Bridges-2 Demo

CyberAccelerate ACCESS Workshop

Emery Etter

emery@psu.edu

RISE Engineer

Institute for Computational and Data Sciences

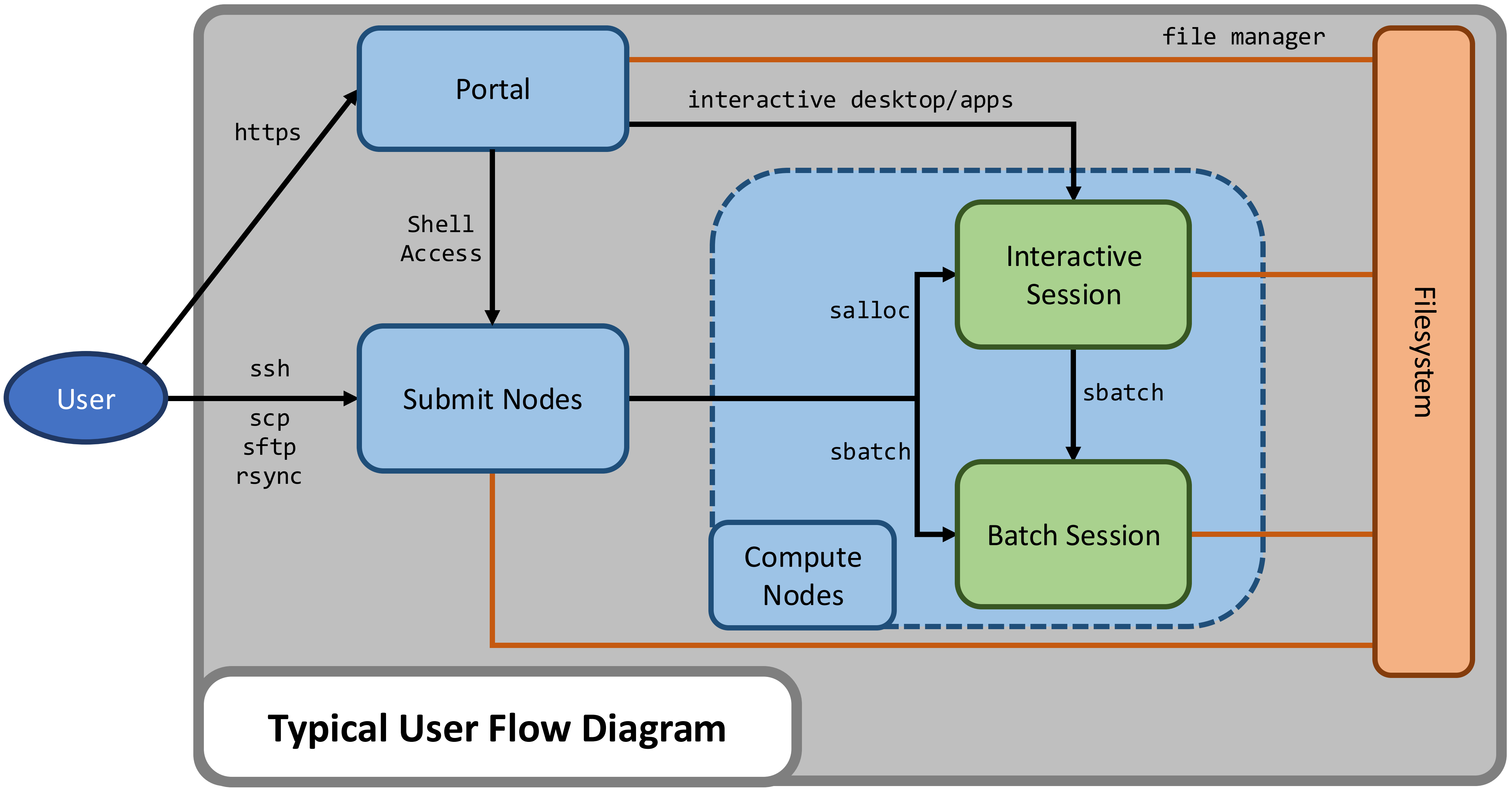
Bridges-2 Computing Cluster

With an ACCESS account through NSF's ACCESS program, users can access various advanced computing resources.

