Open Science Grid

Introduction to Distributed HTC and overlay systems

Tuesday morning, 9:00am

Igor Sfiligoi <isfiligoi@ucsd.edu>
Leader of the OSG Glidein Factory Operations
University of California San Diego

University of California
San Diego

About Me

- Working with distributed computing since 1996
- Working with Grids since 2005
- Leader of the OSG glidein factory ops since 2009
- Deeply involved in overlay system development and deployments
- Mostly worked with Physics communities (CMS, CDF, KLOE)

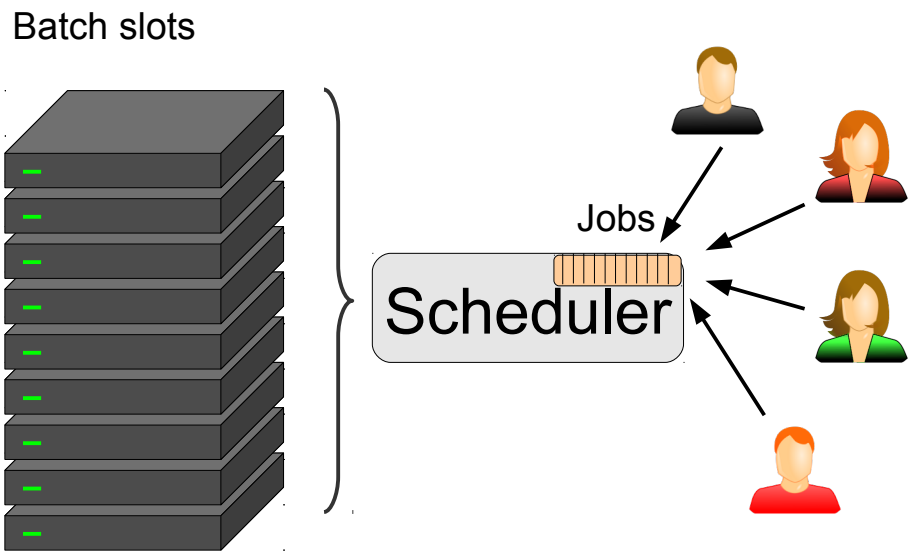
Logistical reminder

- It is OK to ask questions
  - During the lecture
  - During the demos
  - During the exercises
  - During the breaks
- If I don't know the answer,
  I will find someone who likely does

High Throughput Computing

- Yesterday you were introduced to HTC
  - Often called batch system computing
  - A paradigm that emphasizes maximizing useful computation, given a fixed number of resources

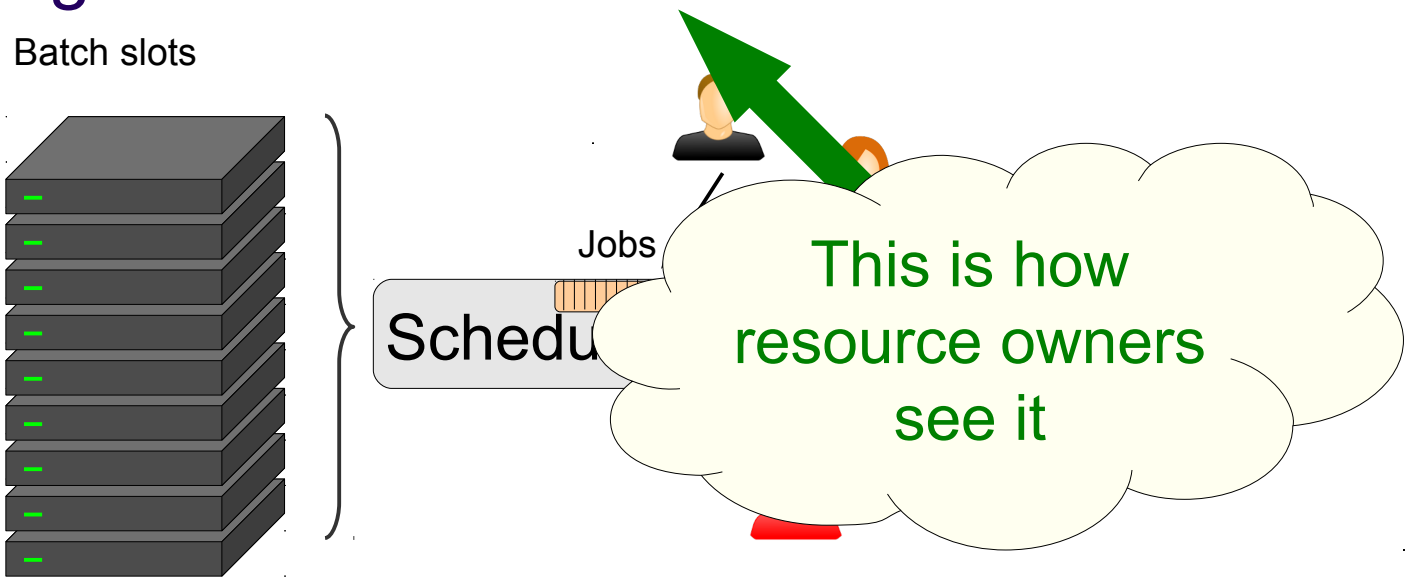
Max ROI
in business-speak



High Throughput Computing

- Yesterday you were introduced to HTC
  - Often called batch system computing
  - A paradigm that emphasizes maximizing useful computation, given a fixed number of resources

Max ROI
in business-speak



High Throughput Computing

- HTC from the **user point of view** is not too much different; it is
  - Maximizing useful computation before a fixed deadline
    - With the deadline being at least a few days in the future
- The available (fixed) resources just set what can be achieved
  - Users (should) expect to get a subset at any point in time

Admittedly very few users presently operate this way

Introducing DHTC

- So what is **Distributed** HTC???
  - HTC is always distributed, right?
  - However, HTC is normally considered within a single cluster
- DHTC is about
  - Globally maximizing useful computation of several independent compute clusters
  - Located in many locations and operated by many entities

Introducing DHTC

- So what is **Distributed** HTC???
  - HTC is always distribut
  - However, HTC is non
    within a single cluster
- DHTC is about
  - Globally maximizing useful computation of several independent compute clusters
  - Located in many locations and operated by many entities

Again, this is how resource owners see it

A user perspective on DHTC

- For users, a DHTC system is just an HTC system that
  - Has more resources
  - Is likely more heterogeneous
  - Unlikely to have a shared file system
  - Has compute nodes far away, network wise
- For some DHTC systems
  - The total amount of available resources may not be known in advance

A user perspective on DHTC

- For users, a DHTC
  just an HTC syste
  - Has more resources
  - Is likely more heterogeneous
  - Unlikely to have a shared file system
  - Has compute nodes far away, network wise
- For some DHTC systems
  - The total amount of available resources may not be known in advance

Tomorrow's topic.

i.e. you will have to do explicit data movement.

