

PostgreSQL: An Unexpected Journey Lessons Learned along the way

Joe Conway
conway@amazon.com
mail@joeconway.com

AWS
October 30, 2026

An Origin Story for This Origin Story

- FOSSASIA 2026: a conversation in the Expo hall
- ”How did you get into Postgres?”
- A digression into my early career
  $\Rightarrow$ the chain of events that led me here
- ”You should write that all down”
- Reminded me of ”The Hobbit: An Unexpected Journey”
  $\Rightarrow$ hence the title
- I hope you find it interesting, enlightening, maybe even inspiring!

aws

The 50 Year Journey

- 1976 – 1980: High School
- 1980 – 1984: Virginia Tech
- 1984 – 1989: US Navy
- 1989 – 1997: CarTech
- 1997 – 2009: Cymer
- 2009 – 2015: credativ
- 2015 – 2022: Crunchy Data
- 2022 – ????: AWS
- The butterfly effect: small actions, big consequences



High School: First Programs (1976 – 1980)

- Two computer programming classes offered
- Took the first as a sophomore: Basic
- IBM 5100 ”suitcase” semi-portable computer
  - Heavy but portable, small green screen
  - Booted into Basic . . . or APL
- Typewriter-style terminal to the BOCES district mainframe
  - Ticker-tape (paper tape), not punch cards
- The second programming course, APL, didn’t fit my Junior year schedule
  - Pushed for, and got, permission to self-teach it

The Hardware





Left: IBM 5100. Photo: Sandstein, CC BY-SA 3.0, via Wikimedia Commons

Right: Teletype ASR-33. Photo: Arnold Reinhold, CC BY-SA 3.0, via Wikimedia Commons

Lessons

- Pursue your interests
- Advocate for yourself

College: The Third Choice



- Three schools ranked; I expected #1 or #2
- #3 was the backup: most economical, farthest from home ⇒ the least comfortable choice
- Then I visited all three – and #3 won
- Mechanical Engineering ⇒ Navy NUPOC ⇒ submarines

Burruss Hall, Virginia Tech. Photo: Eric T Gunther, CC BY 3.0, via Wikimedia Commons

Lessons

- Sometimes your first choice is not the best choice
- Be open to having your mind changed by facts and observations
- Take risks – make the uncomfortable choice sometimes

The Navy (1984 – 1989)





Left: USS Chicago (SSN 721) coming home to San Diego, April 1991.

U.S. Navy photo by PH2 Michael Rodriguez (public domain), via Wikimedia Commons

Everyone Teaches

- Weekly officer training
- Rotating turns: we taught each other
- Preparing to teach a topic $\Rightarrow$ mastering it

Diving Officer: Zero Bubble

- Goal after submerging: neutrally buoyant and perfectly level
  - No speed, neither rising nor sinking: ”zero bubble”
- Before the dive: run the numbers, move water tank to tank
- Standard procedures existed for all of it
- But buoyancy also depends on temperature and salinity
  - And you often had the profiles already
  - Recently surfaced, or frequently traveled waters
- Not part of the standard process – I used them anyway
  $\Rightarrow$ hit zero bubble more often than the typical diving officer



Fire Control Officer: Stacking Dots

- Battle stations role, typically a junior officer
- Console purpose-built to find a firing solution
- Homing torpedoes – but they must get close enough to acquire
- Usually all you have: sonar bearings to the target
- Three dials: target speed, angle on the bow, range
  - Dial until the dots stack in a straight line
  - Straight stack $\Rightarrow$ solution matches the bearing rate

Fire Control Officer: Constraining the Solution

- Nobody told us: it's an unconstrained problem
  - A straight stack is "a" viable solution, not "the" solution
- So constrain it with every scrap of data
  - Periscope divisions + recognized ship profile $\Rightarrow$ range
  - Trained eye on the target $\Rightarrow$ angle on the bow
  - Sonar: ship type + screw RPMs $\Rightarrow$ speed
- Result: $\approx$87% hits vs. fleet average $\approx$50%

Success Depends on Everyone

- On a submarine, survival literally depends on the whole crew
  - One person, wrong thing, wrong time ⇒ could sink the boat
  - One person, right thing, right time ⇒ could save everyone
- Every position and every rank is a valuable teammate
  - Listen carefully, no matter your own rank
- Small crews (≈120 – 130) ⇒ very tight knit
- 40 years on: reunions every couple of years
  ⇒ our non-profit keeps them going

