Определения

Система контроля версий — это специальное программное обеспечение, обеспечивающее работу с постоянно изменяющейся информацией. В задачи системы контроля версий входит хранение истории изменений и обеспечение одновременной работы нескольких разработчиков.

Определения

Репозиторий — хранилище документов вместе с историей их изменения.

Версия, ревизия — зафиксированный набор изменений документов, которому присвоена определенная дата и порядковый номер. Как правило ревизия снабжается комментарием, описывающим сделанные изменения.

Основная версия (HEAD) — последняя версия в репозитории. При первом обращении к репозиторию работа начинается именно с этой версии.

Рабочая копия — набор документов, относящийся к какой либо версии, с которыми производится работа.

Определения

Обновление (check-out) — получение от репозитория новой рабочей копии.

Фиксация (commit) — внесение набора изменений и формирование новой версии (ревизии) в репозитории.

Слияние — объединение разных правок в одном файле в процессе фиксации изменений.

Конфликт — ситуация, при которой автоматическое слияние невозможно, и требуется вмешательство человека.

Ветвь — независимое направление разработки. Может содержать несколько версий.

Определения

Централизованная система контроля версий — это система контроля версий, в которой репозиторий расположен на сервере, а пользователи получают от него только рабочие копии.

Для получения истории изменений и прочей служебной информации также необходимо обратиться на сервер.

Примеры: CVS, SVN, Azure DevOps

Определения

Распределенная система контроля версий — это система контроля версий, в которой каждый клиент работает со своей полной копией репозитория. Сервер выступает только в роли хранилища основного репозитория.

В процессе синхронизации данных с сервером происходит слияние версий.

Примеры: Git, Mercurial

Определения

В рамках распределенной системы контроля версий вводятся следующие понятия:

Клонирование (clone) — первоначальное получение копии репозитория. Для выполнения клонирования необходимо знать расположение репозитория (путь к папке, либо интернет-адрес).

После выполнения **clone**, репозиторий запоминает свое происхождение (**origin**).

Получение (pull) — обновление локального репозитория с целью получить все новые версии с удаленного репозитория.

Отправка (push) — синхронизация данных с целью отправки локальных версий на удаленный репозиторий. Не может быть сделана автоматически при наличии конфликтов.

Git

Распределенная система контроля версий с открытым исходным кодом.

Разрабатывалась для обслуживания процесса разработки ядра ОС Linux.

Является стандартом де-факто для современной программной разработки.

Доступна на сайте: https://git-scm.com. Также доступна онлайн документация на русском языке: https://git-scm.com/book/ru/v2.

С использованием git построены многие системы обмена исходными кодами. Например, github (https://github.com/)

Архитектура Git

Репозиторий git — это папка, содержащая **рабочую копию**, в которой расположена скрытая папка **.git**, хранящая служебную информацию.

Настройки git хранятся в файле **.gitconfig**. В зависимости от расположения, настройки влияют:

- В системной папке: на всех пользователей и все репозитории.
- В профиле пользователя (c:\users\user\.gitconfig) — на все репозитории данного пользователя. Переопределяют системные настройки.
- В репозитории — на данный репозиторий, переопределяет все вышележащие настройки.

Файл **.gitignore** в репозитории определяет игнорируемые файлы.

Архитектура Git

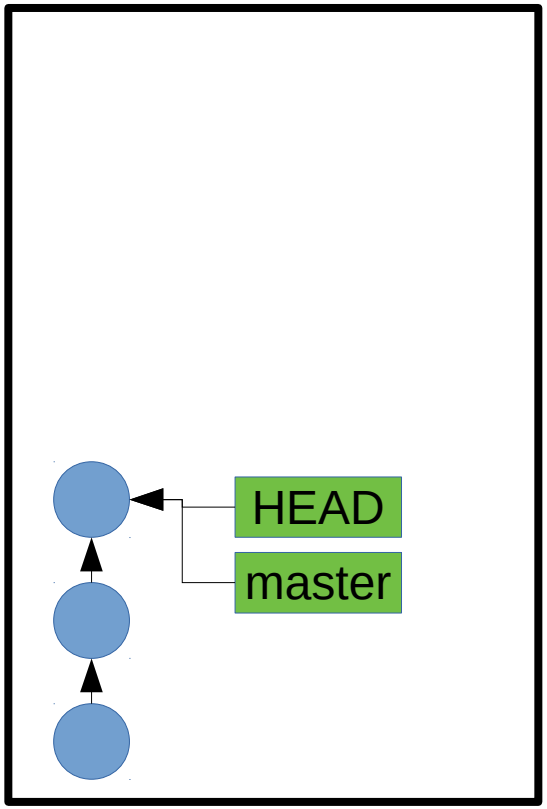
Каждая версия в git содержит в себе следующую информацию:

- Уникальный номер (хэш), состоящий из 40 шестнадцатеричных символов. При отображении допускается сокращение хэша до наиболее короткой, но однозначной последовательности.
- Хэш предыдущей (родительской) версии.
- Сделанные изменения. Описываются только те файлы, которые отличаются от предыдущей версии.
- Информация об авторе версии: его имя и e-mail.
- Комментарий к данной версии, записанный автором при ее создании.
- Время создания версии.

