John Dimm

San Diego, 2024 http://www.johndimm.com/ http://www.linkedin.com/in/johndimm

Full Stack Lead and Data Engineer

Highly effective in roles ranging from SQL maven to full stack developer to hands-on manager to individual contributor.

Skill Summary

- MySQL, PostgreSQL, Lovefield
- SQL query optimization
- React and nextjs for front-end development
- Serverless function apps on Azure
- Data Pipelines and ETL processing
- Amazon Alexa Skills
- Google Chrome Extensions
- Recommender systems

Employment History

2022 – 2024 Gardyn Staff engineer

Gardyn sells an end-user device that grows a wide variety of plants hydroponically in your house or apartment. I took on back-end development of a python API to manage end-user membership and expose data uploaded by the raspberry pi processors in the field, and wrote Javascript and Typescript serverless function apps hosted in Azure to change light and water schedules through phases of plant growth. I designed and implemented a method to create time lapse flipbooks rendered on the client, and later, server-generated videos using ffmpeg. To enable both features, I wrote a multi-threaded python script to select and downscale images captured on the pi.

2020 – present Lean Software Development founder

A sole-proprietorship to support my consulting work.

2020 - 2021 Galigo CTO and co-founder

Eleventh Hour Global, based in Singapore, changed its name to Galigo and I moved into the CTO role in September. We were building a marketplace for green products and a tool to help secretariats of industrial organizations manage their memberships lists.

2020 Eleventh Hour Global Software Engineer

Eleventh Hour is a topical search engine for green products. The standard way to aggregate product information from a group of suppliers is to get a product feed from each of them and aggregate products into a central database. I developed an alternative that requires no effort from suppliers, implemented a rating and review system for products, and created a novel recommender system that does the analysis in postgreSQL.

2018 – 2019 Zuora Senior Software and Data Engineer

Zuora manages subscriptions for companies like 24 Hour Fitness, HBO, and Carbonite. Insights is a tool that aggregates data from 20 sources to present a complete picture of customers. I implemented the utilization model to track pre-paid usage and determine when a customer was likely to go into overage, and created a test harness for integration testing. Later, I created an ETL process to help the accounting department during end-of-quarter reconciliation, taking data from two sources, loading into a local postgres database, and generating a

combined report. Finally, I was in a 4-person team that created a new, simplified aggregator for internal use. I learned more than I ever wanted to know about subscriptions, rate plans, rate plan charges, and rate plan charge tiers.

2015 – 2018 Acculitx, Inc. Senior Data Engineer

Accuscore went through a series of promising customer evaluations, but it never caught fire. A year later, the system I set up was still processing data and generating reports, with no manual intervention.

2011 – 2014 SYSTRAN Director of Software Engineering

A pioneer in Machine Translation, SYSTRAN was for many years the engine behind the free translation services offered by Microsoft, Google, Yahoo, and Alta Vista, where it was called Babel Fish. My group in San Diego consists of software engineers, computational linguists, classically trained linguists, and lexicographers. We use hybrid approaches and domain-specific parallel corpora to train specialized translation models and extract dictionaries.

2008 – 2011 Photometria, Inc. Director of Engineering, Software Developer

E-commerce using Amazon and linkshare. Recommendations of products based on co-occurrence. Google Maps mashup showing worldwide activity. Computer vision algorithms in openCV. Fast Thin Plate Spline rendering using openGL for the iPad.

2007 – 2008 Veoh Networks Director, Search & Analytics

My international team was responsible for metrics and recommendations. Due to an exponential rise in log volume, the Veoh web metrics system stopped functioning a week before I arrived. We put together a stop-gap ETL system using MySQL and perl in a few weeks, later replaced by Oracle.

2004 - 2007 Websense Director, Reporting and Database Development

I led a team to resolve one of Websense's biggest problems: we could not assume partition support in MSDE/SQL Server and our customer's databases suffered declining performance as they filled up. We came up with a simple partitioning technique that worked even on the low-end database MSDE. It allowed users to go beyond the size limitation of MSDE, using multiple databases and UNION ALL queries. The system allowed Websense to satisfy its largest clients, among them Boeing, Rolls Royce, and British Telecom.

2000 - 2004 Websense Chief Scientist

My team developed tools for Websense's impossible core task: classify the entire internet into about 80 categories. We developed a pipeline to manage and report on computer-assisted human classification of web pages. Our Digital Fingerprint was a heat-map created from Support Vector Machine scores of membership in each category.

Highlights

Time lapse for Gardyn

Platform: Azure, python, react/nextjs, ffmpeg

The Gardyn device has cameras that capture a photo of your plants every 30 minutes. The goal of the project was to create a time lapse video of plant growth, using selected images from that stream hosted on Azure. The first version did all the work on the server in a daily job using ffmpeg.

