Eiffel for beginners

Magnus Bäck, Axis Communications
Emil Bäckmark, Ericsson
Mattias Linnér, Ericsson

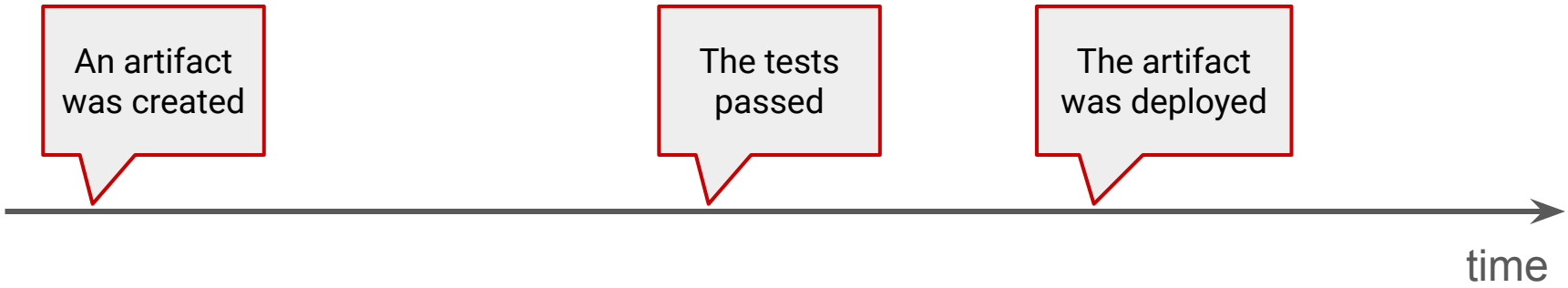
Eiffel

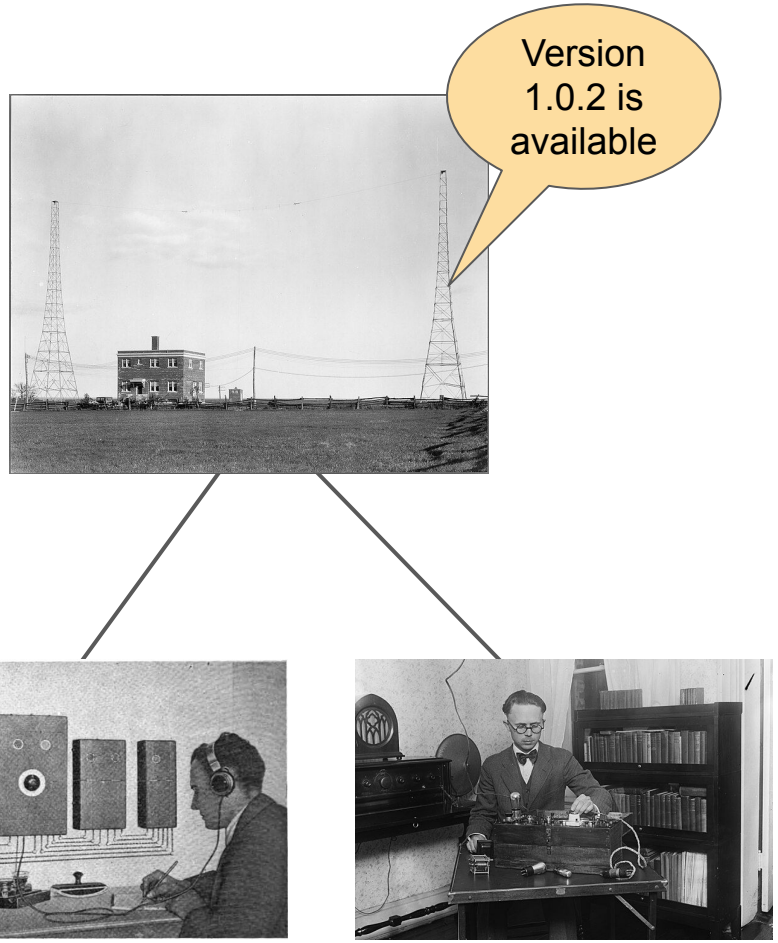
Agenda

- Eiffel Introduction
- <Break>
- Eiffel Use Cases at Ericsson
- Eiffel Use Cases at Axis Communications
- (Eiffel Community Landscape)

Prelude

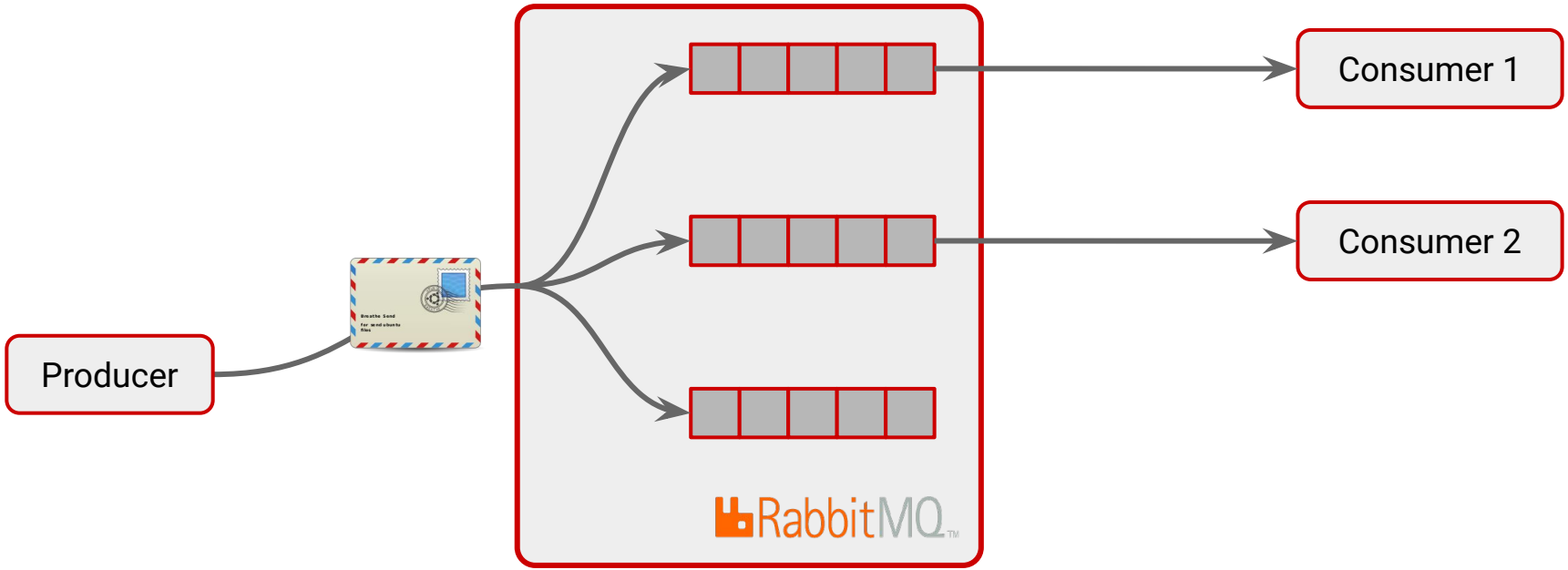
Eiffel


An artifact was created
The tests passed
The artifact was deployed
time

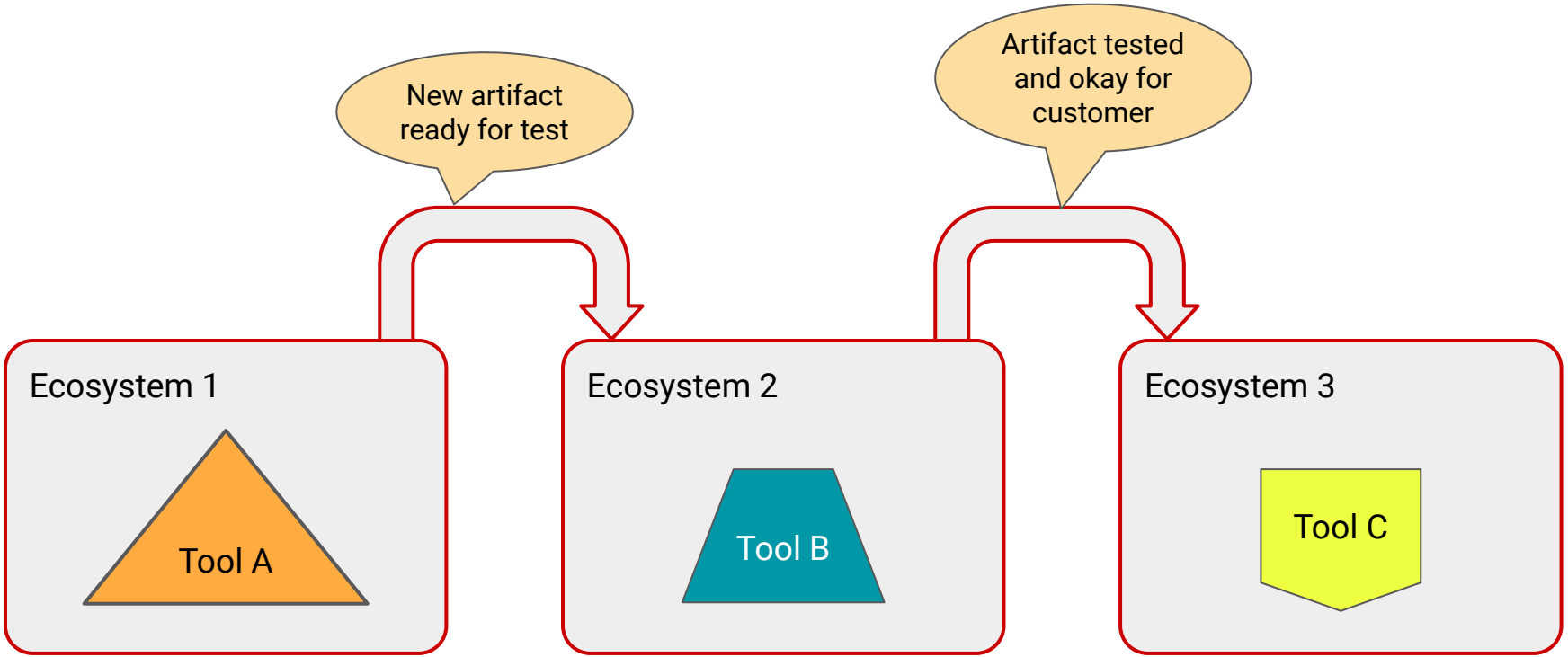


Eiffel

Asynchronous messaging



Using events to implement ecosystem interoperability



Eiffel

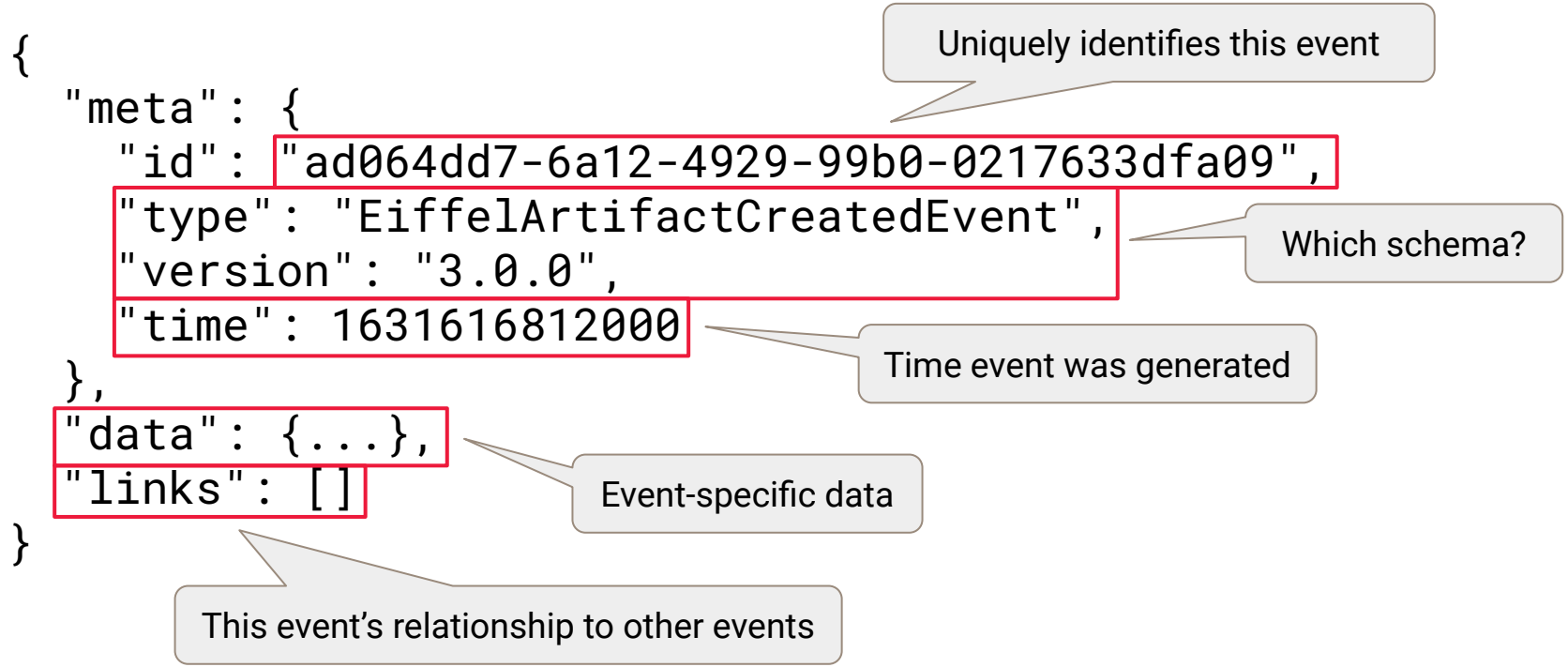
Activities

- Builds
- Test executions
- Promotions
- Deployments

Entities

- Source code
- Build artifacts
- Test results
- Tickets (issues)
- Environments

What does an Eiffel event look like?