Дерево версий Git

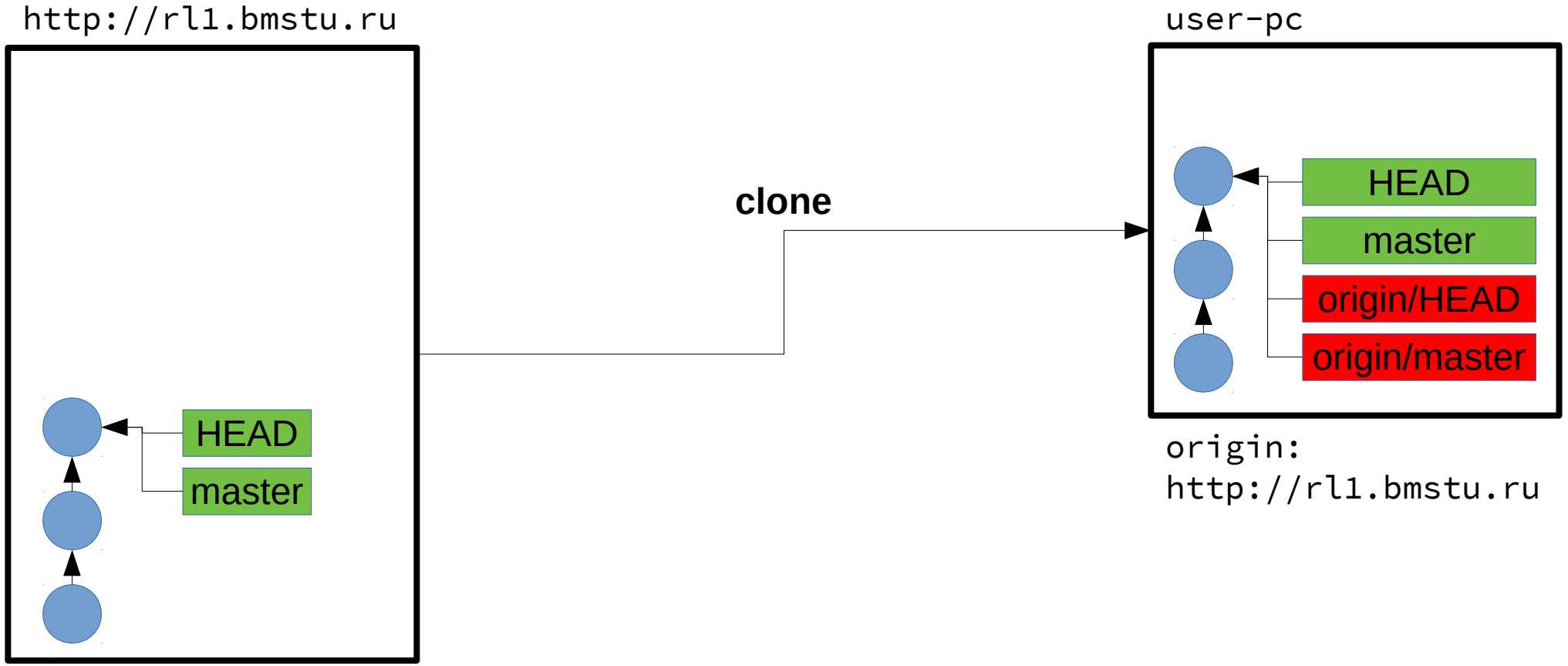
Версии в репозитории складываются в дерево. В вершине этого дерева расположена основная версия — **HEAD**. Основная ветвь репозитория носит название **master**.

http://rl1.bmstu.ru



Дерево версий Git

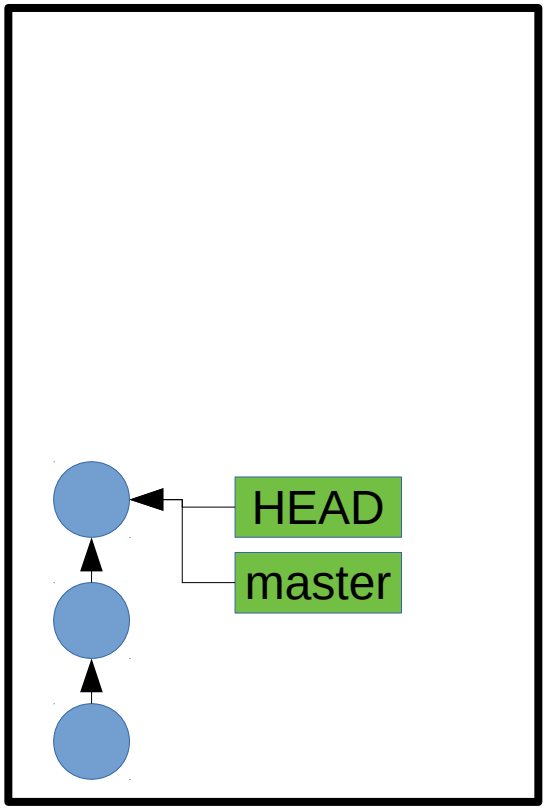
При клонировании репозитория создается его **полная копия** на локальной машине. Данный репозиторий хранит информацию о своем происхождении (**origin**), а также положение веток в исходном репозитории.



