Simplifying CICS Configuration Harness the Power of Code

Stewart Francis
stewartfrancis@uk.ibm.com
CICS TS Development Lead and Developer Productivity

Why we’re doing this…

Skills

Simplification

Practices

Challenges facing the next generation of CICS system programmers

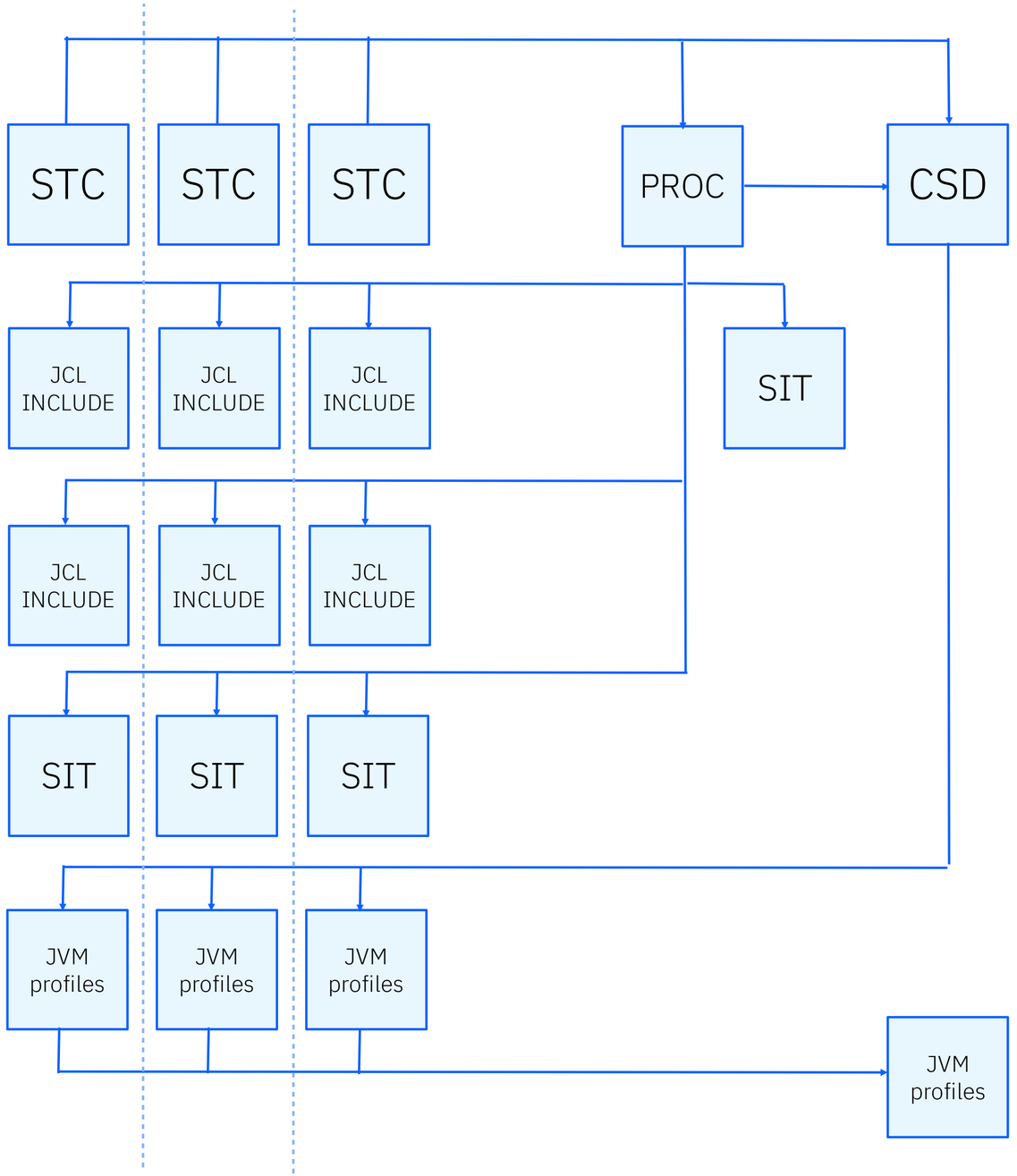
- Takes a long time for early tenure system programmers to become comfortable with the platform
- Not something most universities teach

AS-IS SCENARIO

Stan teaches Julian

CICS system programmer Stan is teaching his junior colleague Julian about how CICS regions are built, configured and managed.

