
Data Pipelines con Luigi

09/05/2019


Speaker: Alejandro Rodríguez Díaz

PyData Salamanca meetup

¿Qué vamos a ver?

1. Motivación
2. ¿Pipelines?
3. Pipelines con Luigi
4. Otras posibilidades
5. Conclusión

¿Por qué crear data pipelines?



```
import pandas as pd
import numpy as np
import urllib2

req1 = urllib2.Request('http://just some api/entries/15')
req2 = urllib2.Request('http://just some api/entries/15')
res1= urllib2.urlopen(req1).read()
res2 = urllib2.urlopen(req2).read()
df_1 = pd.read_csv('./some.scv')
df_2 = pd.read_csv('./another.csv', dtype={'duration': int})
df_1.dropna(how='all')
df_2.dropna(how='all')
df_2 = df_2.rename(columns = {'program':'id'})

pd.merge(df1, df2, on='id')
df.to_csv('aggregated.csv')
```

• • •



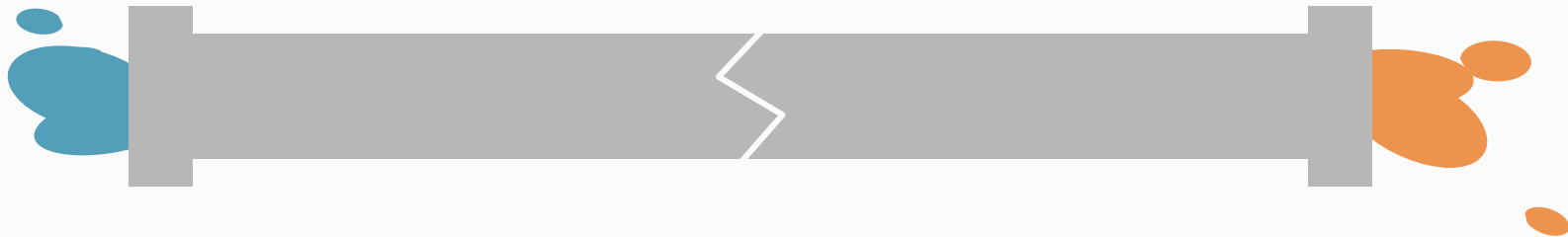
```
import pandas as pd
import numpy as np
import urllib2
```

```
pandas.parser.CParserError: Error tokenizing data. C error: Expected 2 fields in line 3, saw 12
```

```
df_1 = pd.read_csv('./some.csv')
df_2 = pd.read_csv('./another.csv', dtype={'duration': int})
df_1.dropna(how='all')
df_2.dropna(how='all')
df_2 = df_2.rename(columns = {'program':'id'})

pd.merge(df1, df2, on='id')
df.to_csv('aggregated.csv')

• • •
```



```
import pandas as pd
import numpy as np
import urllib2
```

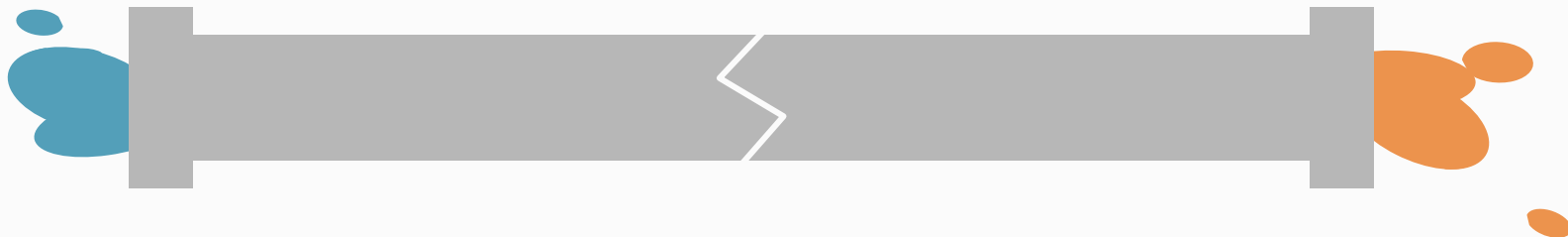
```
pandas.parser.CParserError: Error tokenizing data. C error: Expected 2 fields in line 3, saw 12
```

```
ImportError: /usr/lib/arm-linux-gnueabi/libgfortran.so.3: invalid ELF header
```

```
df_2.dropna(how='all')
df_2 = df_2.rename(columns = {'program':'id'})
```

```
pd.merge(df1, df2, on='id')
df.to_csv('aggregated.csv')
```

• • •



```
import pandas as pd  
import numpy as np  
import urllib2
```

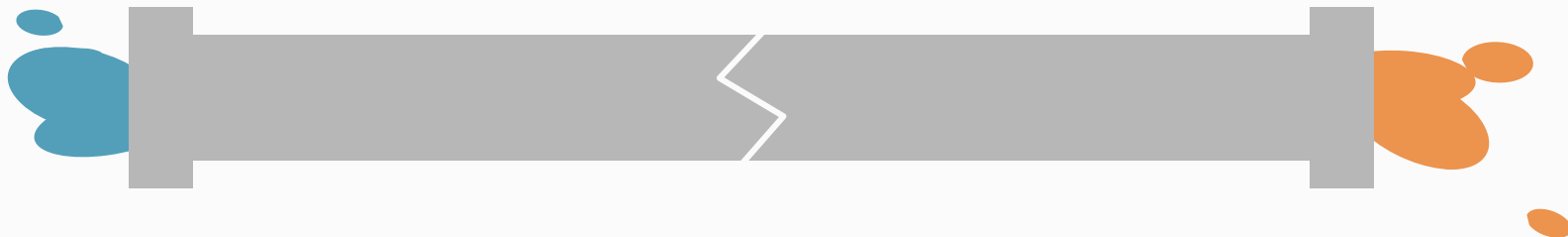
```
pandas.parser.CParserError: Error tokenizing data. C error: Expected 2 fields in line 3, saw 12
```

```
ImportError: /usr/lib/arm-linux-gnueabi/libgfortran.so.3: invalid ELF header
```

```
403 forbidden
```

```
pd.merge(df1, df2, on='id')  
df.to_csv('aggregated.csv')
```

• • •



```
File "example-logging-exception.py", line 5, in get_number  
    return int('foo')  
ValueError: invalid literal for int() with base 10: 'foo'
```

```
pandas.parser.CParserError: Error tokenizing data. C error: Expected 2 fields in line 3, saw 12
```

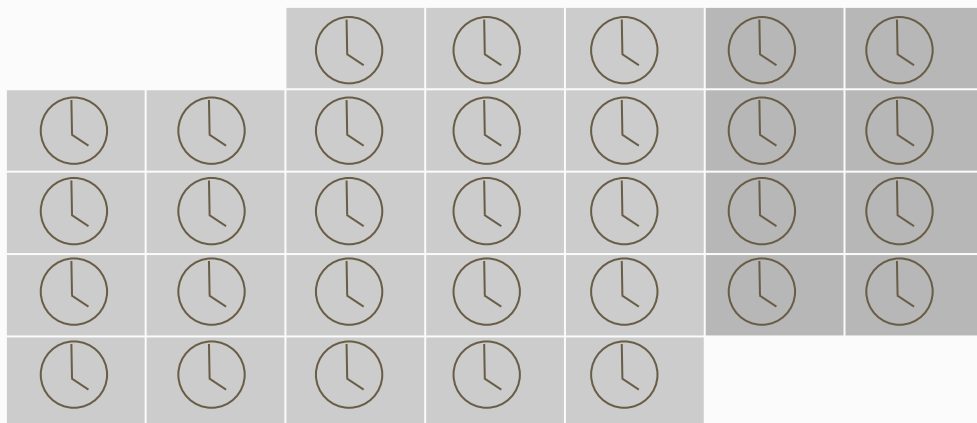
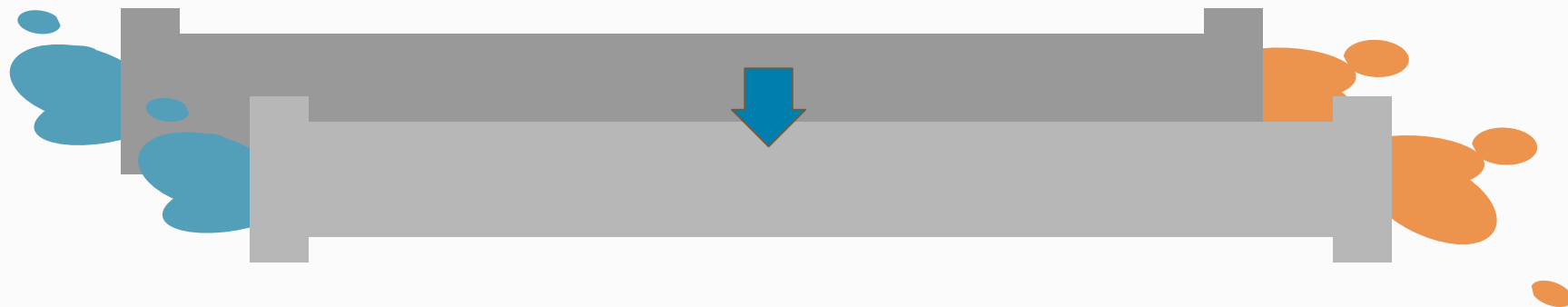
```
ImportError: /usr/lib/arm-linux-gnueabi/libgfortran.so.3: invalid ELF header
```

```
403 forbidden
```

```
pd.merge(df1, df2, on='id')  
df.to_csv('aggregated.csv')
```

