Apache Airflow @ Agari

Sid Anand (@r39132)

Apache Airflow Meet-up

June 2016

Agari

What We Do!

Agari : What We Do

Agari : What We Do

Agari : What We Do

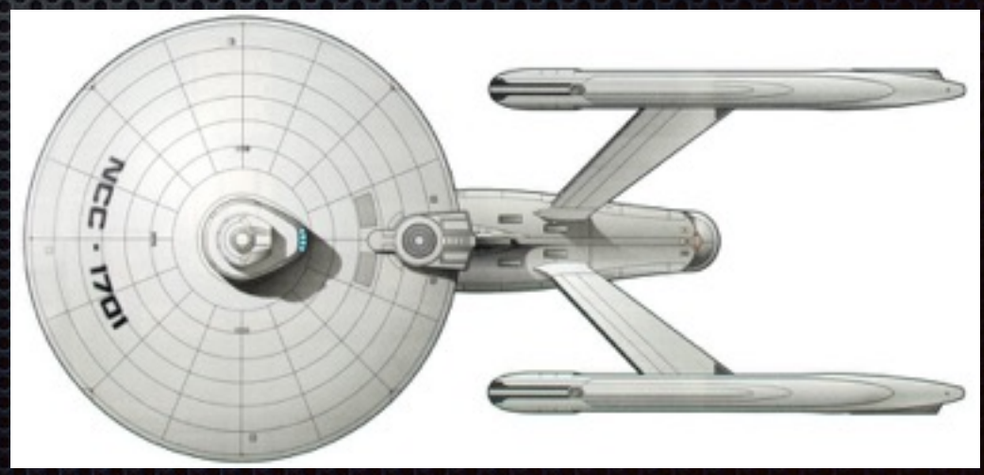
Agari : What We Do

AGARI

Agari : What We Do

AGARI

AGARI



Agari : What We Do

Enterprise Customers

email metadata

Agari's Current Product

apply trust models

email md + trust score

Agari : What We Do

Enterprise Customers

email metadata

Agari's Future Product

apply trust models

email md + trust score

Apache Airflow @ Agari

How Do We Use It?

2 Classes of Orchestration

New Product
(Enterprise Protect)

build trust models

apply trust models (message scoring)

Operational Automation

cron++ (general job scheduler)

2 Classes of Orchestration

New Product
(Enterprise Protect)

build trust models

apply trust models (message scoring)

Operational Automation

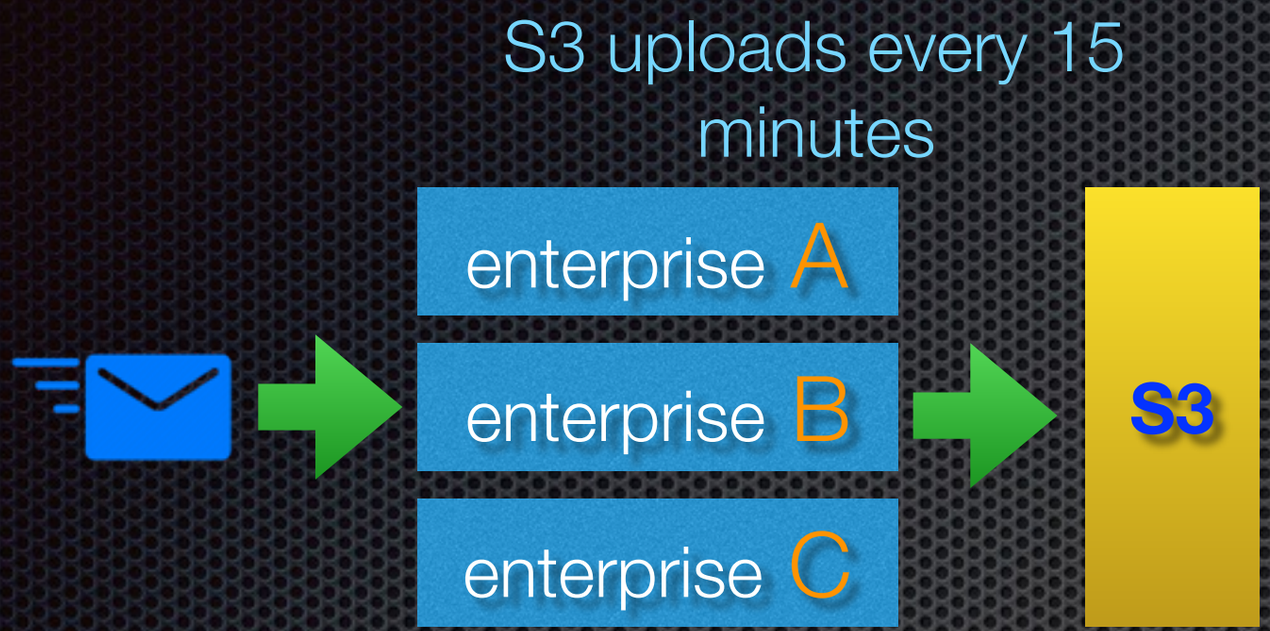
cron++ (general job scheduler)

This Talk

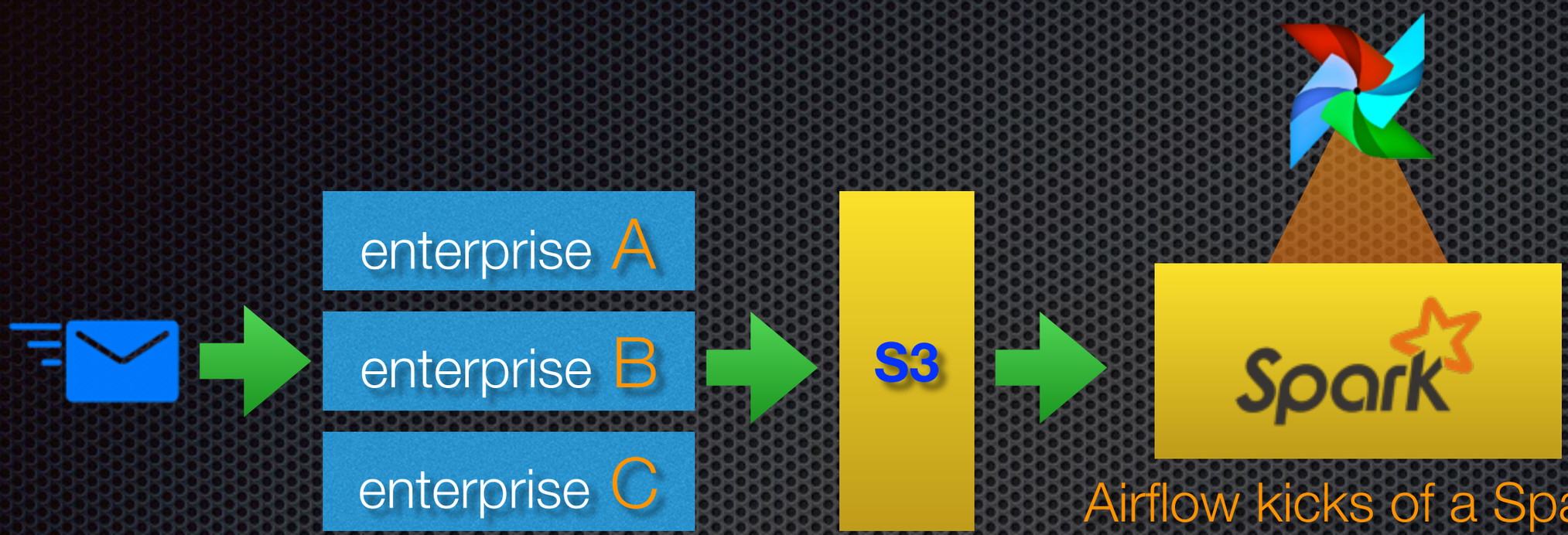
Use-Case : Message Scoring

Batch Pipeline Architecture

Use-Case : Message Scoring

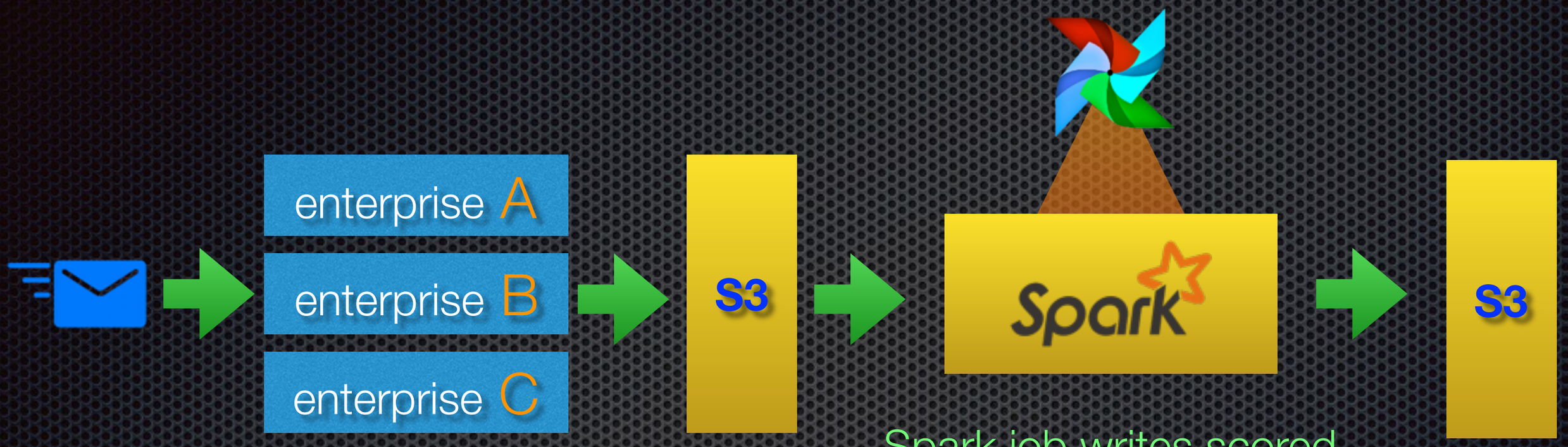


Use-Case : Message Scoring



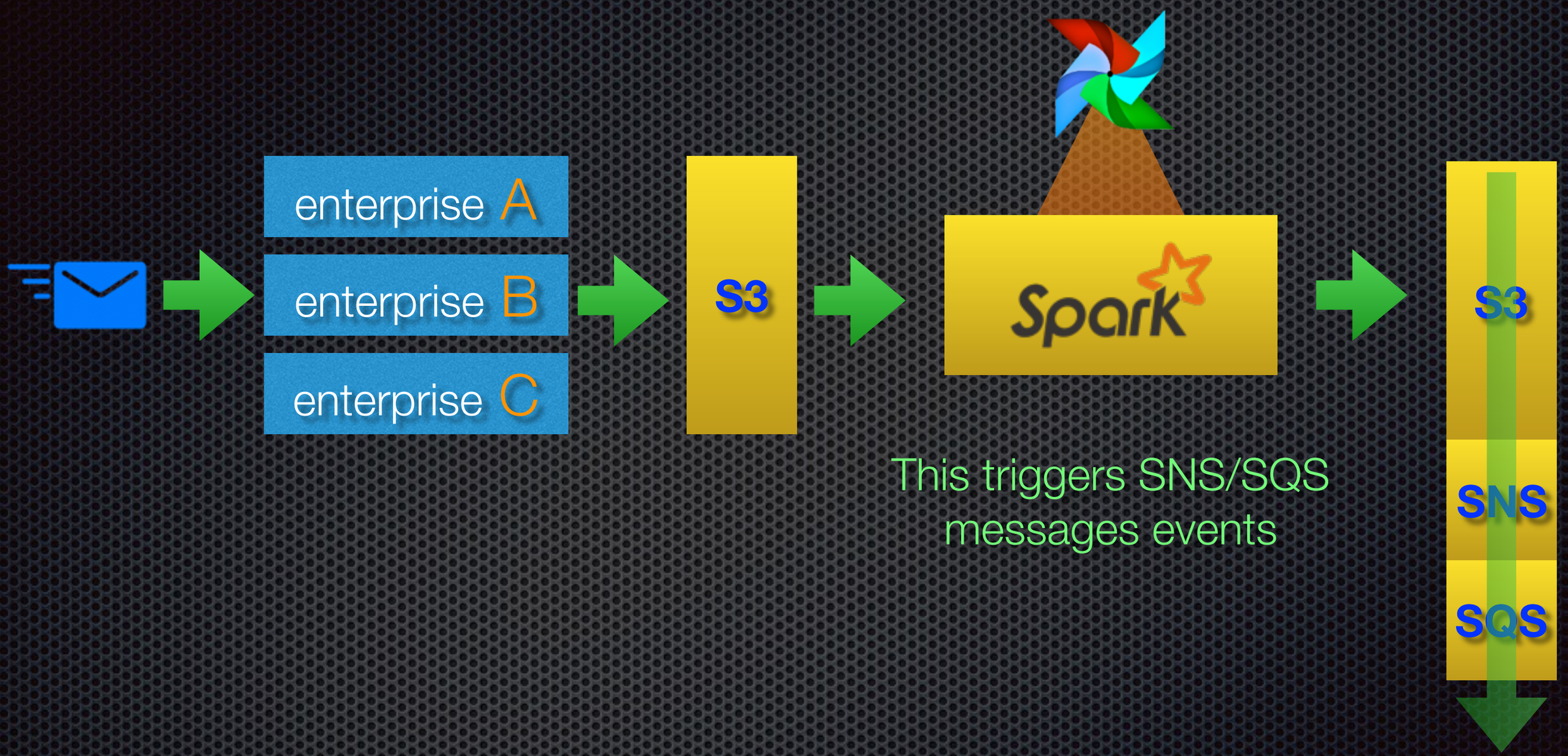
Airflow kicks of a Spark message scoring job every hour

