@rantav
totango

WHO AM I?

WHO AM I?

- A developer
  - Google, Microsoft, Outbrain, Gigaspaces, Totango etc
  - FLOSS: Hector, flask-restful-swager, meteor-migrations, monitoring…
  - Contributor to **Luigi**
- reversim.com
- Gormim

We help developers find their dream job!

Curated weekly email of top companies seeking talented **software engineers** in **Israel**

Enter your email...

Subscribe

Nir Dremer

Nir builds products at the intersection of technology, business & psychology. He founded MediaBox and led products at Imperva (NYSE: IMPV) & BitBand

Ran Tavory

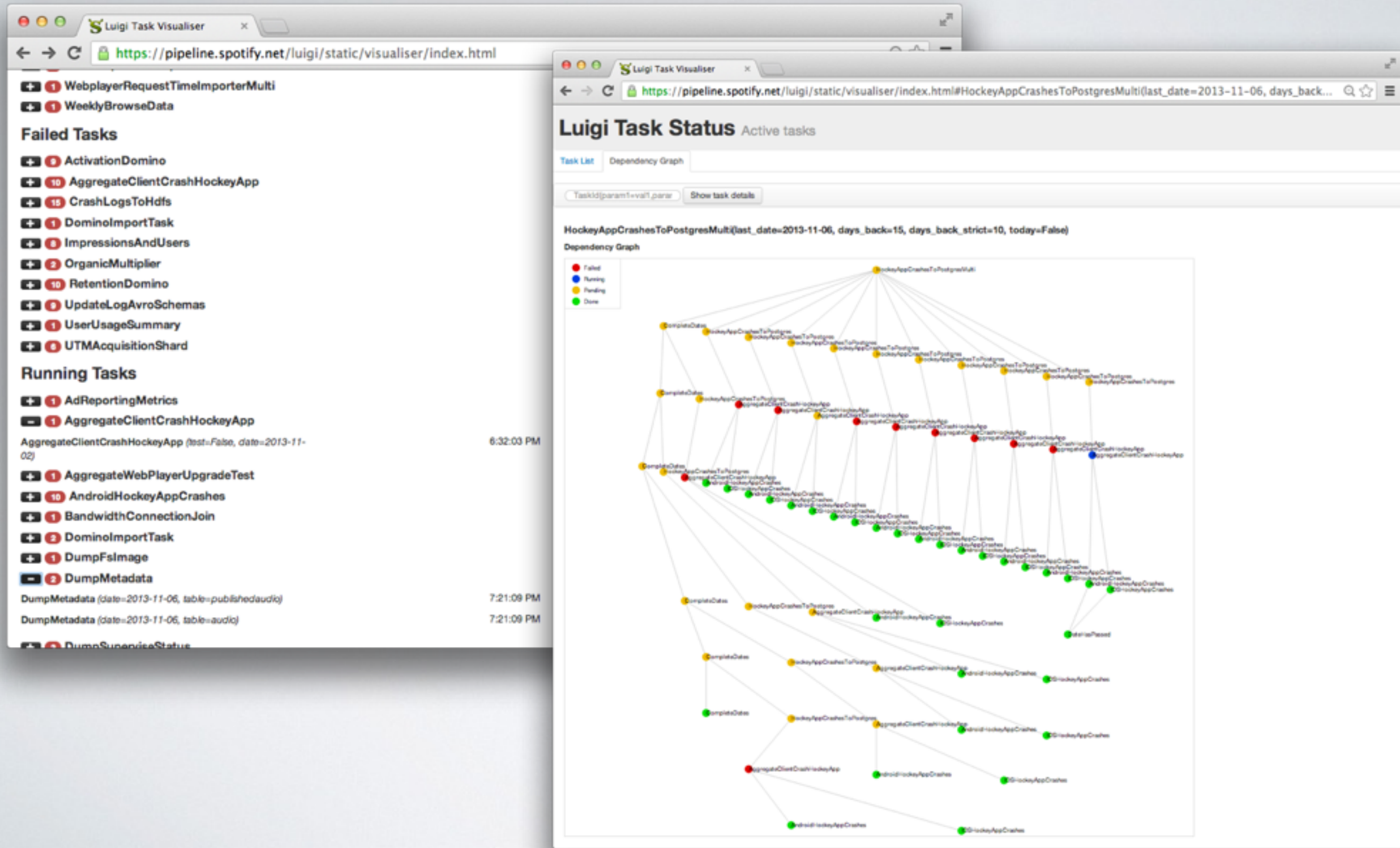
Ran is domain expert in web infrastructure and scaling. He founded SocialStudios and built products at Google, YouTube, Microsoft and Outbrain. Ran is

WHAT IS LUIGI?

WHAT IS LUIGI?

- A **Workflow Engine**.
  - Who the fuck needs a workflow engine?
- You do!!!
  - If you run hadoop (or other ETL jobs)
  - If you have dependencies b/w them (who doesn't?!)
  - If they fail (`s/if/when/`)
- Luigi doesn't replace Hadoop, Scalding, Pig, Hive, Redshift.
  - It **orchestrates** them

SCREENSHOTS



HOW DO **YOU** ETL YOUR DATA?

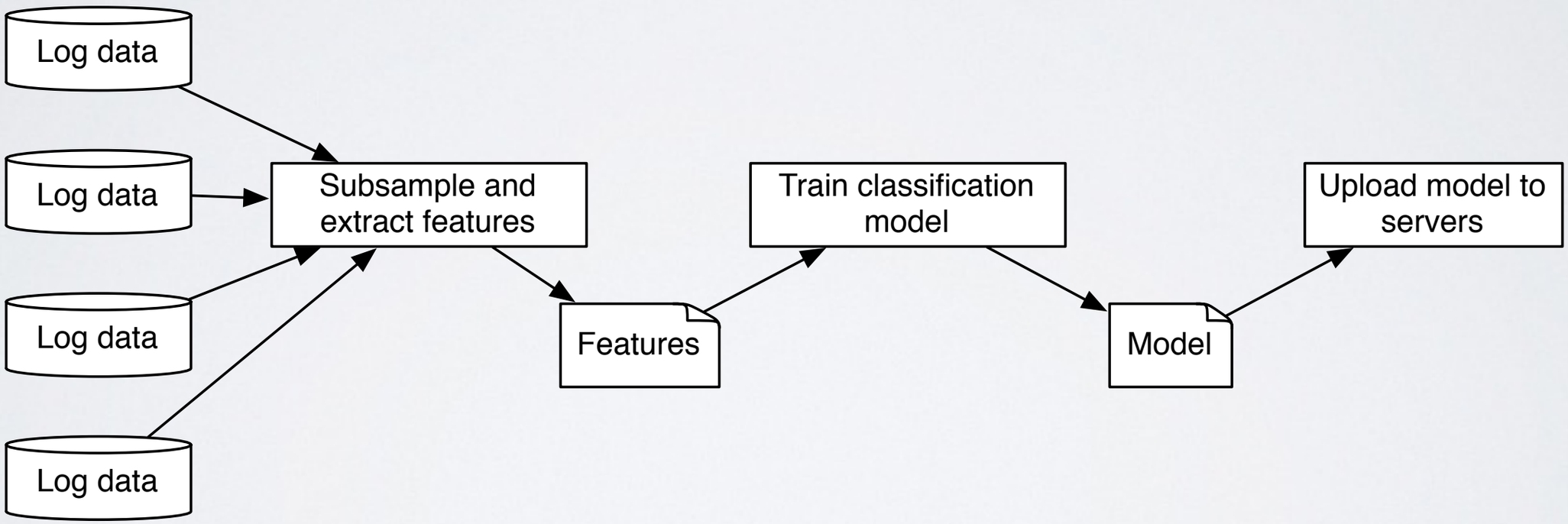
- Hadoop
- Spark
- Redshift
- Postgres
- Ad-hoc java/python/ruby/go/...

