36th International Conference on Software Engineering
ICSE 2014
Hyderabad, India · May 31 - June 7, 2014

How Do Centralized and Distributed Version Control Systems Impact Software Changes?

Caius Brindescu
Mihai Codoban
Sergii Shmarkatiuk
Danny Dig

OSU
Oregon State
UNIVERSITY

Oregon State
UNIVERSITY

GitHub is the main “forge” for OSS projects

SourceForge
300K repos

GitHub
4.6M repos

What's the difference?

	Git	SVN
History	Local to every user	On the server
Commits	Private, local	Centralized, public
Branching and merging	Cheap	Expensive
History	Modifiable	"Set in stone"

What's the difference?

	Git	SVN
History	Local to every user	On the server
Commits	Private, local	Centralized, public
Branching and merging	Cheap	Expensive
History	Modifiable	"Set in stone"

What's the difference?

	Git	SVN
History	Local to every user	On the server
Commits	Private, local	Centralized, public
Branching and merging	Cheap	Expensive
History	Modifiable	"Set in stone"

What's the difference?

	Git	SVN
History	Local to every user	On the server
Commits	Private, local	Centralized, public
Branching and merging	Cheap	Expensive
History	Modifiable	"Set in stone"

What are we missing?

Developers Ⓓ

Managers Ⓜ

Researchers Ⓡ

Tool Builders Ⓣ

What are we missing?

Developers Ⓓ

Managers Ⓜ

Researchers Ⓡ

Tool Builders Ⓣ

What are we missing?

Developers  Ⓓ

Are they using the tools to their full potential?

Managers Ⓜ

Researchers Ⓡ

Tool Builders Ⓣ

What are we missing?

Developers

Ⓓ

Are they using the tools to their full potential?

Managers

Ⓜ

Researchers

Ⓡ

Tool Builders

Ⓣ

What are we missing?

Developers  Ⓓ

Are they using the tools to their full potential?

Managers  Ⓜ

Is switching to Git good?

Researchers Ⓡ

Tool Builders Ⓣ

What are we missing?

Developers  Ⓓ

Are they using the tools to their full potential?

Managers  Ⓜ

Is switching to Git good?

Researchers Ⓡ

Tool Builders Ⓣ

What are we missing?

Developers  Ⓓ

Are they using the tools to their full potential?

Managers  Ⓜ

Is switching to Git good?

Researchers  Ⓡ

How does this new paradigm affect mining software repositories?

Tool Builders Ⓣ

What are we missing?

Developers  Ⓓ

Are they using the tools to their full potential?

Managers  Ⓜ

Is switching to Git good?

Researchers  Ⓡ

How does this new paradigm affect mining software repositories?

Tool Builders Ⓣ

What are we missing?

Developers  Ⓓ

Are they using the tools to their full potential?

Managers  Ⓜ

Is switching to Git good?

Researchers  Ⓡ

How does this new paradigm affect mining software repositories?

Tool Builders  Ⓣ

Are they building the right tools?

Survey

820 participants

Survey

820 participants

85% from **industry**

56% have **over 10 years** experience

51% work in **teams of 6** or larger

Repository Analysis

132 repositories

358K commits

409M LOC

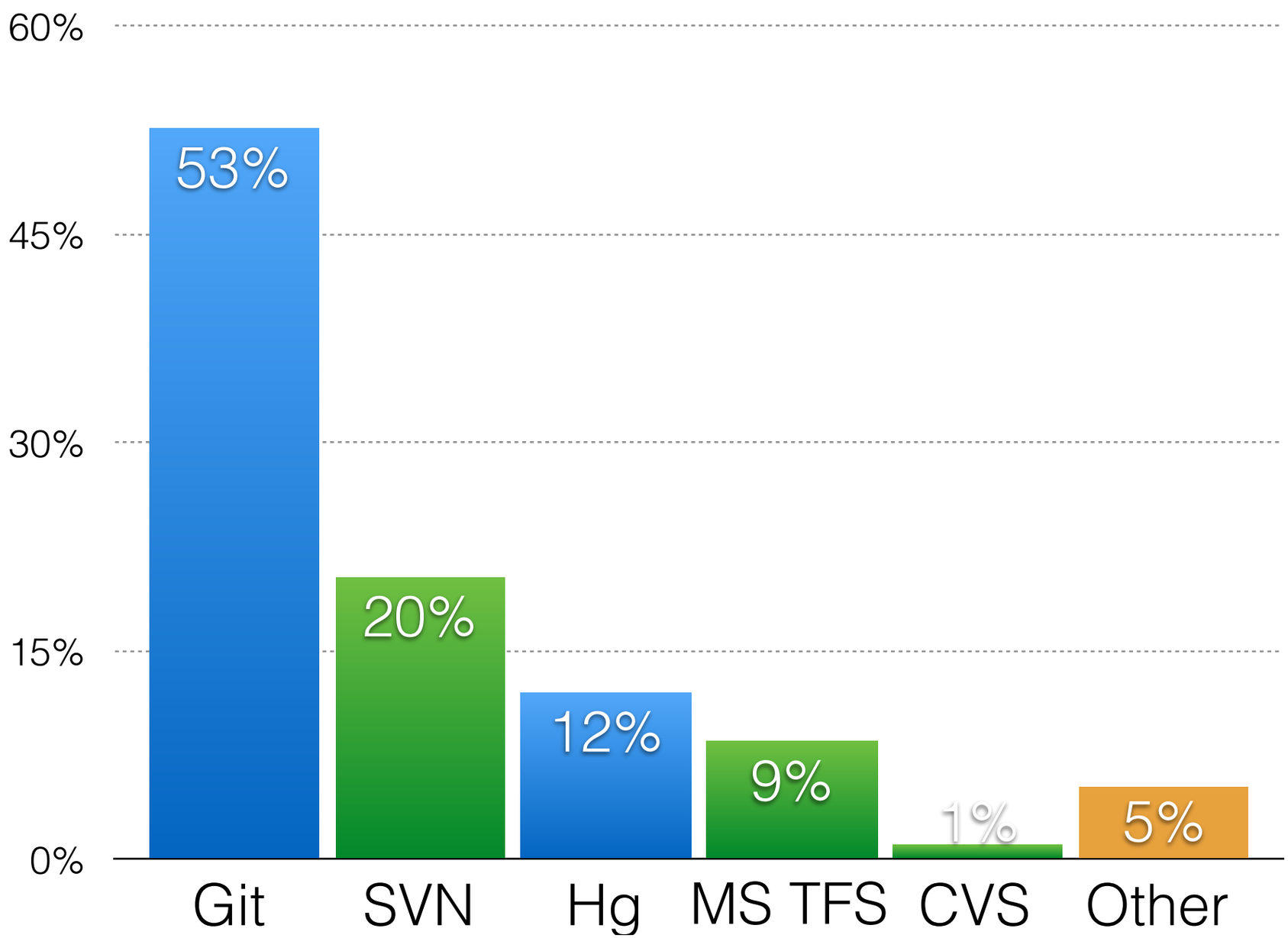
Repository Analysis

52 SVN 51 Git 29 Hybrid

358K commits

409M LOC

Git is the most used VCS



We identified 3 themes

1. Impact of VCS on developer's behavior

RQ 1: Does the type of VCS affect the size of commits?
RQ 2: Do developers split their commits into logical units of change? How do they do it?
RQ 3: How often and why do developers squash their commits?
RQ 4: Why do developers prefer one Version Control System over another?
RQ 5: Does the VCS influence the frequency with which developers commit?

2. Impact of the team size on the VCS

RQ 6: Does team size affect the choice of VCS?
RQ 7: Are larger teams more likely to use Issue Tracking Systems (ITS)?
RQ 8: Does team size affect the size of commits?
RQ 9: Does team size influence commit squashing?

3. Impact of the VCS on the software process

RQ 10: Does the type of VCS influence the presence and the number of issue tracking labels (ITL)?
RQ 11: Is there a correlation between the number of ITLs in the commit message and the commit size?
RQ 12: How does the size of commits vary in time?

