МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЧЕРКАСЬКИЙ ДЕРЖАВНИЙ БІЗНЕС-КОЛЕДЖ

***Циклова комісія програмування***

**РЕФЕРАТ**

з навчальної дисципліни: „Основи алгоритмізації та програмування ”

на тему: «Системи контролю версій»

**Виконали:**

Студенти 2 курсу

спеціальності: «ІПЗ»

групи: 2П-20

Дворяківська Єлизавета Георгіївна

Чухало Ілля Віталійович

**Перевірив викладач:**

Марченко С.В.

Черкаси 2021

**ЗМІСТ:**

Вступ

1. Поняття системи контролю версій

1.1 Розподілені системи керування версіями

2. Види систем контролю версій

2.1 CVS

2.2 Subversion

2.3 Arch

2.4 OpenCM

2.5 Aegis

2.6 Monotone

2.7 BitKeeper

2.8 Perforce

2.9 Darcs

Висновок

Список літератури.

**ВСТУП**

Сьогодні у світі, де існує безліч складних систем, існує необхідність видозміни електронних документів на різних стадіях їх розробки. За час свого існування електронний документ може бути схильний до великої кількості змін. Однак часто так буває, що для подальшої роботи потрібна не тільки остання версія документа, але й різні попередні варіанти.

Безперечно, можна зберігати кілька різних варіантів необхідного документа, але цей спосіб неефективний. Нам доводиться витрачати купу часу та сил, необхідно особливу увагу та велика ймовірність помилки. Крім того, нам доводиться зберігати величезну кількість практично ідентичних документів.

Внаслідок цього були розроблені програмні засоби, які спрощують цей механізм. Ці засоби називаються системами контролю версій. Існує кілька таких систем, кожна з яких актуальна за певних умов їх використання.

Метою даного реферату є розгляд різноманітних систем контролю версій.

Відповідно до поставленої метою необхідно вирішити такі завдання:

* визначити поняття системи контролю версій;
* проаналізувати існуючі системи контролю версій;
* озглянути основні види такого роду систем із зарубіжної та російської практики.

**1. Поняття системи контролю версій**

Системи управління версіями (Version Control System або Revision Control System) являють собою програмне забезпечення для полегшення діяльності з інформацією, що швидко змінюється. Система контролю версій надає можливість зберігати кілька варіантів одного й того самого документа. За необхідності можна повернутися до старих версій, можна дізнатися, ким було зроблено ті чи інші зміни тощо.

Такі системи в більшості своїй використовуються при розробці програмного забезпечення, щоб можна було зберігати вихідні коди програм, що розробляються. Система контролю версій дозволяє розробникам зберігати попередні версії файлів із розробки та діставати їх звідти. Вона зберігає інформацію про версію кожного файлу (і повну структуру проекту) у колекції, яка зазвичай називається репозиторієм. Проте дані системи можуть використовуватися і в інших галузях знань, які включають величезну кількість електронних документів, що часто змінюються. Наприклад, вони все частіше застосовуються в САПР, як правило, у складі систем управління даними про продукт (PDM). Керування версіями використовується у інструментах конфігураційного керування.

Усередині репозиторію може бути кілька паралельних ліній розробки, зазвичай званих гілками. Це може бути корисним для зберігання стабільної або випущеної версії гілки, одночасно продовжуючи роботу над робочою версією. Інший варіант – це відкрити виділену гілку для роботи над екпериментальною можливістю.

Система контролю версій також дозволяє користувачам дати ярлик знімку гілки (часто званих як тэги) для легкого діставання. Це корисно для позначення індивідуальних релізів або найсвіжіших робочих версій, призначених для використання.

Використання системи контролю версій безумовно обов'язково для розробника, якщо проект більше кількох сотень рядків або якщо для проекту спільно працюють кілька розробників. Використання хорошої системи контролю версій безумовно краще, ніж використання вузькоспрямованих спеціальних методів, які використовують деякі розробники для управління різними ревізіями свого коду.

