Современные методы коллективной разработки программного обеспечения

Системы управления версиями

Проблема контроля версий является одной из фундаментальных проблем программной инженерии в силу:

1)Невозможности практической реализации линейной стратегии разработки программных средств

2)Коллективного характера процесса разработки

Даже при индивидуальной работе над проектом разработчик вынужден хранить две и более версий системы, чтобы иметь возможность вернуться на предыдущие стадии разработки

«Ручная» работа с несколькими версиями весьма утомительна и непродуктивна.

При коллективной разработке ситуация существенно усложняется, посклько возникают проблемы:

- семантической совместимости фрагментов программного кода, созданных отдельными разработчиками

- оповещения об изменениях, вносимых в ранее написанный код каждым из разработчиков.

Следствием этого стала перегрузка отдельных разработчиков и их групп проектами, исходным кодом и документацией

Стала очевидной необходимость пересмотра подходов к организации процесса создания ПС

Одним из аспектов нового подхода к организации разработки стало использование систем управления версиями – Version Control Systems (MCS)

Важнейшей частью любой системы контроля версий является репозиторий – хранилище данных, с которыми ведется работа.

Как правило, эти данные хранятся в репозиториях в виде файлов.

В дальнейшем наряду с термином «репозиторий» мы будем использовать его русский эквивалент «хранилище

Под проектом понимается совокупность файлов, включающая:

- исходные тексты на различных языках программирования

- исполняемые, ресурсные и библиотечные файлы, необходимые для сборки программного продукта

-исходные тексты файлов отправки

- сценарии программ инсталляции

- сопроводительная документация проекта

В репозиториях VCS могут храниться данные из нескольких проектов.

Система контроля версий (СКВ) – это система, регистрирующая изменения в одном или нескольких файлах с тем, чтобы в дальнейшем была возможность вернуться к определенным старым версиям этих файлов

VCS предоставляет следующие возможности:

- поддержка хранения файлов в репозитории

- поддержка истории версий файлов в репозитории

- нахождение конфликтов при изменении исходного кода и обеспечение синхронизации при работе в многопользовательской среде разработки

- отслеживание авторов изменений

Понятие «версия» в разных системах трактуется двумя различными способами

- одни системы (CVS)

Проект, в целом, номера версии не имеет, а состоит из множества файлов с разными номерами версий.

Для других систем понятие «версия» относится не к отдельному файлу, а к репозиторию целиком.

Номера версии отдельного файла здесь, фактически, не существует

Поэтому применительно к таким системам контроля версий под версией файла понимается та версия репозитория, в которой этот файл был изменен в последний раз.

Классификация

Централизованные/распределенные – в централизованных системах контроля версий вся работа производится с центральным репозиторием, в распределенных – у каждого разработчика есть локальная копия репозитория

Блокирующие/не блокирующие – блокирующие системы контроля версий позволяют наложить запрет на изменение файла, пока один из разработчиков работает над ним, в неблокирующих один файл может одновременно изменяться несколькими разработчиками.

Для текстовых данных/для бинарных данных – для VCS для текстовых данных очень важна поддержка слияния изменений, для VCS с бинарными данными важна возможность блокирования

Ежедневный цикл работы

Обычный цикл работы разработчика выглядит следующим образом:

Обновление рабочей копии.

- Разработчик выполняет операцию обновления рабочей копии (update) насколько возможно

Модификация проекта

- Разработчик локально модифицирует проект, изменяя входящие в него файлы в рабочей копии

Фиксация изменений

Завершив очередной этап работы над заданием разработчик фиксирует (commit) свои изменения, передавая их на сервер. VCS может требовать от разработчик перед фиксацией выполнить обновление.

Working copy – рабочая (локальная) копия документов

Repository depot – хранилище

Revision – версия документа. Новые изменения (changeset) создают новую ревизию репозитория

Check-in, commit, submit – фиксация изменений

Check-out, clone – извлечение документа из хранилища и создание рабочей копии

Update, sync – синхронизация рабочей копии до некоторого заданного состояния хранилища (в.т.ч и к более старому состоянию, чем текущее).

Merge, integration – слияние независимых изменений

Conflict – ситуация, когда несколько пользователей сделали изменения одного и того же участка документов

Origin – имя главного сервера

Ветвь (branch) – направление разработки проекта, независимое от других

Ветвь представляет собой копию части (как правило одного каталога) хранилища, в которую можно вносить свои изменения, не влияющие на другие ветви

