HTCondor System Architecture and Administration Introduction

European HTCondor Workshop 2024

Todd Tannenbaum
Center for High Throughput Computing
University of Wisconsin-Madison

Center for High Throughput Computing

HTCondor Software Suite

PATh PARTNERSHIP to ADVANCE THROUGHPUT COMPUTING

HTCSS = HTCondor Software Suite

AP = Access Point

EP = Execution Point

CM = Central Manager

CE = Compute Entrypoint

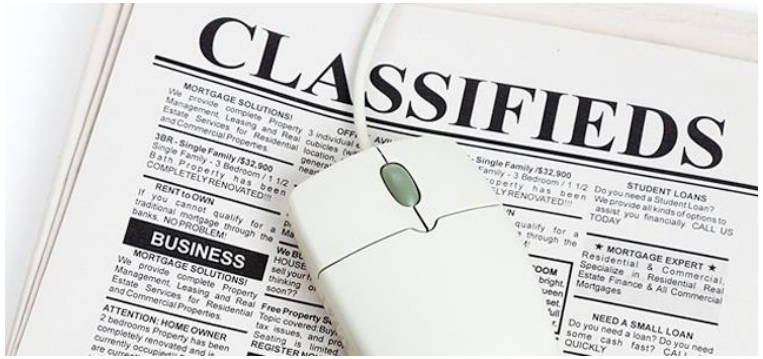
HTCondor Pool = CM + EP(s)

HTCondor System = CM + AP(s) + EP(s)

Job Matching and Class Ad Attributes

Class Ads

- HTCondor stores a list of information about each job and each computer.
- This information is stored as a “Class Ad”



- Class Ads have the format:

```
AttributeName = value
```

can be a boolean, number, string, or expression

ClassAd Values

- Literals
  - Strings (“RedHat6”), integers, floats, boolean (true/false), …
- Expressions
  - Similar look to C/C++ or Java : operators, references, functions
  - References: to other attributes in the same ad, or attributes in an ad that is a candidate for a match
  - Operators: +, -, *, /, <, <=,>, >=, ==, !=, &&, and || all work as expected
  - Built-in Functions: if/then/else, string manipulation, regular expression pattern matching, list operations, dates, randomization, math (ceil, floor, quantize,…), time functions, eval, …

5
5

ClassAd Examples

AP Job Ad

```
Type  = "Job"
Requirements =
    HasMatlabLicense
     == True &&
    Memory >= 1024
Rank = kflops + 1000000 *
 Memory
Cmd= "/bin/sleep"
Args = "3600"
Owner = "gthain"
NumJobStarts = 8
KindOfJob = "simulation"
Department = "Math"
```

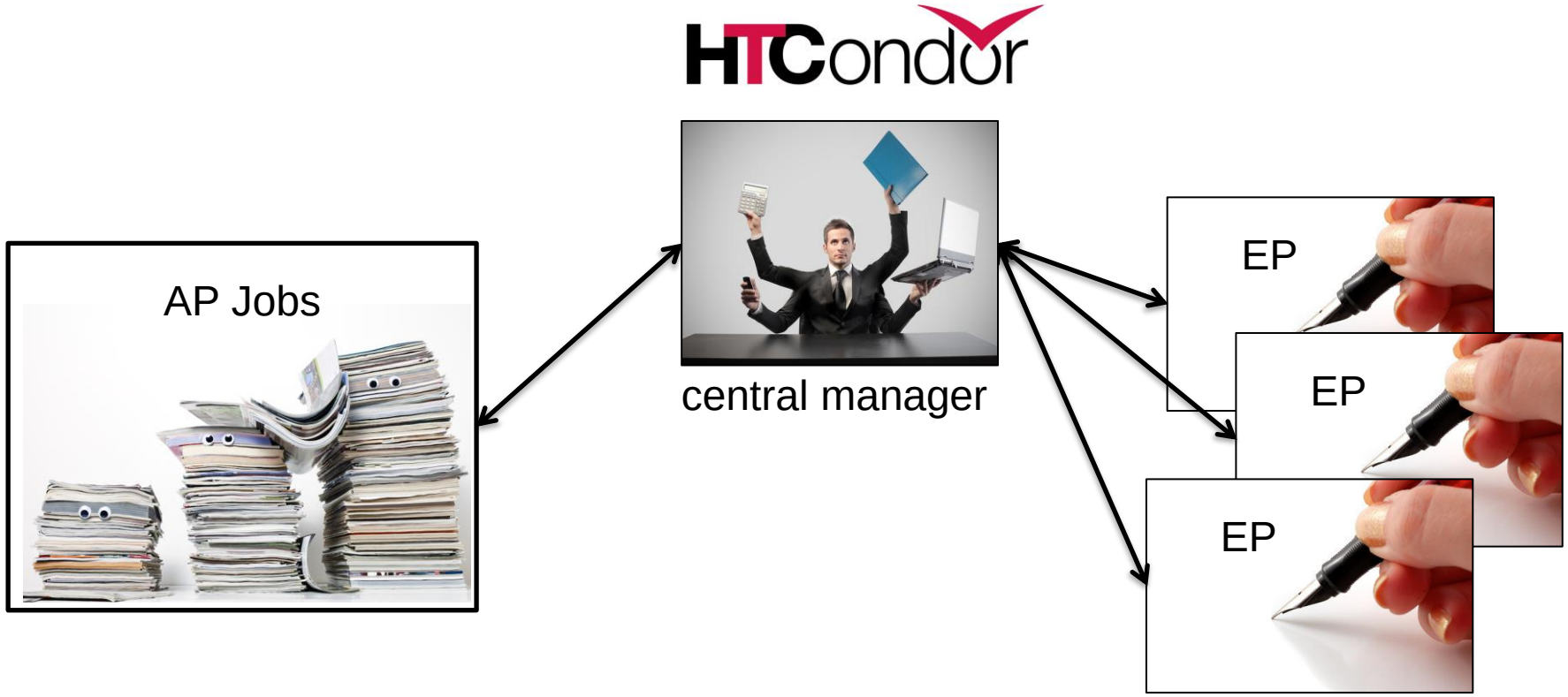
EP Machine Slot Ad

```
Type = "Machine"
Cpus  = 40
Memory = 2048
Requirements =
  (Owner == “gthain”)  ||
  (KindOfJob == “simulation”)
Rank = Department == "Math"
HasMatlabLicense = true
MaxTries = 4
kflops = 41403
```


