# Measuring & Modeling HPC User Productivity: Whole-Experiment Turnaround Time

SE-HPC 2007 Workshop Presentation

Michael O. McCracken, Nicole Wolter, Allan Snavely
UCSD CSE & SDSC

Thanks also to:
Lorin Hochstein (UNL), Taiga Nakamura (UMD), Victor Basili (UMD)

Work supported by the DARPA HPCS program

# Background

- Productivity while using existing code
- HPC users face an **optimization problem**
- Objective: lower experiment turnaround time
    - Experiment: Everything from moving data to analysis and archival
- Many ways to affect total time to solution
    - Change code, system, job shape
    - Alter problem or IO needs
    - Attempt to influence center policy
- Users have a hard time predicting effects

## Talk Outline

- Questions users ask when planning experiments
- The main question: **How can we help them plan better?**
  - Expert and Non-Expert users
- An illustrative story
- Questions along the way
- Playing “what-if”?

# Questions Users Ask When Planning

- **How long** will this experiment take?
- When do I have to start running to **use all my allocation**?
- Would I be done faster if I ran it on a **different system**?
- Is it **worth my time** to optimize code performance?
- Will I be done faster if I ran it on more processors?
- How much time am I **losing in the queue**?
- **Should I pay** for express queue priority?

How can we help users plan experiments?

Show them their **bottlenecks** and quantify the **effects of their options**.

# An illustrative story

- An HPC user (call him Bob) has a large simulation
  - Uses an entire Terascale system
  - ~20 48-hour jobs
  - several Terabytes of output per job
- Bob believes that a single dedicated run slot would be most productive.
  - Queue hurts larger jobs more
  - System reliability at scale
- Quantitative impact of queue wait on different systems

# How can we help?

- How do we find out what Bob's specific bottlenecks are?
- How do we get Bob to tell us what he's doing?
- Can we develop a model of Bob's whole experiment?
- Can we use simulation to predict the effects of a change:
  - In code performance, system choice, or policy effects?
- *Can we simulate multiple workflows to guide system choice and site policy?*

## How do we find out what Bob's specific bottlenecks are?

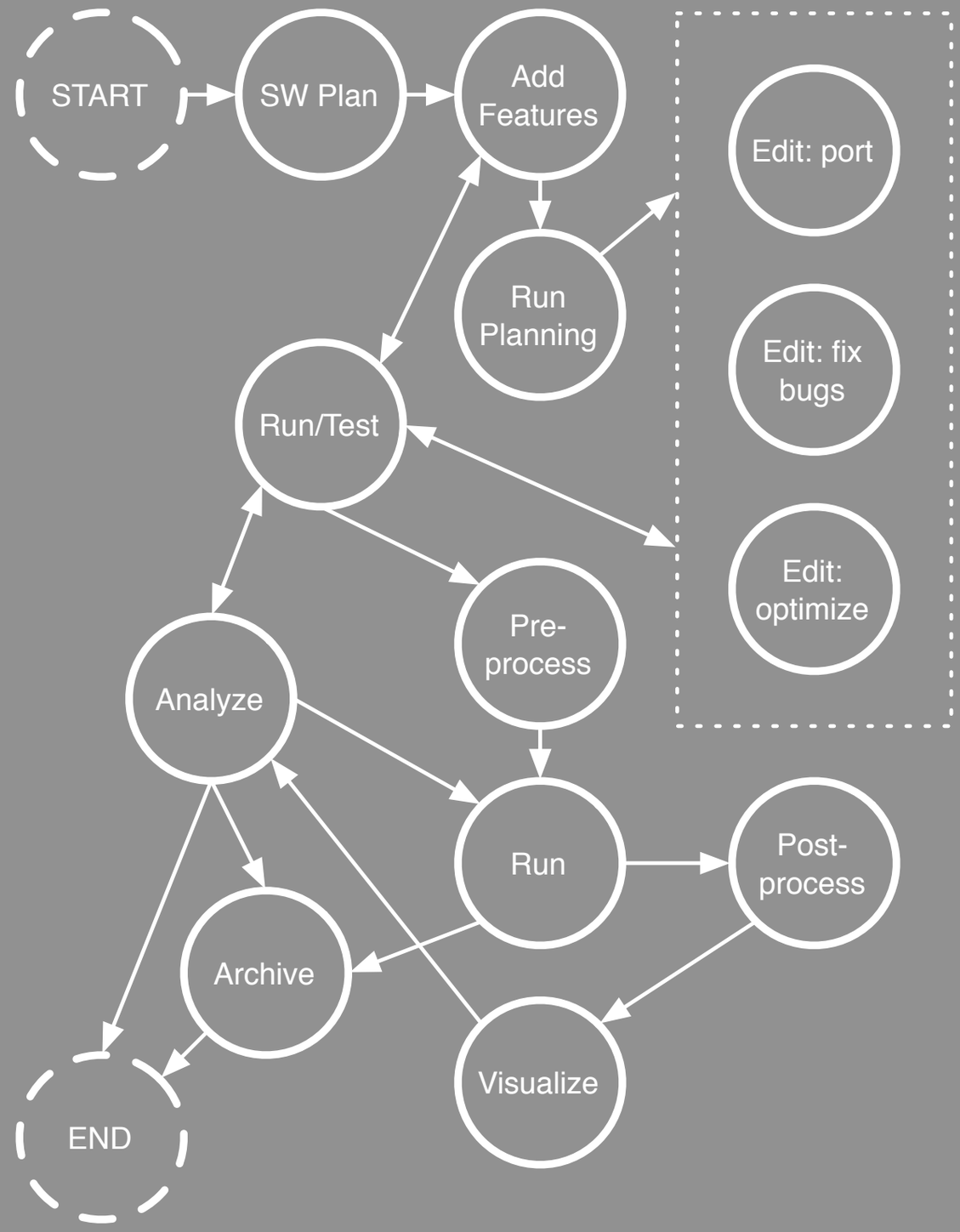
- We need to get a description of his workflow
  - How many jobs he's running
  - How much data they produce
  - What he does with the data after it's produced
- We need to know what systems he could be using, and performance characteristics of those systems
- It's important not to leave out steps that are potential bottlenecks
  - Application characterizations can ignore non-computational bottlenecks
- We need to know **everything he does to complete his experiment**

