

# SURVIVAL GUIDE TO CONTINUOUSLY ALIGN AND UPGRADE YOUR SPRING DEPENDENCIES



**Raquel Pau**

[raquel.pau@broadcom.com](mailto:raquel.pau@broadcom.com)

**Vicente Soriano**

[vicente.soriano@broadcom.com](mailto:vicente.soriano@broadcom.com)



WHO ARE WE?

# Vicente Soriano

Software Engineer @ VMware Broadcom

 [in/visomar](https://www.linkedin.com/in/visomar)

 [vicente.soriano@broadcom.com](mailto:vicente.soriano@broadcom.com)







WHO ARE WE?

# Raquel Pau

Product Manager @ VMware Broadcom

 [in/raquel-pau](https://www.linkedin.com/in/raquel-pau)

 raquel.pau@broadcom.com



# AGENDA

- Basics of dependency management
- Minor and major upgrades in action
- Upgrade orchestration

# WHAT IS A DEPENDENCY?

- “[...] A relationship between software components where one component relies on the other to function correctly [...]”
- An imported library that can be leveraged to not reimplement something (or to not reinvent the wheel)

# WHAT IS A DEPENDENCY?

e.g.

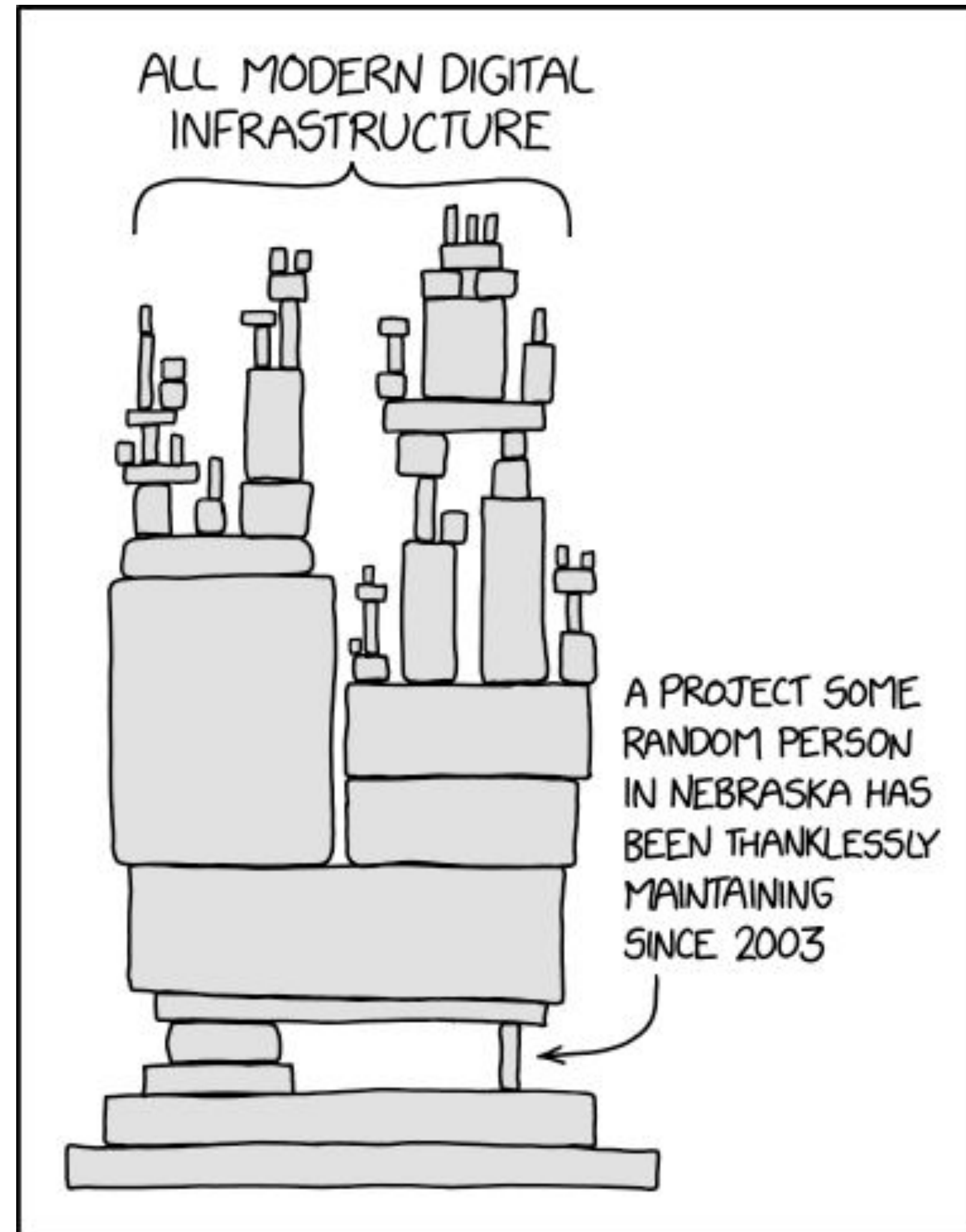
```
<dependency>  
  <groupId>org.springframework.boot</groupId>  
  <artifactId>spring-boot-starter-test</artifactId>  
  <scope>test</scope>  
</dependency>
```

# WHAT IS A DEPENDENCY?

We benefit **HUGELY** from the projects and collaborations that happen in the OSS world

# WHAT IS A DEPENDENCY?

## Drawbacks



Randall Munroe  
<https://xkcd.com/2347/>



# WHAT IS A VULNERABILITY?

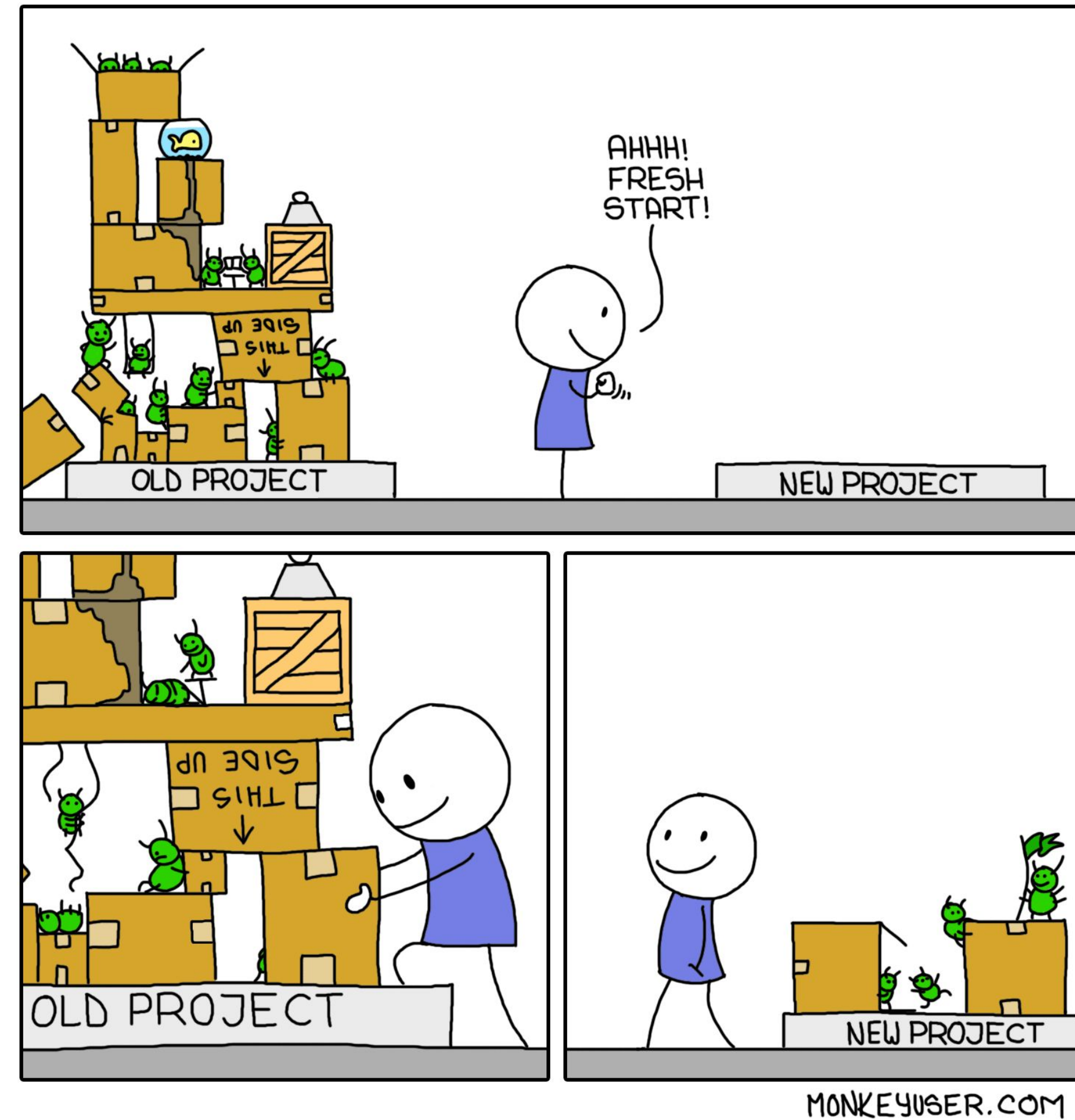
By [www.cve.org](https://www.cve.org):

- "An instance of one or more weaknesses in a Product that can be exploited, causing a negative impact to confidentiality, integrity, or availability; a set of conditions or behaviors that allows the violation of an explicit or implicit security policy."

