Microservices based Stream and Batch Data Processing

Glenn Renfro | Sabby Anandan

Glenn Renfro

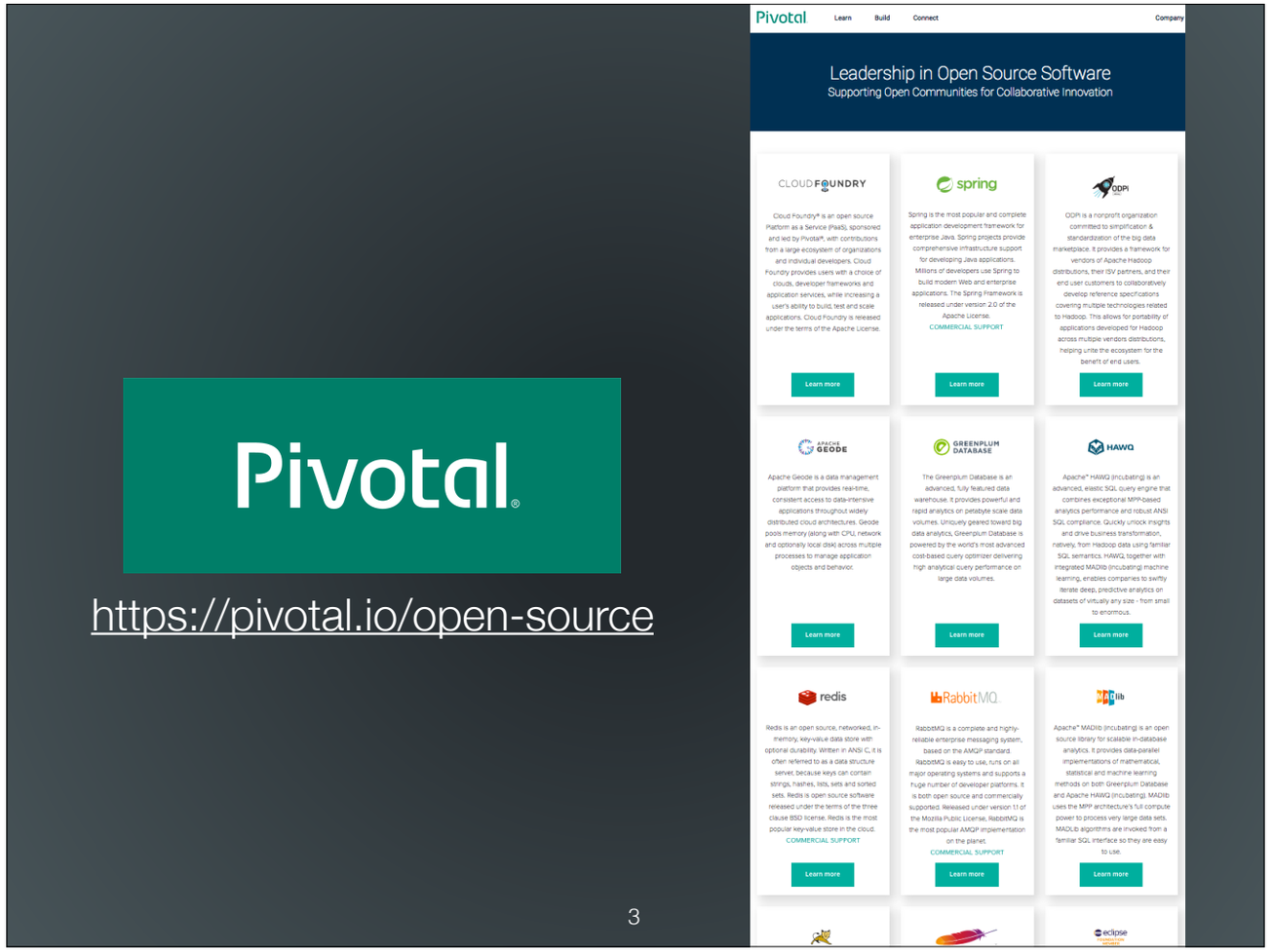
@cppwfs

@cppwfs

Sabby Anandan

@sabbyanandan

@sabbyanandan



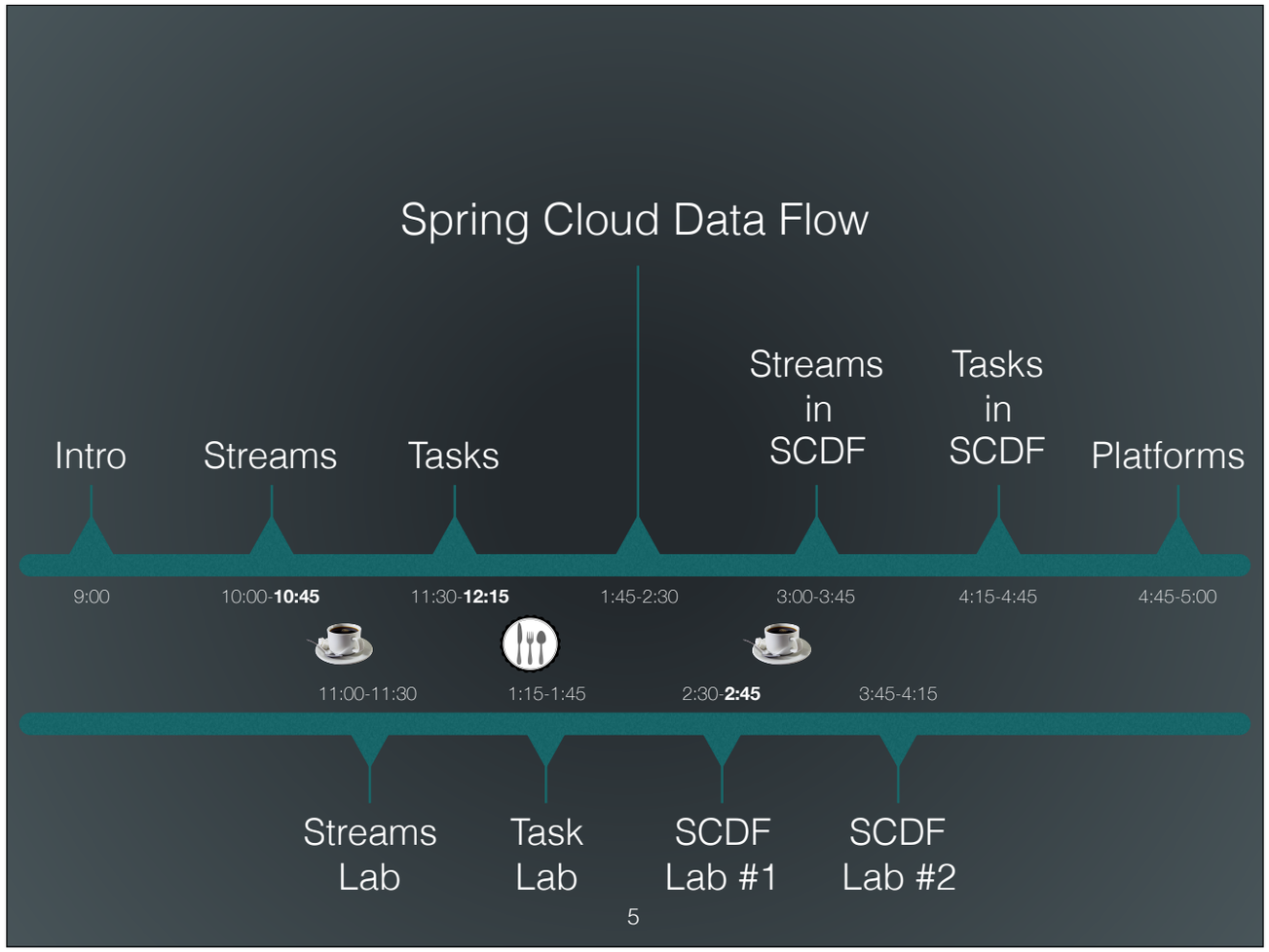
We are both, Glenn and I, are from Pivotal and as a company, we are all about open-source. You can navigate to https://pivotal.io/open-source to review the open-source portfolio at Pivotal. Of course, Spring and the portfolio of projects is one of our primary contributions to the open-source ecosystem.

App frameworks, PaaS, application server, messaging middleware… you name it; we are a proud to embrace, adapt, and contribute to open-source.

We have a lot of collateral to cover today; both, Glenn and I are looking forward to sharing and in reciprocation, we are here to listen and learn from you all, so please do not hesitate to interrupt us wth questions as we move forward with the presentations.

We have very carefully curated the content for this workshop. Before we dive into the core of this workshop, we would like to set some expectations, so let's begin with house-keeping.

1) This workshop is all about Spring Cloud Data Flow.
2) You will learn specifics both from business and technical angle.
3) We will dig deeper into SCDF and the ecosystem of projects.
4) As we dig-in, we will make sure there's room for hands-on lab, questions and comments - we want this to be a interactive session. Questions are welcome at ay time in the presentation or labs.
5) There's dedicated time for labs. We will get to the details momentarily.
6) To keep up with the time and the topics we would like to cover, we will do extended discussions on use-cases/requirements in breaks and/or after the workshop. So, please make note of those questions as we progress today - we can circle back as time permits.
7) We have sent out the class-prep email a week back. If you haven't already, don't worry, we will set it up just before the first lab session.
8) ..

