Architecture, Development Model and Future Trends of Web Search Engines

Marcelo De Barros

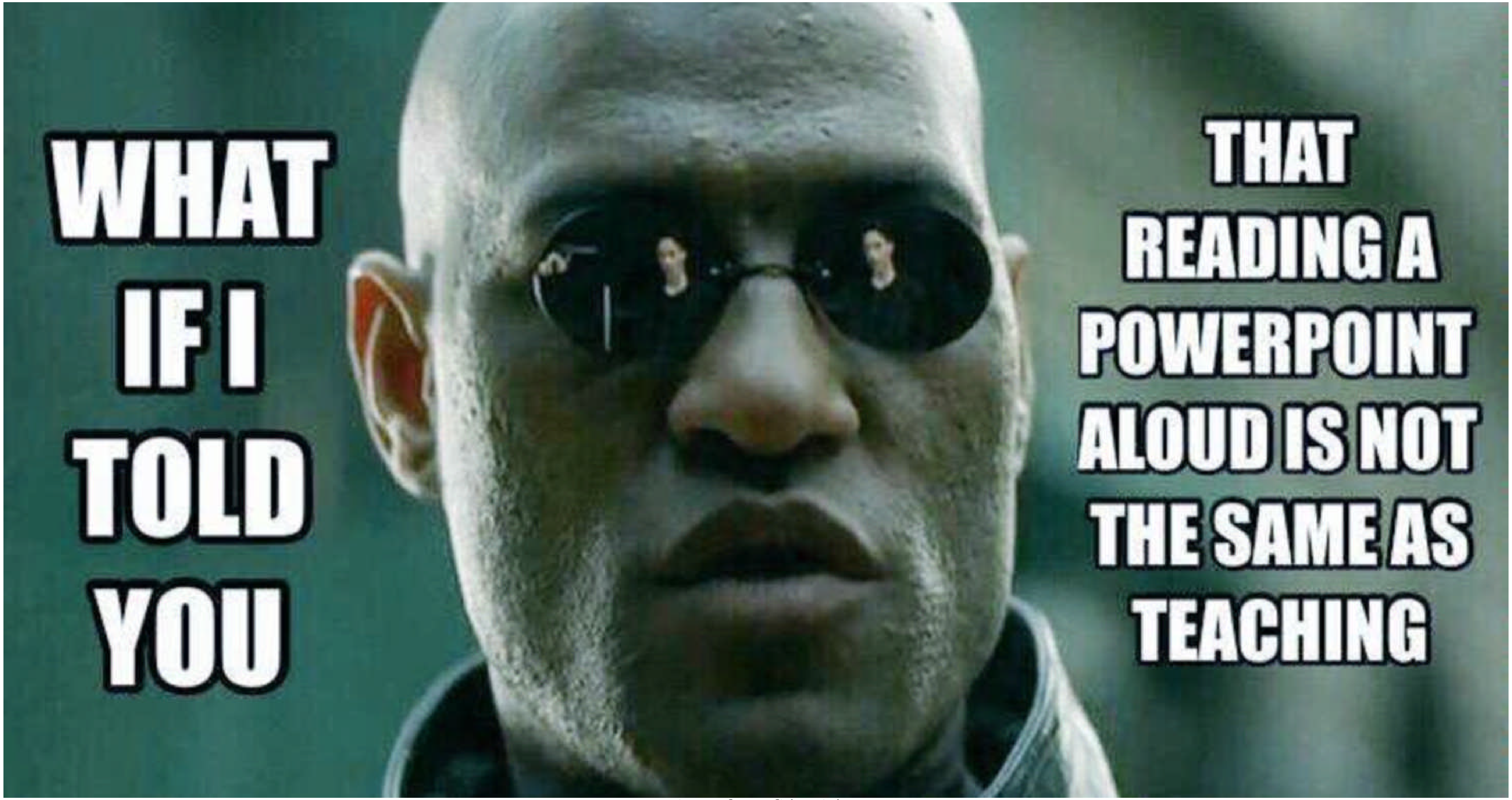
Bing UX Features and Shared Tools Team

Plan for the next hour

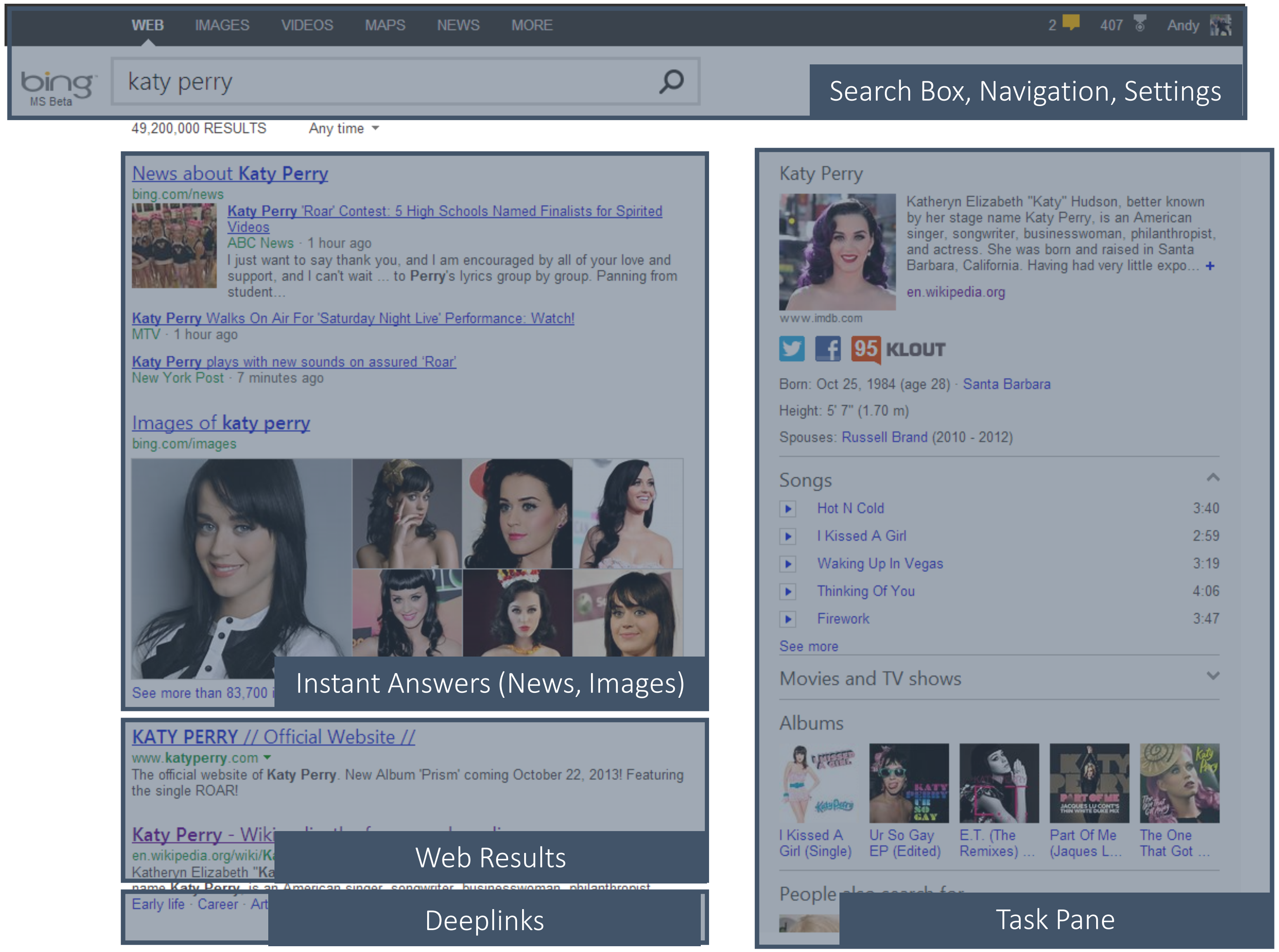
- I’ll present you a simplified view of Search Engines architecture.
- I’ll try **not** to use jargon without explaining it. Stop me if I forget.
- I’ll talk about the future trends around search engines (my own opinion).
- You ask questions if you have them.
- If I can’t answer, I’ll follow up with someone who can.

This deck represents an overview of Search Engines.

