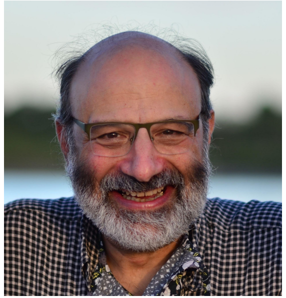
Why we need User Behavior Insights

MICES, Berlin
June 2024

Stavros Macrakis & Charlie Hull

Who are we?

- **Stavros Macrakis**, Senior Technical Product Manager for the OpenSearch Project, focusing on document and e-commerce search.
- Worked on search at Lycos, FAST, GLG, Google, and AWS for almost 20 years and is passionate about search relevance.



- **Charlie Hull,** UK Director & Marketing Director at OpenSource Connections
- 25 years in search as a project lead, manager, trainer, speaker, blogger
- Past lives included electronic engineering & circus performer

Who are OpenSource Connections ?

We've helped clients around the globe to get good search & AI done: whether they are stuck with search & AI or whether they want to get their search & AI to the next level.

Our focus areas:

- Understand & Delight Your Users
- Expert Search & AI Engineering
- Build Powerful People and Process

What we offer:

- Consulting
- Training
- Specialist Staffing

What is OpenSearch?

Open source search and analytics suite
used by thousands of companies worldwide

Applications:

- E-commerce search
- Document search
- Log analytics

Features:

- Apache 2.0 license
- Fork of Elastic 7.10
- AWS hosted service available

All search needs behavioral data

Search data for e-commerce

- Standard today
  - Aggregated metrics
    - Page flow
    - Abandonment
    - Click frequency
    - Sales
- But how can we tune search results for e-commerce?
  - Granular data for analysis
    - What searches lead to sales?
    - What is the user journey?
    - What sequence of searches do users use?
    - Which facets are most useful?

Every corpus and ever user population is different

Looking for items, not documents (heavy on structured data)

- E-commerce search and recommendations: looking for a thing to buy
  - Real estate
  - Books
  - Plumbing parts
  - Electronics
  - Media – movies, TV, …
  - Availability constraints: hotel rooms, flights, fractional jets, …
  - Geographic constraints: ride share, repair services, maid services, …
- Job search; placements
- Expert search
- Photos by description or by similarity; SmugMug very different from ShutterStock

… and various kinds of text-heavy search

- Document search and recommendations (heavy on text)
  - Academic documents
  - News
  - Intranet, highly heterogenous, inconsistent formats
  - Knowledge management
  - Legal and regulatory, both high-precision (relevant regulation) and high-recall (discovery)
  - FAQs, call centers, troubleshooting
- Embedded
  - Company search within a financial services app
  - Person search within email
  - Applications

Search

=

Relevance ranking

E-commerce search
=
Relevance ranking + merchandising

User Behavior Insights (UBI)

Goal: Collect all information needed for evaluating search
at high granularity

- What did the user search for?
- Were the results reasonable?
- How did they refine the search – modified query, facets?
- What search or sequence of searches led to a purchase of the pink Pet Rock with Overalls?
- What did they click on? … other actions… view details, add to basket, checkout basket
- How are actions related *causally*, not chronologically?

What *should* all search implementations do?

Berlin Buzzwords 2022

- Log search actions
- Understand application data
- Evaluate quality of results
- Talk to users and sponsors
- Regression testing
- Performance testing
- Multilingual coverage
- Privacy

Customer

- Search specialist engineers
  - Search is a *discipline*
- Logging
- Quality evaluation
- Incorporating multiple signals
- Tuning

What *should* all search implementations do, but *don't*?

Berlin Buzzwords 2022 gaps

- Log search actions
- Understand application data
- Evaluate quality of results
- Talk to users and sponsors
- Regression testing
- Performance testing
- Multilingual coverage
- Privacy

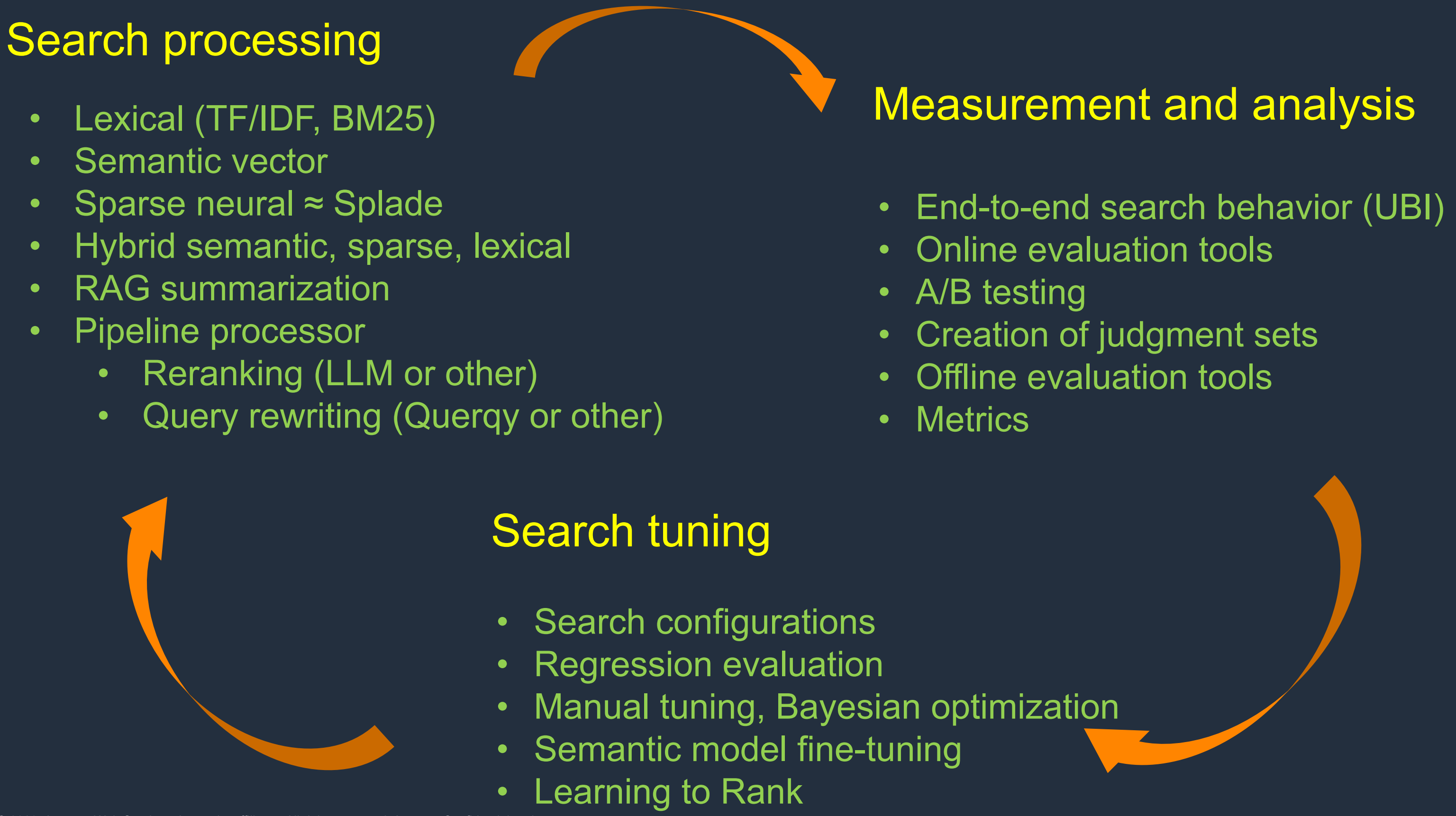
Customer gaps

- Search specialist engineers
    - Search is a *discipline*
- Logging
- Quality evaluation
- Incorporating multiple signals
- Tuning

Calculating search results: search processing

- Lexical (TF/IDF, BM25)
- Semantic vector
- Sparse neural ≈ Splade
- Hybrid semantic, sparse, lexical
- RAG summarization
- Pipeline processor
  - Reranking (LLM or other)
  - Query rewriting (Querqy or other)

The virtuous circle of search relevance



Search processing

- Lexical (TF/IDF, BM25)
- Semantic vector
- Sparse neural ≈ Splade
- Hybrid semantic, sparse, lexical
- RAG summarization
- Pipeline processor
  - Reranking (LLM or other)
  - Query rewriting (Querqy or other)