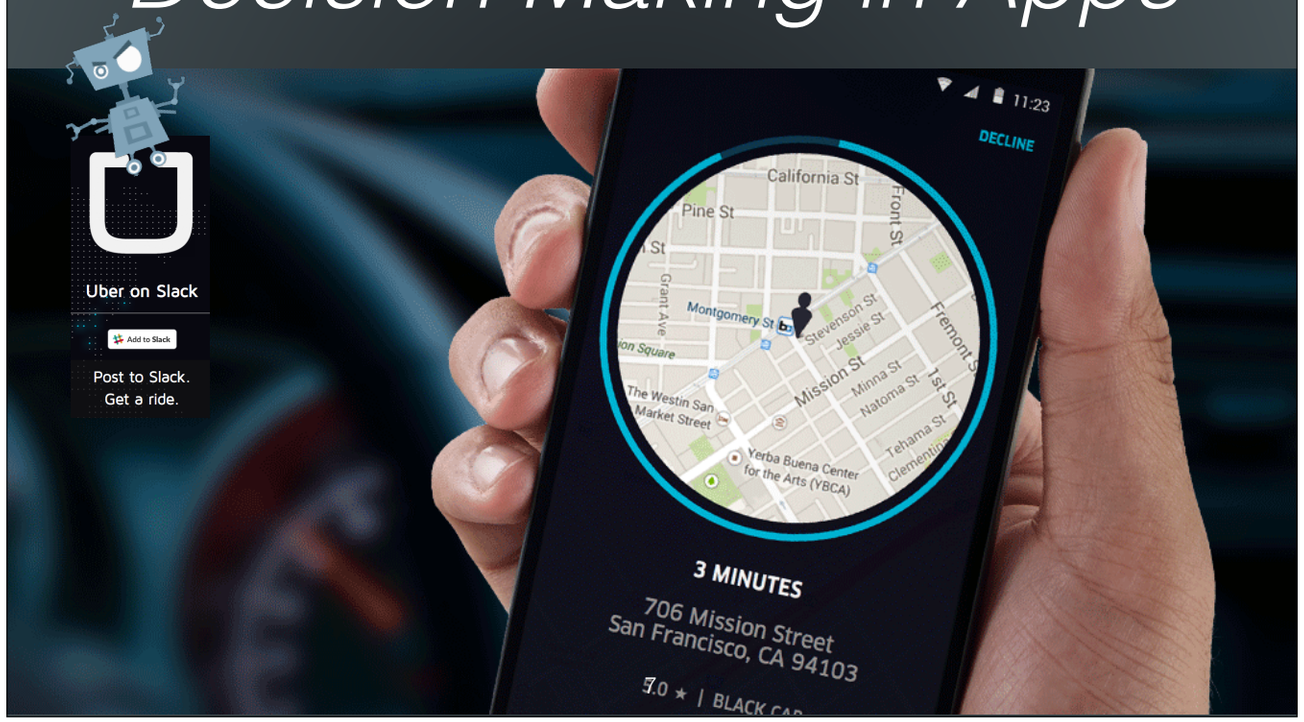


.45 - Introduction (45 mins)
1.15 - SCSt (45 hour) + Lab (30 min)
1.15 - Tasks (45 min) + Lab (30 min)
1.5 - Architecture (1 hour) + Lab (30 min)
1.5 - Streams in SCDF (1 hour) + Lab (30 min)
0.5 - Tasks in SCDF (30 min)
0.5 - Platforms (30 min)

Let’s begin with few questions.

1) How many of you are already familiar with Spring and Spring Boot?
2) This is a micsroservice architecture focussed workshop. How many don’t know what it means?
3) Anyone heard of 12-factor applications?
4) Who here is in Data Engineering, Data Science or generally involved in Data teams at your company?
5) Anyone with no Java experience at all?

Rise of Data Integration and Decision Making in Apps

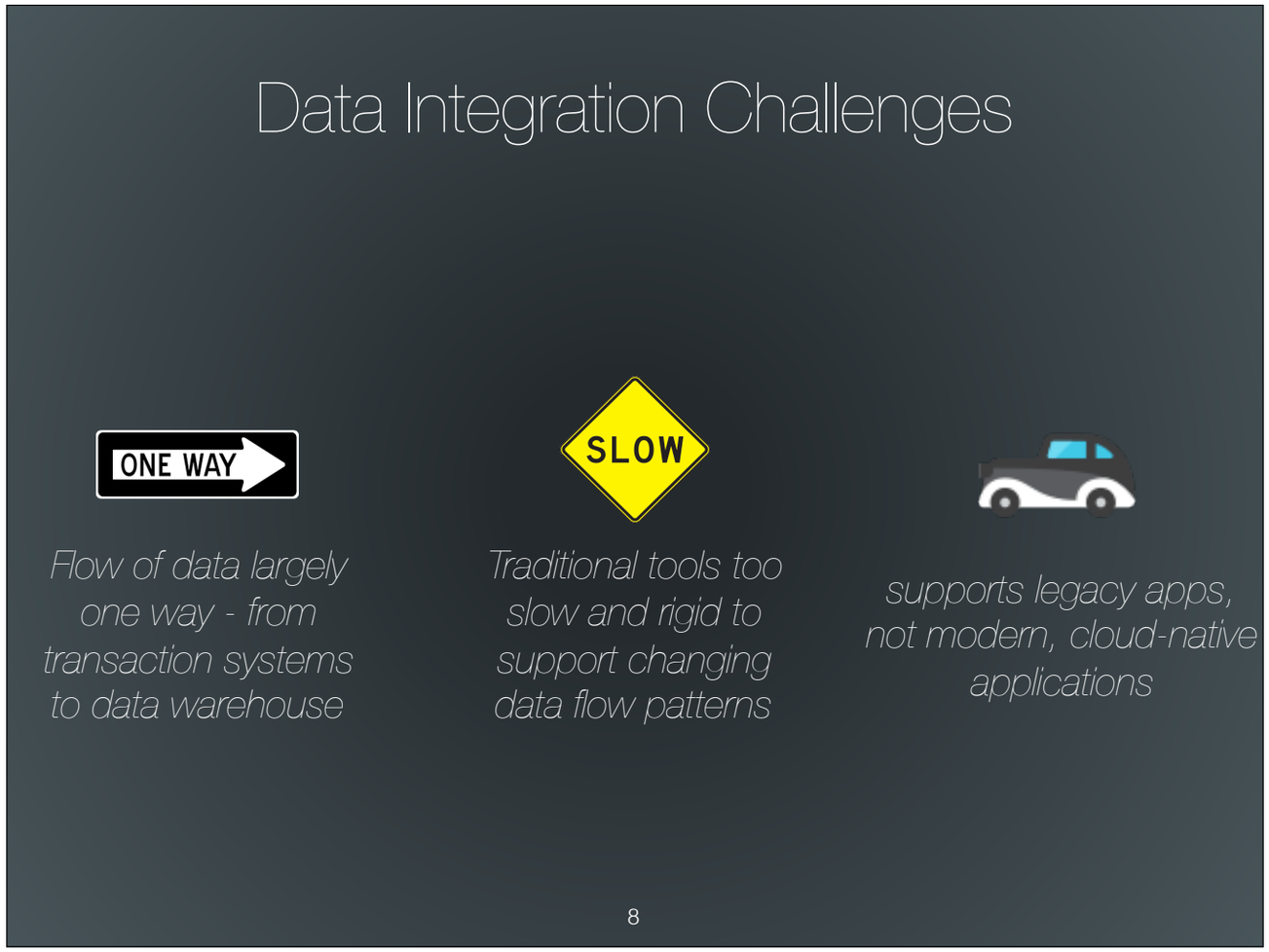


Data = Insights = Business Value

Insights are Perishable.

We have data in our hands; everything we do, every action tells a story. Sometimes opening an app is all you'd have to do, a lot of context specific insights are at your service. Take Uber app for example - who here use Uber/Lyft like services? (almost everyone)

