

DESIGNING DATA PIPELINES THAT DON'T HATE YOU SIX MONTHS LATER

A Sci-Fi Kung Fu Inspired Story About Data Engineering Fundamentals

Chris Birie | Stir Trek



ABOUT ME





YES I USED AI TO GENERATE THESE ALL PHOTOS
IN THIS PRESENTATION...

DEAL WITH IT!!!







Power Command Center

- # announcements
- # general
- # media
- # gaming
- # tech-talk
- # homebrew
- # automation
- # devops
- # projects
- # voice-chat
- # clip-highlights

VOICE CHANNELS

- Lounge
- Watch Party
- Gaming 1
- Gaming 2
- Focus Room



Clean Code
A Handbook of Agile Software Craftsmanship
Robert C. Martin

The Pragmatic Programmer
Hart, Thomas

Design Patterns
Ganglia - Helm - Johnson - Vlissides

Refactoring
Martin Fowler

The Mythical Man-Month
Frederick P. Brooks, Jr.

The Art of Computer Programming
Donald E. Knuth

IDEAS
PLANS
SYSTEMS
LEGACY

EVERYDAY
LEVEL UP

BUILD
PROTECT
INSPIRE

Data Pipelines!
Specifically great data
pipelines that don't hate you
six months later!



```

EXPLORER
  DATA PIPELINE
  > _pychoe
  > configs
  > dags
  > libs
  > models
  > tests
  > __init__.py
  > datacopy.py
  > load.py
  > transform.py
  > validate.py
  > requirements.txt
  > README.md

transform.py
1 def transform(df):
2     # Rename columns
3     df = df.withColumnRenamed('
4
5     # Cast types
6     df = df.withColumn('created_at',
7         cast('timestamp'))
8
9     # Business Logic
10    df = apply_business_rules(df)
11
12    # Join with reference data
13    df = df.join(ref_df, on='custo
14
15    return df
16
17
18 def apply_bu
19     # Some con
20     return
21
22
23
24
25
26
27
28
29
30

```

```

3:00 AM
[ERROR] Schema mismatch: expected 'customer_id' but found 'cust_id'
[ERROR] Missing required column: 'account_status'
[ERROR] Type mismatch: expected 'created_at' as TIMESTAMP, received STRING
[ERROR] Downstream model out of sync with source schema
[ERROR] Duplicate records detected for run_id=2026-04-28
[ERROR] Primary key violation: record already exists
[WARN] Retry produced different row count than original run
[ERROR] Merge failed: duplicate keys found in source batch
[ERROR] Pipeline failed with exit code 1
[WARN] No metrics emitted for job duration, row count, or error count
[WARN] No correlation_id found for failed batch
[ERROR] Alert triggered, but failing step is unknown
[ERROR] Production validation failed: unexpected null rate in customer_id
[ERROR] Transformation output does not match expected contract
[ERROR] Deprecated field still referenced: legacy_customer_id
[ERROR] Contract violation: downstream expects one row per customer
[WARN] Business rule changed but transformation logic was not updated
[ERROR] Reporting layer incompatible with new aggregation grain

```

#data-pipeline-alerts 25

Sarah 2:59 AM
Did the upstream team change the payload again?

Mike 2:59 AM
Can we safely rerun this job or will it double-load everything?

Priya 2:59 AM
Where did it fail? Do we have logs for that?

Alex 3:00 AM
How did this pass QA?

Jordan 3:00 AM
That's not how the process works anymore.

Message #data-pipeline-alerts

What would Cooper do?

MAYBE NOTHING IS REAL

FOCUS COMPARTMENT SHEER WILL

DON'T TRUST THEM ANYONE

Goonies never die



The background is a dark room, likely a home office or a collector's space. On the wall, several movie posters are visible, including 'The Last Starfighter', 'Inception', and 'Ghostbusters'. In the foreground, a desk is cluttered with various items: a smartphone displaying '18 Messages', a Rubik's cube, a small toy car, and several stacks of VHS tapes, including 'The Last Starfighter', 'Galaxy Quest', and 'The Goonies'.

[Freeze Frame]
Record Scratch

You may be wondering how I got here...

**SIX MONTHS
EARLIER...**



Enter the Dragon Bruce.