6

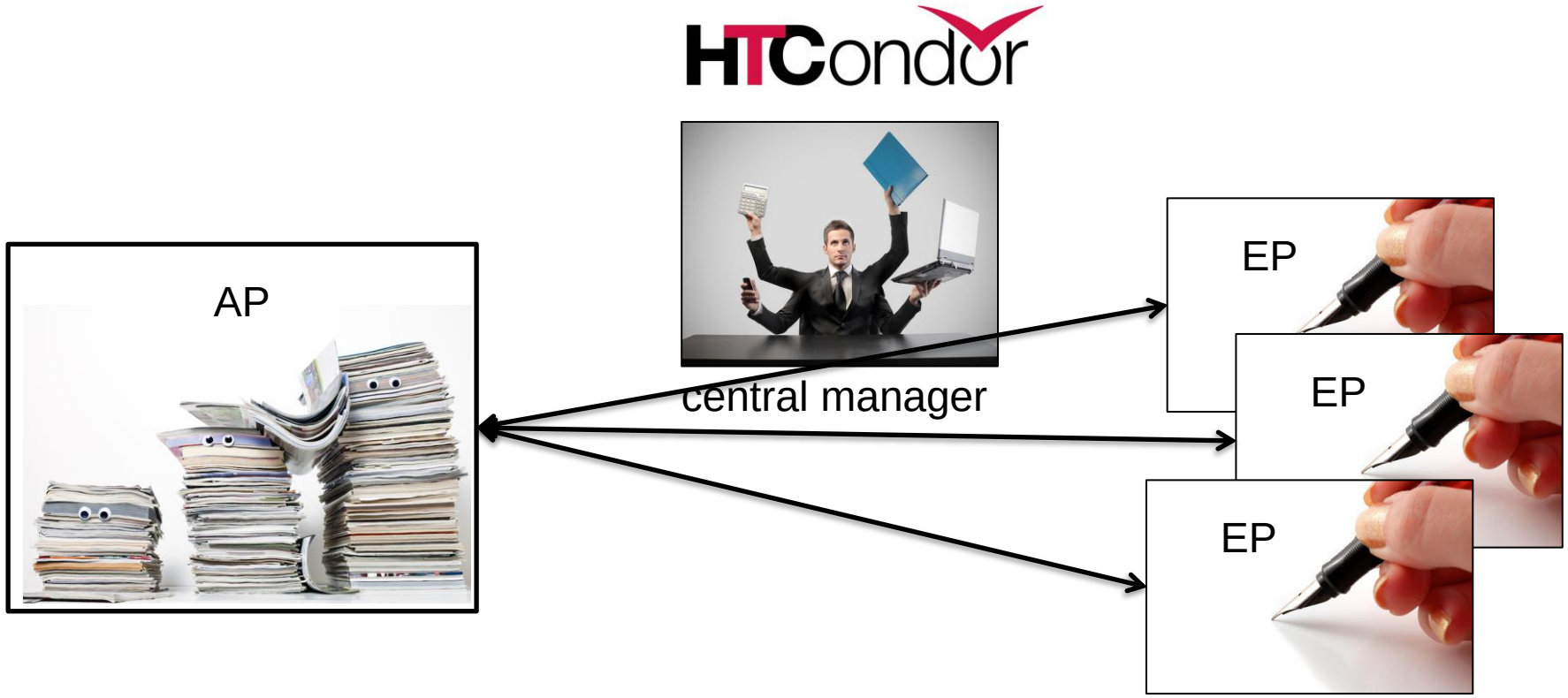
Job Matching

- On a regular basis, the central manager reviews Job resource requests from APs and matches them to EP Slot ads.



