IceCube's SkyDriver

An Application of the Event Workflow Management System for Scalable Solutions of Distributed Workflows

Ric Evans

Research Software Engineer

UW-Madison
IceCube / WIPAC

Throughput Computing 2024 – HTC 24

ICECUBE
SOUTH POLE NEUTRINO OBSERVATORY

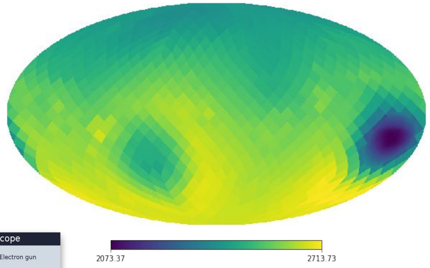
The Event Workflow Management System (EWMS) Question

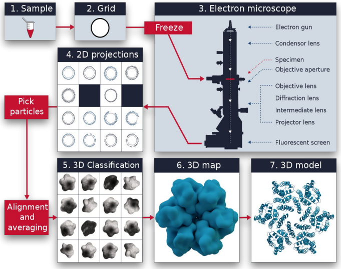
*How can we take a workload,
consisting of millions or billions of tasks,
and group it into tens of thousands of jobs?*

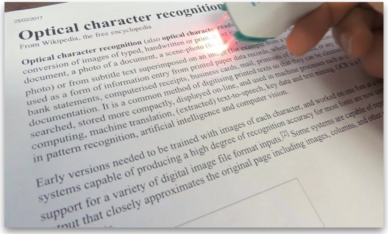
Event-Granular HTC Workflows

To be most efficient, we want to subdivide a workflow into **“smallest” unit of work (“events”)**

- ➢ Multi-Messenger Astrophysics events (IceCube and LIGO triggers)
- ➢ Astronomical observations (images)
- ➢ Cryogenic electron microscopy (cryo-EM) data
- ➢ Optical Character Recognition on pages in a book
- ➢ and more!






28/02/2017

Optical character recognition

From Wikipedia, the free encyclopedia

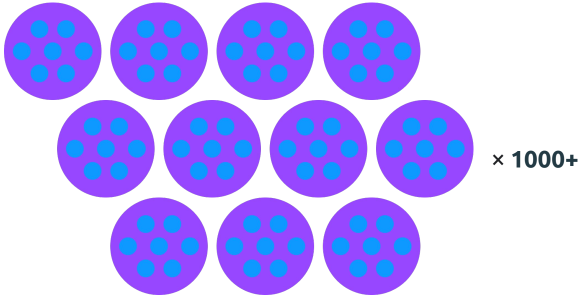
Optical character recognition (also **optical charact[illegible] reade**
conversion of images of typed, handwritten or print[illegible]ext in
document, a photo of a document, a scene-photo ([illegible]
photo) or from subtitle text superimposed on an image (for example from a t
used as a form of information entry from printed paper data records, wheth
bank statements, computerised receipts, business cards, mail, printouts of s[illegible] data, or any
documentation. It is a common method of digitising printed texts so that they can be electron
searched, stored more compactly, displayed on-line, and used in machine processes such as c
computing, machine translation, (extracted) text-to-speech, key data and text mining. OCR is a f
in pattern recognition, artificial intelligence and computer vision.

Early versions needed to be trained with images of each character, and worked on one font at a ti
systems capable of producing a high degree of recognition accuracy for most fonts are now com
support for a variety of digital image file format inputs.[2] Some systems are capable of repr
tput that closely approximates the original page including images, columns, and other

HTCondor’s Traditional Use

HTCondor is great at aggregating distributed resources and orchestrating workflows, but…

- ➢ Imposes **1:1 job-task** mapping
- ➢ Needs O(>30 min) jobs to be most efficient
  - ○ **Task lifetime >> Startup+Scheduling time**



size = runtime

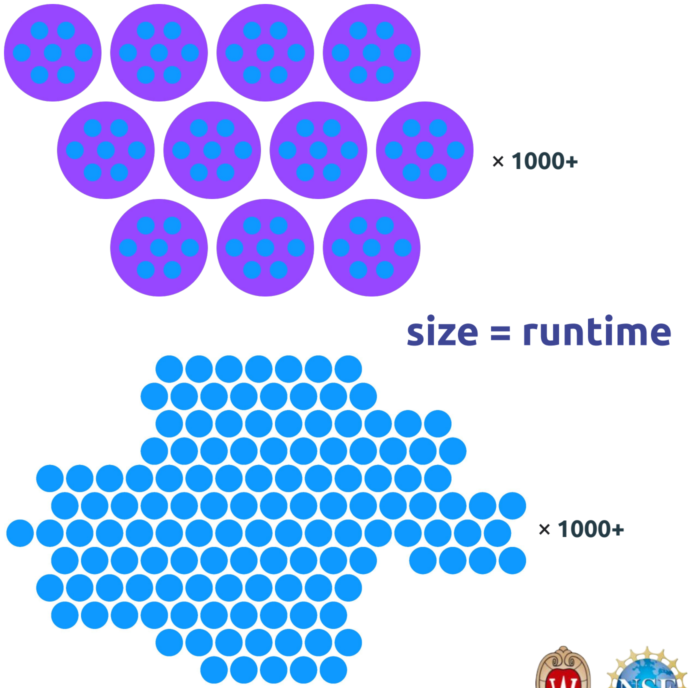
HTCondor’s Traditional Use

HTCondor is great at aggregating distributed resources and orchestrating workflows, but…

- ➢ Imposes **1:1 job-task** mapping
- ➢ Needs O(>30 min) jobs to be most efficient
  - ○ **Task lifetime >> Startup+Scheduling time**

If we want to work on *events*

- ➢ Much **shorter runtime per task**
- ➢ **1:N job-task** mapping
- ➢ **Dynamic allocation** of inputs and outputs

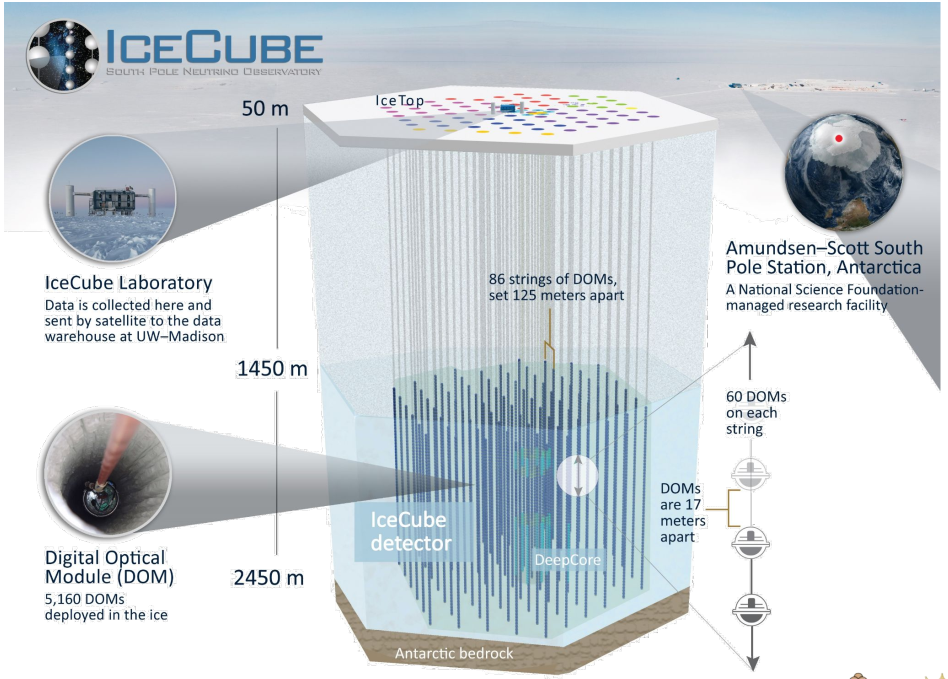


IceCube Neutrino Observatory

Why does IceCube need many, many short-lived tasks?

IceCube Neutrino Observatory

The IceCube Neutrino Observatory is a cubic kilometer neutrino telescope located at the geographic South Pole premier facility for detecting neutrinos > 10 GeV, particularly > 1 TeV astrophysical neutrinos.



IceCube Laboratory
Data is collected here and sent by satellite to the data warehouse at UW–Madison

Digital Optical Module (DOM)
5,160 DOMs deployed in the ice

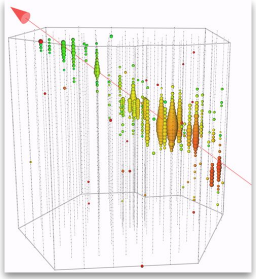
Amundsen–Scott South Pole Station, Antarctica
A National Science Foundation-managed research facility

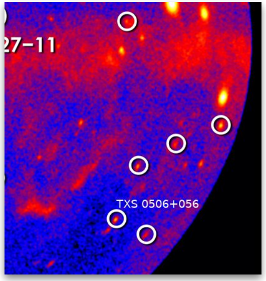
The Situation

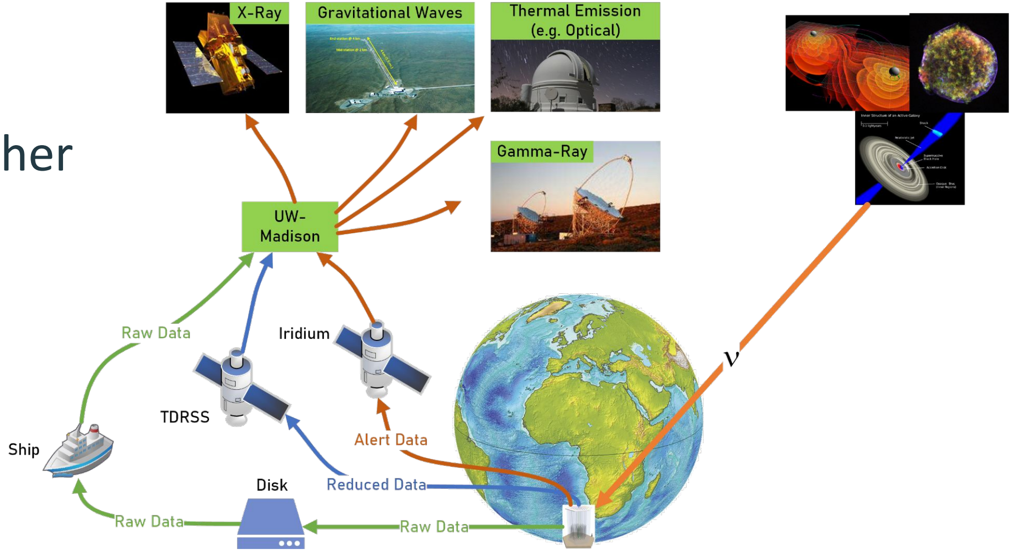
A neutrino is detected by IceCube!

Where did it come from?

Where do we need to point other telescopes for **immediate** follow-up observations?

