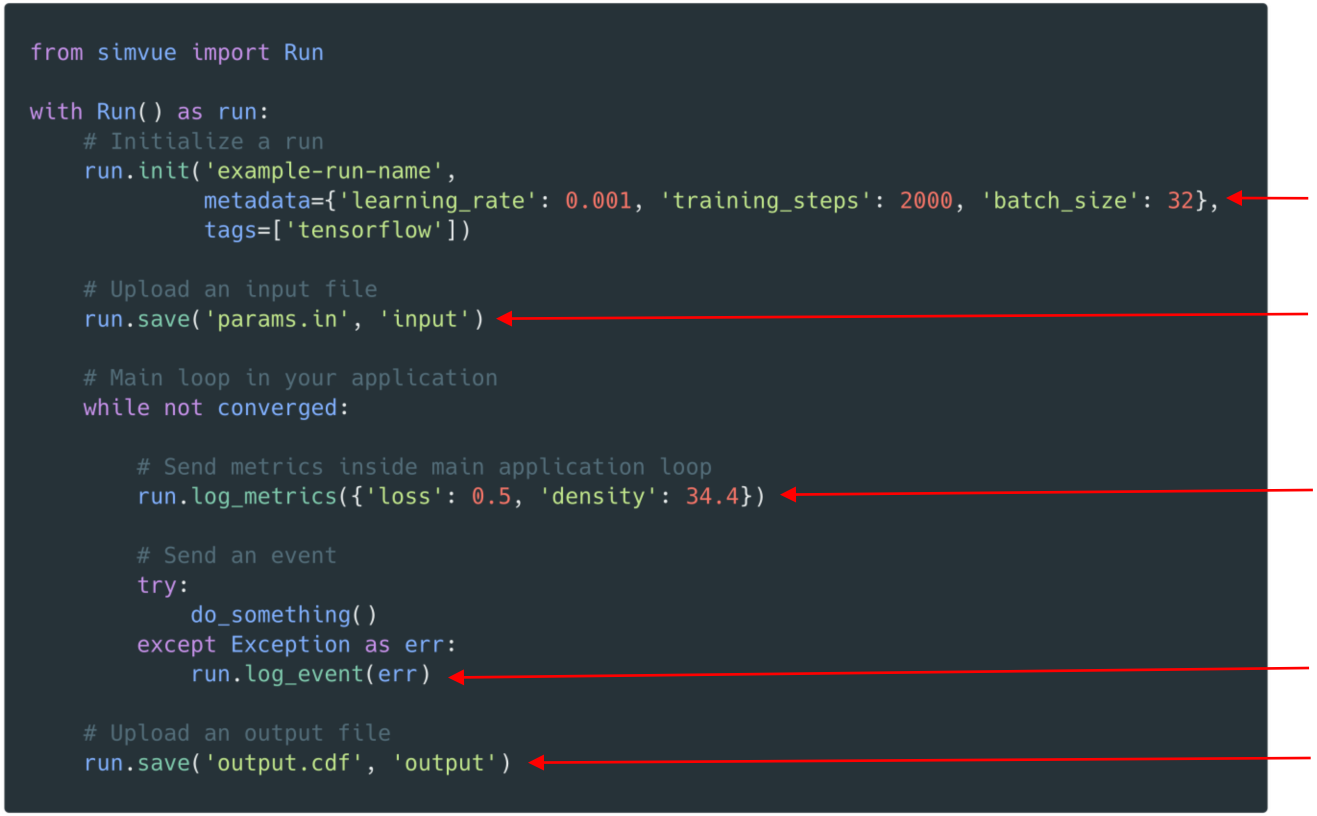
```
WAREHOUSE WALL AND ROOF GEOMETRY

&OBST ID='WALL X-', XB=-0.5,0., -0.5,40.5, 0.,12.5,   RGB=102.0,0.0,204.0, TRANSPARENCY=0.3, SURF_ID='INERT'/
&OBST ID='WALL X+', XB=46.,46.5,-0.5,40.5, 0.,12.5,   RGB=102.0,0.0,204.0, TRANSPARENCY=0.3, SURF_ID='INERT'/
&OBST ID='WALL Y-', XB=0.,46.,  -0.5,0.,   0.,12.5,   COLOR='INVISIBLE',   TRANSPARENCY=0.3, SURF_ID='INERT'/
&OBST ID='WALL Y+', XB=0.,46.,   40.,40.5, 0.,12.5,   RGB=102.0,0.0,204.0, TRANSPARENCY=0.3, SURF_ID='INERT'/
&OBST ID='ROOF Z+', XB=0.,46.,   0.,40.,   12.0,12.5, RGB=102.0,0.0,204.0, TRANSPARENCY=0.3, SURF_ID='Roof'/


GARAGE DOOR GEOMETRY AND OPENING

&HOLE ID='GarageDoor01', XB=-2.,2.,22.,28.,0.,3., RGB=102.0,0.0,102.0, TRANSPARENCY=0.5, DEVC_ID='DoorOpen'/
&HOLE ID='GarageDoor02', XB=-2.,2.,30.,36.,0.,3., RGB=102.0,0.0,102.0, TRANSPARENCY=0.5, DEVC_ID='DoorOpen'/


SMOKE HATCHES IN CEILING, GEOMETRY AND OPENING

&HOLE ID='Hatch01', XB=12.,16.,  24.,28.,  10.,14.,  RGB=102.0,50.0,102.0, TRANSPARENCY=0.5, DEVC_ID='NEVER'/
&HOLE ID='Hatch02', XB=12.,16.,  12.,16.,  10.,14.,  RGB=102.0,50.0,102.0, TRANSPARENCY=0.5, DEVC_ID='NEVER'/
&HOLE ID='Hatch03', XB=24.,28.,  24.,28.,  10.,14.,  RGB=102.0,50.0,102.0, TRANSPARENCY=0.5, DEVC_ID='NEVER'/
&HOLE ID='Hatch04', XB=24.,28.,  12.,16.,  10.,14.,  RGB=102.0,50.0,102.0, TRANSPARENCY=0.5, DEVC_ID='NEVER'/
&HOLE ID='Hatch05', XB=36.,40.,  24.,28.,  10.,14.,  RGB=102.0,50.0,102.0, TRANSPARENCY=0.5, DEVC_ID='NEVER'/
&HOLE ID='Hatch06', XB=36.,40.,  12.,16.,  10.,14.,  RGB=102.0,50.0,102.0, TRANSPARENCY=0.5, DEVC_ID='HatchOpen'/


PALLET LOCATIONS

&OBST ID='PALLET01', XB=10.,18., 28.,36., 0.,8.,   RGB=0.0,204.0,0.0, TRANSPARENCY=0.3, SURF_ID='INERT'/
&OBST ID='PALLET02', XB=10.,18., 16.,24., 0.,8.,   RGB=0.0,204.0,0.0, TRANSPARENCY=0.3, SURF_ID='INERT'/
&OBST ID='PALLET03', XB=10.,18., 4.,12.,  0.,8.,   RGB=0.0,204.0,0.0, TRANSPARENCY=0.3, SURF_ID='INERT'/
&OBST ID='PALLET04', XB=22.,30., 28.,36., 0.,8.,   RGB=0.0,204.0,0.0, TRANSPARENCY=0.3, SURF_ID='INERT'/
&OBST ID='PALLET05', XB=22.,30., 16.,24., 0.,8.,   RGB=0.0,204.0,0.0, TRANSPARENCY=0.3, SURF_ID='INERT'/
&OBST ID='PALLET06', XB=22.,30., 4.,12.,  0.,8.,   RGB=0.0,204.0,0.0, TRANSPARENCY=0.3, SURF_ID='INERT'/
&OBST ID='PALLET07', XB=34.,42., 28.,36., 0.,8.,   RGB=0.0,204.0,0.0, TRANSPARENCY=0.3, SURF_ID='INERT'/
&OBST ID='PALLET08', XB=34.,42., 16.,24., 0.,8.,   RGB=0.0,204.0,0.0, TRANSPARENCY=0.3, SURF_ID='INERT'/
&OBST ID='PALLET09', XB=34.,42., 4.,12.,  0.,8.,   RGB=0.0,204.0,0.0, TRANSPARENCY=0.3, SURF_ID='INERT'/


FIRE LOCATION

&VENT ID='Vent #1', SURF_ID='FIRE', XB=42.,46.,0.,4.,0.,0./
```

Using Simvue from Python

Using Simvue from Python is straightforward:

- Run `pip install simvue`
- Create `simvue.ini` containing server URL & access token