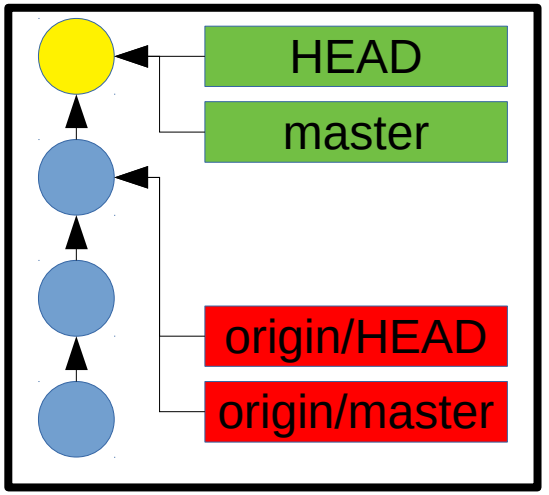
Дерево версий Git

Новая версия добавляется с использованием версии HEAD в качестве родительской, после чего указатели передвигаются. Указатели **origin** остаются неизменными.

http://rl1.bmstu.ru



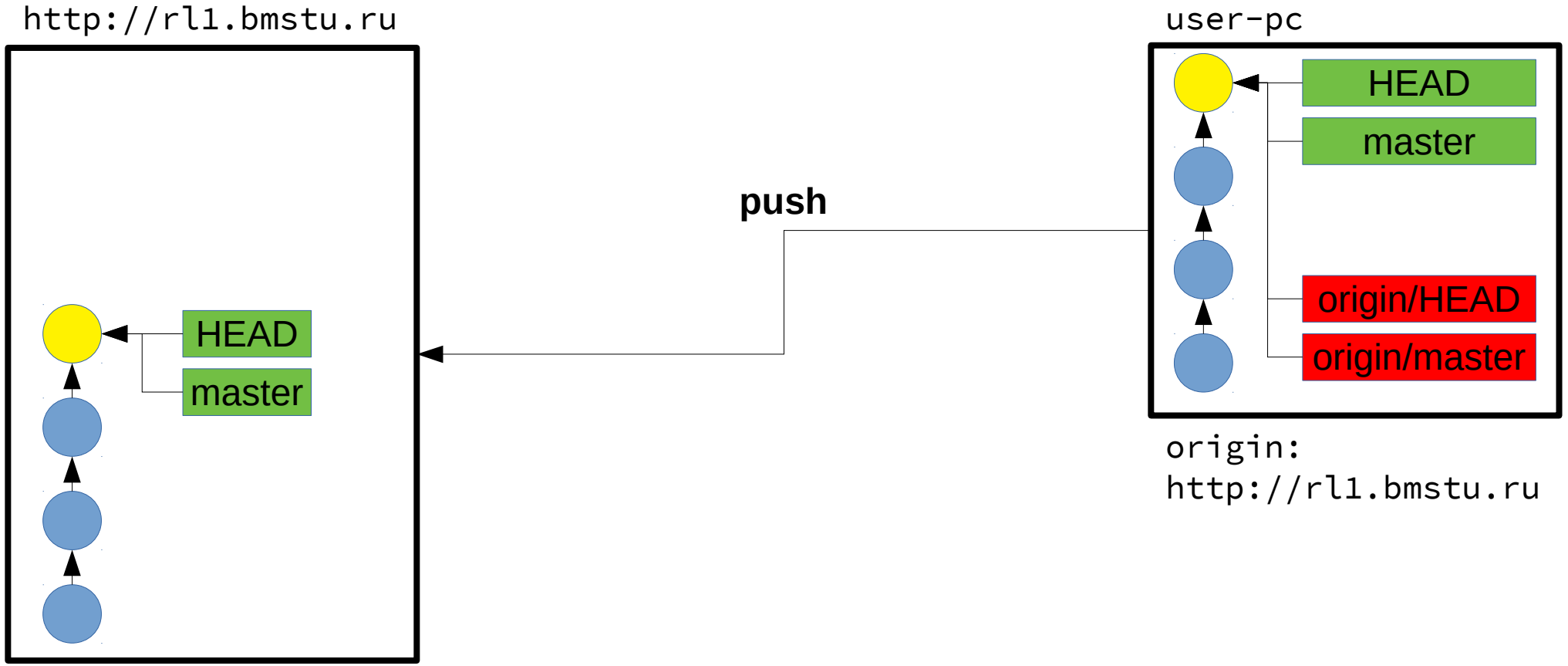
user-pc



origin:
http://rl1.bmstu.ru

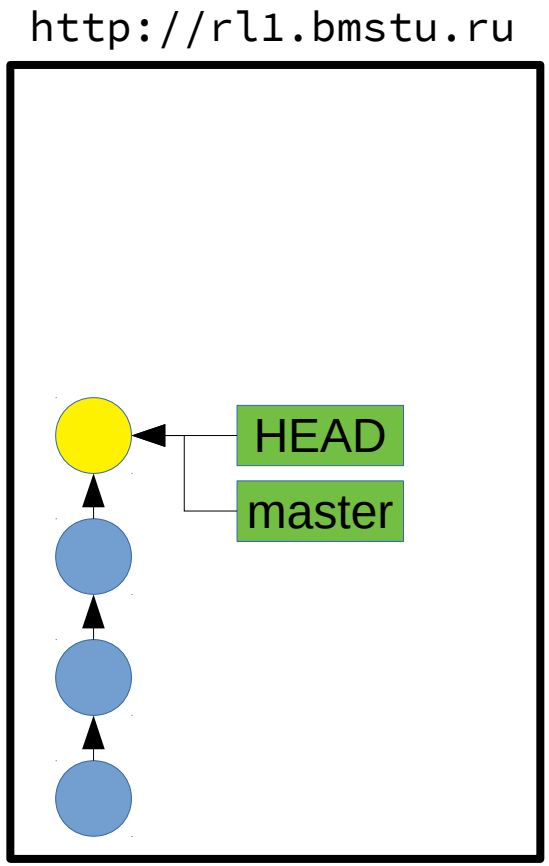
Дерево версий Git

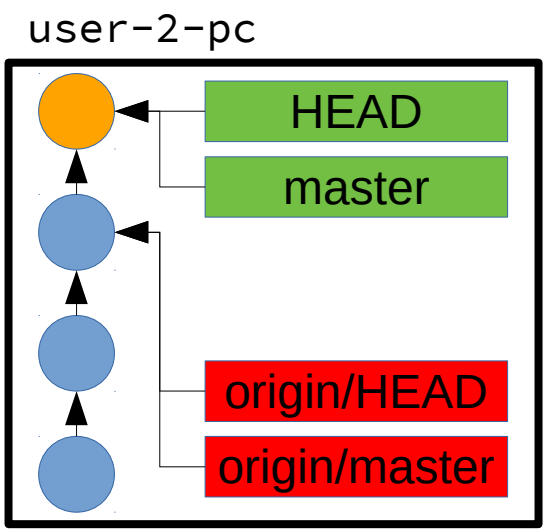