Bruce was brilliant.

Bruce was fast.

Bruce used AI.

Bruce shipped early.

The pipeline worked beautifully

Everyone was happy.

Bruce got a pat on the back.

And probably a promotion.

Bruce moved on. Bruce is not here anymore.



This was Bruce's pipeline.

It is now your problem.

*Pipelines built on shaky fundamentals don't fail on day one.
They fail when pressure arrives.*

BRUCE DIDN'T BUILD A BAD PIPELINE.

He built a pipeline that worked perfectly for the fight he trained for.

The problem is the fight changed.

*“The pipeline didn't fail because Bruce was lazy or bad at his job.
It failed because nobody taught him the fundamentals, or they were ignored.
And then Bruce was gone.”*

Bruce is gone. But his pipeline lives on.

And right now, at 3am, it is absolutely on fire.

BUT WHAT IF WE COULD GO BACK?

*Six months ago, Bruce was building the pipeline.
He didn't know what was coming. Nobody did.
But what if we could travel back and change the outcome?
What would we teach him?*

*Every lesson ahead is one Bruce never learned.
Each principle — a mistake he made so you don't have to.*

Five lessons. Five chances to change the outcome.

THE FIVE LESSONS BRUCE NEVER LEARNED

01

SCHEMA EVOLUTION

Don't Build Your Stance on Sand

02

IDEMPOTENCY

You think a fight is one throw, one kick?

03

OBSERVABILITY

Don't charge him blindly. You've got to listen. Listen.

04

TESTABILITY

What was that? An exhibition?

05

DESIGN FOR CHANGE

Be Water My Friend

*"A warrior who only trains for the
tournament
will fall when the real fight comes."*

— Some Wise Sifu (Probably)



```

3:00 AM
[ERROR] Schema mismatch: expected 'customer_id' but found 'cust_id'
[ERROR] Missing required column: 'account_status'
[ERROR] Type mismatch: expected 'created_at' as TIMESTAMP, received STRING
[ERROR] Downstream model out of sync with source schema
[ERROR] Duplicate records detected for run_id=2026-04-28
[ERROR] Primary key violation: record already exists
[WARN] Retry produced different row count than original run
[ERROR] Merge failed: duplicate keys found in source batch
[ERROR] Pipeline failed with exit code 1
[WARN] No metrics emitted for job duration, row count, or error count
[WARN] No correlation_id found for failed batch
[ERROR] Alert triggered, but failing step is unknown
[ERROR] Production validation failed: unexpected null rate in customer_id
[ERROR] Transformation output does not match expected contract
[ERROR] Deprecated field still referenced: legacy_customer_id
[ERROR] Contract violation: downstream expects one row per customer
[WARN] Business rule changed but transformation logic was not updated
[ERROR] Reporting layer incompatible with new aggregation grain

```

#data-pipeline-alerts 25

Sarah 2:59 AM
Did the upstream team change the payload again?

Mike 2:59 AM
Can we safely rerun this job or will it double-load everything?

Priya 2:59 AM
Where did it fail? Do we have logs for that?

Alex 3:00 AM
How did this pass QA?

Jordan 3:00 AM
That's not how the process works anymore.

Message #data-pipeline-alerts

```

EXPLORER
DATA PIPELINE
  __pycache__
  > configs
  > dags
  > lib
  > models
  > tests
  __init__.py
  datacopy.py
  load.py
  transform.py
  validate.py
requirements.txt
README.md

transform.py
1 def transform(df):
2     # Rename columns
3     df = df.withColumnRenamed({
4
5
6
7
8
9     # Cast types
10    df = df.withColumn('created_at',
11                      cast('timestamp')).
12
13    # Business Logic
14    df = apply_business_rules(df)
15
16    # Join with reference data
17    df = df.join(ref_df, on='cust_
18
19    return df
20
21 def apply_bu
22     # Some con
23     return

```

What would Cooper do?

MAYBE NOTHING IS REAL

FOCUS COMPARTMENT SHEER WILL

DON'T TRUST THEM ANYONE

Goonies never say die





THE LAST STARFIGHTER

INCEPTION

THE GOONIES

DOUTATIME

MATRIX

BACKS TO FUTURE

GHOSTBUSTERS

INTELLIGENCE

GALAXY QUEST

I CODA THEREFORE I CAFFEINATE

DATA > SLEEP

NASA
STARBUCKERS

WHAT IF...?