Use-Case : Message Scoring



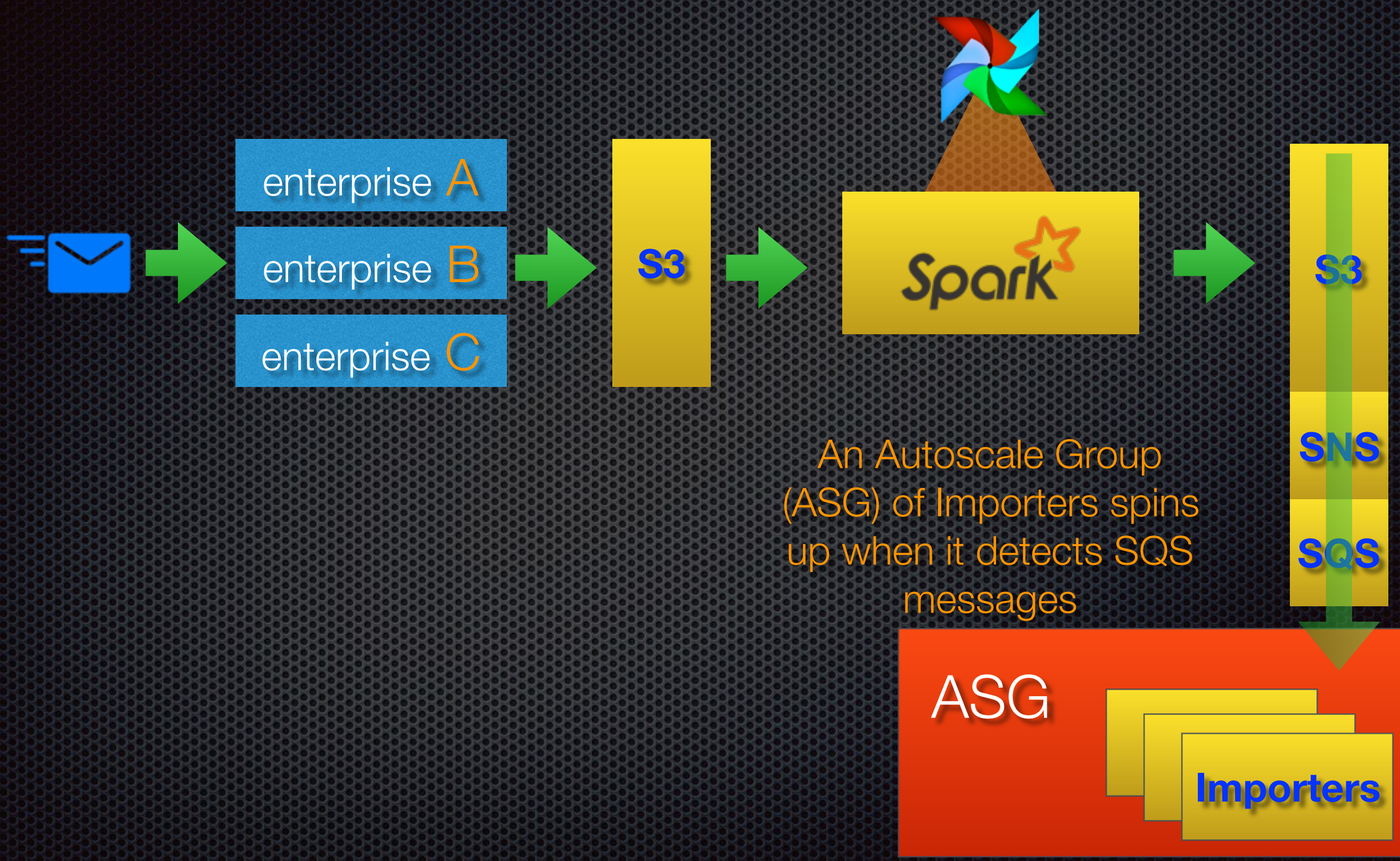
Spark job writes scored messages and stats to another S3 bucket