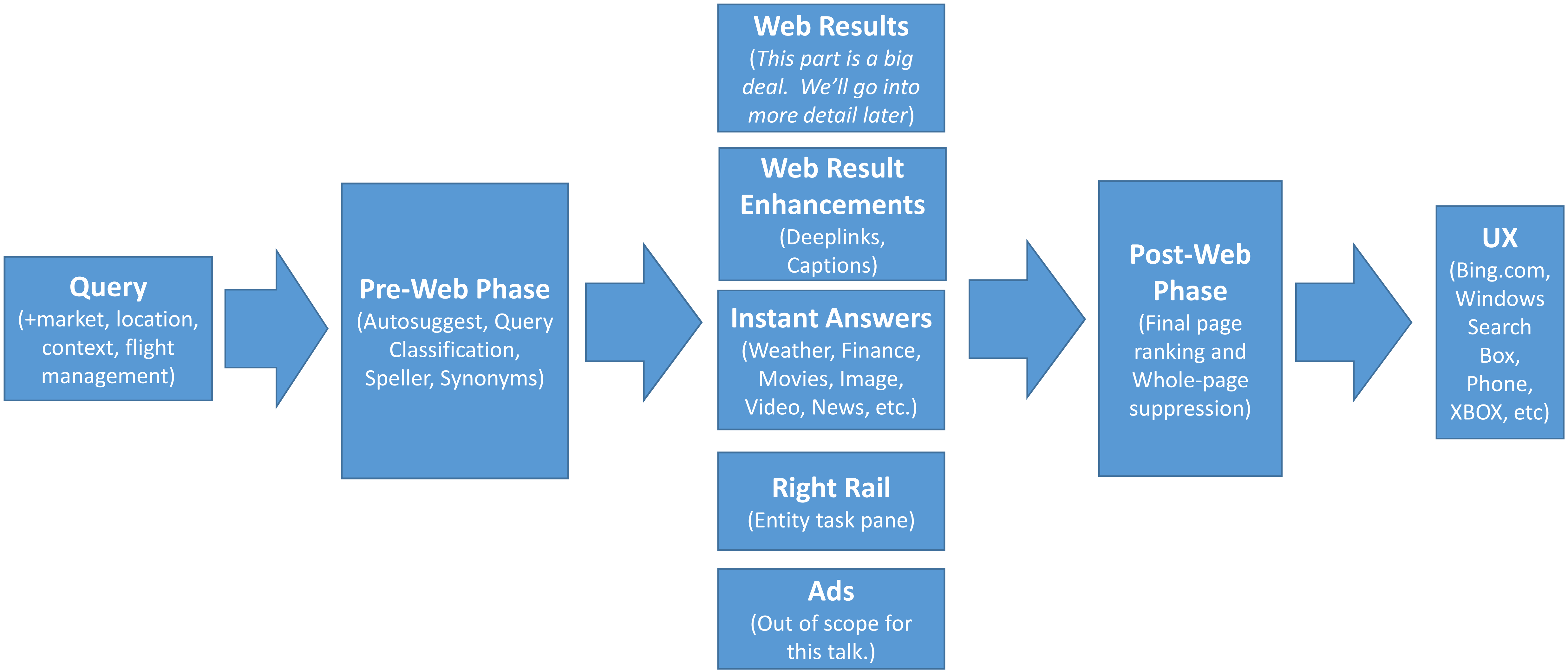
Some technical implementation details will be omitted on purpose ☺


WHAT
IF I
TOLD
YOU
THAT
READING A
POWERPOINT
ALOUD IS NOT
THE SAME AS
TEACHING

The Anatomy of a Bing SERP – Search Engine Results Page

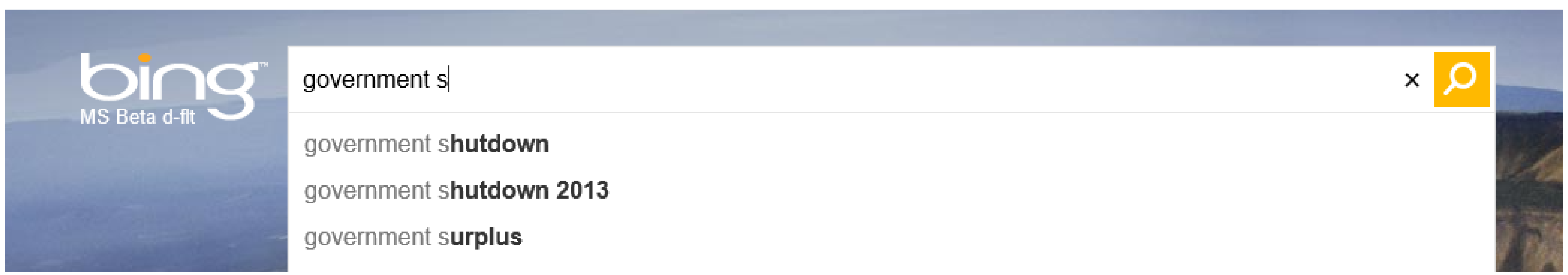


A Runtime Stack in One Slide



The User Query

- Things that a search engine might know even *before* we get to the web search:
  - Your query
  - Your entry point (Windows Search Box, Bing.com, Phone, XBOX, etc.)
  - Your market (country + language)
  - Your location (sometimes…)
  - Your past queries (sometimes…)
  - Your identity from logged-in experiences (sometimes)
  - Which flights (experiments) you are in
- Query Formulation via Autosuggest (traditional *trie* data structure)



Understanding Popular Pages

- Search engines know a number of statistics about the pages:

KATY PERRY // Official Website // (1.25%)
www.katyperry.com Translate this page (0%)
The official website of **Katy Perry**. New Album 'Prism' coming October 22, 2013!
Featuring the single ROAR!

Videos; (0.38%)	Movie (0%)
Tours; (0.2%)	One of The Boys (0%)
Discography; (0.01%)	North America (0%)
Bio; (0.1%)	Tour-Diary (0%)
News; (0.01%)	

See more results (0.02%)

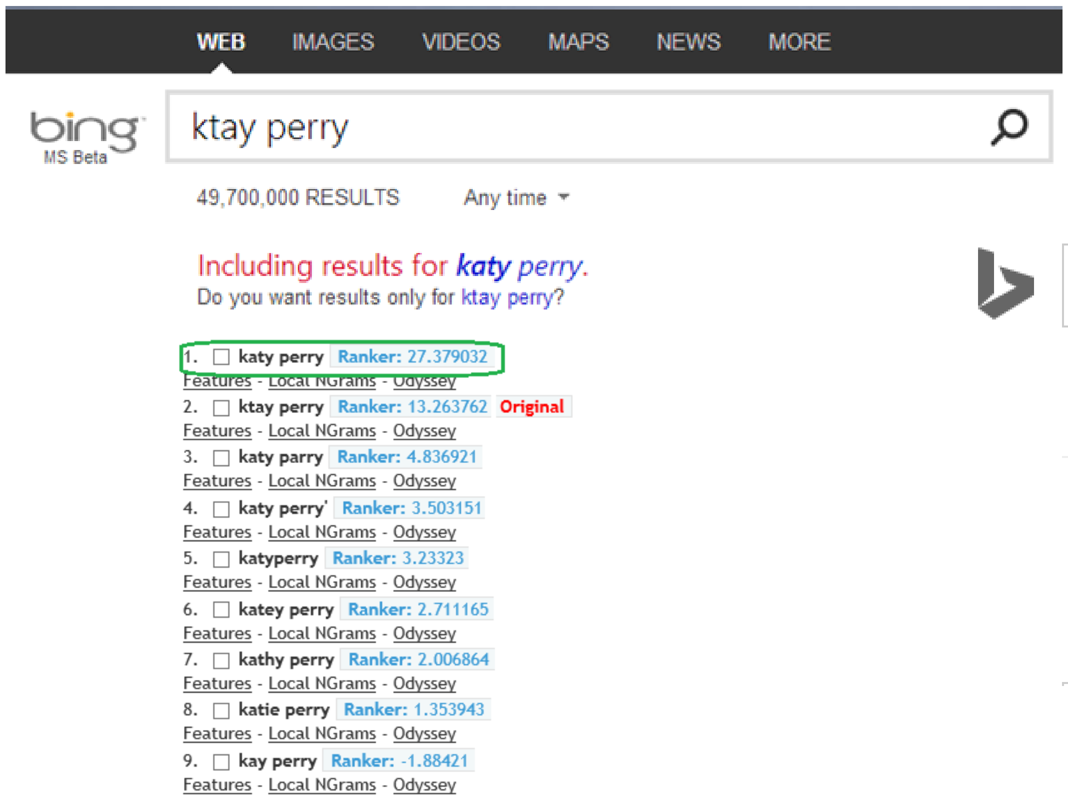
Katy Perry - Wikipedia, the free encyclopedia (1.64%)
en.wikipedia.org/wiki/Katy_Perry Translate this page (0%)

