Методы и средства сборки и развертывания ПО, 25

ФИО преподавателя: Смирнов Михаил Вячеславович

e-mail: smirnovmgupi@gmail.com

Лекция 3

Управление версиями и конфигурациями цифровых продуктов. Определение SCM.


Наше кредо при релизе:
Sux, bugs & plug'n'pray!

Знакомство с цифровым управлением версиями продуктов

- **Что это?** Система правил и инструментов для контроля любых изменений в цифровом продукте (код, дизайн, документация).
- **Зачем нужно?** Чтобы работать в команде, не мешая друг другу, и всегда иметь возможность вернуть рабочую версию.
- **Результат:** Порядок, предсказуемость и надежность на всех этапах создания продукта.

История управления версиями

Локальные системы (1980-е)

- RCS, SCCS
- Работа в одиночку, история на одном компьютере

Централизованные системы (1990-е)

- CVS, Subversion (SVN)
- Командная работа с общим сервером — «единая правда»

Распределенные системы (2000-е →)

- Git, Mercurial
- У каждого — полная копия проекта. Скорость и надежность!

Определение SCM: основы

Software Configuration Management - это дисциплина, которая обеспечивает:

- ***идентификацию*** **-** что именно мы контролируем (код, документация, конфиги)
- ***управление изменениями*** **-** процесс внесения и согласования правок
- ***учет состояния*** **-** фиксация версий и их характеристик
- ***аудируемость*** **-** полная история: кто, что, когда и почему изменил

Главная цель: предсказуемость и качество разработки.

Ключевые понятия SCM

Конфигурация	Версия	Релиз	Среда
• Составные части продукта (код, базы данных, документация)	• Конкретное состояние конфигурации в момент времени	• Версия, переданная пользователям	• Разработка, тестирование, продакшен

Задачи SCM

01

Хранит всю историю изменений каждого файла

02

Позволяет вернуться к любой предыдущей версии

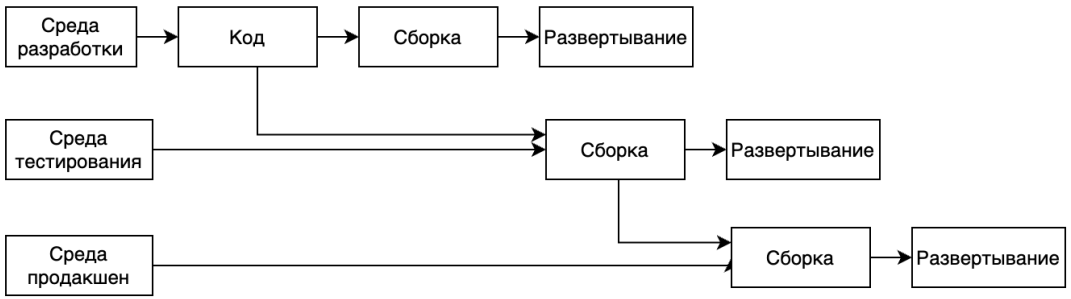
03

Показывает: кто, что и когда изменил

04

Позволяет работать над разными функциями параллельно

Задачи SCM, управление средами и конфигурациями



Управление конфигурацией: Обеспечивает, чтобы на всех этапах использовались правильные версии всех компонентов.

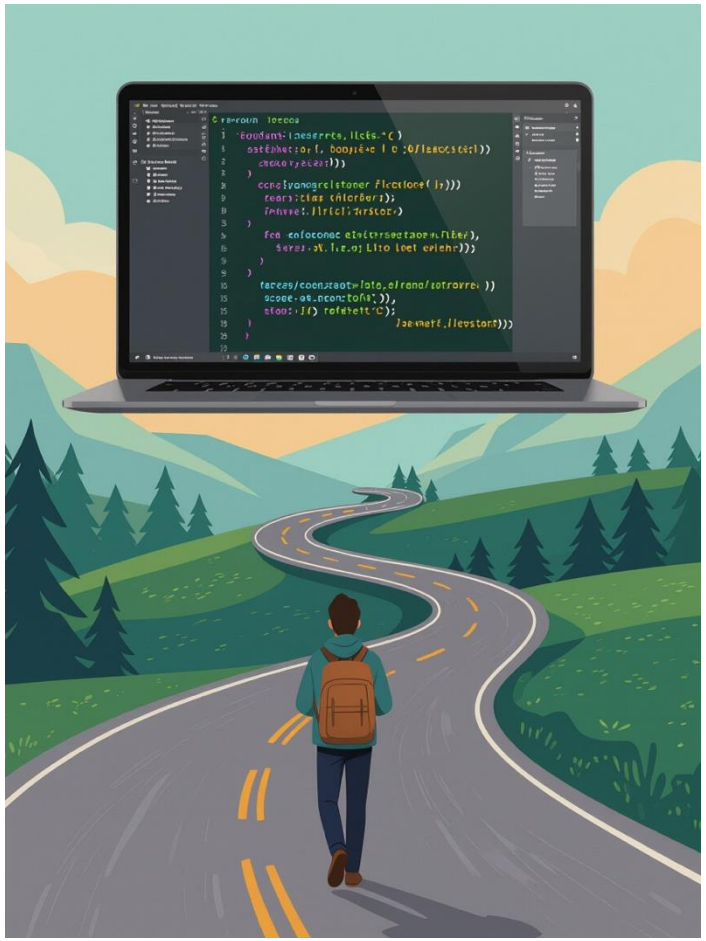
Управление средами: Отделяет среду разработки от тестовой и рабочей, чтобы избежать конфликтов.

