Trunk Based Development

Tony Bjerstedt

What are you going to learn?

The problem with traditional branching strategies

What is Trunk Based Development

How does this help

How does it work

Some tools and techniques

About Me

Family Guy

Photographer

Architect & Developer

Leader

LinkedIn:
https://www.linkedin.com/in/tbjerstedt

Email:
tony@bjerstedttechnologies.com

Twitter
@tbjerstedt

LinkedIn Code

ClickSWITCH

Kawishiwi Falls, Ely, MN

Branches

Why Branch

To separate development from support

Maintain releases separately

Release no code before its time

“**Continuous Integration** is a software development practice where members of a team integrate their work frequently, usually each person integrates at least daily - leading to multiple integrations per day. Each integration is verified by an automated build (including test) to detect integration errors as quickly as possible. Many teams find that this approach leads to significantly reduced integration problems and allows a team to develop cohesive software more rapidly.”

– **Martin Fowler (Blog article, 2006)**

https://www.martinfowler.com/articles/continuousIntegration.html

Branch Chaos

Long lived branches

Never ending sprints

Code Freezes

Merge Day Hell

Approaches to Branching

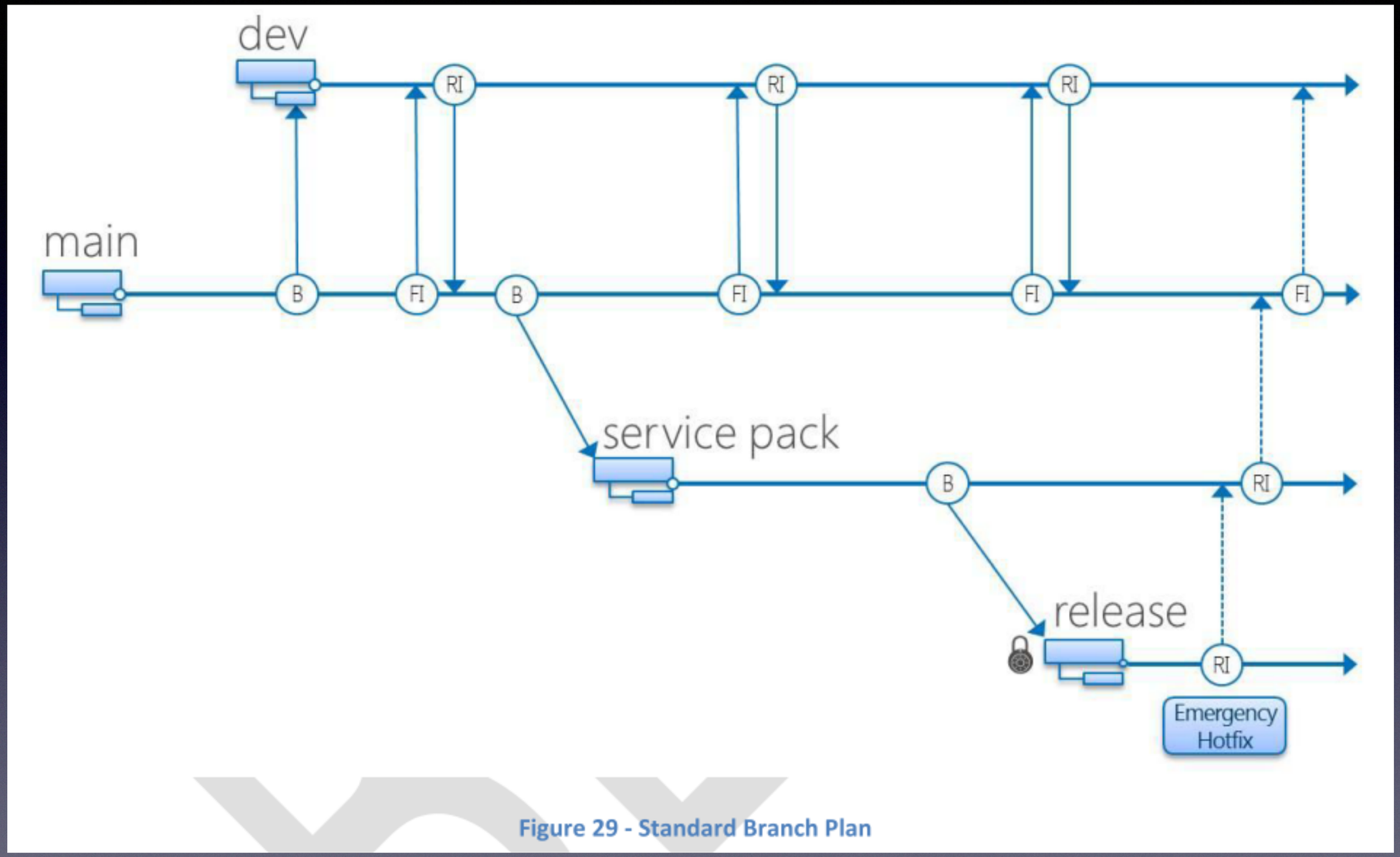


Figure 29 - Standard Branch Plan

Microsoft Recommended: Standard Branch Plan

https://www.infoq.com/news/2012/04/Branching-Guide/

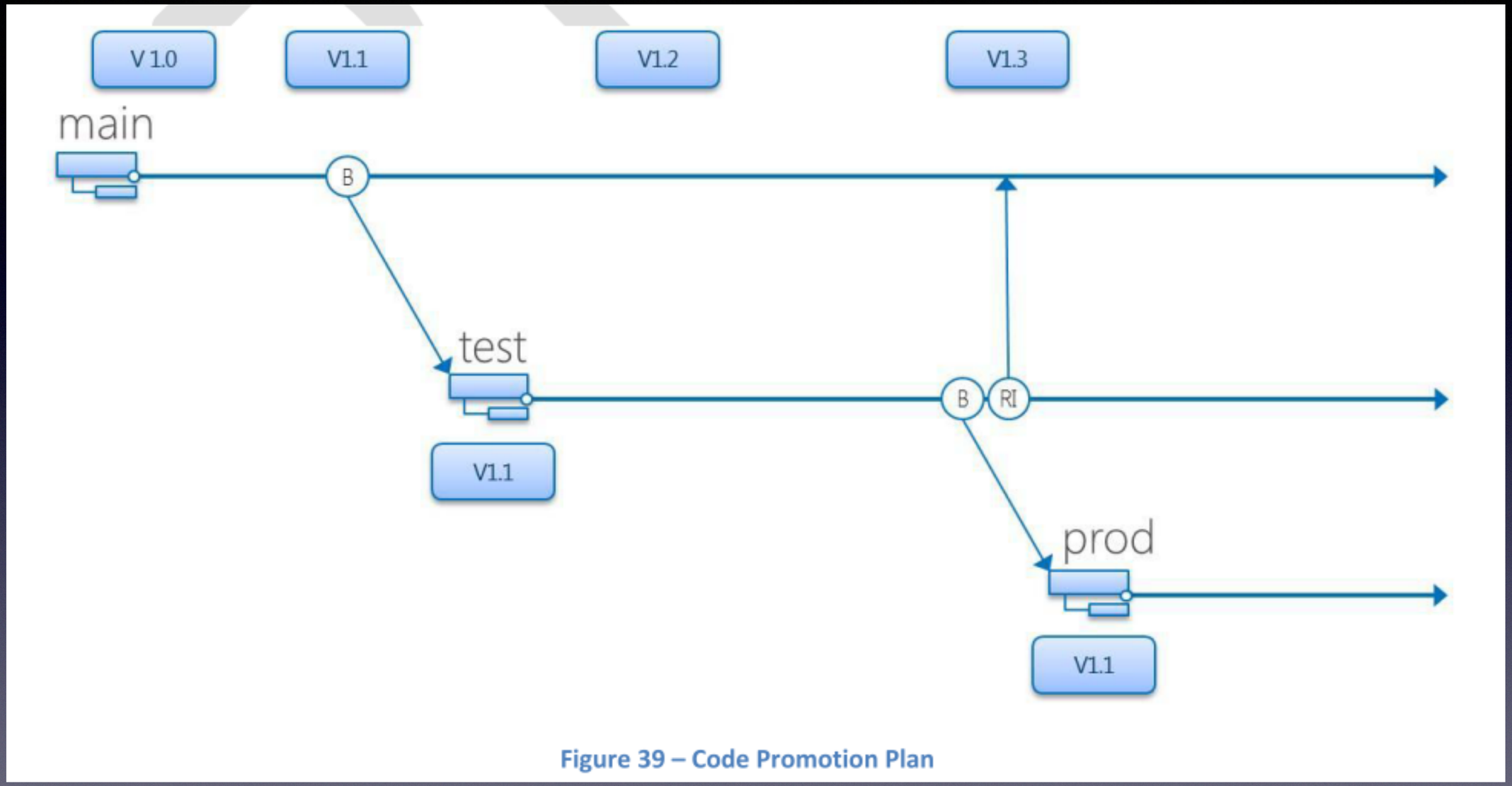
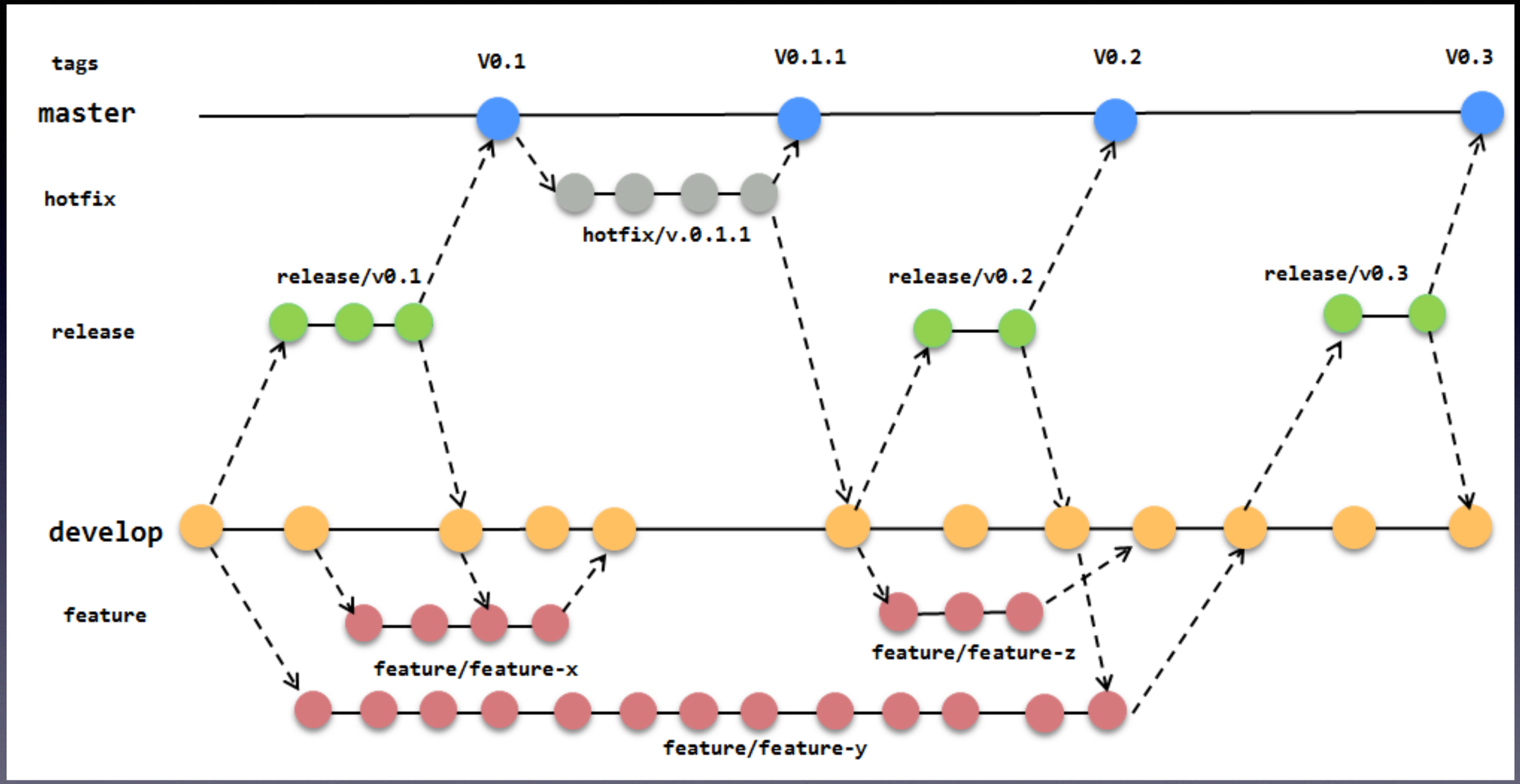