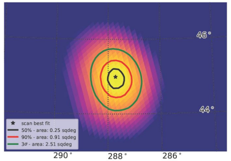




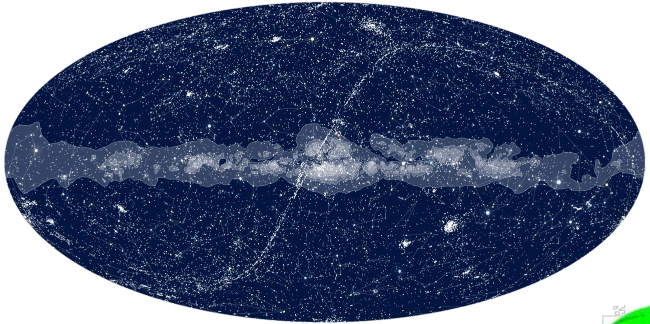


The Problem

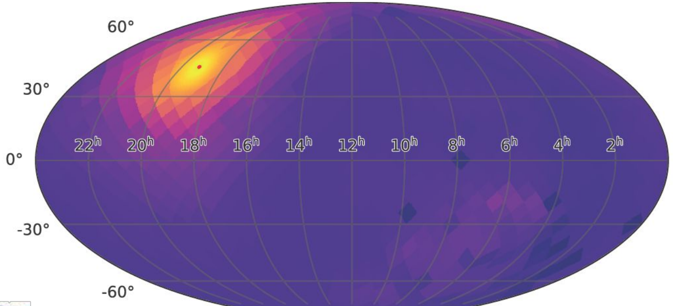
We need to reconstruct a *Sky Map*

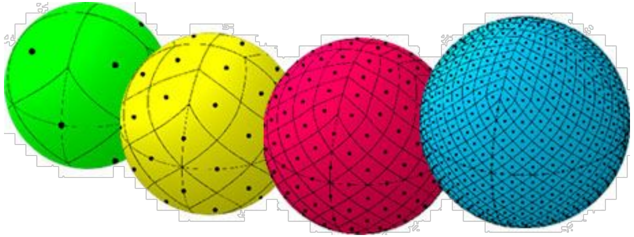


Most accurate and detailed directional reconstruction comes by scanning across the sky in varying granularity: O(100k) pixels



“night sky”



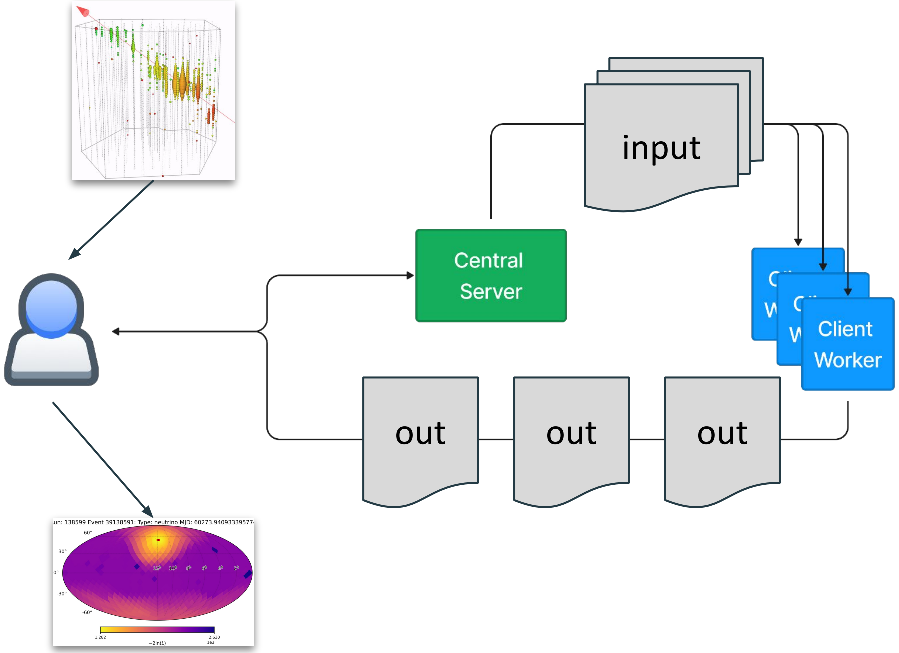


HEALPix algorithm

The Solution

The (Original) Skymap Scanner

1. **Preempt** *N* HTCondor nodes for **immediate availability**
2. **Generate** O(100k) **events** (5-tuples)
3. **Group** O(1k) events into ***N* "input" object**
   - ➢ 1 job gets 1 object, O(1k) events
4. **Submit** to HTCondor for ***N* jobs**
5. **Wait** for every job to finish while collecting *N* transferred output objects
6. **Assemble** resulting skymap
   - ➢ Produce the most **probable direction** and error



The Problem with the Solution

Three False Assumptions

Three False Assumptions

- **We know how to group input events because we have a homogeneous infinitely big compute pool.**

We have a heterogeneous and finite pool

🙁

The Problem with the Solution

Three False Assumptions

- **We know how to group input events because we have a homogeneous infinitely big compute pool.**

  We have a heterogeneous and finite pool

- **Task processes will never fail.**

  CPU crashes happen. *What if last event fails?*

😟

The Problem with the Solution

Three False Assumptions

- **We know how to group input events because we have a homogeneous infinitely big compute pool.**

  We have a heterogeneous and finite pool

- **Task processes will never fail.**

  CPU crashes happen. *What if last event fails?*

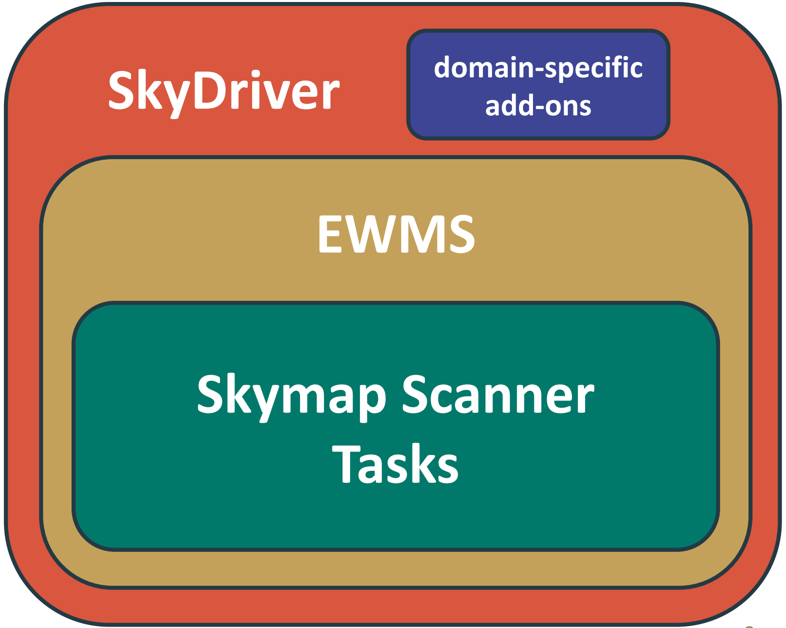
- **No one will be mad if we take away their computing resources.**

  Yes they will, especially before a conference

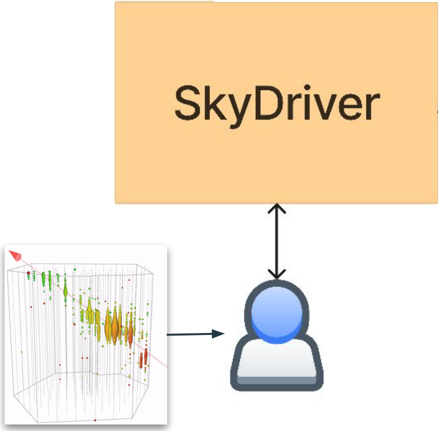
🫠

EWMS Design + SkyDriver Application

- ➢ Design a generalized design, the **Event Workflow Management System**
- ➢ Make an instance of EWMS at IceCube, called **SkyDriver** (with a few domain-specific add-ons)
- ➢ Run **Skymap Scanner tasks** within the SkyDriver service