Use-Case : Message Scoring

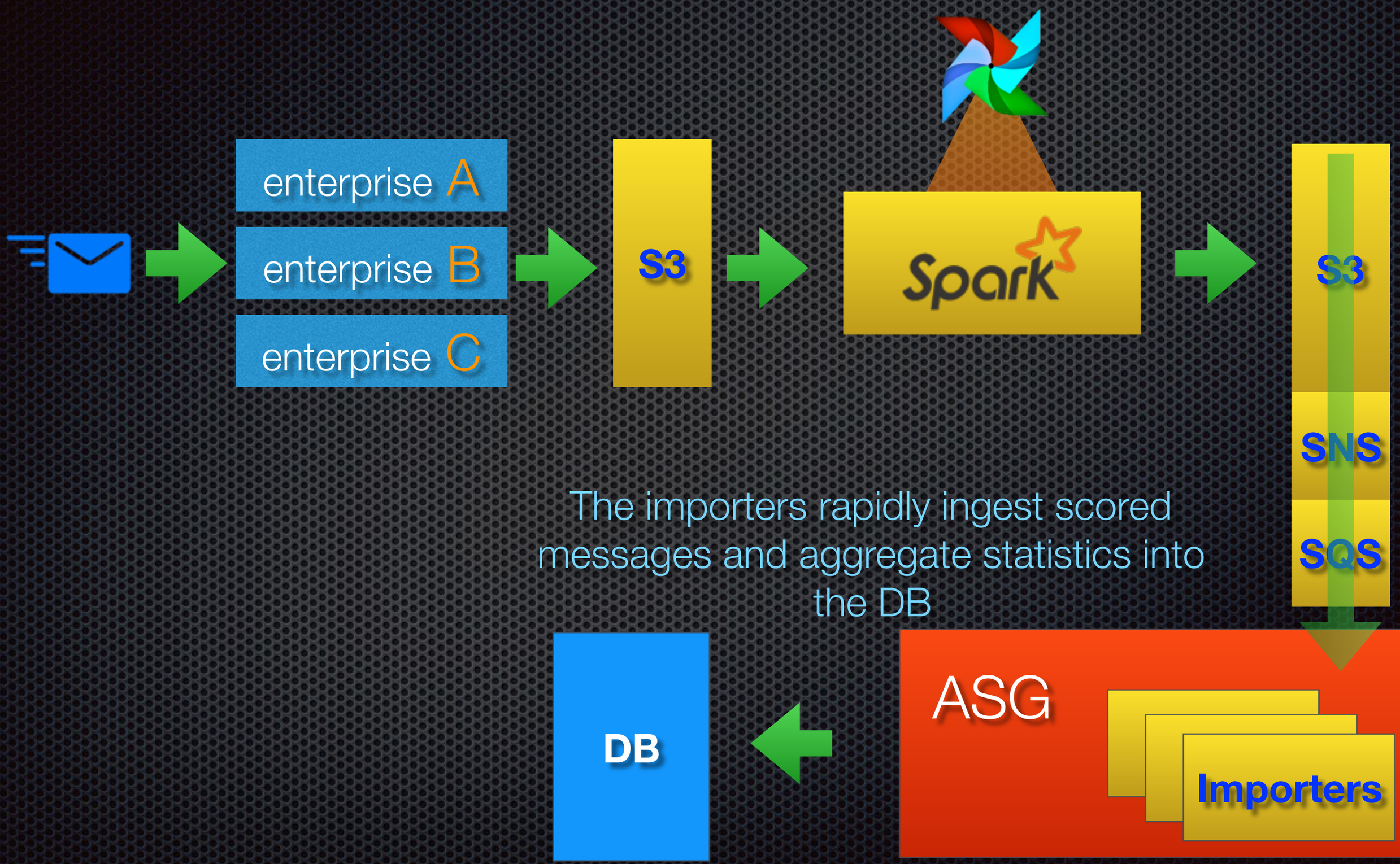


This triggers SNS/SQS messages events

Use-Case : Message Scoring

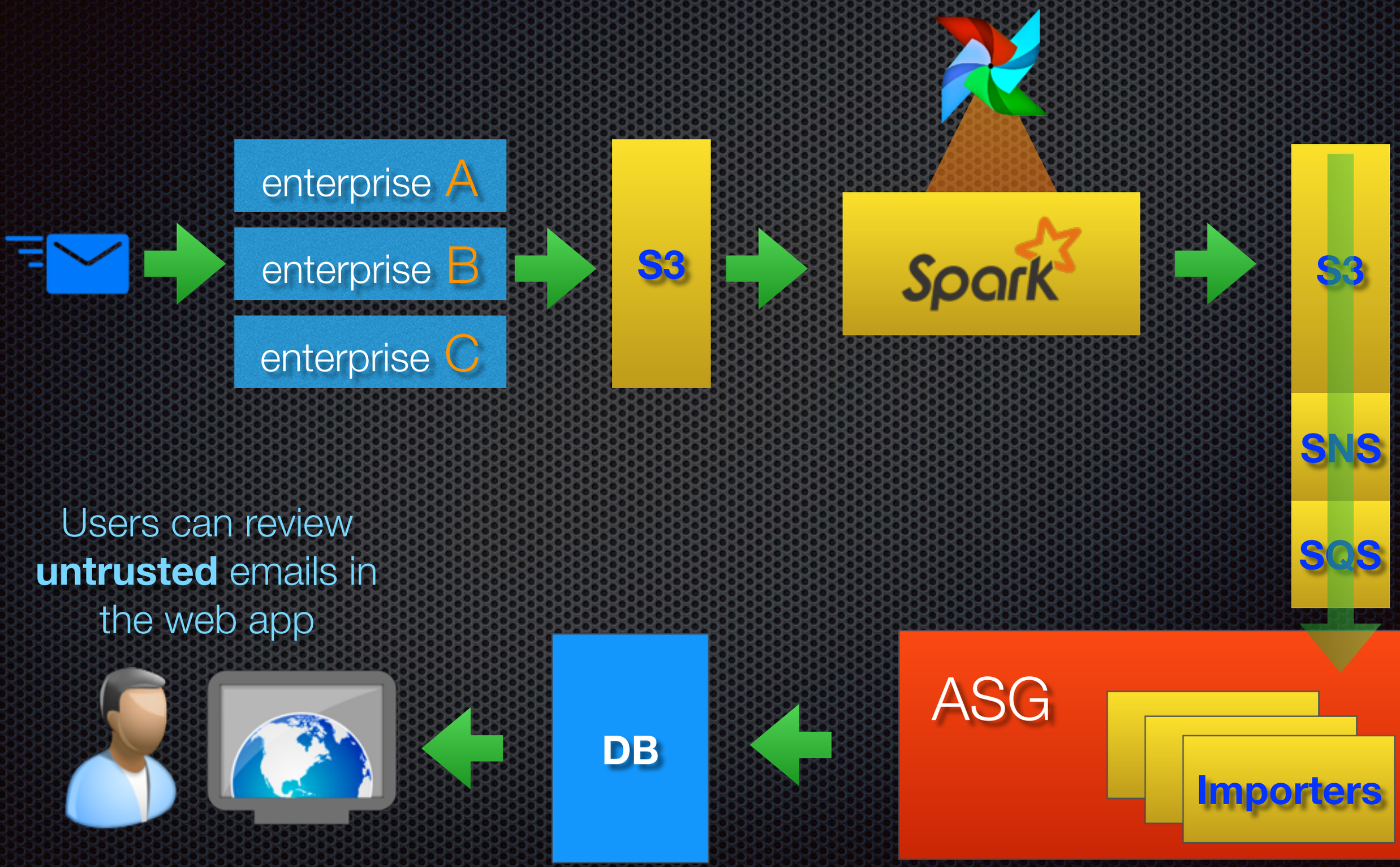


Use-Case : Message Scoring



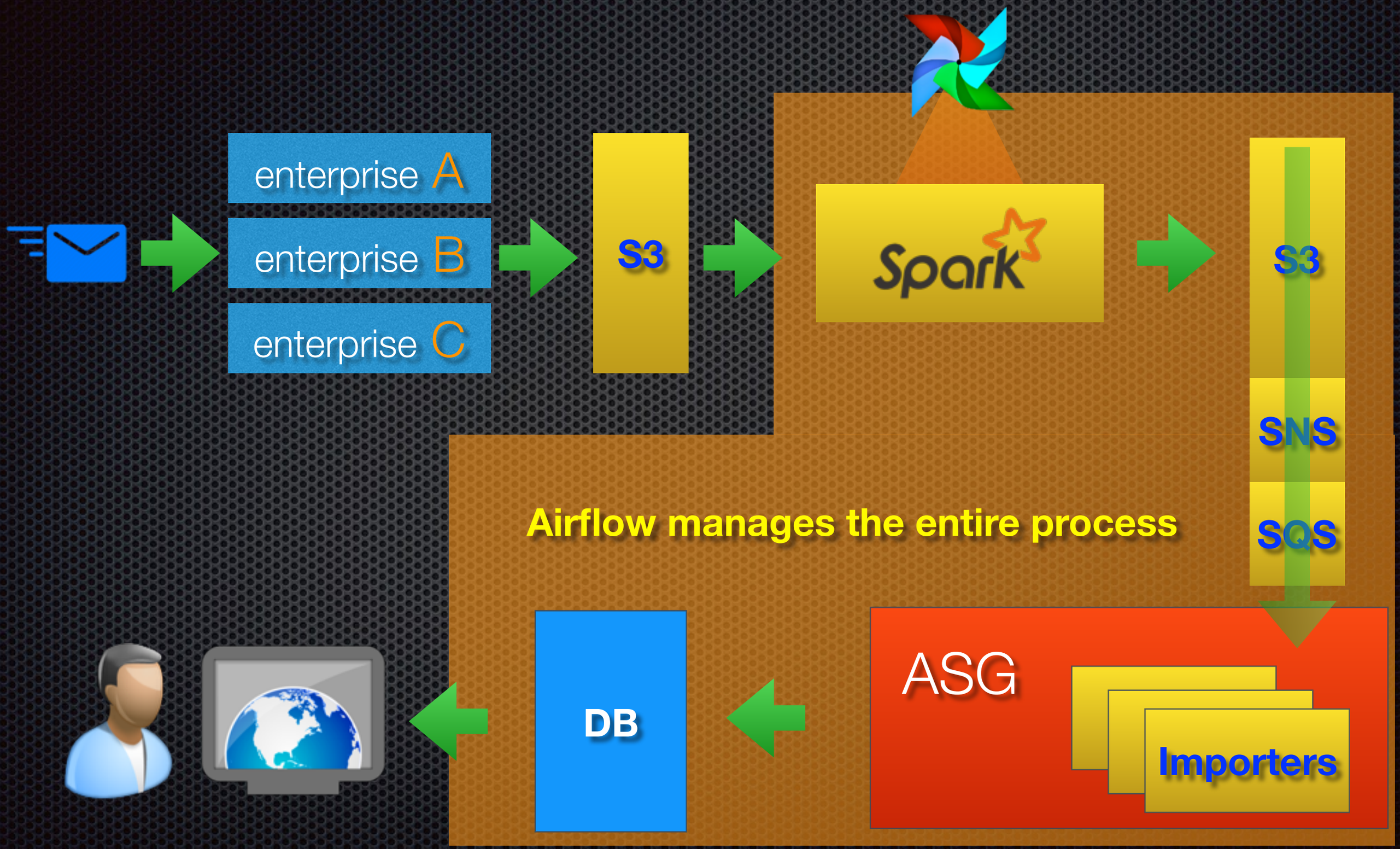
The importers rapidly ingest scored messages and aggregate statistics into the DB