• • •

Automatizando el proceso



- 01 01 * * * /usr/bin/python3 script1.py
- 01 11 * * * /usr/bin/python3 script2.py

Escalando el flujo de datos

Necesidades

- Paralelización
- Automatización
- Gestión de recursos
- Monitorización
- Parametrización
- Gestión de dependencias
- Acceso a recursos externos
- Colaboración

Problemas

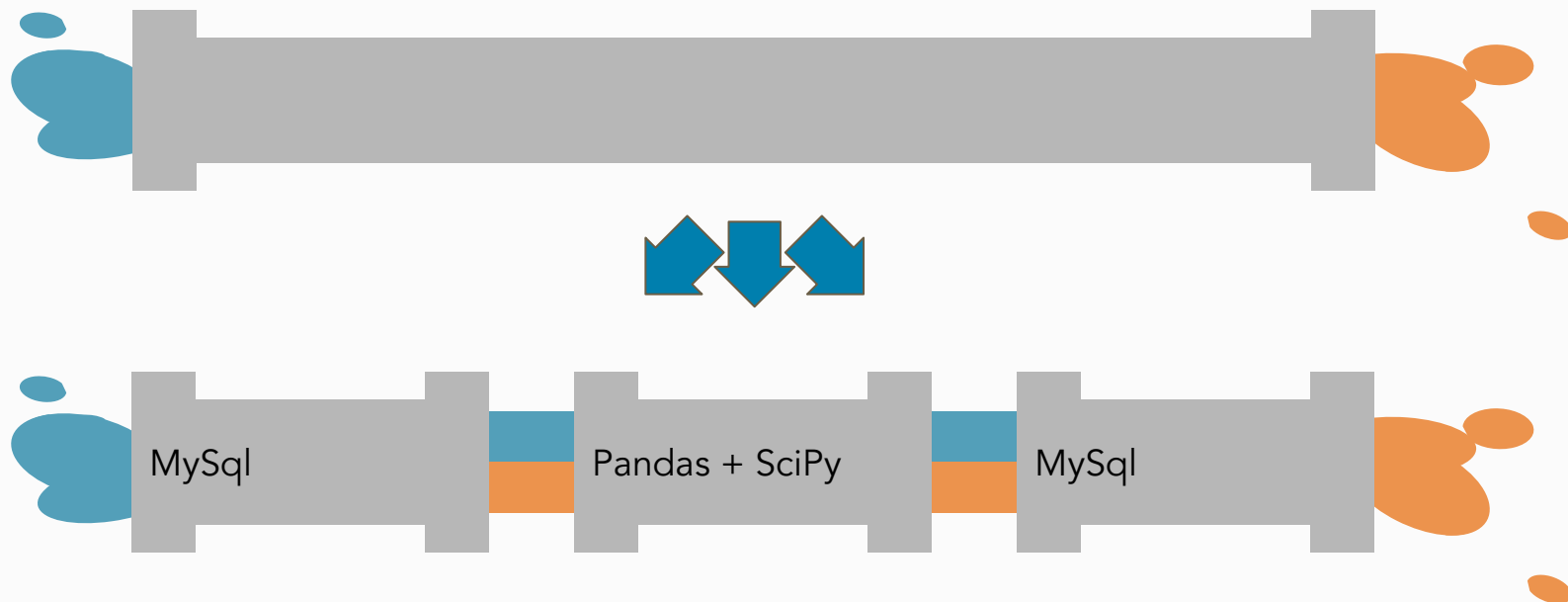
- Concurrencia
- Tipado
- Errores de ejecución
- Acceso a recursos externos
- Uso de servicios externos
- Disponibilidad de los recursos

¿Hasta dónde con un script?





Dividir el proceso = modularidad



Data pipelines

¿Data pipelines?

AWS Data Pipeline configures and manages a data-driven workflow called a pipeline.

<https://docs.aws.amazon.com/cli/latest/reference/datapipeline/index.html>

Moving data between systems can require many steps: from copying data, to moving it from an on-premise location into the cloud, [...]

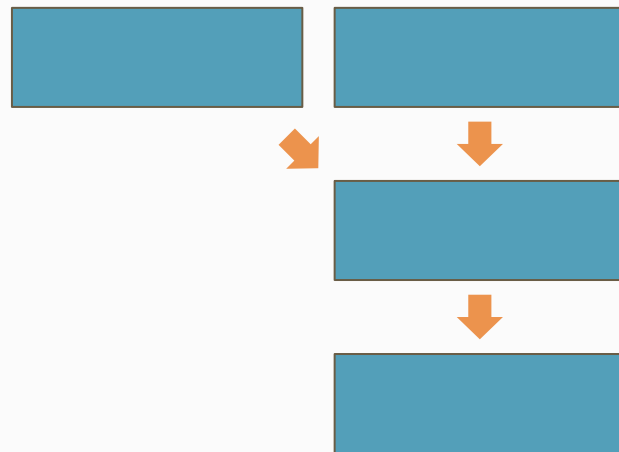
A data pipeline is the sum of all these steps, and its job is to ensure that these steps all happen reliably to all data. These processes should be automated [...]

<https://www.dremio.com/what-is-a-data-pipeline/>

Una data pipeline es una solución software que maneja flujos de datos usando procesos segmentados en otros más pequeños.

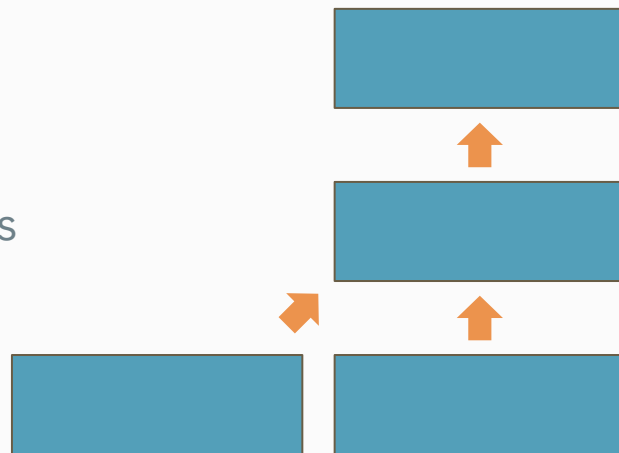
Definiendo el flujo de datos

1. Obtener tabla x de la base de datos
2. Obtener fichero csv disponible a través de ftp
3. Cargar datos
4. Limpiar datos
5. Crear índices
6. Agregar
7. Crear resúmenes
8. Guardar
9. Desplegar

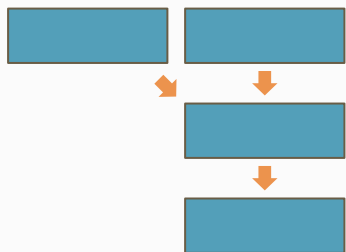


Definiendo el flujo de datos

1. Desplegar los datos que se hayan guardado
2. Guardar los resúmenes que se hayan creado
3. Crear resúmenes a partir de las agregaciones
4. Agregar usando los índices previamente creados
5. Crear los índices usando los datos limpios
6. Limpiar los datos que se hayan cargado
7. Cargar los datos que se hayan descargado
8. Obtener tabla x de la base de datos
9. Obtener fichero csv disponible a través de ftp