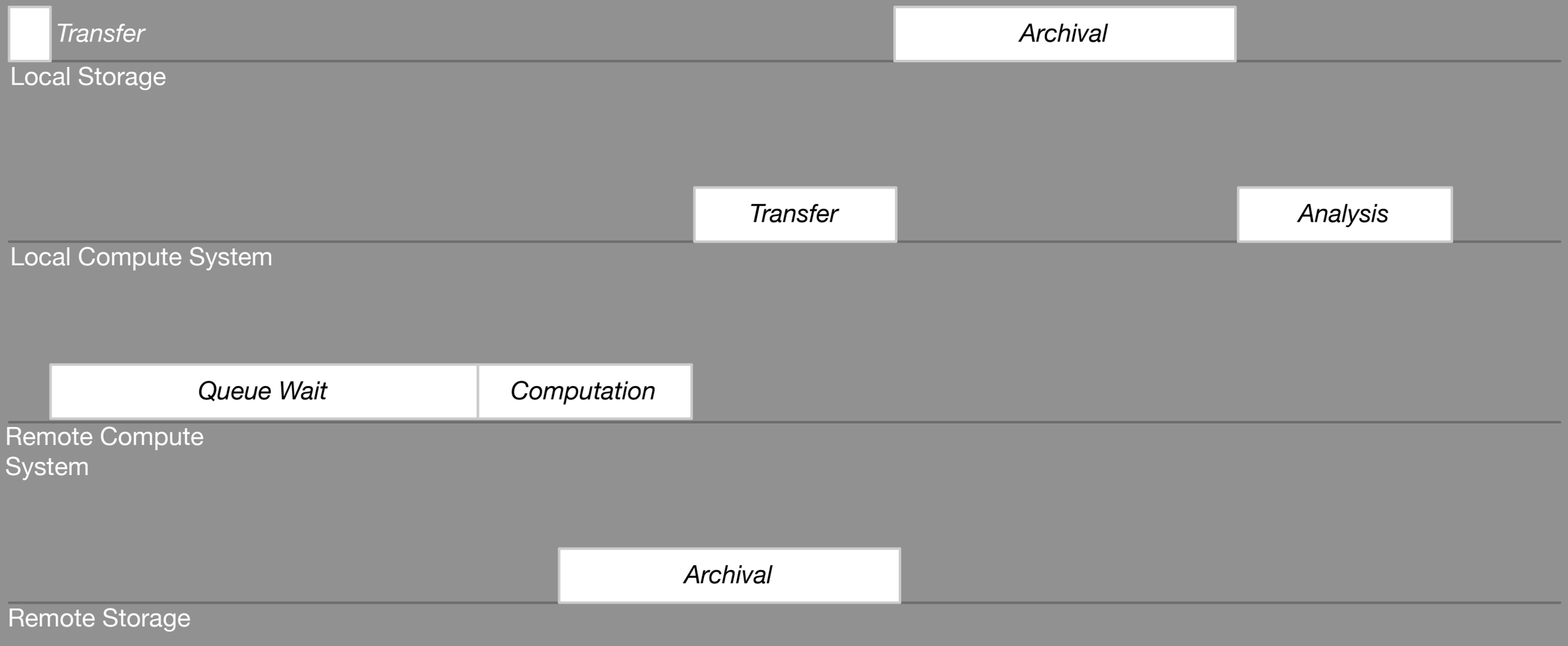
# How can we get him to tell us everything he's doing?

- We could **observe him**
  - Difficult to do in person, long-term
  - Difficult to capture everything automatically
- We could **ask him**
  - Enough detail?
- We use a **conceptual model** to guide the interview
  - Represents a variety of user workflows
  - Experiment tasks at an appropriate level of detail for discussion

# Eliciting user workflow

- We use the model on the right
  - How much time in each state?
  - What path does he take through the states?
  - What different systems does he use in each state?
- Have him draw his own path through the states
- Helps to make sure we don’t ignore any steps
- Helps generate more detailed model for simulation






# Bob’s workflow, for one job

How can we get him to tell us what he’s doing?

