

---

# Airflow Operators need to Die



By Bolke de Bruin

---

# Bolke de Bruin

Airflow since 2015

Now maintaining  
serialization and  
deserialization of Xcom



# Why Operators?

→ **Parallelism and Dependency Management**

Defining dependencies between tasks in a workflow..

→ **Modularity**

Encapsulate a specific type of task or operation

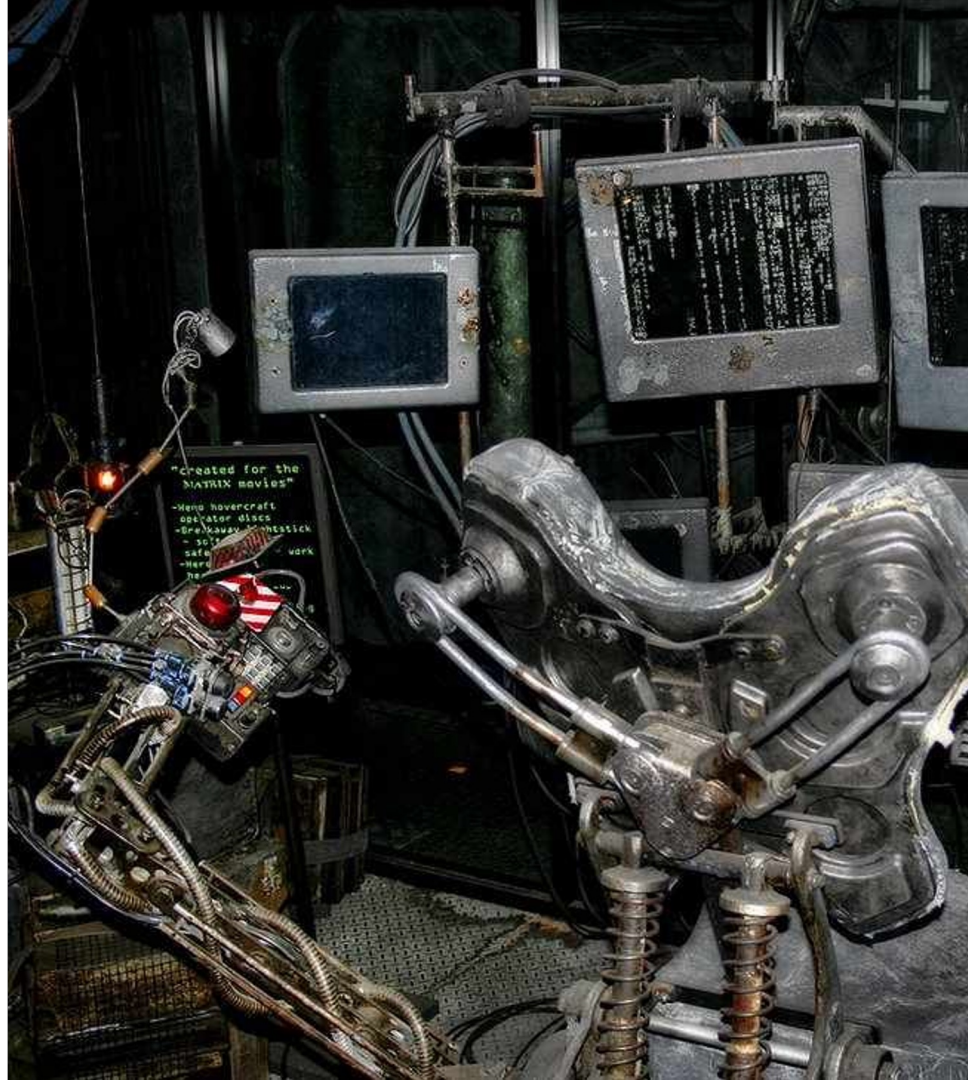
→ **Abstraction**

Abstract away the details of executing specific operations

→ **Extensibility**