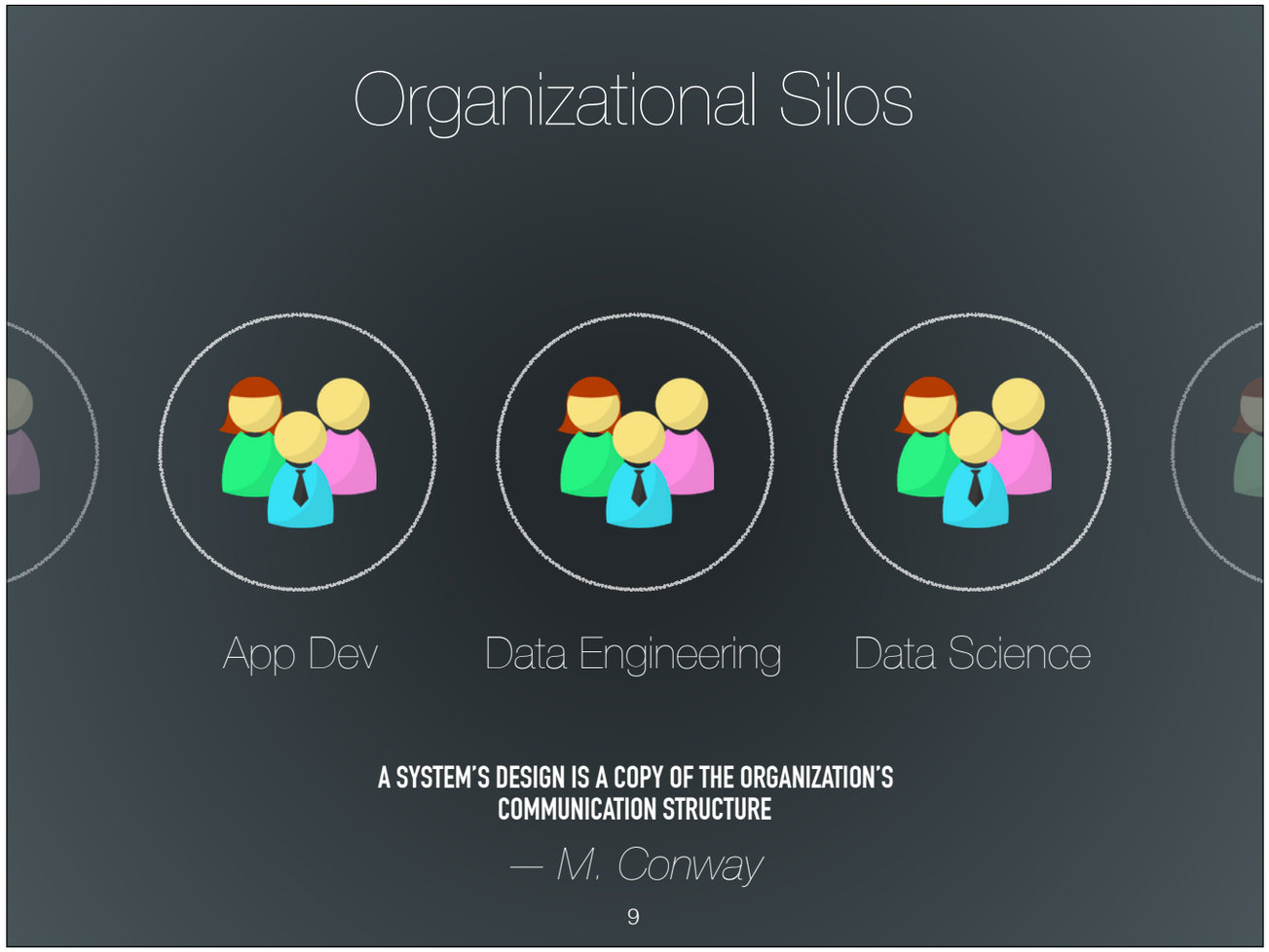
Likewise, every enterprise have plenty of data both from internal and external sources. They want to take advantage of it; it is the new OIL as some say it.



However, in order for such perishable insights to work in our favor and in a timely manner *(because, context is important - what's the point if the Uber App wrong data/map/drive-info?)*, we would have to get around the current set of challenges.

Let's review some of the data integration challenges at a high-level.

1) Legacy ETL/ELT pipelines give ways to linear data processing;
2) Traditional ETL tooling are slow and often don't allow new and dynamic data patterns;
3) Legacy data integration workloads are not easy to modernize and/or port to cloud-native style architectures



As for the organization challenges,

In an enterprise, we see silos. The data organization is disconnected from modern/cloud-native teams at the enterprise.

The tooling, best practices, and the software development processes are significantly different. It is different because of the serviceability and roadmap are different.

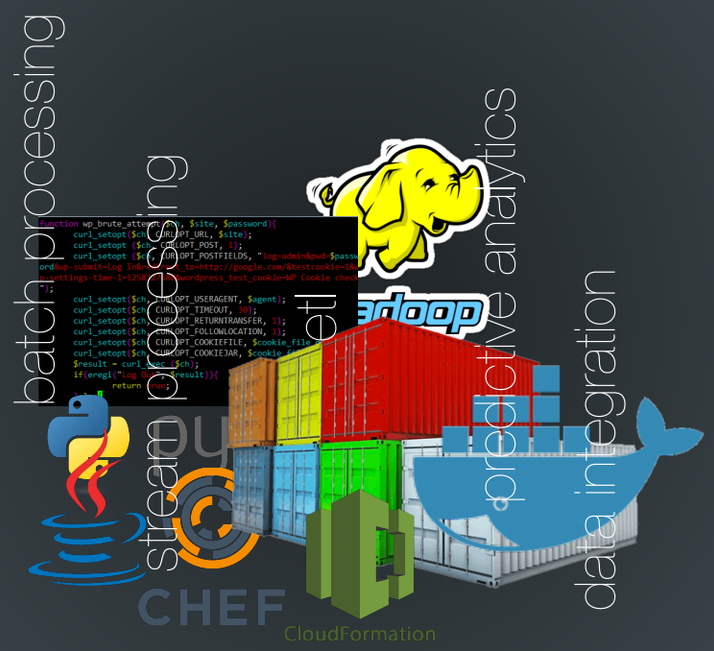
Often times, the tooling and practices followed by the data organization doesn't give room to rapid iterations to deliver quicker value to the business.

From Adrian C.: It's common to talk about "people, process and technology" when we look at changes in IT.

From Adrian C.: As an analogy, think how long it would take to make a model house starting with a ball of clay, compared to a pile of Lego bricks

https://medium.com/@adrianco/evolution-of-business-logic-from-monoliths-through-microservices-to-functions-ff464b95a44d#.xuureb21w

Data Pipelines in Reality



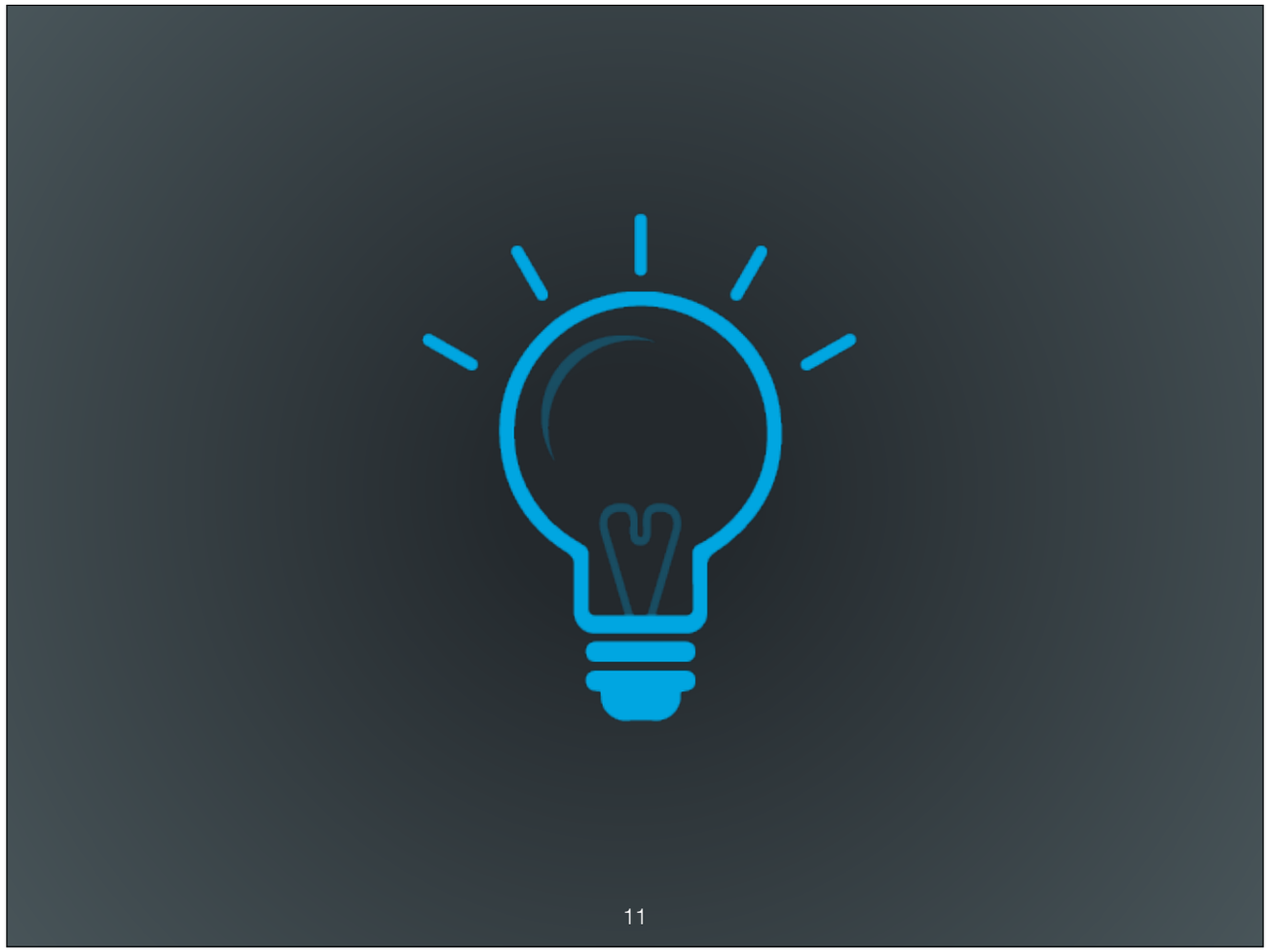
As a last broad type of challenge,

When an enterprise decides to build their own set of tooling and stack, it is not pretty.

Standards, expectations and proven ideas are once again being reinvented - why?

The teams tend to pickup tools as a tactical measure and it ends up being used in the production and because of that, they end up maintaining it like forever.

Given the varying requirements within the enterprise, it gets forked a few times, too. More fragmentation overall!



