TechChannel

How Collaboration and Education Can Help Close the IBM Z and COBOL Skills Gap

→ Dr. Cameron Seay on how we should be presenting COBOL and IBM Z to students

→ Herb Daly and Henri Kuiper on their Advent of Code challenge leaderboard

contents

Education, Collaboration and Fun: Closing the IBM Z and COBOL Skills Gap

It's clear that both the IBM Z platform and the COBOL programming language are here for the long haul, supporting mission-critical applications and key business processes around the world. But how do we maintain interest from new talent? The keys seem to be: education, collaboration and fun.

Dr. Cameron Seay, for example, has taught both COBOL and IBM Z at different schools. It's not that no one is interested in learning about them; it's just that the information needs to be presented to them in a way that highlights competitive advantages.

Beyond education—fostering a collaborative, fun community that encourages newcomers to join is another must. Take the Advent of Code Challenge, for example—an annual challenge comprised of programming puzzles for a variety of skill sets, all of which can be solved in any programming language participants want to use. Herb Daly and Henri Kuiper even created a leaderboard to increase playfulness and competition during the challenges—making coding fun.

In this e-book, you'll learn the importance of education, collaboration and fun in bridging the perceived mainframe and COBOL skills gap.

Keelia Estrada Moeller, Senior Editor

Who Says No One Is Learning COBOL? A Tale of 2 Schools

Dr. Cameron Seay on his experiences teaching COBOL and IBM Z at 2 different schools, and what can be learned from both models

BY DR. CAMERON SEAY

Indisputably, the global economy proves that COBOL still gets used. Almost every sizable bank, insurance company, retailer and government agency runs some of their key business process with COBOL code. The COBOL Working Group (CWG) of the Open Mainframe Project conducted a COBOL survey in the spring of 2021. From the data gathered, the CWG conservatively estimated about 200 billion active lines of COBOL code (see more on the full COBOL survey results).

I aim to refute the premise that college students oppose learning COBOL; they just don't know anything about it. I have taught mainframe technology since 2006, and I have never had a problem convincing students about the viability of the platform. Once you give students the facts—minus the hype—COBOL usually fascinates them.

I had the great privilege in the fall of 2021 to teach two undergraduate COBOL courses at two separate but similar publicly funded universities. The college of engineering at each respective school offered the class as an elective. One college housed it in the computer science department and the other in an information technology program.

Profiles of 2 Universities With COBOL Classes

One school I taught at is a historically Black college/university (HBCU) of about 7,500 students, based in a southern state. The Second Morrill Act initially funded this land-grant college. More than 72% of the student enrollment identifies as African American, with about 50% of students being first-generation—the first in their family to get a college degree.

The other school's enrollment totals around 30,000 students as a public university—but not a land-grant college. This predominantly white institution (PWI) has about 16% African American and 66% white student enrollment. For the rest of the article, I will refer to the HBCU as **"H"** and the PWI as **"P."**

Student Profiles

H's Department of Computer Science housed roughly 100 majors. The class itself consisted of 14 computer science (CS) majors, three business information systems (BIS) majors, one biology major and one mechanical engineering major. Mostly seniors took the class, with only two juniors and one sophomore. All but two students accumulated GPAs of at least 3.0. Three had GPAs of 3.7 or above. Most students first encountered the mainframe in this class. Two of them briefly encountered it during a mainframe bootcamp. The rest had never even logged on to a mainframe system.

At **P**, all 16 students came from the IT department of 1,500 students with a variety of concentrations: cybersecurity, networking, analytics and more. All **P** students had taken a class on mainframe basics like JCL, ISPF, TSO, etc., that I taught the previous semester. All but three had GPA of 3.0 or higher; a couple had GPAs of 3.7 or better.

The two schools contrast in an interesting way. Their endowments differ greatly. **H's** endowment is $63 million; **P's** is $212 million. **H's** enrollment totals about 7,500 and **P's** 29,000. As the larger and richer school, **P** has more resources to support its programs.