# WHAT IS A VULNERABILITY?

"If it works, don't touch it"

## CODE REUSE




<https://www.monkeyuser.com/2018/code-reuse/>

# WHAT IS A VULNERABILITY?

Log4Shell

# WHAT IS A SPRING PROJECT?




[Why Spring](#) [Learn](#) [Projects](#) [Academy](#) [Community](#) [Tanzu Spring](#)

## Projects

From configuration to security, web apps to big data—whatever the infrastructure needs of your application may be, there is a Spring Project to help you build small and use just what you need—Spring is modular by design.


RELEASE CALENDAR



### Spring Boot

Takes an opinionated view of building Spring applications and gets you up and running as quickly as possible.


3.5.0 + 10 versions




### Spring Framework

Provides core support for dependency injection, transaction management, web apps, data access, messaging, and more.


6.2.7 + 8 versions



### Spring Cloud



### Spring Cloud Data Flow



### Spring Security

Overview

Spring Boot

Spring Framework

Spring Cloud

Spring Cloud Data Flow

Spring Data

Spring Integration

Spring Batch

Spring Security

Spring AI

Release Calendar

Security advisories

DEVELOPMENT TOOLS

Spring Tools

Spring Initializr

SPRING

2025



# WHAT IS A SPRING PROJECT?

A collection of Maven artifacts, with a common:

- Purpose or mission
- Repository
- Functionality
- Release cadence
- etc.

# WHAT IS A SPRING PROJECT?

spring initializr

Web, Security, JPA, Actuator, Devtools...

Press Ctrl for multiple adds

TEMPLATE ENGINES

JTE  
Secure and lightweight template engine for Java and Kotlin.

SECURITY

Spring Security  
Highly customizable authentication and access-control framework for Spring applications.

OAuth2 Client  
Spring Boot integration for Spring Security's OAuth2/OpenID Connect client features.

OAuth2 Authorization Server  
Spring Boot integration for Spring Authorization Server.

OAuth2 Resource Server  
Spring Boot integration for Spring Security's OAuth2 resource server features.

Project

Gradle - Groovy

Gradle - Kotlin

Maven

Spring Boot

4.0.0 (SNAPSHOT)

3.5.0 (SNAPSHOT)

3.4.5

3.3.12 (SNAPSHOT)

3.3.0

Project Metadata

Group

com.example

Artifact

demo

Name

demo

ADD DEPENDENCIES... CTRL + B

32

<dependencies>

33

<dependency>

34

<groupId>org.springframework.boot</groupId>

35

<artifactId>spring-boot-starter-security</artifactId>

36

</dependency>

37

<dependency>

38

<groupId>org.springframework.boot</groupId>

39

<artifactId>spring-boot-starter-test</artifactId>

40

<scope>test</scope>

41

</dependency>

42

<dependency>

43

<groupId>org.springframework.security</groupId>

44

<artifactId>spring-security-test</artifactId>

45

<scope>test</scope>

46

</dependency>

47

</dependencies>

48







</dependencies>

# WHAT IS A SPRING PROJECT?

spring-cloud-services-starters		3.5.x	4.0.x			4.1.x		4.2.x
java-cfenv		2.5.x		3.1.x		3.2.x	3.3.x	3.4.x
spring-boot		2.7.x	3.0.x	3.1.x	3.2.x	3.3.x		3.4.x
spring-security		5.8.x	6.0.x	6.1.x	6.2.x	6.3.x		6.4.x
spring-data-jpa		2.7.x	3.0.x	3.1.x	3.2.x	3.3.x		3.4.x
spring-framework		5.3.x	6.0.x		6.1.x			6.2.x
reactor		3.4.x	3.5.x		3.6.x			3.7.x
hibernate		5.6.x	6.1.x	6.2.x	6.4.x	6.5.x		6.6.x

# UPGRADE YOUR APPLICATION

Normally it involves bumping the version numbers of your dependencies

- To a newer patch version, e.g. 3.4.0 → 3.4.5 
- To a newer minor version, e.g. 3.0.x → 3.4.x  
- To a newer major version, e.g. 2.7.x → 3.0.x   



# UPGRADE YOUR APPLICATION

Although in most cases, it will require code changes due to

- Breaking changes between versions
- Deprecations
- Changes in best practices/defaults

# DEMO 1

## SIMPLE UPGRADE TO THE NEXT MINOR



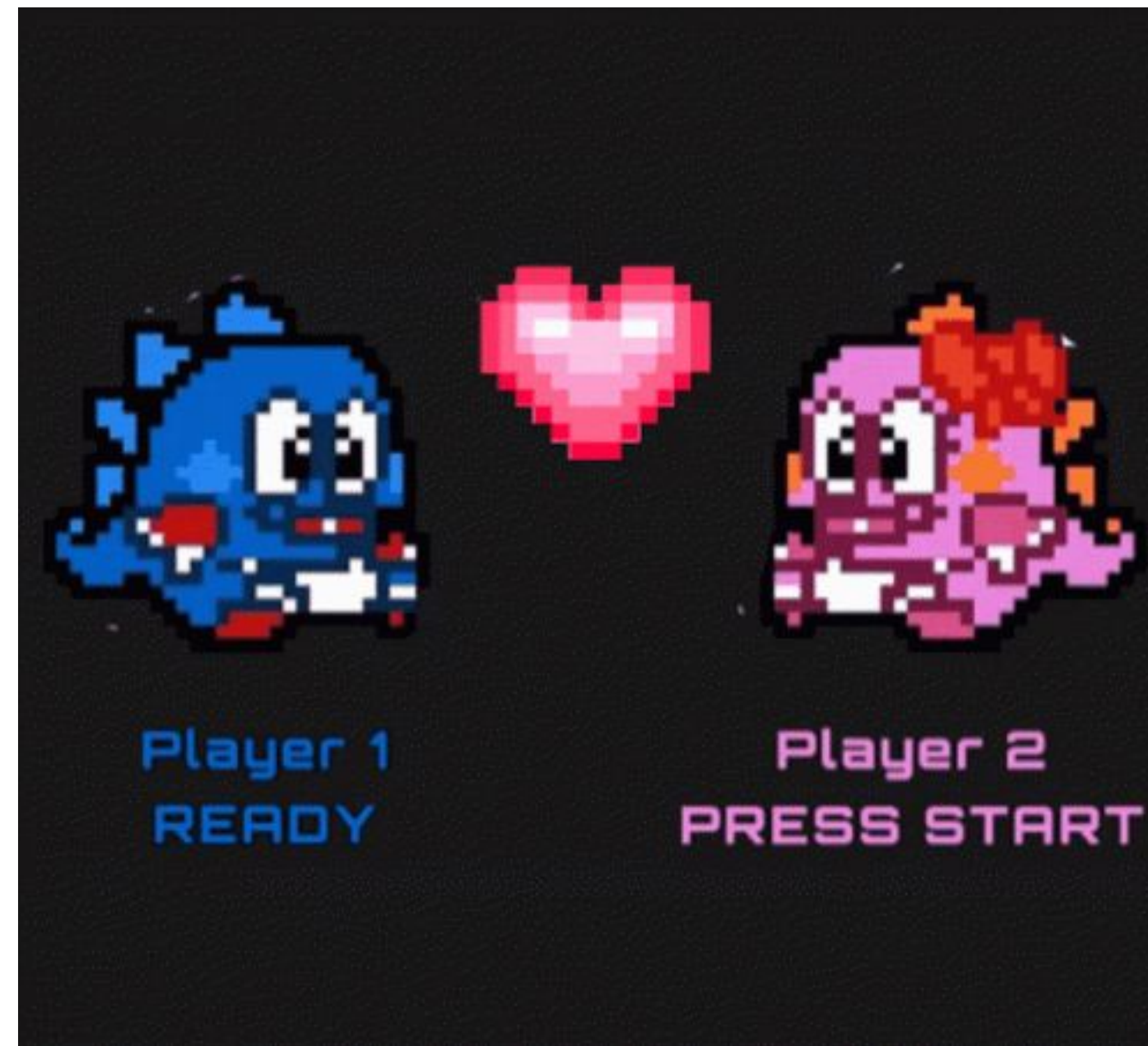
# UPGRADE YOUR APPLICATION - BREAKING CHANGES

What if there are APIs that have changed or are no longer available?




