

Testing PDI solutions

Slawomir Chodnicki
BI Consultant

slawo@twineworks.com

The sample project

```
.  
|-- bin          # entry point scripts  
|-- environments # environment configuration  
|-- etl         # ETL solution  
`-- spec        # tests and helpers
```

Test orchestration

```
$ bin/robot test
```

λ:solution \$ bin/robot test

Run options: exclude {:long_running=>true, :remote=>true, :integration=>true}

db clear command

when db is not empty

exits with exit code 0

clears the db

db reset command

when db is not empty

exits with exit code 0

clears the db

creates a dim_date dimension

including 2000-01-01

including 2015-12-31

creates a dim_time dimension

including 00 AM

including 11 PM

including -1/NA

creates an iso_countries dimension

including US

including DE

ETL

etl/spec/daily/daily_0000_00_00_spec.kjb

completes successfully

etl/spec/daily/daily_2001_01_01_spec.kjb

completes successfully

etl/spec/daily/daily_2017_06_20_reload_spec.kjb

completes successfully

etl/spec/daily/daily_2017_06_20_spec.kjb

```
etl/spec/daily/daily_2001_01_01_spec.kjb
  completes successfully
etl/spec/daily/daily_2017_06_20_reload_spec.kjb
  completes successfully
etl/spec/daily/daily_2017_06_20_spec.kjb
  completes successfully
etl/spec/daily/daily_2017_06_20_to_21_spec.kjb
  completes successfully
etl/spec/dummy/dummy_spec.kjb - verifies the test suite runs ETL as tests
  completes successfully
etl/spec/dwh/load/load_spec.kjb
  completes successfully
etl/spec/dwh/pre_flight/pre_flight_empty_file_spec.kjb
  completes successfully
etl/spec/dwh/pre_flight/pre_flight_missing_file_spec.kjb
  completes successfully
etl/spec/dwh/pre_flight/pre_flight_ok_spec.kjb
  completes successfully
etl/spec/dwh/stage/stage_spec.kjb
  completes successfully
etl/spec/dwh/validate_params/validate_params_spec.kjb
  completes successfully
etl/spec/environment/mysql_driver_spec.kjb
  completes successfully
etl/spec/environment/robot_dir_is_set_spec.kjb - ROBOT_DIR variable is set correctly
  completes successfully
etl/spec/util/day_sequence/day_sequence_spec.kjb
  completes successfully
etl/spec/util/string_cleaner/string_cleaner_spec.kjb - identifiers are cleaned according to project standards
  completes successfully
```









```
etl/spec/dummy/dummy_spec.kjb - verifies the test suite runs ETL as tests
  completes successfully
etl/spec/dwh/load/load_spec.kjb
  completes successfully
etl/spec/dwh/pre_flight/pre_flight_empty_file_spec.kjb
  completes successfully
etl/spec/dwh/pre_flight/pre_flight_missing_file_spec.kjb
  completes successfully
etl/spec/dwh/pre_flight/pre_flight_ok_spec.kjb
  completes successfully
etl/spec/dwh/stage/stage_spec.kjb
  completes successfully
etl/spec/dwh/validate_params/validate_params_spec.kjb
  completes successfully
etl/spec/environment/mysql_driver_spec.kjb
  completes successfully
etl/spec/environment/robot_dir_is_set_spec.kjb - ROBOT_DIR variable is set correctly
  completes successfully
etl/spec/util/day_sequence/day_sequence_spec.kjb
  completes successfully
etl/spec/util/string_cleaner/string_cleaner_spec.kjb - identifiers are cleaned according to project standards
  completes successfully
```

```
jdbc_helper
  dwh_db
    when selecting from steelwheels.customers
      has 126 customers
```

Finished in 1 minute 8.62 seconds (files took 1.25 seconds to load)

28 examples, 0 failures

[Back to Dashboard](#)
[Status](#)
[Changes](#)
[Workspace](#)
[Build Now](#)
[Delete Project](#)
[Configure](#)
[Git Polling Log](#)
Build History
[trend](#)

 #20	Oct 30, 2017 1:35 PM
 #19	Oct 30, 2017 1:34 PM
 #18	Oct 30, 2017 1:32 PM
 #17	Oct 30, 2017 1:30 PM
 #16	Oct 30, 2017 1:27 PM
 #15	Jun 23, 2017 8:24 PM
 #14	Jun 23, 2017 8:14 PM
 #13	Jun 23, 2017 8:13 PM

Project etl-project


[Workspace](#)

[Last Successful Artifacts](#)

[Recent Changes](#)

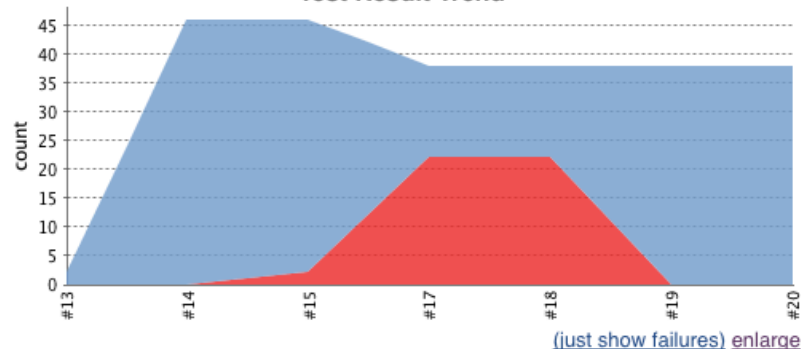
[Latest Test Result \(no failures\)](#)

Permalinks

- [Last build \(#20\), 2 min 23 sec ago](#)
- [Last stable build \(#20\), 2 min 23 sec ago](#)
- [Last successful build \(#20\), 2 min 23 sec ago](#)
- [Last failed build \(#18\), 5 min 38 sec ago](#)
- [Last unsuccessful build \(#18\), 5 min 38 sec ago](#)
- [Last completed build \(#20\), 2 min 23 sec ago](#)

[add description](#)
[Disable Project](#)

Test Result Trend





- [Back to Project](#)
- [Status](#)
- [Changes](#)
- [Console Output](#)
- [View as plain text](#)
- [Edit Build Information](#)
- [History](#)
- [Git Build Data](#)
- [No Tags](#)
- [Test Result](#)**
- [Previous Build](#)
- [Next Build](#)

Test Result

22 failures (±0)



38 tests (±0)

[Took 39 sec.](#)

[add description](#)

All Failed Tests

Test Name	Duration	Age
+ spec.commands.db.reset spec.db reset command when db is not empty exits with exit code 0	18 ms	2
- spec.commands.db.reset spec.db reset command when db is not empty creates a dim date dimension including 2000-01-01		
<ul style="list-style-type: none"> - Error Details Table 'dwh.dim_date' doesn't exist + Stack Trace 	2 ms	2
+ spec.commands.db.reset spec.db reset command when db is not empty creates a dim date dimension including 2015-12-31	1 ms	2
+ spec.commands.db.reset spec.db reset command when db is not empty creates a dim time dimension including 00 AM	2 ms	2
+ spec.commands.db.reset spec.db reset command when db is not empty creates a dim time dimension including 11 PM	2 ms	2
+ spec.commands.db.reset spec.db reset command when db is not empty creates a dim time dimension including -1/NA	2 ms	2
+ spec.commands.db.reset spec.db reset command when db is not empty creates an iso countries dimension including US	2 ms	2
+ spec.commands.db.reset spec.db reset command when db is not empty creates an iso countries dimension including DE	2 ms	2
+ spec.etl.etl spec.ETL etl/spec/daily/daily 0000 00 00 spec.kjb completes successfully	4.4 sec	2
+ spec.etl.etl spec.ETL etl/spec/daily/daily 2001 01 01 spec.kjb completes successfully	2.8 sec	2
+ spec.etl.etl spec.ETL etl/spec/daily/daily 2017 06 20 reload spec.kjb completes successfully	2.4 sec	2
+ spec.etl.etl spec.ETL etl/spec/daily/daily 2017 06 20 spec.kjb completes successfully	2.5 sec	2
+ spec.etl.etl spec.ETL etl/spec/daily/daily 2017 06 20 to 21 spec.kjb completes successfully	2.3 sec	2

[← Previous Build](#)[Next Build →](#)

Stack Trace

spec.commands.db.reset spec.db reset command when db is not empty creates a dim date dimension including 2015-12-31	1 ms	2
spec.commands.db.reset spec.db reset command when db is not empty creates a dim time dimension including 00 AM	2 ms	2
spec.commands.db.reset spec.db reset command when db is not empty creates a dim time dimension including 11 PM	2 ms	2
spec.commands.db.reset spec.db reset command when db is not empty creates a dim time dimension including -1/NA	2 ms	2
spec.commands.db.reset spec.db reset command when db is not empty creates an iso countries dimension including US	2 ms	2
spec.commands.db.reset spec.db reset command when db is not empty creates an iso countries dimension including DE	2 ms	2
spec.etl.etl spec.ETL etl/spec/daily/daily 0000 00 00 spec.kjb completes successfully	4.4 sec	2
spec.etl.etl spec.ETL etl/spec/daily/daily 2001 01 01 spec.kjb completes successfully	2.8 sec	2
spec.etl.etl spec.ETL etl/spec/daily/daily 2017 06 20 reload spec.kjb completes successfully	2.4 sec	2
spec.etl.etl spec.ETL etl/spec/daily/daily 2017 06 20 spec.kjb completes successfully	2.5 sec	2
spec.etl.etl spec.ETL etl/spec/daily/daily 2017 06 20 to 21 spec.kjb completes successfully	2.3 sec	2
spec.etl.etl spec.ETL etl/spec/dwh/load/load spec.kjb completes successfully	2.2 sec	2
spec.etl.etl spec.ETL etl/spec/dwh/stage/stage reload spec.kjb completes successfully	2.2 sec	2
spec.etl.etl spec.ETL etl/spec/dwh/stage/stage spec.kjb completes successfully	2.7 sec	2
spec.etl.etl spec.ETL etl/spec/dwh/validate params/validate params spec.kjb completes successfully	2.2 sec	2
spec.etl.etl spec.ETL etl/spec/util/day sequence/day sequence spec.kjb completes successfully	2.2 sec	2
spec.load.load spec.load phase with fixture: spec/fixtures/load/2017-06-21.json exits with exit code 0	0 ms	2
spec.load.load spec.load phase with fixture: spec/fixtures/load/2017-06-21.json aggregates Gulliver records	0 ms	2
spec.load.load spec.load phase with fixture: spec/fixtures/load/2017-06-21.json transforms user '50 pence' to user tag ' 50_pence'	0 ms	2
spec.load.load spec.load phase with fixture: spec/fixtures/load/2017-06-21.json inserts Libslack into the service table	0 ms	2