При синхронизации, используя **push**, новые версии попадают на сервер. При этом на сервере сдвигаются указатели **HEAD** и **master**, а клиент обновляет у себя положения **origin/HEAD** и **origin/master**.

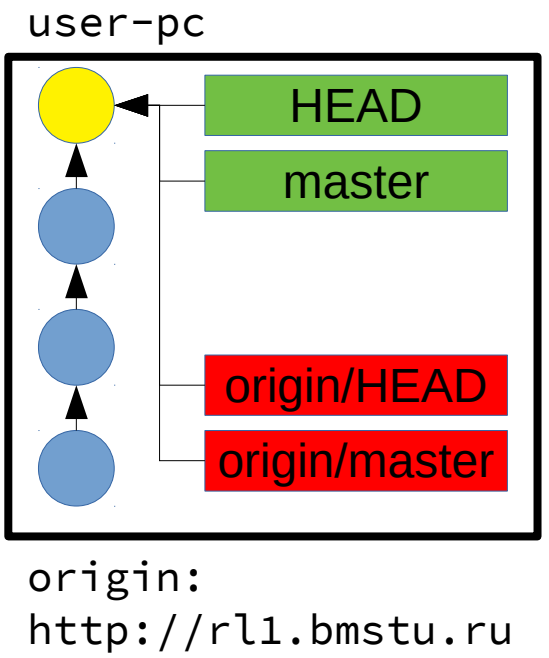


Дерево версий Git

При работе нескольких клиентов с одним репозиторием, может возникнуть ситуация, когда у одного из клиентов окажется старая копия репозитория, и он создаст новую версию в ней.

