# Open OnDemand
# Scientific Community Advocate Group

## Quarterly Meeting @ PEARC19

Moderators: Robert Settlage, Alan Chalker, Steve Gallo

31 July 2019

**OPEN** ⊃**OnDemand**

# Todays Goals and Objectives

1. Introductions

2. Expectations of this Group: the Community Advocates

3. Visibility and XDMoD integration

4. Accessibility and Job Composer – preliminary thoughts

    – Enhancements
    – Pipeline support
    – git

5. Scalability and Cloud extensions – long running jobs

6. Authentication

7. Ease of installation – quick summary, more details in USER meeting

**OPEN** **OnDemand**

# Introductions

Please state home institution and one positive use for OOD locally.

- John-Paul Robinson
- Ray Leto
- Ping Luo
- Jens Mueller
- Martin Cuma
- Jay Jay Billings
- Mark Gardner
- Michael Knox

- Dominic Daninger
- Blake Joyce
- Richard O Anderson
- Lonnie Davis Crosby
- Kristina Plazonic
- Michael Zentner
- Nigel Hartell
- Shawn Doughty

**OPEN OnDemand**

# Expectations of this Group

**You are the Community Advocate.**

Your local community but with an ear to the larger community.

Open OnDemand is a tool for the community. This group was formed to make sure we keep *a) relevant, b) current, and c) focused* on community needs. We need your local user

- input,
- comments, and
- ideas

to be successful.

**OPEN** ᴐnDemand

## Grant Objectives
*Development Activities*

- Visibility (e.g. OOD - XDMoD integration)

- Accessibility (e.g. interface improvements)

- Scalability (e.g. platform and application expansion)

- Engagement (e.g. expand the community)

# Visibility
*XDMoD Integration*

**Job accounting and performance summaries.**

Detailed performance data for completed user jobs.

- Project Summary

- 2019 Progress and Plans

- Accounting Dashboard Examples and Demo

- Feedback and Discussion

# Visibility
*XDMoD Integration*

## 5-Year Plan Summary.

- Year 1: Incorporate XDMoD job accounting and performance summary

- Year 2: Directly link to XDMoD Job Viewer from OnDemand Jobs App

- Year 3: Provide a User Report Card

- Year 4: Allow users to create custom utilization reports

- Year 5: Automated detection of application specific performance via machine learning

# Visibility
*XDMoD Integration - Progress and Plans*

**2019:**

Progress

- Summary dashboard prototype
- User customization of dashboard

Plans

- Role-based, user customizable dashboards
- Report card: User, Project/PI, Center
- Authentication: SSO for OOD and XDMoD

OPEN **OnDemand**

# Visibility
*XDMoD Integration - SCAG Requested Features*

XDMoD Integration

- Historical usage/utilization data
- Queue wait time (drill-down to job size)
- Workgroup usage summary

XDMoD Integration with Job Viewer (SUPReMM)

- Detailed post-job reports
- Memory, Network, GPU Metrics, CPU usage on a per node basis

Planned

- Flag poorly performing jobs
- Allocation support

# Visibility

*XDMoD Integration - User Dashboard*

Wait Time by Queue, Project CPU Usage, Cloud VMs
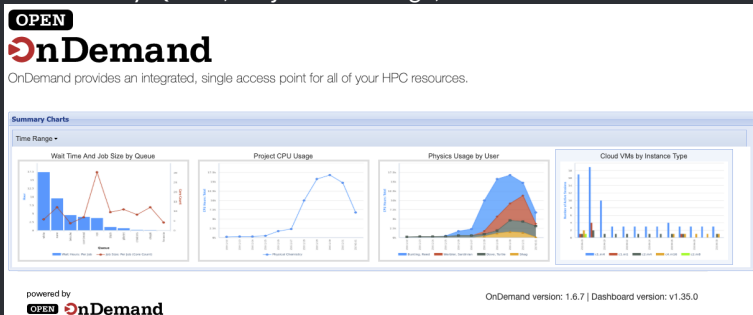
# Visibility
*XDMoD Integration - User Dashboard*

Click a chart to view detail or open directly in XDMoD

# Visibility

## *XDMoD Integration - PI Dashboard*

### Currently under development
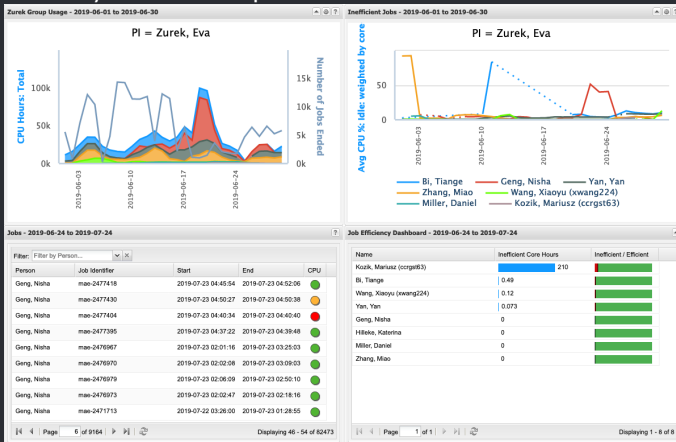
# Visibility
## *XDMoD Integration - Job Viewer*