SCHEMA EVOLUTION

"Don't Build Your Stance on Sand"

BRUCE'S MISTAKE

6 months ago: Bruce trusted the upstream system to not change, and they didn't change during warranty.

Today: The source system changed their database/file layout and source types but didn't let us know. They also sent us some corrupted columns that should have had dates, but it had gibberish

Result: Bruce's pipeline exploded not so silently with an array of errors

THE LESSON

"Adapt what is useful. Reject what is useless."

Bruce trusted what the source sent him and never built a system to question it.

Adapt what is useful — accept valid data, handle schema changes gracefully, version your contracts.

Reject what is useless — validate at the boundary, route bad records before they corrupt what's downstream.

Your pipeline's job is to be paranoid at the front door so nothing weird makes it into the house.

THE FIX

- Nullable fields by default/default values in schema definition
- Versioned schema contracts that are checked regularly
- Defensive parsing/data validation
- Never blindly trust the source, trust but verify

"Your pipeline will outlive the person who designed the source system. Plan for that."

THE PAST HAS CHANGED!



```

EXPLORER
├── DATA_PIPELINE
│   ├── __pycache__
│   ├── configs
│   ├── dags
│   ├── lib
│   ├── models
│   ├── tests
│   ├── util.py
│   ├── viz.py
│   └── load.py
├── transform.py
├── validation.py
├── requirements.txt
└── README.md

transform.py
1 def transform(df):
2     # Rename columns
3     df = df.withColumnRenamed(
4         'customer_id', 'customer_id_v2')
5
6     # Cast types
7     df = df.withColumn('created_at',
8         cast('timestamp'))
9
10    # Business Logic
11    df = apply_business_rules(df)
12
13    # Join with reference data
14    df = df.join(ref_df, on='cast',
15        how='left')
16
17    return df
18
19 def apply_business_rules(df):
20     # Some complex logic
21     return df
22
23
24
25
26
27
28
29
30
31
32
33
34
35

```



```

3:00 AM
[ERROR] Duplicate records detected for run_id=2026-04-28
[ERROR] Primary key violation: record already exists
[WARN] Retry produced different row count than original run
[ERROR] Merge failed: duplicate keys found in source batch
[ERROR] Pipeline failed with exit code 1
[WARN] No metrics emitted for job duration, row count, or error count
[WARN] No correlation_id found for failed batch
[ERROR] Alert triggered, but failing step is unknown
[ERROR] Production validation failed: unexpected null rate in customer_id
[ERROR] Transformation output does not match expected contract
[ERROR] Deprecated field still referenced: legacy_customer_id
[ERROR] Contract violation: downstream expects one row per customer
[WARN] Business rule changed but transformation logic was not updated
[ERROR] Reporting layer incompatible with new aggregation grain

```

#data-pipeline-alerts 20

Mike 2:59 AM
Can we safely rerun this job or will it double-load everything?

Priya 2:59 AM
Where did it fail? Do we have logs for that?

Alex 3:00 AM
How did this pass QA?

Jordan 3:00 AM
That's not how the process works anymore.

+ Message #data-pipeline-alerts

What would Cooper do? MAYBE NOTHING IS REAL FOCUS COMMITMENT SHEER WILL DON'T TRUST THEY ANYONE Goonies never say die.





THE LAST STARFIGHTER
Peter Finkbeiner and Richard

INCEPTION
Christopher Nolan

THE GOONIES
Richard Donner

OUTATIME

MATRIX

BACK TO THE FUTURE

GHOSTBUSTERS

INTERSTELLAR

GALAXY QUEST

I CODA
THEREFORE
I CAFFEINATE

DATA > SLEEP

WHAT IF...?



IDEMPOTENCY

"You think a fight is one thron, one kick?"

BRUCE'S MISTAKE

6 months ago: Bruce built the pipeline to run once per day, start to finish, no interruptions. That was the plan. That assumption was baked into every line of code.

Today: The target database went down halfway through the load. The support team brought it back up and re-ran the pipeline to finish the job.