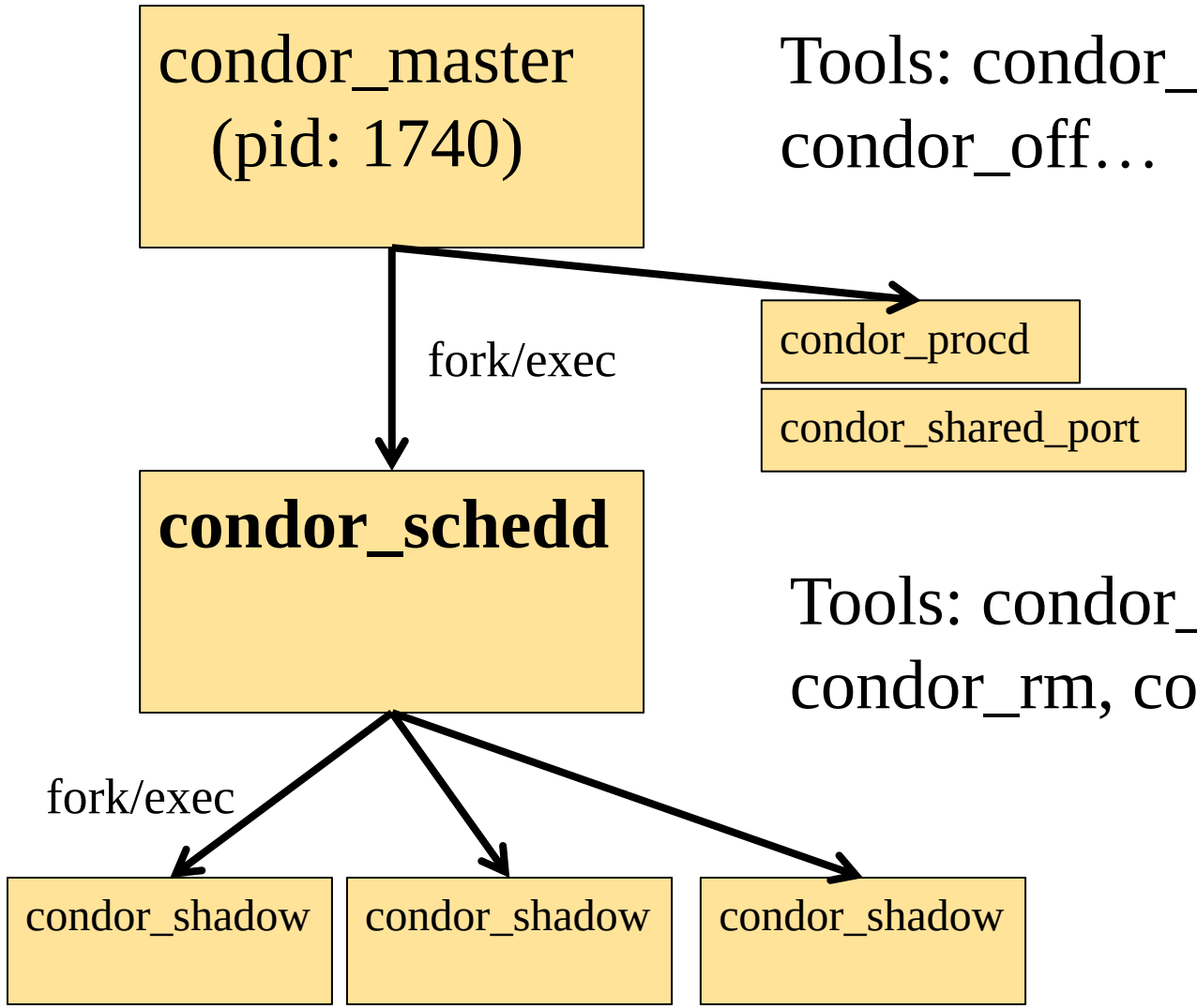
Job Execution

- (Then the AP and EP points communicate directly.)