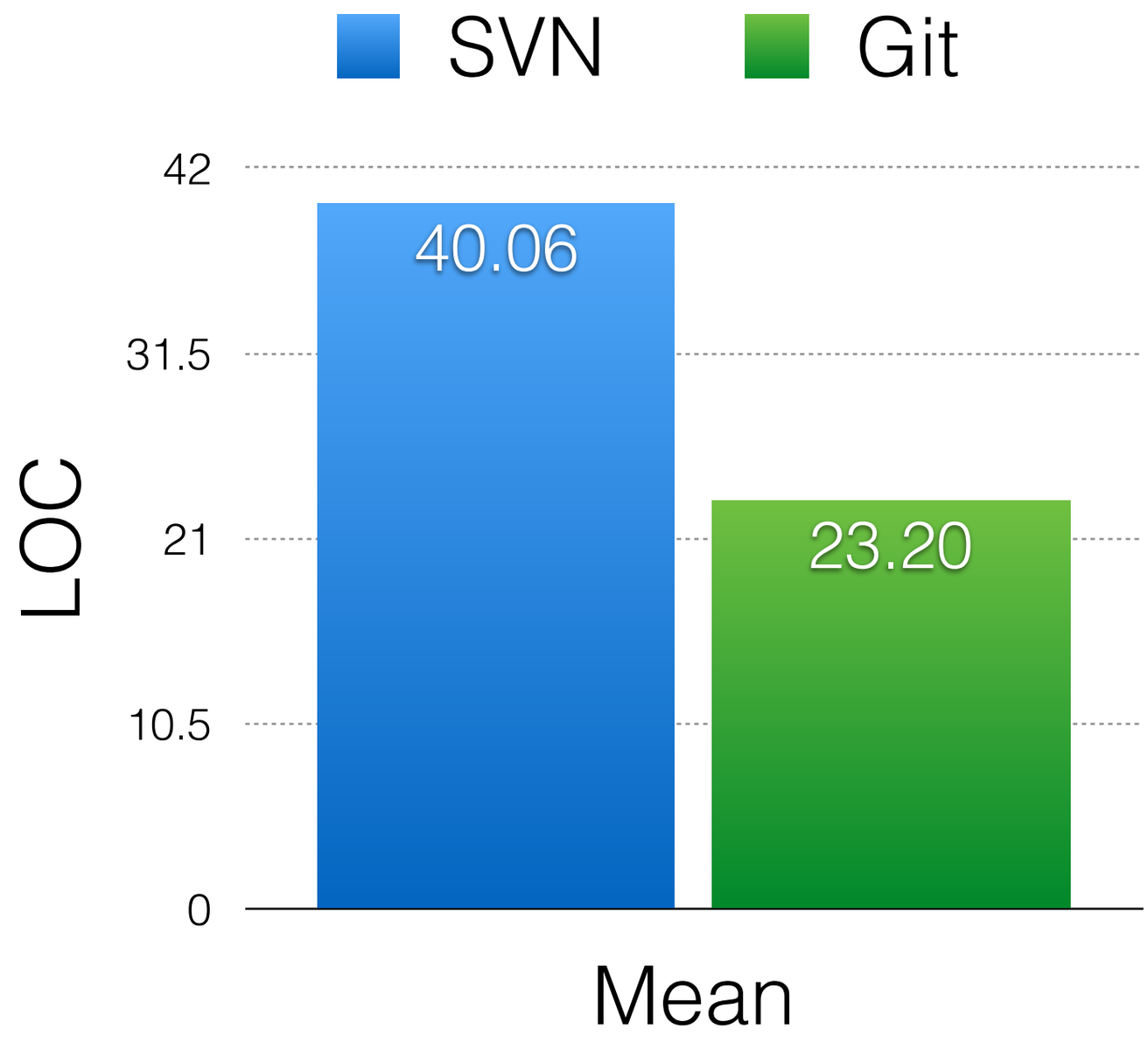
We identified 3 themes

1. Impact of VCS on developer's behavior

RQ 1: Does the type of VCS affect the size of commits?
RQ 2: Do developers split their commits into logical units of change? How do they do it?
RQ 3: How often and why do developers squash their commits?
RQ 4: Why do developers prefer one Version Control System over another?

RQ1: Does the type of VCS affect commit size?

For Git and SVN the difference was statistically significant

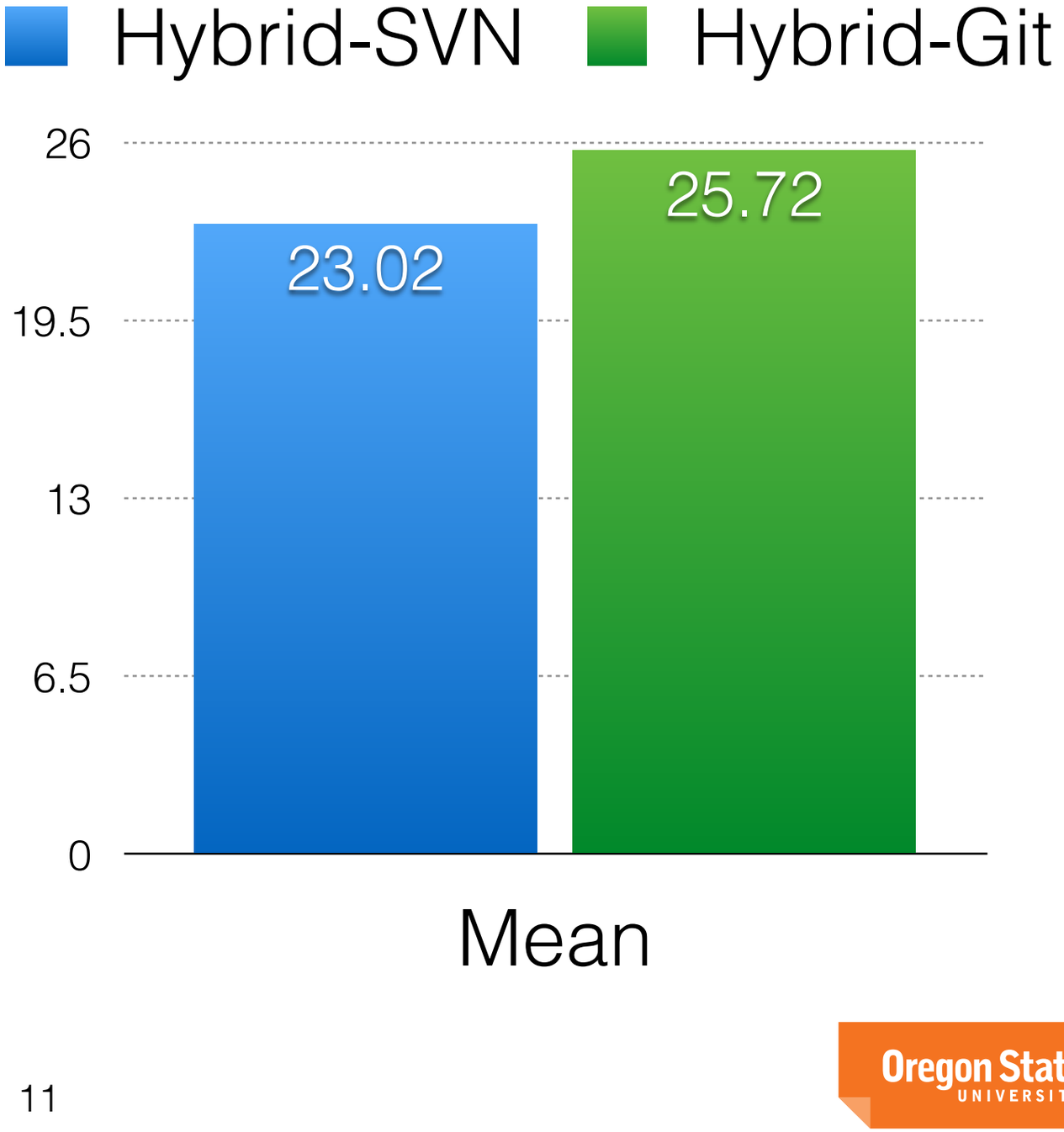


RQ1: Does the type of VCS affect commit size?

"Git promotes the idea that your commit space is ***not inflicting pain*** *on anybody else […] it* ***promotes small frequent commits*** *[…] rather than the 5pm commit"*

RQ1: Does the type of VCS affect commit size?

For repositories that transitioned, there was no statistically significant difference



RQ1: Does the type of VCS affect commit size?

Git repositories have commits size 34% smaller than SVN repositories, in terms of LOC

RQ1: Does the type of VCS affect commit size?

Git repositories have commits size 34% smaller than SVN repositories, in terms of LOC

One possible explanation is that each developer commits to their own local repo, with no need for synchronization or merging their changes.

Hybrid repos keep the same commit size because of existing policies.

RQ1: Does the type of VCS affect commit size?

Git repositories have commits size 34% smaller than SVN repositories, in terms of LOC

One possible explanation is that each developer commits to their own local repo, with no need for synchronization or merging their changes.

Hybrid repos keep the same commit size because of existing policies. ***Old habits die hard***

Implications

Smaller commits makes it easier to **“bisect”** the tree Ⓓ

Git offers better tools for splitting commits ⒹⓇ

Some repositories migrate from one paradigm to the other; this might bias the results Ⓡ

Changing the VCS is not enough Ⓜ

RQ2: Do developers split their changes?