He talks Julian through the complicated web of JCL PROCs, shared and region-specific CSD groups, SIT parameters and zFS configuration files that his organization have created.

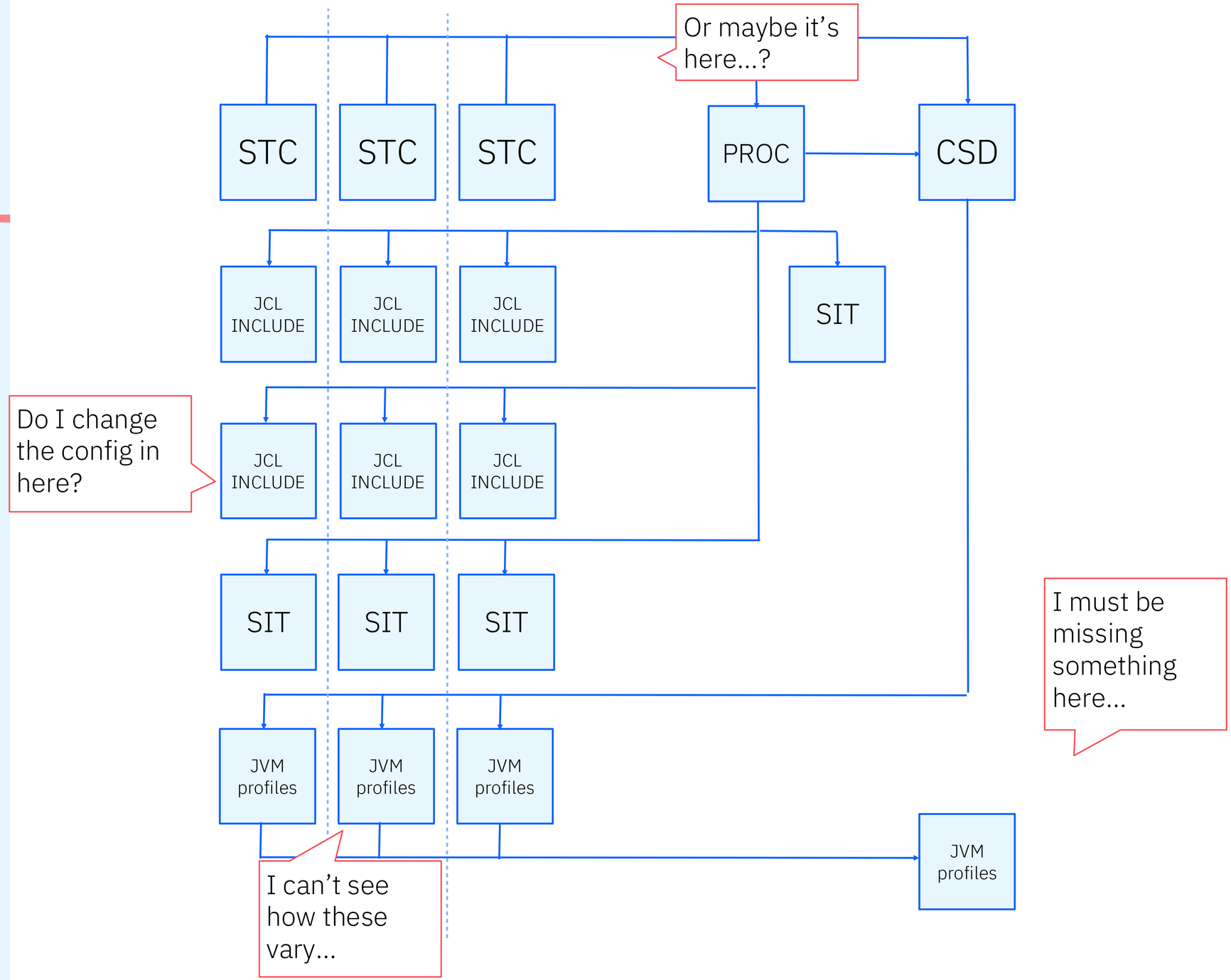


AS-IS SCENARIO

Complicated system

Julian finds it hard to understand the way the complex interrelated systems are put together, and as he's new to z/OS, all the concepts are new to him too.

He finds that he can't look at the system and understand or reason about what will happen at runtime — there is too much complexity locked up in the configuration.



How does YAML config help?

- Design a new configuration model around technology that is already widely adopted in the industry
- Standardize divergent configuration formats into a single unified model (e.g. CSD vs SIT)
- A single high-quality editing experience for all configuration
  - Validation
  - Content assist
  - Integrated documentation
- Introduce new higher-level config options
- Standardize the way configuration is shared between instances
- Rename config options to make it clearer what they do
- Deprecate and remove config options that aren’t used
- Easy to put in Git…

Reminder:
what is configuration-as-code?

Configuration-as-code is the practice of **handling changes, deployments and updates to software and systems via code** rather than manual processes, allowing for a version-controlled, reproducible and **automated method of managing environments**, leading to more consistent and reliable systems.

The essence of configuration-as-code is to treat **system configurations the same way that software code** is treated. This means that configurations are written in a code format, stored in a version control system and applied through automated processes.

This approach brings several benefits, including **consistency, repeatability and the ability to test changes** before applying them to live environments.

https://devops.com/configuration-as-code-trends-and-predictions-for-2024/

Reminder: Core principles of configuration-as-code

Consistent configuration outcomes