Use-Case : Message Scoring



Users can review **untrusted** emails in the web app

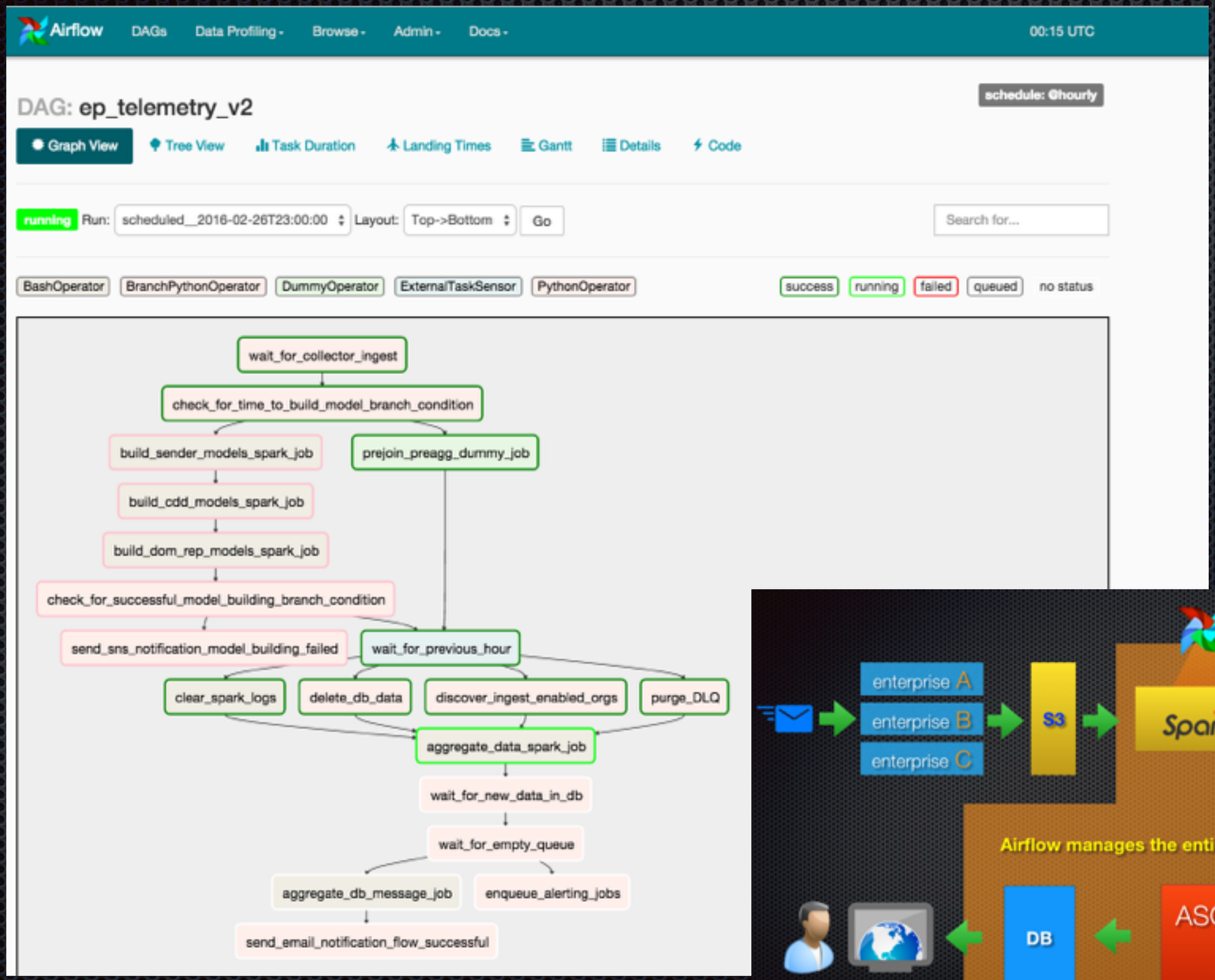
Use-Case : Message Scoring



Use-Case : Message Scoring

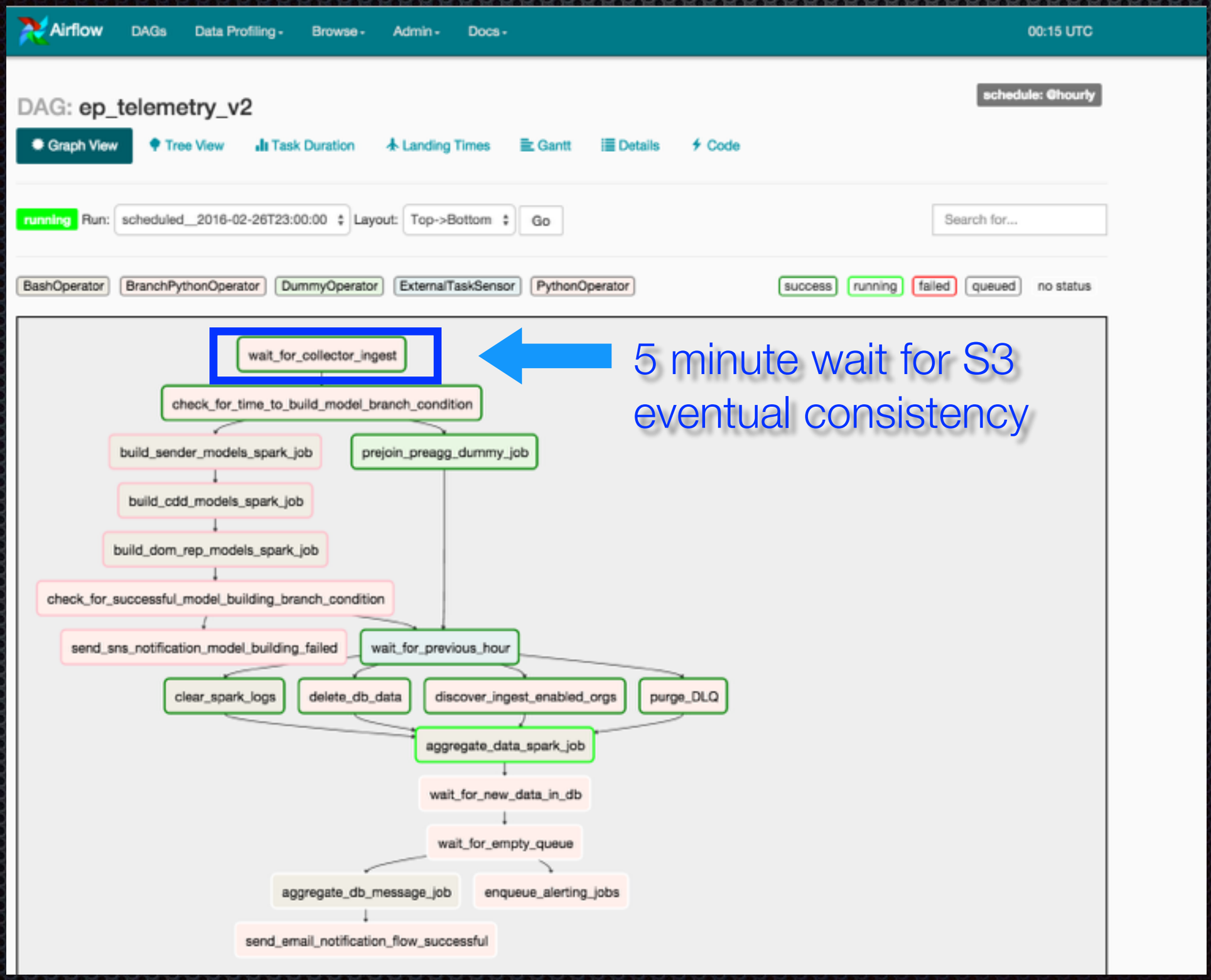
Airflow DAG

Airflow DAG

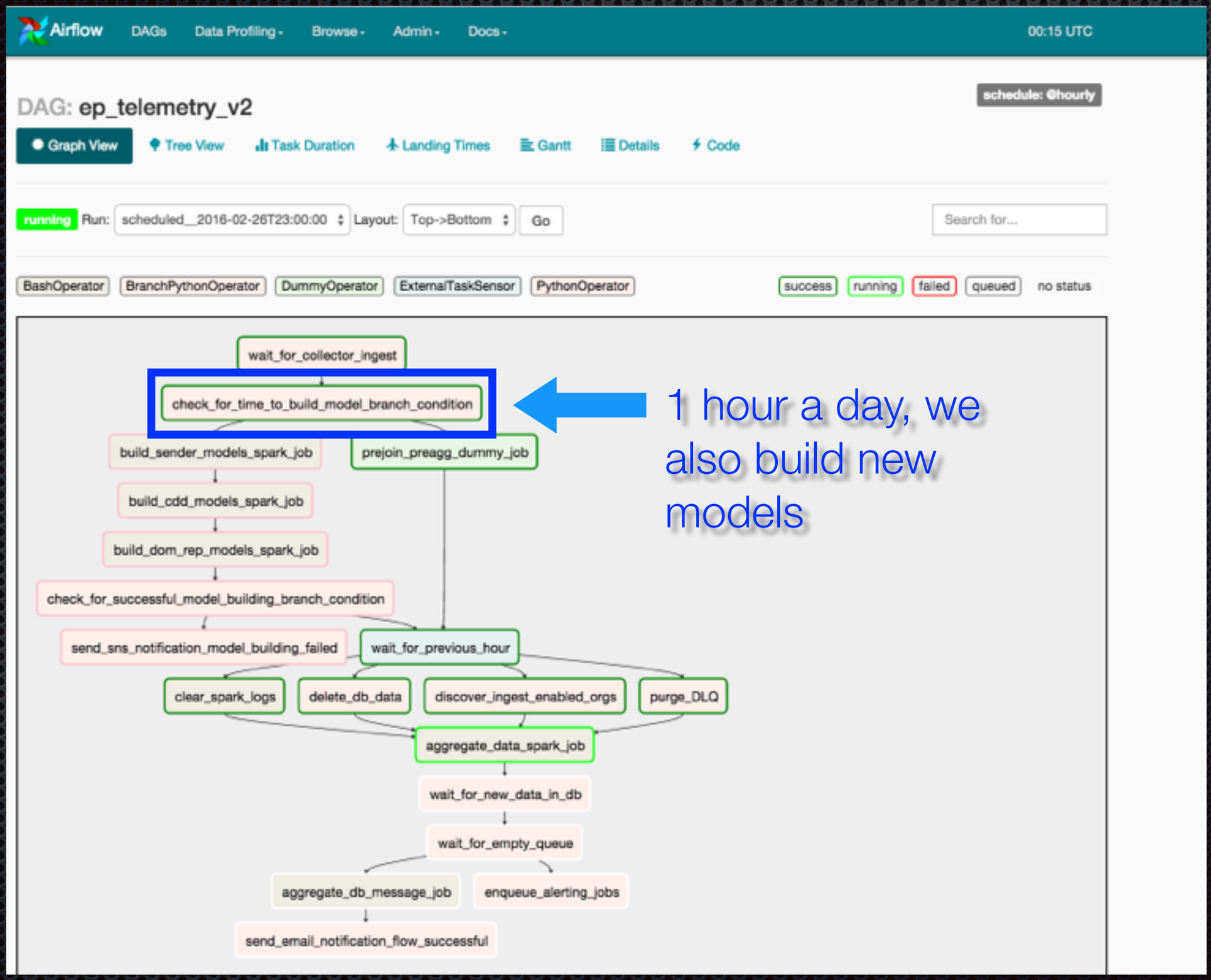

Airflow
DAGs
Data Profiling
Browse
Admin
Docs
00:15 UTC
DAG: ep_telemetry_v2
schedule: @hourly
Graph View
Tree View
Task Duration
Landing Times
Gantt
Details
Code
running
Run: scheduled__2016-02-26T23:00:00
Layout: Top->Bottom
Go
Search for...
BashOperator
BranchPythonOperator
DummyOperator
ExternalTaskSensor
PythonOperator
success
running
failed
queued
no status
wait_for_collector_ingest
check_for_time_to_build_model_branch_condition
build_sender_models_spark_job
prejoin_preagg_dummy_job
build_cdd_models_spark_job
build_dom_rep_models_spark_job
check_for_successful_model_building_branch_condition
send_sns_notification_model_building_failed
wait_for_previous_hour
clear_spark_logs
delete_db_data
discover_ingest_enabled_orgs
purge_DLQ
aggregate_data_spark_job
wait_for_new_data_in_db
wait_for_empty_queue
aggregate_db_message_job
enqueue_alerting_jobs
send_email_notification_flow_successful

enterprise A
enterprise B
enterprise C
S3
Spark
S3
SNS
SQS
Airflow manages the entire process
ASG
Importers
DB

Airflow DAG


Airflow
DAGs
Data Profiling
Browse
Admin
Docs
00:15 UTC
DAG: ep_telemetry_v2
schedule: @hourly
Graph View
Tree View
Task Duration
Landing Times
Gantt
Details
Code
running
Run: scheduled__2016-02-26T23:00:00
Layout: Top->Bottom
Go
Search for...
BashOperator
BranchPythonOperator
DummyOperator
ExternalTaskSensor
PythonOperator
success
running
failed
queued
no status
wait_for_collector_ingest
check_for_time_to_build_model_branch_condition
build_sender_models_spark_job
prejoin_preagg_dummy_job
build_cdd_models_spark_job
build_dom_rep_models_spark_job
check_for_successful_model_building_branch_condition
send_sns_notification_model_building_failed
wait_for_previous_hour
clear_spark_logs
delete_db_data
discover_ingest_enabled_orgs
purge_DLQ
aggregate_data_spark_job
wait_for_new_data_in_db
wait_for_empty_queue
aggregate_db_message_job
enqueue_alerting_jobs
send_email_notification_flow_successful
5 minute wait for S3
eventual consistency