DHTC vs HTC

- So, for jobs with small data needs, a DHTC system is **remarkably similar** to a “regular” HTC system

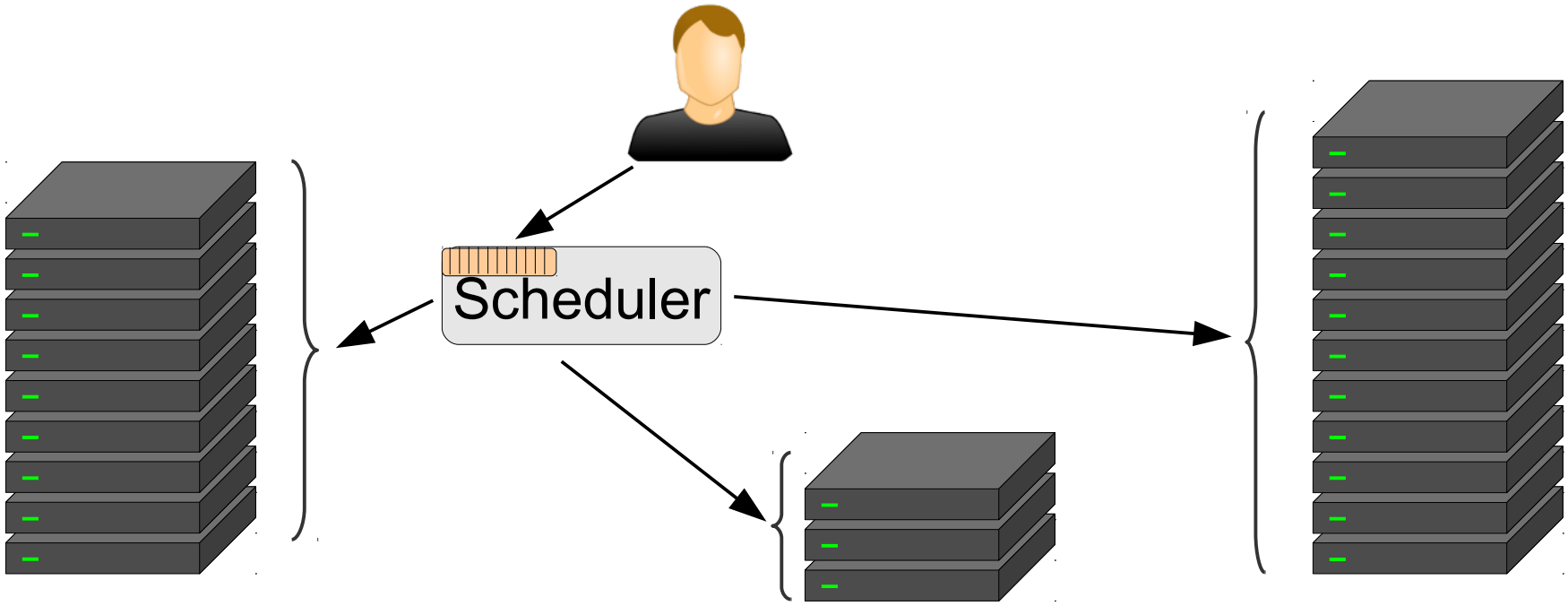
So why do we have a lecture on it???

Why a dedicated DHTC lesson?

- Two main reasons
  - Will have hands-on with a DHTC system
    - So knowing some details may help understand
  - We expect several of you to be more than just “dummy computer users”
    - So you may end up running,
      or at least help operating,
      a DHTC system

Anatomy of DHTC

- Basically, DHTC is about computing on resources **located in more than one place**

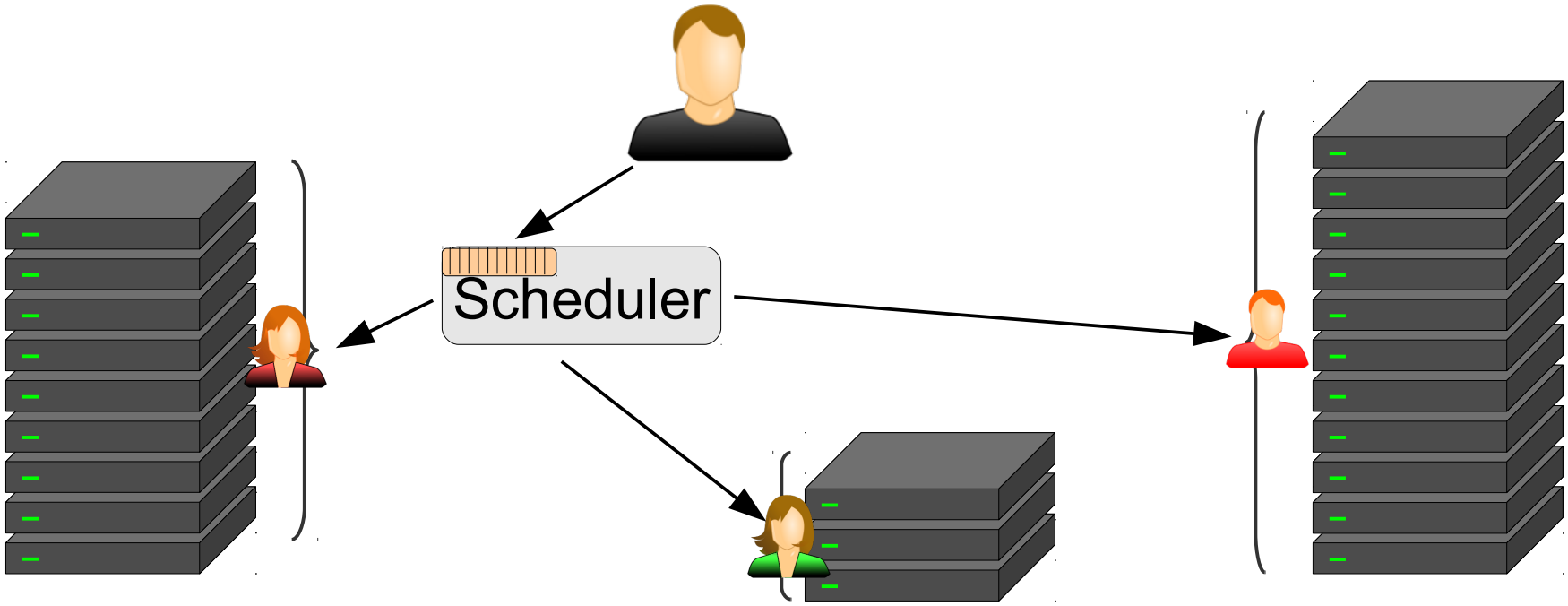


Why DHTC

- Many reasons:
  - Practical
    (a site/entity has a limit to how much HW can host)
  - Political
    (you only get money for HW if it is hosted at X)
  - Economical
    (hosting and operating HW myself is too expensive)
    (someone else can offer you hosted HW for less)
  - Opportunistic
    (HW on site X is temporarily idle, might as well allow others to use them (for free or for pay))