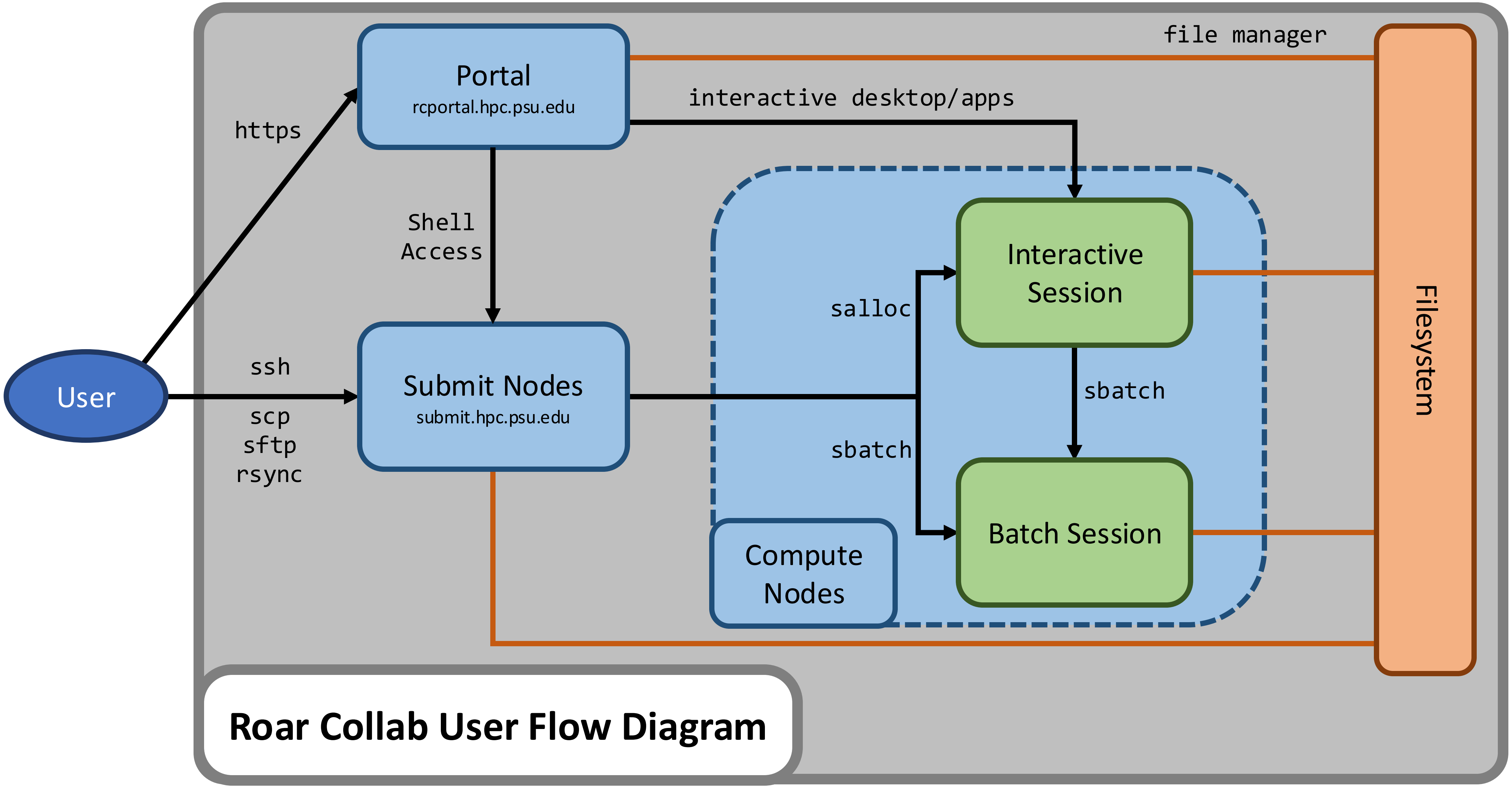
Bridges-2, the flagship cluster offered by Pittsburgh Supercomputing Center (PSC), is one of these available resources.

Bridges-2 Information: www.psc.edu/resources/bridges-2
Bridges-2 User Guide: www.psc.edu/resources/bridges-2/user-guide

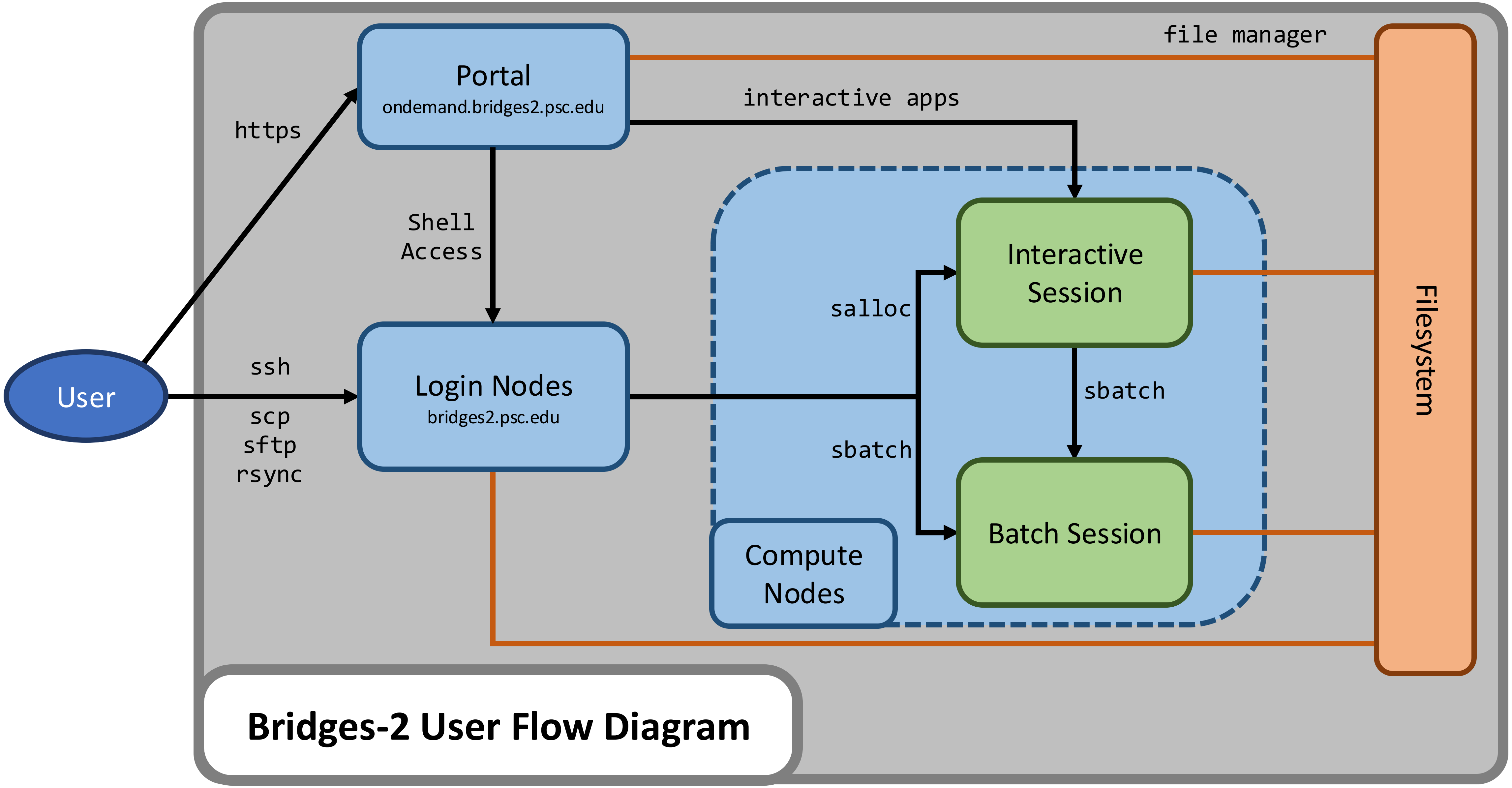
Today's demo utilizes Bridges-2, but this is just one of the many compute resources available via the ACCESS program.