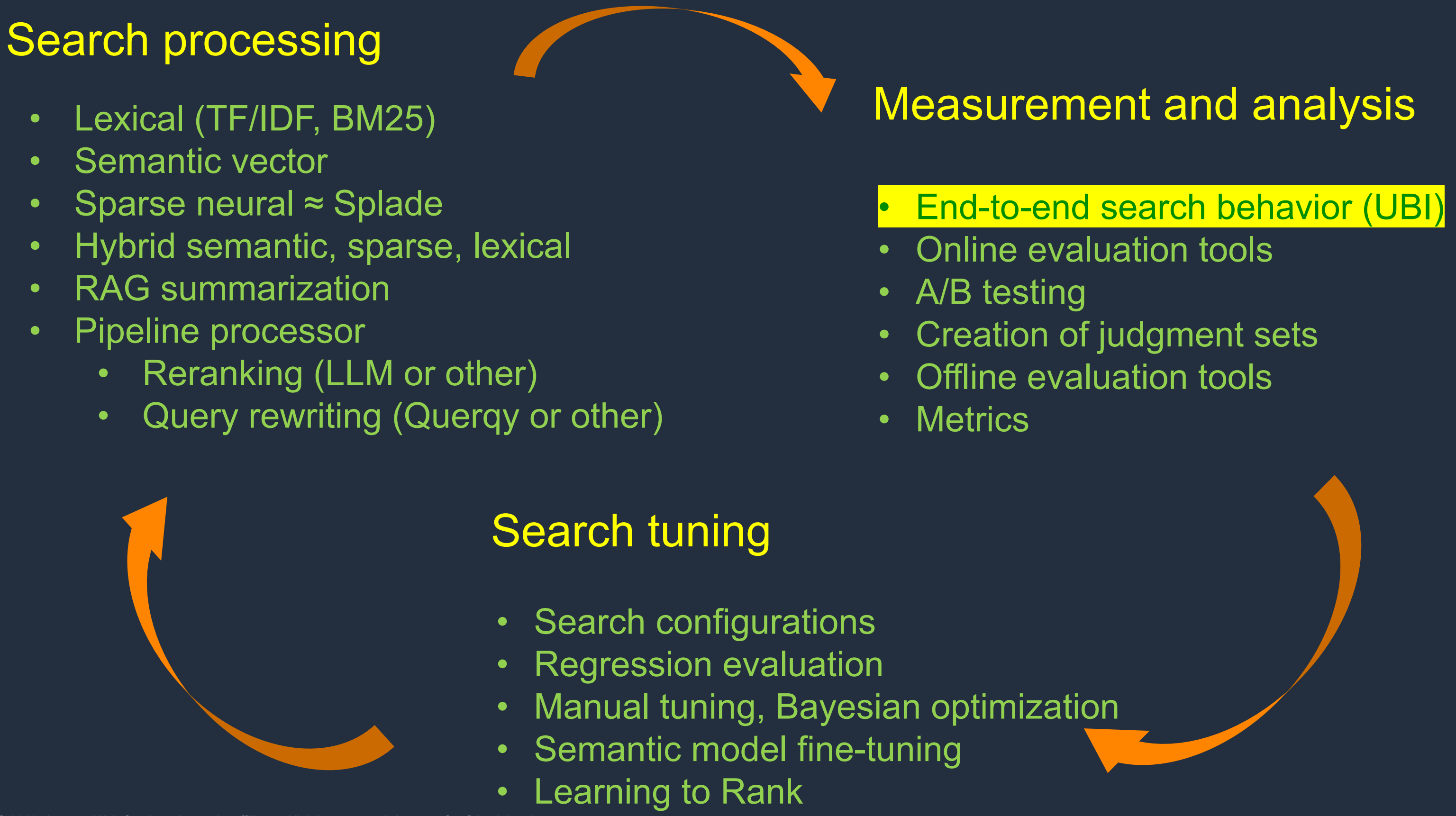
Measurement and analysis

- End-to-end search behavior (UBI)
- Online evaluation tools
- A/B testing
- Creation of judgment sets
- Offline evaluation tools
- Metrics

Search tuning

- Search configurations
- Regression evaluation
- Manual tuning, Bayesian optimization
- Semantic model fine-tuning
- Learning to Rank

The virtuous circle of search relevance



Search processing

- Lexical (TF/IDF, BM25)
- Semantic vector
- Sparse neural ≈ Splade
- Hybrid semantic, sparse, lexical
- RAG summarization
- Pipeline processor
  - Reranking (LLM or other)
  - Query rewriting (Querqy or other)

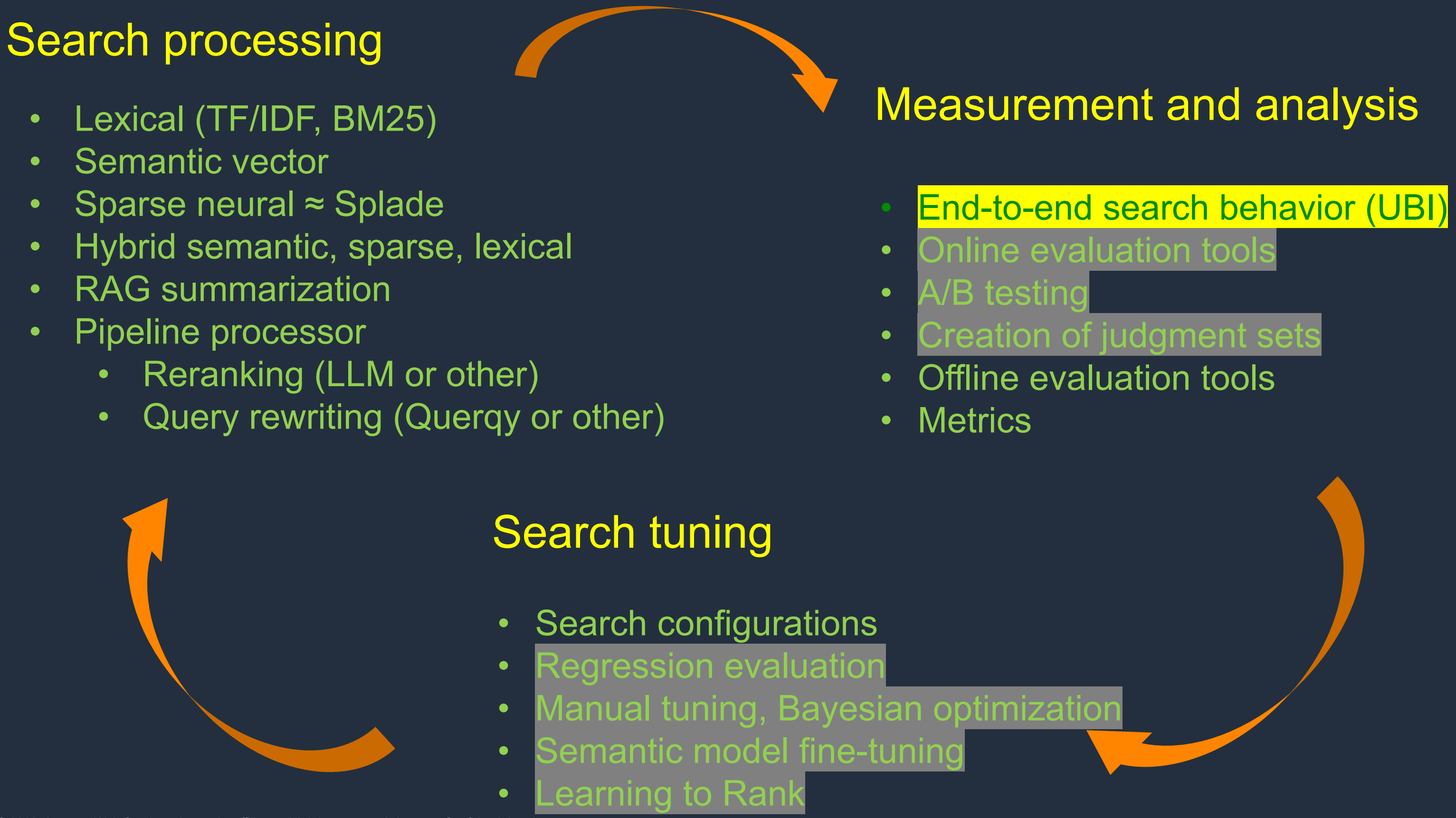
Measurement and analysis

- End-to-end search behavior (UBI)
- Online evaluation tools
- A/B testing
- Creation of judgment sets
- Offline evaluation tools
- Metrics

Search tuning

- Search configurations
- Regression evaluation
- Manual tuning, Bayesian optimization
- Semantic model fine-tuning
- Learning to Rank

The virtuous circle of search relevance



Search processing

- Lexical (TF/IDF, BM25)
- Semantic vector
- Sparse neural ≈ Splade
- Hybrid semantic, sparse, lexical
- RAG summarization
- Pipeline processor
  - Reranking (LLM or other)
  - Query rewriting (Querqy or other)

Measurement and analysis

- End-to-end search behavior (UBI)
- Online evaluation tools
- A/B testing
- Creation of judgment sets
- Offline evaluation tools
- Metrics

Search tuning

- Search configurations
- Regression evaluation
- Manual tuning, Bayesian optimization
- Semantic model fine-tuning
- Learning to Rank

How to choose a retrieval technique?

- Lexical search
- Semantic vector search
- Customization, personalization
- Neural sparse retrieval (similar to SPLADE)
- Image
- Hybrid search: lexical, semantic, multimodal

Priorities

1. Standardizing the schema
2. Collecting end-to-end search behavior events
3. Analyzing and displaying

Principles

- Full data available for analysis
- All data under customer control
- Extensible
- Search engine and data store agnostic
- Collection of PII under customer control
- Secure
- Near real-time

Search tuning with data

- Manual tuning
- Bayesian/grid parameter optimization
- Semantic model fine-tuning
- Learning to Rank
- Combining lexical and neural/vector rankings

