Technology and Software

Brian Bockelman
OSG AHM 2015

OSG Technology

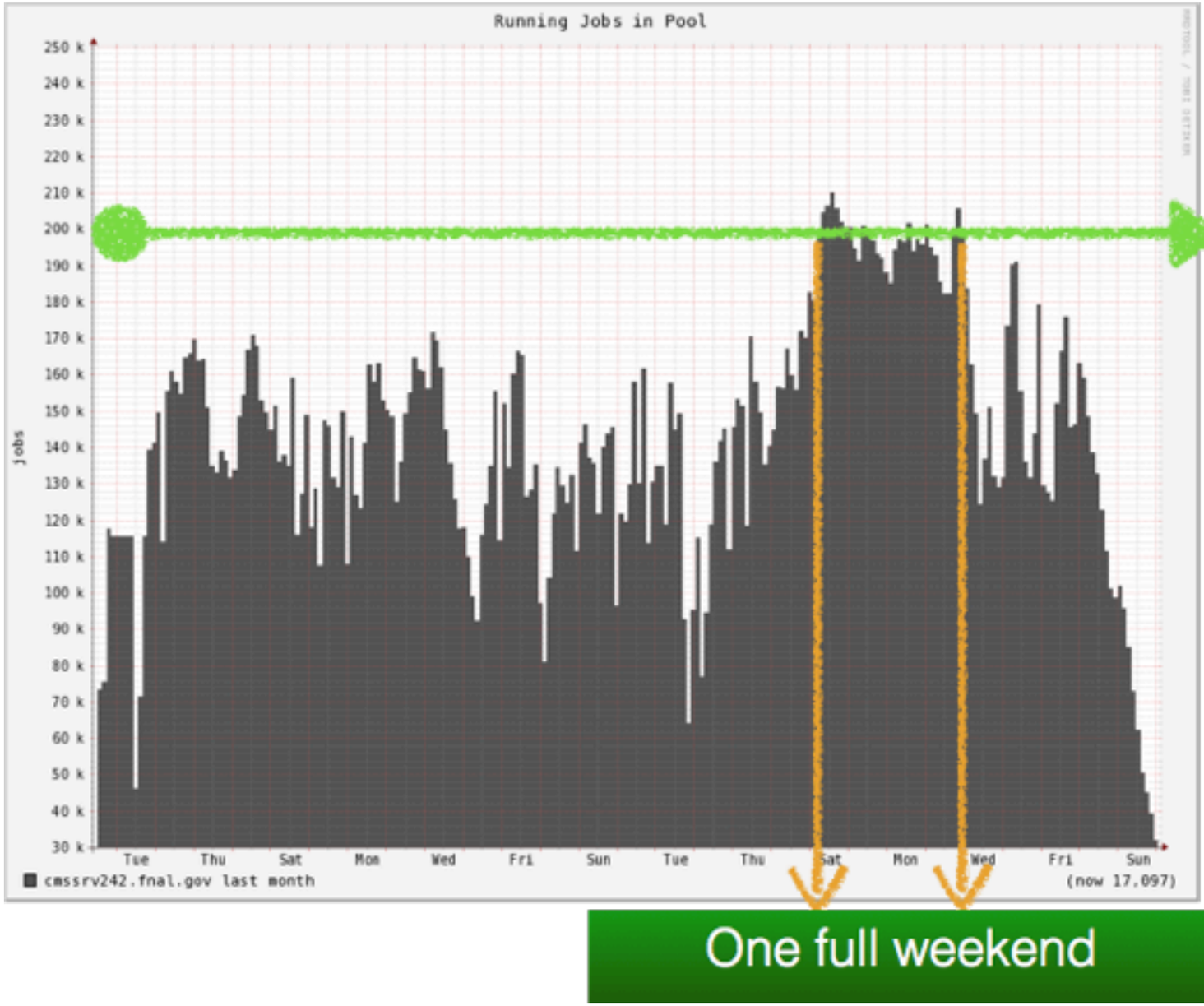
- OSG software allows the OSG and sites to advance the science of DHTC.
- A few thrusts of the upcoming year:
    - Provide a stable environment for LHC Run 2.
    - Design and deliver the next stable series of software.
    - Deploy new technologies into the software stack.
    - Delivering a coherent technology platform for opportunistic VOs.
- OSG Technology does not operate alone: OSG is an integrated ecosystem along with Ops, Security, Production and User Support. Refer also to their presentations.