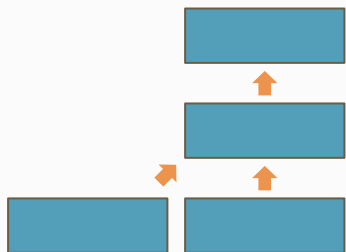


Implementando la data pipeline



De forma explícita

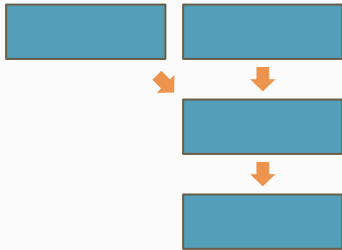
Definiendo cada tarea y el orden de ejecución



De forma implícita

Definiendo cada tarea y sus dependencias

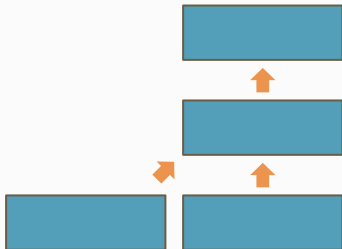
Implementando la data pipeline



De forma explícita

Definiendo cada tarea y el orden de ejecución

Apache Airflow, ActionChain, Mario

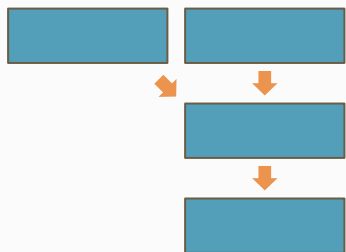


De forma implícita

Definiendo cada tarea y sus dependencias

Makefile, BioMake, Luigi, Nextflow

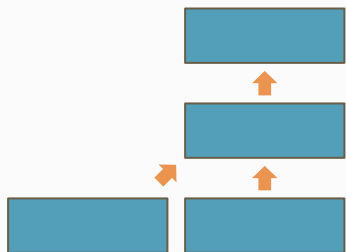
Implementando la data pipeline



De forma explícita