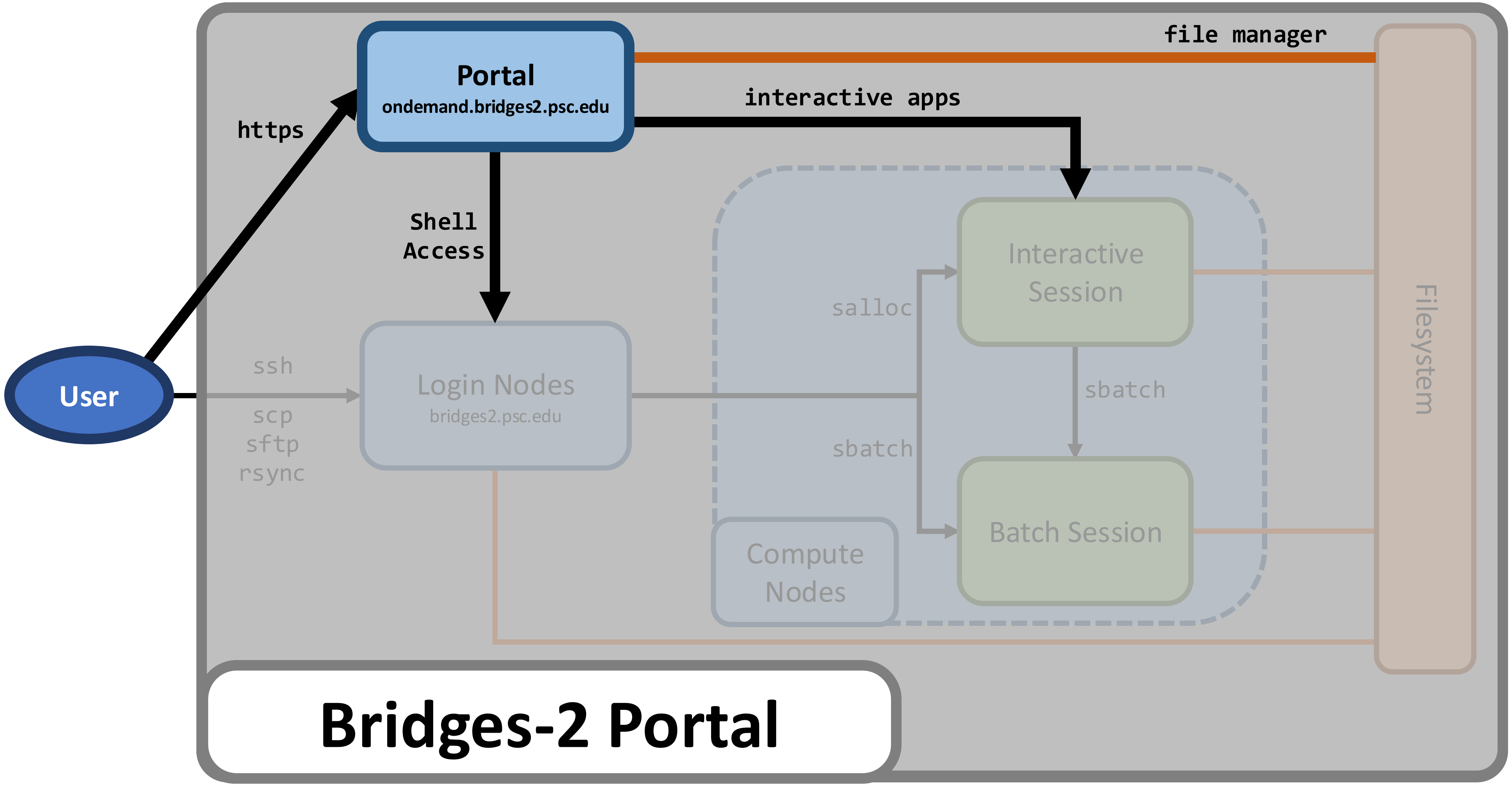
Typical User Flow Diagram



Roar Collab User Flow Diagram



Bridges-2 User Flow Diagram



Bridges-2 Portal

Bridges-2 Portal

The Bridges-2 Portal, powered by Open OnDemand (OOD), allows users to submit and monitor jobs, manage files, and run applications using just a web browser.

To access the Bridges-2 Portal, navigate to the following website in a web browser: ondemand.bridges2.psc.edu

To log into the Bridges-2 Portal successfully, users must log in using valid PSC account credentials.