Stable Running Environment

- This is more of a “mindset change” for the OSG Software team than a particular deliverable.
    - Prioritize bugs / tickets over features.
    - Formalize testing & QA processes.
    - Contrast to the OSG 3.0 transition, where we purposely sacrificed quality for speed of delivery.
    - Separate “aggressive items” onto a separate upgrade track.
- Likely mirrors changes being made in the LHC computing organizations.

Example: HTCondor Scaling

- Collaboration between OSG, CMS, and HTCondor team.
- Hit goal of 200k — but that took around 6 months of coordinated debugging, patching, and software releases.
    - Significant time investment - I don’t think this could have been done without OSG involvement!
- HTCondor-CE validated up to 16K running jobs; hope is to try again this summer to get up to 20k.



Software Stack Basics

- In 2013: OSG 3.1 had 16 releases; OSG 3.2 had 7.
- In 2014: OSG 3.1 had 14 releases; OSG 3.2 had 15.
- In 2013, there were 425 software tickets closed.
  In 2014, there were 404 closed.
- About 25% of our packaging is “pass-through” - straight copy of upstream or EPEL.
  - From 3.1->3.2, we saw a 20% decrease in total number of packages.
  - We have ~150 RPMs we maintain.

Designing OSG 3.3

- In April, we drop all support for OSG 3.1; accordingly, we've begun planning for OSG 3.3.
- New series are our mechanism for introducing large-scale changes.
    - OSG 3.0 -> 3.1 introduced support for RHEL6.
    - OSG 3.1 -> 3.2 dropped about 20% of our packages.
- My goals for OSG 3.3:
    - From OSG 3.2 to 3.3, drop another 20% of our packages (15% drop 3.1 to 3.2).
    - We ship 4 SRM clients; I'd like to drop this to 1 (gfal2).
    - I'd like to remove all Java components from the CE and WN.
    - Drop RHEL5, add RHEL7 support.
    - Remove osg-client?
    - Split metapackages to "basic" and "advanced" variants.

Designing OSG 3.3

- **However**, we are still “requirements gathering” and welcome community input over coffee and lunch breaks this week. All ideas - from boring to crazy - will be listened to!
    - Since OSG 3.2 will be the stable series for a long time yet, 3.3 will have a gradual rollout. I want to err towards more aggressive package removal for 3.3**.0** and add back as needed.
        - The long rollout will help us keep the “stable running environment” goal.
- With an eye toward stability, it may take a few releases before we’d recommend WLCG sites to use the 3.3.x series.

The Software Orphanage

- Over the past several years, we've had the concept of the "software orphanage" - software whose original maintainers have moved on but is still critical to our stakeholders.
- Biggest examples include:
  - bestman2
  - GUMS
- Software in this state are significant drains on our effort; we aim to reduce scope and use cases as best possible.

