Designing Feature DSLs:
Principles and Tradeoffs

Greg Kuhlmann

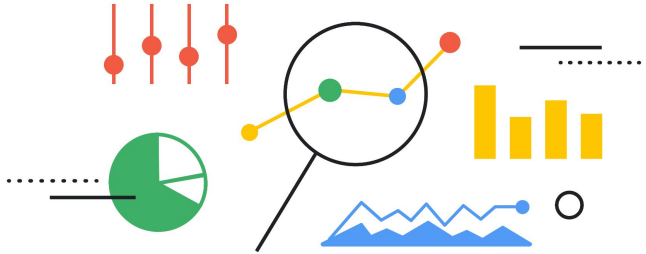
Co-founder & CEO

SUMATRA

FEATURE STORE SUMMIT 2022

Organized by HOPSWORKS

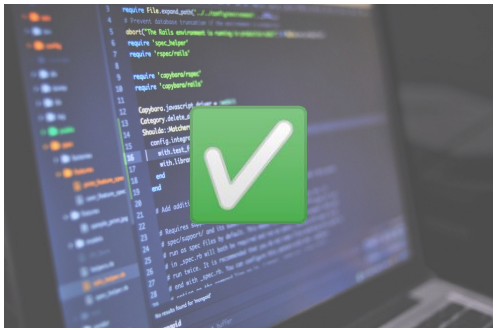
The Feature Engineering Process



Average Trip Length By Region

Idea

Correct Implementation



Research

Production

Feature Languages in the Wild

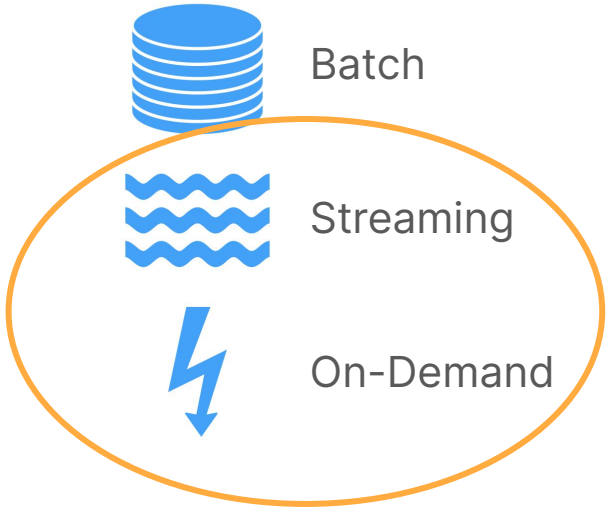
YAML

SQL + Jinja templates

Python decorators

PySpark classes

Clojure macros



Feature Store: Pre-transform → DataFrame API

On-the-fly stateful and stateless transformation

Opportunity for Feature Languages

Q. How do we evaluate a feature language?

A. Data scientist self-sufficiency

```
- name: trips_last_24h
  type: count
  entity: trip
  window: 24h
```

```
@on_demand_feature_view(
   sources=[
       driver_hourly_stats_view,
       input_request
   ],
   schema=[
     Field(name='conv_rate_plus_val1', dtype=Float64),
     Field(name='conv_rate_plus_val2', dtype=Float64)
   ]
)
def transformed_conv_rate(features_df: pd.DataFrame)
    -> pd.DataFrame:
    df = pd.DataFrame()
    df['conv_rate_plus_val1'] =
        (features_df['conv_rate'] + features_df['val_to_add'])
    df['conv_rate_plus_val2'] =
        (features_df['conv_rate'] + features_df['val_to_add_2'])
    return df
```

