Лабораторная работа № 1. Subversion. Основные операции.

Цель работы: Получение навыков работы с системой контроля версий Subversion (хранилищем и рабочими копиями).

Теоретическая часть

1 Хранилище

Subversion является централизованной системой для хранения информации. Ее основа – хранилище, являющееся центром хранения данных. Оно хранит информацию в форме дерева файлов.

Любое количество клиентов может подключиться к хранилищу и прочитать или записать новые версии этих файлов. Записывая данные, клиент делает информацию доступной для остальных; читая данные, клиент получает информацию от других.

Subversion запоминает каждое внесённое изменение: любое изменение любого файла, а также изменения в самом дереве каталогов, такие как добавление, удаление и реорганизация файлов и каталогов. При чтении данных из хранилища клиент обычно видит как последнюю версию дерева файлов, так и предыдущие состояния файловой системы.

2 Правки

При фиксации версии создаётся новое состояние файловой системы – правка (ревизия). Каждая правка получает уникальный номер, на единицу больший номера предыдущей. Начальная правка только что созданного хранилища получает номер 0 и не содержит ничего, кроме пустого корневого каталога.

В служебном каталоге .svn/ для каждого файла рабочего каталога Subversion записывает информацию о двух важнейших свойствах:

- на какой правке основан рабочий файл (рабочая правка файла);
- временной метке, определяющей, когда рабочая копия последний раз обновлялась из хранилища.

Используя эту информацию при соединении с хранилищем, Subversion может сказать, в каком из следующих четырех состояний находится рабочий файл:

1) Файл не изменялся и не устарел.

Файл не изменялся в рабочем каталоге, в хранилище не фиксировались изменения этого файла со времени создания его рабочей правки. Команды svn commit и svn update никаких операций производить не будут;

2) Файл изменялся локально и не устарел.

Файл был изменен в рабочей копии, но в хранилище не фиксировались его изменения. Есть локальные изменения, которые не

были зафиксированы в хранилище, поэтому svn commit выполнит фиксацию ваших изменений, а svn update не сделает ничего;

3) Файл не изменялся и устарел.

В рабочем каталоге файл не изменялся, но был изменен в хранилище. Необходимо выполнить обновление файла для того, чтобы он соответствовал текущей правке. Команда svn commit не сделает ничего, а svn update обновит рабочую копию файла;

4) Файл изменялся локально и устарел.

Файл был изменен как в рабочем каталоге, так и в хранилище. Команда svn commit выдаст ошибку «out-of-date». Файл необходимо сначала обновить; svn update попытается объединить локальные изменения с опубликованными. Если Subversion не сможет совершить объединение, она предложит пользователю разрешить конфликт вручную.

Фундаментальные правила Subversion – «передающее» действие не приводит к «принимаемому», и наоборот. То, что вы готовы внести изменения в хранилище, не означает, что вы готовы принять изменения от других. А если вы все еще работаете над новыми изменениями, то svn update объединит изменения из хранилища с вашими собственными вместо того, чтобы заставить вас опубликовать их.

3 Простейший рабочий цикл

Типичный рабочий цикл с применением Subversion выглядит примерно так:

1) обновление рабочей копии:

svn update

2) внесение изменений:

svn add (delete, copy, move)

3) редактирование файлов под контролем Subversion;

4) анализ изменений:

svn status (diff, revert)

5) слияние изменений, выполненных другими, с вашей рабочей копией:

svn update

svn resolved

6) фиксация изменений:

svn commit

3.1 Создание репозитория

Для создания пустого репозитория необходимо выполнить команду svnadmin create (команда svnadmin поставляется со стандартным пакетом svn):

Пример 1. Создание пустого репозитория:

$ svnadmin create /path/to/rep

Здесь /path/to/rep – это URL того адреса, по которому будет создан репозиторий.

Для импортирования нового проекта в Subversion-хранилище используется svn import.

Команда svn import это быстрый способ скопировать не версионированное дерево файлов в хранилище, создавая при необходимости промежуточные директории.

Пример 2. Создание пустого репозитория в подкаталоге /usr/local/svn/newrepos, затем перенесение всего дерева каталогов в этот репозиторий (file:///usr/local/svn/newrepos/some/project):

```
$ svnadmin create /usr/local/svn/newrepos
$ svn import mytree file:///usr/local/svn/newrepos/some/project \
-m "Initial import"
Adding mytree/foo.c
Adding mytree/bar.c
Adding mytree/subdir
Adding mytree/subdir/quux.h
Committed revision 1.
```

В предыдущем примере выполняется копирование содержимого директории mytree в директорию some/project хранилища.

3.2 Создание рабочей копии репозитория

Для создания рабочей копии используется команда svn checkout

Для того чтобы создать рабочую копию уже существующего репозитория, вам нужно получить какую-либо из подкаталогов хранилища.

Рабочая копия представляет собой обычное дерево каталогов на компьютере.

После изменения файлов рабочей копии нужно «опубликовать» изменения, в результате чего они станут доступными для всех участников проекта.

Рабочая копия содержит дополнительные файлы в подкаталоге .svn. Они помогают определить файлы рабочей копии, содержащие неопубликованные изменения, и файлы, устаревшие по отношению к файлам других участников.

Пример 3. Создание рабочей копии репозитория http://svn.example.com/repos/calc, загружается в текущую директорию:

```
$ svn checkout http://svn.example.com/repos/calc
A calc/Makefile
A calc/integer.c
```

A calc/button.c

Checked out revision 56.

3.3 Внесение изменений в рабочую копию

Изменения, которые можно сделать в рабочей копии:

− изменения файлов;

− изменения в структуре каталогов.