Separating the changes to the working copy into multiple, separate commits

```
file1.txt
file2.txt
file3.txt
```

RQ2: Do developers split their changes?

Separating the changes to the working copy into multiple, separate commits

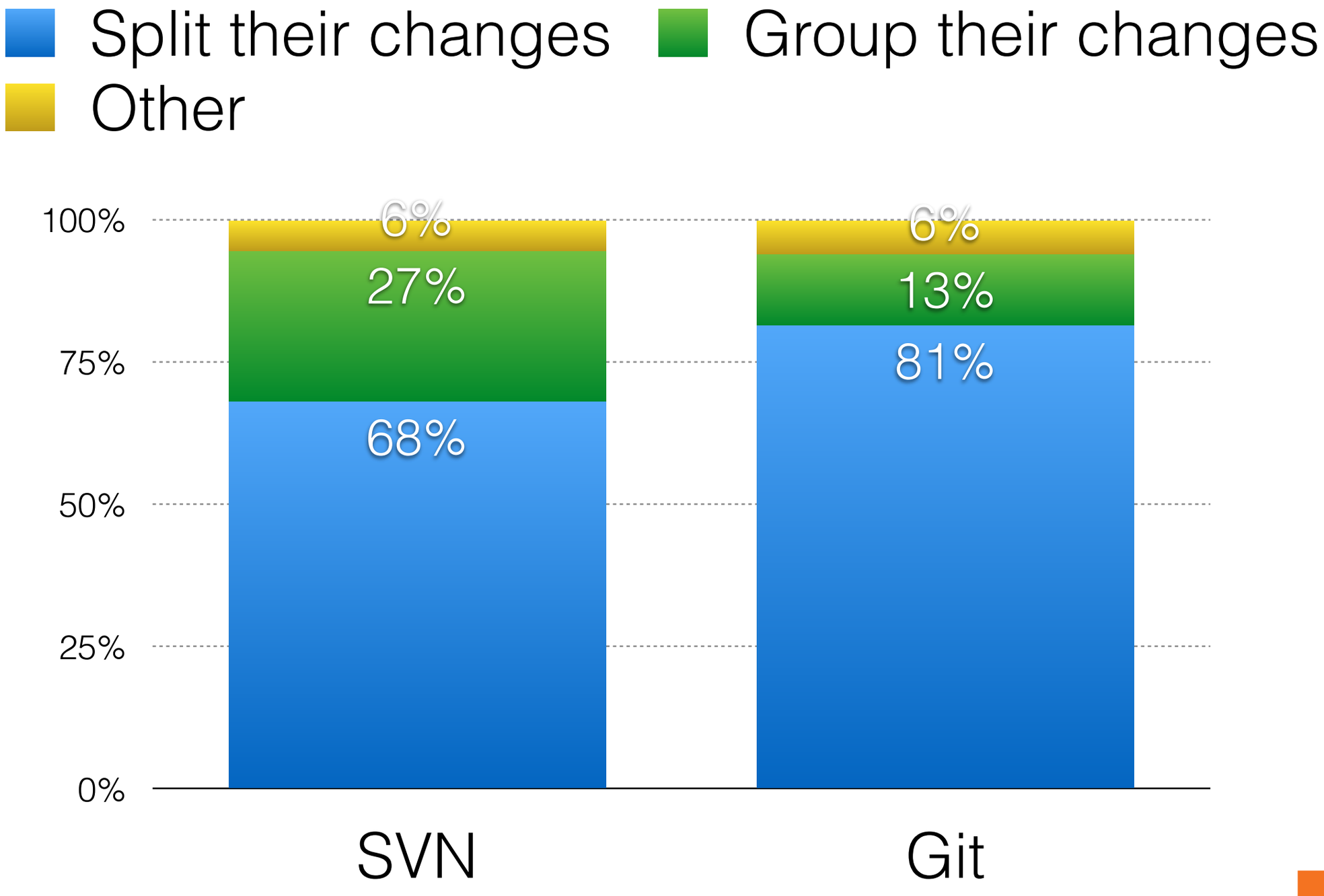
Commit 1

```
file1.txt
```

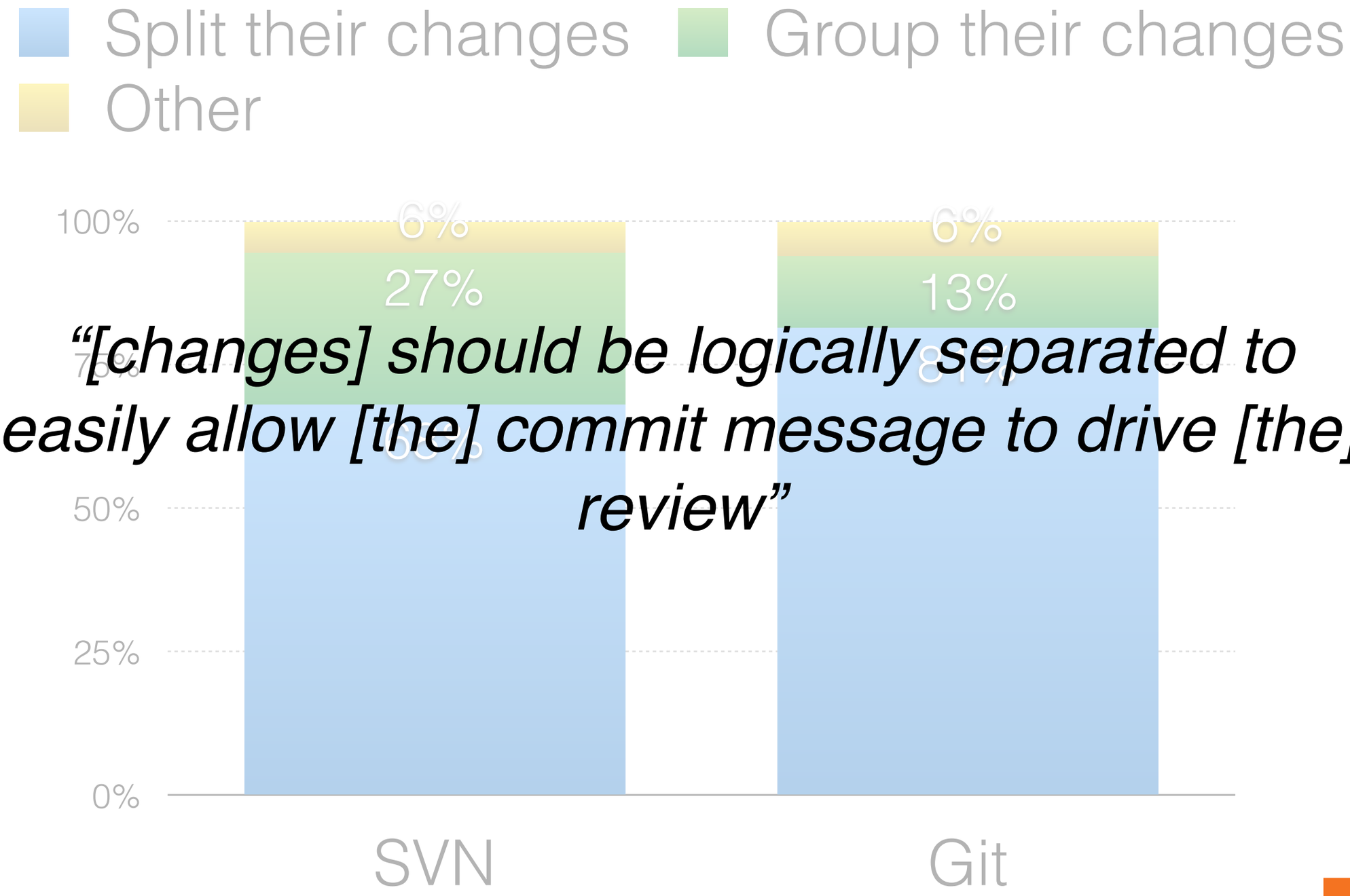
Commit 2

```
file2.txt
file3.txt
```

RQ2: Do developers split their changes?