The students compete head-to-head for the same exact jobs—entry-level mainframe positions or internships. **H** faces the task of somehow offsetting the large disadvantage in resources and bodies while enabling its students to compete. HBCUs have always faced this challenge, yet they have done extremely well, all things considered. The HBCUs develop much more high-touch relationships. They must, because they exist to provide opportunity. Initially for the African American community, they now serve the entire public and admit students that other schools pass on.

The HBCUs make the best assessments they can beyond test scores and high school grades. Many times, they're correct, and the student flourishes. Sometimes, they're wrong, and the student does not.

Class Comparisons

P's students proved their technical proficiency; their IT concentrations all required hands-on work. They took numerous courses in Linux and networking, but interestingly, their curriculum didn't include relational database or programming courses.

Computer science majors made up the majority of **H's** class, and all had taken several programming courses. However, **H's** students had little hands-on experience. They could write code, which **P's** students struggled with. Yet, **H's** students could not build servers or configure a subnet mask, and **P's** students could. Again, **P's** students spent a semester being exposed to IBM Z.

Both classes used the same book, took the same exams and did the same labs. Both classes saw clear outliers—students with performance superior to their classmates'. **H** had three: a CS major, a BIS major and a major in mechanical engineering. **P** had two, both in the same IT concentration.

Since **H** did not have previous exposure to Z, we held a three-week crash course in Z basics: JCL, ISPF, creating data sets, etc. Both **H** and **P** gave an admirable first effort, but the **P** students more easily managed the nuances of Z. I actually gave the **P** students a two-week refresher in Z, but course evaluation responses deemed this unnecessary. The students thought it wasted time.

During the term, I had the classes hand-code several programs, meaning they just copied programs I gave to them. I intended for this to help them acclimate to COBOL syntax, a language unlike any they have worked with. This proved to be a mistake to avoid going forward. For the final project, I had them hand-code a program that calculated a GPA. I asked them to use the same code to read in three numbers and display the average and total of those numbers—a seemingly simple task. Surprisingly, both classes struggled with this. Two outliers at **P** and three at **H** didn't have a problem with it. My concentration on syntax and not having the students do problem-solving in COBOL earlier on likely caused this. I won't make this mistake again.

“It’s not that we have a shortage of students wanting to learn COBOL; we just need to be presenting it in the right way. ”

—Dr. Cameron Seay, Ph.D. adjunct instructor, East Carolina University

Summarizing the COBOL Class Results

By semester’s end, four of the **H** students received mainframe-related job offers (another was already working in a mainframe job), but as far as I know, none of the **P** students were extended any offers. It should be noted here the **P** was the much more high-profile school and getting work was not a real concern of the graduating students. At **H**, however, like most HBCUs, getting a job was a real concern, and the placements were deeply appreciated.

This experience causes me look forward to working with both schools again in the future. Ultimately, my key takeaways from this are:

1. **COBOL is still very attractive to students if is presented correctly.**
2. **Learning COBOL can make students at HBCUs extremely competitive.**
3. **The concern of finding enough COBOL programmers to support existing and new COBOL programs can be addressed if there is sufficient willingness among decision makers to expose students to it correctly.**

In other words—it’s not that we have a shortage of students wanting to learn COBOL; we just need to be presenting it in the right way.

DR. CAMERON SEAY is a Ph.D. adjunct instructor at East Carolina University.

Overcoming Unconscious Biases Around COBOL and Women in Technology

BY KEELIA ESTRADA MOELLER

TechChannel recently launched a video series featuring conversations with women of COBOL, moderated by (and the brainchild of) Misty Decker, product marketing director, Micro Focus.

Decker's broad expertise within the mainframe ecosystem, along with her dedication to correcting misinformation and shifting unconscious biases, ultimately led her to think about how she could help combat misinformation surrounding women in technology, the mainframe and COBOL itself.