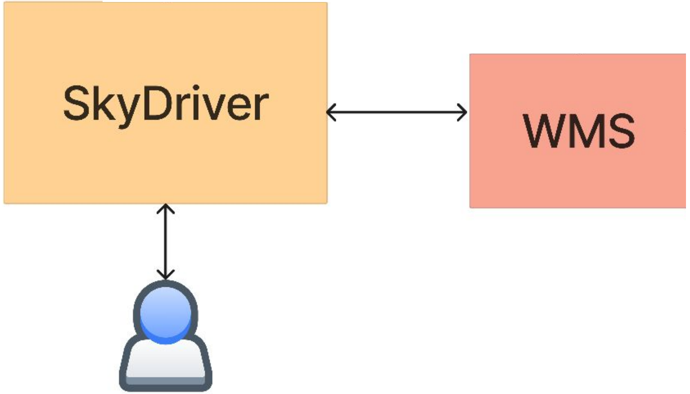


SkyDriver-EWMS Architecture



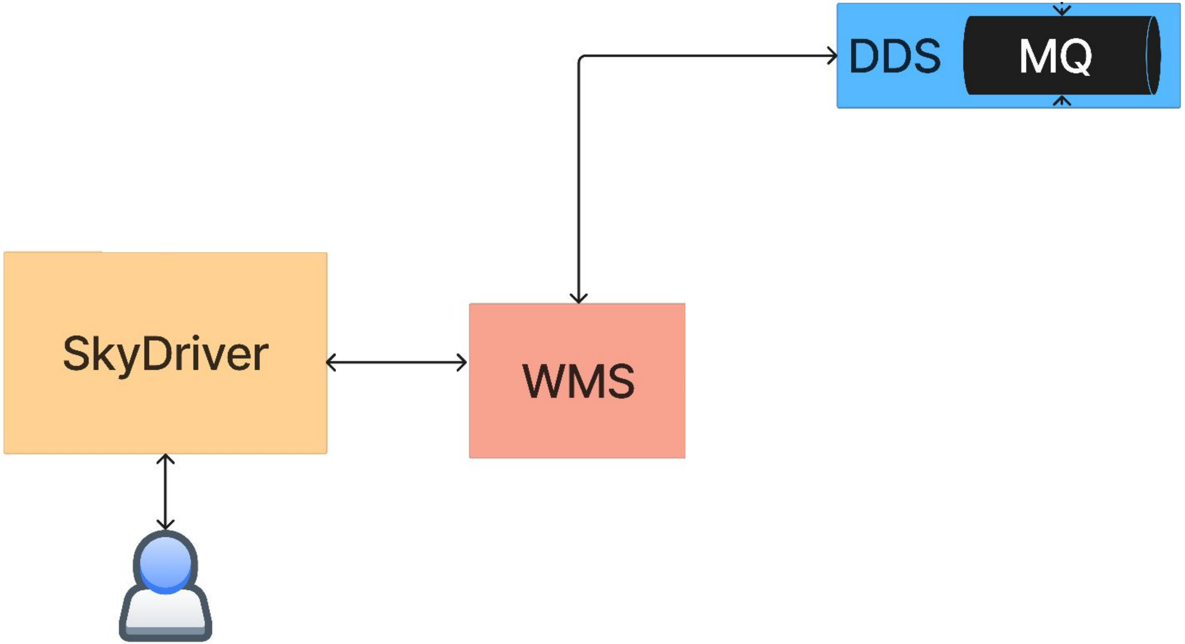
User requests a new scan

SkyDriver-EWMS Architecture



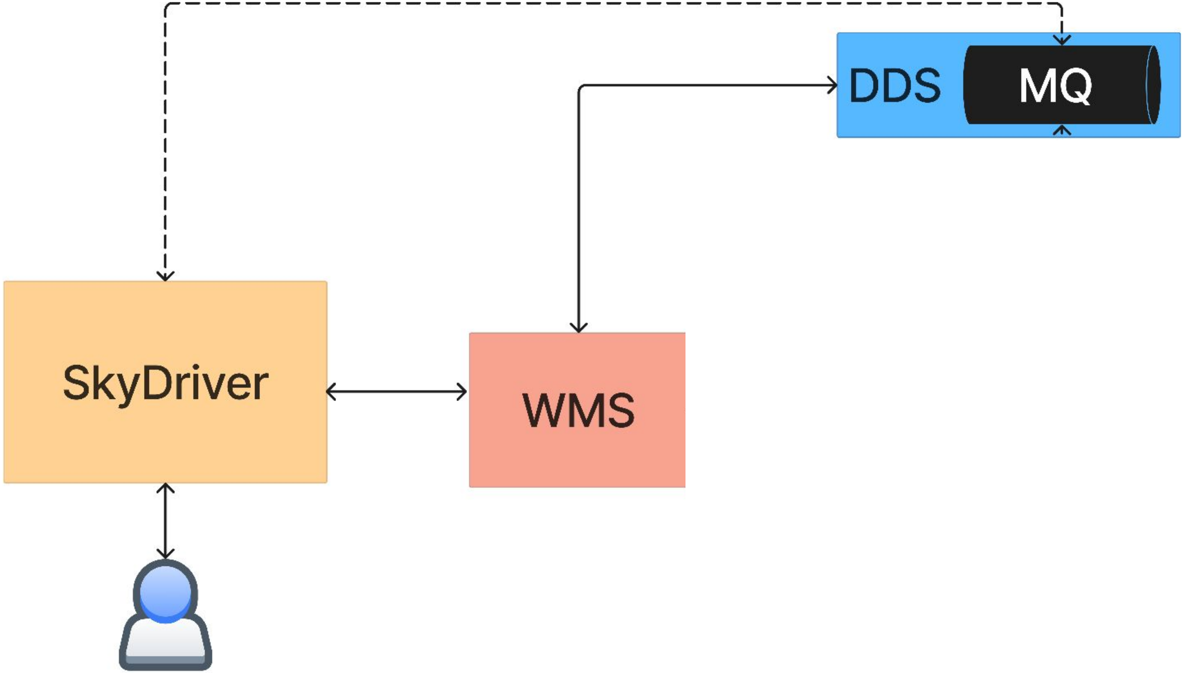
Workflow Management Service

SkyDriver-EWMS Architecture



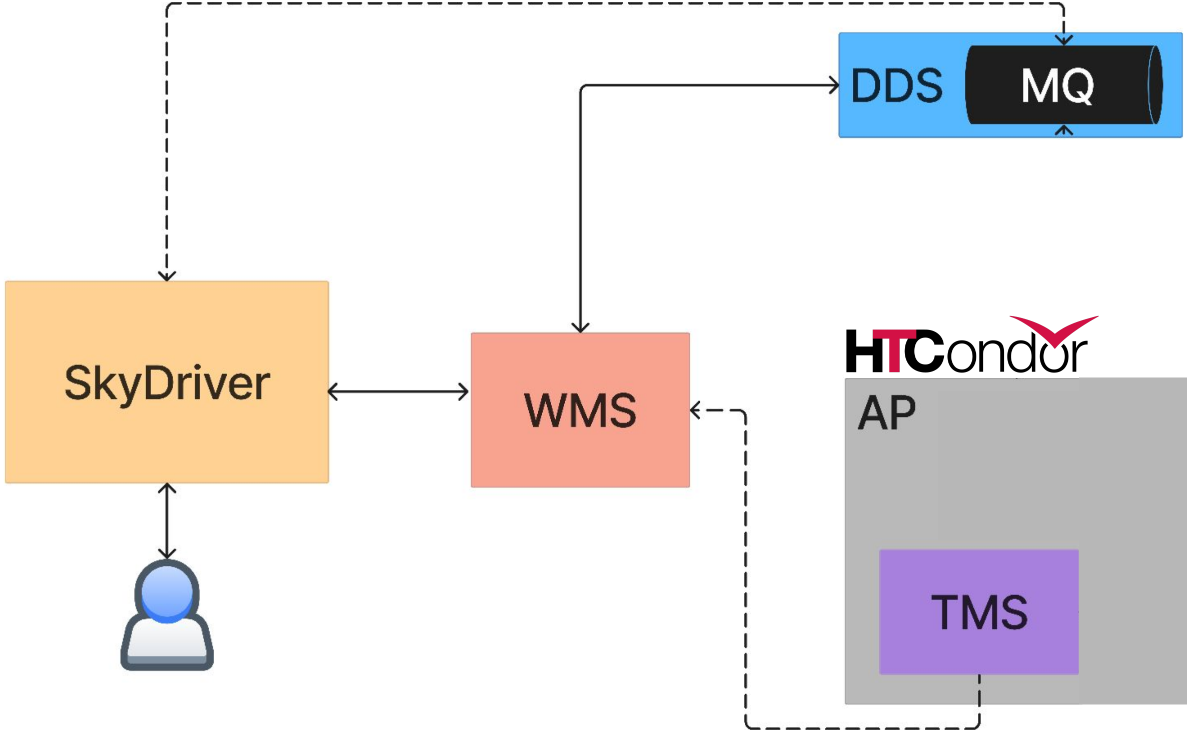
Data Distribution Service & Message Queue Broker

