FROM AIRFLOW IMPORT DAG

Airflow the perfect match in our Analytics Pipeline

Sergio Camilo Fandiño Hernández
Senior Business Intelligence Architect @LOVOO

Airflow Summit 2020

AGENDA

1. Why we met?
2. How we met?
3. The first date!
4. Fun dates!
5. Is there any dynamic in between?
6. Recap and conclusion

About LOVOO

- LOVOO is a dating and social app and the place for chatting, live streaming, watching streams and getting to know people.
- Germany - Dresden & Berlin - 2011
- Acquired by The Meet Group (NASDAQ:MEET) in 2017
- Top 3 Dating App in Europe
- + 280 TB of Data
- ~ 6 TB Monthly Growth
- + 3 TB daily total aggregated data
- + 36 TB Swipes (162,824,303,474)

Analytics

- 1 Head
- 6 Data Analysts
- 2 BI Architects

- Product
- Finance
- Marketing
- Talent Management
- Customer Insights
- CRM

What can you expect?

My main purpose today is to tell you about our journey with Airflow as well as a few different use cases that could also boost the work of your Analytics/BI team on a daily basis.

- Pieces of code (examples)
- Way too many screenshots

AGENDA

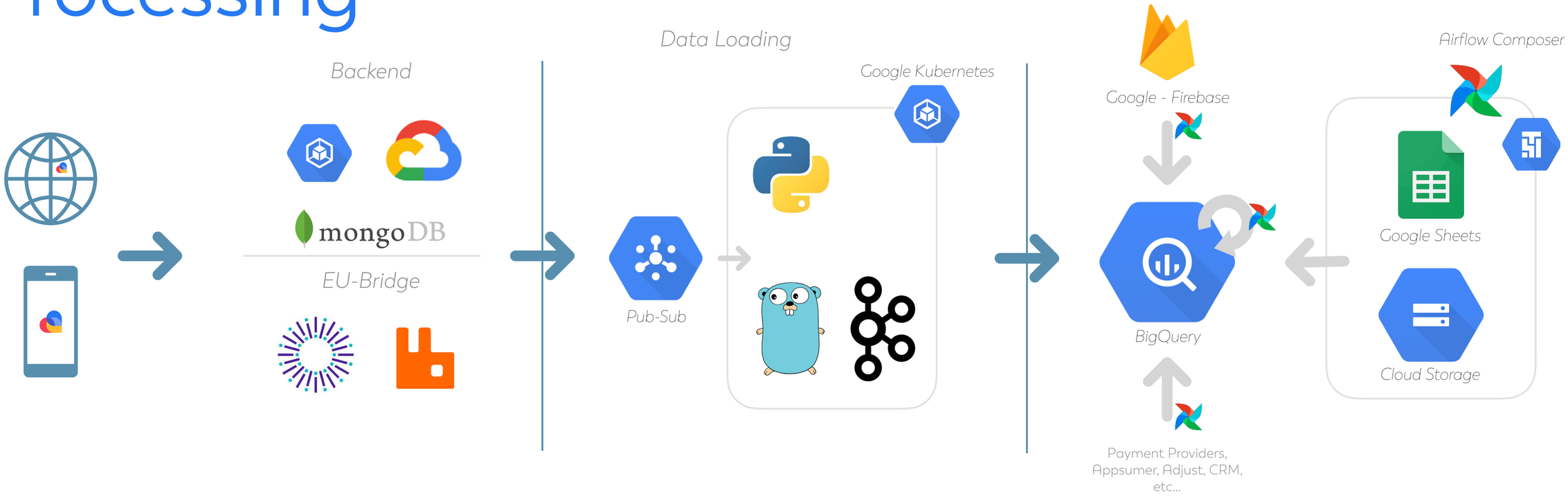
1. Why we met?
2. How we met?
3. The first date!
4. Fun dates!
5. Is there any dynamic in between?
6. Recap and conclusion

On-premise

We went Cloud

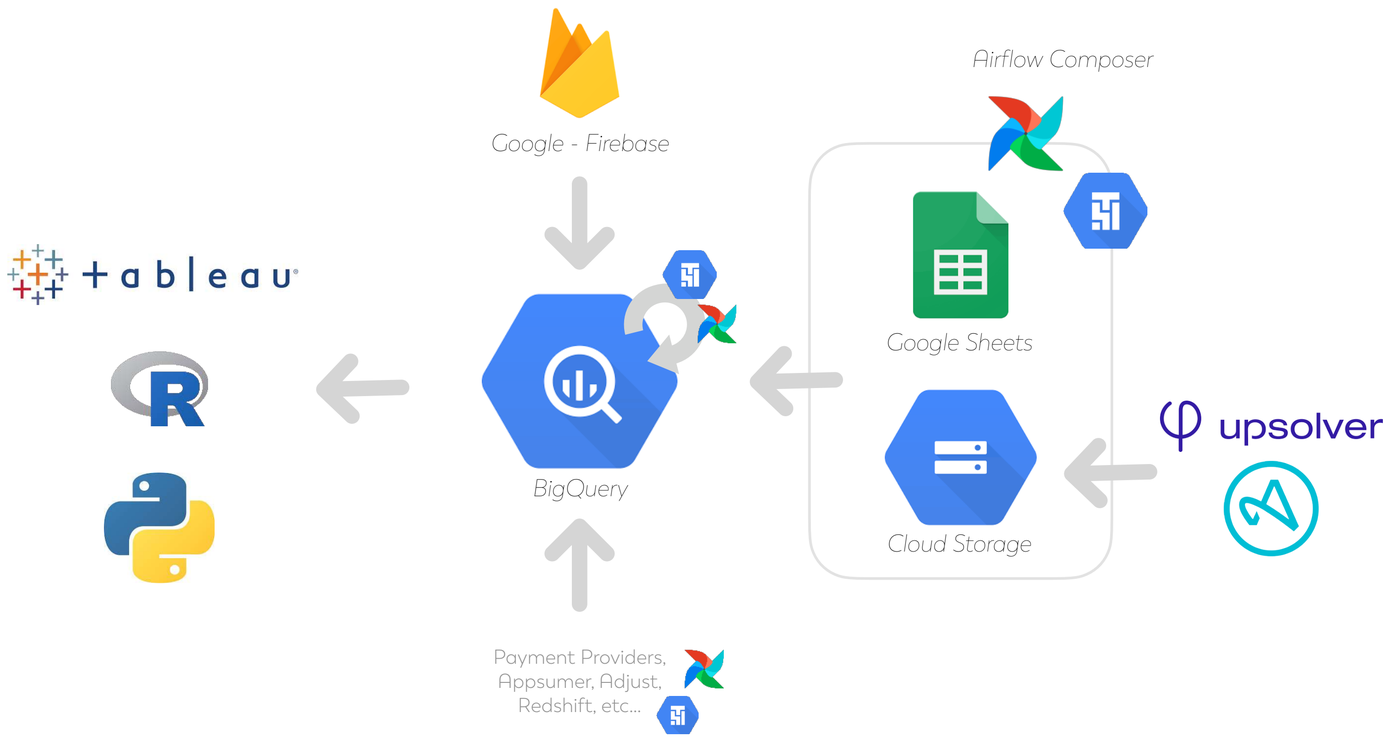
Sergio Camilo Fandiño Hernández
Senior Business Intelligence Architect @LOVOO