Software Testing and Release

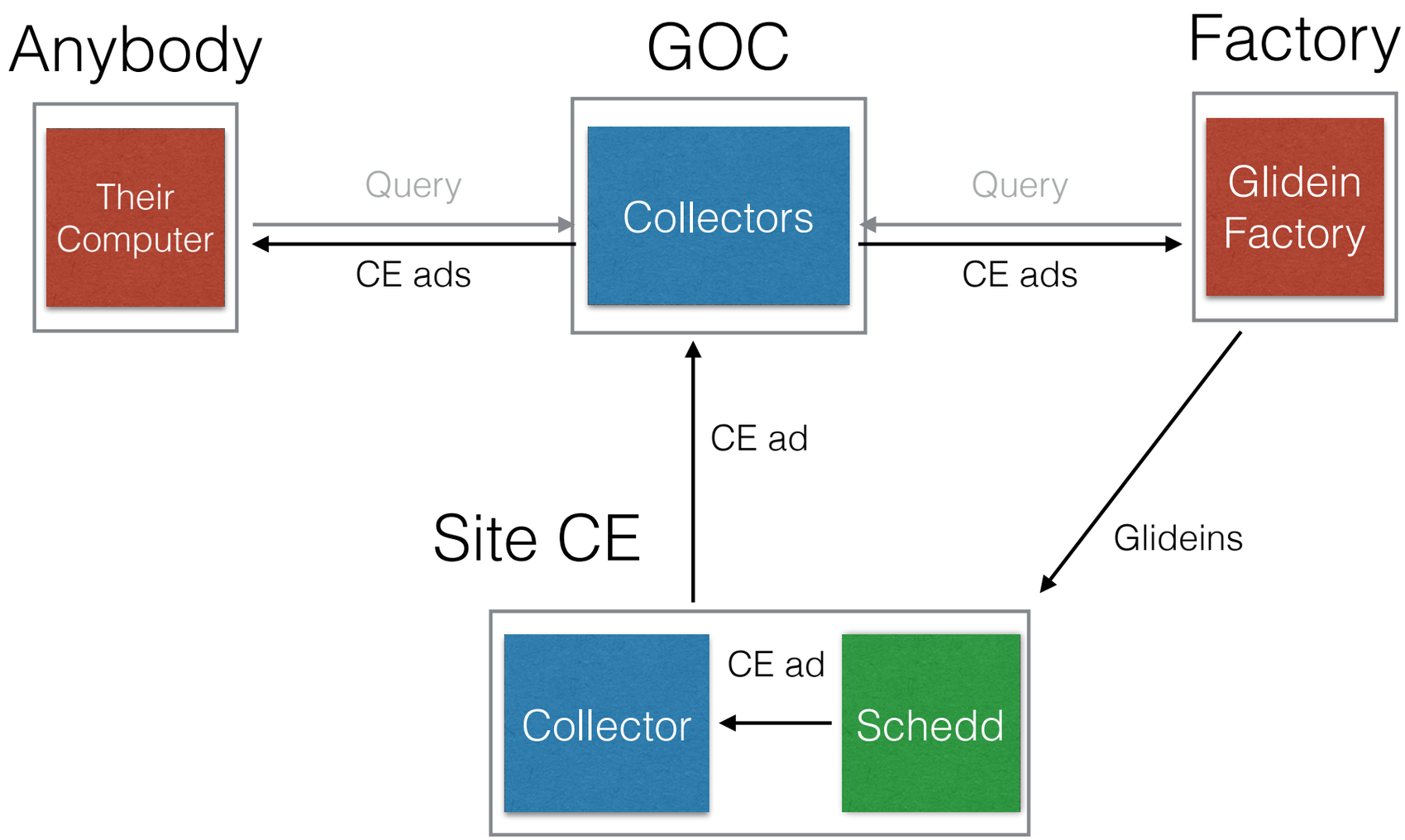
- Software integration testing has hit its stride in the last year.
  - Coverage of software and functionality continuously increases.
    - “osg-test” toolset has become critical - the full matrix of release series and platforms is otherwise overwhelming.
    - Each night, we launch hundreds of VMs for different tests and platforms.
- Release team has also hit its stride. We’ve maintained a constant cadence and .

What new services are we looking at?

CE Collector

- We have been doing information services wrong throughout the life of the grid.
    - Projects like the BDII have been an attempt to generalize the description of the state and queues of an LSF system.
        - You can generalize for other batch systems, but at the core, it is still optimized for a particular use case; works poorly for our needs.
- The CE collector publishes a description of the HTCondor-CE, the resources accessible, and how to access the resources.
    - No monitoring information! No hardcoded concept of the queue!
    - Currently, our one focus is to get core use case - providing the information necessary for provisioning systems - right on HTCondor-CE. Additional use cases may follow.

Architecture (planned)



https://indico.fnal.gov/getFile.py/access?contribId=19&sessionId=8&resId=0&materialId=slides&confId=8580

The (Local) Collector

- We've always wanted more information about payload jobs.
    - Who's running? What are they running? Are they using CPU efficiently?
    - No communication channel between the pilot and site!
- In the next HTCondor-CE release, the CE will allow pilots to send startd ads (representing the payload jobs). The CE admin can view the payload activity with condor_status.
- In the next gWMS release, the pilot will send these ads automatically.
    - Any other pilots are welcome to use the "condor_advertise" tool to inform the site about what they are doing.

(A) Technology Vision

- OSG enables a plethora of different technologies.
  - Many different overlaps; optimized for different use cases.
  - These overlaps are a *good* thing; VOs can pick and choose the technology they use.
- In the next few slides, I'm going to outline a specific technology stack that is starting to develop.
  - Tailored for the use case of small or opportunistic OSG VOs. Used by the OSG VO.
  - Do not view this as a narrowing of our stack - rather a focus on rounding out the services for this use case.
    - As a counter-example — I would like to see OSG Software finish our packaging of PanDA and AutoPyFactory.

A Technology Vision

Jobs -> HTCondor
Software -> OASIS
Data -> ???