Lessons

- You learn best when you teach
- Use all available data to enhance results
  - Diving officer: temperature and salinity
  - Fire control officer: constrain the solution
- Attention to detail
- Shared adversity strengthens team cohesion
- Side notes:
  - Great Video
  - The Navy moved me from east coast USA to San Diego

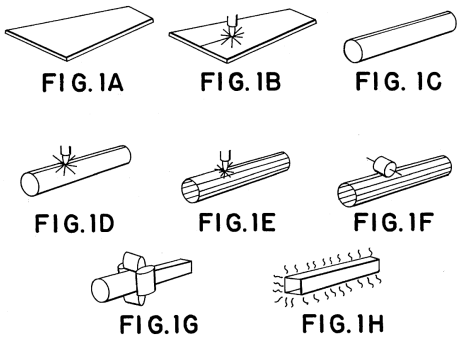
Industry Years



CarTech (1989 – 1997)

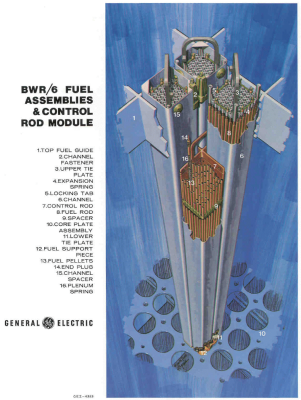
- First civilian job (in San Diego)
- The big product: ”nuclear fuel channels”
  - Thin-walled square tube, special material, tight tolerances
  - ≈125 – 130 mm per face, ≈4 m long, ≈2.5 mm wall
- $4,000 – $5,000 apiece, sold in contracts of ≈250
- Each job: a custom, bespoke contract
- The competition: Hitachi and General Electric
  ⇒ we were the small company
- Also where I met my wife

From Sheet to Channel



US Patent 5,574,761, Figs. 1A – 1H – I am a named inventor

Where the Channel Goes



The square tube wrapping the fuel bundle is the channel – our product.

Illustration: General Electric, ”BWR/6 Fuel Assemblies & Control Rod Module” (GEZ-4383)

The ”Rock”

- Our inspection machine for finished channels
  - Sensors on an outer bridge + a cart riding through the tube
  - Widths at several stations, straightness down the length
- Computer-driven, but. . .
  - Dimensions printed on wide green bar paper, dot matrix
  - Nothing stored
- A human then checked hundreds of numbers per part, by hand
  $\Rightarrow$ tedious, error-prone, a lot of work

The Rock: Don't Waste Good Data

- Learned how the Rock's program worked, then changed it
  - QA enters the drawing requirements into a database
  - Out-of-spec dimensions flagged automatically
  - Every measurement stored
- Added trending charts and statistics
  - Mean $\pm$ 3 standard deviations, per dimension, per point
- Operators could see tooling wear coming
  $\Rightarrow$ dynamic corrections mid-job
- Consistently tighter tolerances $\Rightarrow$ quote tighter tolerances
  $\Rightarrow$ the small company out-executing Hitachi and GE

Barcoded Travelers and Stats

- 30 – 40 steps: bend, weld, square, heat treat, polish, inspect
- ”Travelers”: signed-off written instructions, step by step
- Early 1990s: barcodes on travelers, readers throughout the shop
  - Who, which step, start/stop $\Rightarrow$ Oracle 5 database
- Exported to Borland Paradox + a standard set of queries
  - New quote: select the steps + attributes
    $\Rightarrow$ estimated time and cost per job
- Quoting accuracy: $\pm 20\% \Rightarrow \pm 1 - 2\%$
  $\Rightarrow$ win jobs, confident in the margin



The Megamill

- Ask: mill the wall from 2.5 mm down to 1.25 mm
  - Tolerance on the milled thickness: ≈0.15 mm
  - On parts only straight to ≈1.5 mm over 4 m…
- Early-90s CNC: one G-code program, same cut on every part
  ⇒ straight-line cuts on not-so-straight parts
- Instead: PC generating custom G-code, streamed live over RS-232
  - Measure the actual part profile first
  - Every cut on every part generated to match
- Customer didn't think we could do it
  ⇒ ran successfully for years, at a premium

