HYPERLANE
Data, Code, Lab, Factory

Munich | 23 June 2015 | Hadoop User Group

Assignment

- Cross-Media Audience Measurement System.
- Dozends of data sources. Varyiing quality.
- Distributed Data Science and Engineering Teams.
- Global Ops Hub.

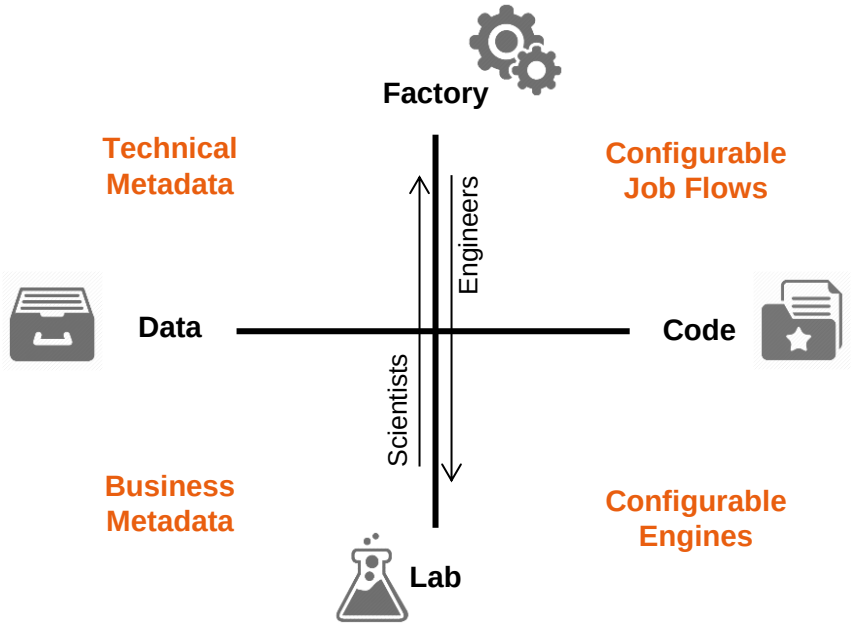
Hyperlane ...but why?

Sculley et al (2014):

Undeclared Consumers:
Consuming the output of a given prediction model as an input to another component of the system. Changes will very likely impact these other parts.

Unstable Data Dependencies:
Some input signals are unstable, meaning that they qualitatively change behavior over time.

Static Analysis of Data:
On teams with many engineers, or if there are multiple interacting teams, not everyone knows the status of every single feature.



Sculley et al (2014):

Glue Code:
Using self-contained solutions often results in a glue code system design pattern, in which a massive amount of supporting code is written to get data into and out of general-purpose packages.

Pipeline Jungles:
The system for preparing data in an ML-friendly format may become a jungle of scrapes, joins, and sampling steps, often with intermediate files output.

