**1. Основные отличия централизованных и распределенных систем контроля версий.**

ЦСКВ используют единственный сервер, содержащий все версии файлов, и некоторое количество клиентов, которые получают файлы из этого централизованного хранилища. Применение ЦСКВ являлось стандартом на протяжении многих лет.

В РСКВ (например Git) клиенты полностью копируют репозиторий. В этом случае, если один из серверов, через который разработчики обменивались данными, умрёт, любой клиентский репозиторий может быть скопирован на другой сервер для продолжения работы. Каждая копия репозитория является полным бэкапом всех данных.

**2. Опишите модели совместного использования файлов системами контроля версий.**

Модель Блокирование-Изменение-Разблокирование

Эта модель запрещает одновременное редактирование файла несколькими пользователями. Эксклюзивность доступа гарантируется блокировками.

Модель Копирование-Изменение-Слияние

В этой модели каждый пользовательский клиент связывается с хранилищем проекта и создает персональную рабочую копию — локальное отражение файлов и каталогов хранилища. После этого пользователи работают одновременно и независимо друг от друга, изменяя свои личные копии.

**3. Какую модель использует TortoiseSVN по умолчанию? По каким причинам эта модель лучше?**

По умолчанию TortoiseSVN использует модель копирование-изменение-слияние. Причиной того то, что пользователи могут работать одновременно, не тратя время на ожидание других. Обычно оказывается, что большинство одновременно вносимых изменений в файл вообще не пересекается; конфликты бывают редко. И время, потраченное на их разрешение, значительно меньше времени, отнимаемого блокировками системы.

**4. В чем отличие команд «Обновить» и «Фиксировать»? Что делает команда «Слить»?**

«Обновить» - обновляет содержимое локальной копии, а «Фиксировать» - отправляет все изменения локальной копии в репозиторий. «Слить»- включает файл/папку в локальную копию проекта.

5. Укажите и кратко опишите этапы стандартного workflow при работе с локальным репозиторием с помощью распределенной системы контроля версий git. Команды commit, checkout, branch, reset, add, status, diff, clone, push, pull.

Этапы стандартного workflow:

* Из ветки main создается ветка develop.
* Из ветки develop создается ветка release.
* Из ветки develop создаются ветки feature.
* Когда работа над веткой feature завершается, она сливается в ветку develop.
* Когда работа над веткой release завершается, она сливается с ветками develop и main.
* Если в ветке main обнаруживается проблема, из main создается ветка hotfix.
* Когда работа над веткой hotfix завершается, она сливается с ветками develop и main.

Команда git commit берёт все данные, добавленные в индекс с помощью git add, и сохраняет их слепок во внутренней базе данных, а затем сдвигает указатель текущей ветки на этот слепок.

Команда git checkout используется для переключения веток и выгрузки их содержимого в рабочую директорию.

Команда git branch - это своего рода “менеджер веток”. Она умеет перечислять ваши ветки, создавать новые, удалять и переименовывать их.

Команда git reset используется в основном для отмены изменений

Команда git add добавляет содержимое рабочей директории в индекс (staging area) для последующего коммита.

Команда git status показывает состояния файлов в рабочей директории и индексе: какие файлы изменены, но не добавлены в индекс; какие ожидают коммита в индексе.

Команда git diff используется для вычисления разницы между любыми двумя Git деревьями

Команда git push используется для установления связи с удалённым репозиторием, вычисления локальных изменений отсутствующих в нём, и собственно их передачи в вышеупомянутый репозиторий.

Команда git pull работает как комбинация команд git fetch и git merge, т.е. Git вначале забирает изменения из указанного удалённого репозитория, а затем пытается слить их с текущей веткой.