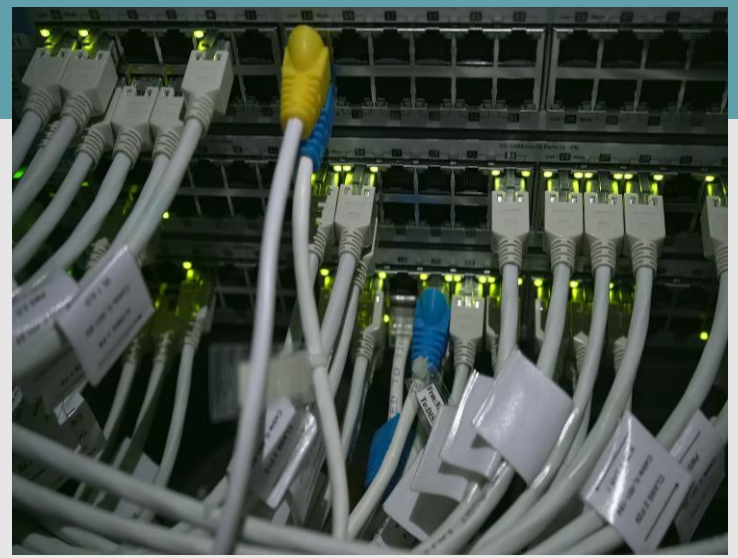
Open Source Software

An Approach to Controlling Usage and Risk in Application Ecosystems



A Brief Introduction

My name is Stan Zajdel and I am a software architect and engineer with over 30 years experience currently working in DevSecOps at a major healthcare insurer.

My focus is on the use of automation, innovation and good coding practices to help developers produce better quality, more resilient and secure software

https://www.linkedin.com/in/zajdelstan/

“Software is eating the World”

Marc Andreeson

“*Open Source* Software is eating the World”

Open Source Software is used Everywhere

Every industry, every business, everything has some type of open source software behind it

Widespread Business Adoption

93% of all companies and organizations use open source software in their applications and systems

All Companies are Becoming Software Companies

Businesses today are going through a digital transformation and open source software is driving that transformation

Predicted Growth

- Predicted growth is at a Compound Annual Growth Rate of 18.2% between 2021-26.
- Expected to reach USD $50 billion by 2026

There are Problems though…

The Problems with Open Source

There are two groups of problems:

- At the open source supply chain level
- The way that developers are using and managing that software

Open Source Supply Chain Problems

Generally an unregulated industry

- No industry-wide process that can provide assurance that the software that is being written is meeting certain quality and security standards

Open Source Supply Chain Problems

Bugs and Vulnerabilities

- Equifax
- Log4shell
- Springshell

Open Source Supply Chain Problems

Intentional Sabotage

- Color
- Faker
- Node-ipc

Open Source Supply Chain Problems

Repository Poisoning

- NPM
  - Klow
  - Klown
  - Okhsa
  - ua-parser-js

Open Source Supply Chain Problems

Inadequate Documentation

- A recent GitHub survey indicated that incomplete documentation is one of the biggest problems encountered in open source

Open Source Supply Chain Problems

Software Bloat

- For many projects there is no central authority that controls what features are added

Developer Usage Problems

Losing track of what was downloaded resulting in

- A lot of unused, exploitable software on servers
- Application bloat

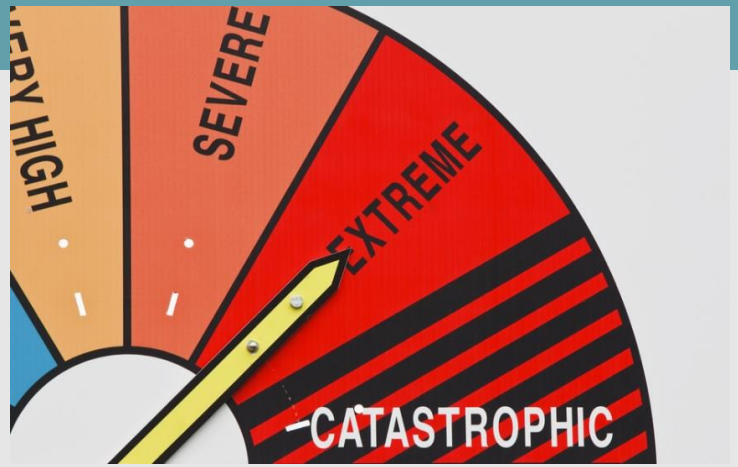
Developer Usage Problems

Developer preferences

- Leading to many different libraries that have the same capabilities
- Adds to developer cross-training time and costs

