Proactive Detection of Collaboration Conflicts

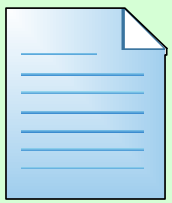
Yuriy Brun
Reid Holmes
Michael D. Ernst
David Notkin

University of Washington
University of Waterloo

Have you ever made a mistake while programming and only realized it later?

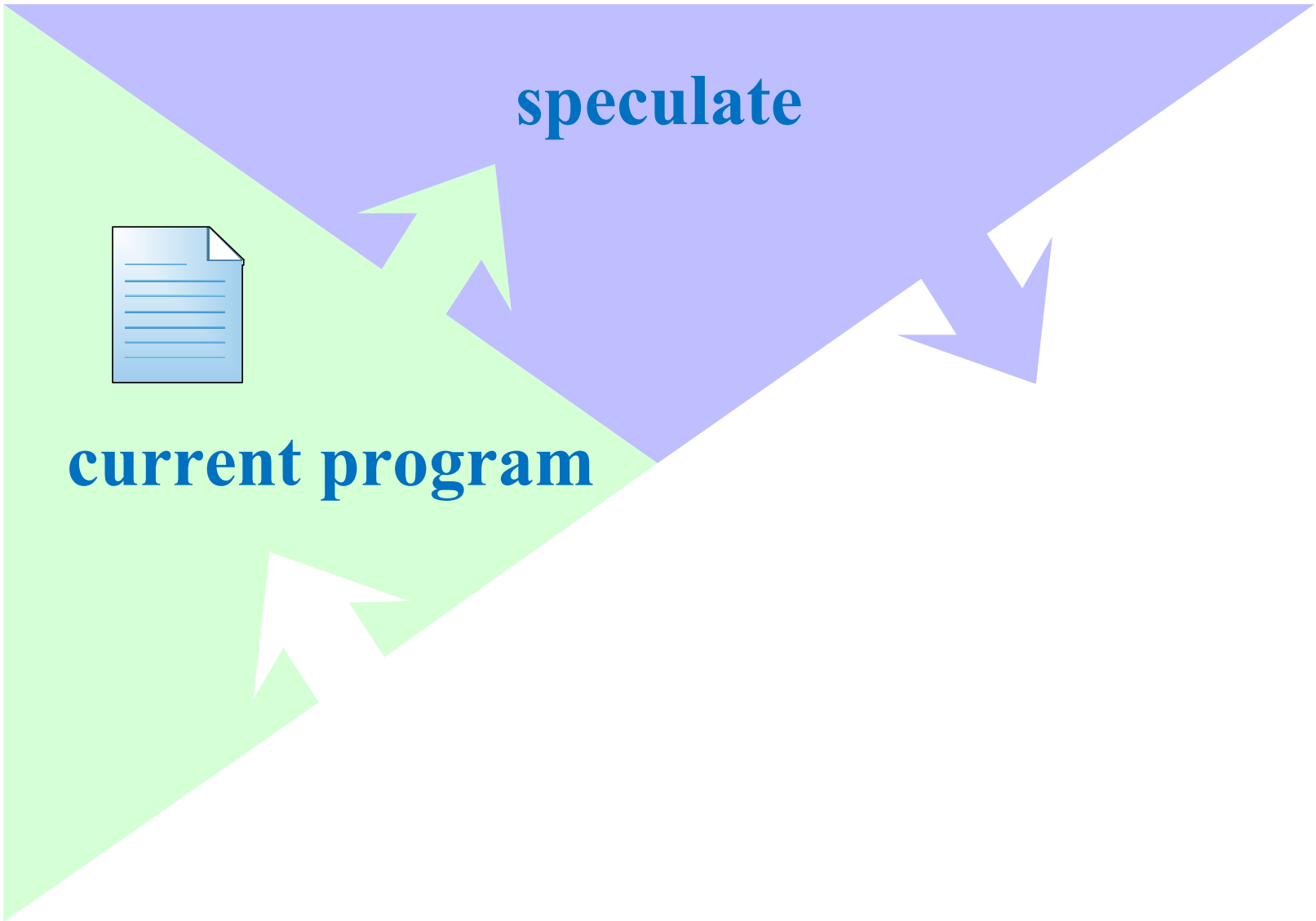
- design decision
- refactoring
- repeated someone else's work