Data Processing



Sergio Camilo Fandiño Hernández
Senior Business Intelligence Architect @LOVOO

Analytics Data-Core



Sergio Camilo Fandiño Hernández
Senior Business Intelligence Architect @LOVOO

AGENDA

1. Why we met?
2. How we met?
3. The first date!
4. Fun dates!
5. Is there any dynamic in between?
6. Recap and conclusion

Orchestration Tool

- Identify what is out there
- Costs?
- Scalability?
- Data sources compatibility?
- Knowledge/Human Resources?

Airflow

- Great community
- Game changer
- Mobile App
- Python
- BigQuery

Google Cloud Composer

- Fully Managed Airflow
- Scalable
- IAP - Secure
- Focus on building the Analytics data pipeline
- Ease of implementation

Google Cloud Composer

- Fully Managed Airflow

Alpha

This is an alpha release of Cloud Composer. This product might be changed in backward-incompatible ways and is not recommended for production use. It is not subject to any SLA or deprecation policy. This product is not intended for real-time usage in critical applications.

- Focus on building the Analytics data pipeline
- Ease of implementation

Confidential Material: This page is confidential. Do not share or discuss until authorized to do so.

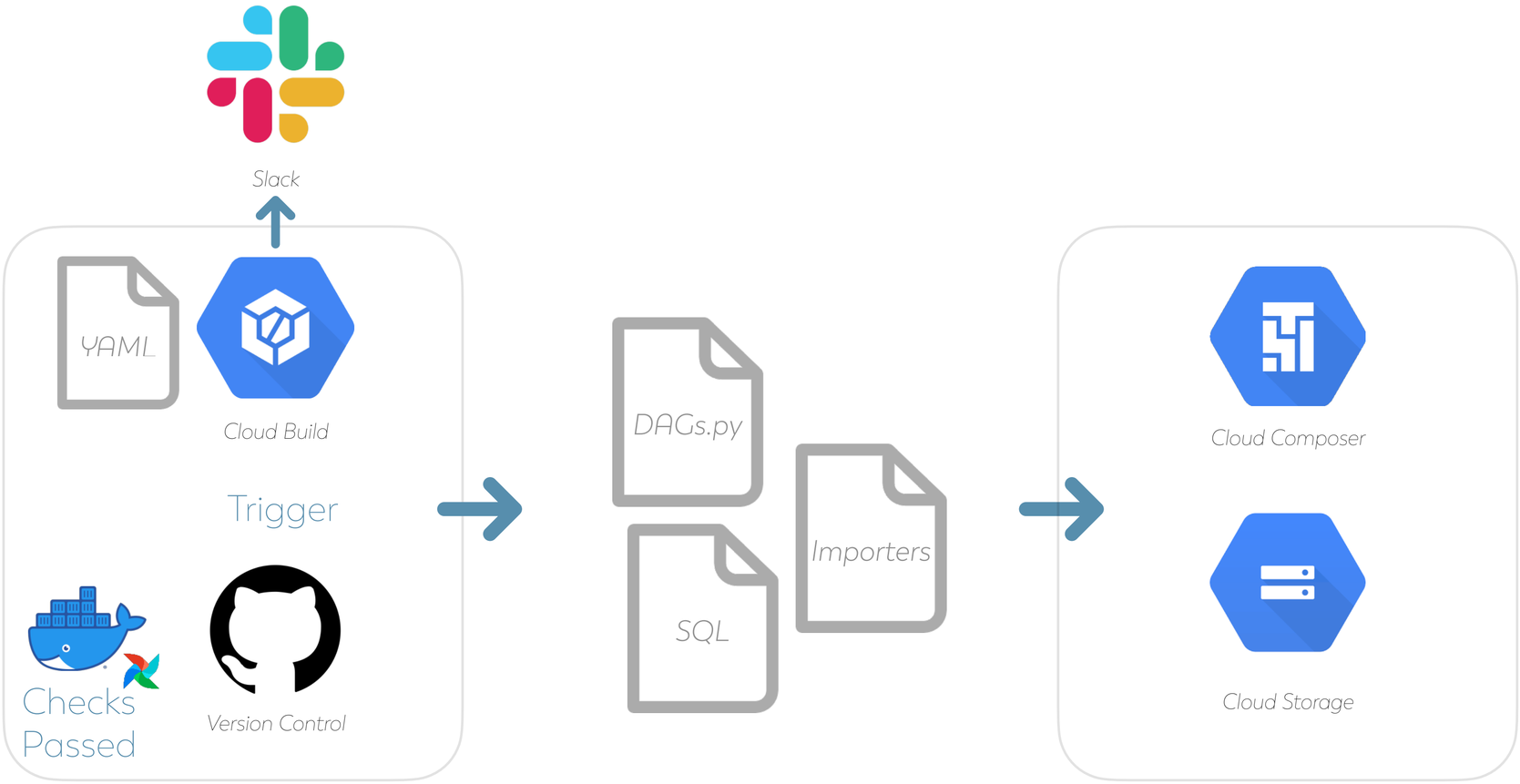
AGENDA

1. Why we met?
2. How we met?
3. The first date!
4. Fun dates!
5. Is there any dynamic in between?
6. Recap and conclusion