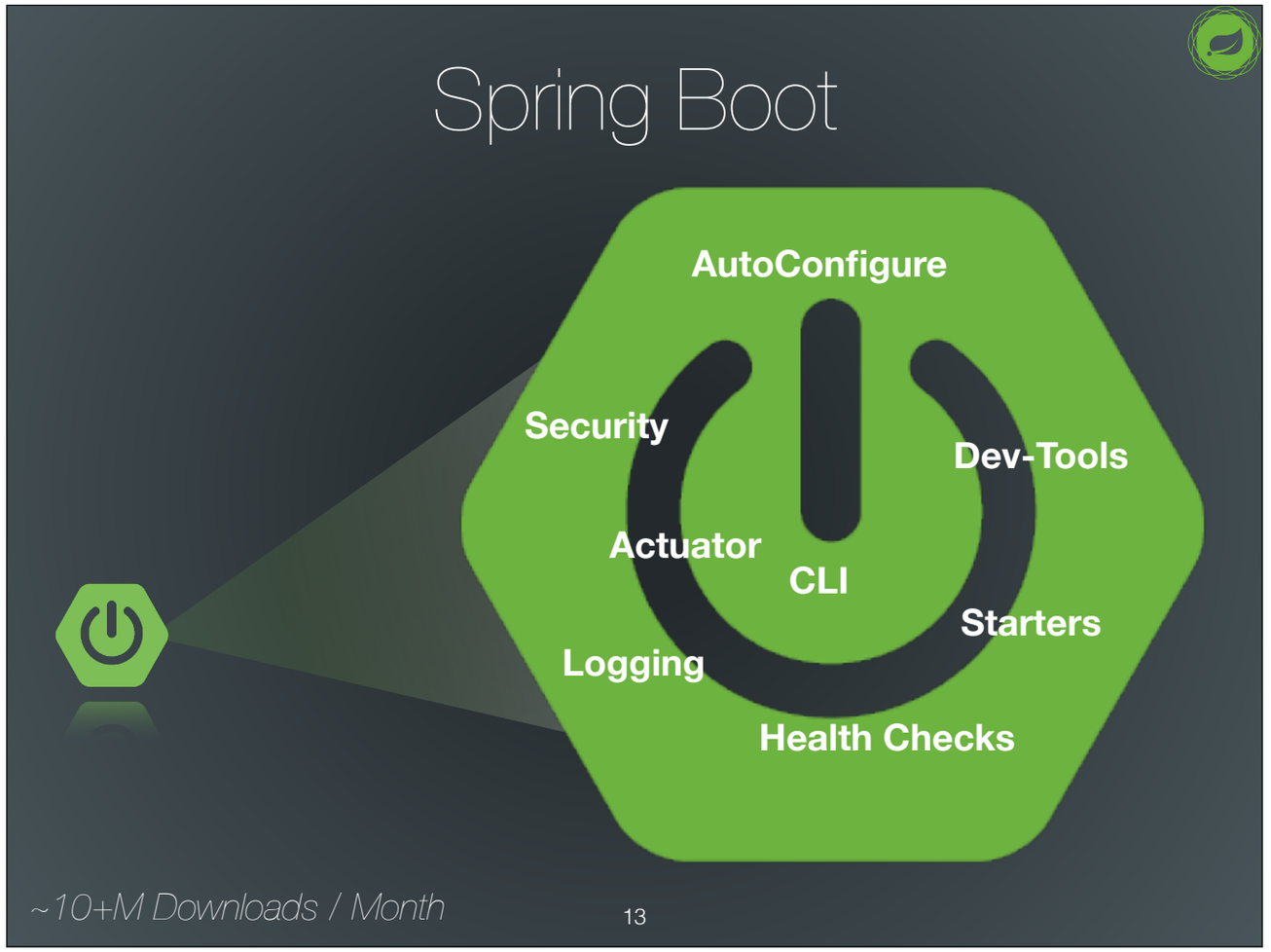
OK, so that’s some of the general set of challenges that we reviewed so far. Now that you’re aware of the problem space, let’s dive into solving them. Let’s go!

We are going to assume you’ve enough background about microservices.

Microservices Turn Out to be Great for Data Processing

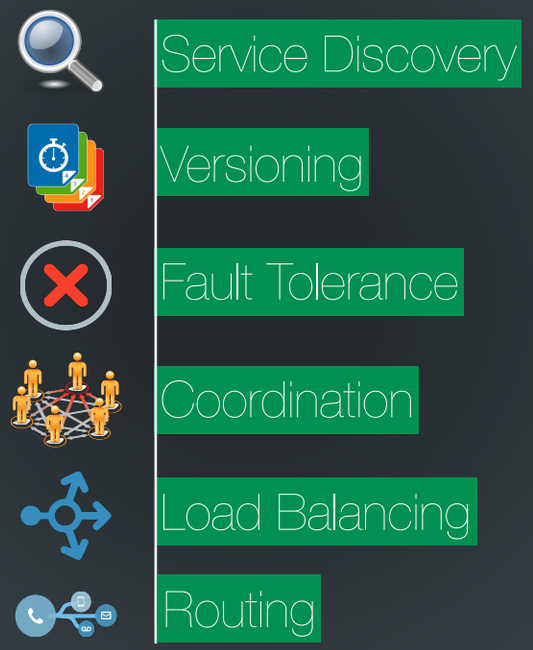
Developing cloud-native applications provide:

- *continuous delivery*
- *higher reuse*
- *scalability and resilience*



Create standalone production grade spring applications that just “runs”!
Auto configure Spring where possible.
Don’t worry about boilerplate configurations; let Spring handle it for you; focus on the business logic instead.

Emerging Challenges



Microservices is no free-lunch; there are challenges, in fact, the distributed systems challenges are pretty significant.

Spring Cloud

NETFLIX | OSS

Hystrix
Eureka
Ribbon
Zuul
Feign

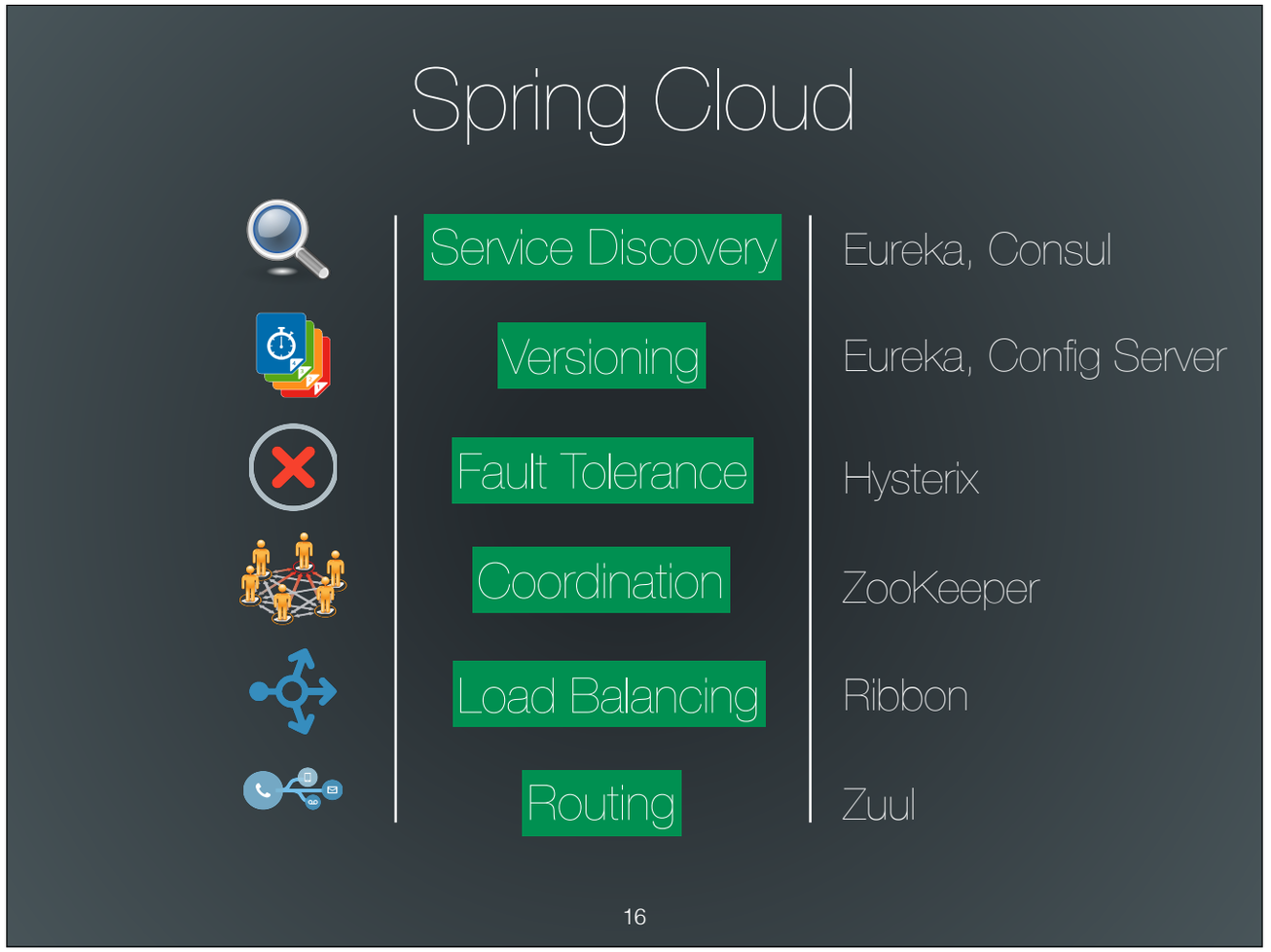
Service Discovery
Config Server
Control Bus