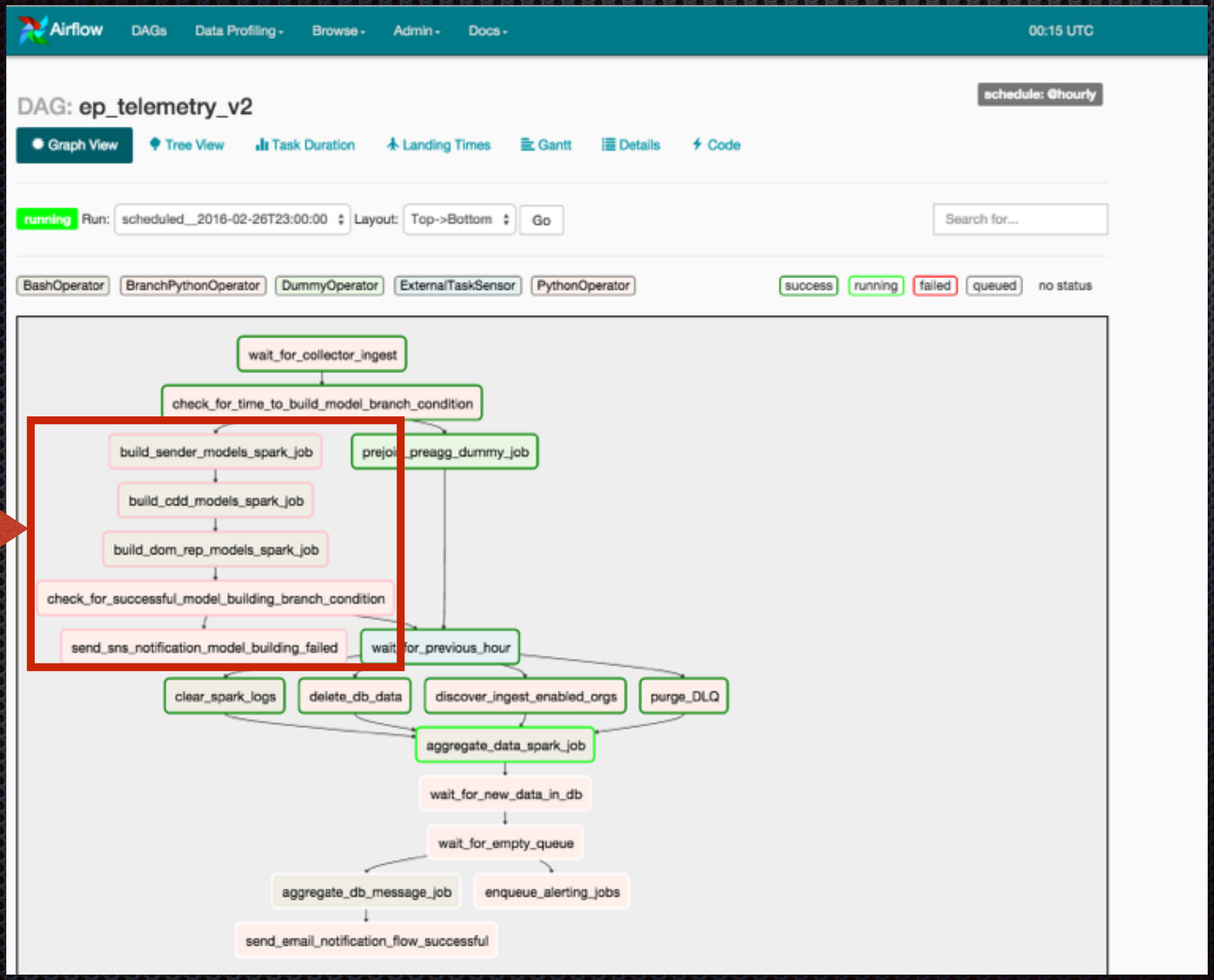
Airflow DAG



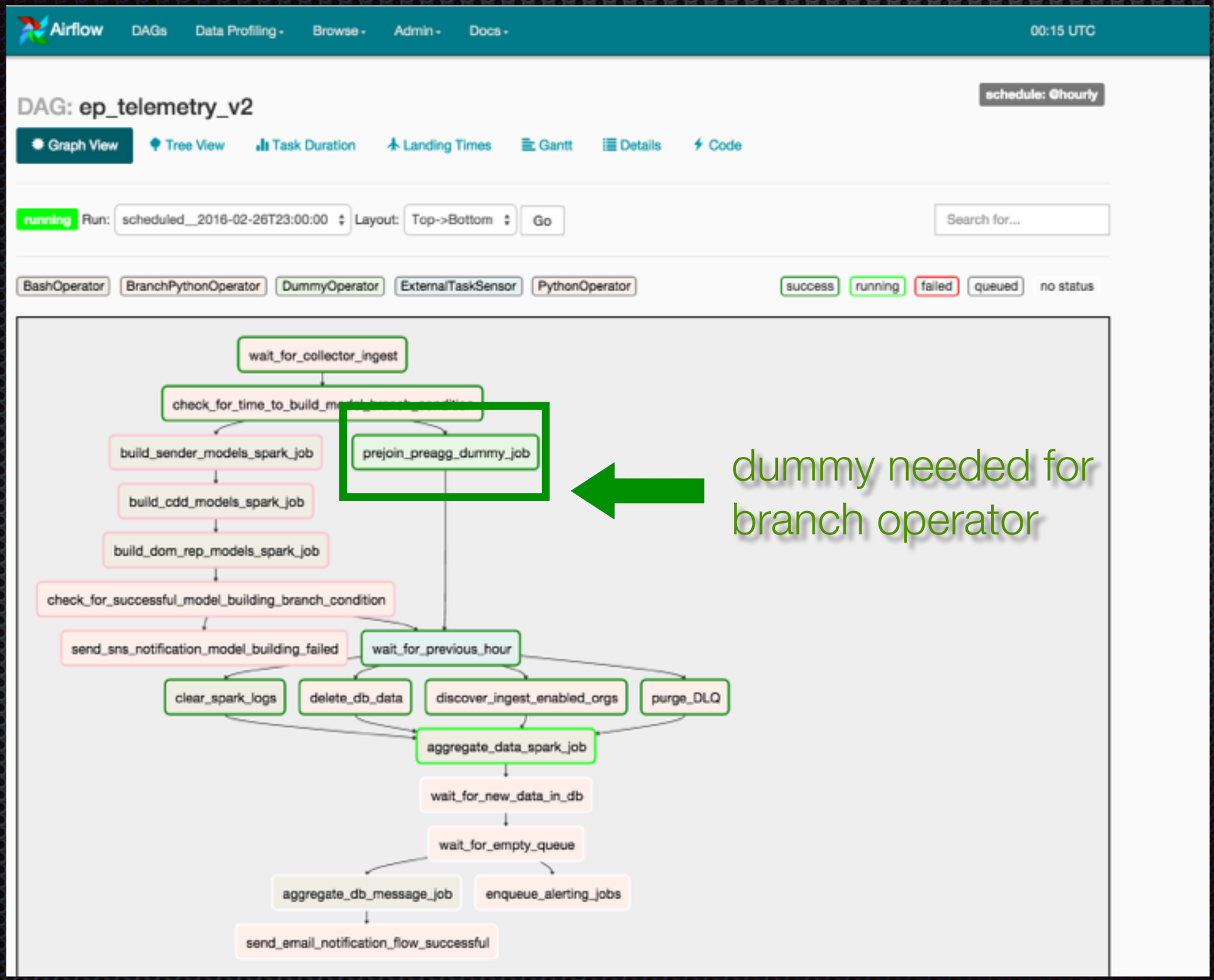
1 hour a day, we also build new models

Airflow DAG

build models

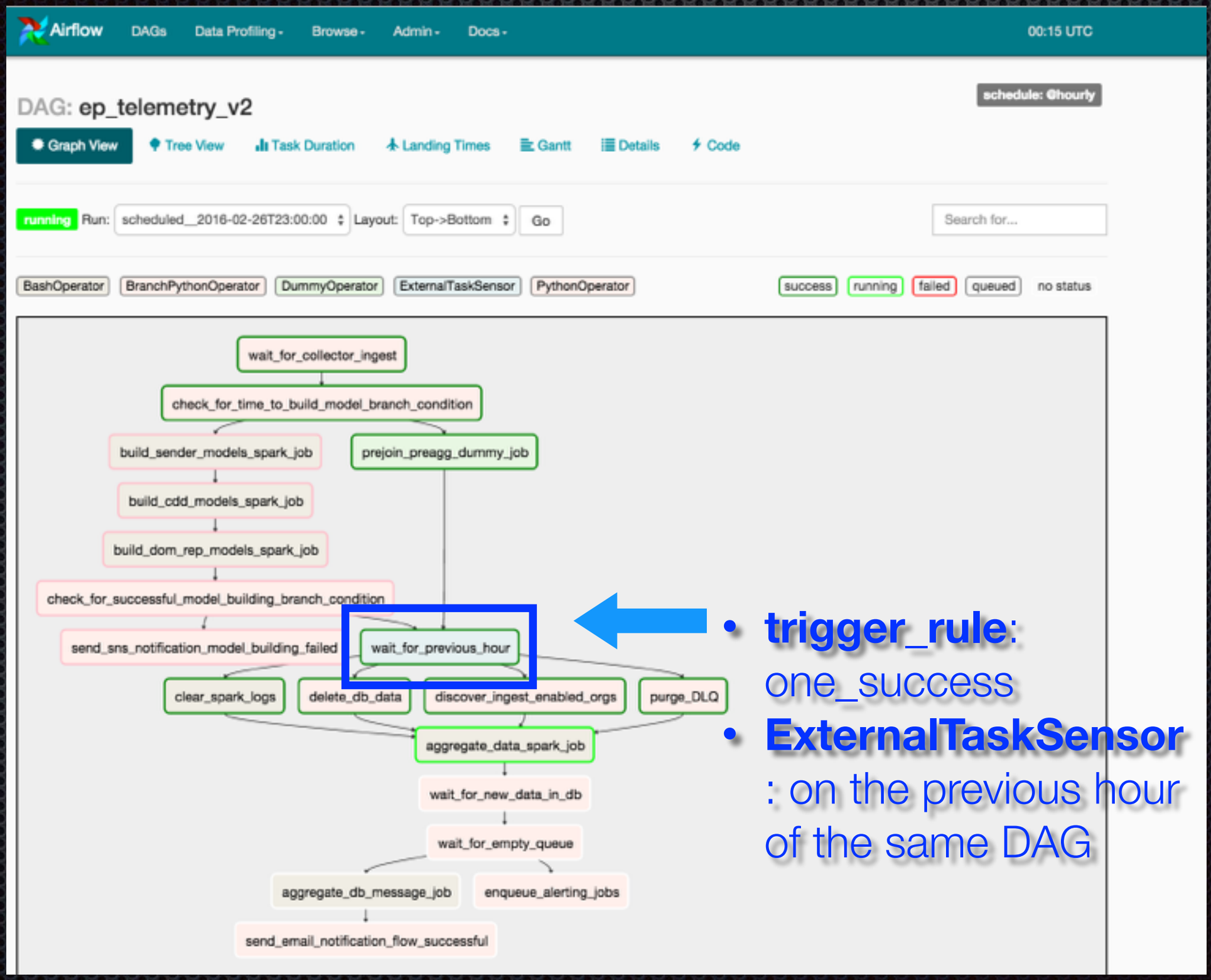


Airflow DAG



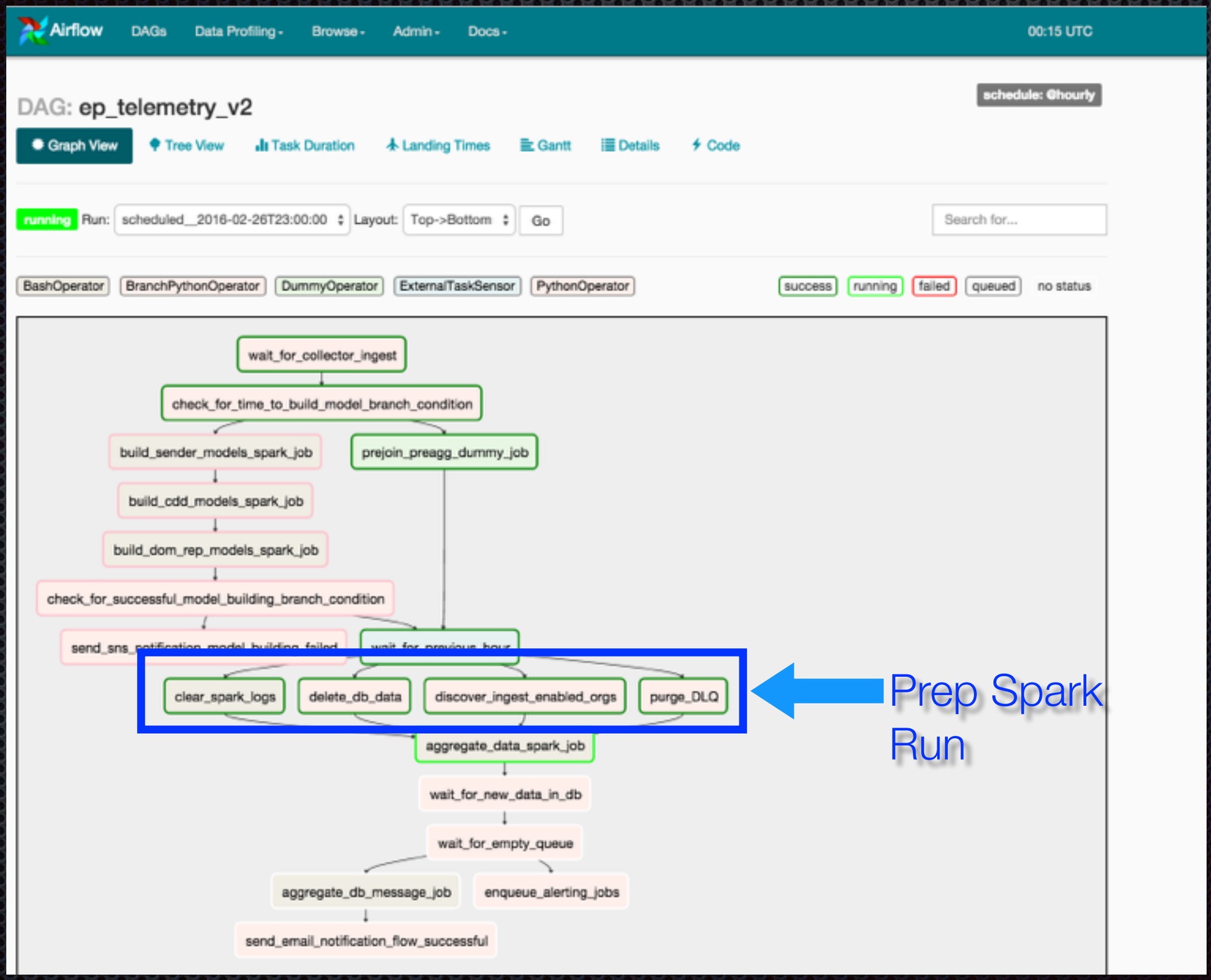
dummy needed for branch operator

Airflow DAG



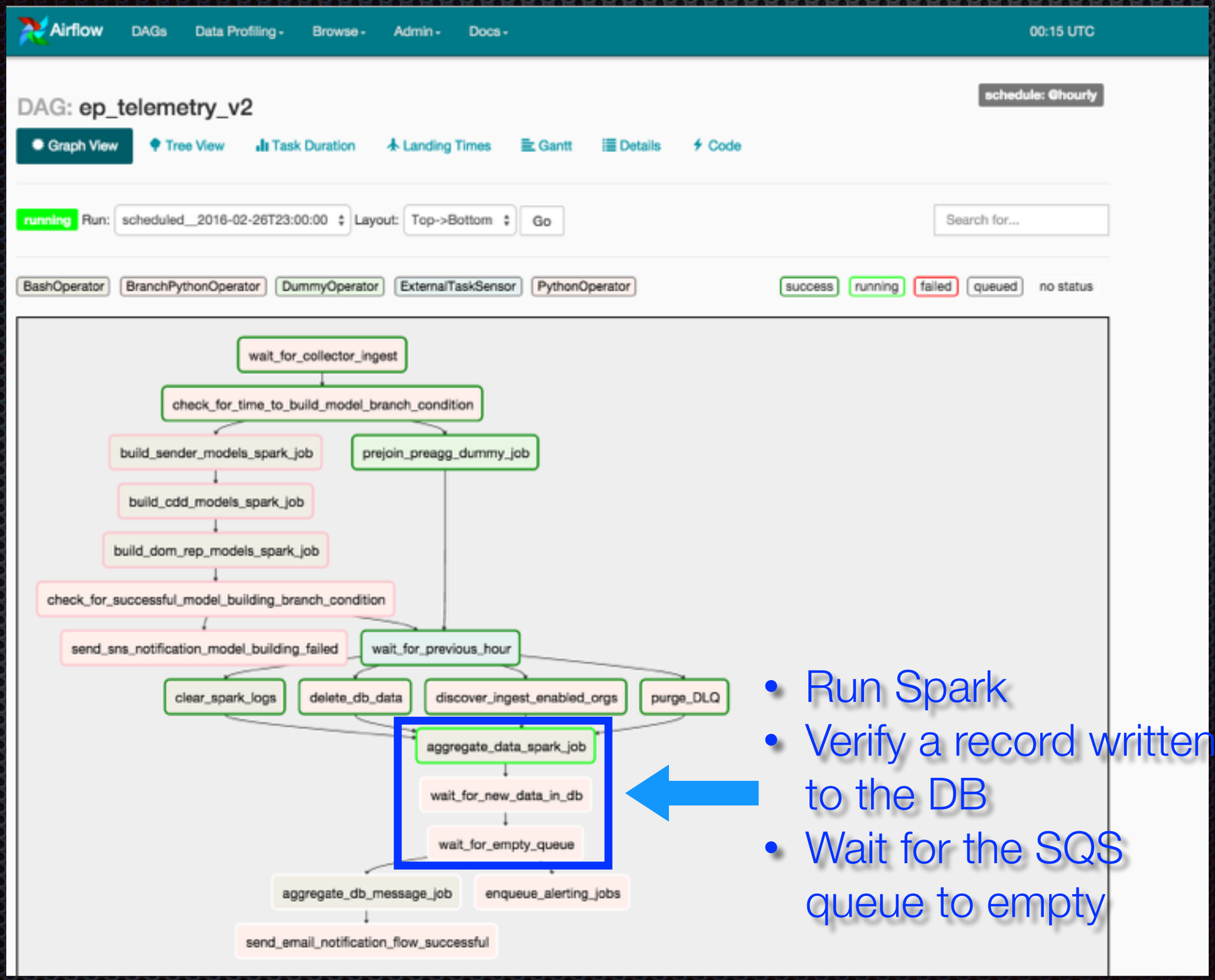
- **trigger_rule**: one_success
- **ExternalTaskSensor** : on the previous hour of the same DAG

Airflow DAG



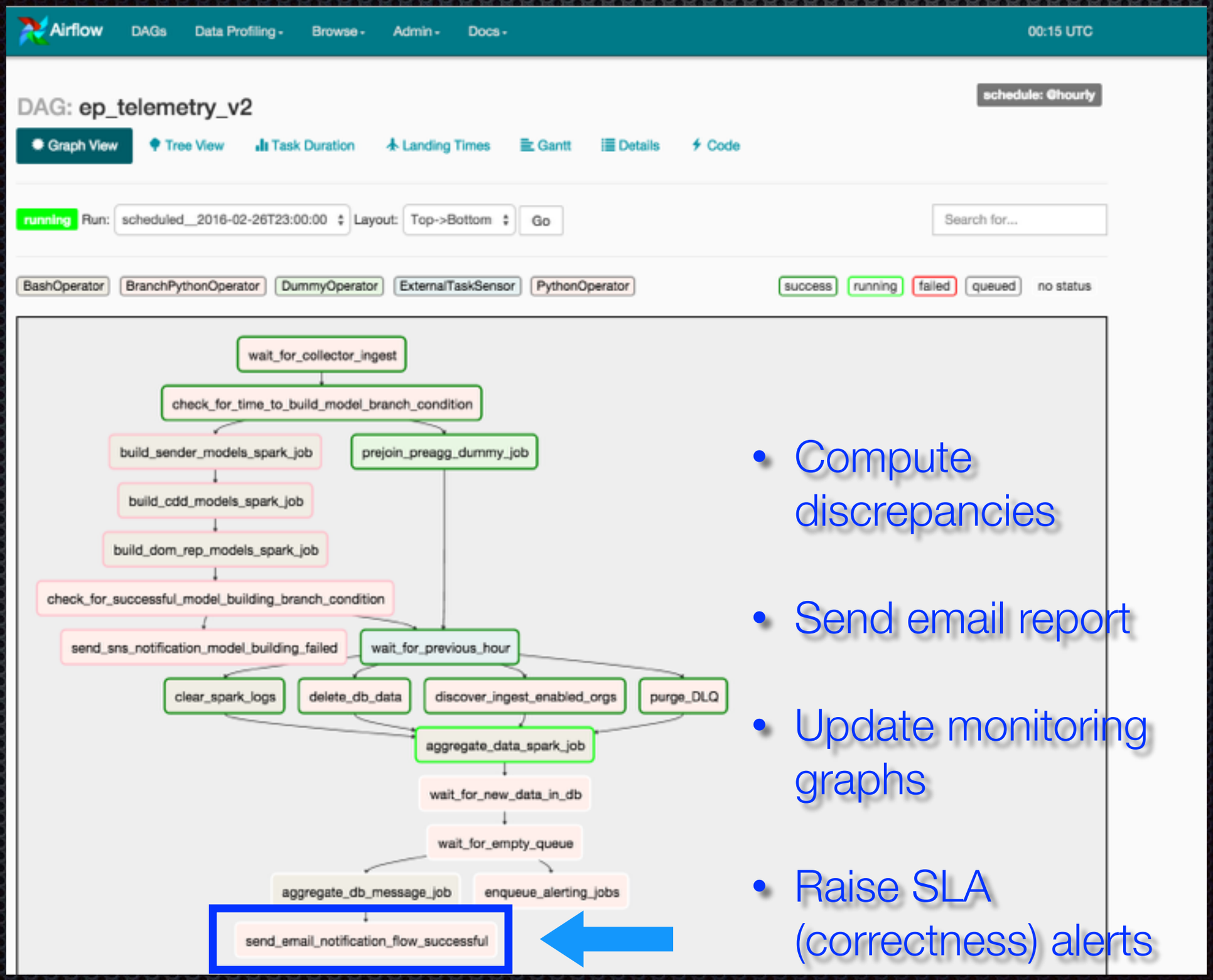
Prep Spark Run

Airflow DAG



- Run Spark
- Verify a record written to the DB
- Wait for the SQS queue to empty

Airflow DAG



- Compute discrepancies
- Send email report
- Update monitoring graphs
- Raise SLA (correctness) alerts

SLAs & Insights

Airflow 

Desirable Qualities of a Resilient Data Pipeline

Correctness

- Data Integrity (no loss, etc…)
- Expected data distributions

Operability

- Fine-grained Monitoring & Alerting of Correctness & Timeliness SLAs
- Quick Recoverability

Timeliness

- All output within time-bound SLAs (e.g. 1 hour)

Cost

- Pay-as-you-go

Desirable Qualities of a Resilient Data Pipeline

Correctness

SLA

- Data Integrity (no loss, etc…)
- Expected data distributions

Operability

- Fine-grained Monitoring & Alerting of Correctness & Timeliness SLAs
- Quick Recoverability

Timeliness

SLA