Lessons

- Don’t waste good data
  - Statistics, process improvement, estimating costs
- Automate, automate, automate
  - Data collection, measurements vs. specs, reports/certs
- Don’t accept ”cannot be done”
  - Megamill reduced-wall channels
- Sometimes dating a coworker works out! ;-)

Cymer (1997 – 2009)

- How I got there: a CarTech colleague had moved to Cymer
  - He told me about it ⇒ I followed him over
- First real high tech company
- Capital equipment for the semiconductor lithography industry
- Very expensive lasers, in clean rooms all over the world
- ≈90% of the world market at the time

The Side Hustle (1999)

- My wife's brother had a friend who needed a project-tracking app
- My plan: Microsoft SQL Server and ASP
- His plan: Red Hat Linux . . . and something called PostgreSQL
- PostgreSQL 6.4 – my first introduction
- My first post to pgsql-hackers, answered by Jan Wieck
  - "That is a stupid idea" ;-)
- $\Rightarrow$ my introduction to the Postgres community

CymerOnline: Monitoring Before We Called It That

- To diagnose one laser in the field
  - Suit up, carry a laptop into the clean room
  - Hook up to the laser's RS-232 port, snapshot the metrics
  - Walk out, get undressed, then puzzle over the snapshot
- So: reach that port from outside the clean room
  - Poll periodically $\Rightarrow$ store it in Postgres
  - Serve it back as trend charts on a web page
- Monitoring, before people were talking about monitoring
- Successful test $\Rightarrow$ became a product we sold



Oracle ⇒ Postgres

- Shipped on both Oracle and Postgres
- The embedded Oracle license got too expensive
  ⇒ port everything to Postgres only
- Same data, same hardware, same application
  - Most things faster, or the same, on Postgres
  - A few things slower ⇒ minor application tweaks
- Ran better on Postgres than on Oracle

CBAD

- One central database aggregating the whole field fleet
- The ”Cymer Big-Ass Database” – 2002 – 2003, ≈3 TB of Postgres
  - Not impressive today – very large then
- Predict part failures before they happened
  - Replace during scheduled downtime, not unscheduled
  - Unscheduled downtime in a fab: ≈$1M per hour
- Every laser counts its own pulses
  - Fleet-wide pulse rates ⇒ the industry’s up and down cycles
  - ⇒ reported to Wall Street quarterly

POD and PL/R

- The factory: test and QA dimensional data, no good way to use it
- Straight out of the CarTech playbook: collect, trend, analyze
- Built POD – ”parametric online data” – backed by Postgres
- Wanted better analytics than Postgres offered then
- I knew R; Postgres takes pluggable procedural languages
  $\Rightarrow$ wrote PL/R, the R language handler for Postgres
- PL/R functions doing the analysis on factory data

MSC: Flat Panel Display Annealing

- Transferred out of IT into engineering
- My first capital equipment: I owned the control system
- My team built the MSC – master system controller
- One tool, many embedded controllers
  - High power laser, beam steering, motion stages, focusing optics
- Operators define recipes $\Rightarrow$ the MSC executes them
- Annealing silicon on glass $\Rightarrow$ flat panel displays

In the Clean Room





EUV

- Director of engineering: first full-scale beta EUV light source
- Reused the flat panel MSC code as the EUV master system controller
- Postgres ran the system – and captured what it produced
  - Metrology, temperatures, voltages, from every embedded system
  - Pulsing at 50 kHz – collected and stored at that rate
- ASML bought Cymer $\Rightarrow$ today's most advanced chips
- $\approx$15 more years to a production-worthy system
- My era was 2007 – 2009; I make no claims about today

Lessons

- PostgreSQL!
- Open source!
- Software running hardware is fun!
- Be technically fearless

All-In on Postgres



credativ (2009 – 2015)

- PGCon in Ottawa, 2009: dinner with Michael Meskes
  - One of the founders of credativ, in Germany
- He talked me into standing up credativ LLC in the US
- My first (and only) time as a founder
- First job whose actual purpose was open source – mostly Postgres
  - Before this, Postgres was part of the job, never the point