Speculative analysis: Predict the future and analyze it

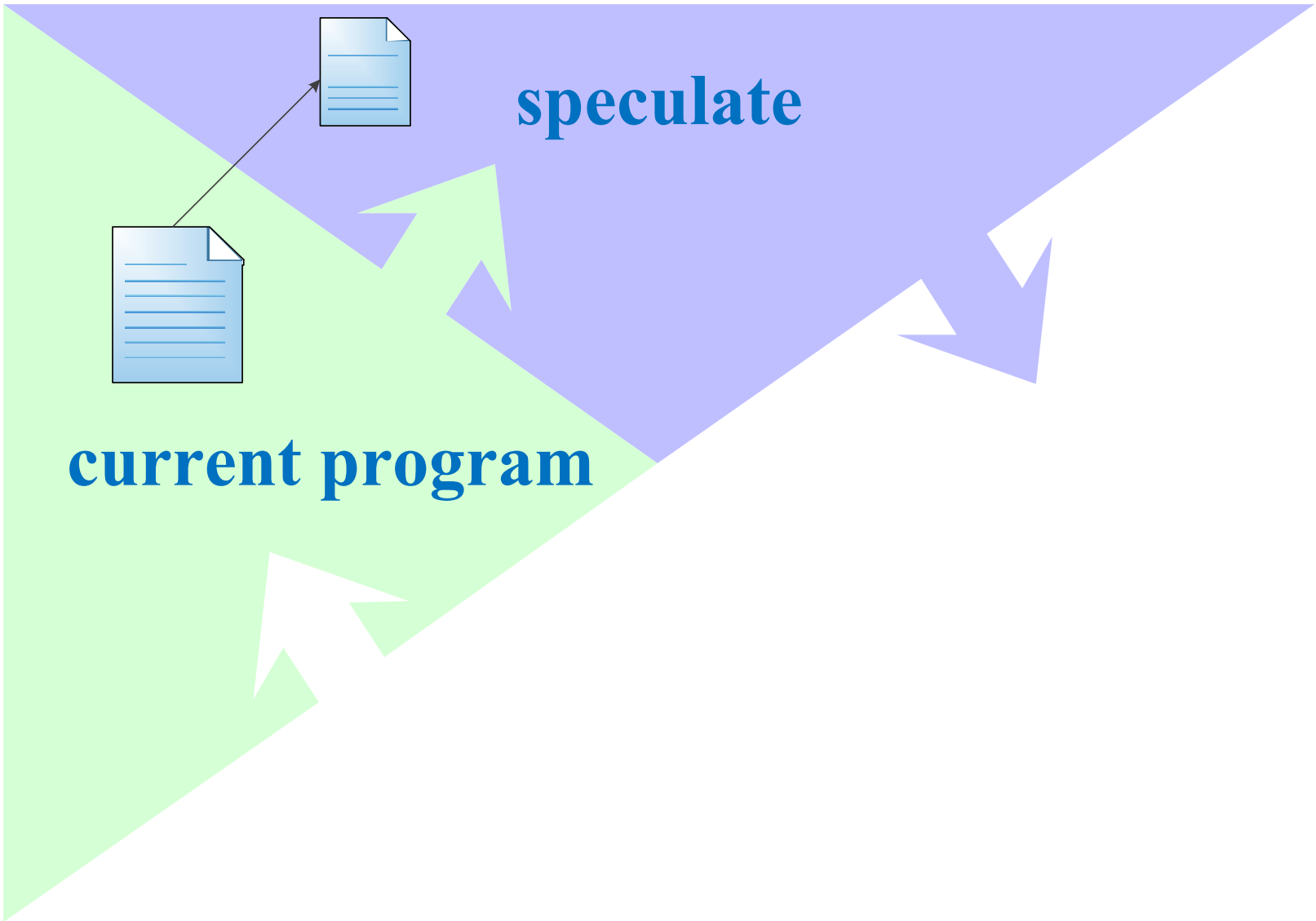


current program

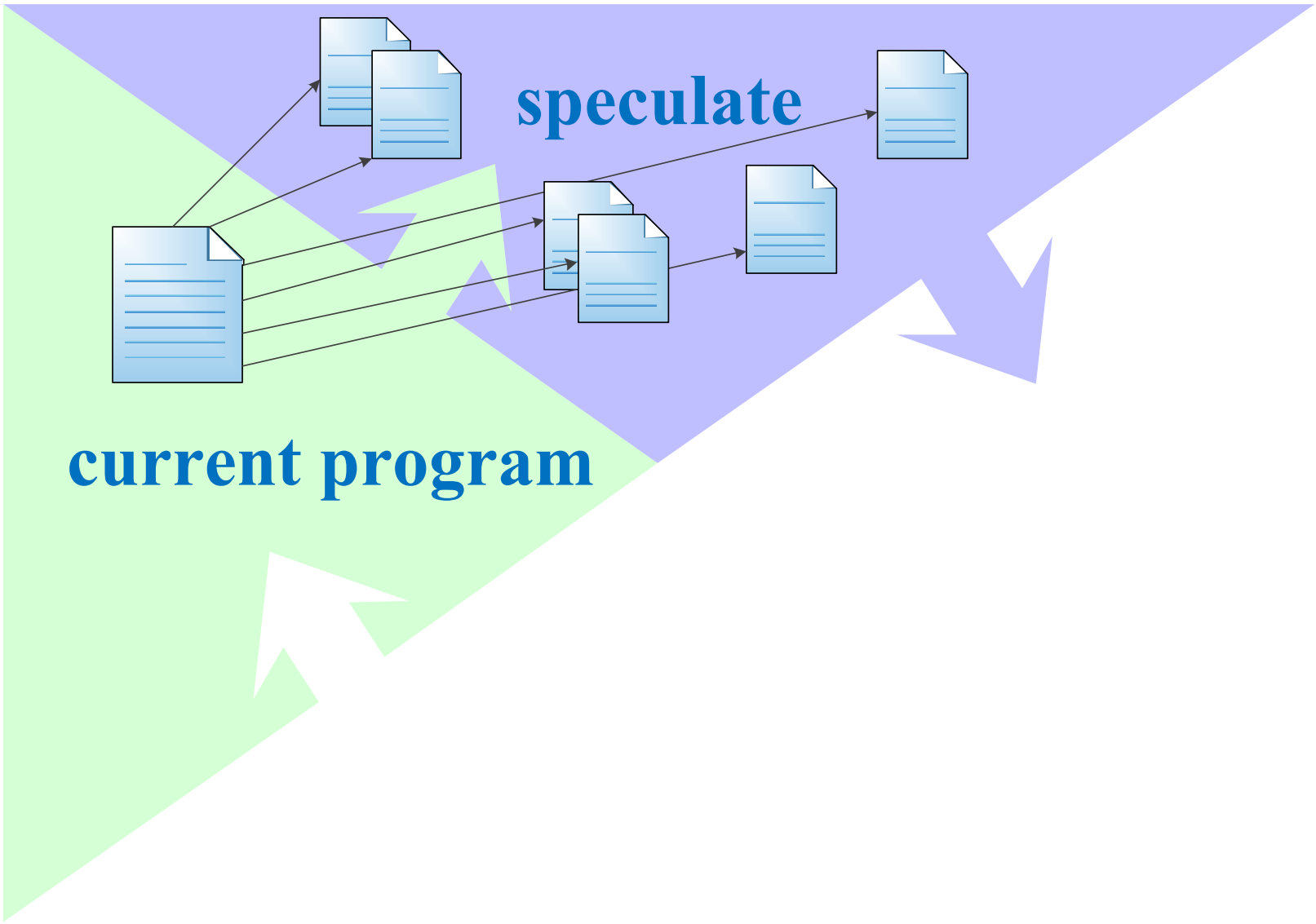
Speculative analysis: Predict the future and analyze it



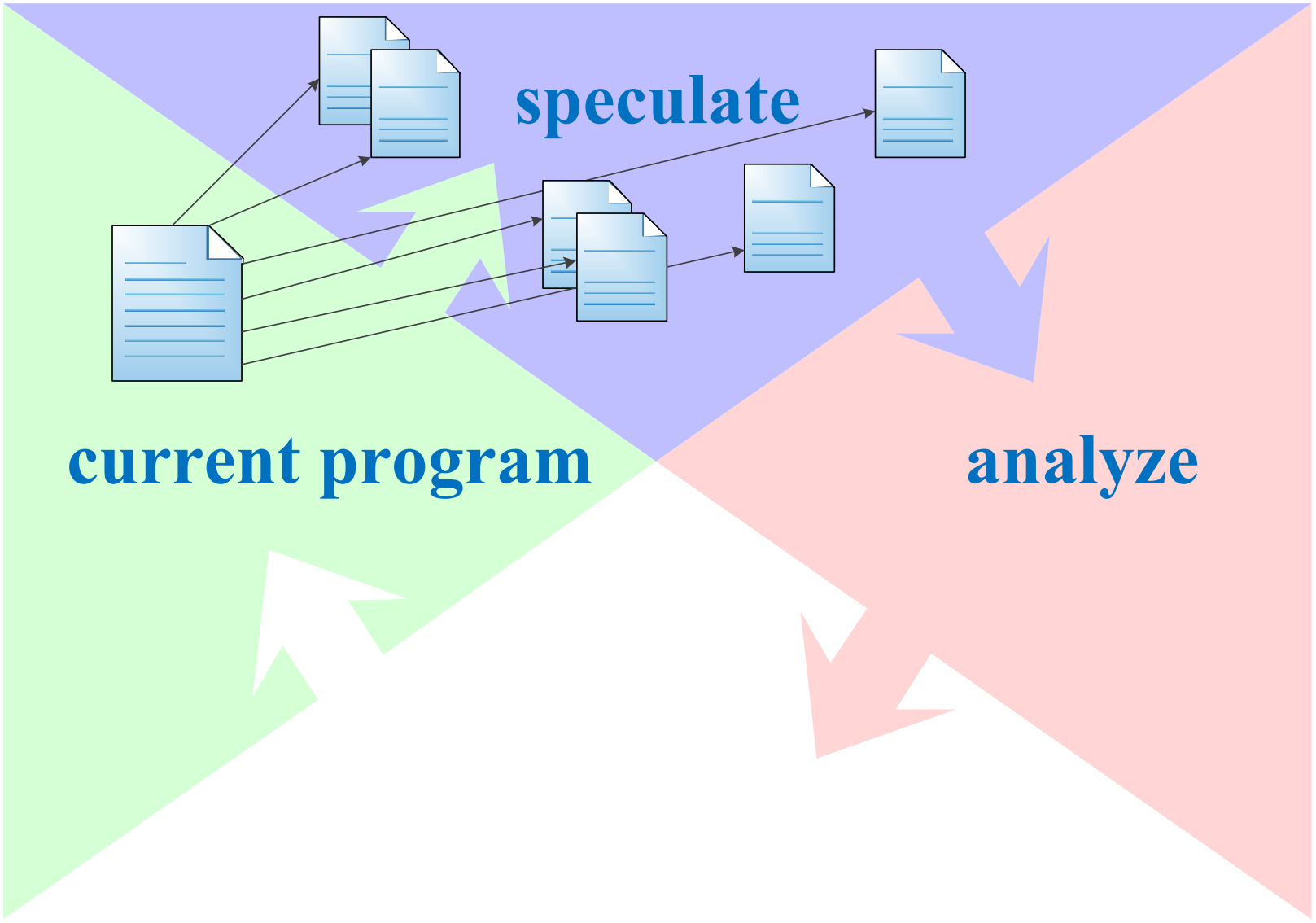
Speculative analysis: Predict the future and analyze it