- All output within time-bound SLAs (e.g. 1 hour)

Cost

- Pay-as-you-go

Correctness : Email Reporting

airflow@agari.com 6:20 PM (16 hours ago)

to AirflowJobComp.

Hi EP Folks! The EP Data Pipeline (Airflow) Loaded Data in the EP_STAGE environment for day: 2016-04-28 12:00:00 GMT/UTC
Airflow

Day	RDAs	Messages
2016-04-28 00:00:00	22058	261254

Message Counts by Org and by Pipeline Stage Output

Org_ID	Collector Msgs	Agg Msgs	DB Msgs	Discrepancy	Discr %	NRT DB Msgs	NRT Discrepancy	NRT Discr %	Skipped	Notes
1	340	266	266	0	0.0	340	-74	-21.8	74	
5	43115	43115	43115	0	0.0	358	42757	99.2	0	
6	156	153	153	0	0.0	0	153	98.1	3	
7	73	68	68	0	0.0	0	68	93.2	5	
12	315	111	111	0	0.0	0	111	35.2	204	
14	35648	28731	28731	0	0.0	0	28731	80.6	6917	
16	35383	34010	34010	0	0.0	0	34010	96.1	1373	
25	20530	20530	20530	0	0.0	0	20530	100.0	0	
54	30003	26190	26190	0	0.0	0	26190	87.3	3813	
56	108084	108080	108080	0	0.0	0	108080	100.0	4	

orgs

For each org, we check for duplicate or missing data as a count & percentage

Correctness : Email Reporting

airflow@agari.com 6:20 PM (16 hours ago)

to AirflowJobComp.

Hi EP Folks! The EP Data Pipeline (Airflow) Loaded Data in the EP_STAGE environment for day: 2016-04-28 12:00:00 GMT/UTC
Airflow

Day	RDAs	Messages
2016-04-28 00:00:00	22058	261254

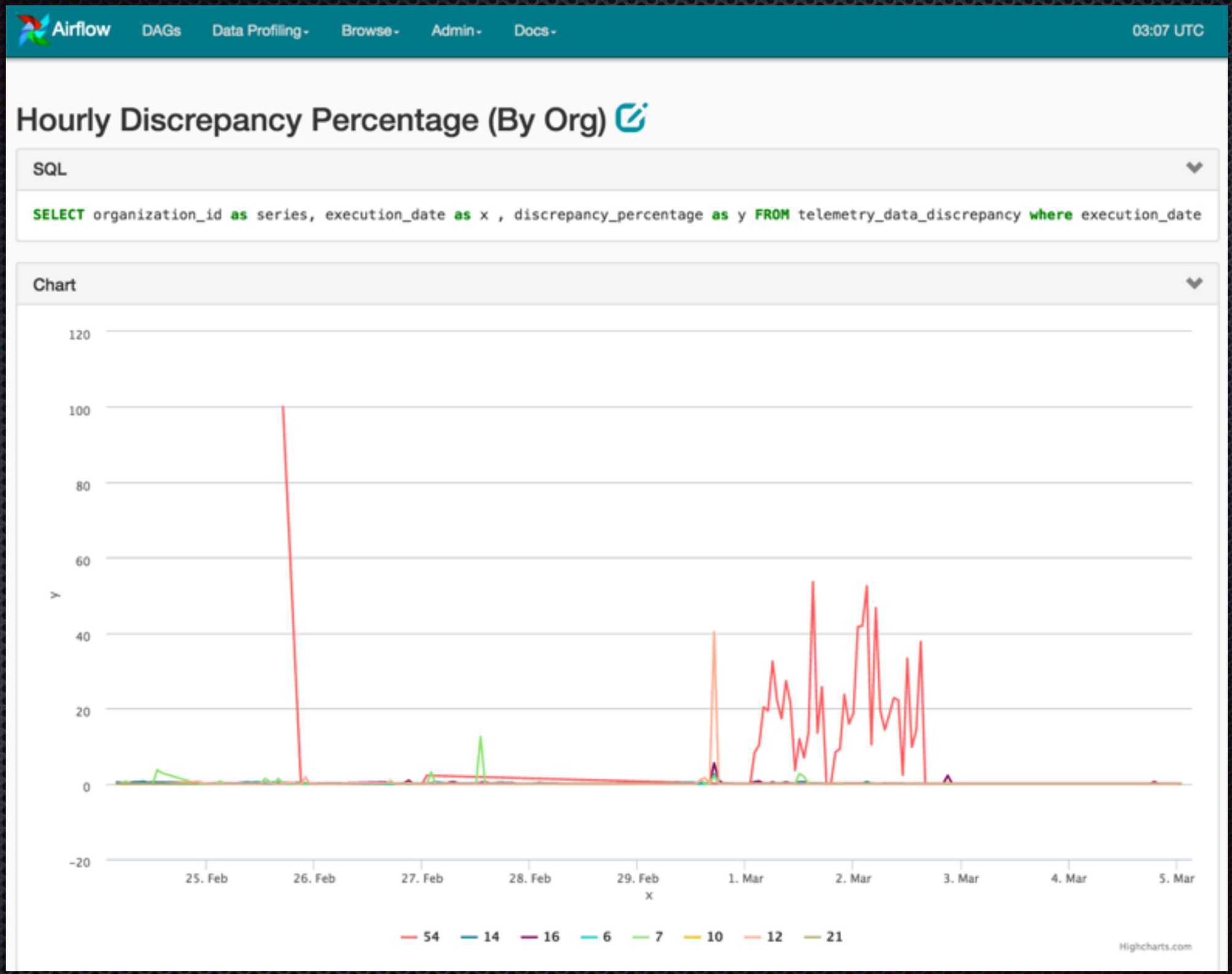
Message Counts by Org and by Pipeline Stage Output