Figure 39 – Code Promotion Plan

Microsoft Recommended Code Promotion Plan

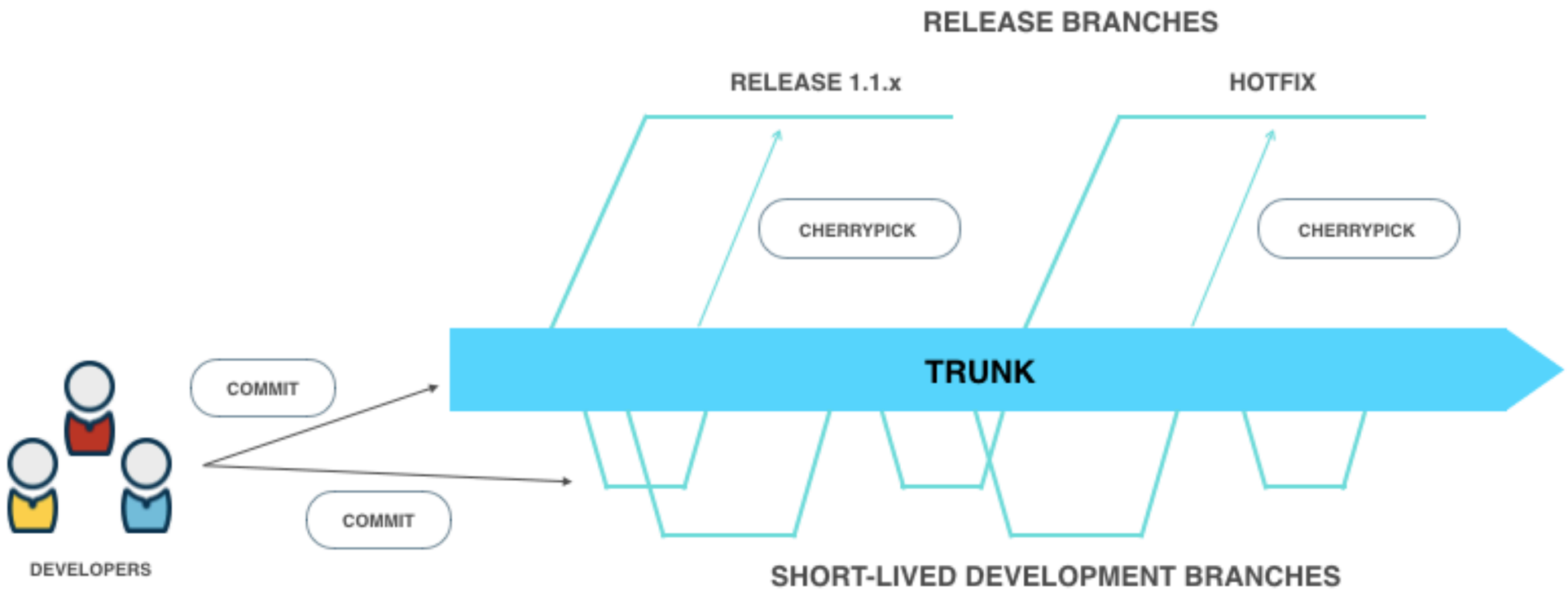
https://www.infoq.com/news/2012/04/Branching-Guide/