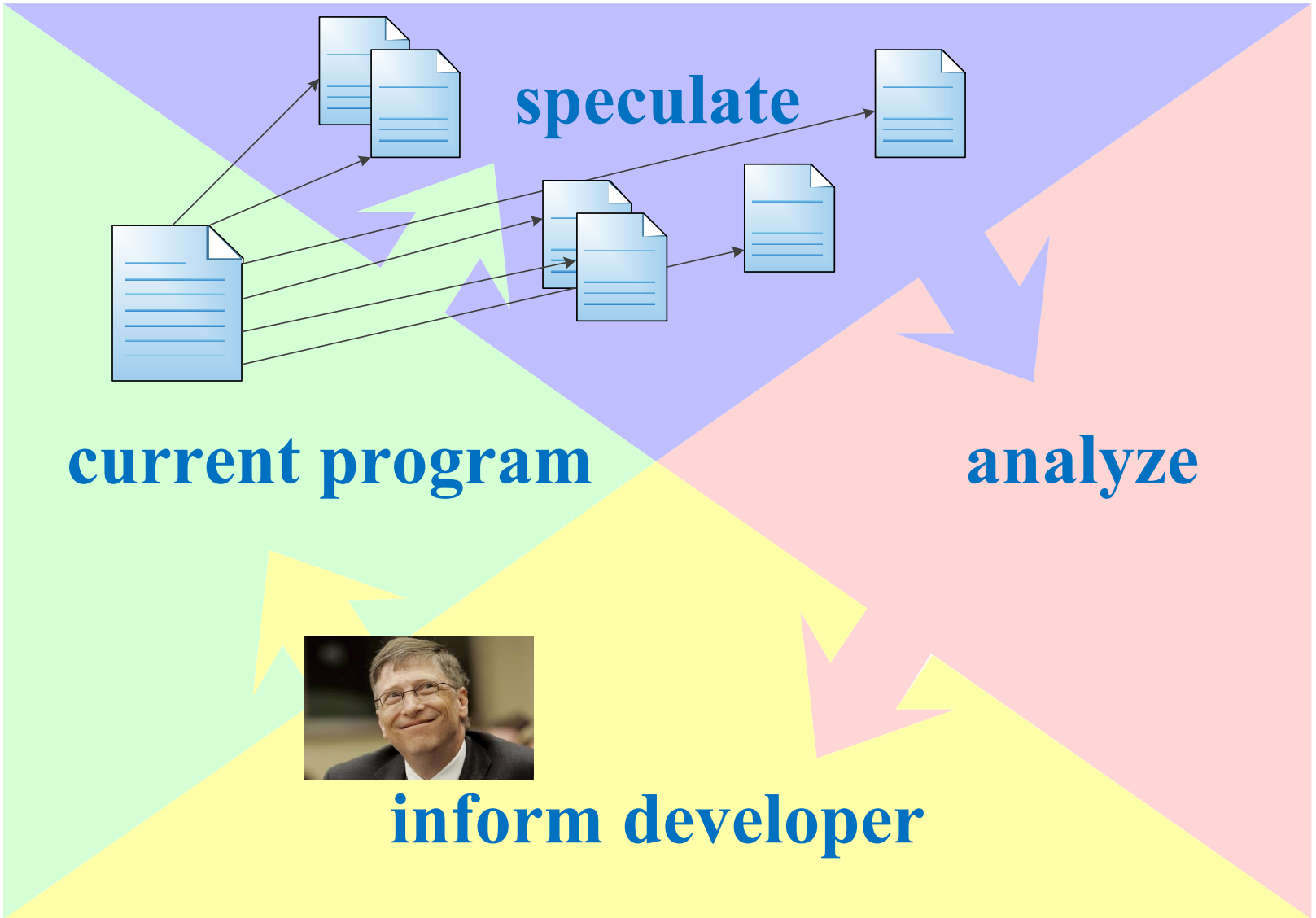
Speculative analysis: Predict the future and analyze it



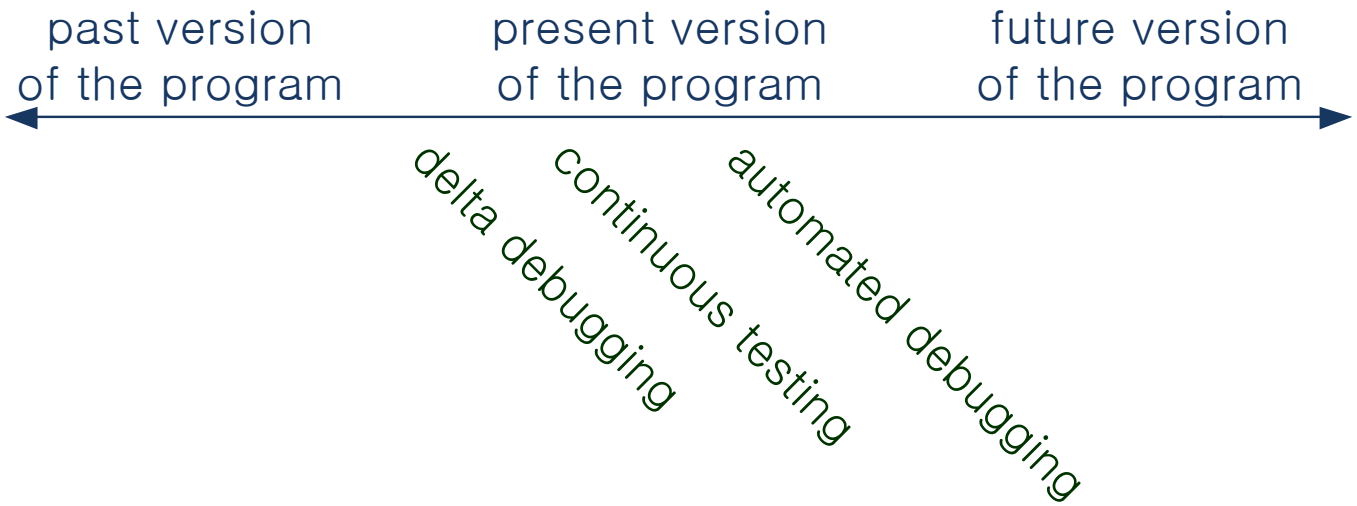
Speculative analysis: Predict the future and analyze it



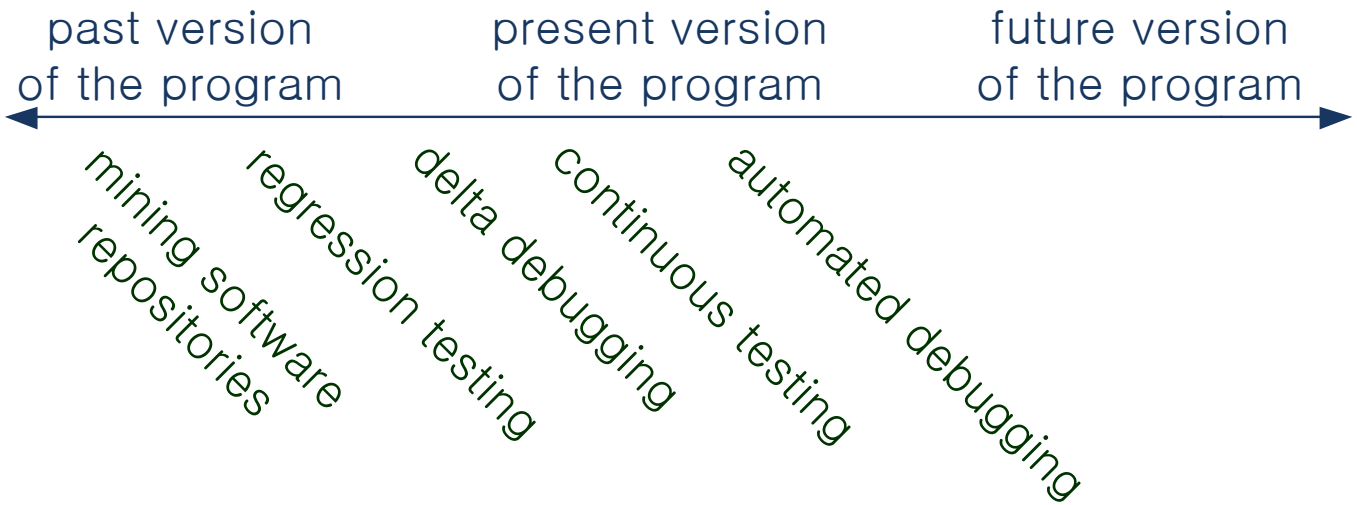
Speculative analysis: Predict the future and analyze it



