МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

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

Колледж ГГУ

Специальность 09.02.07 Информационные системы и программирования

**Отчет**

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

**на тему «Системы контроля версий ПО»**

ВЫПОЛНИЛ:

Студент группы ИСП-О-17

Шашков И.С.

ПРОВЕРИЛА:

Прокуронова А.Ю.

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

п. Электроизолятор

1. г.

**Общее определение:**

**Система управления версиями** (от [англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *Version Control System, VCS* или *Revision Control System*) — [программное обеспечение](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5) для облегчения работы с изменяющейся информацией. Система управления версиями позволяет хранить несколько версий одного и того же документа, при необходимости возвращаться к более ранним версиям, определять, кто и когда сделал то или иное изменение, и многое другое.

Такие системы наиболее широко используются при разработке программного обеспечения для хранения [исходных кодов](https://ru.wikipedia.org/wiki/%D0%98%D1%81%D1%85%D0%BE%D0%B4%D0%BD%D1%8B%D0%B9_%D0%BA%D0%BE%D0%B4) разрабатываемой программы. Однако они могут с успехом применяться и в других областях, в которых ведётся работа с большим количеством непрерывно изменяющихся электронных документов. В частности, системы управления версиями применяются в [САПР](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%B0%D0%B2%D1%82%D0%BE%D0%BC%D0%B0%D1%82%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D0%B8_%D0%BF%D1%80%D0%BE%D0%B5%D0%BA%D1%82%D0%BD%D1%8B%D1%85_%D1%80%D0%B0%D0%B1%D0%BE%D1%82), обычно в составе систем управления данными об изделии ([PDM](https://ru.wikipedia.org/wiki/PDM)). Управление версиями используется в инструментах [конфигурационного управления](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BD%D1%84%D0%B8%D0%B3%D1%83%D1%80%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%BE%D0%B5_%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5) (*Software Configuration Management Tools*).

**Система контроля версий Git:**

Эта система была создана для управления разработкой ядра Linux и использует подход, который в корне отличается от CVS и SVN.

В основу [**Git**](http://git-scm.com/) закладывались концепции, призванные создать более быструю распределенную систему контроля версий, в противовес правилам и решениям, использованным в CVS. Так как Git разрабатывалась главным образом под Linux, то именно в этой ОС она работает быстрее всего.

Git также работает на Unix-подобных системах (как MacOS), а для работы на платформе Windows используется пакет mSysGit.

Программный код может быть недоступен, когда используется компьютер без репозитория. Для решения этой проблемы есть обходные пути и некоторые разработчики полагают, что быстродействие Git является справедливой платой за неудобства.

Кроме того, в Git есть множество инструментов для навигации по истории изменений. Каждая рабочая копия исходного кода содержит всю историю разработки, что крайне полезно, когда программируешь без Интернет-соединения.

**Преимущества:**

* Прекрасно подходит для тех, кто ненавидит CVS/SVN
* Значительное увеличение быстродействия
* Дешевые операции с ветками кода
* Полная история разработки доступная оффлайн
* Распределенная, пиринговая модель

**Недостатки:**

* Высокий порог вхождения (обучения) для тех, кто ранее использовал SVN
* Ограниченная поддержка Windows (по сравнению с Linux)

**Система контроля версий Apache Subversion (SVN):**

[**SVN**](http://subversion.apache.org/) создавалась как альтернатива CVS с целью исправить недостатки CVS и в то же время обеспечить высокую совместимость с ней.

Как и CVS, SVN это бесплатная система контроля версий с открытым исходным кодом. С той лишь разницей, что распространяется под лицензией Apache, а не под Открытым лицензионным соглашением GNU.

Для сохранения целостности базы данных SVN использует так называемые атомарные операции. При появлении новой версии к финальному продукту применяются либо все исправления, либо ни одно из них. Таким образом, код защищают от хаотичных частичных правок, которые не согласуются между собой и вызывают ошибки.

Многие разработчики переключились на SVN, так как новая технология унаследовала лучшие возможности CVS и в то же время расширила их.

В то время как в CVS операции с ветками кода дорогостоящие и не предусмотрены архитектурой системы, SVN создана как раз для этого. То есть, для более крупных проектов с ветвлением кода и многими направлениями разработки.

В качестве недостатков SVN упоминаются сравнительно низкая скорость и нехватка распределенного управления версиями. Распределенный контроль версий использует пиринговую модель, а не  централизованный сервер для хранения обновлений программного кода. И хотя пиринговая модель работает лучше в open source проектах, она не идеальна в других случаях. Недостаток серверного подхода в том, что когда сервер падает, то у клиентов нет доступа к коду.

**Преимущества:**

* Система на основе CVS
* Допускает атомарные операции
* Операции с ветвлением кода менее затратны
* Широкий выбор плагинов IDE
* Не использует пиринговую модель

**Недостатки:**

* Все еще сохраняются ошибки, связанные с переименованием файлов и директорий
* Неудовлетворительный набор команд для работы с репозиторием
* Сравнительно небольшая скорость

**Главные различия:**

1. GIT распределяется, а SVN - нет. Другими словами, если есть несколько разработчиков работающих с репозиторием у каждого на локальной машине будет ПОЛНАЯ копия этого репозитория. Разумеется есть и где-то и центральная машина, с которой можно клонировать репозиторий. Это напоминает SVN. Основной плюс в том, что если вдруг у вас нет доступа к интернету, сохраняется возможность работать с репозиторием. Потом только один раз сделать синхронизацию и все остальные разработчики получат поолную историю.
2. GIT сохраняет метаданные изменений, а SVN целые файлы. Это экономит место и время.
3. Система создания branches, versions и прочее в GIT и SVN отличаются значительно. В GIT проще переключатся с ветки на ветку, делать merge между ними. В общем GIT я нахожу немного проще и удобнее, но бывают конечно иногда сложности. Но где их не бывает?