Distributed Version Control Systems

CS 595 Technical Presentation
Wednesday, May 4, 2011
3:10pm

Thesis

Distributed version control system (DVCS) such as **git** have many advantages that make them much more scalable than centralized version control systems (CVCS) such as Subversion.

Outline

- **Why version control?**
- Version control system concepts
- Version control system models
  - Centralized (CVCS)
  - Distributed (DVCS)
- Case Study
  - CVCS to DVCS migrations
  - A class submission system for group projects
- Conclusion

Development Teams

- Example: Imperial Software, Inc.
- Problem:
    - Bugs cause code changes
    - Code changes cause other problems

Effects of Code Changes

- "The amount of space to store the source code…may be several times that needed for any particular version."
- "Fixes made to one version of a module sometimes fail to get made to other versions."
- "When changes occur it is difficult to tell exactly what changed and when."
- "When a customer has a problem it is hard to figure out what version he has."

(Rochkind 1975)

Software Maintainers

- Example: **Linus Torvalds**
- Problems: (Torvalds 2007)
  - Separating good changes from bad changes
  - Merging other people's changes
  - Release management
  - Need guarantee that code is valid/trustworthy

Outline

- Why version control?
- **Version control system concepts**
- Version control system models
  - Centralized (CVCS)
  - Distributed (DVCS)
- Case Study
  - CVCS to DVCS migrations
  - A class submission system for group projects
- Conclusion

Features provided by SCCS

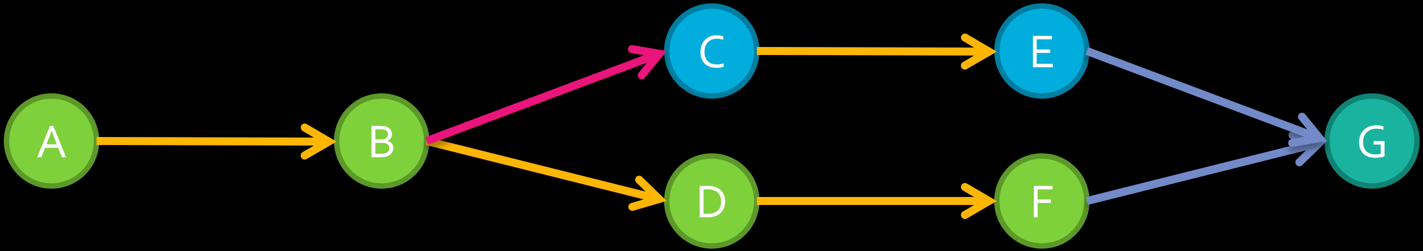
- Storage
- Protection
- Identification
- Documentation

(Rochkind 1975)

Modern VCS Features

- Branches
  - Independent subprojects
  - Release branches vs. Feature branches
- Merging
  - Re-integrating a branch

(O'Sullivan 2009)



Outline

- Why version control?
- Version control system concepts
- **Version control system models**
  - **Centralized (CVCS)**
  - Distributed (DVCS)
- Case Study
  - CVCS to DVCS migrations
  - A class submission system for group projects
- Conclusion

Centralized VCS (CVCS)