Більшість систем управління версіями використовують централізовану модель, коли є єдине сховище документів, кероване спеціальним сервером, який виконує більшу частину функцій з управління версіями. Користувач, який працює з документами, повинен спочатку отримати потрібну версію документа зі сховища; зазвичай створюється локальна копія документа. Може бути отримана остання версія або будь-яка з попередніх, яка може бути обрана за номером версії або датою створення, іноді за іншими ознаками. Після того, як в документ внесено потрібні зміни, нова версія міститься у сховищі. На відміну від простого збереження файлу, попередня версія не стирається, а також залишається в сховищі і може бути отримана звідти в будь-який час. Сервер може використовувати т.з. дельта-компресію - такий спосіб зберігання документів, у якому зберігаються лише зміни між послідовними версіями, що дозволяє зменшити обсяг збережених даних.

Дуже часто над одним і тим самим проектом працює кілька людей. Якщо один з них змінюватиме вихідний файл, і одночасно з цим інша людина виконуватиме аналогічну операцію, то можлива така ситуація, що якісь зміни можуть не зберегтися. Системи контролю версій працюють з проблемами і мають певний перелік їх вирішення. Здебільшого ці системи можуть автоматично об'єднувати такі зміни, які роблять різні члени команди розробників. Але слід зазначити, що такого роду об'єднання найчастіше виконується для текстових файлів і з певною умовою: зміни відбувалися у різних частинах файлу. Це обмеження має місце, оскільки здебільшого системи контролю версій спрямовані підтримку процесу розробки програмних продуктів, а початкові коди перебувають у текстових файлах. Якщо автоматично виконати об'єднання не вийшло, то система пропонує виправити ситуацію вручну.

Найчастіше здійснити об'єднання неможливо ні з допомогою системи, ні вручну. Яскравим прикладом є ситуація, коли формат файлу дуже складний або невідомий. Окремі системи контролю версій дозволяють користувачеві заблокувати файл у сховищі. Ця операція не дозволяє іншим користувачам отримати робочу копію або перешкоджає зміні робочої копії файлу і забезпечує, таким чином, винятковий доступ лише користувачеві, який працює з документом.

Багато систем управління версіями надають ряд інших можливостей:

* Дозволяють створювати різні варіанти одного документа, т.з. гілки, із загальною історією змін до точки розгалуження та з різними – після неї.
* Дають можливість дізнатися, хто і коли додав або змінив конкретний набір рядків у файлі.
* Ведуть журнал змін, до якого користувачі можуть записувати пояснення про те, що і чому вони змінили цю версію.
* Контролює права доступу користувачів, дозволяючи або забороняючи читання або зміну даних, залежно від того, хто запитує цю дію.

**1.1 Розподілені системи керування версіями**

Існують системи управління версіями, які замість традиційної клієнт-серверної використовують розподілену модель. Такі системи, загалом, не потребують централізованого сховища: вся історія зміни документів зберігається кожному комп'ютері. Фактично кожен комп'ютер, крім робочої копії, зберігає локальну копію всього сховища. У деяких системах робоча копія сама є сховищем.

Коли користувач такої системи виконує звичайні дії, такі як видалення певної версії документа, створення нової версії тощо, він працює зі своєю локальною копією сховища. У міру внесення змін копії, що належать різним розробникам, починають розрізнятися і виникає необхідність їх синхронізації. Така синхронізація може здійснюватися за допомогою обміну патчами або про набори змін між користувачами.

Описана модель аналогічна до створення окремої гілки для кожного розробника в класичній системі управління версіями (у деяких розподілених системах перед початком роботи з локальним сховищем потрібно створити нову гілку). Поки розробник змінює лише свою гілку, його робота не впливає на інших учасників проекту та навпаки. Однак при необхідності злиття гілок (або синхронізацію локальних сховищ у розподіленій моделі) можуть виникнути конфлікти.

Основна перевага розподілених систем полягає в їхній гнучкості. Кожен розробник може вести роботу незалежно, оскільки йому зручно, зберігаючи проміжні варіанти документів і передаючи результати іншим учасникам, коли вважає за потрібне. При цьому обмін наборами змін може здійснюватись за різними схемами. У невеликих колективах учасники можуть обмінюватися змінами за принципом «кожний з кожним», за рахунок чого відпадає необхідність у створенні виділеного сервера. Велика спільнота, навпаки, може використовувати централізований сервер, з яким синхронізуються копії всіх учасників. Можливі і складніші варіанти - наприклад, зі створенням груп до роботи з окремих напрямів усередині більшого проекту.