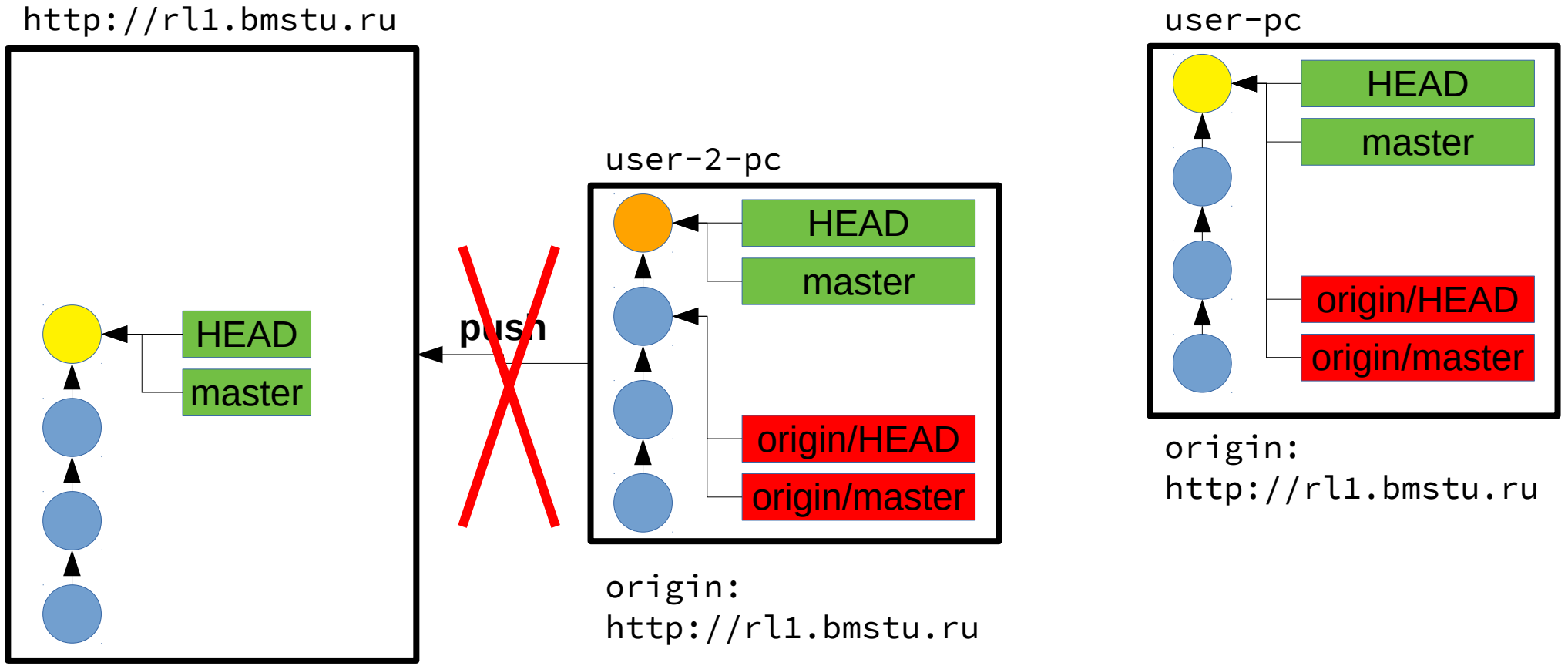






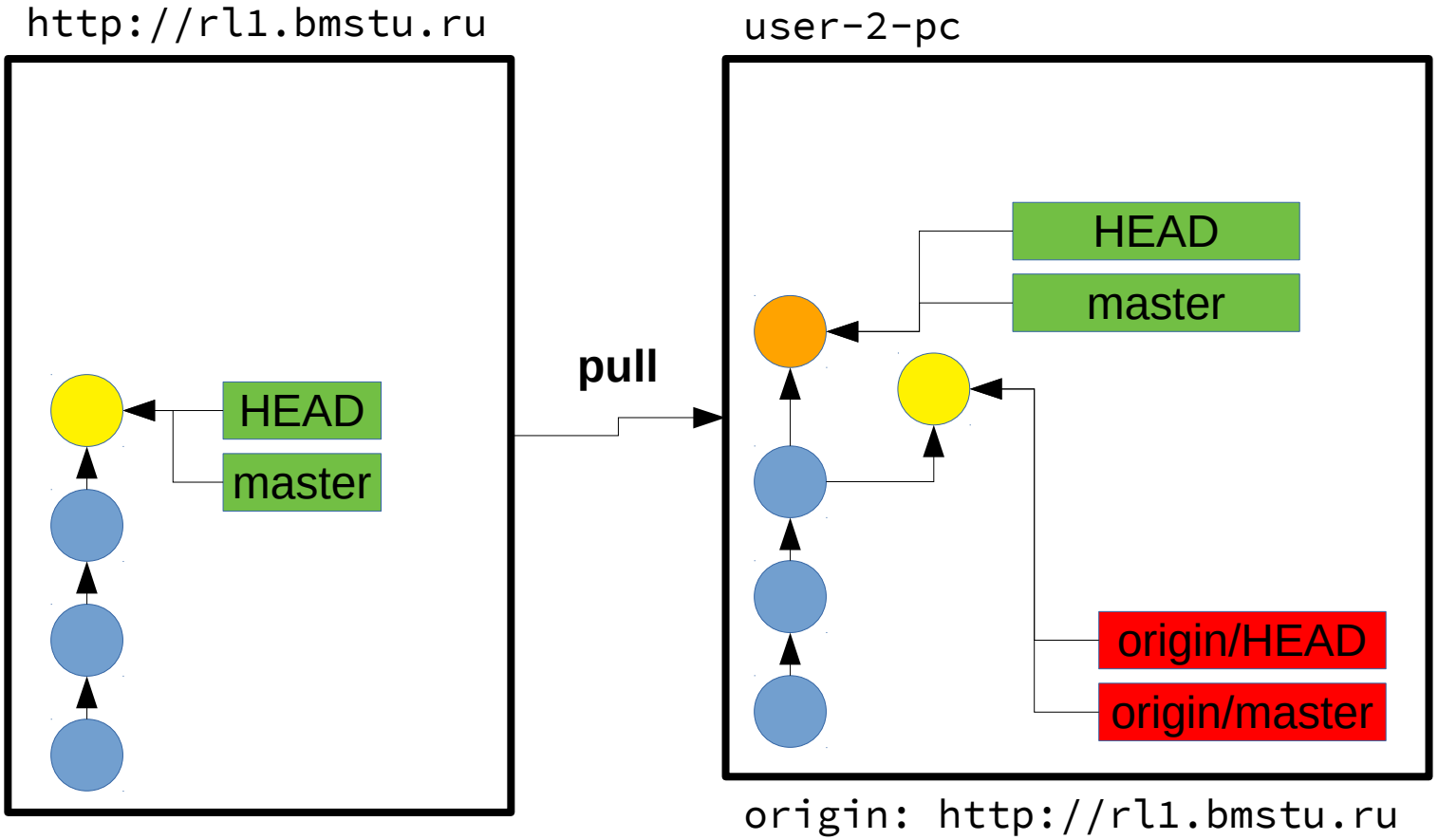
Дерево версий Git

Команда **push** будет отклонена сервером. Указатель **master** уже передвинулся на новую версию, и добавить новую версию к версии **origin/master** невозможно, так как репозиторий про нее не знает.



