

Mainframe programming and the modern developer

Published January 2017

For those of you who don't know – which, to be honest, is probably most of you – before I started working with Bloor Research, I worked as a software developer for a relatively modern, forward-thinking tech company. I enjoyed the comforts and luxuries afforded by most modern-day programmers – IDEs such as Eclipse and Visual Studio, for instance – as well as then new, up-and-coming tools such as Docker and Selenium. I even worked in an innovation lab for about six months of my career, in which my job description was to – essentially – play around with interesting new technologies. I mention all this to emphasise the fact that, as far as my development experience is concerned, I'm used to the newest, the most innovative and the most modern. Mainframes, COBOL and job control language simply weren't a part of my vocabulary: if you had asked me when I was a developer, I would have said, unequivocally, that I could never work with a mainframe.

In fact, that would have been true up until only a few weeks ago, when Compuware showed me Topaz and, in particular, Topaz Workbench. I won't discuss it in detail, but Topaz is a suite of mainframe development and testing tools designed to help developers of varying skillsets understand and work on any program, no matter how old or complex. Topaz Workbench, a modern Eclipse-based IDE that's available at no additional cost to maintenance paying customers of Compuware's solutions, allows you to access the company's full range of developer, testing and maintenance tools, as well as non-Compuware products and distributed solutions. Of course, just working inside Eclipse is a big advantage, but what's more, having seen it in action, it's remarkable just how easy it is to use. It doesn't have a learning curve, there's no new language or set of commands or syntax you have to learn, or two-hundred-page manual you have to read: you just use it like any other tool (in this regard, it's actually much better than many modern tools, including Selenium and Docker, but that's beside the point). It makes mainframes accessible in a way that they traditionally aren't, particularly to modern programmers like myself who have barely touched a command line, let alone a COBOL copybook.



Compuware

1 Campus Martius
Detroit, MI 48226 USA
Tel. (313) 227 7300

www.compuware.com

The Topaz suite of products, including Topaz for Total Test, the newest product Compuware just released – a system for unit testing COBOL programs – make it clear that Compuware is hellbent on bringing mainframe programming kicking and screaming into the 21st century. Given that many 20th century mainframe developers are already, or soon will be, retired this is not just good, it's absolutely necessary. As an analyst, I welcome this as a move in the right direction; as a developer, I can even imagine working in a mainframe environment – and that's saying quite a lot.

Daniel Howard

Associate Analyst