An update on the ExaWorks Project

Parsl & FuncX Fest ‘22

Daniel Laney

LLNL-PRES-835634
## The Team

<table>
<thead>
<tr>
<th>Name</th>
<th>Institution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dan Laney</td>
<td>Lawrence Livermore National Laboratory</td>
</tr>
<tr>
<td>Kyle Chard</td>
<td>Argonne National Laboratory</td>
</tr>
<tr>
<td>Shantenu Jha</td>
<td>Brookhaven National Laboratory</td>
</tr>
<tr>
<td>Rafael Ferreira da Silva</td>
<td>Oak Ridge National Laboratory</td>
</tr>
<tr>
<td>Dong H. Ahn</td>
<td>Lawrence Livermore National Laboratory</td>
</tr>
<tr>
<td>Todd Munson</td>
<td>Argonne National Laboratory</td>
</tr>
<tr>
<td>Aymen Alsaadi</td>
<td>Brookhaven National Laboratory</td>
</tr>
<tr>
<td>Ben Clifford</td>
<td>Argonne National Laboratory</td>
</tr>
<tr>
<td>James Corbett</td>
<td>Lawrence Livermore National Laboratory</td>
</tr>
<tr>
<td>Mihael Hategan</td>
<td>Argonne National Laboratory</td>
</tr>
<tr>
<td>Ketan Maheshwari</td>
<td>Oak Ridge National Laboratory</td>
</tr>
<tr>
<td>Andre Merzky</td>
<td>Brookhaven National Laboratory</td>
</tr>
<tr>
<td>Zeke Morton</td>
<td>Lawrence Livermore National Laboratory</td>
</tr>
<tr>
<td>Mikhail Titov</td>
<td>Brookhaven National Laboratory</td>
</tr>
<tr>
<td>Matteo Turilli</td>
<td>Brookhaven National Laboratory</td>
</tr>
<tr>
<td>Andreas Wilke</td>
<td>Argonne National Laboratory</td>
</tr>
<tr>
<td>Justin M. Wozniak</td>
<td>Argonne National Laboratory</td>
</tr>
</tbody>
</table>

## Previous Contributors

- Stephen Herbein  
  Lawrence Livermore National Laboratory
- Yadu Babuji  
  Argonne National Laboratory
## Exascale Computing Project (ECP)

Seven-year, $1.8B project that aims to accelerate R&D, acquisition, and deployment of **exascale** computing capability to DOE

Six core national laboratories are focused on software, applications, hardware, system engineering and testbed platforms

| Performant mission and science applications @ scale |
|---------------------------------|---------------------------------|---------------------------------|---------------------------------|
| Aggressive R&D Project | Mission apps & integrated S/W stack | Deployment to DOE HPC Facilities | Hardware tech advances |

### Application Development (AD)
- Develop and enhance the predictive capability of applications critical to the DOE
- 24 applications including national security, energy, earth systems, economic security, materials, and data

### Software Technology (ST)
- Deliver expanded and vertically integrated software stack to achieve full potential of exascale computing
- 67 unique software products spanning programming, models and runtimes, math libraries, data and visualization

### Hardware and Integration (HI)
- Integrated delivery of ECP products on targeted systems at leading DOE HPC facilities
- 6 US HPC vendors focused on exascale node and system design, application integration and software deployment to facilities
Scientific computing workflows underlie a significant number of projects in the Exascale Computing Project (ECP) portfolio.

Many teams are creating infrastructures to:
- Couple multiple applications
- Manage jobs, sometimes dynamically
- Orchestrate compute/analysis and manage data

There is duplication of effort in these infrastructures.

These customized workflows incur significant costs to port, maintain and scale.

These tools do not always interface with facilities smoothly.

The costs could be minimized by creating a reliable, scalable, portable software development kit (SDK) for workflows.
Our approach will ensure exascale readiness of a wide range of ECP workflows and improve their long-term sustainability

Partner with ECP AD and other teams for co-design and adoption of ExaWorks SDK to address their workflow problems

Engage with DOE compute facilities to support deployment and use of workflows at scale; contribute requirements for next-generation systems

Curate community SDK to enable robust, scalable, portable, performant workflows; progressively increase the availability of composable workflow components

Lead the workflows community towards interoperability and reuse; build the case for future standardization for long-term sustainability

The ExaWorks SDK is packaged, deployed, and tested using E4S and ECP CI infrastructure
ExaWorks is *not* funded to build another workflow system

We are funded to provide a **production-grade Software Development Kit (SDK)** for exascale workflows

**Our SDK democratizes access** to hardened, scalable, and interoperable workflow management technologies and components

**Approach**
- Community policies for software quality (based on E4S)
- Open community-based design and implementation process
- Ensure scalability of components on **Exascale Systems**
- Standard packaging and testing
- Work toward shared capabilities in the SDK
ExaWorks SDK brings together five seed technologies currently impacting ECP applications

- **Scientific workflows SDK includes four seed technologies**
  - **Flux** – hierarchical resource and job management software
  - **Parsl** – flexible and scalable parallel programming library for Python
  - **RADICAL** – component-based workflow middleware
  - **Swift/T** – high performance dataflow computing
We are engaging Workflow Communities and Computing Facilities

- **Workflows Community Summit: Researchers**
  - Brought together workflows leaders to develop a vision for community activities
  - [https://doi.org/10.5281/zenodo.4606958](https://doi.org/10.5281/zenodo.4606958)

- **Workflows Community Summit: Developers**
  - Explored technical approaches for realizing the community vision
  - [https://doi.org/10.5281/zenodo.4915801](https://doi.org/10.5281/zenodo.4915801)

- **Workflows Community Summit: Facilities**
  - Small group of facility representatives discussing facilities perspectives, challenges, and opportunities


