

XP CONFERENCE INDIA 2016

Powered By  SolutionsIQ

19 - 20 August 2016
Bangalore

www.xpconference.in

Bashing Cultural Monsters in Continuous Integration

Vivek Ganesan

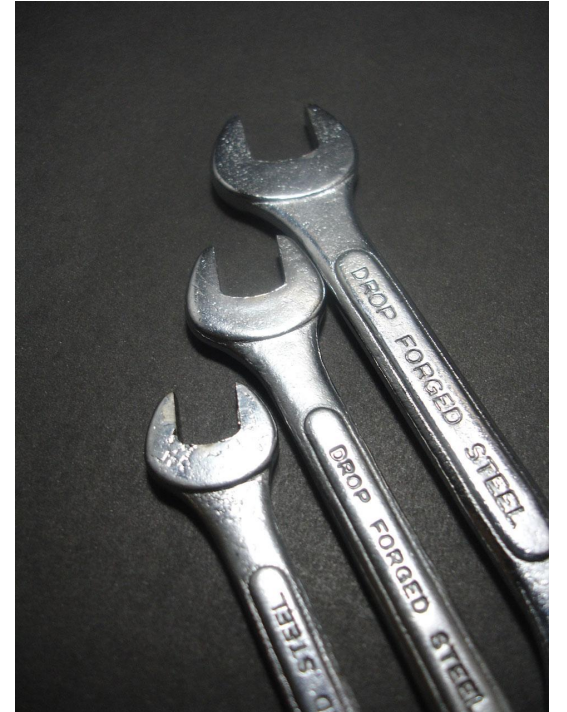
Twitter: @vivek_ganesan

Gainsight

Running Jenkins is NOT Continuous Integration!

Tools that you use \neq What you do.

How do you use the tools? = What you do.





Then, What is CI?

**For Continuous Integration to be useful,
following are 'Mandatory'**

- **Continuous**
Never ending. May be periodical. Good to start with daily frequency.
- **Integration**
Different teams'/people's codes are integrated and tested
- **Stimulus - Response**
The adrenaline rush and events that follow any build failure to bring it back to green

Have you ever observed what happens during the Stimulus-Response?



Tip

You can effectively observe only when you do not have stakes in something.

If you have stakes, you get 'involved' instead of studying what happens.

— Blame Game, Mostly!

- Raj broke the build. What a careless guy!
- Why should I fix it? It is Raj who broke the build!
- Dear Boss,
I am BLOCKED until Raj fixes the build. Why should I suffer for someone else's mistakes?
-The Escalator
- Poor soul Raj! He needs some more training.
- Raj, you are a good developer. But, you could have done better at avoiding some build failures this year!