Bridges-2 Portal Menu Bar

Apps: Lists all available Portal apps

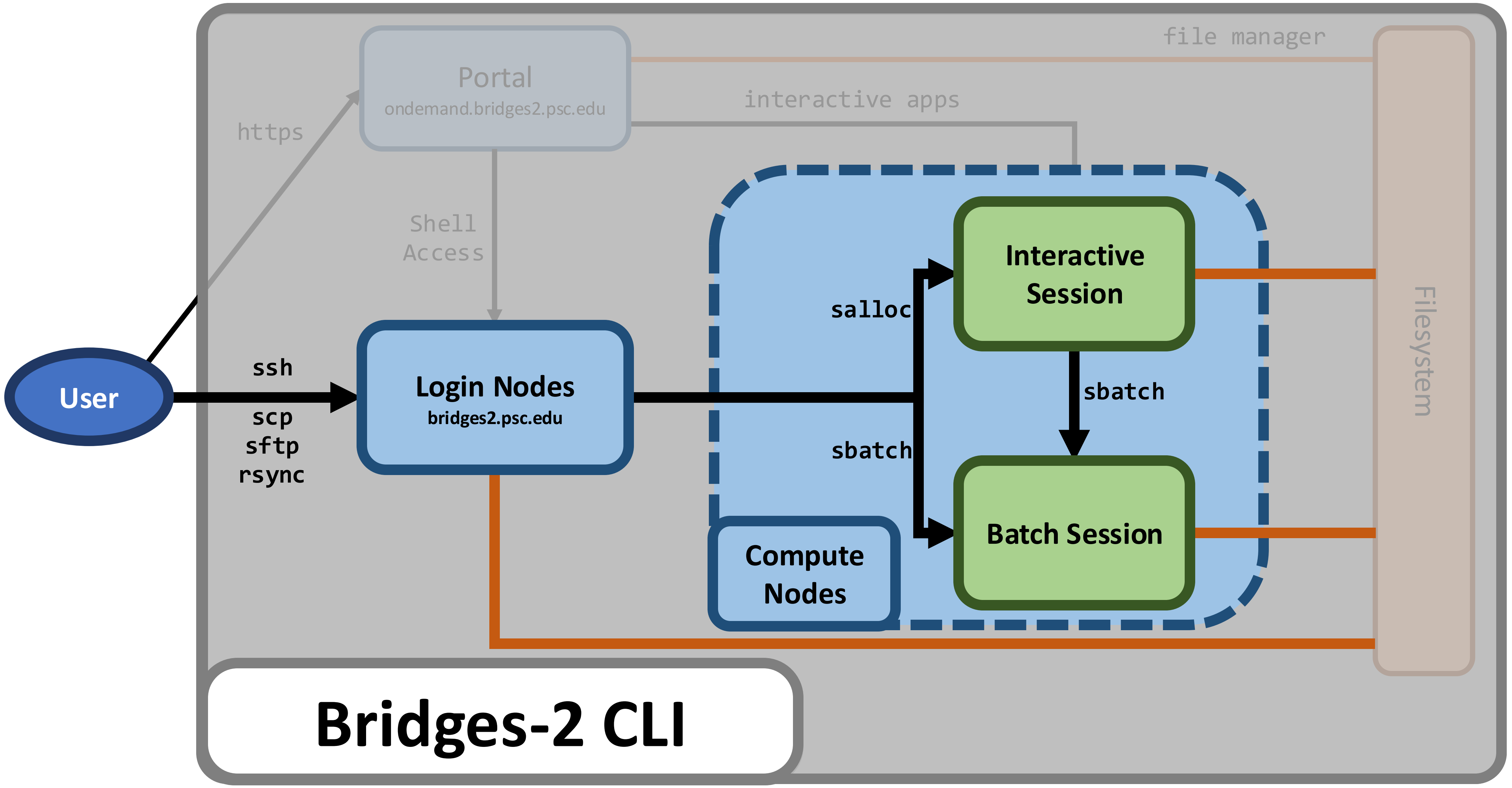
Files: Provides a convenient graphical file manager and lists primary file locations

Jobs: Lists active jobs and allows use of the Job Composer

Clusters: Provides shell access to submit nodes on RC

Interactive Apps: Provides access to Jupyter and RStudio interactive apps

My Interactive Sessions: Lists any active sessions



Bridges-2 CLI

Key Notation

Green highlight identifies a `command` that can be run in a terminal session.

Light `$BLUE` highlight with a dollar sign identifies a user environment variable.

Grey highlight identifies `something` within a file.

Angle brackets like **<this>** represent the need to insert a string. Replace **<this>** with the string, and do not keep the angle brackets.

Computing Environment Basics

- Data Storage and Data Transfer
- Accessing and Utilizing Software
- Performing Computational Tasks

Data Storage and Data Transfer

Bridges-2 Data Storage

Users can access personal, project-specific, and job-specific storage locations.

Storage	Quota	Backup Policy	Use Case
$HOME	25 GB	Daily Snapshot	Batch scripts, source code, and parameter files
$PROJECT	Project-dependent	No Backup	Primary project storage
$LOCAL	Job-dependent	No Backup	Job-specific node-local storage
$RAMDISK	Job-dependent	No Backup	Job-specific IO memory

The `my_quotas` and `projects` commands provide further details on the available storage locations and the usage of those storage locations

Bridges-2 Data Transfer

File transfer utilities like `scp`, `sftp`, and `rsync` can be used with the `bridges2.psc.edu` hostname to transfer smaller files to/from Bridges-2.

Globus, a web-based file transfer tool, should be used for larger file transfers. With Globus, users can easily, reliably, and securely transfer data to/from Bridges-2.

Details on using Globus for file transfers to/from Bridges-2 are provided in the Bridges-2 User Guide.

Accessing and Utilizing Software

Software Stack

Software stacks often use Lmod to package available software.

Lmod is a useful tool for managing user software environments using **environment modules** that can be dynamically added or removed using module files.

Lmod alters environment variables, most notably the $PATH variable.

Lmod Commands

Command	Action
`module avail`	Lists all modules that are available to be loaded
`module show <module_name>`	Shows the contents of a module
`module spider <module_name>`	Searches the module space for a match
`module load <module_name>`	Loads a module or multiple modules if given a space-delimited list of modules
`module load <module>/<version>`	Loads a module of a specific version
`module unload <module_name>`	Unloads a module or multiple modules if given a space-delimited list of modules
`module list`	Lists all currently loaded modules
`module purge`	Unloads all currently loaded modules
`module use <path>`	Adds an additional path to $MODULEPATH to expand module scope
`module unuse <path>`	Removes a path from $MODULEPATH to restrict module scope