All Tests

Package	Duration	Fail	(diff) Skip	(diff) Pass	(diff) Total	(diff)
spec.commands.db	1 min 18 sec	8	0	3	11	
spec.demo	39 sec	0	0	3	3	
spec.environment	39 sec	0	0	2	2	
spec.etl	39 sec	10	0	7	17	

spec.commands.db.reset spec.db reset command when db is not empty creates a dim time dimension including -1/VA	2 ms	2
spec.commands.db.reset spec.db reset command when db is not empty creates an iso countries dimension including US	2 ms	2
spec.commands.db.reset spec.db reset command when db is not empty creates an iso countries dimension including DE	2 ms	2
spec.etl.etl spec.ETL etl/spec/daily/daily 0000 00 00 spec.kjb completes successfully	4.4 sec	2
spec.etl.etl spec.ETL etl/spec/daily/daily 2001 01 01 spec.kjb completes successfully	2.8 sec	2
spec.etl.etl spec.ETL etl/spec/daily/daily 2017 06 20 reload spec.kjb completes successfully	2.4 sec	2
spec.etl.etl spec.ETL etl/spec/daily/daily 2017 06 20 spec.kjb completes successfully	2.5 sec	2
spec.etl.etl spec.ETL etl/spec/daily/daily 2017 06 20 to 21 spec.kjb completes successfully	2.3 sec	2
spec.etl.etl spec.ETL etl/spec/dwh/load/load spec.kjb completes successfully	2.2 sec	2
spec.etl.etl spec.ETL etl/spec/dwh/stage/stage reload spec.kjb completes successfully	2.2 sec	2
spec.etl.etl spec.ETL etl/spec/dwh/stage/stage spec.kjb completes successfully	2.7 sec	2
spec.etl.etl spec.ETL etl/spec/dwh/validate params/validate params spec.kjb completes successfully	2.2 sec	2
spec.etl.etl spec.ETL etl/spec/util/day sequence/day sequence spec.kjb completes successfully	2.2 sec	2
spec.load.load spec.load phase with fixture: spec/fixtures/load/2017-06-21.json exits with exit code 0	0 ms	2
spec.load.load spec.load phase with fixture: spec/fixtures/load/2017-06-21.json aggregates Gulliver records	0 ms	2
spec.load.load spec.load phase with fixture: spec/fixtures/load/2017-06-21.json transforms user '50 pence' to user tag ' 50_pence'	0 ms	2
spec.load.load spec.load phase with fixture: spec/fixtures/load/2017-06-21.json inserts Libslack into the service table	0 ms	2

All Tests

Package	Duration	Fail	(diff) Skip	(diff) Pass	(diff) Total	(diff)
spec.commands.db	1 min 18 sec	8	0	3	11	
spec.demo	39 sec	0	0	3	3	
spec.environment	39 sec	0	0	2	2	
spec.etl	39 sec	10	0	7	17	
spec.jdbc_helper	39 sec	0	0	1	1	
spec.load	39 sec	4	0	0	4	

[Back to Dashboard](#)

- [Status](#)
- [Changes](#)
- [Workspace](#)
- [Build Now](#)
- [Delete Project](#)
- [Configure](#)
- [Git Polling Log](#)

Project etl-project

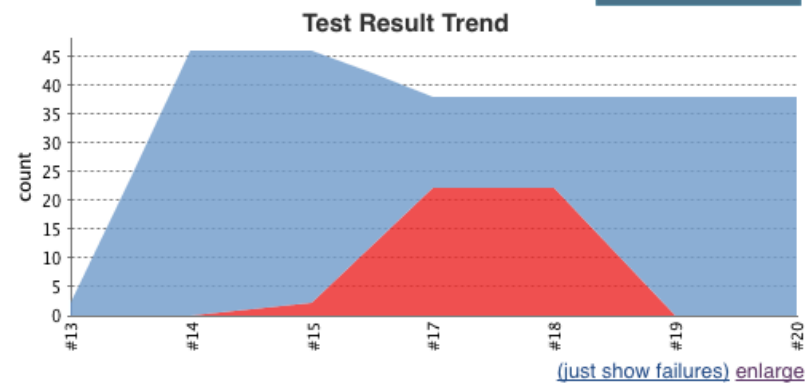
- [Workspace](#)
- [Last Successful Artifacts](#)
- [Recent Changes](#)
- [Latest Test Result \(no failures\)](#)

Build History [trend](#)

Build	Time
#20	Oct 30, 2017 1:35 PM
#19	Oct 30, 2017 1:34 PM
#18	Oct 30, 2017 1:32 PM
#17	Oct 30, 2017 1:30 PM
#16	Oct 30, 2017 1:27 PM
#15	Jun 23, 2017 8:24 PM
#14	Jun 23, 2017 8:14 PM
#13	Jun 23, 2017 8:13 PM

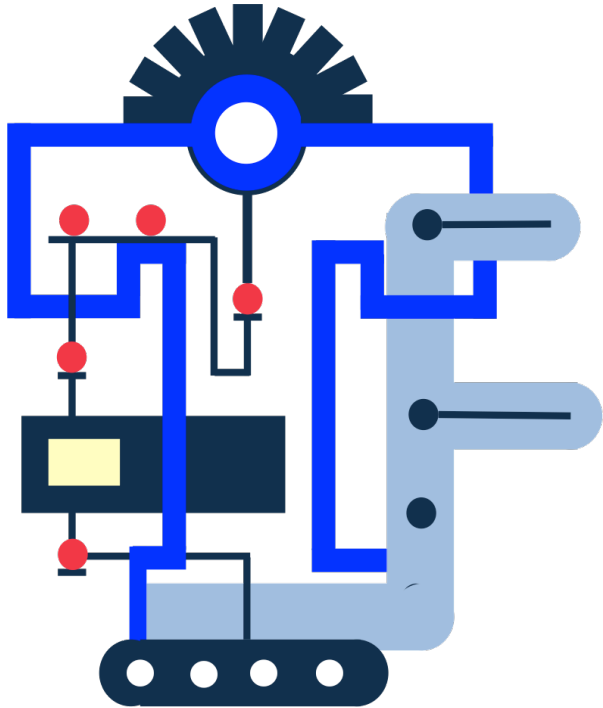
[add description](#)

[Disable Project](#)



Permalinks

- [Last build \(#20\), 2 min 23 sec ago](#)
- [Last stable build \(#20\), 2 min 23 sec ago](#)
- [Last successful build \(#20\), 2 min 23 sec ago](#)
- [Last failed build \(#18\), 5 min 38 sec ago](#)
- [Last unsuccessful build \(#18\), 5 min 38 sec ago](#)
- [Last completed build \(#20\), 2 min 23 sec ago](#)



Testable solutions

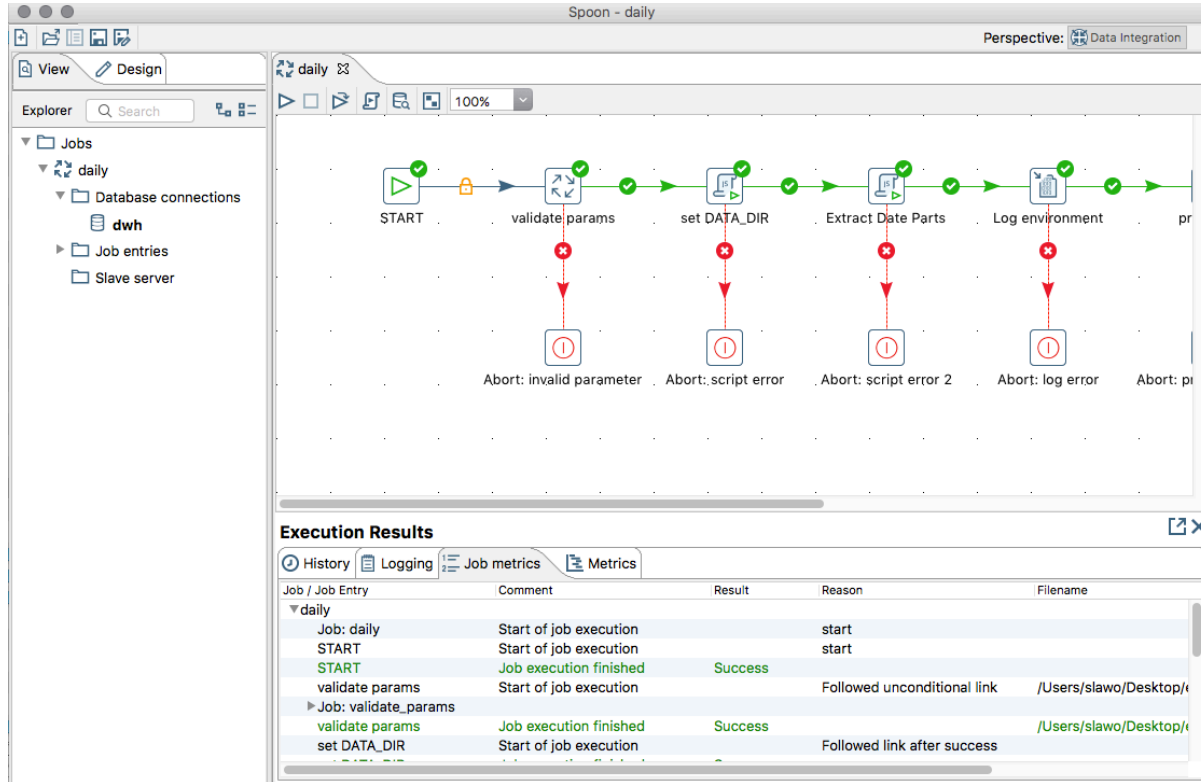
- configure all data sources/targets and paths through kettle variables or parameters
- local environment - **not in version control**
- test environment - **reference environment**
- production environment - **optional**

```
environments
  `-- local                # dev environment - not in version control
    |-- environment.sh     # shell environment variables
    |-- my.cnf             # database config file
    `-- .kettle            # KETTLE_HOME
      |-- shared.xml       # database connections
      `-- kettle.properties # kettle variables
  `-- test                 # test environment - in version control
  `-- production         # other environments
```