Config Server

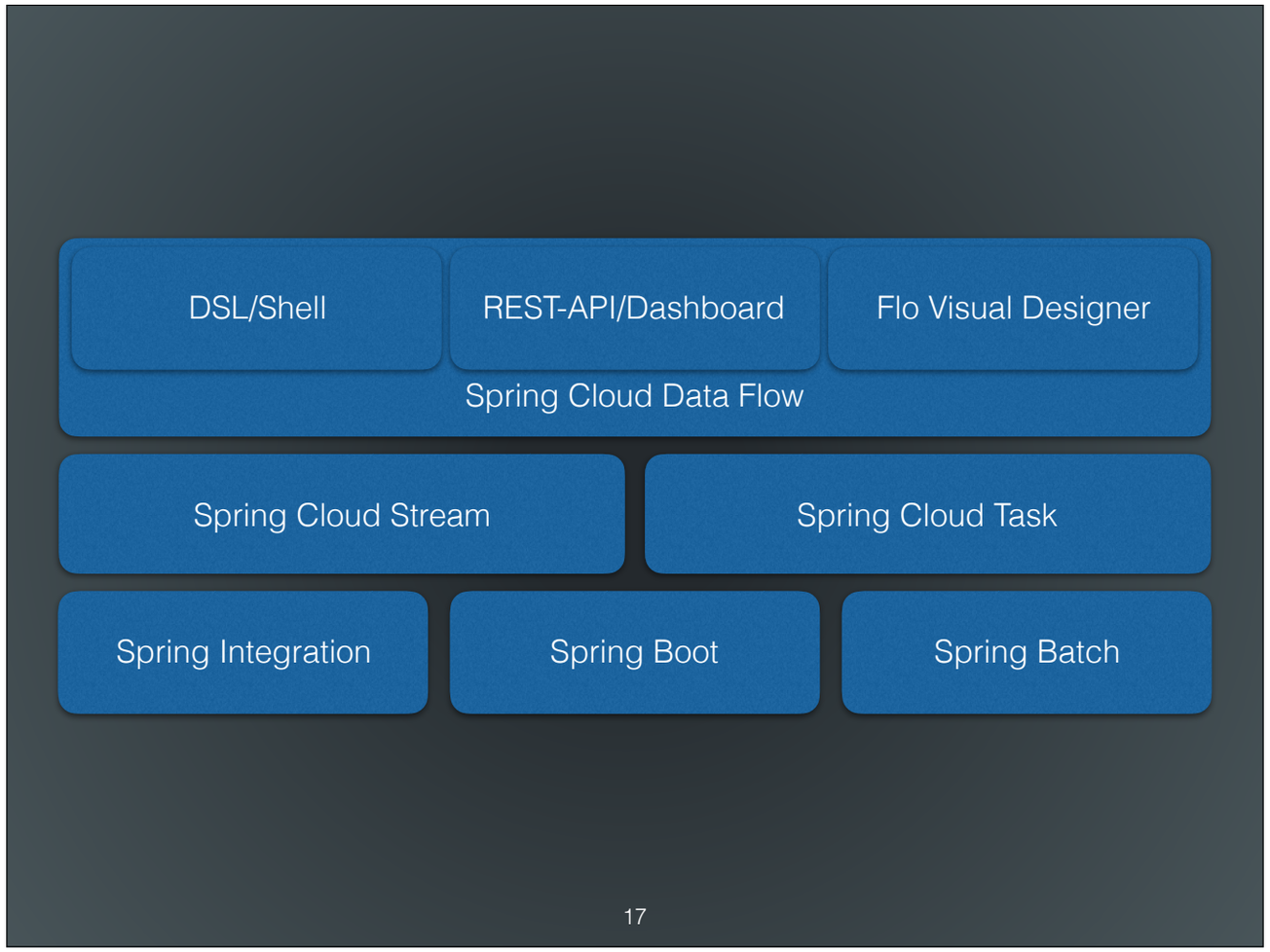
Zookeeper

Distributed Tracing

Here comes Spring Cloud! Takes the distributed systems challenges head-on!

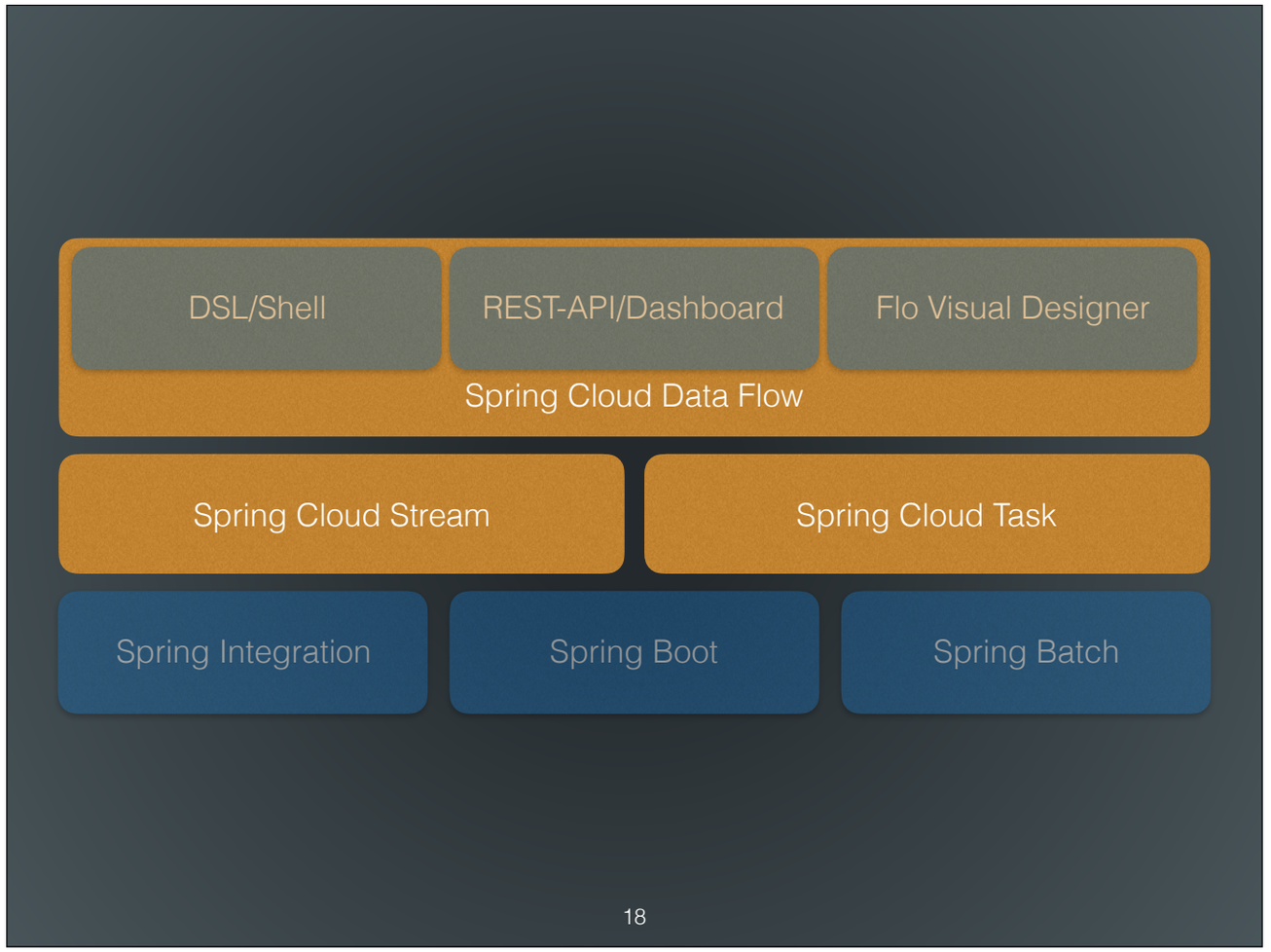


Microservices is no free-lunch; there are challenges, in fact, the distributed systems challenges are pretty significant.



Spring Cloud Data Flow is built from the group up with microservice architecture in mind. The product is made up of several microservice projects and sub-modules. This picture highlights the top-level projects alone and we will cover all of them today.

You'll see this visual a few times today. This is a running roadmap for this workshop. Based on the topic we are to cover, we will see color-coded boxes in this visual - you will get an idea what we are cover.



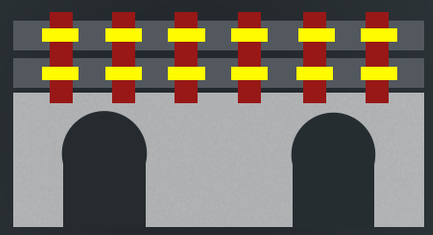
This is an example.

Now, we are going to introduce you all to SCDF and the top-level projects it builds on.