Gitflow

https://fpy.cz/pub/slides/git-workshop/#/step-1

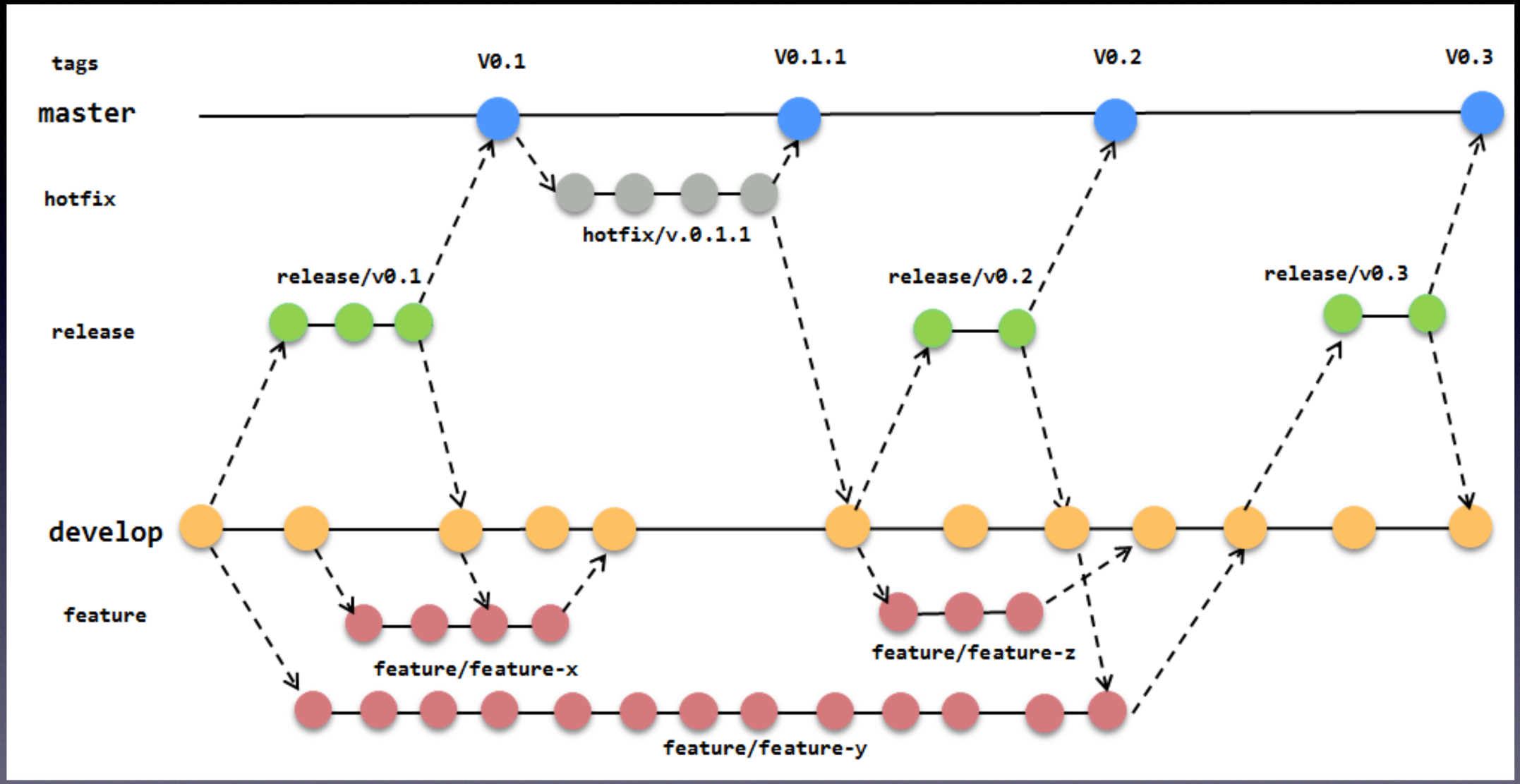
CI - TRUNK BASED DEVELOPMENT



Trunk Based Development

https://www.gocd.org/2018/05/30/ci-microservices-feature-toggles-trunk-based-development/

More about Git Flow



Gitflow

https://fpy.cz/pub/slides/git-workshop/#/step-1

Git Flow Pros & Cons

Pros:

Git Flow

Open-source projects

Junior developers

Pull requests / code review

Cons:

Can slow new projects

Hard to iterate quickly

Invites merge conflicts

Feature separation - testing 2+ new features

Unpredictable release times

Trunk-Based Development

Trunk Based Development

One branch serves as trunk (normally “master”)

Release branches when a new release is desired

Many short, lived development branches with frequent commits

Do not break the trunk. Commits are reviewed and must pass automated tests.

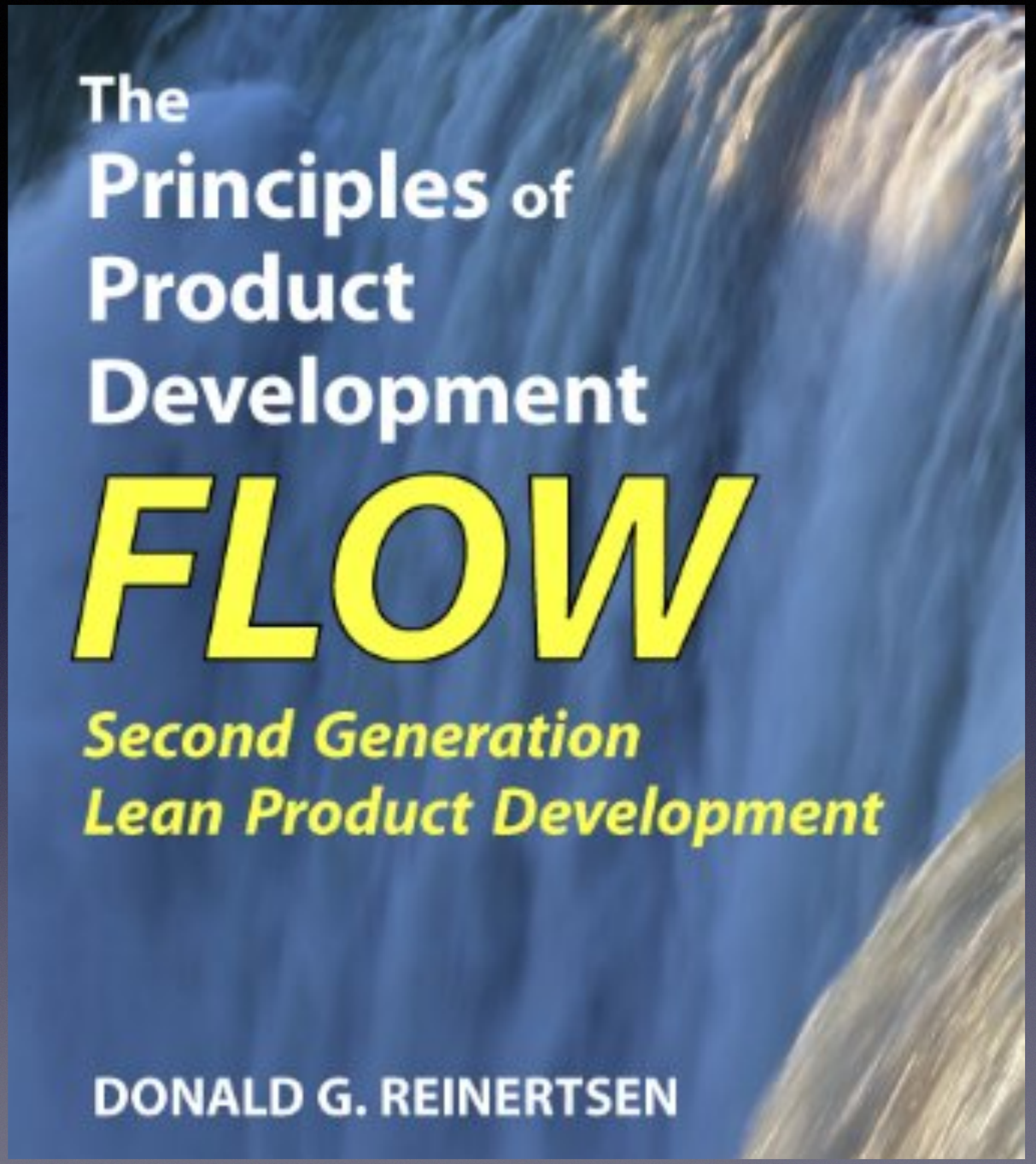
Why?

Achieving Flow by

Small batch size

Rapid Feedback

Limited WIP



“Branches create distance between developers and we do not want that”

–Frank Compagner, Guerrilla Games

https://trunkbaseddevelopment.com/5-min-overview/

Building Software

Development

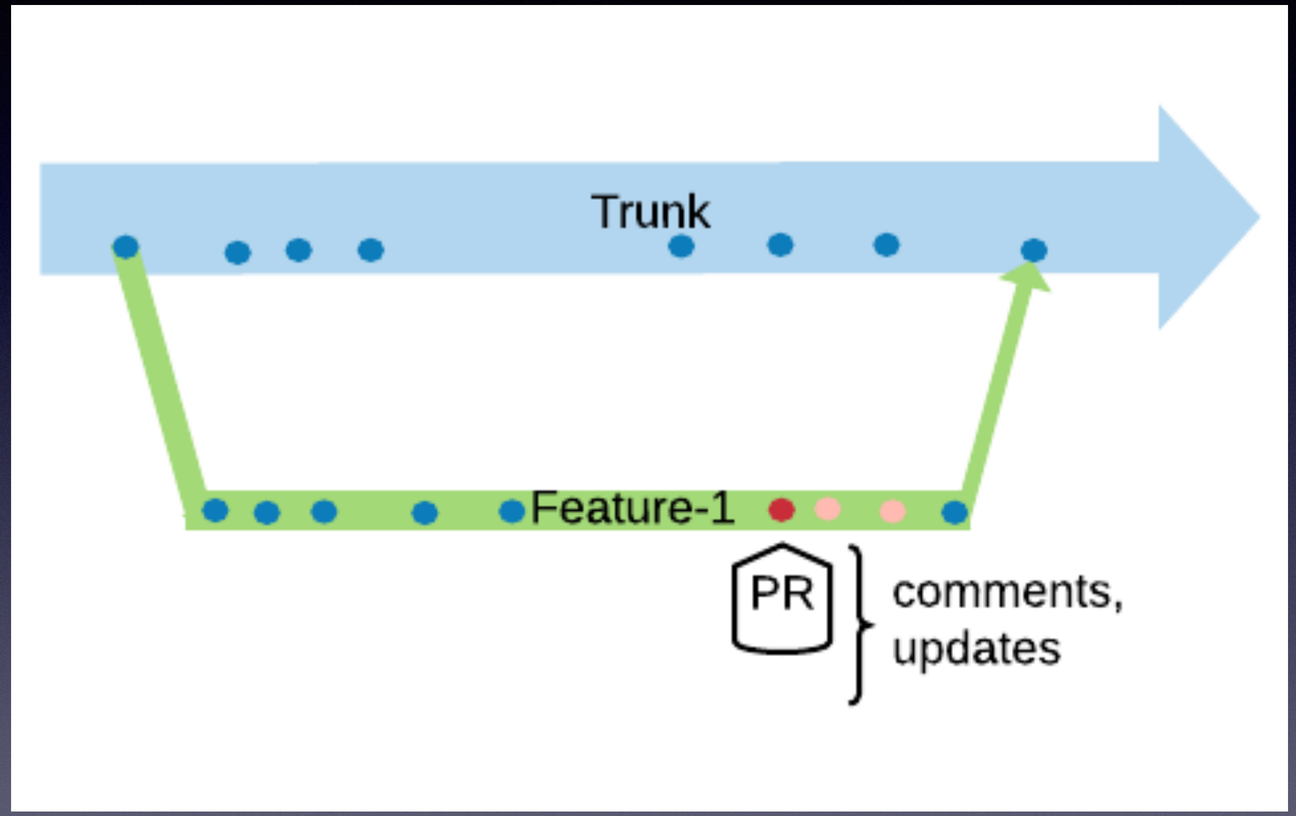
Short-lived (< 1 day)