# UPGRADE YOUR APPLICATION - BREAKING CHANGES

We better start using a tool that helps us with the upgrade












# UPGRADE YOUR APPLICATION - BREAKING CHANGES








## OpenRewrite

Semantic code search and transformation

 738 followers    San Francisco, CA    <https://docs.openrewrite.org>    @openrewrite    @moderne-and-openrewrite

 <https://www.moderne.ai/community>    [team@moderne.io](mailto:team@moderne.io)

 Overview    Repositories 68    Projects 1    Packages    People 5

README.md

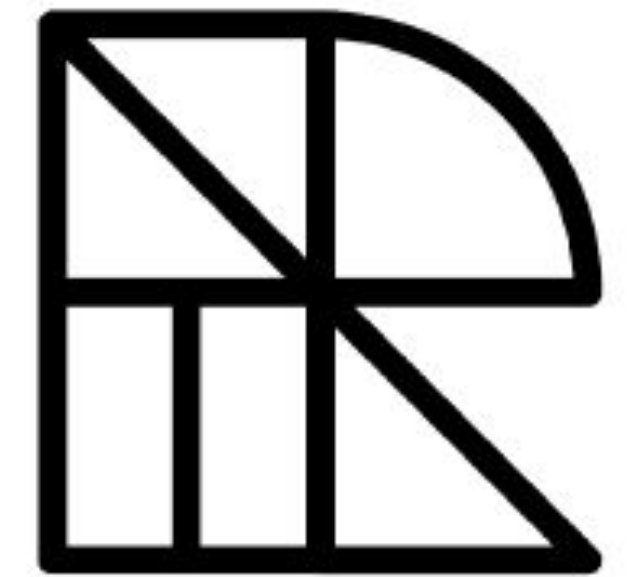
### Get fast, repeatable source code refactoring

[OpenRewrite](#) (founded and managed by [Moderne](#)) is a community-driven open source project that enables developers to effectively eliminate technical debt within their repositories. It consists of an auto-refactoring engine that runs prepackaged, open source refactoring recipes for common framework migrations, security fixes, and stylistic consistency tasks.

OpenRewrite is Apache2 licensed and maintained by Moderne, the company that scales the use of OpenRewrite recipes across large codebases and multiple repositories.

Start with the [quickstart guide](#) and let OpenRewrite handle the boring parts of software development for you. We also welcome you to join our [community Slack](#).

# ADVANTAGES OF OPENREWRITE



- Applies refactors with accuracy. No false positives.
- Respects code formatting after an upgrade.
- You can write your own code refactor/recipe or use any of the +3000 recipes.
- It runs as a Maven or Gradle plugin.



# DEMO 2

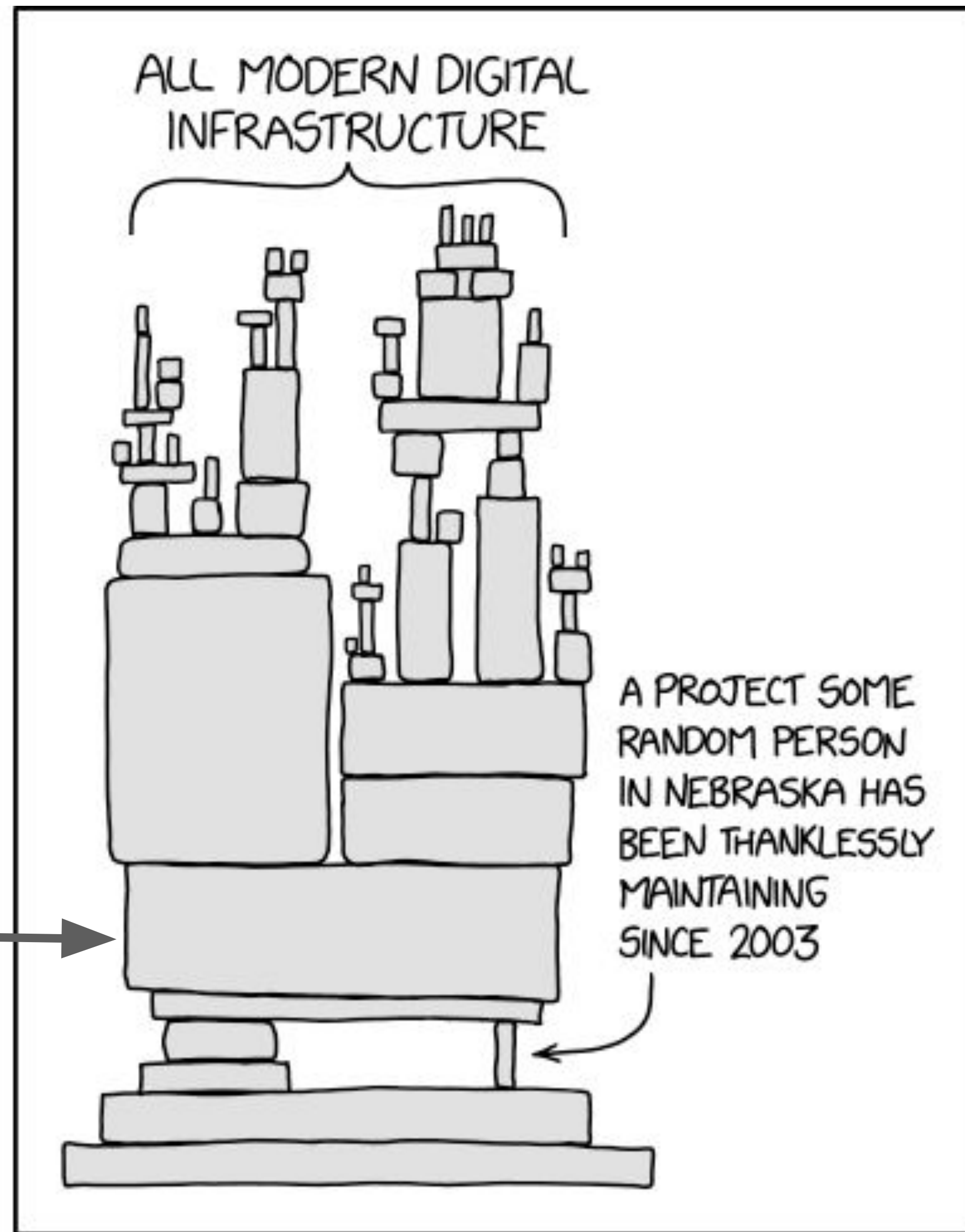
## UPGRADE WITH BREAKING CHANGES



# UPGRADE YOUR APPLICATION - CHALLENGES

Let's see the most common scenario for major upgrades



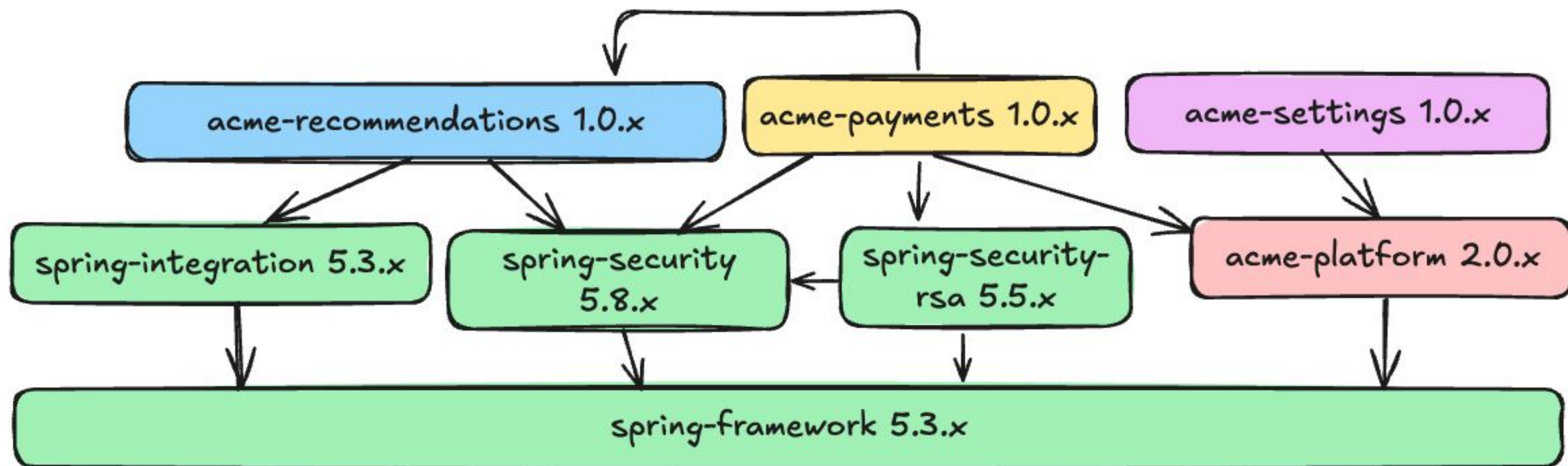


# UPGRADE ORCHESTRATION - CHALLENGES

1. Projects need to be upgraded and released in a specific order.
2. All projects need to build recipes? What if there are +100 ?
3. Some (OSS) projects might need to be replaced.