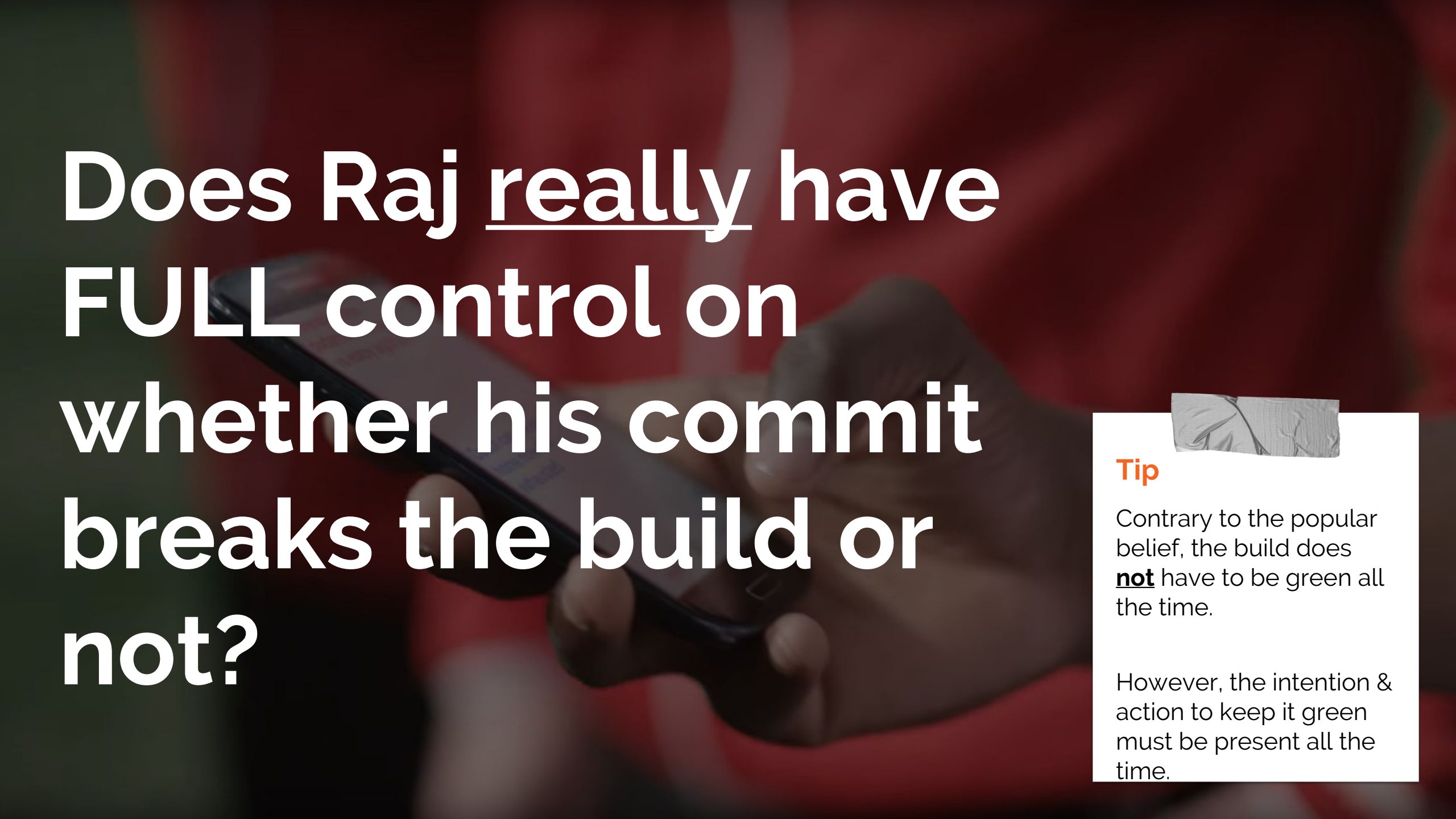
How does this
ADVERSELY
AFFECT Raj & the
organization?





What happens to the following in this situation?

- Commit frequency
- Size of each commit
- Ability to take risks
- Willingness to face uncertainty
- Bug count
- General motivation
- Anything else that we don't know about?

A hand holding a smartphone against a blurred red background. The text is overlaid on the left side of the image.