TODO List

- SQL Scripts —> Data Modeling
- DAGs
- Permissions - Service Accounts
- Data Importers
- Create a Composer Environment
- How do we deploy? —> CI/CD

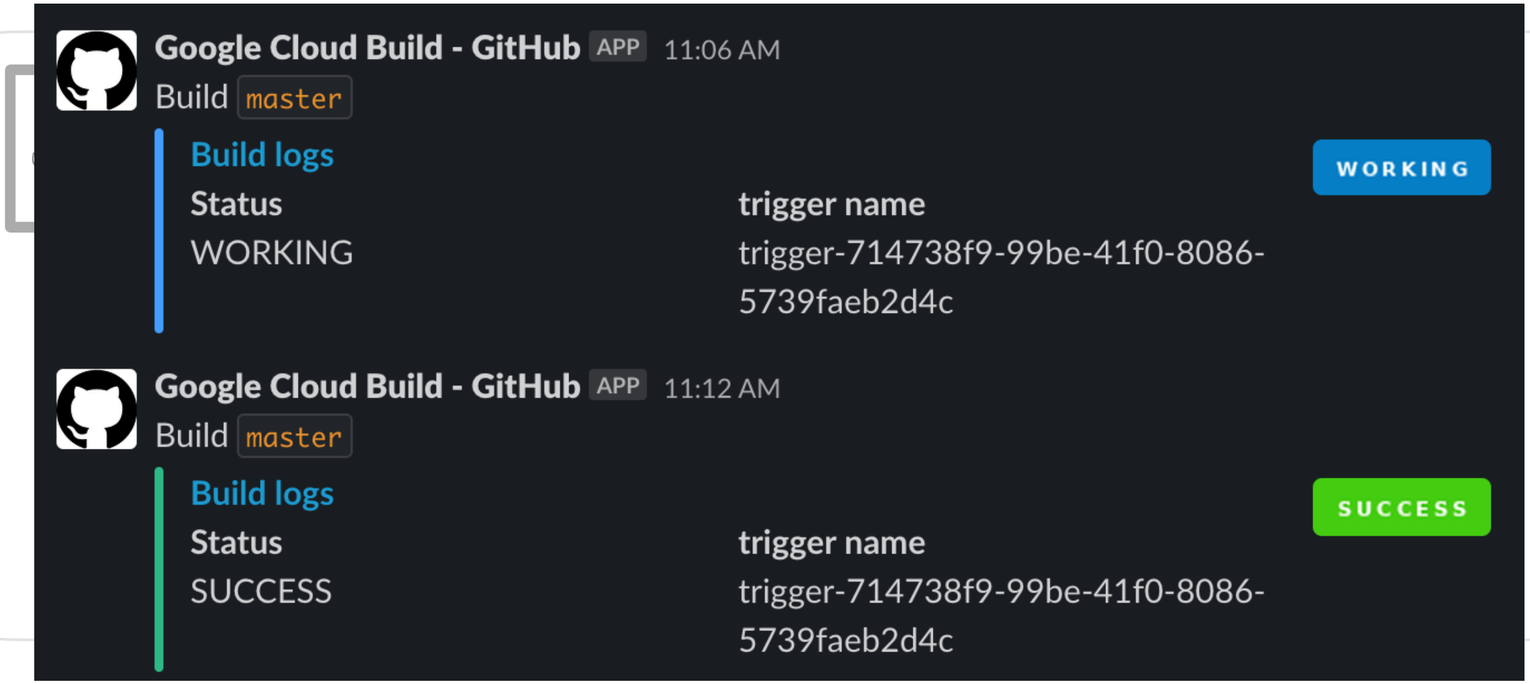
CI/CD



Sergio Camilo Fandiño Hernández
Senior Business Intelligence Architect @LOVOO

CI/CD

Slack



Sergio Camilo Fandiño Hernández
Senior Business Intelligence Architect @LOVOO

AGENDA

1. Why we met?
2. How we met?
3. The first date!
4. Fun dates!
5. Is there any dynamic in between?
6. Recap and conclusion

DAGs

DummyOperator SlackAPIPostOperator SubDagOperator TimeRangeExternalTaskSensor

BashOperator BigQueryCheckOperator BigQueryOperator PythonOperator BranchPythonOperator

Operators

- 26 DAGs
- Sub-DAGs
- Branching
- Jinja Templating
- Hooks
- Pools
- Trigger rules