Локальный SCM: личное путешествие

Основной принцип (RCS (Revision Control System):
Система хранит историю изменений в **скрытой папке** на том же компьютере, где ведется работа.

Как это работает:

- Вы работаете с файлом project.txt
- После значимых изменений фиксируете версию командой ci -l project.txt
- Система сохраняет разницу между версиями в локальную базу (папка .rcs)
- Для просмотра истории используете команду rlog project.txt



Локальный SCM: личное путешествие

Ключевые характеристики:

✅ **Простота:** Минимум команд и понятная структура

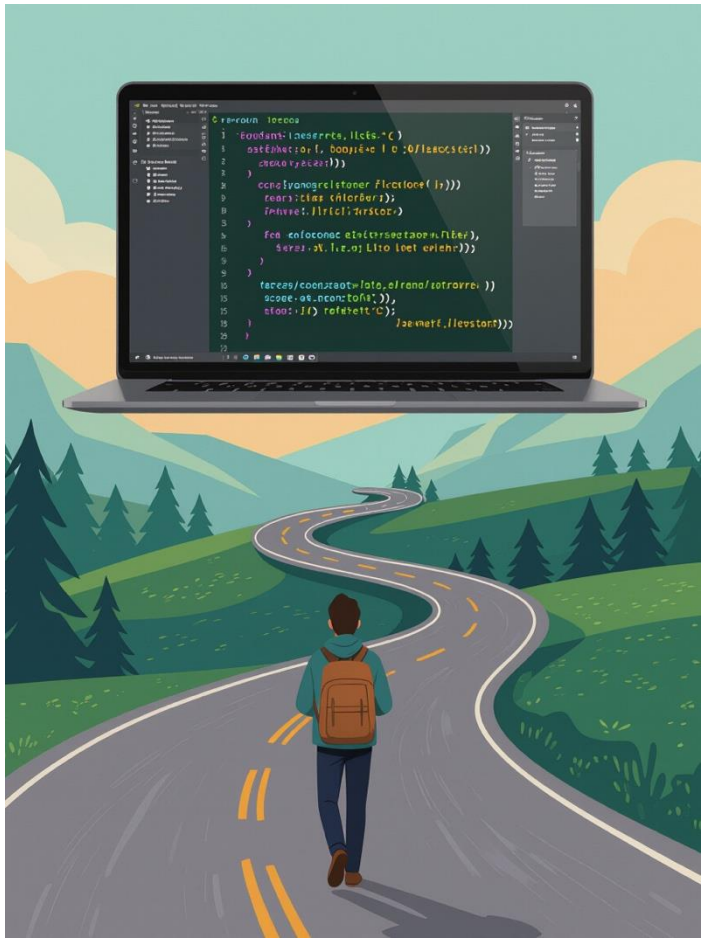
✅ **Скорость:** Все операции локальные, не нужен интернет

✅ **Легкость освоения:** Идеально для индивидуальной работы

❌ **Риск потери:** Если сгорит диск — пропадет и проект, и его история

❌ **Нет командной работы:** Нельзя работать над проектом вместе

❌ **Ограниченность:** Только линейная история, нет ветвления



Централизованная SCM: подход сообщества

Основной принцип:

Есть единый центральный сервер, который хранит официальную историю версий проекта. Каждый разработчик работает со своей «рабочей копией» файлов.