Episode 1 released in mid-November, and featured Decker herself, along with TechChannel Senior Editor Keelia Estrada Moeller and Open Mainframe Project/ The Linux Foundation Senior PR Manager Maemalynn Meanor. The three discussed what misinformation they've each encountered surrounding COBOL, and how they're working to dispel these misconceptions.

The next session in the series will focus on mainframe skills, highlighting insight from Melissa Christie—a mainframe student and a North America winner of Master the Mainframe 2020. Christie submitted her final project on COBOL as well. Alongside Dr. Gina Bullock of NC A&T, who teaches COBOL—who Decker is still working to finalize as a speaker—Christie will discuss her student experience and have a chance to ask this educator questions.

The third session will highlight modernization, featuring Susan Drennan from Micro Focus, who has many years of experience helping customers modernize their COBOL applications, along with Marianne Bellotti—the author of "Kill It with Fire: Manage Aging Computer Systems (and Future Proof Modern Ones)," a book about neglected legacy IT and modernizing those systems.

Watch the video series

A New Way to Play With COBOL: Following the Leaderboard in the Advent of Code Challenge

Herb Daly and Henri Kuiper on their Advent of Code challenge leaderboard, the importance of fun in IBM Z engagement and what's next for their collaboration

BY DAVA STEWART

When **Herb Daly**, an academic based at the University of Wolverhampton in the U.K., attended an Educator Council hosted by **Guide Share Europe** (GSE) in 2012, he expected to enjoy a few days "somewhere sunny"—on the French Riviera, to be exact—and to enjoy networking and learning with other IT professionals. He didn't expect to meet a lifelong friend and collaborator, but that's what happened. Daly and **Henri Kuiper**, mainframe security engineer at a Dutch governmental organization and founder and owner of **zdevops**, have been collaborating ever since they met that year, bringing a sense of playfulness and excitement to their projects that is so important for maintaining engagement at mainframe conferences and other events.

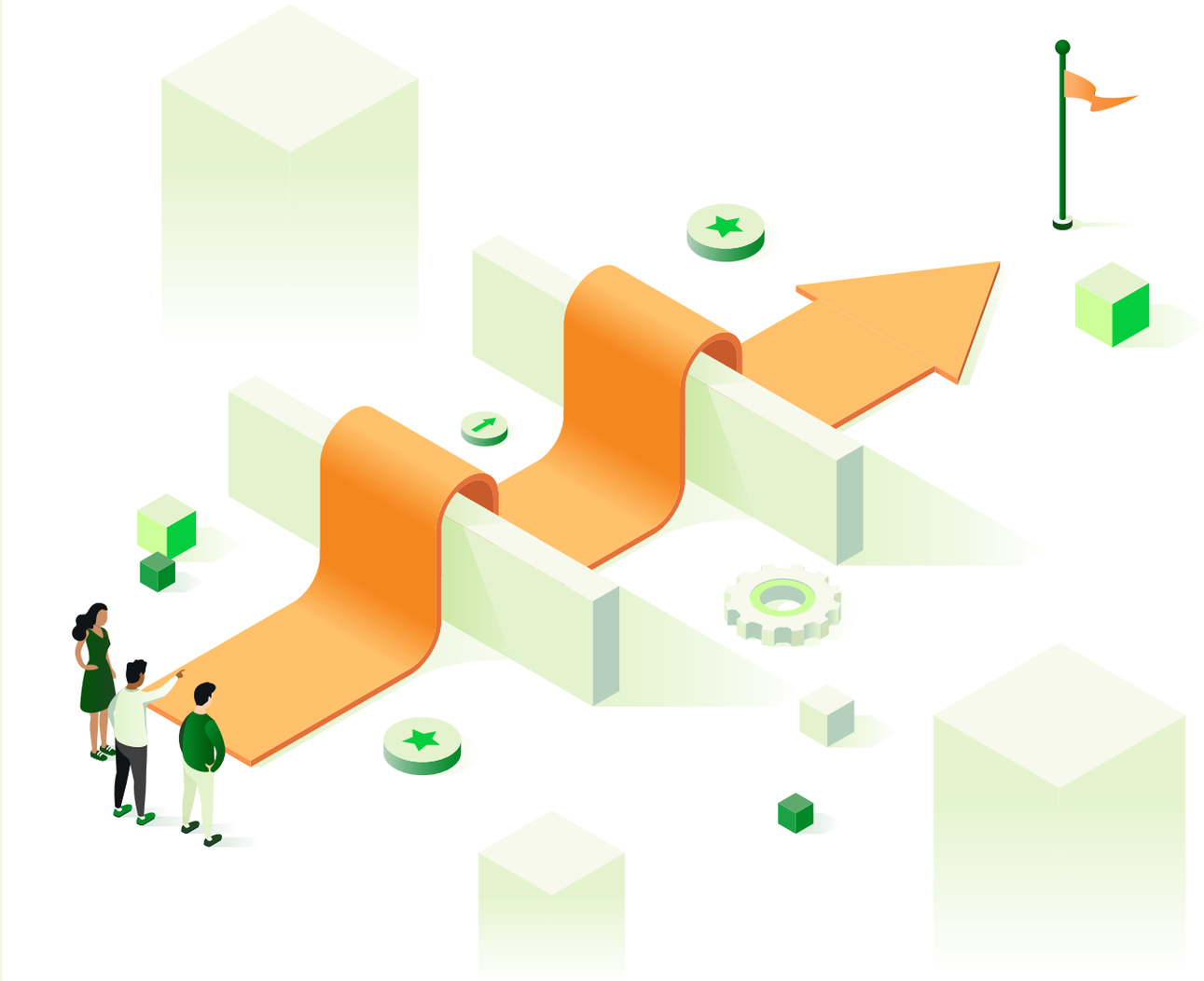
Their most recent collaborative project involved creating a leaderboard for those who wanted to participate in the 2021 **Advent of Code challenge** using programming languages associated with the mainframe, such as COBOL, REXX, PL/1 or Assembler. Advent of Code is an annual advent calendar comprised of programming puzzles that appeal to a variety of skill sets and skill levels and can be solved in any programming language participants want to use, according to **Eric Wastl**, the creator.