Wearing All the Hats



- Incorporation, bank accounts, payroll
- Sales, contracts, keeping everybody happy
- Chasing new business while delivering the existing contracts
- Six years, generally successful the whole way
  $\Rightarrow$ but it wasn't going to scale the way I wanted

Lessons

- Chasing business while doing business is hard
- Real tax advantages to running your own business
  - At least in the US – Australia may well differ!
- I loved being my own boss



Crunchy Data (2015 – 2022)

- March 2015: Crunchy Data came calling
- About a dozen people, profitable but small annual revenue
- I was ready to let someone else chase the business
  $\Rightarrow$ and just be in the business of delivering
- A Postgres company $\Rightarrow$ Postgres all day, all the time
- Joined June 2015

What We Built

- A dozen people ⇒ about 100 by the time I left
- Revenue up roughly 10X
- That scale is what made Crunchy Bridge possible
  - I'd pushed for Postgres-as-a-service for a couple of years
  - Bootstrapped – they didn't want to take VC money
- I interviewed, attracted or hired about half the company
- 2025: acquired by Snowflake, IMHO mostly due to Crunchy Bridge, three years after I left



Lessons

- Not a founder? Don't count on the "exit" to retire you
  - The founders did really well
  - Early and instrumental was nowhere near enough
  - I am not retired, and there's a reason for that
- A big pile of options $\neq$ a great deal
  - How many shares outstanding? How many authorized?
  - So what percent of the company is this, really?
  - Can it be diluted in later rounds?

aws

Lessons

- Founders rarely volunteer those numbers
  - The pile of options is meant to feel like enough
  - Often in place of salary a bigger company would pay
- Don't believe everything in the public releases on acquisitions
- Options are not RSUs
  - RSUs: you own them when they vest
  - Options: you own nothing until you pay to exercise

AWS (2022 – ????)

- Left Crunchy in April 2022
- My first really huge employer
  - Bigger than anywhere else I'd worked – bigger than the Navy
- Hired to build the PostgreSQL upstream contributor team
- From the ground up – it did not exist



Why AWS Was Ready – and How I Got There

- Years of selling services built on Postgres
  - The community's criticism: not contributing back in a big way
  - A big company that didn't yet understand how to
- Then people inside got involved; a Postgres company was acquired
  - They made the case: it's right, and it's good for Amazon
- Jonathan Katz had left Crunchy for AWS a few months earlier
  - I asked him: are they really supporting open source now?
  - His answer: yes – he believed they were sincere
- He pointed me to Jim Mlodgenski inside AWS
  - Jim's boss wanted to discuss creating this team



The Team Today

- Well over a dozen people, 100% of their time upstream
- Not internal services – the PostgreSQL project
  - pgsql-hackers, the Postgres way, the Postgres processes
- For the last three years running, top three on every metric I track
  - #1 on many of them, #2 or #3 on the rest
- Best job I have ever had



Lessons

- Big companies can beat startups in important ways
- Happy people are productive people
- PostgreSQL is going strong
  - 30 years as an open source project
  - 40 years since inception
  - Here's to another 40!

The Chain

- Interested in computer programming and Engineering in High School
- ⇒ Mechanical Engineering at Virginia Tech
- ⇒ the degree took me into the submarine force
- ⇒ the submarine force took me from the east coast . . .
- ⇒ . . . to San Diego, where the boat was home ported
- ⇒ out of the Navy, I stayed ⇒ CarTech
- ⇒ CarTech is where I met my wife
- ⇒ a CarTech colleague went to Cymer ⇒ I followed



The Chain, Continued

- ⇒ at Cymer, my wife's brother introduced me to Postgres
- ⇒ Cymer's use cases made me a Postgres developer
- ⇒ and brought me into the Postgres community
- ⇒ community connections got me into credativ
- ⇒ more community connections got me into Crunchy
- ⇒ a Crunchy colleague got me into AWS
- ⇒ and the rest is history, as they say



The Butterfly Effect

- Break that chain anywhere and I'm not standing here
  - I might never have gotten involved in Postgres at all
- I'm thankful every link was there
- Think about that when you make your decisions
  - Any one of them can lead to great things
  - Be willing to take chances
  - Believe that things will ultimately work out



Questions?

Thank You!
conway@amazon.com
mail@joeconway.com