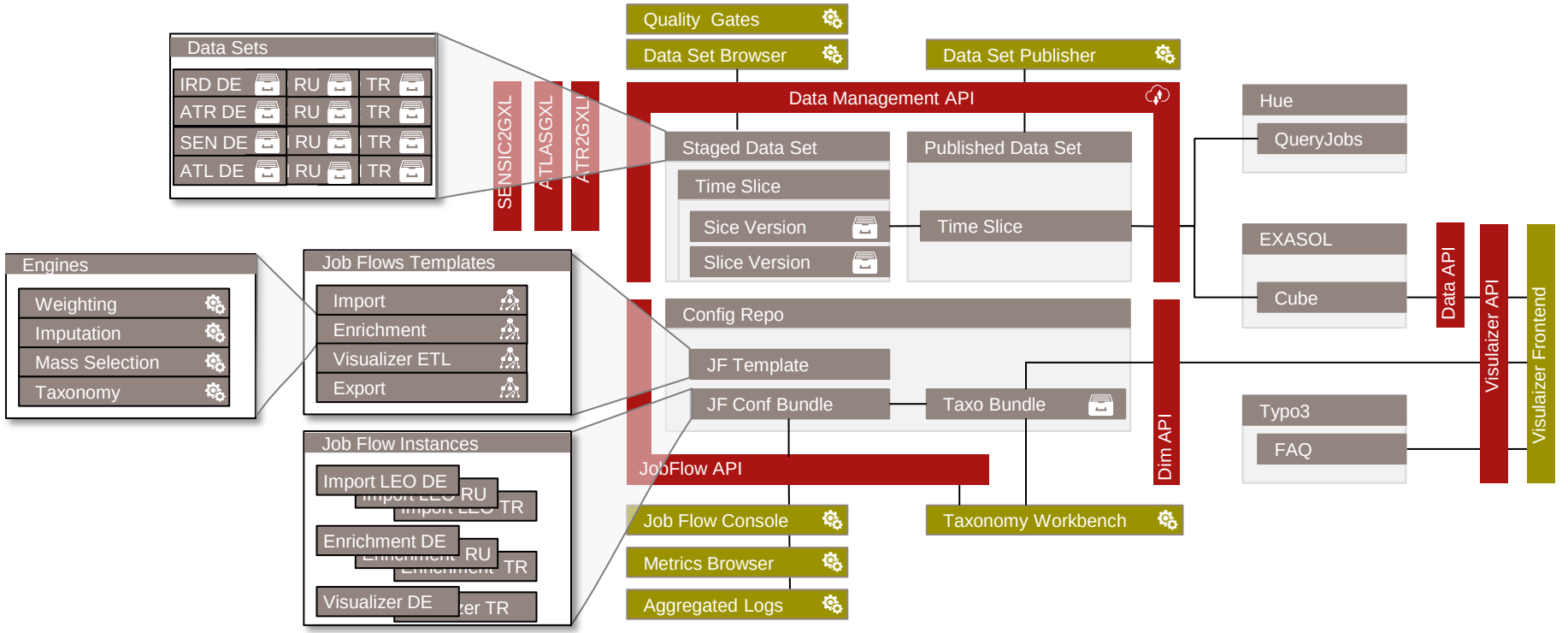
Configuration Debt:
In a mature system which is being actively developed, the number of lines of configuration can far exceed the number of lines of the code.

User Stories

- As a Data Provider..
- I want to provide either (time-based) increments or full snapshots of my data in order to avoid expensive transfomations.
- I want to provide data using the most recent schema even if this schema evolves over time in order to avoid expensive schema mappings.
- I want to provide new versions of data that has already been delivered in order to reflect reworks.
- I want to tag and comment my data (-increments, -versions) in order to retain qualitative and business meta-data that has an impact on how the data can/should (not) be used. This can include lineage data encoded in tags.

- As a Data Consumer..
- I want to find and browse (sampled) data sets based on their business meta data, tags, comments etc. in order to find relevant data and understand the status this data has.
- I want to access data using consistent schema and format whenever possible (no matter if the the data was written using evolving schemas or particular technical representations or not) in ordert to avoid writing expensive and error prone mapping code.
- I want to have (automated) access to growing data sets (Feeds) filtered by tags / business meta data in order to feed it into data pipelines.
- I want to be exposed to immutable data while I‘m working on it in order to avoid breaking processing jobs due to concurrent write/update operations (potentially using a different schema) during the run time of my job.
- I want to be able to use standard tooling (Hive, PIG, JDBC, etc.) to access data in order to avoid lock-in and to benefit from the momentum in the engineering and data science community.
- I want to understand at what data center a particular data set is physically stored in order to adhere to regulation and to optimize the processing (e.g. avoid remote data access).