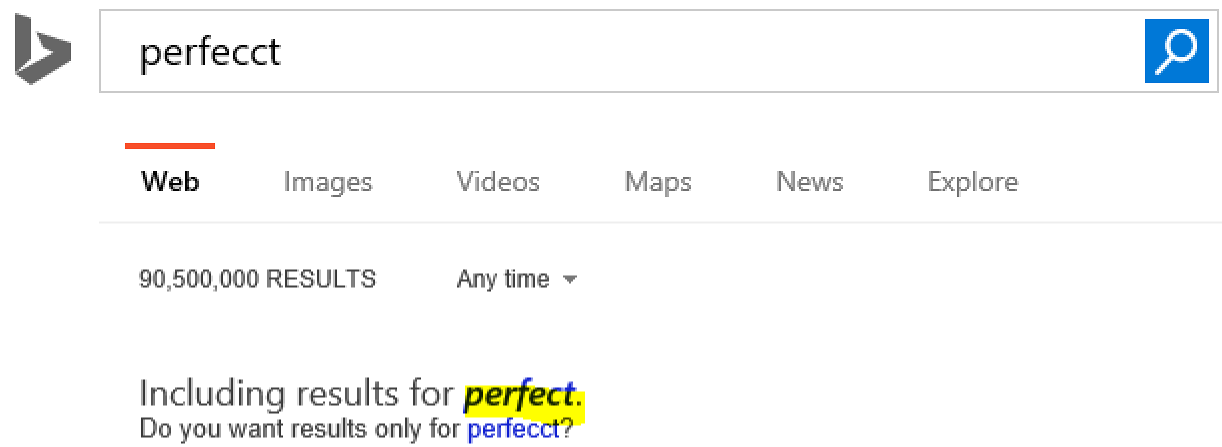
Sample Data
(not real numbers)

- Such information helps in ranking decisions, as well as caching decisions and placement decisions

Query Rewriting: Spelling and Synonyms (*pre-web*)

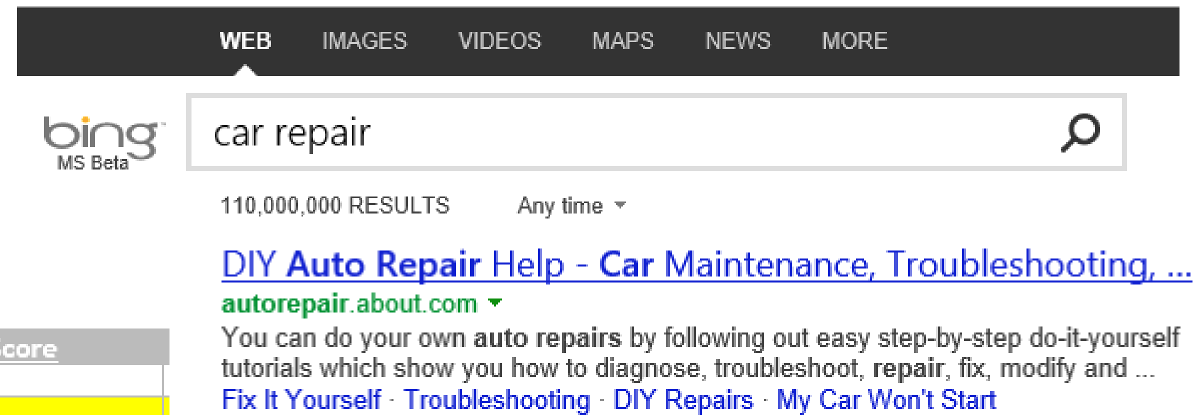
- Spelling:
  - Dictionary (per language)
  - Logs (words proximity, clustering techniques, ranking within clusters)





Query Rewriting: Spelling and Synonyms (*pre-web*)

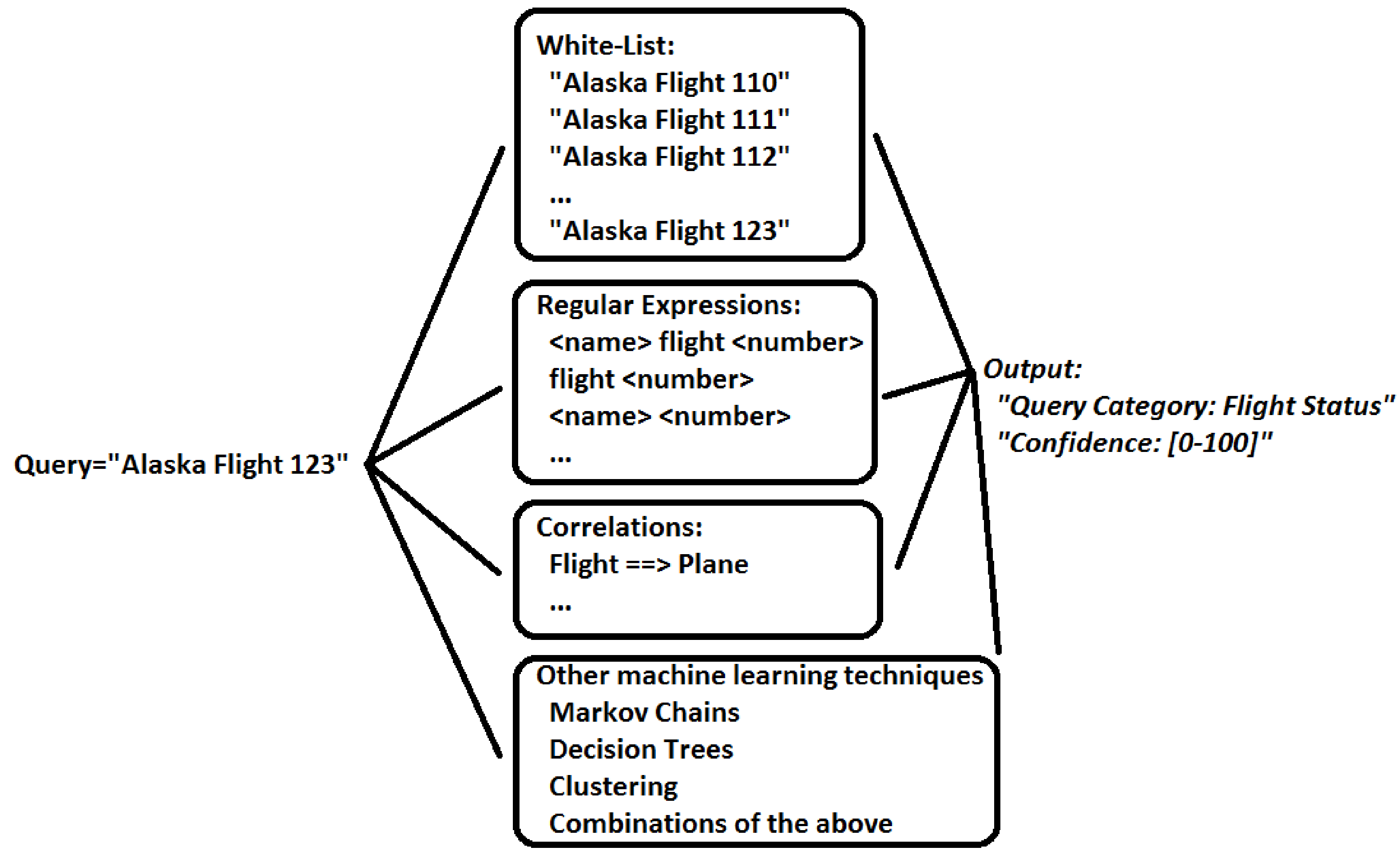
- Synonyms:
  - Clustering techniques



Term	Candidate	Tier1 Score
[-] car		
	car	N/A
	auto	999966
	automobile	999885
	automotive	999431
	carr	996370
	cars	999991
	motor vehicle	918356
	truck	999848
	van	998102
	vehicle	999862
	vehicles	999752
[-] repair		
	repair	N/A
	fix	999851
	fixing	1000000
	maintenance	999855
	owners	999849