Performing Computational Tasks

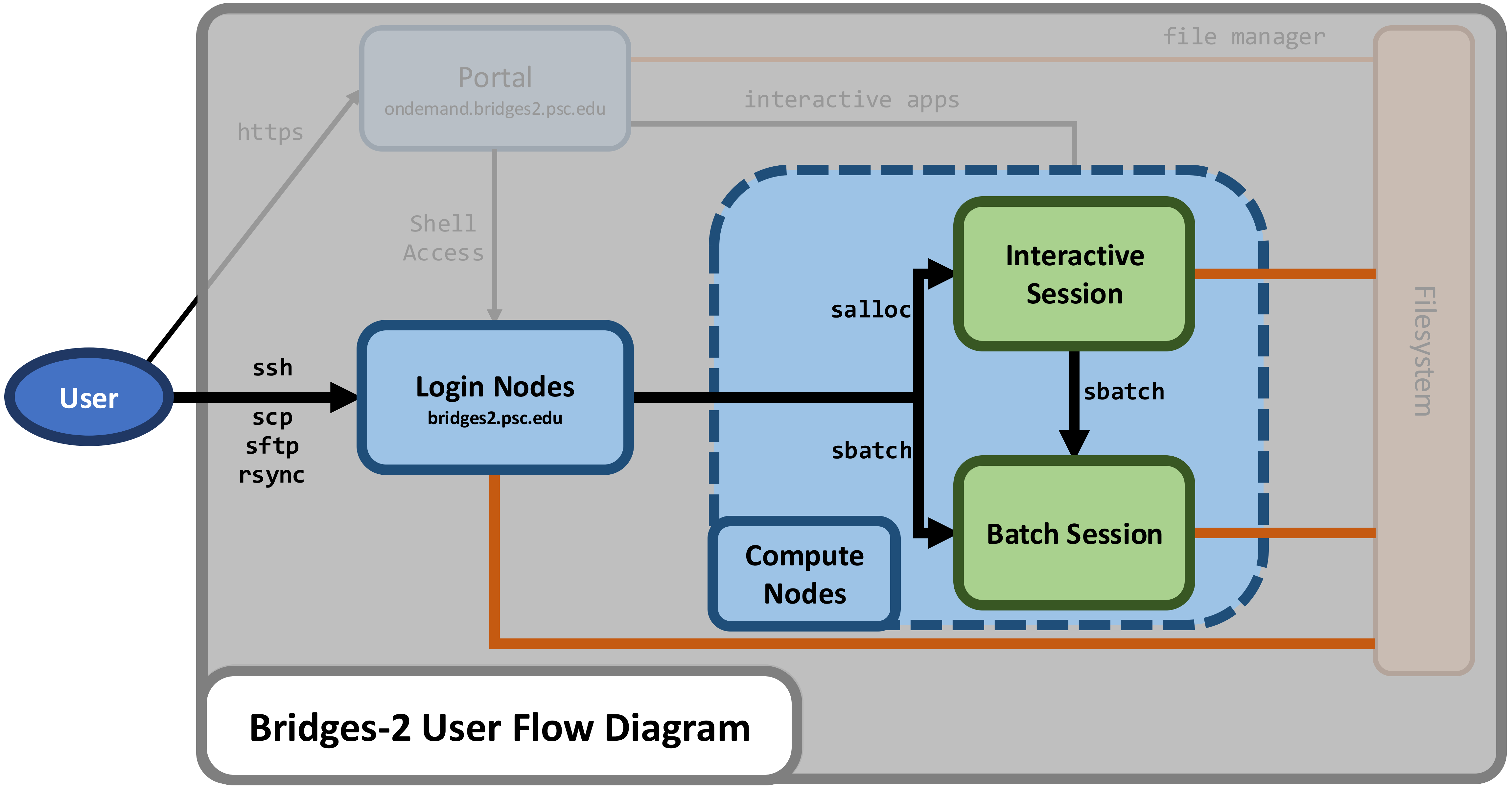
Shared Computing Systems

The clusters are shared computational resources. To perform computationally-intensive tasks, users must request compute resources and be provided access to those resources to perform the tasks.

The request/provision process allows the tasks of many users to be scheduled and carried out efficiently to avoid resource contention.

Slurm (**S**imple **L**inux **U**tility for **R**esource **M**anagement) is often utilized as the job scheduler and resource manager for shared computing clusters.

Slurm is rapidly rising in popularity and research computing cluster use Slurm. ICDS clusters (both RC and RR) and Bridges-2 all utilize Slurm.



Bridges-2 User Flow Diagram

Slurm Job Submission

Requesting an interactive compute session can be accomplished using the `salloc` command.

Submitting a batch job with a submission script can be accomplished using the `sbatch` command.

Detailed information for each of these commands (salloc, sbatch) can be found in Slurm's documentation.

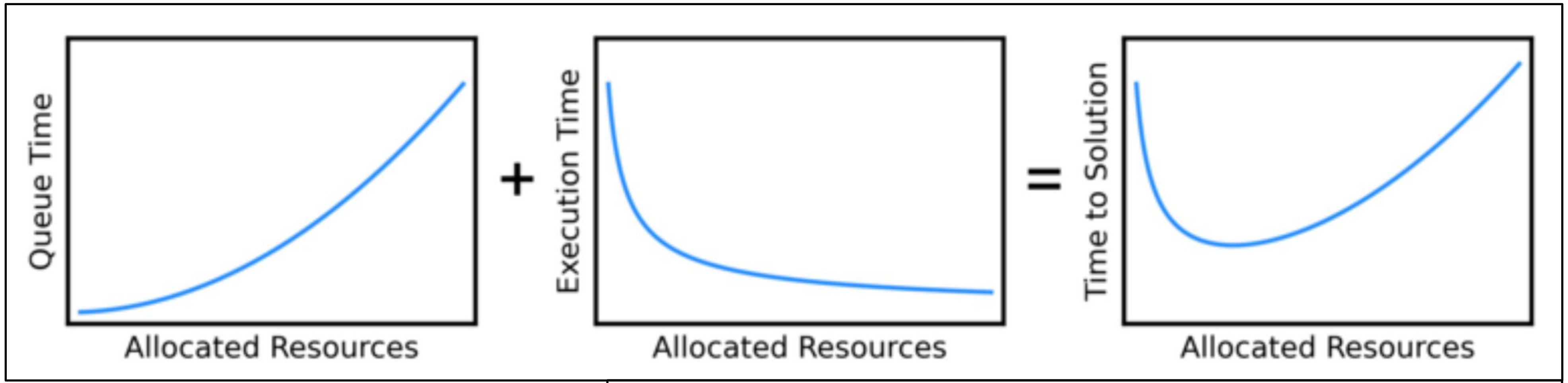
Common Resource Directives

Directive	Description
-J or --job-name	Specify a name for the job
-A or --account	Charge resources used by job to specified compute account
-p or --partition	Request a specific partition
-N or --nodes	Request the number of nodes for the job
-n, --ntasks, or --ntasks-per-node	Request the number of tasks for the job
-C or --constraint	Specify any required node features for job
--mem or --mem-per-cpu	Specify the memory required per node or CPU, respectively
-t or --time	Set a limit on the total run time of the job
--requeue	Specify that the batch job should be eligible for requeuing
-e or --error	Instruct Slurm to connect the batch script's standard error to a non-default file
-o or --output	Instruct Slurm to connect the batch script's standard output to a non-default file

Time To Solution

Requesting more resources for a job will increase its time in the queue as it must wait longer for resources to be allocated.