I had an idea of doing it in a simpler and more flexible way. I made a react/nextjs proof-of-concept webapp to create a video-like experience but without actually creating a video. Instead, it shows the set of selected photos one after another in the same html img element, producing a slideshow or flipbook. It was not obvious that this approach would result in a smooth animation, but it did, and the feature allowed users to select their own time range. The same approach worked on the mobile

app, with moderations.

I later added a video-generation feature to allow sharing, this time done on demand rather than in a nightly job.

Acculitx Driver Score Portal

Platform: AWS, Python, MySQL, PHP, celery, javascript, angular, react, auth0
With: Peter Ellegaard (physics)

Accuscore generates a FICO-like score for driving behavior using accelerometer data gathered by an on-board device or cellphone. The first version used the database only for storage, passing large amounts of data back and forth between the database and the application. Ingest was driven by Python and done line by line. All analysis was done in Python, doing multiple passes through many thousands of records. I moved this work into the database using stored procedures, and ingested files in batch. Trip files that took minutes to process were now being done in a second or two. I replaced the bloated Joomla front-end with React, and extended it to show each driver's week-over-week scores across several dimensions.

The summarized raw physical measurements formed an unwieldy distribution, and some time had been spent trying to tame it. Playing with the data in R, I found that the data is log-normal – the log of the same readings very clearly form a normal curve. We were able to reduce six dimensions to a single score by computing a simple weighted average, and the result was again a normal curve, as desired.

2018: What Looks Good?

Platform: Yelp Dataset Challenge, React, MySQL
Link: http://www.johndimm.com/yelp_db_caption/app/

I wanted to create an interface that lets you start by thinking about *what* you want to eat and decide later *where* you want to eat. The Yelp Dataset Challenge makes available a good chunk of anonymized data. Using only the restaurant table, the photographs, and photo captions, it was possible to extract the latent concept of a Dish. The key insight was that when people take pictures of food, the captions often have a useful feature: they just say what it is. “Sushi”, “Steak”, and “Grilled calamari” are common captions. We can identify dishes from the co-occurrence of such phrases in caption text, then expand coverage by including captions that contain those phrases. The UI relies on two recommender systems built on the relation between restaurants and their dishes, to find related dishes and related restaurants.

Web Speech API Experiments

Platform: Google Chrome, jQuery, Systran/Google/Azure translation, WebRTC, Alchemy API, YahooBOSS, flickr API

When I was managing the team of computational linguists at Systran, the Web Speech API was released in Google Chrome with support for continuous speech recognition and I wrote several apps to make use of it. **TalkShow** is an experiment in “ambient computing” – it runs in the background while you talk, converts speech to text, analyzes the text to extract noun phrases and proper nouns, does live queries against Flickr and Bing, and displays a slideshow of possibly relevant images. **Fun With Speech** is a set of minimal demos of several speech and language related features and was the basis for my talk at the 2014 jQuery conference. The **Translating Telephone** was based on the open source video conferencing system WebRTC. Each participant speaks in their own language and the other users see the video feed with subtitles in their own language. Translation was done by Systran, Google, or Azure.

Veoh Video Recommendations

Platform: MySQL, Oracle, Hadoop, pig
With: Alexander Sherback (SQL guru), Ted Dunning (scientist)

Users often upload videos without a good description, making them invisible to standard text search. It turns out that a system designed to find related videos also solves this search problem.

People rarely bother to provide ratings on videos. One key insight in the new recommendations system I designed and implemented was to use passive viewing behavior instead of explicit ratings. The longer the

viewing session, the more likely the user was really interested in the video. Brief viewing sessions were rejected, since they are often the result of a misleading title or thumbnail. In this way, we skimmed off the top of our mountain of data to extract information reflecting true user interest.

We noticed that the method of grouping and the method of finding surprising features of a group are independent. We could group people by video viewing behavior and use that to predict search queries. Or group people by search query and then look at the videos they liked. This last technique is formally the same as a search engine: the input is a query and the output is an ordered list of videos.

An interesting consequence of this search method: "Paco de Lucia" yields videos on flamenco guitar even though we had no Paco de Lucia at the time. That is because users who searched for him ended up watching other flamenco videos, and behavioral search makes use of that fact. Of course, the text-based search engine finds nothing for this query. Behavioral search can find relevant results even when there is no textual evidence.

We started with a naïve SQL solution running on Oracle and ended up with an elegant Pig script running on a 20-node Hadoop cluster.