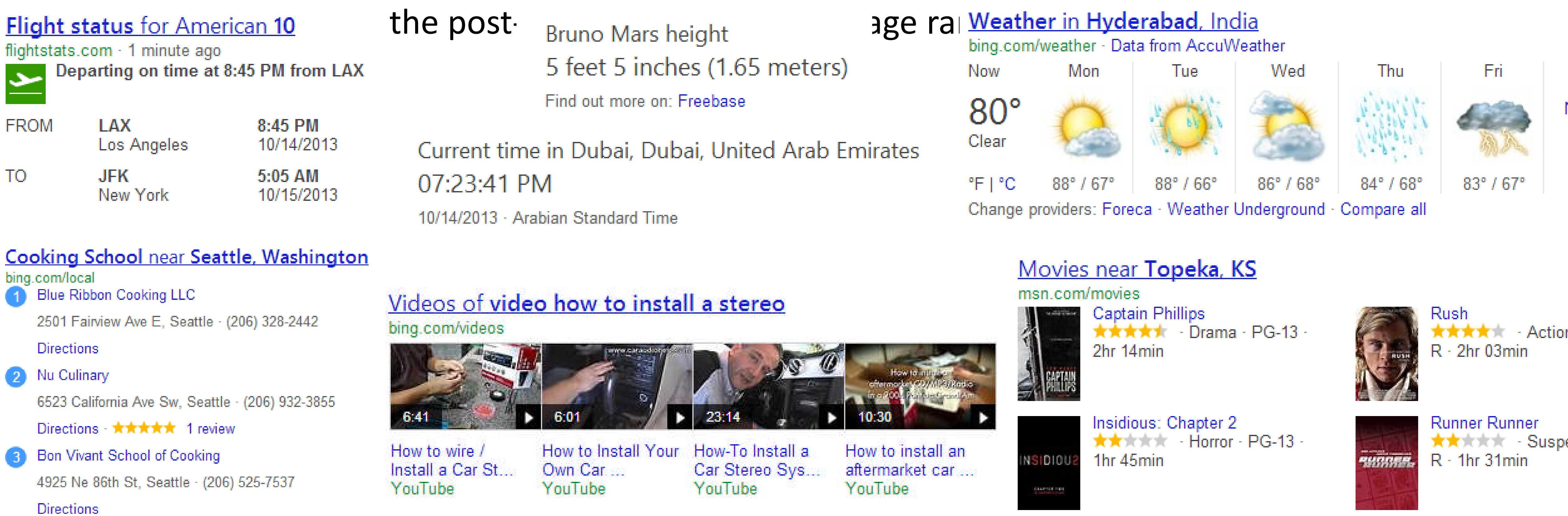
Query Rewriting: Query Classification (*pre-web*)

- Query Classification:
  - Fast Classifiers
  - White-List
  - Regular Expressions
  - Correlations
  - Other techniques



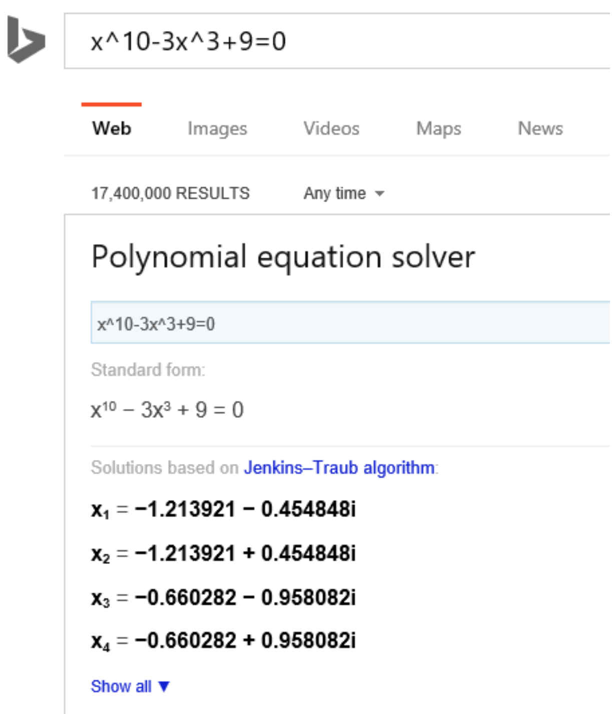
Instant Answers: a Federated Model

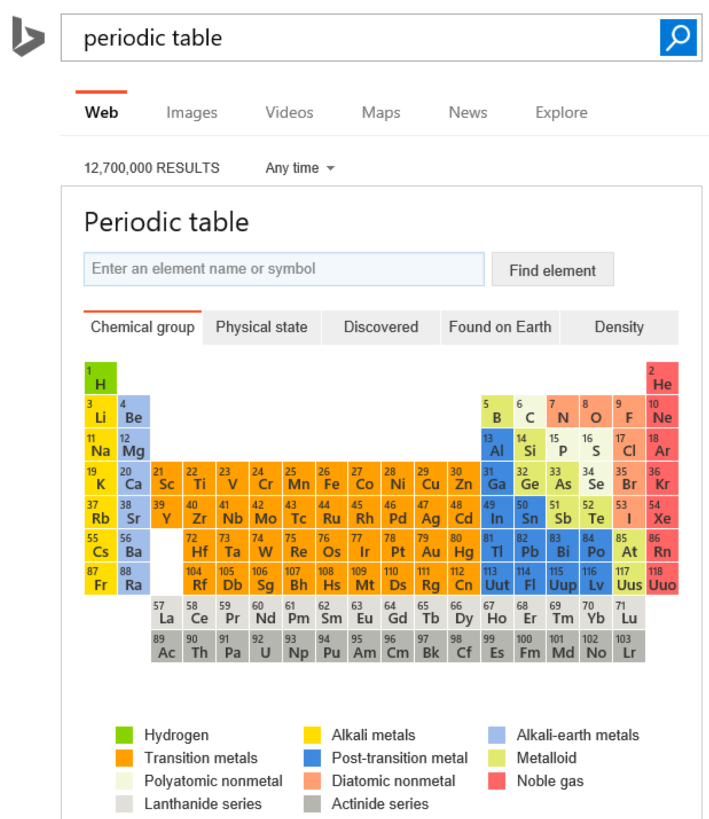
- After pre-web components are run, the query is federated out (dispatched) to dozens of *Answer Services*
- *Anybody* can ship an Answer service, and *any* answer can trigger for *any* query
- Answers vary widely in complexity. Some (like Flights/News/Stocks) have up-to-the-minute data requirements. Some (like Image/Video) have full indices and relevance stacks.

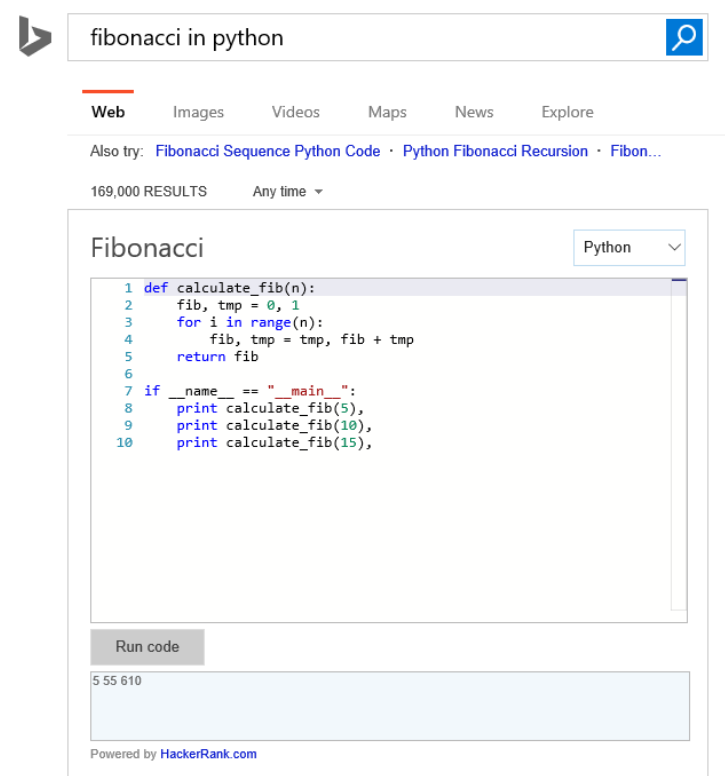


Instant Answers: Targeted Experiences

- Instant answers are a great way to meet users' demands
- Users no longer have patience for the traditional blue links ☺