Delete after merge

Never break build/tests

Always be **release ready**

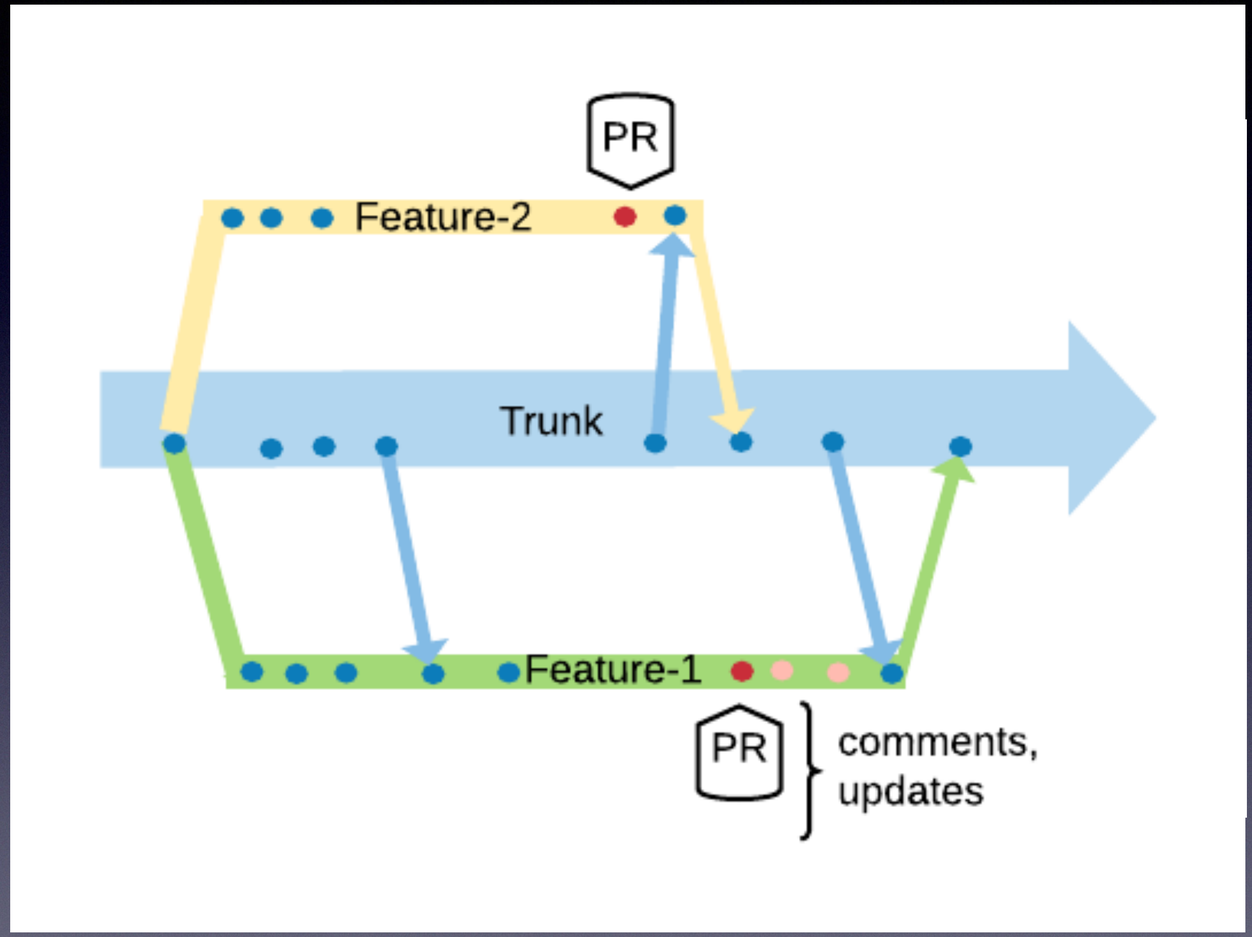
Automated builds/tests



Merge from Master often

Fewer Conflicts

Easier to Resolve

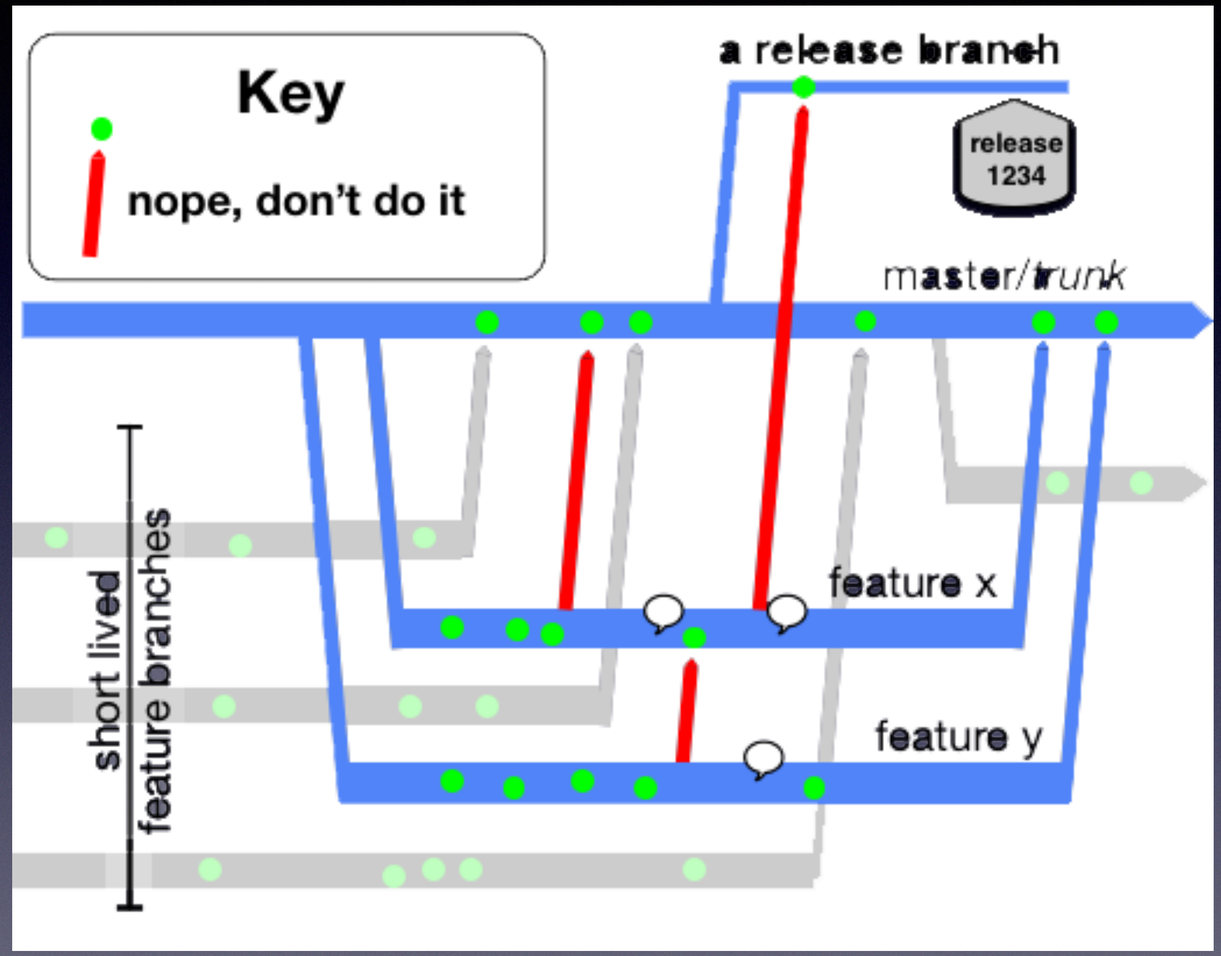


Don’t

Merge prematurely

Merge branch to branch

Merge directly to release



https://trunkbaseddevelopment.com/short-lived-feature-branches/slfb_bad_sharing.png