A Global Contest

The Advent of Code challenge is popular, with thousands of participants. The first person to solve the puzzle of the day gets 100 points, the second 99 points and so on. The 201st person to solve and all those after are awarded a star. People, including those without application development experience, use a wide variety of approaches to solving the problems. “Some are even solving the puzzles in **SQL** and Excel,” says Kuiper.

Here’s the introduction to the first puzzle of 2021:

> “You’re minding your own business on a ship at sea when the overboard alarm goes off! You rush to see if you can help. Apparently, one of the Elves tripped and accidentally sent the sleigh keys flying into the ocean!
>
> “Before you know it, you’re inside a submarine the Elves keep ready for situations like this. It’s covered in Christmas lights (because of course it is), and it even has an experimental antenna that should be able to track the keys if you can boost its signal strength high enough; there’s a little meter that indicates the antenna’s signal strength by displaying 0-50 stars.
>
> “Your instincts tell you that in order to save Christmas, you’ll need to get all fifty stars by December 25th.
>
> “Collect stars by solving puzzles. Two puzzles will be made available on each day in the Advent calendar; the second puzzle is unlocked when you complete the first. Each puzzle grants one star. Good luck!”



The story goes on to present measurements the submarine takes of the ocean floor using sonar and asks how many measurements are larger than the previous measurement. The answer is the solution to the first puzzle. Users are invited to identify themselves on GitHub, Google, Twitter or Reddit through **OAuth**.

Making Coding in COBOL Fun

Daly and Kuiper say that the puzzles are sort of a Rosetta Stone of programming languages because “the problem is the same, the data is the same, the functional result of the solution must be the same or otherwise you don’t

score your stars, but you know in JavaScript it's four lines. In COBOL it's 50 lines; in REXX it's 52 lines," says Kuiper.