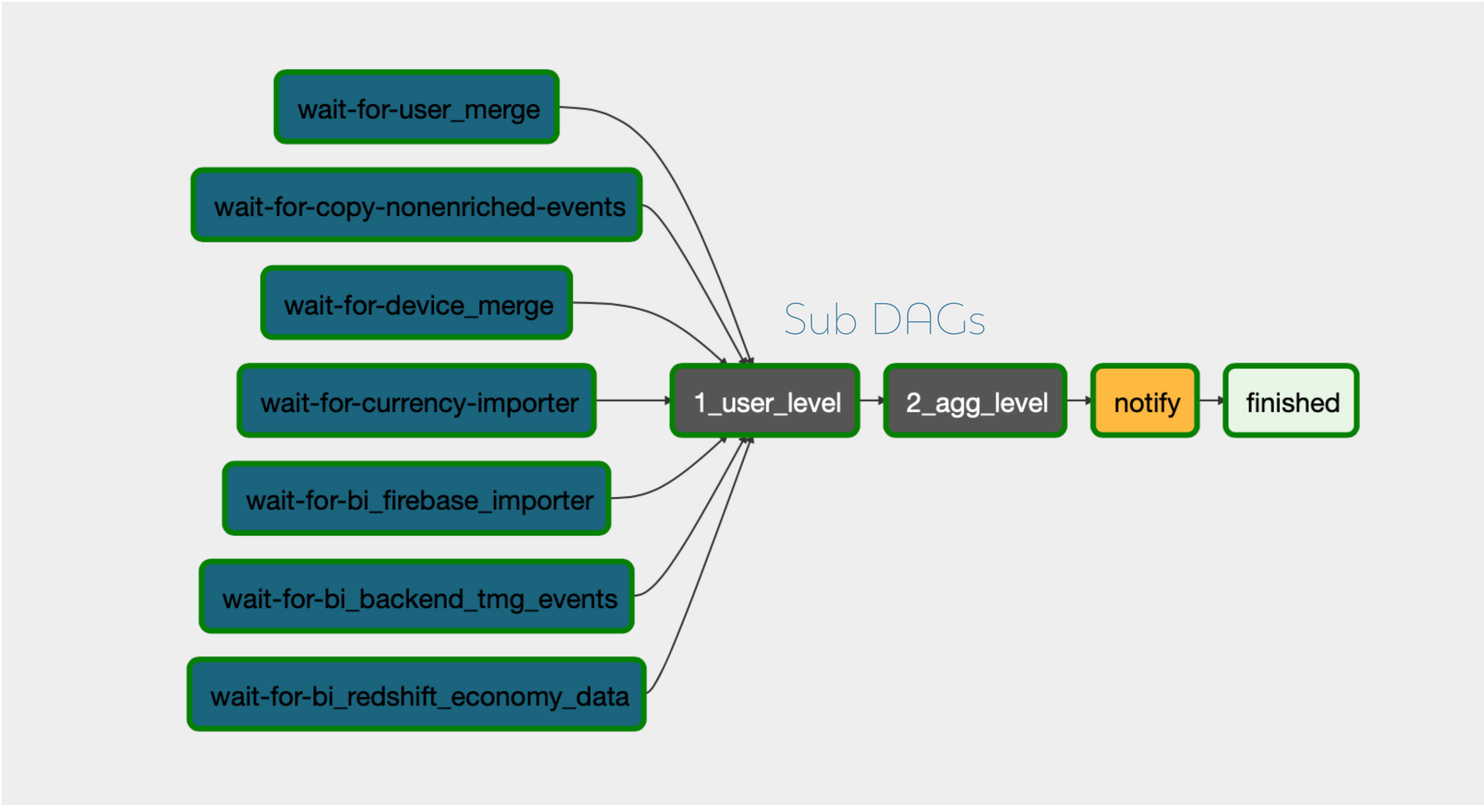
		DAG	Schedule
	On	adevents-repair	0 4 * * *
	On	airflow_monitoring	None
	On	analytics_jobs	0 5 * * *
	On	analytics_jobs_live	0 5 * * *
	On	antispam-creditfarm-detection	@daily
	On	antispam-reputation-modeltraining	@daily
	On	appsumer-importer	00 11 * * *
	On	appsumer-importer-hayi	00 12 * * *
	On	bi_backend_tmg_events	30 2 * * *
	On	bi_data_check	40 3 * * *
	On	bi_firebase_importer	40 4 * * *
	On	bi_firebase_live_events	40 4 * * *
	On	bi_marketing_events_jobs	50 6 * * *
	On	bi_marketing_jobs	50 9 * * *
	On	bi_payment_provider_apis	40 4 * * *
	On	bi_redshift_economy_data	30 4 * * *

TimeRangeExternalTaskSensor

SubDagOperator

The Core

Analytics - Workflow



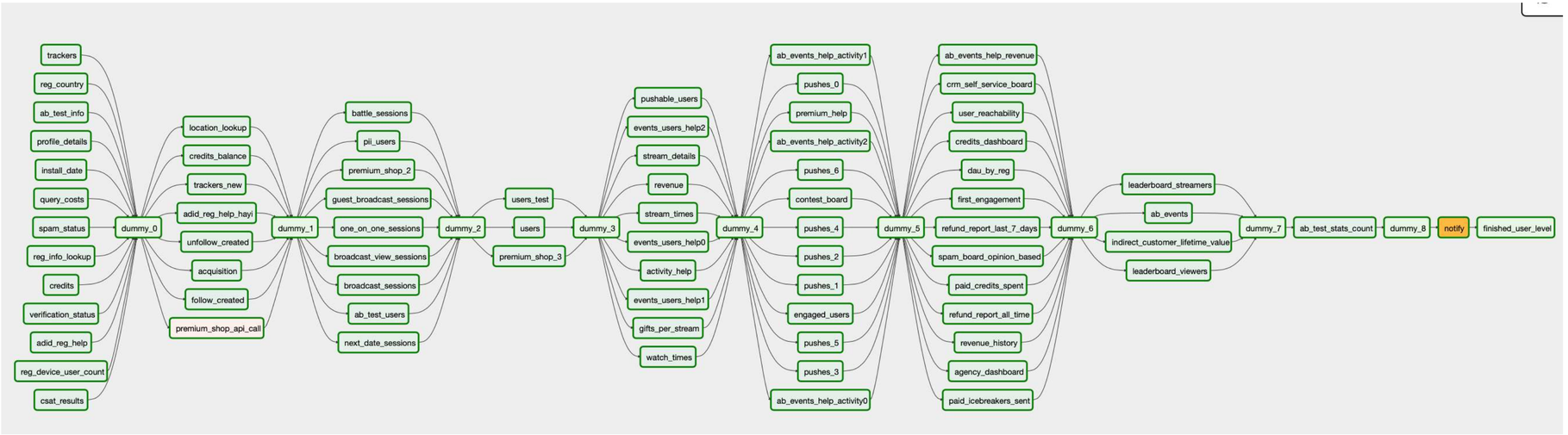
Sergio Camilo Fandiño Hernández
Senior Business Intelligence Architect @LOVOO