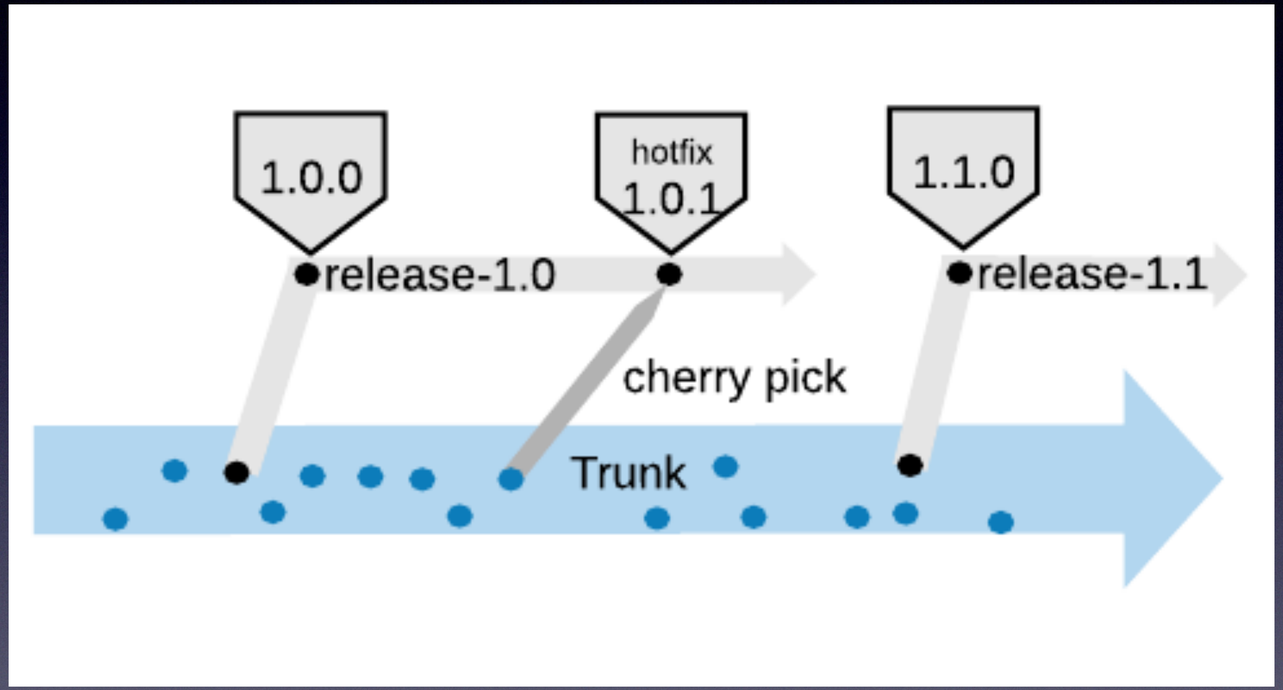
Release Management

Release from Branches

Identify a commit and create release branch

Label release in release branch and deploy it

For hotfix, cherry pick needed commit(s)



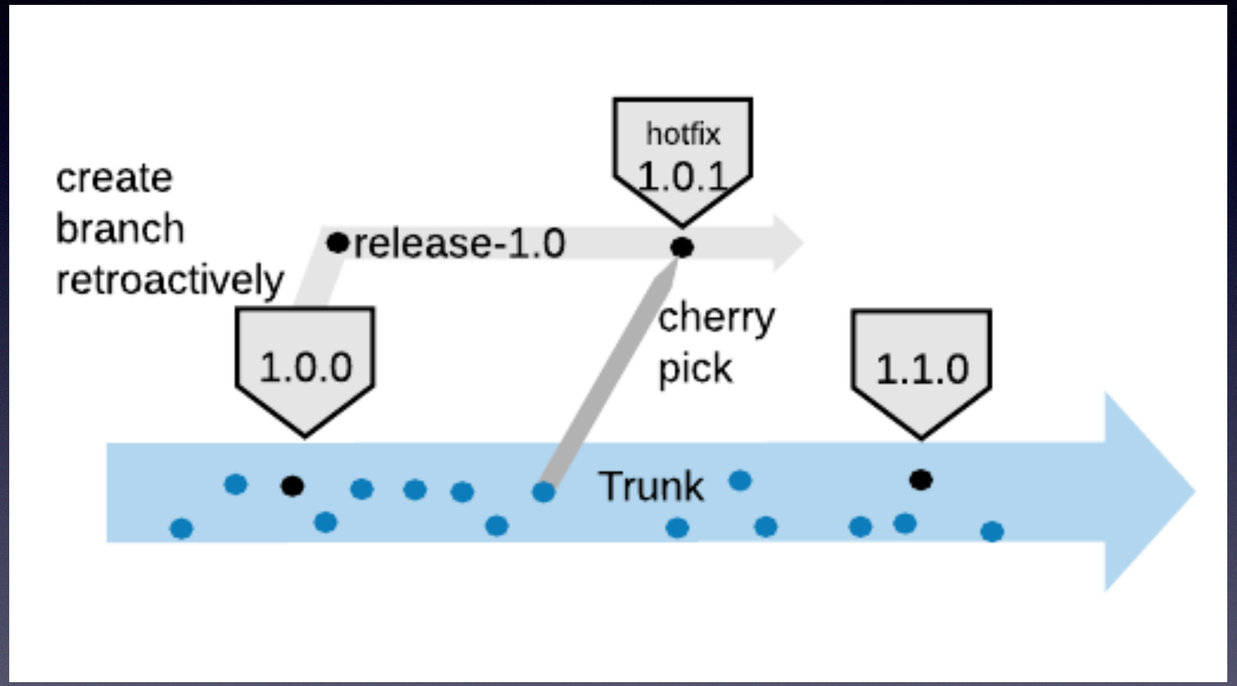
Release from Trunk

Identify a commit and label it.

Deploy the release.

For a hot fix:

- Create branch from label
- Cherry pick commits



Techniques and Tools

Feature Toggles

Release no feature before its time

Should be short lived

Use tooling to manage

Build vs. Buy

- Libraries, good list at http://featureflags.io/feature-flag-introduction/
- Commercial Products