Concise

Domain-Oriented

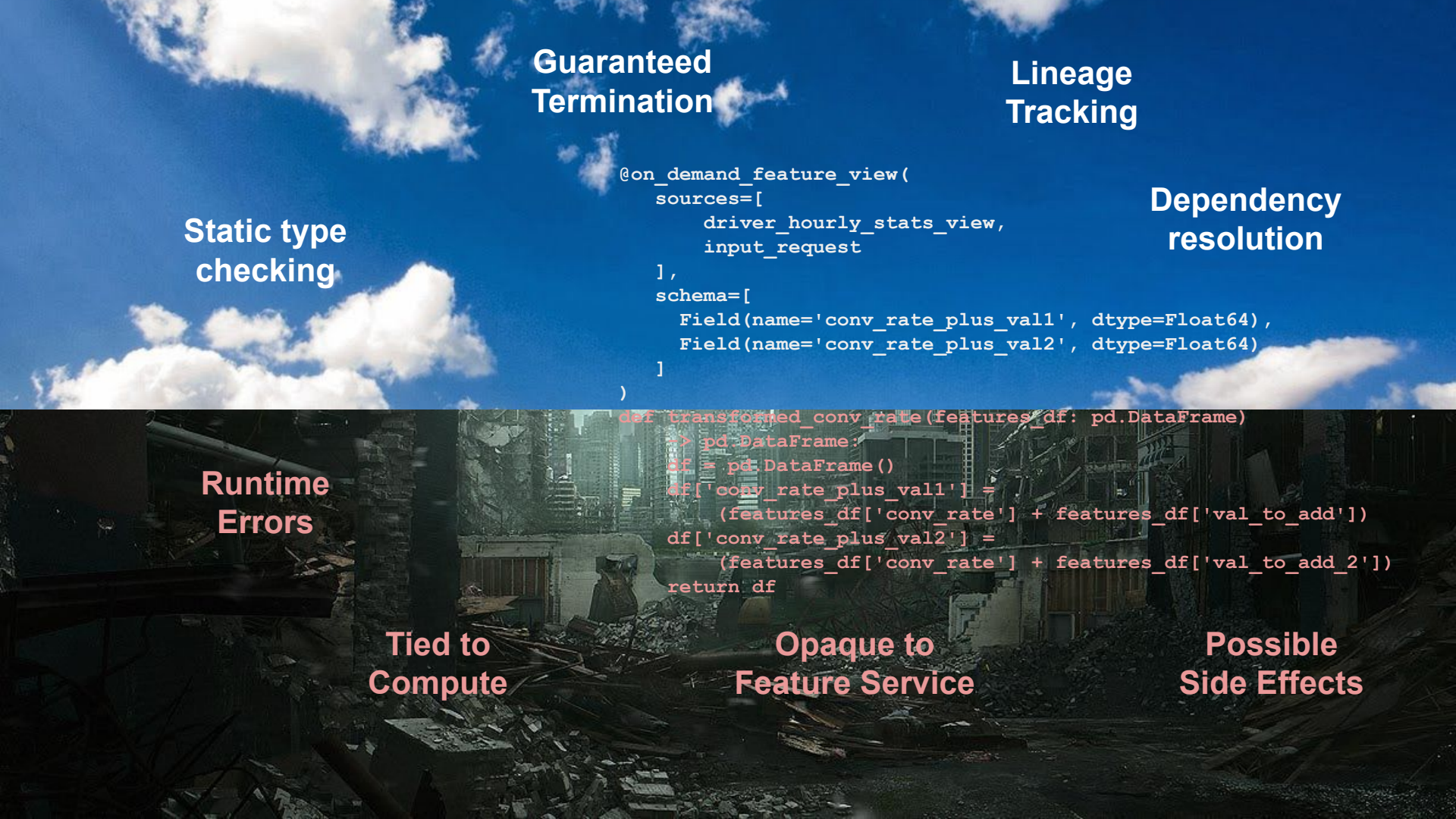
What not How

Flexible

General Purpose

“Real programming”

How do we get the best of both worlds?

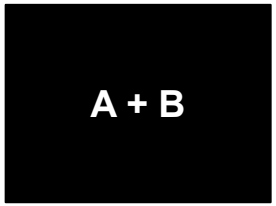

Guaranteed Termination
Lineage Tracking
Static type checking
Dependency resolution
@on_demand_feature_view(
sources=[
driver_hourly_stats_view,
input_request
],
schema=[
Field(name='conv_rate_plus_val1', dtype=Float64),
Field(name='conv_rate_plus_val2', dtype=Float64)
]
)
def transformed_conv_rate(features_df: pd.DataFrame)
-> pd.DataFrame:
df = pd.DataFrame()
df['conv_rate_plus_val1'] =
(features_df['conv_rate'] + features_df['val_to_add'])
df['conv_rate_plus_val2'] =
(features_df['conv_rate'] + features_df['val_to_add_2'])
return df
Runtime Errors
Tied to Compute
Opaque to Feature Service
Possible Side Effects

What if it were features all the way down?