Add a few additional lines to your code:



```python
from simvue import Run

with Run() as run:
    # Initialize a run
    run.init('example-run-name',
             metadata={'learning_rate': 0.001, 'training_steps': 2000, 'batch_size': 32},
             tags=['tensorflow'])

    # Upload an input file
    run.save('params.in', 'input')

    # Main loop in your application
    while not converged:

        # Send metrics inside main application loop
        run.log_metrics({'loss': 0.5, 'density': 34.4})

        # Send an event
        try:
            do_something()
        except Exception as err:
            run.log_event(err)

    # Upload an output file
    run.save('output.cdf', 'output')
```

Specify metadata, tags

Save an input file

Log custom metrics
(CPU, GPU, memory resource usage metrics automatically collected)

Log an error message

Save an output file

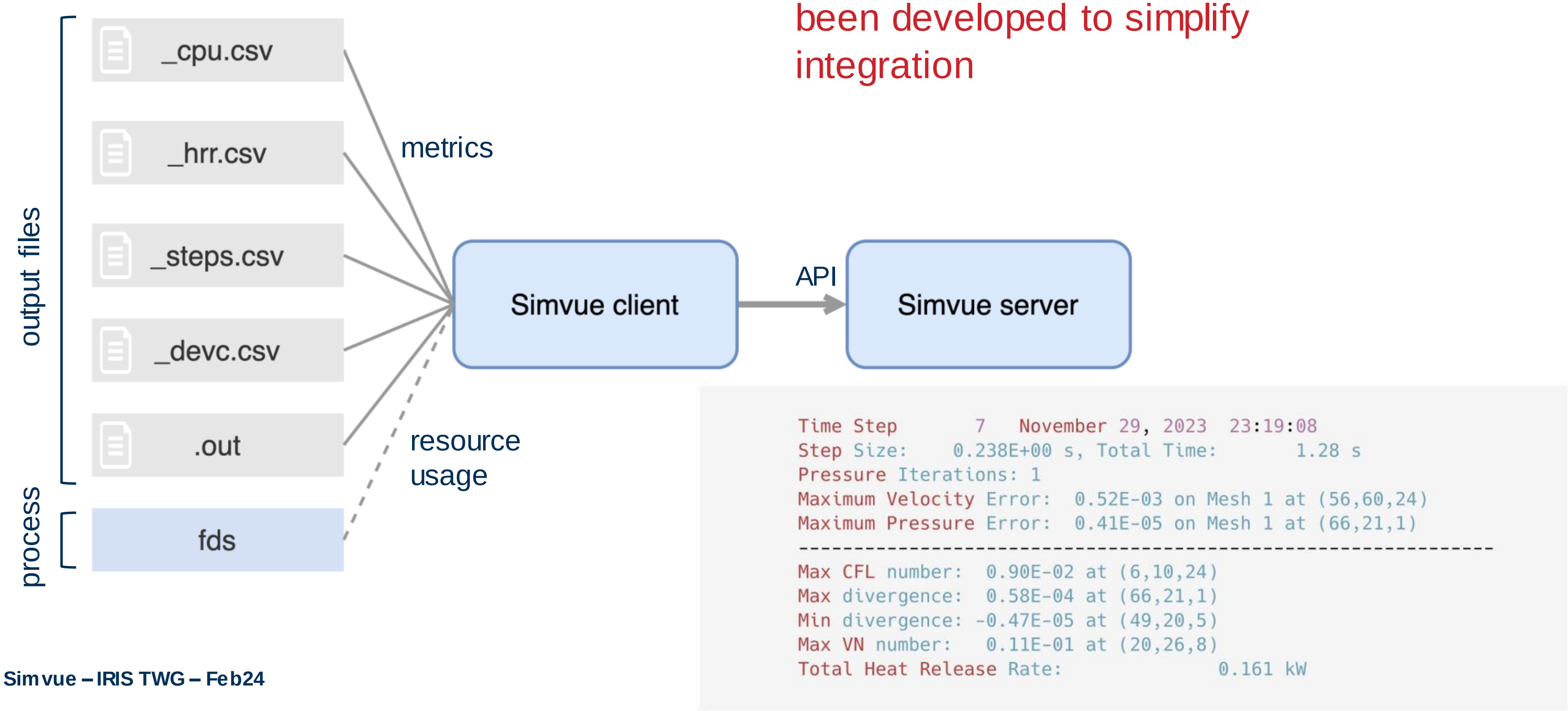
Integration with other languages

Python monitoring script running in parallel to the main application

- Avoids requiring modifications to complex & legacy simulation codes
- Read metadata from input files
- Tail & parse log files to extract metrics & events
- Upload data files

FDS example:

A generic multi-file parser has been developed to simplify integration

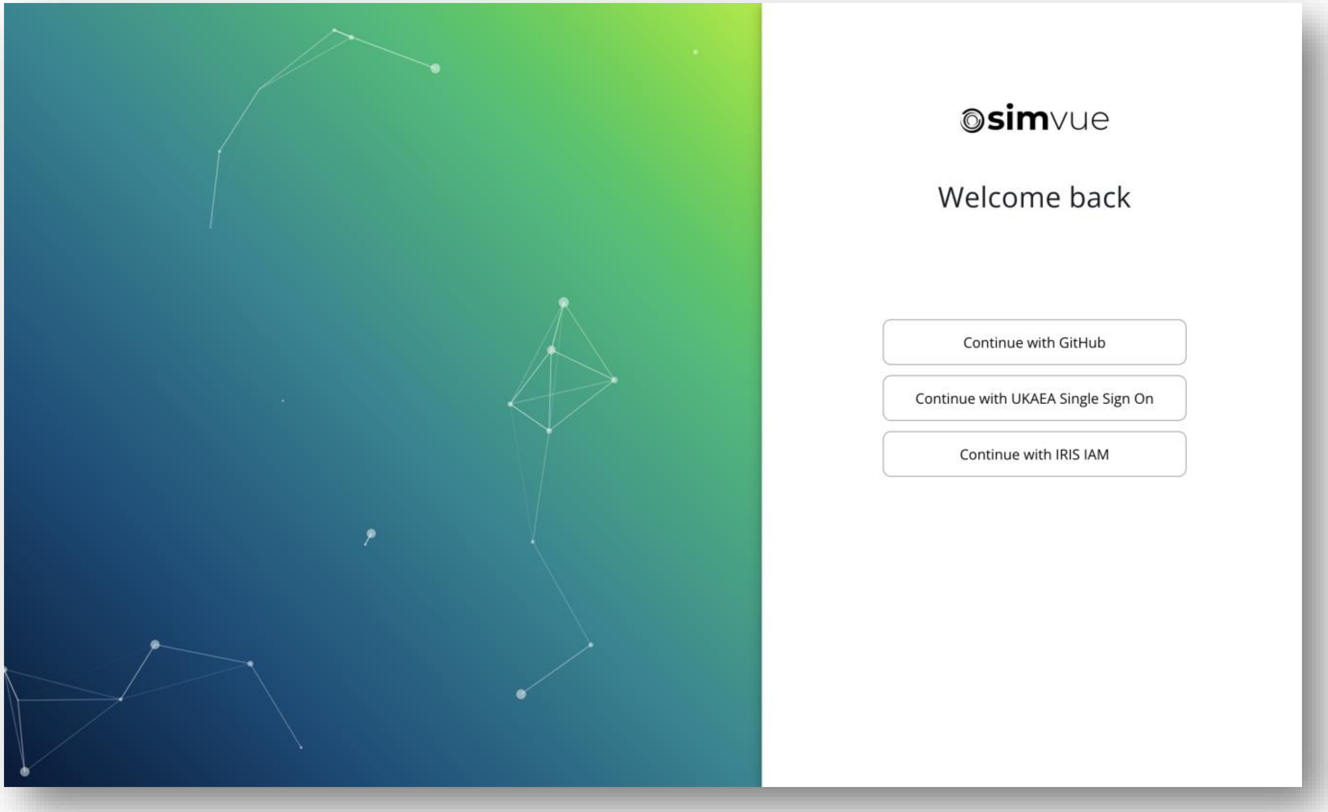


```
Time Step       7   November 29, 2023  23:19:08
Step Size:    0.238E+00 s, Total Time:       1.28 s
Pressure Iterations: 1
Maximum Velocity Error:  0.52E-03 on Mesh 1 at (56,60,24)
Maximum Pressure Error:  0.41E-05 on Mesh 1 at (66,21,1)
----------------------------------------------------------------
Max CFL number:  0.90E-02 at (6,10,24)
Max divergence:  0.58E-04 at (66,21,1)
Min divergence: -0.47E-05 at (49,20,5)
Max VN number:   0.11E-01 at (20,26,8)
Total Heat Release Rate:             0.161 kW
```