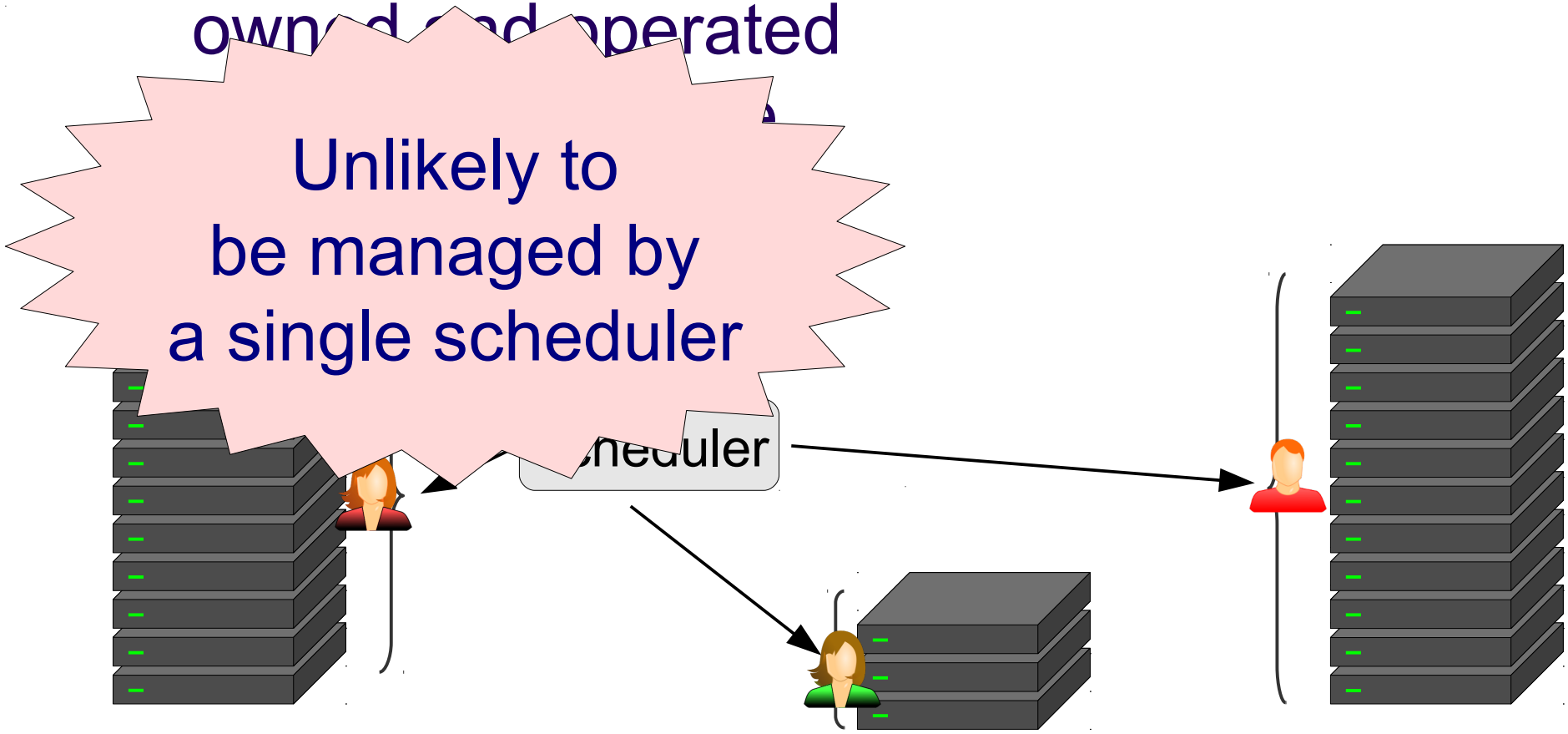
Multiple owners

- Different clusters most likely to be owned and operated by different people



Multiple owners

- Different clusters most likely to be owned and operated



Why no global scheduler?

- Local users, local policies
- Existing infrastructure
- Different technology preferences
- Being able to work when WAN goes down
- Money & politics
- ...

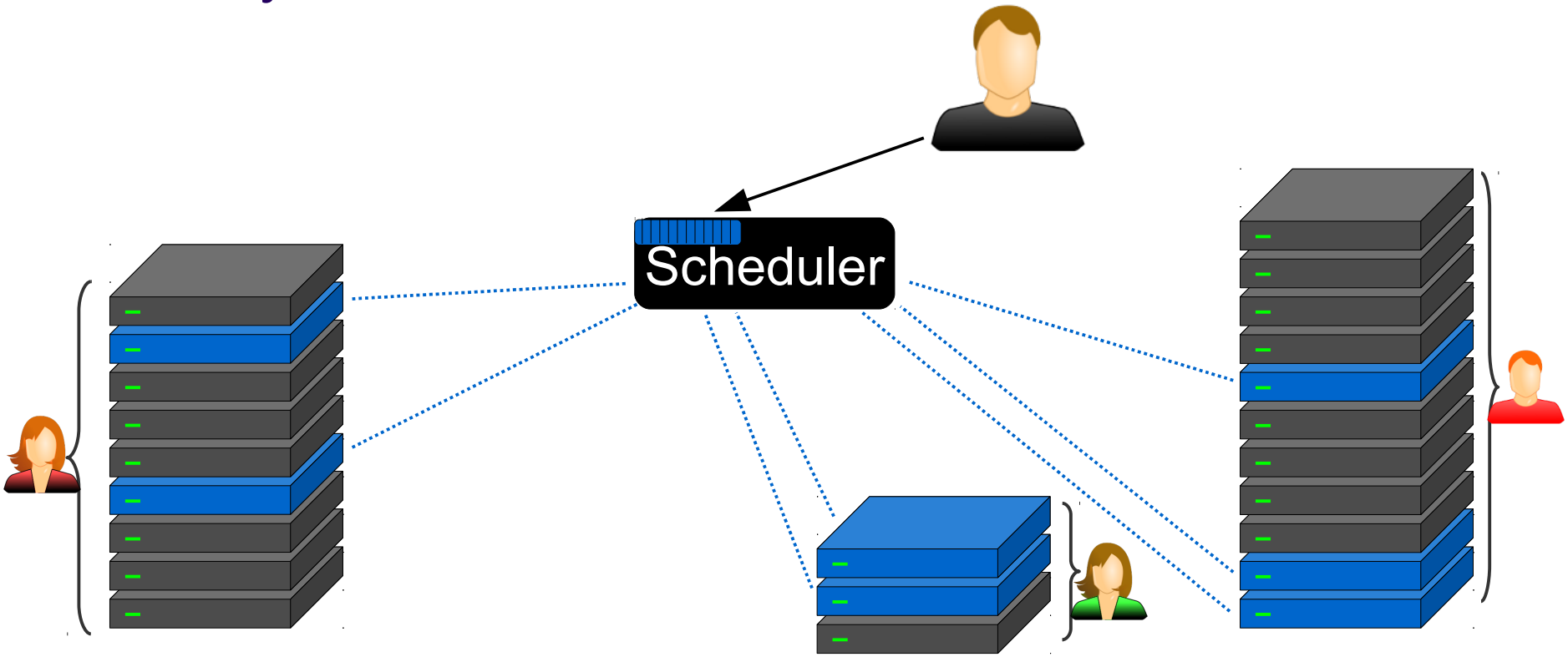
Why no global scheduler?

