

Scalable Drug Discovery with Apache Beam

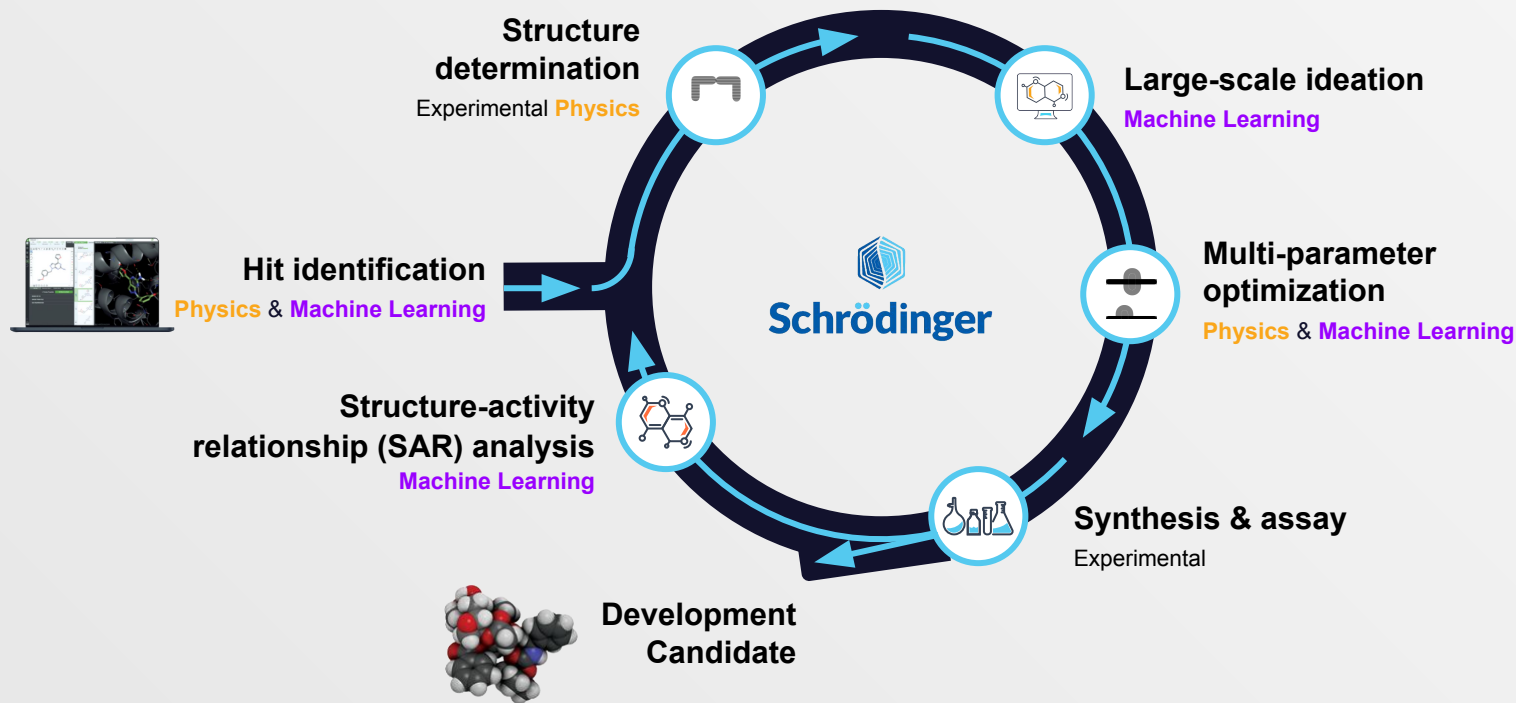


Agenda



- Schrodinger
 - The Drug Discovery Platform
 - A Brief History of Schrodinger Workflow Engines
- Schrodinger Beam → Seam
 - SeamRunner
 - Seam Transform Catalog
- Applications
 - AutoDesigner: R-Group Enumeration
 - Crystal Structure Prediction
- Future Direction

The Schrödinger Drug Discovery Platform



How Schrodinger Software is Run

User



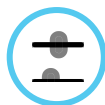
Large-scale ideation

Machine Learning



**Structure-activity
relationship (SAR) analysis**

Machine Learning



**Multi-parameter
optimization**

Physics & Machine Learning



Schrödinger

How Schrodinger Software is Run

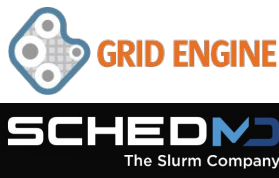
User



```
$SCHRODINGER/run  
glide input.mae
```