Bottom Line

- 84% of all codebases contain an open source vulnerability
- Developers spend on average, 4 hours per found vulnerability
- Average cost that a business incurs when breached is ~$4.24 million
- Open source usage is only going to increase

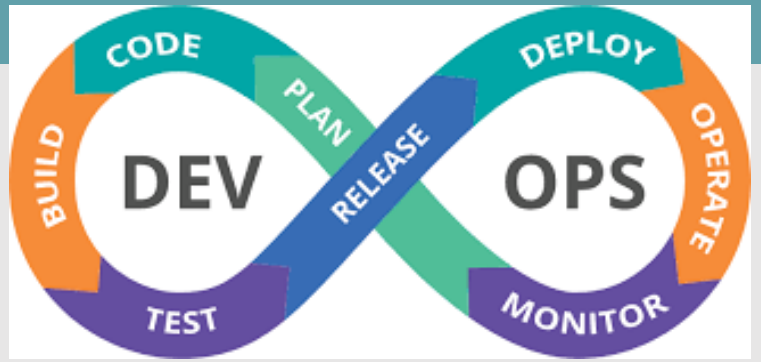


How Can We Solve this Problem?

- The solution is to reduce the risks that open source imposes on applications by stringently **regulating its use where it is being used**
- Automation is the key

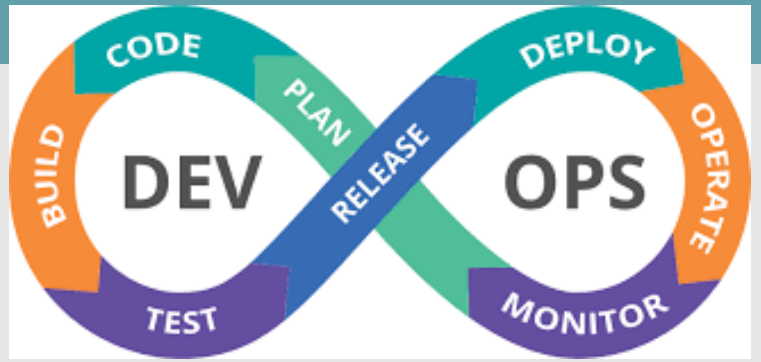
Leveraging Automation to Reduce Open Source Risk

- A DevSecOps pipeline can be leveraged as a solution

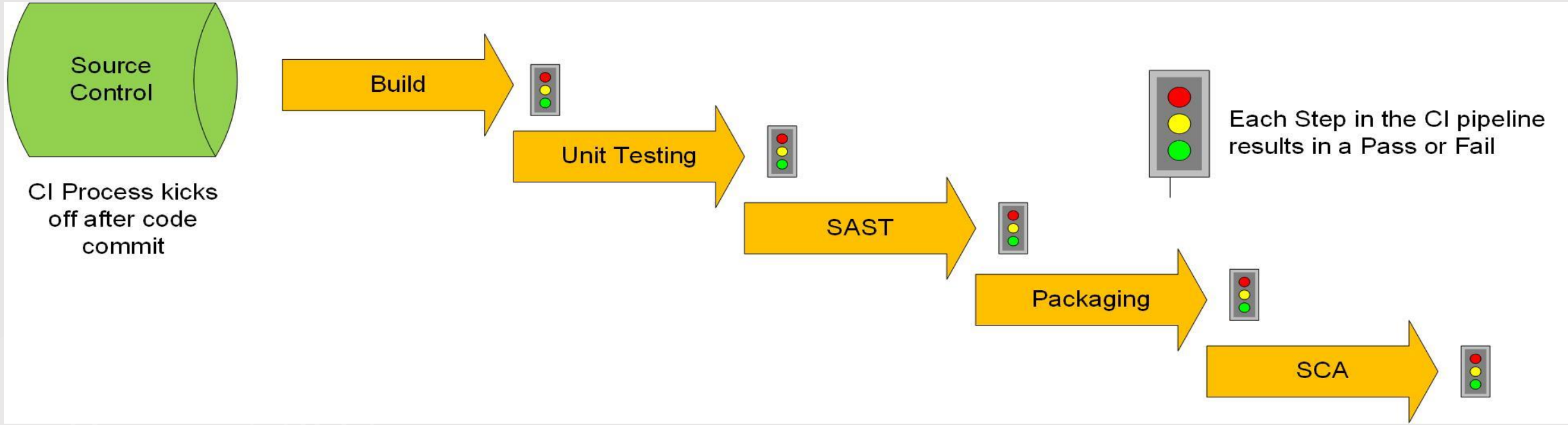


What is DevOps? DevSecOps?