- Local users, local policies
- Existing in
- Diff
- Be
  wh
- Mor
- ...

But users still want
to have a
single scheduler
with resources from
multiple clusters

Stitch together leased resources

- Imagine leasing a subset of resources
  - Once you have them,
    you decide what to do with them

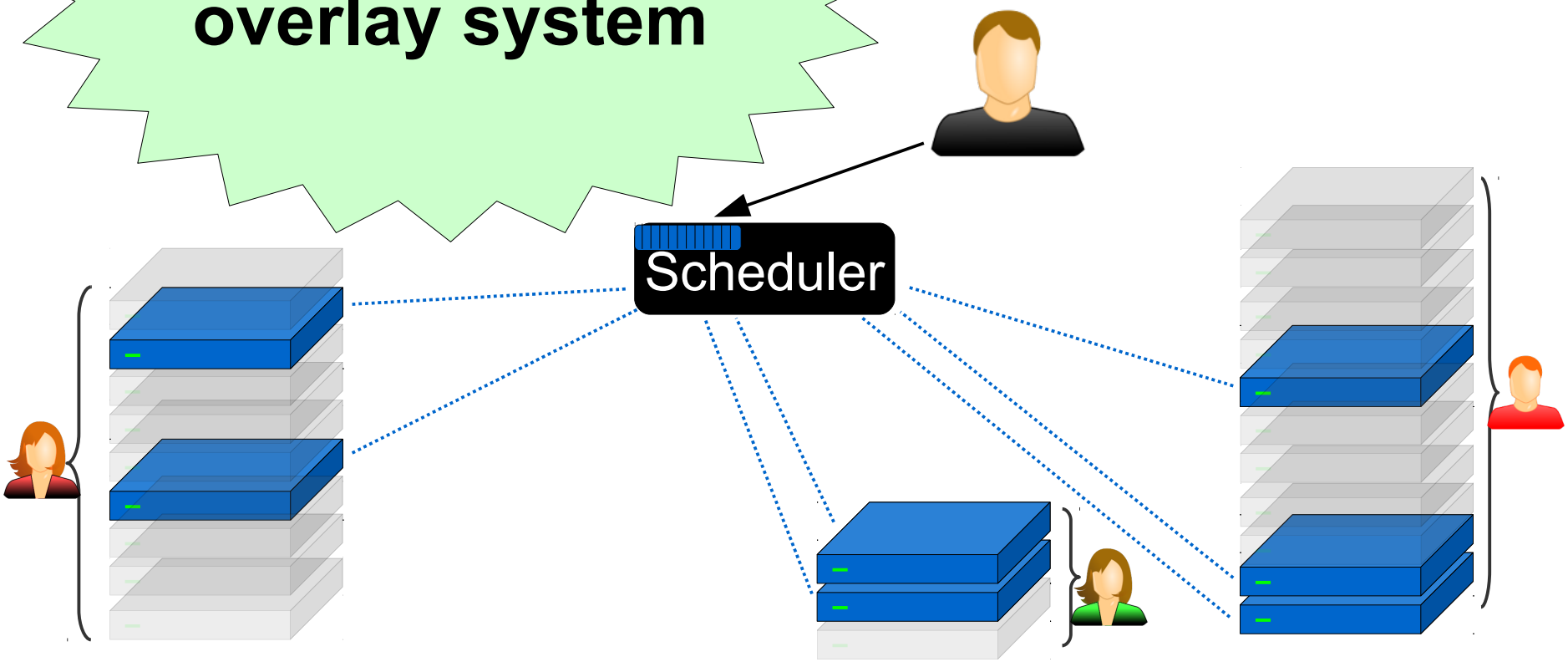


Stitch together leased resources

subset of resources

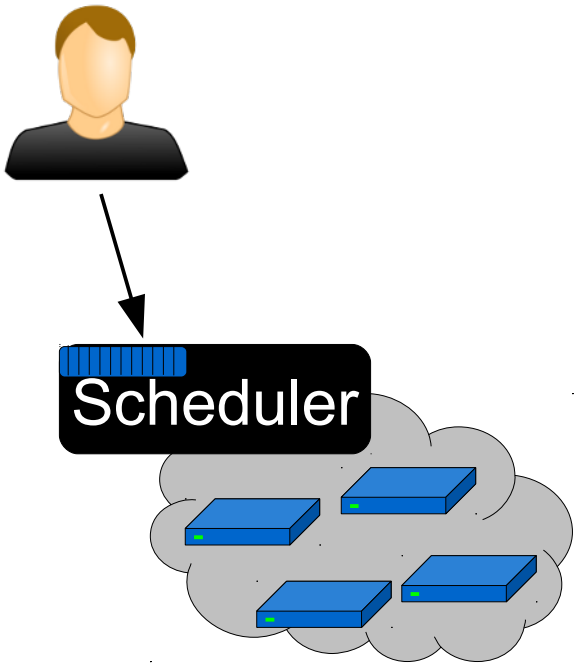
with them

We call this an
overlay system



Overlay systems

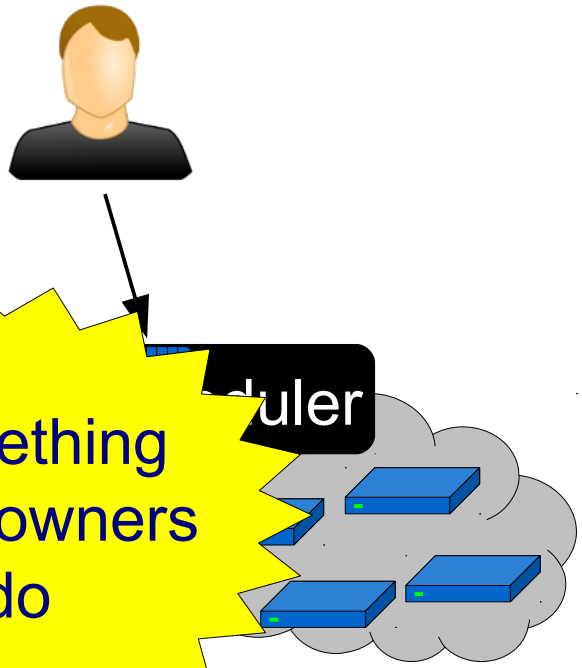
- From the user point of view, it is just **a single, global** scheduler
  - DHTC drawbacks still apply, of course (e.g. slow networking)