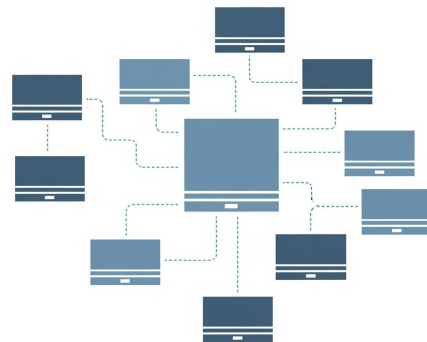
Job

Scheduler

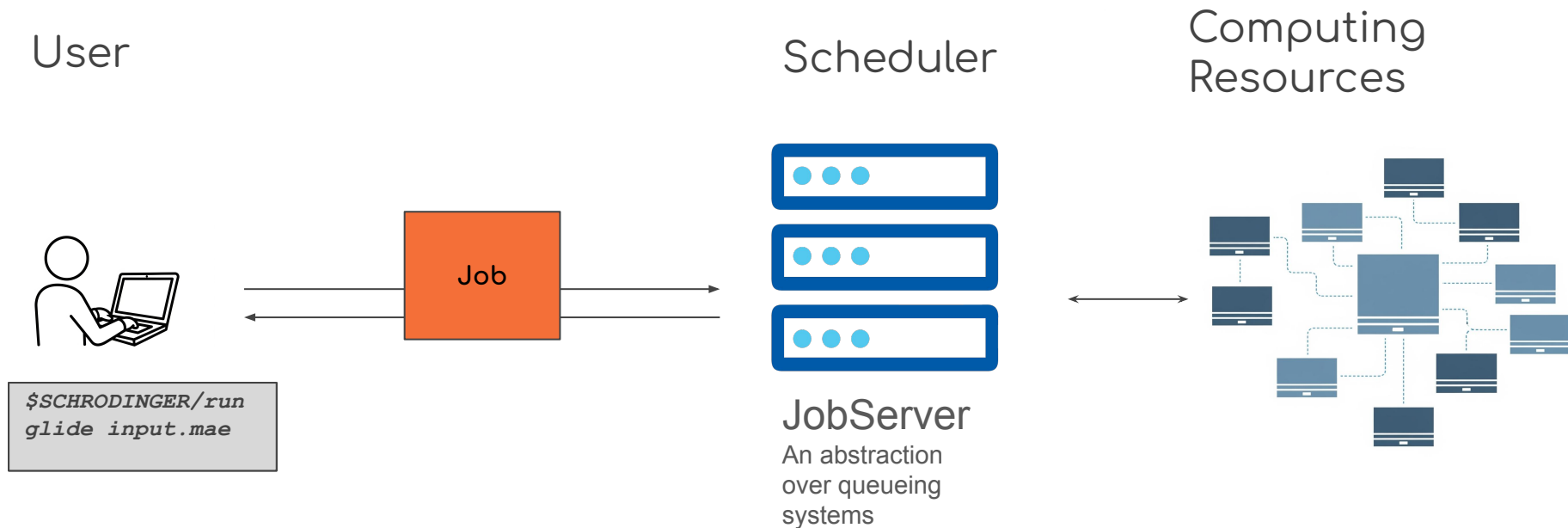


Portable Batch System (PBS)