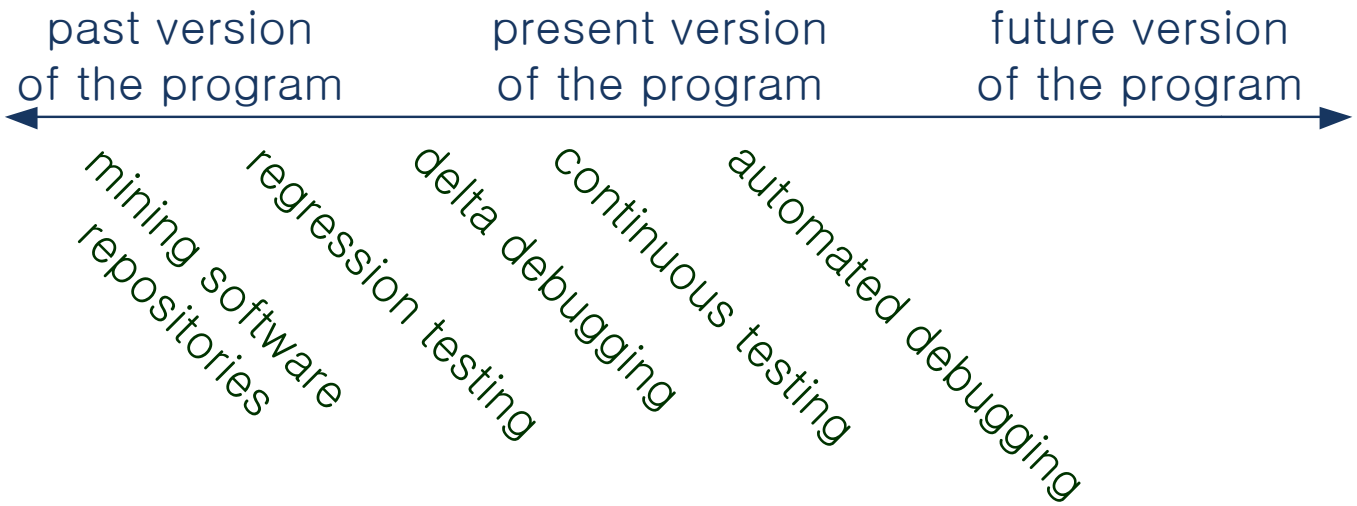
Exploring the future



Exploring the future



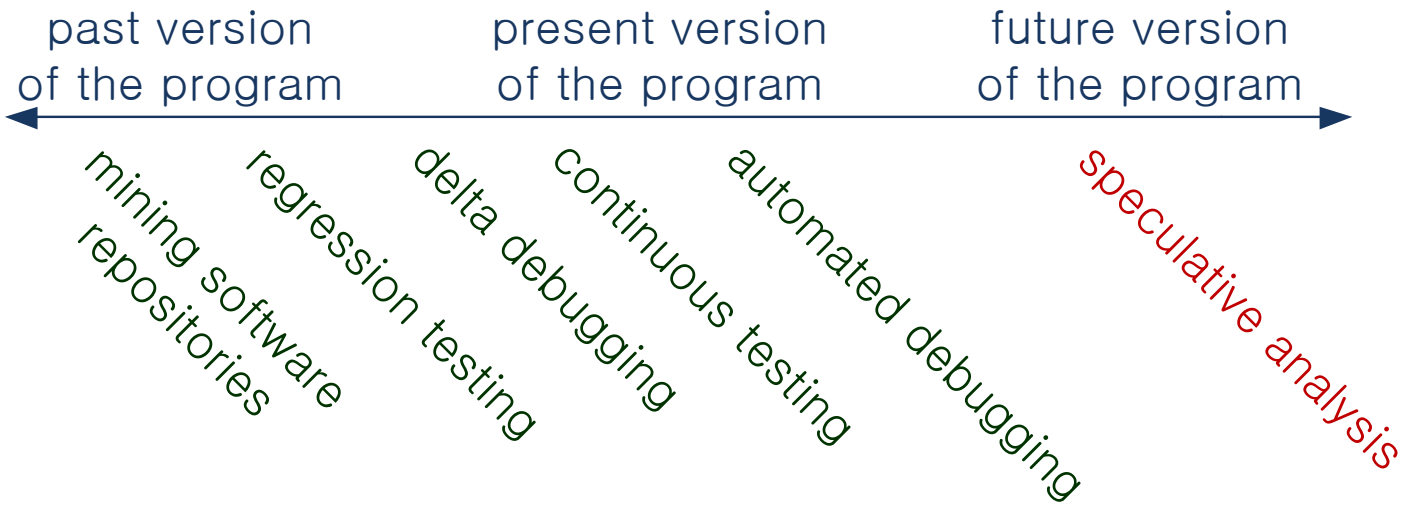
Exploring the future



Continuous development

- execution [Henderson and Weiser 1985; Karinthi and Weiser 1987]
- compilation [Childers et al. 2003; Eclipse]
- testing [Saff and Ernst 2003, 2004]
- version control integration [Guimarães and Rito-Silva 2010]

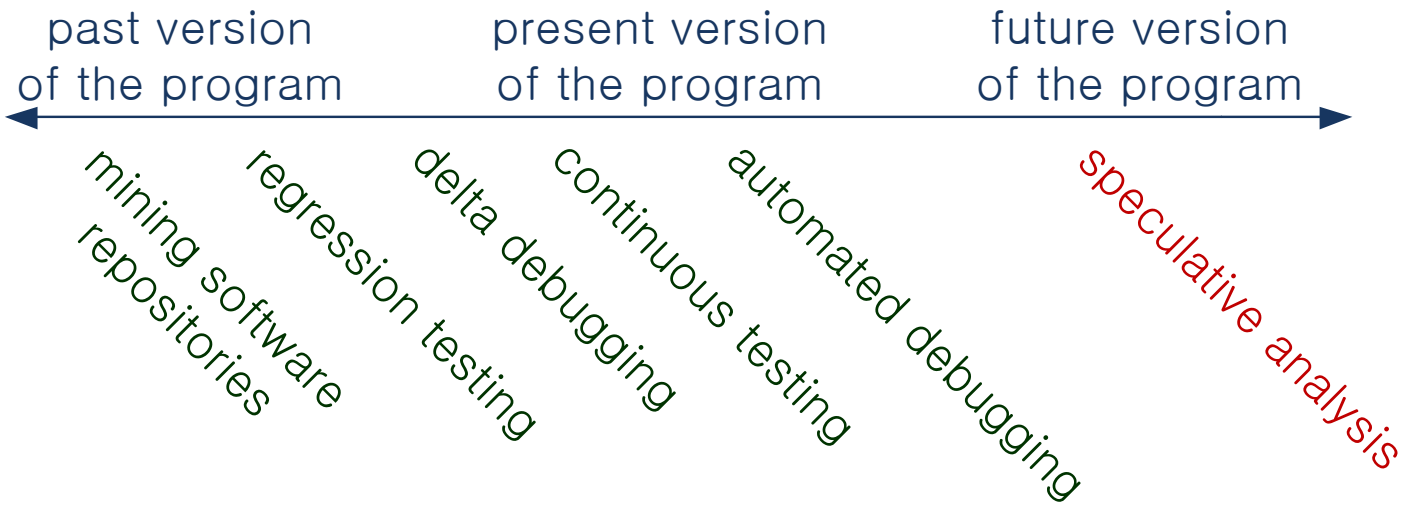
Exploring the future



Continuous development

- execution [Henderson and Weiser 1985; Karinthi and Weiser 1987]
- compilation [Childers et al. 2003; Eclipse]
- testing [Saff and Ernst 2003, 2004]
- version control integration [Guimarães and Rito-Silva 2010]

Exploring the future



Continuous development

- execution [Henderson and Weiser 1985; Karinthi and Weiser 1987]
- compilation [Childers et al. 2003; Eclipse]
- testing [Saff and Ernst 2003, 2004]
- version control integration [Guimarães and Rito-Silva 2010]

Speculative analysis is **predictive**.

Contributions

- Speculative analysis
- Speculative analysis for collaborative development
  Crystal: prototype tool
- Utility of speculative analysis for collaborative development

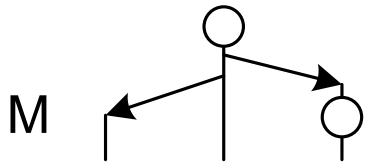
Version-control terminology

Proactive conflict detection applies to both centralized and decentralized version control.

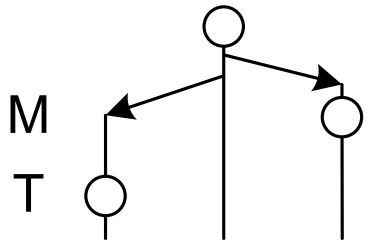
Terminology:

	decentralized	centralized
local commit:	commit	save
incorporate:	push and pull	commit and update

