Домашняя работа 6. Система контроля версий

21.02.2022

Дедлайн: 10:00 07.03.2022
Баллов: 10

В командах по 3 человека требуется спроектировать систему контроля версий, представляющую из себя консольное приложение и умеющую:

- commit с commit message, датой коммита и автором;
- работу с ветками: создание и удаление;
- checkout по имени ревизии или ветки;
- log — список ревизий вместе с commit message в текущей ветке;
- merge — сливает указанную ветку с текущей;
  - должен быть предусмотрен механизм разрешения конфликтов;
- работа с удалёнными репозиториями: clone, fetch/pull, push.

При этом код системы должен позволять себя использовать как библиотеку, но предполагается также наличие консольного интерфейса.

Что надо сделать:

- диаграмму компонентов и диаграмму/диаграммы классов;
  - можно сделать одну большую диаграмму, где отображены одновременно и компоненты, и классы, а можно по одной диаграмме классов для каждого компонента, но помните о необходимости явно показать взаимосвязи между компонентами — то есть, на одной диаграмме рисовать классы с других диаграмм, без атрибутов и операций;
- сдавать, как обычно, пуллреквестом в репозиторий одного из членов команды, либо исходника с диаграммой, либо .md-файла со ссылкой на проект в облачном редакторе;
  - не забудьте расшарить;
  - не забудьте указать, кто в команде;
- текстовое описание не требуется, поясняйте непонятные моменты в комментариях на диаграмме;
  - однако любая неоднозначность и непонятность будет трактована не в вашу пользу;
- нельзя подсматривать в Git Book и другую архитектурную документацию систем контроля версий (там всё написано, хотя конкретно git нельзя назвать примером хорошей архитектуры, и у нас про это ещё отдельная пара будет).

Ожидается выбор архитектурного стиля и высокоуровневой структуры приложения. Решения в духе «куча классов, хаотично соединённых стрелками» высокого балла не получат. Даже «куча компонентов, хаотично соединённых стрелками» не пойдёт, разделение на модули должно отражать архитектурный замысел, и он должен быть понятен из диаграммы (и/или пояснён комментарием).

Обратите внимание на следующие вещи.

- Как представляются файлы, коммиты, ветки, репозиторий?
- Как выполняется компрессия и выполняется ли вообще? Насколько просто получить текущую, предыдущую, произвольную версии?
- Каков жизненный цикл файла?
- Как выполняется работа с файловой системой?
- Как выполняется работа с пользователем? Как представляются команды?
- Как выполняется работа с сервером со стороны клиента?
- Какова архитектура серверной части?

Эту систему реализовывать будет не надо.