Overlay systems

- From the user point of view, it is just **a single, global** scheduler
  - DHTC drawbacks still
    a course
    orking)

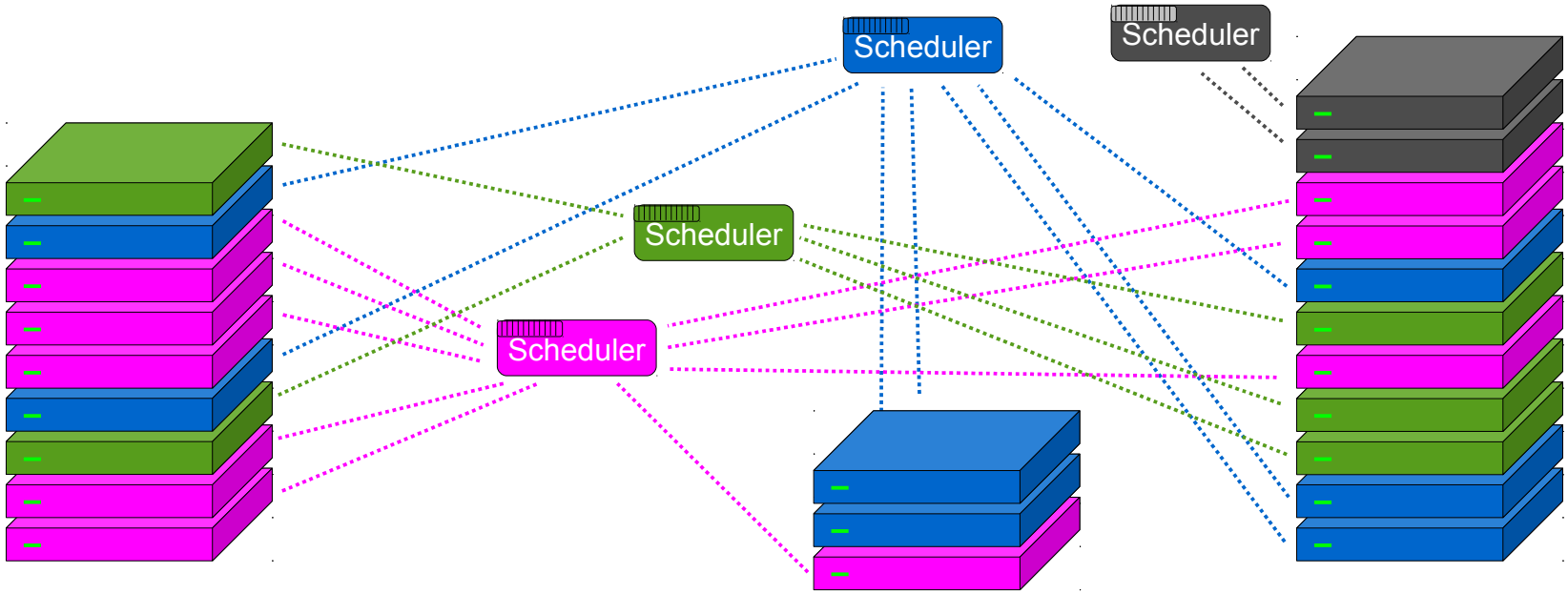
We just need someone to actually create the overlay



Not something resource owners will do

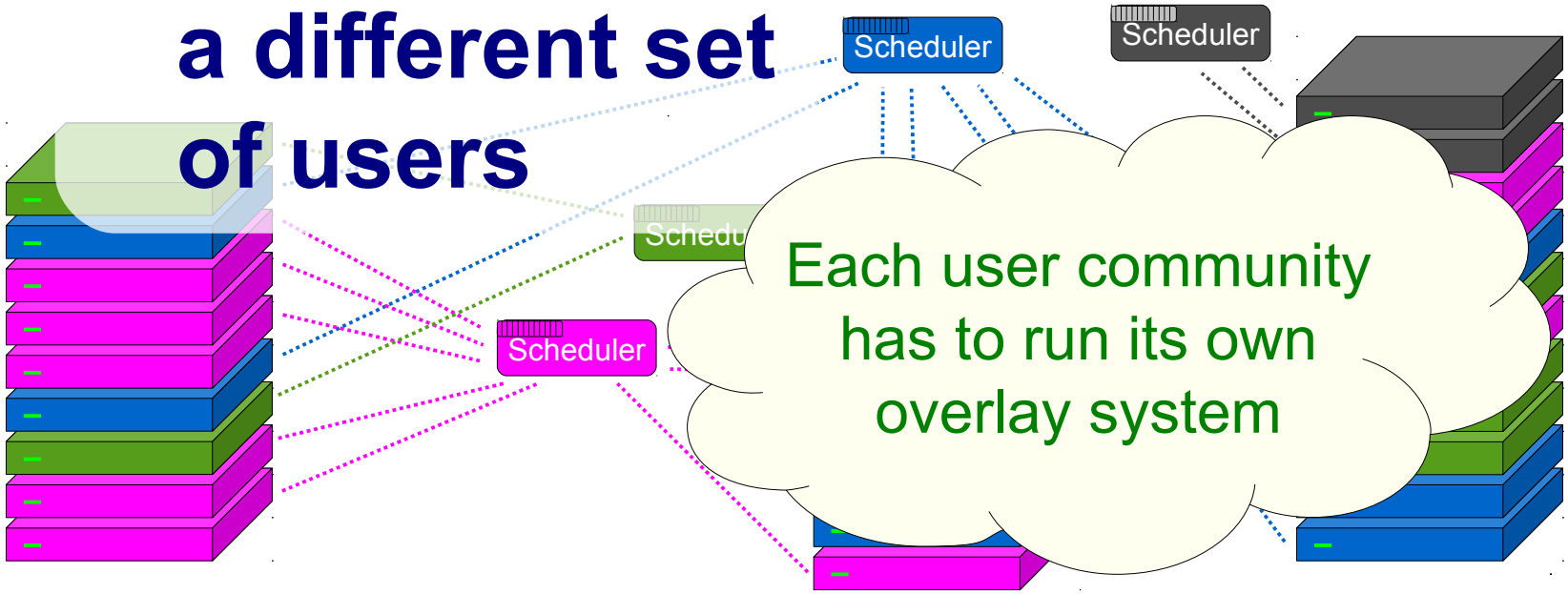
Many independent schedulers

- By creating overlays, there can be several independent schedulers
  - Each handling a subset of resources
  - Each serving a different set of users