Architecture & Job Startup

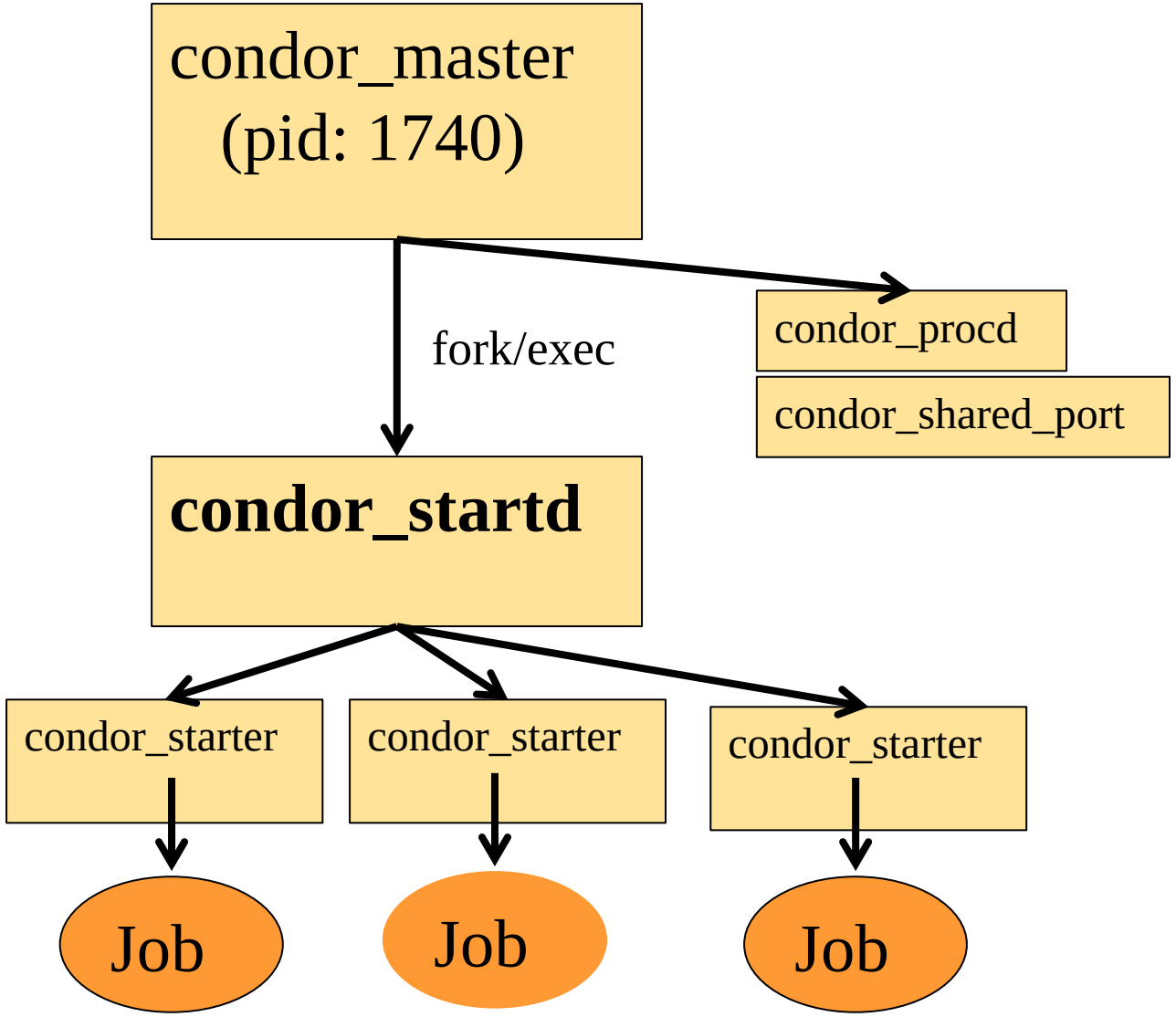
AP Core Process View



Tools: condor_reconfig, condor_on, condor_off…

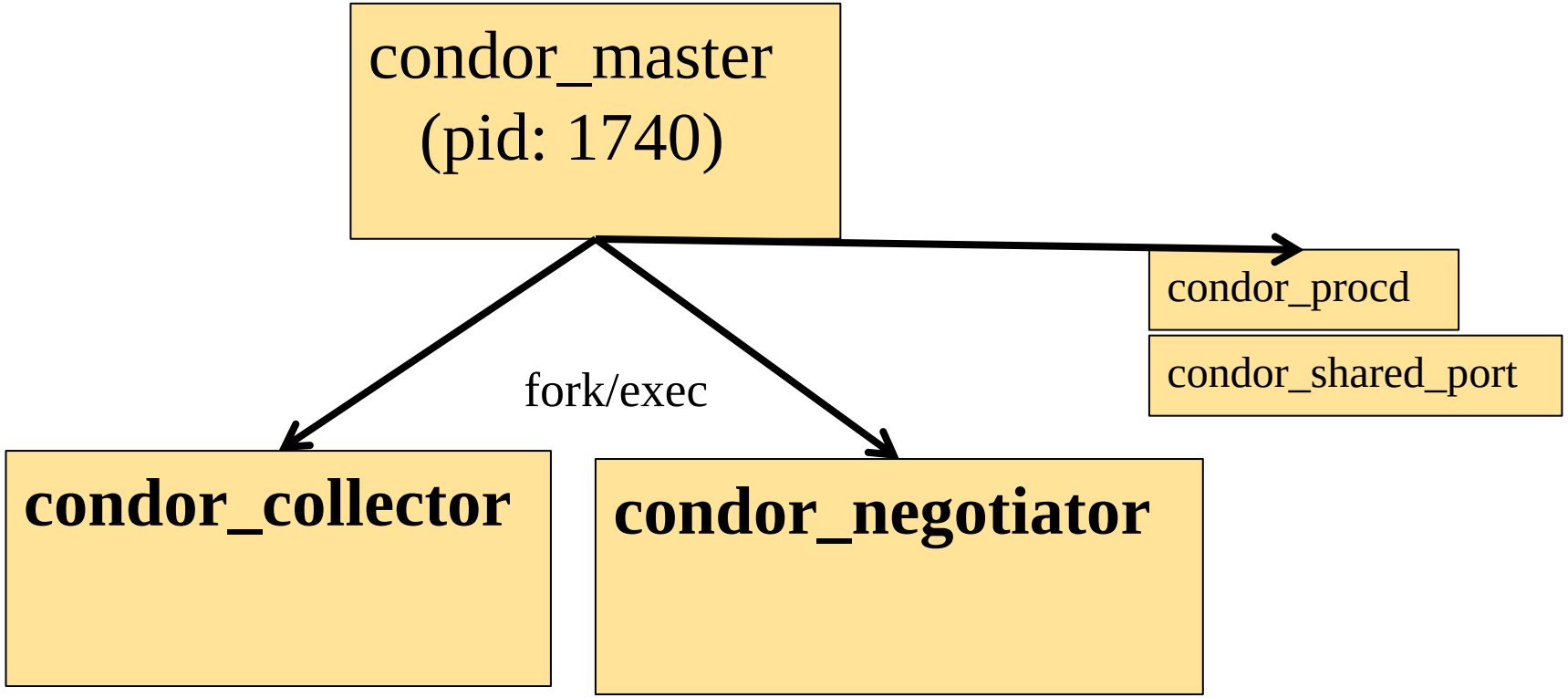
Tools: condor_submit, condor_q, condor_rm, condor_hold, …