Org_ID	Collector Msgs	Agg Msgs	DB Msgs	Discrepancy	Discr %	NRT DB Msgs	NRT Discrepancy	NRT Discr %	Skipped	Notes
1	340	266	266	0	0.0	340	-74	-21.8	74	
5	43115	43115	43115	0	0.0	358	42757	99.2	0	
6	156	153	153	0	0.0	0	153	98.1	3	
7	73	68	68	0	0.0	0	68	93.2	5	
12	315	111	111	0	0.0	0	111	35.2	204	
14	35648	28731	28731	0	0.0	0	28731	80.6	6917	
16	35383	34010	34010	0	0.0	0	34010	96.1	1373	
25	20530	20530	20530	0	0.0	0	20530	100.0	0	
54	30003	26190	26190	0	0.0	0	26190	87.3	3813	
56	108084	108080	108080	0	0.0	0	108080	100.0	4	

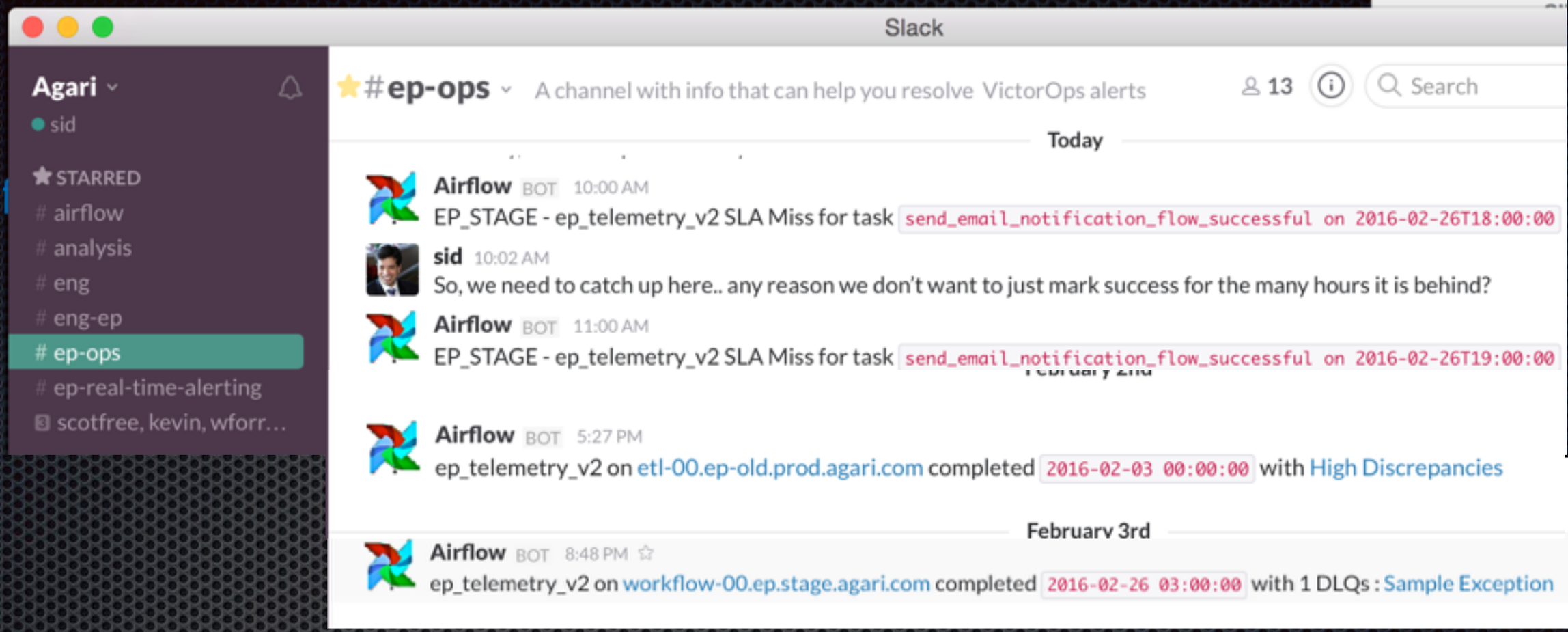
orgs

These are the 3 stages of the pipeline. We can detect where a discrepancy is coming from - often related to a code push!