Typically, parallelized code will see a reduction in overall runtime as more resources are requested.



https://researchcomputing.princeton.edu/support/knowledge-base/scaling-analysis

The total time to solution is ultimately reduced when the job strikes a balance by requesting the minimal amount of resources needed to provide a reasonable speedup.

Requesting Resources

The goal of a resource request is to attain the necessary computational resources to complete a task while minimizing the time to solution.

An optimal resource request is the minimal amount of computational resources that allows a computational task to run to successful completion.

Let's examine **1 laptop-worth of resources** as a reference. My laptop, for example, has a 12-core processor and 36 GB of RAM.

If a computational task can run on a laptop without crashing the device, then there is absolutely no need to submit a resource request larger than this.

Bridges-2 Demo

Access Login Node

HANDS ON

Choose one of two options to initiate an interactive session on a submit node:

- From a terminal session, run the following command:

```
ssh <userid>@bridges2.psc.edu
```

- Access ondemand.bridges2.psc.edu and from the banner select

```
Clusters > Bridges-2 Shell Access
```

After completing the login process, the prompt will indicate that the interactive session is on a login node.

Download Session Files

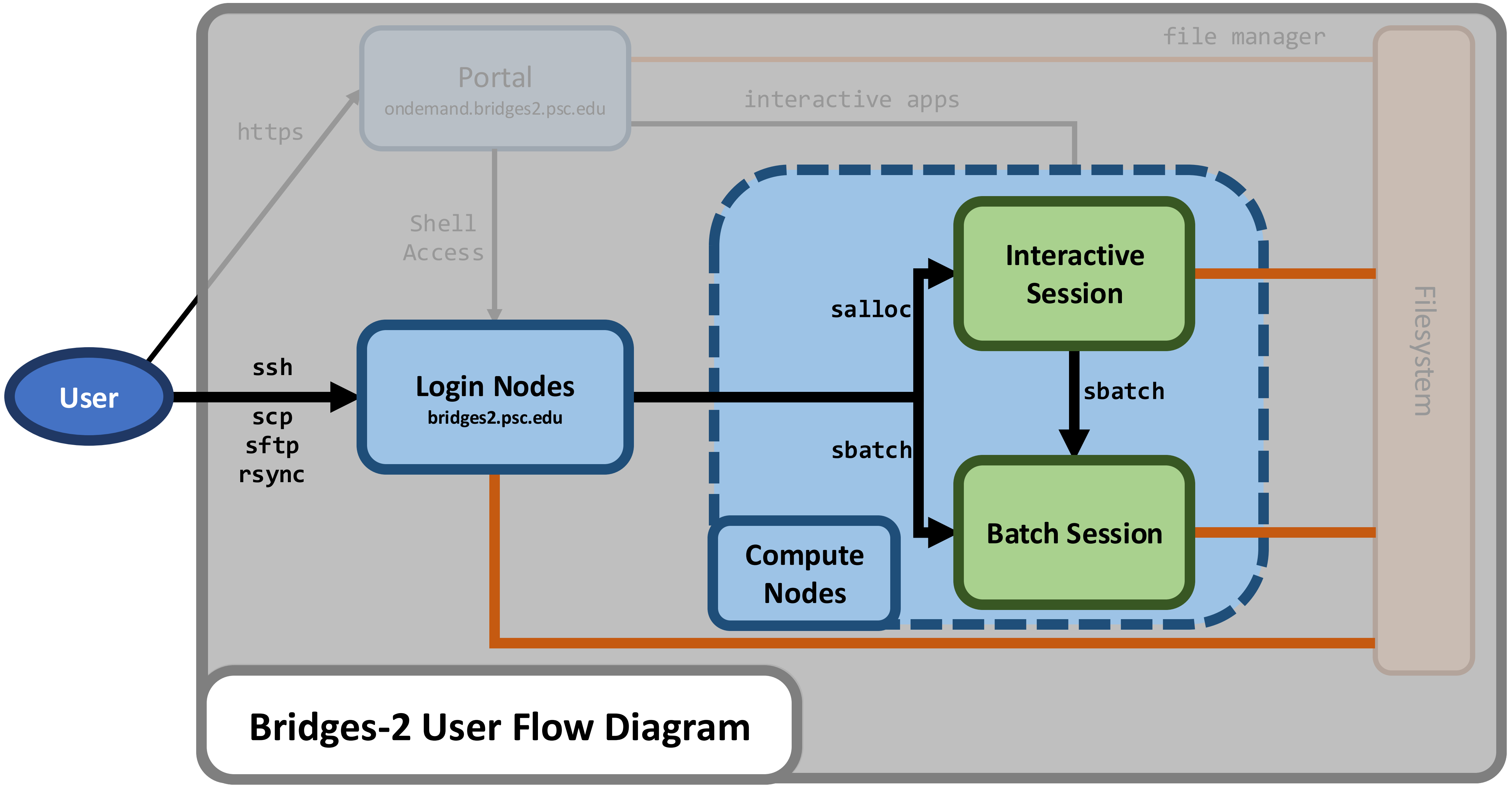
HANDS ON

Within the session on Bridges-2, navigate to either your $HOME directory with the following command:

```
cd    OR    cd $HOME
```

Download the session files and enter the directory with

```
git clone https://github.com/ecetter/intro2bridges2.git
cd intro2bridges2
```