Eiffel

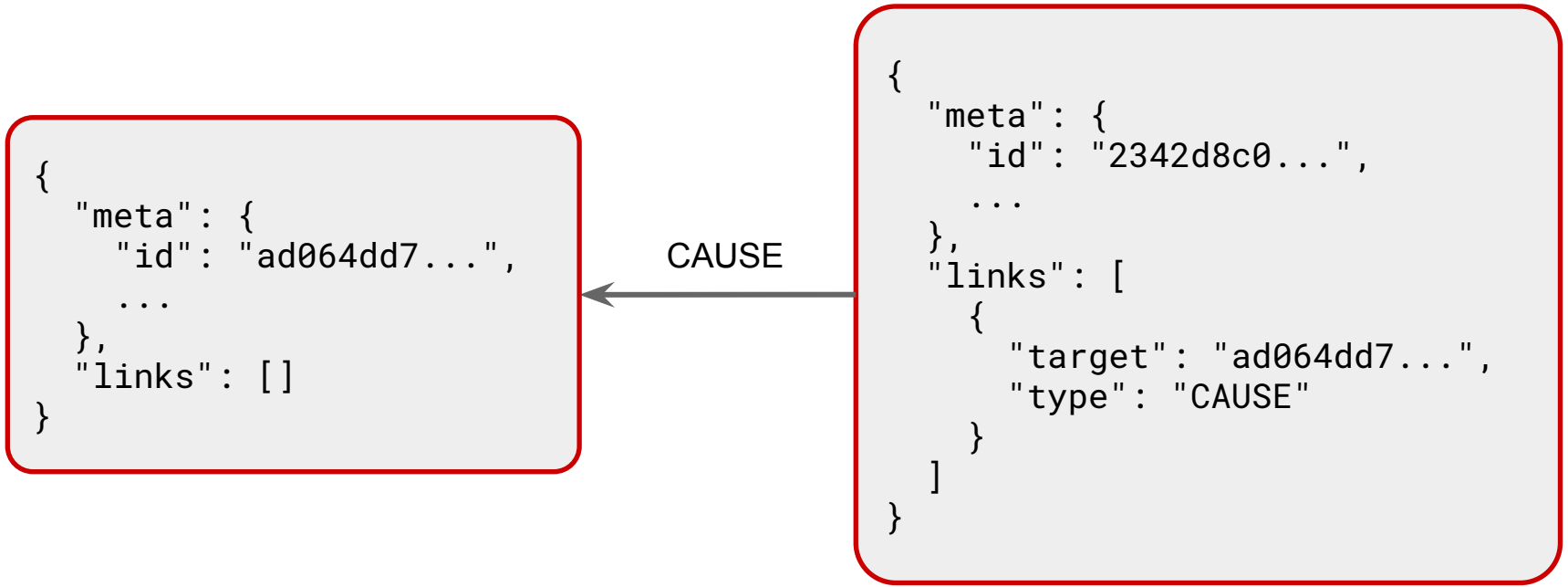
Activities

- Builds
- Test executions
- Promotions
- Deployments

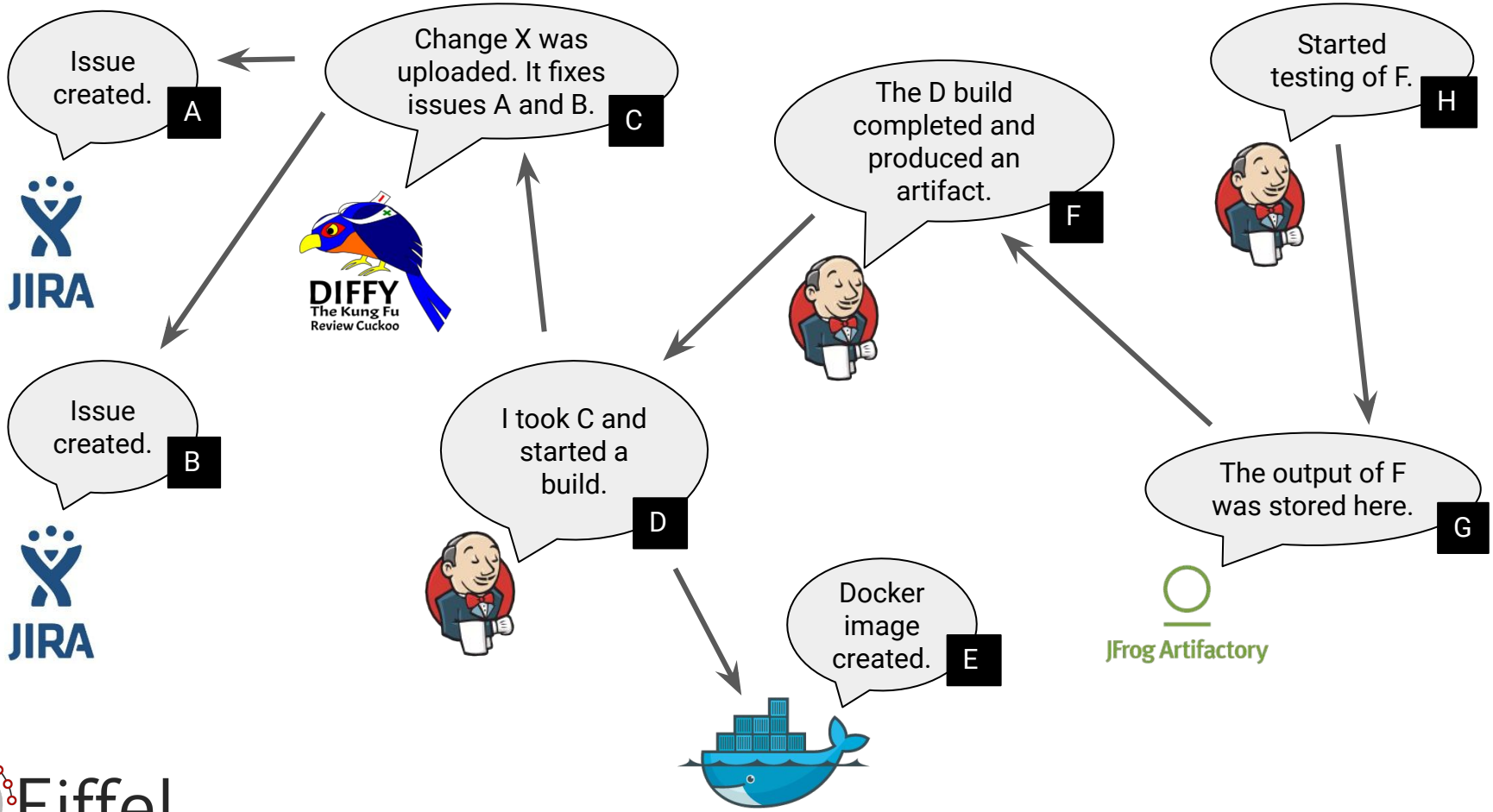
Entities

- Source code
- Build artifacts
- Test results
- Tickets (issues)
- Environments

Links describe relationships between events



Eiffel


Issue created.
A
Change X was uploaded. It fixes issues A and B.
C
The D build completed and produced an artifact.
F
Started testing of F.
H
JIRA
DIFFY
The Kung Fu Review Cuckoo
Issue created.
B
I took C and started a build.
D
The output of F was stored here.
G
Docker image created.
E
JIRA
JFrog Artifactory

Walkthrough of Eiffel event types

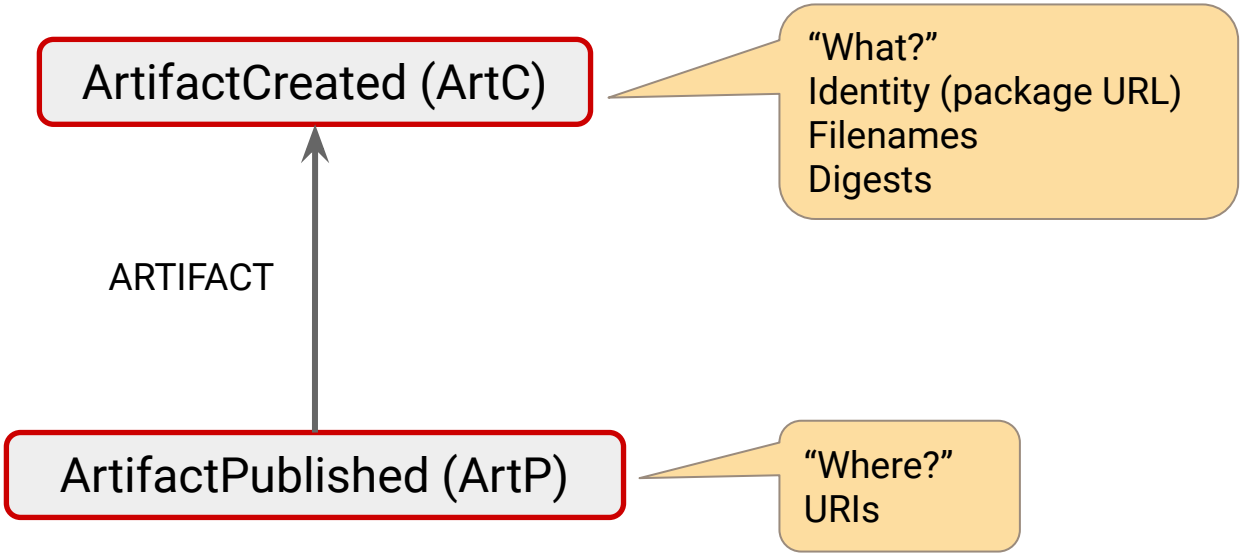
Artifacts

An Eiffel **artifact** is some kind of data that's produced. Often one or more files produced by a build system.

Artifacts are identified with a Package URL (https://github.com/package-url/purl-spec):

pkg:deb/debian/curl@7.50.3-1?arch=i386&distro=jessie
pkg:docker/cassandra@sha256:244fd47e07d1004f0aed9c
pkg:gem/ruby-advisory-db-check@0.12.4
pkg:maven/org.apache.xmlgraphics/batik-anim@1.9.1?packaging=sources
pkg:npm/foobar@12.3.1
pkg:pypi/django@1.11.1
pkg:generic/openssl@1.1.10g

Artifacts

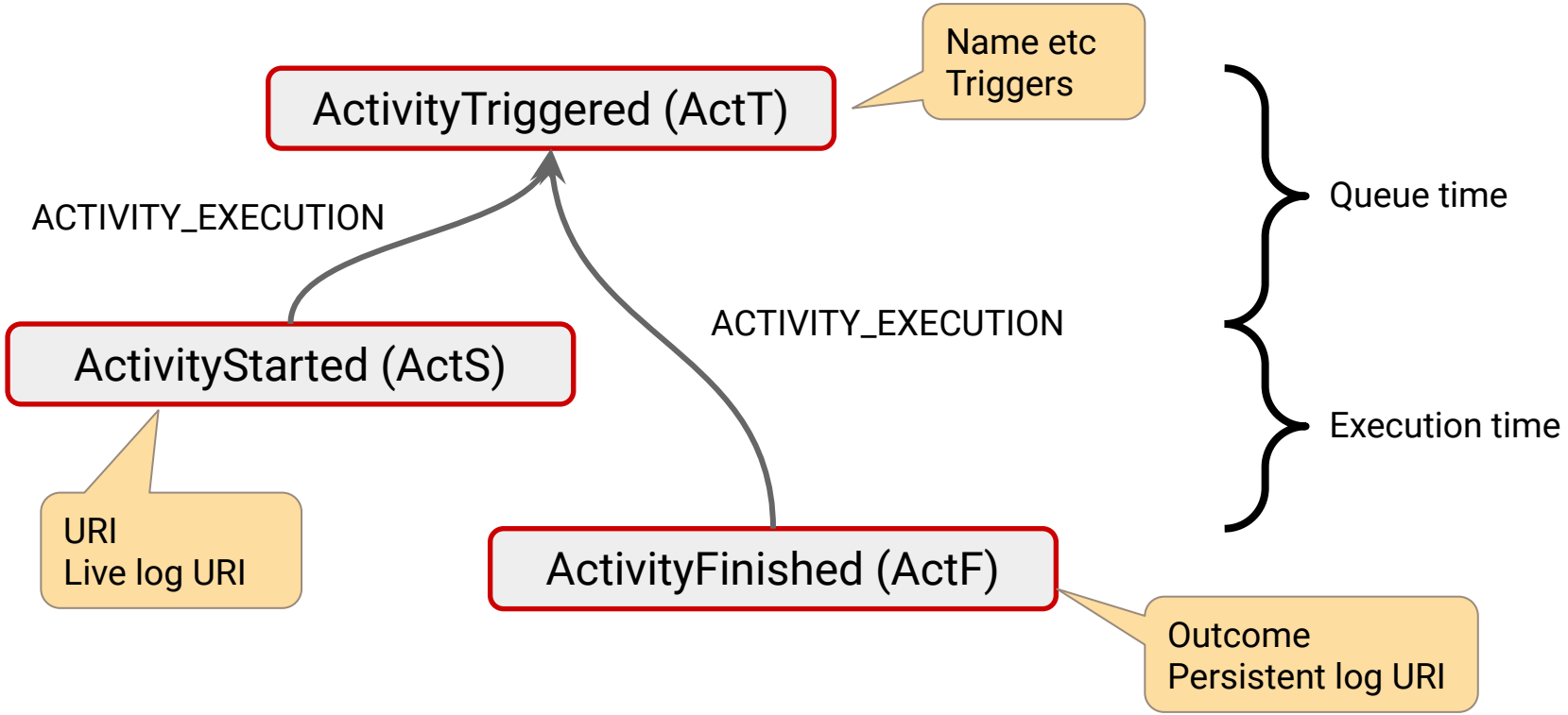

ArtifactCreated (ArtC)
"What?"
Identity (package URL)
Filenames
Digests
ARTIFACT
ArtifactPublished (ArtP)
"Where?"
URIs

Activities

An Eiffel **activity** describes something that’s taking place.