First Workflows Community Summit: 45 participants, 27+ workflow systems

Second Workflows Community Summit: 75 participants

Third Workflows Community Summit: 9 participants, 8 facilities/centers (ALCF, OLCF, NERSC, LC, BNL, PSC, NREL, NCSA)

[https://exaworks.org/summit.html](https://exaworks.org/summit.html)
A portable, flexible next gen job scheduler for emerging workflows

- Open-source project in active development at flux-framework GitHub organization with ~15 team members

- Single-user and multi-user (a.k.a. system instance) modes
  - Single-user mode has been used in production for ~3 years
  - Multi-user mode is having its debut on LLNL’s Linux clusters

- Plan of record for LLNL’s El Capitan exascale system

- Designed with heterogeneous systems and advanced workflows in mind

- Rich Python and C/C++ API’s
**Parsl: a parallel programming library for Python**

**Apps** define opportunities for **parallelism**

- Python apps call Python functions
- Bash apps call external applications

Apps return “futures”: a proxy for a result that might not yet be available

Apps run concurrently respecting dataflow dependencies. Natural parallel programming!

Parsl scripts are independent of where they run. Write once run anywhere!

Parsl scales to 100,000s of tasks on the largest HPC systems

```python
@python_app
def hello():
    return 'Hello World!'
print(hello().result())
```

Hello World!

```bash
@bash_app
def echo_hello(stdout='echo-hello.stdout'):
    return 'echo "Hello World!"'
echo_hello().result()
with open('echo-hello.stdout', 'r') as f:
    print(f.read())
```

Hello World!
RADICAL Cybertools: scalable Python abstractions for workflows

**RADICAL EnTk represents an ensemble application as a set of Pipelines.**

Two (pythonic) collections of objects:

- **Set**: contains objects that have no relative order with each other
- **Sequence/List**: contains objects that have a linear order, i.e. object ‘i’ depends on object ‘i-1’

- Pipelines can thus represent general DAG structures
- Pipelines can coordinate and communicate
Swift/T: Enabling high-performance scripted workflows

Write site-independent scripts, translates to MPI

Automatic task parallelization and data movement

Invoke native code, script fragments in Python and R

Rapidly subdivide large partitions for MPI jobs in multiple ways (MPI 3.0)

$ spack install stc

$ conda install -c lightsources2-tag swift-t
We are working closely with ECP Applications to impact deliverables

- **Approach**: Continuous engagement with ECP applications to address their workflow challenges and implement best practices, scalable, and performant workflows using the ExaWorks SDK.

- **ExaAM’s** complex workflow simulates laser melt-pool additive manufacturing processes.

- **Colmena (ExaLearn)**: open-source Python framework for ML-steering of simulation campaigns at scale.

- **CANDLE**: Relies on Swift/T for rapid development, scalability, and portability of DL-oriented cancer application suite on DOE systems.

- **COVID**: National Virtual Biotechnology Lab used billions of core hours harnessed rapidly and effectively for heterogeneous workflows.

- **Gordon Bell Prizes**: 3 of the 4 finalists used ExaWorks technologies

ExaWorks technologies were leveraged in 3 of 4 finalists and the Winner of the SC21 Gordon Bell Covid-19 Competition

The Winner:

DeepDiveMD -- an extension of RADICAL tools -- workflow infrastructure adaptively couples ML + NAMD simulation workflow

Effective speedup of 1 order of magnitude sampling efficiency: with DeepDriveMD observed 25% more conformations of the knee bending in only 12% of the time!

RADICAL components of the ExaWorks tool set brought scalability, reliability, and agility to the project
ExaWorks RoadMap

Exascale Workflows | Community
PSI/J was designed through an open community process

- Our survey, interviews, and co-design meetings highlighted need for portability layer for schedulers
- Community desired a light-weight user-space API
- Initial Python implementation is nearing version 1.0 release
  - Support for Slurm, LSF, Cobalt, Flux, RCT, SAGA
  - Working to add next set of schedulers (e.g., PBS)
  - Architected to allow seamless contributions from the community

---

1. Problem definition

2. Community specification

3. Open discussion

4. Community SDK component
ExaWorks is working towards a production quality continuous integration and deployment infrastructure for workflow tools.

We have developed a GitLab CI infrastructure.

We have set up CI at LLNL, ORNL, and ANL for the SDK components.

We are testing PSI/J on an ECP testing cluster.

We have developed a testing server to collect results of tests and enable dashboards and reporting from multiple sites.

We are creating Status Dashboard to view what tests have been run on which systems.
PSI/J: Portable Submission Interface for Jobs

A set of interfaces that allow the specification and management of “jobs”

Support for Slurm, LSF, Cobalt, Flux, PBS

Open document to define a language-independent specification
Community specification
http://exaworks.org/job-api-spec/specification.html
PSI/J Python binding provides an intuitive Python-futures based API for job management

- **PSI/J Python binding**
  - Python library with asynchronous interface for interacting with job schedulers
  - Support for Slurm, LSF, Cobalt, Flux, RCT, SAGA
  - Working to add next set of schedulers (e.g., PBS)
  - Architected to allow seamless contributions from the community

- Eventually the PSI/J specification will cover more advanced job-management functionality, such as job submission on remote clusters (“layer 1”).
  - All effort so far has been on “Layer 0”, in which PSI/J talks only to the local resource manager.

- We have integrated PSI/J into both RADICAL CyberTools and Parsl
Learn more…

https://exaworks.org
- Join our Slack Channel
- Read the documentation

Tutorial Sessions
- ISC-HPC (May 2022)
- PEARC (July 2022)

Engagements
- Get in touch to discuss how ExaWorks components can benefit your project
Thank you!

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under contract DE-AC52-07NA27344. Lawrence Livermore National Security, LLC