- share **nothing**
- reproducible results:
 - you can run it
 - your team can run it
 - ci-server can run it

```
$ bin/robot spoon
```


Configuration management



The screenshot shows the Spoon - daily interface. The main workspace displays a job execution flow with the following steps: START, validate_params, set DATA_DIR, Extract Date Parts, Log environment, and pr. Each step has a green checkmark above it, indicating successful execution. Below the flow, there are four red 'X' marks and arrows pointing to abort nodes: 'Abort: invalid parameter', 'Abort: script error', 'Abort: script error 2', and 'Abort: log error'. The 'Execution Results' table at the bottom provides a detailed log of the job execution.

Job / Job Entry	Comment	Result	Reason	Filename
daily				
Job: daily	Start of job execution		start	
START	Start of job execution		start	
START	Job execution finished	Success		
validate_params	Start of job execution		Followed unconditional link	/Users/slavo/Desktop/...
Job: validate_params				
validate_params	Job execution finished	Success		/Users/slavo/Desktop/...
set DATA_DIR	Start of job execution		Followed link after success	

Initialize db

```
$ bin/robot db reset
```

```
clearing database [ OK ]
```

```
initializing database
```

```
2017/10/20 15:32:37 - Kitchen - Start of run.
```

```
2017/10/20 15:32:37 - reset_dwh - Start of job execution
```

```
...
```

```
...
```

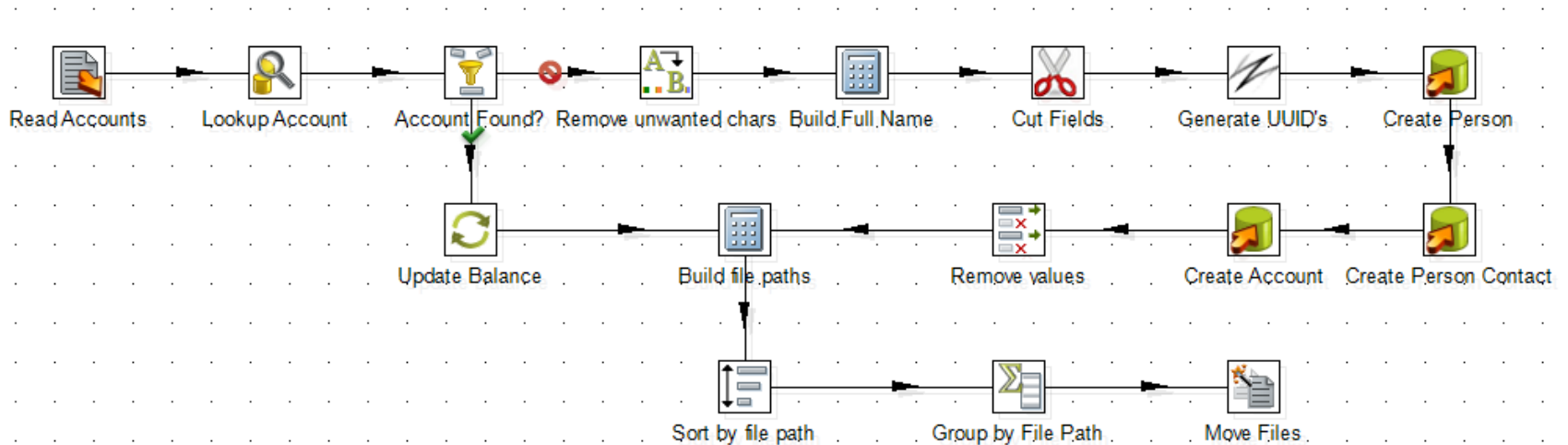
```
2017/10/20 15:32:41 - Kitchen - Finished!
```

```
2017/10/20 15:32:41 - Kitchen - Processing ended after 4 seconds.
```

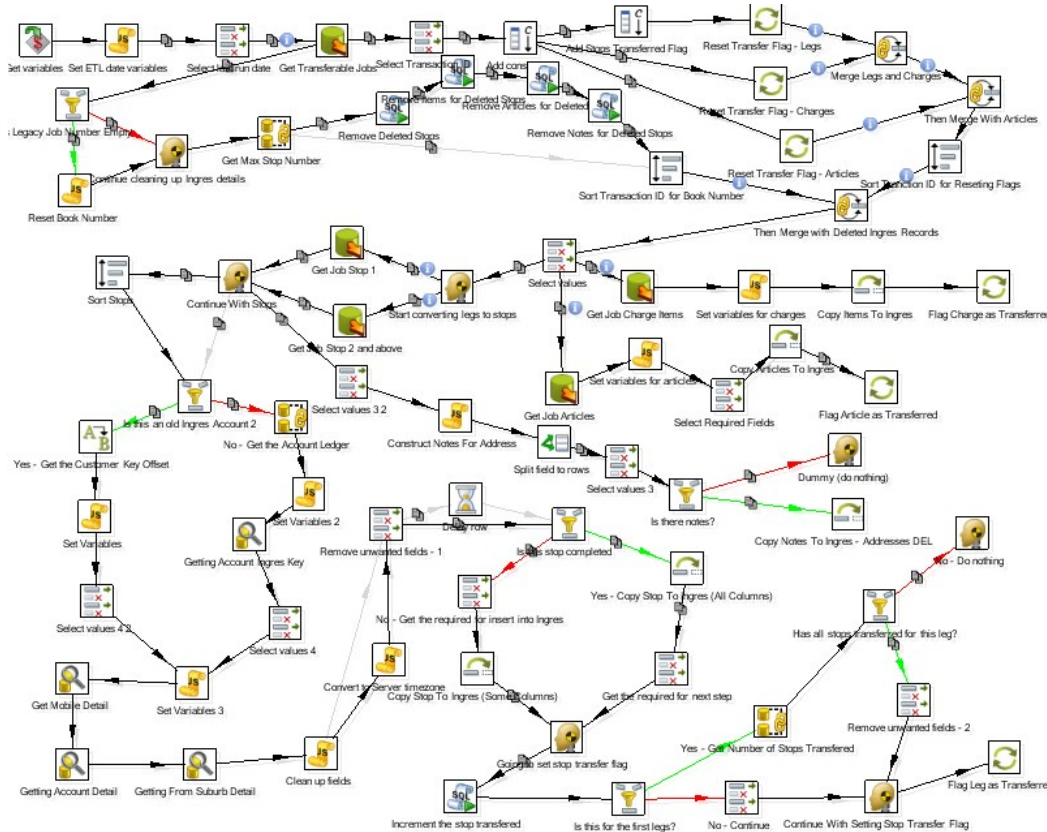
```
[ OK ]
```

- Define sub-systems/phases
 - define pre-requisites
 - data expected in certain sources
 - define outcomes
 - data written to certain sinks
- A sub-system/phase of the ETL process is responsible for a small set of related side-effects to happen

Sub-systems/Phases



Sub-systems/Phases



- Define entry points with a full functional contract.
- An entry point implements an application feature.

```
λ:solution $ bin/robot load help
```

LOAD COMMANDS

Convenience commands to execute the main entry points of the ROBOT loading process.

load

Runs the main ROBOT job, loading data for a given date.

usage: **robot load date**

date is date to load in YYYY-MM-DD format or **yesterday**

example: **robot load 2015-09-22**

Above command runs the daily loading job for 2015-09-22.

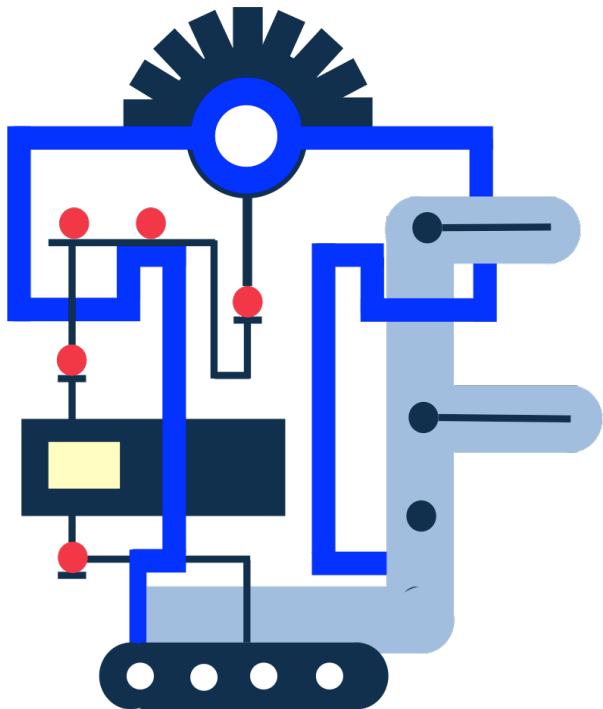
example: **robot load yesterday**

Above command runs the daily loading job for the previous day.