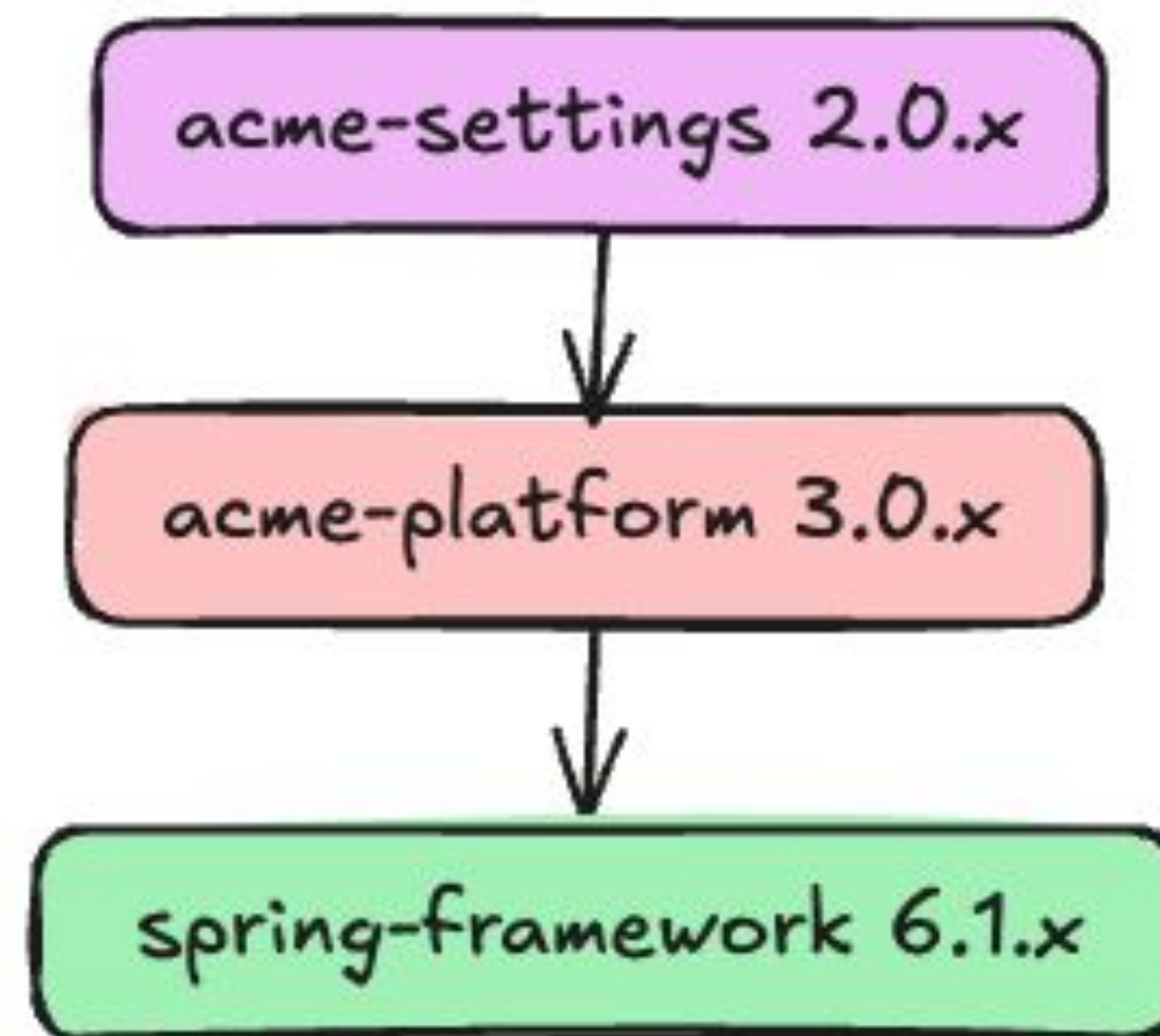
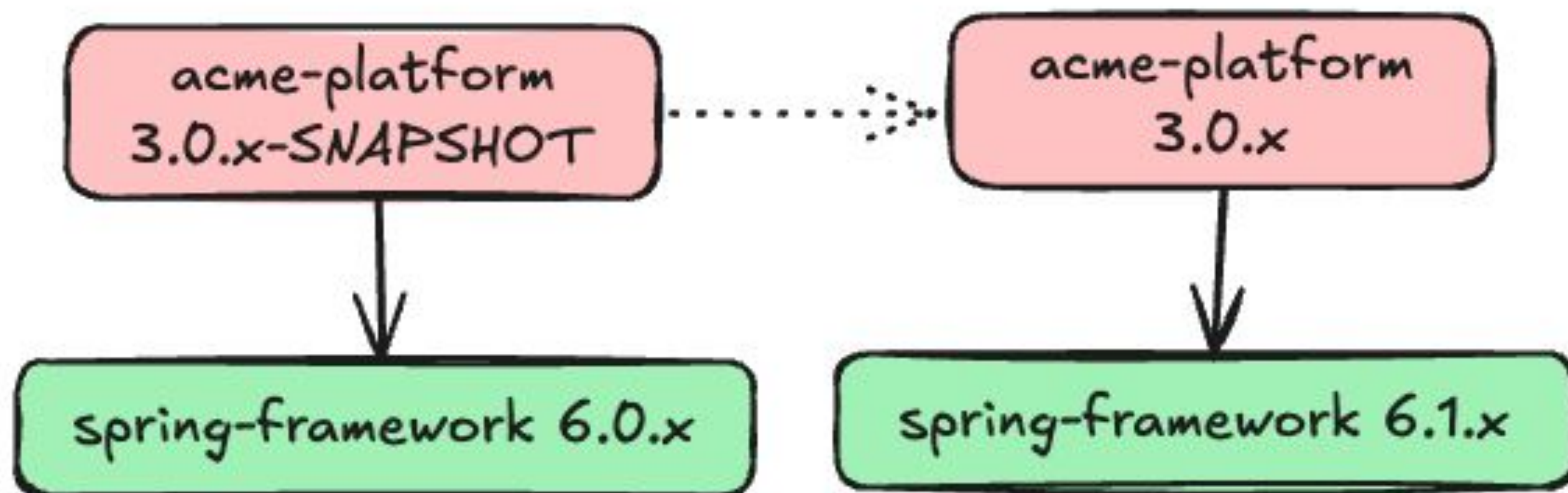
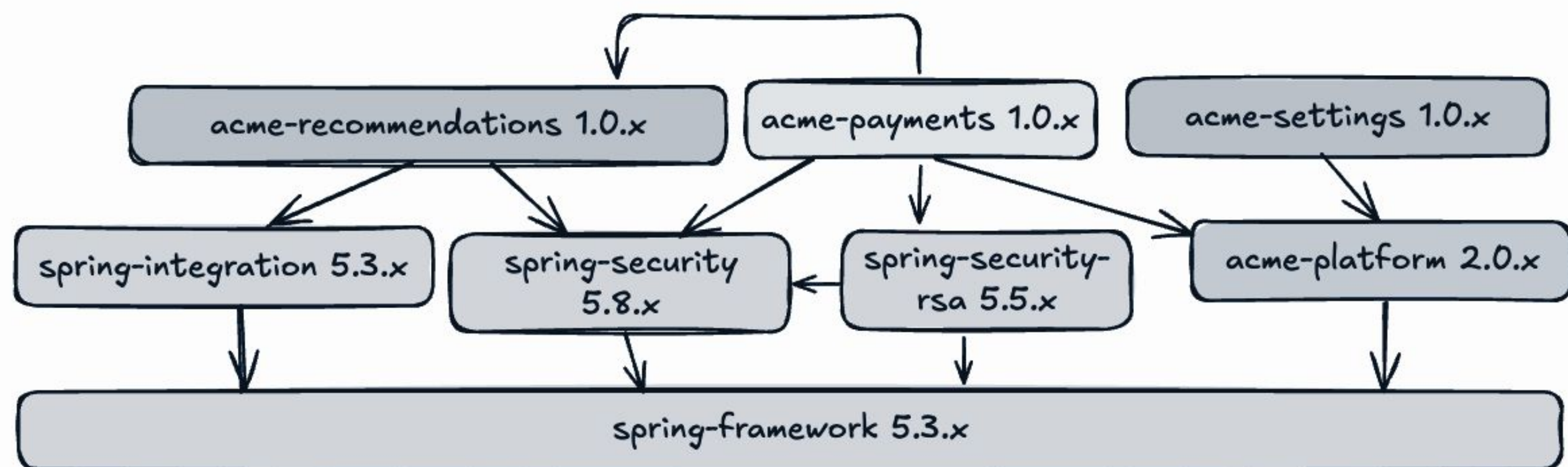