One thing Daly and Kuiper noticed, though, was that not everyone seemed to be using languages associated with the mainframe. They attribute this to the fact that mainframe languages are often treated very formally—but the challenges give users the freedom to show their skills and approach the solutions more creatively.

"We hoped that the novelty of the challenge would encourage people to be playful with mainframe languages," says Daly. Playfulness factors into their collaborations often. They cite Portia Tung, who runs **School of Play**, where "happier adulthood through lifelong play" is promoted, as an inspiration.

"When you learn to skateboard, there's no curriculum, right?" notes Daly. "You keep falling, and you keep getting back up. You're with your friends and you're all doing the same thing, and you're having fun. We saw Advent of Code as a way to encourage the mainframers to engage more playfully with mainframe languages."

Private Leaderboards for Specific Groups

Just as participants use a variety of approaches to solving the daily challenges, they also use the Advent of Code challenge itself in different ways. Some people, particularly those who dominate the global leaderboard, see it as a speed contest. Others consider it a way to prepare for interviews, complete company training, practice skills or as part of university coursework. Some, of course, just consider it good fun.

Because there are so many reasons to participate in the Advent of Code challenge, and because the puzzles are published at midnight Eastern time, which can be difficult for participants in other time zones who want to enjoy the competitive factor of the challenges, private leaderboards came about.

Private leaderboards have more options for how points are awarded, and participants have to have a code in order to join. Each participant can join up to 100 private leaderboards. Regardless of where they are competing, all of the puzzles are the same. "It really brings people together to jointly discuss the same thing," says Kuiper. "You know you're talking different languages, different solutions, different technologies. In the end, it's all the same. That's pretty amazing." See Figure 1 for a visual of this year's leaderboard (featuring the Top 20).

This was the second year Daly and Kuiper ran the leaderboards for mainframes. They had significantly more participants than in 2020, partly due to their efforts in getting the word out about it, and partly due to having a sponsor (**Micro Focus**), which meant prizes for the winners.

And the Winners Are ...

The winners for the 2021 challenge (selected by Daly and Kuiper based on the leaderboard based on accumulated points) include:

1. **Galois Girl**
2. **Craig Schneiderwent**
3. **COBOL-Erik**
4. **Johansorlin**
5. **MalachiHealy** and **https://pastebin.com/r4HMUd30**

Each winner received the following prizes (donated by Micro Focus):

- "Grace Hopper Queen of Computer Code" children's book
- Blue shopping tote bag
- Digital skip rope
- Blue ceramic mug
- Brain teaser game
- LED retractable charging cable
- Power bank
- COBOL 60th birthday celebration buttons

Figure 1. Top 20 on the Advent of Code leaderboard

Other noted mainframe language solvers this year included:

1. **Dougie Lawson**: PL/1
2. **Wizard of z/OS** (Kuiper himself): REXX and Python on z/OS. You can also **see videos** of Kuiper solving the puzzles on his Advent of Code YouTube Channel.

A Growing Effort

Daly and Kuiper are already planning for next year. "There are a lot of things that we've learned that we'd like to do different next year," says Kuiper. For instance, they want to have more of an onboarding process somewhere so that they have a better idea of who the competitors are and can avoid the problem of "leaderboard squatters."

"We were in something of a hurry this year," says Daly. "Now we have some examples so we can talk about it in advance next year and we'll have a much longer runway. It'd be nice to get more people every year, and the challenge changes every year." He says that participating in a month-long event like the Advent of Code challenge is a bit like running a half-marathon; most people think they want to for a long time before they actually commit.