RUNNING **ONE** JOB IS EASY
RUNNING **MANY** IS HARD

- 100s of concurrent jobs, 1000s Daily.
- Job dependencies.
  - E.g. first copy the file, then crunch it.
- Errors / retries
  - Idempotency
- Monitoring / Visuals

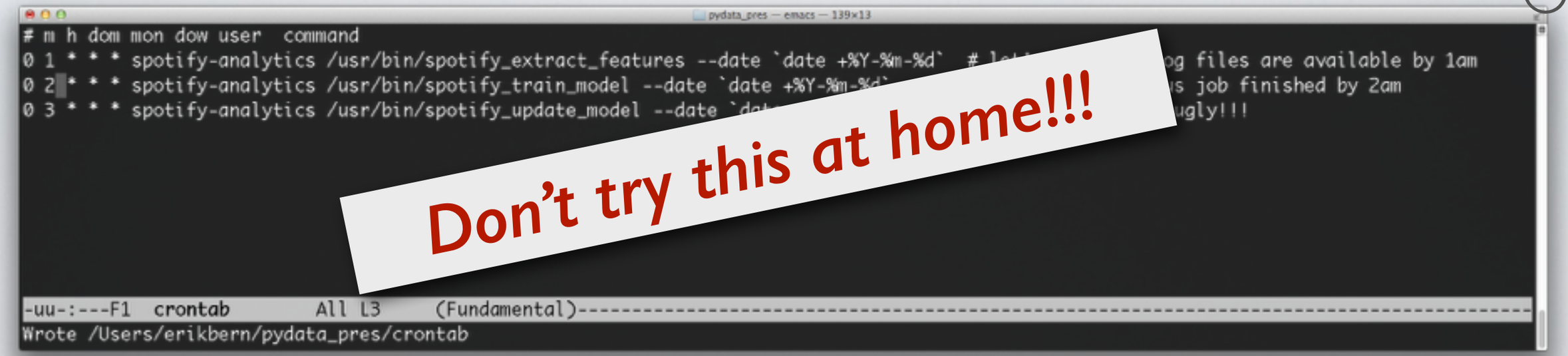
THE **WRONG** WAY TO DO IT

EXAMPLE WORKFLOW



THE CRON PHENOMENON

THE **WRONG** WAY TO DO IT



ENTER LUIGI

ENTER **LUIGI**

- Like Makefile - but in python
    - And - For data
- Integrates well with data targets
    - Hadoop, Spark, Databases
    - Atomic file/db operations
- Visualization
- CLI - really nice developer interface!

LUIGI TASK

```
class MyTask(Task):
    def output(self):
        pass

    def requires(self):
        pass

    def run(self):
        pass
```

LUIGI TASK

```
class AggregateArtists(Task):
    def output(self):
        return HdfsTarget("data/artist_streams.tsv")

    def requires(self):
        return Streams()

    def run(self):
        with self.input().open('r') as in_file:
            ... # read stuff from in_file

        with self.output().open('w') as out_file:
            ... # write stuff to out_file

if __name__ == "__main__":
    luigi.run()
```

Store output in HDFS

Run - actual work
Read from input
Write to output

Run with __main__

RUN FROM THE CLI

Run on the command line:

```
$ python dataflow.py Aggreg
DEBUG: Checking if AggregateArtists() is complete
INFO: Scheduled AggregateArtists()
DEBUG: Checking if Streams() is complete
INFO: Done scheduling tasks
DEBUG: Asking scheduler for work...
DEBUG: Pending tasks: 1
INFO: [pid 74375] Running   AggregateArtists()
INFO: [pid 74375] Done      AggregateArtists()
DEBUG: Asking scheduler for work...
INFO: Done
```

TASK PARAMETERS

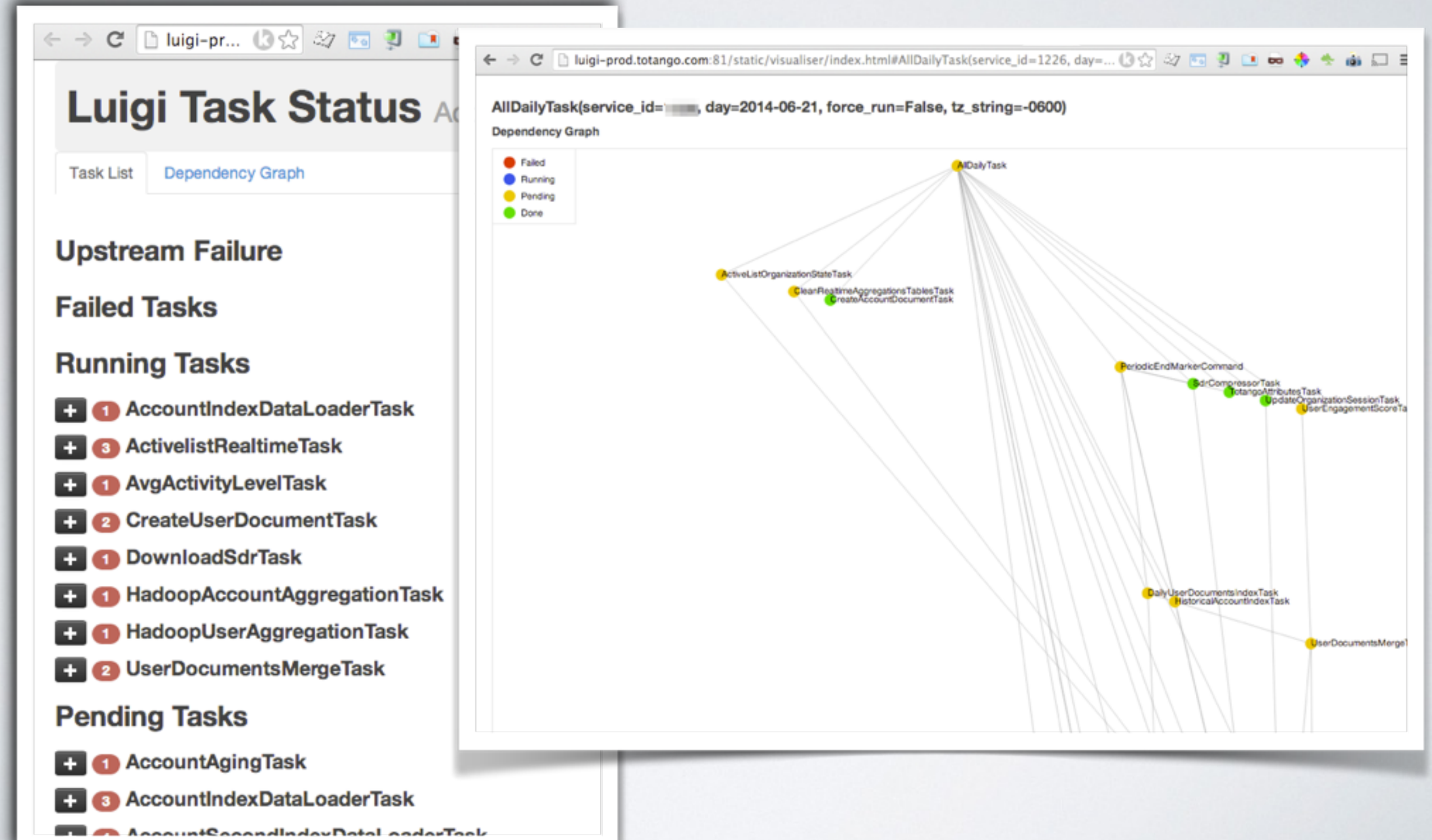
```
class AggregateArtists(Task):
    date = DateParameter()
```

```
$ python dataflow.py AggregateArtists --date 2013-03-05
```