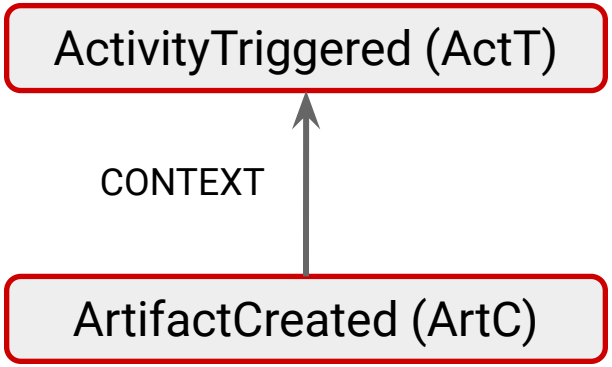
Activities typically don’t carry meaning themselves, but they act as useful containers for artifacts and test executions.

Activities



Eiffel

Activities



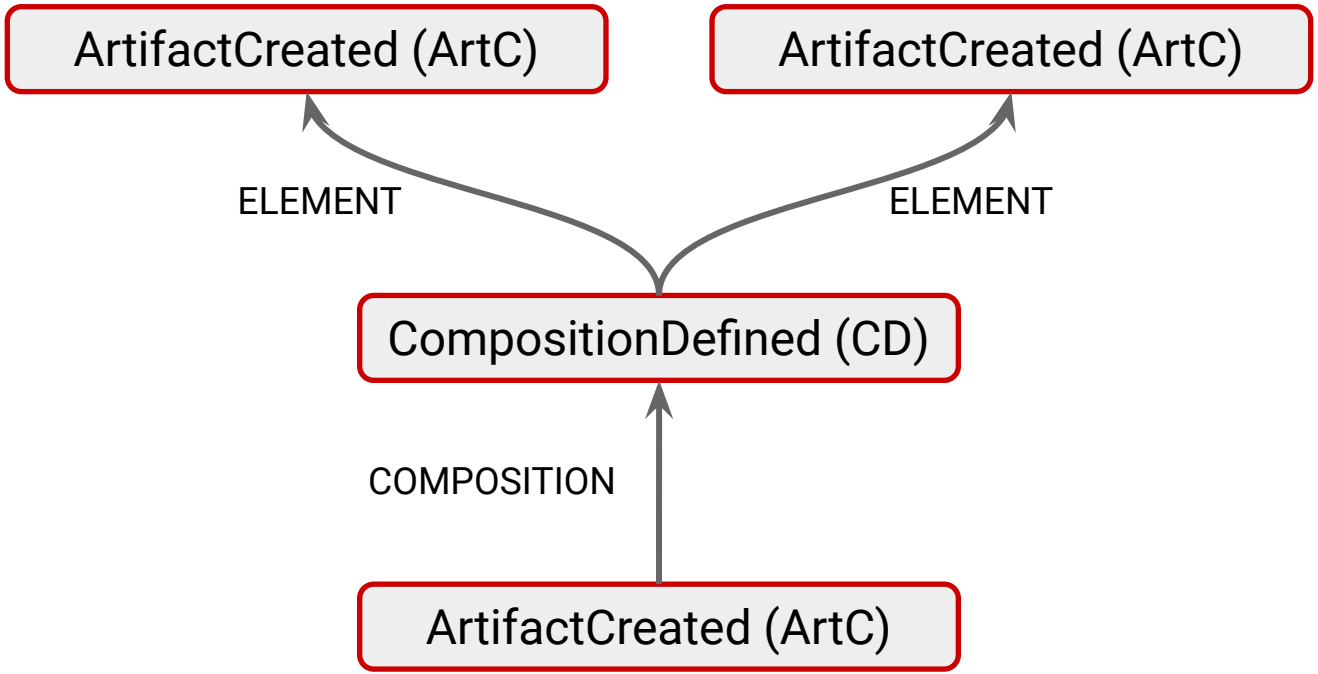
Compositions

An Eiffel **composition** groups artifacts, source changes, or other compositions.

A composition has a name and optionally a version.

Artifacts use compositions to indicate what they were created from.

Compositions

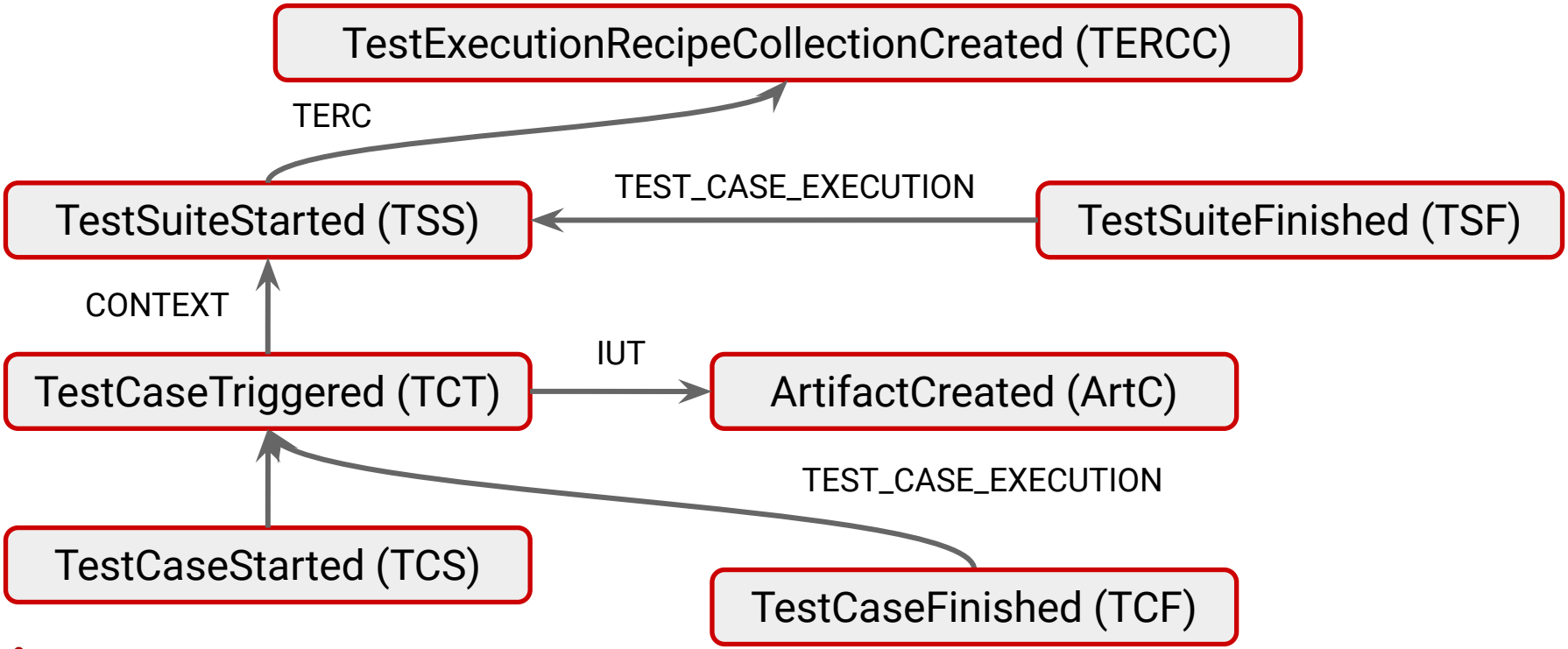


Tests

Eiffel defines various test-related events that describe what goes on during testing.

Like artifact creations tests are often connected to an Eiffel activity.

Tests



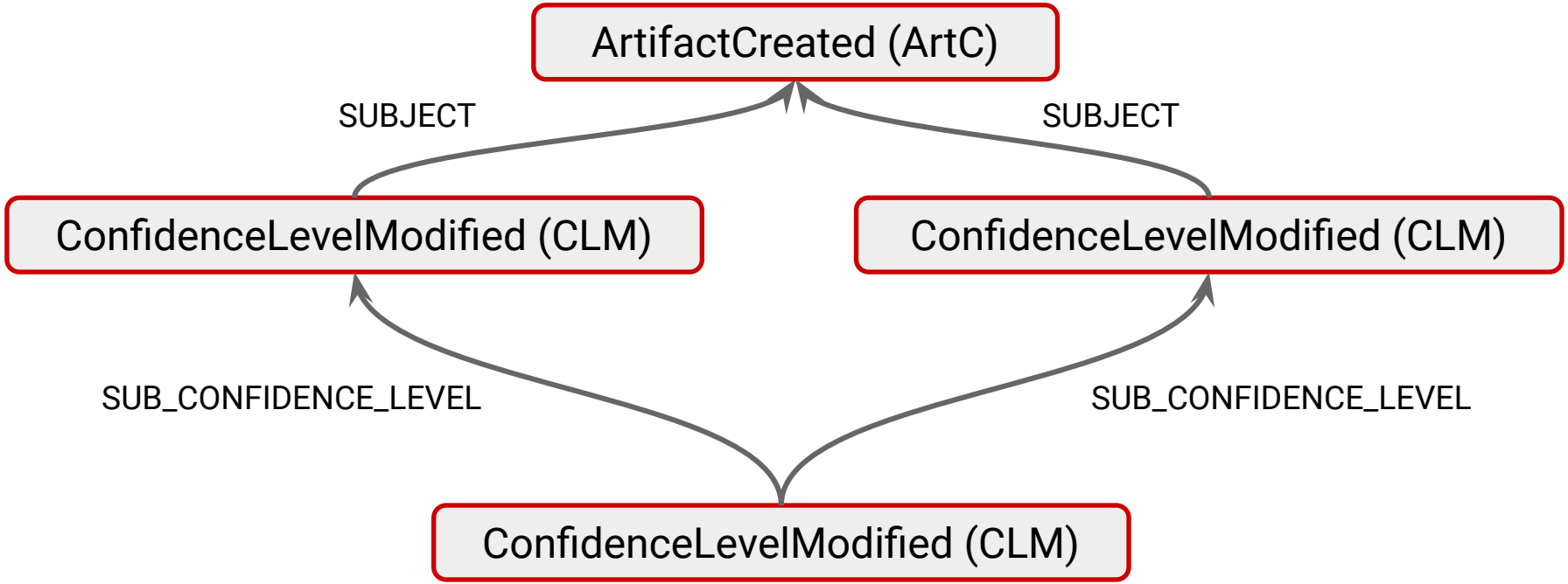
Eiffel

Confidence levels

An Eiffel **confidence level modification** declares that a set of Eiffel entities has been evaluated and found to achieve or not achieve a particular level of confidence for someone.

Example: “We’re confident that baseline X is releasable”.

Confidence levels

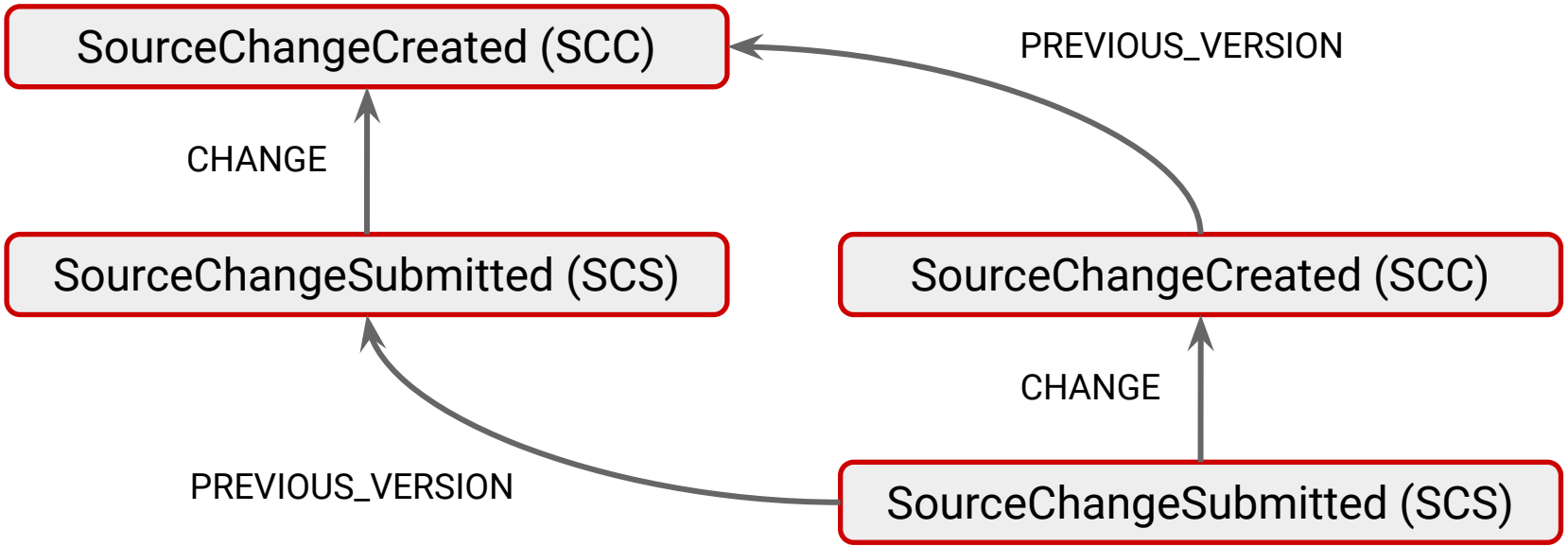


Source changes

Eiffel **source change** events describe what goes on in source code repositories.