Organizations can create custom Operators to suit their needs

Apache Airflow boosts  
over **100 Operators**  
in **> 80** provider  
packages



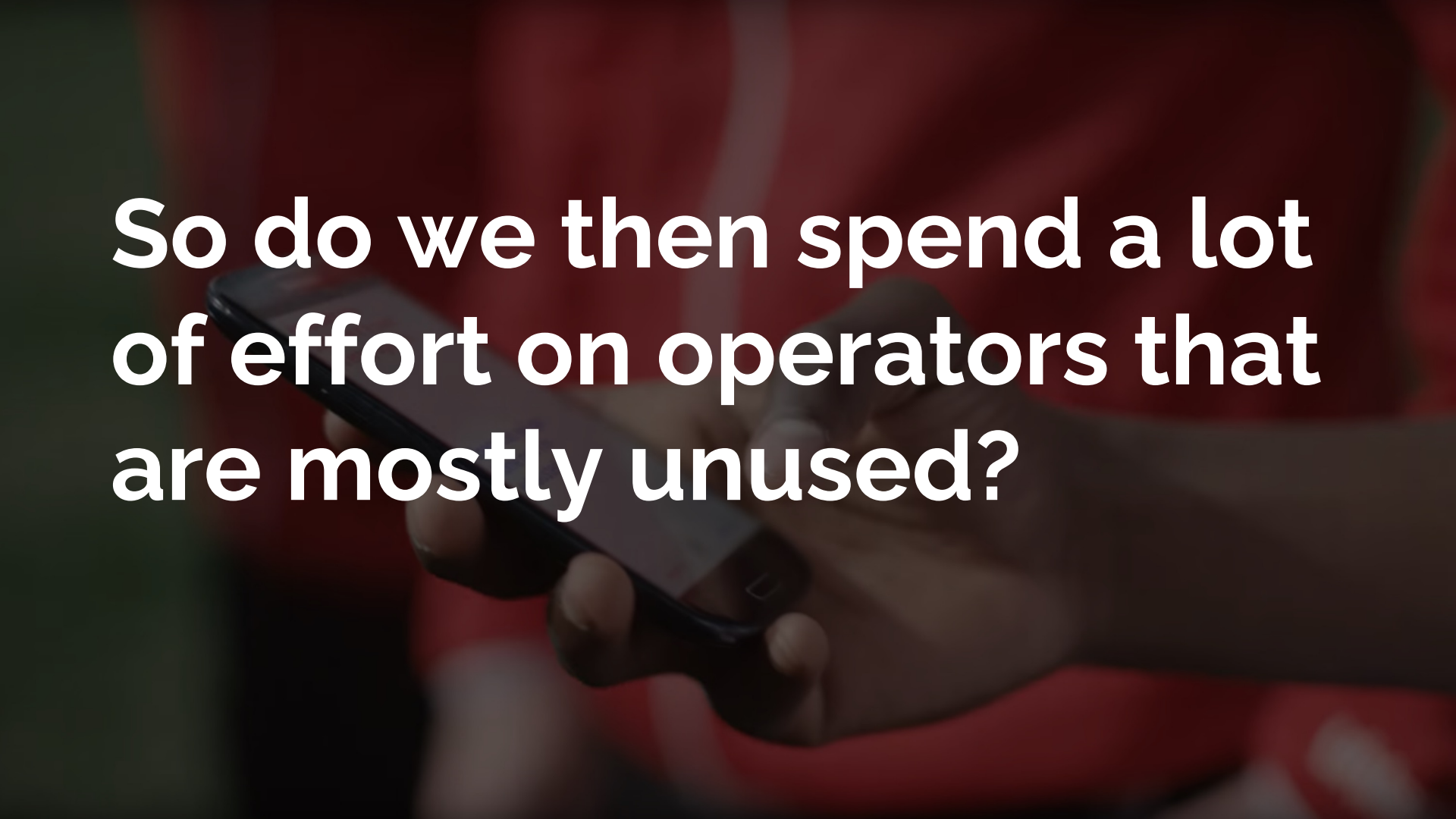
—

# What are the most used Operators in Airflow?

---

# PythonOperator & BashOperator.

(> 90% as reported by Astronomer, > 50% by Google)

A hand holding a black smartphone, with the screen facing the viewer. The background is a blurred red surface. The text is overlaid on the image in a large, white, sans-serif font.