If invoked on 2015-09-23, it is equivalent to **robot load 2015-09-22**.

If invoked on 2015-09-24, it is equivalent to **robot load 2015-09-23**.

This form is convenient for invoking the ROBOT load from a scheduling utility like cron.



Testing ETL
solutions

- Computation tests
- Integration tests
- Functional tests
- Non-functional tests

- Single unit of ETL under test
- Performs a computation (no side-effects)
- What is a “unit” in PDI?
 - Job?
 - Transformation?
 - Sub-transformation (mapping)?

A simple computation test



Test job

`spec/dwh/validate_params/validate_params_spec.kjb`

A simple computation test



Test job

`spec/dwh/validate_params/validate_params_spec.kjb`

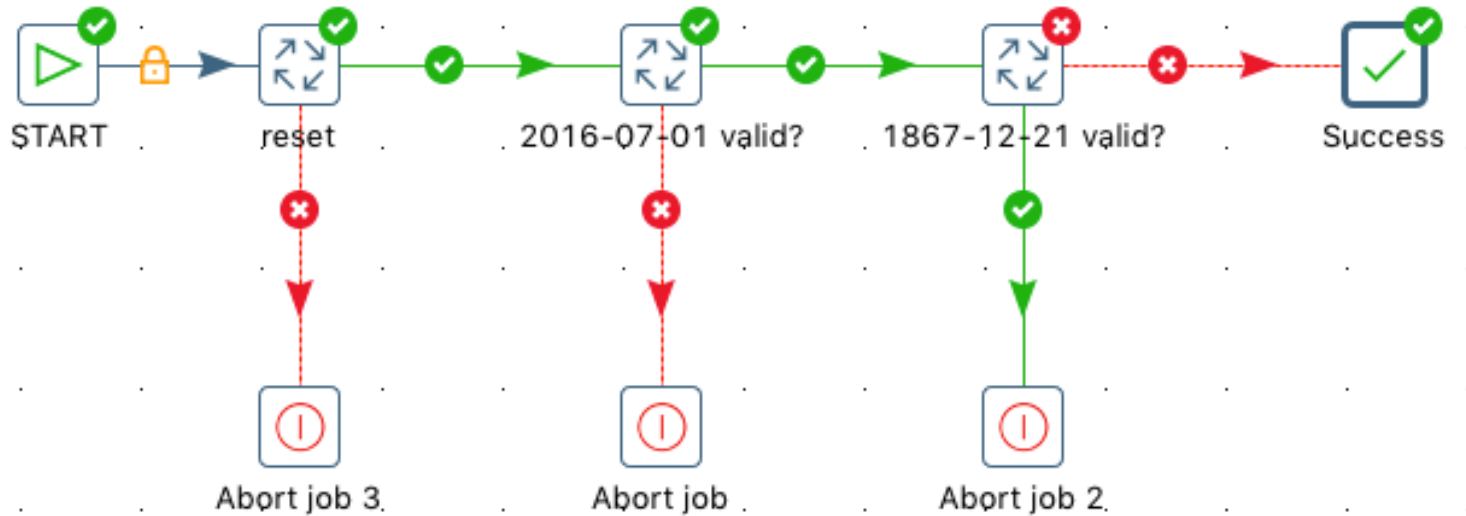
The job calls

`etl/dwh/validate_params.kjb`

- with `DATA_DATE=2016-07-01` and expects it to succeed
- with `DATA_DATE=1867-12-21` and expects it to fail

The job succeeds if all expectations are met. It fails otherwise.

Test jobs

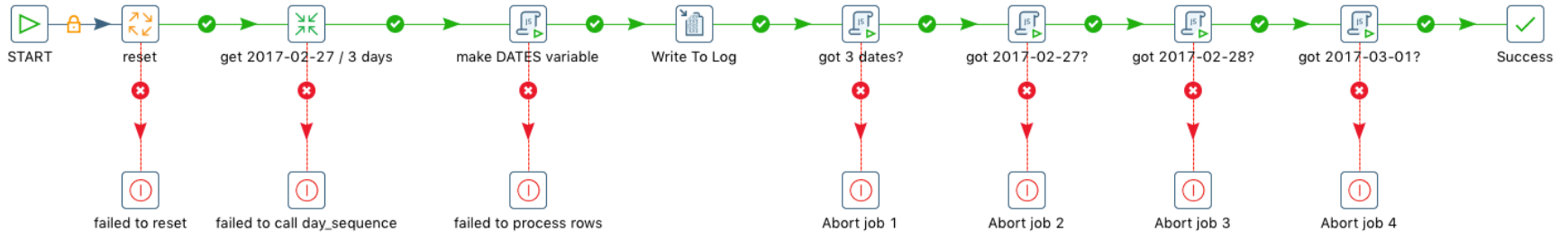


Test transformation results

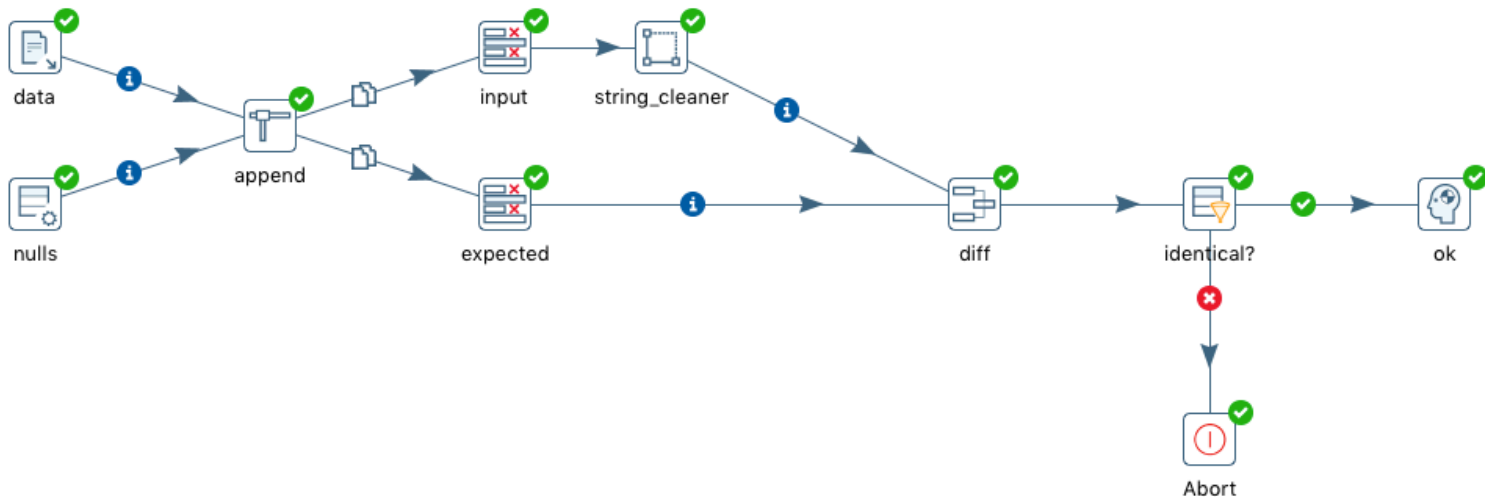
Rows of step: read dates (5 rows)

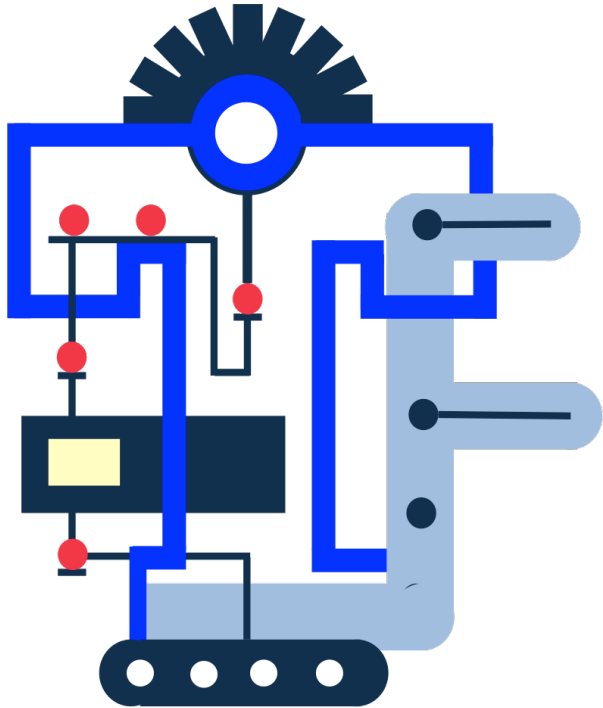
# ^	date_id	year	month	day
1	2000-01-01	2000	1	1
2	2000-01-02	2000	1	2
3	2000-01-03	2000	1	3
4	2000-01-04	2000	1	4
5	2000-01-05	2000	1	5