Applications

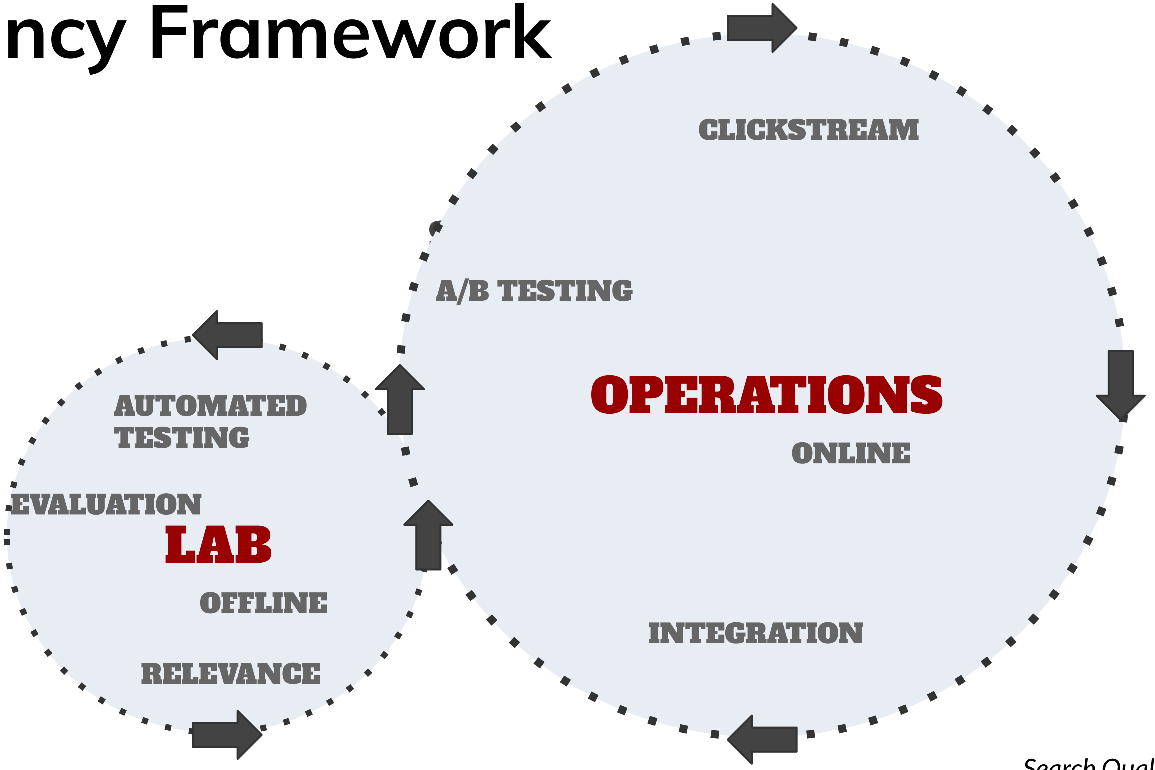
- Analysis of customer journeys
- Dashboarding of KPIs, anomaly detection
- Investigation by relevance engineers, data scientists, and merchandisers
- Collecting judgment data for offline evaluation
- Debugging
- Tuning of typeahead
- Real-time feedback to relevance calculation
- Input to machine learning models
  (Bayesian optimization, Personalize, Learning to Rank, …)

The User Behavior Insights project

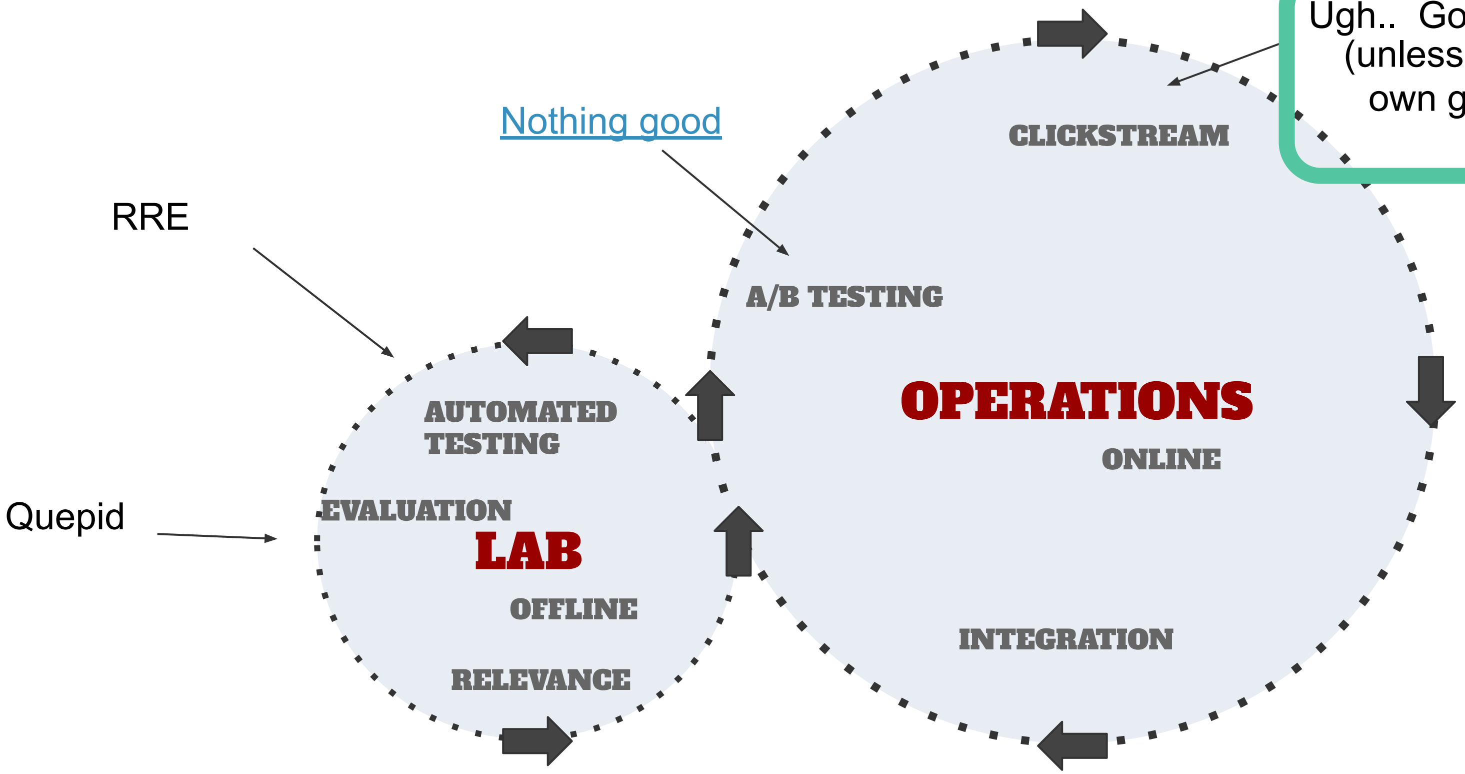
- Discussion at Haystack/MICES 2022
- OpenSearch RFC 4619 (9/2022); RFC 12084 (1/2024, updated 4/2024)
- Community debated names
- Frequent presentations and discussion at OpenSearch community meetings
- All work done in the open on github
- Apache 2.0 license

Why are we so excited about UBI?

Relevancy Framework



Search Quality: A Business-Friendly Perspective
Peter Fries - [Haystack 2018]



Ugh.. Google Analytics? (unless you have your own giant analytics team!)

State of play

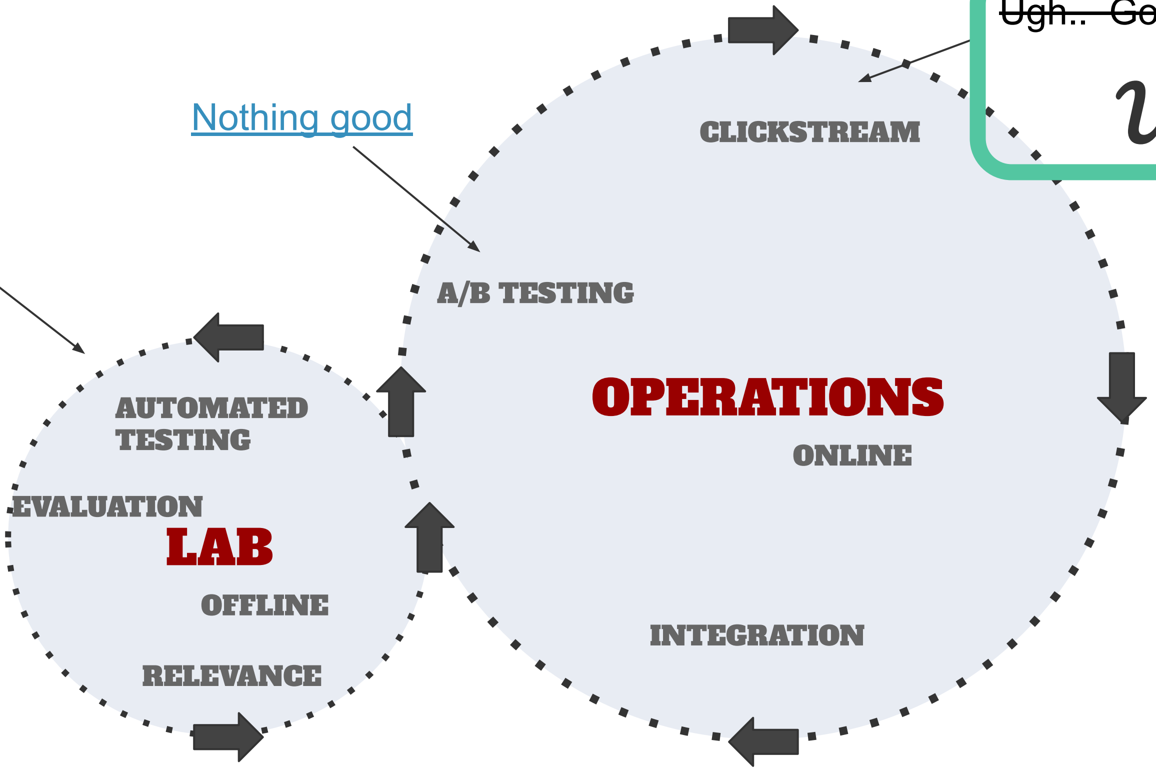
~~Ugh.. Google Analytics?~~

UBI!

Nothing good

RRE

Quepid



State of play

What makes up UBI?

1. A shared schema for
   a. defining what the user is asking for (Query)
   b. Defining what documents/objects/records/answers the user is receiving back.
   c. Defining how we track and organize the resulting Events and make the causal connections between the Query and the follow on actions (click through, add to cart, etc)

What makes up UBI?