Types of Feature Toggles

Release Toggles

Early Access

Incremental Rollouts

Experiment Toggles

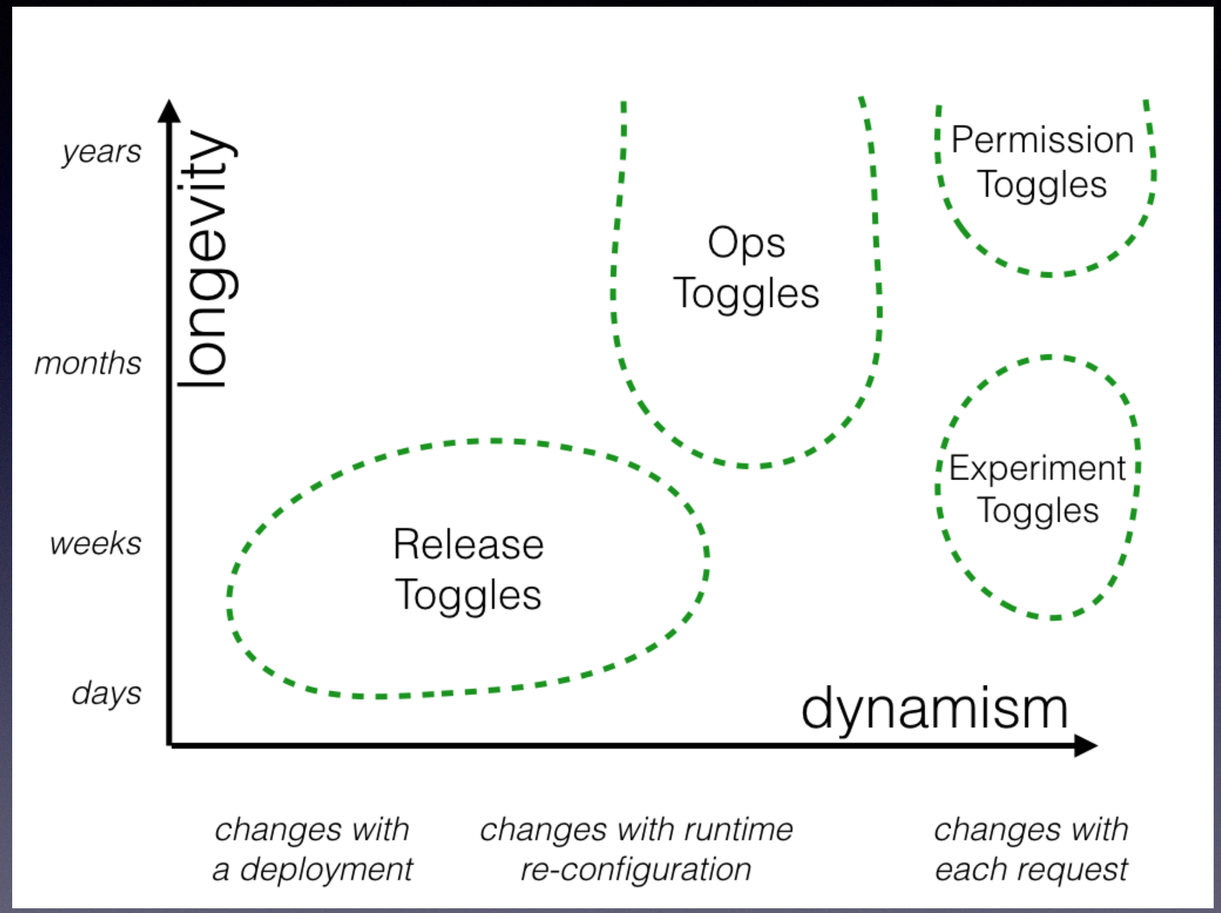
A/B Testing

Ops Toggles

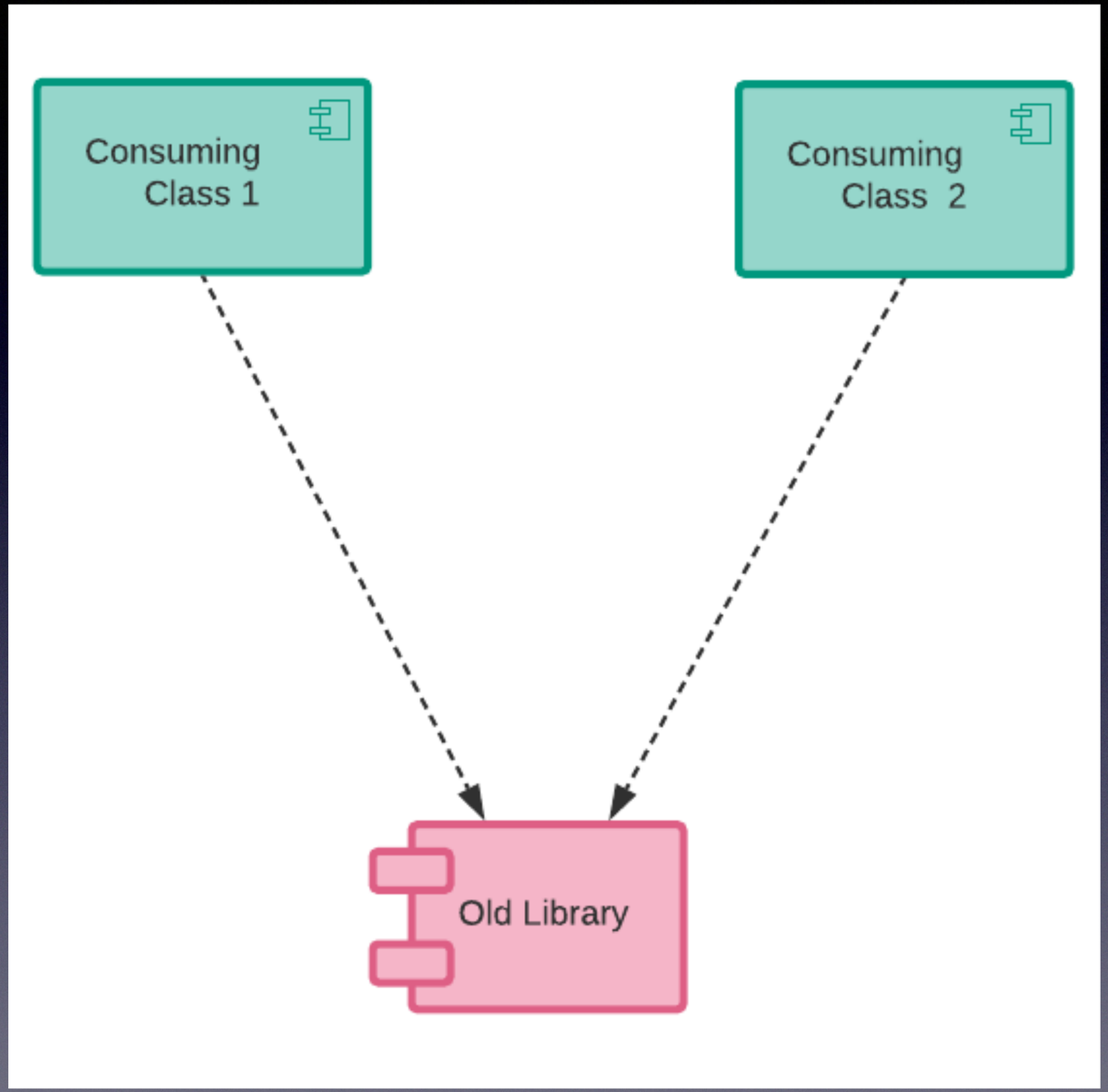
Kill Switch

Permission Toggles

Premium Features

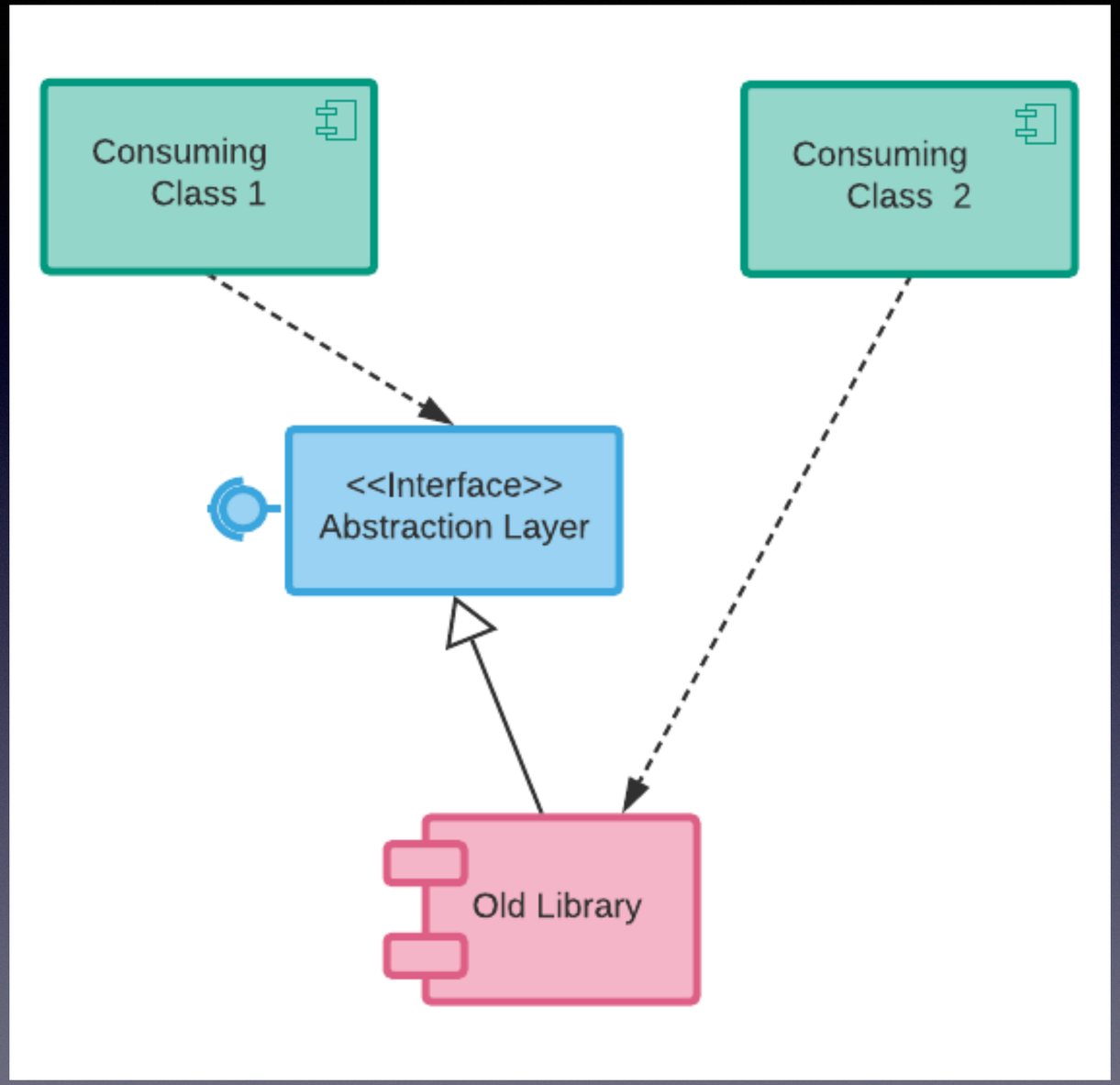


Branch by Abstraction



Branch by Abstraction

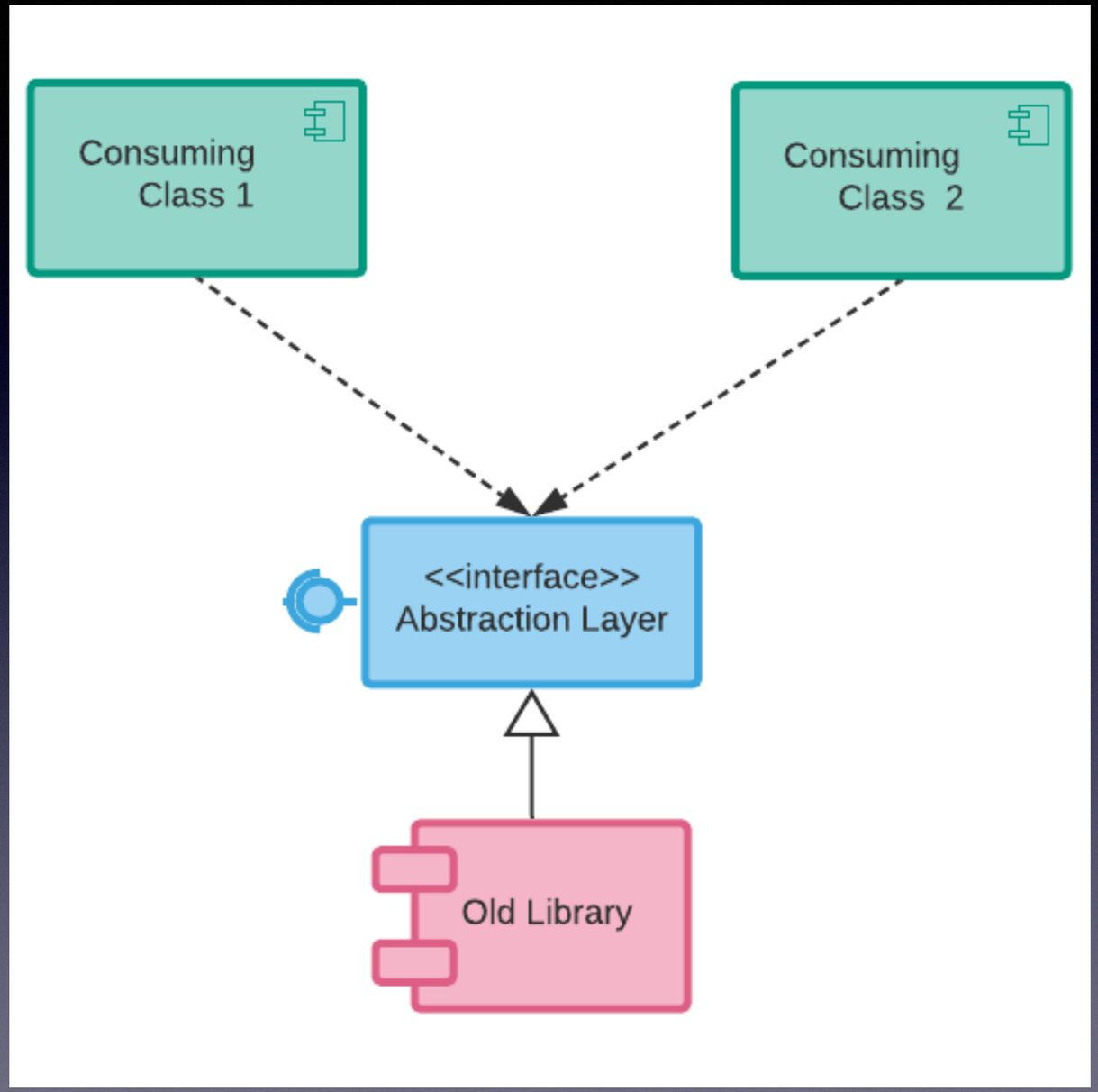
Add an abstraction layer



Branch by Abstraction

Add an abstraction layer

All classes use it

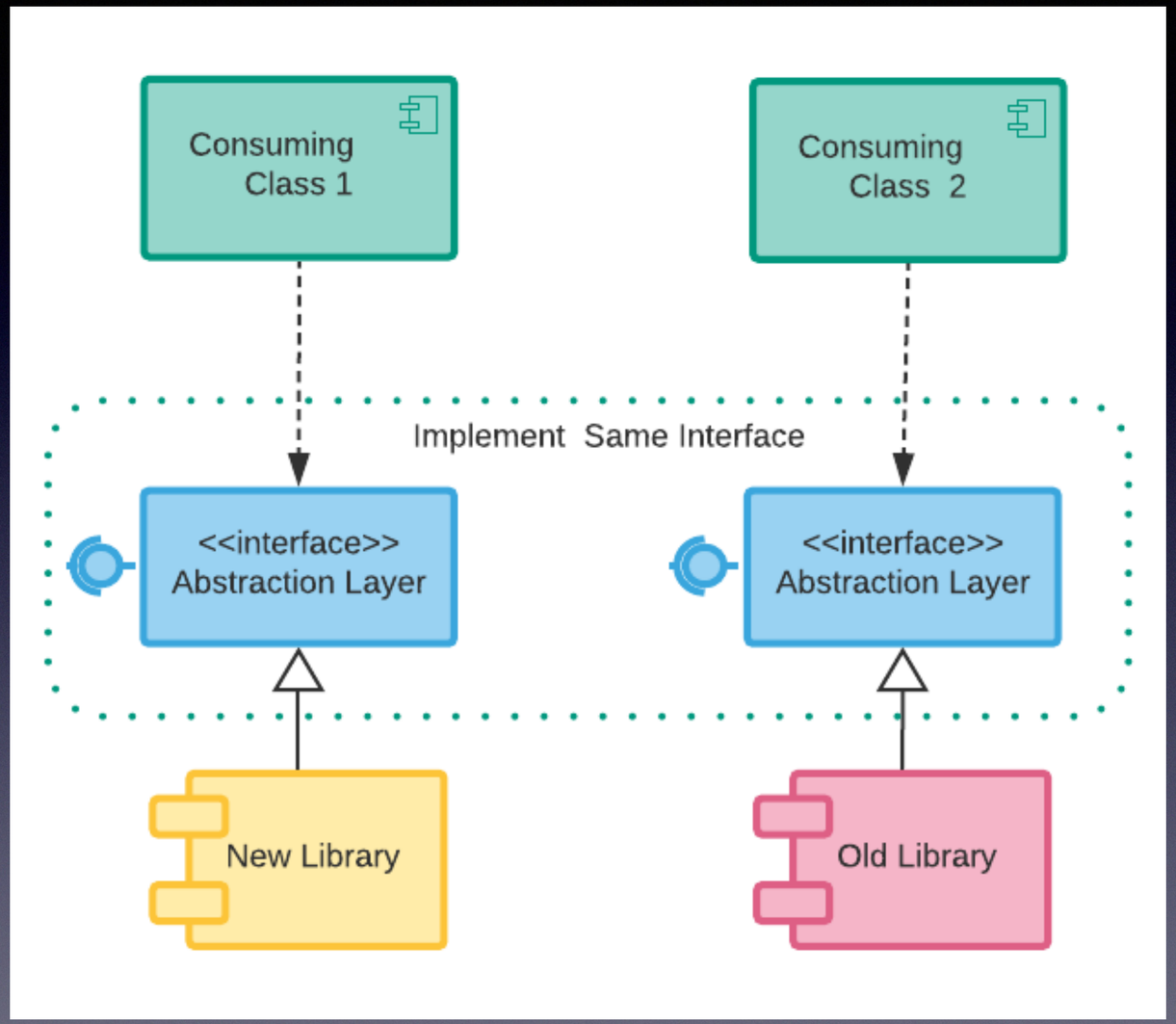


Branch by Abstraction

Add an abstraction layer

All classes use it

Add new library - same abstraction



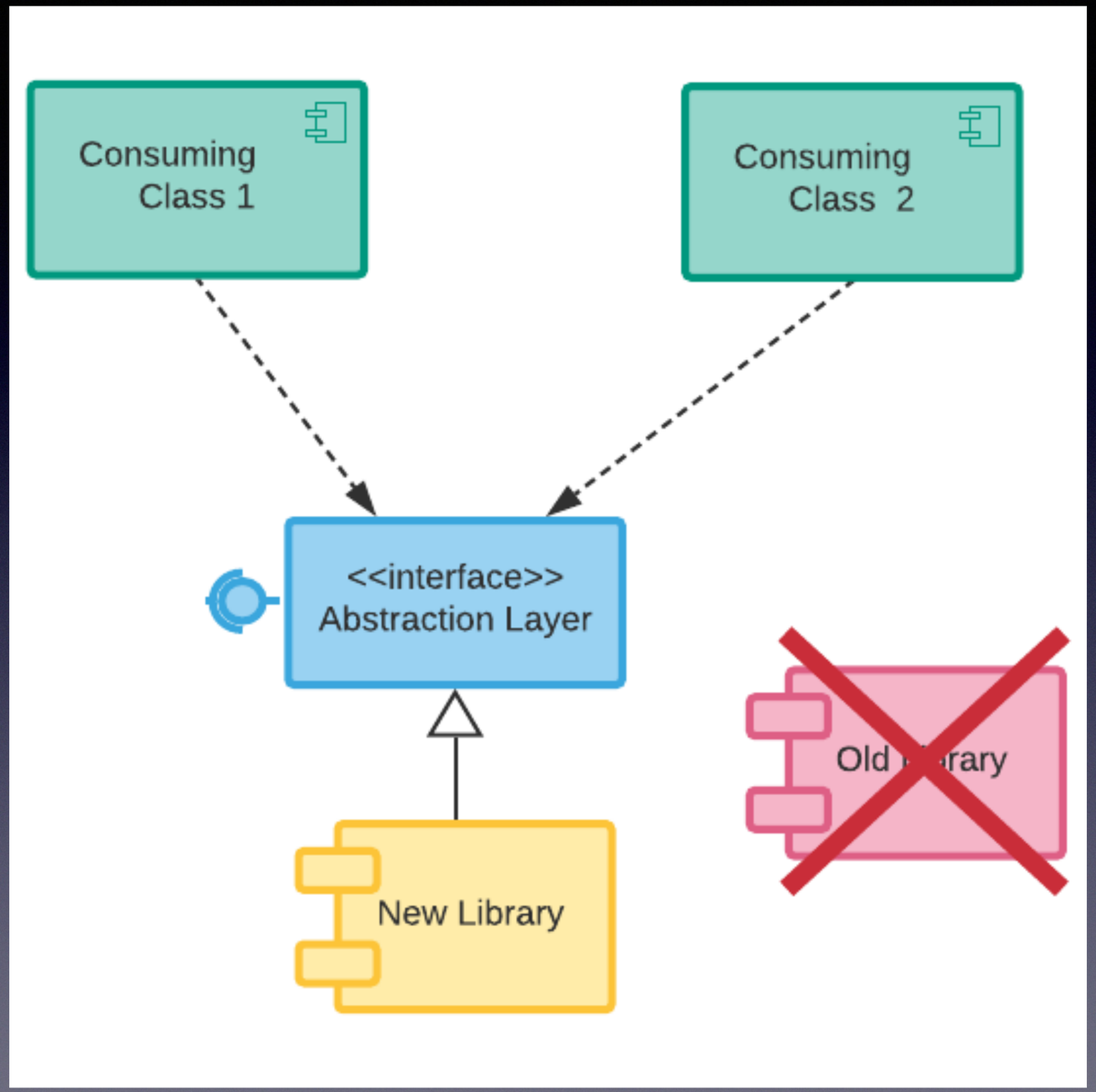
Branch by Abstraction

Add an abstraction layer

All classes use it

Add new library - same abstraction

Remove old library



Branch by Abstraction

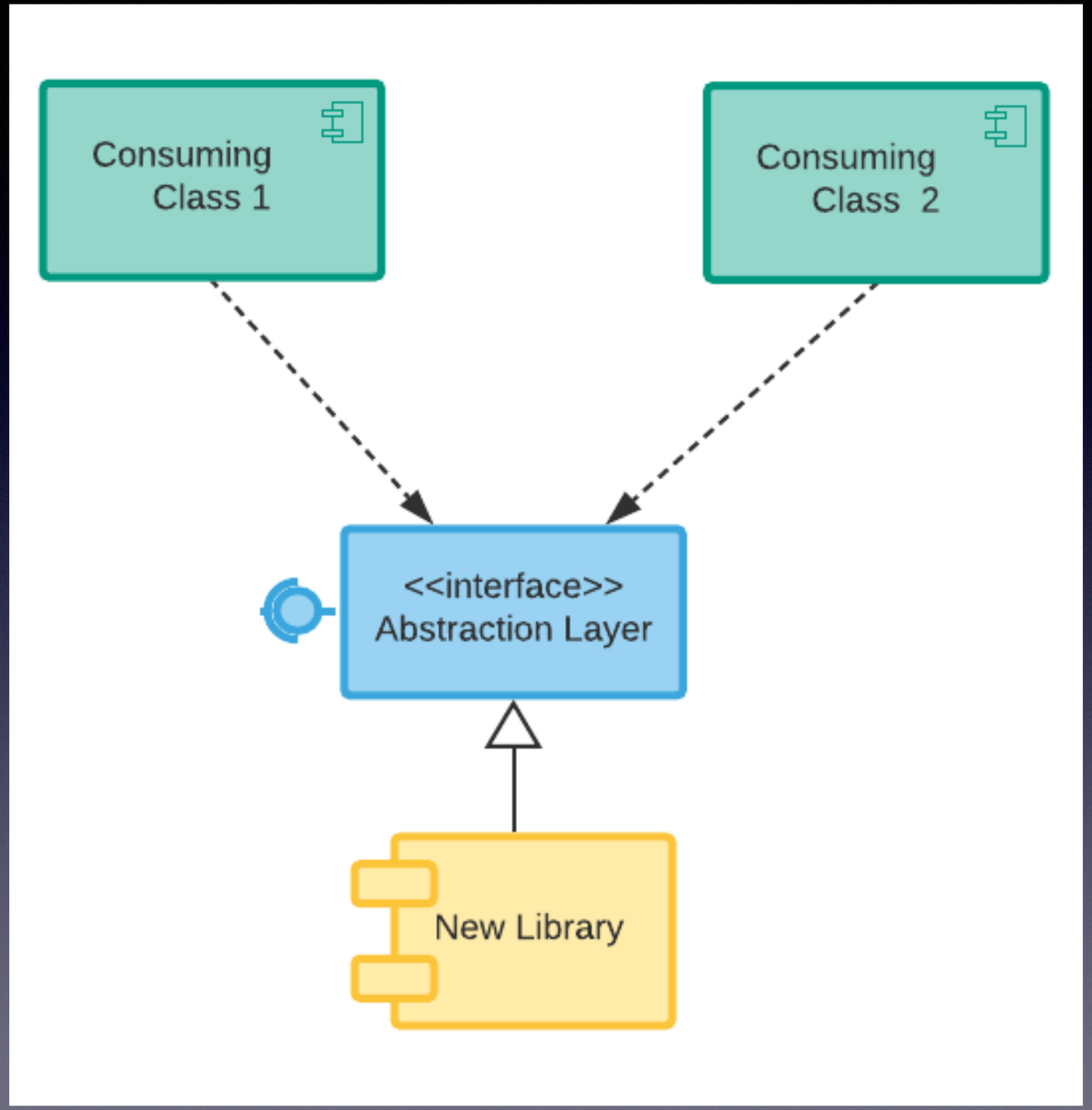
Add an abstraction layer

All classes use it

Add new library - same abstraction

Remove old library

Final Product



Prerequisites

Solid Development Infrastructure

Automated Build Pipeline

Automated Tests

Trunk Based Development Pros & Cons

Pros:

- Experienced developers
- Rapid iterations
- New projects