This principle means that applying the same configuration multiple times will always result in the same system state. Whether you run the configuration once or several times, the result remains unchanged. This eliminates the risk of unintended changes or configuration drift.

Declarative definitions

This principle is designed to describe what the final set-up should look like, rather than relying on writing out each step that the system should take. This leads into declarative specifications and makes configuration easier to read and maintain.

Version control

This principle allows tracking changes, identifying who made a particular change, when it was made, and why. It also enables rollback to a previous configuration state if necessary. Using a modern SCM means that no changes are made without review.

Continuous validation

DevOps and CI/CD pipelines allow for code to be iteratively changed and reviewed, continuous validation allows for regular testing of the configuration code to ensure it’s correct and effective, so validation checks can be made automatically and run whenever a change is made.

Drivers for configuration-as-code

An early tenure systems admin can easily stand up and maintain middleware regions in a manner that's consistent across the industry.

A development team can easily stand up a test environment for their functional, regression, performance tests without extensive setup by system admins.

Auditability

Storing configuration in a source code management system allows for an audit trail through any changes that are made, pull requests verified, and a history of changes to a particular system.

Rollback

The ability to gather previous versions of configuration allows for agile and quick rollbacks during any changes made to the system.

Consistency

For the IBM Z stack, the formatting for configuration-as-code is being published as a consistent set of YAML documents across the ecosystem, so a developer and system programmer will have the flexibility to work across the entire stack.

Skills

The skills needed to edit and make changes to the configuration will be industry-standard, which will make it easier for early-tenure professionals to understand and learn how to use IBM Z.