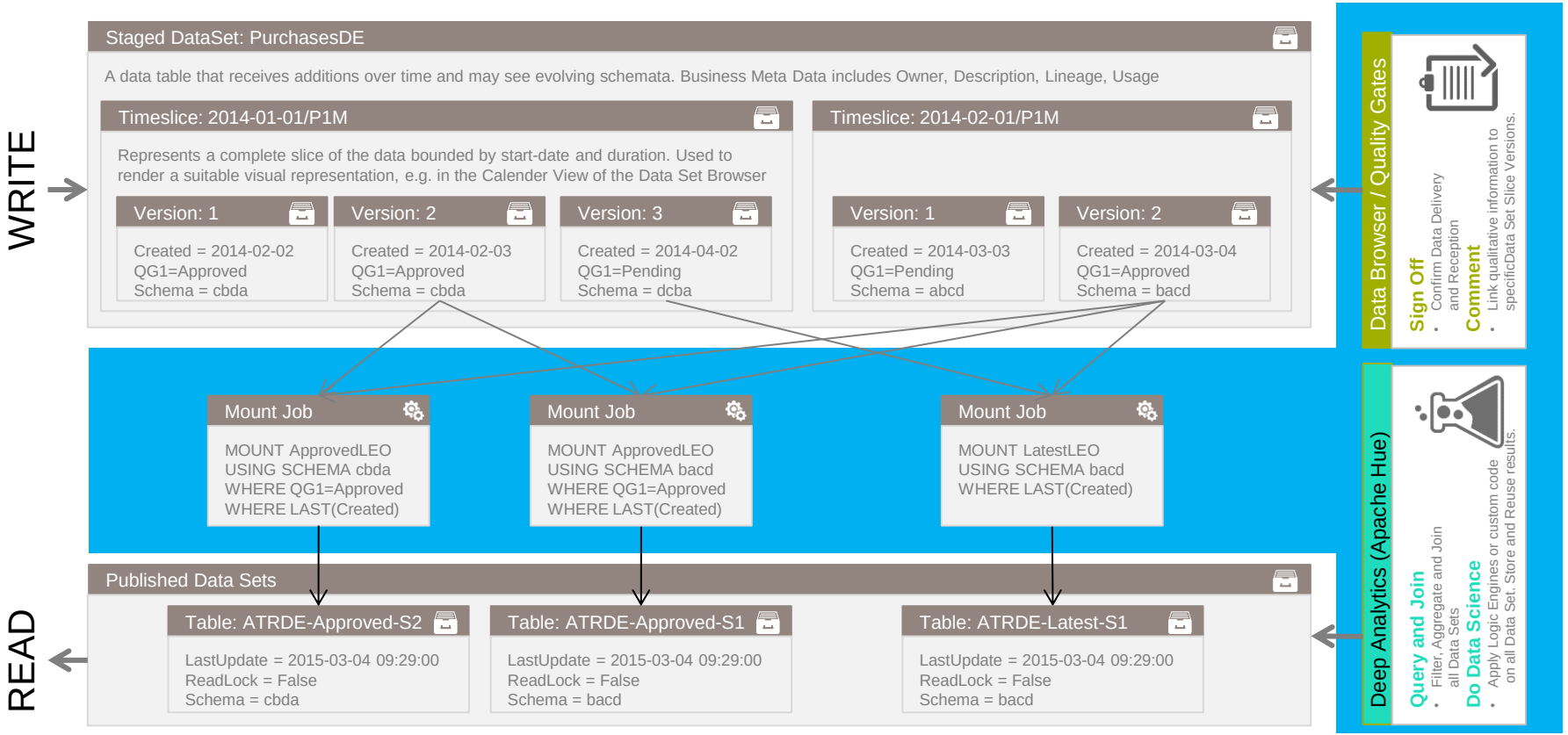
Hyperlane – Functional Architecture



HYPERLANE DATA

Berlin | 16 March 2015 | Get Together

Hyerlane - Data Set Management



Meta Data Management

Challenges:

- Versioned Data, Quality Gates, Comments as Meta Data, Logging
- Wording: Dataset, Instance, Partition, Timeslice? Dataset & Timeslice

Make or Buy?

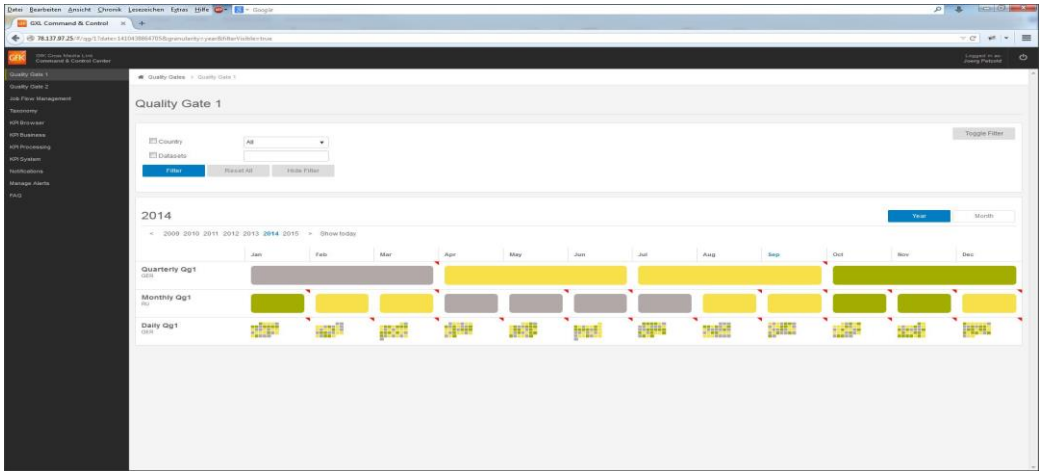
- Cloudera Navigator
  + Policy Engine, - Vendor-Lock-In, Inflexible
- Waterline Data
  + very good for unstructured data, + profiling, + tag propagation, +lineage, - no qg, no key value tags, - product fairly new