- Fix merge conflicts early
- Good fit for web based projects or those with automated push

Cons:

- Open source projects
- Historic projects
- Installed projects with long lived release branches
- No automated build
- Poor tests

Summary

Many branching strategies

Trunk based development can accelerate CI/CD

Early discovery of merge issues

Questions

TECH Masters

www.TechMasters-TC.com

Tuesdays, 7:45-9:00 am, Improving Enterprises, Bloomington

Sources

Allen, Jonathon. (23-Apr-2012) *Microsoft's Branching and Merging Guidelines*. https://www.infoq.com/news/2012/04/Branching-Guide/

Ecker, Robert. (22-Mar-2016). *From Git Flow to Trunk Based Development*. https://team-coder.com/from-git-flow-to-trunk-based-development/

featureflags.io. (retrieved 29-Sep-2019). *Feature Flags, Toggles, Controls: The Hub for Feature Flag Driven Development*. http://featureflags.io/

Fowler, Martin. (3-Sep-2009). *FeatureBranch*. https://martinfowler.com/bliki/FeatureBranch.html

Fowler, Martin. (29-Oct-2010). *FeatureToggle*. https://martinfowler.com/bliki/FeatureToggle.html

Fowler, Martin. (7-Jan-2014). *BranchByAbstraction*. https://martinfowler.com/bliki/BranchByAbstraction.html

