МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное

Учреждение высшего образования

**«Гжельский государственный университет»** (ГГУ)

Колледж ГГУ

Специальность 09.02.07 Информационные системы и программирования

**Реферат**

**По дисциплине «Моделирование и анализ программного обеспечения»**

**на тему «Системы контроля версий программного обеспечения»**

ВЫПОЛНИЛА:

Студентка группы ИСП-О-17

Янгабозова Анастасия

ПРОВЕРИЛА:

Прокуронова А.Ю.

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

п. Электроизолятор

2019 г.

**Системы контроля версий**

**Система контроля версий** – это система, регистрирующая изменения в одном или нескольких файлах с тем, чтобы в дальнейшем была возможность вернуться к определённым старым версиям этих файлах.

**Функции системы:**

* Хранение нескольких версий одного и того же файла;
* хранение истории разработки;
* при необходимости возвращение к более ранним версиям документа;
* определение, кто и когда сделал изменение;
* совмещение изменений сделанных разными разработчиками;
* реализация альтернативных вариантов проекта.

**Области применения систем контроля версий**

Области, в которых ведётся работа с большим количеством непрерывно изменяющихся электронных документов:

* В инструментах конфигурационного управления (Software Configuration Management Tools).
* В САПР, обычно в составе систем управления данными об изделии (Product Data Management (PDM)).

**Типы СКВ**

* Локальные (RCS)
* Централизованные (CVS, Subversion и Perforce)
* Децентрализованные (Git, Mercurial, Bazaar, Darcs)
* Блокирующие - позволяют наложить запрет на изменение файла, пока один из разработчиков работает над ним.
* Не блокирующие - один файл может одновременно изменяться несколькими разработчиками.
* Для текстовых данных.
* Для бинарных данных (важна возможность блокировки).

**Локальные системы контроля версий**

Основываются на простой базе данных, в которой хранятся изменения нужных фалов.

**Централизованные системы контроля версий**

Позволяют сотрудничать разработчиками.

Есть центральный сервер, на котором хранятся все файлы под версионным контролем, и ряд клиентов, которые взаимодействуют с этим сервером.

*Преимущества:*

* все знают, кто и чем занимается в проекте;
* у администраторов есть чёткий контроль над тем, кто и что может делать

*Недостатки:*

централизованный сервер является уязвимым местом всей системы:

* если сервер не работает, то разработчики не могут взаимодействовать, и никто не может сохранить новой версии своей работы;
* если отсутствует подключение сети у разработчика, в это время он также не может взаимодействовать с сервером;
* если же повреждается диск с центральной базой данных и нет резервной копии, вы теряете абсолютно всё — всю историю проекта, разве что за исключением нескольких рабочих версий, сохранившихся на рабочих машинах пользователей.

**Децентрализованные системы контроля версий**

В отличие от централизованных систем клиенты не просто выгружают последние версии файлов, а полностью копируют весь репозиторий.

Преимущества:

* Так как каждый раз, когда клиент забирает, свежую версию файлов, он создаёт себе полную копию всех данных, то в случае сбоев на сервере, через который шла работа, любой клиентский репозиторий может быть скопирован обратно на сервер, чтобы восстановить базу данных.
* Возможность работать с несколькими удалёнными репозиториями, таким образом, можно одновременно работать поразному с разными группами людей в рамках одного проекта. Так, в одном проекте можно одновременно вести несколько типов рабочих процессов, что невозможно в централизованных системах.

**Система контроля версий Subversion**

**Subversion** – современная система контроля версий, обладающая широким набором инструментов, позволяющих удовлетворить любые нужды для управления версиями проекта с помощью централизованной системы контроля. В Интернете множество ресурсов посвящено особенностям Subversion, что позволяет быстро и качественно решать все возникающие в ходе работы проблемы. Простота установки, подготовки к работе и широкие возможности позволяют ставить subversion на одну из лидирующих позиций в конкурентной гонке систем контроля версий.