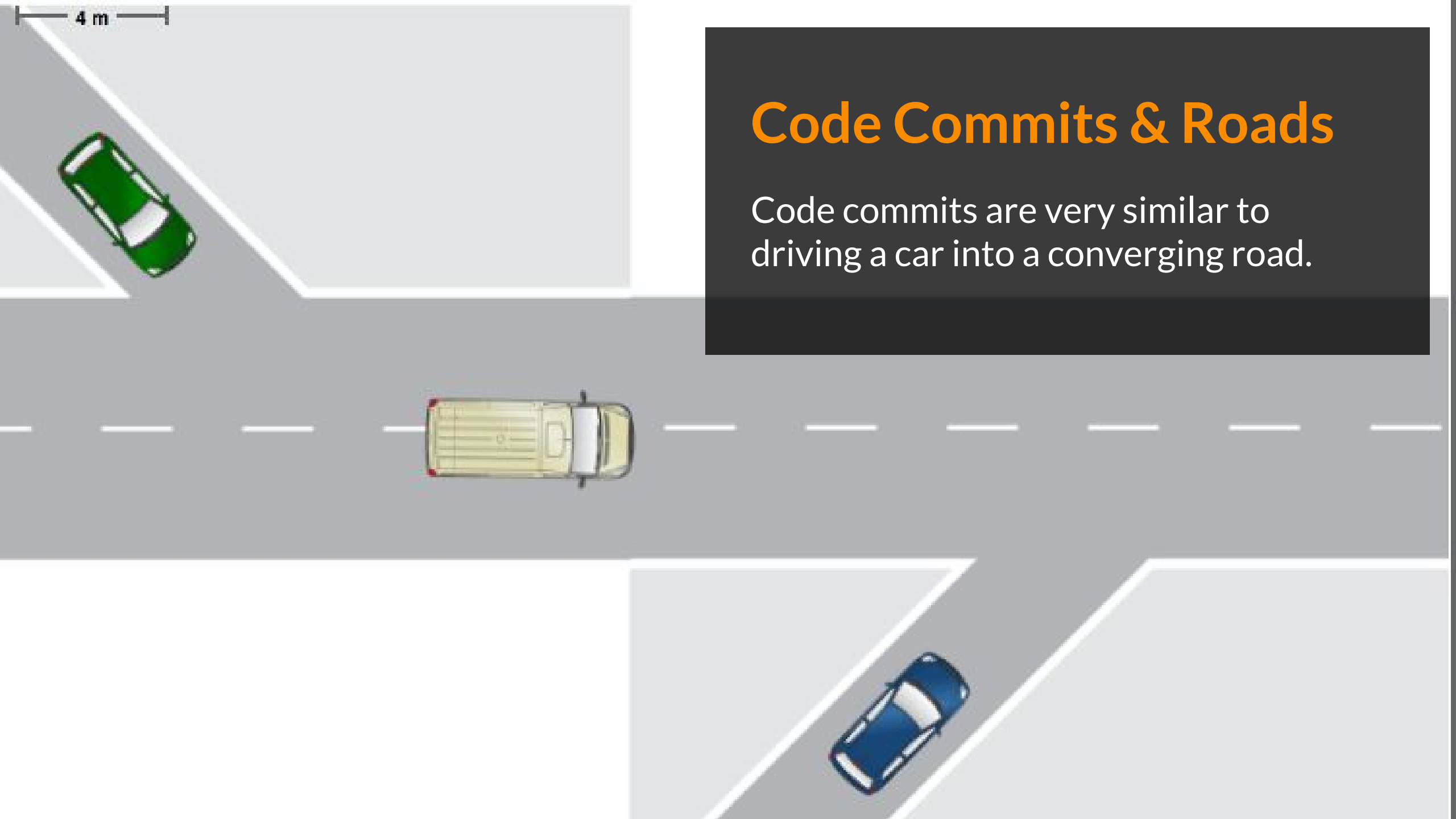
Does Raj really have FULL control on whether his commit breaks the build or not?



Tip

Contrary to the popular belief, the build does **not** have to be green all the time.

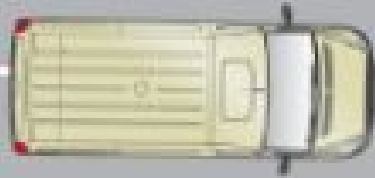
However, the intention & action to keep it green must be present all the time.



Code Commits & Roads

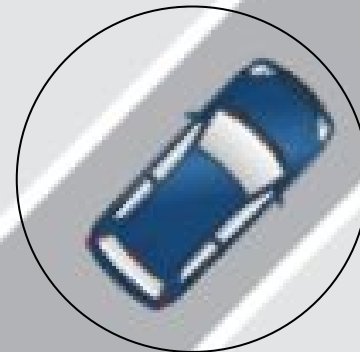
Code commits are very similar to driving a car into a converging road.

4 m



Code Commits & Roads

Code commits are very similar to driving a car into a converging road.