1. A shared schema for
   a. defining what the user is asking for (Query)
   b. Defining what documents/objects/records/answers the user is receiving back.
   c. Defining how we track and organize the resulting Events and make the causal connections between the Query and the follow on actions (click through, add to cart, etc)

2. Code for collecting specific search related events by users (often JavaScript)

What makes up UBI?

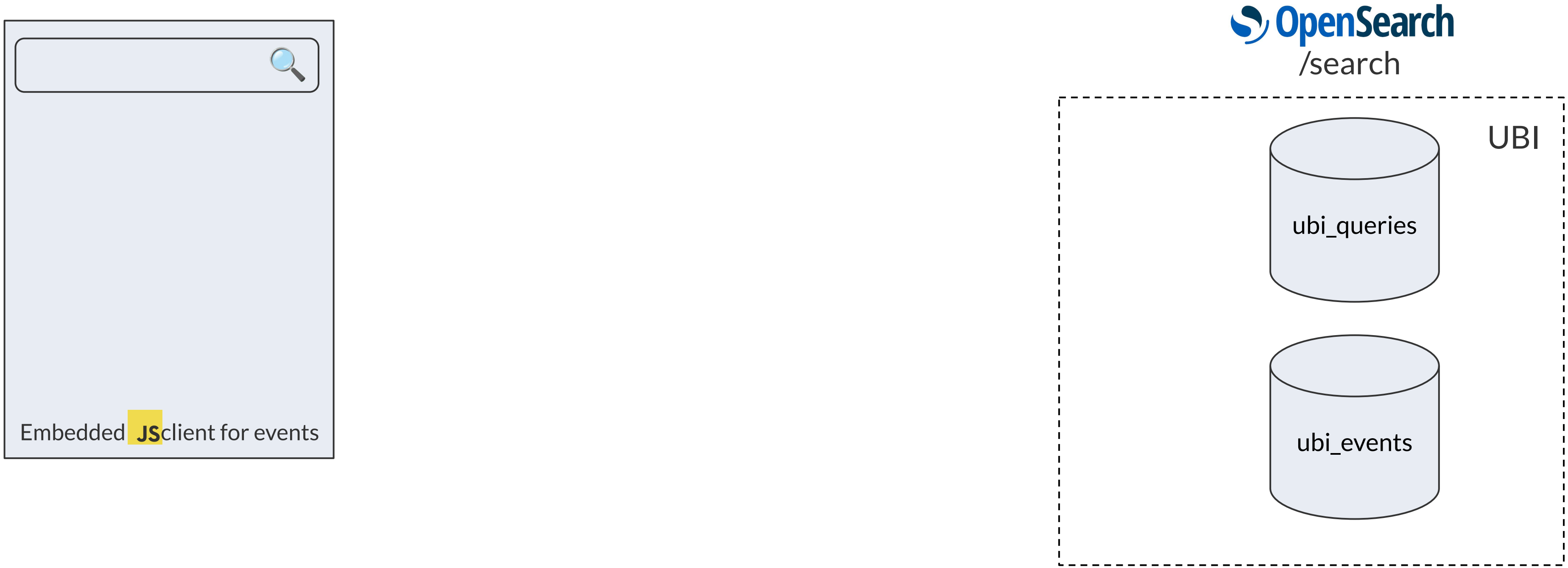
1. A shared schema for
   a. defining what the user is asking for (Query)
   b. Defining what documents/objects/records/answers the user is receiving back.
   c. Defining how we track and organize the resulting Events and make the causal connections between the Query and the follow on actions (click through, add to cart, etc)

2. Code for collecting specific search related events by users (often JavaScript)

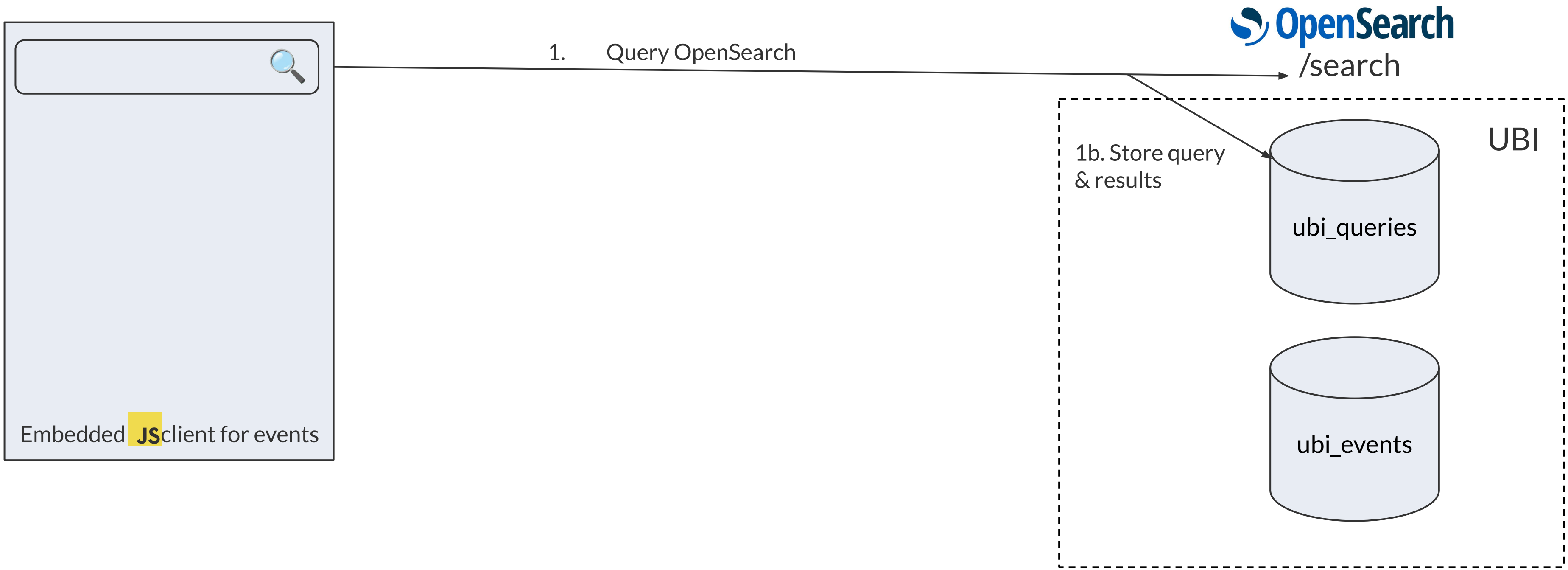
3. An implementation in OpenSearch (and Solr!) to facilitate this process

What makes up UBI?

1. A shared schema for
   a. defining what the user is asking for (Query)
   b. Defining what documents/objects/records/answers the user is receiving back.
   c. Defining how we track and organize the resulting Events and make the causal connections between the Query and the follow on actions (click through, add to cart, etc)
2. Code for collecting specific search related events by users (often JavaScript)
3. An implementation in OpenSearch (and Solr!) to facilitate this process