SkyDriver-EWMS Architecture



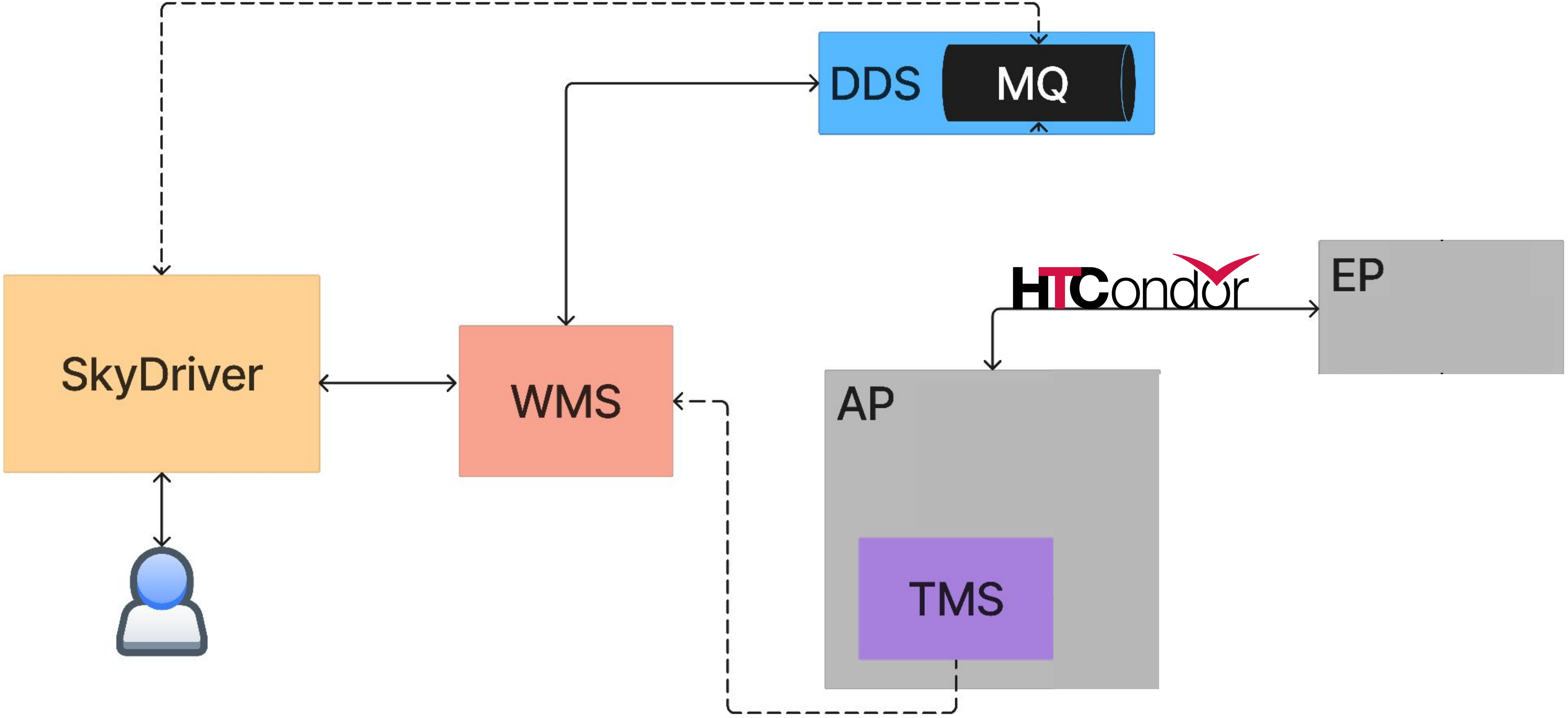
SkyDriver sends events to MQ

SkyDriver-EWMS Architecture



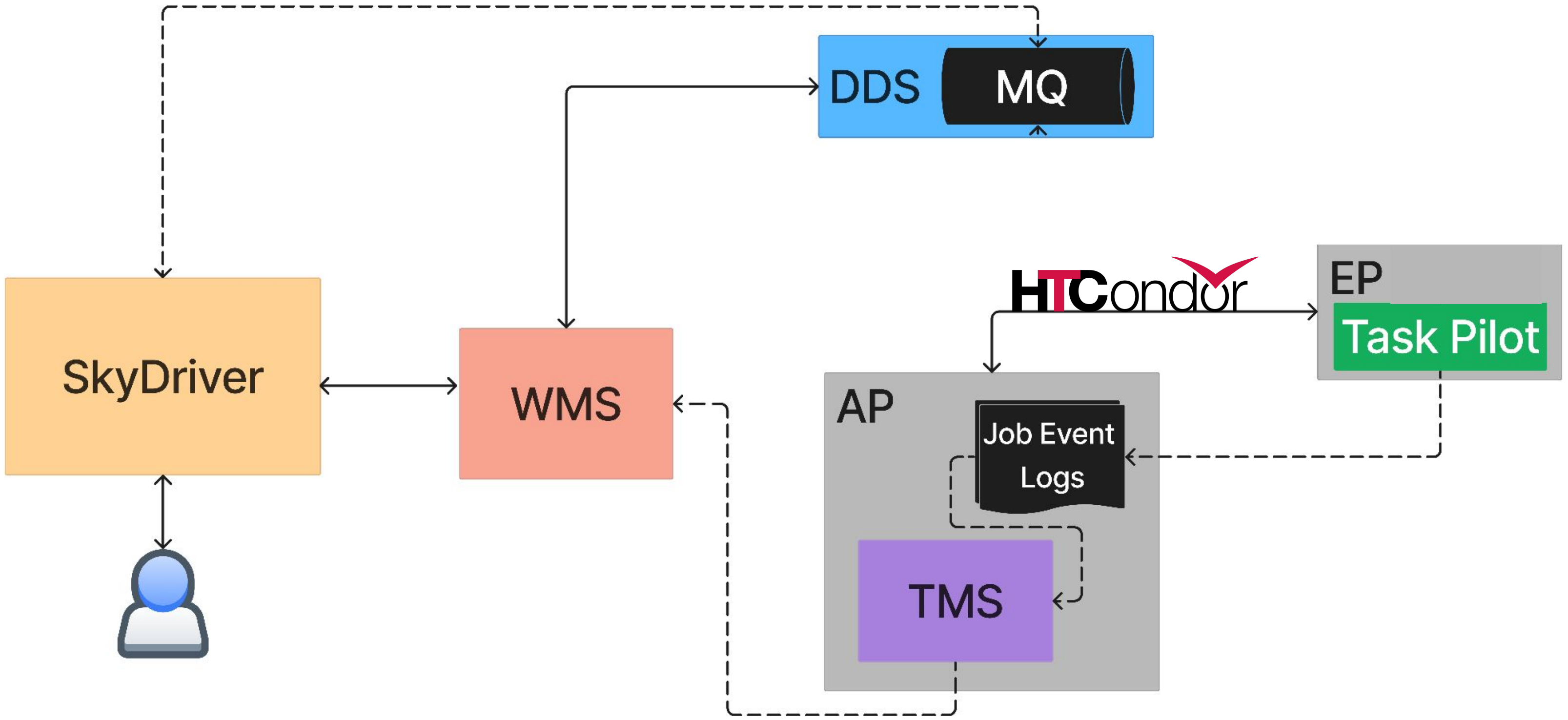
Task Management Service on HTCondor Access Point

SkyDriver-EWMS Architecture



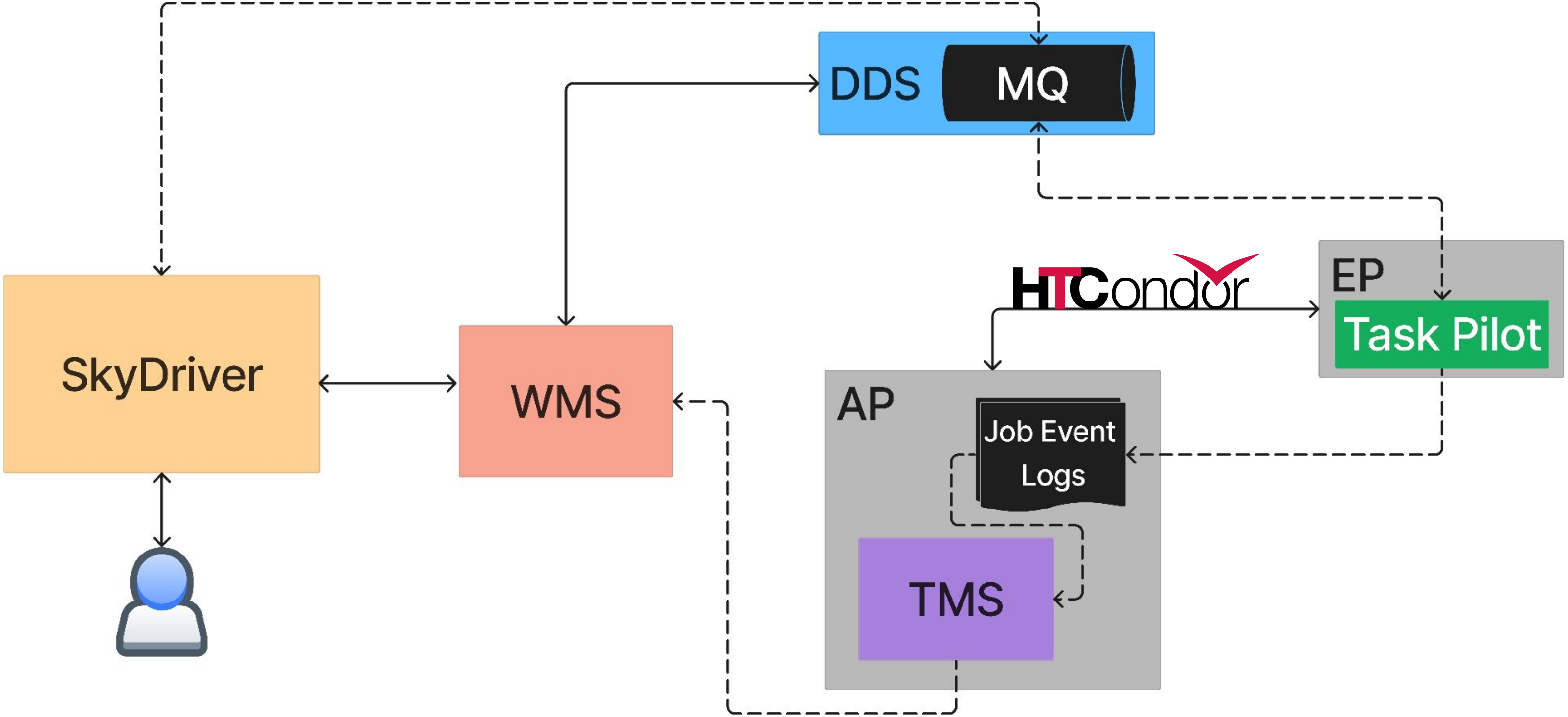
HTCondor starts up jobs

SkyDriver-EWMS Architecture



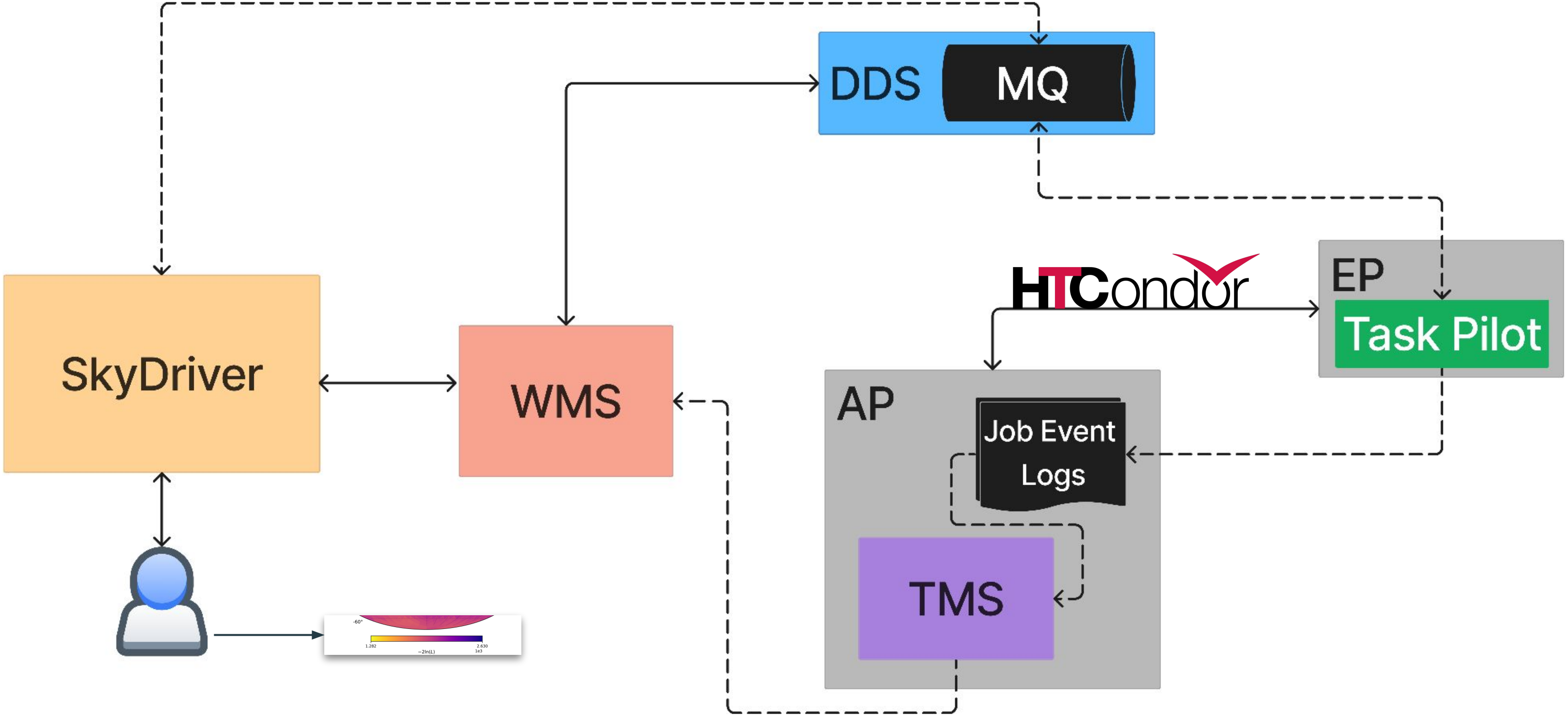
Task Pilot (Worker) on HTCondor Execution Point