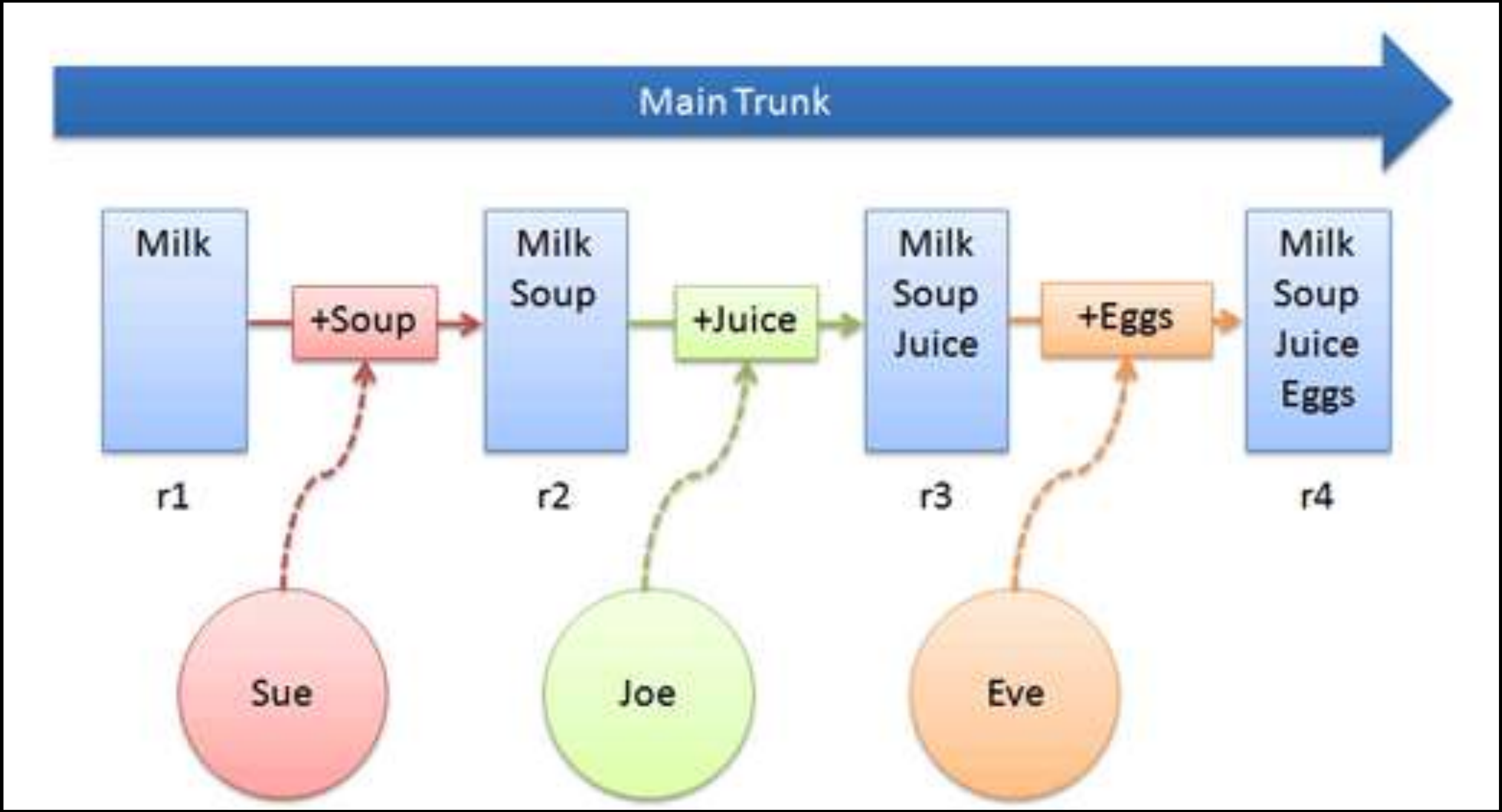


Image source: http://betterexplained.com/wp-content/uploads/version_control/distributed/centralized_example.png