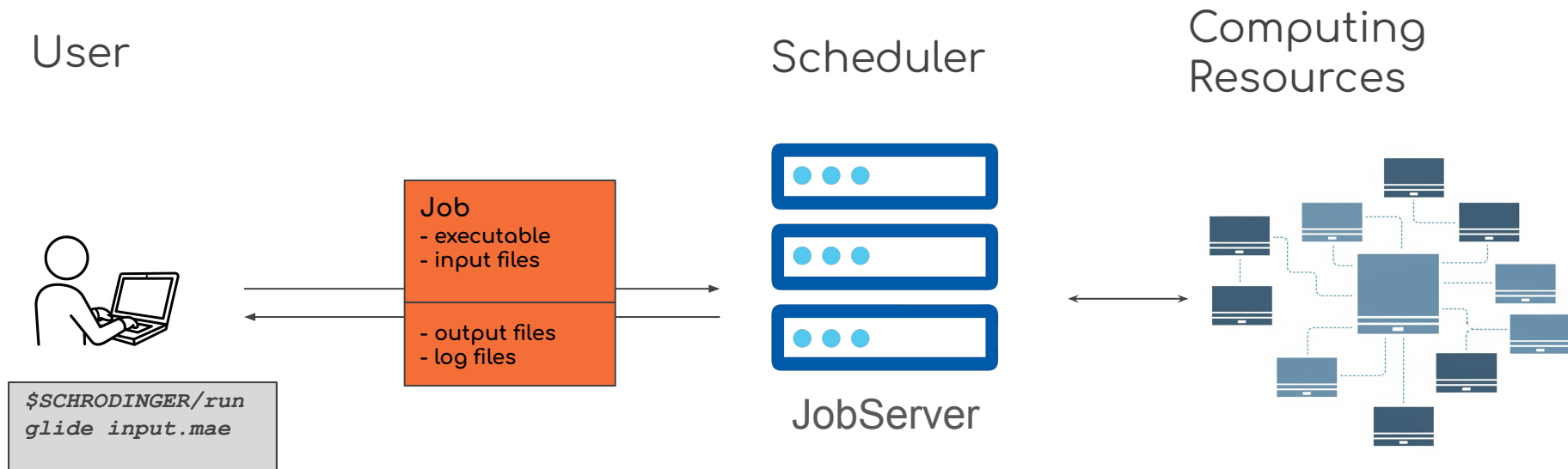
Computing
Resources



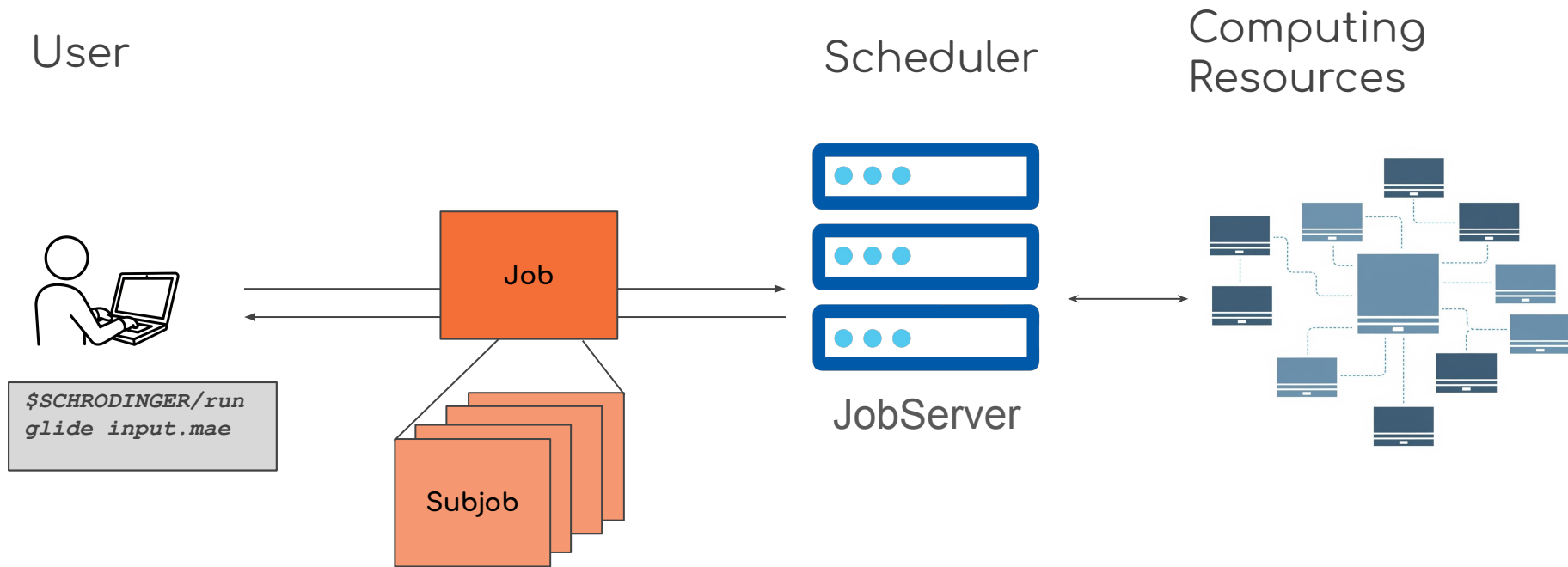
How Schrodinger Software is Run



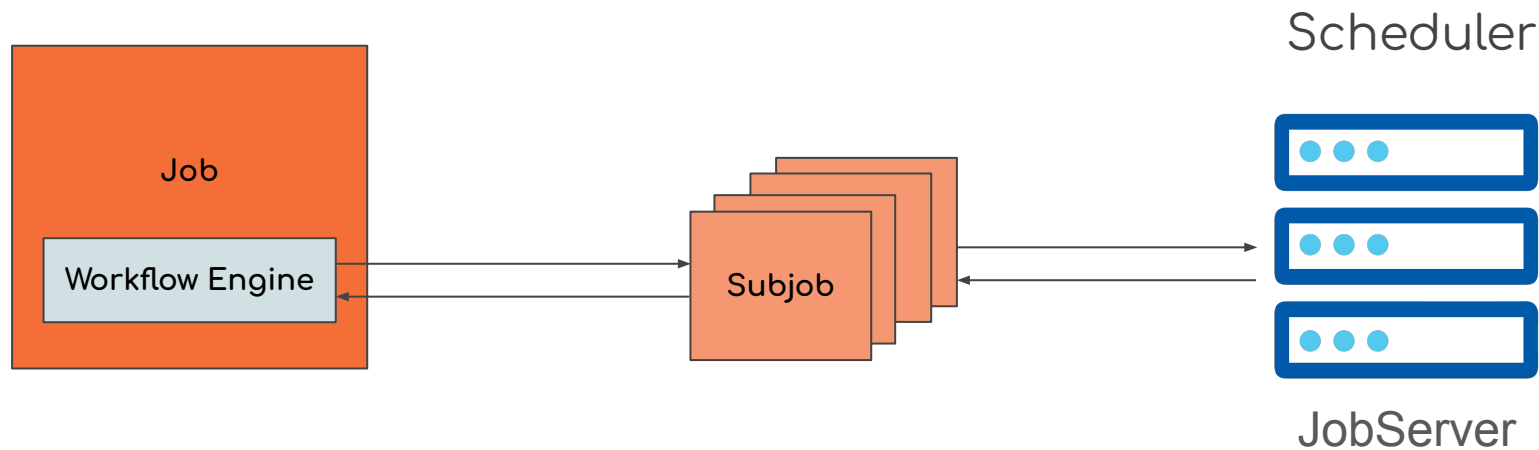
How Schrodinger Software is Run



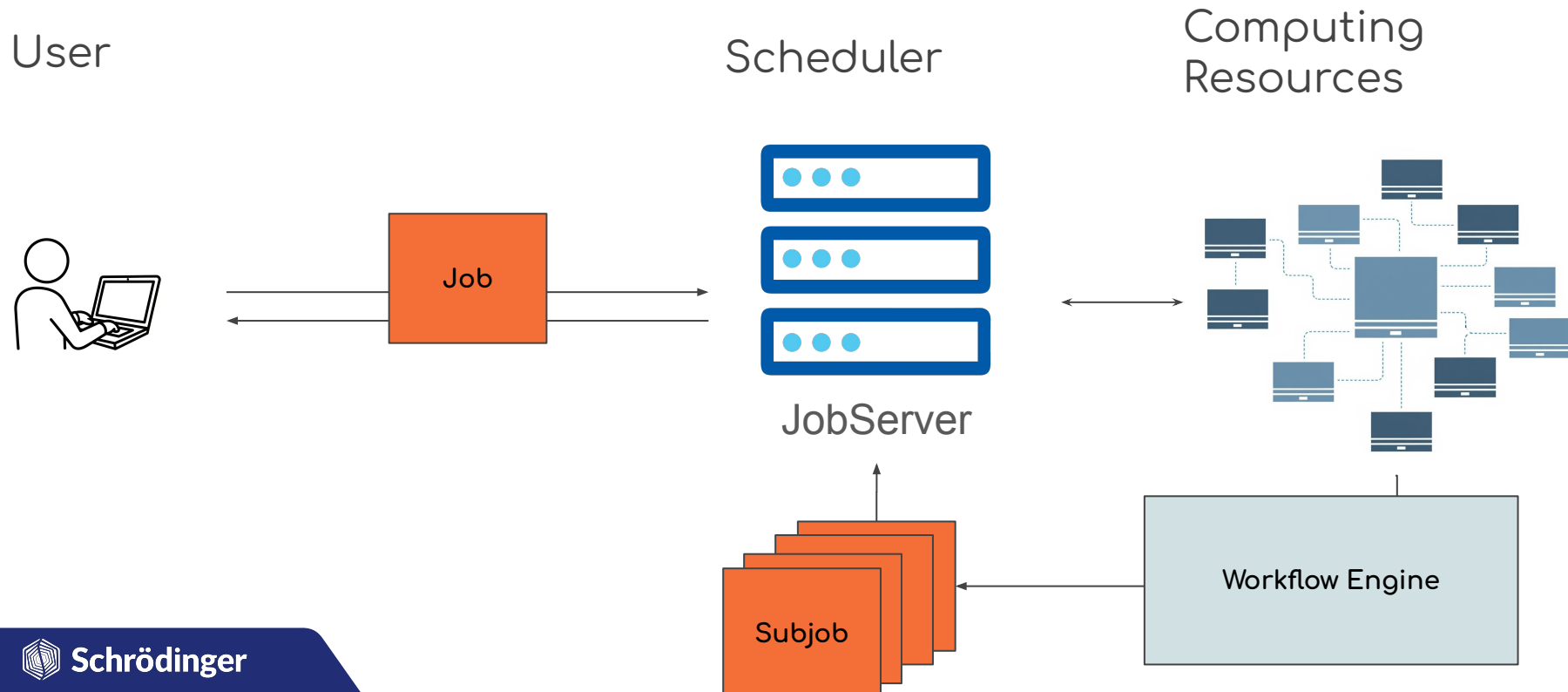
How Schrodinger Software is Run