Result: Primary key violations. Duplicate records. Audit flags on doubled counts. The fix took longer than the original outage.

THE LESSON

"Expect to be hit more than once. The fighter who trains for one clean exchange is not prepared for the real fight."

A pipeline that can only safely run once is a pipeline waiting to fail.

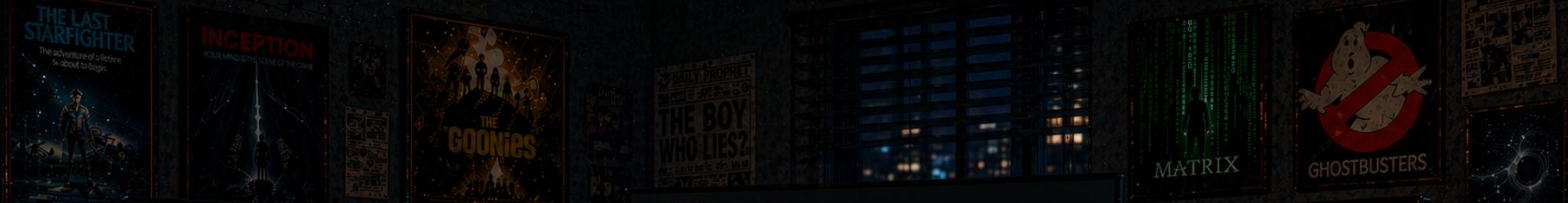
The database will come back up mid-load. The orchestrator will retry on timeout. Someone will ask for a backfill. In a real fight, you don't get one clean exchange — plan for all it.

THE FIX

- Deterministic keys
- Upsert over insert
- Safe reprocessing windows
- Ask yourself, is this safe to retry?

"If running it twice breaks it, it was already broken."

THE PAST HAS CHANGED!



```

DIPLOMER
- DATA_PIPELINE
  > __pycache__
  > configs
  > dags
  > lib
  > models
  > tests
  > util.py
  > extract.py
  > load.py
  > transform.py
  > validate.py
  > requirements.txt
  > README.md

transform.py x schema_util.py x load.py x
1 def transform(df):
2     # Rename columns
3     df = df.withColumnRenamed(
4         "customer_id", "customer_id_v2")
5
6     # Cast types
7     df = df.withColumn("created_at",
8         cast("timestamp"))
9
10    # Business Logic
11    df = apply_business_rules(df)
12
13    # Join with reference data
14    df = df.join(ref_df, on="customer_id", how="left")
15    return df
16
17 def apply_business_rules(df):
18     # some code
19     return df
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
  
```

```

3:00 AM
[ERROR] Pipeline failed with exit code 1
[WARN] No metrics emitted for job duration, row count, or error count
[WARN] No correlation_id found for failed batch
[ERROR] Alert triggered, but failing step is unknown

[ERROR] Production validation failed: unexpected null rate in customer_id
[ERROR] Transformation output does not match expected contract
[WARN] No unit tests found for transform.py
[WARN] No fixture data available for this edge case

[ERROR] Deprecated field still referenced: legacy_customer_id
[ERROR] Contract violation: downstream expects one row per customer
[WARN] Business rule changed but transformation logic was not updated
[ERROR] Reporting layer incompatible with new aggregation grain
  
```

#data-pipeline-alerts 15

Priya 2:59 AM
Where did it fail? Do we have logs for that?

Alex 3:00 AM
How did this pass QA?

Jordan 3:00 AM
That's not how the process works anymore.

Message #data-pipeline-alerts

What would Cooper do? MAYBE NOTHING IS REAL. FOCUS COMMITMENT SHEER WILL. DON'T TRUST THEM ANYONE. Goonies never say die.





THE LAST STARFIGHTER
The Last Starfighter

INCEPTION
Inception

THE GOONIES
The Goonies

OUTATIME
OUTATIME

MATRIX
MATRIX

Coors LIGHT
Coors LIGHT

BACK TO THE FUTURE
Back to the Future

GHOSTBUSTERS
GHOSTBUSTERS

INTERSTELLAR
Interstellar

GALAXY QUEST
Galaxy Quest

I CODA
THEREFORE
I CAFFEINATE

DATA > SLEEP
DATA > SLEEP

WHAT IF...?