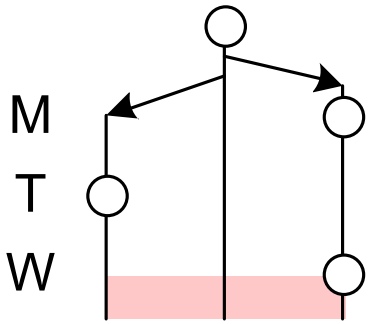
The Gates conflict



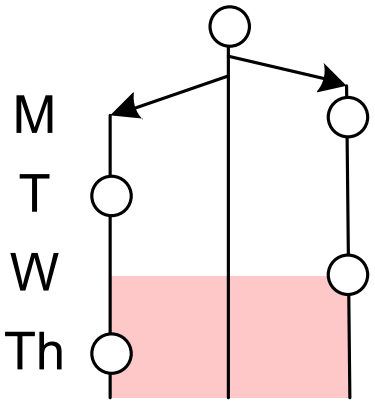
The Gates conflict



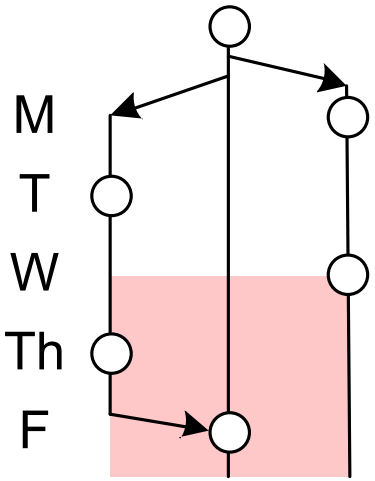
The Gates conflict



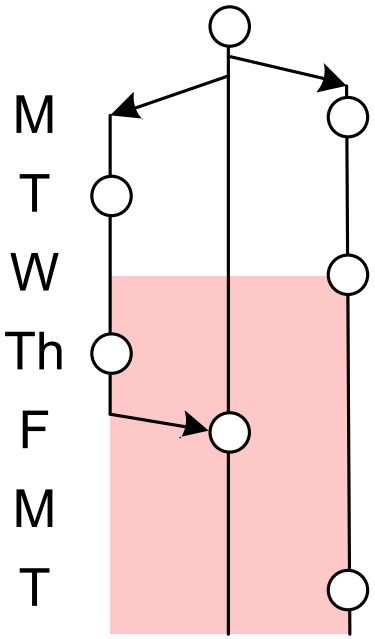
The Gates conflict



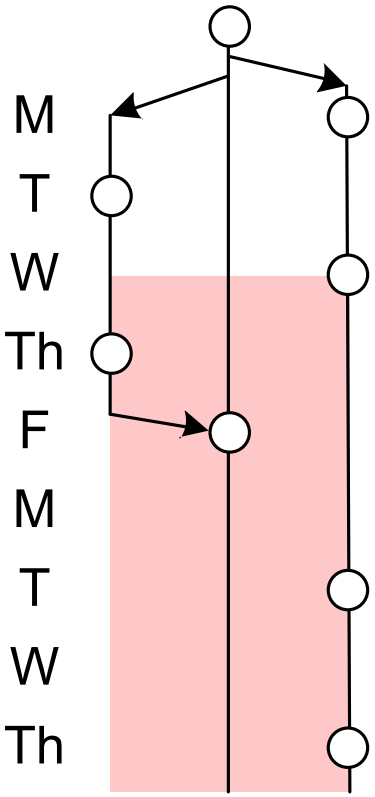
The Gates conflict



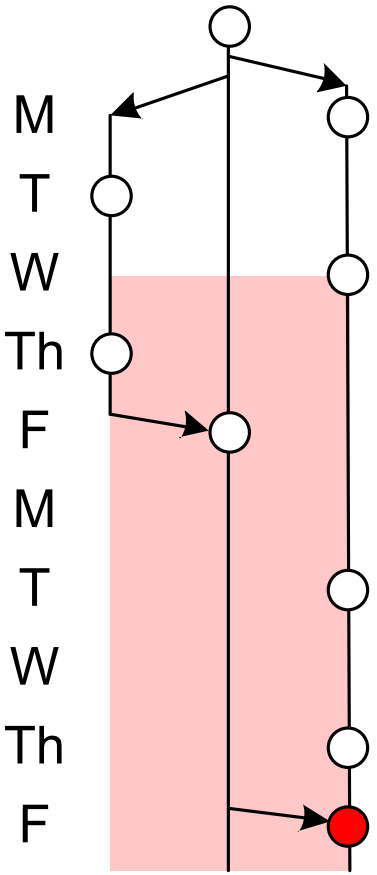
The Gates conflict



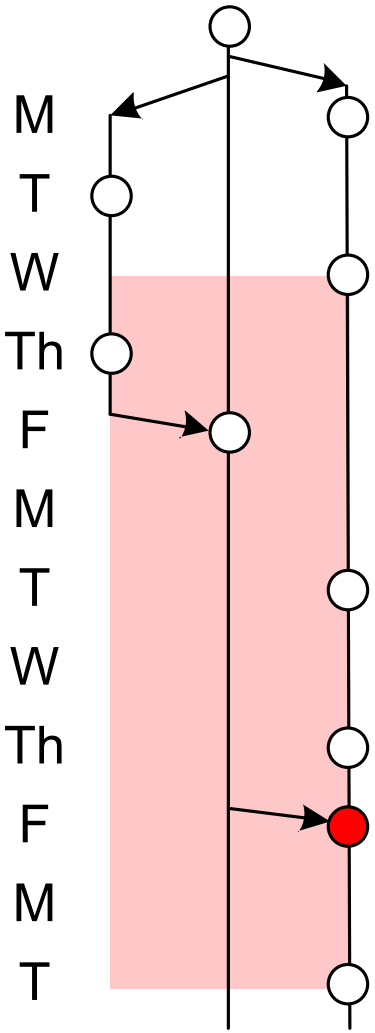
The Gates conflict



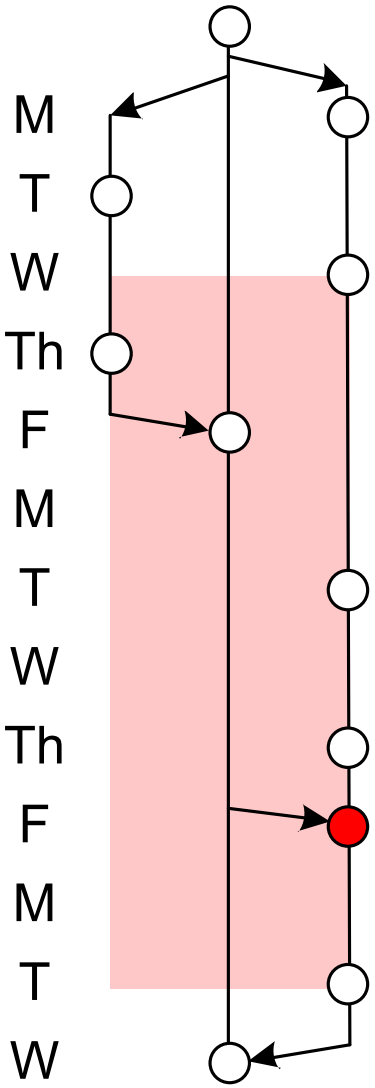
The Gates conflict



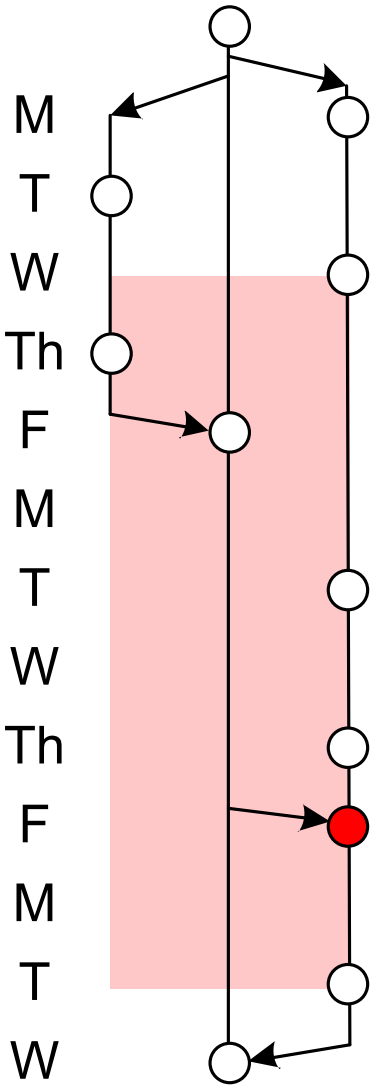
The Gates conflict



The Gates conflict

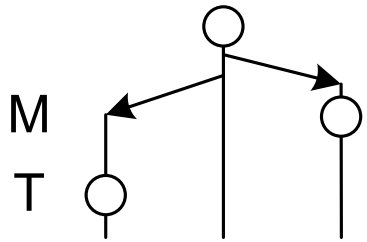


The Gates conflict



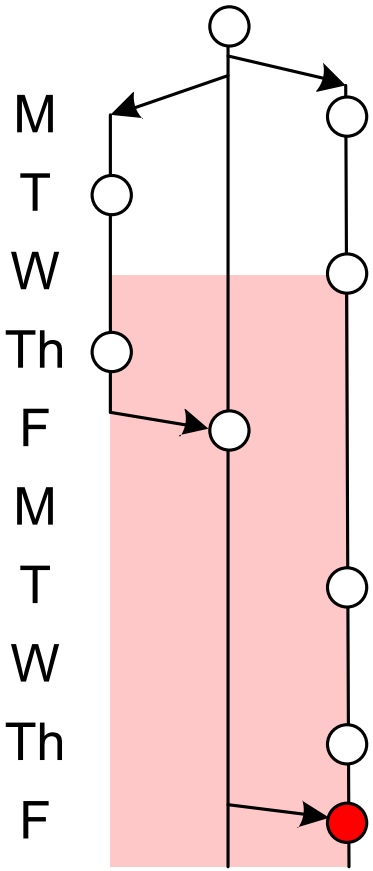
The information was all there, but the developers didn't know it.