- DevOps enables software development (Dev) and operations (Ops) teams to accelerate delivery through automation, collaboration, fast feedback and iterative improvement.
- DevSecOps adds security to the mix



Typical DevSecOps Continuous Integration Pipeline



How Can We Leverage a DevSecOps Pipeline to Solve the Problem?

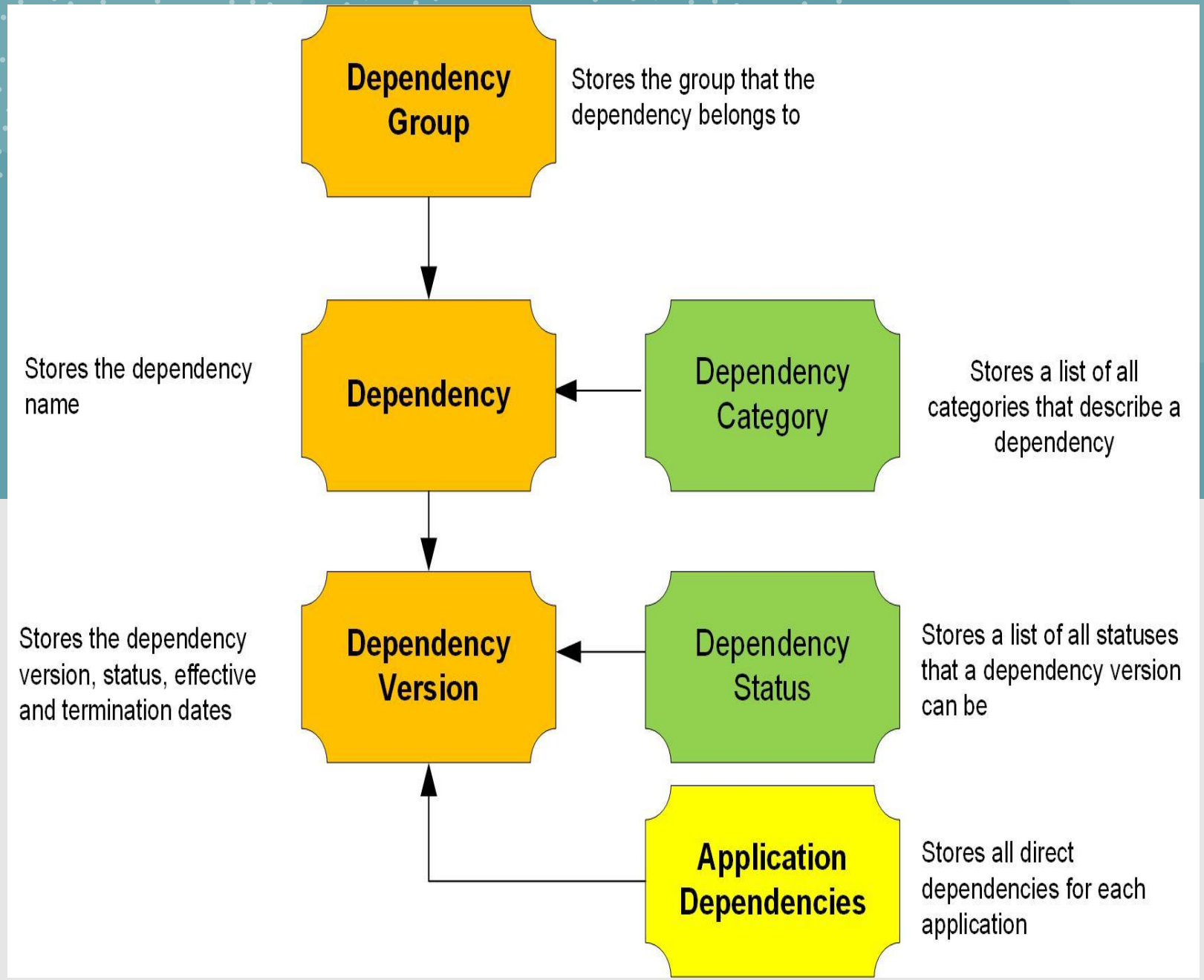
Software Bill of Materials (SBOM)

An SBOM answers the following questions:

- What components are we using?
- How many different versions of this software are being used?