Как это работает на практике:

Разработчик А делает checkout → получает файлы с сервера

Разработчик Б делает checkout → получает те же файлы

Оба работают локально со своими копиями

Разработчик А делает commit → отправляет изменения на сервер

Разработчик Б делает update → получает изменения от А

Разработчик Б делает commit своих изменений

Централизованная SCM: подход сообщества

Ключевые характеристики:

✅ **Единая версия:** Все знают, где «официальная» версия проекта

✅ **Контроль доступа:** Администратор может управлять правами

✅ **Полная история:** Вся история хранится в одном месте

✅ **Лучше, чем локальные:** Реальная командная работа возможна

❌ **Единая точка отказа:** Сервер упал — работа остановилась

❌ **Риск потери истории:** Если сервер умрет без бэкапа — всё пропало

❌ **Требует сеть:** Каждый коммит требует подключения к серверу

❌ **Конфликты коммитов:** Одновременная работа над одними файлами проблематична

Распределенная SCM: свобода выбора

Революционный принцип:

Каждый разработчик имеет на своем компьютере **полную копию всего репозитория**, включая всю историю изменений, все ветки и теги.

Как это работает на практике:

Разработчик А делает git clone → получает полную копию проекта

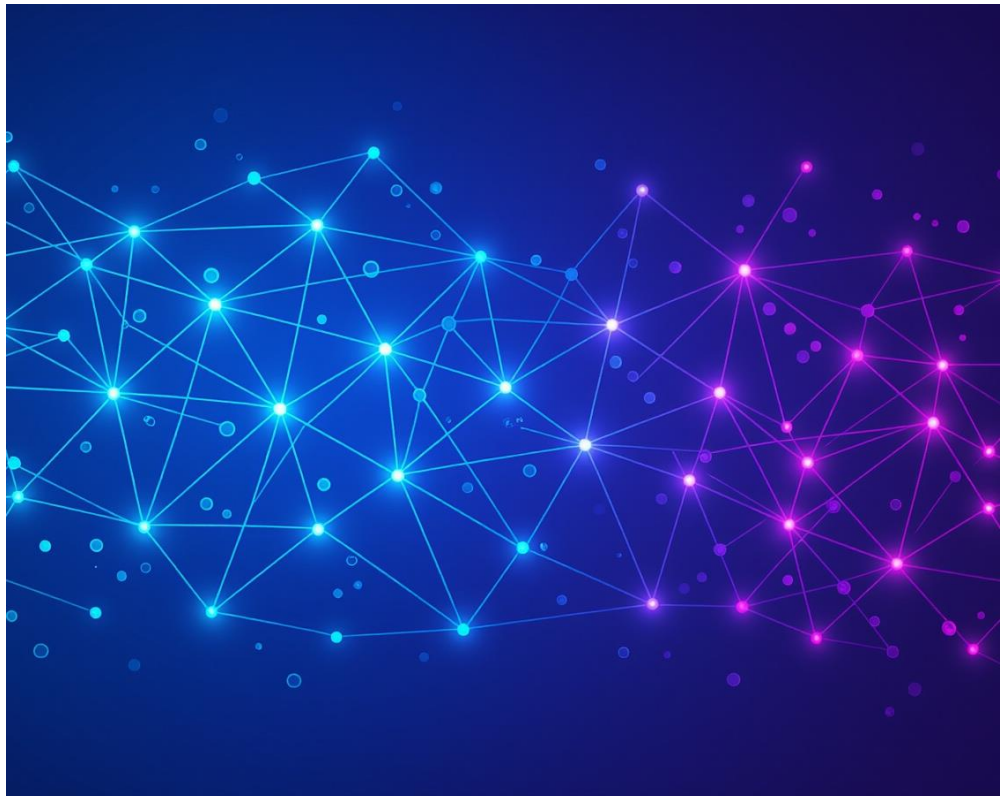
Разработчик Б делает git clone → тоже получает полную копию

Оба работают **локально, делают коммиты в свои репозитории**

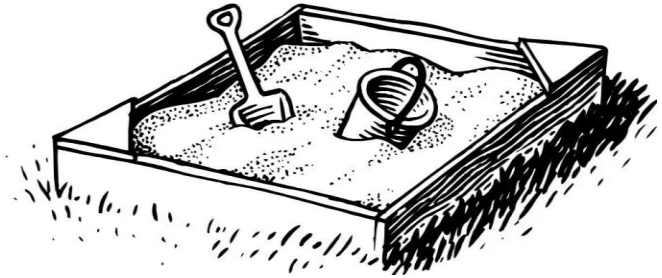
Разработчик А делает git push → отправляет изменения на удаленный сервер

Разработчик Б делает git pull → забирает изменения от А к себе

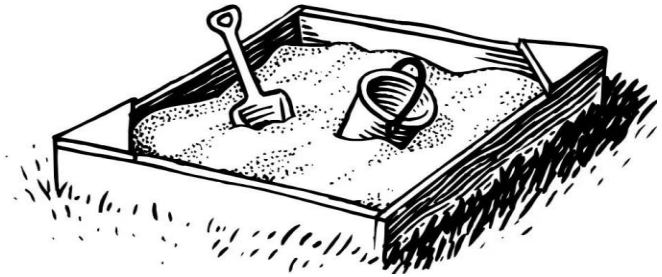
Разработчик Б продолжает работу, имея актуальную версию



Распределенная SCM: «Ветвление» и «Слияние»



ПРОЕКТ



Распределенная SCM: причины создания ветвей

• Физическая. Ветвится физическая конфигурация системы, ветви создаются для файлов, компонентов и подсистем

• Функциональная. Ветви создаются для средств, логических изменений, устранения дефектов, расширения системы и других сущностей, затрагивающих поставку функциональности (например, обновлений, релизов, продуктов и т.п.)