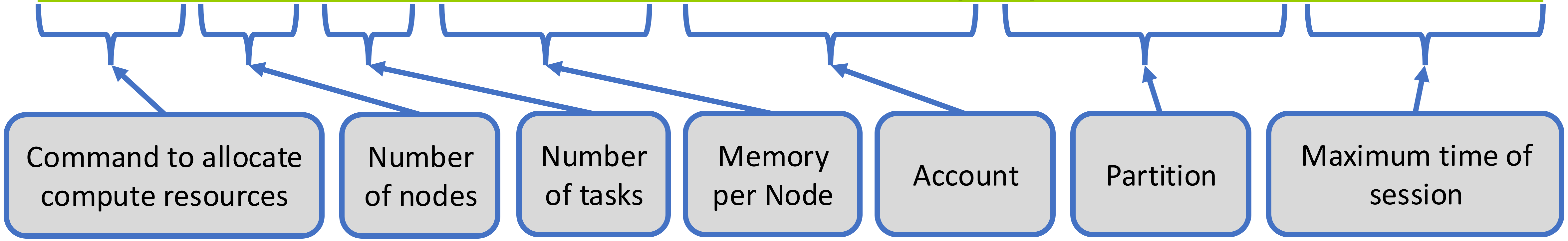
Bridges-2 User Flow Diagram

Interactive Compute Session

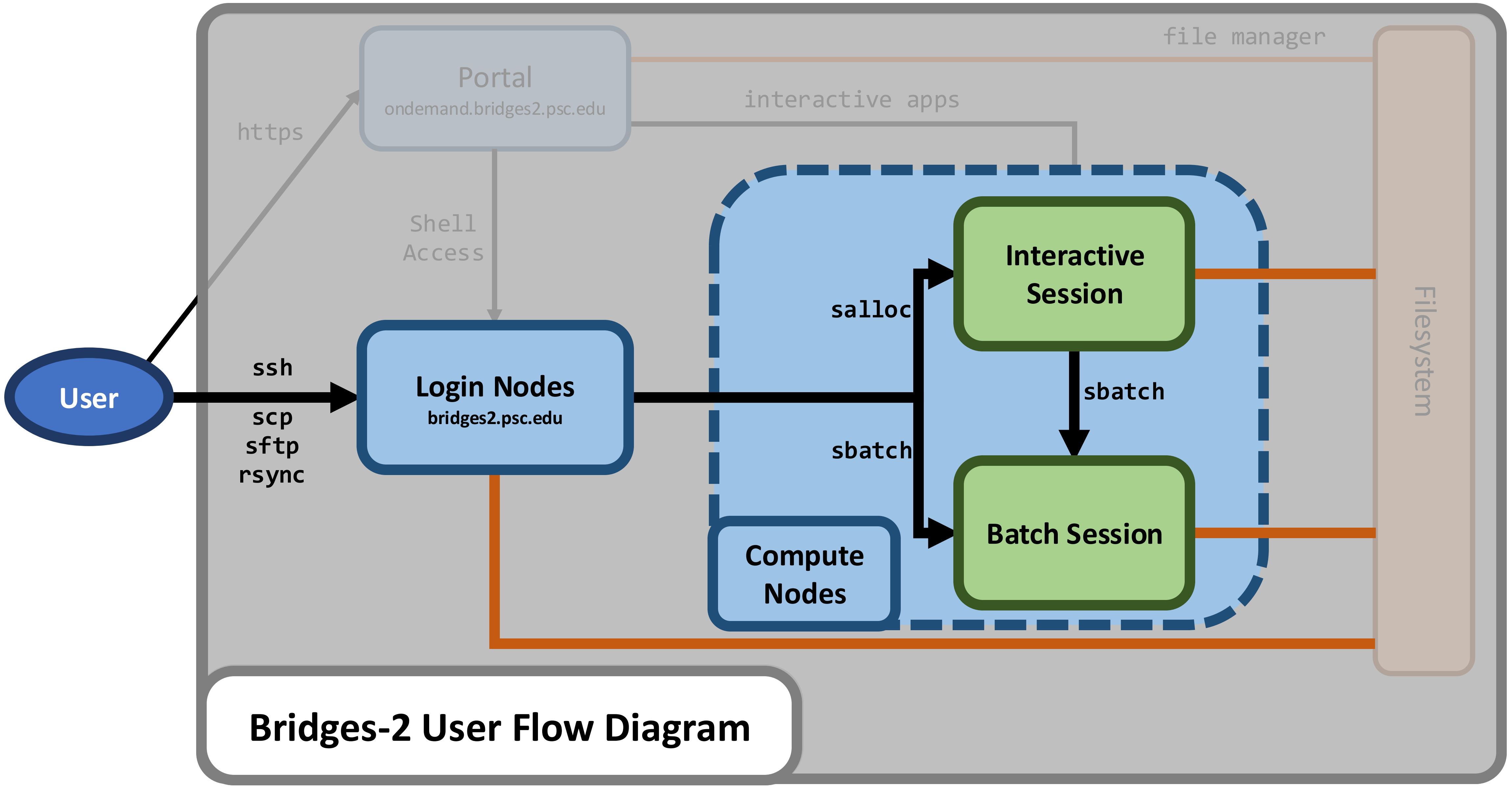
HANDS ON

To launch an interactive session on a compute node, run the following command on a login node:

```
salloc -N 1 -n 4 --mem 4GB -A cis240128p –p RM-shared -t 1:00:00
```



Note that the prompt changes from a login node type to a compute node type after the resources are granted.