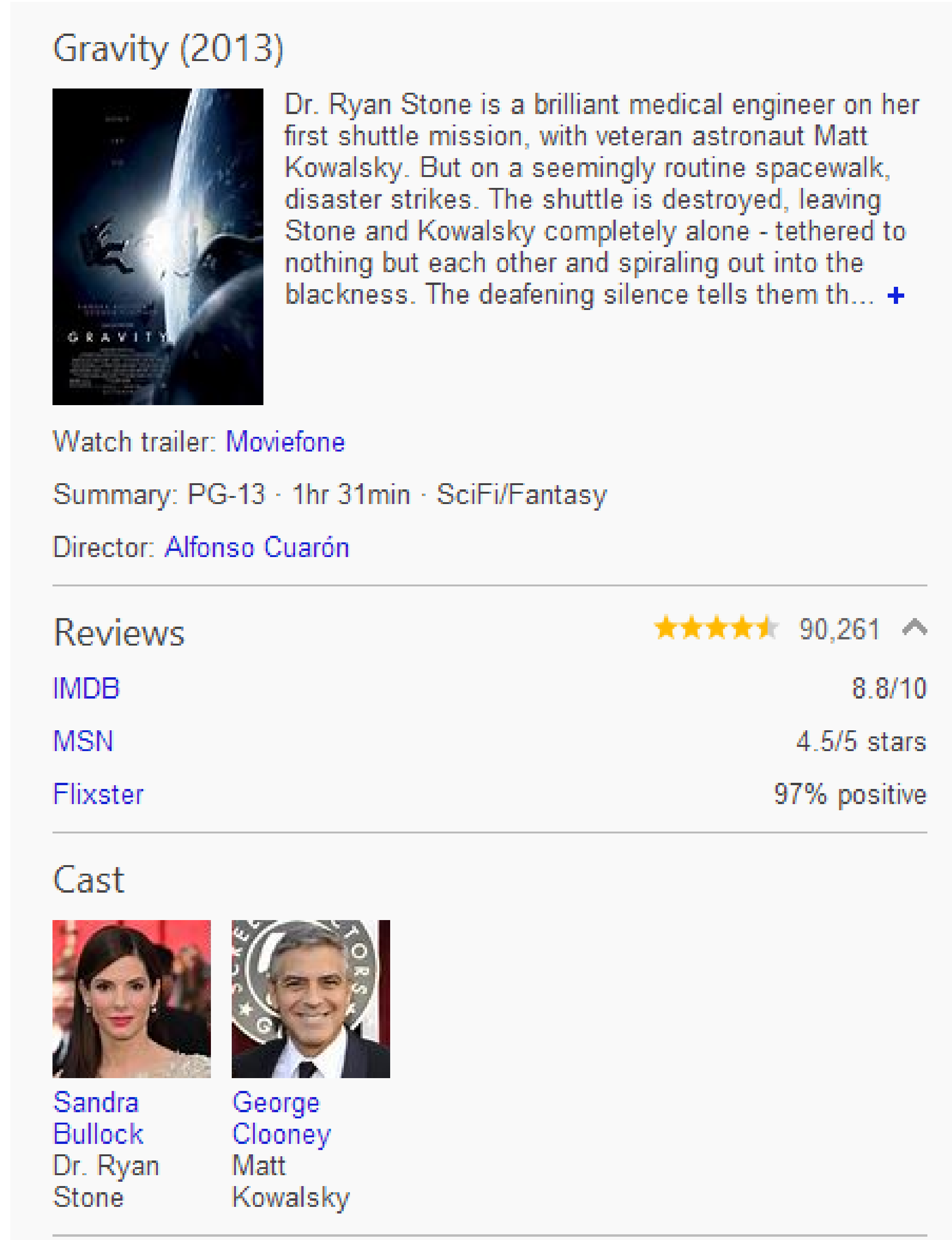






Entity (or Side) Pane

- The Entity Pane is a special kind of Instant Answer
- It pulls in content from various answers and displays it all together in one place
- Search engines keep a graph of entities on the Web



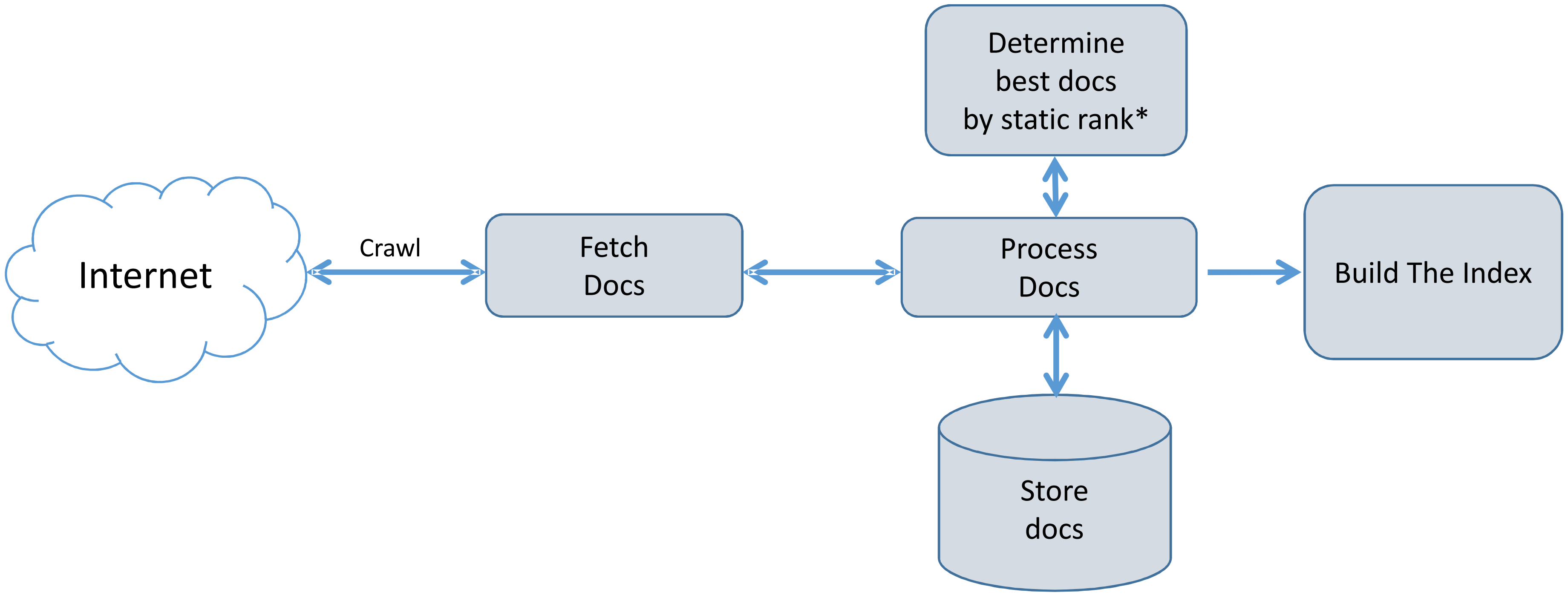
An Aside on Web Relevance

Web Relevance

- Objective: find the 10 (sometimes more, sometimes less) most relevant blue links for the query and put them in the right order on the page
- How this happens in 6 oversimplified steps:
  1. Acquire billions of web documents and index them
     "this is a hard problem from many angles, mainly from a scalability and storage standpoint"
  2. Match each user query to some possibly relevant web docs
  3. Use machine learning to rank the candidate web docs
  4. Return the top ten (give or take) to the user
  5. Do this globally
  6. Do this in a blink of an eye!!!

Where do the documents come from?