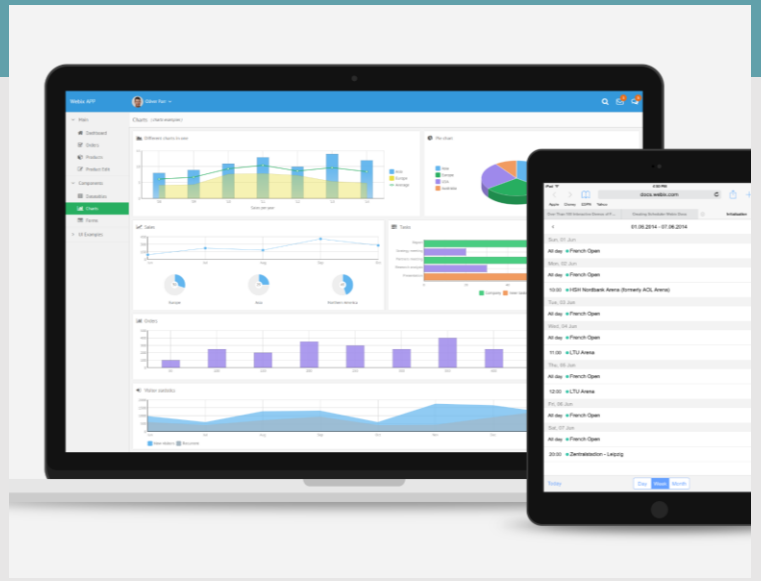
Dependency Reference Database

- Keeps a reference of all libraries that are in the application ecosystem
- Maintains an SBOM for the entire ecosystem as well as for individual applications

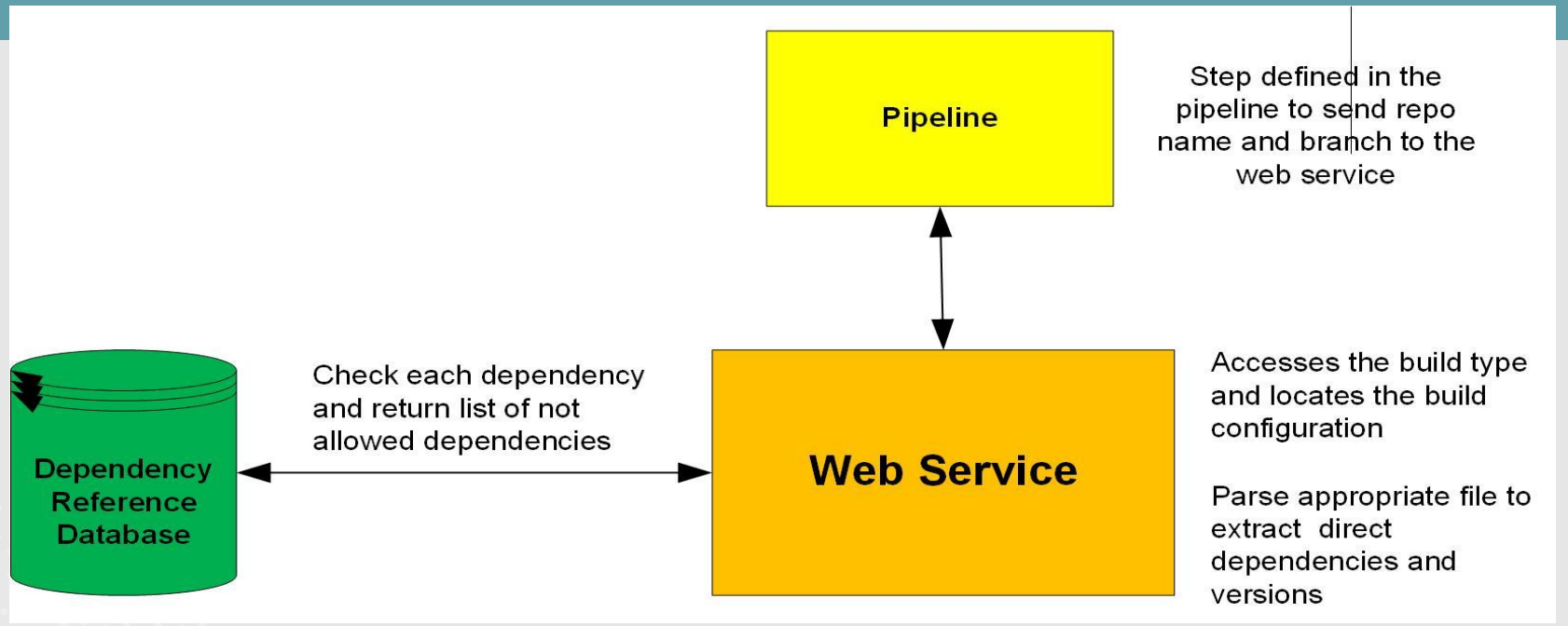


Web Application into the SBOM

- A web app for developers and decision-makers to view, update and categorize the SBOM information
- Developers are required to consult the web app for approved software to use