Gadzinowski, Konrad. *Trunk-based Development vs. Git Flow*. https://www.toptal.com/software/trunk-based-development-git-flow

Hammant, Paul. (2017-2018). *Trunk Based Development*. Entire website: https://trunkbaseddevelopment.com/

Sources, page 2

Hodgson, Pete. (9-Oct-2017). *Feature Toggles (aka Feature Flags)*. https://martinfowler.com/articles/feature-toggles.html

Marker, Sheroy. (30-May-2018). *Continuous Delivery of Microservices - Trunk Based Development and Feature Toggles*. https://www.gocd.org/2018/05/30/ci-microservices-feature-toggles-trunk-based-development/

Mówiński, Kamil. (28 Feb 2018). *Escape from Merge Hell: Why I Prefer Trunk-Based Development Over Feature Branching and GitFlow.* https://stxnext.com/blog/2018/02/28/escape-merge-hell-why-i-prefer-trunk-based-development-over-feature-branching-and-gitflow/

Pytloun, Filip. Slides from *Git Workshop*. https://fpy.cz/pub/slides/git-workshop/#/step-1

Reinertsen, Donald G. (Celeritas Publishing, 2012) *The Principles of Product Development Flow: Second Generation Lean Product Development*. **ASIN: B007TKU0O0**

Wikipedia. (2019) *Feature Toggle*. Retrieved 29-Sep-2019. https://en.wikipedia.org/wiki/Feature_toggle