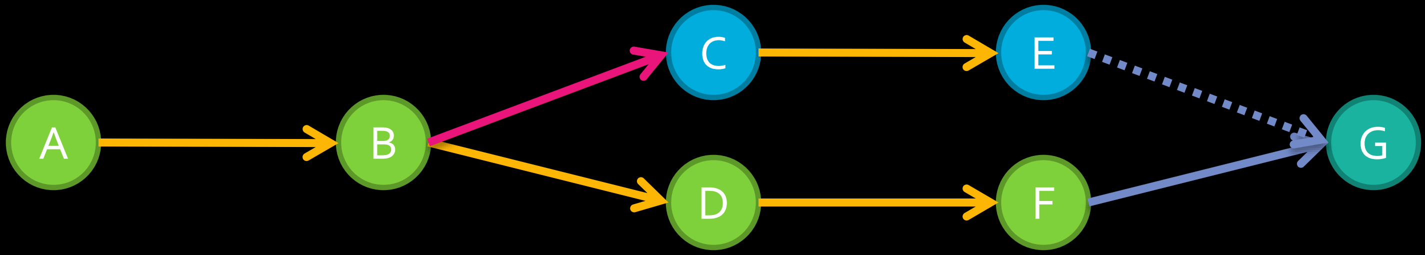
CVCS Implementation: SVN

- Storage
  - Central repository, working copy
- Protection
  - Commit access
- Identification
  - Sequence number
- Documentation
  - Log message, timestamp, author

(Pilato, et. al., 2008)

Branches in SVN

- Create is a server-side copy
  - All branches visible to all developers
- Merge is difficult
  - Must linearize history (Pilato 2008)
  - Renames present problems (O'Sullivan 2009)



Centralized Model Problems

- All commits visible to all developers (O'Sullivan 2009)
  - vader593 runs `svn update; make`
  - vader593 walks away to get coffee
  - vader593 comes back to a build error
  - What happened?
    - admiral494 committed a bad one-line change