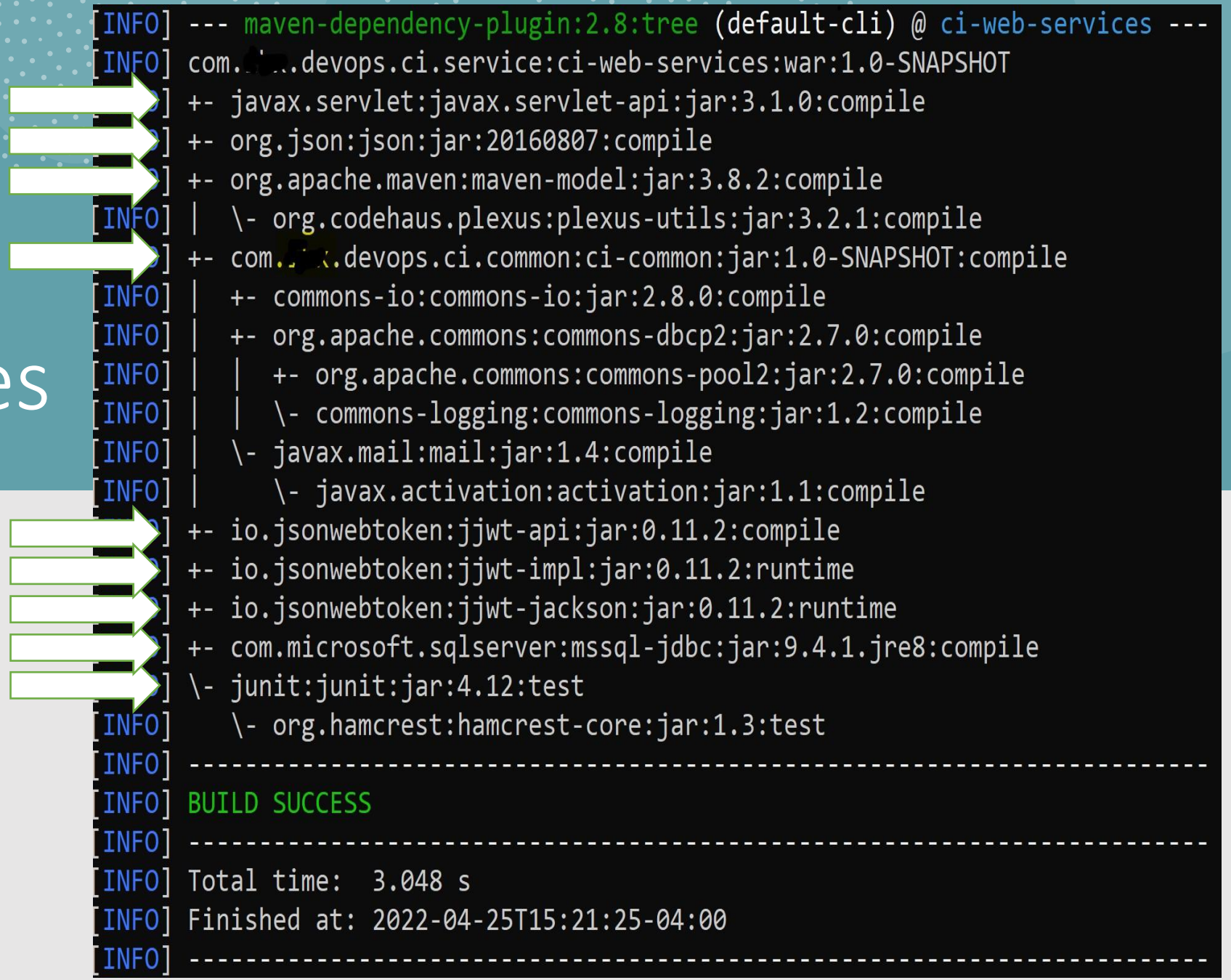
Dependency Reference Service



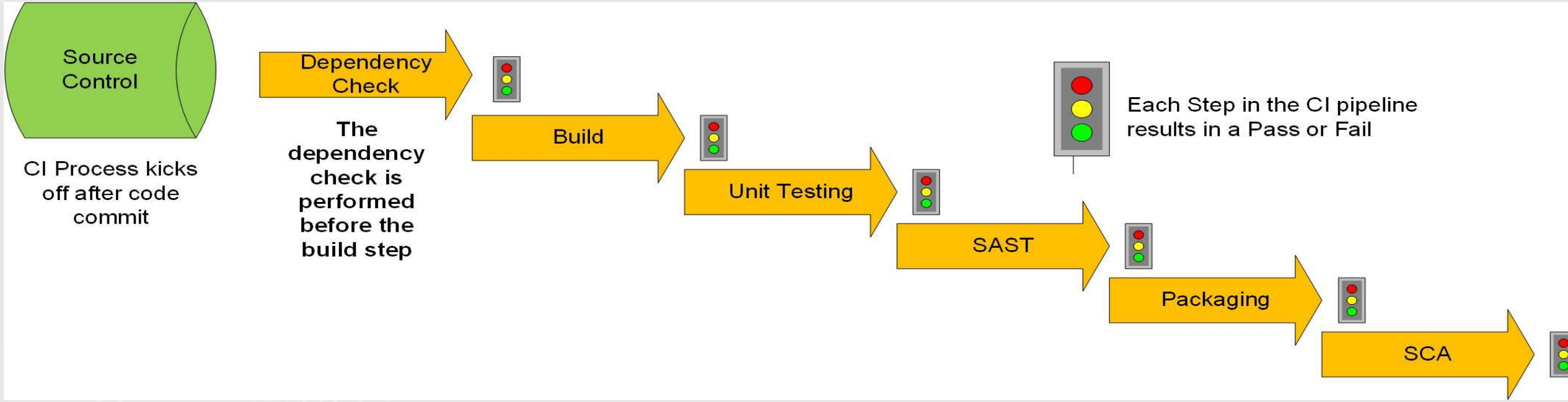
Maven Dependency Tree

```
[INFO] --- maven-dependency-plugin:2.8:tree (default-cli) @ ci-web-services ---
[INFO] com.[redacted].devops.ci.service:ci-web-services:war:1.0-SNAPSHOT
[INFO] +- javax.servlet:javax.servlet-api:jar:3.1.0:compile
[INFO] +- org.json:json:jar:20160807:compile
[INFO] +- org.apache.maven:maven-model:jar:3.8.2:compile
[INFO] |  \- org.codehaus.plexus:plexus-utils:jar:3.2.1:compile
[INFO] +- com.[redacted].devops.ci.common:ci-common:jar:1.0-SNAPSHOT:compile
[INFO] |  +- commons-io:commons-io:jar:2.8.0:compile
[INFO] |  +- org.apache.commons:commons-dbcp2:jar:2.7.0:compile
[INFO] |  |  +- org.apache.commons:commons-pool2:jar:2.7.0:compile
[INFO] |  |  \- commons-logging:commons-logging:jar:1.2:compile
[INFO] |  \- javax.mail:mail:jar:1.4:compile
[INFO] |     \- javax.activation:activation:jar:1.1:compile
[INFO] +- io.jsonwebtoken:jjwt-api:jar:0.11.2:compile
[INFO] +- io.jsonwebtoken:jjwt-impl:jar:0.11.2:runtime
[INFO] +- io.jsonwebtoken:jjwt-jackson:jar:0.11.2:runtime
[INFO] +- com.microsoft.sqlserver:mssql-jdbc:jar:9.4.1.jre8:compile
[INFO] \- junit:junit:jar:4.12:test
[INFO]    \- org.hamcrest:hamcrest-core:jar:1.3:test
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  3.048 s
[INFO] Finished at: 2022-04-25T15:21:25-04:00
[INFO] ------------------------------------------------------------------------
```

Identified Direct Dependencies



Dependency Reference Service on the Pipeline



Preliminary Results

Preliminary Results