**So do we then spend a lot of effort on operators that are mostly unused?**

# But there are other issues...

You cannot use them *within* Taskflow functions or within PythonOperator

O(n):

- HiveToMySql
- MySqlToHive
- VerticaToHive
- GlacierToGcs

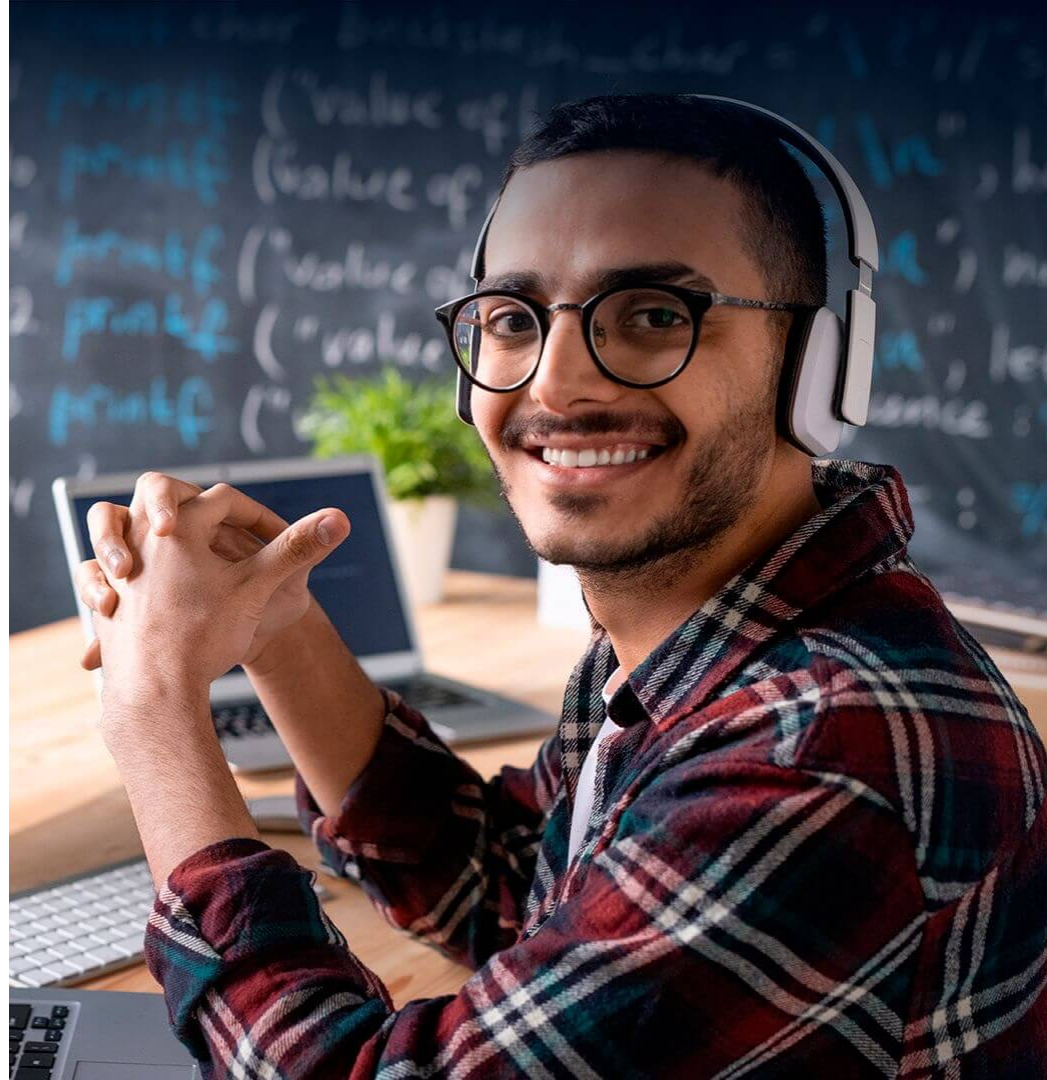
It's hard to capture lineage from Operators



## Meet Marcos.

Marcos is a one-man-army data scientist and lives and breathes Notebooks.