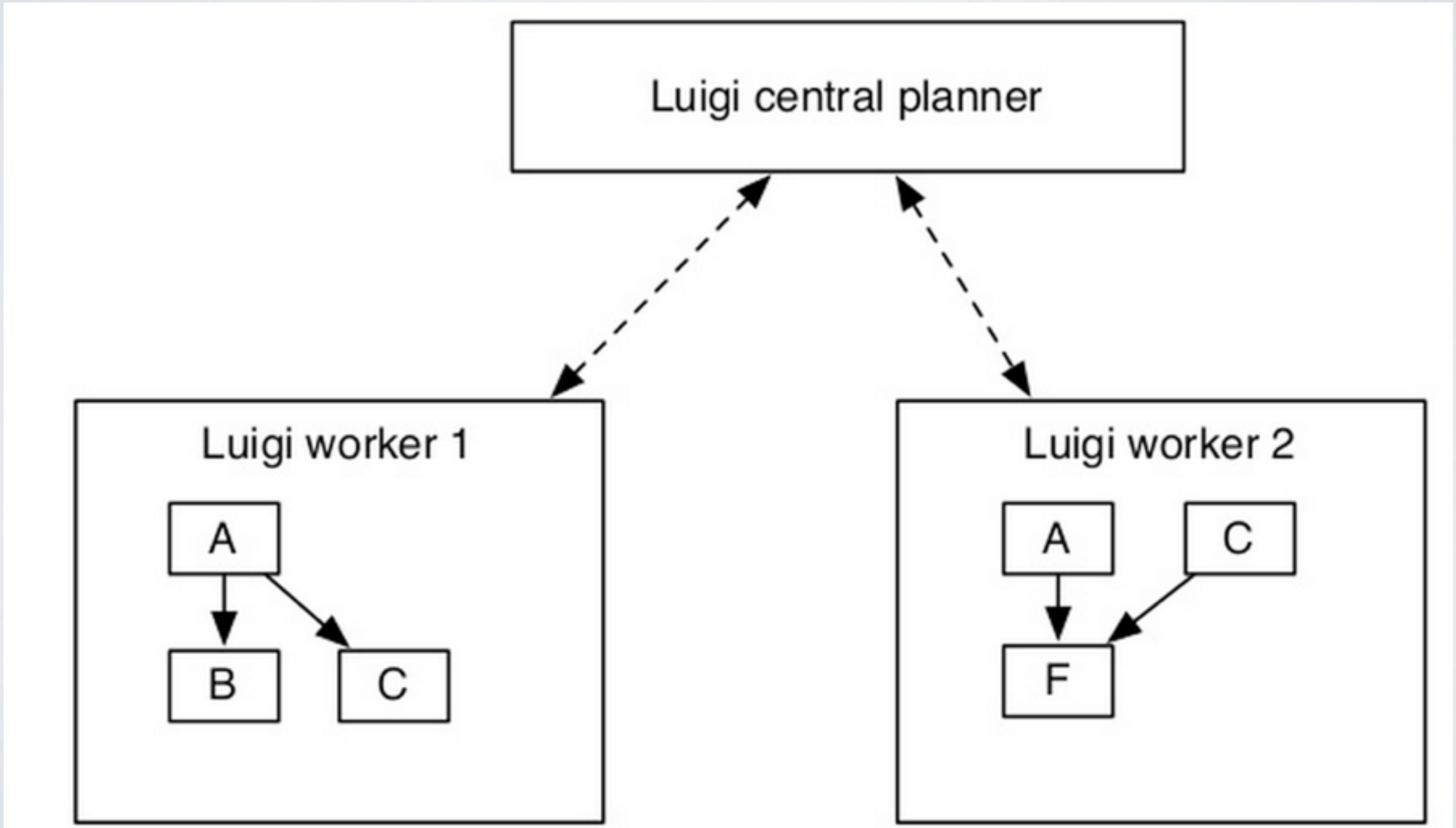
AWESOME HADOOP (MR) SUPPORT

```
class AggregateArtists(luigi.hadoop.JobTask):
    def requires(self):
        return Streams()

    def output(self):
        return HdfsTarget("data/artist_streams.tsv")

    def mapper(self, line):
        timestamp, artist, track = line.split('\t')
        yield artist, 1

    def reducer(self, artist, streams):
        yield artist, sum(streams)
```

WEB UI


luigi-pr...
Luigi Task Status
Task List
Dependency Graph
Upstream Failure
Failed Tasks
Running Tasks
1 AccountIndexDataLoaderTask
3 ActivelistRealtimeTask
1 AvgActivityLevelTask
2 CreateUserDocumentTask
1 DownloadSdrTask
1 HadoopAccountAggregationTask
1 HadoopUserAggregationTask
2 UserDocumentsMergeTask
Pending Tasks
1 AccountAgingTask
3 AccountIndexDataLoaderTask
luigi-prod.totango.com:81/static/visualiser/index.html#AllDailyTask(service_id=1226, day=...
AllDailyTask(service_id=[redacted], day=2014-06-21, force_run=False, tz_string=-0600)
Dependency Graph
Failed
Running
Pending
Done
AllDailyTask
ActiveListOrganizationStateTask
CleanRealtimeAggregationsTablesTask
CreateAccountDocumentTask
PeriodicEndMarkerCommand
SdrCompressorTask
TotangoAttributesTask
UpdateOrganizationSessionTask
DailyUserDocumentsIndexTask
HistoricalAccountIndexTask

PROCESS SYNCHRONIZATION



USED BY

CAPITAL ONE LABS

Spotify®

iHeart RADIO

foursquare®

bitly

MORTAR

VisualDNA™

UPPSALA UNIVERSITET

SEMI-DEEP DIVE

Programming for Luigi

LUIGI TASKS

- Implement 4 method:

```
def input(self) (optional)

def output(self)

def run(self)

def depends(self)
```

LUIGI TASKS

- Or extend one of the predefined tasks
  - **S3CopyToTable**
  - RedshiftManifestTask
  - SparkJob
  - HiveQueryTask
  - HadoopJobTask
  - …

EXAMPLE
LOCAL WORDCOUNT

```
class WordCount(luigi.Task):
    date_interval = luigi.DateIntervalParameter()

    def requires(self):
        return [InputText(date) for date in self.date_interval.dates()]

    def output(self):
        return luigi.LocalTarget('/var/tmp/text-count/%s' % self.date_interval)

    def run(self):
        count = {}
        for file in self.input():
            for line in file.open('r'):
                for word in line.strip().split():
                    count[word] = count.get(word, 0) + 1

        # output data
        f = self.output().open('w')
        for word, count in count.iteritems():
            f.write("%s\t%d\n" % (word, count))
        f.close()
```

EXAMPLE
HADOOP WORDCOUNT

```
class WordCount(luigi.hadoop.JobTask):
    date_interval = luigi.DateIntervalParameter()

    def requires(self):
        return [InputText(date) for date in self.date_interval.dates()]

    def output(self):
        return luigi.hdfs.HdfsTarget('/tmp/text-count/%s' % self.date_interval)

    def mapper(self, line):
        for word in line.strip().split():
            yield word, 1

    def reducer(self, key, values):
        yield key, sum(values)
```