EP Core Process View



Tools:
condor_drain,
condor_vacate…

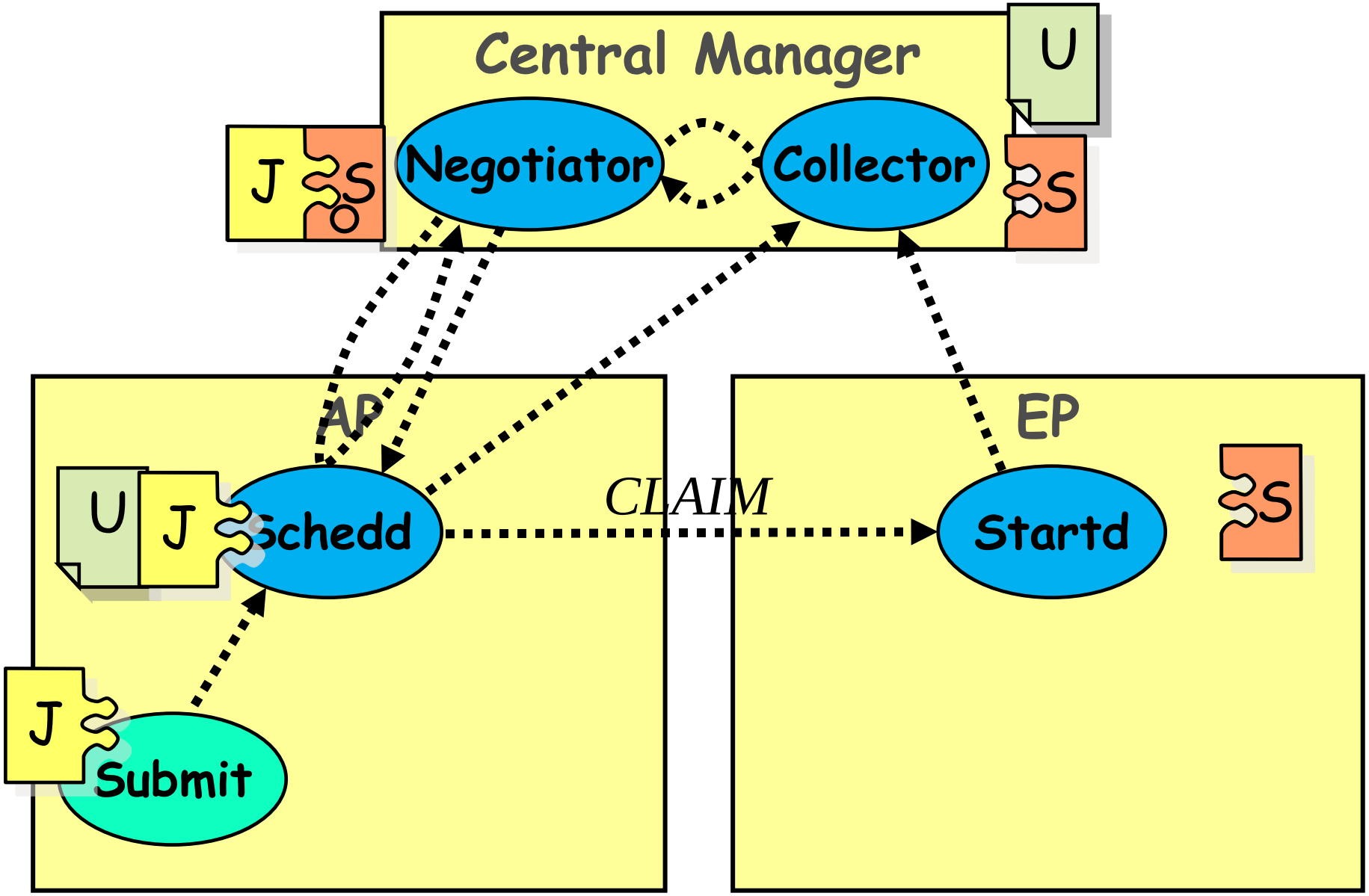
Central Manager Process View



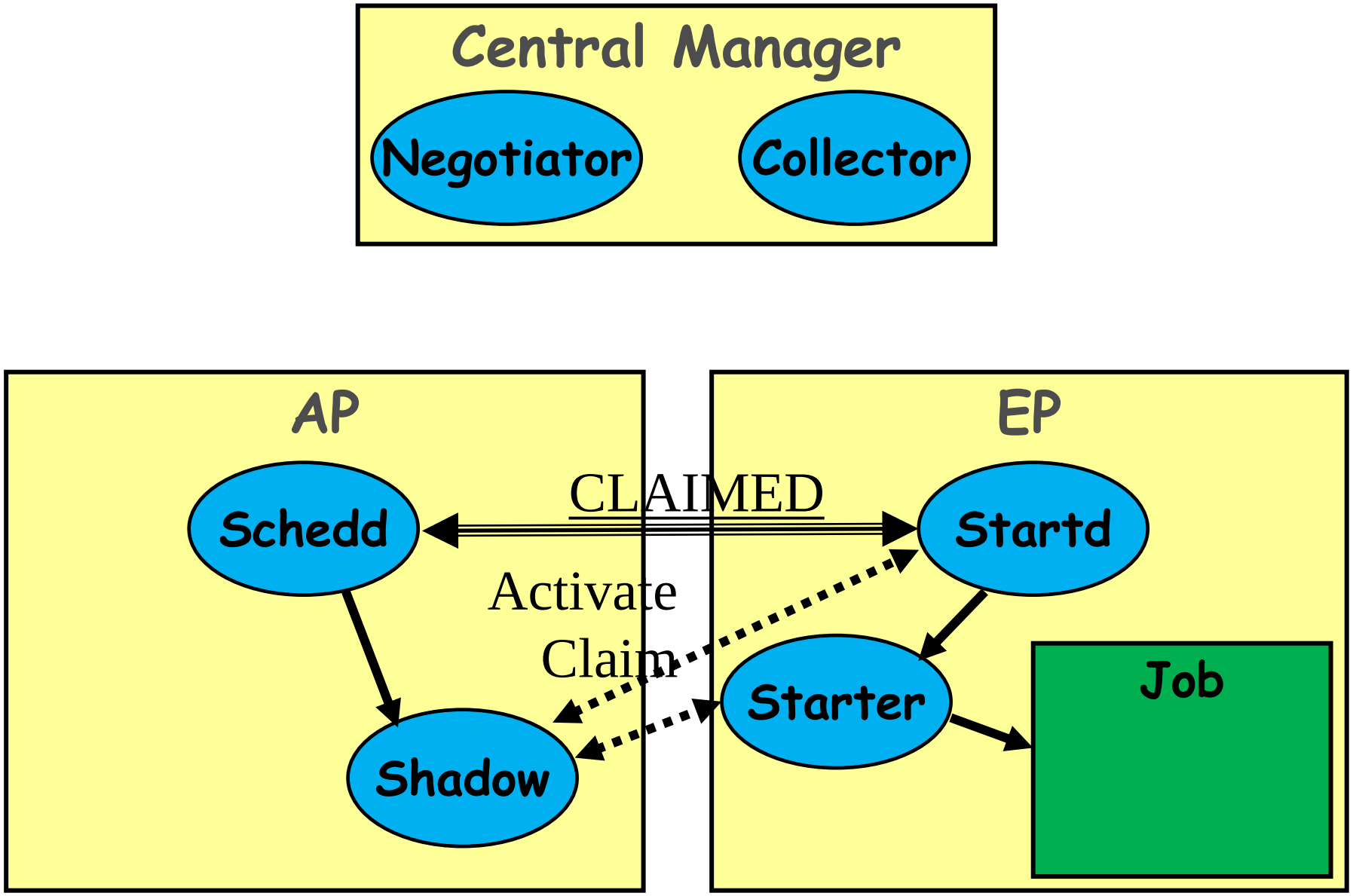
Tools:
condor_status,
…

Tools:
condor_userprio
…

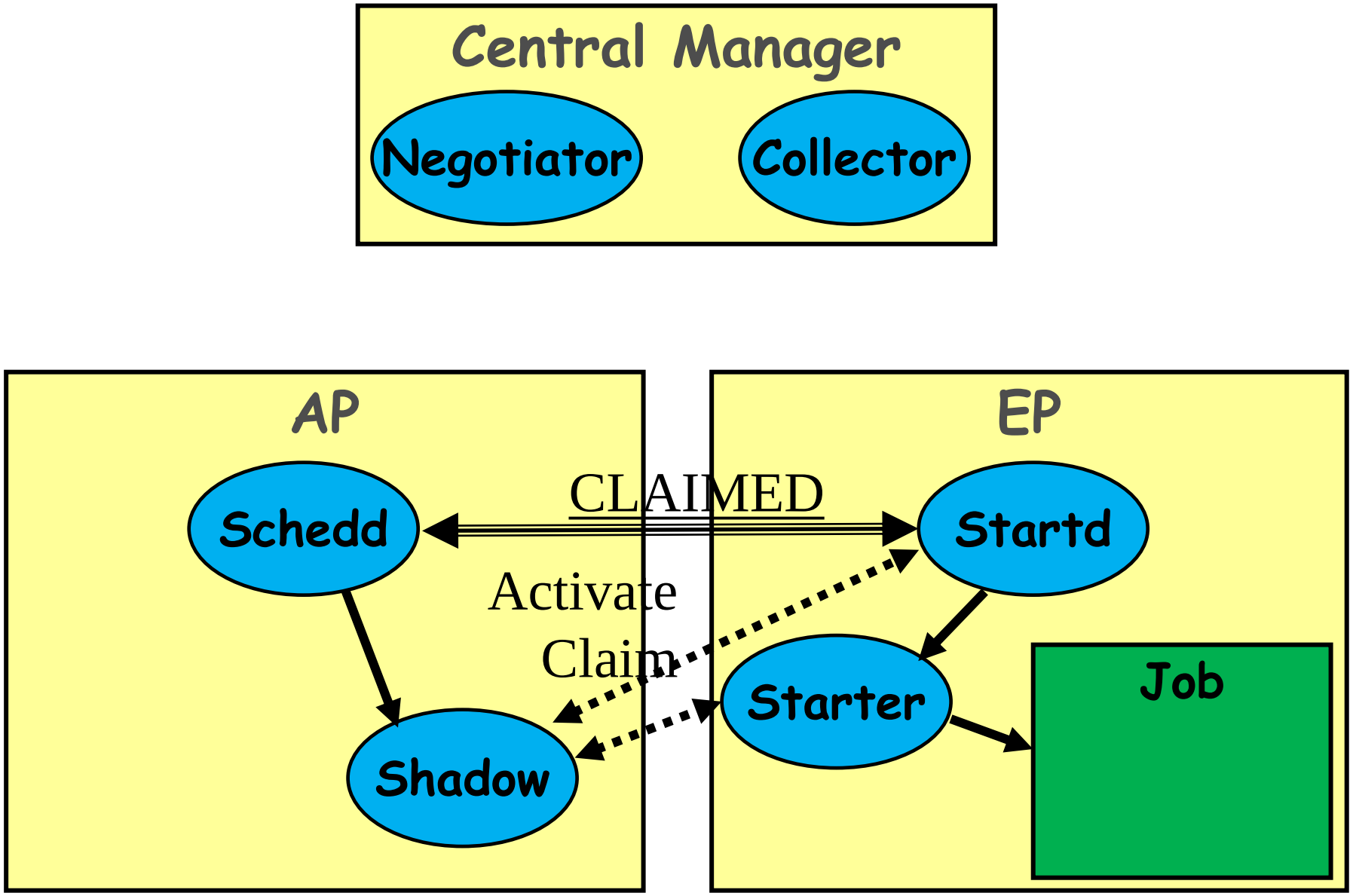
Claiming Protocol



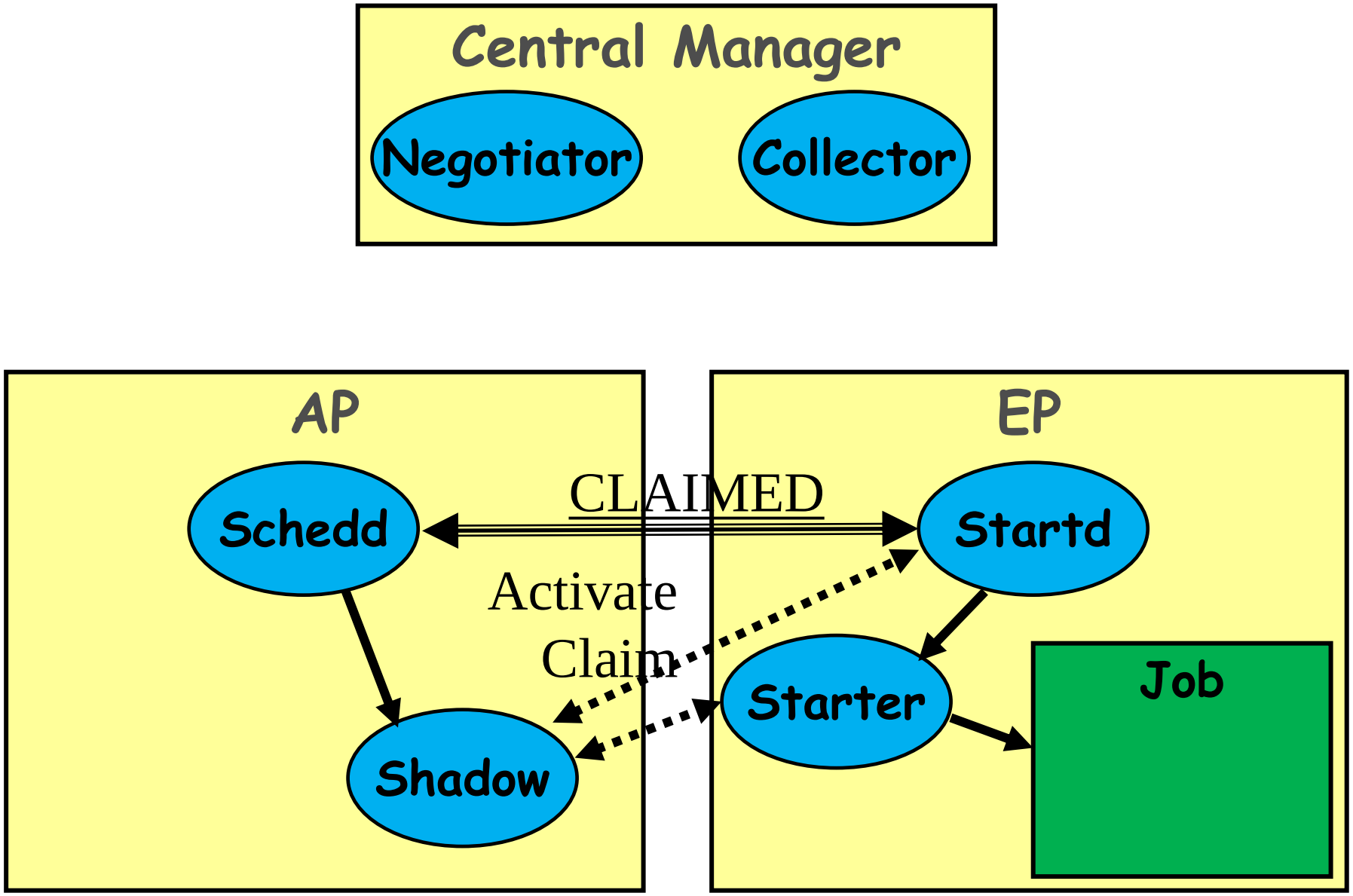
Claim Activation



Repeat until Claim released



Repeat until Claim released



Now the Manual will make more sense!

REFERENCE MANUALS

Users' Manual

Administrators' Manual

Introduction

Starting Up, Shutting Down and Reconfiguring the System

Introduction to Configuration

Configuration Macros

HTCondor-wide Configuration File Entries

Daemon Logging Configuration File Entries

DaemonCore Configuration File Entries

Network-Related Configuration File Entries

Shared File System Configuration File Macros

condor_master Configuration File Macros

condor_startd Configuration File Macros

condor_schedd Configuration File Entries