Upgrade to spring-framework 6.1.x to stay in OSS support

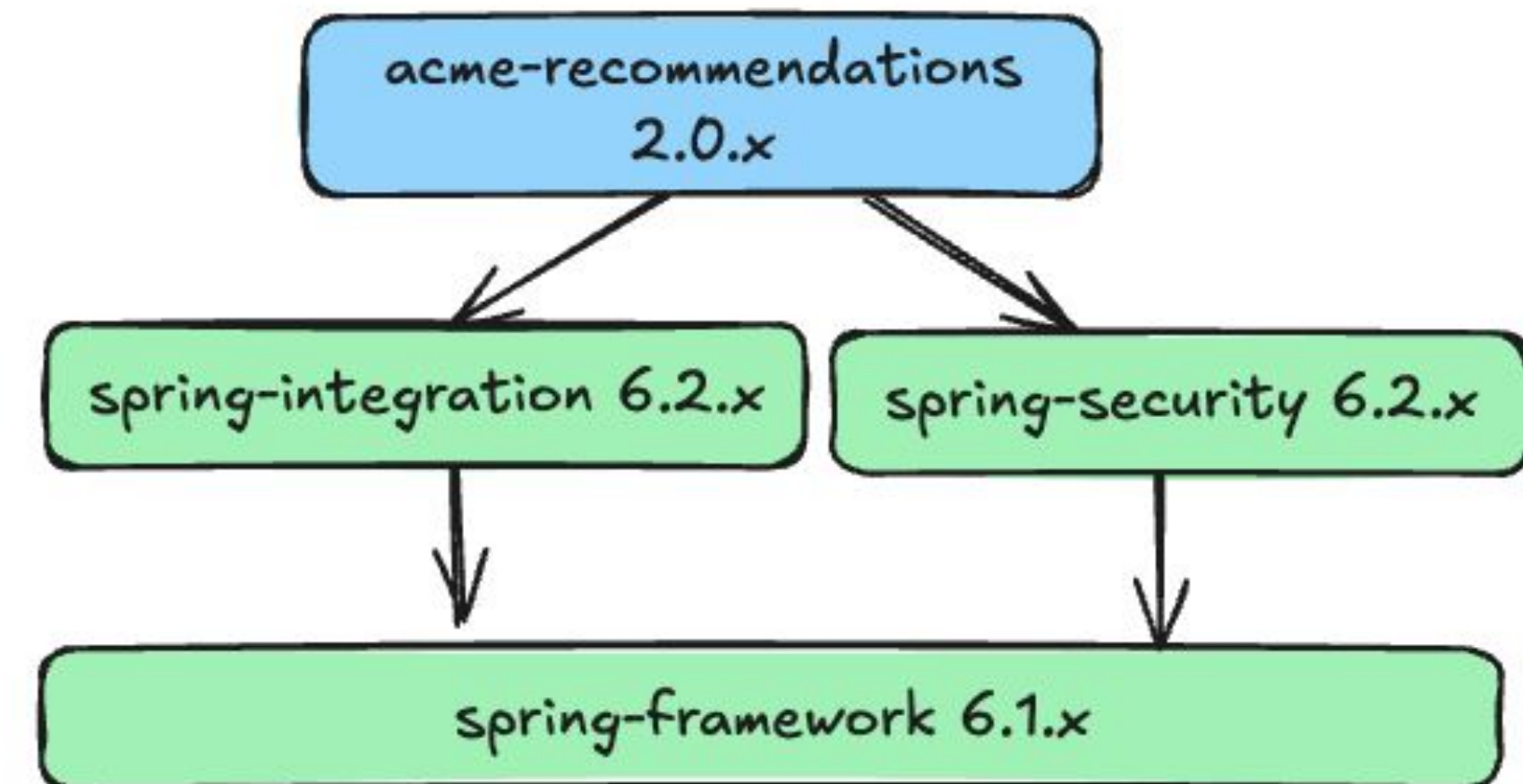