LUIGI TARGETS

- HDFS
- Local File
- Postgres / MySQL, Redshift, ElasticSearch
- ... Easy to extend

DEFINING A **TARGET**

- Implement:

```
def exists(self)
```

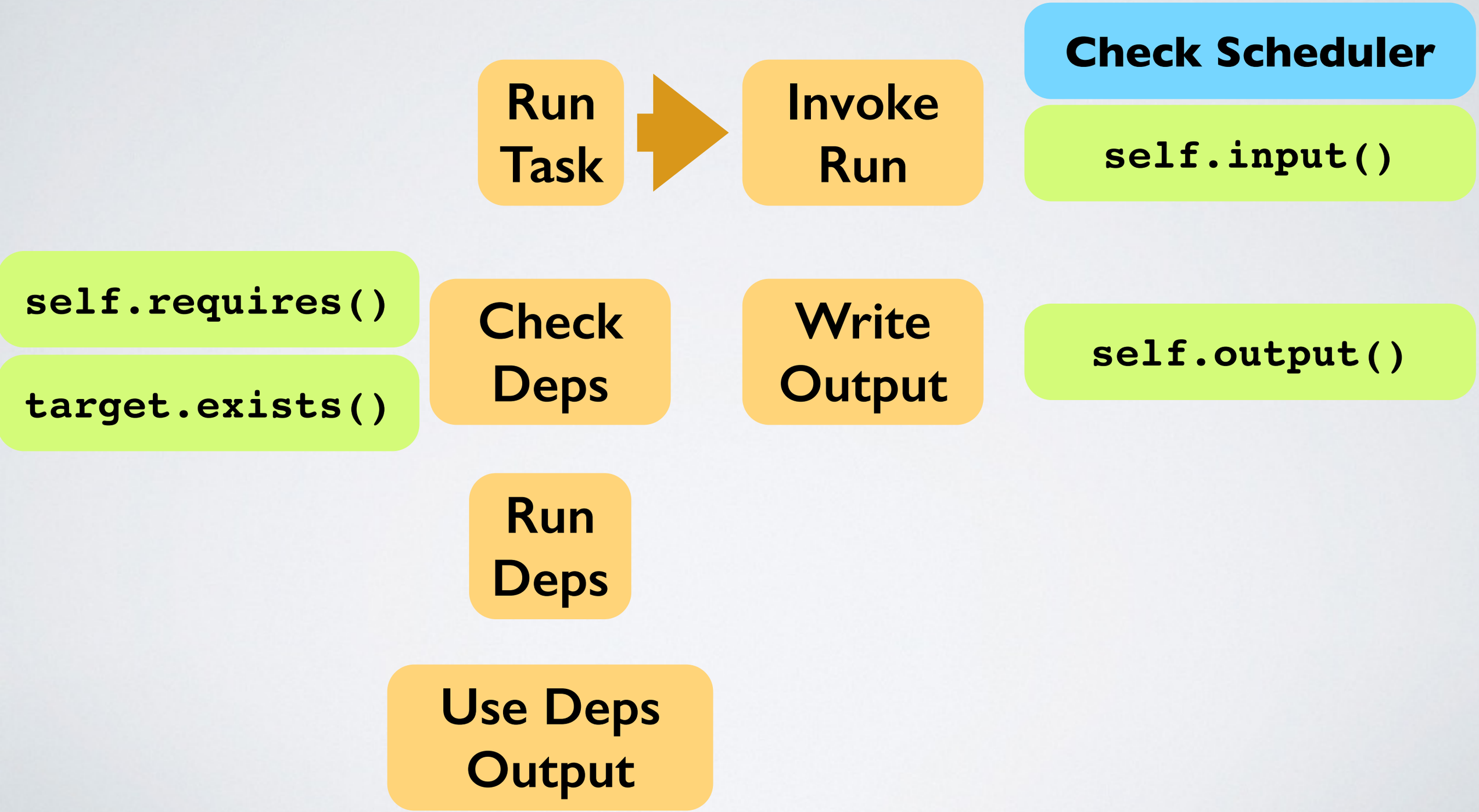
And optionally:

`connect` or **`open / close`**

EXAMPLE MYSQL TARGET

```
class MySqlTarget(luigi.Target):

    def touch(self, connection=None):
      ...

    def exists(self, connection=None):
      cursor = connection.cursor()
      cursor.execute("""SELECT 1 FROM {marker_table}
                        WHERE update_id = %s
                        LIMIT 1""".format(marker_table=self.marker_table),
                     (self.update_id,)
      )
      row = cursor.fetchone()
      return row is not None

  def connect(self, autocommit=False):
    ...

  def create_marker_table(self):
    ...
```

THE GRAND SCHEME



OPEN SOURCE

BY

Spotify™

authors, spotify

erikbern #1
188 commits / 9,116 ++ / 4,218 --

freider #2
115 commits / 6,142 ++ / 2,083 --

That's Joe!

jcrobak #3
23 commits / 2,661 ++ / 949 --

That's me!

rantav #4
13 commits / 967 ++ / 510 --

JonNelson #5
9 commits / 129 ++ / 65 --

miku #6
7 commits / 724 ++ / 7 --

WHAT DID I DO?

- Add **Redshift** support
- Add **MySQL** support
- Various small features (improved **notifications**, **dep.py, historydb** etc)
- Various bug reports
  - And fixes!

LUIGI @ TOTANGO

- Daily computation
- Hourly computation
- Ad-hoc data loading (for data analysis activities, to redshift)