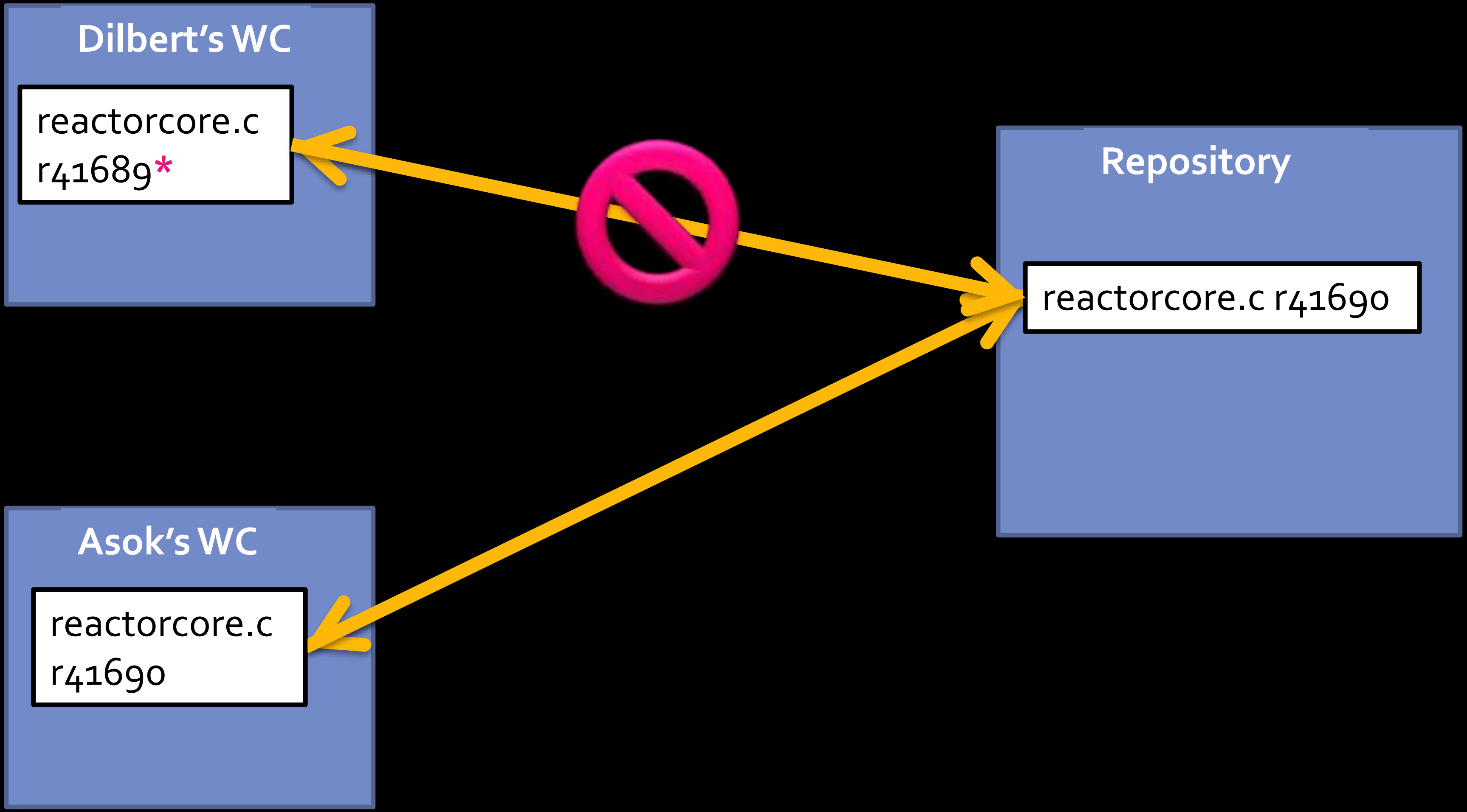
Centralized Model Problems

- Conflicts detected after commit attempt (O'Sullivan 2009)

```
From: pointyhairedboss
To: dilbert
A customer found a bug that we think is in
/deathstar/reactorcore.c.  Could you please
look into it?

From: pointyhairedboss
To: asok
I've noticed that all the C files in
/deathstar/ contain TABs.  I read somewhere
that TABs are bad.  Could you please fix this?
```

Centralized Model Problem



Outline

- Why version control?
- Version control system concepts
- **Version control system models**
  - Centralized (CVCS)
  - **Distributed (DVCS)**
- Case Study
  - CVCS to DVCS migrations
  - A class submission system for group projects
- Conclusion

Distributed VCS (DVCS)