Test transformation results



Test sub-transformations

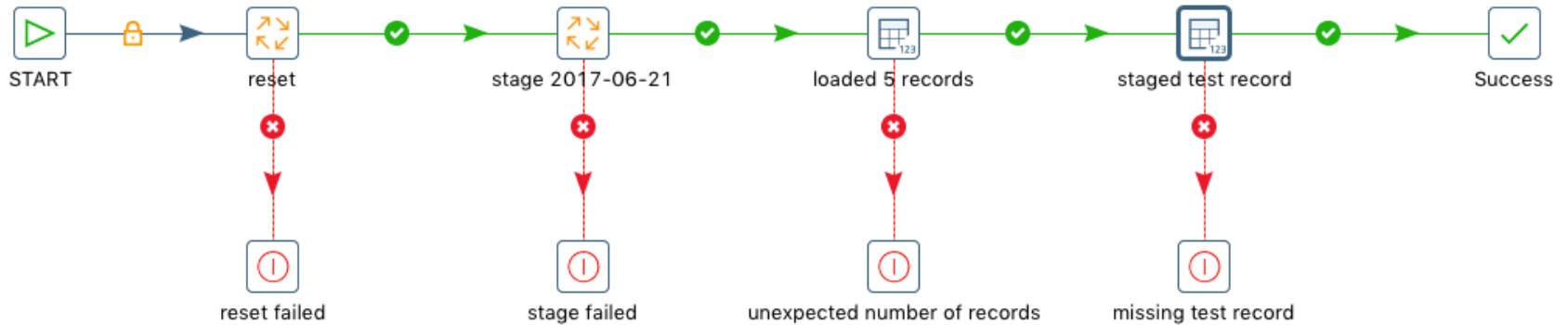




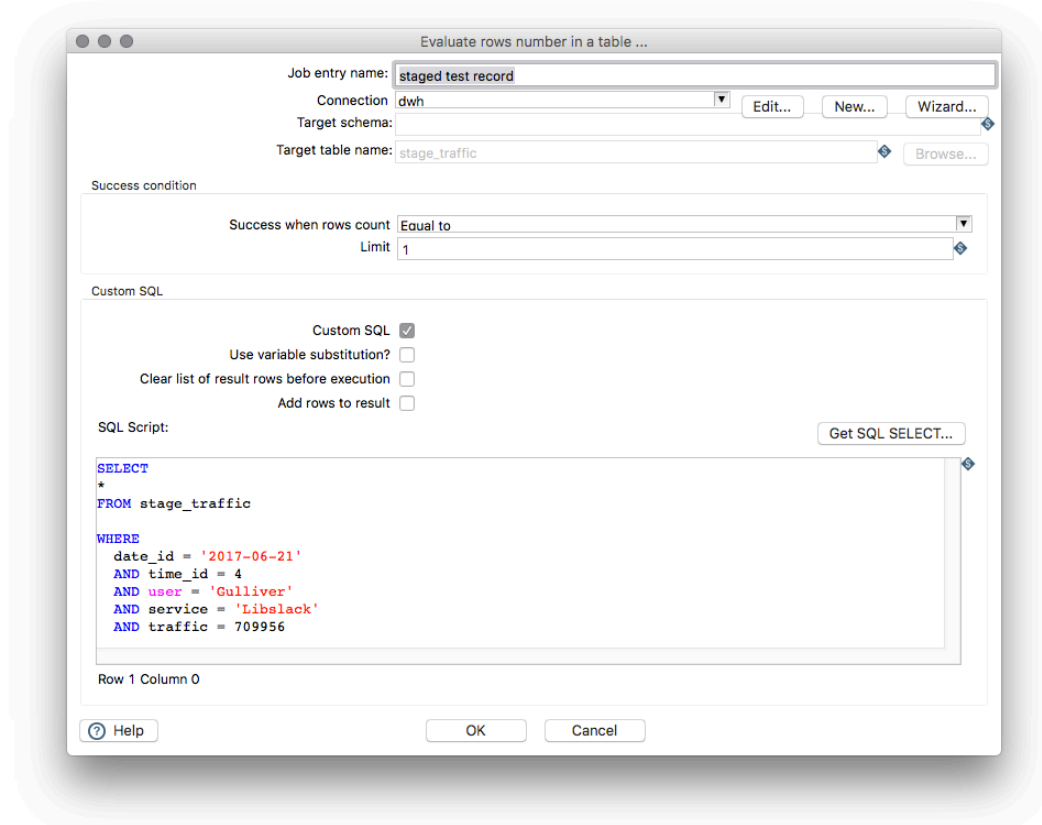
Integration tests

- ETL responsible for a set of related side-effects under test
- Most common case in ETL testing
 - Test individual phases of a batch process

