

1. Введение в системы версионирования

Раздел познакомит с основными процессами совместной работы над файлами и кодом, а также с инструментами, которые были созданы для обеспечения этих процессов.

1.1. Совместная работа над данными

Предположим, что у вас есть черновик важного письма в простом текстовом документе, над которым вы хотите работать совместно с коллегами. Каждый из них будет ответственен за различные аспекты этого черновика. В случае возникновения вопросов необходим способ определения автора фрагмента.

ⓘ Attention

Каким образом можно организовать подобный процесс?

Варианты решения

- внешний накопитель
- электронная почта
- общий сетевой диск

ⓘ Attention

Как вы думаете, какой из этих способов обладает наибольшим и наименьшим количеством возможностей?

Рассмотрим каждый из способов, задав себе четыре вопроса:





1. Можно ли работать с данными одновременно?
2. Можно ли работать с данными автономно?
3. Можно ли отследить историю изменений?
4. Можно ли установить авторство изменений?

Классификация инструментов для совместной работы

Внешний накопитель







Основная копия файла всегда находится на диске, который может быть в произвольный момент времени только у одного человека. Получив внешний накопитель, пользователь может работать с файлом автономно. Историю изменений и их авторство невозможно отследить, т.к. дата, время и авторство сохраняется только для последнего изменения.

-  Одновременная работа
-  Автономная работа
-  История изменений
-  Авторство

Электронная почта



Основная копия файла пересылается в электронном письме. Во избежание конфликтов каждый пользователь работает над файлом по очереди. Получив письмо, пользователь может работать с файлом автономно. Историю правок можно отследить путем сравнения нескольких версий файла, отправленных в разное время, а авторство можно установить по отправителю письма с версией, где искомый фрагмент подвергся изменению.

-  Одновременная работа
-  Автономная работа
-  История изменений
-  Авторство

Общий сетевой диск



Основная копия файла доступна всем участникам, однако в силу того, что файл является простым текстовым документом, одновременное внесение правок может привести к потере части изменений. Файл также не доступен без постоянной связи с сетевым диском, что исключает возможность автономной работы. Историю изменений и их авторство невозможно отследить, т.к. дата, время и авторство сохраняется только для последнего изменения.

- Одновременная работа
- Автономная работа
- История изменений
- Авторство

Итоговое сравнение

Электронная почта	● ● ● ●
Внешний накопитель	● ● ● ●
Общий сетевой диск	● ● ● ●

Все перечисленные требования равноценно применимы к любому процессу совместной работы над кодом. Для построения подобных процессов были созданы инструменты, обеспечивающие частичное или полное выполнение всех упомянутых требований.

1.2. Почему Git?

1. Полная децентрализованность и автономность

Git не требует наличия централизованного хранилища и позволяет осуществлять обмен данными путем передачи обновлений между автономными репозиториями.

2. Исчерпывающая история изменений с указанием авторства

Механизм позволяет восстанавливать хронологию модификации кода и осуществлять возврат к произвольной её точке.

3. Гарантии целостности хранимых данных

Консистентность данных и истории обеспечивается с помощью криптографических операций.

4. Быстрое переключение контекста

Возможность хранить изолированные контексты позволяет работать над несколькими задачами параллельно без пересечения изменений.

5. Популярность

Git используется в подавляющем большинстве компаний, связанных с разработкой, в т.ч. Google, Facebook, Microsoft, Twitter, Netflix и других.

Забавный факт: На Github есть [зеркало ↗](#) официального репозитория Apache Subversion.

Подробнее ознакомиться с отличительными чертами Git можно на [сайте проекта ↗](#).

1.3. Другие системы версионирования

Помимо Git существуют другие системы версионирования кода. Обсуждение их преимуществ и недостатков выходит за рамки данного курса. В этом разделе собраны ссылки на источники информации о них.

SVN



- [Subversion — официальный сайт ↗](#)
- [Wikipedia — Apache Subversion ↗](#)
- [Perforce — Git vs. SVN – What Is The Difference? ↗](#)
- [Tabnine — SVN vs. Git: Which is right for you in 2020? ↗](#)

Mercurial



- [Mercurial - официальный сайт ↗](#)
- [Wikipedia - Mercurial ↗](#)
- [Atlassian - Git vs. Mercurial: why Git? ↗](#)
- [Perforce - Mercurial vs. Git: How Are They Different? ↗](#)
- [Хабр - Сходство и различие между Mercurial и Git ↗](#)

Сравнения трех и более систем контроля версий



- [Wikipedia — Comparison of version-control software ↗](#)
- [Medium — Version Control Software Comparison: Git, Mercurial, CVS, SVN ↗](#)