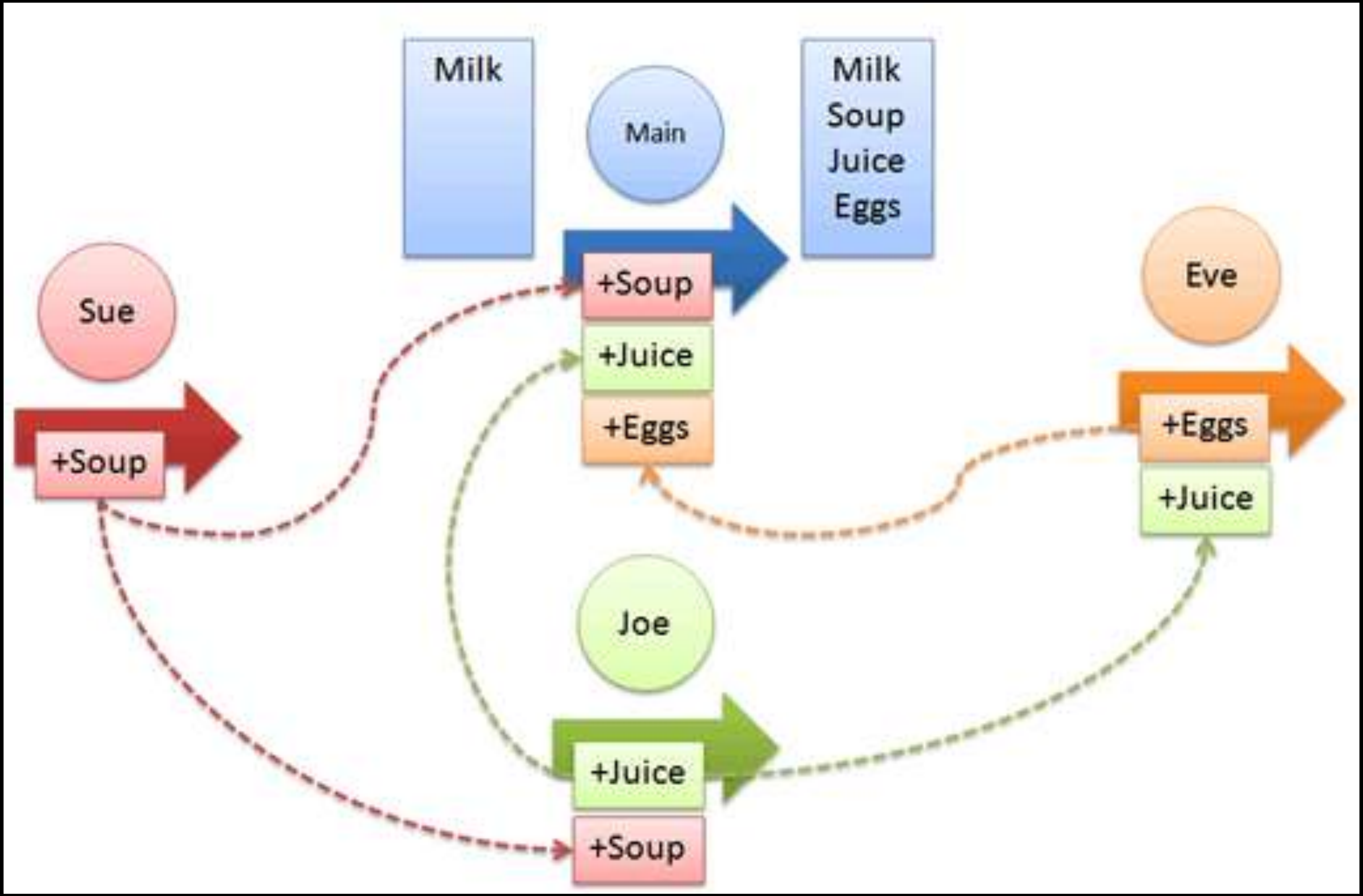


Image source: http://betterexplained.com/wp-content/uploads/version_control/distributed/distributed_example.png

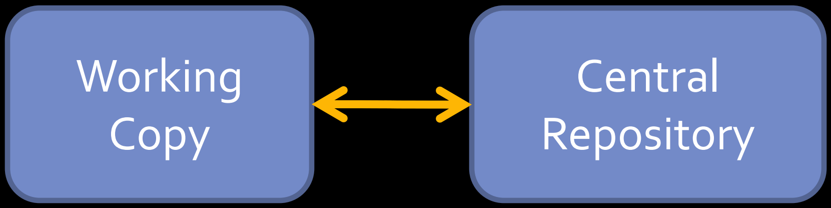
DVCS Implementation: git

- Storage
  - Each developer has entire project history
- Identification
  - SHA-1 hash
- Protection
  - SHA-1 hashes, “Pull” model
- Documentation
  - Log message, timestamp, author

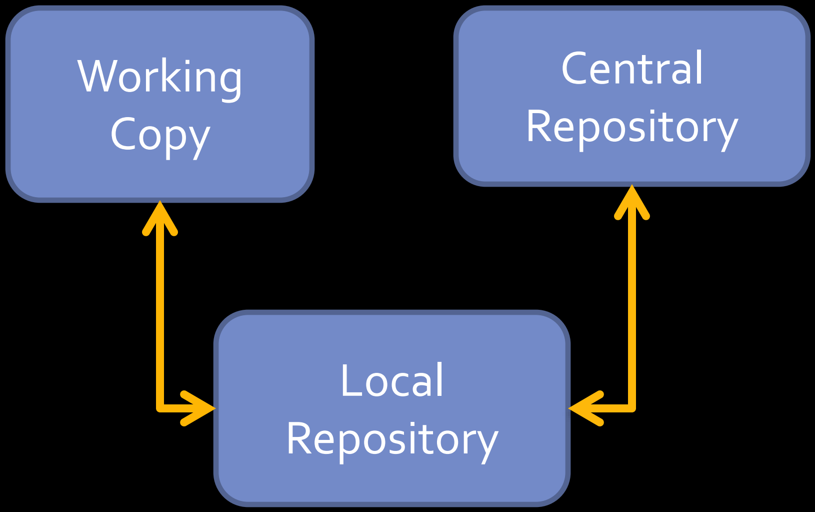
(Chacon)