Airflow Summit 2020

BigQueryOperator

The Core

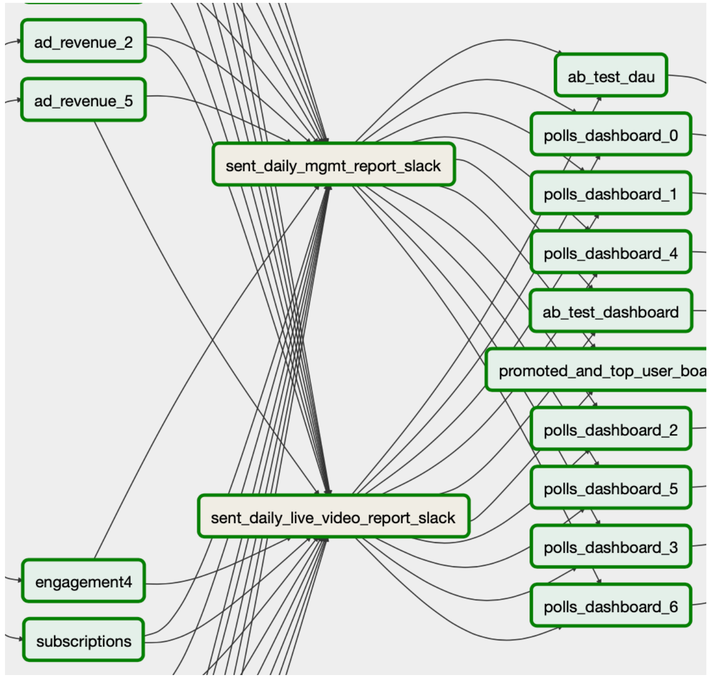
Sub DAG



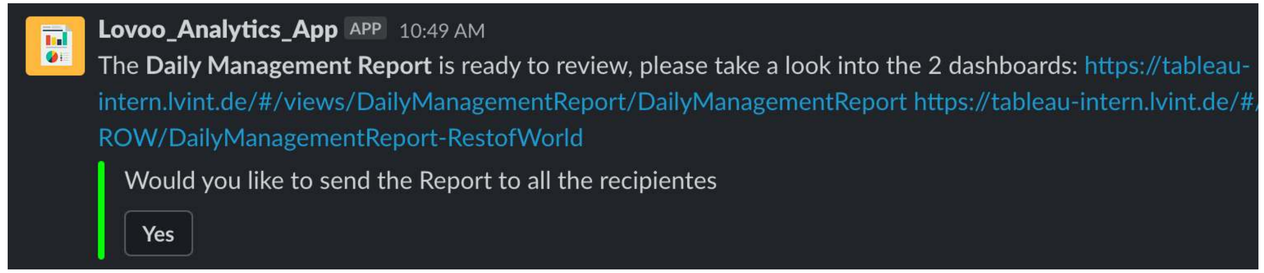
Airflow Summit 2020

BashOperator

Reports!



Slack Webhook

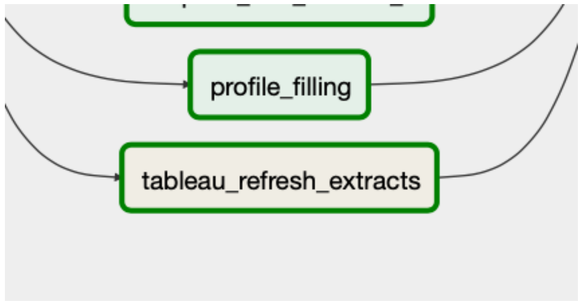


tableau

Sergio Camilo Fandiño Hernández
Senior Business Intelligence Architect @LOVOO

BashOperator

Tableau Extracts



bash_command

```
1 curl -X POST http://35.205.226.12:8007
```

tableau

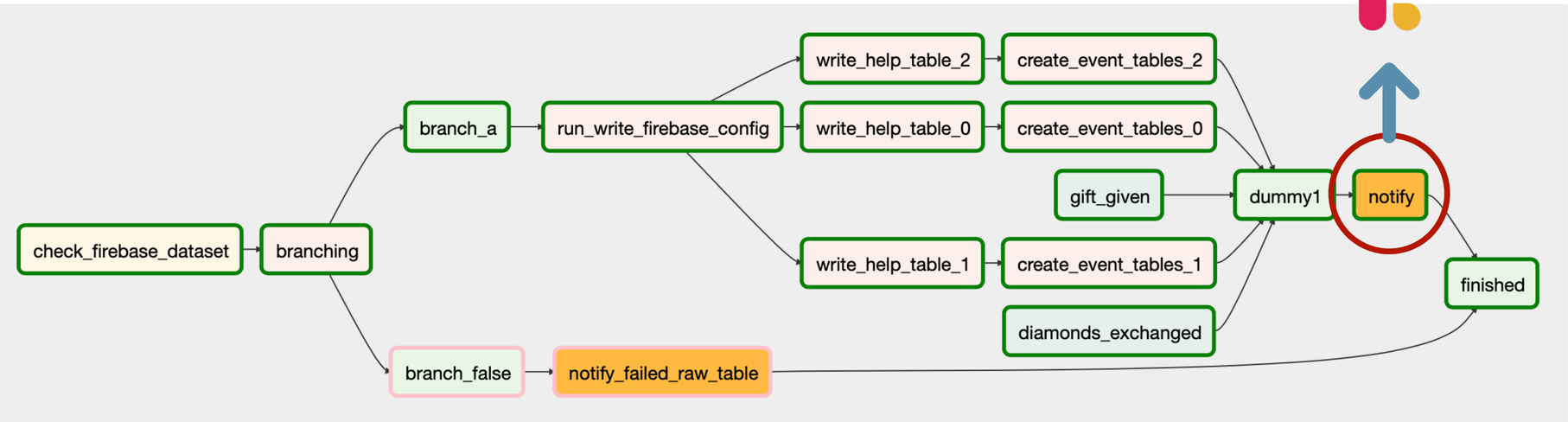
```
6 $tabcmd runschedule "Daily Extract Refreshes (9:00 AM)"
```

Airflow Summit 2020

SlackAPIPostOperator

Is Airflow finished?

by the way, this is branching...