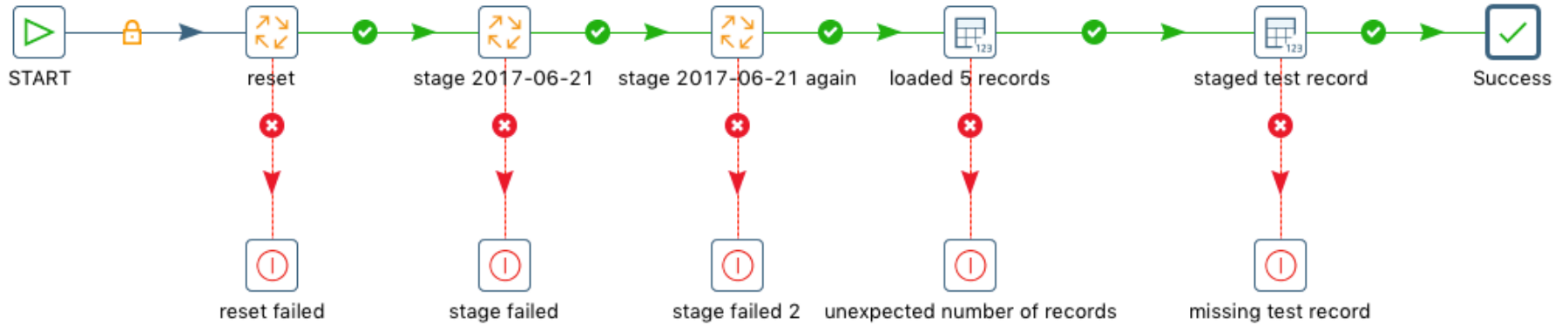
Integration tests

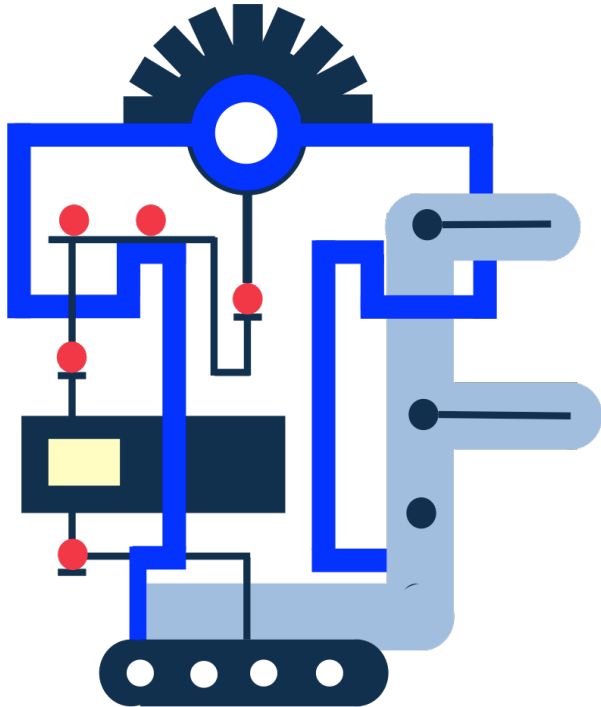


Integration tests



Integration tests

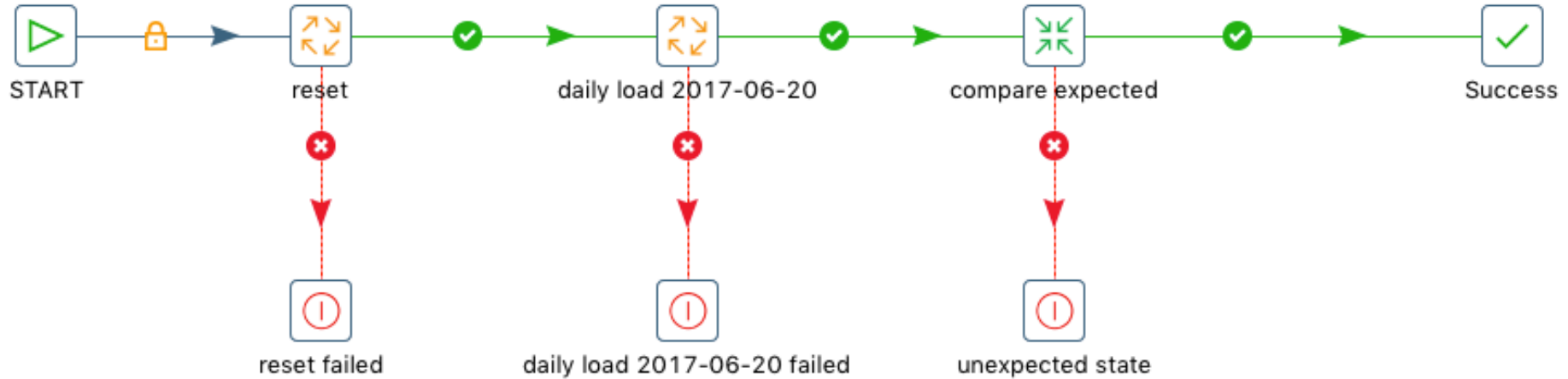




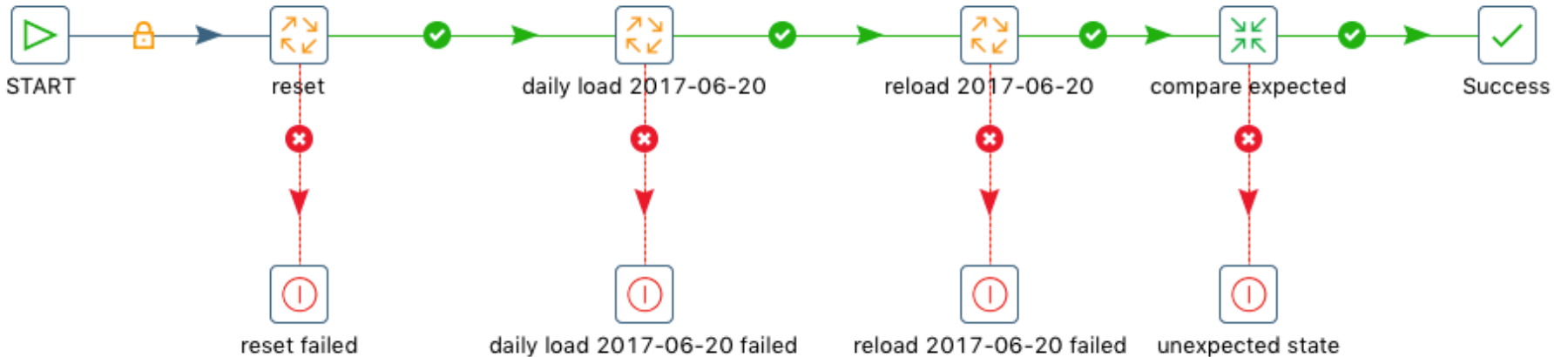
Functional testing

- Entry point of ETL solution under test
- Assertions reflect invocation contract
 - Behavior on happy path
 - Behavior on errors
 - Behavior on incorrect invocation

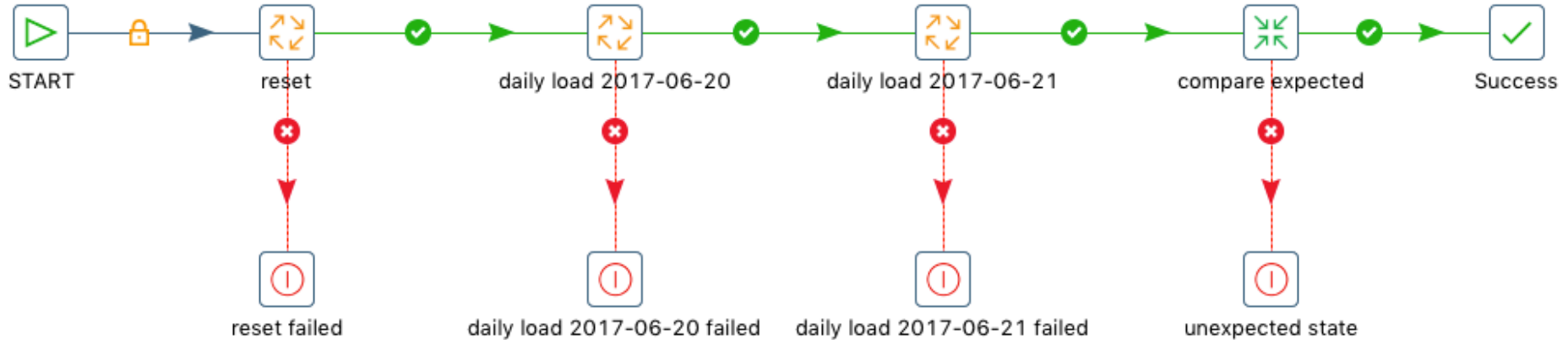
Test the daily run

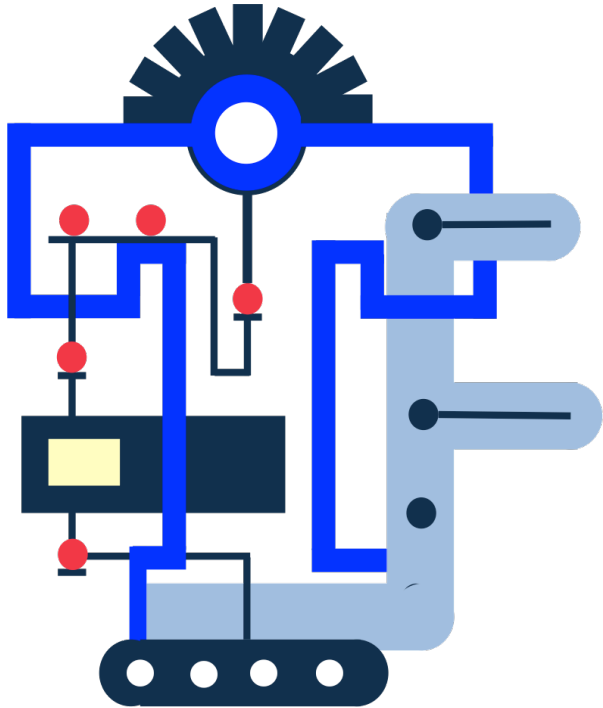


Test the daily run



Test the daily run



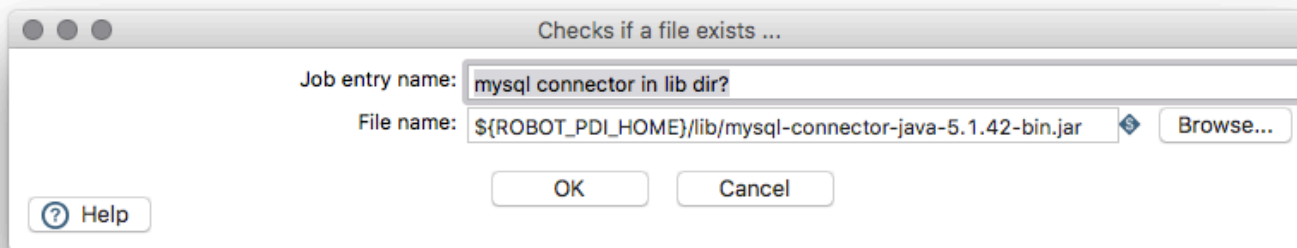
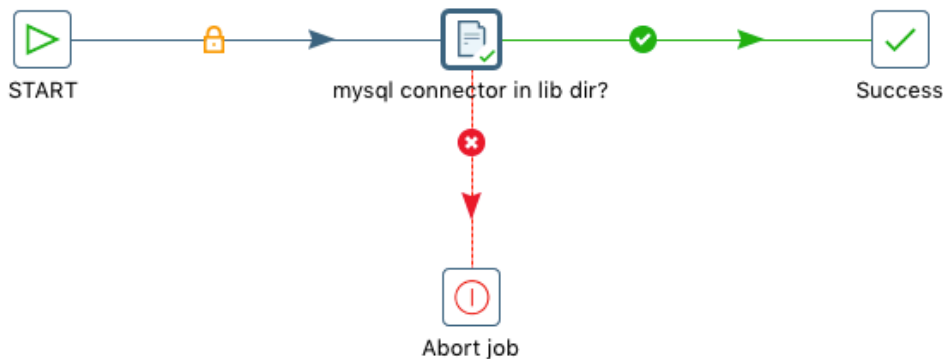


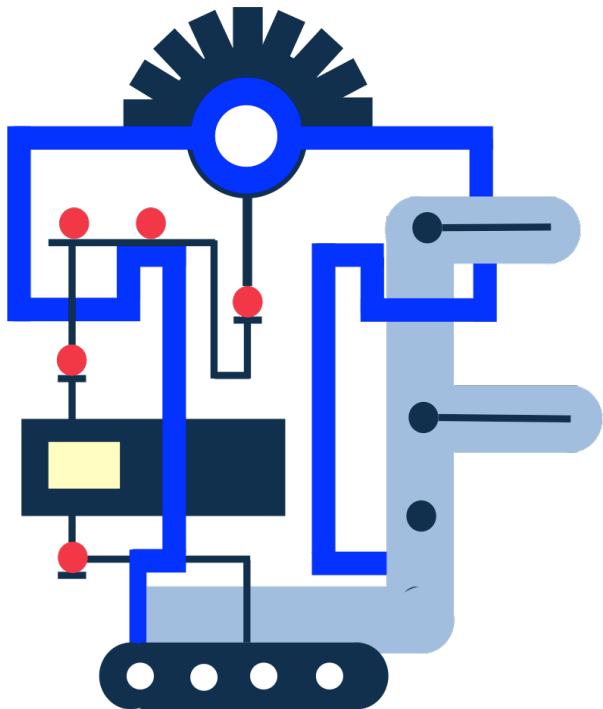
Non-functional tests

- Performance
 - how long does workload x take?
- Stability
 - what does it take to break it?
 - How much memory is too little?
 - What happens when loading unexpected data?
(truncated file, column too long, 50MB XML in string field, badly formatted CSV reads as single field, empty files)

- Security
 - Verify configuration assumptions automatically
- Compliance
 - We must use version x of library y

Test compliance





Scripting tests

JRuby is the ruby language on the JVM

<http://jruby.org>

Maintained by Redhat.

Runs Rails on JBoss 😊

Rspec is a testing framework for ruby

<http://rspec.info/>

<https://relishapp.com/rspec>

\$ bin/robot test

- Includes helper files in **spec/support**
- traverses the **spec** folder looking for files whose names end in **_spec.rb** and loads them as **tests**

```
describe "dwh clear job" do
```

```
end
```

```
describe "dwh clear job" do
  describe "when db is not empty" do

    end
  end
end
```

```
describe "dwh clear job" do
  describe "when db is not empty" do
    before :all do
      dwh_db.load_fixture "spec/fixtures/steelwheels/steelwheels.sql"
    end
  end
end
```

`spec/support/spec_helpers.rb`

```
def dwh_db  
  ...  
end
```

Returns a JDBC database object.

