The Promises and Perils of Mining Git

Topic 1 – Presented by Soumaia Bouhouia

Table of contents

01

Overview of the Paper

Short description of the paper and its relevance to the course material.

02

SCM systems

Their use in research.

03

CSCM systems

What are they, SVN and its usage today compared to Git.

04

DSCM systems

What they are, and pros and cons in regards to research.

Table of contents

05

SVN VS. Git

Promises and perils of Git and how it compares to SVN.

06

Authors' Inquiries

Questions raised by the authors of the paper.

07

Further analysis

Some of the questions were addressed in a 2014 analysis.

08

Bibliography

Links to the papers used for this presentation.

01

Overview of the Paper

Short description of the paper and its relevance to the course material.

Overview

- ***The authors:***
  - Christian Bird[*], Peter C. Rigby[†], Earl T. Barr[*], David J. Hamilton[*], Daniel M. German[†], Prem Devanbu[*].
- ***Date of publication:***
  - 05 June 2009
- ***Goal of the paper:***
  - Investigating whether the repositories created using Git would be as good as centralized options for gathering data and performing analyses.

(*: University of California, †: University of Victoria)

02

SCM systems

Their use in research.

Utilizing SCM Systems in Research

1. Reconstruction of the software creation process
2. Creation of recommender systems
3. Study of evolution patterns
4. Bug predictions
5. Exploring collaborative processes

03

CSCM systems

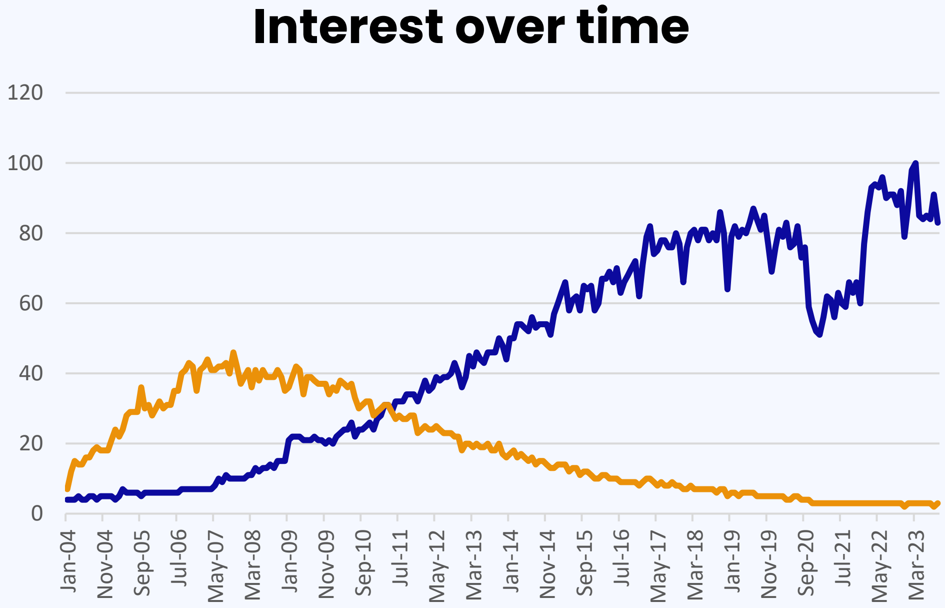
What they are, SVN, and its usage today compared to Git.

What is a CSCM system?

- Stands for **Centralized Source Code Management** system.
- Developers connect to a **central** source/server to access the repository.
- Changing a file leads to **only** the difference (**delta**) being stored.
- Complicated to retrieve previous versions of the code if the code on the centralized server becomes corrupted. [3]
- **SVN** (Apache Subversion) is one such system.

Git vs. SVN: A 20-Year Comparison

Generated using Google Trends!



● **Git**

Starts picking up traction in 2007.

● **SVN**

Peaks in mid-2007.

> "Numbers represent search interest relative to the highest point on the chart for the given region and time."

04

DSCM systems

What they are, and pros and cons regarding research.

What is a DSCM system?

- Stands for **Decentralized Source Code Management** system.

- Enables:
  - Working **independently** on **local** repository copies.
  - Offline work while **retaining access** to the **complete** project **history**.
  - Creating and merging **branches** at **minimal cost**.
  - Developers to commit individual changed lines within a file [2].

- **Git** is one such system.