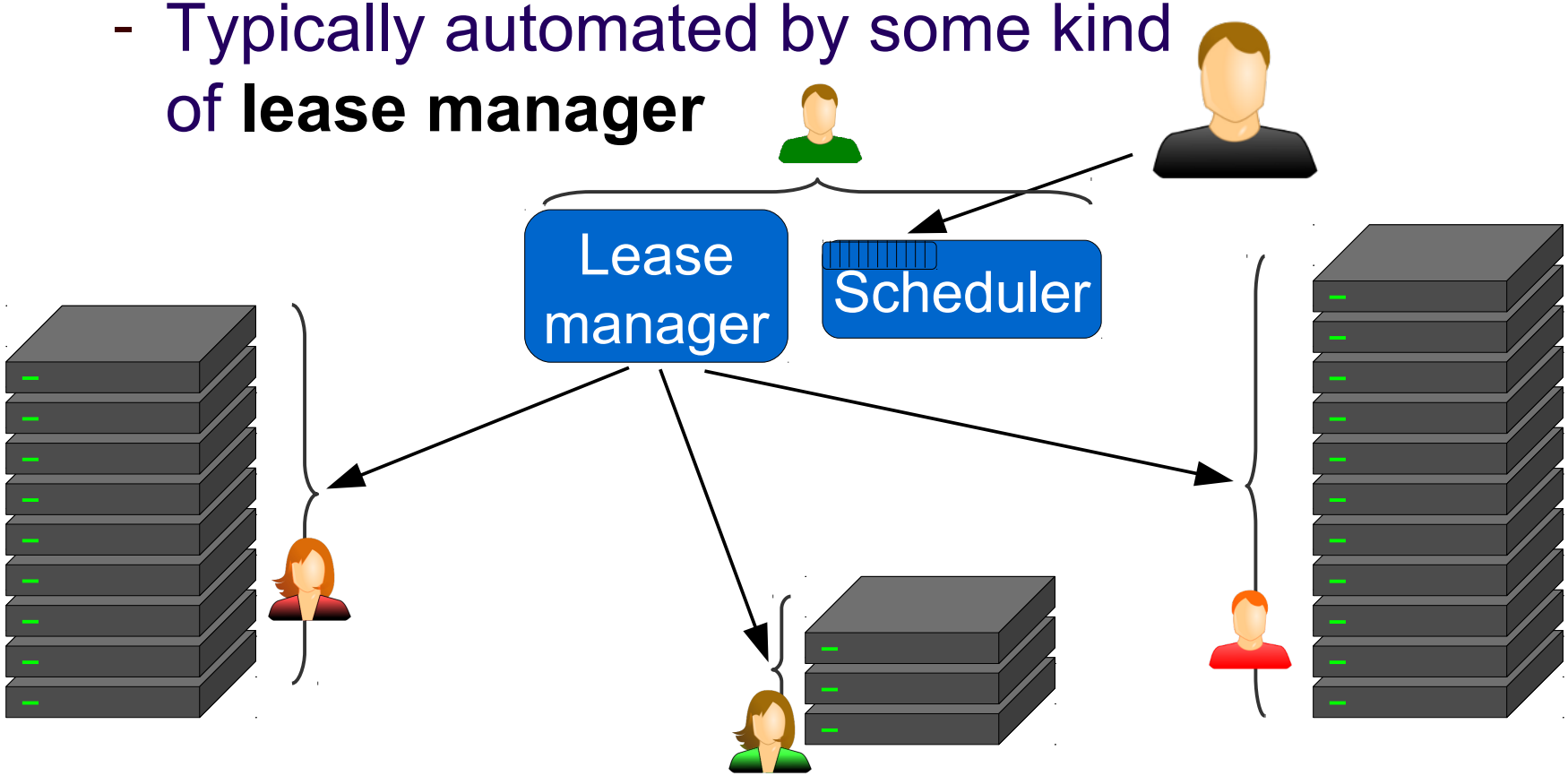
Many independent schedulers

- By creating overlays, there can be several independent schedulers
  - Each handling a subset of resources
  - **Each serving a different set of users**