TOTANGO'S SETUP

Luigi Scheduler

invoke

Luigi Workers

coordinate

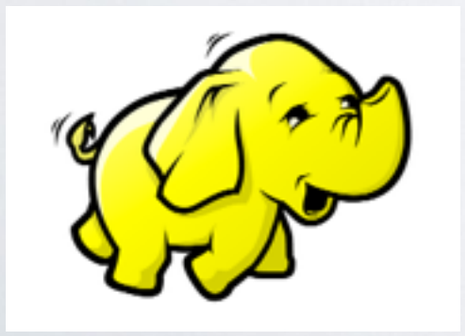
report

librato™

logentries

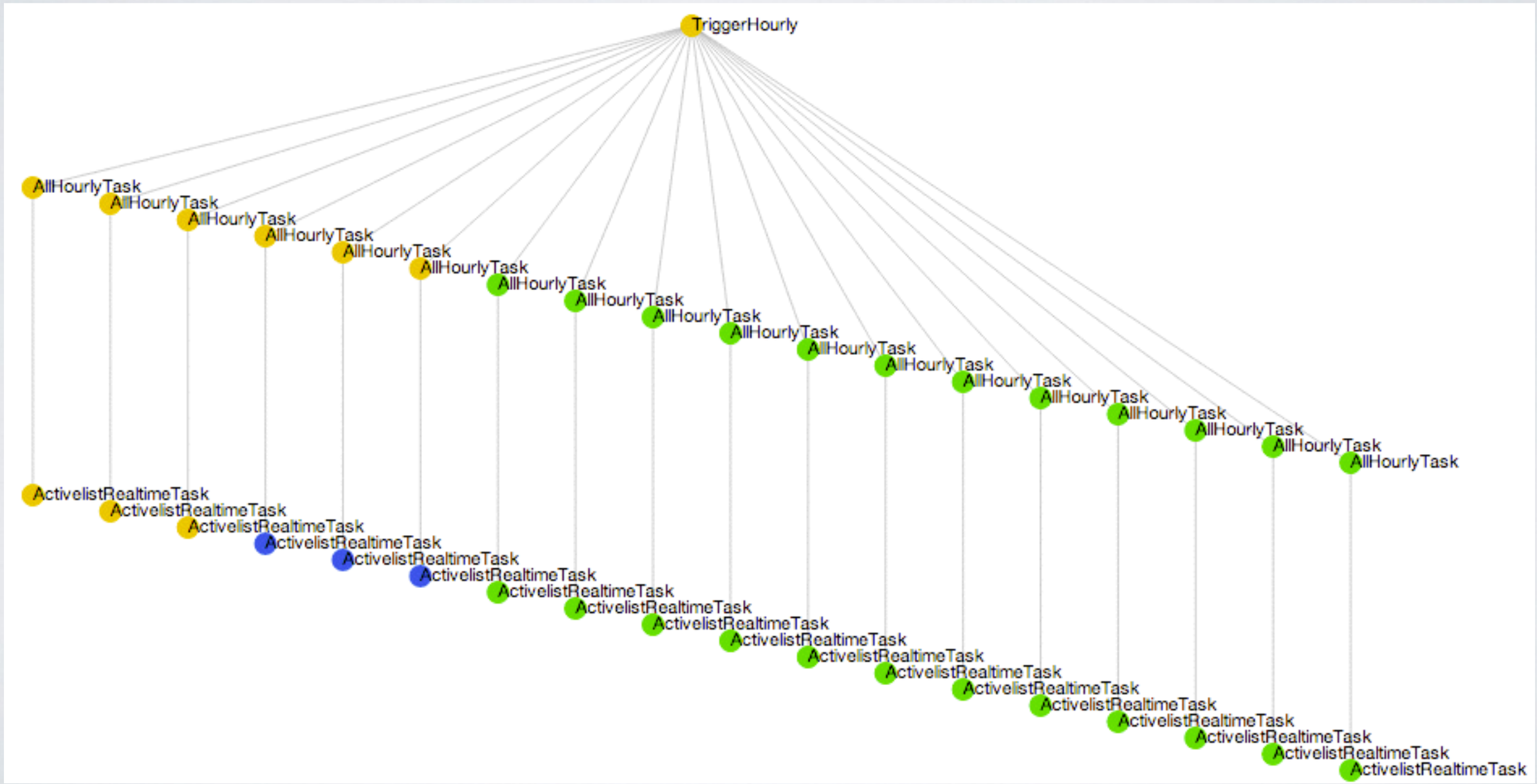