Connects on demand, and closes automatically when test-suite ends.

spec/support/spec_helpers.rb

```
def dwh_db  
  ...  
end
```

In addition

`dwh_db.load_fixture(path)` allows loading a sql or json fixture file

`dwh_db.reset()` triggers **\$ bin/robot db reset**


```
describe "dwh clear job" do
  describe "when db is not empty" do
    before :all do
      dwh_db.load_fixture "spec/fixtures/steelwheels/steelwheels.sql"
    end
  end
end
```

```
describe "dwh clear job" do
  describe "when db is not empty" do
    before :all do
      dwh_db.load_fixture "spec/fixtures/steelwheels/steelwheels.sql"
      @result = run_job "etl/dwh/util/clear.kjb"
    end
  end
end
```

spec/support/spec_helpers.rb

```
def run_job file, params
  ...
end
```

Runs a kettle job and returns a map

```
{
  :successful? => true/false,
  :log         => "log text",
  :result      => [row1, row2, row3, ...]
}
```

```
describe "dwh clear job" do
  describe "when db is not empty" do
    before :all do
      dwh_db.load_fixture "spec/fixtures/steelwheels/steelwheels.sql"
      @result = run_job "etl/dwh/util/clear.kjb"
    end
  end
end
```

```
describe "dwh clear job" do
  describe "when db is not empty" do
    before :all do
      dwh_db.load_fixture "spec/fixtures/steelwheels/steelwheels.sql"
      @result = run_job "etl/dwh/util/clear.kjb"
    end

    it "completes successfully" do
      expect(@result[:successful?]).to be true
    end
  end
end
```

```
describe "dwh clear job" do
  describe "when db is not empty" do
    before :all do
      dwh_db.load_fixture "spec/fixtures/steelwheels/steelwheels.sql"
      @result = run_job "etl/dwh/util/clear.kjb"
    end

    it "completes successfully" do
      expect(@result[:successful?]).to be true
    end

    it "clears the db" do
      expect(dwh_db.query("SHOW TABLES").to_a.length).to eq 0
    end
  end
end
```

Test orchestration

```
$ bin/robot test
```

λ:solution \$ bin/robot test

Run options: exclude {:long_running=>true, :remote=>true, :integration=>true}

db clear command

when db is not empty

exits with exit code 0

clears the db

db reset command

when db is not empty

exits with exit code 0

clears the db

creates a dim_date dimension

including 2000-01-01

including 2015-12-31

creates a dim_time dimension

including 00 AM

including 11 PM

including -1/NA

creates an iso_countries dimension

including US

including DE

ETL

etl/spec/daily/daily_0000_00_00_spec.kjb

completes successfully

etl/spec/daily/daily_2001_01_01_spec.kjb

completes successfully

etl/spec/daily/daily_2017_06_20_reload_spec.kjb

completes successfully

etl/spec/daily/daily_2017_06_20_spec.kjb


```
etl/spec/daily/daily_2001_01_01_spec.kjb  
  completes successfully  
etl/spec/daily/daily_2017_06_20_reload_spec.kjb  
  completes successfully  
etl/spec/daily/daily_2017_06_20_spec.kjb  
  completes successfully  
etl/spec/daily/daily_2017_06_20_to_21_spec.kjb  
  completes successfully  
etl/spec/dummy/dummy_spec.kjb - verifies the test suite runs ETL as tests  
  completes successfully  
etl/spec/dwh/load/load_spec.kjb  
  completes successfully  
etl/spec/dwh/pre_flight/pre_flight_empty_file_spec.kjb  
  completes successfully  
etl/spec/dwh/pre_flight/pre_flight_missing_file_spec.kjb  
  completes successfully  
etl/spec/dwh/pre_flight/pre_flight_ok_spec.kjb  
  completes successfully  
etl/spec/dwh/stage/stage_spec.kjb  
  completes successfully  
etl/spec/dwh/validate_params/validate_params_spec.kjb  
  completes successfully  
etl/spec/environment/mysql_driver_spec.kjb  
  completes successfully  
etl/spec/environment/robot_dir_is_set_spec.kjb - ROBOT_DIR variable is set correctly  
  completes successfully  
etl/spec/util/day_sequence/day_sequence_spec.kjb  
  completes successfully  
etl/spec/util/string_cleaner/string_cleaner_spec.kjb - identifiers are cleaned according to project standards  
  completes successfully
```

```
etl/spec/dummy/dummy_spec.kjb - verifies the test suite runs ETL as tests
  completes successfully
etl/spec/dwh/load/load_spec.kjb
  completes successfully
etl/spec/dwh/pre_flight/pre_flight_empty_file_spec.kjb
  completes successfully
etl/spec/dwh/pre_flight/pre_flight_missing_file_spec.kjb
  completes successfully
etl/spec/dwh/pre_flight/pre_flight_ok_spec.kjb
  completes successfully
etl/spec/dwh/stage/stage_spec.kjb
  completes successfully
etl/spec/dwh/validate_params/validate_params_spec.kjb
  completes successfully
etl/spec/environment/mysql_driver_spec.kjb
  completes successfully
etl/spec/environment/robot_dir_is_set_spec.kjb - ROBOT_DIR variable is set correctly
  completes successfully
etl/spec/util/day_sequence/day_sequence_spec.kjb
  completes successfully
etl/spec/util/string_cleaner/string_cleaner_spec.kjb - identifiers are cleaned according to project standards
  completes successfully
```

```
jdbc_helper
  dwh_db
    when selecting from steelwheels.customers
      has 126 customers
```

Finished in 1 minute 8.62 seconds (files took 1.25 seconds to load)

28 examples, 0 failures

`spec/etl/etl_spec.rb`

Recursively traverses **etl/spec** looking for files whose names end in **_spec.kjb**, and dynamically generates a **describe** and **it** block for it.

Hence all such job files are part of the test suite.

```
describe "ETL" do

  Dir.glob("./**/*_spec.kjb").each do |path|

    describe "#{path}" do
      it "completes successfully" do
        @result = run_job path.to_s, {}
        expect(@result[:successful?]).to be true
      end
    end

  end

end

end
```

Rspec runs in two phases

Phase 1: collects tests, recording the structure as given by the describe blocks.

Phase 2: filters found tests as per command line parameters and executes them

Run only tests containing the word 'clear' in their name or enclosing describe blocks:

```
$ bin/robot test --example 'clear'
```

Run only tests tagged 'long_running':

```
$ bin/robot test --tag 'long_running'
```

Run only tests in **spec/commands**

```
$ bin/robot test spec/commands
```

Test orchestration

```
$ bin/robot test
```

λ:solution \$ bin/robot test

Run options: exclude {:long_running=>true, :remote=>true, :integration=>true}

db clear command

when db is not empty

exits with exit code 0

clears the db

db reset command

when db is not empty

exits with exit code 0

clears the db

creates a dim_date dimension

including 2000-01-01

including 2015-12-31

creates a dim_time dimension

including 00 AM

including 11 PM

including -1/NA

creates an iso_countries dimension

including US

including DE

ETL

etl/spec/daily/daily_0000_00_00_spec.kjb

completes successfully

etl/spec/daily/daily_2001_01_01_spec.kjb

completes successfully

etl/spec/daily/daily_2017_06_20_reload_spec.kjb

completes successfully

etl/spec/daily/daily_2017_06_20_spec.kjb


```
etl/spec/daily/daily_2001_01_01_spec.kjb
  completes successfully
etl/spec/daily/daily_2017_06_20_reload_spec.kjb
  completes successfully
etl/spec/daily/daily_2017_06_20_spec.kjb
  completes successfully
etl/spec/daily/daily_2017_06_20_to_21_spec.kjb
  completes successfully
etl/spec/dummy/dummy_spec.kjb - verifies the test suite runs ETL as tests
  completes successfully
etl/spec/dwh/load/load_spec.kjb
  completes successfully
etl/spec/dwh/pre_flight/pre_flight_empty_file_spec.kjb
  completes successfully
etl/spec/dwh/pre_flight/pre_flight_missing_file_spec.kjb
  completes successfully
etl/spec/dwh/pre_flight/pre_flight_ok_spec.kjb
  completes successfully
etl/spec/dwh/stage/stage_spec.kjb
  completes successfully
etl/spec/dwh/validate_params/validate_params_spec.kjb
  completes successfully
etl/spec/environment/mysql_driver_spec.kjb
  completes successfully
etl/spec/environment/robot_dir_is_set_spec.kjb - ROBOT_DIR variable is set correctly
  completes successfully
etl/spec/util/day_sequence/day_sequence_spec.kjb
  completes successfully
etl/spec/util/string_cleaner/string_cleaner_spec.kjb - identifiers are cleaned according to project standards
  completes successfully
```

```
completes successfully
etl/spec/dummy/dummy_spec.kjb - verifies the test suite runs ETL as tests
  completes successfully
etl/spec/dwh/load/load_spec.kjb
  completes successfully
etl/spec/dwh/pre_flight/pre_flight_empty_file_spec.kjb
  completes successfully
etl/spec/dwh/pre_flight/pre_flight_missing_file_spec.kjb
  completes successfully
etl/spec/dwh/pre_flight/pre_flight_ok_spec.kjb
  completes successfully
etl/spec/dwh/stage/stage_spec.kjb
  completes successfully
etl/spec/dwh/validate_params/validate_params_spec.kjb
  completes successfully
etl/spec/environment/mysql_driver_spec.kjb
  completes successfully
etl/spec/environment/robot_dir_is_set_spec.kjb - ROBOT_DIR variable is set correctly
  completes successfully
etl/spec/util/day_sequence/day_sequence_spec.kjb
  completes successfully
etl/spec/util/string_cleaner/string_cleaner_spec.kjb - identifiers are cleaned according to project standards
  completes successfully
```

```
jdbc_helper
  dwh_db
    when selecting from steelwheels.customers
      has 126 customers
```