Accessing the web UI

The web UI is written in React & makes use of Auth.js for authentication

- Can use any OAuth provider
- Successfully tested integration with IRIS IAM

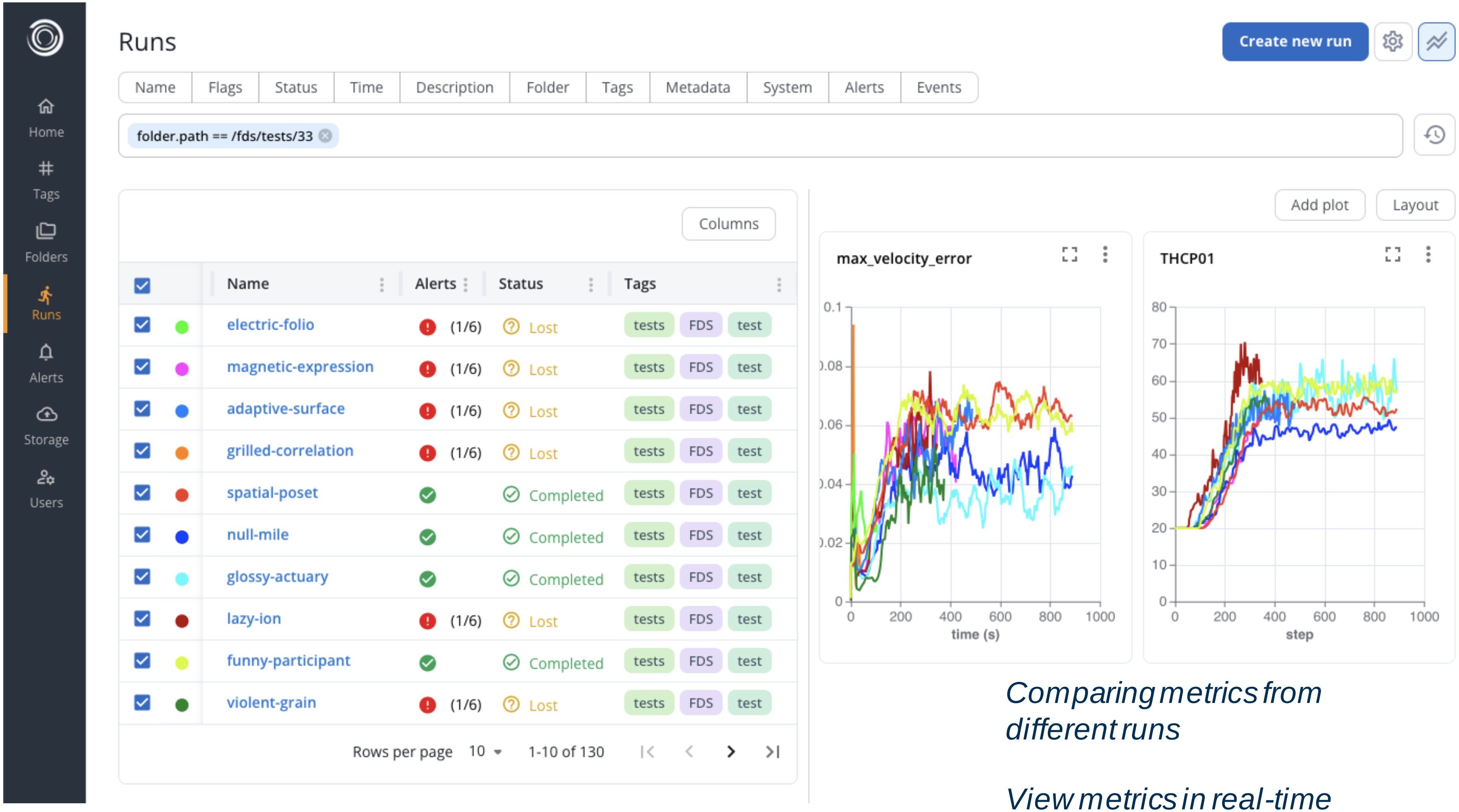


simvue

Welcome back

Real time tracking – Fire simulations

Everything becomes accessible & queryable from a web UI (and/or API)



Runs

folder.path == /fds/tests/33

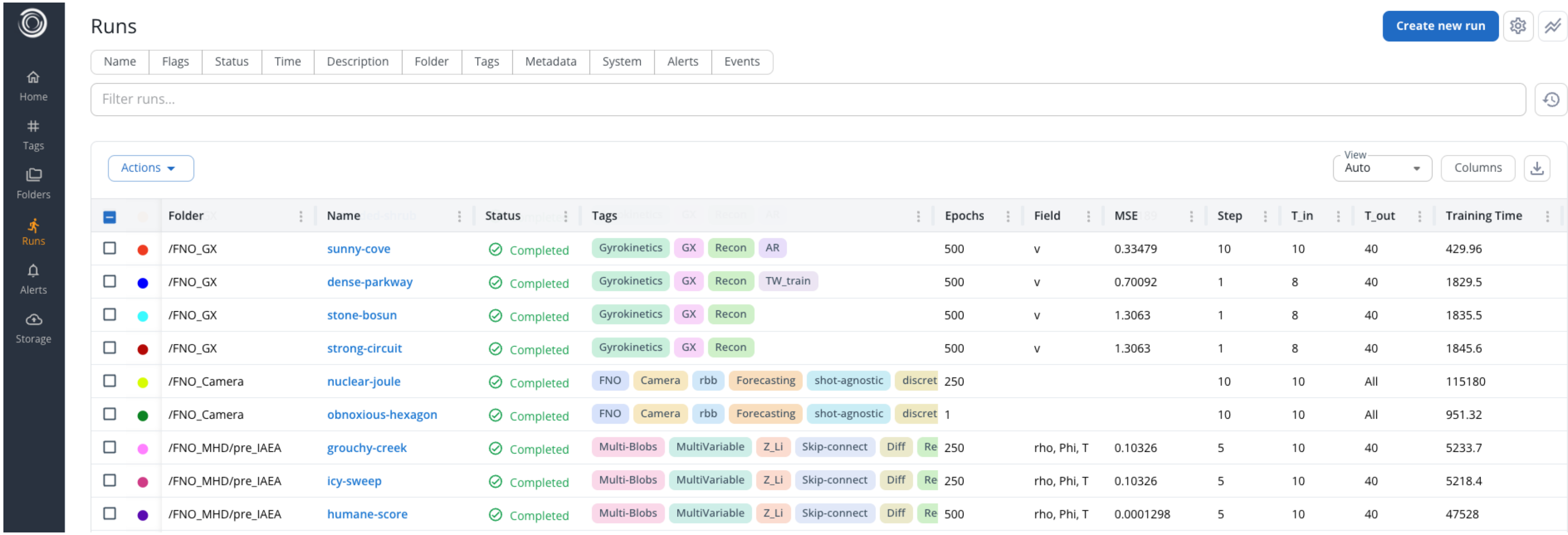
☑		Name	Alerts	Status	Tags
☑	●	electric-folio	! (1/6)	? Lost	tests FDS test
☑	●	magnetic-expression	! (1/6)	? Lost	tests FDS test
☑	●	adaptive-surface	! (1/6)	? Lost	tests FDS test
☑	●	grilled-correlation	! (1/6)	? Lost	tests FDS test
☑	●	spatial-poset	✓	✓ Completed	tests FDS test
☑	●	null-mile	✓	✓ Completed	tests FDS test
☑	●	glossy-actuary	✓	✓ Completed	tests FDS test
☑	●	lazy-ion	! (1/6)	? Lost	tests FDS test
☑	●	funny-participant	✓	✓ Completed	tests FDS test
☑	●	violent-grain	! (1/6)	? Lost	tests FDS test