Data Architecture

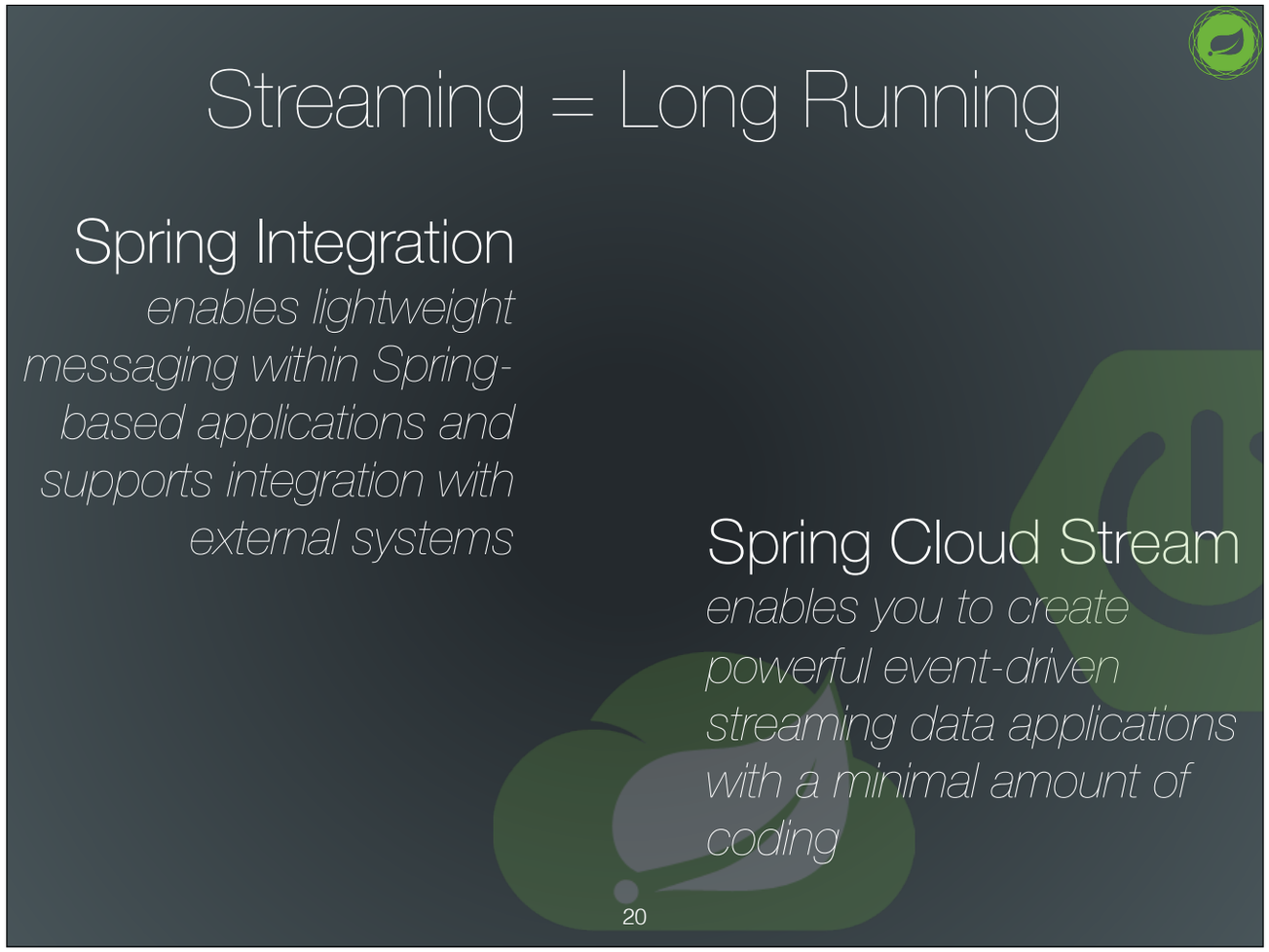
Stream

longer running services



Batch

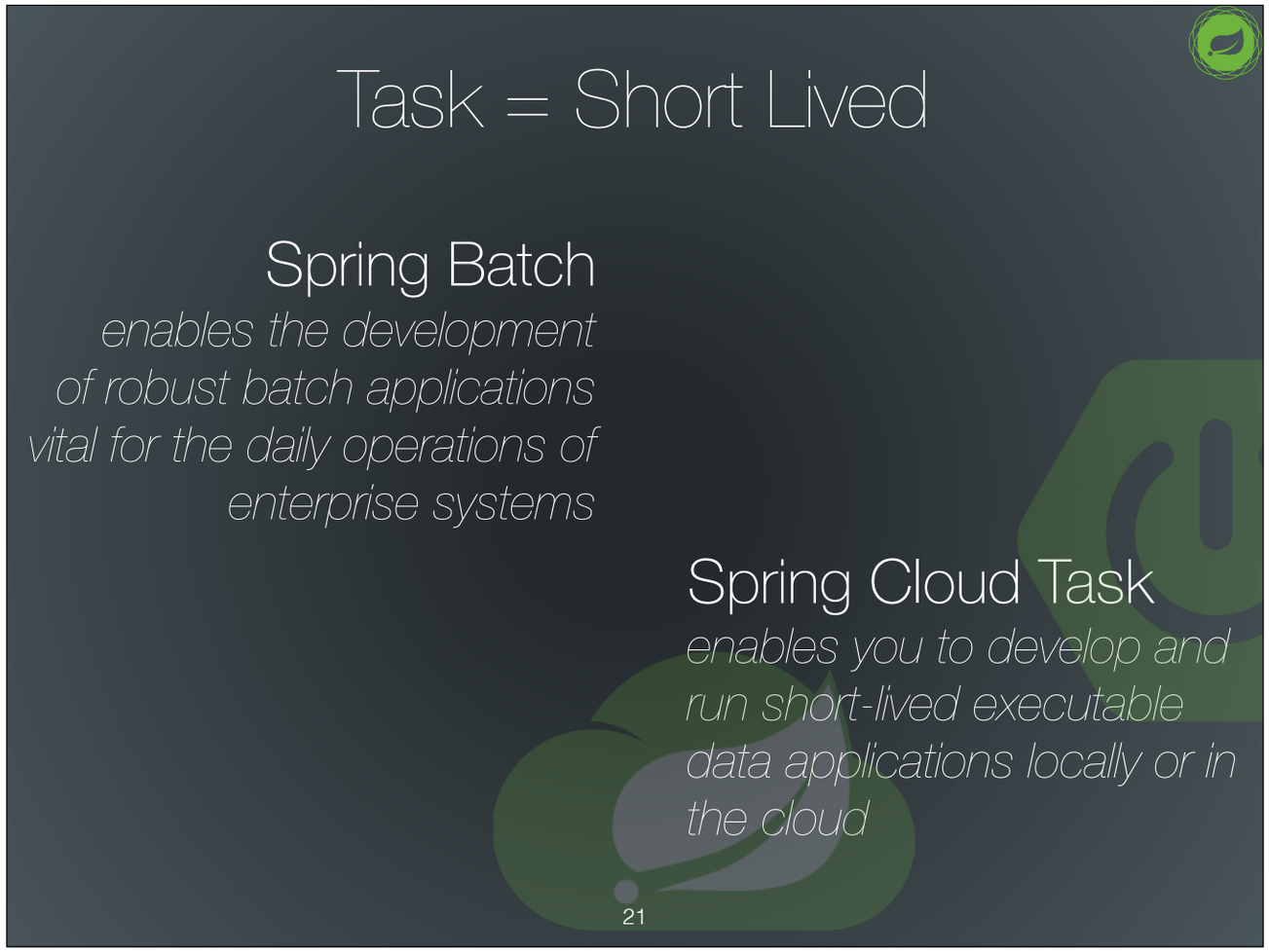
short lived processes



In Spring's portfolio, we have have had Spring Integration (SI) for about 9+ yrs in production. This project allows you to connect to external system such as file-systems, messaging middlewares, hadoop, sql-no-sql stories, etc.

To further simplify developing integration applications with SI at its foundation, Spring Cloud Stream project was spun-out. Apart from SI, this project also builds upon Spring Boot and Spring Cloud patterns and best practices, too.

In 1-line, SCSt provides event-driven microservices framework, so you can quickly build standalone production grade streaming applications. You're developing a well-known and familiar Spring Boot applications at the end of the day.

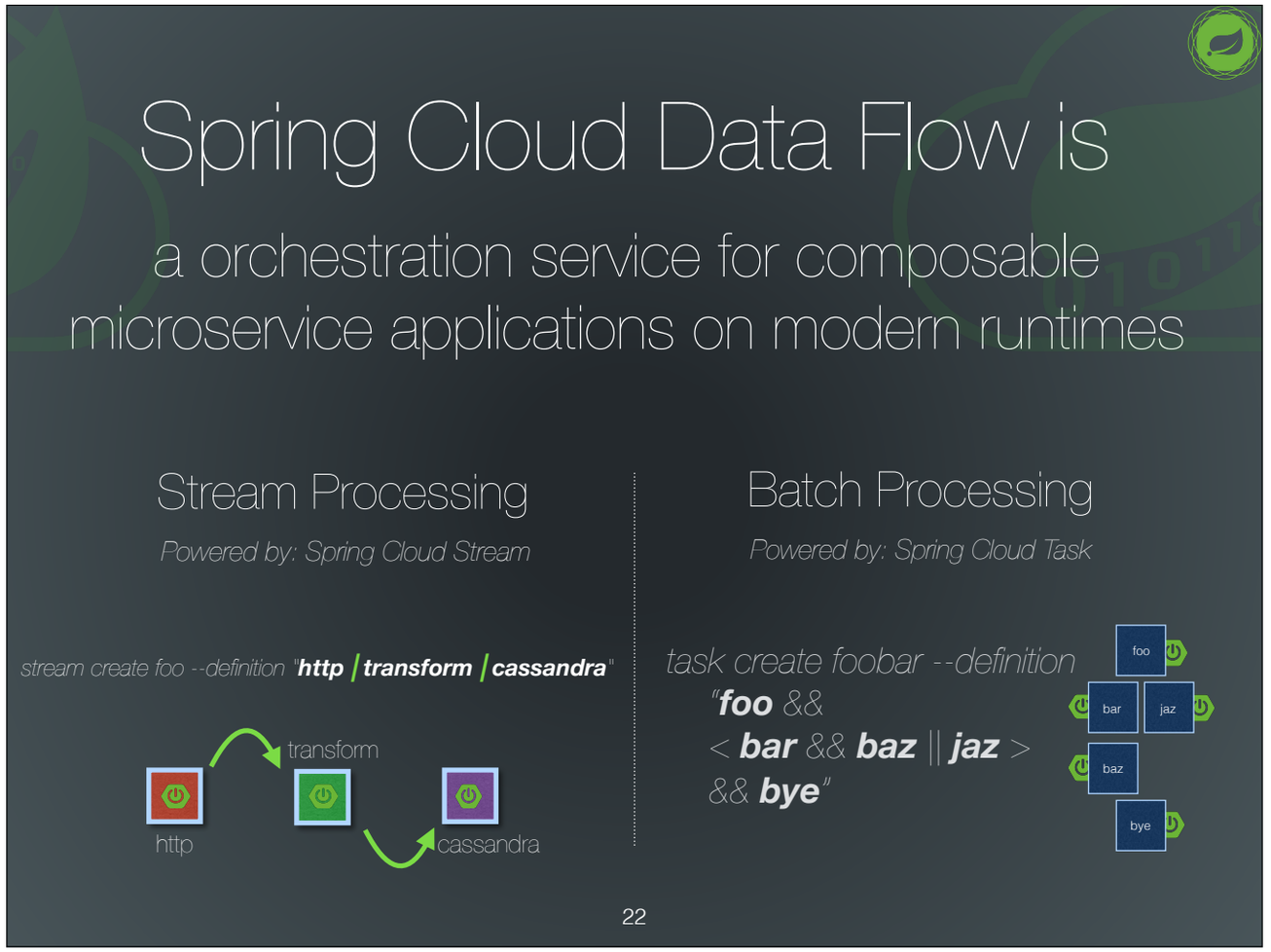


We have seem streaming solutions, but there's batch/task based workloads that is equally important in enterprise setting.