• Связанные со средами. Ветви создаются для модификации разных аспектов платформы сборки или выполнения (компиляторов, оконных систем, библиотек, оборудования, операционных систем и т.п.). Иногда ветвь создается для всей платформы.

• Организационная. Ветвление рабочих усилий команды. Ветви создаются для задач, проектов, ролей и групп.

• Процедурная. Ветвление рабочего поведения команды. Ветви создаются для поддержки политик, процессов и состояний.

Распределенная SCM: свобода выбора

Ключевые характеристики:

✅ **Надежность:** Потеря сервера не страшна - у каждого есть полная копия

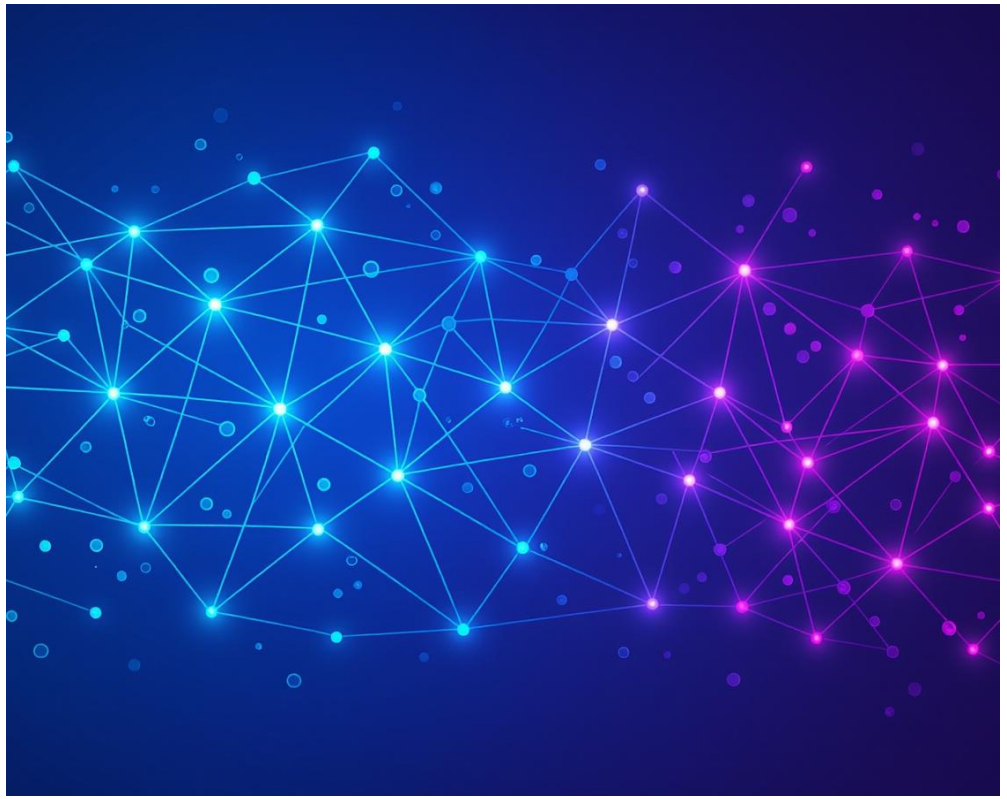
✅ **Скорость:** Все операции (кроме push/pull) локальные, мгновенные

✅ **Гибкость:** Можно работать оффлайн, экспериментировать в ветках

❌ **Сложность:** Больше команд и концепций для изучения

❌ **Размер репозитория:** Локальная копия содержит всю историю

❌ **Конфликты слияния:** Требуют ручного разрешения



Сравнительная таблица вариантов SCM систем

Критерий	Локальные	Централизованные	Распределенные
Работа в команде	Нет	Да	Да
Нужен сервер	Нет	Да (обязательно)	Да (для синхронизации)
Скорость	Высокая	Зависит от сети	Очень высокая (локально)
Надежность	Низкая	Низкая (единая точка отказа)	Очень высокая
Сложность	Низкая	Средняя	Высокая

Чтение на дом

- Д. Хамбл, Д. Фарли, Непрерывное развертывание ПО, стр. 368-398.

Спасибо за внимание!