Rows per page 10 ▾ 1-10 of 130

Comparing metrics from different runs

View metrics in real-time

Real time tracking - Machine Learning Dashboard



☐		Folder	Name	Status	Tags	Epochs	Field	MSE	Step	T_in	T_out	Training Time
☐	●	/FNO_GX	sunny-cove	✓ Completed	Gyrokinetics GX Recon AR	500	v	0.33479	10	10	40	429.96
☐	●	/FNO_GX	dense-parkway	✓ Completed	Gyrokinetics GX Recon TW_train	500	v	0.70092	1	8	40	1829.5
☐	●	/FNO_GX	stone-bosun	✓ Completed	Gyrokinetics GX Recon	500	v	1.3063	1	8	40	1835.5
☐	●	/FNO_GX	strong-circuit	✓ Completed	Gyrokinetics GX Recon	500	v	1.3063	1	8	40	1845.6
☐	●	/FNO_Camera	nuclear-joule	✓ Completed	FNO Camera rbb Forecasting shot-agnostic discret	250			10	10	All	115180
☐	●	/FNO_Camera	obnoxious-hexagon	✓ Completed	FNO Camera rbb Forecasting shot-agnostic discret	1			10	10	All	951.32
☐	●	/FNO_MHD/pre_IAEA	grouchy-creek	✓ Completed	Multi-Blobs MultiVariable Z_Li Skip-connect Diff Re	250	rho, Phi, T	0.10326	5	10	40	5233.7
☐	●	/FNO_MHD/pre_IAEA	icy-sweep	✓ Completed	Multi-Blobs MultiVariable Z_Li Skip-connect Diff Re	250	rho, Phi, T	0.10326	5	10	40	5218.4
☐	●	/FNO_MHD/pre_IAEA	humane-score	✓ Completed	Multi-Blobs MultiVariable Z_Li Skip-connect Diff Re	500	rho, Phi, T	0.0001298	5	10	40	47528

Tags

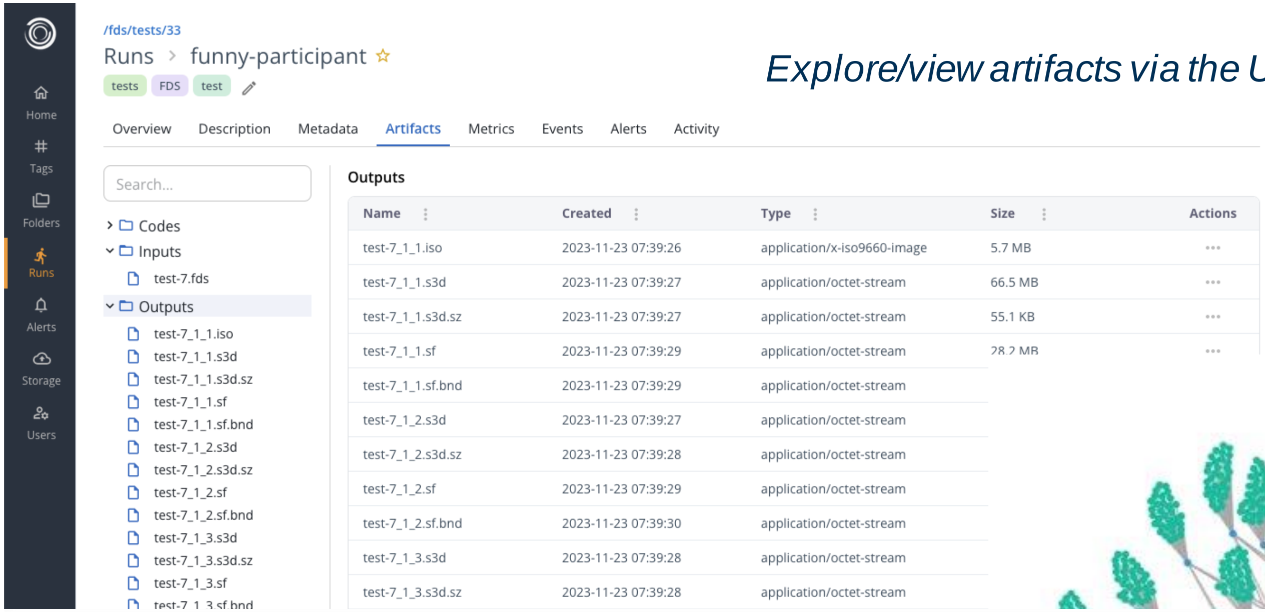
Run metadata

Artifacts

UK Atomic Energy Authority

Information about simulations & artifacts stored in the database as graphs

Artifacts themselves are stored in object storage, e.g. S3



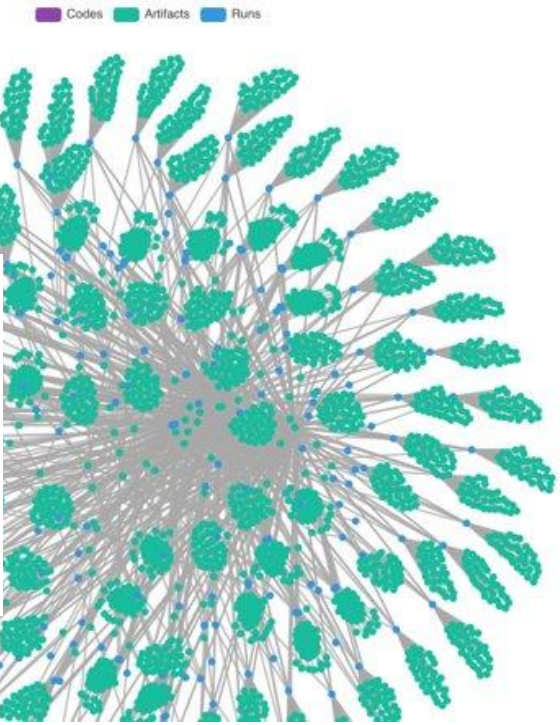
- Codes
- Inputs
  - test-7.fds
- Outputs
  - test-7_1_1.iso
  - test-7_1_1.s3d
  - test-7_1_1.s3d.sz
  - test-7_1_1.sf
  - test-7_1_1.sf.bnd
  - test-7_1_2.s3d
  - test-7_1_2.s3d.sz
  - test-7_1_2.sf
  - test-7_1_2.sf.bnd
  - test-7_1_3.s3d
  - test-7_1_3.s3d.sz
  - test-7_1_3.sf
  - test-7_1_3.sf.bnd

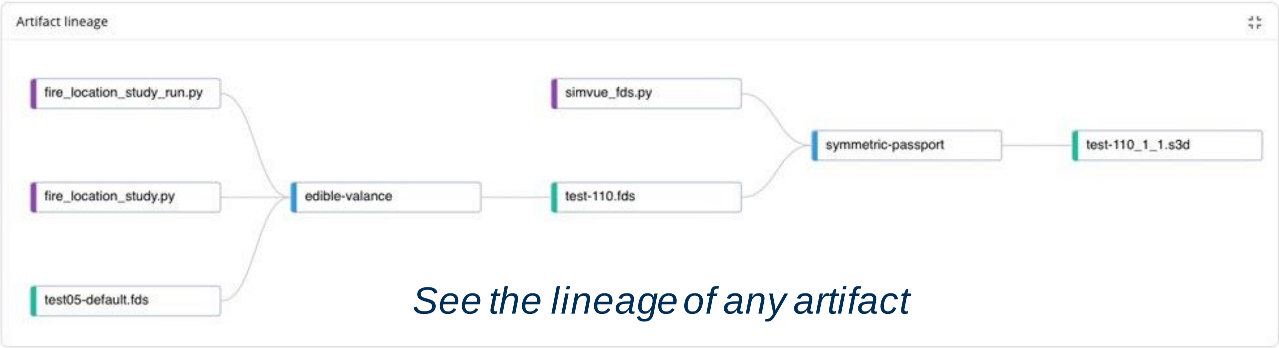
Outputs

Name	Created	Type	Size	Actions
test-7_1_1.iso	2023-11-23 07:39:26	application/x-iso9660-image	5.7 MB	•••
test-7_1_1.s3d	2023-11-23 07:39:27	application/octet-stream	66.5 MB	•••
test-7_1_1.s3d.sz	2023-11-23 07:39:27	application/octet-stream	55.1 KB	•••
test-7_1_1.sf	2023-11-23 07:39:29	application/octet-stream	28.2 MB	•••
test-7_1_1.sf.bnd	2023-11-23 07:39:29	application/octet-stream		
test-7_1_2.s3d	2023-11-23 07:39:27	application/octet-stream		
test-7_1_2.s3d.sz	2023-11-23 07:39:28	application/octet-stream		
test-7_1_2.sf	2023-11-23 07:39:29	application/octet-stream		
test-7_1_2.sf.bnd	2023-11-23 07:39:30	application/octet-stream		
test-7_1_3.s3d	2023-11-23 07:39:28	application/octet-stream		
test-7_1_3.s3d.sz	2023-11-23 07:39:28	application/octet-stream		

Explore/view artifacts via the UI

Dependencies of an artifact





See the lineage of any artifact

Alerting

Alerts enable users to automatically check if simulations meet specified criteria

- e.g. is the simulation converging?