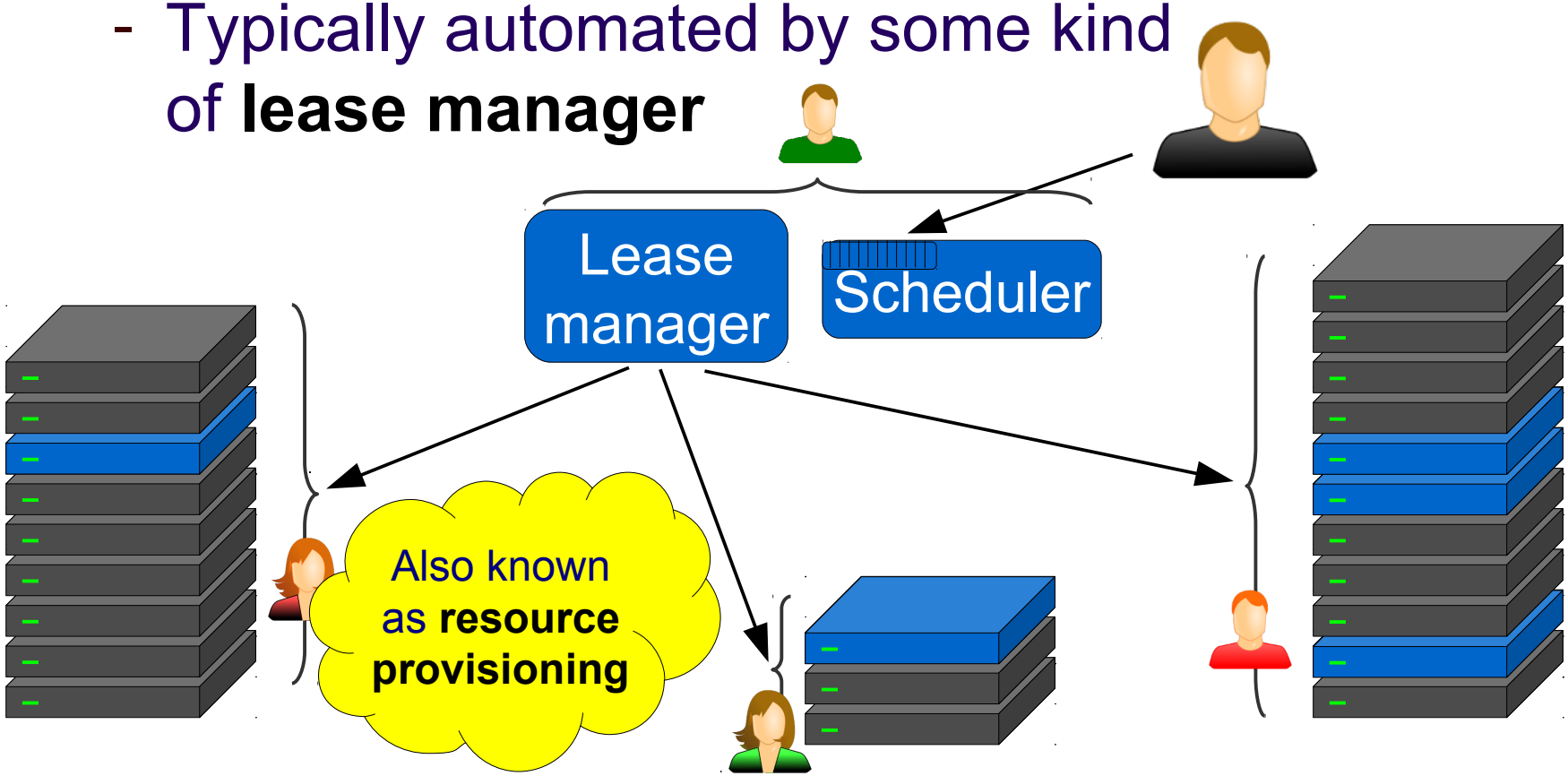
The need for a lease manager

- The overlay system has to get exclusive rights to a set of resources
  - Typically automated by some kind of **lease manager**



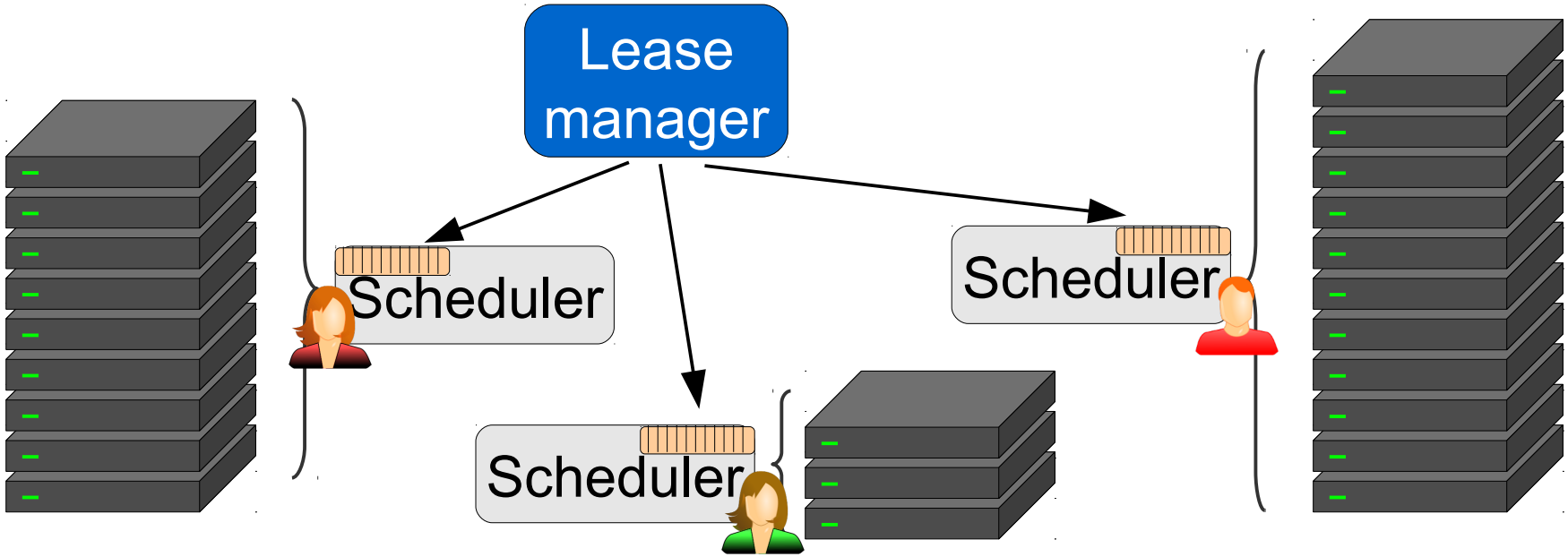
The need for a lease manager

- The overlay system has to get exclusive rights to a set of resources
  - Typically automated by some kind of **lease manager**



Layers of HTC

- Resource providers typically want to maximize their resource use
  - Thus configure them as HTC
  - i.e. they have their own layer of scheduling



Layers of HTC

- Resource providers typically want to maximize their resource use
  - Thus configure them as HTC
  - i.e. they have their own layer of scheduling