Thanks to the links, source change events can represent the whole history of a repository.

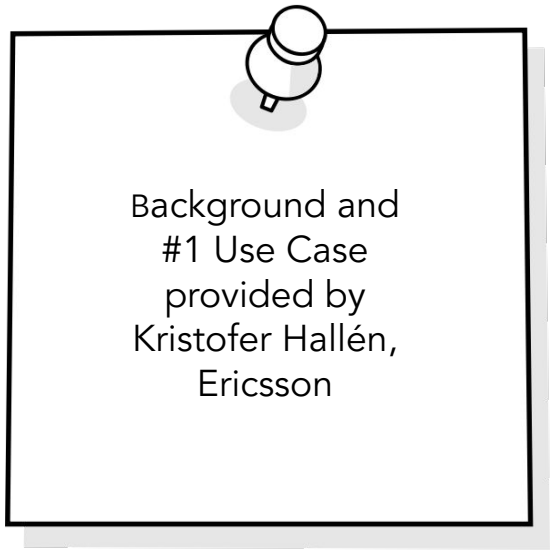
Source changes



Ericsson use cases

Ericsson Use Cases - Background

ERICSSON



Eiffel

Ericsson Use Cases - Background

ERICSSON

Ericsson Loves Events in CI/CD

- since 2012 -

"There must be something we did right since we are still getting value from concepts defined more than ten years ago"

CICD Architect
Ericsson

1,000,000s of Eiffel events per day

Connecting pipelines, triggering

Collecting product information

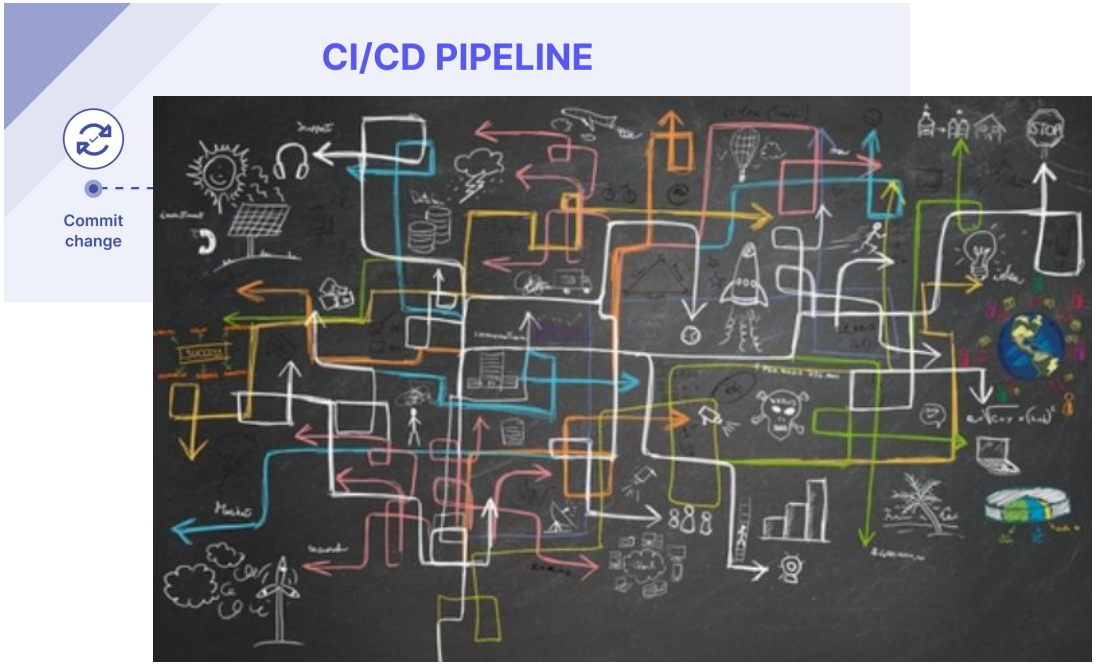
Providing data for visualization of products and pipelines

Eiffel

Ericsson Use Cases - Background

ERICSSON

Our world …



… is not like in the books

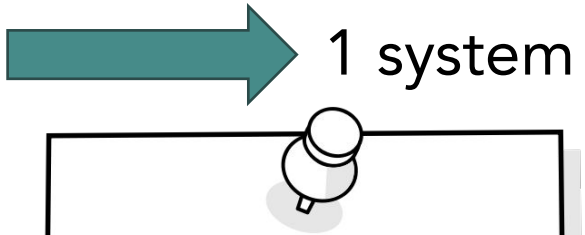
Eiffel

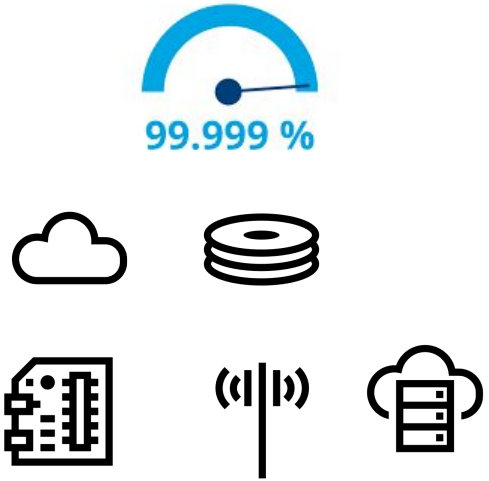
Ericsson Use Cases - Background

ERICSSON

Our world …

- 1,000s of software components
- 1,000s of commits per day
- 100,000s of test runs per day
- 1,000s of developers, on three continents
- From embedded to cloud platforms
- Radio signals
- ~40 development and CI/CD tool stacks.



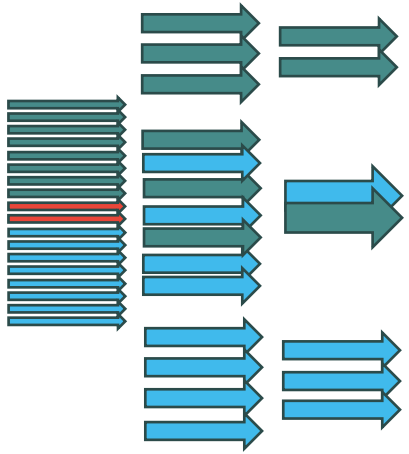


Eiffel

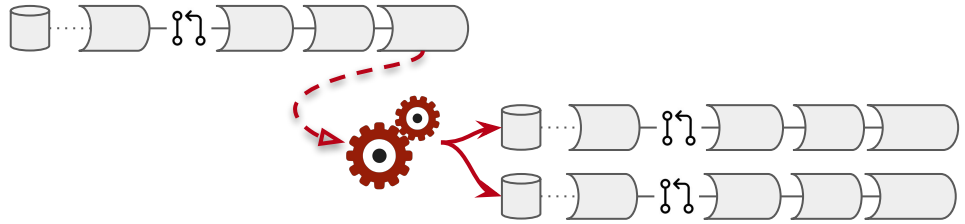
Ericsson Use Cases - Intro

ERICSSON

1. SW Pipeline Assembly
   - Assemble Huge SW Pipelines in Days



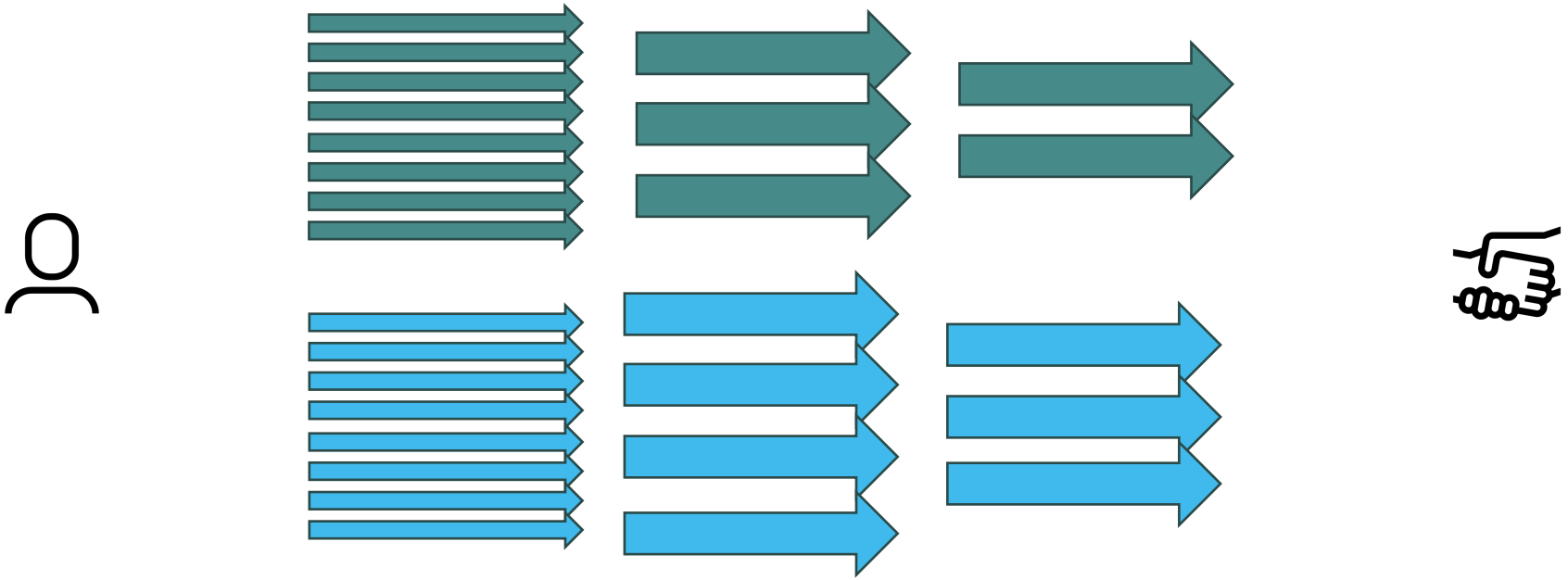
2. Dependency Updates
   - General purpose ‘Dependabot’