He works for an auction company and deals with lots, hammer values, nas, gas etc.



SparkNLP-RegExp-Inter... Python ▼ ☆

File Edit View Run Help [La...](#) [Provide feedback](#) ▶ Run all ■ Terminated ▼ 📅 Schedule Share

```
66 LOTS = "/mnt/lake/presentation/atlas/FactLots"
67 DIM_LOTS = "/mnt/lake/presentation/atlas/DimLot"
68 OUTPUT = "/mnt/lake/presentation/Customer_Service/IntercomAuctions1"
69 PLATFORM_ID = "TWK"
70 EXPERIMENT = "dbfs:/databricks/mlflow-tracking/3567689888134771"
71 EXPERIMENT_ID = "3567689888134771"
72 INTERCOM_IN = "/mnt/lake/standardised/troostwijk/intercom/conversations"
73 MODEL_NAME = "intercom_auction_regex_matcher"
```

Command took 0.05 seconds -- by b.debruin@tbauctions.com at 11/09/2023, 08:02:22 on 13.2-LARGE-ICEBERG

Cmd 2

```
1 if mlflow.active_run():
2     display("Found a current active MLFlow run. Ending the current one and
3         restarting. You probably do not want this")
4     mlflow.end_run()
5
6 mlflow.start_run(experiment_id=EXPERIMENT_ID)
7 mlflow.log_param("platform", PLATFORM_ID)
```

▼ (1) MLflow run  
Logged 1 run to an experiment in MLflow. [Learn more](#)

Out[438]: 'TWK'

Command took 0.78 seconds -- by b.debruin@tbauctions.com at 11/09/2023, 08:02:22 on 13.2-LARGE-ICEBERG

Cmd 3

```
1 # this grabs the intercom data and transforms it so we can use it
2 intercom_df = spark.read.format("delta").load(INTERCOM_IN)
3 intercom_df = (
4     intercom_df.withColumn("year", F.year(F.col("created_at").cast("timestamp")))
5     .withColumn("dayofyear", F.dayofyear(F.col("created_at").cast("timestamp")))
6     .withColumn("month", F.month(F.col("created_at").cast("timestamp")))
7 )
8
9 # remove html from message subject and body
10 intercom_df = intercom_df.withColumn("subject", F.regexp_replace(intercom_df.subject, "<.*>", ""))
```



# Rethinking the Operator

*Long live the Operator!*

—

**Wouldn't it be nice if...**

# ...Operators could deal with File-like objects residing anywhere?

So that GlacierToGCS (and basically every Transfer Operator) could just become:

**CopyFileObject(src, dst)**

Capturing lineage at the same time?

# ...Taskflow and Operators would be truly integrated

So that this becomes possible

```
@task  
  
def my_task():  
    for i in range(10):  
        do_something()  
    S3ToMySQLOperator(sql)
```

**Instead of:**

```
@task  
  
def my_task():  
    hook = MySQLHook()  
    hook.execute(sql)
```

# ...Operators know how to deal with DataFrames

```
import(dataframe, snowflake_table)
df = export(snowflake_table)
```

Removing the need for *DatastoreToDatastore*  
Operators

—

**This would reduce the  
number of Operators by  
~70%**

**Splitting between Tasks and Data Aware  
Operators**



A hand holding a smartphone against a red background. The text is overlaid on the image.

**The Astro SDK covers some  
of this, but not all**

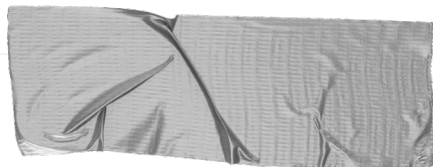
---

# Let's embrace and extend

**BlobAPI**

**DataFrameAPI**

**Remove  
unnneeded  
Operators**



**Thank you!**

[bdbruin@gmail.com](mailto:bdbruin@gmail.com)