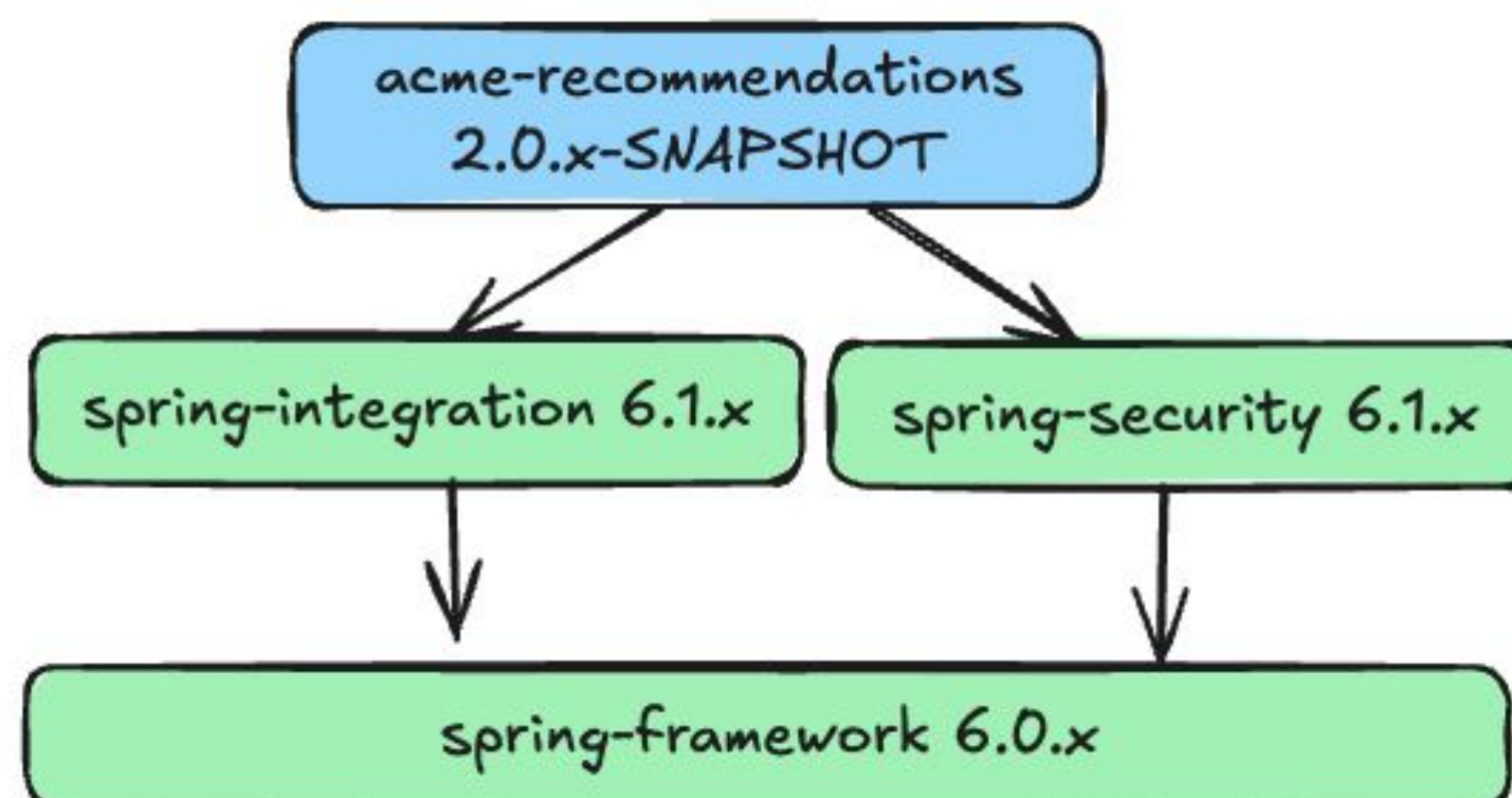
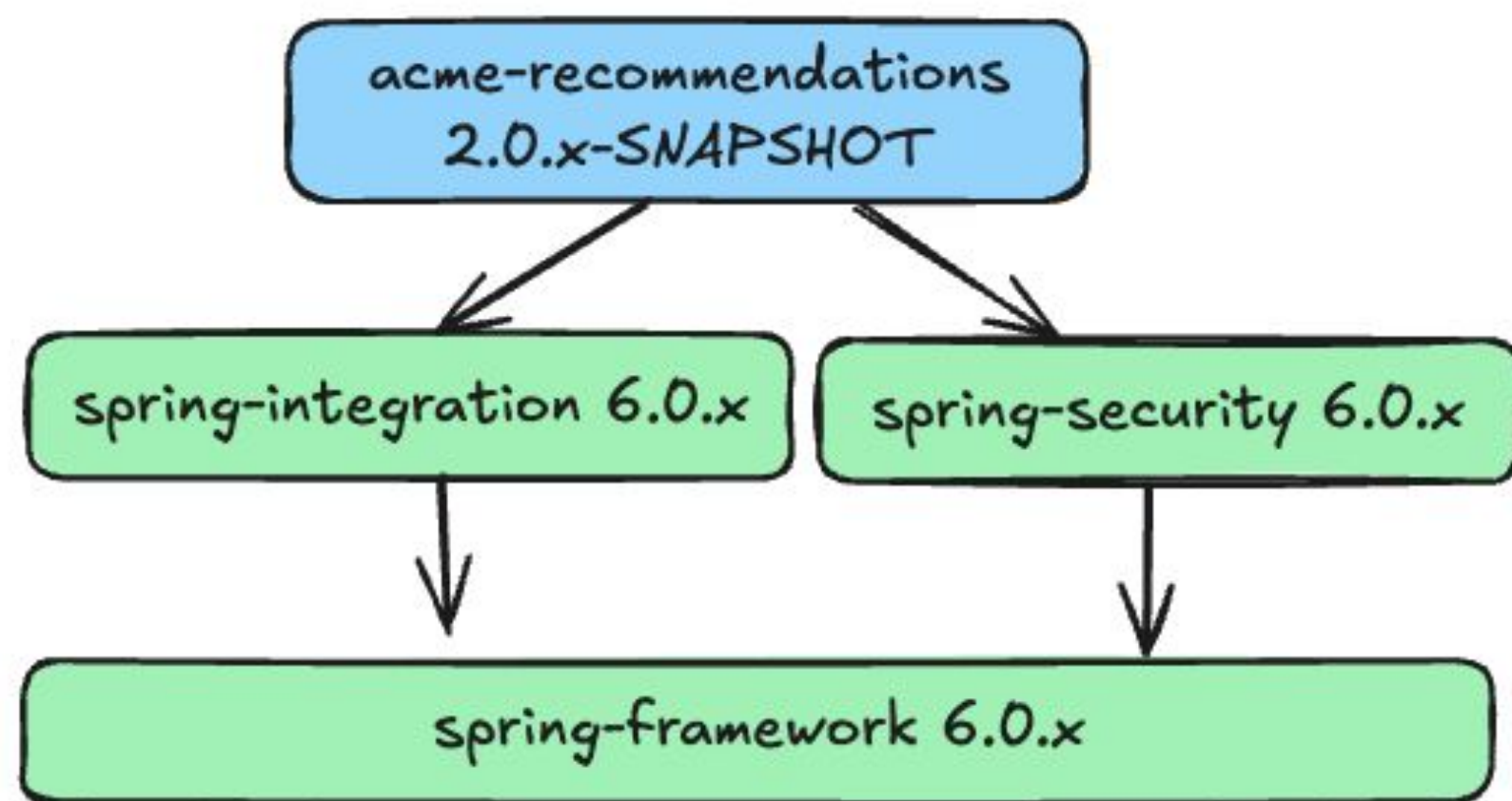
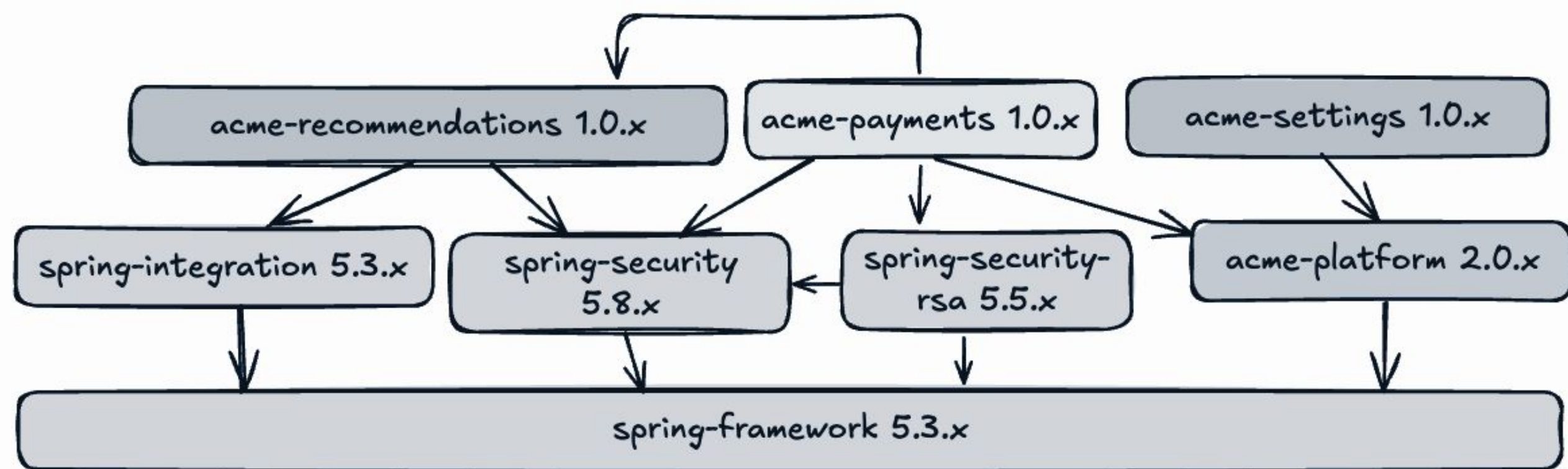