In Spring's portfolio, we have have had Spring Batch (SB) for about 9+ yrs in production. This project allows you to build batch applications rapidly.

To further simplify developing batch applications with SB at its foundation, Spring Cloud Task as project was spun-out. Apart from SB, this project also builds upon Spring Boot and Spring Cloud patterns and best practices, too.

In 1-line, SCT provides short-lived microservices framework, so you can quickly build standalone production grade task applications. They are short-lived because the application runs and it is live ***only*** as far as the business logic embedded in it is operational. Again, Spring Boot applications all the way.

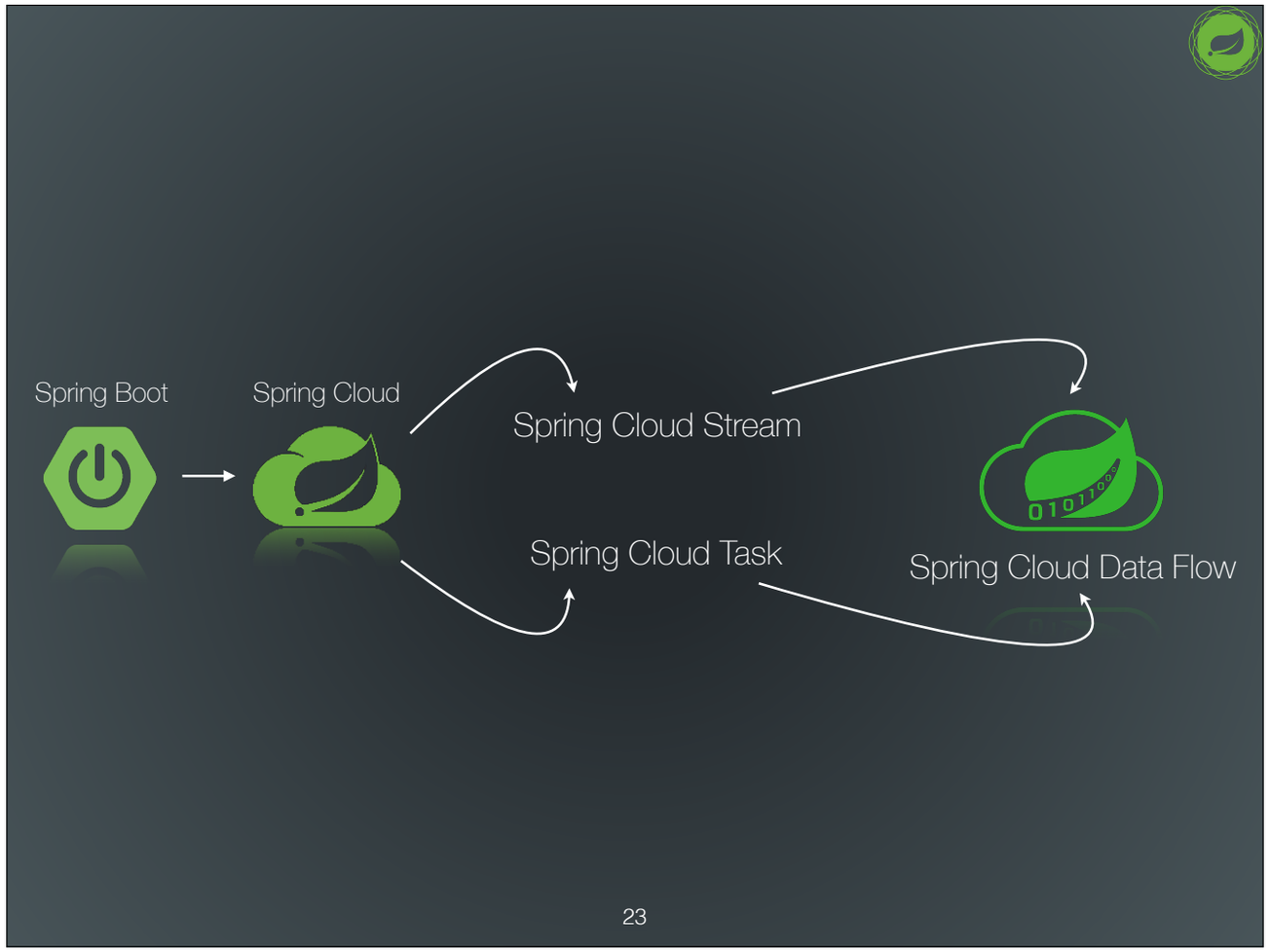


You can build production grade SCSt and SCT applications as standalone stack; however, once you have a several of them and you'd need an easy to orchestrate and operationalize them in in the wild.

That's where Spring Cloud Data Flow comes in.

We call it: "a orchestration service for composable microservice applications on modern runtimes"

Let's review SCSt and SCT in the context of SCDF.



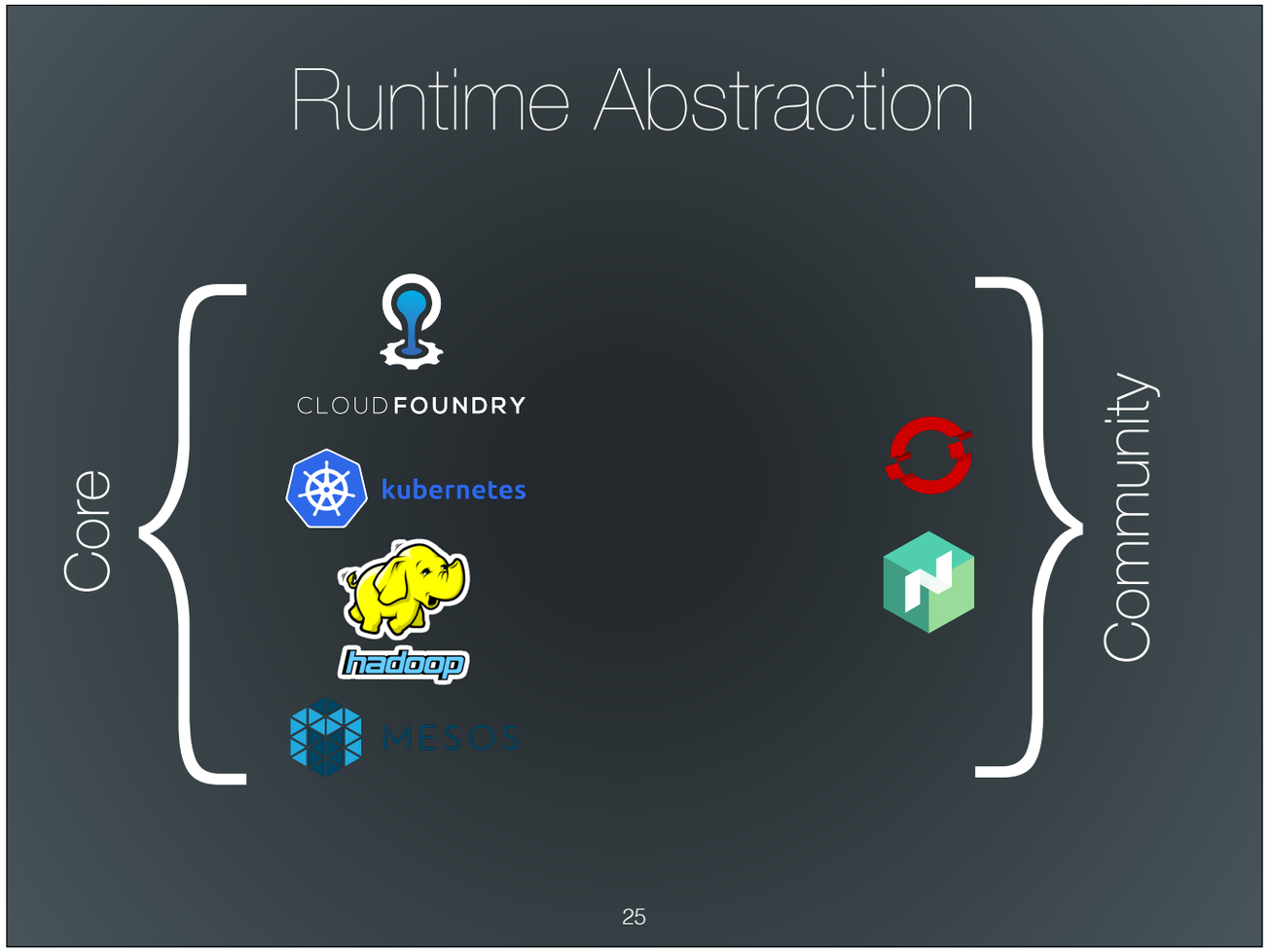
You can build production grade SCSt and SCT applications as standalone stack; however, once you have a several of them and you'd need an easy to orchestrate and operationalize them in in the wild.