Correctness : Monitoring

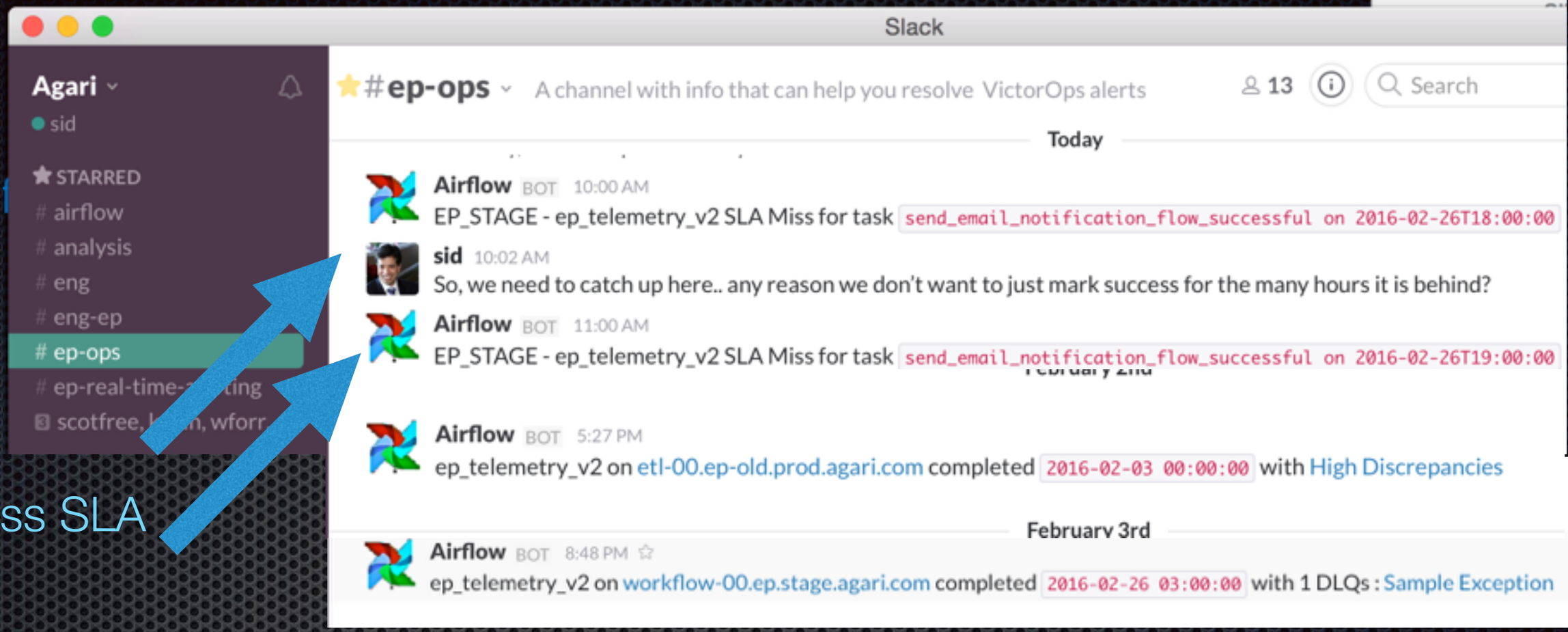


Correctness & Timeliness : Alerting

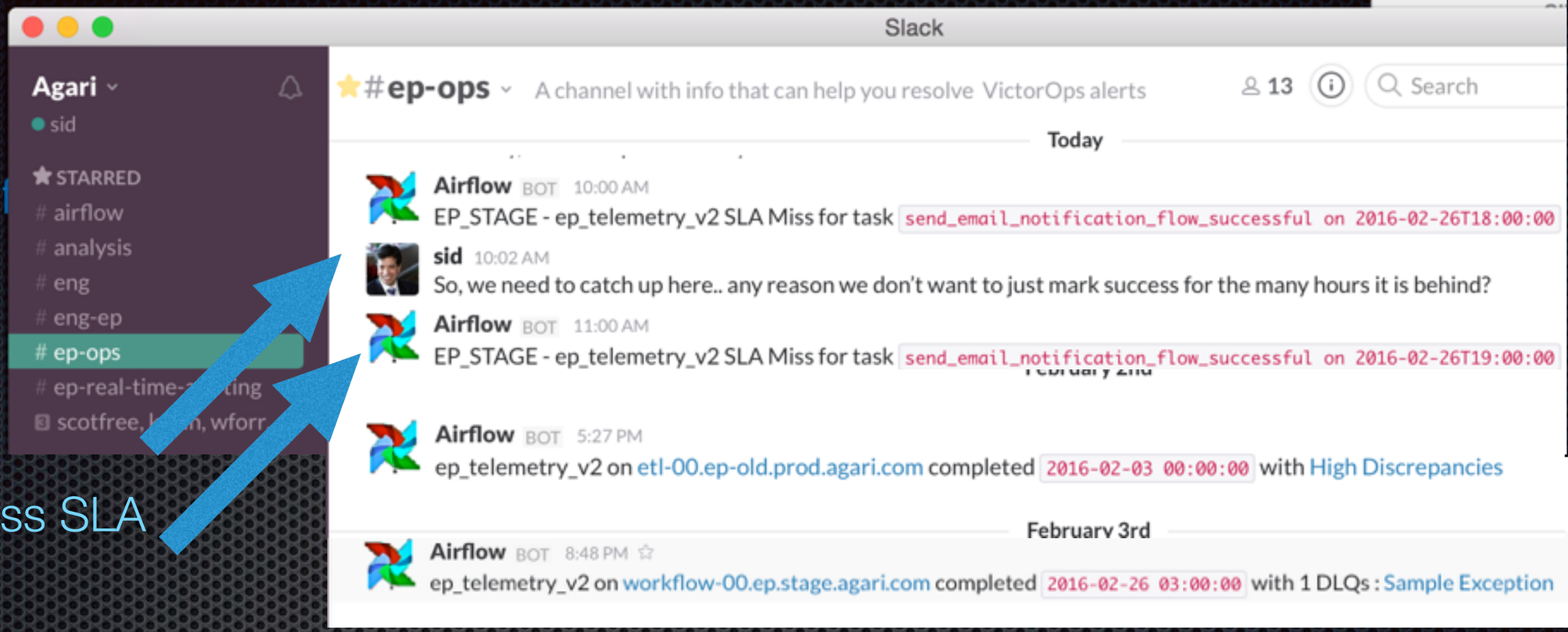


Correctness & Timeliness : Alerting

Timeliness SLA miss



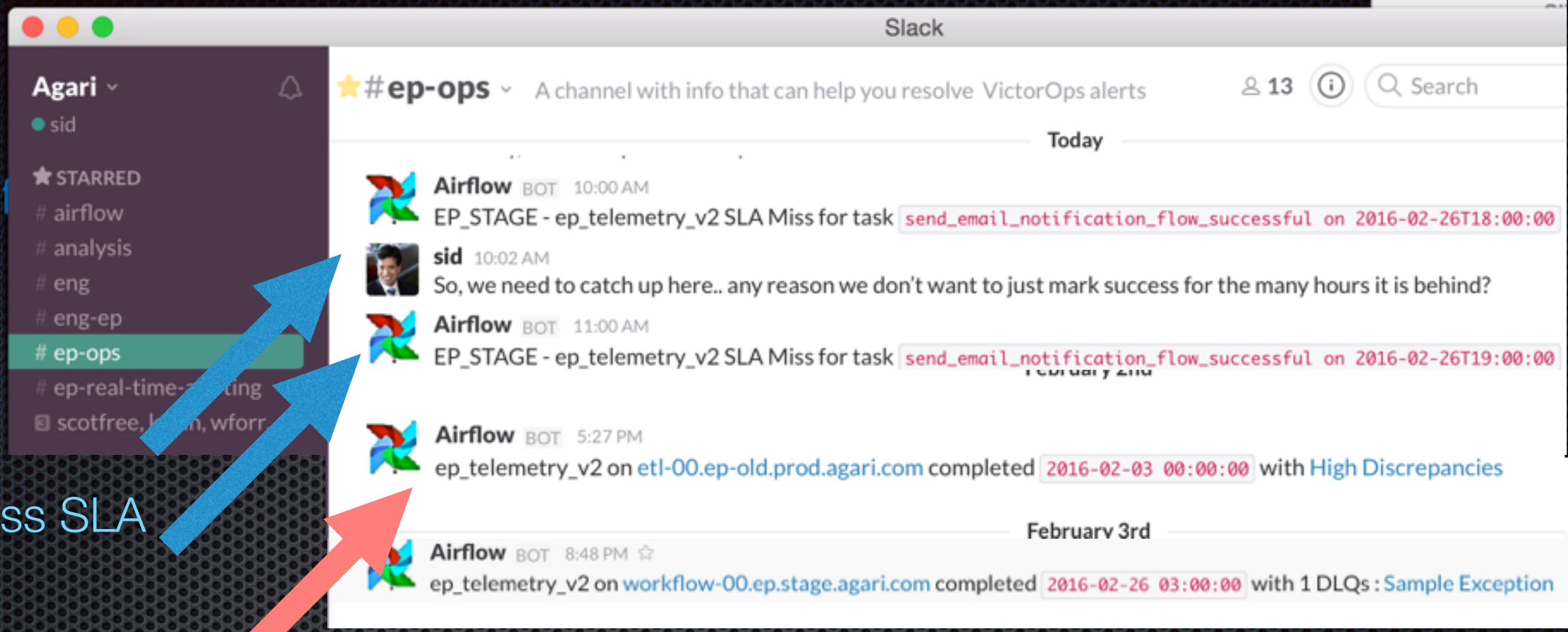
Correctness & Timeliness : Alerting



Timeliness SLA miss

```
dag = DAG(DAG_NAME,
          schedule_interval='@hourly',
          default_args=default_args,
          sla_miss_callback=sla_alert_func)
```

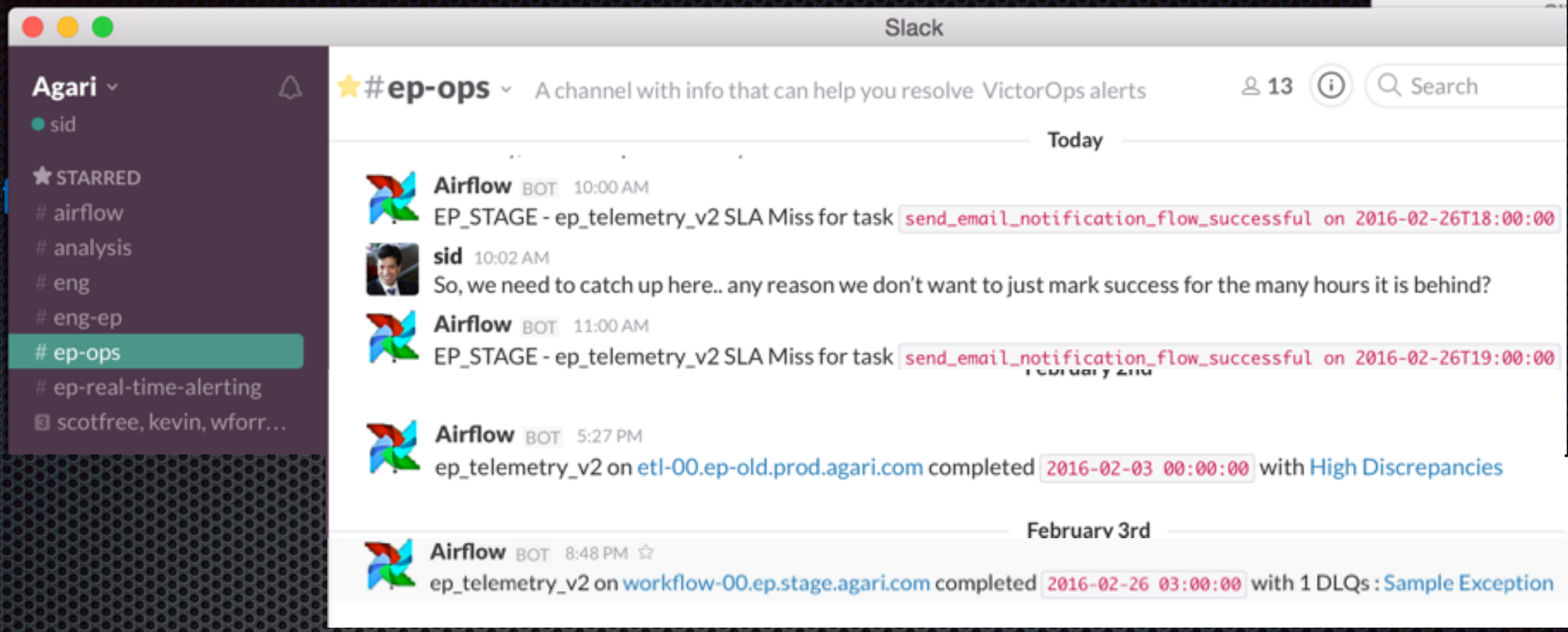
Correctness & Timeliness : Alerting

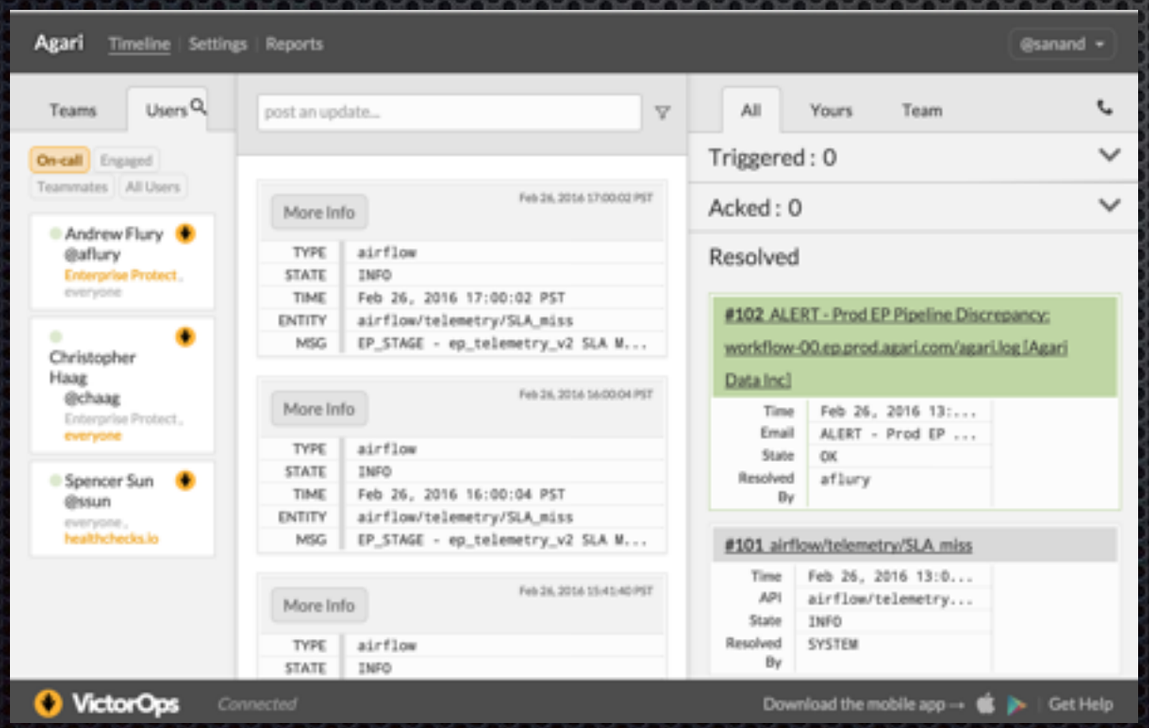


Timeliness SLA miss

Correctness SLA miss

Correctness & Timeliness : Alerting





Timeliness & Correctness SLA misses sent to PagerDuty/VictorOps

Operations

Managing Airflow

Agari in AWS

- Agari runs in the AWS US-West-2 Region & leverages multiple Availability Zones (AZs) for Fault Tolerance

- We have 3 environments (Dev, Stage, Prod), each with its own AWS account

Dev

- A Developer's playground
  - developers prototype new services here

Agari in AWS

- Agari runs in the AWS US-West-2 Region & leverages multiple Availability Zones (AZs) for Fault Tolerance
- We have 3 environments (Dev, Stage, Prod), each with its own AWS account

Dev

- A Developer's playground
  - developers prototype new services here

Stage

- A Pre-Production Env
  - All new code & system upgrades are baked here prior to Prod deployment
  - It gets a copy of prod data & stage data : helps us test for scale!

Agari in AWS

- Agari runs in the AWS US-West-2 Region & leverages multiple Availability Zones (AZs) for Fault Tolerance
- We have 3 environments (Dev, Stage, Prod), each with its own AWS account

Dev

- A Developer's playground
  - developers prototype new services here

Stage

- A Pre-Production Env
  - All new code & system upgrades are baked here prior to Prod deployment
  - It gets a copy of prod data & stage data : helps us test for scale!

Prod

- Prod Env

Agari in AWS

Security

- Each environment has its own AWS account for isolation
- We use Security Groups & VPCs to lock every EC2 instance down
  - In other words, EC2 nodes only accept inbound traffic from other hosts in the same or peered VPCs
- Exception : Each env has one bastion host
  - The bastion host provides indirect access to the nodes in the VPC

Agari's Airflow in AWS

- An Apache Reverse-proxy, running on the bastion host, does:
  - SSH termination &
  - Authentication via mod_auth

- The workflow machine hosts
  - airflow webserver
  - airflow scheduler (running LocalExecutor)

- A Postgresql DB instance hosts the Airflow database

Fault Tolerance

- At Agari, we use Monit to keep processes running on any EC2 instance, e.g. :
  - Apache webserver on the bastion host
  - Airflow webserver & scheduler on the workflow host
  - Postgresql on the DB host
- In Prod, we run 2 schedulers for increased scalability and fault-tolerance

Airflow Github Repos

Apache/
Airflow

r39132/
Airflow
(fork)

Agari/
Airflow
(fork)

Airflow Github Repos

Apache/
Airflow

r39132/
Airflow
(fork)

Agari/
Airflow
(fork)

Deployed to
stage & prod

Airflow Github Repos

Apache/
Airflow

r39132/
Airflow
(fork)

Agari/
Airflow
(fork)

Agari / Airflow pinned
to a release version
(e.g. 1.6.2)

Airflow Github Repos

r39132 / Airflow kept in sync upstream master

Apache/ Airflow

r39132/ Airflow (fork)

Agari/ Airflow (fork)

Airflow Github Repos

r39132 / Airflow sends PR 191

Airflow Github Repos

Apache/
Airflow

r39132/
Airflow
(fork)

Agari/
Airflow
(fork)

PR 191 is cherry-picked into Agari / Airflow

Airflow Github Repos

Apache/
Airflow

r39132/
Airflow
(fork)

Agari/
Airflow
(fork)

Deployed to
stage & prod

What is deployed is 1.6.2 + cherry-picked (e.g. PR 191) changes

Acknowledgments

None of this work would be possible without the contributions of the strong team below

- Vidur Apparao
- Stephen Cattaneo
- Jon Chase
- Andrew Flury
- William Forrester
- Chris Haag
- Mike Jones
- Scot Kennedy
- Thede Loder
- Paul Lorence
- Kevin Mandich
- Gabriel Ortiz
- Jacob Rideout
- Josh Yang
- Julian Mehnle

Questions? (@r39132)