- Detailed post-job reports
- Memory, Network, GPU Metrics, CPU usage on a per node basis
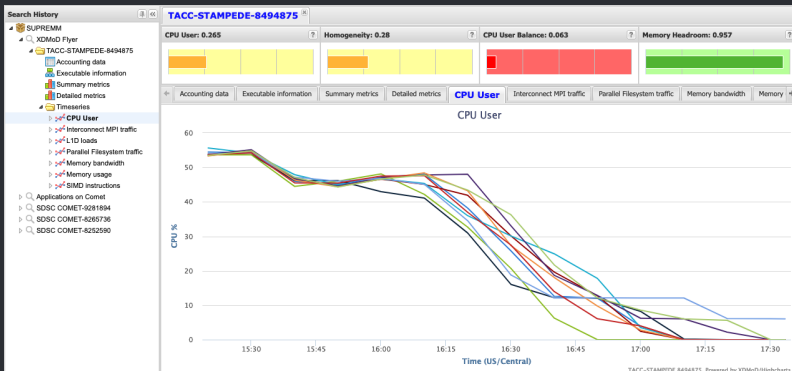
# Visibility
*XDMoD Integration - Demo and Discussion*

- Live integration demo

- Discussion and feedback

# Accessibility
*Interface improvements*

## 5-Year Plan Summary.

- Streamlined interactive app launching (Y1)

- IDEs accessible from within OOD (Y1)

- Simple Desktop home page (Y1/2)

- Domain specific workspaces (Y1/3)

- Job composer updates (Y2)

- Visual editor for job dependencies (Y3)

- Changes / applications as identified by the SCAG (Y3)

- Common API

## Accessibility
*SCAG and user feedback – summarized by Dave:*

Need to further enable novice users.

Need to provide more functionality to enhance the power user.

1. Enhancements
    – Unified job viewer - running, completed, interactive and batch
    – Integrated app plugin - batch and interactive i.e. Job Composer templates + web forms
    – Files + jobs - file type indicates app
    – Software catalog - every installed module
    – Open file in app
    – remove restriction on where job folders live

2. Pipeline support

3. git

**OPEN** ⊃n**Demand**

# Accessibility

*Unified job viewer*

running  completed, interactive and batch

# Accessibility
*Integrated app plugin*

batch and interactive i.e. Job Composer templates + web forms

# Accessibility

*Files + jobs*

file type determines the app
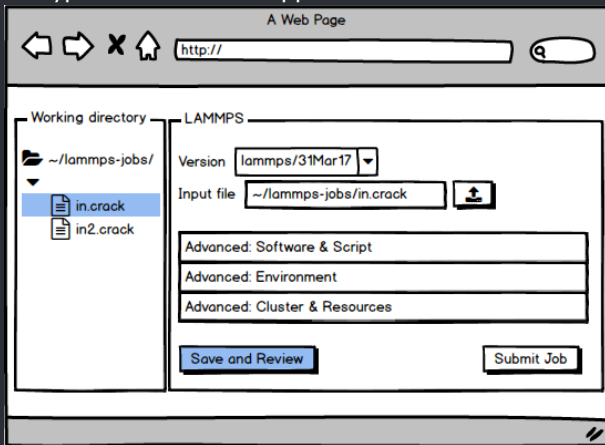
# Accessibility
*Software catalog*

file type determines the app

# Accessibility
*Open file in app*
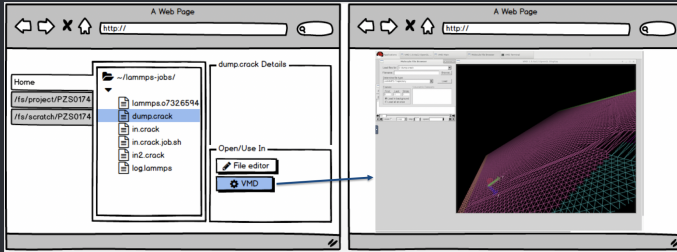
# Scalability

- API limitations (e.g. persistence mechanisms) (Y1/2)

- Backend server communication (e.g. containers and secure reverse proxies) (Y3)

- Process management (e.g. background job processing) (Y4)

- Server load balancing (e.g. PUNs across multiple nodes) (Y5)

# Scalability

Cloud extensions; long running or interactive work without a scheduler.

- Kubernetes

- ssh + fork - launch on login node

- OpenStack, AWS, GoogleCloud

  1. Provision VMs, do interactive work without walltime restriction
  2. Utilize "cloud burst" features of Slurm, other schedulers

- Challenges include data transfer, ensuring VM processes are per user and protecting user started root processes abuse

**OPEN** ◗n Demand

# Scalability

Looking for feedback.



- Default to CILogon or other method besides Apache Basic auth?

- Provide more helpful examples to setup with campus auth?

## Scalability
*Ease of installation: The first barrier we impose is on ourselves.*

Community feedback:

Installation is sometimes difficult.

- Interactive Apps are difficult - simplify plugin architecture and provide better debugging experience

- Authentication is hard. See above.

- Getting the Apache config right is hard; simplify infrastructure to avoid regenerating config using ood-portal-generator

- Cluster config is hard; provide auto validation/error reporting

- Configuration is spread across many files. Centralize into a few files.

**OPEN** ⊃n Demand

## Scalability
*Ease of installation: The first barrier we impose is on ourselves.*

Community feedback:

Installation is sometimes difficult.

- Interactive Apps are difficult - simplify plugin architecture and provide better debugging experience

- Authentication is hard. See above.

- Getting the Apache config right is hard; simplify infrastructure to avoid regenerating config using ood-portal-generator

- Cluster config is hard; provide auto validation/error reporting

- Configuration is spread across many files. Centralize into a few files.

**OPEN** ⊃nDemand

# Thank you

Homework.

Work with your local community to:

- discover under served users

- determine how we can extend the interface to better support new users/use cases

- other thoughts/comments/ideas …