Airflow Summit 2020

SlackAPIPostOperator

Is Airflow finished?

airflow-bot APP 6:47 AM
bi_firebase_live_events: Finished

airflow-bot APP 6:53 AM
bi_firebase_importer: Finished

airflow-bot APP 7:04 AM
1/2 `analytics_jobs_live-1_user_level_live`: Finished
2/2 `analytics_jobs_live-2_agg_level_live`: Finished
Analytics Live Pipeline Completed `analytics_jobs_live`: Finished

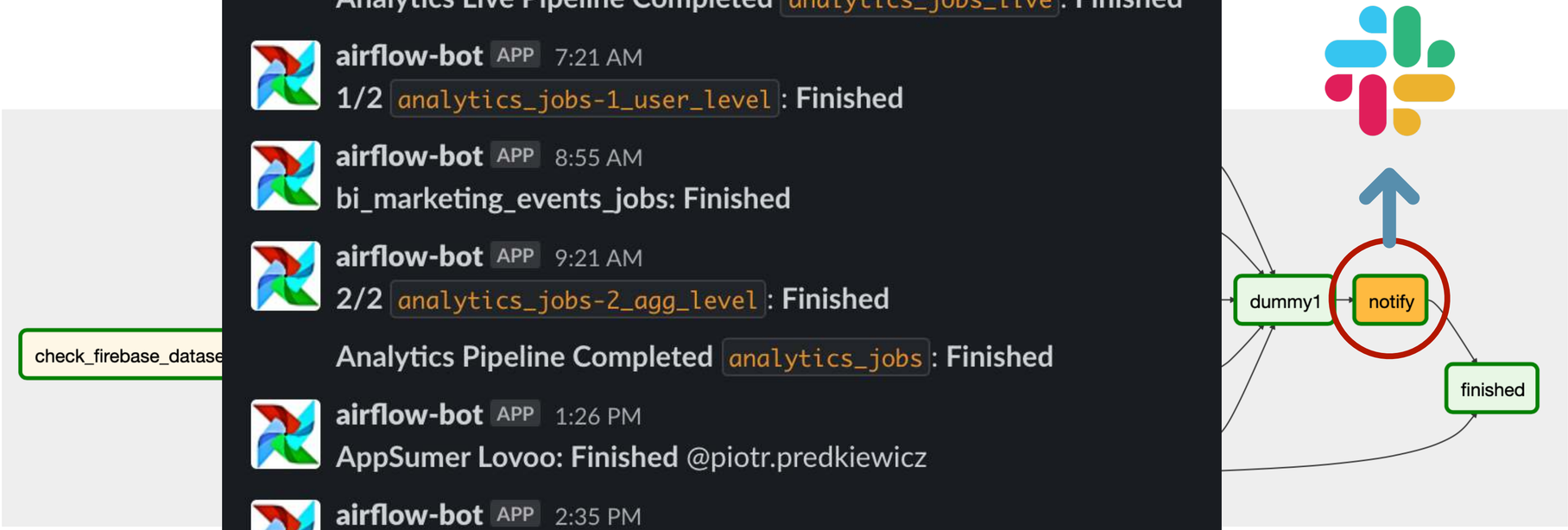
airflow-bot APP 7:21 AM
1/2 `analytics_jobs-1_user_level`: Finished

airflow-bot APP 8:55 AM
bi_marketing_events_jobs: Finished

airflow-bot APP 9:21 AM
2/2 `analytics_jobs-2_agg_level`: Finished
Analytics Pipeline Completed `analytics_jobs`: Finished

airflow-bot APP 1:26 PM
AppSumer Lovoo: Finished @piotr.predkiewicz

airflow-bot APP 2:35 PM
AppSumer Hayi: Finished @piotr.predkiewicz



Error Alerting

```
'on_failure_callback': on_failure_callback,
```


```
def on_failure_callback(context):
    operator = SlackAPIPostOperator(
        task_id='notify_fail',
        channel="#the_channel",
        token='your_Slack_bot_token',
        username='airflow-bot',
        text= str('*Airflow requires an action* {} Task: {}').format(
            str(context['dag']), str(context['task_instance']))
    )
    return operator.execute(context=context)
```

Airflow Summit 2020

PythonOperator

Integrating Data Sources

this code belongs to the DAG.py file

```
t1a = PythonOperator(
    task_id='load_table_lovoo_transaction_groups_{}'.format(i),
    python_callable=import_day_callable,
    provide_context=True,
    templates_dict={'exec_date': exec_date, 'table_name':'lovoo_transaction_groups'},
    dag=dag)
```

Airflow Summit 2020

PythonOperator

Integrating Data Sources

this code belongs to the DAG.py file

```
from BI.redshift_importer import import_datalake_redshift_data

def import_day_callable(**kwargs):
    exec_date = kwargs.get('templates_dict').get('exec_date')
    table_name = kwargs.get('templates_dict').get('table_name')
    return import_datalake_redshift_data(table_name,
                                         'load_job_dataframe_to_bq',
                                         exec_date=exec_date)
```

Airflow Summit 2020

PythonOperator

Integrating Data Sources

this code belongs to the importer.py file

```
def postgreSQL_connection():
    try:
        # Using a Hook for getting the Redshift credentials from the Airflow connections -
        connection = BaseHook.get_connection("redshift_tmg")
        password = connection.password
        host = connection.host
        dbname = connection.schema
        user = connection.login
        port = connection.port

        conn = psycopg2.connect("dbname='{}' user='{}' port='{}' host='{}' password='{}'".f

        cursor = conn.cursor()

    except Exception as e:
        print("I am unable to connect to the database: " + str(e))

    return cursor,conn
```