*Достоинства:*

1. Система команд, схожая с CVS.
2. Поддерживается большинство возможностей CVS.
3. Разнообразные графические интерфейсы и удобная работа из консоли.
4. Отслеживается история изменения файлов и каталогов даже после их переименования и перемещения.
5. Высокая эффективность работы, как с текстовыми, так и с бинарными файлами.
6. Встроенная поддержка во многие интегрированные средства разработки, такие как KDevelop, Zend Studio и многие другие.
7. Возможность создания зеркальных копий репозитория.
8. Два типа репозитория – база данных или набор обычных файлов.
9. Возможность доступа к репозиторию через Apache с использованием протокола WebDAV.
10. Наличие удобного механизма создания меток и ветвей проектов.
11. Можно с каждым файлом и директорией связать определенный набор свойств, облегчающий взаимодействие с системой контроля версии.
12. Широкое распространение позволяет быстро решить большинство возникающих проблем, обратившись к данным, накопленным Интернет-сообществом.

*Недостатки:*

1. Полная копия репозитория хранится на локальном компьютере в скрытых файлах, что требует достаточно большого объема памяти.
2. Существуют проблемы с переименованием файлов, если переименованный локально файл одним клиентом был в это же время изменен другим клиентом и загружен в репозиторий.
3. Слабо поддерживаются операции слияния веток проекта.
4. Сложности с полным удалением информации о файлах попавших в репозиторий, так как в нем всегда остается информация о предыдущих изменениях файла, и непредусмотрено никаких штатных средств для полного удаления данных о файле из репозитория.

**Система контроля версий Git**

**Git** – гибкая, удобная и мощная система контроля версий, способная удовлетворить абсолютное большинство пользователей. Существующие недостатки постепенно удаляются и не приносят серьезных проблем пользователям. Если вы ведете большой проект, территориально удаленный, и тем более, если часто приходится разрабатывать программное обеспечение, не имея доступа к другим разработчикам (например, вы не хотите терять время при перелете из страны в страну или во время поездки на работу), можно делать любые изменения и сохранять их в локальном репозитории, откатываться, переключаться между ветками и т.д.).

Git – один из лидеров систем контроля версий..

*Достоинства:*

1. Надежная система сравнения ревизий и проверки корректности данных, основанные на алгоритме хеширования SHA1 (Secure Hash Algorithm 1).
2. Гибкая система ветвления проектов и слияния веток между собой.
3. Наличие локального репозитория, содержащего полную информацию обо всех изменениях, позволяет вести полноценный локальный контроль версий и заливать в главный репозиторий только полностью прошедшие проверку изменения.
4. Высокая производительность и скорость работы.
5. Удобный и интуитивно понятный набор команд.
6. Множество графических оболочек, позволяющих быстро и качественно вести работы с Git’ом.
7. Возможность делать контрольные точки, в которых данные сохраняются без дельта компрессии, а полностью. Это позволяет уменьшить скорость восстановления данных, так как за основу берется ближайшая контрольная точка, и восстановление идет от нее. Если бы контрольные точки отсутствовали, то восстановление больших проектов могло бы занимать часы.
8. Широкая распространенность, легкая доступность и качественная документация.
9. Гибкость системы позволяет удобно ее настраивать и даже создавать специализированные контроля системы или пользовательские интерфейсы на базе git.
10. Универсальный сетевой доступ с использованием протоколов http, ftp, rsync, ssh и др.

*Недостатки:*

1. Unix – ориентированность. На данный момент отсутствует зрелая реализация Git, совместимая с другими операционными системами.
2. Возможные (но чрезвычайно низкие) совпадения хеш - кода отличных по содержанию ревизий.
3. Не отслеживается изменение отдельных файлов, а только всего проекта целиком, что может быть неудобно при работе с большими проектами, содержащими множество несвязных файлов.
4. При начальном (первом) создании репозитория и синхронизации его с другими разработчиками, потребуется достаточно длительное время для скачивания данных, особенно, если проект большой, так как требуется скопировать на локальный компьютер весь репозиторий.