SO,  
WHAT'S  
THE  
UPGRADE PLAN?

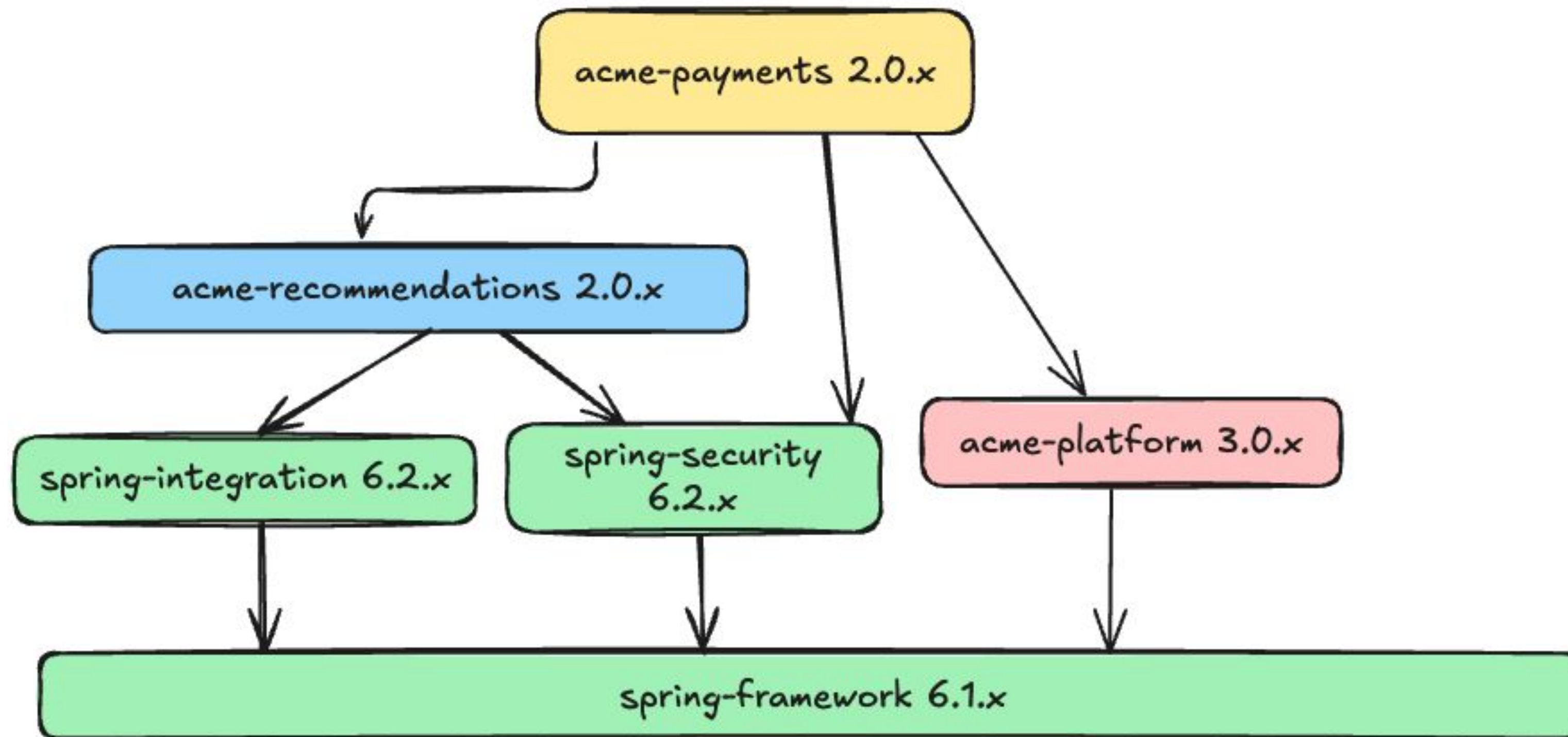
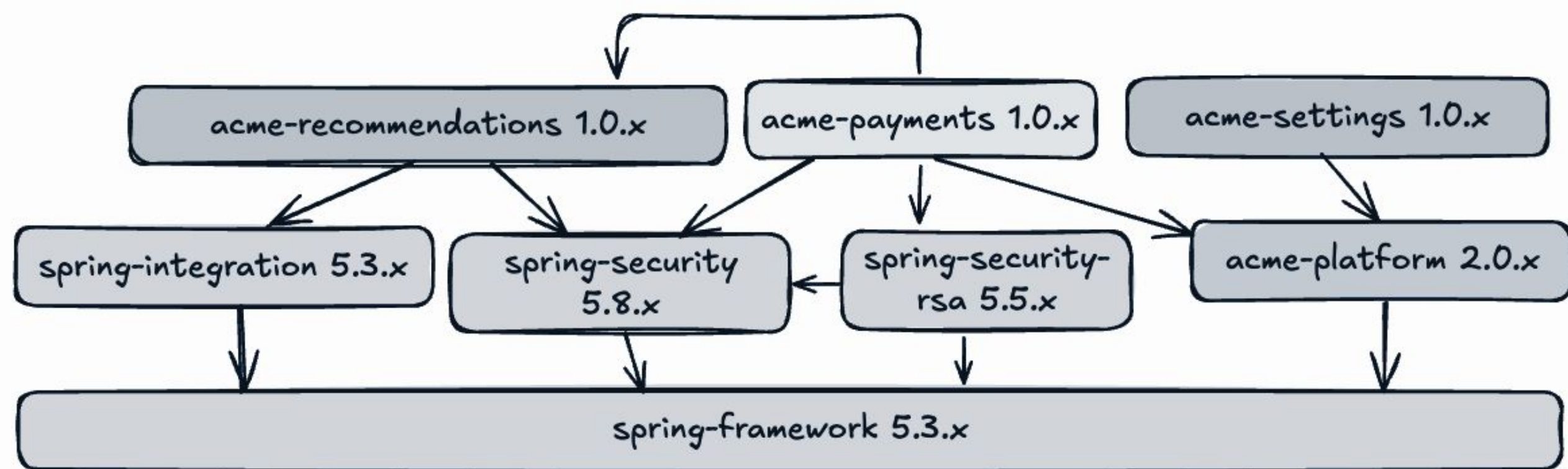


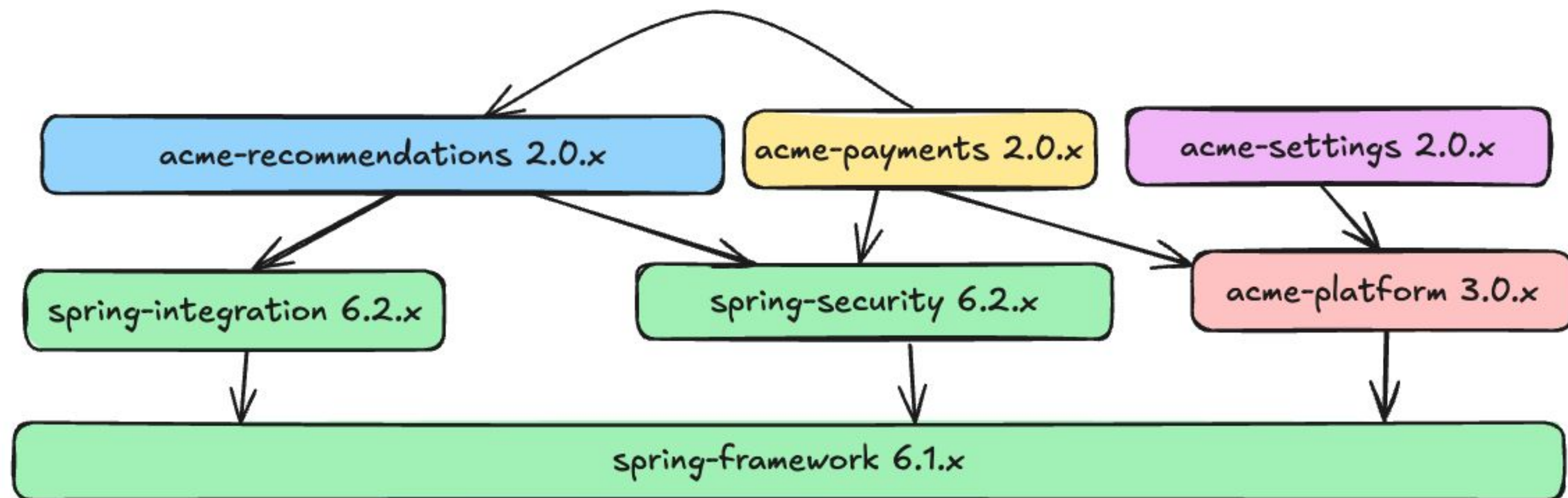














# LESSONS USING OPEN REWRITE FOR UPGRADE ORCHESTRATION

Incremental upgrades are critical to understand the changes.

Upgrade my dependencies to latest or whatever, I don't know #982

<> Code ▾

 Open visomar wants to merge 1 commits into `main` from `just-upgrade-my-dependencies` 

 Conversation 1

 Commits 1

 Checks 6

 Files changed 420

+31,052 -11,165 

# LESSONS USING OPEN REWRITE FOR UPGRADE ORCHESTRATION

Incremental upgrades are critical to understand the changes.

```
---
type: specs.openrewrite.org/v1beta/recipe
name: org.openrewrite.java.spring.boot3.UpgradeSpringBoot_3_4
displayName: Migrate to Spring Boot 3.4
description: |
  Migrate applications to the latest Spring Boot 3.4 release. This recipe
tags:
- spring
- boot
recipeList:
- org.openrewrite.java.spring.boot3.UpgradeSpringBoot_3_3
- org.openrewrite.java.spring.framework.UpgradeSpringFramework_6_2
```