Simplifying configuration for the z/OS stack

APPLICATION

Application source code

Application middleware configuration



Deb Early Tenure App Dev

Angie App Architect

Rohan DevX Engineer

OS & SOFTWARE

Middleware subsystems

z/OS



Zach Sr z/OS Sys Prog

Chris Jr z/OS Sys Prog

David DBA

SYSTEM

Firmware

Hardware


Greg Infra Architect

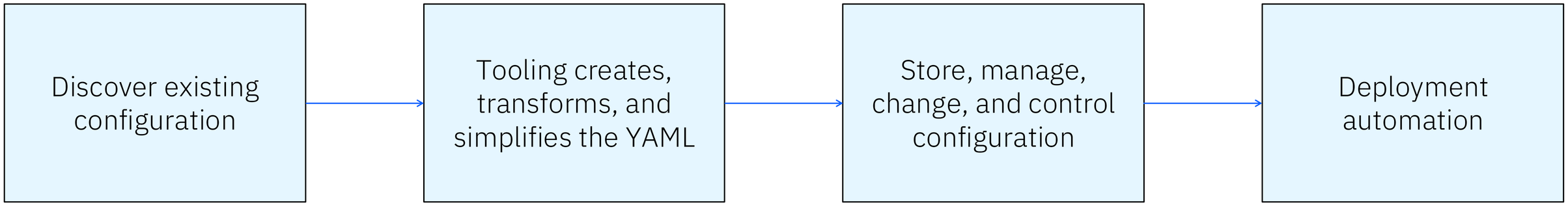
Luis IT operator

Addressing configuration complexity was ranked the highest priority by customers because of the current complexity across the stack and its impact on getting a system set up, the overall learning curve and in complicating problem determination and resolution.

> “It’s scary to admit but I don’t really know how my system is configured.”

> “It can’t just be PARMLIB. It has to be the full stack.”

IBM Z Vision for configuration-as-code



CICS TS Resource Builder

- Provides YAML-native representation for CICS resource definitions
- The `zrb` tool can be used to translate this YAML representation into DFHCSDUP commands
- DFHCSDUP can be used to execute the resulting commands, adding the definitions to the CSD
- zconfig has integrated support for zrb, and does all this for you!

config > ! cics.resources.yaml > ...

Mortgages (cics.model.json)

```yaml
resourceDefinitions:
  - program:
      name: MTGPROG1
      group: MTGGRP1
  - transaction:
      name: M001
      group: MTGGRP1
      program: MTGPROG1
```

zrb build –m model.yaml –r defs.yaml –o csdup.txt

Prerequisites and limitations

General requirements:

- Access to z/OS UNIX System Services (USS)
- z/OS 2.4 or later
- IBM® Open Enterprise SDK for Python 3.11 or later
- Z Open Automation Utilities (ZOAU) 1.3

CICS TS requirements:

- A currently supported CICS TS release

CICS limitations:

- Only supports single CICS TS regions (also, no CPSM)
- Not all configuration is supported yet (both for discover and apply)

Accessing the closed beta

1. **Request access** for yourself and any colleagues who will need to download the tool and supporting resources, or view the documentation.

2. You will **receive an email** once access is granted.

3. The tool **resources** are:
   - ibm.biz/zconfig-beta
   - ibm.biz/zconfig-docs

4. **Try** the tool out, and be sure to give **feedback** directly or in the beta forum.

ibm.biz/zconfig-join

Experience more with IBM

Visit us at the IBM Booth #113

After a full day of technical sessions, take a break with us!

Connect with our experts, snap a photo with the z17 Plexi or the latest Telum II, and get an up-close look at our Spyre Accelerator.

Come back each day for fresh topics and demos at our expert stations.

Think 2026

Join 5000+ senior business and technology leaders who are seizing the AI revolution to unlock unprecedented growth and productivity at **Think 2026**.

Find out more information using the QR code below.

IBM Digital Asset Haven

IBM Digital Asset Haven is the operational backbone for financial institutions and regulated enterprises entering the digital asset economy.

Find out more information using the QR code below.

Your feedback is important!

SHARE
EDUCATE ▸ NETWORK ▸ INFLUENCE

Submit a session evaluation for each session you attend:

www.share.org/evaluation