How Schrodinger Software is Run

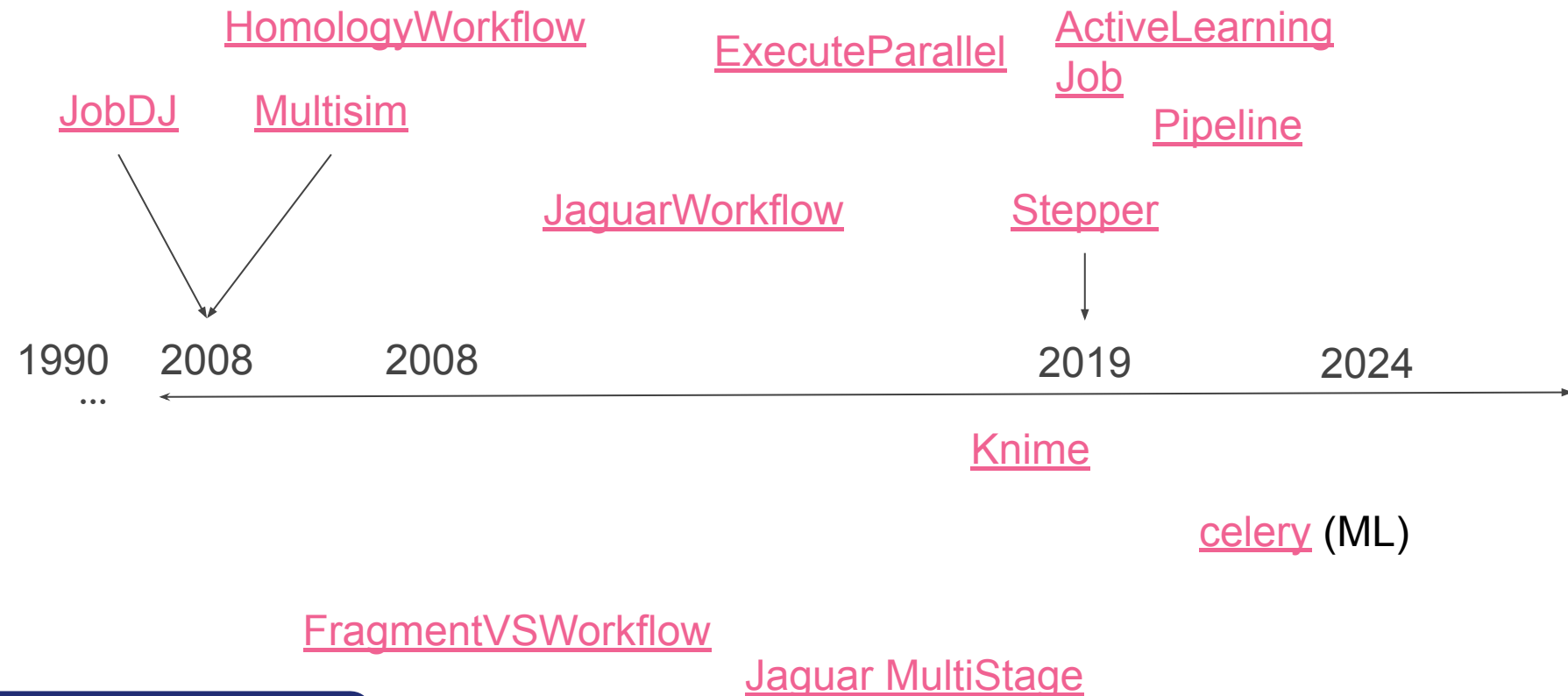


How Schrodinger Software is Run



Workflow Engines at Schrödinger

EpikX Local Train



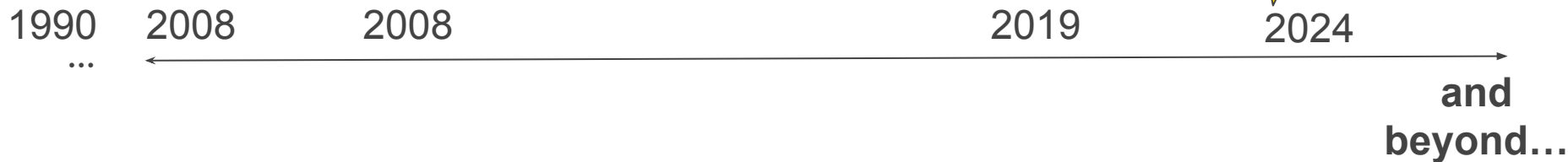
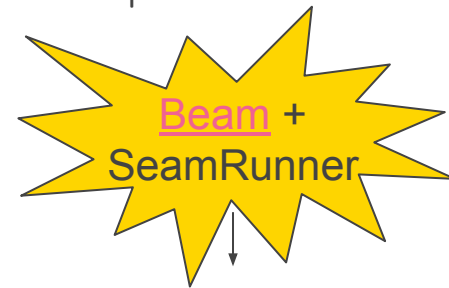
Workflow Engine Feature Matrix

	Observability	Worker Autoscaling	Cloud Integrations	Text-based declarative workflows	Checkpointing	Expressive and usable API
stepper	✓	✓	✓	✗	✗	✗
active learning	✗	✗	✓	✗	✓	✗
meta	✗	✗	✗	✓	✗	✗
multisim	✗	✗	✗	✓	✗	✗
...						✗

Workflow Engines at Schrödinger

SeamRunner

- Beam runner built to execute workflows on JobServer-enabled compute clusters



Workflow Engine Feature Matrix

	Observability	Worker Scaling	Cloud Integrations	Text-based declarative workflows	Checkpointing	Expressive and usable API
SeamRunner	✓	✓	✓	✓	✓	
Apache Beam						✓