That's where Spring Cloud Data Flow comes in.

We call it: "a orchestration service for composable microservice applications on modern runtimes"

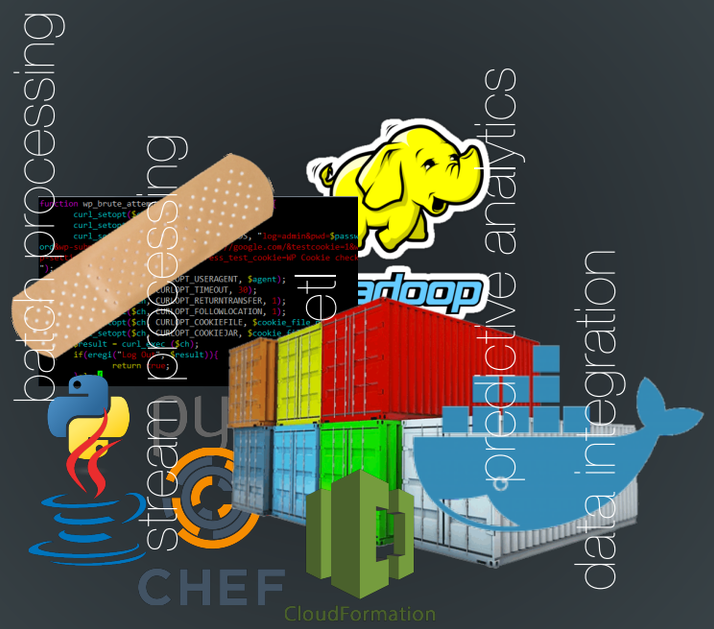
Let's review SCSt and SCT in the context of SCDF.


Demo: Twitter Analytics



SCDF builds an abstraction that’s easy to extend. We are not too opinionated. You have options to operationalize SCDF in variety of runtimes.

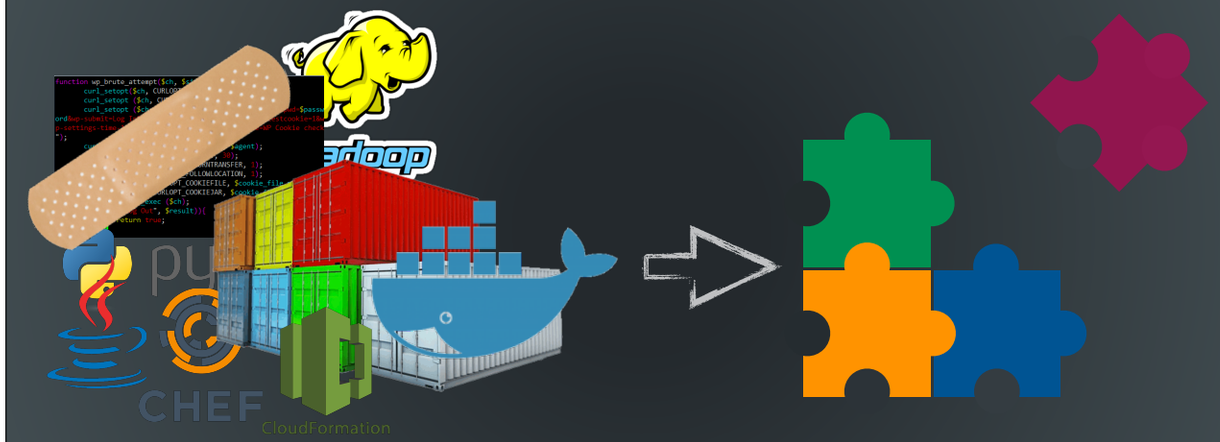
Data Pipelines in Reality



Let’s revisit the challenges that we reviewed in the beginning.

You don’t have to fire-fight the operational requirements constantly; why do you need to build your homegrown data pipeline solution when you have several options?
There’s no tool that fits all the requirements.
Instead of choosing a technology or building your own stack, please embrace the ones that matter for your business.

Data Pipelines in Reality



Do you want to be in fire-fight mode constantly or you’d prefer being more productive and focus on business values that matter to your customers and of course your organization?

Repeatable Process

Spring Boot Apps

Consistent Dev, Testing, and CI Tooling

Extensible Abstractions

Declarative DSL and Pipelines

Drag & Drop GUI and Pipelines

REST-APIs and Rapid Dashboarding

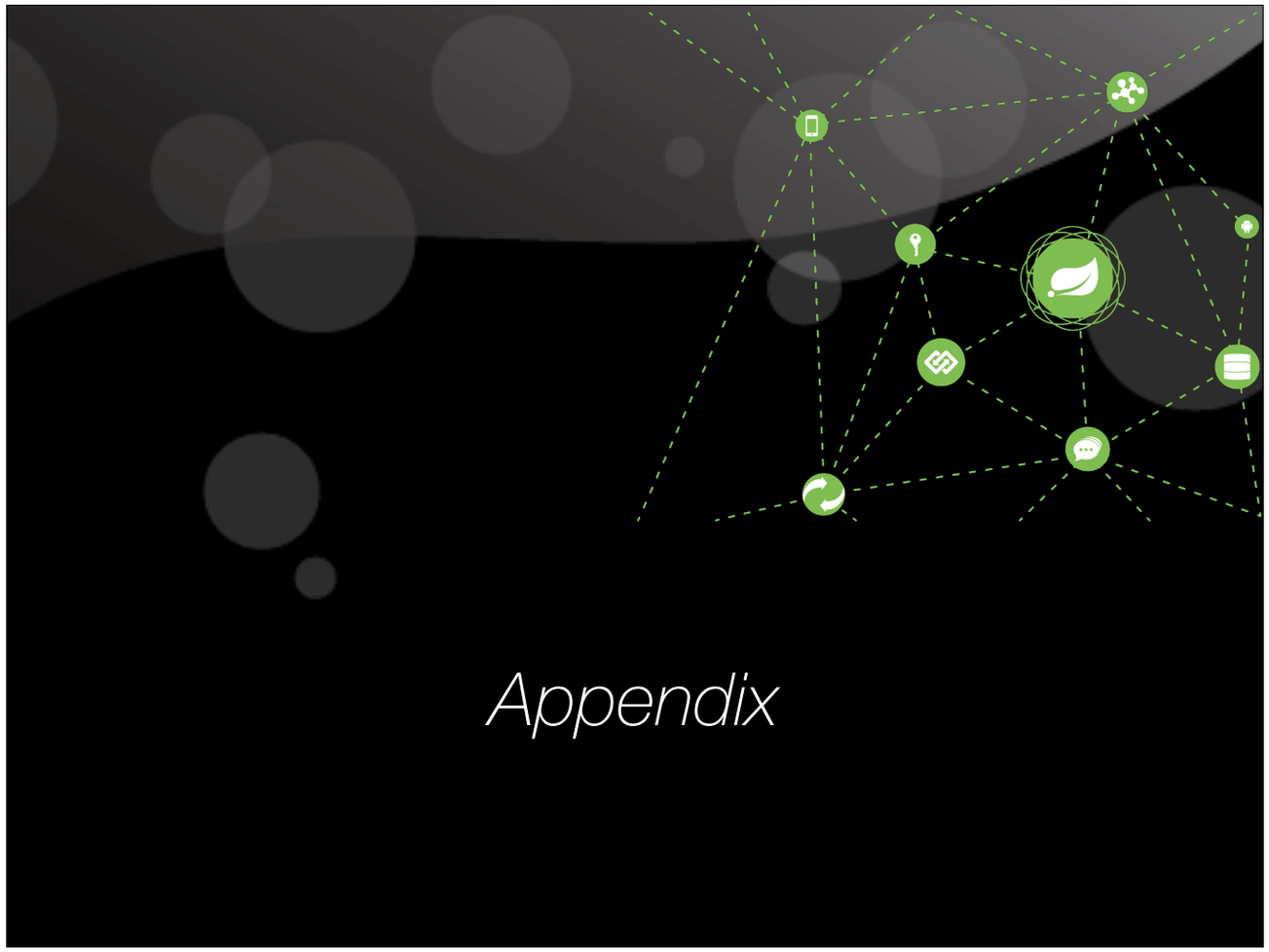
Unified Stream and Batch Processing

“Simple things should be simple; complex things should be possible”

Consistency

App Dev | Data Engineering | Data Science

Spring, Spring Boot and SCDF offer consistent set of tools, patterns, and best practices. Silos could meltdown to form a more coherent experience across the all the teams.

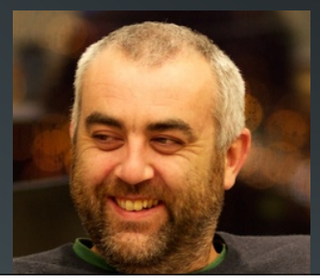


Appendix

Microservices

a single application as a suite of small services, each running in its own process and communicating with lightweight mechanisms, often an HTTP resource API

James Lewis and Martin Fowler
ThoughtWorks



Microservices

a single application as a suite of small services, each running in its own process and communicating with lightweight mechanisms, often an an **HTTP** resource API