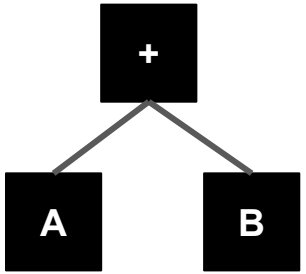
Not precious artifacts

Lightweight

Composable



❌



✅

```
-- example.scowl

  last_login := LatestTime<login>(by acct_id)
```

```
-- example.scowl


  last_login := LatestTime<login>(by acct_id)
  time_since := Minutes(EventTime()-last_login)
```

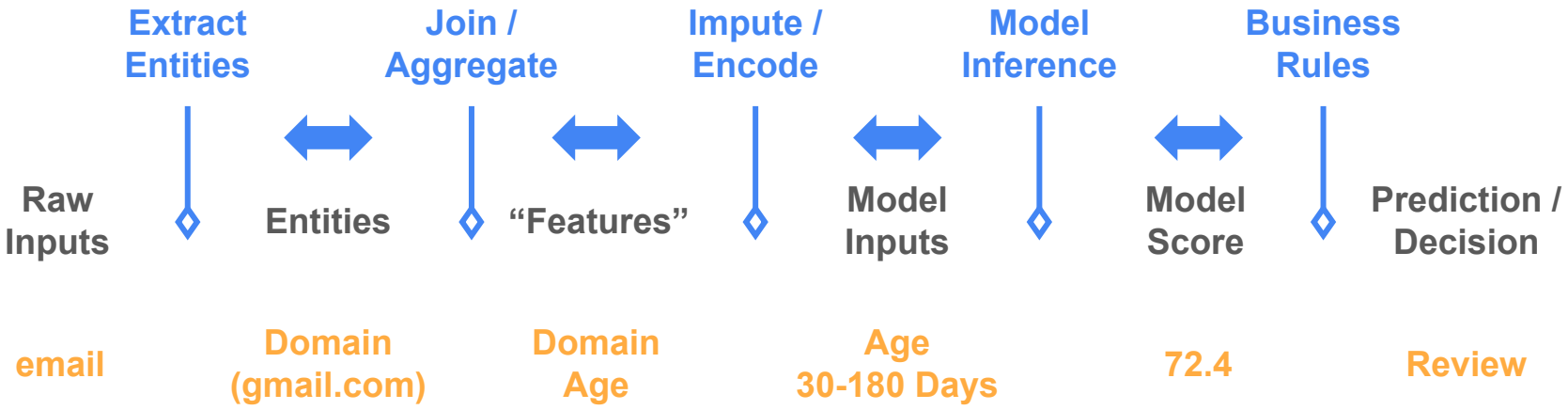
```
-- example.scowl

func MinutesSinceLogin(acct_id: string) {
  last_login := LatestTime<login>(by acct_id)
  time_since := Minutes(EventTime()-last_login)
  return time_since
}
```

```
-- example.scowl

func MinutesSinceLogin(acct_id: string) {
  last_login := LatestTime<login>(by acct_id)
  time_since := Minutes(EventTime()-last_login)
  return time_since
}

buyer_duration := MinutesSinceLogin(buyer_id)
seller_duration := MinutesSinceLogin(seller_id)
```