Подкоманды, наиболее часто используемые при внесении изменений:

− svn add <file> – запланировать для добавления в хранилище. При фиксации <file> станет компонентом своей родительской директории;

− svn delete <file> – запланировать удаление из хранилища;

− svn copy <file1> <file2> – создать элемент <file2> как копию <file1>;

− svn move <file1> <file2> – <file2> будет запланирован для добавления как копия <file1>, а <file1> будет запланирован для удаления.

3.4 Фиксация (публикация) изменений в хранилище

Публикация (коммит) представляет собой «фиксирование» данных в репозитории. Для публикации изменений следует воспользоваться командой commit. Можно использовать ключ –m для указания поясняющего сообщения в одной строке:

Пример 4. Фиксация изменений с сообщением «First message»:

$ svn commit -m "First message."

Sending

button.c

Transmitting file data

Committed revision 2.

3.5. Просмотр истории изменений

Svn log показывает вам развернутую информацию: лог сообщения, присоединенные к правкам, с указаной датой изменений и их автором, а также измененные в правке пути файлов.

Пример 5. Команда svn log:

```
$ svn log
------------------------------------------------------------------------
r3 | sally | Mon, 15 Jul 2002 18:03:46 -0500 | 1 line
Added include lines and corrected # of cheese slices.
------------------------------------------------------------------------
r2 | harry | Mon, 15 Jul 2002 17:47:57 -0500 | 1 line
Added main() methods.
------------------------------------------------------------------------
r1 | sally | Mon, 15 Jul 2002 17:40:08 -0500 | 1 line
Initial import
```

Обратите внимание на то, что по умолчанию лог сообщения выводятся в обратном хронологическом порядке. При необходимости порядок вывода информации об истории изменений можно изменить.

3.5. Другие полезные команды

svn help

Клиент для командной строки Subversion является самодокументируемым — в любой момент команда svn help <subcommand> покажет описание синтаксиса, параметров и поведения подкоманды subcommand.

svn info

Позволяет просматривать информацию о репозитории, такую как URL родительского репозитория, номер текущей ревизии, дату последних изменений.

svn status

Служит для того, чтобы узнать состояние любого элемента в вашей рабочей копии. Можно использовать перед фиксацией для просмотра изменений, которые войдут в следующую фиксацию.

Метки

Можно считать, что метки — это специальные обозначения, прикреплённые к коммиту. С помощью них удобно искать коммиты, в которых приложение было в определённом состоянии (например, релиз 1.0, релиз 1.2 и т.д.).

Для меток в Subversion нет специальной команды, они выглядят как дешевые копии (cheap copies), или ссылки:

Пример 6. Создание метки 1.0 в директории tags и сообщением "Release 1.0":

```
svn copy http://svn.example.com/project/trunk \
        http://svn.example.com/project/tags/1.0 -m "Release 1.0"
```

Практическая часть

Используя клиент Subversion необходимо проделать, задокументировать и отразить в отчете следующие задания:

1. Создать репозиторий в любой выбранной пустой директории.
2. Создать рабочую копию в любой выбранной пустой директории.
3. Проверить, на какое хранилище ссылается созданная рабочая копия.
4. Просмотреть последнюю дату изменения файлов в репозитории.
5. Создать три текстовых файла (f1.txt, f2.txt, f3.txt) с несколькими содержательными строками внутри каждого из них.

6. Добавить в отслеживаемые все созданные файлы.
7. Просмотреть перед фиксацией сделанные изменения.
8. Зафиксировать изменения с любым осмысленным сообщением.
9. Удалить файл f3.txt и зафиксировать его удаление с любым осмысленным сообщением.
10. Посмотреть номер текущей ревизии.
11. Посмотреть историю коммитов, задокументировать.
12. Совершить еще 2 изменения и коммита, поставить на одном из них метку «Лабораторная_1», просмотреть полученные результаты.

Содержание отчёта

По результатам выполнения работы оформляется отчет в соответствии с требованиями ГОСТ 7.32-2017 «Отчет о научно-исследовательской работе. Структура и правила оформления», включающий:

- титульный лист;
- цель работы;
- описание структуры хранилища во время выполнения (при выполнении операций, меняющих состояние хранилища);
- выполняемые команды с комментариями и результаты их выполнения;
- выводы.

Контрольные вопросы:

1. Опишите жизненный цикл коммитов в Svn.
2. Что представляет из себя коммит?
3. Как хранятся изменения между различными файлами в Svn?
4. Зачем создаётся каталог .svn в каждой директории?
5. Перечислите известные вам достоинства и недостатки Subversion.
6. После операции svn update имеем A file1. Поясните значение данного статуса.
7. Откуда берется копия файла для замены при осуществлении команды svn revert?
8. Опишите быстрый способ скопировать неверсионированное дерево в хранилище.
9. Действия, вызванные svn commit и svn update, при условии, что файл изменялся локально и устарел.
10. Перечислите основные команды subversion для внесения изменений в рабочую копию.

11. По каким протоколам можно получить доступ к хранилищу Subversion?

12. В чём отличие модели «блокирование – изменение – разблокирование» от «копирование – изменение – слияние»?

13. В каком порядке выводится история коммитов при вызове команды svn log?

14. Как в Subversion обозначается последняя (самая новая) правка хранилища, номер правки элемента рабочей копии, правка, в которой элемент последний раз редактировался, и правка, предшествующая последней правке?

15. Какие два типа игнорирования файлов существуют в Subversion? Чем они отличаются?

16. Опишите назначение игнорирования файлов?