TOTANGO'S SETUP

Luigi Worker

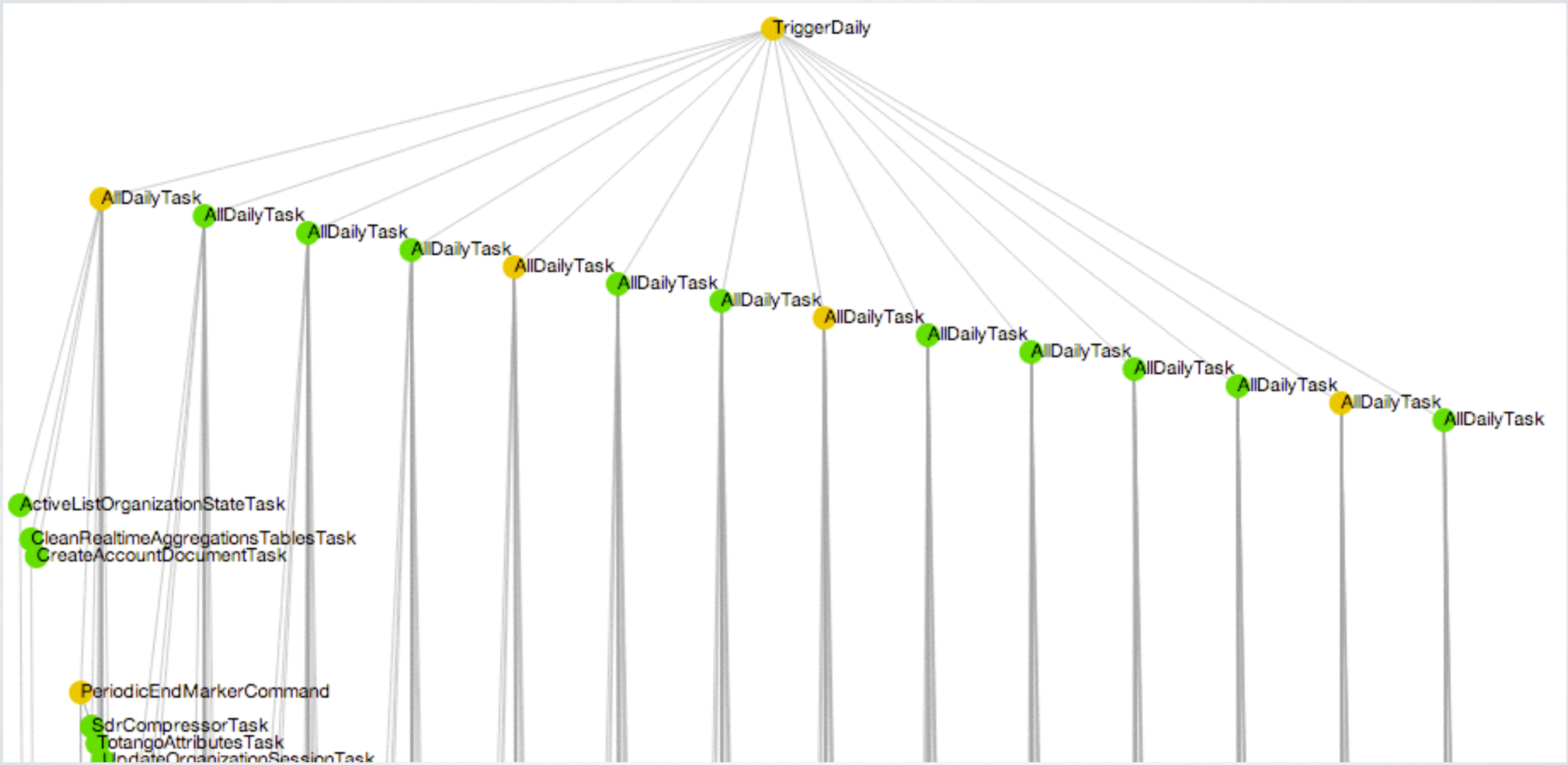


MOAR SCREENSHOTS

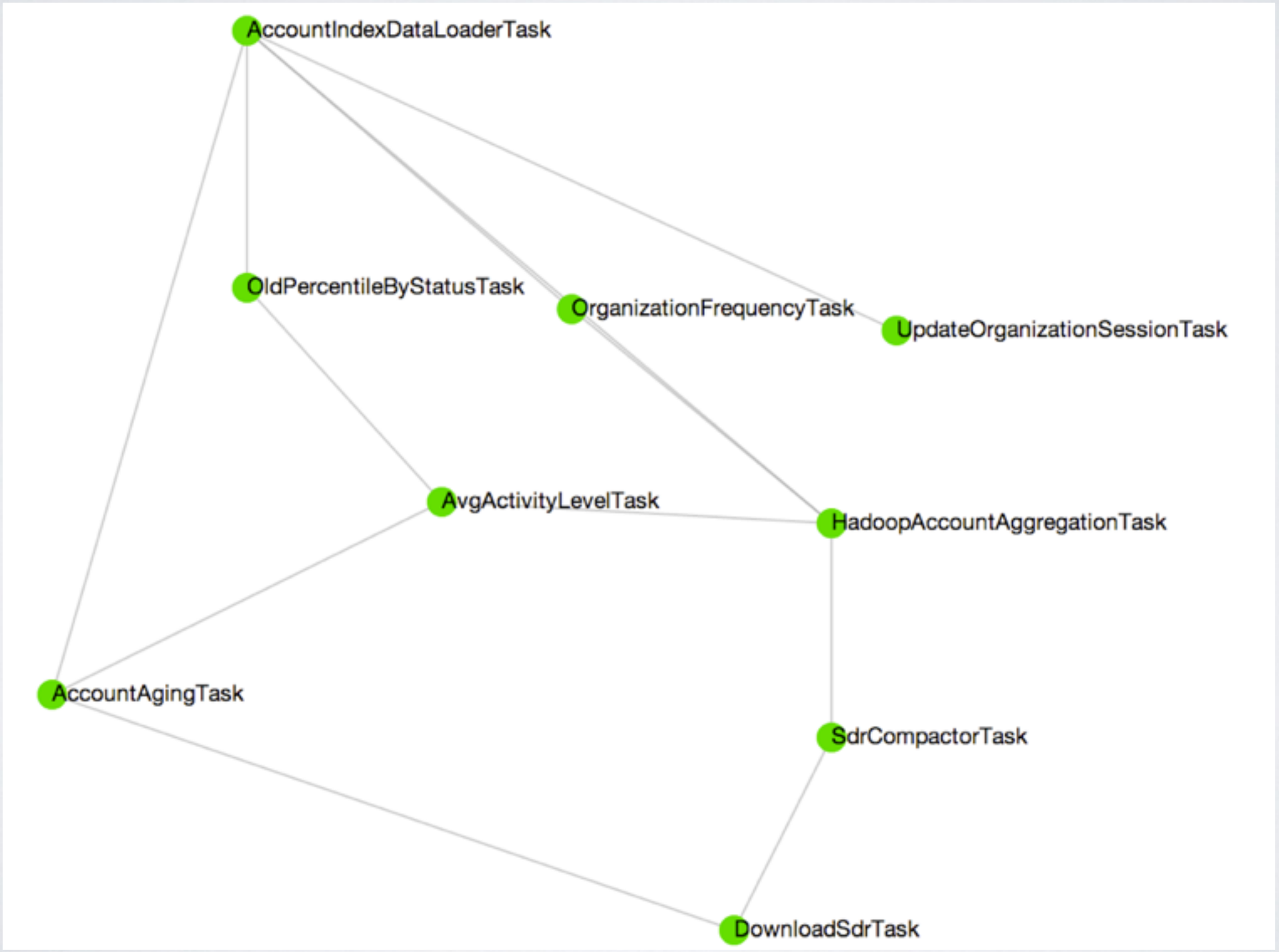
MOAR SCREENSHOTS