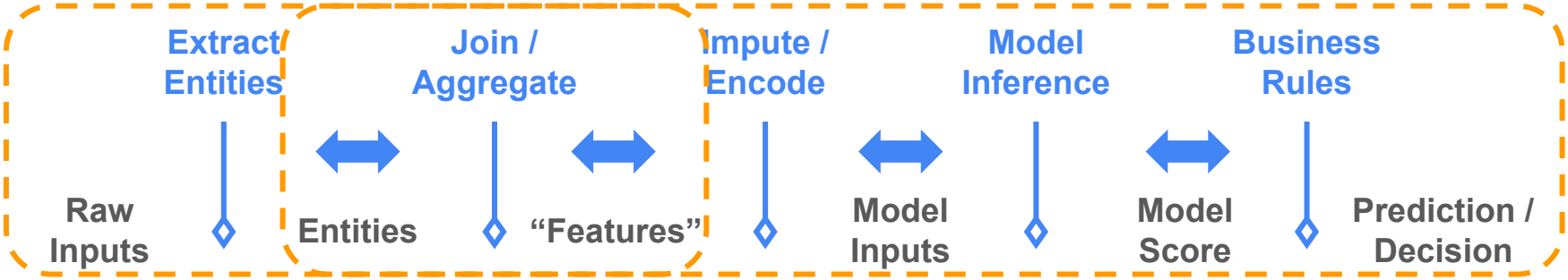
Functions...you invented functions. 🙄

Ah, yes, but functions over *features*. 🤩

Features: Deep AND WIDE

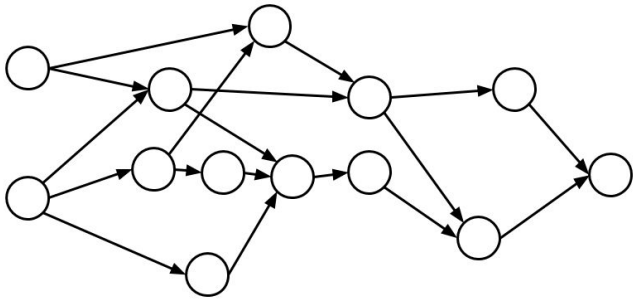


One big, beautiful DAG



Type checking end-to-end 👍

Feature versioning for free 👍

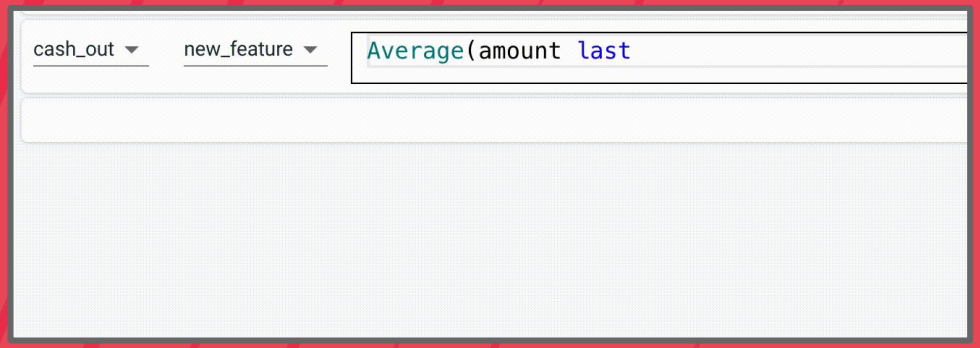


Logging, monitoring, lineage-tracking 👍

Online-offline reproducibility **of the entire DAG** 🔥

```
20
21 is_search := url.Contains('/s?search')
22 cidr_16 := CIDRBlock(ip, 16)
23 cidr_browser := cidr_16 + user_agent_str
24 unique_searches_2h
25
26
27
28
29
30
```

Syntax Highlighting & Autocomplete



REPL

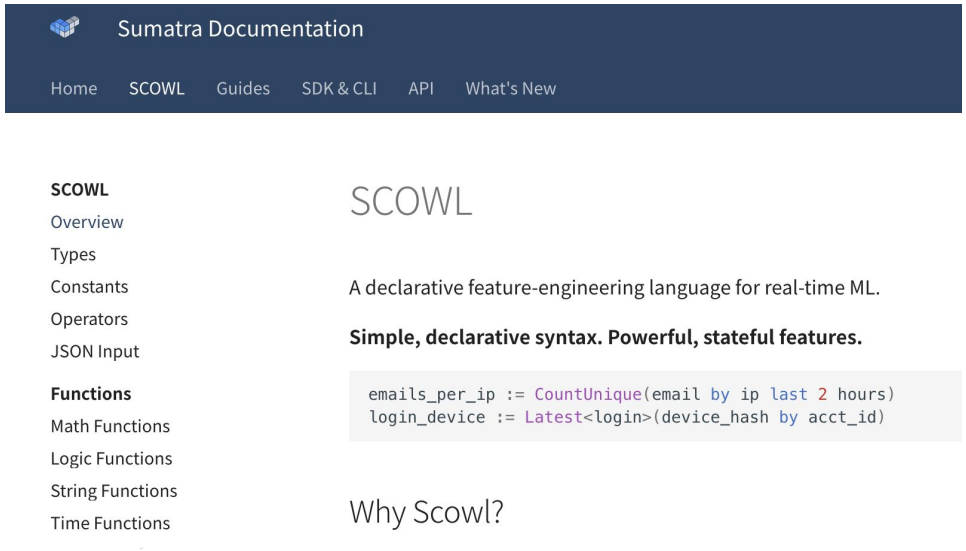
What else do we want from a feature language?

Type inference

Complex data types

Missing values (Maybe monad)

Error isolation



docs.sumatra.ai

Acknowledgments

Bob Nugmanov, DoorDash

Max Halford, Carbonfact

Adriano Freitas, Cherre

Thank you!

Greg Kuhlmann

Co-Founder / CEO, Sumatra

greg@sumatra.ai