UBI data flow



1. Query OpenSearch

OpenSearch /search

1b. Store query & results

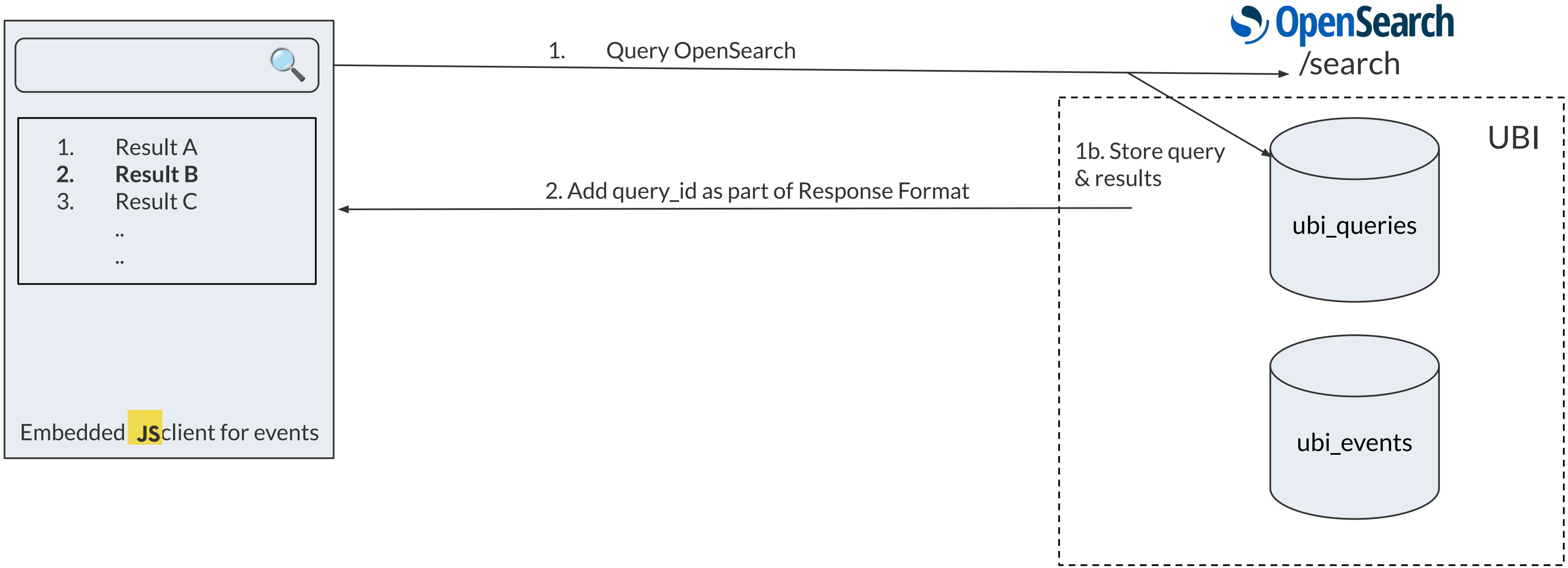
UBI

ubi_queries

ubi_events

Embedded JS client for events

UBI data flow

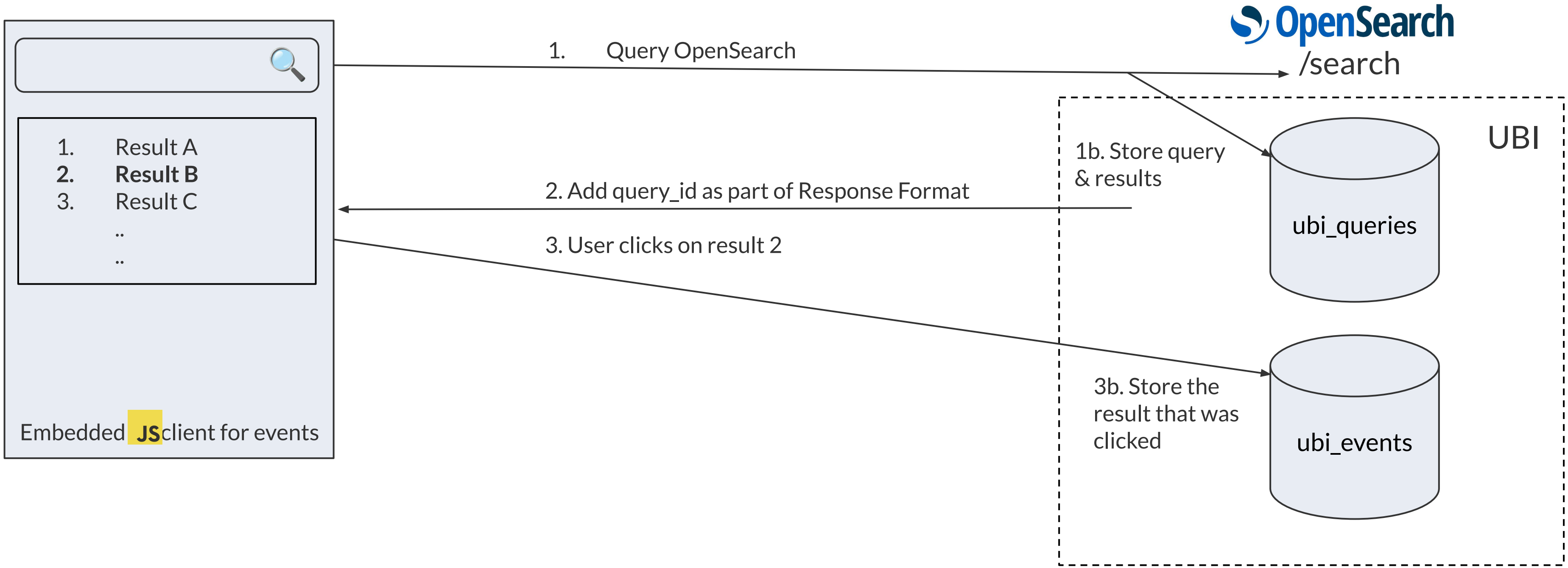


1. Query OpenSearch

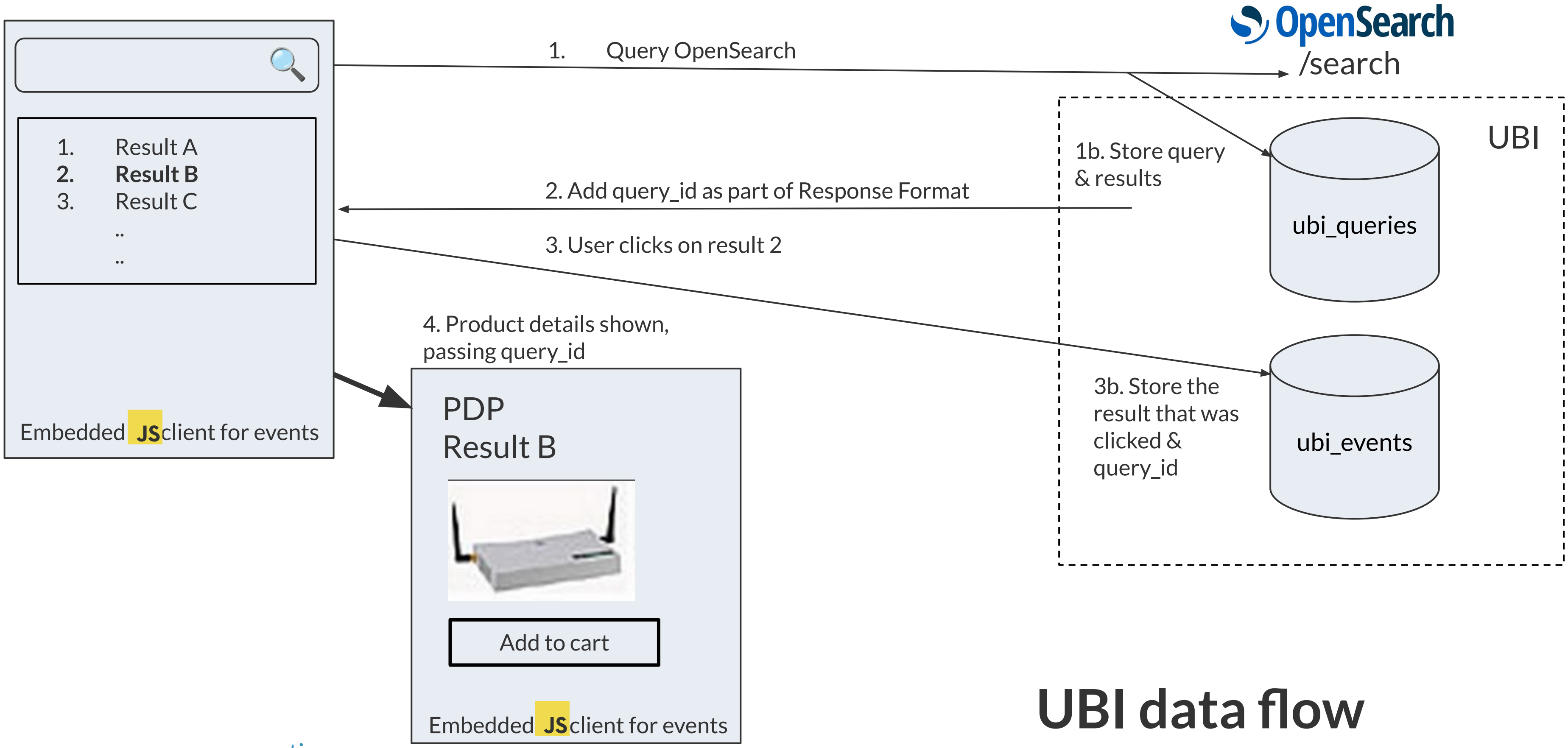
2. Add query_id as part of Response Format

1b. Store query & results

UBI data flow


OpenSearch
/search
1. Query OpenSearch
UBI
1b. Store query & results
ubi_queries
2. Add query_id as part of Response Format
3. User clicks on result 2
1. Result A
2. Result B
3. Result C
..
..
3b. Store the result that was clicked
ubi_events
Embedded JS client for events

UBI data flow



OpenSearch

/search

1. Query OpenSearch

1b. Store query & results

2. Add query_id as part of Response Format

3. User clicks on result 2

3b. Store the result that was clicked & query_id

4. Product details shown, passing query_id

UBI

ubi_queries

ubi_events

1. Result A
2. **Result B**
3. Result C

..

..