Your Commit

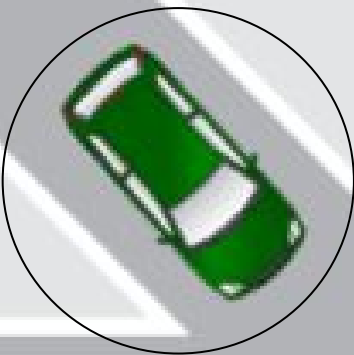
Code Commits & Roads

Code commits are very similar to driving a car into a converging road.

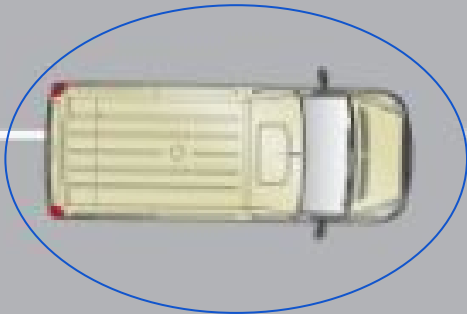
Team mate's
Pending
Commit

Your Commit

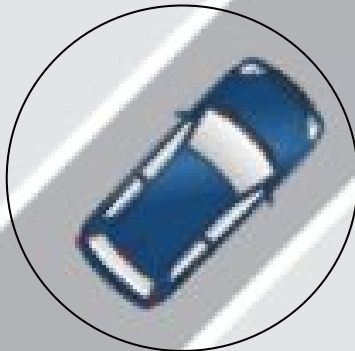
4 m



Team mate's
Pending
Commit

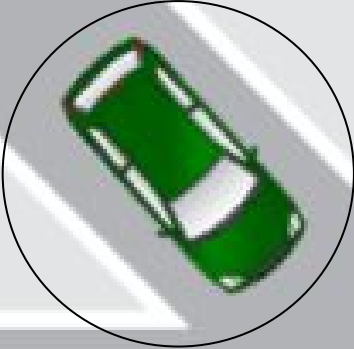


Some commit
done after your
last pull

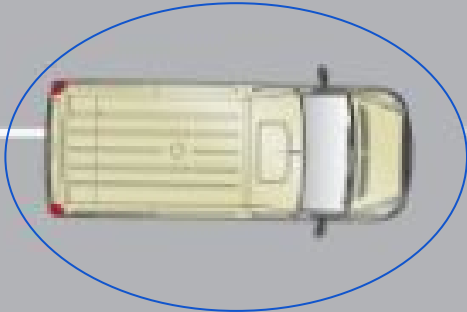


Your Commit

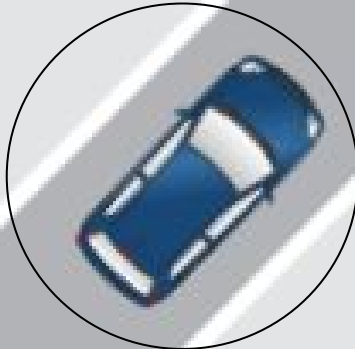
4 m



Team mate's
Pending
Commit



Some commit
done after your
last pull



Your Commit

Challenge

Can you be 100% sure about driving
into the main road without an accident?

Challenge - 2

What would you do if you really have to be sure not to hit another vehicle?

Team mate's
Pending
Commit

Some commit
done after your
last pull

Your Commit

—

This is just a piece of ridiculous joke!



—
This is just a piece of ridiculous joke!

**What if the car
could not run
properly even
before merge?**

—
This is just a piece of ridiculous joke!

What if **the car**
could not run
properly even
before merge?



Tip

There are two types of failures:

1. Avoidable
2. Unavoidable

What if a developer creates an avoidable failure?

Eg: Missed a semi-colon and checked-in

Solution: Find ways to help the developer to avoid it

Best Case: Automate the avoidable failure check before commit (using pre-commit hooks or equivalent)

Worst Case: Police a checklist for commit.