## Can we develop a model of his entire experiment?

- His experiment fits into a common pattern
    - Initial data transfer, then a sequence of compute jobs with some analysis/post-processing and archival at each step.
- A common set of parameters:

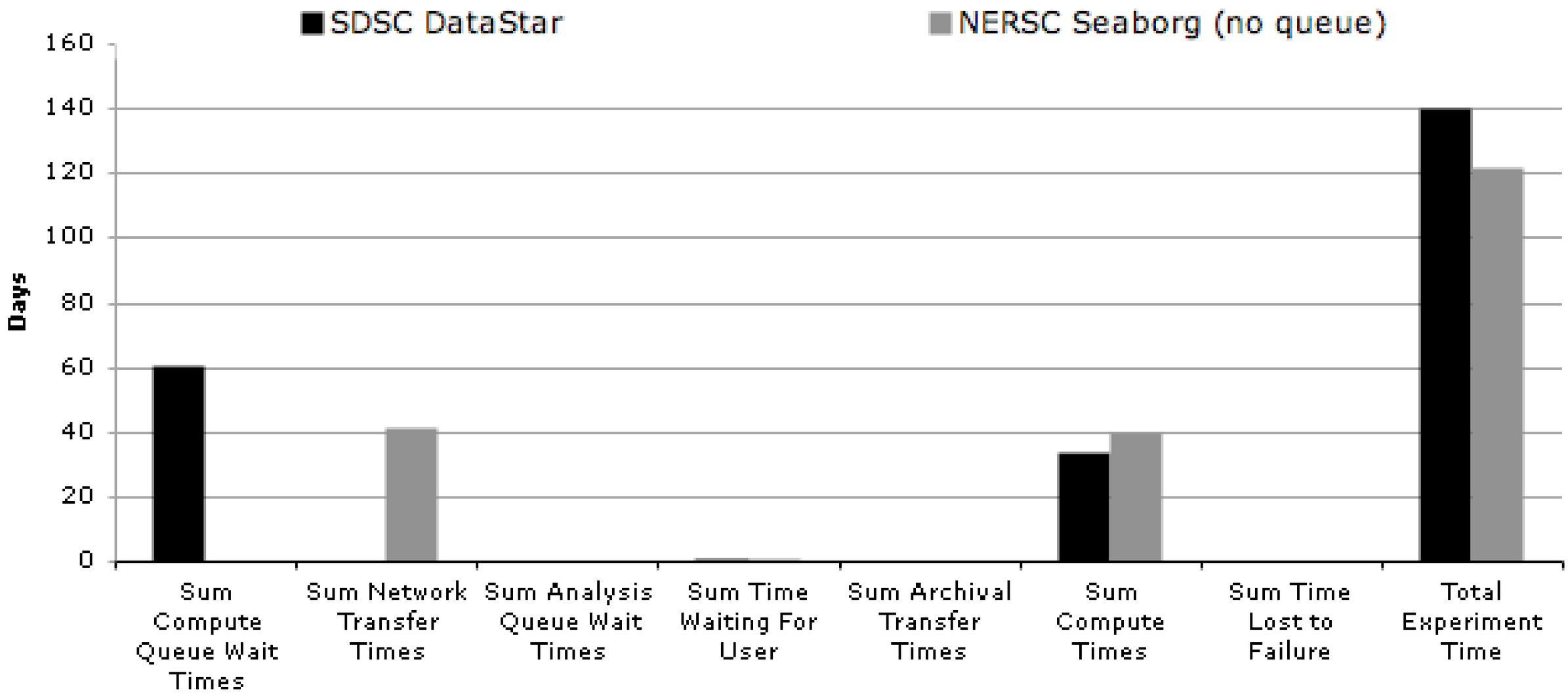
| Experiment Parameters |
| --- |
| Input Data Size |
| Number of Jobs |
| Compute time per job |
| Output size per job |
| Analysis time per job |
| Compute/Archive Systems Used |

| System Parameters |
| --- |
| Network Bandwidth |
| Queue wait time* |
| Failure rates* |

# Can we use simulation to predict the effects of a change?

- A discrete event simulator
- Verifying Bob's workflow:
  - His time prediction with current queue wait times: "greater than six months"
  - Our simulator predicted bounds of 27-29 weeks, or 6-7 months
- Does Bob need this?
- Does anyone else need this?

# What-ifs: dedicated run vs faster system with no network



# How have we helped?

- Generated a conceptual model of HPC workflow
- Used the model to get quantitative characterizations of Bob's workflow
- Used characterization to parameterize a simulation of Bob's experiment
- Accurately simulated whole-experiment turnaround time for multiple situations
    - Giving him a way of saying exactly how much time could be saved by dedicated runs
- What else can we use this information for?
    - Simulate entire workloads to evaluate total impact of policy decisions
    - Evaluate experiment throughput of current systems
    - Potentially map experiments to best groups of systems