Дерево версий Git

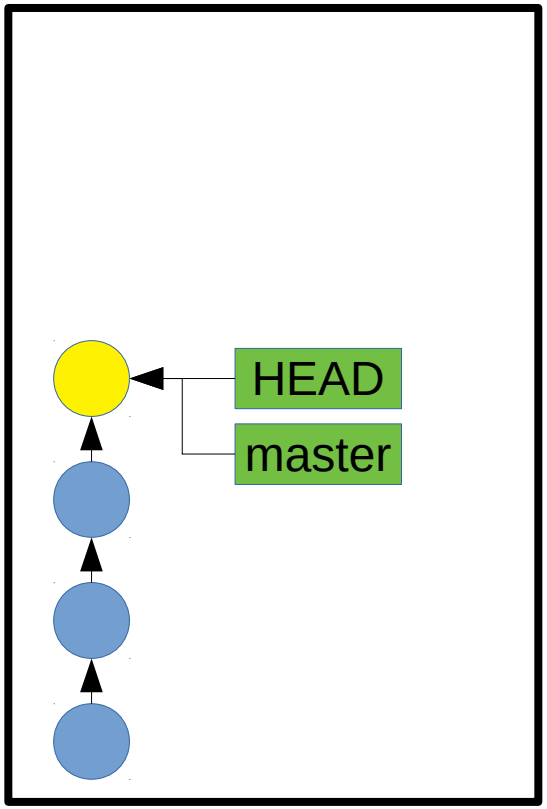
Клиенту 2 в такой ситуации необходимо выполнить команду **pull** и привести свой репозиторий в состояние, соответствующее серверу. Его версия не будет потеряна.



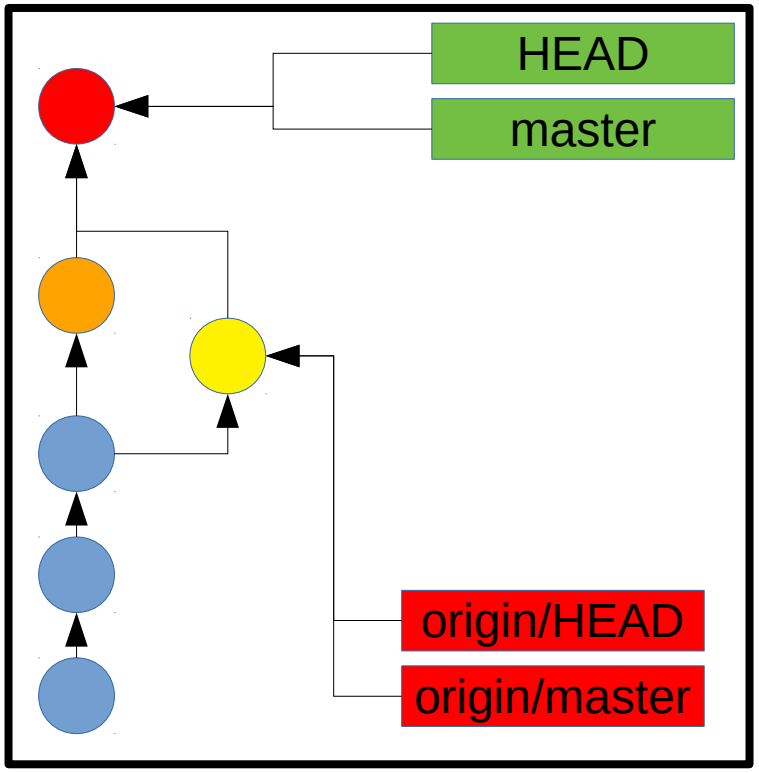
Дерево версий Git

При выполнении **pull** происходит **слияние**, ветвление устраняется. Такое состояние репозитория может быть отправлено на сервер при помощи команды **push**.