Special Report: COBOL Survey Results Prove Pervasiveness, Value and a Bright Future

BY REG HARBECK

Last year, the COBOL Working Group (CWG) of the Open Mainframe Project (OMP) rolled out its COBOL survey after weeks—months, even—of careful consideration and design. The survey aimed to cover:

- Job role and experience level of the respondent and willingness to discuss their answers further
- Respondent's organization's size and category (e.g., finance, manufacturing, distribution, retail, government, etc.)
- COBOL context of their organization—relative and absolute size of COBOL portfolio, current and projected shrinking and expansion of COBOL presence, and location and sufficiency of current and anticipated COBOL development workforce

Analyzing the data turned out to be a major undertaking. But weeks later, a compelling picture began to emerge of the world of COBOL today and in the foreseeable future. Three particularly significant results emerged:

1. 250 Billion Lines of COBOL in Use

Based on our extrapolation, there are 250 billion lines of COBOL in production use worldwide. That's apparently the largest it's ever been—and slightly larger than the 240 billion that some estimated in the 1990s. Some folks take this to mean that COBOL's not going anywhere, while others point out that it is indeed going somewhere—just not "away."

Figure 1, below, is an illustration of this from one segment of the COBOL-using world—the financial services industry, broken up by number of lines of COBOL per individual organization surveyed. At the extremes, more than 15% of such organizations had fewer than 1 million lines of COBOL, while slightly more than 7% each had over a billion.

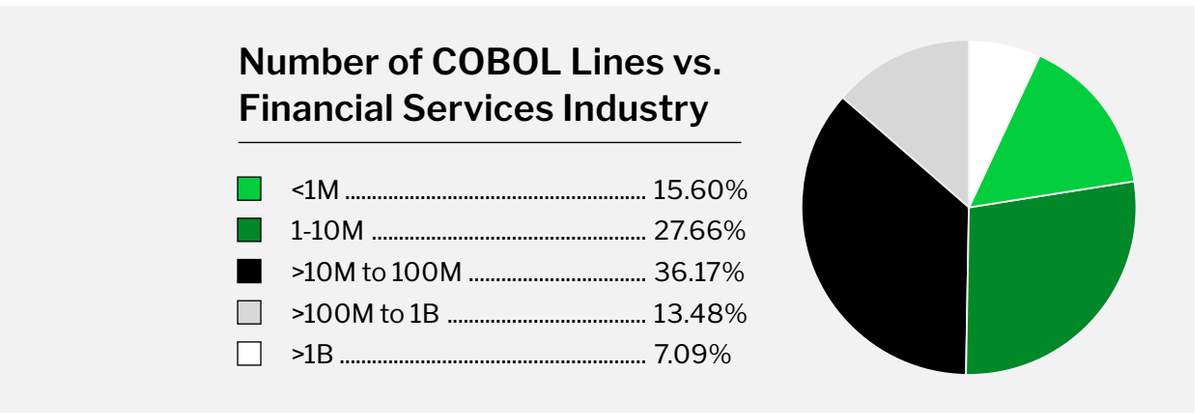


Figure 1

2. The Future of COBOL Is Bright

The next major finding demonstrates that the future of COBOL is recognized as relevant and persistent—with fully 58% of respondents willing to admit that their COBOL applications will be sticking around for the next half decade or more.

3. The COBOL Skills Challenge

Given how COBOL was designed to be easy to read and write, the existence of a skills challenge is almost funny. The language was designed not to require advanced computing abilities to learn and develop with it. Organizations that have their own internal training programs need not spend their time worrying where they'll find new COBOL programmers—they can just hire competent and willing new staff and get them proficient and functioning in an acceptably finite time.

Get the full COBOL survey results

TechChannel

TechChannel.com is home to a variety of content to help you get started on your IT journey.

Subscribe for the latest TechChannel content

This e-book was published by

TechChannel

953 Westgate Drive, Suite 107, St. Paul, MN 55114 // (612) 339-7571

Staff list

Senior Editor: Keelia Estrada Moeller

Copy Editor: Emma Pitzl

Art Director: Kelsey Hanscom

Senior Marketing Manager: Noelle Heaslip

Audience Development Director: Linda Holm

Publishing Director: Mari Adamson-Bray
(612) 336-9241 // mbray@techchannel.com

Account Executive: Darryl Rowell
(612) 313-1781 // drowell@techchannel.com

Sr. Strategic Account Manager: Lisa Kilwein
(480) 428-9780 // lkilwein@techchannel.com