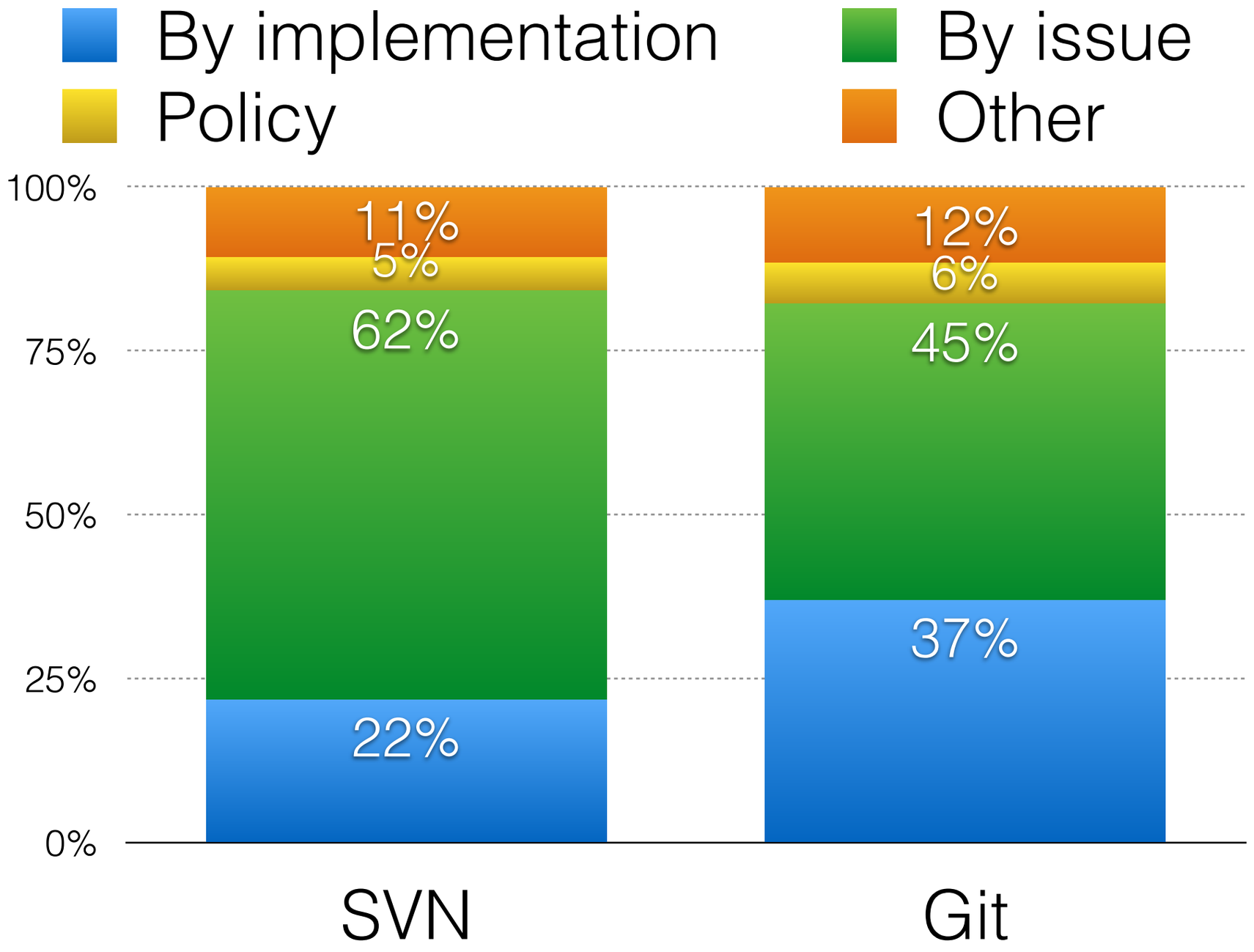


RQ2: Do developers split their changes?

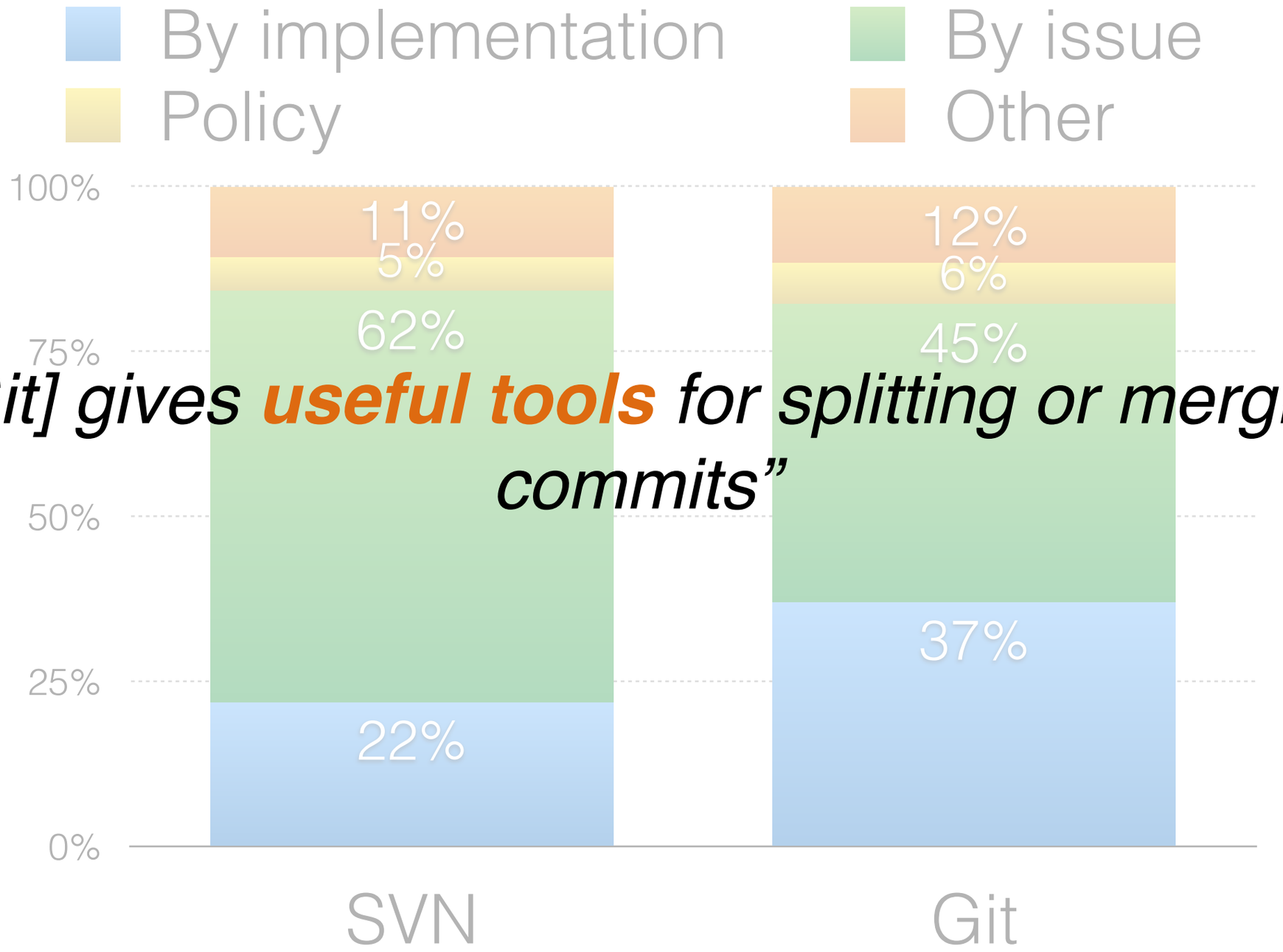


"[changes] should be logically separated to easily allow [the] commit message to drive [the] review"

RQ2: Do developers split their changes?



RQ2: Do developers split their changes?



*"[Git] gives **useful tools** for splitting or merging commits"*

RQ2: Do developers split their changes?

76% of developers split their commits. The percentage is higher for Git (81.25%), compared to SVN (67.89%).

RQ2: Do developers split their changes?

76% of developers split their commits. The percentage is higher for Git (81.25%), compared to SVN (67.89%).

We attribute this to an easier commit process.

RQ2: Do developers split their changes?

76% of developers split their commits. The percentage is higher for Git (81.25%), compared to SVN (67.89%).

We attribute this to an easier commit process.

Overall, developers choose to split their commits based on the issue they belong to.

RQ2: Do developers split their changes?

For Git, more users (37%) split changes based on implementation details that in SVN (22%).

RQ2: Do developers split their changes?

For Git, more users (37%) split changes based on implementation details that in SVN (22%).

"Each commit is one cohesive change […] (like 'sphere class can now calculate its own volume') - user level features usually take many commits."

Implications

Doing this makes it easier to perform other operations such as **cherry-picking**.