http://rl1.bmstu.ru



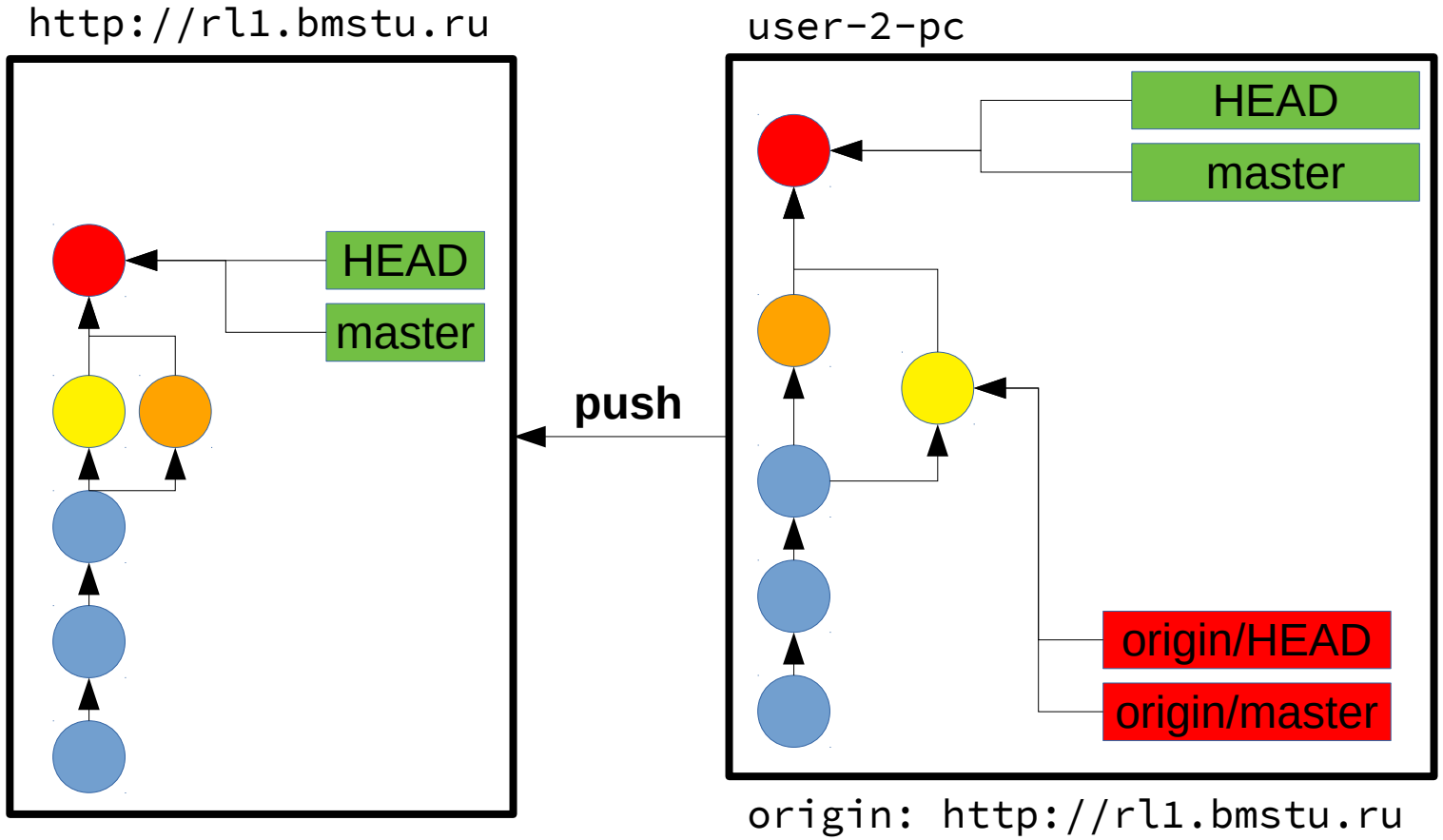
user-2-pc



origin: http://rl1.bmstu.ru

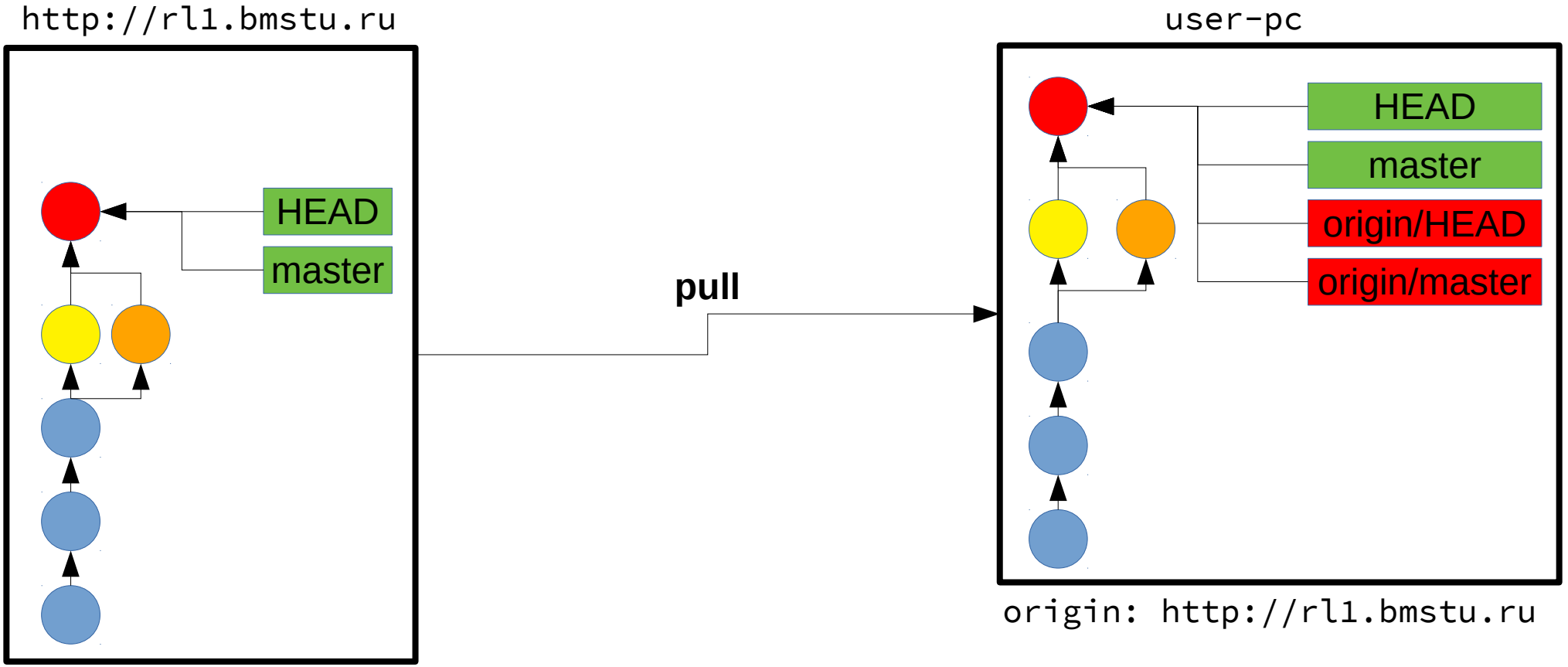
Дерево версий Git

При выполнении **pull** происходит **слияние**, ветвление устраняется. Такое состояние репозитория может быть отправлено на сервер при помощи команды **push**.



Дерево версий Git

Чтобы избежать подобной проблемы, следует выполнять команду **pull** до того, как будет происходить добавление новых версий в репозиторий.



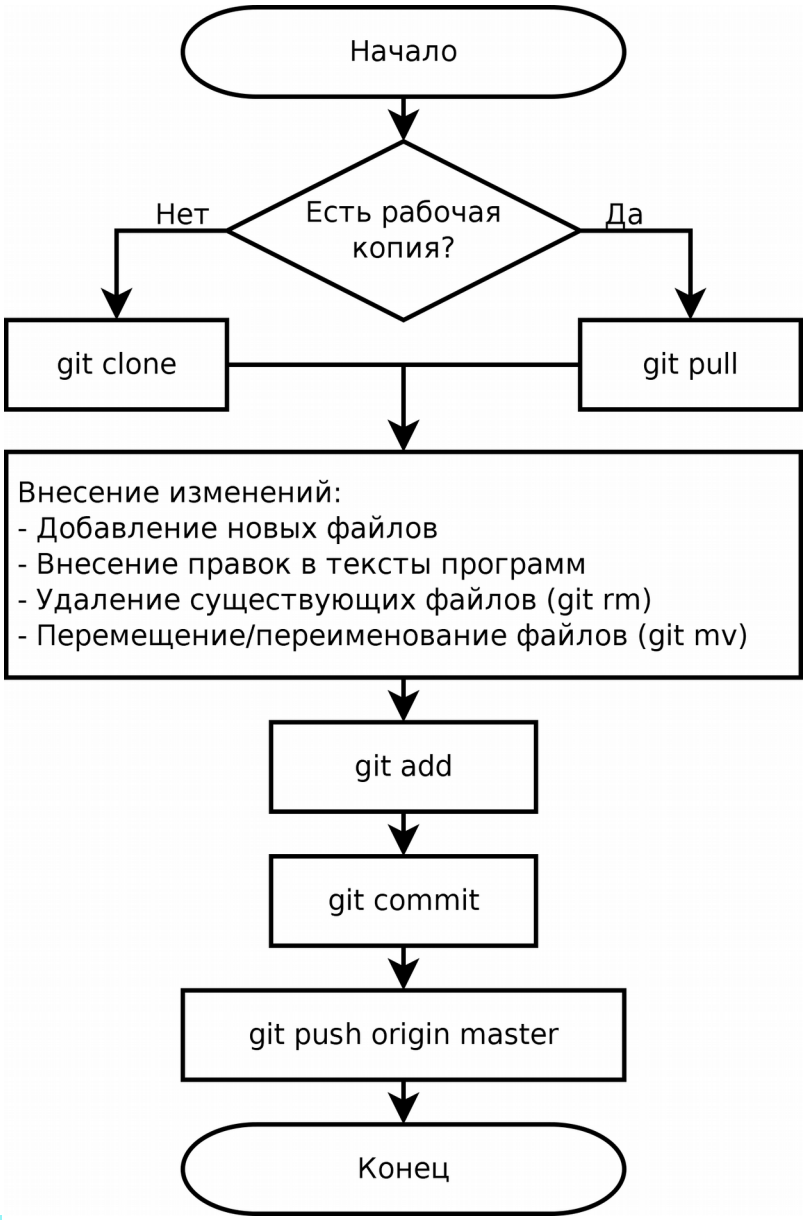
Работа с Git

Работа с git может быть представлена в виде блок-схемы.

Приведенные в блок-схеме команды одинаковы для всех клиентов git.

Работа с файлами между получением рабочей копии и созданием новой версии аналогична работе с любыми другими файлами, кроме двух случаев:

- Перемещение или переименование файлов осуществляется командой **git mv**
- Удаление файлов осуществляется командой **git rm**.



Клиенты Git

- **git** — клиент с интерфейсом командной строки. Используется различными графическими обертками или напрямую. Доступен на сайте https://git-scm.com.
- **Tortoise Git** — графический клиент для операционной системы Microsoft Windows. Встраивается в контекстное меню проводника. Доступен на сайте https://tortoisegit.org/.
- **Source Tree** — графический клиент git для Windows и Mac OS. https://www.sourcetreeapp.com/
- **SmartGit** — кросс-платформенный графический клиент. https://www.syntevo.com/smartgit/