Finished in 1 minute 8.62 seconds (files took 1.25 seconds to load)

28 examples, 0 failures

[Back to Dashboard](#)

[Status](#)

[Changes](#)

[Workspace](#)

[Build Now](#)

[Delete Project](#)

[Configure](#)

[Git Polling Log](#)

Build History

[trend](#)

● #20	Oct 30, 2017 1:35 PM
● #19	Oct 30, 2017 1:34 PM
● #18	Oct 30, 2017 1:32 PM
● #17	Oct 30, 2017 1:30 PM
● #16	Oct 30, 2017 1:27 PM
● #15	Jun 23, 2017 8:24 PM
● #14	Jun 23, 2017 8:14 PM
● #13	Jun 23, 2017 8:13 PM

Project etl-project



[Workspace](#)



[Last Successful Artifacts](#)



[Recent Changes](#)



[Latest Test Result](#) (no failures)

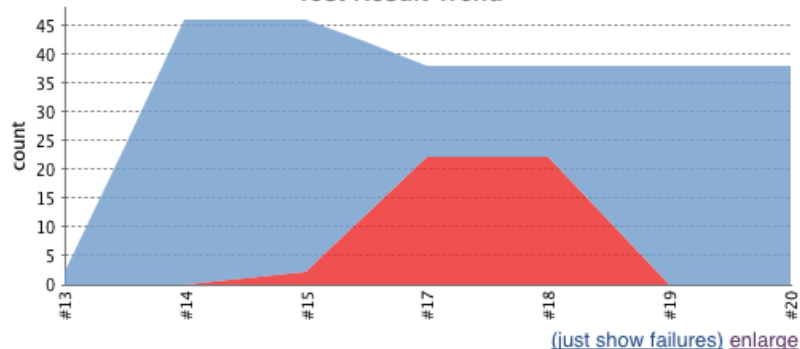
Permalinks

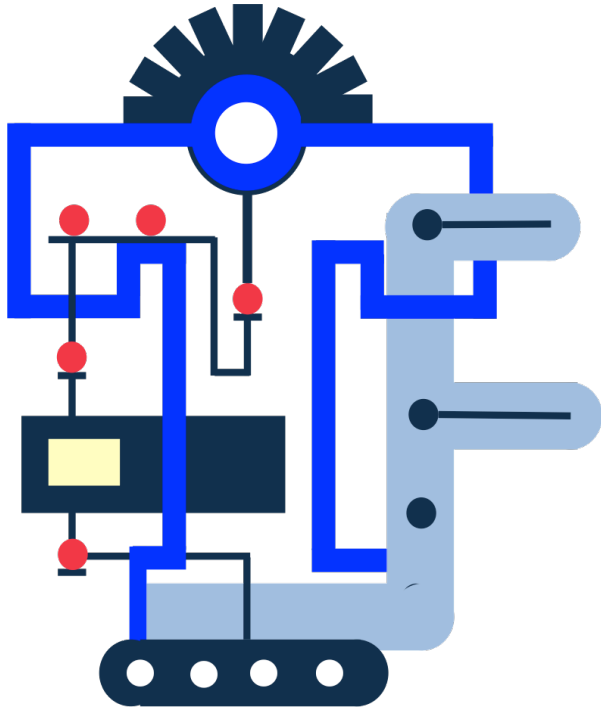
- [Last build \(#20\), 2 min 23 sec ago](#)
- [Last stable build \(#20\), 2 min 23 sec ago](#)
- [Last successful build \(#20\), 2 min 23 sec ago](#)
- [Last failed build \(#18\), 5 min 38 sec ago](#)
- [Last unsuccessful build \(#18\), 5 min 38 sec ago](#)
- [Last completed build \(#20\), 2 min 23 sec ago](#)

[add description](#)

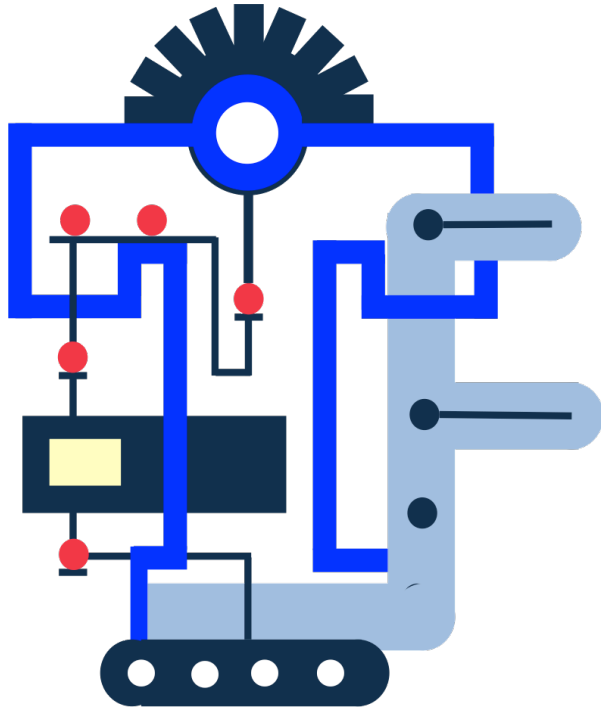
[Disable Project](#)

Test Result Trend

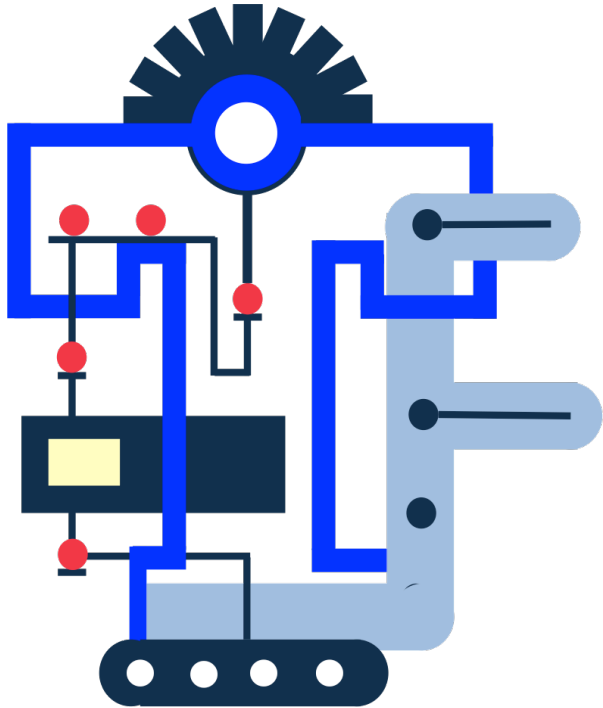




Thank you!



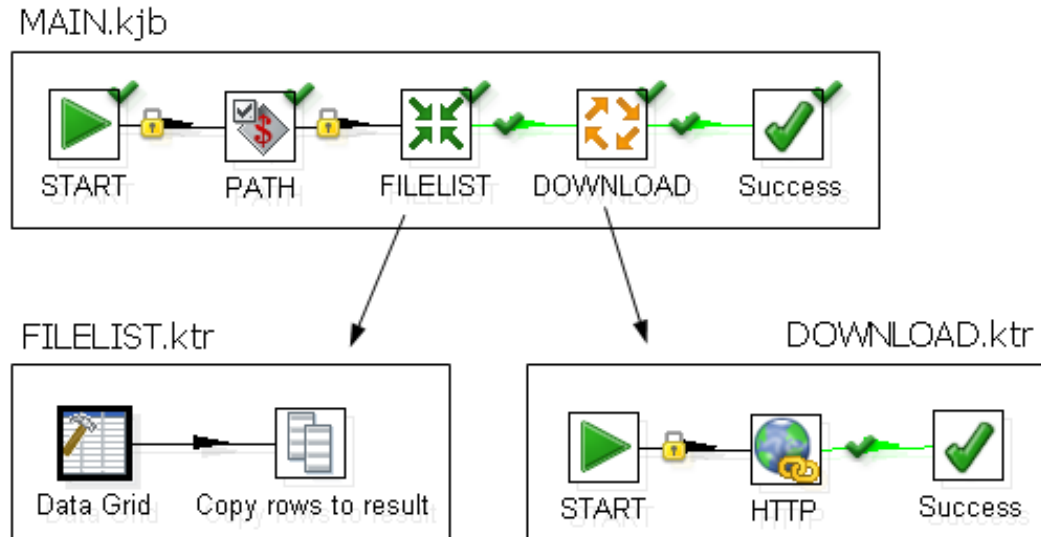
Backup Slides



Testing in practice

Test what you run

- Verify behavior of the entity you run directly



- Helpers
 - utility code/etl of components reused to make tests about the what, not about the how
 - fixture loaders
 - assertion helpers
 - data comparison helpers

- Data Fixtures
 - sets of test data, encoded in a convenient way, easily loaded into data sources and sinks
- JSON, CSV, SQL, XML, YAML
 - Use whatever is easiest to maintain for the team
- **Generate data fixtures** through parameterized scripts if you need to generate datasets with consistent relationships

- File Fixtures
 - sets of test files acted upon during a run
- Maintain file fixtures separate from source location expected by ETL
- If fixture files are changed as part of the test, copy them to a temporary location before running tests
- Create a unique source location per test run, if the file location is shared (like sftp)