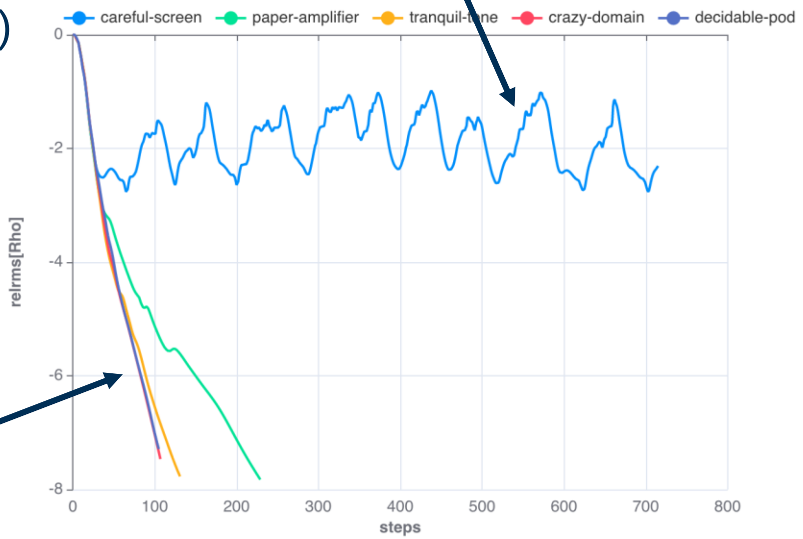
Alert source:

- Metrics
  - Single metric (e.g. "loss") or wildcard ("VIS*")
- Events
  - Log messages (e.g. from stdout/err or log file)
- User
  - Explicitly trigger an alert

Alert rule: when the alert should fire

- Metric below threshold, above threshold, inside of range, outside of range
- Event contains specified pattern

Not converging



Converging

Residuals from CFD calculations can be used to assess convergence

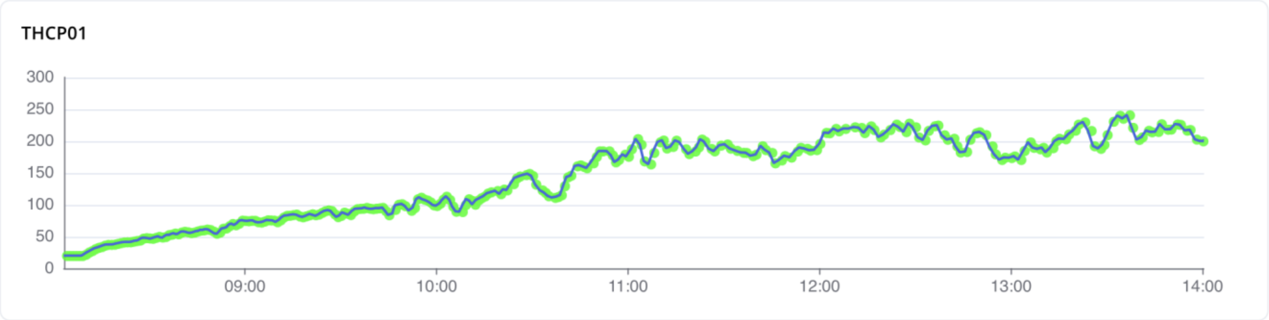
Metrics & alerting: FDS

Metrics can be used to check the results of a simulation

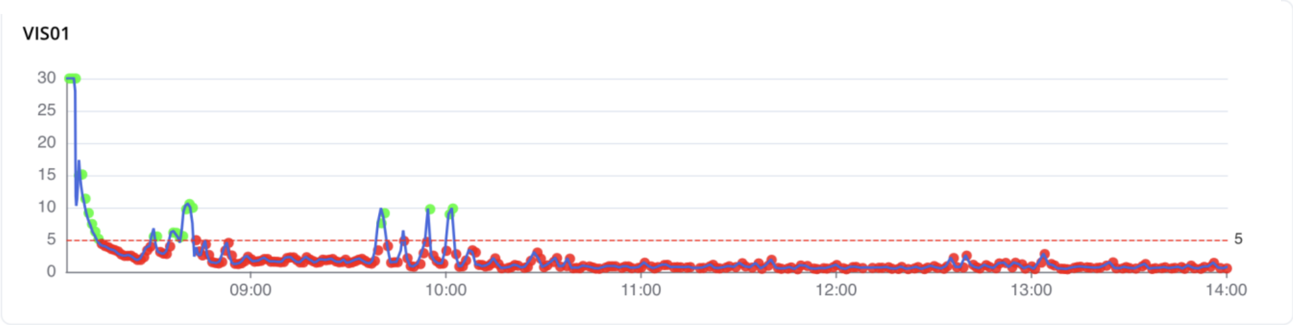
In FDS "sensors" can be placed at arbitrary places which generate metrics

Alerts can be used to:

- Check if thermocouple temperatures near the roof are < 500 C
- Check if visibility at eye level is > 5m
- Check if fractional effective does at head level < 0.5



Thermocouple temperature



Visibility

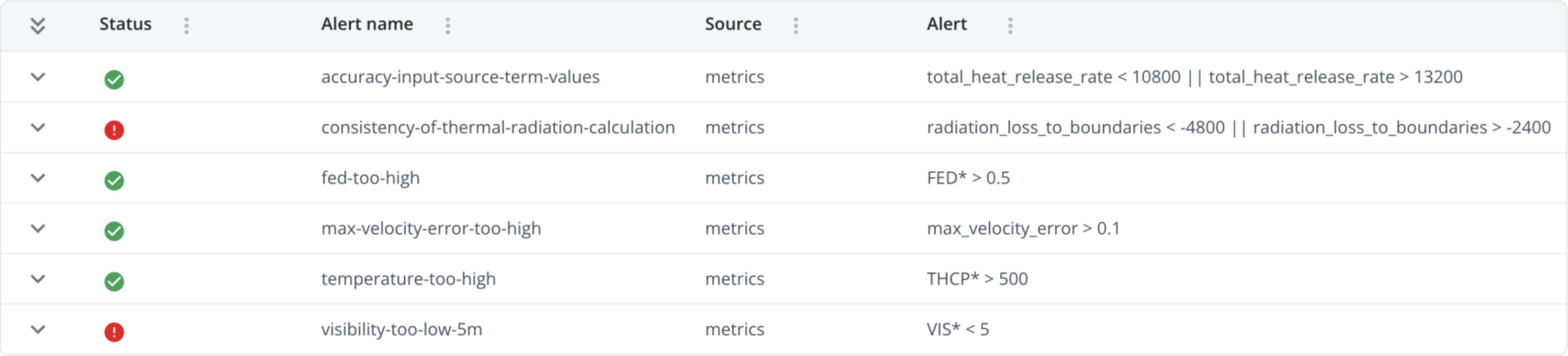
Green computing

Can use alerts to detect & automatically kill failing or unuseful simulations

- Save time, money, energy

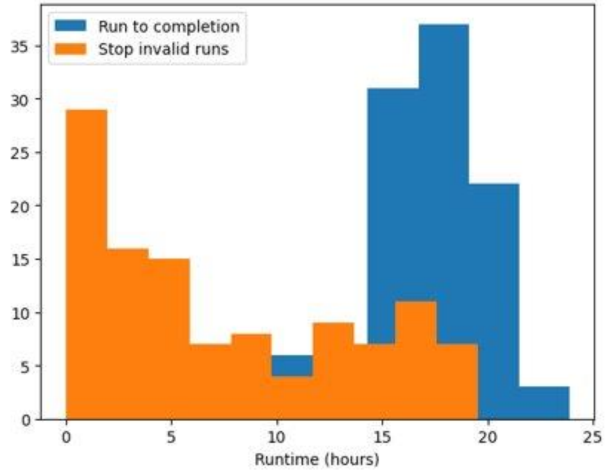
In this FDS example 6 alerts are monitoring 18 metrics