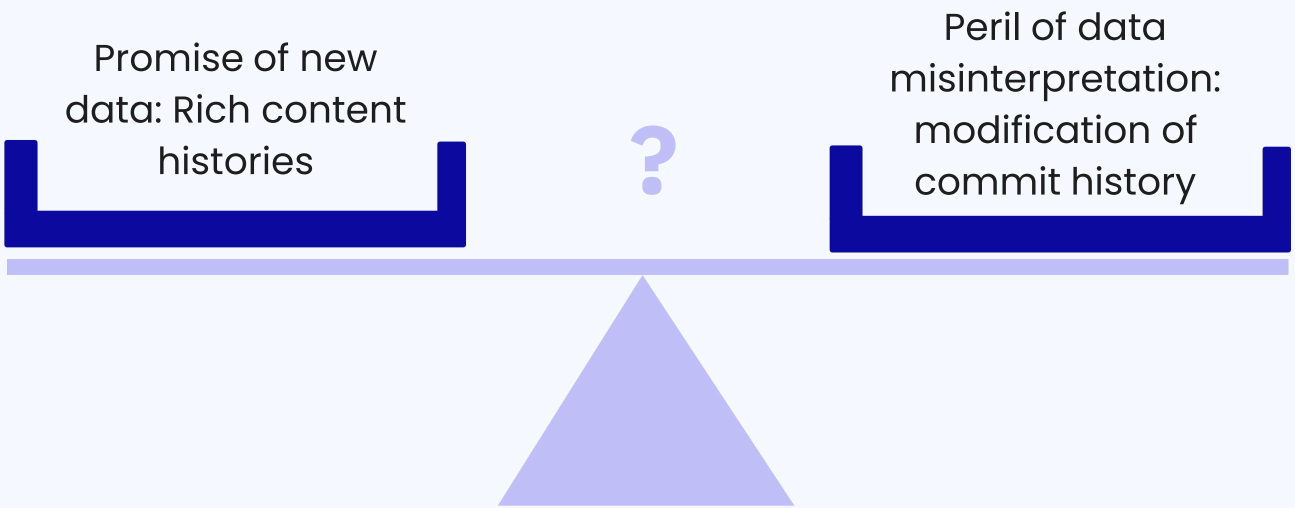
Growing Adoption of DSCM systems

- **Increasing** adoption of **DSCM** systems in software projects.

- DSCM data provides **valuable** insights for research. However, it also poses **conceptual** and **practical challenges** [1].

- The paper's authors conducted a comparative analysis between SVN and Git, where SVN represents CSCM systems, while Git represents DSCM systems.

Promises and Perils



Promise of new data: Rich content histories

?

Peril of data misinterpretation: modification of commit history

05

SVN VS. Git

Promises and Perils of Git and how it compares to SVN.

Promises and Perils (1/4)

1

Possible to recover more data than what makes it in the stable codebase.

1

Git's nomenclature is different from CSCMs.

2

Automatic creation of implicit branches.

2

Easier to recover the history of a project.

Promises and Perils (2/4)

3 Need to use different analysis methods since Git uses DAGs instead of a mainline.

4 Git history can be rewritten.

5 Cannot always determine what branch a commit was made on.

6 Difficult to track where and if the merge occurred.

Promises and Perils (3/4)

3

Git records the information needed to correct perils 3 to 6 in private logs.

7

The accessible data may only contain selected commits.

4

The signed-off-by and other attributes create a “paper trail.”

5

Git records info explicitly about the contributors that are not part of the core set of devs.

Promises and Perils (4/4)

6

All metadata, including history, is local.

7

Tracks content of the files, including history of lines as they are moved or copied.

8

Git is faster and uses less space than SCMs.

9

Most SCMs can be converted to Git with their history intact.

Testing some Promises and Perils

4

“Paper Trail”

They were able to construct a signed-off-by network, which demonstrates the role of a community member and how big their role is in the project.

6

Merge Source

They were able to detect the source of the merge 97.9% of the time.

5

Author info

They examined the data from the project Ruby on Rails which switched to Git in 2008. Jump in the number of authors at that point.

06

Authors’ Inquiries

Questions raised by the authors of the paper.

Research Inquiries

- Does changing from a centralized to a distributed SCM system affect how the project team communicates or develops the project?
- Does adopting a DSCM system encourage more focused development while possibly diminishing awareness of the broader project?
- Do developer teams sometimes work together separately from the main repository for extended periods?

07

Further analysis

A portion of the questions raised have been addressed in an analysis conducted in 2014.

Question and answer

Question

Does changing from a centralized to a distributed SCM system affect team communication or project development?

Answer

DSCM systems like Git often facilitate smaller, more frequent commits due to the possibility of making fine-grained change selection and lack of conflict fears with local repos. Yet, projects transitioning from SVN to Git maintained commit size and frequency, possibly retaining previous CSCM commit policies [2].

Interesting observations

Observation 1

Developers who answered that they used CSCM systems said that they found this type of workflow easier mainly because they are used to it, not because of its features.

Observation 2

As the team size gets bigger, the size of the commits might slightly decrease, but there's no strong evidence to prove this.

Expectation

Large teams would perform smaller commits to better express changes.

08
Bibliography

1. Bird, Christian, et al. “The Promises and Perils of Mining Git .” *IEEE Xplore*, 5 June 2009, ieeexplore.ieee.org/abstract/document/5069475.

2. Brindescu, Caius, et al. “How Do Centralized and Distributed Version Control Systems Impact Software Changes?” ACM Conferences, 1 May 2014, dl.acm.org/doi/10.1145/2568225.2568322.

3. Zolkifli, Nazatul Nurlisa, et al. “Version Control System: A Review.” *Procedia Computer Science*, Elsevier, 29 Aug. 2018, www.sciencedirect.com/science/article/pii/S1877050918314819.