Eiffel

Ericsson Use Cases - SW Pipeline Assembly
The Starting Point

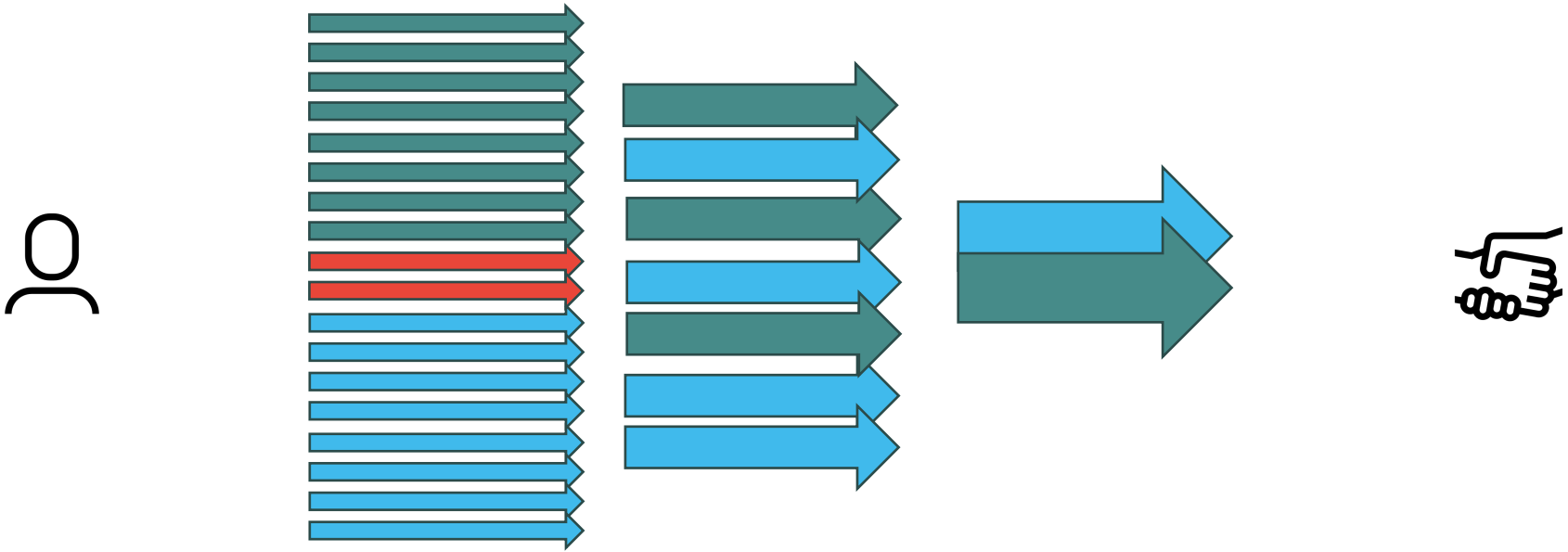
ERICSSON



Eiffel

Ericsson Use Cases - SW Pipeline Assembly
The Challenge

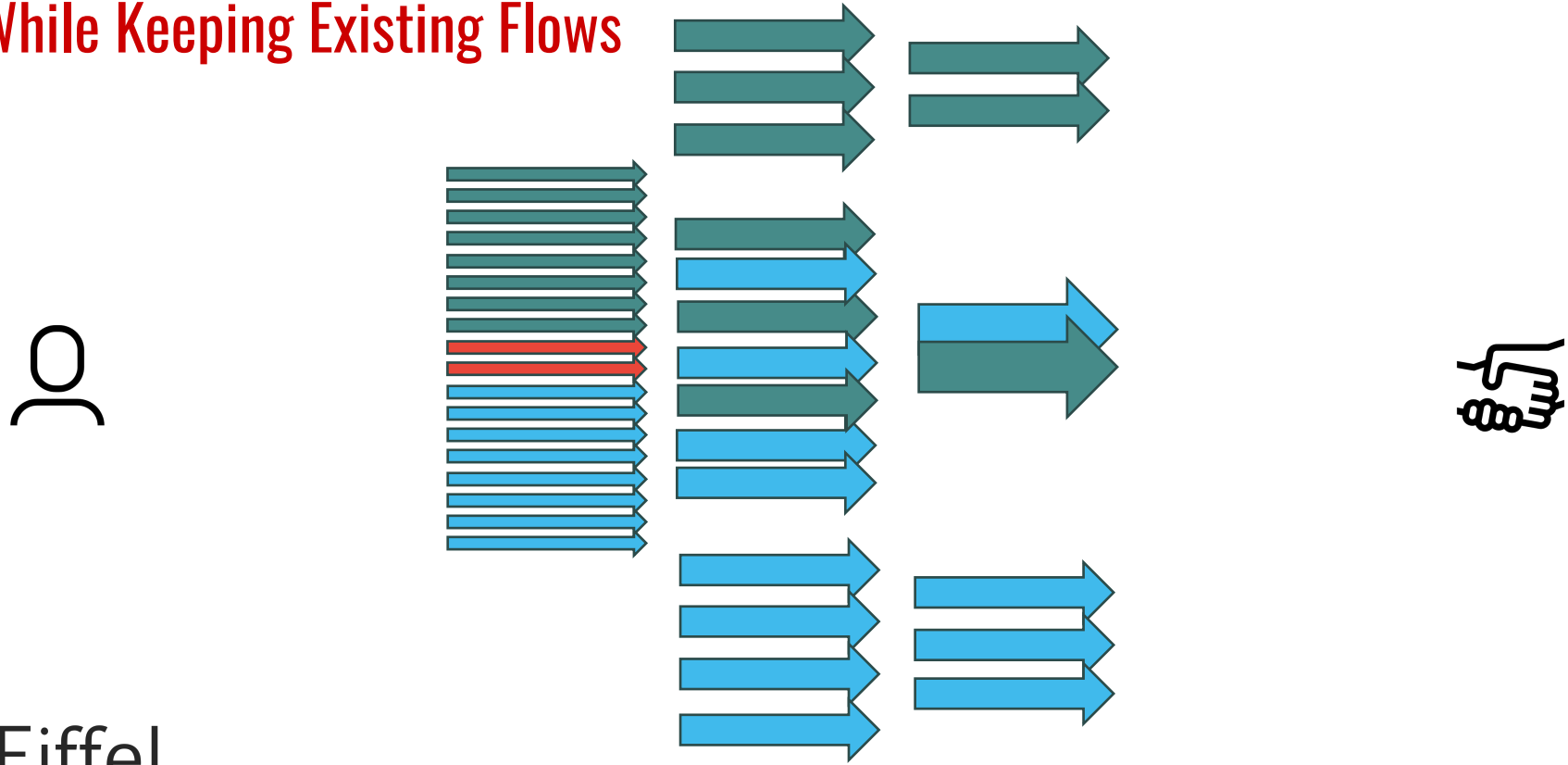
ERICSSON



Eiffel

Ericsson Use Cases - SW Pipeline Assembly While Keeping Existing Flows

ERICSSON



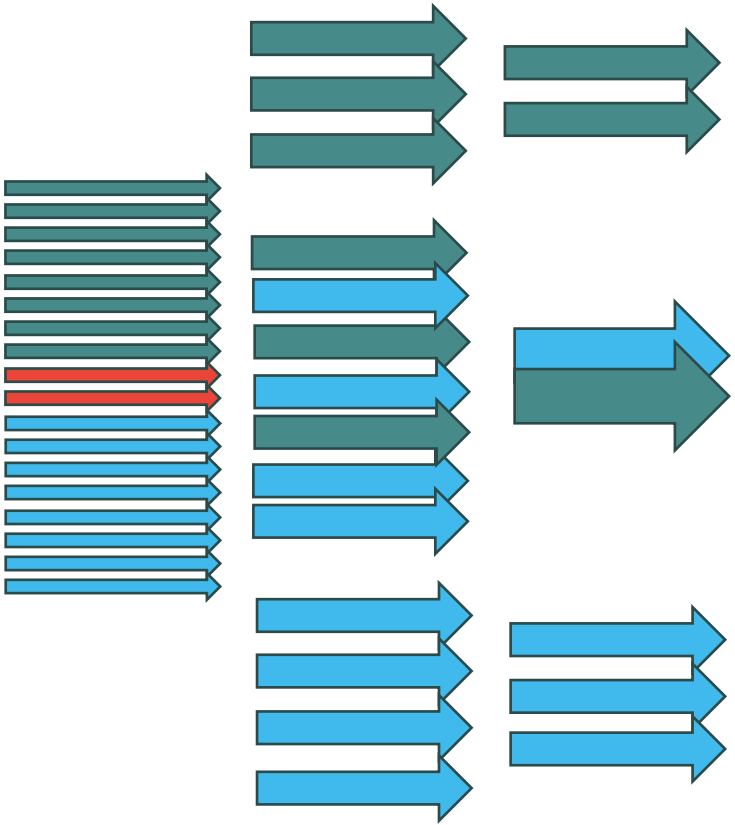
Eiffel

Ericsson Use Cases - SW Pipeline Assembly Thanks to the Events!

ERICSSON

- Decoupled but Connected pipelines
- Visualized pipeline progress
- Collecting product information
- Collecting quality information