Jobs -> HTCondor

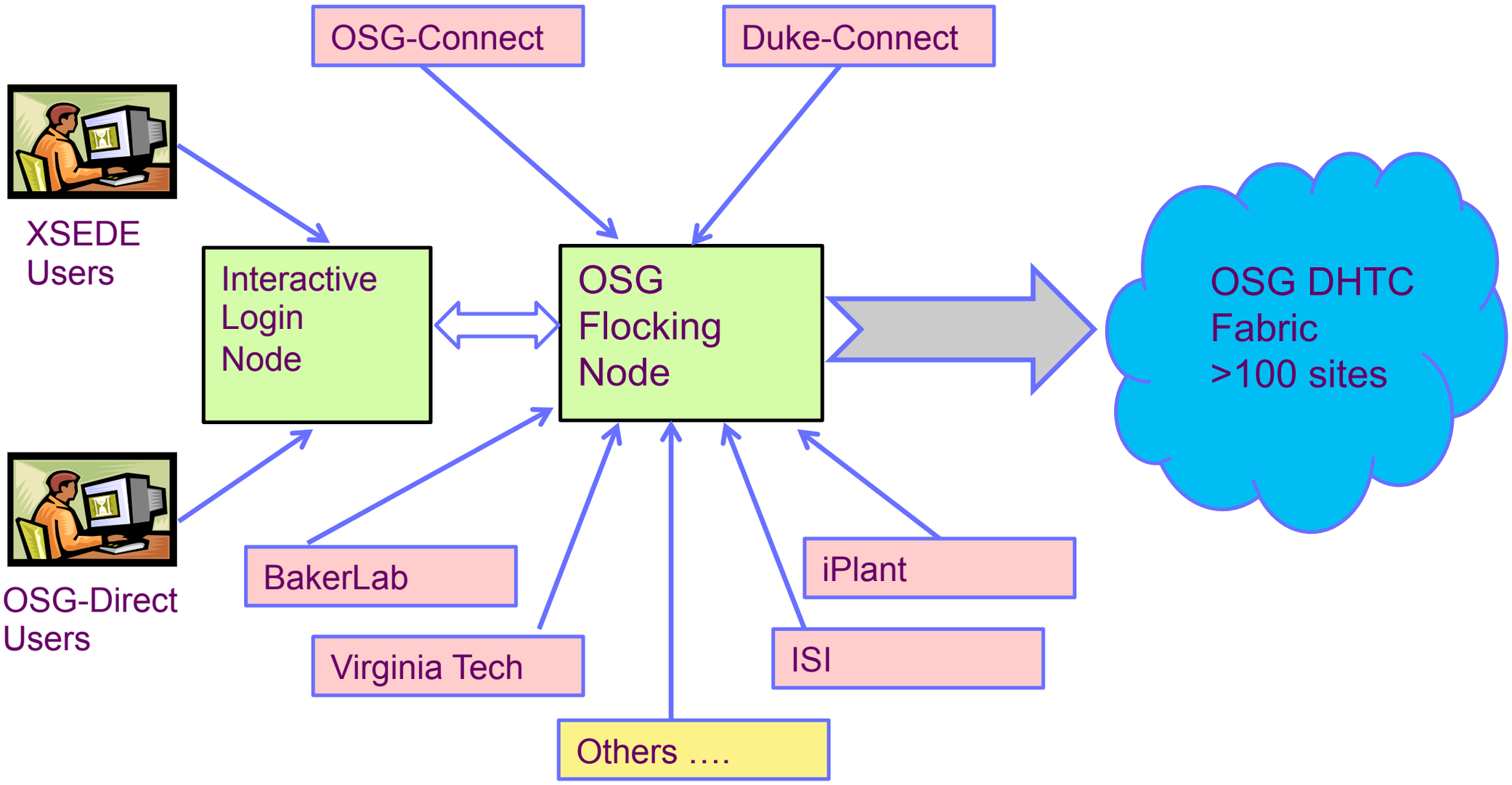
One of the most mature services offered by OSG is management of HTCondor pools.

Jobs -> HTCondor

- This is the oldest and most mature component.
  - Simple idea: Use “normal” HTCondor to run your jobs.
  - Variety of workflow managers can be layered on top - popular choices are homegrown, DAGMan, or DAGMan+Pegasus.