Embedded JS client for events

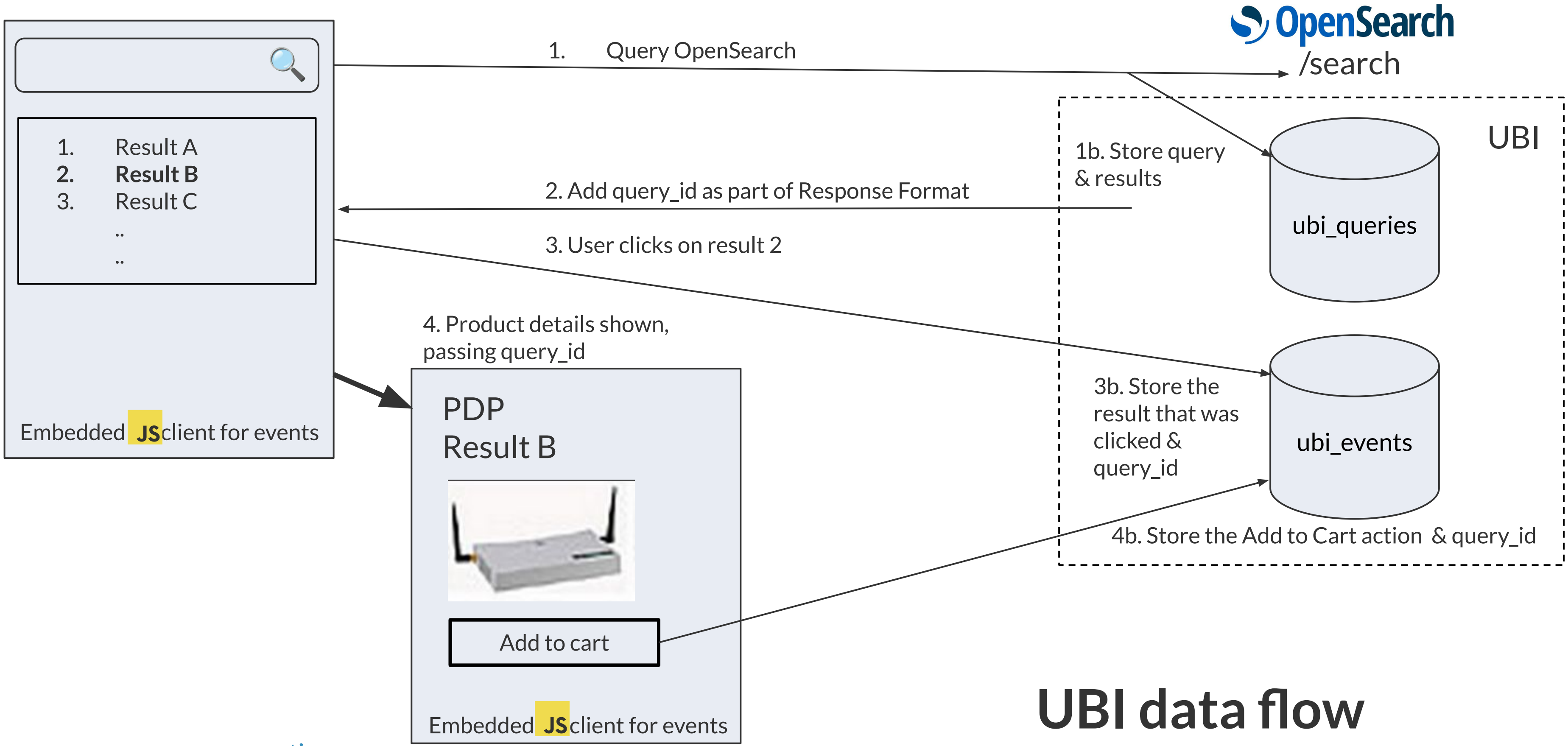
PDP

Result B

Add to cart

Embedded JS client for events

UBI data flow



OpenSearch

/search

1. Query OpenSearch

1. Result A
2. **Result B**
3. Result C

..

..

Embedded JS client for events

1b. Store query & results

UBI

ubi_queries

2. Add query_id as part of Response Format

3. User clicks on result 2

4. Product details shown, passing query_id

PDP

Result B

Add to cart

Embedded JS client for events

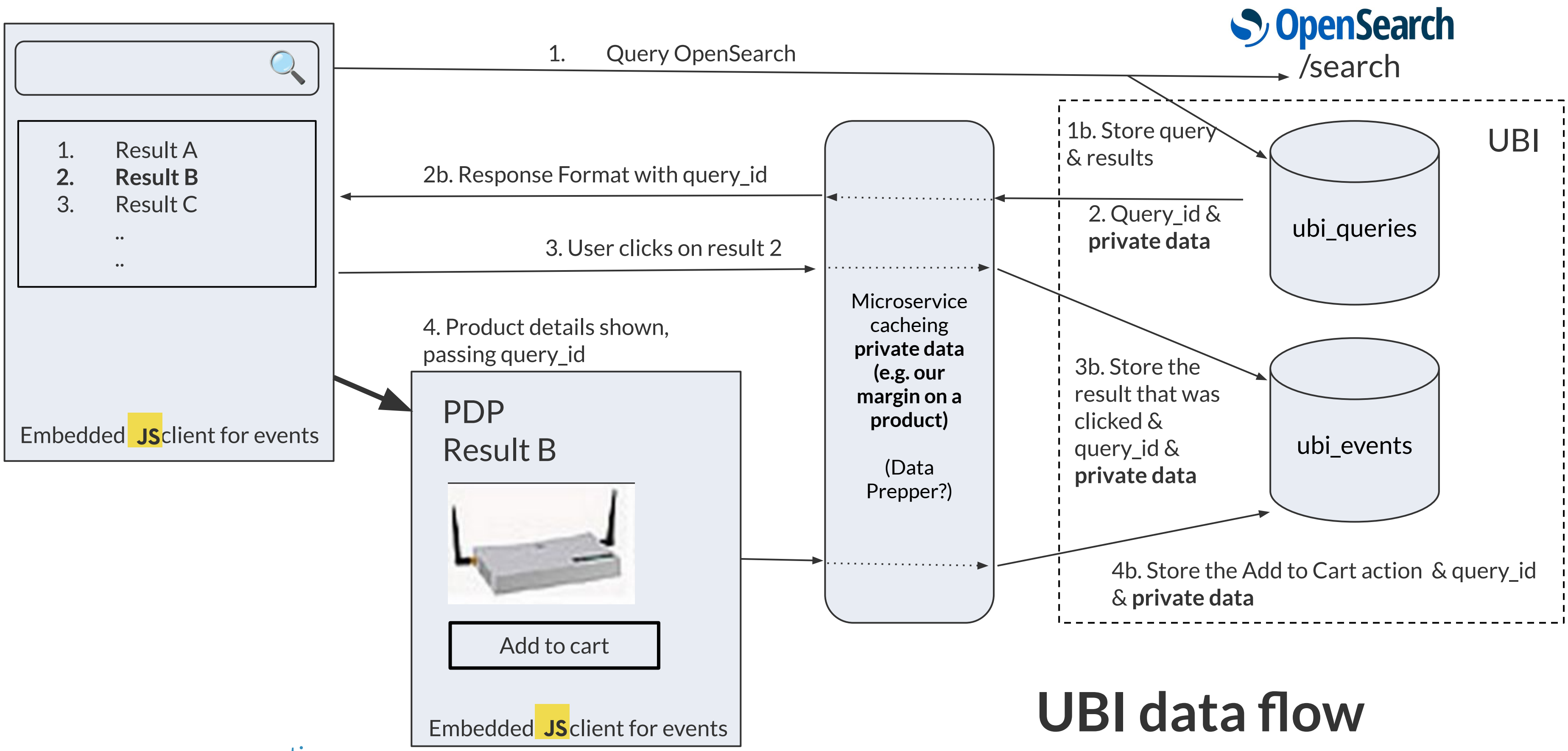
3b. Store the result that was clicked & query_id

ubi_events

4b. Store the Add to Cart action & query_id

UBI data flow

UBI data flow



OpenSearch /search

1. Query OpenSearch

1b. Store query & results

2. Query_id & **private data**

2b. Response Format with query_id

3. User clicks on result 2

3b. Store the result that was clicked & query_id & **private data**

4. Product details shown, passing query_id

4b. Store the Add to Cart action & query_id & **private data**

1. Result A
2. **Result B**
3. Result C

..

..

Embedded JS client for events

PDP

Result B

Add to cart

Embedded JS client for events

Microservice cacheing **private data (e.g. our margin on a product)**

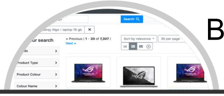
(Data Prepper?)

UBI

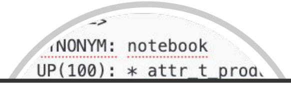
ubi_queries

ubi_events

Solr Elasticsearch



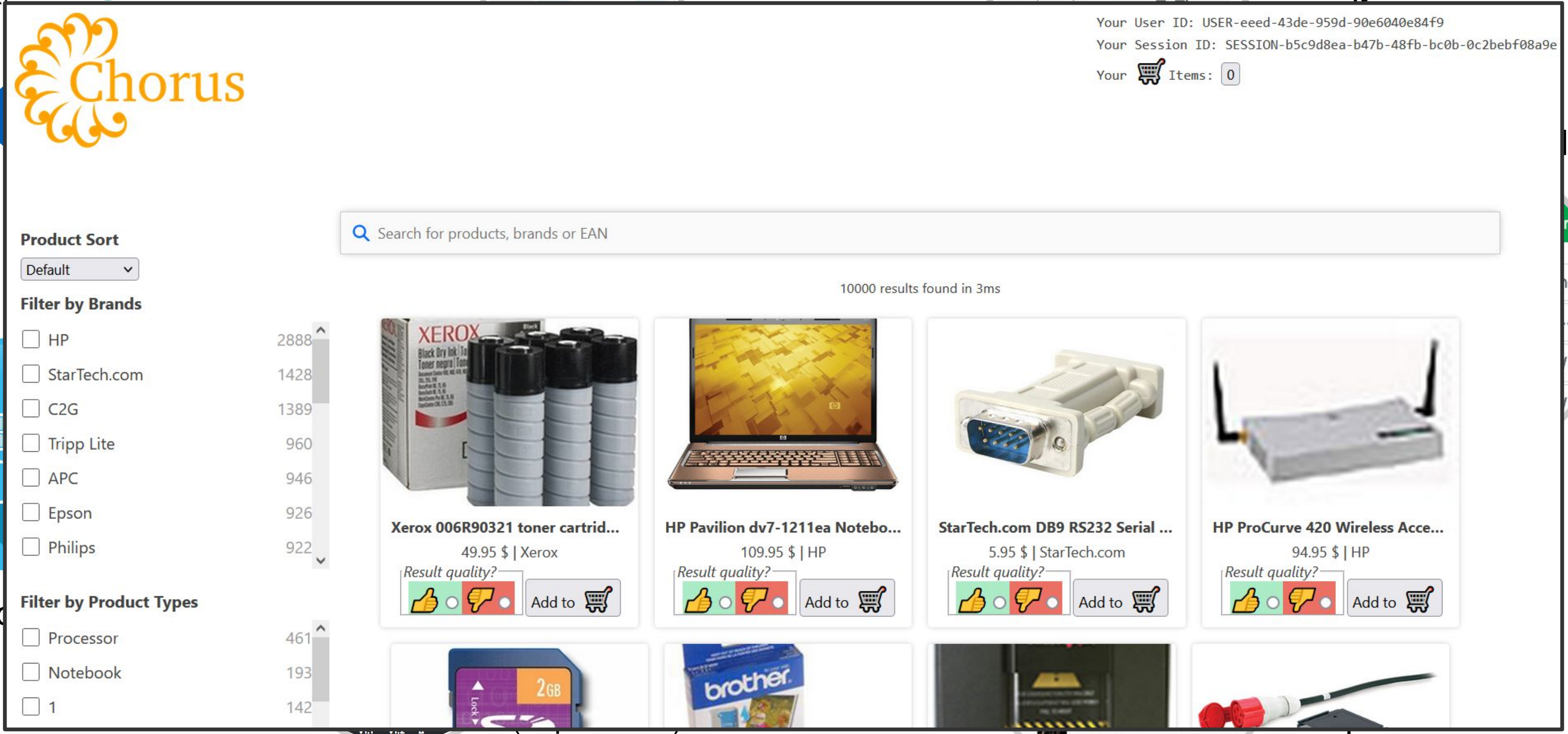
Blacklight
(Solr)



