**ВВЕДЕНИЕ**

С приходом более совершенных IDE и языков программирования, вопрос взаимодействия с коллегами является ключевым для многих разработчиков. В основном чтобы решить эту проблему, начали появляться централизованные системы контроля версий (ЦСВ или CVSs – Centralized Version Control systems). Примерами таких систем являются:

− CVS;

− Subversion;

− Perforce.

В них используется следующий метод сохранения информации: имеется один сервер, который хранит все версии файлов, а также некоторое количество клиентов, на которые поставляется информация с сервера.

Такой подход имеет много преимуществ, особенно относительно RCS. RCS (Revision Control System) является одной из самых первых (разработана в 1985 году) систем управления версиями. Для каждого файла, зарегистрированного в системе, она хранит полную историю изменений, причём для текстовых файлов используется эффективный алгоритм дельта-компрессии, когда хранится только последняя версия и все межверсионные изменения. Система позволяет также хранить версии бинарных файлов, но без использования этого механизма, то есть каждая версия бинарного файла хранится полностью.

Основное преимущество CVSs над RCS в том, что разработчики могут работать с одним и тем же файлом одновременно. Кроме этого, есть ещё и другие плюсы, например, каждый разработчик в любой момент времени может знать, что делают все остальные; кроме этого, администраторы систем, имея точно такую же возможность, могут эффективнее управлять своими кадрами и всем продуктом в целом.

Но, несмотря на решение достаточно большого спектра проблем, у таких систем есть свои недостатки. Первый и наиболее существенный – единая точка отказа. Если сервер по каким-то причинам выходит из строя, то все работают только со своими локальными копиями, не имея возможности смотреть изменения других разработчиков.

Чтобы решить эту проблему, и, в свою очередь забыть о проблемах RCS, существуют децентрализованные системы контроля версий (ДСКВ или DVCS – Distributed Version Control System). В качестве примеров можно привести следующие ДСКВ:

− Git;

− Mercurial;

− Darcs;

− Bazaar.

Как следует из названия, в таких системах клиенты не просто забирают последнюю (или выбранную, но всегда одну) версию проекта: на самом деле они полностью копируют его содержание и историю. Таким образом, при таком подходе у системы появляются два больших преимущества перед VCS:

− В любой момент времени разработчик может посмотреть состояние проекта в какой-то момент времени даже без наличия доступа к центральному серверу;

− Каждая копия проекта это, по сути дела, полная резервная копия проекта. Таким образом, если сервер выйдет из строя, то любой разработчик сможет восстановить его состояние на любой другой машине.

На предприятиях Республики Беларусь используется достаточно много различных версий VCS. Это объясняется спецификой заказов (а именно необходимостью поддерживать различные устаревшие проекты). Тем ни менее, основную часть VCS составляют mercurial и git-системы. На них пишутся большинство новых приложений и проектов. Во время прохождения практики мною были изучены возможности именно git-систем в рамках сразу двух проектов. Таким образом, можно сделать вывод, что именно они являются преобладающими.

Тем ни менее, также большое количество проектов написаны с использованием svn-систем. В основном это достаточно долгосрочные проекты для крупных международных организаций, то есть те, старт которых был дал примерно в прошлом десятилетии.

На основании полученных данных, в моей дипломной работе будут использоваться git как децентрализованная система и SVN (subversion) как централизованная.

Основная причина в использовании git заключается в его популярности. Git является одним из основных инструментов, которые требуют заказчики, а также в том, что именно для работы с git написана самая популярная система для работы с файлами – GitHub. Его популярность, в свою очередь, обусловлена достаточно большим набором возможностей и команд:

− поддержка меток,

− функция сохранения временного состояния (англ. stash),

− качественный веб-интерфейс «из коробки»,

− продуманная система команд, позволяющая удобно встраивать git в скрипты.

Основная причина использования Mercurial состоит в том, что легче для понимания и изучения, особенно для разработчиков, для который первой системой контроля версий был SVN и подобные. Также Mercurial исторически лучше поддерживает Windows, в нём у начинающего разработчика будет меньше проблем с настройкой.

Основные причины использования Subversion – наследие и legacy-код. Однако, существуют и плюсы SVN относительно git:

− отслеживает версии не только файлов, но и директорий.

− атомарность многофайловых фиксаций.

− поддерживаются наборы изменений (англ. changeset).

− с каждым файлом и директорией может быть связан произвольный набор свойств, состоящих из названия и значения. Свойства тоже находятся под управлением версиями.

− одинаково эффективно работает как с текстовыми, так и с двоичными файлами.

Исходя из всего вышеперечисленного, в данной дипломной работе имеет смысл рассматривать сразу три системы, используя Mercurial в качестве промежуточного звена.