- Fast
- Cheap
- Minimal impact on existing pipelines



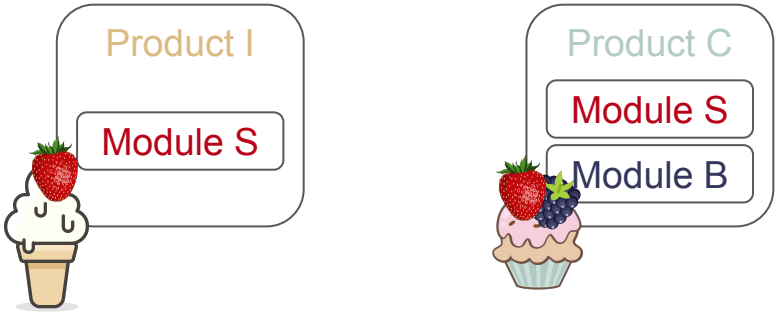
Eiffel

Ericsson Use Cases - SW Pipeline Assembly Reflections

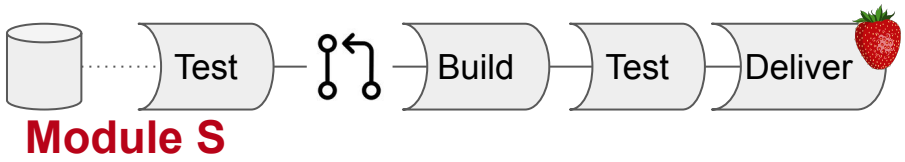
ERICSSON

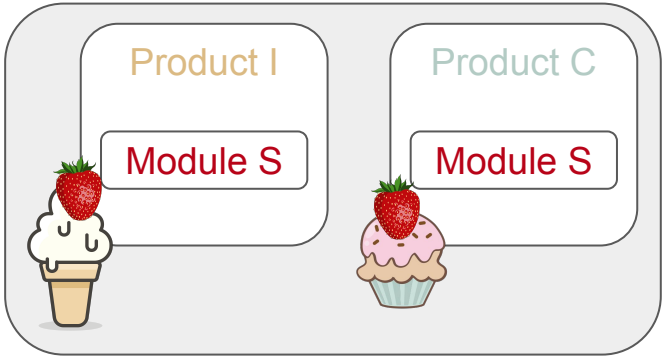
Eiffel

Ericsson Use Cases - Dependency Updates

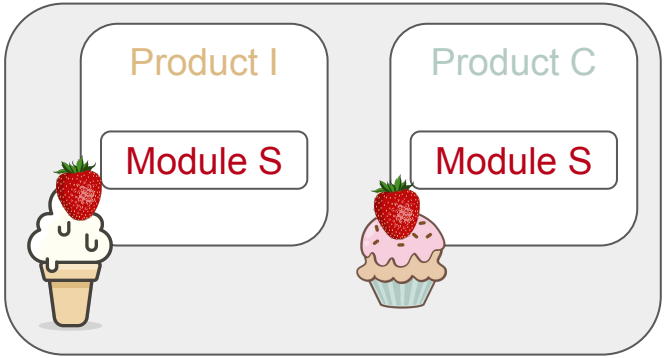


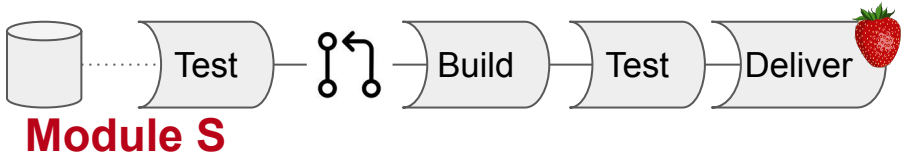
Ericsson Use Cases - Dependency Updates

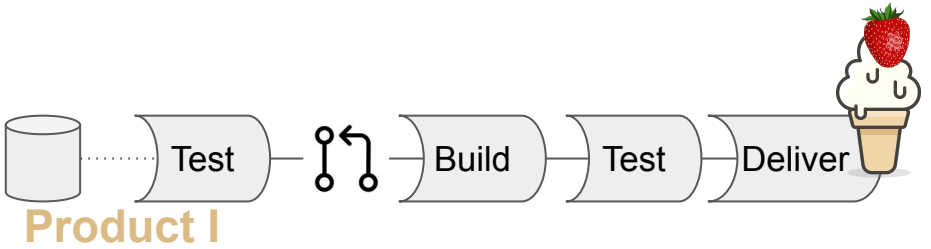


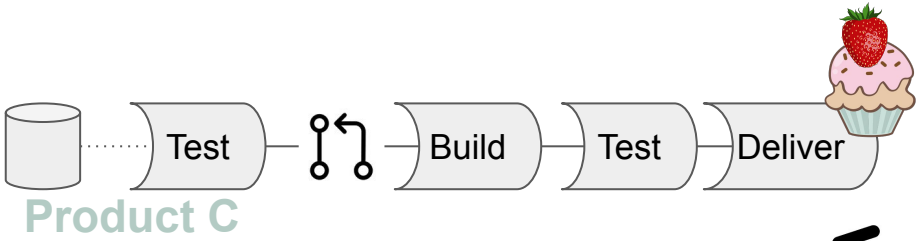


Ericsson Use Cases - Dependency Updates





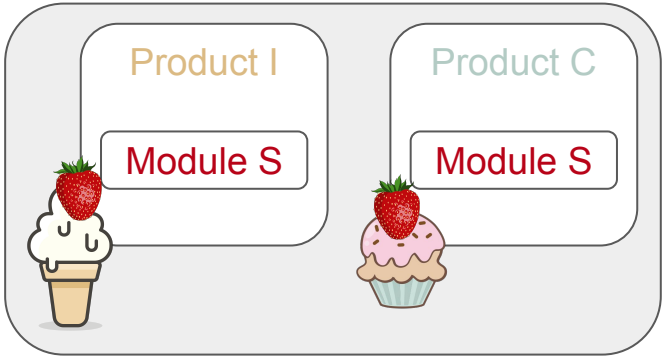


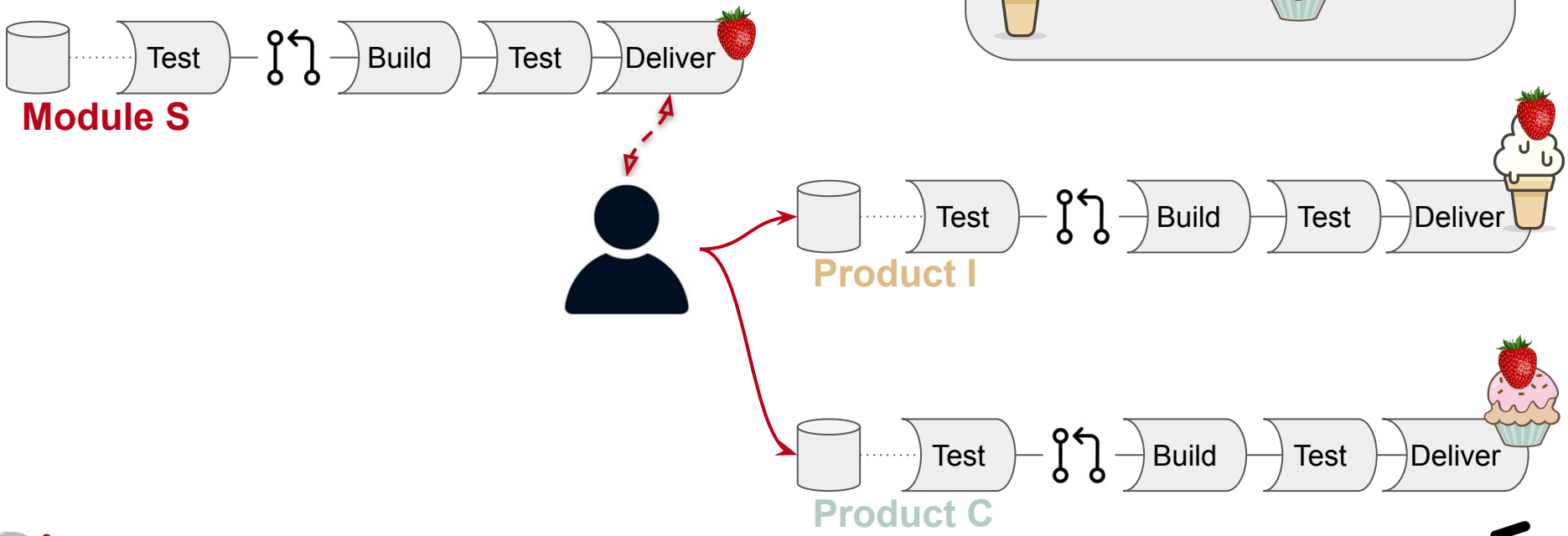


Eiffel

ERICSSON

Ericsson Use Cases - Dependency Updates

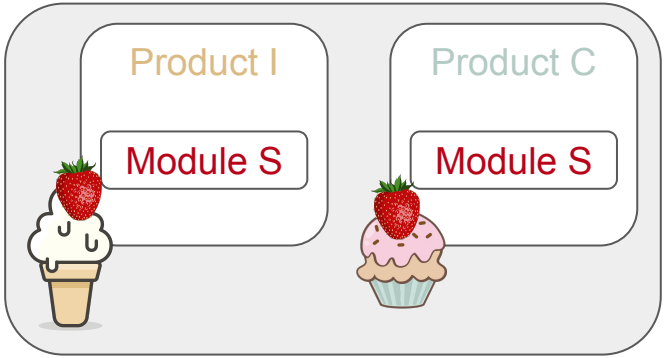


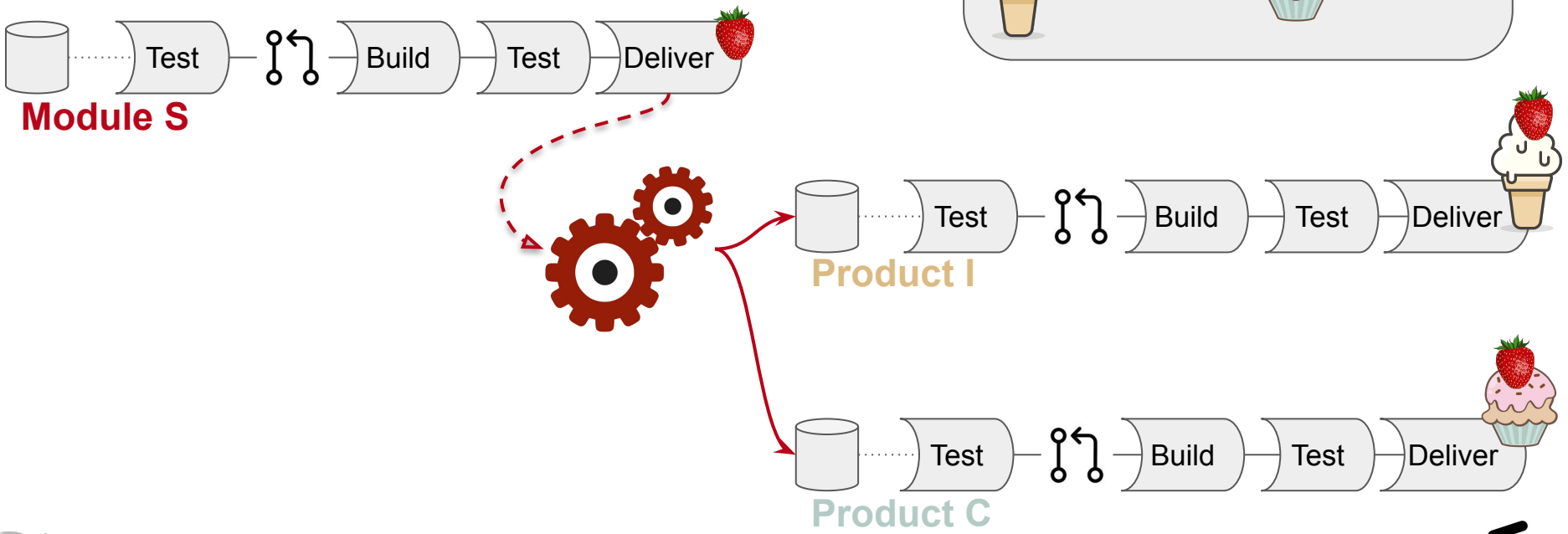


Eiffel

ERICSSON

Ericsson Use Cases - Dependency Updates

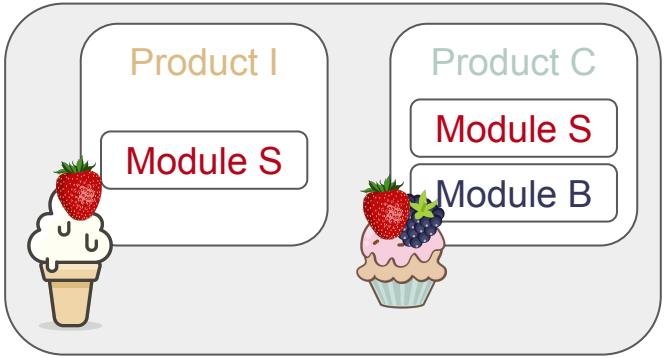


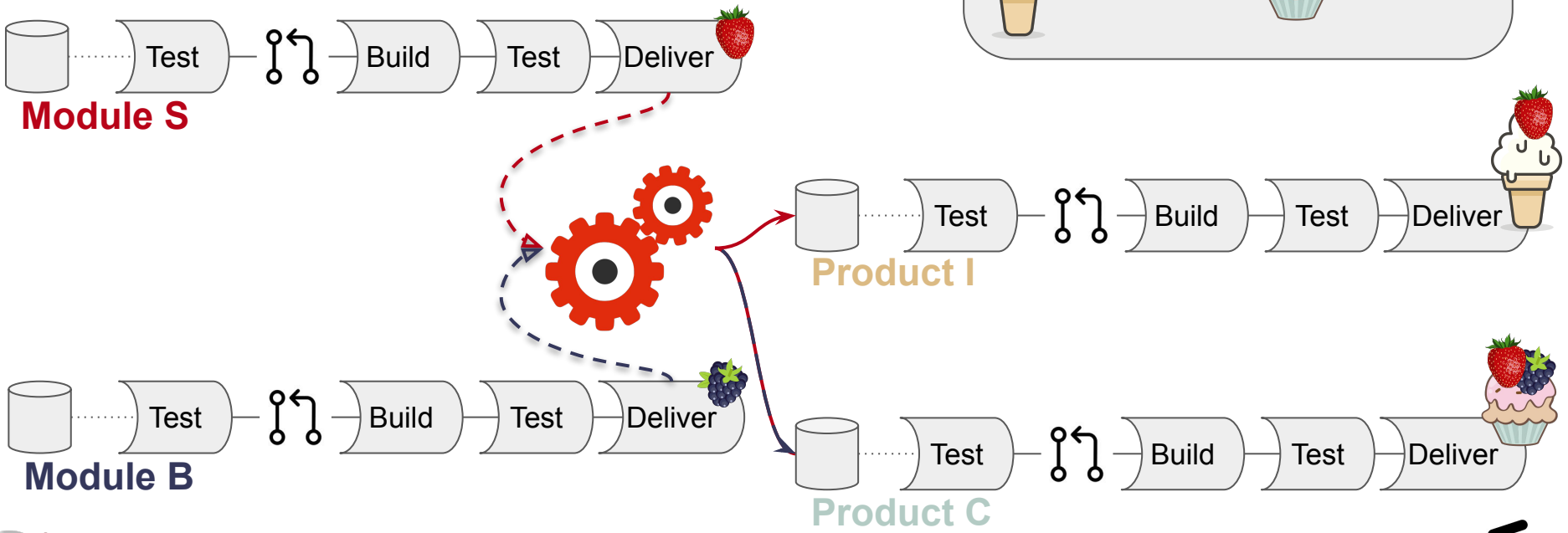


Eiffel

ERICSSON

Ericsson Use Cases - Dependency Updates



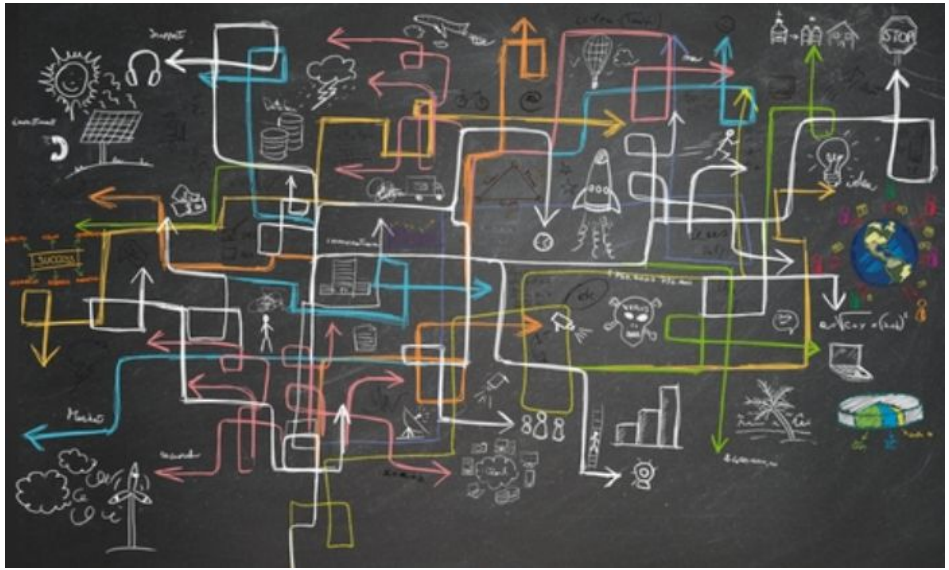


Eiffel

ERICSSON

Ericsson Use Cases - Dependency Updates

What if the dependency tree looks like this?

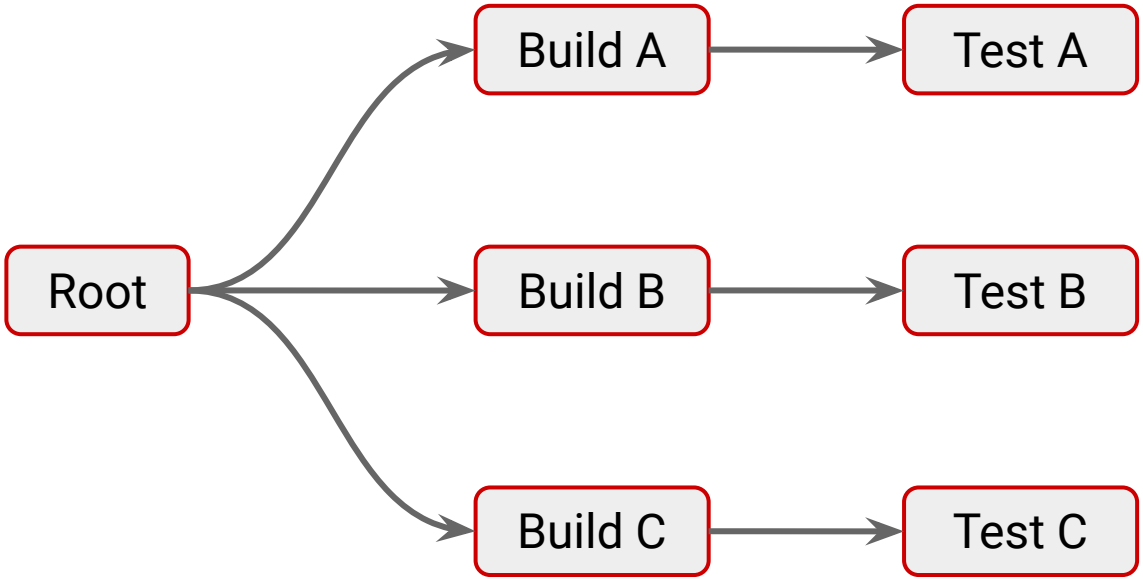


Eiffel

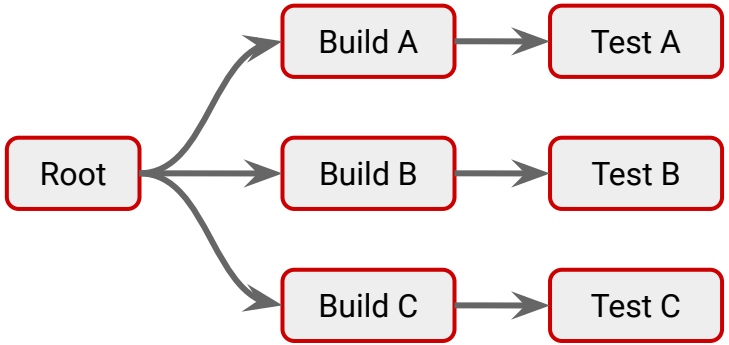
ERICSSON

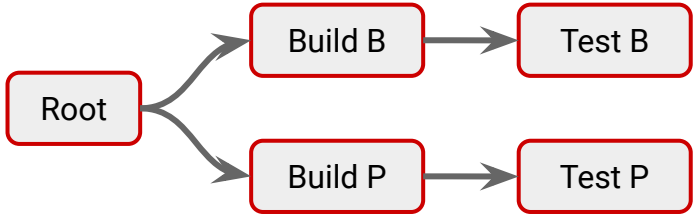
Axis use cases

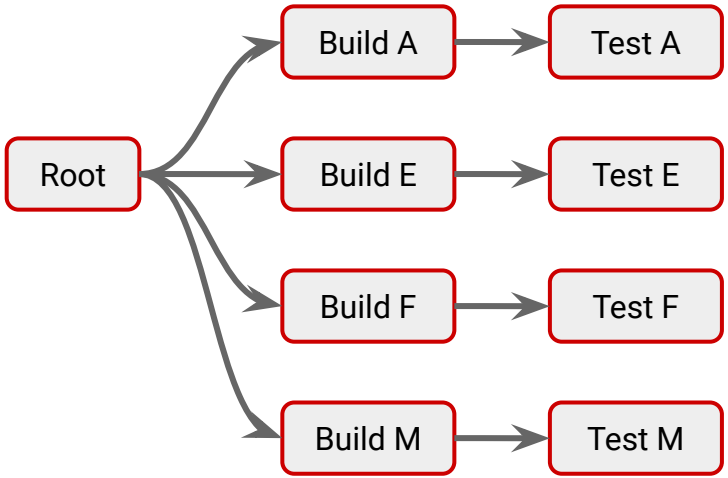
A simple pipeline for testing products running AXIS OS