Definiendo cada tarea y el orden de ejecución

Apache Airflow, ActionChain, Mario



De forma implícita

Definiendo cada tarea y sus dependencias

Makefile, BioMake, Luigi, Nextflow



<https://raw.githubusercontent.com/spotify/luigi/master/doc/luigi.png>

Pipelines con Luigi

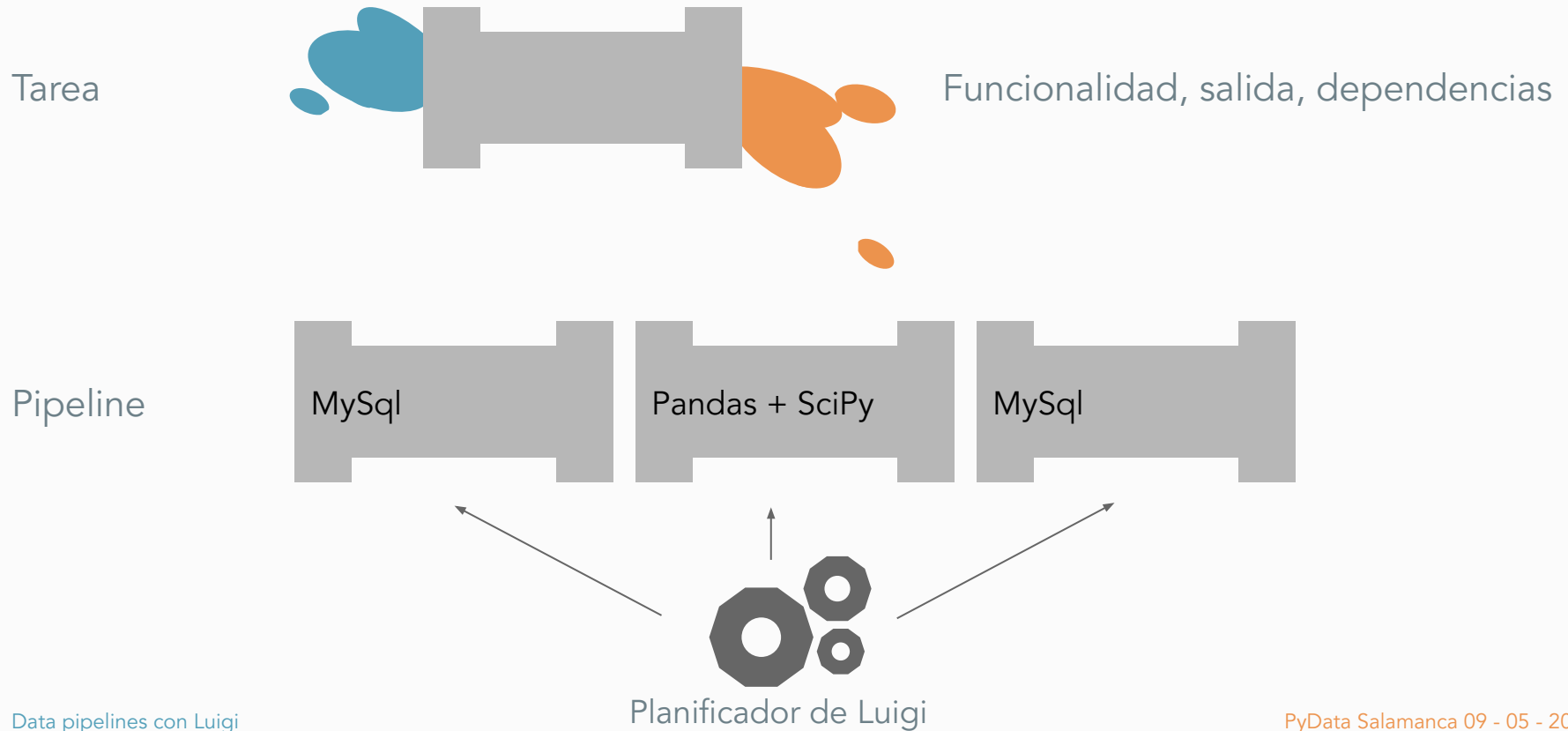
Pipelines con Luigi

Tarea

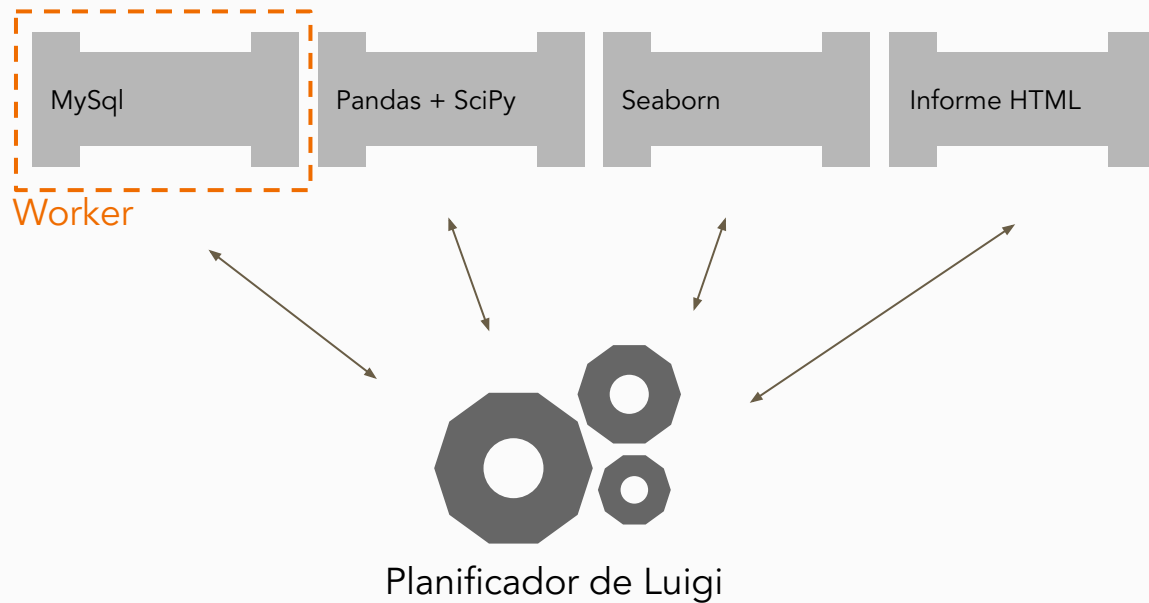


Funcionalidad, salida, dependencias

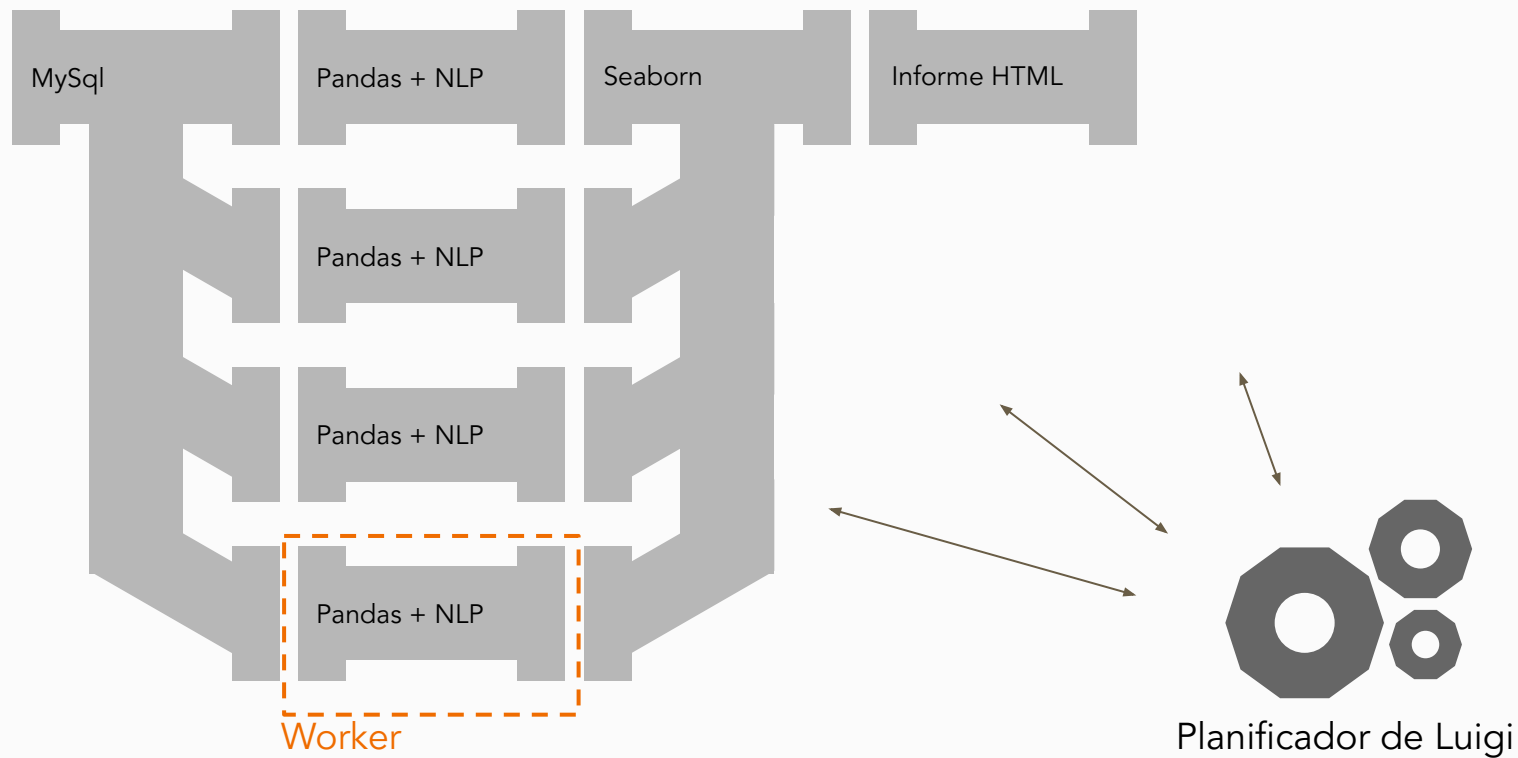
Pipelines con Luigi



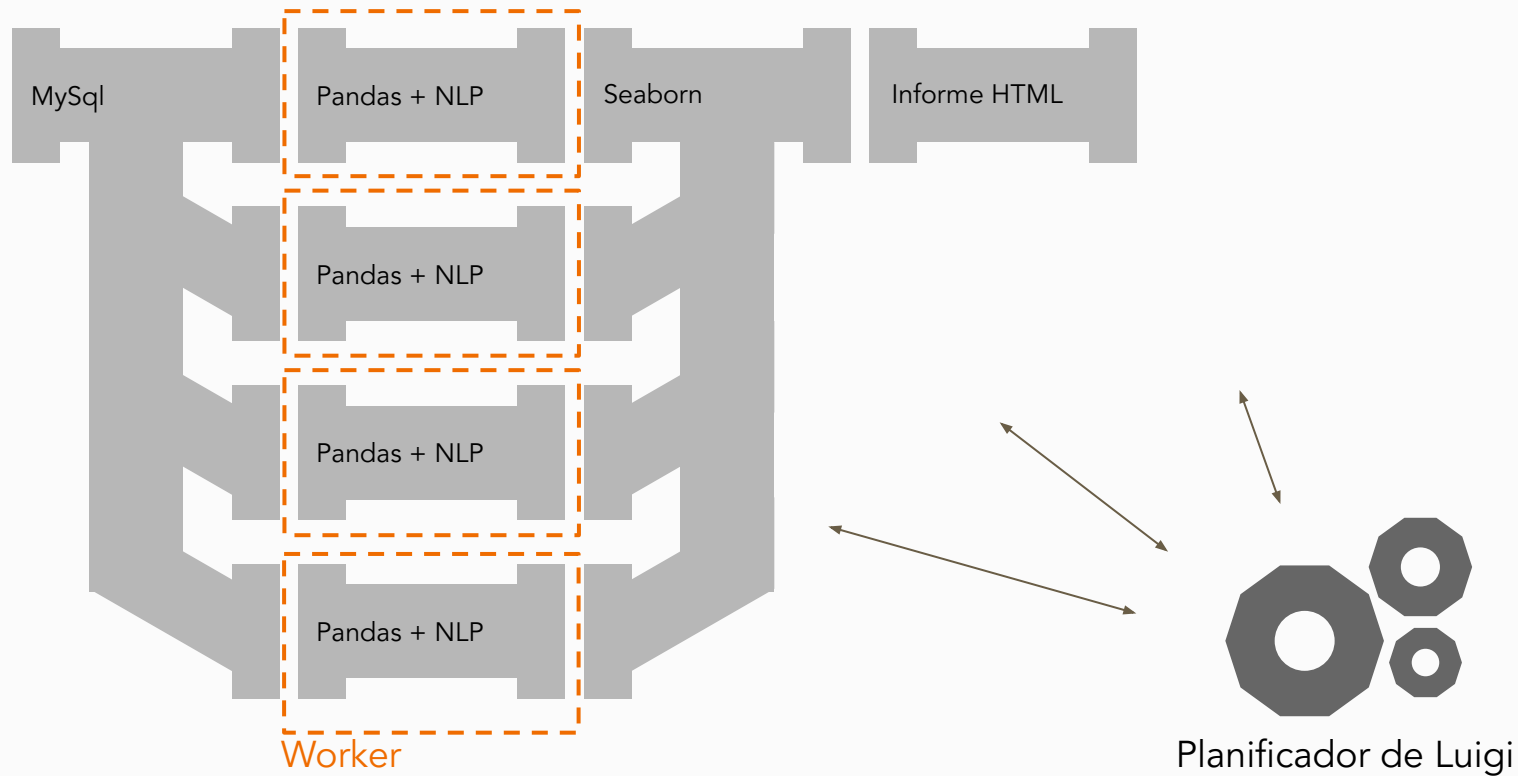
Planificador de Luigi



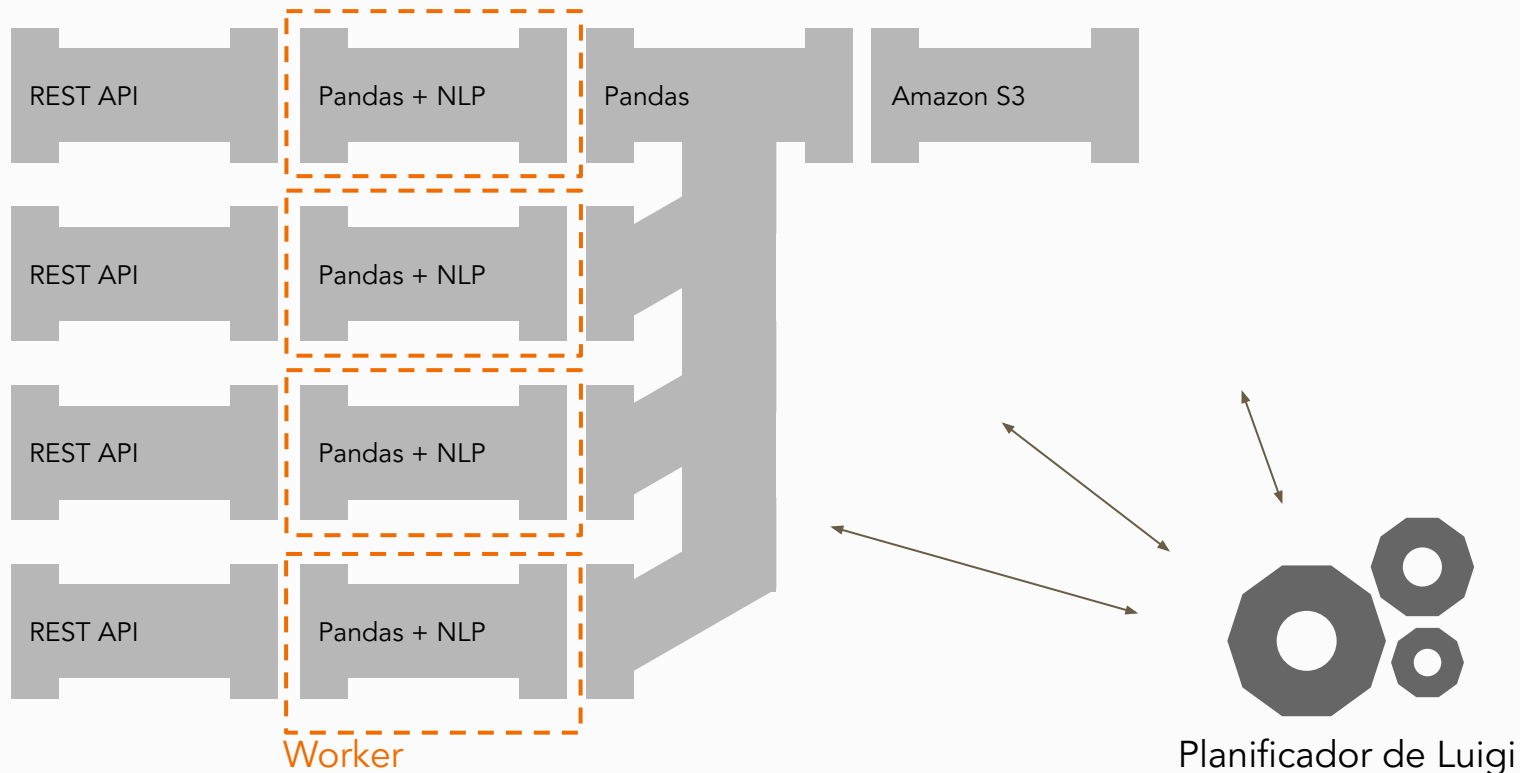
Planificador de Luigi



Planificador de Luigi



Planificador de Luigi



Luigi Task Visualiser

localhost:8082/static/visualiser/index.html#

min_web

Luigi Task Status

Task ListDependency GraphWorkersResources

Running

TASK FAMILIES

PENDING TASKS0

RUNNING TASKS0

BATCH RUNNING TASKS0

DONE TASKS0

FAILED TASKS0

UPSTREAM FAILURE0

DISABLED TASKS0

UPSTREAM DISABLED0

Show10▼entries

Filter table:

Filter on Server☐

Name	Details	Priority	Time	Actions
No data available in table				

Showing 0 to 0 of 0 entries

PreviousNext

Luigi Task Visualiser
+
localhost:8082/static/visualiser/index.html#
min_web

Luigi Task Status
Task List
Dependency Graph
Workers
Resources
Running

TASK FAMILIES

- 1 ExtractSlides
- 20 ExtraProcessing
- 1 MergeSlides
- 1 Pipeline
- 20 Pptx2Pdf