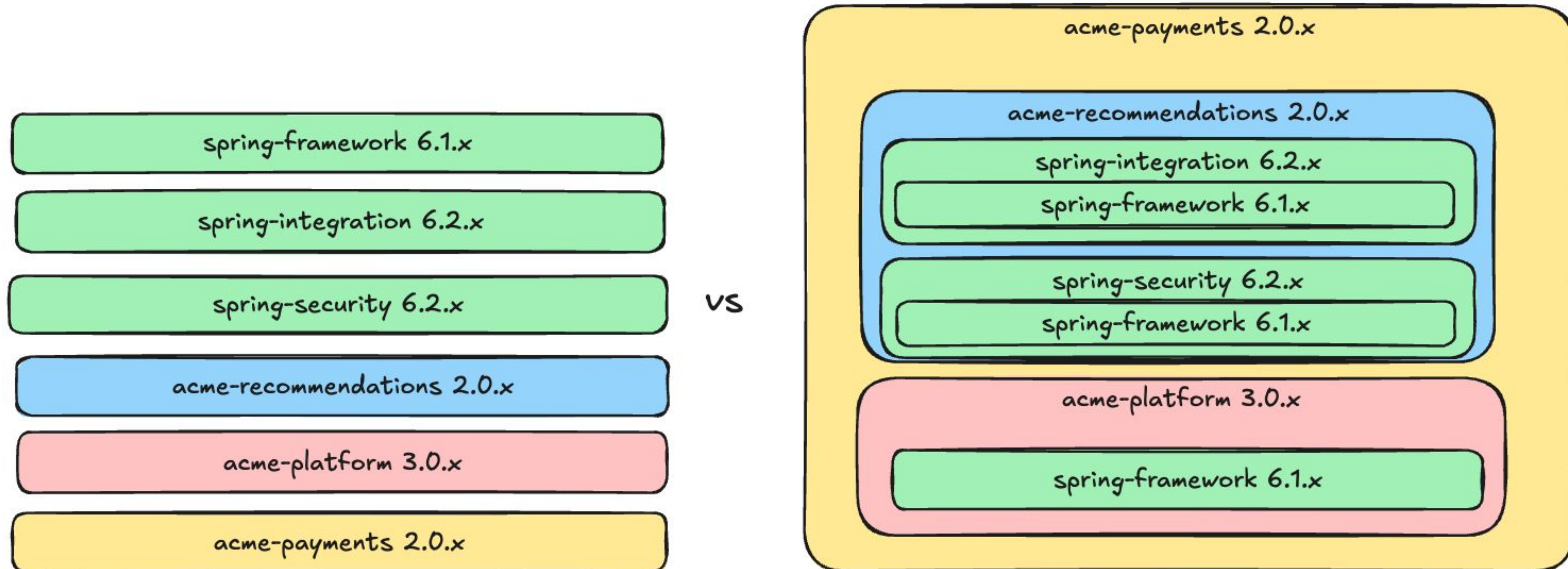
```
---
type: specs.openrewrite.org/v1beta/recipe
name: org.openrewrite.java.spring.boot3.UpgradeSpringBoot_3_3
displayName: Migrate to Spring Boot 3.3
description: |
  Migrate applications to the latest Spring Boot 3.3 release. This recipe
tags:
- spring
- boot
recipeList:
- org.openrewrite.java.spring.boot3.UpgradeSpringBoot_3_2
- org.openrewrite.java.spring.boot3.SpringBootProperties_3_3
```

```
---
type: specs.openrewrite.org/v1beta/recipe
name: org.openrewrite.java.spring.boot3.UpgradeSpringBoot_3_2
displayName: Migrate to Spring Boot 3.2
description: |
  Migrate applications to the latest Spring Boot 3.2 release. This recipe
tags:
- spring
- boot
recipeList:
- org.openrewrite.java.spring.boot3.UpgradeSpringBoot_3_1
```



# LESSONS USING OPENREWRITE FOR UPGRADE ORCHESTRATION

Recipes should be restricted at project level. Recipes should not be composed by the recipes associated to their dependencies.



# LESSONS USING OPEN REWRITE FOR UPGRADE ORCHESTRATION

Recipes should be restricted at project level because:

- Including the recipes associated to their dependencies does not scale. It implies to always look and release new versions if new recipes appear.
- Recipes are quite often published after releasing software.



# TANZU / APPLICATION ADVISOR



It is a CLI tool that **calculates and applies the upgrade plan** across your different projects.

**Just Better for Spring.** Includes an additional Tanzu enterprise recipes for a higher coverage for Spring upgrades.

```
👤 Fetching and processing upgrade plan details [02m 26s] ok
- Step 1:
  * Upgrade spring-cloud-services-starters from 3.3.x to 3.4.x
- Step 2:
  * Upgrade spring-cloud-services-starters from 3.4.x to 3.5.x
- Step 3:
  * Upgrade spring-data-jpa from 2.7.x to 3.1.x
  * Upgrade hibernate-orm from 5.6.x to 6.2.x
  * Upgrade spring-cloud-services-starters from 3.5.x to 4.0.x
  * Upgrade spring-cloud-config from 3.1.x to 4.0.x
  * Upgrade spring-framework from 5.3.x to 6.0.x
  * Upgrade spring-retry from 1.3.x to 2.0.x
  * Upgrade micrometer from 1.9.x to 1.10.x
  * Upgrade spring-ws from 3.1.x to 4.0.x
  * Upgrade spring-cloud-commons from 3.1.x to 4.0.x
  * Upgrade java-cfenv from 2.4.x to 3.1.x
  * Upgrade spring-boot from 2.7.x to 3.1.x
  * Upgrade spring-security from 5.7.x to 6.1.x
  * Upgrade spring-data-commons from 2.7.x to 3.1.x
- Step 4:
  * Upgrade spring-cloud-commons from 4.0.x to 4.1.x
  * Upgrade spring-data-jpa from 3.1.x to 3.3.x
  * Upgrade hibernate-orm from 6.2.x to 6.4.x
  * Upgrade java-cfenv from 3.1.x to 3.2.x
  * Upgrade spring-cloud-services-starters from 4.0.x to 4.1.x
  * Upgrade spring-cloud-config from 4.0.x to 4.1.x
  * Upgrade spring-framework from 6.0.x to 6.1.x
  * Upgrade spring-boot from 3.1.x to 3.3.x
  * Upgrade spring-security from 6.1.x to 6.3.x
  * Upgrade spring-data-commons from 3.1.x to 3.3.x
  * Upgrade micrometer from 1.10.x to 1.12.x
- Step 5:
  * Upgrade spring-cloud-commons from 4.1.x to 4.2.x
  * Upgrade spring-data-jpa from 3.3.x to 3.4.x
  * Upgrade hibernate-orm from 6.4.x to 6.6.x
  * Upgrade java-cfenv from 3.2.x to 3.3.x
  * Upgrade spring-cloud-services-starters from 4.1.x to 4.2.x
  * Upgrade spring-cloud-config from 4.1.x to 4.2.x
  * Upgrade spring-framework from 6.1.x to 6.2.x
  * Upgrade spring-boot from 3.3.x to 3.4.x
  * Upgrade spring-security from 6.3.x to 6.4.x
  * Upgrade spring-data-commons from 3.3.x to 3.4.x
  * Upgrade micrometer from 1.12.x to 1.14.x
```

# DEMO 3

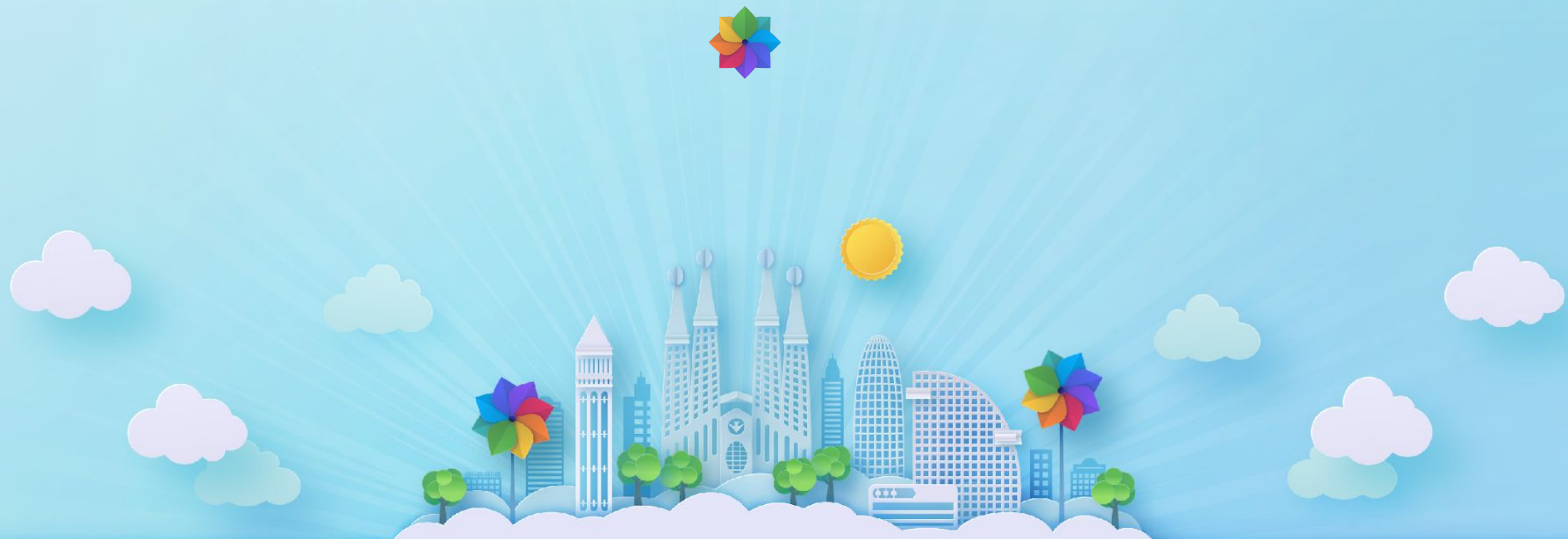
## COMING UP WITH AN UPGRADE PLAN





# QUESTIONS

AND (HOPEFULLY) ANSWERS





# THANKS!!!



Raquel Pau

Vicente Soriano

## And please, rate the session and leave some feedback!