Generation of the Index = the process of crawling/storing docs and building the index



*Static Rank = the query-independent importance score that we assign to every document on the web

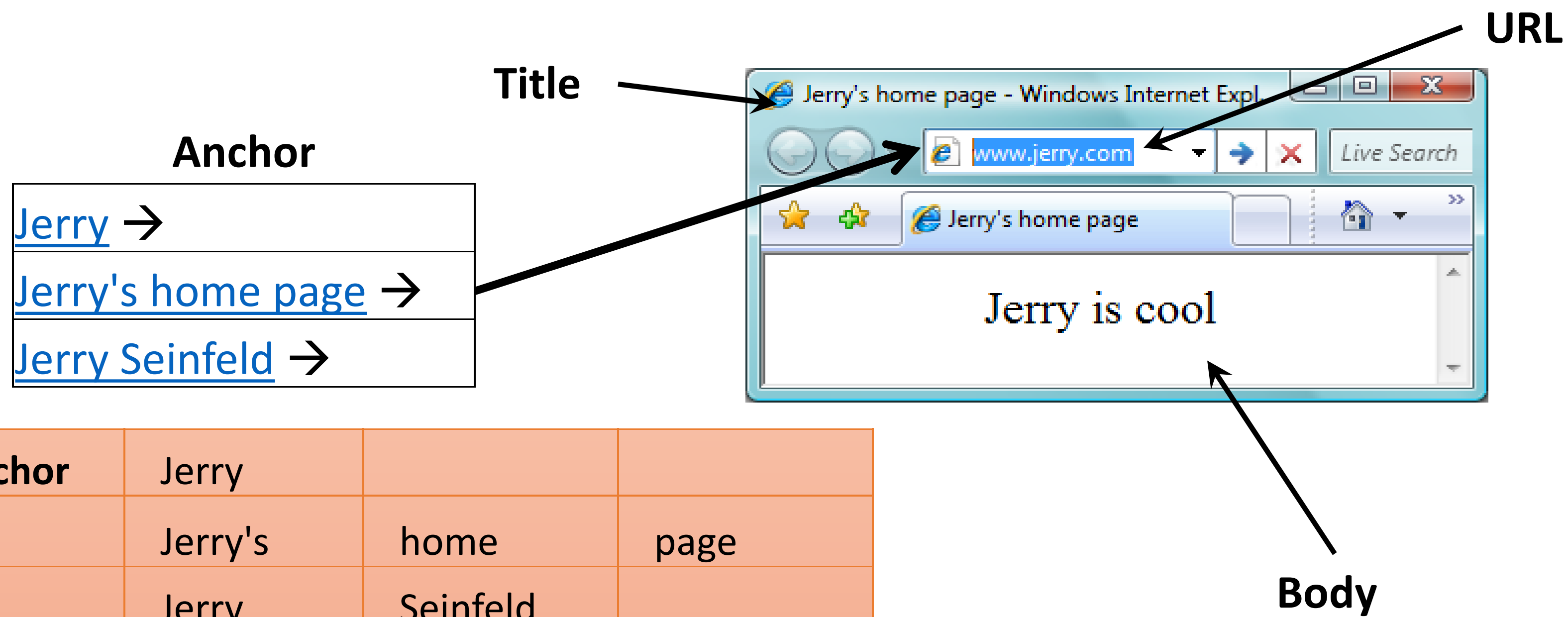
How are documents served for a query?

Index-Serve = the process of hosting docs and returning them for incoming queries

- Search Engines have multiple-tiers platform to balance for freshness, relevance, index depth, and cost. Bing for example:
  - Fresh tier
    - **Millions** of documents
    - Doc discovery to hosting takes <1min
  - Main tier
    - **Billions** of documents
    - Updates in <1 day
  - Depth tier
    - **Many Billions** documents
    - Updates in <7 days
- Includes both En-us docs as well as global docs

How is a query matched to documents?

There are four basic streams (text sources): **Anchor, URL, Title**, and **Body (or AUTB)**



Anchor	Jerry		
	Jerry's	home	page
	Jerry	Seinfeld	
URL	www	Jerry	com
Title	Jerry's	home	page
Body	Jerry	is	cool

- AUTB is just the basics that Bing uses. Other engines might use other streams. We also rely heavily on Speller and Synonym expansion.

So Far, We Have a Big Pile of Documents

- We’ve matched a few thousand (or more) documents to your query
- Now we just need to get them in the right order
- How do we do that? Machine learning!

Machine Learning is like Guess Who

The Steps of Machine Learning

Machine learning helps a machine answer human questions (e.g. what are the best docs for this query?) by quantifying human questions into scores.

1. First, create some examples *where you know the right answer.* This is called **training data**.
2. Figure out some important *easy* and *quantifiable* questions to ask of those examples. These questions are called **features**.
3. Use the training data to "learn" how to get the known examples right by adjusting weights until the numbers work out. This is called the **training process**.
4. Then, for *new* examples, the system can take an educated guess at the right answer. This is called **generalization**.
5. **Measure** how well you do. Do this early and often.
6. Then go back and fix the problems. This is called **tuning**. Rinse and repeat.

Web Ranking Features

Features can be for the query, the doc, or both. Here are just a few examples of many that are used by different search engines:

- Do the query and doc belong to the same **category**? (sports, movies, etc.)
- Do the query and doc come from the same **geographical origin**?
- How many times does the query term appear in the doc? (**frequency**)
- Does the query have any **known phrases**? e.g. {*star wars* trailer}
- How important is the doc? (Remember **static rank**?)
- We also look at **doc clicks**.
- Has the doc been classified as **junk/spam/adult**?
- What **query terms** have people used in the past to get to (click on) this doc? (queries association technique)
- And many, many more!

*In the end, each query/doc pair gets a **dynamic rank** score. The docs are ordered by this score.*

How do we gather training data?

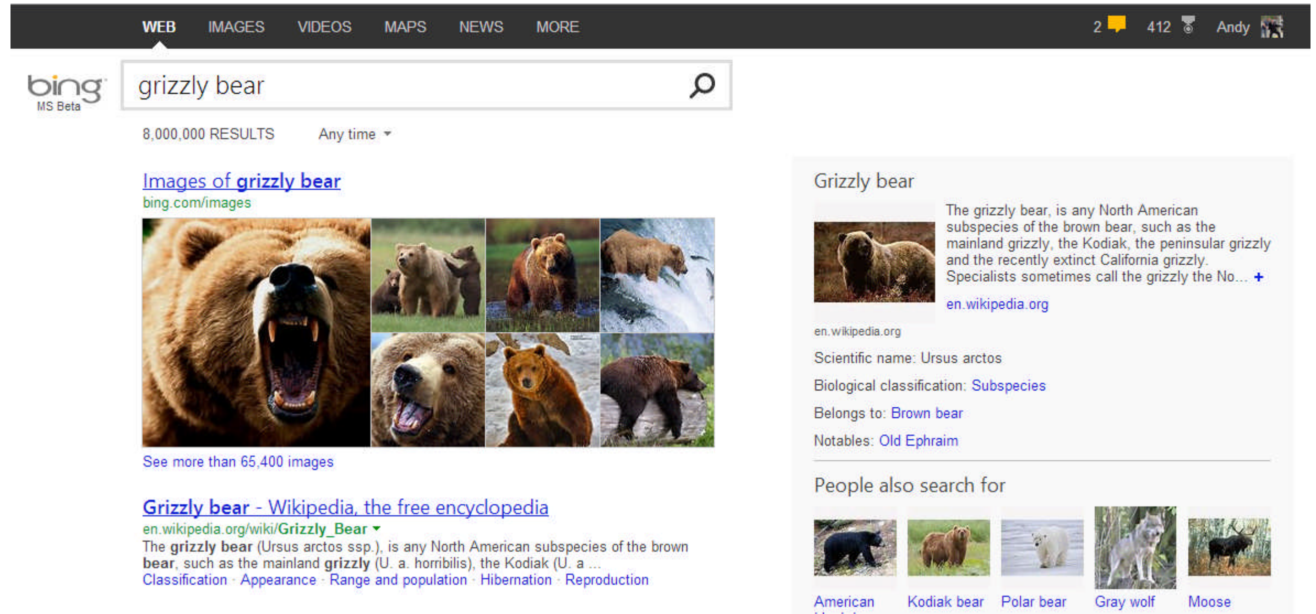
Relevance Measurement: judges assess query/doc pairs on a five-point scale. This is used for both training and testing.
The process of pulling in the top N docs for a query and storing them is called **scraping**.
We use these judgments to train, test and measure our rankers (machine learning models).