Metadata:

1. Semantics: What does it actually mean?
2. Quality: Checked and Approved
3. Validity: What can/can‘t we do with this data?

Tool: Data Set Browser + QGs

DEMO TIME!



HYPERLANE CODE

Berlin | 16 March 2015 | Get Together

Configurable Engines

- Business Logic is packaged as Pig Scripts
- Takes Parameters and invokes R or JAVA code
- Easy to re-use through Apache Hue and Apache Oozie
- No UI

readme.md

Weighting Engine

Given the mass (a subset of panelists) and a description (constraints) of the real world online population, this engine computes weights, such that weighted l
(for the mass) can be considered representative for the actual online population.

This engine applies two phase:

1. Compute input data for the weighting R script
2. Run the R script on the data produces in step 1.

https://stash.gfk.com/projects/GO/repos/marketingscience/browse/weighting/specification/

Input Data

Name	Type	Description
Mass selected and Poss/Socio imputed data (output of Device Possession Imp.)	Semicolon separated file	Output of Device Possession Imputation
setting-constraints	Comma separated file	Settings for the weighting
weighting-constraints	Comma separated file	Constraints for the weighting

Parameters

Name	Description	Example
inputLocation	Location of the input file (i.e. Device Possession Imputation engine output)	/user/hdfs/GO-262/output/sozio_mass_completed_201407_
outputLocation	Location of the output produced by the engine	Output of Device Possession Imputation
queueName	Name of the queue	root.default
appPath	Base path of the oozie workflow	/user/hyperlane/engines/weighting//

Output Data

Pig Wrapper

```
SET mapred.min.split.size 1000000000;
SET pig.maxCombinedSplitSize 1000000000;
SET pig.exec.reducers.max 1;
SET pig.splitCombination 'true';
SET mapreduce.map.java.opts '-Xmx8G';
SET mapred.child.java.opts '-Xmx8G';

DEFINE R_SCRIPT `Rscript $r_script_name --min-vsize=100M --min-nsize=16M 1>&2`
        INPUT(stdin USING PigStreaming(';'))
        OUTPUT('output.ssv' USING PigStreaming(';'))
        SHIP('$r_script_name', '$r_resources_name', '$weighting_constraints', '$weighting_settings');

Data = LOAD '$input_location' USING PigStorage(';');

Data = STREAM Data THROUGH R_SCRIPT;

STORE Data INTO '$output_location' USING PigStorage(';');
```

Pig UDF

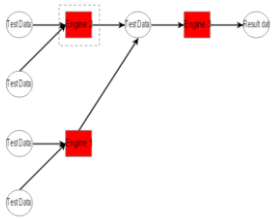
```
package com.gfk.hyperlane.engine.taxonomy;

import java.util.Arrays;
import java.util.List;

import org.apache.pig.backend.executionengine.ExecException;
import org.apache.pig.data.Tuple;

/**
 * @author thevis
 *
 */
public class AddTaxonomy extends EnhanceOrUpdate {


    static String[] COLUMNS = {
        "TAXONOMY_COMPANY_LABEL",
        "TAXONOMY_COMPANY_ID",
        "TAXONOMY_BRAND_LABEL",
        "TAXONOMY_BRAND_ID",
        "TAXONOMY_PRODUCT_LABEL",
        "TAXONOMY_PRODUCT_ID",
        "TAXONOMY_ACTIVITY_LABEL",
        "TAXONOMY_ACTIVITY_ID",
        "TAXONOMY_CONTENT_LABEL",
        "TAXONOMY_CONTENT_ID",
        "TAXONOMY_CONTEXT_LABEL",
        "TAXONOMY_CONTEXT_ID"
    };

    public AddTaxonomy(String taxonomies, String a2c) {
        super(taxonomies, a2c, AddTaxonomy.class.getSimpleName());
    }

    /* (non-Javadoc)
     * @see com.gfk.hyperlane.engine.taxonomy.EnhanceOrUpdate#getSpecificOutputColu
     */
```