- Note: more complex FDS simulations could have hundreds of metrics each



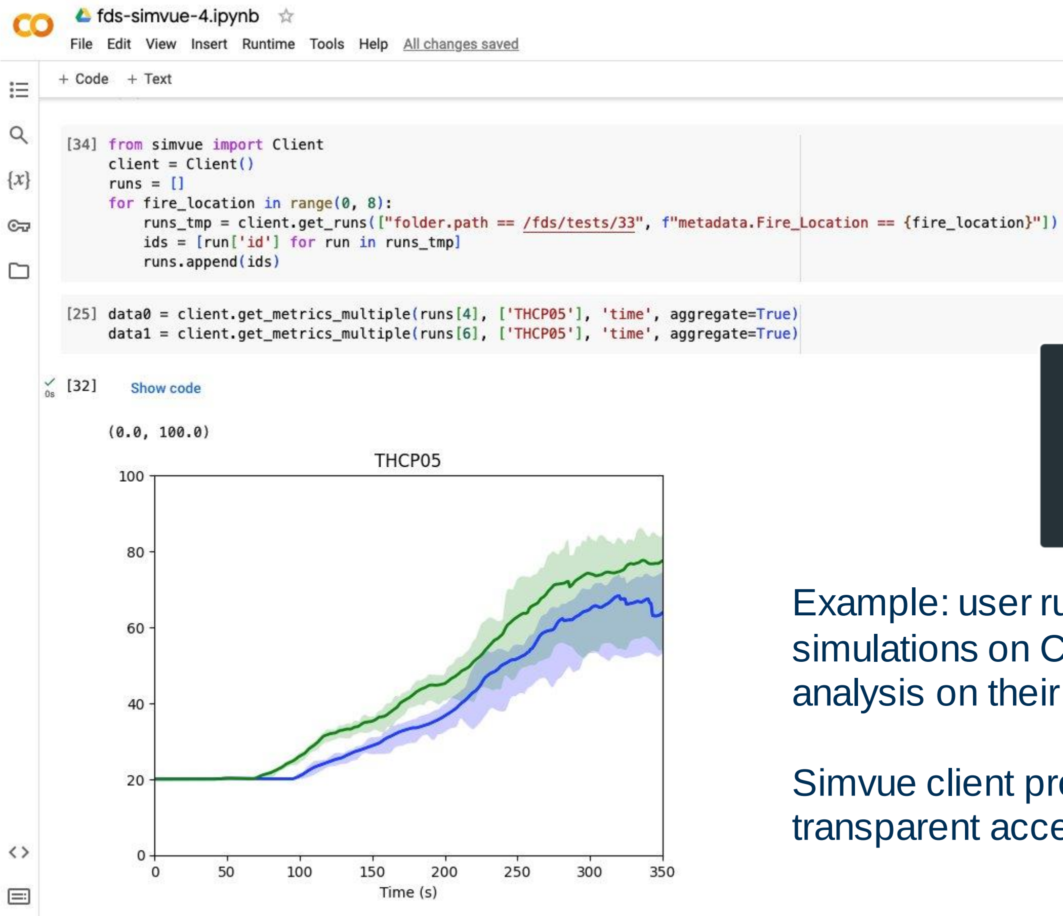
⌄	Status	Alert name	Source	Alert
⌄	✓	accuracy-input-source-term-values	metrics	total_heat_release_rate < 10800 || total_heat_release_rate > 13200
⌄	!	consistency-of-thermal-radiation-calculation	metrics	radiation_loss_to_boundaries < -4800 || radiation_loss_to_boundaries > -2400
⌄	✓	fed-too-high	metrics	FED* > 0.5
⌄	✓	max-velocity-error-too-high	metrics	max_velocity_error > 0.1
⌄	✓	temperature-too-high	metrics	THCP* > 500
⌄	!	visibility-too-low-5m	metrics	VIS* < 5

Aborting simulations as soon as possible reduces total CPU hours required by a factor of 2



Python client

Everything in Simvue (including data) can be accessed from anywhere

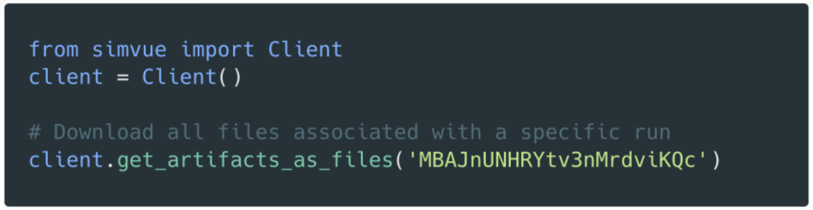


```
[34] from simvue import Client
     client = Client()
     runs = []
     for fire_location in range(0, 8):
         runs_tmp = client.get_runs(["folder.path == /fds/tests/33", f"metadata.Fire_Location == {fire_location}"])
         ids = [run['id'] for run in runs_tmp]
         runs.append(ids)
```

```
[25] data0 = client.get_metrics_multiple(runs[4], ['THCP05'], 'time', aggregate=True)
     data1 = client.get_metrics_multiple(runs[6], ['THCP05'], 'time', aggregate=True)
```

```
(0.0, 100.0)
```

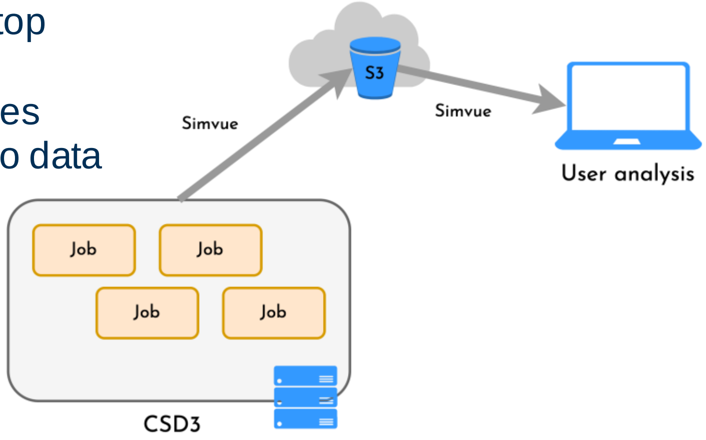
Download data using Python



```
from simvue import Client
client = Client()

# Download all files associated with a specific run
client.get_artifacts_as_files('MBAJnUNHRYtv3nMrdviKQc')
```

Example: user runs HPC simulations on CSD3, does analysis on their laptop

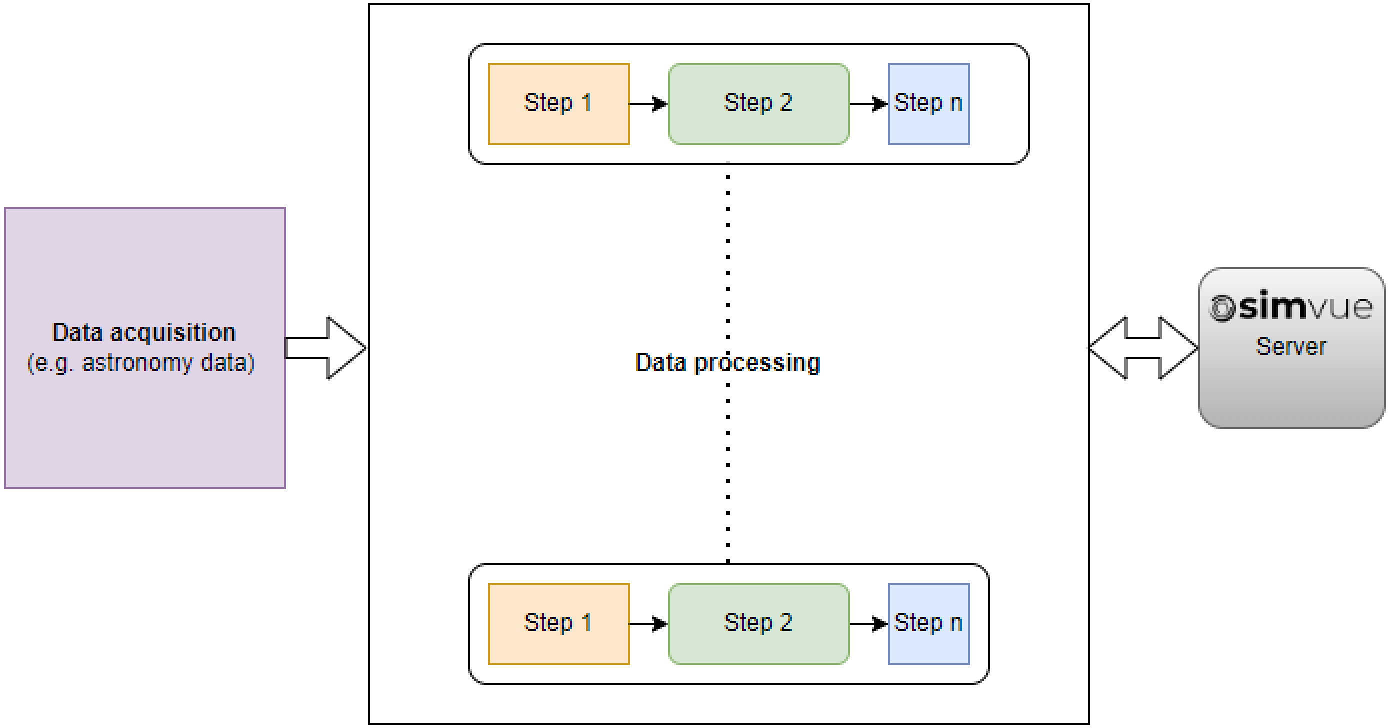
Simvue client provides transparent access to data