Examples of what we found:

- Over 2000 Maven libraries and versions in the ecosystem
- ~96 Java JSON versions of parsers
- ~16 versions of Spring-web and Struts
- Multiple versions of internal libraries

Preliminary Results

- ~1000 reported vulnerabilities on 398 components used in ~750 repos
- On average ~2-4 new vulnerabilities reported per day

Governance

Governance
Creation of Guidelines

- Create guidelines on how to introduce and select open source software.
- The guidelines should be followed to ensure that software is introduced in a responsible, systematic and governed way.

Governance Committee

- The committee should consist of SMEs, Security, and Management
- Consult security tooling, outside code repositories and internal sources

Governance
Research, Reduce and Categorize

- Identify and eliminate duplicate functionality
- Ensure minimal vulnerabilities
- Standardize on best fit

Governance
Migrating to the Approved Components

- After the SBOM has been vetted and categorized all existing applications must be migrated to use the approved components
- A phased approach works best

Governance
New Component Vetting Process

- New components and versions are automatically added for a trial period
- An email is sent to the governance committee notifying that a new component or version has been detected

Governance
New Component Vetting Process

- Is there an approved component that provides the same functionality?
- How many vulnerabilities and what is the severity?
- Is it actively maintained?
- Who is the maintainer?
- How do they view security?
- What is their issue history?

