

# Have we a Human Ecosystem?

James Davenport<sup>1</sup>  
masjhd@bath.ac.uk

Departments of Computer Science and Mathematical Sciences  
University of Bath  
Fellow, Software Sustainability Institute

**HIS 2022**  
HIGH INTEGRITY SOFTWARE CONFERENCE

**#HISConf2022**  
[his-conference.co.uk](http://his-conference.co.uk)

---

<sup>1</sup>Many thanks to Capgemini staff and members of SC<sup>2</sup> [ABB<sup>+</sup>16],  
H2020-FETOPEN-2015-CSA 712689



**Anna Baas** @venite · 2h

A friend learned COBOL and received a codebase where the last change was done in the 90s... by his. mum.



**nobody** @imaguid · 20m

that's not how inheritance is supposed to work in programming



# Why am I talking about COBOL here?

Because in 2020 the USA ran out of COBOL programmers (as well as masks, ventilators etc.).

- 1 New Jersey Gov. Phil Murphy has issued a call for volunteers who know how to code in the 60-year-old COBOL. “Literally, we have systems that are 40-plus-years-old”.
- 2 Connecticut has also admitted that it’s struggling to process the large volume of unemployment claims with its “40-year-old system comprised of a COBOL mainframe and ...”
- 3 In 2008, then-California Gov. Schwarzenegger wanted to issue minimum-wage checks to 200,000 state workers, but the state’s controller said it would need at least six months to reprogram California’s COBOL-based payroll system to issue them.
- 4 A 2018 report by the inspector general for the Social Security Administration found that the administration maintained more than “60 million lines of COBOL” (probably not unique lines).
- 5 A friend got hired back at “>\$1000/day” for COBOL maint.

# COBOL (continued)

Over half the states found their COBOL systems couldn't cope with the volume of claims. [She20]

"IBM is addressing the call for skilled COBOL coders by working in tandem with the Linux Foundation's Open Mainframe Project to create three new initiatives to address the immediate need:

- 1 **Calling all COBOL Programmers Forum.** A new forum where developers and programmers who would like to volunteer or are available for hire can post their profiles and credentials.
- 2 **COBOL Technical Forum.** A new resource for being actively monitored by experienced COBOL programmers providing free advice and expertise.
- 3 **Open Source COBOL Training.** A brand new open source course designed to teach COBOL to beginners and refresh experienced professionals. IBM partnered with the clients and universities to develop this in-depth course which is available for free to anyone".

## But we shouldn't mock the Americans

[Fin19] reports a local fiasco.

*Ulster Bank's ageing systems make it impossible for the lender to routinely charge large corporate customers in the Republic of Ireland for deposits that run into millions of euros, according to two people briefed on the issue.*

*"You wouldn't believe how hard it is to change".*

*Ulster Bank has implemented a manual workaround that allows it to levy an ad hoc charge on a small number of the largest depositors.*

COBOL integers are decimal and unsigned by default, and a signed integer (PIC S99V99) takes more space than an unsigned.

Remember there's no `#include`.

Contrary to [Tho19], this isn't "programmers being unnecessarily specific": I'd have written the same code in 1975.

# COBOL: Why so difficult

After all, similarly-aged languages like FORTRAN don't have this problem.

- 1 The FORTRAN ones tend to be in active development (e.g. Met Office).
- 2 COBOL has no equivalent of `#include` (and a preprocessor providing it was my firm's most successful product in 1975), which gives a technical debt problem.
- 3 COBOL has no equivalent of `#define`, so instead of  
`#define BATCHSIZE 50000`  
you have 50000 (and 50001, 49999 etc.) throughout the code.
- 4 Y2K people wrote lots of *ad hoc* tools



Who knows what my successor in 40 years time will write about SPARK/cvc-5/Isabelle/..., or even about C?

*We've tried to migrate using tools to port from mainframe code to Java. But you end up with Java code which looks like COBOL, so your Java programmes can't understand the code and your mainframe programmers can't change it. [Sum22]*

The problem is the lack of a human ecosystem to understand the technology.

# And it's not just the language

A friend (aged ~72) works for a major bank, maintaining mission-critical systems.

40 year old system: finally decommissioned.

30 year old system: now runs on Linux hardware, so the Board aren't worried any more.

20 year old system: I can't get anyone interested in updating this.



Though I'm the only person who knows how it works, and there's no succession plan.

The fundamental problem isn't the technology, it's the management attitude to software, and the lack of "preventative maintenance" (which would keep the maintenance team current). Also really poor human succession planning.  
"Are there zombies lurking in your bank?" [Lum23]



Italian railways are migrating away from Relay-based Railway Interlocking Systems (RRIS).

- 1 The factual documentation is on manually drawn paper diagrams (in a mixed state of preservation).
- 2 The intellectual documentation (“why” etc.) is in informal documents and engineers’ heads.