Sergio Camilo Fandiño Hernández
Senior Business Intelligence Architect @LOVOO

Airflow Summit 2020

PythonOperator

Integrating Data Sources

this pseudo-code belongs to the importer.py file

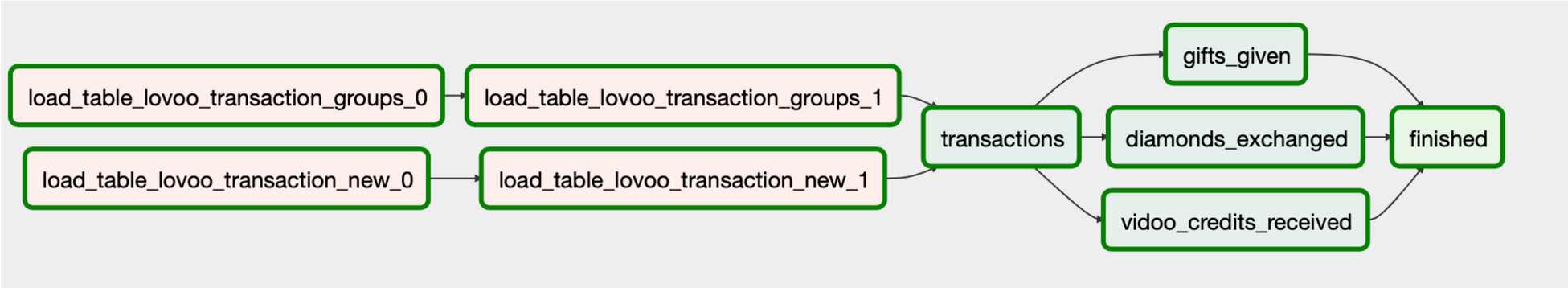
```
def import_datalake_redshift_data(table_name, method_type, exec_date, **kwargs):

    # Cursor & Connection
    cursor,conn = postgreSQL_connection()

    - Create dynamically a SQL query using the input parameters table_name and exec_date
    query = "select * from a_datalake.{} where data_updated_at::date >= '{}'".format(table_name, exec_date)

    - Use the query to request the data using the cursor
    cursor.execute(query)

    - use any method to upload the data to BigQuery
    df = cursor.fetchall()
    df = pd.DataFrame(df)
    job = client.load_table_from_dataframe(
    df, table_name, job_config=job_config
    )
    return whether it was successful or not
```

Airflow Summit 2020

PythonOperator

Integrating Data Sources

2 Tables - 2 Days -> ELT in BQ



Sergio Camilo Fandiño Hernández
Senior Business Intelligence Architect @LOVOO

PythonOperator

Data Importers

- Redshift
- Firebase (very dynamic)
- Google Cloud Storage (Adjust, Merger)
- Appsumer, Shopify, Paypal, AppStore, Adyen
- S3 Storage

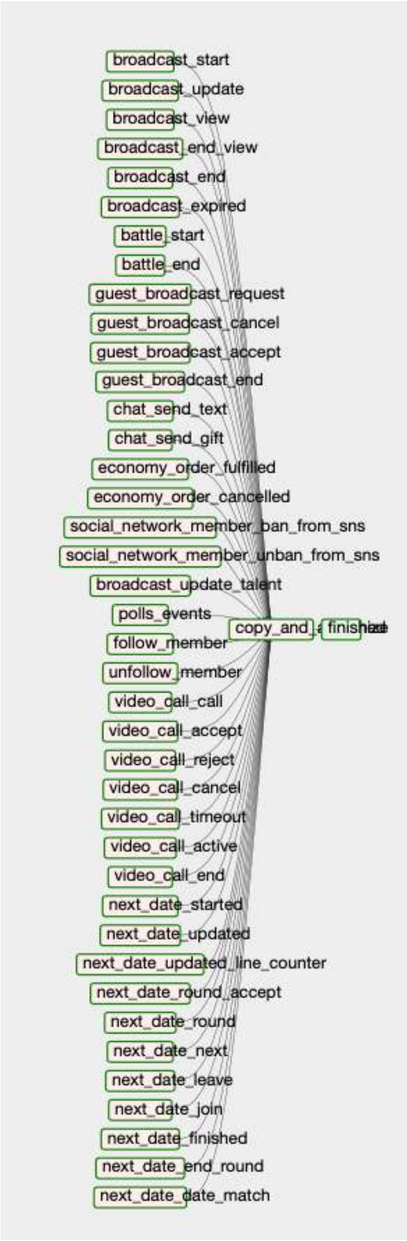
AGENDA

1. Why we met?
2. How we met?
3. The first date!
4. Fun dates!
5. Is there any dynamic in between?
6. Recap and conclusion

Airflow Summit 2020

PythonOperator

Creating Tasks Dynamically

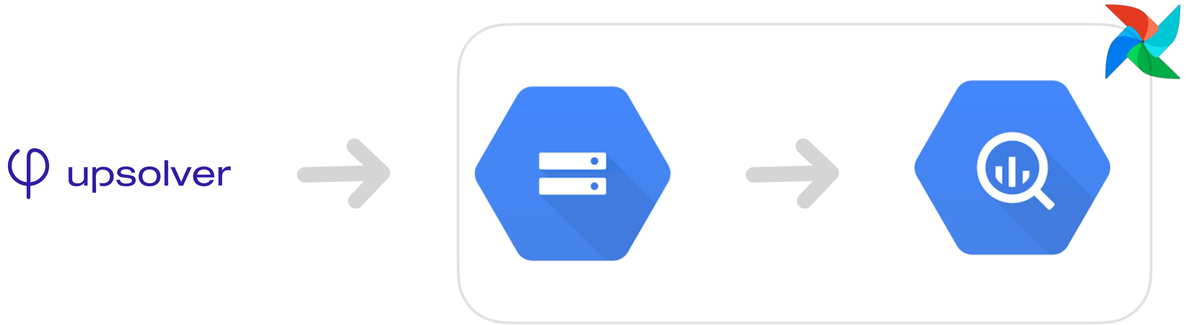


Sergio Camilo Fandiño Hernández
Senior Business Intelligence Architect @LOVOO

PythonOperator

Creating Tasks Dynamically

1. Creating a plain text with meaningful structure
2. Create a task based on a PythonOperator
3. Define and write your Callable (your custom code)



PythonOperator

Creating Tasks Dynamically

JSON File


```
{
  "broadcast_start": {
    "id": "81ce53e3-2ca6-48a2-88f7-493f7fc2c364",
    "format": "json"
  },
  "broadcast_update": {
    "id": "de7cbe04-9b53-4b4c-bc18-4d065ed3830e",
    "format": "json"
  },
  "broadcast_view": {
    "id": "2a4a0093-baee-47a2-8817-aebdb469b1b1",
    "format": "json"
  },
  "broadcast_end_view": {
    "id": "3af65f80-94a3-42c4-8ed9-502134605d27",
    "format": "json"
  },
  "broadcast_end": {
    "id": "37fa9fdf-8384-41a7-83aa-1263814b3585",
    "format": "json"
  },
```