Indirection

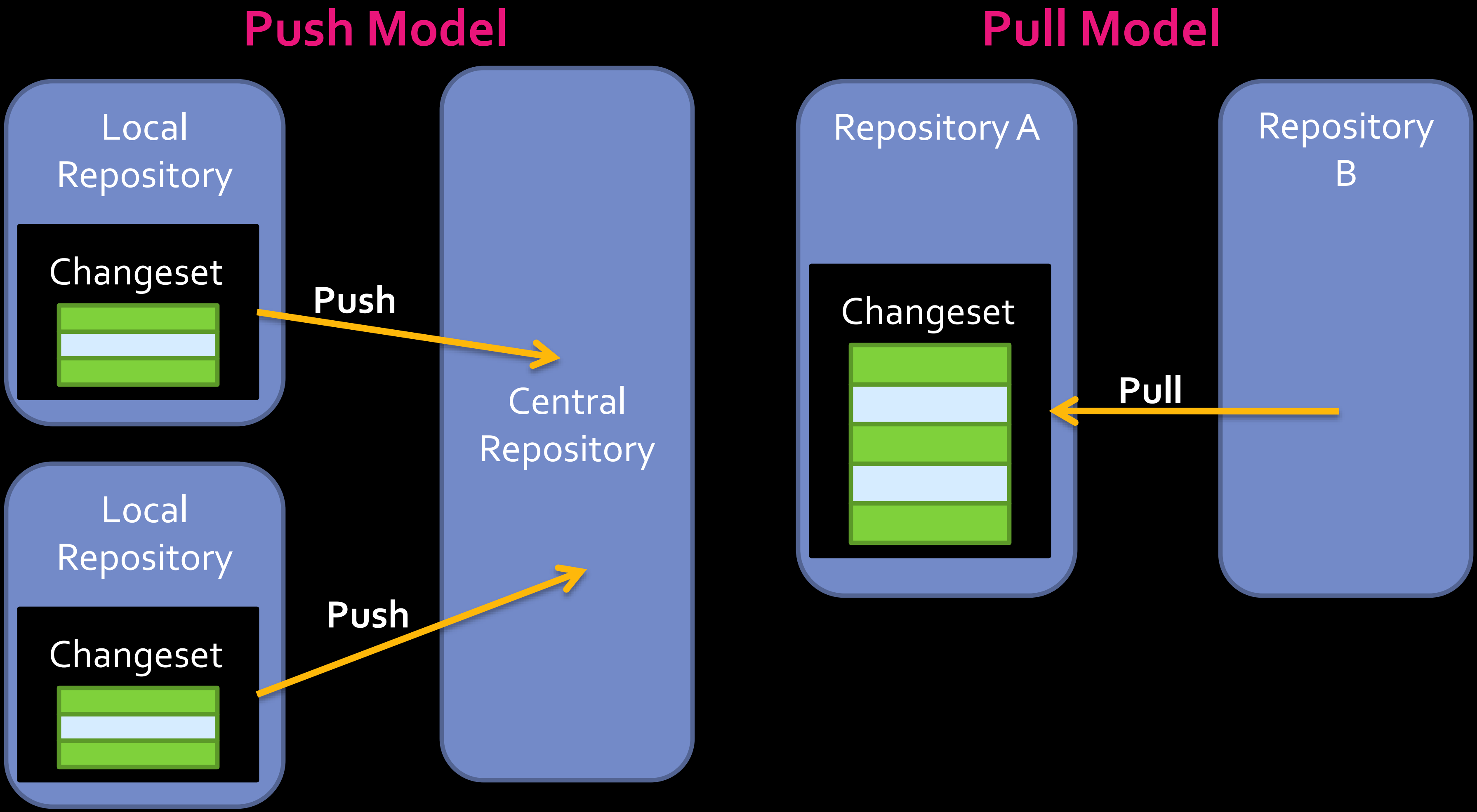
Subversion



Git

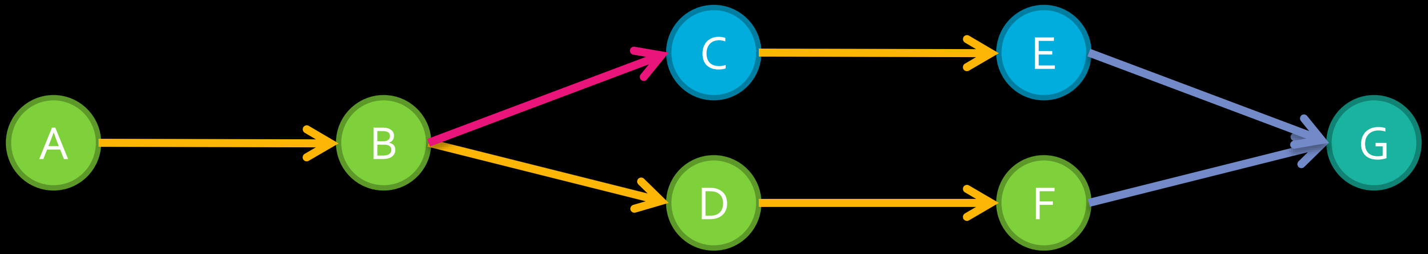


Publishing Code Changes



Branches in git

- Can be implicit or explicit
- Lightweight
- Merging preserves tree structure



Solutions to CVCS Problems

- CVCS problem:
  - All commits visible to all developers (O'Sullivan 2009)
- DVCS solution:
  - Each developer works on a separate branch

Solutions to CVCS Problems

- CVCS Problem
  - Conflicts detected after commit attempt (O'Sullivan 2009)
- DVCS solution
  - Detected at merge time, not compile time

Outline

- Why version control?
- Version control system concepts
- Version control system models
  - Centralized (CVCS)
  - Distributed (DVCS)
- **Case Study**
  - **CVCS to DVCS migrations**
  - **A class submission system for group projects**
- Conclusion

de Alwis Case Study

- Brian de Alwis, Jonathan Sillito
- Study on various projects moving from CVCS to DVCS
  - Perl
  - OpenOffice
  - Python
  - NetBSD

Reasons for Switching

- Developers without commit access
- “Simple automatic merging”
- “Improved support for experimental changes”
- Offline development

Outline

- Why version control?
- Version control system concepts
- Version control system models
  - Centralized (CVCS)
  - Distributed (DVCS)
- Case Study
  - CVCS to DVCS migrations