“Moreover, the work on these documents requires a strong legacy domain knowledge which is vanishing in these days” [ABC<sup>+</sup>20]

B 1 is “solved” via scanning, editing *and* verifying [ABC<sup>+</sup>22]

A 2 is solved via capture with Controlled Natural Language [ABC<sup>+</sup>20]

# Italian Railways Workflow [ABC<sup>+</sup>20, Figure 1]

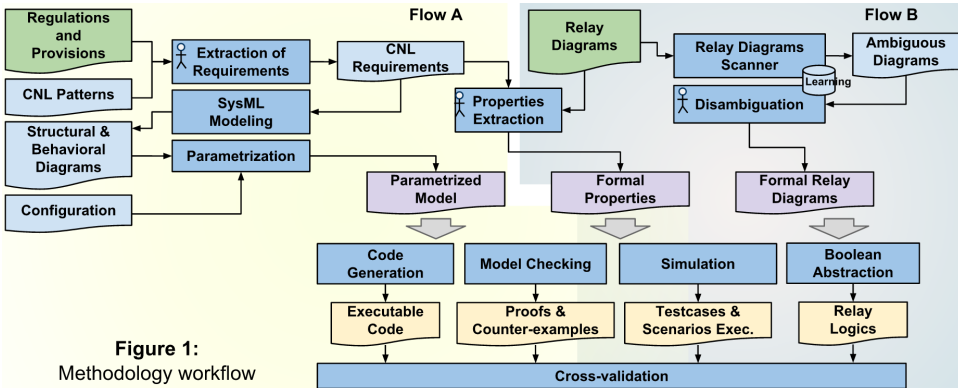


Figure 1:

Methodology workflow

Hopefully our successors will not be building anything this complex to understand our artefacts, but how sure can we be?

B We can reasonably hope that our artefacts are fully formal.

A But “strong legacy domain knowledge”?

# Paris Métro Line 14 [D'22]

1998 Opened; fully automatic; critical software verified by Atelier B. **major showcase.**



This software (rather than the construction process) was signed off, with no change process [D'17, Minute 14:30].

2003,2007 Extensions opened (but already in the verified software).

2010s Planning for new extensions, not in software.



Atelier B only runs on Sun workstations, ported to Linux (but not necessarily 100% identical, and previous software didn't verify out-of-the-box).

2020 New extension, with new trains running 2× faster.



The new trains were 8 carriages, not 6, so not every door opens at every station; significant modification.



But it's running formally verified GoA4 [Cue20].

- 1 Cobol: Do you keep programming in a sufficiently similar language?
- 2 Atelier B: Is your proof infrastructure still maintained?
- 3 Ligne 14: Just because a theorem verifies automatically today doesn't mean it will verify *automatically* in 20 years time. Is it worth/possible recording some major steps of the automatic proof, rather than “relying on upwards compatibility”





This may be more serious as more machine learning (sledgehammer etc.) is used in proof systems.

- 4 Italy: Are you reliant on “strong legacy domain knowledge”?



“The Empire can't maintain anything” — Hober Mallow [Asi44].

-  E. Abraham, B. Becker, A. Bigatti, B. Buchberger, A. Cimatti, J.H. Davenport, M. England, P. Fontaine, S. Forrest, D. Kroening, W. Seiler, and T. Sturm.  
SC<sup>2</sup>: Satisfiability Checking meets Symbolic Computation (Project Paper).  
*In Proceedings CICM 2016*, pages 28–43, 2016.
-  A. Amendola, A. Becchi, R. Cavada, A. Cimatti, A. Griggio, G. Scaglione, A. Susi, A. Tacchella, and M. Tessi.  
A model-based approach to the design, verification and deployment of railway interlocking system.  
*In Proceedings International Symposium on Leveraging Applications of Formal Methods ISOLA 2020*, pages 240–254, 2020.



A. Amendola, A. Becchi, R. Cavada, A. Cimatti, A. Ferrando, L. Pilati, G. Scaglione, A. Tacchella, and M. Zamboni.  
NORMA: a tool for the analysis of Relay-based Railway Interlocking Systems.

*Tools and Algorithms for the Construction and Analysis of Systems TACAS 2022*, pages 125–142, 2022.



I. Asimov.

The Big and the Little [reprinted as “Merchant Princes” in *Foundation*, Gnome Press, 1951].

*Astounding*, 33:7–54, 1944.



A. Cimatti.

Migrating Railway Systems.

*Personal Communication at Dagstuhl February 2022*, 2022.



O. Cuenca.

Paris inaugurates Line 14 extension.

<https://www.railjournal.com/passenger/metros/paris-inaugurates-line-14-extension/>, 2020.



D. Déharbe.

Talk at SCSC Summer School.

<https://vimeo.com/showcase/5271198>, 2017.



D. Déharbe.

Évolution de la ligne 14.

*Personal Communication 11 August, 2022.*



Financial Times.

Ulster Bank's computer says no to negative rates.

<https://www.ft.com/content/ff12856a-e07d-11e9-9743-db5a370481bc>, 2019.



L. Lumley.

Are there zombies lurking in your bank?

<https://www.thebanker.com/Are-there-zombies-lurking-in-your-bank-1700640547>, 2023.





E. Shein.

COBOL programmers are in demand to fight the coronavirus pandemic.

<https://www.techrepublic.com/article/cobol-programmers-are-in-demand-to-fight-the-coronavirus-2020>.



S. Sumner.

Computing gathered a group of UK-based IT leaders together to discuss their legacy concerns, including security, skills and cost issues.

<https://www.computing.co.uk/news/4057289/cios-discuss-legacy-issues-dont-write-mainframe,2022>.



M. Thomas.

Ethics, old software and negative interest rates.

*Letter to Financial Times 4 October, 2019.*