What are current users of legacy SCM solutions for the Mainframe looking for?

… they want to Modernize!

Although all current customers recognize the great value that legacy solutions like Serena ChangeMan, CA Endevor or Harvest, Compuware ISPW and IBM ISPW gave them, today they want to make the next step and they want their SCM (Software Change Management) to be Modern, adapted to the current software development and deployment practices.

What are the requirements of a modern SCM solution?

Moving from Waterfall to Agile

We personally don't see this as a religion as both methodologies have their pros and cons. And some Agile principles can easily be applied in a current mainframe setup.

When people say they want to move from waterfall to agile, moving to a more modern tooling is what they usually mean. They see the mainframe as an isolated dinosaur and that dinosaur needs to be a part of their enterprise IT and they want reuse (standardisation) of tooling. And maybe some projects need to use the waterfall methodology and others projects need Agile.

Modern IDE's

Also mainframe customers want to move from text based editors like ISPF (although we think this is a great editor).

Why? The modern IDE's that are Eclipse based or IDE's like VS Code offer standard integrations with productivity tools and are important to attract young graduates.

Integrate with other (open source) tools and standards

That brings us to our second point. What people really want is a kind of "plug and play" sustainable solution. Example: the current legacy SCM systems have their own proprietary source version control. Enterprises want standards. Standards that evolve and mature. A few years ago, CVS was the go-to version control repository, soon it got replaced by Subversion, where today Git is the de facto standard.

Git, by example, is not just a standard, but it is also open source (free/no cost) and widely accepted in both the distributed world and having attraction in mainframe land. For issue tracking and project management Jira seems to be the standard.

Same for testing. There are both open source and commercial solutions that can be used for just distributed or mainframe or both. Like SonarQube.

Finally, as they want sustainability they want to make sure that if Git has a successor they can easily move without having to change everything.

Parallel development

Modern development means that you want to have the ability to do parallel development.
Modern version control solutions like GIT make this possible through their branching.

You can work on as many releases as you want. You can have a major new release, together with a minor upgrade and the ability to fox bugs what is currently in production.

Cost effective

Cost effective doesn't mean that everything is for free. Even Open Source comes at a cost: implementation, maintenance, management,.. and you need skilled resources.

What customers want is an ideal cost effective mix that gives the best value for money. And they are willing to spent money as in the end they want quality, minimal risk, cost effectiveness and a fast time to market.

Enterprise solution

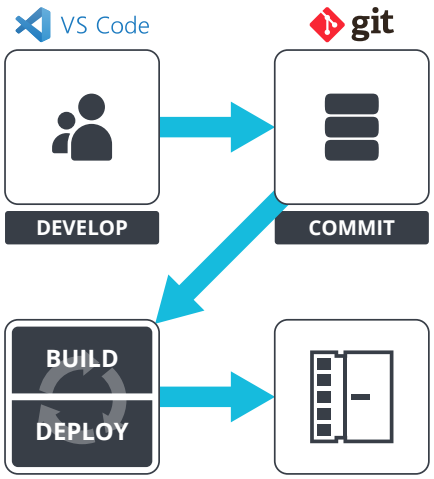
The ideal scenario is that for each step in the process you can have one solution, that can be used enterprise wide.

Like Git for version control and this for all kind of sources you may have, including your mainframe programs.

What are mainframe customers missing here?

Again, in our opinion mainframe environments are not that different from other environments.

You develop code (COBOL, Assembler, ...) using a modern IDE , you version the code (in Git) and then, just like in Java you need to build and deploy that code.



How does that work?

Mainframes are using JCL (Job Control Language) as scripting language to build and deploy your code.

Today there are solutions that allow you to build and deploy your code on non-mainframe platforms (Windows or Linux) or that allow you to do this on the mainframe, all starting from Git.

But first some mainframe lingo.

Mainframes use PDS's or Partitioned Data Sets to store sources and executables. A PDS is mostly the same as a Windows directory with a number of files.

Some other naming differences between the distributed and mainframe domain are:

Distributed	Mainframe
Executable	Load Module
Build	Compile
Deploy	Promote
Start build	Submit (Sub)

In other words to have a modern SCM process (nowadays referrerd to as "CI / CD") for mainframes you need to:

1. Transfer the sources from Git to a PDS
2. Generate the compile and promote JCL
3. Submit and then run the JCL (job) on the mainframe
4. Retrieve the results (compile listing, load modules,..)

Kobee

Kobee is a CI/CD solution that meets these requirements.

Kobee makes it easy to integrate your mainframe into a contemporary software development and deployment way of working.

It's more than just an alternative for:

- Serena Change Man (now Micro Focus Change Man ZMF)
- CA Endevor
- Harvest (now Broadcom Endevor or Harvest)
- Compuware ISPW (now BMC ISPW)
- CA Panvalet
- CA Librarian
- IBM SCLM

We offer a full implementation service. All we need is the current version of you desired compile and deploy JCL's.

- Easy integration with tools like Jira, Git and testing tools
- Project setup:
  Branch based project streams, where each project stream has his own life cycle with a build level, one or more testing and production levels.
- Fully customisable Build and Deploy environments using:
  - Predefined Phases for compile and promote with predefined models for JCL cards
  - Release or Package based (just these sources you want, not a full branch) build and deploy
  - Machine, Environment or Phase based parameters
- Pre- and Post-approval management
- Easy to setup and use (web based interface)
- Can be used for non-mainframe environments too. By importing dedicated Phases (Build and Deploy scripts)
- Project dependencies:
  You can have separate projects, let us say one for mainframe COBOL, one for Java. The dependency features makes sure that both travel through the life cycle together.

Kobee is a commercial yet cost effective solution trusted by other companies such as:

REALE MUTUA VIVIUM ASSURANCES mediaocean

Contact us

info@ikan.be
www.kobee.io

IKAN Development
Motstraat 30
2800 Mechelen, Belgium
Tel. +32 15 238427

info@kobee.io
www.kobee.io