Data processing application

Using the client API

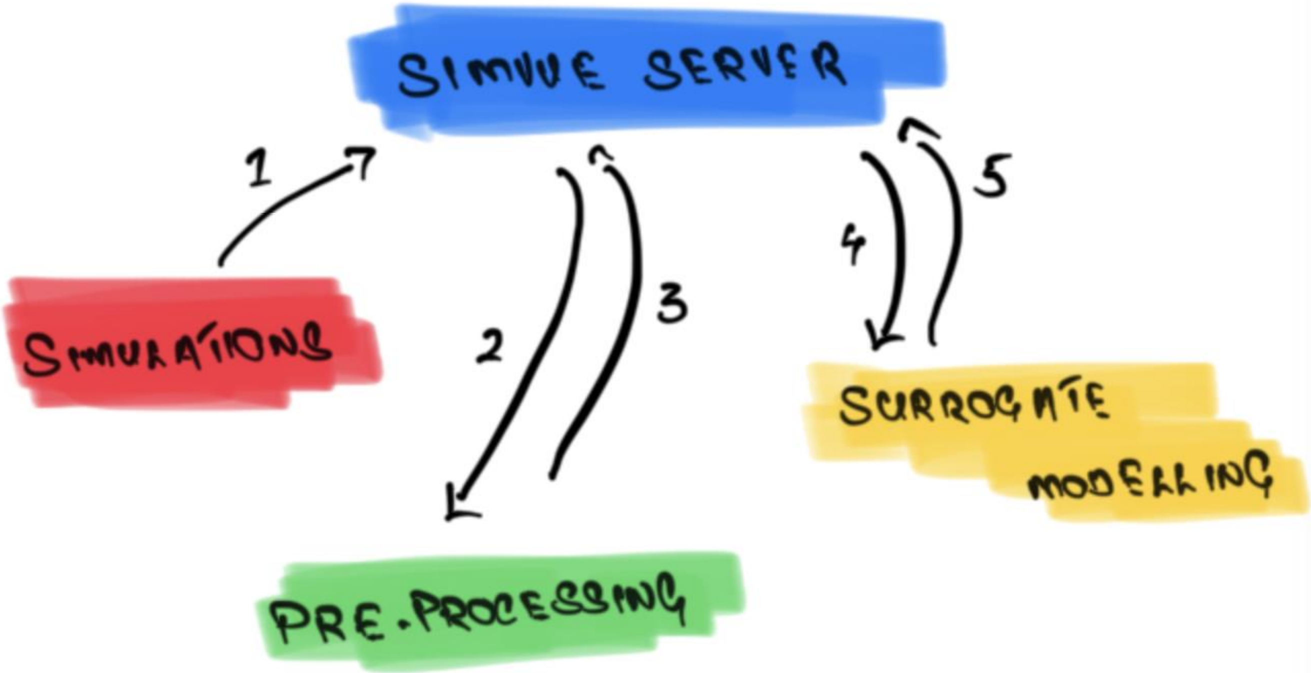
Simvue – Data processing application



- Real time resource utilisation
- Alerting
- Trigger jobs

- Single dashboard
- Store metric data
- Resource management

Surrogate Modelling Workflow



Simvue integration examples

MOOSE

Multiphysics Object-Oriented Simulation Environment

SU2 code

PyTorch

OpenFOAM®

TensorFlow

FDS-SMV

Fire Dynamics Simulator (FDS) and Smokeview (SMV)

Summary

UK Atomic Energy Authority

Innovation of Simvue used in Fusion energy

Evaluating application of Simvue for fire risk safety simulation

Facilitating green computing

Benefits in application of Simvue for Design of Experiments using AI/ML

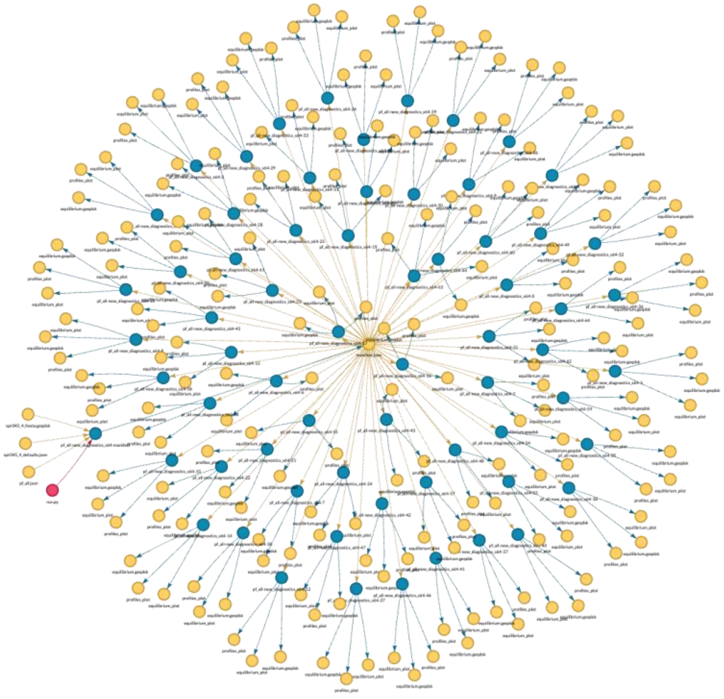
Speaking to many other application domains & building an online community

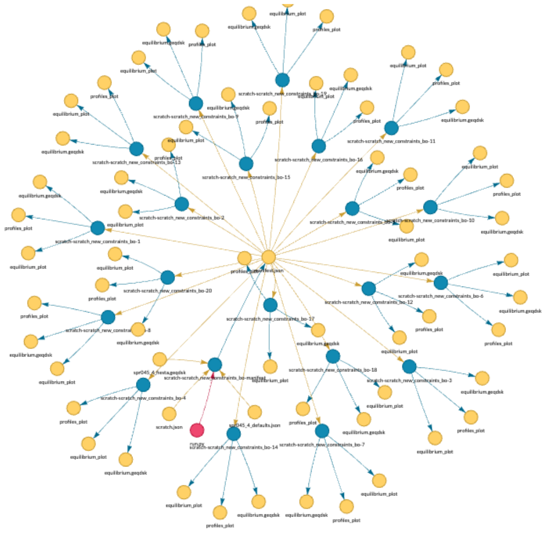
Explore application of Simvue for uses in IRIS

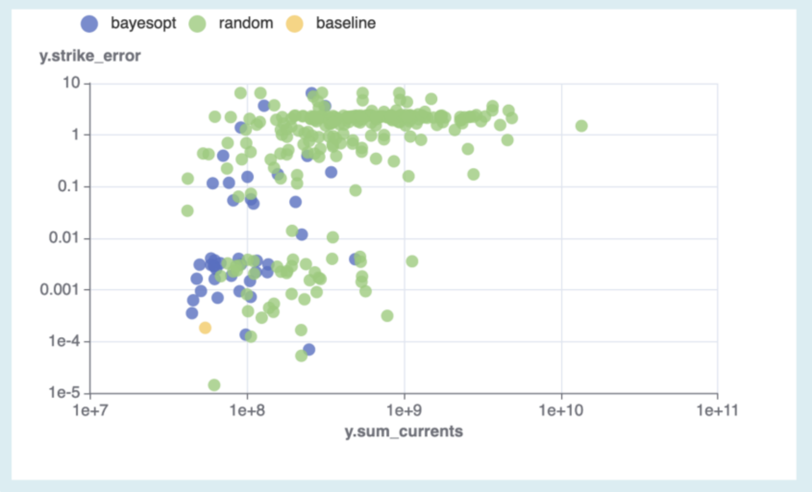
Thank you!