Ⓓ

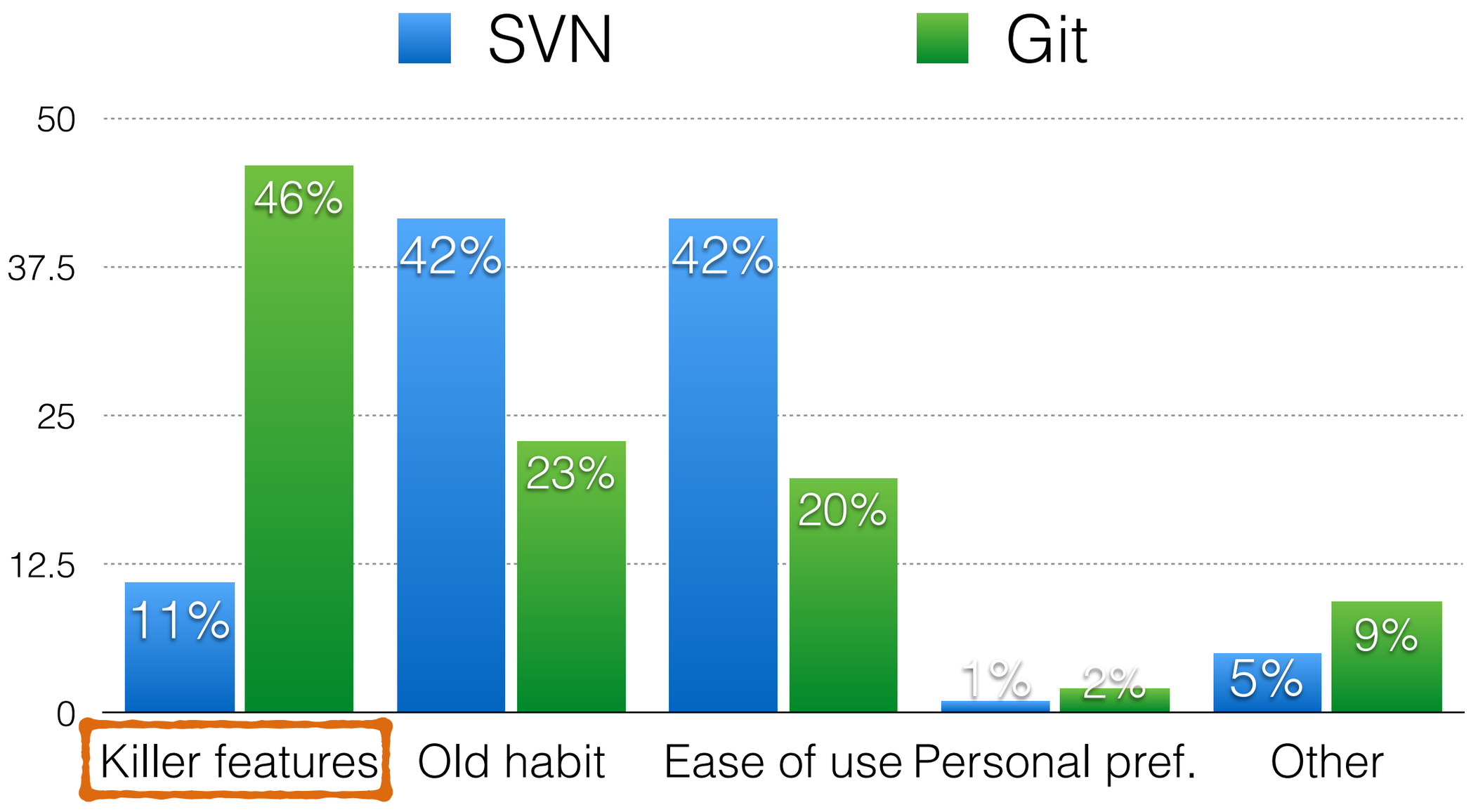
Implications

Doing this makes it easier to perform other operations such as **cherry-picking**.  Ⓓ

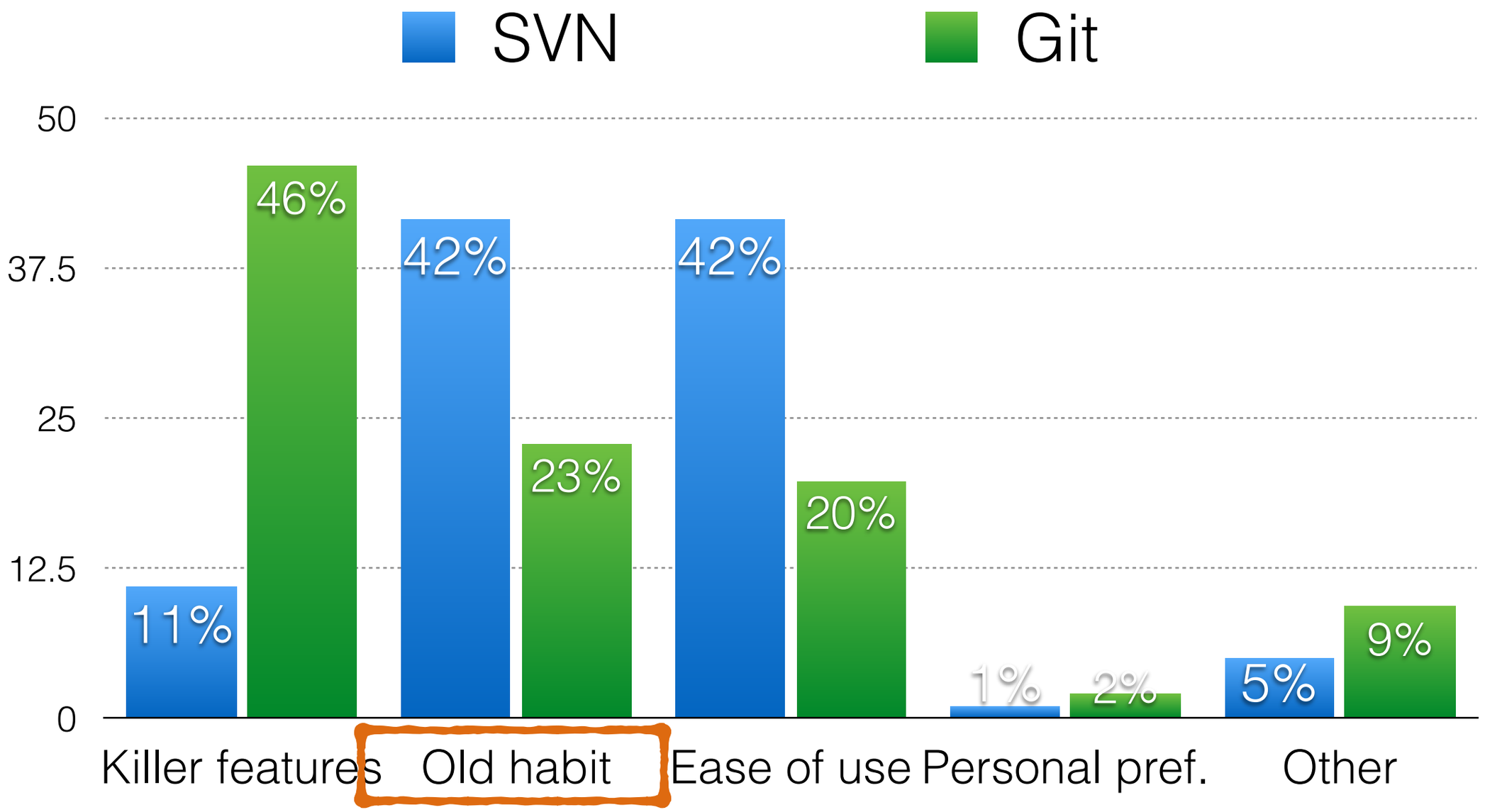
For mining software repositories, Git might be better since it allows **smaller atomic changes**. Ⓡ

Splitting changes is a **manual and tedious process**. Tool builders could make their tools support this process Ⓣ