  - A class submission system for **group projects**
- **Conclusion**

Conclusion

Distributed version control system (DVCS) such as **git** have many advantages that make them much more scalable than centralized version control systems (CVCS) such as Subversion.

References

Chacon, Scott. "About Git." *Git: The fast version control system.* 6 April 2011. http://git-scm.com/about

de Alwis, Brian, and Jonathan Sillito. "Why are software projects moving from centralized to decentralized version control systems?" 2009 *ICSE Workshop on Cooperative and Human Aspects on Software Engineering.* pp. 36-39. 2009.

Laadan, Oren, et. al. 2010. Teaching operating systems using virtual appliances and distributed version control. In *Proceedings of the 41st ACM technical symposium on Computer science education* (SIGCSE '10). ACM, New York, NY, USA, 480-484. DOI=10.1145/1734263.1734427 http://doi.acm.org/10.1145/1734263.1734427

O'Sullivan, Bryan. 2009. Making Sense of Revision-control Systems. *Queue* 7, 7, Pages 30 (August 2009), 11 pages. DOI=10.1145/1594204.1595636 http://doi.acm.org/10.1145/1594204.1595636

Pilato, C. Michael, et. al. Version Control with Subversion, Second Edition. O'Reilly. 2008.

Torvalds, Linus. "Tech Talk: Linus Torvalds on git." 14 May 2007. Online video clip. YouTube.

Questions?