PENDING TASKS
0

RUNNING TASKS
0

BATCH RUNNING TASKS
0

DONE TASKS
21

FAILED TASKS
20

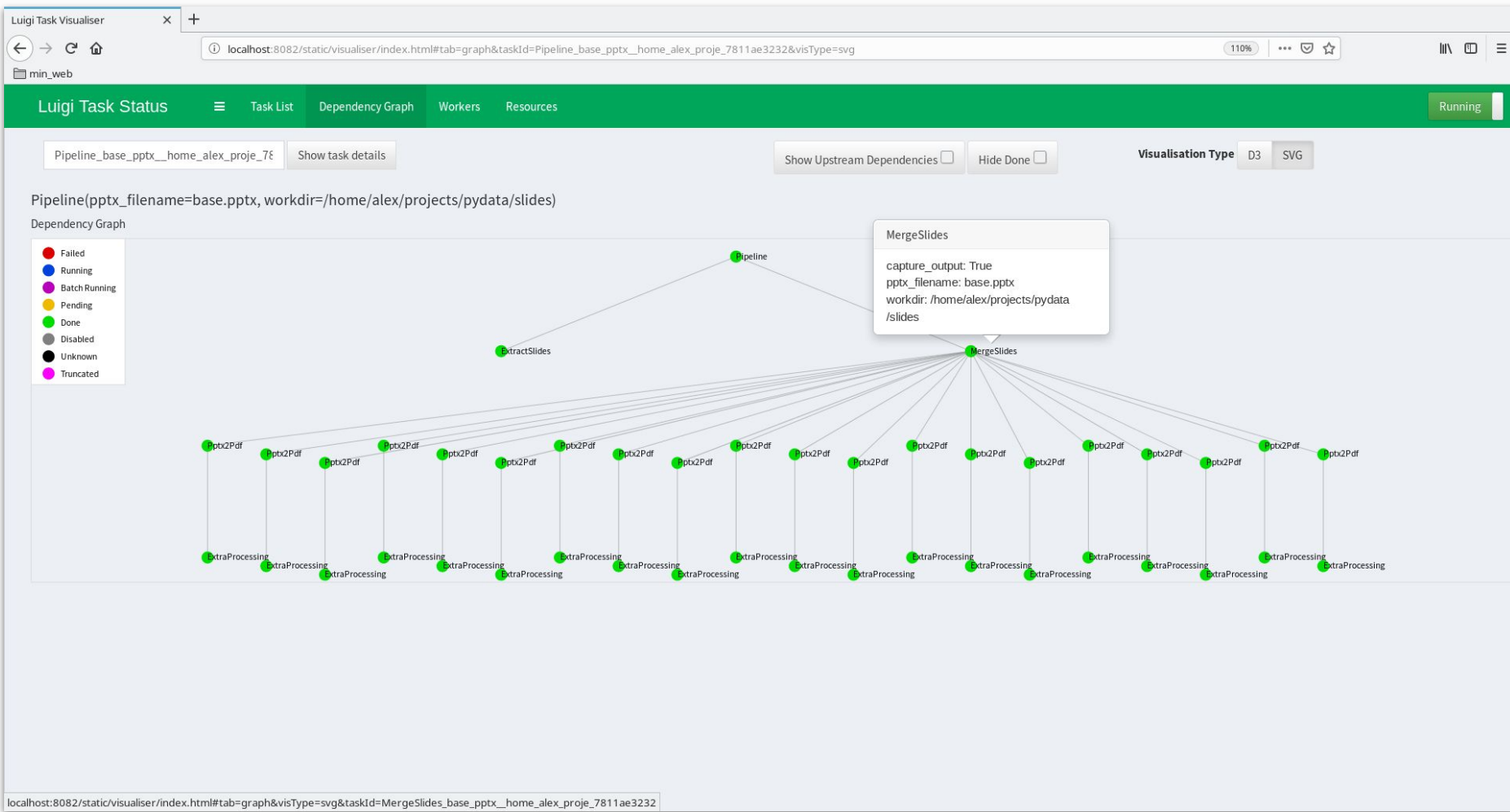
UPSTREAM FAILURE
2

DISABLED TASKS
0

UPSTREAM DISABLED
0

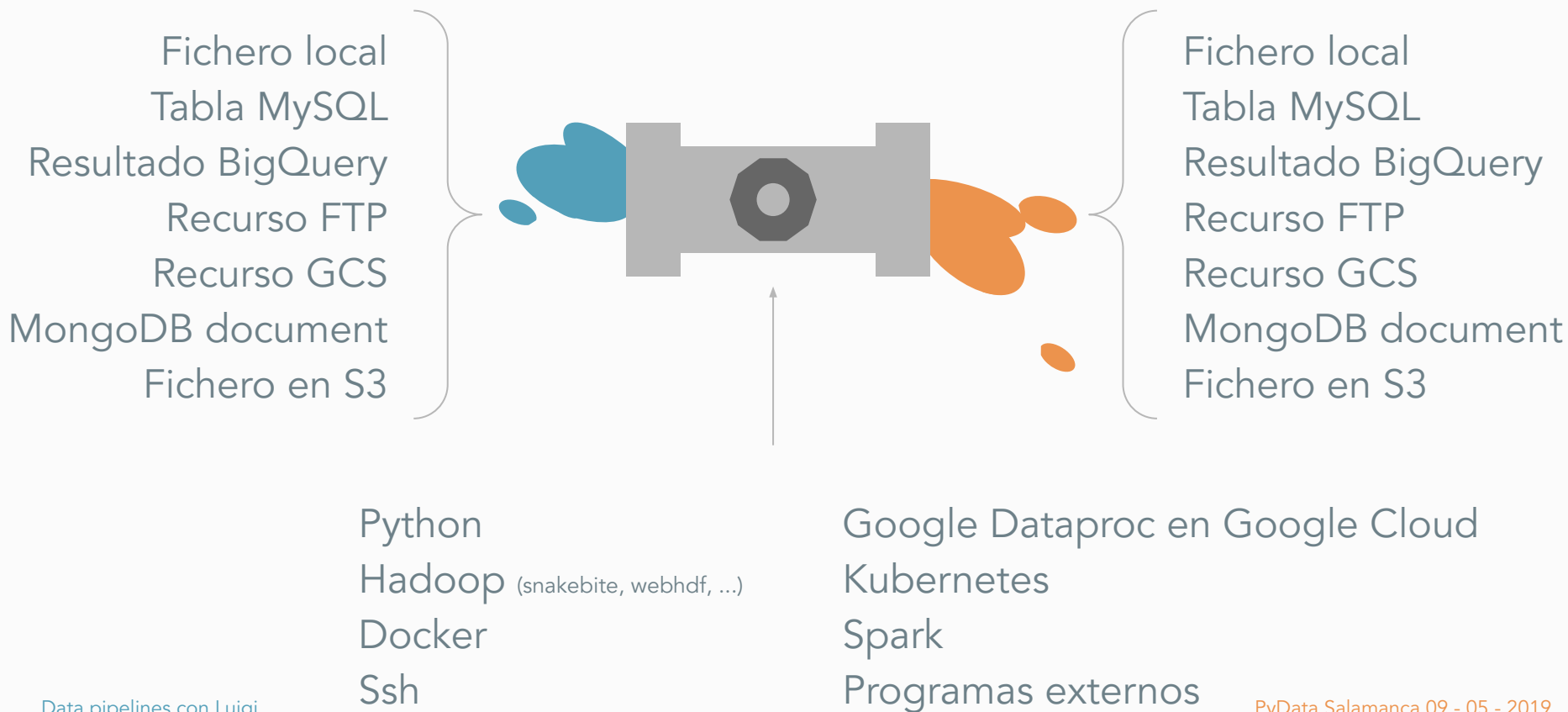
Show 10 entries
Filter table:
Filter on Server

	Name	Details	Priority	Time	Actions
UPSTREAM_FAILED	MergeSlides	capture_output=True, pptx_filename=base.pptx, workdir=/home/alex/projects/pydata/slides	0	5/3/2019, 12:13:21 AM	
UPSTREAM_FAILED	Pipeline	pptx_filename=base.pptx, workdir=/home/alex/projects/pydata/slides	0	5/3/2019, 12:13:21 AM	
FAILED	Pptx2Pdf	capture_output=True, pptx_filename=base_raw_13.pptx, workdir=/home/alex/projects/pydata/slides	0	5/3/2019, 12:13:21 AM	
FAILED	Pptx2Pdf	capture_output=True, pptx_filename=base_raw_14.pptx, workdir=/home/alex/projects/pydata/slides	0	5/3/2019, 12:13:21 AM	
FAILED	Pptx2Pdf	capture_output=True, pptx_filename=base_raw_19.pptx, workdir=/home/alex/projects/pydata/slides	0	5/3/2019, 12:13:21 AM	
FAILED	Pptx2Pdf	capture_output=True, pptx_filename=base_raw_7.pptx, workdir=/home/alex/projects/pydata/slides	0	5/3/2019, 12:13:21 AM	
FAILED	Pptx2Pdf	capture_output=True, pptx_filename=base_raw_1.pptx, workdir=/home/alex/projects/pydata/slides	0	5/3/2019, 12:13:21 AM	

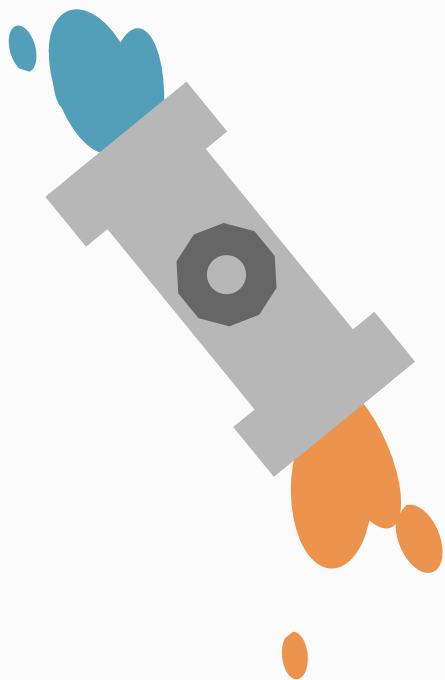


Tareas en Luigi

Tareas en Luigi



Tareas en Luigi



```
import luigi

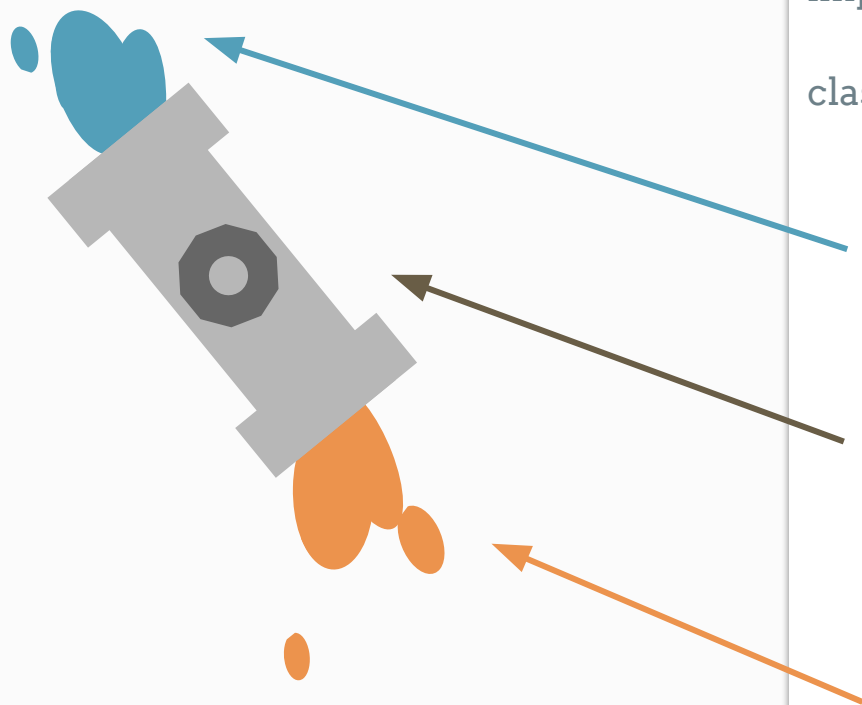
class Tarea(luigi.Task):
    fecha: str = luigi.Parameter()

    def requires(self):
        return TareaPrevia(param = 'un parámetro')

    def run(self):
        ...
        with self.output().open('w') as f:
            f.write( ... )

    def output(self):
        return luigi.LocalTarget('ruta al recurso')
```