What could well-informed developers do?



- Avoid conflicts

What could well-informed developers do?



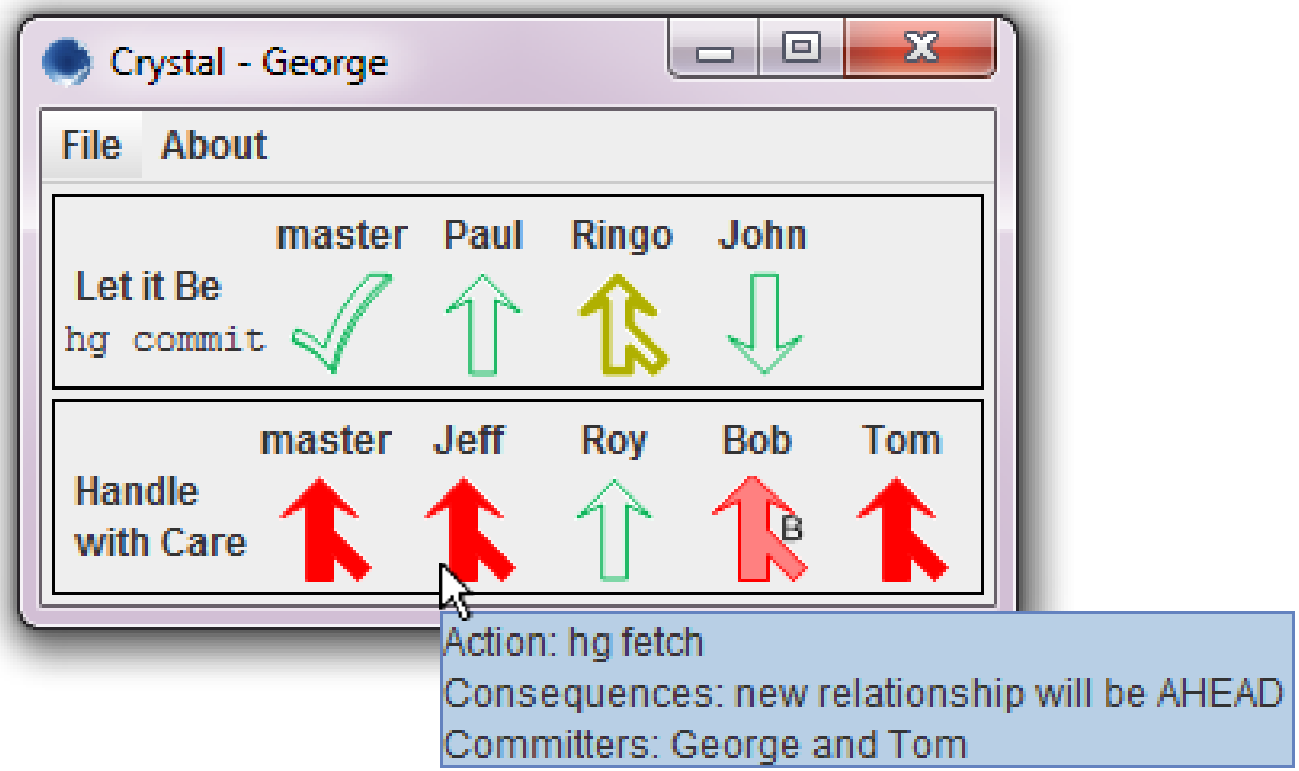
- Avoid conflicts
- Reduce conflict severity

Introducing Crystal: A proactive conflict detector

DEMO

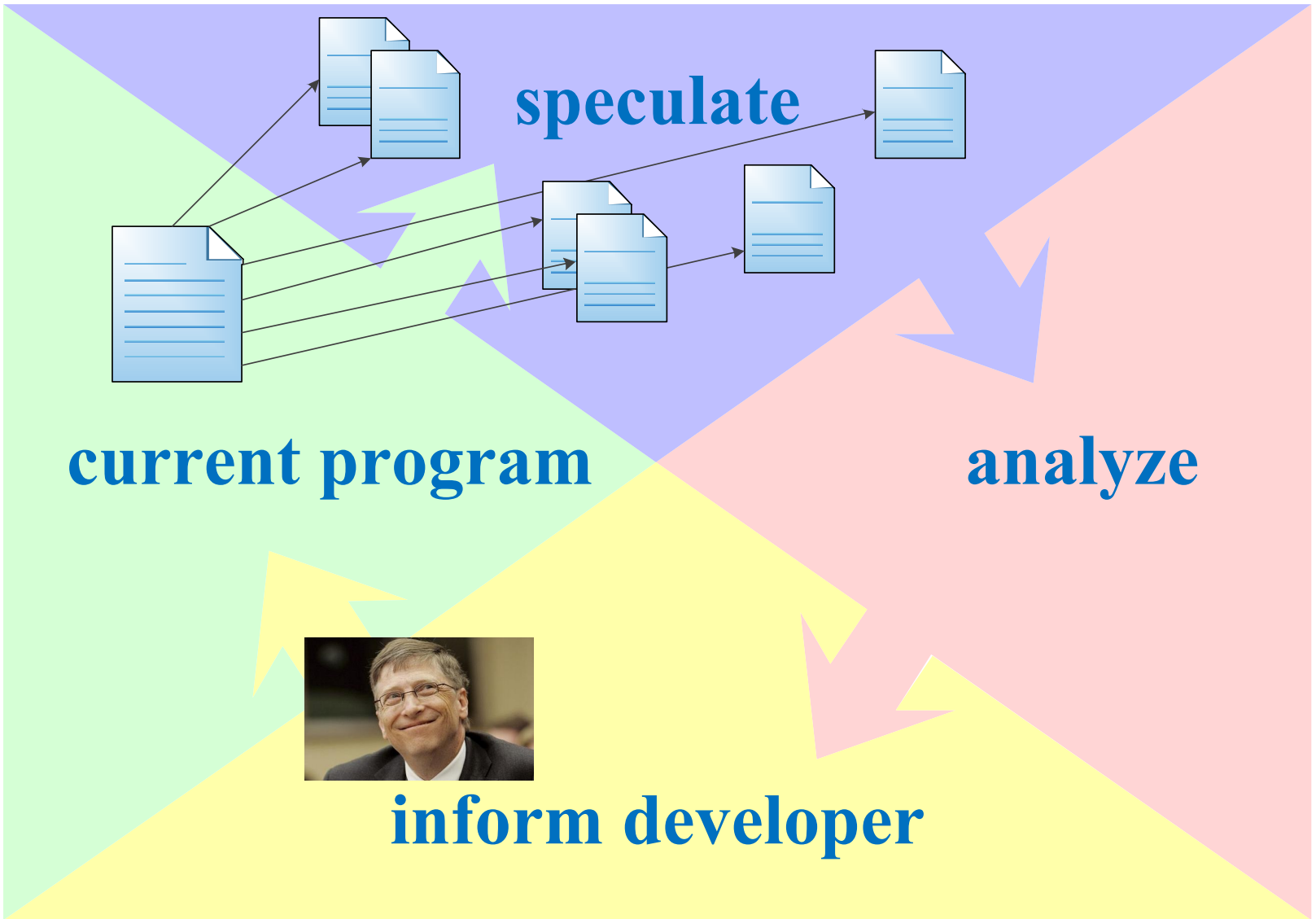
Introducing Crystal: A proactive conflict detector

DEMO



http://crystalvc.googlecode.com

Speculative analysis in collaborative development



Reducing false positives in conflict prediction

Collaborative awareness

- Palantír [Sarma et al. 2003]
- FASTDash [Biehl et al. 2007]
- Syde [Hattori and Lanza 2010]
- CollabVS [Dewan and Hegde 2007]
- Safe-commit [Wloka et al. 2009]
- SourceTree [Streeting 2010]

Reducing false positives in conflict prediction

Collaborative awareness

- Palantír [Sarma et al. 2003]
- FASTDash [Biehl et al. 2007]
- Syde [Hattori and Lanza 2010]
- CollabVS [Dewan and Hegde 2007]
- Safe-commit [Wloka et al. 2009]
- SourceTree [Streeting 2010]

Crystal analyzes **concrete artifacts**,
eliminating false positives and false negatives.

Utility of proactive collaborative conflict detection

- Are textual collaborative conflicts a real problem?
- How dangerous are safe merges?
- Do higher-order collaborative conflicts exist?