Additional slides

Design Of Experiments



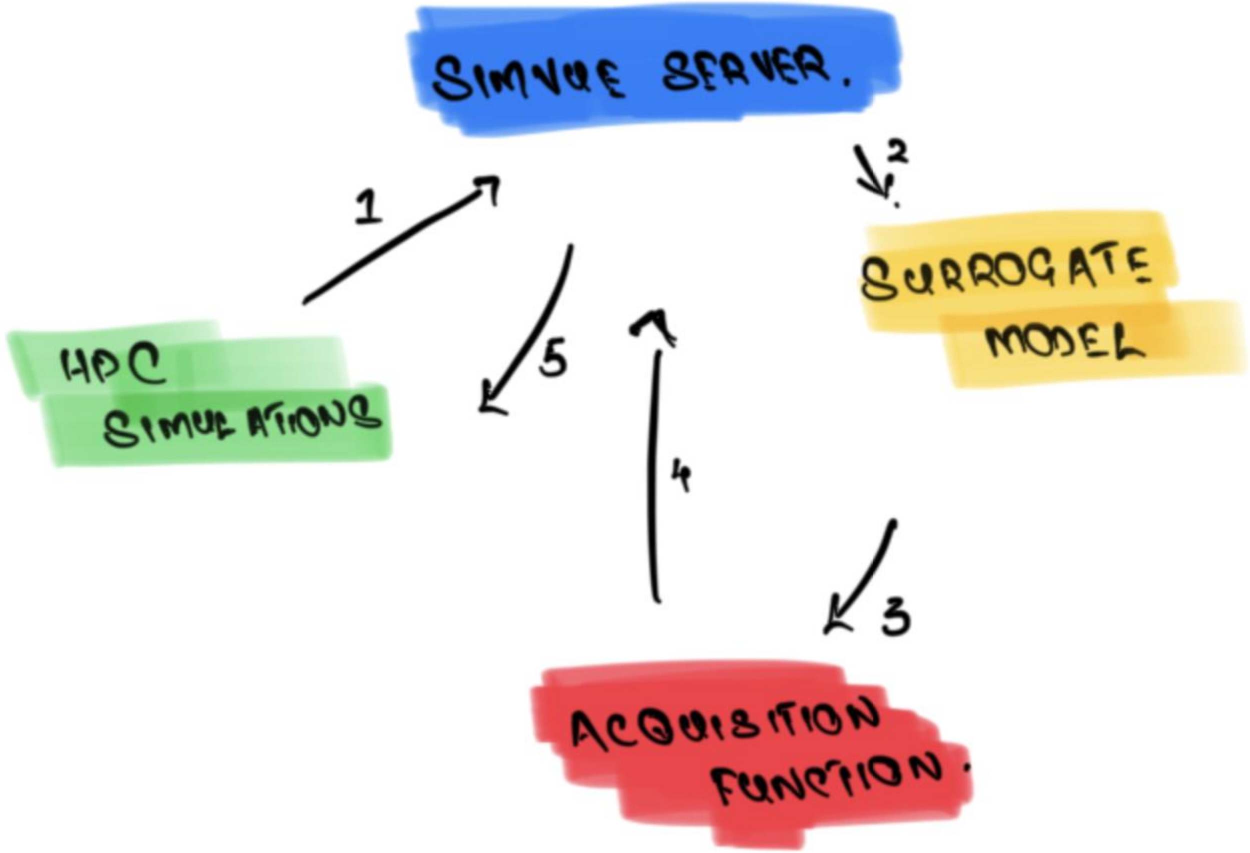




Simvue automatically sets up a graph-based workflow, seamlessly integrating across your simulation and your AI/ML models.

Work done by Timothy Nunn

Bayesian Optimisation for PF Coil Placement



Simvue's Software Innovation Journey

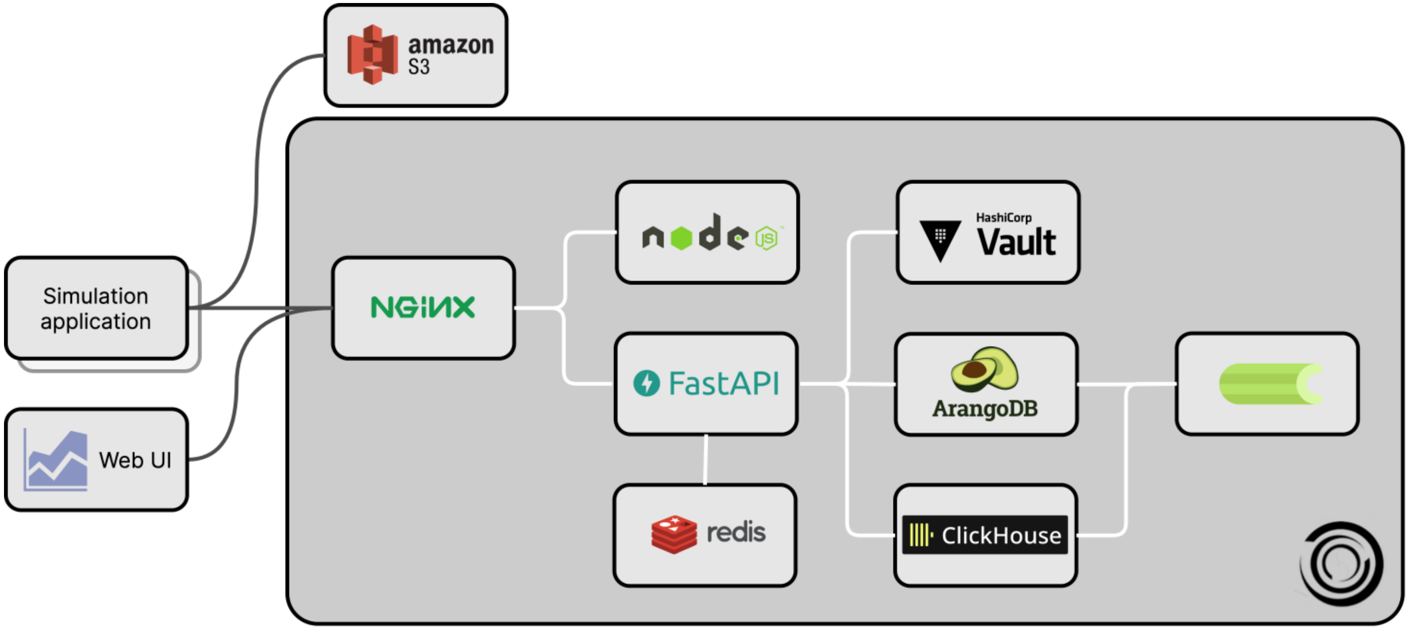
Satisfy a need in Fusion Energy	Suitability outside Fusion Energy –	Business model and add more application domains –	Commercialisation
"Creation Phase"	**"Explore Phase"**	**"Expand Phase"**	**"Let's go! Phase"**
Think generic design Feedback from users Discussion with Innovation team IP & Licence evaluation	Application to Fire Safety Simulation Generalise application Design patterns	STFC (IRIS) Health (HDRUK) Weather (Met Office) Fire safety	Product enhancement, development Development support model Active use by a global community

Technology

Internally uses only open-source software

Data stored in external cloud object storage

Deployment using Docker Compose or Kubernetes



Simvue as a batch system

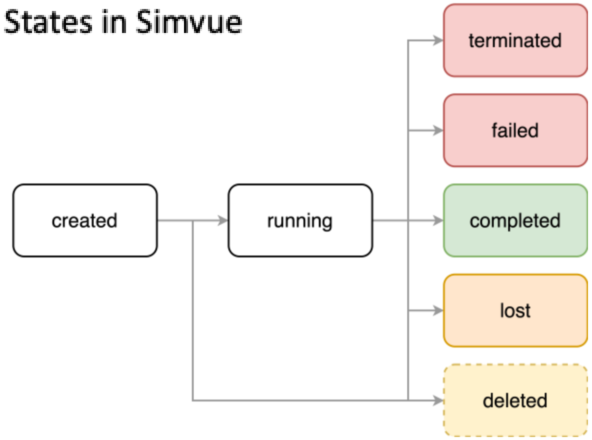
Create runs in Simvue

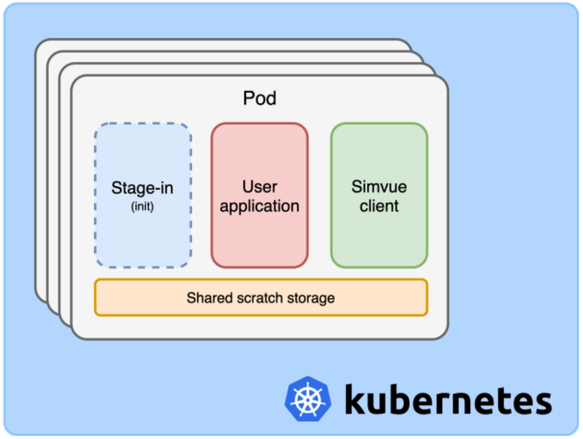
- Leave in the "created" state
- Include input files, container image name, required resources as metadata

Wrote a PoC service which ensures Kubernetes jobs are created for each run defined in Simvue

Use pods with user application & Simvue client in separate containers

- Allows use of existing containers to be used without modification
- Use an init container to stage-in data before job runs





Jisc

OCRE Open Clouds for Research Environments

Amazon Elastic Kubernetes Service