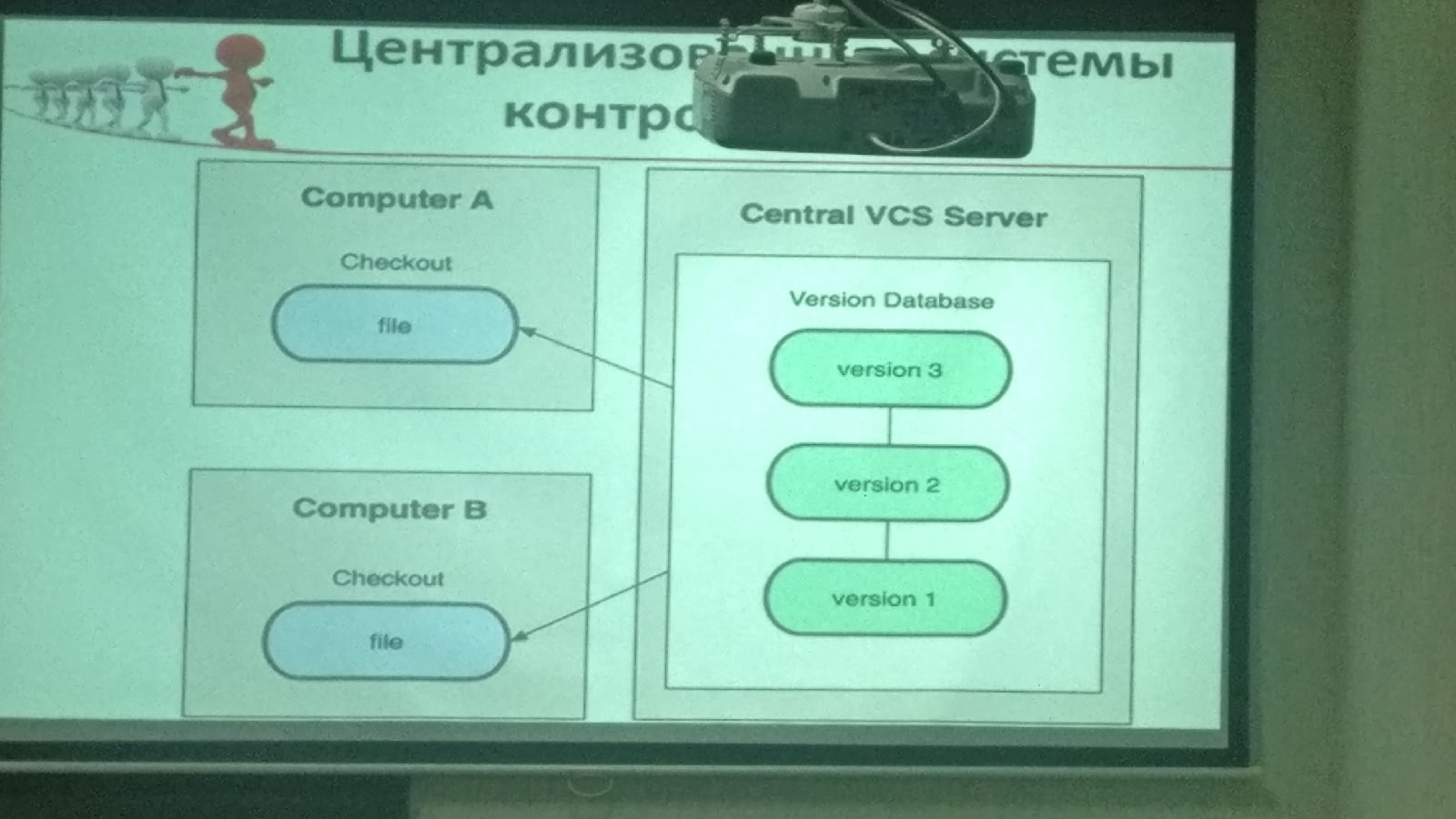
Документы в разных ветвях имеют одинаковую историю до точки ветвления и разные – после нее

Изменения из одной ветви можно переносить в другую

Ствол (trunk, mainline, master) – основная ветвь разработки проекта.



Централизованные системы контроля версий



Распределение системы контроля версий





Принцип работы централизованной системы контроля версий

1. Последняя версия скачивается
2. Модифицируется
3. Измененная версия загружается на сервер

Достаточно скопировать только те изменения, которых нет в локальном проекте пользователя, таким образом каждый пользователь имеет свой личный репозиторий

Можно создавать ветки контроля версий проекта и отслеживать сложные изменения

Git – распределенная система контроля версий, разработанная Линусом Торвалдсом. Изначально Git предназначалась для использования в процессе разработки ядра Linux, по позже стала использоваться и во многих других проектах

На данных момент Git является самой быстрой распределенной системой, использующей самое компактное хранилище ревизий.

Mercurial – распределенная система, написанная на языке Python с несколькими расширениями на C

Из использующих Mercurial проектов можно назвать такие, как, Mozilla и MoinMoin

Bazaar – система, разработка которой поддерживается компанией Canonical – известной своими дистрибутивом Ubuntu и сайтом nauchpad. Система в основном написана на языке Python и используется такими проектами, как, например,MySQL

Codeville – написання на Python распределенная система использующая инновационный алгоритм объединения алгоритм объединения изменений (merge). Система используется, например, при разработке оригинального клиента BitTorrent.