R Code

```
lwrgui <- function(path.in, path.out, path.constraints, path.settings, path.logs, jobstamp,
  ## This Tool is a GfK SE Marketing Sciences product. ##
  ## Contact Philipp Gaffert for questions and details.#
  ## Key developers are: Philipp Gaffert  (lead, core)
  ##                     Markus Ziegler   (GUI logic, KPI)
  ##                     Dr. Markus Lilienthal (outer core)
  ##                     Dr. Marek Jiruse (cleaning, error management)
  ##                     Dr. Volker Bosch (theory and former versions)
  ## June, 12th 2013
  ##
  # weighted.minmax=FALSE : minimum, maximum bounds are set relative to the design weight
  # dropcons=FALSE        : weighting ignores the constant, i.e. sample / population size is
  # jobstamp              : integer job ID
  # path.func             : location of R sources
  # path.in               : input data
  # path.out              : output folder
  # path.constraints      : path to constraints file
  # path.settings         : path to settings file


  #---------------------------#
  # Set input file definitions #
  #---------------------------#

  #path for R functions
  if (is.null(path.func)) {path.func <- getwd()}

  #make important parameters global
  jobstamp<<-jobstamp
  path.out<<-path.out
  path.logs<<-path.logs
  options(scipen=12)
```

Configurable Job Flows

JobFlow Templates



layout.json

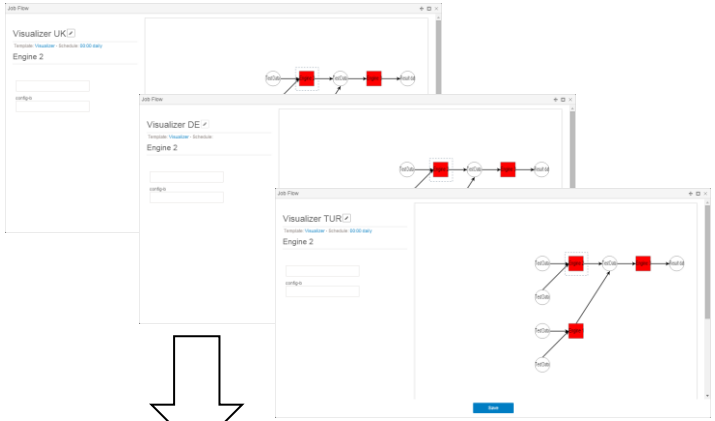
```
{
  "id": 4,
  "type": "circle",
  "name": "Test Data 4",
  "configuration": []
},
{
  "id": 5,
  "type": "rectangle",
  "name": "Engine 1",
  "configuration": [
    {
      "name": "config-a",
      "type": "string"
    }
  ]
},
{
  "id": 6,
  "type": "rectangle",
  "name": "Engine 2",
  "configuration": [
    {
      "name": "config-a",
      "type": "string"
    },
    {
      "name": "config-b",
      "type": "string"
    }
  ]
},
{
  "id": 7,
```

workflow.xml

```
<workflow-app name="HiveServer2-security" xmlns="uri:oozie:workflow:0.4">
  <credentials>
    <credential name="hive2" type="hive2">
      <property>
        <name>hive2.jdbc.url</name>
        <value>jdbc:hive2://test-cdh5-hue.ent.cloudera.com:10000/default</value>
      </property>
      <property>
        <name>hive2.server.principal</name>
        <value>hive/test-cdh5-hue.ent.cloudera.com@ENT.CLOUDERA.COM</value>
      </property>
    </credential>
  </credentials>
  <start to="simple_select"/>
  <action name="simple_select" cred="hive2">
    <hive xmlns="uri:oozie:hive-action:0.2">
      <job-tracker>${jobTracker}</job-tracker>
      <name-node>${nameNode}</name-node>
      <job-xml>hive-site3.xml</job-xml>
      <configuration>
        <property>
          <name>oozie.action.sharelib.for.hive</name>
          <value>hive2</value>
        </property>
        <property>
          <name>oozie.launcher.action.main.class</name>
          <value>org.apache.oozie.action.hadoop.Hive2Main</value>
        </property>
      </configuration>
      <script>select_generic1.sql</script>
      <param>connectString=${connectString}</param>
      <param>tableName=${tableName}</param>
    </hive>
    <ok to="end"/>
    <error to="kill"/>
  </action>
  <kill name="kill">
```