Pilot jobs

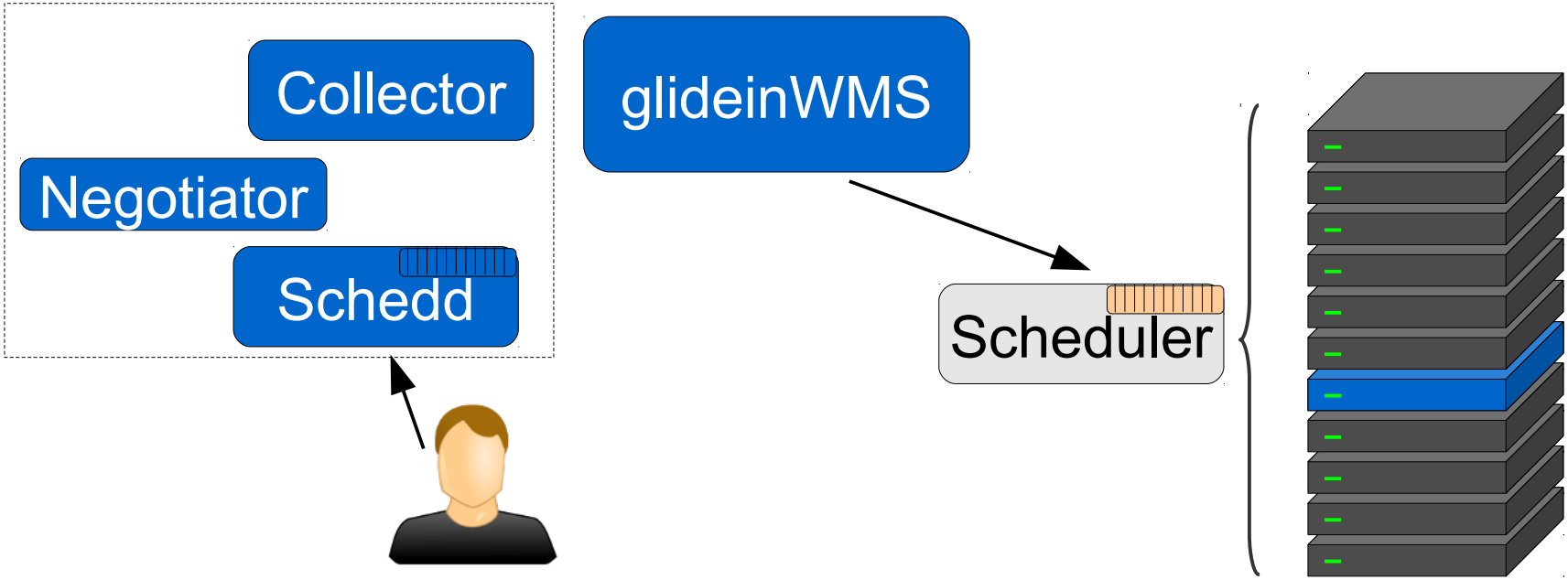
- The lease requests are known as **pilot jobs**
- Each pilot job holds the lease for the lifetime of the job
  - Which is typically relatively short
  - User jobs thus cannot run longer than the limits imposed by the sites

Overlay systems

- Many possible implementations
- We will concentrate on **glideinWMS**
  - Based on HTCondor
  - The one used by most user communities on OSG
- Others available
  - PANDA, DIRAC, ALIEN, CycleCloud, ...

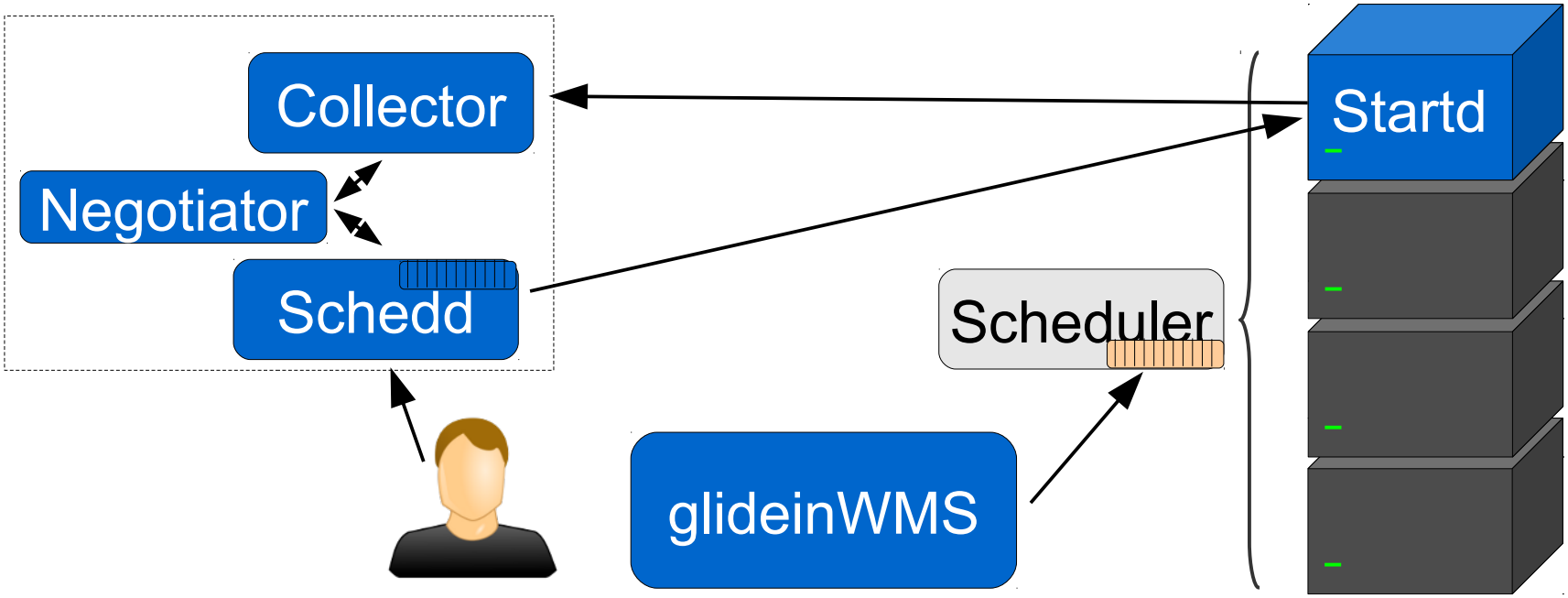
glideinWMS

- A HTCondor based overlay system
  - i.e. looks like a regular HTCondor system to the users
  - Adds a resource provisioning service
    (i.e. the lease manager)



HTCondor pilots

- HTCondor pilot == A glidein job
  - Basically a properly configured HTCondor Startd



Two level matchmaking

- The system now has **two matchmaking points**
  - The **glideinWMS** decides when and where to send glideins
  - The **Condor negotiator** decides which job runs on which glidein
- The two must treat jobs the same way
  - Or we end up with either unused glideins or jobs that never start

Moving policy in glideinWMS

- In glideinWMS, **user jobs never have requirements**
- All policy is implemented by system administrators
  - Users **just provide parameters**

Example policy

```
(DESIRED_SITES=?="any") || stringListMember(GLIDEIN_Site,DESIRED_SITES)
```

Example job JDL

```
+DESIRED_SITES = "UCSD,Wisconsin"
queue
```

Example job JDL

```
+DESIRED_SITES = "any"
queue
```

Know your system

- The matchmaking is thus less flexible
  - You can only work within the frame of the system policy
- But arguably easier to use
  - No complex boolean expressions to write
- Be sure to ask for the system policy of your system

Down to practice

- This is all for the theoretical part
- Next we have the hands-on session

Questions?

- Questions? Comments?
  - Feel free to ask me questions later:
    Igor Sfiligoi <isfiligoi@ucsd.edu>
- Upcoming sessions
  - Now - 11:00am
    - Hands-on exercises
  - 11:00am – 11:15am
    - Break
  - 11:15am –
    - Next lecture – How to get the needed computing

Beware the power

Courtesy of fanpop.com