Tip

Developers don't intentionally cause havoc, unless you steal their coffee mug.

FAILURE IS NOT AN OPTION

Tip

Instead of punishing the build breaker, appreciate the build fixer, even if they both are same :)

This motivates people to fix broken builds immediately.

IT IS A *REQUIREMENT*

And, what after an unavoidable failure? All hell breaks loose?

No, my friend! You wait until someone fixes the build.

Remember! This build fixer need not be a build breaker.

After build fixed, publicly appreciate the build fixer and exclaim “Woohoo! Another integration issue fixed early!”



What to track?

Don't track who broke the build and how many times.

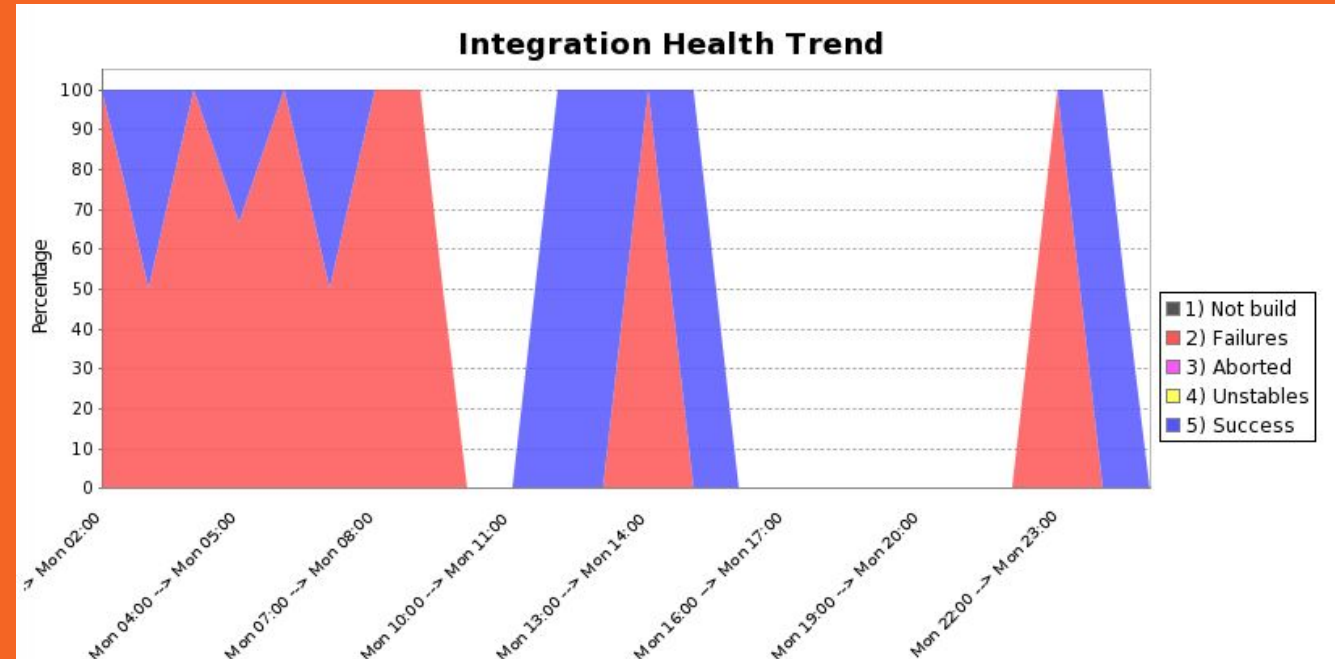
Now that we care about who fixes the build, track these:

→ Total Build Red Time

Average time in a day for which build was Red. Trend it transparently & aim for reducing the same.

→ No. of Unavoidable Failures

This is the no. of integration issues that you fixed early :)





Questions???

Create an atmosphere free of blame and full of trust and transparency.

For more (free) agility tips visit my site

<http://www.vivekganesan.com> or follow me on Twitter at [@vivek_ganesan](https://twitter.com/vivek_ganesan)