SkyDriver-EWMS Architecture



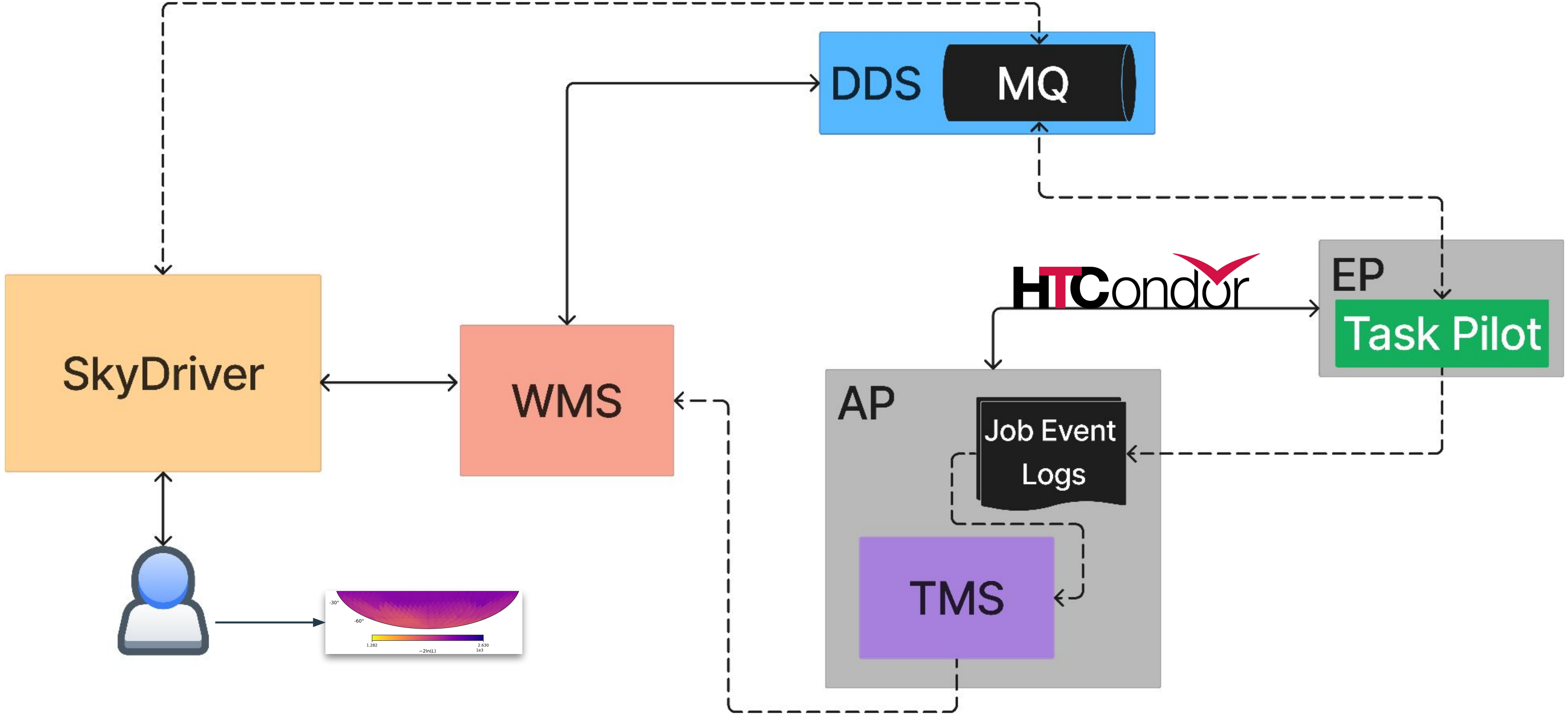
Workers retrieve input-events & send output-events via the MQ

SkyDriver-EWMS Architecture



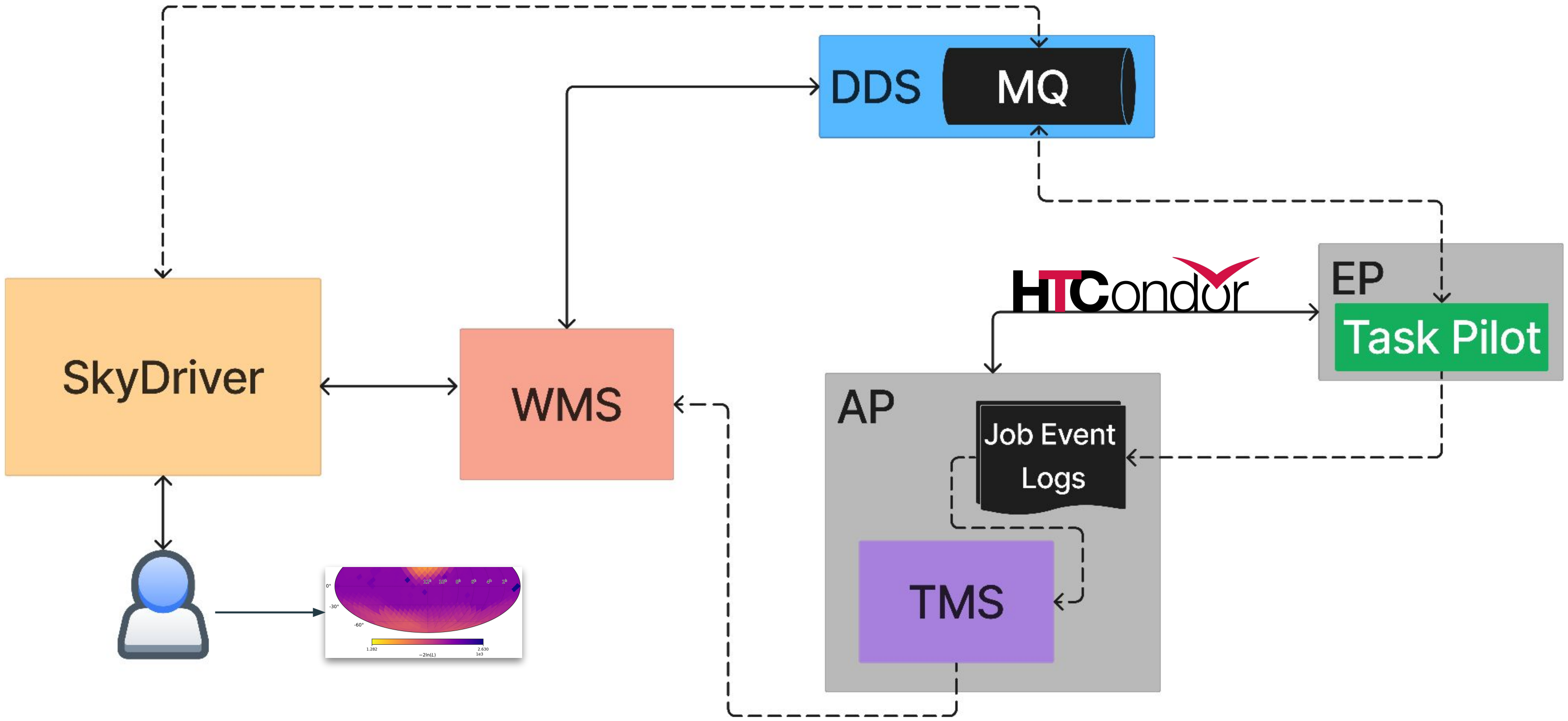
SkyDriver receives output events from MQ

SkyDriver-EWMS Architecture



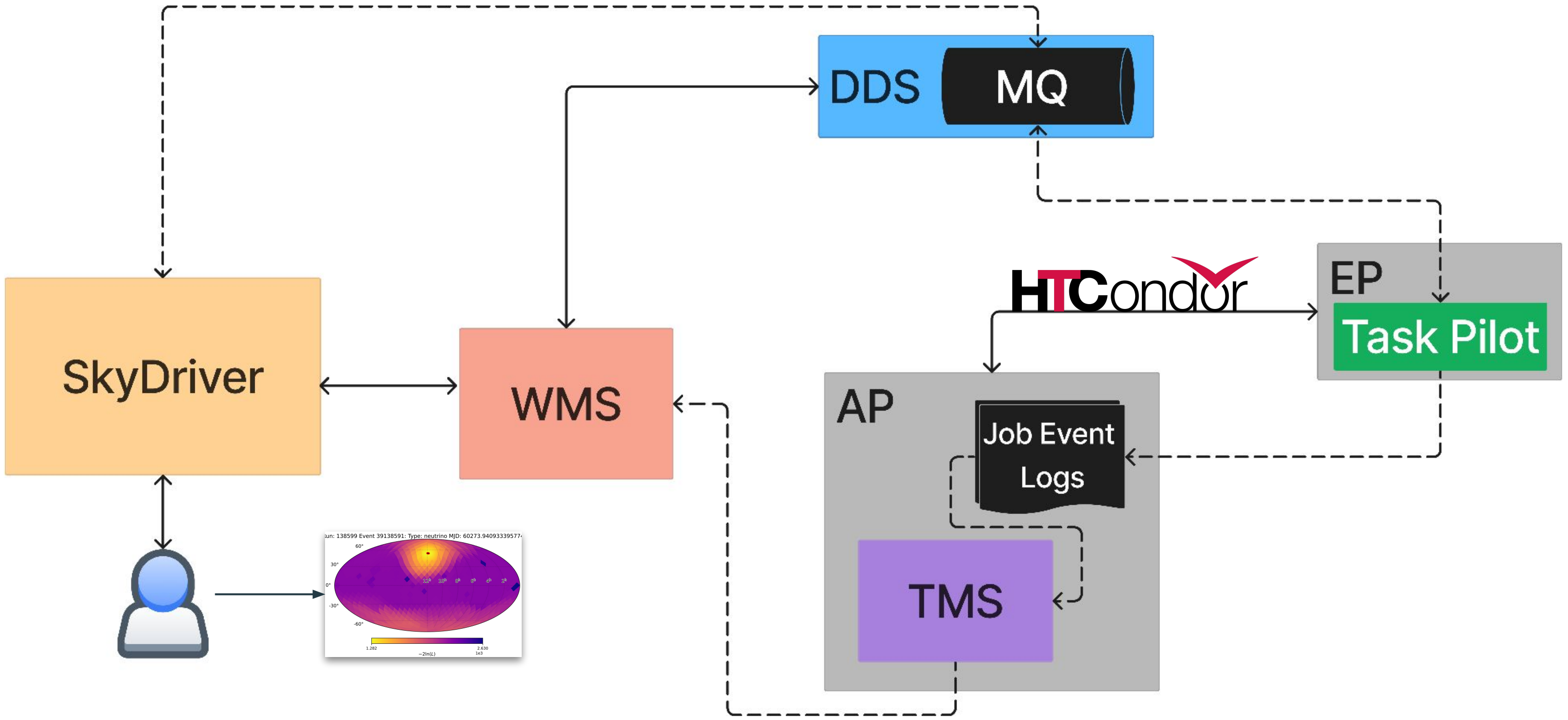
SkyDriver receives output events from MQ

SkyDriver-EWMS Architecture



SkyDriver receives output events from MQ

SkyDriver-EWMS Architecture



SkyDriver receives output events from MQ

Motivation & Goals

How can we help HTCondor support multiple events per job?

1:N job-task pattern

What does EWMS need to do?

➢ **Complement HTCondor's Capabilities**

Thrive in heterogeneous, dynamic environments
(faster CPUs do more work, etc.)

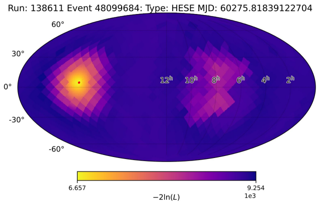
What does EWMS need to do?

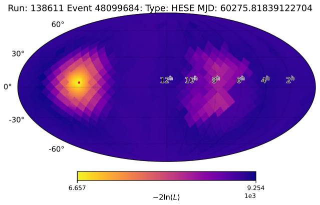
- **Complement HTCondor's Capabilities**

  Thrive in heterogeneous, dynamic environments (faster CPUs do more work, etc.)

- **Support Scientific Reproducibility**

  Build a robust, repeatable system





Our Goals

What does EWMS need to do?

- ➢ **Complement HTCondor's Capabilities**

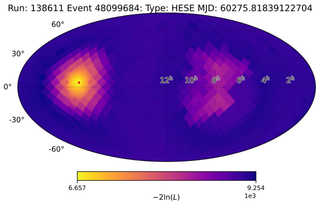
  Thrive in heterogeneous, dynamic environments (faster CPUs do more work, etc.)