condor_shadow Configuration File Entries

condor_starter Configuration File Entries

condor_submit Configuration File Entries

Read the Docs v: latest

condor_schedd Configuration File Entries

These macros control the *condor_schedd*.

SHADOW

This macro determines the full path of the *condor_shadow* binary that the *condor_schedd* spawns. It is normally defined in terms of `$(SBIN)`.

START_LOCAL_UNIVERSE

A boolean value that defaults to `TotalLocalJobsRunning < 200`. The *condor_schedd* uses this macro to determine whether to start a **local** universe job. At intervals determined by `SCHEDD_INTERVAL`, the *condor_schedd* daemon evaluates this macro for each idle **local** universe job that it has. For each job, if the `START_LOCAL_UNIVERSE` macro is `True`, then the job's `Requirements` expression is evaluated. If both conditions are met, then the job is allowed to begin execution.

The following example only allows 10 **local** universe jobs to execute concurrently. The attribute TotalLocalJobsRunning is supplied by *condor_schedd* 's ClassAd:

```
START_LOCAL_UNIVERSE = TotalLocalJobsRunning < 10
```

STARTER_LOCAL

The complete path and executable name of the *condor_starter* to run for **local** universe jobs. This variable's value is defined in the initial configuration provided with HTCondor as

```
STARTER_LOCAL = $(SBIN)/condor_starter
```

This variable would only be modified or hand added into the configuration for a pool to be upgraded from one running a version of HTCondor that existed before the **local** universe to one that includes the **local** universe, but without utilizing the newer, provided configuration files.

LOCAL_UNIV_EXECUTE

A string value specifying the execute location for local universe jobs. Each running local universe job will receive a uniquely named subdirectory within this directory. If not specified, it defaults to `$(SPOOL)/local_univ_execute`.