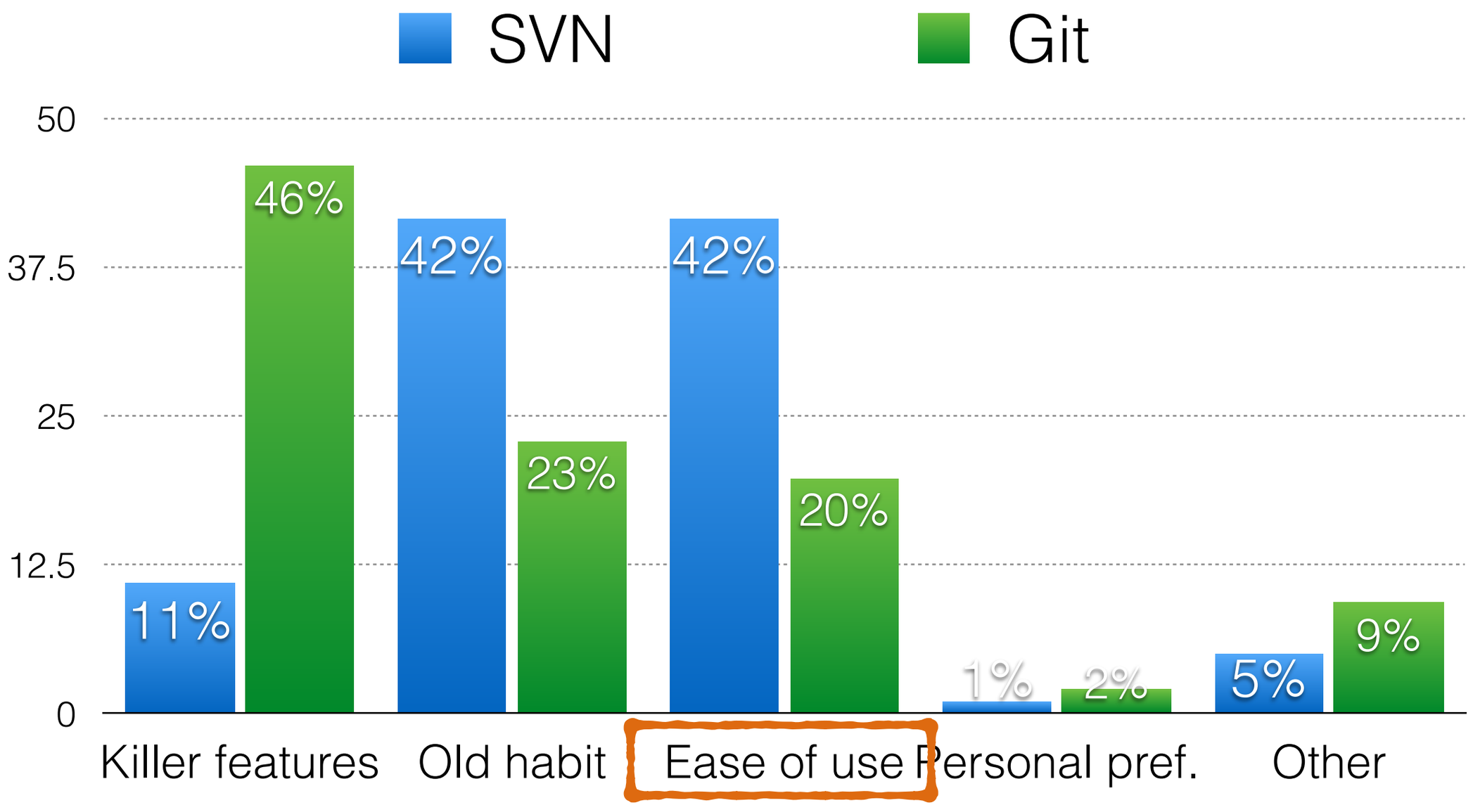
RQ3: Why do developers prefer one VCS over another?



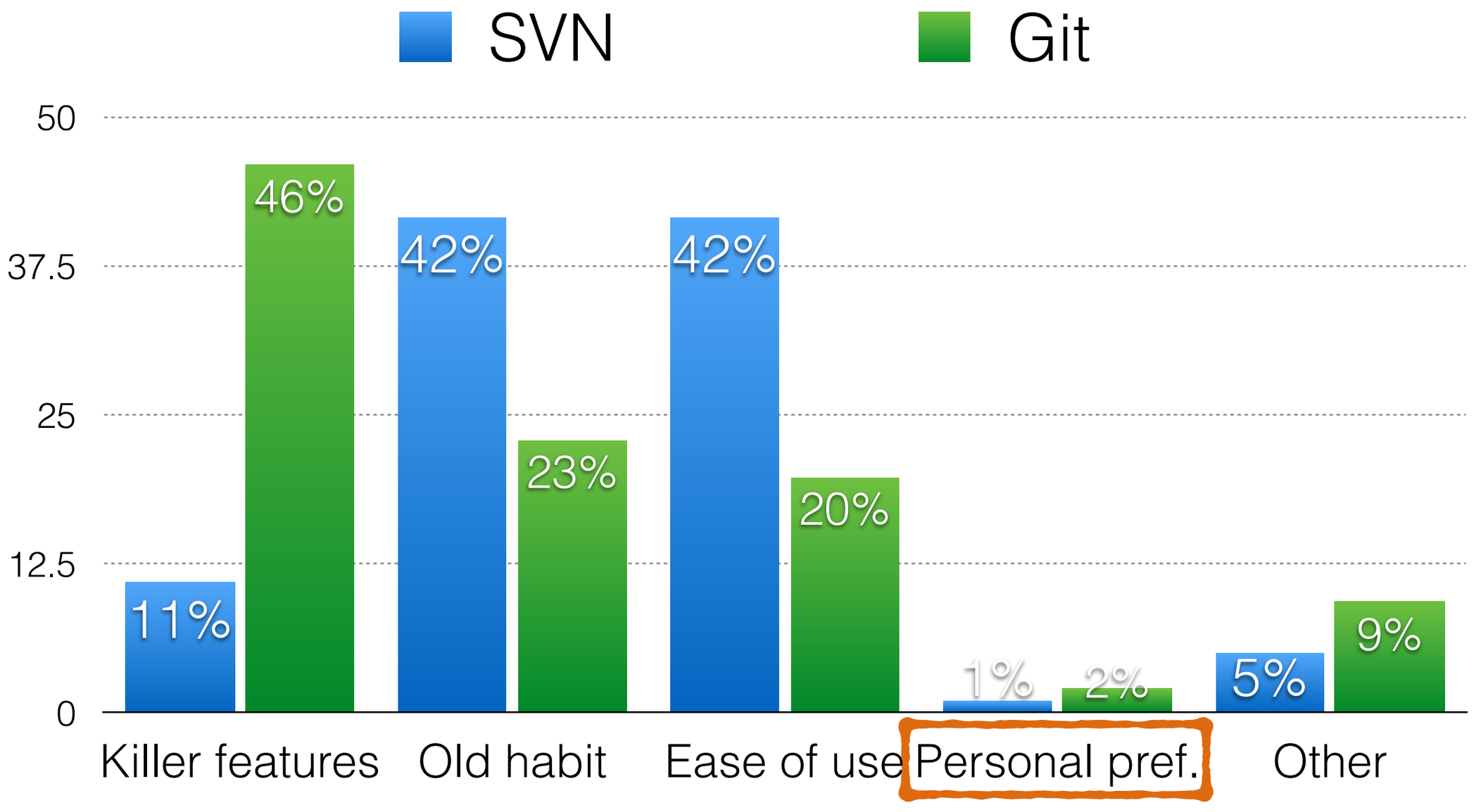
RQ3: Why do developers prefer one VCS over another?



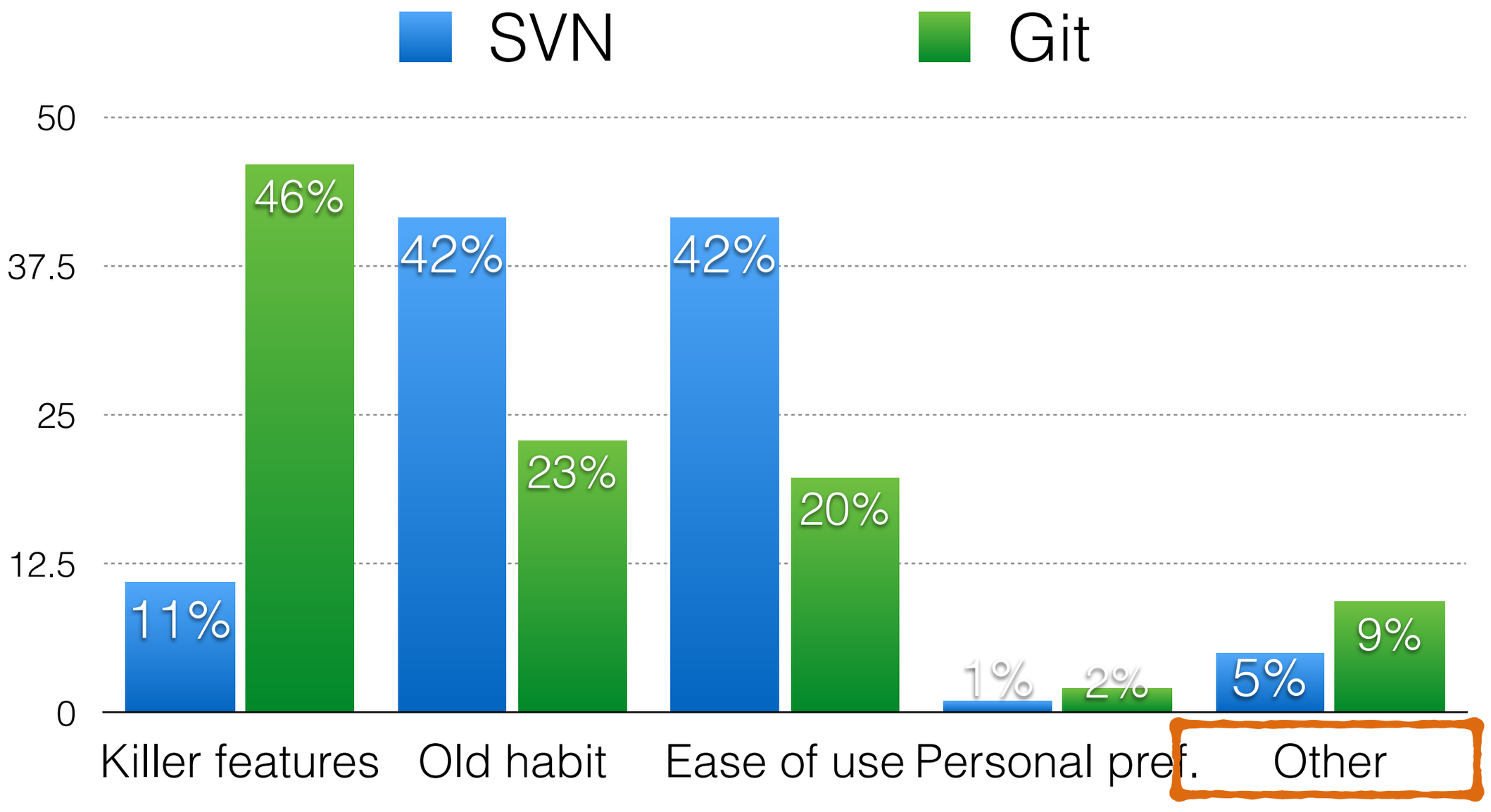
RQ3: Why do developers prefer one VCS over another?