Governance Grievance Process

- A process to allow developers the opportunity to “plead” their case and provide valid reasons for why they should be allowed to use a new component that was rejected

Developer Education and Awareness

Developer Education and Awareness

- Studies have shown that developers make suboptimal choices 69% of the time when selecting and updating open source components
- Do societal forces influence these decisions?

Developer Education and Awareness

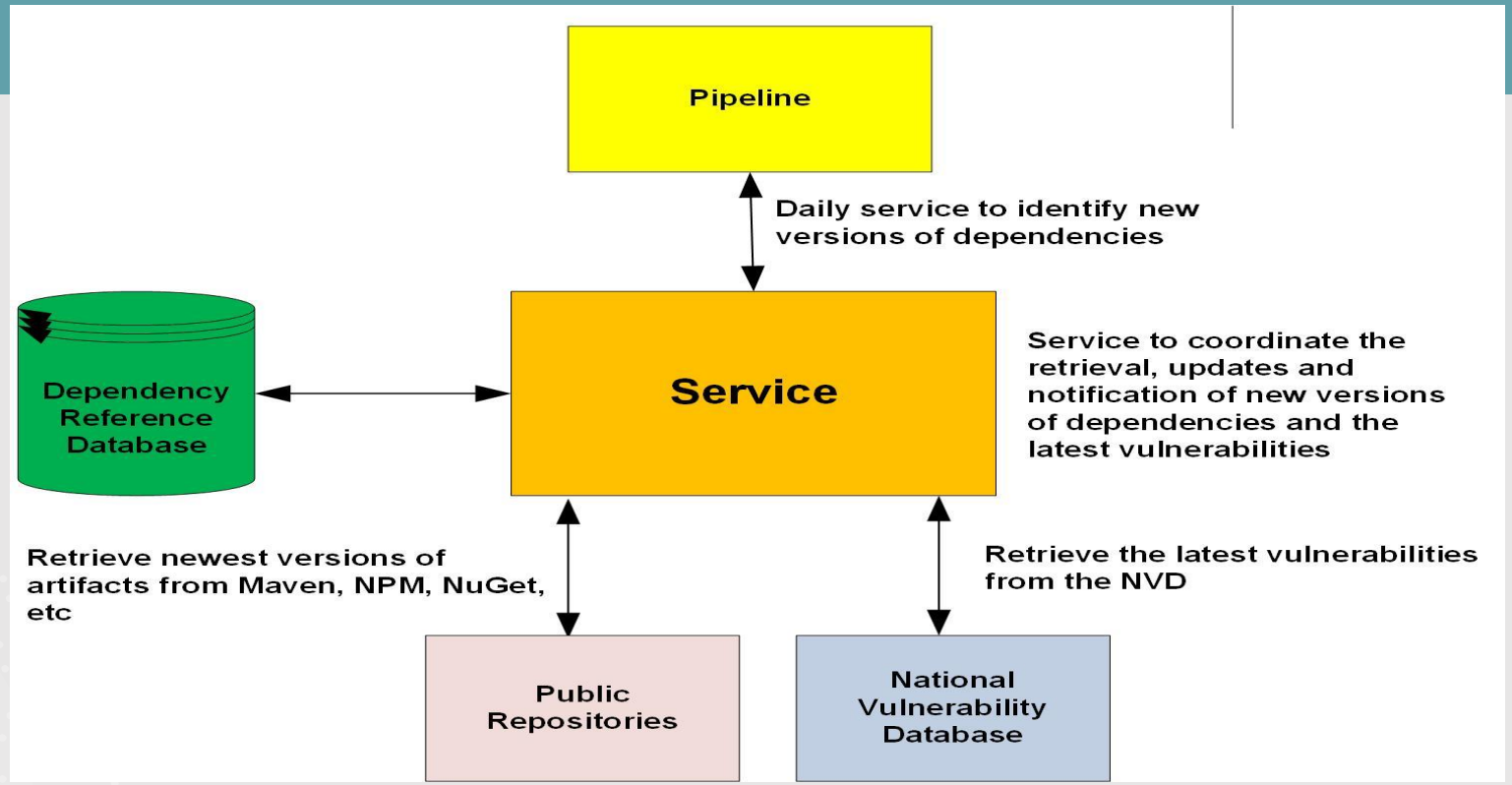
- Developers need to be taught to focus more attention to what software they are using and to ensure that their build configurations are always kept up to date
- Can this be automated?