Actual pipelines for testing products running AXIS OS

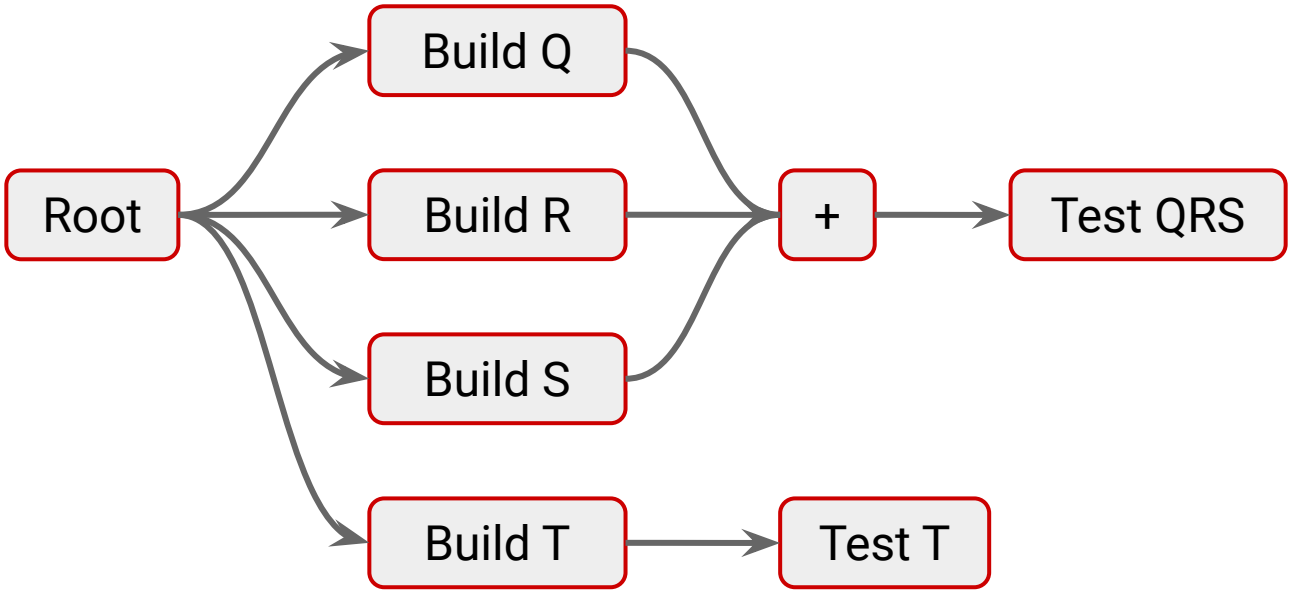






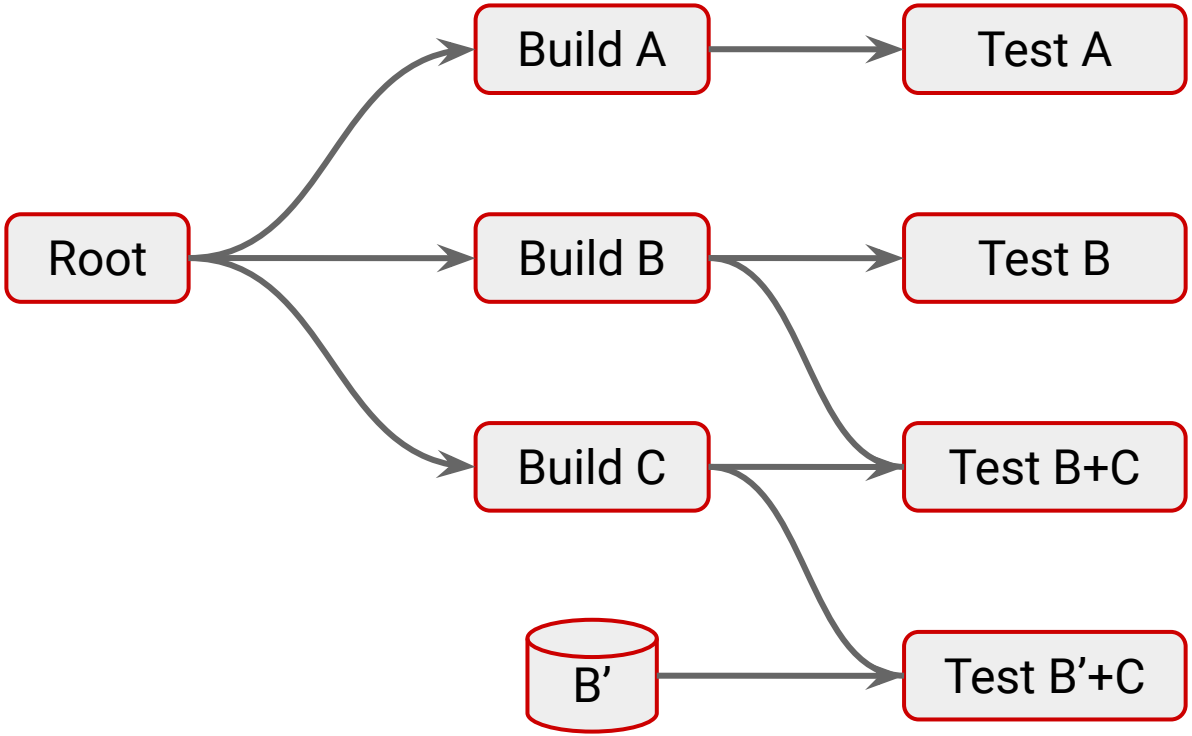
Eiffel

Fan-ins



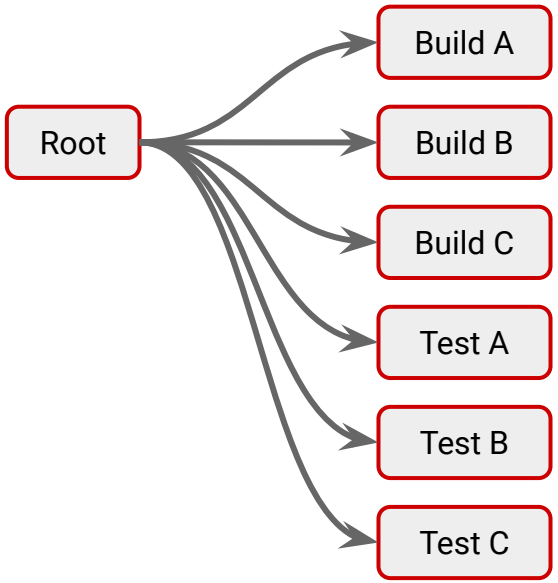
Eiffel

Testing systems of products



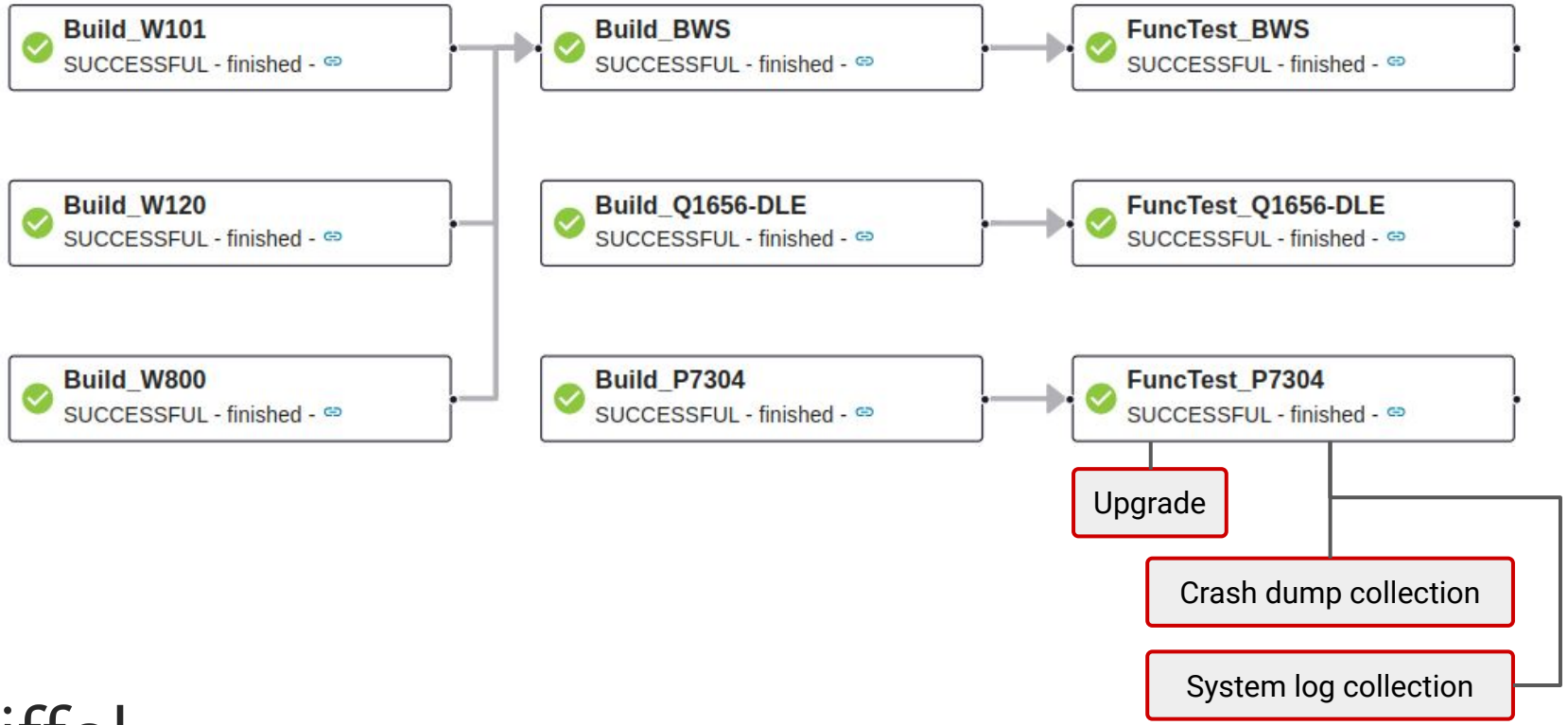
Eiffel

Solution: Implement the scheduling in a Jenkins pipeline



Eiffel

Solution: Eiffel-driven visualizer



Eiffel

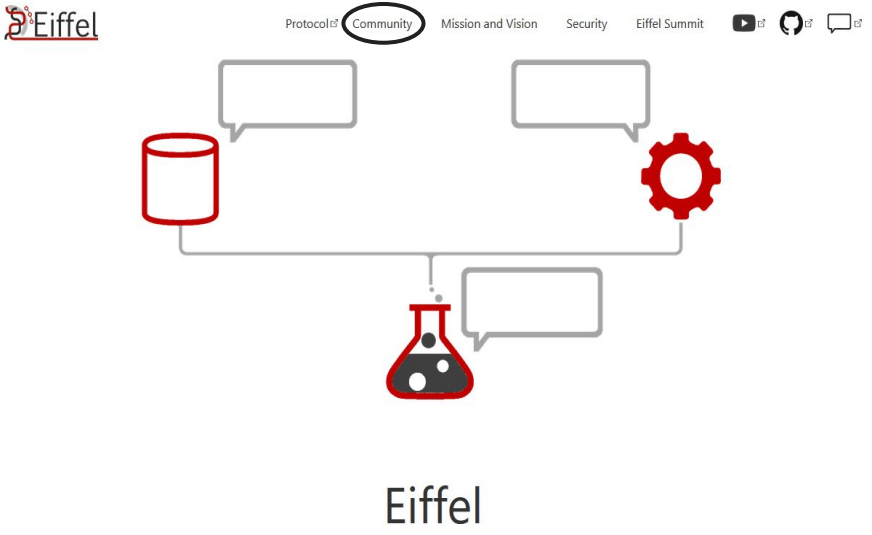
The Eiffel landscape

An overview of the projects in our landscape

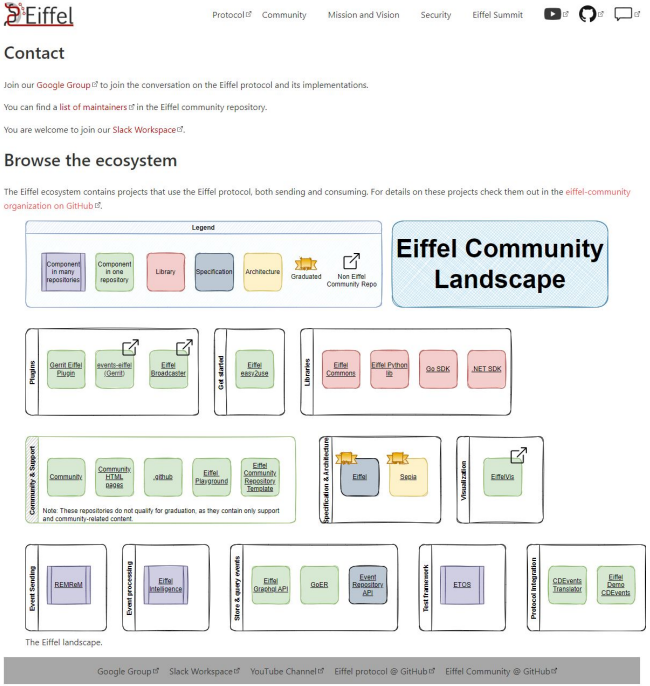
Mattias Linnér - Ericsson

How to find the landscape

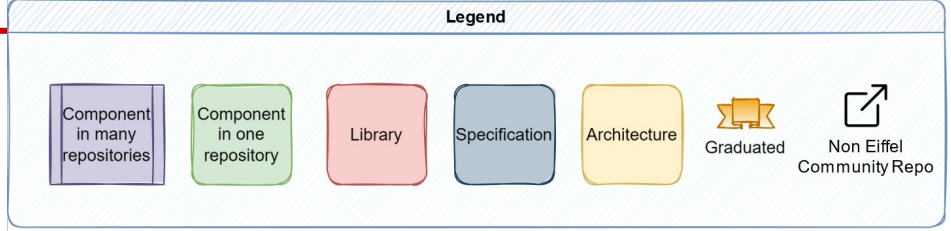
https://eiffel-community.github.io/

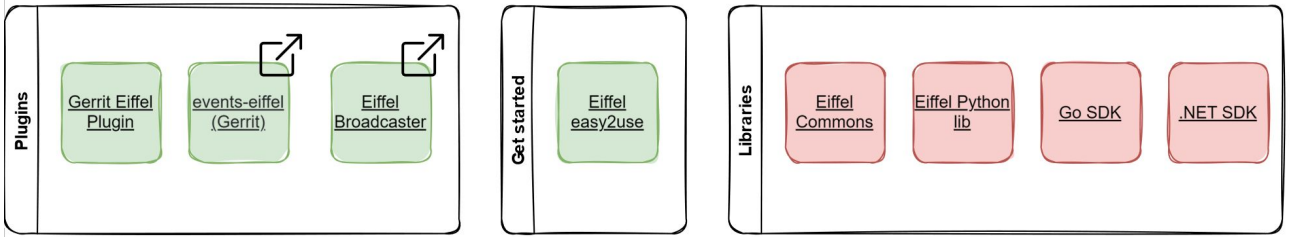


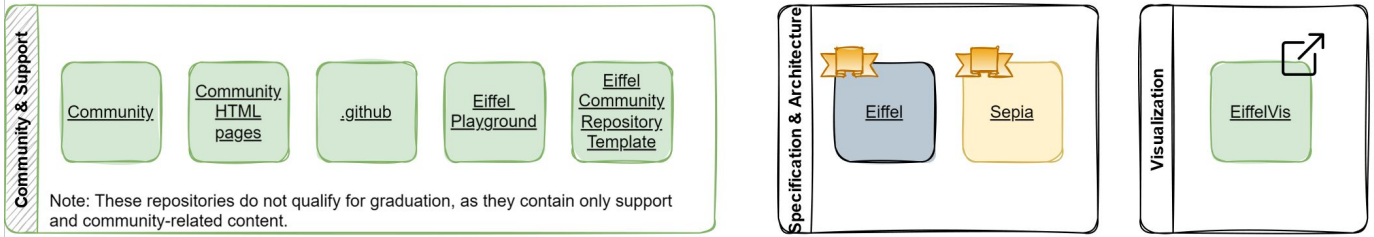
https://eiffel-community.github.io/community.html

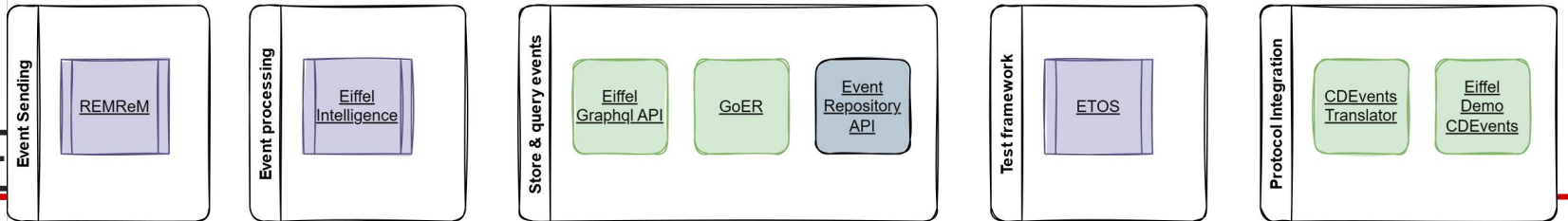


Eiffel Community Landscape









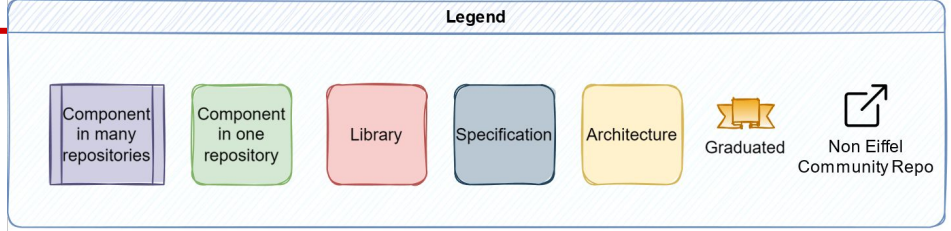
Lifecycle

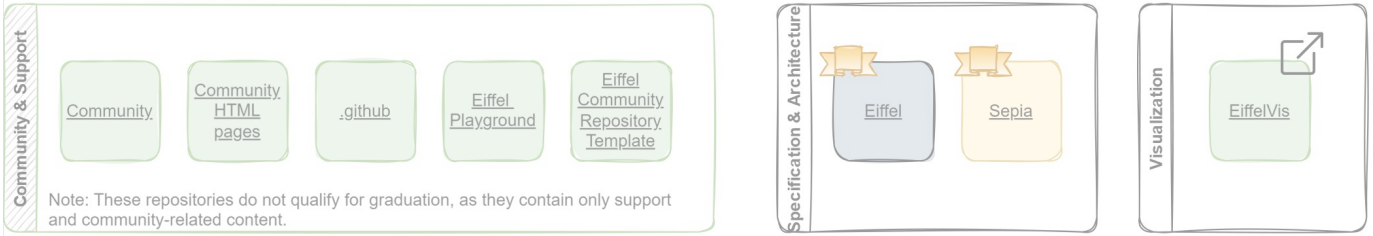