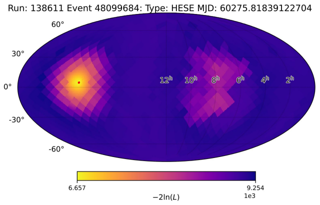
- ➢ **Support Scientific Reproducibility**

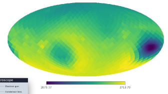
  Build a robust, repeatable system

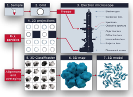
- ➢ **A Service-First Design**

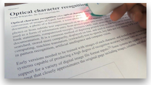
  Build a platform, not an application


Run: 138611 Event 48099684: Type: HESE MJD: 60275.81839122704


Run: 138611 Event 48099684: Type: HESE MJD: 60275.81839122704







Our Goals

What does EWMS need to do?

- ➢ **Complement HTCondor's Capabilities**

  Thrive in heterogeneous, dynamic environments (faster CPUs do more work, etc.)

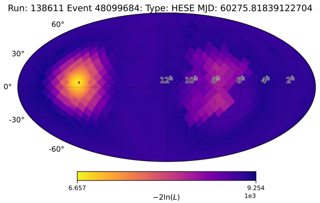
- ➢ **Support Scientific Reproducibility**

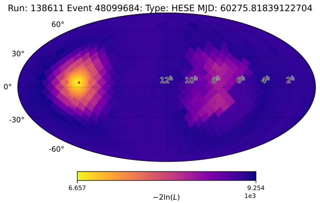
  Build a robust, repeatable system

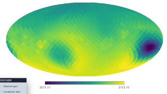
- ➢ **A Service-First Design**

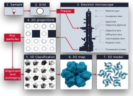
  Build a platform, not an application

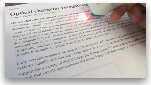
- ➢ **Make everyone happy :)**


Run: 138611 Event 48099684: Type: HESE MJD: 60275.81839122704


Run: 138611 Event 48099684: Type: HESE MJD: 60275.81839122704







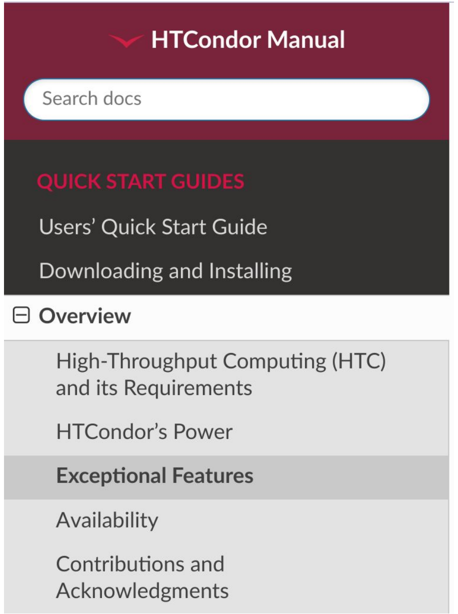
1. Complement HTCondor's Capabilities

How can we work with heterogeneous clusters?

How do we complement HTCondor?

A few of HTCondor's *Exceptional Features*:

- ➢ Guaranteed execution
- ➢ Extreme scalability
- ➢ Parallelization without reimplementation
- ➢ Success in heterogeneous environments
- ➢ Adaptable to user requirements



HTCondor Manual

Search docs

QUICK START GUIDES

Users' Quick Start Guide

Downloading and Installing

⊟ Overview

High-Throughput Computing (HTC) and its Requirements

HTCondor's Power

Exceptional Features

Availability

Contributions and Acknowledgments

Paraphrased from the HTCondor Manual

Complement HTCondor

How can a job have dynamically allocated inputs, outputs, and tasks?

File-transfer system for task I/O (of events) will not suffice:

- ➢ 1:N tasks are complex
- ➢ No dynamic scaling task per job

Complement HTCondor

How can a job have dynamically allocated inputs, outputs, and tasks?

File-transfer system for task I/O (of events) will not suffice:

- ➢ 1:N tasks are complex
- ➢ No dynamic scaling task per job

Message passing (MQ):

- ➢ Separates event I/O from job mechanics
  - ○ Additional **input(s)** are given when needed
  - ○ **Outputs are immediately** relayed in real-time
- ➢ Doesn’t care about fluxuations in job count
  - ○ ***Can we increase/decrease number of jobs?***

▢ guaranteed execution
▢ extreme scalability
▢ parallelization without reimplementation
☑ success in heterogeneous environments
▢ adaptable to user requirements

On Choosing an MQ Protocol

Many possible protocols

- Low-level and foundational decision
- **Expensive to change after implemented**

Credit: Jessie Thwaites

(Not) Choosing an MQ Protocol

Many possible protocols

- ➢ Low-level and foundational decision
- ➢ **Expensive to change after implemented**

Created software to be flexible with **any** of these:

- ➢ RabbitMQ
- ➢ Apache Pulsar
- ➢ NATS.io

□ guaranteed execution
☑ extreme scalability
□ parallelization without reimplementation
☑ success in heterogeneous environments
☑ adaptable to user requirements

Pilot-Based Workers

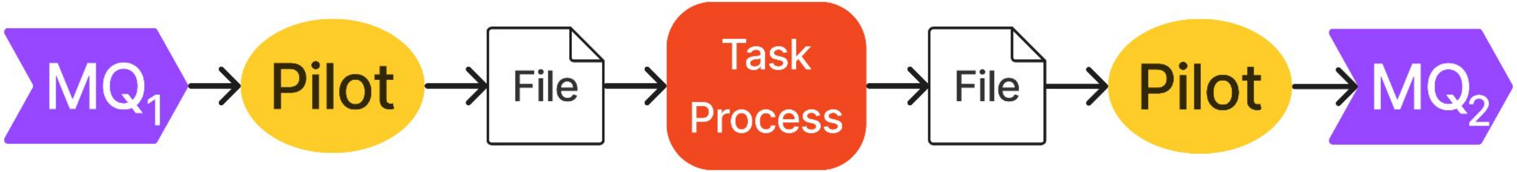
Resilient to CPU crashes – Built-in failover mechanism

- ➢ **Ack-last & fail-fast paradigm**
  - ○ Acknowledge input event only when task is done
  - ○ MQ will redeliver to another worker when no ack
  - ○ “Dead Letter” queue for problem events

Backward compatible – invisible from user’s POV

- ➢ Existing physics algorithms use files as input

☑ guaranteed execution
☑ extreme scalability
☑ parallelization without reimplementation
☑ success in heterogeneous environments
☑ adaptable to user requirements



2. Support Scientific Reproducibility

How can we be assured science results are not due to software bugs?

Versioning & Containerization

What software was used in this analysis?

- Need to document **version** identifier with results

What else can affect the software?

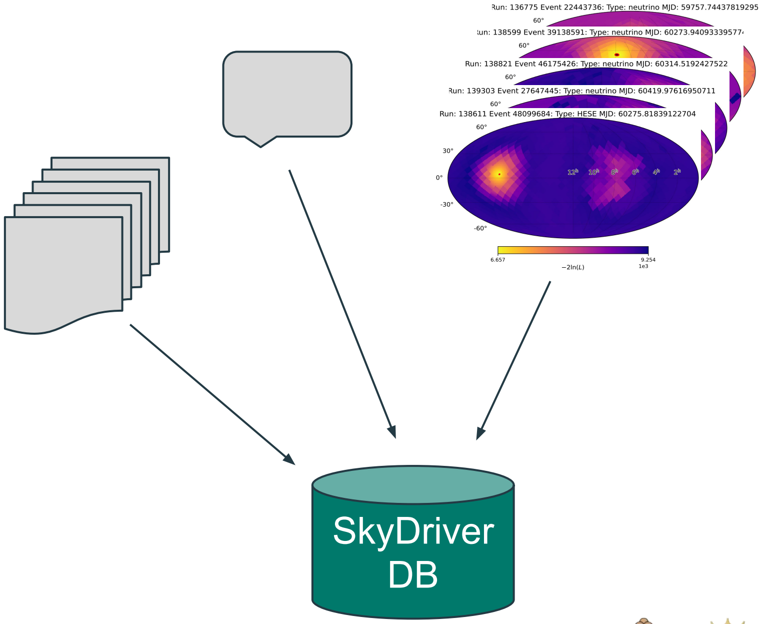
- Need to know what we're testing is what we're running in production
- Using **containers** guarantees consistent reuse

NASA wind tunnel test

Put it all in a centralized database!

For every run of SkyDriver, store:

- **Startup parameters**
- **User-defined tags**
    - Used to find results, limited in size
- **Metadata**
    - Timestamps, basic runtime stats
- **Results A.K.A. Skymaps**


Run: 136775 Event 22443736: Type: neutrino MJD: 59757.74437819295
un: 138599 Event 39138591: Type: neutrino MJD: 60273.94093339577
Run: 138821 Event 46175426: Type: neutrino MJD: 60314.5192427522
Run: 139303 Event 27647445: Type: neutrino MJD: 60419.97616950711
Run: 138611 Event 48099684: Type: HESE MJD: 60275.81839122704
60°
30°
0°
-30°
-60°
12h
10h
8h
6h
4h
2h
6.657
9.254
1e3
−2ln(L)

SkyDriver DB

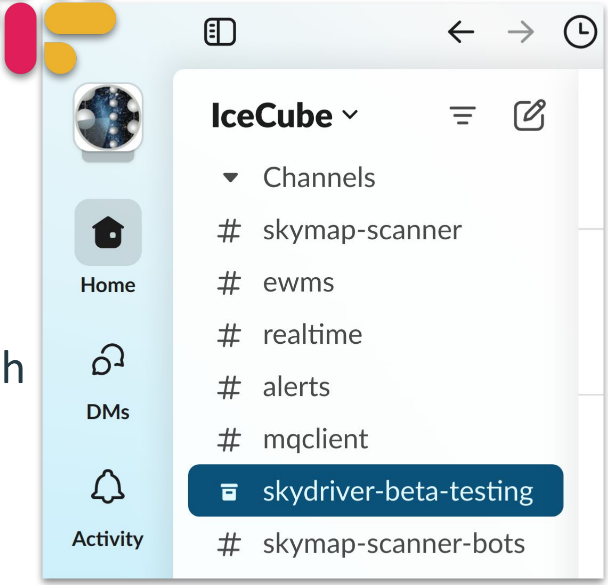
Include Users Throughout the Process

Feedback-Driven Enhancements

- ➢ Don’t spend time designing a solution for no problem

Open Beta Testing (Gamma Testing)

- ➢ Advertised as a prototype, un-ready system, with an end date goal (Q4 2023) – team effort
- ➢ Created slack channel for this purpose, closed channel when beta testing was completed



IceCube

Channels

- # skymap-scanner
- # ewms
- # realtime
- # alerts
- # mqclient
- skydriver-beta-testing
- # skymap-scanner-bots

3. A Service-First Design

What do users need to know to be successful?