Developer Education and Awareness

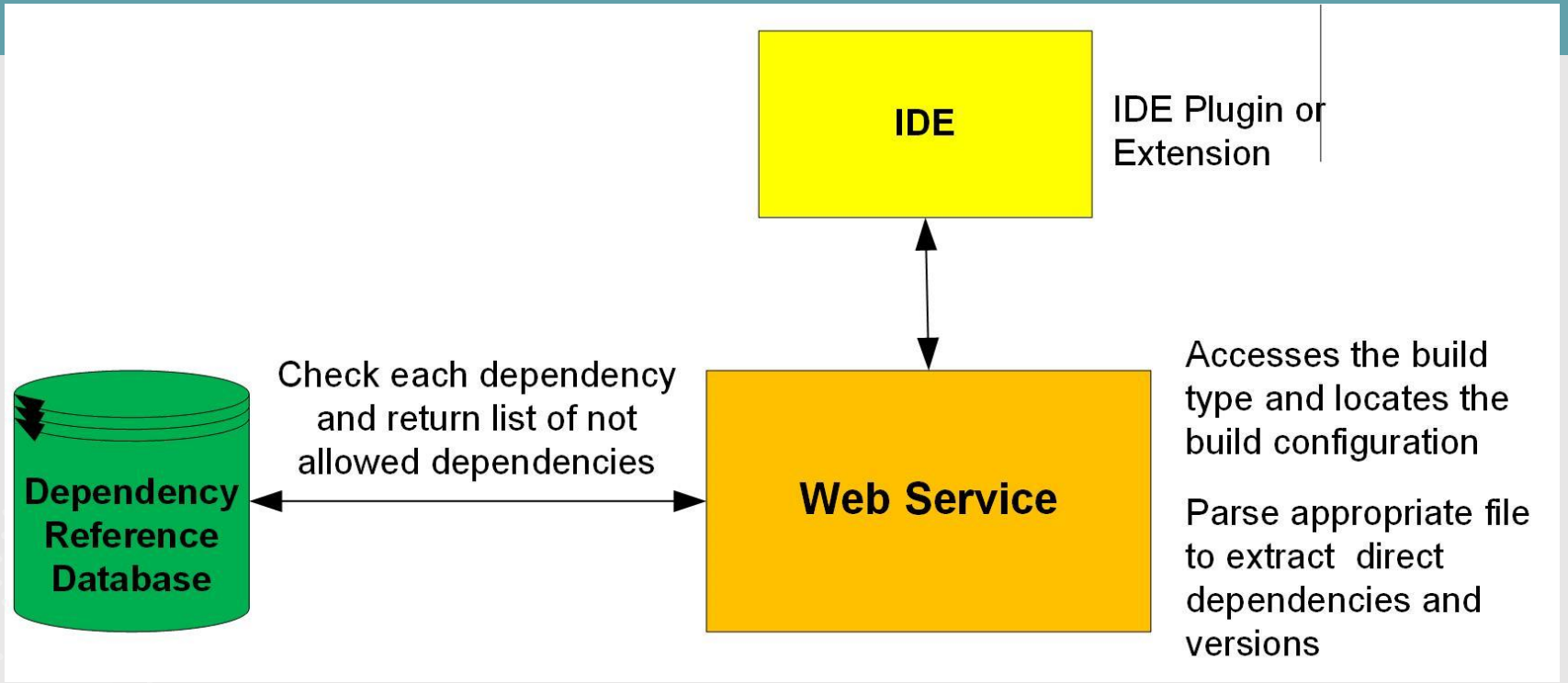
- Development staff should be trained to always consult the reference library to find software that is sanctioned for use as well as latest versions available before looking outside

Future Capabilities

Future Capabilities
Automatically Track New Versions and Vulnerabilities



Future Capabilities
Dependency Reference Service in the IDE



Potential Research Focus Areas

Open Source Supply Chain Certification Process

- Software has been well-tested
- Vulnerabilities are kept to a minimum
- Fully documented
- Assurance that the software performs as intended.

Societal Forces thatDrive Software Adoption

- The affects that social coding environments where popularity, number of users or followers, links and a host of other types of social information are used in the technical decision-making process such as the adoption of libraries and frameworks

Better Library and Framework Development

- Frameworks are often bloated.
- No process to control the type or number of features being added.
- Banana, gorilla, jungle syndrome
- More bloat = bigger attack surface

Thank you!