OBSERVABILITY

Don't charge him blindly. You've got to listen. Listen.

BRUCE'S MISTAKE

6 months ago: Bruce got the data flowing and called it done. No logging beyond basic errors. No metrics. No alerts. The pipeline ran — and that was enough.

Today: The pipeline failed silently for three days. No alerts fired. No dashboards moved. The business found out from a customer complaint about stale data.

Result: Three days of bad data downstream. A frantic audit to determine what was affected. A 3am incident that was completely invisible until it wasn't.

THE LESSON

"The warrior who fights without information fights twice."

Bruce's pipeline was running blind. He knew whether the job finished — not whether the data was right.

A pipeline that finishes is not the same as a pipeline that works.

THE FIX

- Row counts at every stage
- Null rate tracking
- Latency monitoring
- Alert on anomalies, not just exceptions

"Hope is not a monitoring strategy."

THE PAST HAS CHANGED!



```

EXPLORER
- DATA_PIPELINE
  > __pycache__
  > configs
  > dags
  > lib
  > models
  > tests
  > job.py
  > main.py
  > load.py
  > transform.py
  > validate.py
  > requirements.txt
  > README.md

transform.py
1 def transform(df):
2     # Pipeline columns
3     df = df.withColumnRenamed({
4         # Cast types
5         # Business logic
6         df = apply_business_rules(df)
7     })
8     # Join with reference data
9     df = df.join(ref_df, on="cust",
10                return_df)
11
12 def apply_business
13     # some code
14     return
15

```

3:00 AM

```

[ERROR] Production validation failed: unexpected null rate in customer_id
[ERROR] Transformation output does not match expected contract
[WARN] No unit tests found for transform.py
[WARN] No fixture data available for this edge case

[ERROR] Deprecated field still referenced: legacy_customer_id
[ERROR] Contract violation: downstream expects one row per customer
[WARN] Business rule changed but transformation logic was not updated
[ERROR] Reporting layer incompatible with new aggregation grain

```

#data-pipeline-alerts 10

Alex 3:00 AM
How did this pass QA?

Jordan 3:00 AM
That's not how the process works anymore.

Sam 3:01 AM
Can we get an ETA on a fix?

Message #data-pipeline-alerts

What would Cooper do?

MAYBE NOTHING IS REAL

FOCUS COMMITMENT SHEER WILL

DON'T TRUST THEM ANYONE

Goonies never say die.



It's not a bug
It's an extreme feature!



THE LAST STARFIGHTER
The Last Starfighter
The Last Starfighter

INCEPTION
INCEPTION
INCEPTION

THE GOONIES
THE GOONIES
THE GOONIES

OUTATIME

MATRIX

PHONE

BACK TO THE FUTURE
BACK TO THE FUTURE
BACK TO THE FUTURE

GHOSTBUSTERS
GHOSTBUSTERS
GHOSTBUSTERS

INTERSTELLAR
INTERSTELLAR
INTERSTELLAR

GALAXY QUEST
GALAXY QUEST
GALAXY QUEST

I CODA
THEREFORE
I CAFFEINATE

DATA > SLEEP

WHAT IF...?



TESTABILITY

"What was that? An exhibition?"

BRUCE'S MISTAKE

6 months ago: Bruce built the pipeline and tested it in production. It worked. Data came in, data went out. Looked right. He shipped it.

Today: A bug was found — records with null values in a key field were being silently dropped. The only way to fix it was to run the corrected version in production and watch what happened.

Result: Every change is a live gamble. Every fix introduces new risk. Nobody has confidence in anything because nothing has ever been tested outside of the real environment.

THE LESSON

"I fear not the man who has practiced 10,000 kicks once, but the man who has practiced one kick 10,000 times." — Bruce Lee

You don't find out if a technique works in a real fight. You already know — because you ran it a thousand times in the gym.

Your pipeline should work the same way. If you've never run it outside of production, you haven't trained at all.

THE FIX

- Pure transformation functions
- Parameterized inputs
- Fixture-based unit tests
- Test without consequences

"Train like you would fight. Don't fight to find out if you trained."