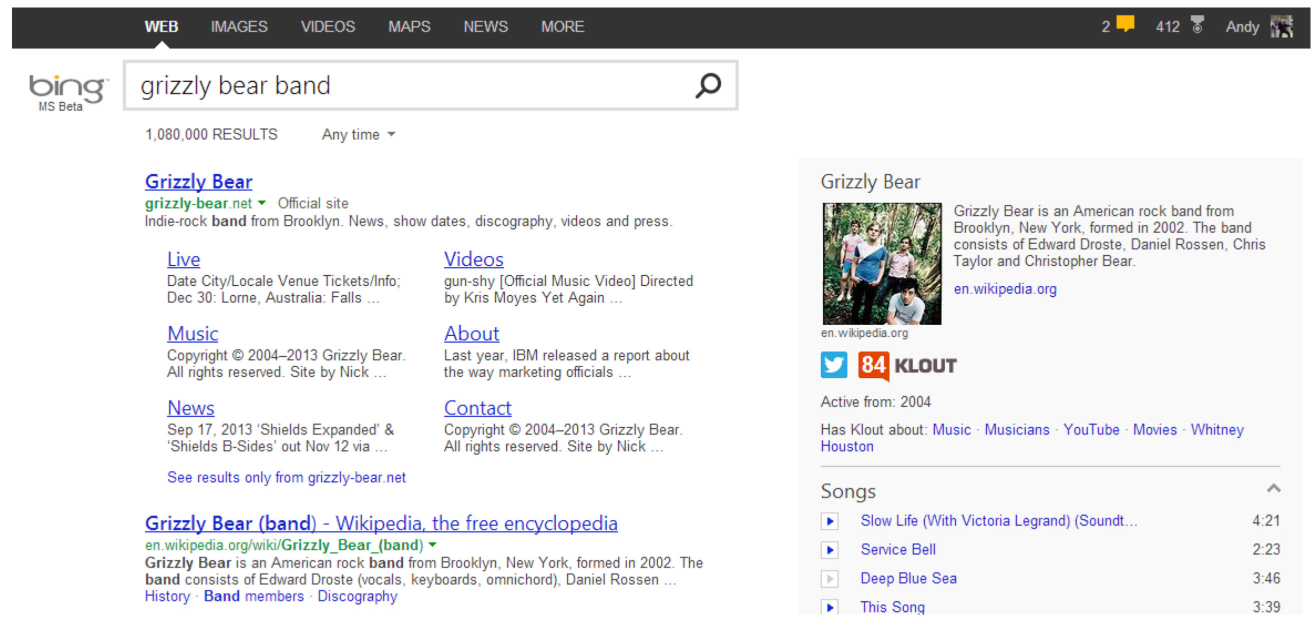
Bringing all together – recap!

1. A query comes in via one of several entry points
2. Some contextual information comes with the query
3. A few core services (e.g. Speller, Alteration) process the query
4. The query is “federated out” to Web, Answers, Task Pane, etc.
5. A subset of answers trigger for the query
6. The web ranker matches *many* documents and returns the top 10
7. All of this takes few milliseconds…
8. Now, we have a big pile of stuff waiting to be rendered on the page

Page Coherence

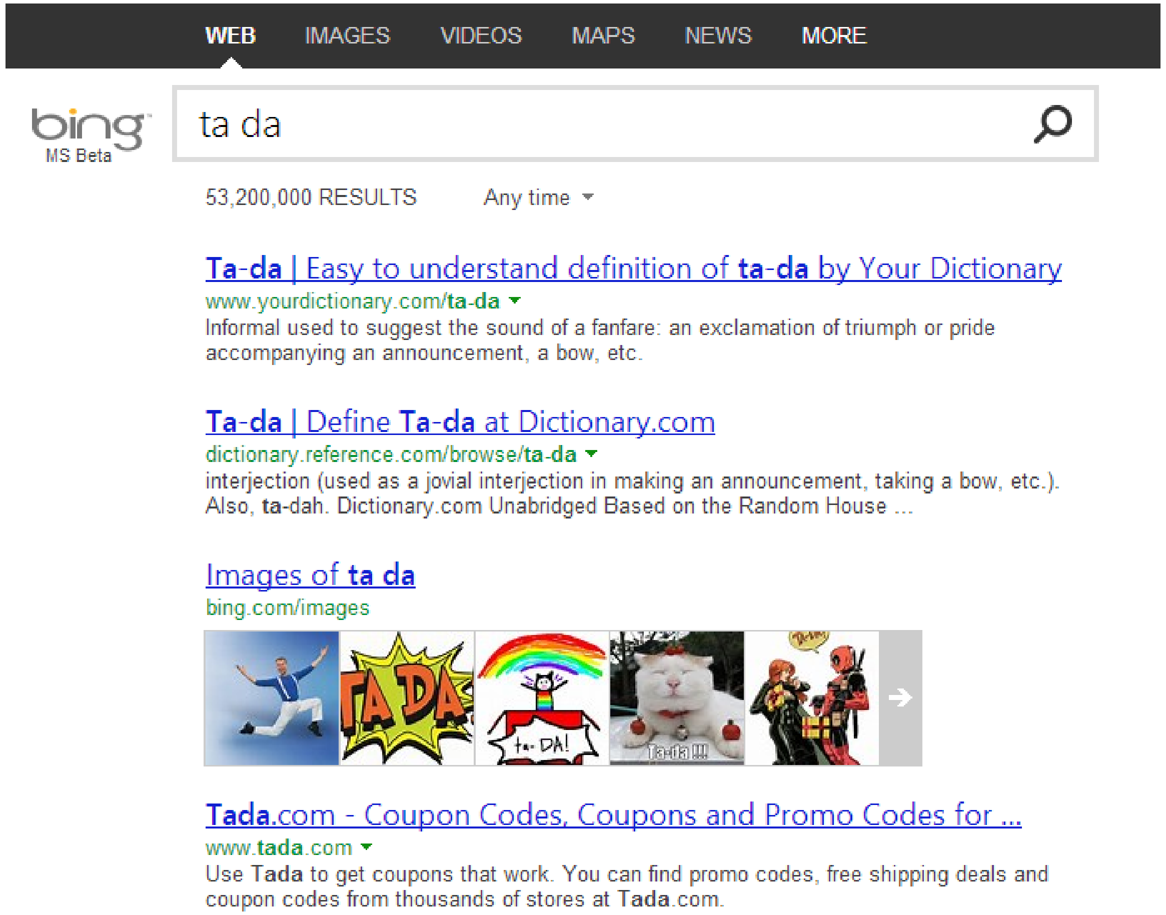
- It doesn't look good to show apples and oranges intertwined…
  - Jaguar: The Car? The Animal? The City?
- Need to apply **suppression**, and then
- Need to apply **final ranking**
- **Coherence** between web docs and answers is a key component
- **Past data** (user clicks) is also important
- The job of suppression is to **minimize defects**
  - A defect = irrelevant or otherwise bad content for a query
  - Components that perform poorly lose credibility
- The job of the final page ranking is to **push the best stuff to the top and the less-good stuff toward the bottom**
  - This is done via a metrics derived from click-info





UX (User Experience, or UI)

- After Whole-Page Relevance decides what to show, it passes the final content to the UX layer
- The content is rendered beautifully on the page
- The layout is customized by entrypoint, but the content is (mostly) the same
- UX Server: ASP.Net
- UX Client: Java Script (Libraries) + HTML + CSS3