Tareas en Luigi



```
import luigi
```

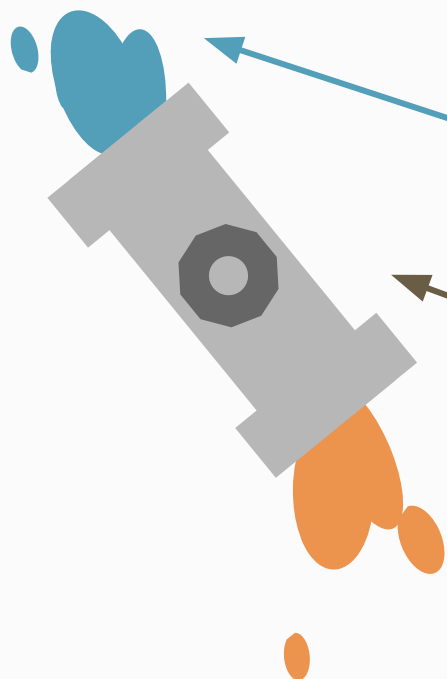
```
class Tarea(luigi.Task):  
    fecha: str = luigi.Parameter()
```

```
    def requires(self):  
        return TareaPrevia(param = '...')
```

```
    def run(self):  
        ...  
        with self.output().open('w') as f:  
            f.write(...)
```

```
    def output(self):  
        return luigi.LocalTarget('ruta al recurso')
```

Tareas en Luigi



```
import luigi
```

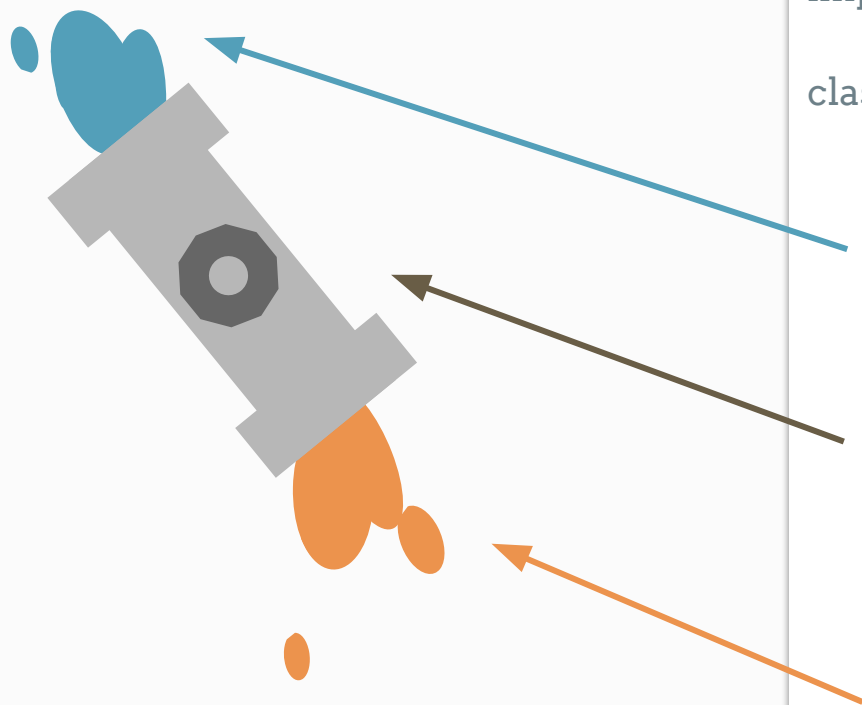
```
class Tarea(luigi.contrib.ExternalProgramTask):  
    fecha: str = luigi.Parameter()
```

```
def requires(self):  
    return TareaPrevia(param = '...')
```

```
def program_args(self):  
    url = "..."  
    return ['wget', '-O', self.output().path, url]
```

```
def output(self):  
    return luigi.LocalTarget('ruta al recurso')
```

Tareas en Luigi



```
import luigi

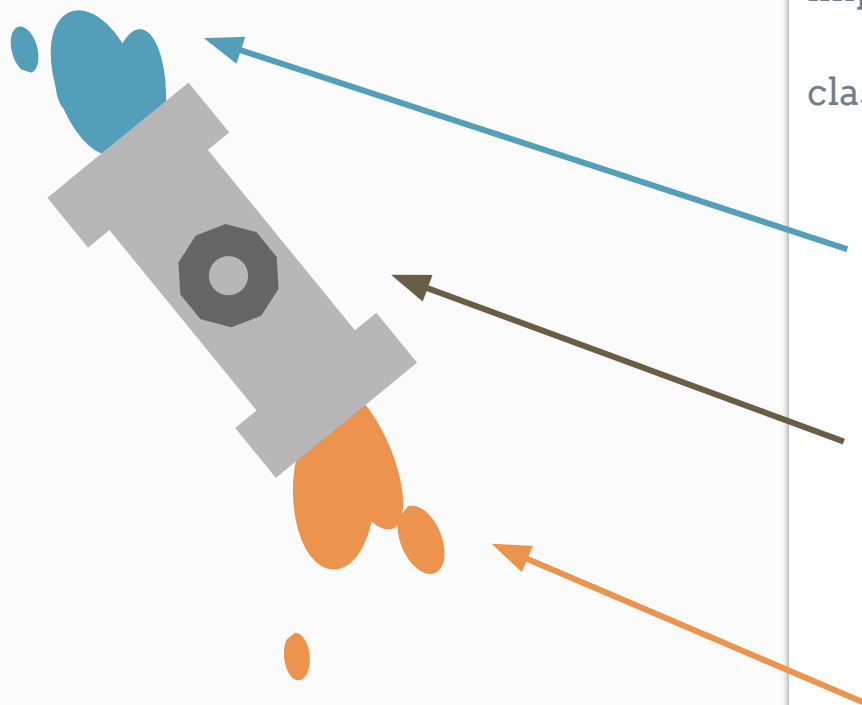
class Tarea(luigi.Task):
    fecha: str = luigi.Parameter()

    def requires(self):
        return {
            'dep_1': TareaA(param = '...'),
            'dep_2': TareaB()
        }

    def run(self):
        ...

    def output(self):
        return luigi.LocalTarget('ruta al recurso')
```

Tareas en Luigi



```
import luigi

class Tarea(luigi.Task):
    fecha: str = luigi.Parameter()

    def requires(self):
        return TareaPrevia(param = '...')

    def run(self):
        ...

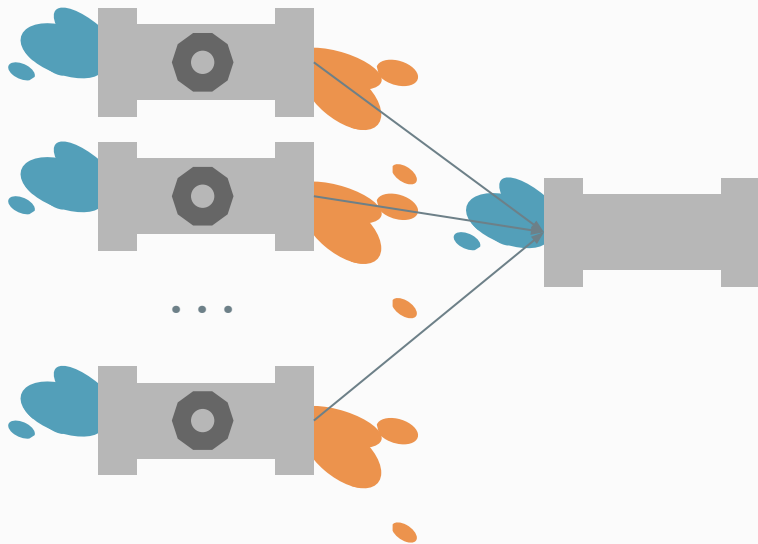
    def output(self):
        return {
            'out1': luigi.LocalTarget('ruta'),
            'out2': luigi.LocalTarget('ruta')
        }
```

¿Qué más ofrece Luigi?