THE PAST HAS CHANGED!



```

DIPLOMER
DATA_PIPELINE
  > __ycache__
  > configs
  > logs
  > lo
  > models
  > news
  > _pk__py
  > extract.py
  > load.py
  > transform.py
  > validate.py
  > replacement.txt
  > README.md
transform.py
1 def transform(df):
2     # Remove nulls
3     df = df.withColumnRenamed(
4         'cast', 'timestamp')
5
6
7     # Cast types
8     df = df.withColumn("created_at",
9         cast("timestamp"))
10
11
12     # Business logic
13     df = apply_business_rules(df)
14
15
16     # Join with reference data
17     df = df.join(ref_df, on="cast")
18
19
20     return df
21
22
23
24
25 def apply_business
26     # some code
27
28
29     return
30
  
```

3:00 AM

[ERROR] Contract violation: downstream expects one row per customer

[WARN] Business rule changed but transformation logic was not updated

[ERROR] Reporting layer incompatible with new aggregation grain

#data-pipeline-alerts 5

Jordan 3:00 AM
That's not how the process works anymore.

Sam 3:01 AM
Can we get an ETA on a fix?

Message #data-pipeline-alerts

What would Cooper do?

MAYBE NOTHING IS REAL

FOCUS COMMITMENT SHEER WILL

DON'T TRUST THEM ANYONE

Goonies never say die.



It's not a bug it's an extreme feature





THE LAST STARFIGHTER
The Last Starfighter

INCEPTION
Inception

THE GOONIES
The Goonies

OUTATIME

MATRIX

BACK TO THE FUTURE
Back to the Future

GHOSTBUSTERS
Ghostbusters

INTERSTELLAR
Interstellar

GALAXY QUEST
Galaxy Quest

I CODA
THEREFORE
I CAFFEINATE

DATA > SLEEP

WHAT IF...?



DESIGN FOR CHANGE

"Be water, my friend"

BRUCE'S MISTAKE

6 months ago: Bruce's pipeline was tight, clean, and exactly right for the problem in front of him. It handled the volume, the schema, the delivery cadence. It was built for the fight he knew was coming.

Today: Volume is 100x what it was. Requirements shifted. The source system restructured. The business wants to backfill 18 months of history.

Result: The pipeline didn't bend. It shattered. A full rewrite — not because the original was bad, but because it was built for one specific fight and the fight changed.

THE LESSON

"Notice that the stiffest tree is most easily cracked, while the bamboo or willow survives by bending with the wind."

Water doesn't resist the container. It adapts. Bruce's pipeline was concrete — perfectly shaped for the original mold. When the mold changed, the concrete cracked. Build pipelines that bend.

THE FIX

- Modular pipeline stages
- Backfill-friendly design
- Smart partition strategies
- Loose coupling between layers

"Volume is the easy scaling problem. Changing requirements is the real test."

THE PAST HAS CHANGED!



THE LAST
STARFIGHTER

INCEPTION

THE
GOONIES

MATRIX

GHOSTBUSTERS

OUTATIME

INTERSTELLAR

INTERSTELLAR

GALAXY QUEST

BACK TO THE FUTURE

3:00 AM

DATA
>
SLEEP

Bruce Redeemed.

BEFORE

- ▶ Silent failures at 3am
- ▶ Doubled data on reruns
- ▶ Untestable, unable to do any meaningful performance tests or stress tests
- ▶ Explodes when schema changes
- ▶ Full rewrite when requirements shift

AFTER

- ▶ Anomaly alerts catch issues early and fix them or move on not breaking the pipeline
- ▶ Idempotent — safe to retry
- ▶ Full test suite, no consequences
- ▶ Schema evolution handled gracefully
- ▶ Modular — bends instead of breaks

BEFORE YOU GO WATCH THE MOVIE

- 01 Build for schema change.
- 02 Make it idempotent.
- 03 Observe everything.
- 04 Test without consequences.
- 05 Design for Changes...Be water.

Bruce left before he could fix it.

You're still here.

You can fix it!

Start Small, Do Something

"Go enjoy the movie. You've already seen the real disaster film."

Chris Birie

Software & Data Martial Artist

chris.birie@gmail.com



Designing Data Pipelines That Don't Hate You Six Months Later

Feedback



LinkedIn