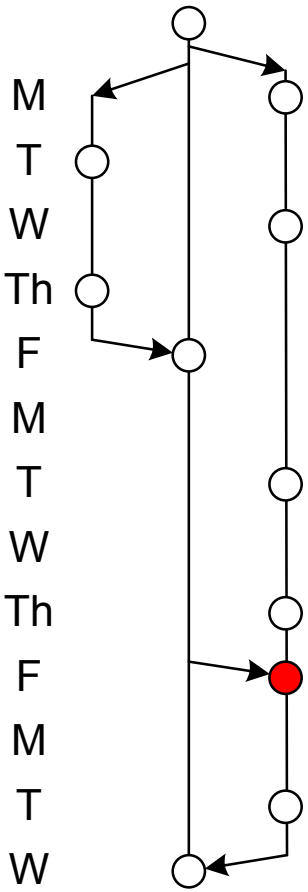
Are textual collaborative conflicts a real problem?

histories of 9 open-source projects:

size:	26K–1.4MSLoC
developers:	298
versions:	140,000

Perl5, Rails, Git, jQuery, Voldemort, MaNGOS, Gallery3, Samba, Insoshi

Are textual collaborative conflicts a real problem?

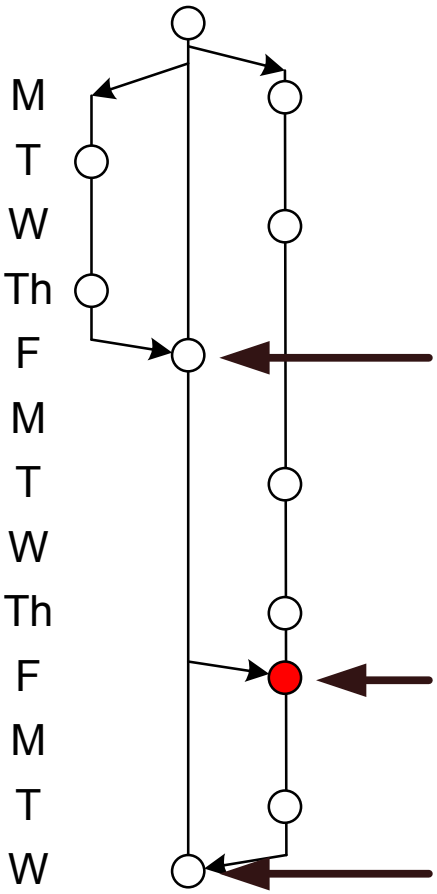


histories of 9 open-source projects:

size:	26K–1.4MSLoC
developers:	298
versions:	140,000

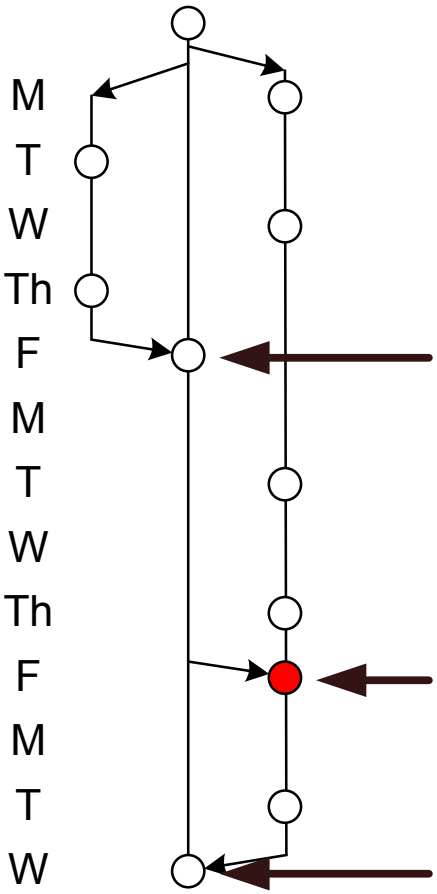
Perl5, Rails, Git, jQuery, Voldemort, MaNGOS, Gallery3, Samba, Insoshi

Are textual collaborative conflicts a real problem?



RQ1: How frequent are textual conflicts?

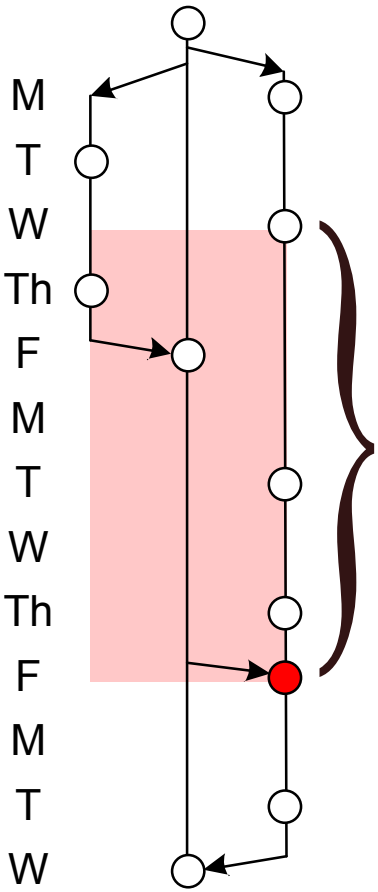
Are textual collaborative conflicts a real problem?



RQ1: How frequent are textual conflicts?

16% of the merges have textual conflicts.

Are textual collaborative conflicts a real problem?

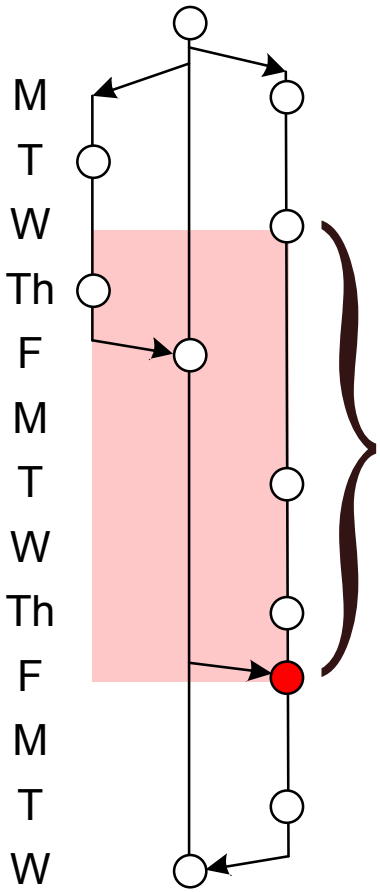


RQ1: How frequent are textual conflicts?

16% of the merges have textual conflicts.

RQ2: How long do textual conflicts persist?

Are textual collaborative conflicts a real problem?



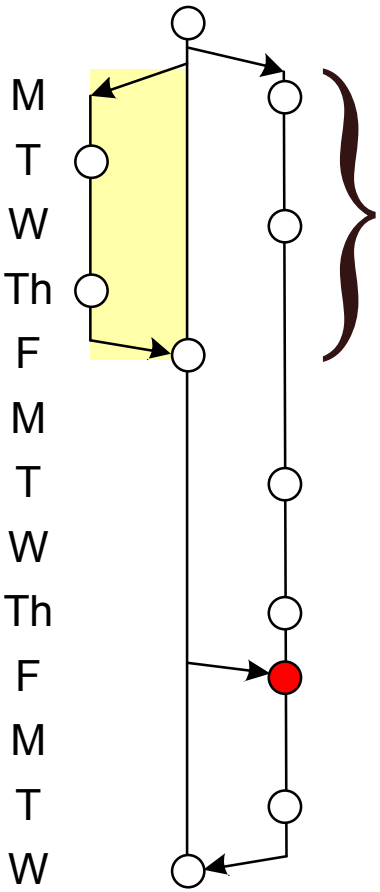
RQ1: How frequent are textual conflicts?

16% of the merges have textual conflicts.

RQ2: How long do textual conflicts persist?

Conflicts live a mean of 9.8 and median of 1.6 days.
The worst case was over a year.

Are textual collaborative conflicts a real problem?



RQ1: How frequent are textual conflicts?

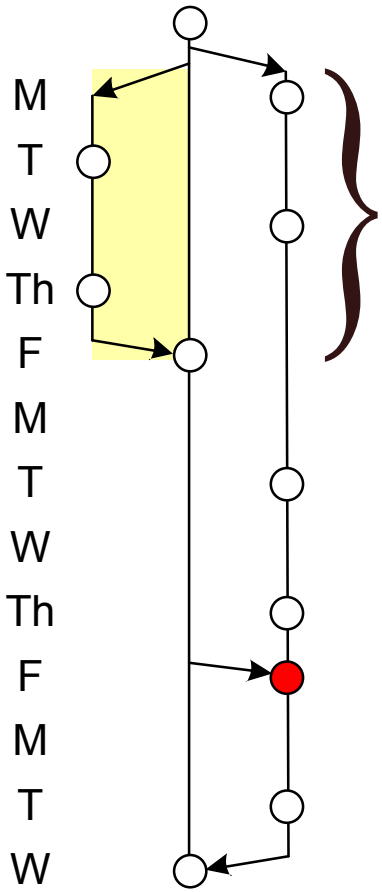
16% of the merges have textual conflicts.