PythonOperator

Creating Tasks Dynamically

this code belongs to the DAG.py file

```
# Iterates over all the Mapping file and extracts the event name for generating all the task-events
for event_name in event_mapping:
    event_name_task = PythonOperator(
        task_id=str(event_name),
        provide_context=True,
        python_callable=run_import_day,
        templates_dict={'exec_date': exec_date,'event_name': event_name,
                        'dataset': dataset, 'bucket_name': bucket_name},
        dag=dag)
```

Sergio Camilo Fandiño Hernández
Senior Business Intelligence Architect @LOVOO

PythonOperator

Creating Tasks Dynamically

this code belongs to the DAG.py file

```
# Function that will be called by the Python operator and will write a table partition in BQ
def run_import_day(**kwargs):
    dataset = 'events_input_analytics_tmg_backend'
    bucket_name = 'lovoo-tmg-transfer'
    import_gcs_to_bq(exec_date=kwargs.get('templates_dict').get('exec_date'),
                     event_name=kwargs.get('templates_dict').get('event_name'),
                     dataset=kwargs.get('templates_dict').get('dataset'),
                     bucket_name=kwargs.get('templates_dict').get('bucket_name'),
    )
```

Sergio Camilo Fandiño Hernández
Senior Business Intelligence Architect @LOVOO

PythonOperator

Creating Tasks Dynamically

this is your custom code (Pseudo-Code)

```
def import_gcs_to_bq(exec_date, event_name, dataset, bucket_name,**op_kwargs):

    # read the structured JSON file
    event_mapping = json.load(read_file)

    # mapping the id and the event_name
    id_event = event_mapping[event_name]['id']

    # gathering the blobs inside the bucket - array of paths
    path_array.append('gs://{0}/{1}/exec_date_file.json'.format(bucket_name, id_event))

    # BigQuery Job to Load the JSON files to a table
    load_job = bq_client.load_table_from_uri(
        tuple(path path_array), table_dest, job_config=job_config
    )
```

Sergio Camilo Fandiño Hernández
Senior Business Intelligence Architect @LOVOO

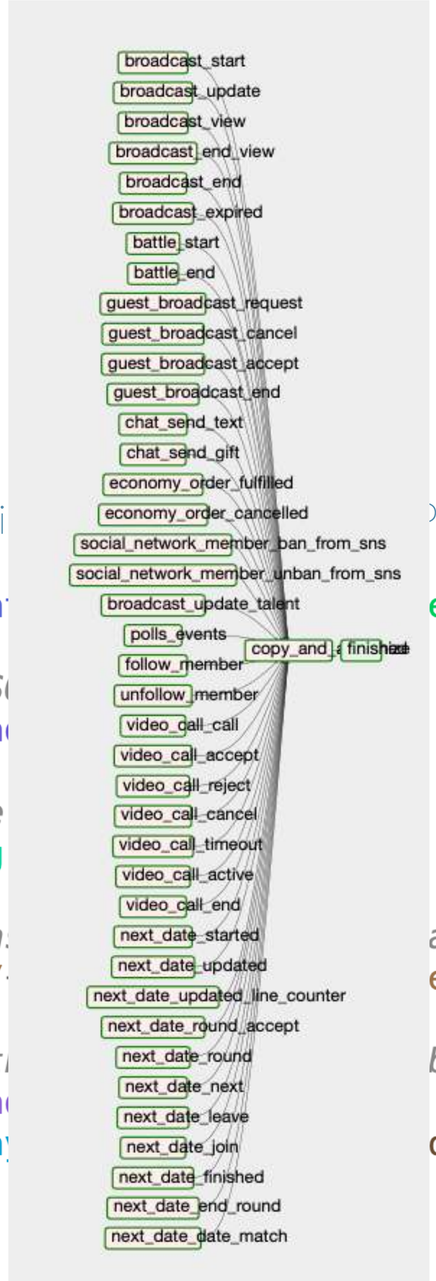
Airflow Summit 2020

PythonOperator

Creating Tasks Dynamically

this i … Pseudo-Code)

```
def import_gcs_to_bq(exec_da … et, bucket_name,**op_kwargs):

    # read the structured JS
    event_mapping = json.loa

    # mapping the id and the
    id_event = event_mapping

    # gathering the blobs in … ay of paths
    path_array.append('gs://… e.json'.format(bucket_name, id_event))

    # BigQuery Job to Load t … ble
    load_job = bq_client.loa
        tuple(path path_arra … onfig=job_config
    )
```



Sergio Camilo Fandiño Hernández
Senior Business Intelligence Architect @LOVOO

AGENDA

1. Why we met?
2. How we met?
3. The first date!
4. Fun dates!
5. Is there any dynamic in between?
6. Recap and conclusion

Recap and Conclusion

```
return KubernetesPodOperator(
    startup_timeout_seconds=60 * 10,  # we need seconds here as int, 10min now
    task_id= 'appsumer_import_' + iso_date.replace('-','_'),
    namespace='default',
    image=task_kwargs.get('image'),
    cmds=task_kwargs.get('command'),
    secrets=[appsumer_pass, service_account],
    env_vars=env_vars,
    name=task_kwargs.get('name'),
    is_delete_operator_pod=True,
    dag=dag,
    dt=dt,
    pool="appsumer_pool",
    get_logs=True,
    resources=resources,
    affinity={
        'nodeAffinity': {
            # requiredDuringSchedulingIgnoredDuringExecution means in order
            # for a pod to be scheduled on a node, the node must have the
            # specified labels. However, if labels on a node change at
            # runtime such that the affinity rules on a pod are no longer
            # met, the pod will still continue to run on the node.
            'requiredDuringSchedulingIgnoredDuringExecution': {
                'nodeSelectorTerms': [{
                    'matchExpressions': [{
                        'key': 'kuberunoperator',
                        'operator': 'In',
                        'values': [
                            'true',
                        ]
```

kubernetes

Recap and Conclusion

- Using an Alpha version (Google Composer) in Production was challenging!
- Focus on what's important - Google Cloud Composer
- Airflow leverages a bunch of Operators OOTB
- Always room for improvement
- No magic recipe to use - stay flexible

Gracias.

Feedback and Questions

LinkedIn:

https://www.linkedin.com/in/fandinohernandez/

Email: sergio.fandino@lovoo.com

July 16, 2020 Berlin - Germany

THE MEET GROUP

LOVOO

Airflow Summit 2020