Patrones comunes en Luigi

Wrapper tasks

Requerir la ejecución de varias tareas sin producir ningún resultado.



```
import luigi
```

```
class Tarea(luigi.WrapperTask):
```

```
    def requires(self):
```

```
        yield TareaA('un parámetro')
```

```
        yield TareaB('un parámetro')
```

```
        yield TareaC('un parámetro')
```

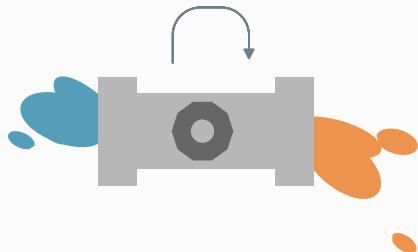
```
        yield TareaD('un parámetro')
```

Patrones comunes en Luigi

Tareas Range

Ejecutar una tarea para cada uno de los valores en un rango de tiempo

[2019-05-6, 2019-05-12)



```
import luigi

class Tarea(luigi.Task):
    fecha: str = luigi.DateParameter()

    def requires(self):
        ...
    def run(self):
        ...
    def output(self):
        ...
```

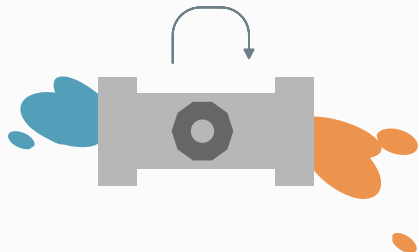
```
luigi --module all_reports RangeDaily --of Tarea --start 2019-05-06 --stop 2019-05-12
```


Patrones comunes en Luigi

Tareas Range

Ejecutar una tarea para cada uno de los valores en un rango de tiempo

[2019-05-6, 2019-05-12)



```
import luigi

class Tarea(luigi.Task):
    fecha: str = luigi.DateParameter()

    def requires(self):
        ...
    def run(self):
        ...
    def output(self):
        ...
```

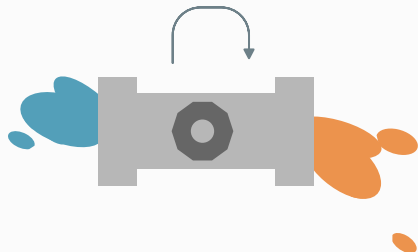
```
luigi --module all_reports RangeDaily --of Tarea --start 2019-05-06
```

Patrones comunes en Luigi

Tareas Range

Ejecutar una tarea para cada uno de los valores en un rango de tiempo

[2019-05-6, 2019-05-12)



```
import luigi

class Tarea(luigi.Task):
    fecha: str = luigi.DateParameter()

    def requires(self):
        ...

    def run(self):
        ...

    def output(self):
        ...
```

```
luigi --module all_reports RangeDaily --of Tarea --start 2019-05-06 --of-params '{"p1": "123", "p2": 1}'
```

Patrones comunes en Luigi

Parámetros globales

Propagación de parámetros globales.



```
class A(luigiWrapper.Task):  
    log_dir: str = luigi.Parameter()  
    def requires(self):  
        return None  
  
...  
  
class B(luigiWrapper.Task):  
    log_dir: str = luigi.Parameter()  
    def requires(self):  
        return A(log_dir=self.log_dir)  
  
...  
  
class C(luigiWrapper.Task):  
    log_dir: str = luigi.Parameter()  
    def requires(self):  
        return B(log_dir=self.log_dir)  
  
...
```

Patrones comunes en Luigi

Parámetros globales

Propagación de parámetros globales.



```
class GlobalParams(luigi.Config):  
    log_dir = luigi.Parameter()  
  
class A(luigiWrapper.Task):  
    def requires(self):  
        return None  
  
    ...  
  
class B(luigiWrapper.Task):  
    def requires(self):  
        return A()  
  
    ...  
  
class C(luigiWrapper.Task):  
    def requires(self):  
        return B()  
  
    def output(self):  
        log_path = os.path.join(Global().log_dir, 'c.log')  
        return luigi.LocalTarget(log_path)
```

Patrones comunes en Luigi

Parámetros globales

Propagación de parámetros globales.



```
class GlobalParams(luigi.Config):  
    log_dir = luigi.Parameter()
```

```
class A(luigiWrapper.Task):  
    def requires(self):  
        return None  
  
    ...
```

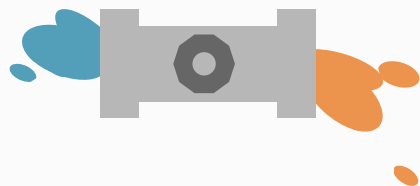
```
class B(luigiWrapper.Task):  
    def requires(self):  
        return A()  
  
    ...
```

```
luigi --module mi_modulo C --GlobalParams-log-dir './logs'
```

Patrones comunes en Luigi

Manejadores de eventos

Ejecutar código Python en diferentes puntos del ciclo de vida de la tarea.



```
import luigi
```

```
class Tarea(luigi.Task):
```

```
...
```

```
@Tarea.event_handler(luigi.Event.PROCESSING_TIME)
```

```
def task_procc_handler(task, elapsed):
```

```
    task.elapsed_time = elapsed
```

```
    # Mandar email con el tiempo de ejecución
```

```
@Tarea.event_handler(luigi.Event.START)
```

```
def task_start_handler(task):
```

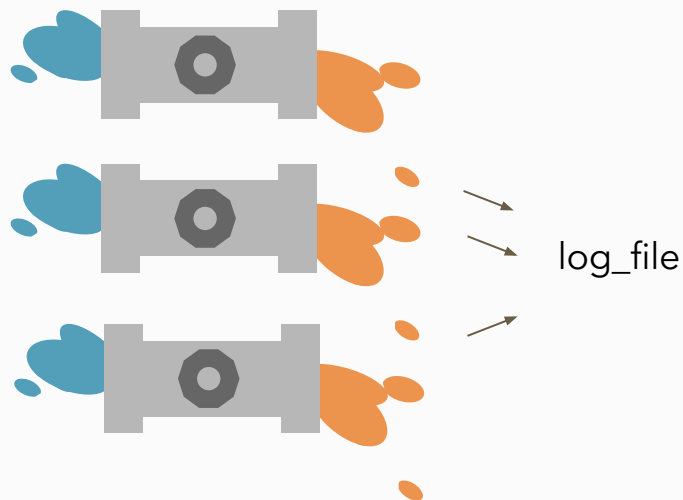
```
    task.dep_list = task.deps()
```

```
    task.start_time = datetime.utcnow()
```

Patrones comunes en Luigi

Manejo de concurrencia

Manejo sencillo de acceso concurrente a recursos compartidos.



```
import luigi

class Tarea(luigiWrapper.Task):
    @property
    def resources(self):
        return { GlobalParams().log_file: 1 }

    def requires(self):
        ...

    def run(self):
        ...

    def output(self):
        ...
```

Conclusiones

- Las data pipelines son soluciones software para el manejo eficaz de datos.
- Luigi es una alternativa liviana que provee un planificador de tareas, un portal para monitorizar, y útiles para facilitar la modularidad y homogeneidad en las tareas.
- Usando los diferentes tipos de tareas, recursos (targets) y patrones de uso, podemos crear cualquier tipo de flujo de procesamiento.

Muchas gracias

Recursos de interés

Repositorio con un compendio de data pipeline frameworks

<https://github.com/pditommaso/awesome-pipeline>

Comparativa de data pipeline frameworks

<https://towardsdatascience.com/data-pipelines-luigi-airflow-everything-you-need-to-know-18dc741449b7>

Repositorio con una estructura para proyectos de *data science* con Luigi

https://github.com/ffmmjj/luigi_data_science_project_cookiecutter