- OSG offers three levels of service:
  - **We run all HTCondor components** and provide a submit host; just bring your jobs! (“OSG-Connect”)
  - Link your HTCondor submit host to our pool; **you run the submit host**, we run the pool (“OSG-Flock”)
  - **Run your own HTCondor** submit host and pool; we’ll fill it with worker nodes. (“GFactory service”)

Access to OSG DHTC Fabric via OSG VO



All access operates under the OSG VO using glideinWMS

Slide courtesy of Chander Seghal

HTCondor Service

- Each comes with some “cost”: the more components OSG runs, the more policies and restrictions the VO/user has to follow.
    - The more components the VO/user runs, the higher their operational cost.

- **OSG-Connect**: In this case, OSG decides on fairshare of users. OSG configures the schedd, and users must log in to a central host.
- **OSG-Flock**: VO still has to use the OSG pool; we still decide fairshare and use the OSG VO (sites cannot prioritize between OSG users).
- **GFactory Service**: VO is in control of fairshare, all configs, and DNs.
- The HTCondor Service is often the “hook” we first give to new VOs!

Software -> OASIS

- In the “old bad days”, VOs were told to write and maintain their software into an NFS directory on every site.
    - It was their problem to figure out how to do this, share space with others, detect and fix corruptions, etc. **GOOD LUCK**.
- For the last 2 years, OSG has provided a hosted service, OASIS.
    - Get your VO enabled, login to our central server, install your software, and hit the “publish” button.
    - OASIS handles file integrity (checksumming) and distribution (cache management).
    - Implementation is a specialized install of CVMFS as well as a few wrappers.

Software -> OASIS

- New service delivered last year:
    - The **VO runs a repo**, we sign and replicate it.
- Looking forward to the next year:
    - As the service becomes popular and gains users, the users like to push boundaries.
    - Working to make the boundaries more explicitly stated in the acceptable use policy.
    - Exploring what other resources we should protect with **technical mechanisms**. In many prior cases, we could previously survive with **social mechanisms**. Typically, the most critical component is the Stratum-1 servers that mirror the data.
- Even further: looking at possible modifications to CVMFS client so pilots can (safely) define the configuration. The VO could run & replicate their own repo - no need for OSG to sign it!

OASIS

- I want to help the CVMFS technology evolve so 'VOs can help themselves' without involving OSG.
    - In HTCondor, we say "submit locally, run globally". In OASIS, I want "**install locally, use globally**".
        - Many technical obstacles left to go: as we decrease the potential "gotchas", we can decentralize.
- I want the relationship with CVMFS to better mirror our relationship with HTCondor -
    - Steady stream of goals and deliverables.
    - Keep development external from OSG.
- For sites, we'd like to provide more automated client configuration (analogous to how OSG Software distributes CA certificates).

A Technology Vision

Jobs -> HTCondor
Software -> OASIS
Data -> ???

A Technology Vision

Jobs -> HTCondor
Software -> OASIS
Data -> StashCache?

Motivation -
Why not use what exists?