And so will admin tools

```
$ condor_status
Name                OpSys      Arch   State     Activity LoadAv Mem

slot1@chevre.cs.wi LINUX      X86_64 Unclaimed Idle      0.190 20480
slot2@chevre.cs.wi LINUX      X86_64 Claimed   Busy      0.992 20480
slot3@chevre.cs.wi LINUX      X86_64 Claimed   Busy      1.000 20480
slot4@chevre.cs.wi LINUX      X86_64 Unclaimed Idle      0.000 20480
```

Condor Installation Basics

http://htcondor.org/htcondor/download

HTCondor Download

htcondor.org/htcondor/download/

HTCondor Software Suite

Software | Help & Support | Community | About | Contact Us

- Overview
- Download
- Documentation
- Release Highlights
- Release Plan
- News
- Full Release Notes

HTCondor Download

23.9.6 Feature Release

August 8, 2024

- Add config knob to not have cgroups count kernel memory for jobs on EL9
- Remove support for numeric unit suffixes (k,M,G) in ClassAd expressions
- In submit files, request_disk & request_memory still accept unit suffixes
- Hide GPUs not allocated to the job on cgroup v2 systems such as EL9
- DAGMan can now produce credentials when using direct submission
- Singularity jobs have a contained home directory when file transfer is on
- Avoid using IPv6 link local addresses when resolving hostname to IP addr
- New 'htcondor credential' command to aid in debugging

Feature releases distribute HTCondor's new features and also incorporates bug fixes. Most people should use choose this channel and stay up-to-date with HTCondor's latest features.

Consultation of the Detailed Notes recommended prior to upgrading.

Installation Guide | Release Notes

Find or add yourself to the global network of HTCondor users!

HTCSS User Map

Version Number Scheme

- Major.minor.release
  - Represents Year (we hope)
- Major.minor.release
  - If minor is zero (a.b.c): Stable series
    - Very stable, mostly bug fixes
    - Current: 23.0.12
    - Examples: 23.0.1,
      - 24.0.1 coming soon to a repo near you
  - If minor is non-zero (a.b.c): Developer series
    - New features, may have some bugs
    - Current: 23.10.1

The Guarantee

- All minor releases in a stable series interoperate
  - E.g. can have pool with 23.0.1, 23.0.9, etc.
  - But not WITHIN A MACHINE:
    - Only across machines
- The Reality
  - We work really hard to do better
    - 23.9 with 24.0, etc.
    - Part of HTC ideal: can never upgrade in lock-step
    - We document when we break this.

Let’s Install "minicondor"

- Either with tarball (usually for non-root installs)
- Or native packages (for Linux as root)

```
curl -fsSL https://get.htcondor.org |
sudo GET_HTCONDOR_PASSWORD="myPassword" /bin/bash
-s -- --no-dry-run --minicondor cm.toddt.org
```

Or for Docker people:

```
docker run -it htcondor/mini bash
```

3 Separate machines

- Central Manager
- Execute Machine
- Submit Machine

Central Manager

```
curl -fsSL https://get.htcondor.org |
sudo GET_HTCONDOR_PASSWORD="myPassword"
/bin/bash -s -- --no-dry-run -central-manager
cm.toddt.org
```

Access Point

```
curl -fsSL https://get.htcondor.org |
sudo GET_HTCONDOR_PASSWORD="myPassword"
/bin/bash -s -- --no-dry-run --submit
cm.toddt.org
```

Execution Point

```
curl -fsSL https://get.htcondor.org |
sudo GET_HTCONDOR_PASSWORD="yourPassword"
/bin/bash -s -- --no-dry-run --execute
cm.toddt.org
```

Voila! Pool Running

- But how is it configured by default?
  Config files go into /etc/condor/config.d

condor_config_val –dump (show all config knobs!)

condor_config_val –summary (show only customized knobs)

- Q: Who can submit jobs? A: Anyone who can login to AP
- Q: Who can use administrative commands? A: root
- Q: Scheduling policy? A: Fair-share across users
- Q: What UID is used to run the jobs?

What UID should jobs run as?

- Three Options (all require root):
  - Nobody UID
    - Safest from the machine's perspective
  - The submitting User
    - Most useful from the user's perspective
    - May be required if shared filesystem exists
  - A "Slot User"
    - Bespoke UID per slot
    - Good combination of isolation and utility

UID_DOMAIN SETTINGS

```
UID_DOMAIN = same_string_on_submit
TRUST_UID_DOMAIN = true
SOFT_UID_DOMAIN = true
```

If UID_DOMAINs match, jobs run as user, otherwise "nobody"

Slot User

```
SLOT1_USER = slot1
SLOT2_USER = slot2
…
STARTER_ALOW_RUNAS_OWNER = false
EXECUTE_LOGIN_IS_DEDICATED=true
```

Job will run as slotX Unix user

FILESYSTEM_DOMAIN

- HTCondor can work with NFS
  - But how does it know what nodes have it?
- WhenSubmitter & Execute nodes share
  - `FILESYSTEM_DOMAIN` values
    - e.g `FILESYSTEM_DOMAIN = domain.name`
- Or, submit file can always transfer with
  - `should_transfer_files = yes`
- If jobs always idle, first thing to check

I want my custom the scheduling policy

- First write your policy "in English"
- Then use the power of ClassAds to express it!
  - ClassAd expressions at AP control which jobs are allowed, what attributes must be specified, …
  - ClassAd expressions at the EP control when/who can use the EP, when to evict jobs, when to run the job inside a container, …
  - ClassAd expressions at the Central Manager control which ads are accepted into the pool, best-fit/first-fit policy, …
- Need help? You can ask on htcondor-users email list

Thank you!

Questions?

Join us on the htcondor-users email list!

https://htcondor.org/mail-lists/#user

This work is supported by NSF under Cooperative Agreement OAC-2030508 as part of the PATh Project. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the NSF

PATh PARTNERSHIP to ADVANCE THROUGHPUT COMPUTING