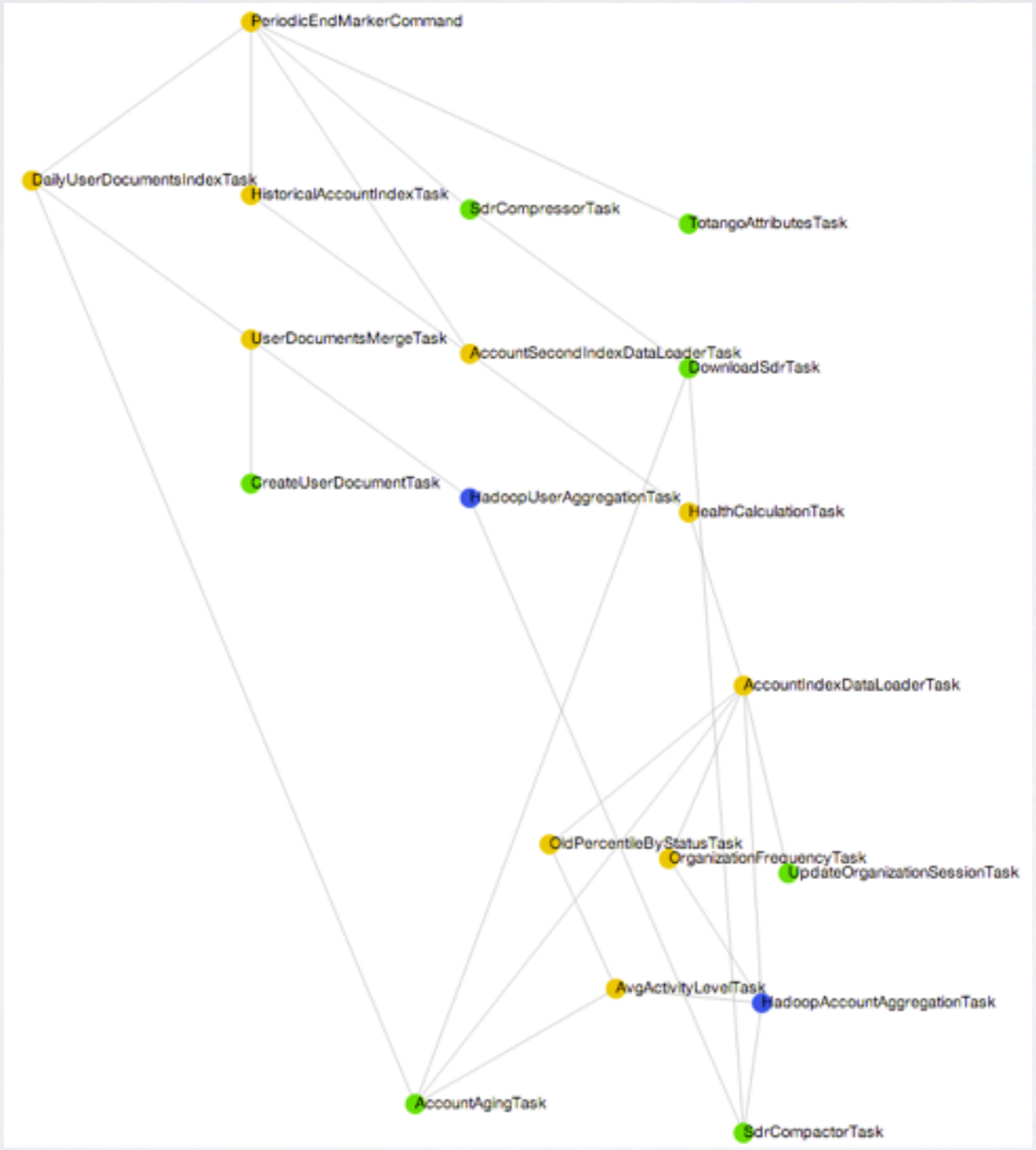
MOAR SCREENSHOTS



MOAR SCREENSHOTS

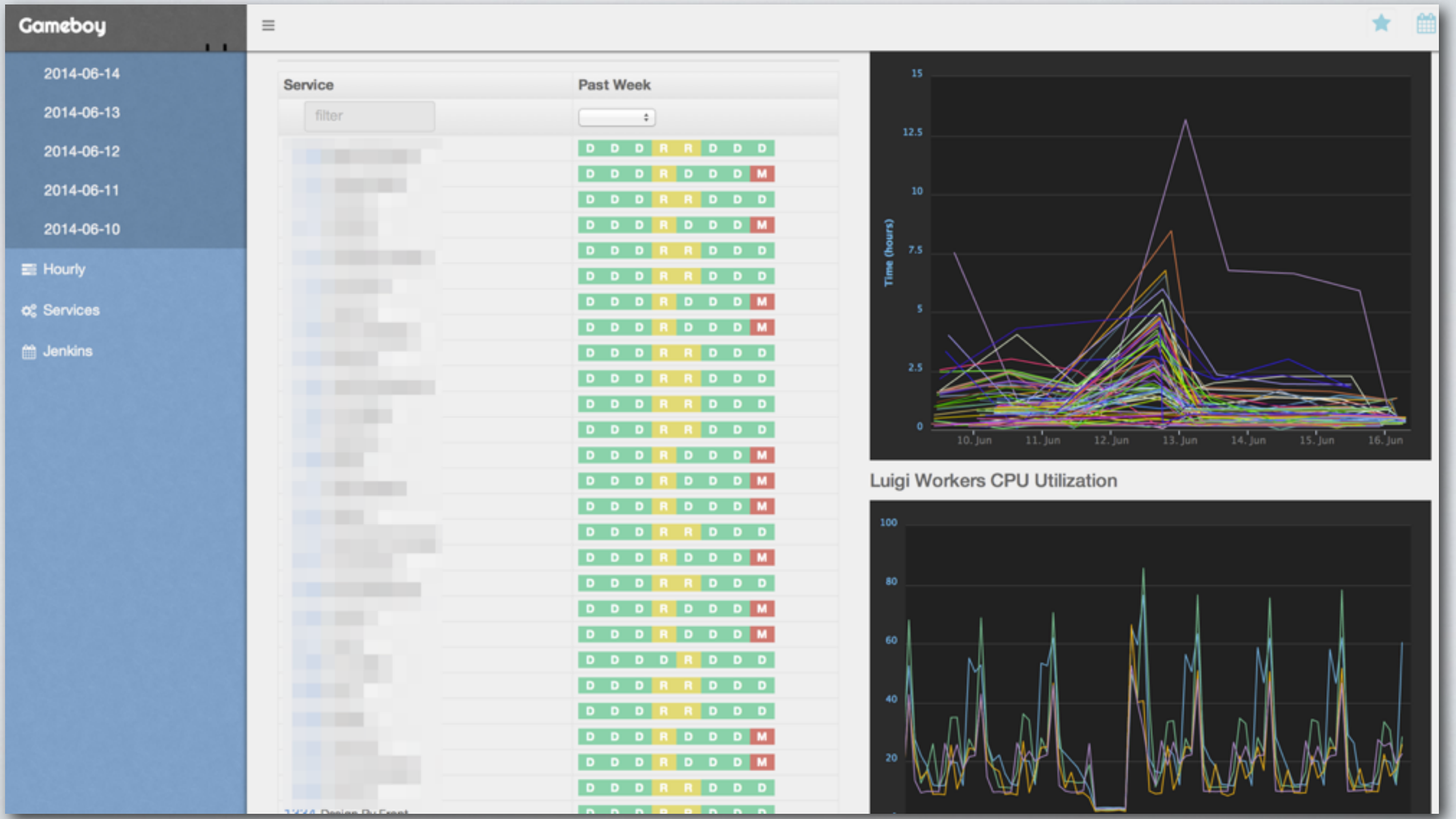


MOAR SCREENSHOTS

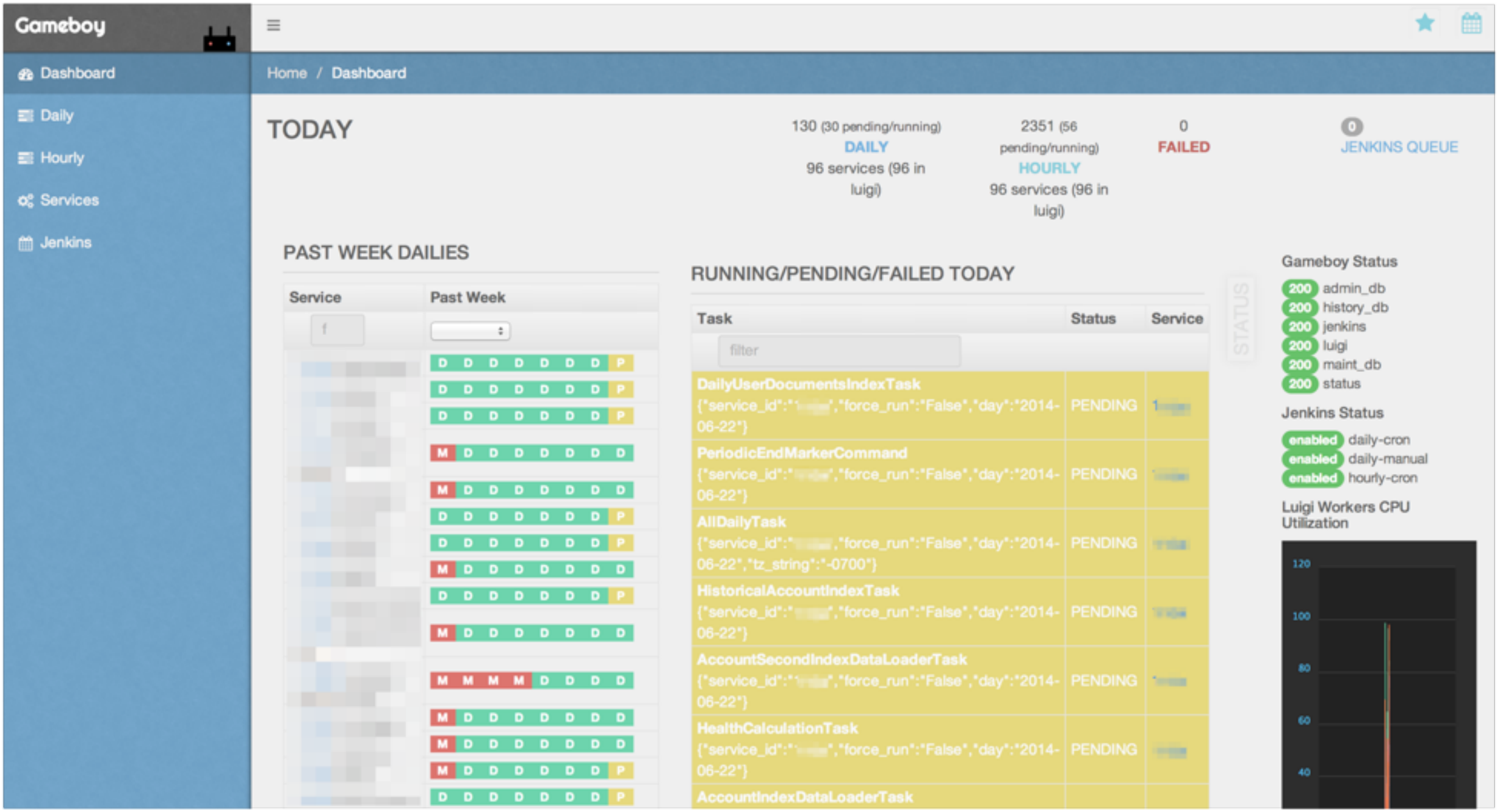


GAMEBOY!!!