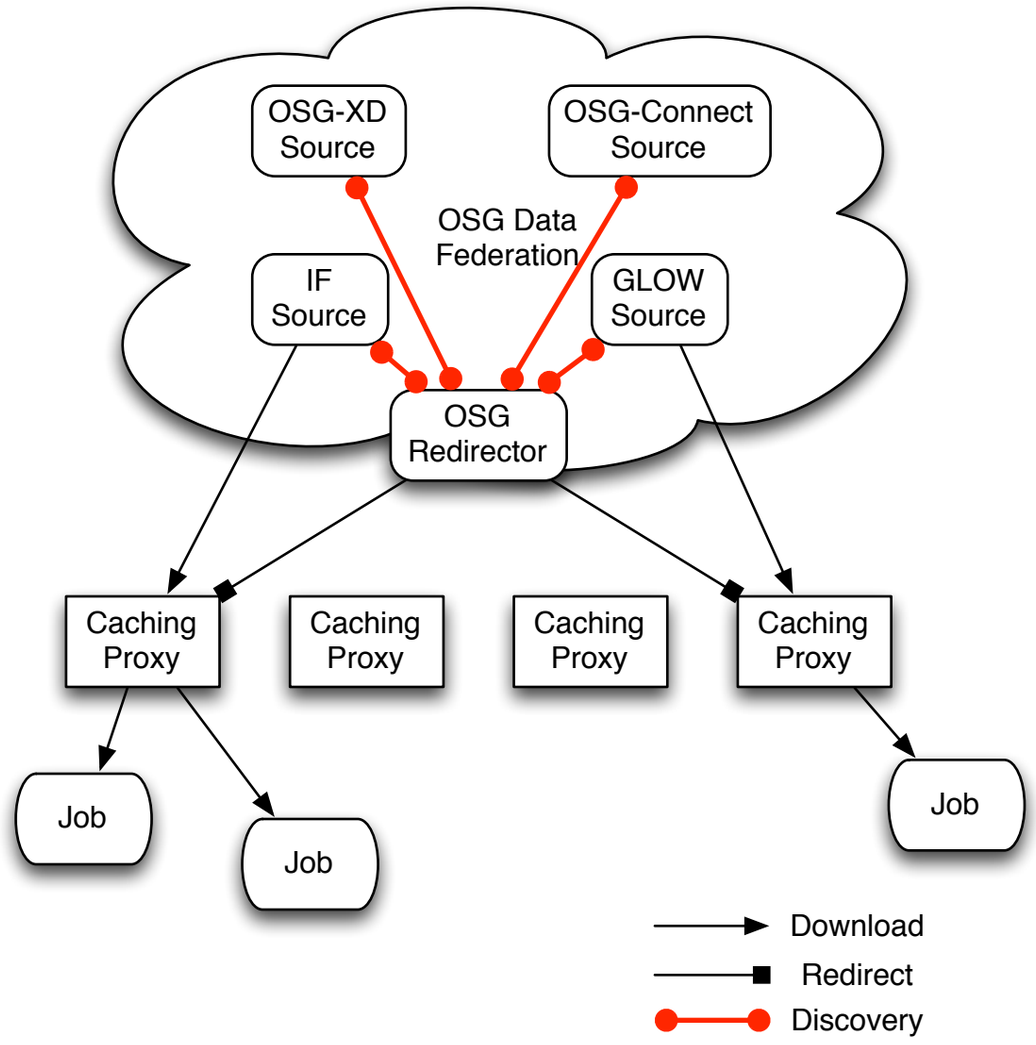
Opportunistic Computing is like giving away empty airline seats; the plane was going to fly regardless.

Opportunistic Storage is like giving away real estate.

(paraphrased from Mike Norman)

Data -> StashCache

- Using technologies from SLAC XRootD, AAA, and OSG-Connect.
- Introduce a cache-based paradigm for opportunistic VOs.
- See yesterday's talk!



Scale and Scope

- The possible origin servers are limited to OSG VOs.
    - During the pilot phase, we have a single origin server (OSG-Connect).
- **No service requirement for each site.**
- Each cache has minimum size (>10TB) and performance (10Gbps to WAN).
    - This allows us to provide reasonable lower bounds on acceptable working set size.
- Scale system so it can support ~10k running jobs.
- Scope of the system is limited to data stage-in, not stage-out.

Data Access Methods

- Xrootd is not a familiar protocol for users. Goal is to provide reasonable UIs to VOs; users don't care about protocols, they **care about interfaces**.
    - This means application protocol is implementation-defined; if protocol B is more relevant in 3 years, we can use that.
    - C.f., accessing "google.com" from Chrome does not use HTTP. Few users seem to care as long as the browser (the interface) works.
- To upload files, VOs can provide users with a writeable shared filesystem exported by the origin server.
    - Users first must "cp" their data to this mount point, then can access the files from their jobs.
    - Top-level directory name is assigned to VO by OSG; VO manages the namespace within their directory.
- User interfaces:
    - **"cp"-like**
    - **HTCondor file transfer**
    - **POSIX**

(An) OSG Technology Platform

- We hope this technology platform allows us to continue to expand the usability of DHTC for opportunistic VOs.

- I want to exploit these three good ideas to the fullest extent possible.

    - Over the course of the year, we will work to improve the integration between the three.

- Especially with StashCache, we will start with the tightly controlled environment - **OSG-Connect**.

    - OSG-Connect has been essential in leading the charge in providing a vision of providing a service that looks like a familiar environment (“virtual cluster”).

    - We will work hard to make sure the platform is available to all VOs.

- As we better understand the best approaches, we will then roll out improved packaging.

Parting Thoughts

- As always, we will temper our excitement for new features with a focus on stable running environments.

- OSG 3.3 planning is ongoing; will be our mechanism for defining large changes.

- As the core of the HTCondor-CE is starting to settle, we are looking to leverage other parts of the HTCondor ecosystem (collector).

- The combination of HTCondor, OASIS, and StashCache could form a powerful platform for opportunistic VOs. We will work to grow this throughout 2015!