Bridges-2 User Flow Diagram

Submission Script Example

```
#!/bin/bash

#SBATCH --job-name=pi_serial
#SBATCH --account=cis240128p
#SBATCH --partition=RM-shared
#SBATCH --nodes=1
#SBATCH --ntasks=1
#SBATCH --mem=1G
#SBATCH --time=00:20:00
#SBATCH --output=out/slurm-%x.out
#SBATCH --error=out/slurm-%x.err

module load python numpy

echo "Job ID is $SLURM_JOB_ID."
echo "Executing on the machine: " $(hostname)

python pi_serial.py > out/pi_serial.out
```

Submission Script Example

```
#!/bin/bash
```

Shebang

```
#SBATCH --job-name=pi_serial
#SBATCH --account=cis240128p
#SBATCH --partition=RM-shared
#SBATCH --nodes=1
#SBATCH --ntasks=1
#SBATCH --mem=1G
#SBATCH --time=00:20:00
#SBATCH --output=out/slurm-%x.out
#SBATCH --error=out/slurm-%x.err
```

Scheduler/ Resource Directives

```
module load python numpy

echo "Job ID is $SLURM_JOB_ID."
echo "Executing on the machine: " $(hostname)

python pi_serial.py > out/pi_serial.out
```

Job Payload

Common Environment Variables

Variable	Description
SLURM_JOB_ID	ID of the job
SLURM_JOB_NAME	Name of job
SLURM_NNODES	Number of nodes
SLURM_NODELIST	List of nodes
SLURM_NTASKS	Total number of tasks
SLURM_NTASKS_PER_NODE	Number of tasks per node
SLURM_QUEUE	Queue (partition)
SLURM_SUBMIT_DIR	Directory of job submission

Filename Pattern Symbols

Both standard output and standard error are directed to the same file by default, and the file name is "slurm-%j.out", where the "%j" is replaced by the job ID.

The output and error filenames are customizable, however.

Symbol	Description
%j	Job ID
%x	Job name
%u	Username
%N	Hostname where the job is running
%A	Job array's master job allocation number
%a	Job array ID (index) number

For example:

```
#SBATCH --output=%x-%j.out
#SBATCH --error=%x-%j.err
```

Submit A Job

HANDS ON

Enter the *pi_example* directory and launch a job:

```
cd pi_example
sbatch submit-pi_serial.slurm
```

The job will be placed in the job queue until it gains enough priority to be assigned compute resources.

Monitor A Job

HANDS ON

Find out what node this job is running on with

```
squeue -u <userid>
```

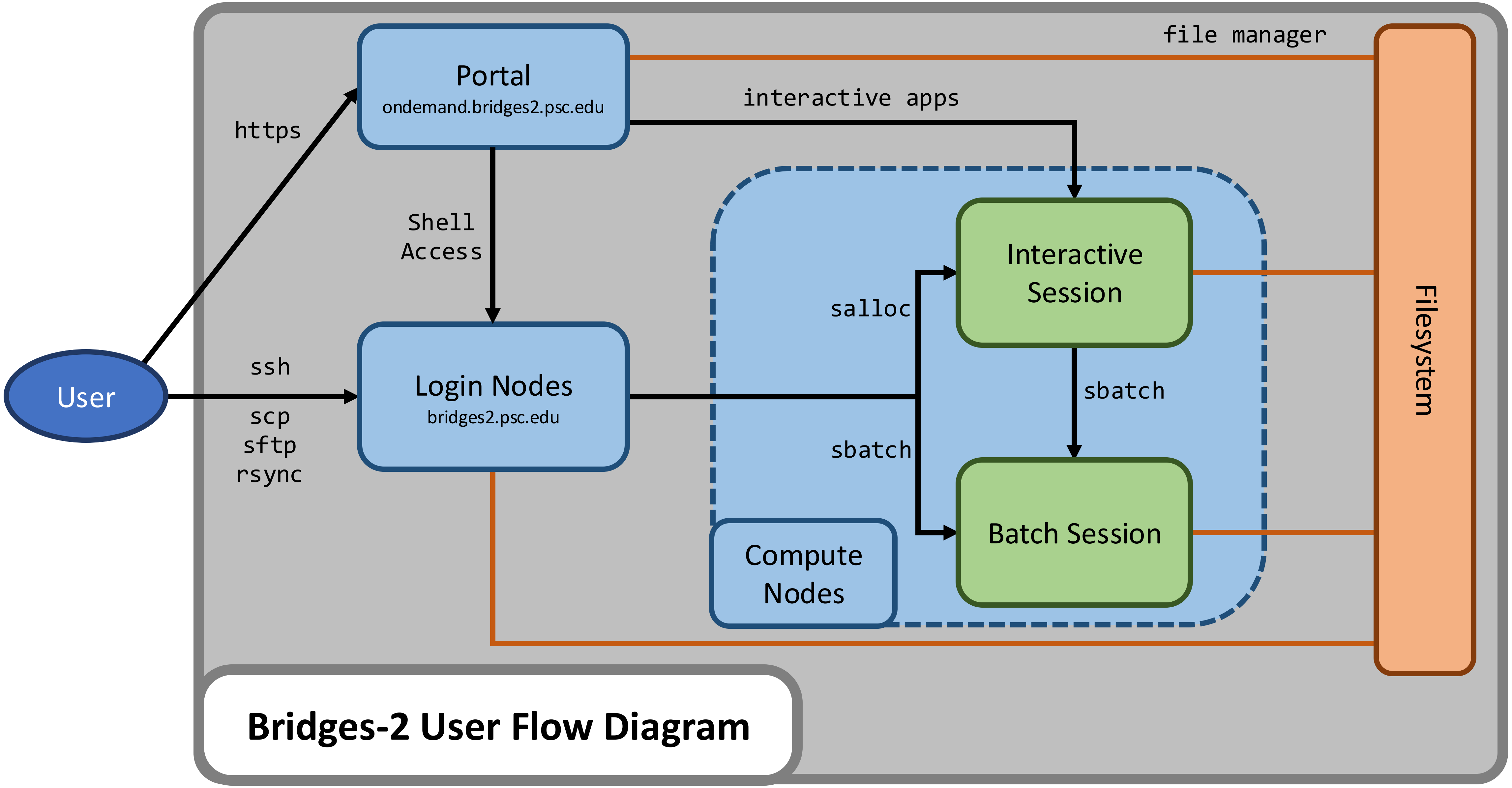
A useful environment variable is the SQUEUE_FORMAT variable and can be set, for example, with the following command:

```
export SQUEUE_FORMAT="%.7i %9P %35j %.8u %.2t %.12M %.12L %.5C %.7m %.4D %R"
```

Further details on the usage of this variable are available in Slurm's squeue documentation.

Another useful job monitoring command is: `scontrol show job <jobid>`

Cancel a job with: `scancel <jobid>`



Bridges-2 User Flow Diagram

Bridges-2 User Guide

www.psc.edu/resources/bridges-2/user-guide