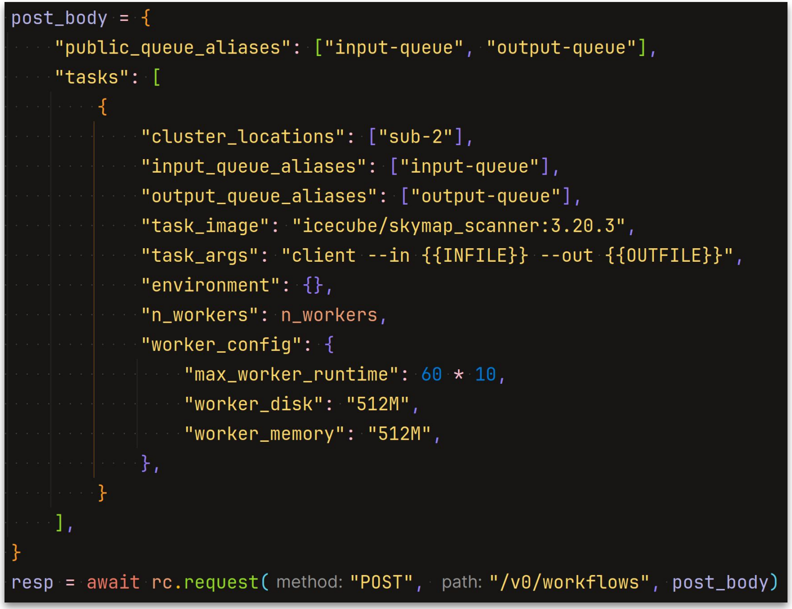
A Service-First Design

How do we get people to use our system?

*If our system is **not simple** to onboard,*
*it **won't be used!***

HTTP / REST user interface

- Standardized **JSON input** – auto-documented
  - Validation by **JSON Schema & OpenAPI**
- **Multiple image versions** available, including feature-branch versions
  - SkyDriver uses Skymap Scanner Images
  - Allows users to **test customizations**



```
post_body = {
    "public_queue_aliases": ["input-queue", "output-queue"],
    "tasks": [
        {
            "cluster_locations": ["sub-2"],
            "input_queue_aliases": ["input-queue"],
            "output_queue_aliases": ["output-queue"],
            "task_image": "icecube/skymap_scanner:3.20.3",
            "task_args": "client --in {{INFILE}} --out {{OUTFILE}}",
            "environment": {},
            "n_workers": n_workers,
            "worker_config": {
                "max_worker_runtime": 60 * 10,
                "worker_disk": "512M",
                "worker_memory": "512M",
            },
        }
    ],
}
resp = await rc.request( method: "POST",  path: "/v0/workflows", post_body)
```

Looking Back and Forward

How's EWMS going?

Challenges

- ***How generalized of a system do we want?***
- **Many unique tools:** Kubernetes, Helm, Docker, Python Packaging, REST, Input Validation, …
- Some errors only appear **at massive workflow scales**
- Removing **tech debt** from original Skymap Scanner
  - **Created "organically"**
  - *"How does this work?"… "I don't remember."*
- Small development team size **(1.1 full-time)**

Oversimplified Timeline

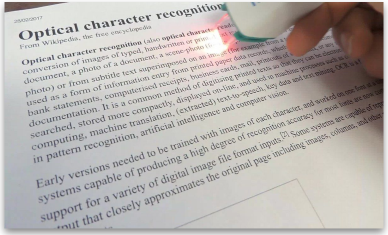
2022: MQ-equipped Skymap Scanner

2022-23: SkyDriver

2024: Generalized EWMS

EWMS: Ongoing and Upcoming Features

- ➢ Release generalized EWMS *(currently in alpha)*
- ➢ Automatic job scaling by detecting MQ usage and availability of compute resources in HTCondor pool
- ➢ Real-time monitoring dashboard
- ➢ Support scheduling for DAG workflows



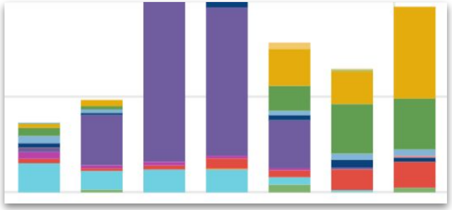
28/02/2017

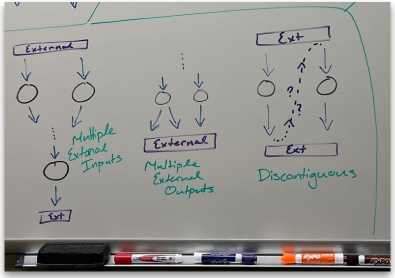
Optical character recognitio

From Wikipedia, the free encyclopedia

Optical character recognition (also **optical characte** reade
conversion of images of typed, handwritten or printed text in
document, a photo of a document, a scene-photo (fo
photo) or from subtitle text superimposed on an image (for example from a
used as a form of information entry from printed paper data records, wheth
bank statements, computerised receipts, business cards, mail, printouts of s
documentation. It is a common method of digitising printed texts so that they can be electron
searched, stored more compactly, displayed on-line, and used in machine processes such as c
computing, machine translation, (extracted) text-to-speech, key data and text mining. OCR is a f
in pattern recognition, artificial intelligence and computer vision.

Early versions needed to be trained with images of each character, and worked on one font at a ti
systems capable of producing a high degree of recognition accuracy for most fonts are now com
support for a variety of digital image file format inputs.[2] Some systems are capable of repr
tput that closely approximates the original page including images, columns, and other



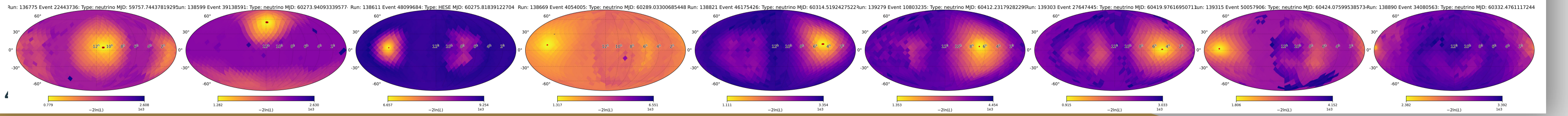


Looking Back and Forward

Summary

How can we take a workload, consisting of millions or billions of tasks, and group it into tens of thousands of jobs?

- **Complement HTCondor's Capabilities**
  Using message passing-equipped worker pilots to thrive in heterogeneous, dynamic environments
- **Support Scientific Reproducibility**
  Providing dependable software, developed with user feedback
- **A Service-First Design**
  Putting the user's POV first, simple interfaces and removed complexities
- **Made everyone happy :)**



Acknowledgements

PIs (EWMS)

- Miron Livny
- Brian Bockelman
- Benedikt Riedel

Developers

- Ric Evans (me)
- Benedikt Riedel
- David Schultz

IceCubers

- Massimiliano Lincetto
- Tianlu Yuan
- Claudio Kopper
- Erik Blaufuss
- Christina Lagunas
- Robert Stein

National Science Foundation Funding

- OAC #2103963 + OPP #2042807

THE ICECUBE COLLABORATION

AUSTRALIA
University of Adelaide

BELGIUM
UCLouvain
Université libre de Bruxelles
Universiteit Gent
Vrije Universiteit Brussel

CANADA
Queen's University
University of Alberta–Edmonton

DENMARK
University of Copenhagen

GERMANY
Deutsches Elektronen-Synchrotron
ECAP, Universität Erlangen-Nürnberg
Humboldt–Universität zu Berlin
Karlsruhe Institute of Technology
Ruhr-Universität Bochum
RWTH Aachen University
Technische Universität Dortmund
Technische Universität München
Universität Mainz
Universität Wuppertal
Westfälische Wilhelms-Universität Münster

ITALY
University of Padova

JAPAN
Chiba University

NEW ZEALAND
University of Canterbury

SOUTH KOREA
Sungkyunkwan University

SWEDEN
Stockholms universitet
Uppsala universitet

SWITZERLAND
Université de Genève

TAIWAN
Academia Sinica

UNITED KINGDOM
University of Oxford

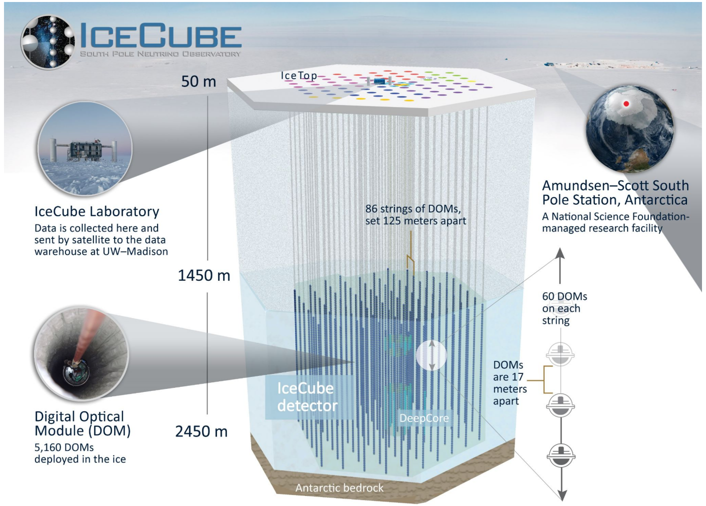
UNITED STATES
Clark Atlanta University
Columbia University
Drexel University
Georgia Institute of Technology
Harvard University
Lawrence Berkeley National Lab
Loyola University Chicago
Marquette University
Massachusetts Institute of Technology
Mercer University
Michigan State University
Ohio State University
Pennsylvania State University
South Dakota School of Mines and Technology
Southern University and A&M College
Stony Brook University
University of Alabama
University of Alaska Anchorage
University of California, Berkeley
University of California, Irvine
University of Delaware
University of Kansas
University of Maryland
University of Nevada, Las Vegas
University of Rochester
University of Texas at Arlington
University of Utah
University of Wisconsin–Madison
University of Wisconsin–River Falls
Yale University

FUNDING AGENCIES

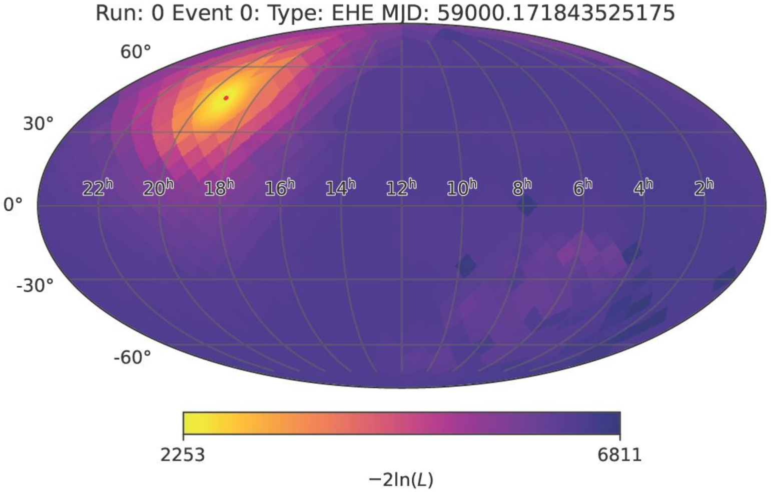
Fonds de la Recherche Scientifique (FRS-FNRS)
Fonds Wetenschappelijk Onderzoek-Vlaanderen (FWO-Vlaanderen)
Federal Ministry of Education and Research (BMBF)
German Research Foundation (DFG)
Deutsches Elektronen-Synchrotron (DESY)
Japan Society for the Promotion of Science (JSPS)
Knut and Alice Wallenberg Foundation
Swedish Polar Research Secretariat
The Swedish Research Council (VR)
University of Wisconsin Alumni Research Foundation (WARF)
US National Science Foundation (NSF)

icecube.wisc.edu

Thank You!