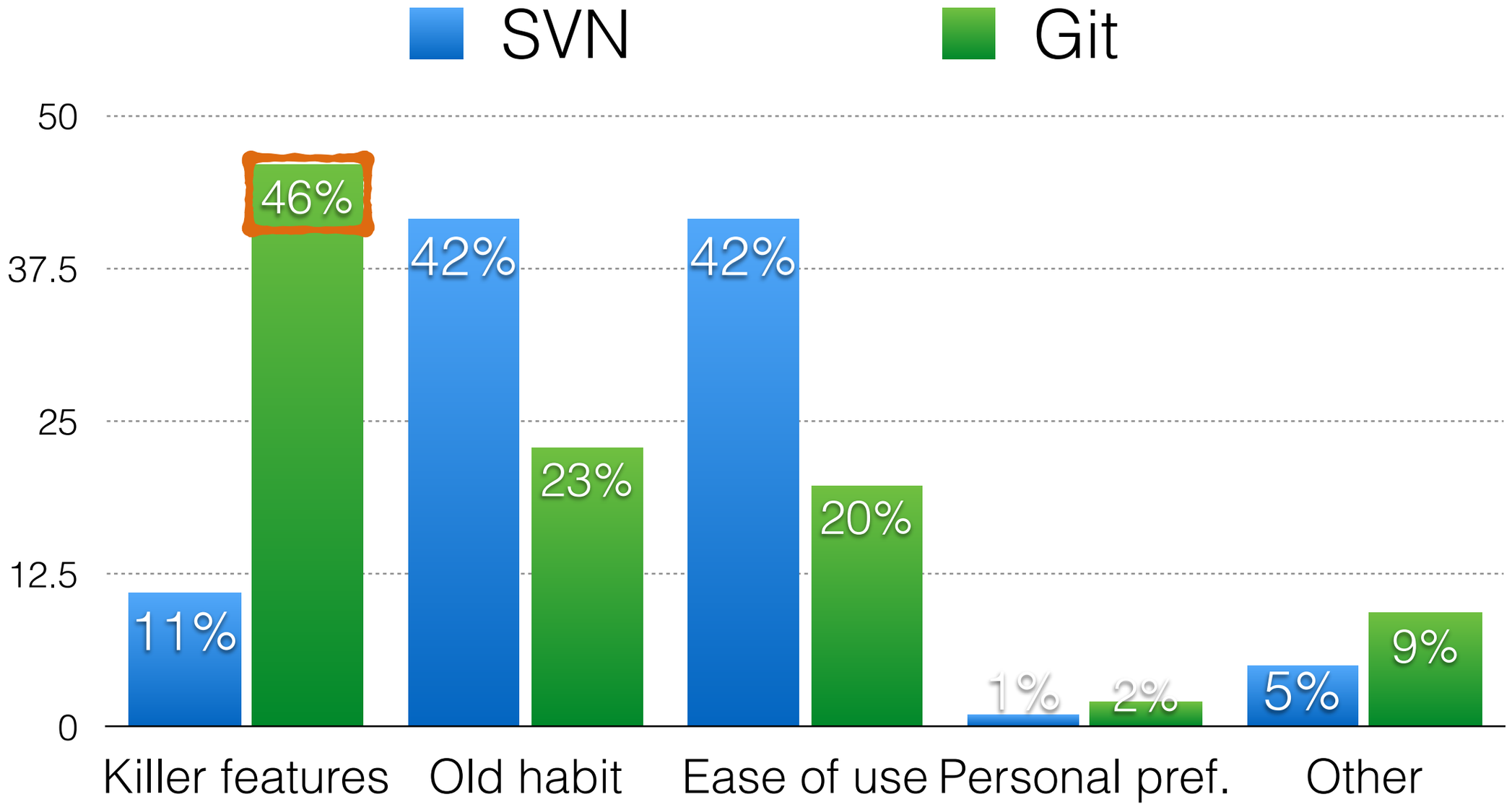
RQ3: Why do developers prefer one VCS over another?



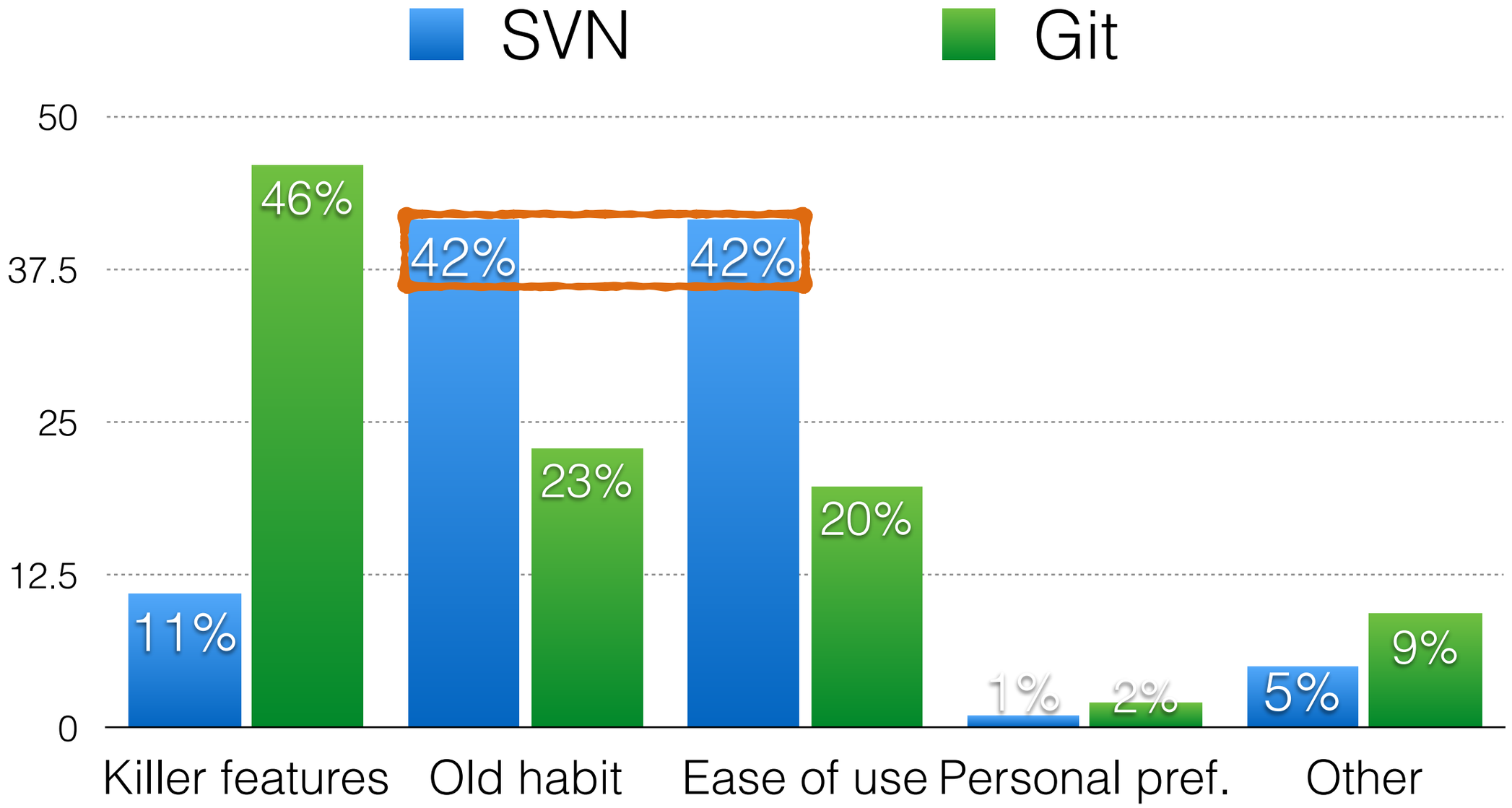
RQ3: Why do developers prefer one VCS over another?