Scientists & Engineers

JobFlow Instances



de/config.json

```
{
    "tags": {
        "allowUnknownTags": true,
        "dictionaries": ["jsdoc","closure"]
    },
    "source": {
        "includePattern": ".+\\.js(doc)?$",
        "excludePattern": "(^|\\/|\\\\)_"
    },
    "plugins": [],
    "templates": {
        "cleverLinks": false,
        "monospaceLinks": false
    }
}
```

uk/config.json

```
{
    "tags": {
        "allowUnknownTags": true,
        "dictionaries": ["jsdoc","closure"]
    },
    "source": {
        "includePattern": ".+\\.js(doc)?$",
        "excludePattern": "(^|\\/|\\\\)_"
    },
    "plugins": [],
    "templates": {
        "cleverLinks": false,
        "monospaceLinks": false
    }
}
```

tur/config.json

```
{
    "tags": {
        "allowUnknownTags": true,
        "dictionaries": ["jsdoc","closure"]
    },
    "source": {
        "includePattern": ".+\\.js(doc)?$",
        "excludePattern": "(^|\\/|\\\\)_"
    },
    "plugins": [],
    "templates": {
        "cleverLinks": false,
        "monospaceLinks": false
    }
}
```

Operations Hub

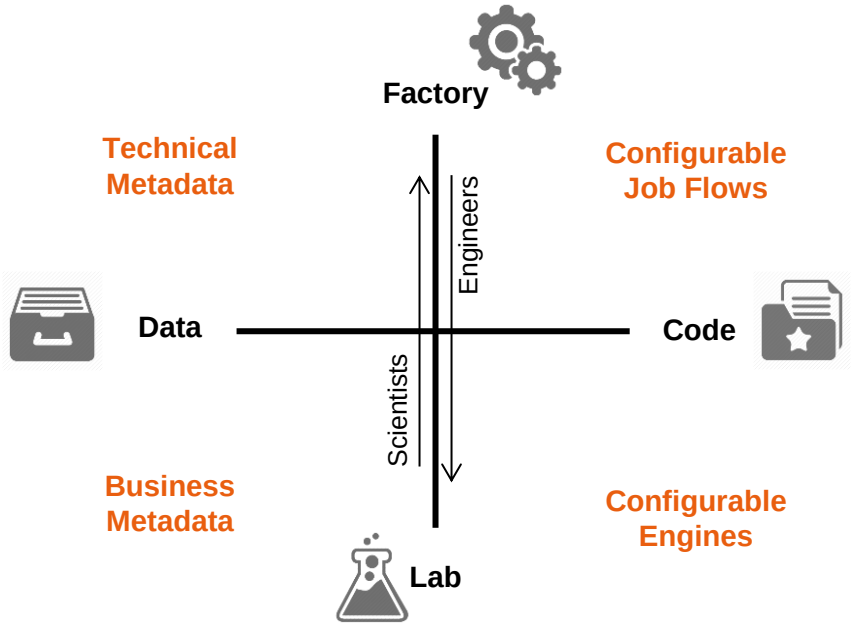
HYPERLANE WRAP UP

Berlin | 16 March 2015 | Get Together

Hyperlane. Wrap Up.

Improved findability
of relevant data sets using a data catalog that features rich meta data (context, lineage, evolving schemas).

Better collaboration
and strict quality controls enforced though sign-offs and clear accountabilities for providers and consumers of data sets.



More transparency for **productionized Job Flows** visually represented as interactive DAGs used for documentation, configuration and monitoring.

Easier fusion, imputation and weighting of data sets through **reusable and extensible "Engines"** implemented in PIG, JAVA and R.