ICECUBE
SOUTH POLE NEUTRINO OBSERVATORY
IceTop
50 m
IceCube Laboratory
Data is collected here and sent by satellite to the data warehouse at UW–Madison
1450 m
Digital Optical Module (DOM)
5,160 DOMs deployed in the ice
2450 m
86 strings of DOMs, set 125 meters apart
IceCube detector
DeepCore
Antarctic bedrock
Amundsen–Scott South Pole Station, Antarctica
A National Science Foundation-managed research facility
60 DOMs on each string
DOMs are 17 meters apart



Run: 0 Event 0: Type: EHE MJD: 59000.171843525175
60°
30°
0°
-30°
-60°
22h
20h
18h
16h
14h
12h
10h
8h
6h
4h
2h
2253
6811
−2ln(L)

The Problem

Two IceCube Use Cases

CASE 1: Massive Scale

Real-time Scans

Fast & Resource Intensive -> High Priority

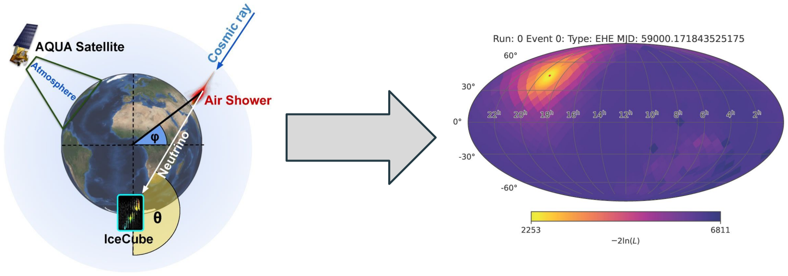
➔ O(10k+) CPUs, spun up ASAP

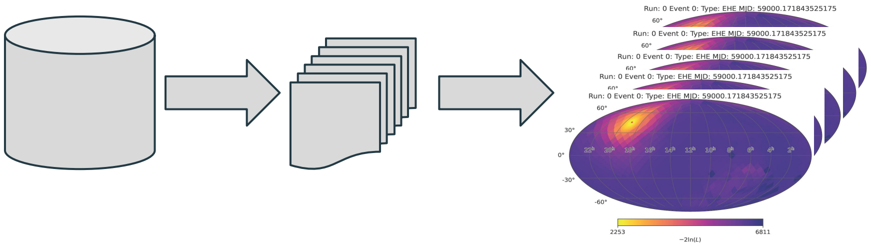
CASE 2: Moderate Scale

Historical Catalog & Simulation

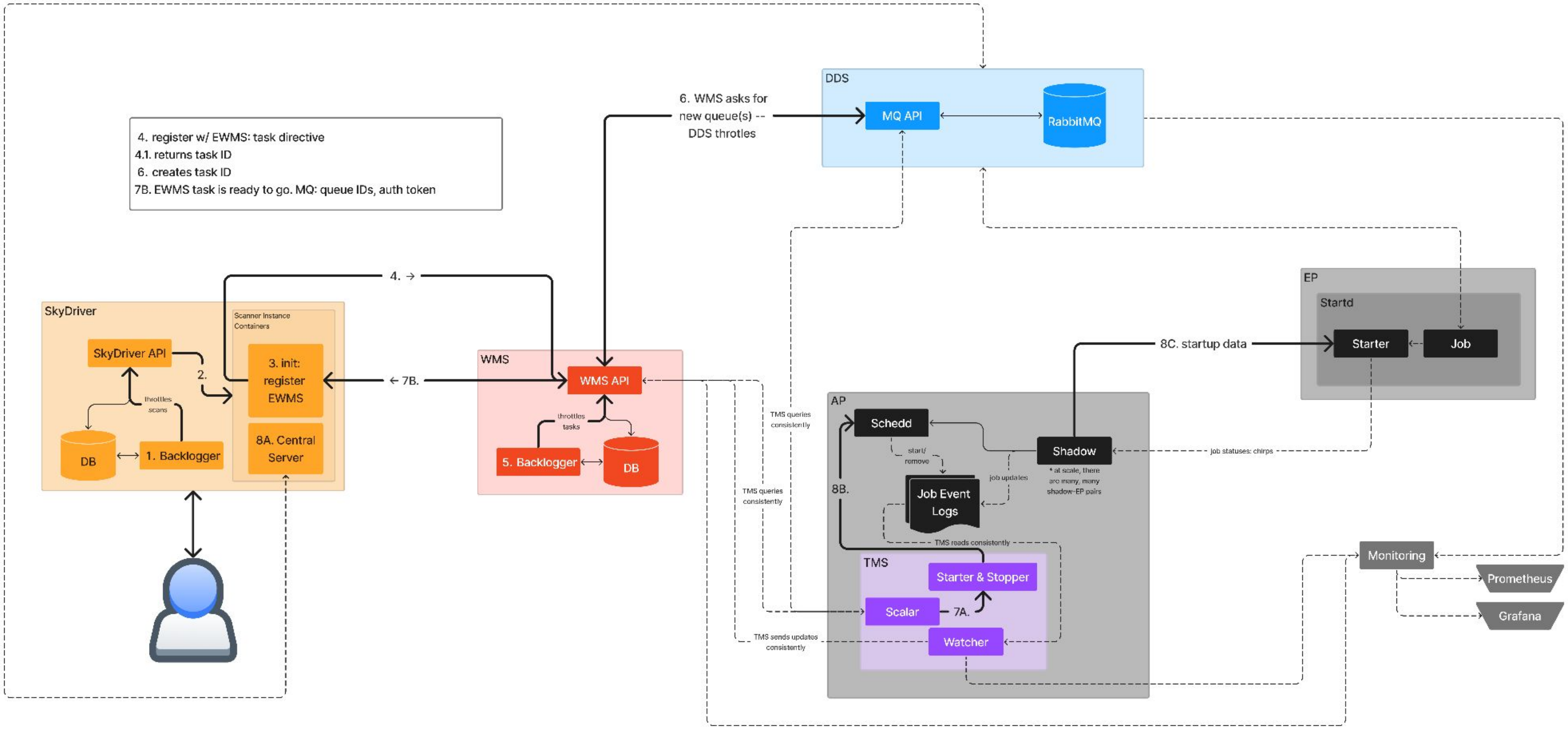
Steady/Predictable -> Lower Priority

➔ Varying # of CPUs, subject to availability





SkyDriver-EWMS Architecture



4. register w/ EWMS: task directive
4.1. returns task ID
6. creates task ID
7B. EWMS task is ready to go. MQ: queue IDs, auth token

Development Methodology

Minimum viable product

- ➢ Wait to implement enhancements until needed

Test every enhancement & bug fix

- ➢ Use non-domain specific data & workflows

Do enhancements in order of priority

- ➢ Track in GitHub

Test, scale up, test, scale up, test, …

1. **Test at no scale** – *fast*

➢ Test individual components

2. **Test at mini scale** – *cheap*

➢ 1 or 2 jobs in automated CI environment (Github Actions)

3. **Test at large scale** – *conservative*

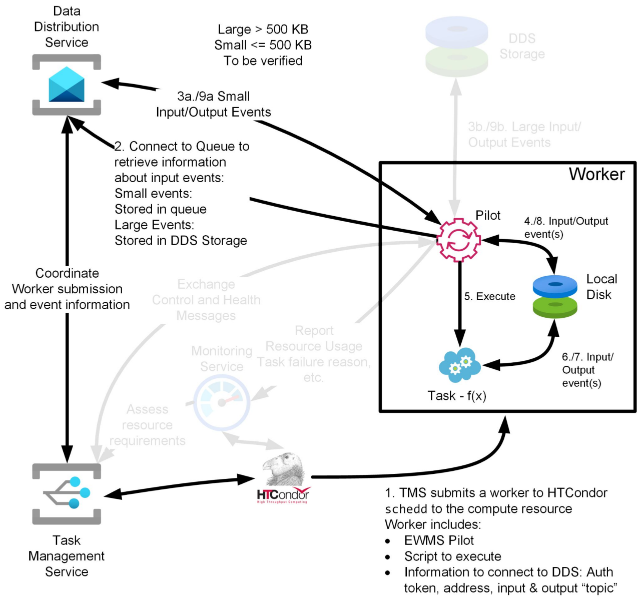
➢ Use production cluster w/ downsized configuration

4. **Test at full scale**

➢ Use production configuration

5. **Publish Release**

SkyDriver – Worker / Scanner Client POV



Data Distribution Service

Large > 500 KB
Small <= 500 KB
To be verified

DDS Storage

3a./9a Small Input/Output Events

3b./9b. Large Input/ Output Events

2. Connect to Queue to retrieve information about input events:
Small events:
Stored in queue
Large Events:
Stored in DDS Storage

Worker

Pilot

4./8. Input/Output event(s)

Local Disk

5. Execute

6./7. Input/ Output event(s)

Task - f(x)

Coordinate Worker submission and event information

Exchange Control and Health Messages

Report Resource Usage Task failure reason, etc.

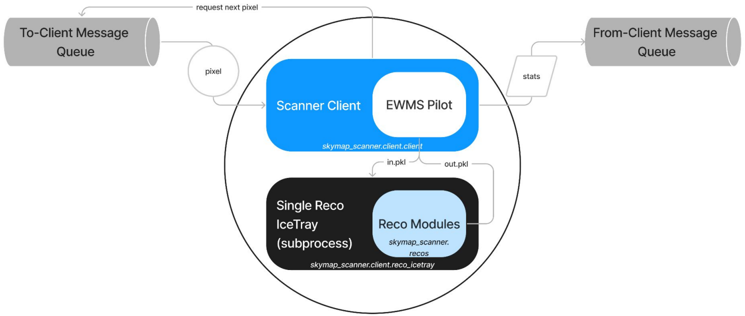
Monitoring Service

Assess resource requirements

HTCondor

Task Management Service

1. TMS submits a worker to HTCondor `schedd` to the compute resource
Worker includes:
- EWMS Pilot
- Script to execute
- Information to connect to DDS: Auth token, address, input & output "topic"



To-Client Message Queue

request next pixel

pixel

From-Client Message Queue

stats

Scanner Client

EWMS Pilot

skymap_scanner.client.client

in.pkl

out.pkl

Single Reco IceTray (subprocess)

Reco Modules

skymap_scanner.recos

skymap_scanner.client.reco_icetray