- Sandbox stage is for the projects that are at the early stages of their development lifecycle such as newly created projects or the projects with heavy initial development going on.

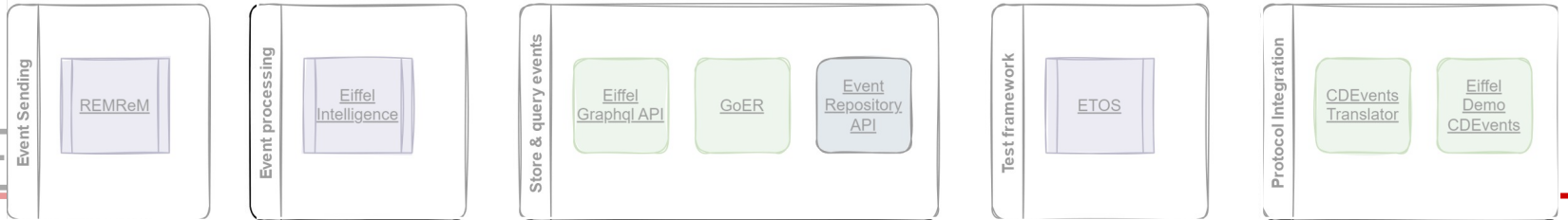
- Graduated stage is for the projects that achieved production level functionality and quality. Eiffel Community expects projects in Graduated stage to be active, both within the community and towards their users.

- Archived stage is for the projects that do not expect further development or maintenance and may very well be in dormant state.

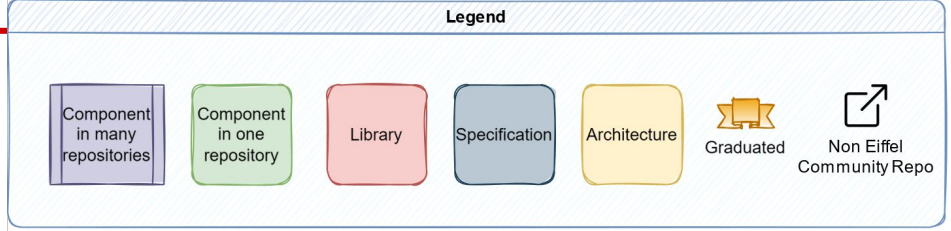
Eiffel Community Landscape

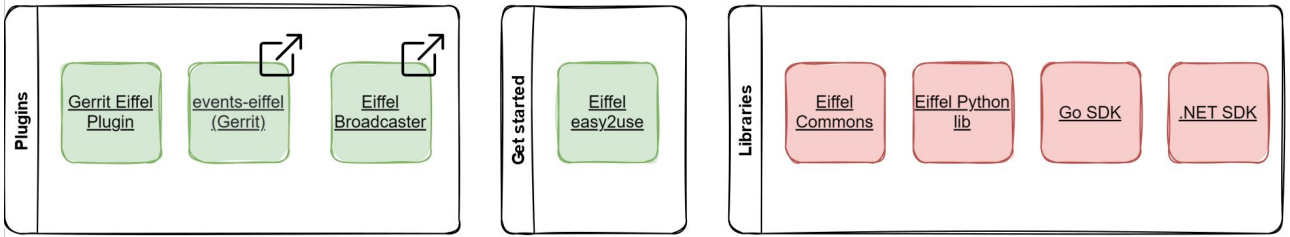


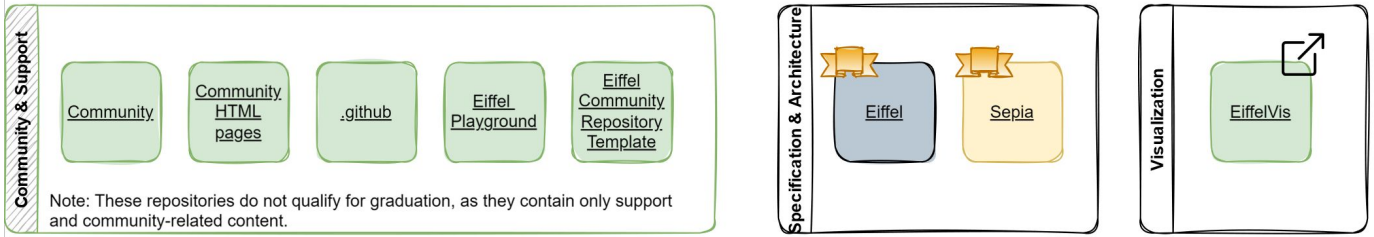


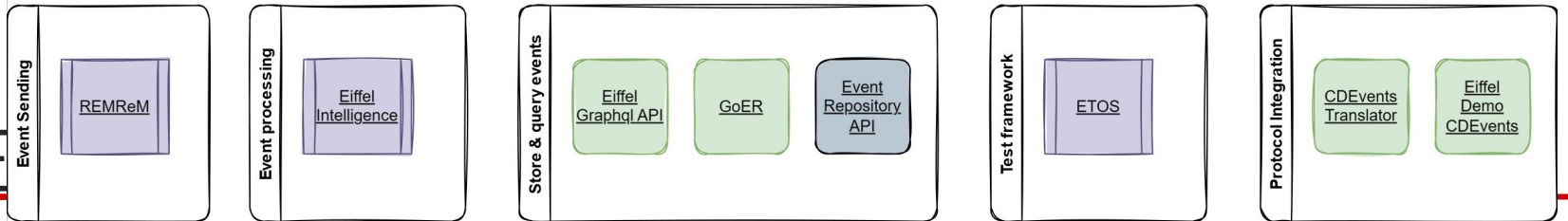


Eiffel Community Landscape









SDKs

Project	Language	Multi-Edition support	meta.security support	Comment
eiffel-remrem-semantics	Java	No	No	Not a full SDK
eiffel-pythonlib	Python	No	No	Untyped
eiffelevents-sdk-go	Go	Yes	Yes	
eiffelevents-sdk-dotnet	.NET	Yes	Yes	

Questions?