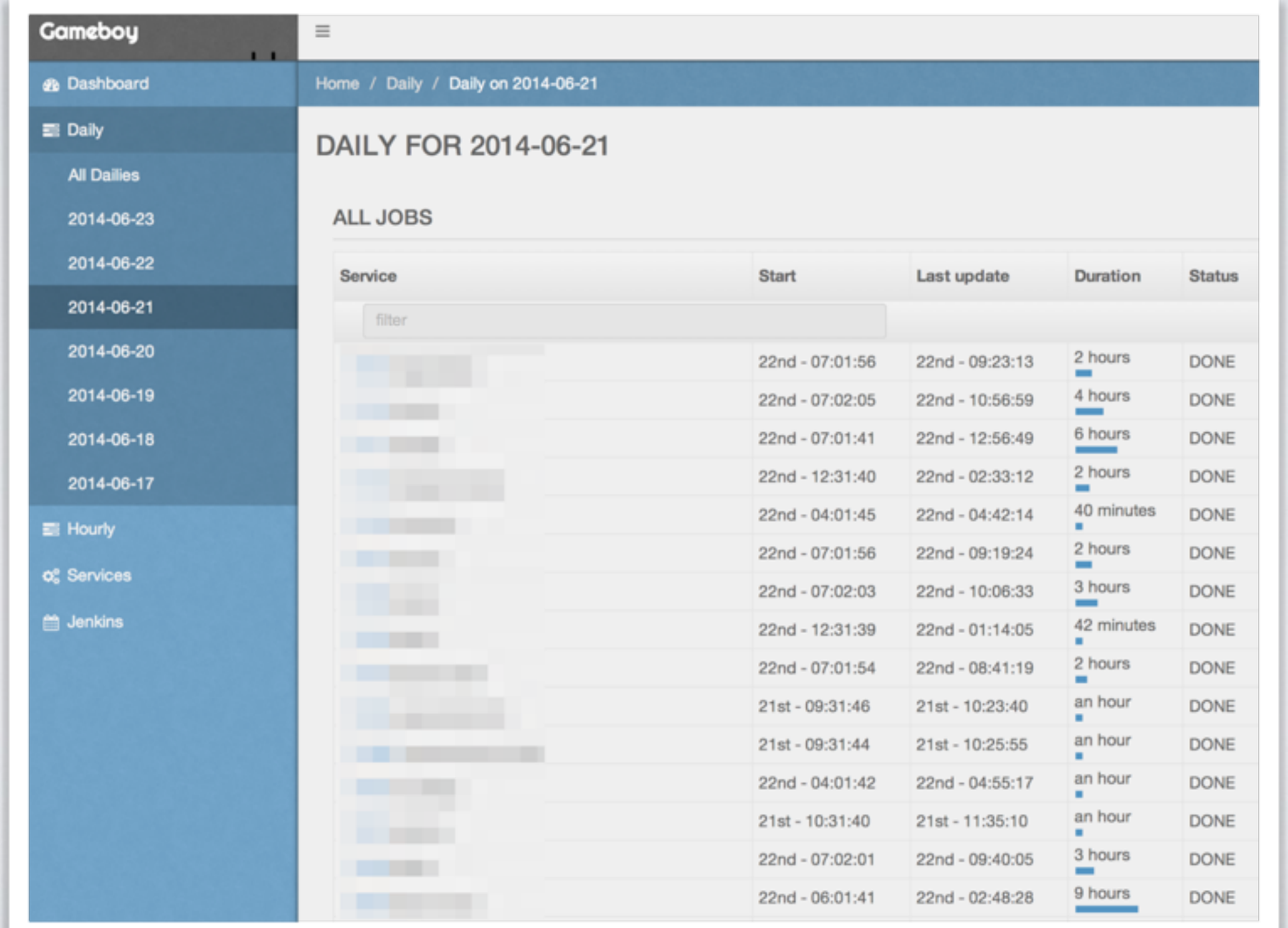
AND... **GAMEBOY**


Gameboy
2014-06-14
2014-06-13
2014-06-12
2014-06-11
2014-06-10
Hourly
Services
Jenkins
Service
Past Week
filter
Time (hours)
15
12.5
10
7.5
5
2.5
0
10. Jun
11. Jun
12. Jun
13. Jun
14. Jun
15. Jun
16. Jun
Luigi Workers CPU Utilization
100
80
60
40
20

GAMEBOY


Gameboy
Dashboard
Daily
Hourly
Services
Jenkins
Home / Dashboard
TODAY
130 (30 pending/running)
DAILY
96 services (96 in luigi)
2351 (56 pending/running)
HOURLY
96 services (96 in luigi)
0
FAILED
0
JENKINS QUEUE
PAST WEEK DAILIES
Service
Past Week
RUNNING/PENDING/FAILED TODAY
Task
Status
Service
filter
DailyUserDocumentsIndexTask {"service_id":"...","force_run":"False","day":"2014-06-22"} PENDING
PeriodicEndMarkerCommand {"service_id":"...","force_run":"False","day":"2014-06-22"} PENDING
AllDailyTask {"service_id":"...","force_run":"False","day":"2014-06-22","tz_string":"-0700"} PENDING
HistoricalAccountIndexTask {"service_id":"...","force_run":"False","day":"2014-06-22"} PENDING
AccountSecondIndexDataLoaderTask {"service_id":"...","force_run":"False","day":"2014-06-22"} PENDING
HealthCalculationTask {"service_id":"...","force_run":"False","day":"2014-06-22"} PENDING
AccountIndexDataLoaderTask
STATUS
Gameboy Status
200 admin_db
200 history_db
200 jenkins
200 luigi
200 maint_db
200 status
Jenkins Status
enabled daily-cron
enabled daily-manual
enabled hourly-cron
Luigi Workers CPU Utilization
120
100
80
60
40

GAMEBOY


Gameboy
Dashboard
Daily
All Dailies
2014-06-23
2014-06-22
2014-06-21
2014-06-20
2014-06-19
2014-06-18
2014-06-17
Hourly
Services
Jenkins
Home / Daily / Daily on 2014-06-21
DAILY FOR 2014-06-21
ALL JOBS
Service
Start
Last update
Duration
Status
filter
22nd - 07:01:56
22nd - 09:23:13
2 hours
DONE
22nd - 07:02:05
22nd - 10:56:59
4 hours
DONE
22nd - 07:01:41
22nd - 12:56:49
6 hours
DONE
22nd - 12:31:40
22nd - 02:33:12
2 hours
DONE
22nd - 04:01:45
22nd - 04:42:14
40 minutes
DONE
22nd - 07:01:56
22nd - 09:19:24
2 hours
DONE
22nd - 07:02:03
22nd - 10:06:33
3 hours
DONE
22nd - 12:31:39
22nd - 01:14:05
42 minutes
DONE
22nd - 07:01:54
22nd - 08:41:19
2 hours
DONE
21st - 09:31:46
21st - 10:23:40
an hour
DONE
21st - 09:31:44
21st - 10:25:55
an hour
DONE
22nd - 04:01:42
22nd - 04:55:17
an hour
DONE
21st - 10:31:40
21st - 11:35:10
an hour
DONE
22nd - 07:02:01
22nd - 09:40:05
3 hours
DONE
22nd - 06:01:41
22nd - 02:48:28
9 hours
DONE

GAMEBOY IS

- A Totango specific controller for Luigi
  - The transition process (to Luigi)
  - Provide **high level** overview
  - Manual **re-run** of tasks
  - Monitor **progress**, **performance**, **run times**, **queues**, **worker load** etc...
  - Implemented using **Flask** and **AngularJS**

WHAT ELSE IS OUT THERE?

WHAT ELSE IS OUT THERE?

- Oozie
- Azkaban
- AWS Data Pipeline
- Chronos
- spring-batch
- Dataswarm (facebook)
- River (outbrain internal)
- What's your favorite WF engine? (did you build one?)

REFS

- https://github.com/spotify/luigi
- Facebook's Dataswarm https://www.youtube.com/watch?v=M0VCbhfQ3HQ
- Outbrain's River https://www.youtube.com/watch?v=EzsckTggDiM