SeamRunner Features

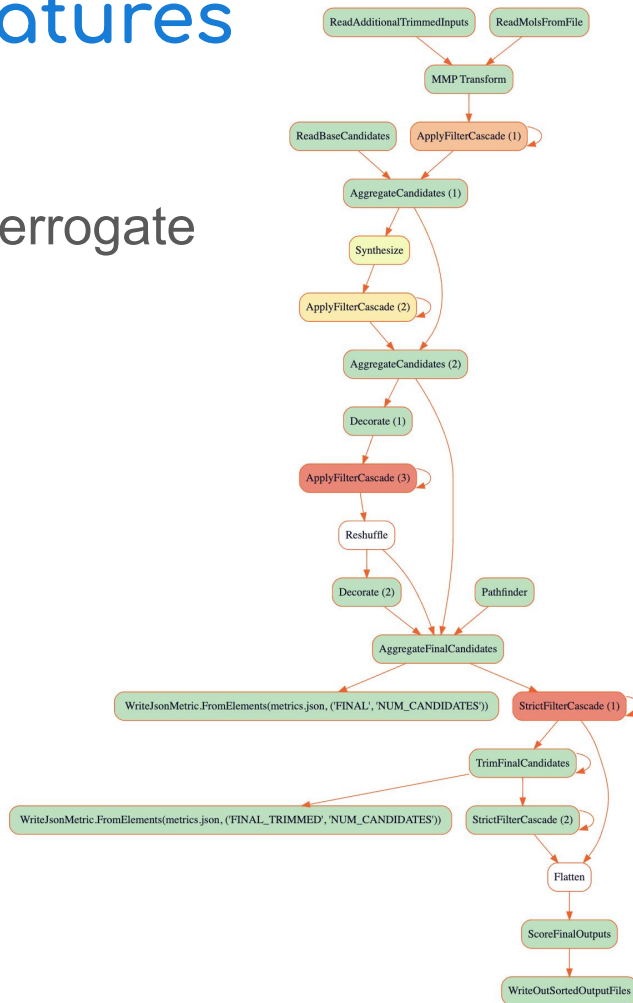
- Log aggregation

```
2025-05-12T13:35:49 EDT
LOG 2025-05-07T00:15:28.005 seam_log pipeline.clean.log[0]
May07 12:15:28AM [seam_orchestrator]-|- USER =====
May07 12:15:28AM [seam_orchestrator]-|- USER Executing pipeline
May07 12:15:28AM [seam_orchestrator]-|- USER Using following non-default SeamRunner options:
May07 12:15:28AM [seam_orchestrator]-|- USER SeamOptions(max_dead_bundles=3,
max_worker_failure_rate=0.25,
max_workers=256,
worker_timeout=25.0)
May07 12:15:28AM [seam_orchestrator]-|- USER
May07 12:15:28AM [seam_orchestrator]-|- USER =====
May07 12:15:28AM [seam_orchestrator]-|- USER
May07 12:15:49AM [seam_orchestrator]-|- USER -----
May07 12:15:49AM [seam_orchestrator]-|- USER Running stage 1/171
May07 12:15:49AM [seam_orchestrator]-|- USER (EXECUTABLE_STAGE)
May07 12:15:49AM [seam_orchestrator]-|- USER Includes transforms:
May07 12:15:49AM [seam_orchestrator]-|- USER - Pathfinder [1/11]
May07 12:15:49AM [seam_orchestrator]-|- USER
May07 12:15:49AM [seam_orchestrator]-|- USER (Full set of transforms in this stage:)
May07 12:15:49AM [seam_orchestrator]-|- USER - Design/Pathfinder/ReadMolsFromFile/Create/FlatMap(<lambda at core.py:3970>)
May07 12:15:49AM [seam_orchestrator]-|- USER - Design/Pathfinder/ReadMolsFromFile/Create/Map(decode)
May07 12:15:49AM [seam_orchestrator]-|- USER - Design/Pathfinder/ReadMolsFromFile/FlatMap(<lambda at chemio.py:397>)
May07 12:15:49AM [seam_orchestrator]-|- USER - Design/Pathfinder/ReadMolsFromFile/FlatMap(_read_mols_from_file)
May07 12:15:49AM [seam_orchestrator]-|- USER
May07 12:15:49AM [seam_orchestrator]-|- USER Input PColls:
May07 12:15:49AM [seam_orchestrator]-|- USER - 208_Impulse_outputs(tag='main_input'): 1 elements (2.0 B)
May07 12:15:49AM [seam_orchestrator]-|- USER ...
```



SeamRunner Features

- **Workflow Watcher**
 - Visualize and interrogate your pipelines



Info

Status: Completed
Start Time: 2025-Jun-27 08:28 AM
End Time: 2025-Jun-27 08:28 AM
Duration: 7 seconds
Compute Time: 7 seconds

View

Color By

- ☒ Compute Time
- ☐ License Requirements

Actions

[Download Logs](#)

☐ Only sanitized logs

Transform Information

Name: LigandPrep
Licenses: • LIGPREP_MAIN: 1

Inputs: • 45 Mol(s)

Outputs: • 70 Structure(s)

CPU Time: 5 seconds

Display Data

PTransform Type: [LigandPrep](#)



Seam Tooling

- **Resource Management**
 - Per-transform licenses
 - GPU requirements
 - Memory requirements

```
@with_license_requirements({license.LIGPREP_MAIN: 1})  
@requires_min_ram("16GB")  
class PrepareLigand(beam.PTransform):  
  
    ...
```

Seam Transform Catalog

Seam Transform Catalog

See also:

- [Beam transform catalog](#)
- [Seam YAML transform catalog](#)

Transform	Description
ReadStructuresFromFile	Read a file (or files) containing structures and return a PCollection of Structure objects.
WriteStructuresToFile	Write a PCollection of Structure objects to a file.
ReadMolsFromFile	Read a file (or files) containing molecules
WriteMolsToFile	Write a PCollection of Mol objects to a file.
FixedSample	Split a PCollection into two PCollections: a random sample of the input PCollection and the remaining elements.
Top	Split a PCollection into two PCollections: the largest n elements of the input PCollection and the remaining elements.
RandomSample	Create a PCollection made up of a random sample of the input PCollection.
Smallest	Create a PCollection containing the smallest element(s) of the input PCollection.
IPythonInspect	Starts an IPython shell to interactively inspect the contents of a PCollection.