Engineering Development Rhythm

Inner Dev Loop
- Feature development
- Concludes at checkin

Outer Dev Loop
- Build validation
- Concludes at PROD deployment

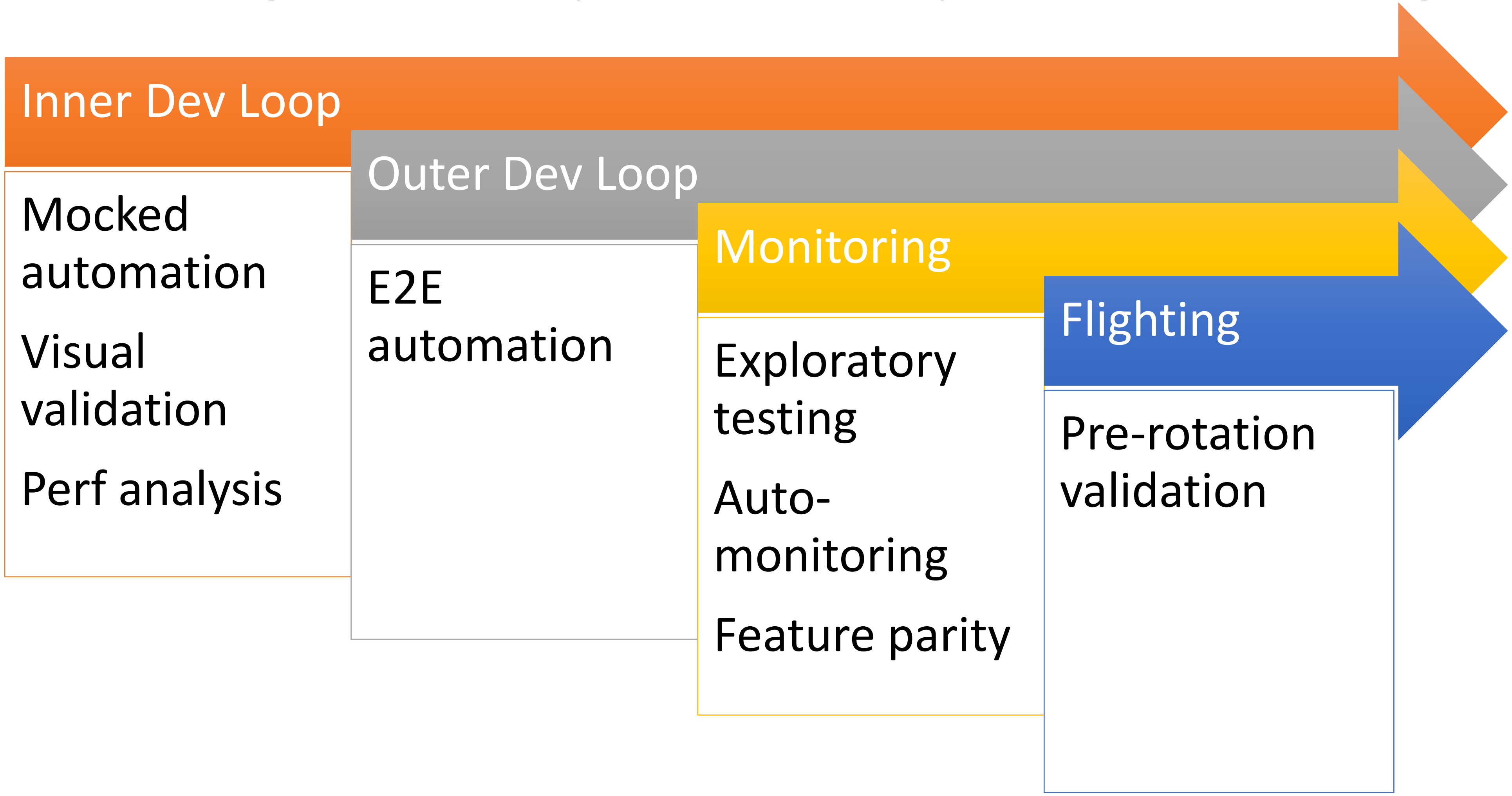
Monitoring
- Live Site quality
- Continuous

Flighting
- Controlled exposure of features

Development is composed of discrete states

https://www.youtube.com/watch?v=SiPtRjiCe4U

Engineering Development Rhythm - Testing



Testing is composed of overlapping states

Engineering Development Rhythm

- Hundreds of engineers across many continents!
- Shipping multiple times a day (millions of lines of code):
  - *Continuous Delivery* ➔ *"your check-in will go to production soon!!!"*
- Tens of thousands of automated tests
  - If any fails ➔ don't ship
  - Don't write tests? Well, good luck shipping to hundreds of millions of users!!!
- Flight everything ➔ Analyze the data ➔ Ship or fail fast!!!

Guardrail Metrics	Treatment	Control	Delta [%]	Pval
Quick Back 20	0.2295	0.2281	0.0014 [0.60%]	< 0.001
Algo Pane Load Time(Overall PLT)	1212	1208	4.055 [0.34%]	< 0.001
Revenue /UU	1.088	1.075	0.0130 [1.21%]	< 0.001
Truncated Revenue / UU	0.8571	0.8504	0.0067 [0.79%]	< 0.001
Distinct Queries / UU	14.67	14.67	–	1.001
Average Log Record Size (in KB)	111.4	111.1	0.2545 [0.23%]	< 0.001

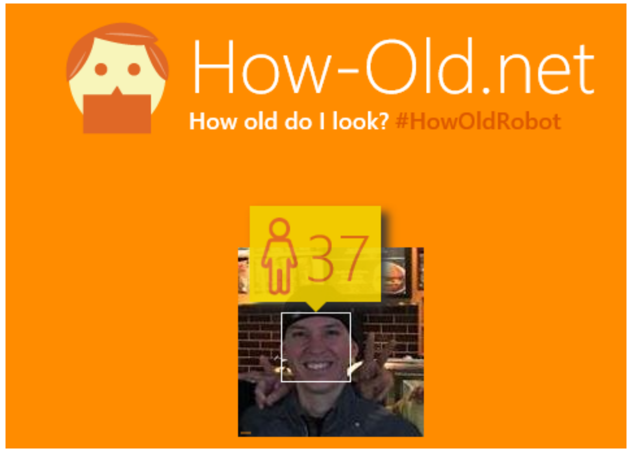
Search Engines Future Trends and Challenges

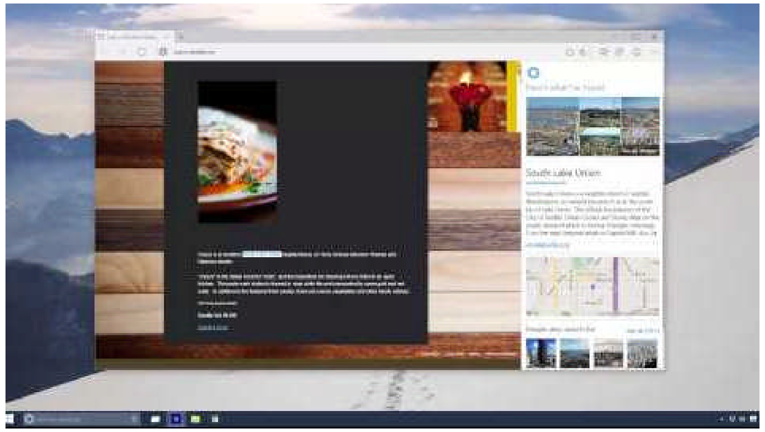
Search Engines Future Trends and Challenges

- **Data**: not every data is in the index…
  - Offline data – other formats
  - Live data - happening now, I mean, really, NOW!!!

Search Engines Future Trends and Challenges

- **AI**: it is only in its infancies
  - Image and Video Understanding
    - Media → Features → Syntax → Semantics
  - Personal Assistant (Cortana, Siri, Google Now, Alexa)



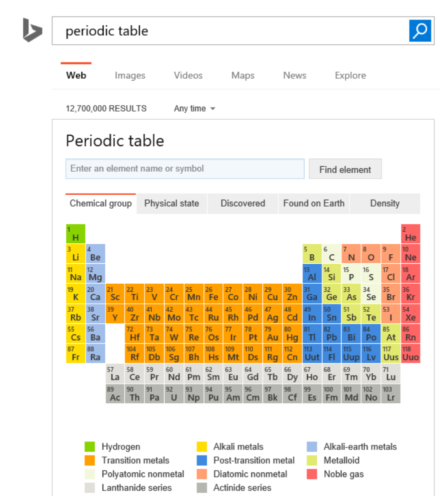


Search Engines Future Trends and Challenges

- **Fundamentals**: more connections, ~~less~~ no patience
  - Internet of Things (IoT)
    - Availability across devices (phones, wearables, cars, things)
    - Poses unique User Interface challenges
    - Poses unique privacy concerns
  - Performance:
    - Not fast… but NOW!!!!
    - Fun experiment: slowdown flight ➔ revenue hit!
    - Pushing the limits of techniques
      - Algorithms, distributed computation, hardware, networks, caching, programming languages, etc.
    - Faster data analysis
      - Data is becoming cheaper…
      - However, useful information from the massive data sets is hard!

Search Engines Future Trends and Challenges

- **Collaboration**: search is also about connecting services
  - No more blue-links: the answer must be right there!
  - Many specialized companies
    - *Servicefication* of platforms



Q&A