Розподілена система контролю версій дозволяє клонувати віддалений репозиторій, роблячи точну копію. Вона також дозволяє розповсюджувати зміни з одного репозиторію на інший. У нерозподілених VCS розробнику потрібен репозиторій у тому, щоб зафіксувати зміни у ньому. Це робить розробника без доступу до репозиторію людиною другого гатунку. З розподіленою VCS кожен розробник може схиляти головний репозиторій, попрацювати з нього і потім поширити свої зміни на головний репозиторій.

**2. ВИДИ СИСТЕМ КОНТРОЛЮ ВЕРСІЙ**

Система контролю версій може бути будь-якої форми та розмірів, але є основні положення про їхню архітектуру. Деякі системи підтримують Атомарні Фіксації, які означають, що стан всього репозиторію змінюється повністю. Без Атомарних Фіксацій кожен файл або частина змінюється окремо і тому стан всього репозиторію в будь-якій точці не може бути зафіксовано.

Більшість звичайних VCS (Version Control System) систем дозволяють поєднувати зміни між гілками. Це означає, що зміни, зафіксовані в одній галузі, будуть зафіксовані в головній лінії або в іншій галузі за допомогою однієї автоматичної або, принаймні, напівавтоматичної операції.

**2.1 CVS**

Традиційно де-факто системою контролю версій стала CVS (Система Паралельних Версій), але потім з'явилося багато інших, які допомагають краще. З усіх цих можливостей CVS підтримує лише об'єднання змін.

Це зріла та відносно надійна система контролю версій. Багато проектів з відкритими вихідними кодами, включаючи KDE, GNOME та Mozilla, використовують CVS. Більшість центрів відкритих вихідних кодів, такі як SourceForge, пропонують CVS як сервіс, тому її використовують у багатьох інших проектах.

Система CVS має архітектуру клієнт-сервер. Зазвичай сервер та клієнт з'єднуються через локальну мережу або через Інтернет, але можуть працювати і на одній машині, якщо необхідно вести історію версій локального проекту. Серверне ПЗ зазвичай працює під управлінням Unix, тоді як CVS клієнти доступні у всіх популярних операційних системах.

Сервер зберігає в репозиторії поточну версію (версії) проекту та історію змін, а клієнт з'єднується з ним, щоб отримати потрібну версію або записати нову. Отримавши з сервера необхідну версію, клієнт створює локальну копію проекту (або його частини) - так звану робочу копію. Коли файли, що у робочої копії, внесено необхідні зміни, вони пересилаються на сервер.

Користувачі мають можливість порівнювати порівняти різні версії файлів, переглянути історію змін або навіть отримати історичний образ проекту на певну дату. Багато проектів формату «Відкритих ресурсів» дозволяють анонімний доступ до читання. Ця можливість передбачає, що користувачі можуть отримати доступ до версій файлів без пароля.

CVS також може містити різні гілки проекту. Наприклад, стабільна версія проекту може становити одну гілку, в яку вносяться лише виправлення помилок, тоді як активна розробка може вестися в паралельній гілці, що включає значні покращення або зміни з моменту виходу стабільної версії.

Крім того, варто також відзначити, що дана система використовує механізм дельта-компресії, щоб ефективніше зберігати різноманітні версії одного і того ж файлу.

Незважаючи на свою популярність, CVS має обмеження. Наприклад, вона не підтримує перейменування файлів та директорій. Крім того, бінарні файли не обробляються досить добре. CVS – нерозподілена система та атомарні фіксації змін не підтримуються. Оскільки вже є найкращі альтернативи, які містять ширший набір функцій, ви, можливо, вважаєте за краще почати новий проект, використовуючи щось інше.

Як плюс, CVS дуже добре документована у своїй книзі та у багатьох онлайн посібниках. Також існує безліч графічних клієнтів та доповнень.