Querqy

UI

Doc



Chorus

Your User ID: USER-eeed-43de-959d-90e6040e84f9
Your Session ID: SESSION-b5c9d8ea-b47b-48fb-bc0b-0c2bebf08a9e
Your 🛒 Items: 0

Product Sort

Default

Filter by Brands

□ HP	2888
□ StarTech.com	1428
□ C2G	1389
□ Tripp Lite	960
□ APC	946
□ Epson	926
□ Philips	922

Filter by Product Types

□ Processor	461
□ Notebook	193
□ 1	142

Search for products, brands or EAN

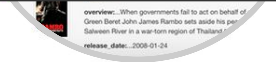
10000 results found in 3ms

Xerox 006R90321 toner cartrid...
49.95 $ | Xerox
Result quality? 👍 ○ 👎 ○ Add to 🛒

HP Pavilion dv7-1211ea Notebo...
109.95 $ | HP
Result quality? 👍 ○ 👎 ○ Add to 🛒

StarTech.com DB9 RS232 Serial ...
5.95 $ | StarTech.com
Result quality? 👍 ○ 👎 ○ Add to 🛒

HP ProCurve 420 Wireless Acce...
94.95 $ | HP
Result quality? 👍 ○ 👎 ○ Add to 🛒


overview: ...When governments fail to act on behalf of
Green Beret John James Rambo sets aside his pe
Salween River in a war-torn region of Thailand
release_date: 2008-01-24

Let’s watch it working on Chorus Electronics...

Enabling UBI logging per query

```
GET _search
 {
  "ext":
    "ubi": {
      "user_query":  "the wind rises",
    }
   },
   "query": {
     "query_string": {
       "query": "the wind AND (rises OR rising)"
     }
   }
 }
```

We track what the user types
AND the search engine query

Passing in our own explicit query_id

```
GET _search
 {
  "ext":
    "ubi": {
      "query_id": "USER52-SESSION2-QUERY3",
      "user_query": {
          "original_query":"the wind rises",
          "aggregates": ....
       }
     }
   },
   "query": {
     "query_string": {
       "query": "the wind AND (rises OR rising)"
     }
   }
 }
```

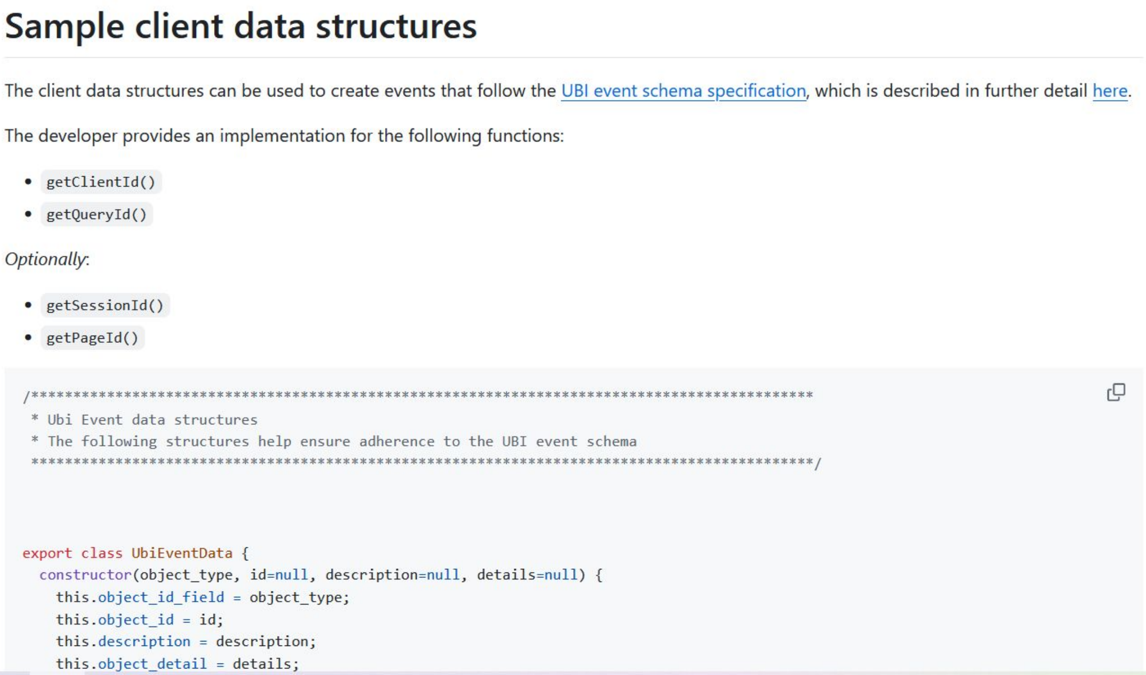
If you don’t pass in a query_id then UBI will create one for you. In both cases we return it in the query response

Event logging

Events logging is just a standard OS document! You can thus scale UBI just as you would anything else in OpenSearch

```
POST /ubi_events/doc
 {
  "query_id": "USER52-SESSION2-QUERY3" ,
  "application": "chorus",
  "action_name": "add_to_cart",
  "user_id": "1",
  "event_attributes": {
   "object": {
     "object_id": "RESULT_B",
     ....
     }
   "position": {...},
   "data": {...},
   }
 }
```

Docs



Sample client data structures

The client data structures can be used to create events that follow the UBI event schema specification, which is described in further detail here.

The developer provides an implementation for the following functions:

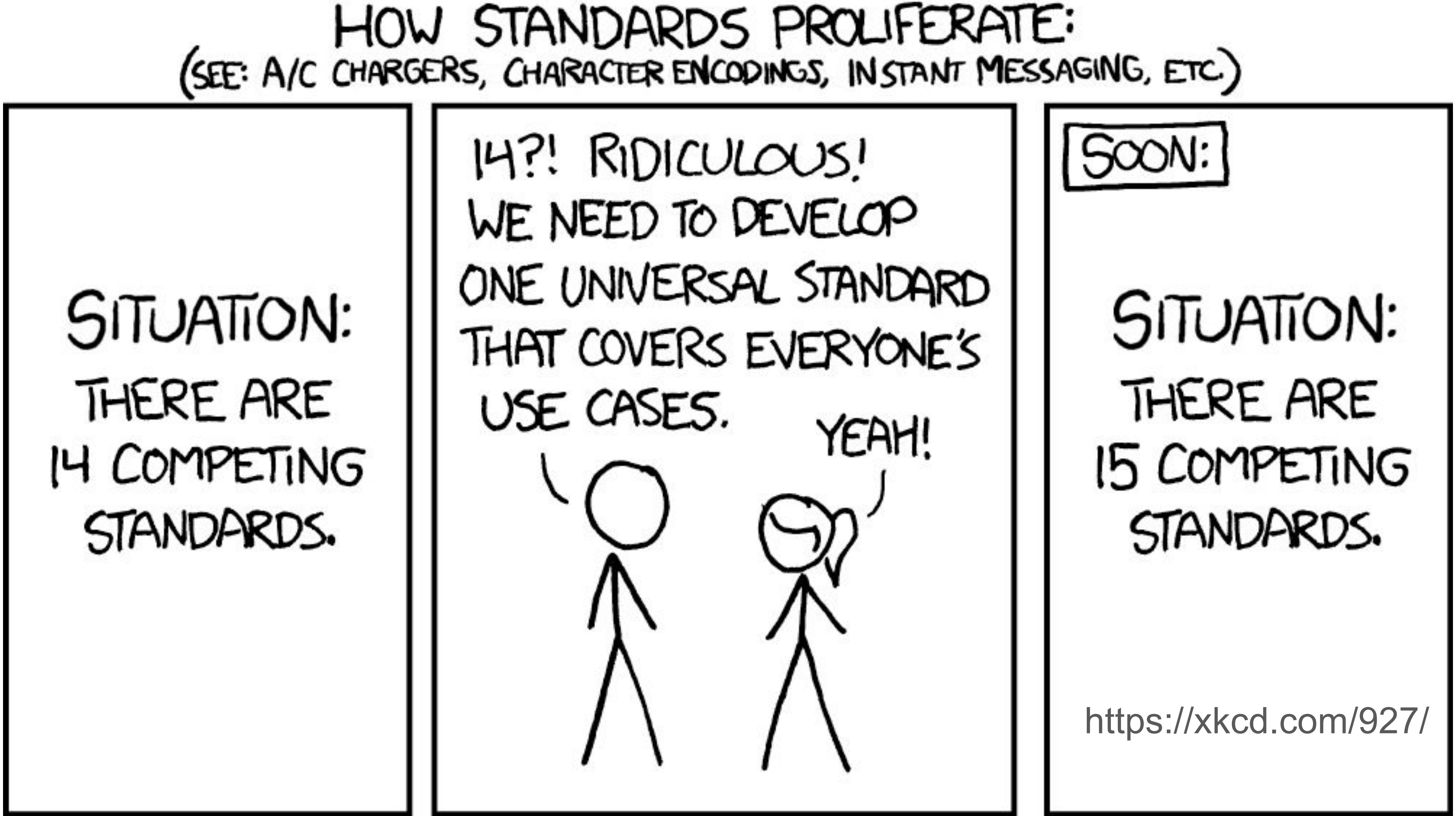
- `getClientId()`
- `getQueryId()`

Optionally:

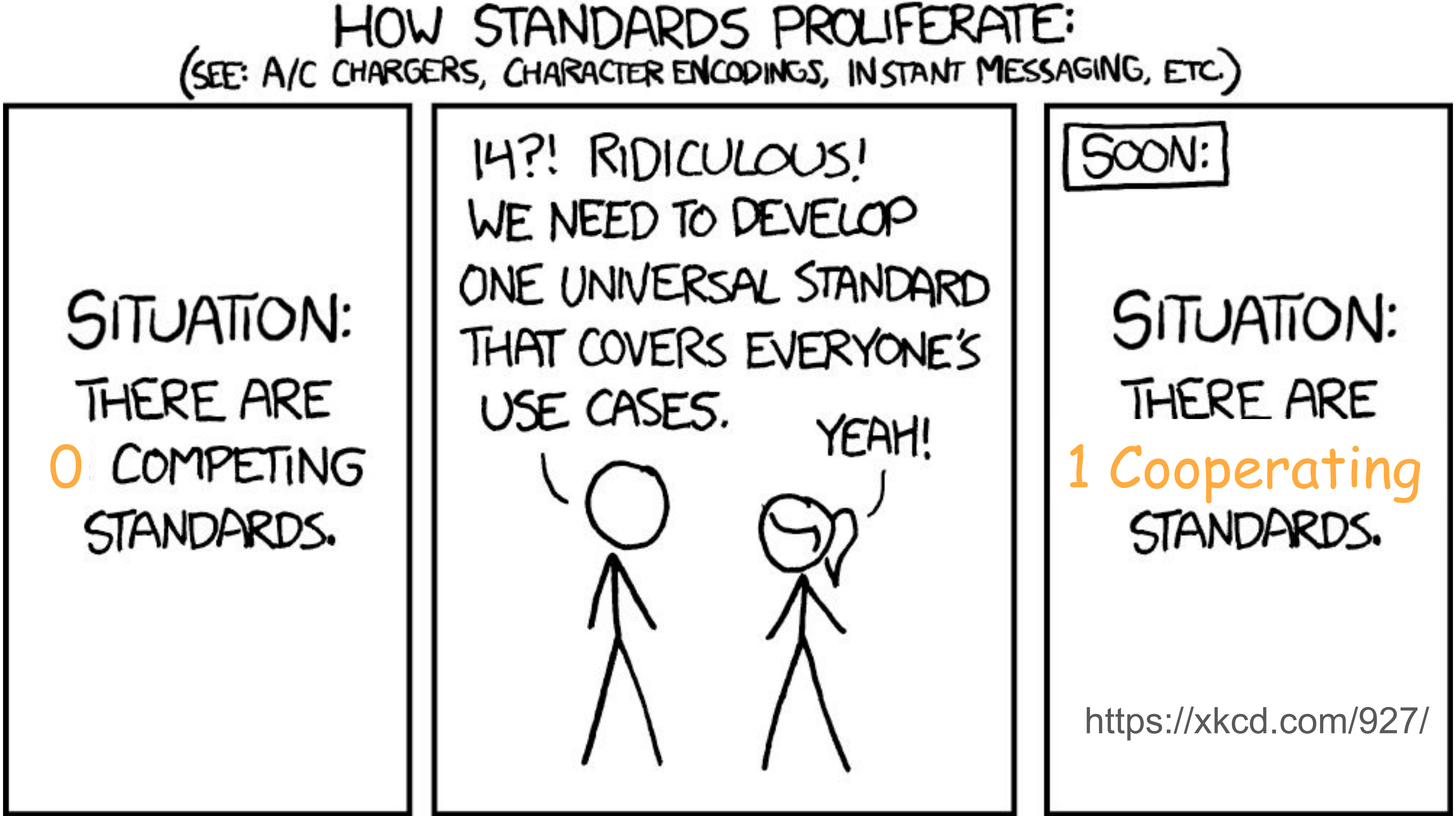
- `getSessionId()`
- `getPageId()`

```
/*********************************************************************************************
 * Ubi Event data structures
 * The following structures help ensure adherence to the UBI event schema
 *********************************************************************************************/



export class UbiEventData {
  constructor(object_type, id=null, description=null, details=null) {
    this.object_id_field = object_type;
    this.object_id = id;
    this.description = description;
    this.object_detail = details;
```

Now we know what you're thinking about a 'standard' schema....


HOW STANDARDS PROLIFERATE:
(SEE: A/C CHARGERS, CHARACTER ENCODINGS, INSTANT MESSAGING, ETC.)
SITUATION: THERE ARE 14 COMPETING STANDARDS.
14?! RIDICULOUS! WE NEED TO DEVELOP ONE UNIVERSAL STANDARD THAT COVERS EVERYONE'S USE CASES.
YEAH!
SOON:
SITUATION: THERE ARE 15 COMPETING STANDARDS.

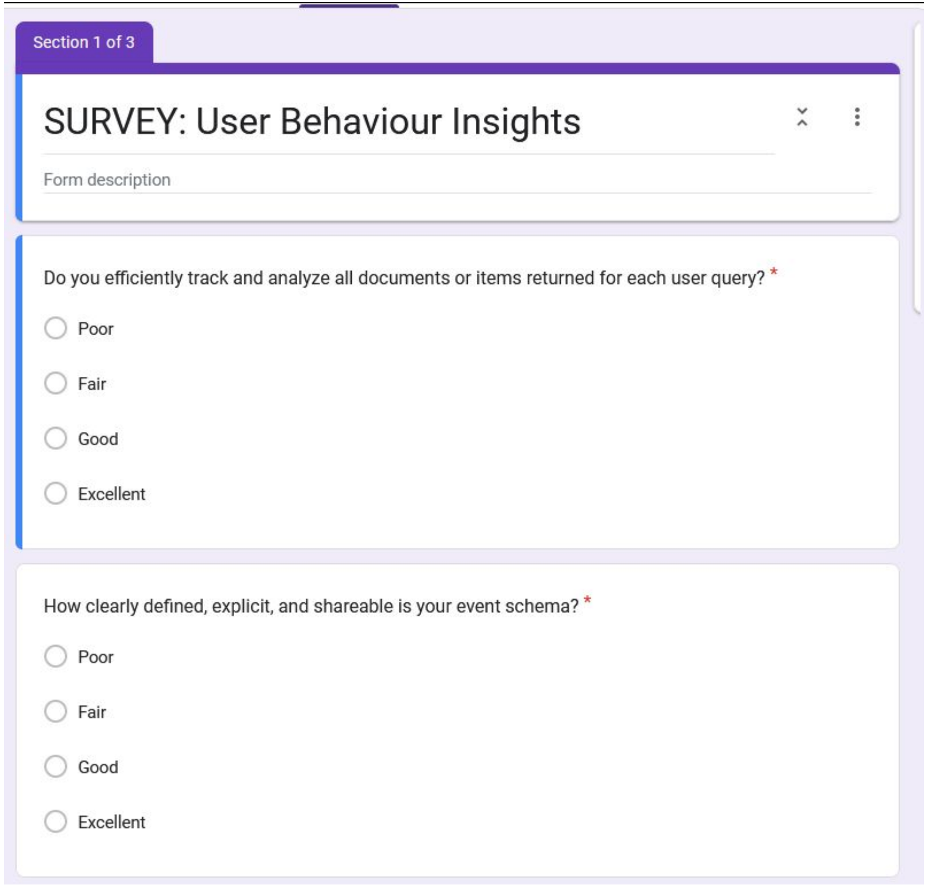
https://xkcd.com/927/


HOW STANDARDS PROLIFERATE:
(SEE: A/C CHARGERS, CHARACTER ENCODINGS, INSTANT MESSAGING, ETC.)
SITUATION: THERE ARE 0 COMPETING STANDARDS.
14?! RIDICULOUS! WE NEED TO DEVELOP ONE UNIVERSAL STANDARD THAT COVERS EVERYONE'S USE CASES.
YEAH!
SOON:
SITUATION: THERE ARE 1 Cooperating STANDARDS.

https://xkcd.com/927/

The schema, docs & code

- 1.0 release is now available
- We want this to be as widely adopted as possible in the community - every engine, every search! Can you help?
- https://github.com/opensearch-project/user-behavior-insights
  - **Live as of today!**
  - Builds for v2.14.0 and upcoming v2.15.0 of OpenSearch
- OpenSearch RFC for this work: https://github.com/opensearch-project/OpenSearch/issues/12084
- Original pull request for UBI:
- https://github.com/opensearch-project/OpenSearch/pull/13546
- Chorus w/ UBI: https://github.com/o19s/chorus-opensearch-edition/
- Online Demo: http://chorus-opensearch-edition.dev.o19s.com:4000/ and http://chorus-opensearch-edition.dev.o19s.com:5601



Section 1 of 3

SURVEY: User Behaviour Insights

Form description

Do you efficiently track and analyze all documents or items returned for each user query? *

○ Poor

○ Fair

○ Good

○ Excellent

How clearly defined, explicit, and shareable is your event schema? *

○ Poor

○ Fair

○ Good

○ Excellent

We need your input!
Just a few questions I promise!
https://forms.gle/7jdnLMJdxqY5j4qb8

Any questions?

Stavros Macrakis
macrakis@amazon.com

Eric Pugh (leading from OSC)
epugh@opensourceconnections.com

Github
OS 2.14, 2.15:
https://github.com/opensearch-project/user-behavior-insights