Websense Explorer
Platform: SQL Server for Windows, MySQL for Unix, perl
With: Vince Star (developer) and Don Aymar (graphic designer)

Logs of URL access by a company's employees are summarized by date, category, user, and other dimensions. Websense had a traditional batch-oriented template-based reporting program. The Explorer interface lets you select a value for one dimension and drill down by another in the same click. It was a game-changing product that gave instant information, a sense of free movement through the data, and showed users immediately the issues Websense was designed to address. It was touted as the flagship product of Websense for several years and was often cited as the deciding feature in a purchase.

Personal Projects

2024: Wall of Comics
Platform: React/Next
Link: https://silverage.vercel.app

I collected Marvel comics as a kid for a few years. I still have them, and they are worth something now. I wrote this tool to display them all, using the Filterpanel to let users select subsets.

2021: Collaboration Graph Browser
Platform: React/Next and PostgreSQL
Link: http://www.johndimm.com/collaboration.html

A research tool for students and collectors. I used musicbrainz's public database to extract the collaboration graph, and developed this front-end interface. It answers the question: What were the musicians on this album doing before and after? The same concept was also applied to the IMDb database for movies and tv shows.

2021: Filterpanel
Platform: React/Next
Link: https://filterpanel-csv.vercel.app/

A software tool that generates a filter panel UI, as seen on Amazon and many other e-commerce sites. Filtering is a powerful adjunct to natural language search. This version downloads a large set of search results and reads them into local memory. All analysis is done in javascript. In spite of that, or maybe because of it, response is super fast.

2019: TalkShow for Alexa

Platform: Amazon Alexa Skill
Link: https://www.amazon.com/John-Dimm-TalkShow/dp/B0816KVF14/ref=sr_1_1?keywords=TalkShow&qid=1579326798&s=digital-skills&sr=1-1

A port of my Chrome web app TalkShow, this one uses the entire user utterance as query to Flickr and Bing through Azure Cognitive Services. Results are displayed in a slideshow done using the Amazon Presentation Language and the AutoPage command.

2019: Breakdown for Mint and Personal Capital
Platform: Google Chrome Extension, React, SQL using Lovefield
Link: https://chrome.google.com/webstore/detail/breakdown-for-mint-and-pe/npkcbabhgmplkmaaljpcbombbgkggilm?h1=en

Breakdown slices and dices personal financial data. The interface works on all kinds of multidimensional data. For personal finance, it loads your downloaded Mint or Personal Capital data into Lovefield, the browser-based local database written in javascript by a team at Google.

Patents

2008: Search System using Patterns of Usage

From the filing: “The fact that recommendation technology can be used to implement a search function appears to be completely novel.”

Collaborative filtering is normally used to group people by a given behavior (products purchased, web pages viewed), and then use that to predict future behavior of the same sort. You might want to buy this product because it was bought by people who bought some of the things you bought in the past.

The twist here is to switch in the middle, from one kind of behavior to another. For the query-to-document mapping, it looks at the users who have done a given query, and then instead of looking at the other queries they have done (which produces similar queries), it looks at all the videos they have watched. It doesn't matter how they got to them: from the given query or from some other query or by browsing or whatever. People who performed this query disproportionately watched these videos.

2007: System and method of monitoring and controlling application files, Patent number: 7185015, with Harold Kester, Ron Hegli, and Mark Andersen

Websense wanted to create a categorized database of Windows applications, in order to filter and control them. This was seen as a logical extension of their core business of filtering websites by category. The patent dealt with a practical problem: we couldn’t possibly buy or somehow acquire every piece of software in the world. Our solution was to ship software with our product that examines and generates hashes for the application files our client software finds on the user’s computers and send that information back to Websense for classification. I wrote the Windows Portable Executable hashing routines and a created a GUI in C# to display the captured version info, link to websites, and facilitate the manual classification process.

Publications

2001: Presenting content one slide at a time

Platform: Unix or Windows, World Wide Web
Language: Perl, Javascript
Link: https://people.apache.org/~jim/NewArchitect/webtech/2001/04/dimm/index.html

The lowly slide show is an underrated user interface paradigm. This article examines four implementations.

1991 : A tiny Windows Class Library

Platform: Windows
Language: C++
Link: https://www.unz.com/print/ProgrammersJournal-1991nov-00016/

Soon after the release of the first Windows C++ compiler by Borland, this was the feature article in the November/December 1991 issue of *Programmer's Journal.* Before OWL or MFC, it showed how a simple class library in Borland C++ lets developers avoid the usual Windows boilerplate code. Create a generic window in two statements.

Education

- University of Washington: Ethnomusicology, Philosophy, Linguistics, Mathematics. B.S. Mathematics, B.A. Philosophy, cum laude, Phi Beta Kappa.
- Reed College: Physics major

Foreign Languages

I spent four years working in Paris and my extended family is French, so I can speak and read it.