RQ2: How long do textual conflicts persist?

Conflicts live a mean of 9.8 and median of 1.6 days.
The worst case was over a year.

RQ3: How long do textually-safe merges persist?

Are textual collaborative conflicts a real problem?



RQ1: How frequent are textual conflicts?

16% of the merges have textual conflicts.

RQ2: How long do textual conflicts persist?

Conflicts live a mean of 9.8 and median of 1.6 days. The worst case was over a year.

RQ3: How long do textually-safe merges persist?

Textually-safe merges live a mean of 11.0 and median of 1.9 days.

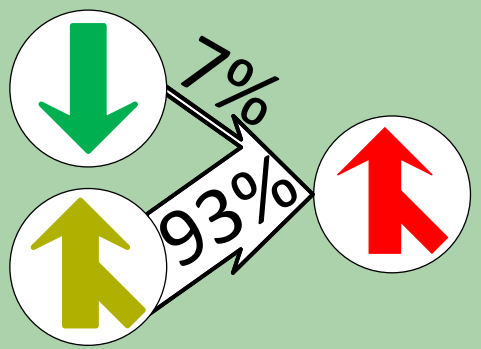
How dangerous are safe merges?

RQ4: Where do textual conflicts come from?

How dangerous are safe merges?

RQ4: Where do textual conflicts come from?

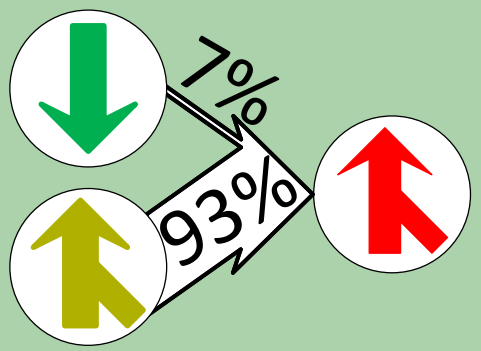
93% of textual conflicts developed from safe merges.



How dangerous are safe merges?

RQ4: Where do textual conflicts come from?

93% of textual conflicts developed from safe merges.

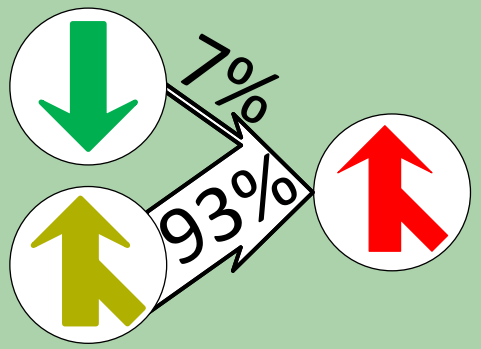


RQ5: Do textually-safe merges devolve into conflicts?

How dangerous are safe merges?

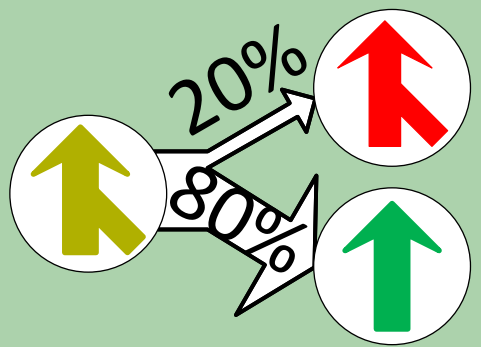
RQ4: Where do textual conflicts come from?

93% of textual conflicts developed from safe merges.



RQ5: Do textually-safe merges devolve into conflicts?

20% of textually-safe merges developed into conflicts.



Do higher-order collaborative conflicts exist?

program	conflicts			safe merges
	textual	build	test	
Git	17%	<1%	4%	79%
Perl5	8%	4%	28%	61%
Voldemort	17%	10%	3%	69%

RQ6: Does merged code fail to build or fail tests?

One in three conflicts are of higher-order.

Crystal is in the wild

"Crystal handles several projects and users effortlessly and presents the necessary information in a simple and understandable way."

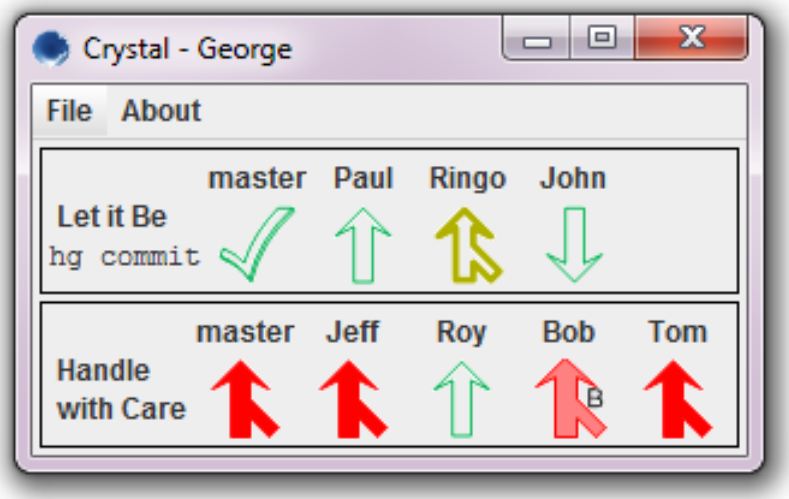
– a user

Microsoft Beacon

- A centralized version control-based tool.
- Microsoft product groups will use Beacon to help identify conflicts earlier in the development process.
- We will conduct user studies to measure effects on developers.

Contributions

- Introduced speculative analysis to guide future actions.
- Developed Crystal to precisely detect conflicts and unobtrusively inform developers.
- Analyzed 9 projects with over 140,000 versions: conflicts are frequent and persistent.



http://crystalvc.googlecode.com

Jacob T. Biehl, Mary Czerwinski, Greg Smith, and George G. Robertson. FASTDash: A visual dashboard for fostering awareness in software teams. pages 1313–1322, San Jose, CA, USA, 2007. ISBN 978-1-59593-593-9. doi: 10.1145/1240624.1240823.

Bruce Childers, Jack W. Davidson, and Mary Lou Soffa. Continuous compilation: A new approach to aggressive and adaptive code transformation. In *IPDPS*, 2003.

Prasun Dewan and Rajesh Hegde. Semi-synchronous conflict detection and resolution in asynchronous software development. In *ECSCW*, pages 159–178, Limerick, Ireland, 2007.

Eclipse.

Mário Luís Guimarães and António Rito-Silva. Towards real-time integration. In *CHASE*, pages 56–63, Cape Town, South Africa, May 2010.

Lile Hattori and Michele Lanza. Syde: A tool for collaborative software development. In *ICSE Tool Demo*, pages 235–238, Cape Town, South Africa, May 2010. ISBN 978-1-60558-719-6. doi: 10.1145/1810295.1810339.

Peter Henderson and Mark Weiser. Continuous execution: The VisiProg environment. In *ICSE*, pages 68–74, London, England, UK, Aug. 1985.

R. R. Karinthi and M. Weiser. Incremental re-execution of programs. pages 38–44, St. Paul, MN, USA, 1987. ISBN 0-89791-235-7. doi: 10.1145/29650.29654.

David Saff and Michael D. Ernst. Reducing wasted development time via continuous testing. In *ISSRE*, pages 281–292, Denver, CO, USA, Nov. 2003. ISBN 0-7695-2007-3.

David Saff and Michael D. Ernst. An experimental evaluation of continuous testing during development. In *ISSTA*, pages 76–85, Boston, MA, USA, July 2004. doi: 10.1145/1007512.1007523.

Anita Sarma, Zahra Noroozi, and André van der Hoek. Palantír: Raising awareness among configuration management workspaces. In *ICSE*, pages 444–454, Portland, OR, May 2003. ISBN 0-7695-1877-X.

Steve Streeting. Sourcetree. `http://www.sourcetreeapp.com`, 2010.

Jan Wloka, Barbara Ryder, Frank Tip, and Xiaoxia Ren. Safe-commit analysis to facilitate team software development. In *ICSE*, pages 507–517, Vancouver, BC, Canada, May 2009. ISBN 978-1-4244-3453-4. doi: 10.1109/ICSE.2009.5070549.