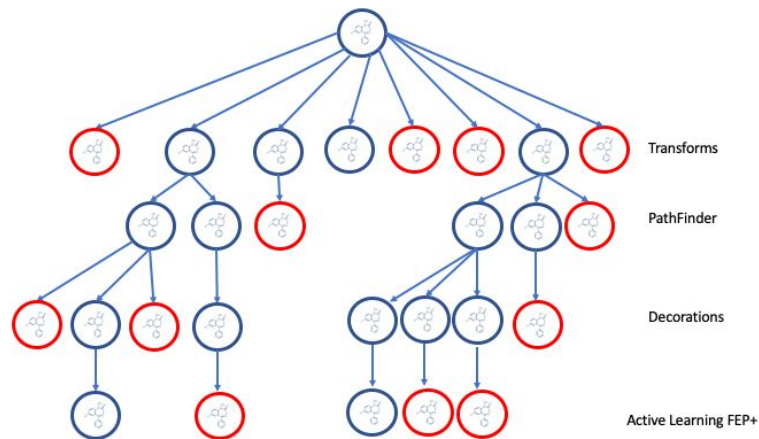
Scientific Transforms

Transform	Description
Dock	Dock ligands into a receptor using Glide.
ProteinPrep	Clean up and prepare a protein using PrepWizard
LigandPrep	Clean up and prepare a ligand and generate tautomers.
Refine	Refine a protein receptor using Prime.

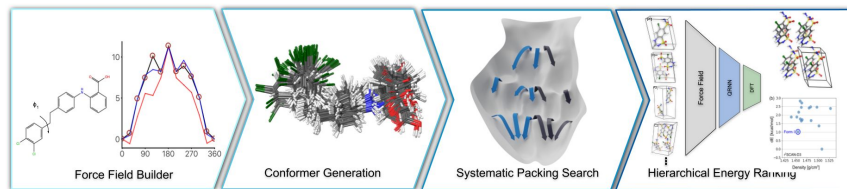


Seam-based Applications

AutoDesigner: R-Group Enumeration



Crystal Structure Prediction



Scaled to 10k CPU cores and TB scale data
PCollections


Challenges

Writing a
runner is hard!


Getting Started With Implementing a Runner

External > Inbox x

◆ Summarize this email




Joey Tran Thu, Jun 22, 2023, 8:39 AM ★
Hi Beam community! I'm interested in trying to implement a runner with my company's execution environment but...




Jack McCluskey via user Thu, Jun 22, 2023, 10:08 AM ☆
Hey Joey, The best resource to look at, at the moment, is likely Robert Burke's Prism runner that he is implementi...

40



Joey Tran Sun, Sep 10, 2023, 9:42 AM ☆
Implemented side inputs successfully. It was made easy once I saw how fn_api_runner does it. Is the WindowGro...



Robert Bradshaw <robertwb@google.com> Mon, Sep 11, 2023, 2:06 PM ☆ ↩ ⋮
to me ▾
Yes, this sounds like a good idea.



Challenges

Developer adoption

- Beam learning curve can be steep



Future Direction

- Streaming support
- YAML-based workflows
- Improvements in scientific prototyping usability
- Worker/Server-based execution

```
pipeline:  
  type: chain  
  transforms:  
    - type: CreateStructures  
      config:  
        elements:  
          - smiles: "CC(=O)C1=CC=C(C=C1)C(=O)O"  
            name: "Aspirin"  
          - smiles: "Cn1cnc2c1c(=O)n(C)c(=O)n2C"  
            name: "Caffeine"  
          - smiles: "CC(C)[C@H]1CCC(C)CC1O"  
            name: "Menthol"  
          - smiles: "CC(=O)NC1=CC=C(C=C1)OC(C)C=CC=CC2=CC=C(C=C2)O"  
            name: "Capsaicin"  
          - smiles: "CN1CCCC1C2=CN=CC=C2"  
            name: "Nicotine"  
    - type: LigPrep  
      config:  
        arg_string: "-nt -epik -s 32"
```



Acknowledgements

AutoDesigner / Seam Team

Nithin Chintala

René Kanters

Pieter Bos

Namit Negi

Nayan Mathur

PyDev Team

Thomas Oh

Alex Malao

Marco Sanchez-Ayala

Crystal Structure Prediction Team

Efrem Braun

Ethan Alguire

Ding Wei

Renke Huang

Apache Beam Dev

Robert Bradshaw

Danny McCormick

Jack McCluskey

Kenneth Knowles

Robert Burke

dev@beam.apache.org



QUESTIONS?

Email questions to
joey.tran@schrodinger.com