RQ3: Why do developers prefer one VCS over another?



RQ3: Why do developers prefer one VCS over another?



RQ3: Why do developers prefer one VCS over another?

"You get to commit to a local repository and make your changes public only when they are ready"

RQ3: Why do developers prefer one VCS over another?

“You get to commit to a local repository and make your changes public only when they are ready”

“I found the commit process very straightforward […]”

RQ3: Why do developers prefer one VCS over another?

Git is preferred because of its "killer features"

SVN is preferred because of it's easier to use and because of familiarity

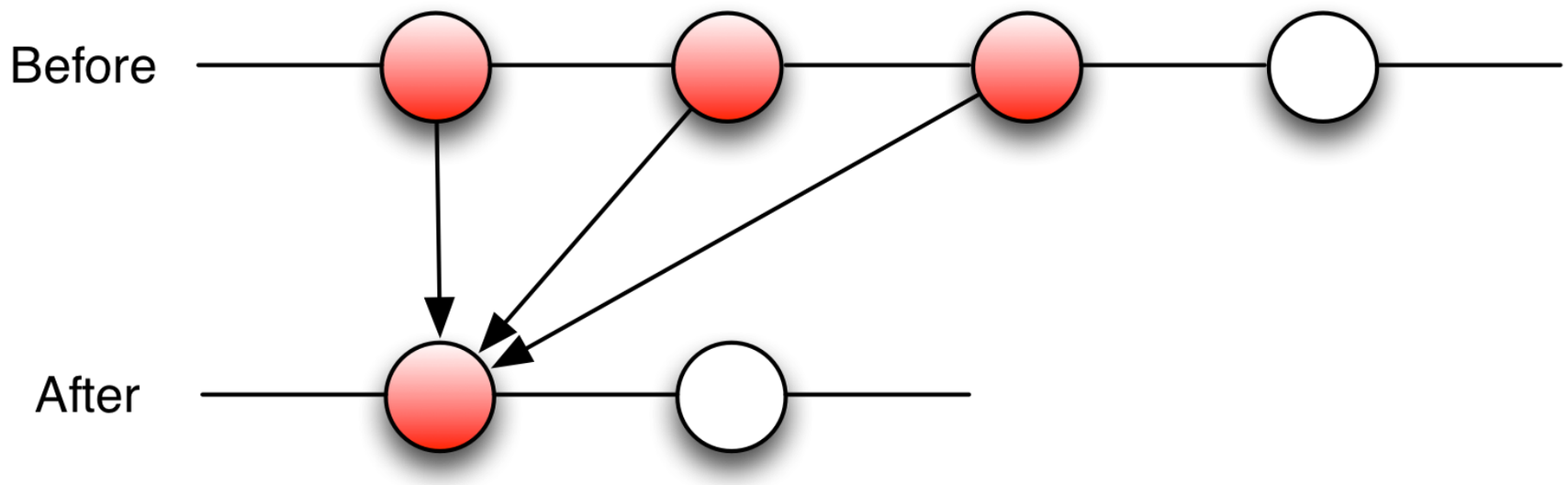
Implications

Tool builders should focus on features that complement the **developer's workflow**. Ⓣ

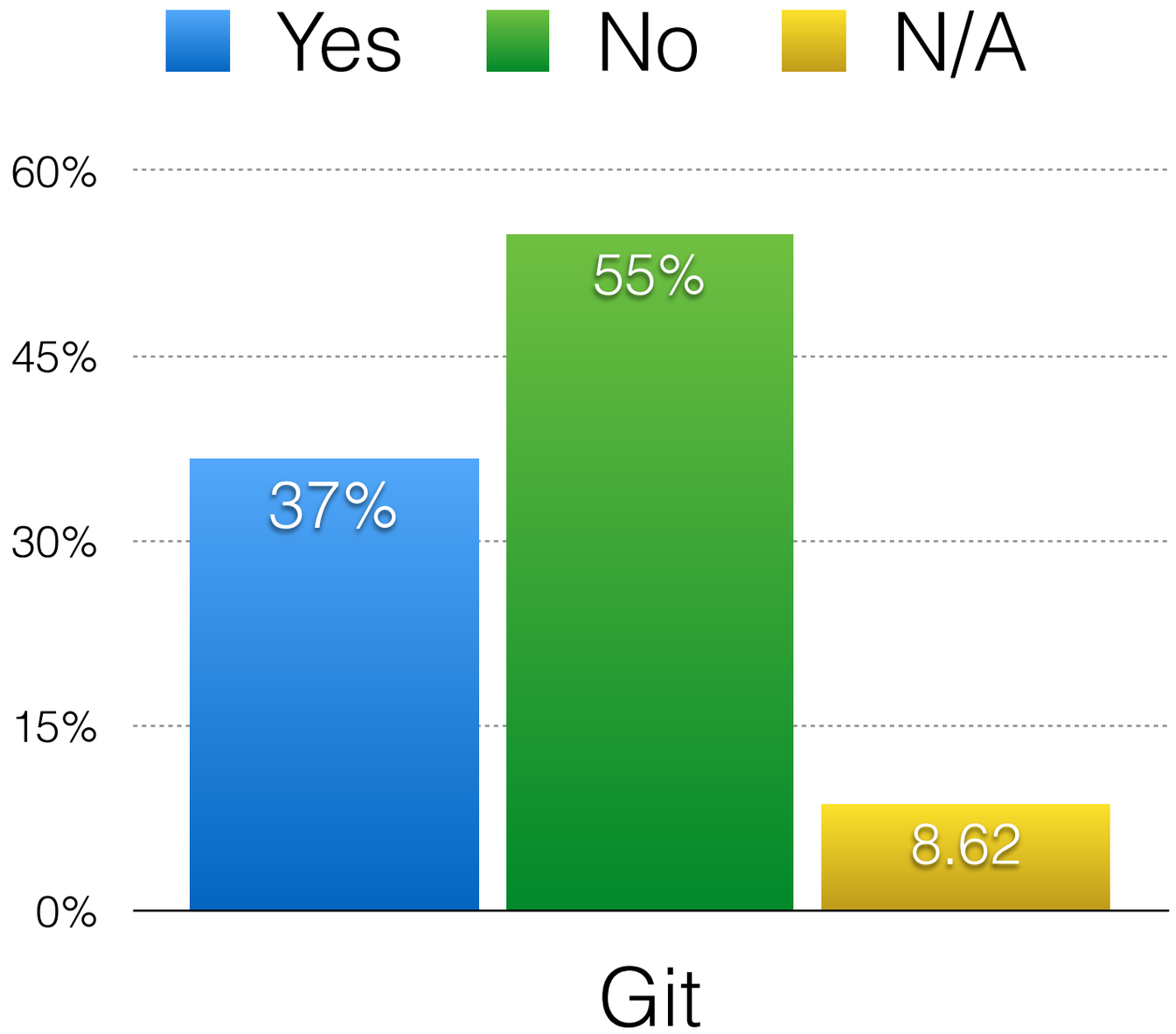
While Git has a **steep learning curve**, it does allow for **better ways to manage your changes**. Ⓓ

RQ4: Do developers squash their commits

What is squashing?



RQ4: Do developers squash their commits



RQ4: Do developers squash their commits

Why do they do it?

Developers using Git mention two different reasons

(a) grouping several changes together

(b) they only care about the final solution, not the path they took to get there

RQ4: Do developers squash their commits

Over 1/3 of developers squash their commits

Large teams squash commits more often then small ones

Implications

Git allows users to **change history** before they make it public or available to others. ®

Tool builder could allow for **non-destructive history** modifications, e.g.: **hierarchical commits** Ⓣ

Threats

Squashing

Age bias

OSS vs. Proprietary software

Conclusions

The commit size is smaller in Git than SVN.

Developers split their changes more often in Git, using a finer granularity.

1/3 of developers use squashing to change the history.

Teams of all sizes predominantly prefer Git (71%)

cope.eecs.oregonstate.edu/VCStudy