T.C

İZMİR BAKIRÇAY ÜNİVERSİTESİ

MİMARLIK VE MÜHENDİSLİK FAKÜLTESİ BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

2022-2023 BAHAR DÖNEMİ

BİL 102-YAZILIM MÜHENDİSLİĞİ TEMELLERİ ÖDEV II

220601059 NİDA ELVİN MERTOĞLU

İÇİNDEKİLER:

Versiyon Kontrol Sistemi(VKS):

Versiyon kontrolü aynı zamanda kaynak kod yönetimi olarak adlandırılan zaman içinde dosyalarda yapılan değişiklikleri yönetmemize ve bu değişiklikleri bir veri tabanında depolamamıza ve daha sonra bu oluşturulan versiyonları çağırabilmemize olanak tanır. Versiyon kontrol sistemi, yazılım ekiplerinin zaman içinde kaynak kodda yapılan değişiklikleri takip edip yönetmesine yardımcı olan yazılım aracıdır. Versiyon kontrol sistemleri, geliştirme ortamları arttıkça yazılım ekiplerinin daha hızlı ve akıllı bir şekilde çalışmasını sağlar. DevOps ekipleri için özellikle faydalıdır çünkü başarılı dağıtımları hızlandırmalarına ve geliştirme süresini kısaltmalarına olanak sağlar. Koddaki her değişiklik, versiyon kontrol yazılımı tarafından belirli bir veri tabanı biçiminde kaydedilir. Bir hata yapılırsa, programcılar geri gidebilir ve kodun önceki yinelemelerini gözden geçirerek düzeltmeye yardımcı olurken tüm ekipte en az kesintiye neden olur. Versiyon kontrol sistemi, seçilen dosyaların bir önceki versiyona döndürülmesi, projenin tamamının bir önceki versiyona döndürülmesi, zaman içerisinde yapılan değişikliklerin karşılaştırılması ve takip edilmesi, probleme neden olabilecek değişikliklerin kimin tarafından yapıldığı, kimin bir problemden ne zaman bahsettiği gibi birçok işlemin gerçekleştirilebilmesini sağlar. Yukarıda bahsedildiği üzere versiyon kontrol sistemleri bazı kaynaklarda revizyon kontrolü veya kaynak kod kontrolü olarak da adlandırılabilir.

Versiyon Kontrol Sistemi Neden Önemlidir?

Versiyon kontrol sistemi, proje üzerinde yapılan değişiklikleri takip etmek ve her ekip üyesinin en son sürümle çalışmasını sağlamak adına önemlidir. Bu sayede yazılım ekibinin daha hızlı ürün geliştirmesini sağlar. Ekiplerin dünya çapında işbirliği yapmasını ve görünürlüğün artmasına yardımcı olur. Tüm yazılım geliştirme ekibinde koordinasyonu, paylaşımı ve işbirliğini kolaylaştırır. Ekiplerin dağıtılmış ve eş zamansız ortamlarda

çalışmasına, kod ve yapıların değişikliklerini ve sürümlerini yönetmesine ve birleştirme çakışmalarını ve ilgili anormallikleri çözmesine olanak tanıması açısından önemlidir.

Versiyon Kontrol Sisteminin Avantajları:

Her dosyanın tam uzun vadeli değişiklik geçmişi kayıt alınır. Bu, çok sayıda kişi tarafından zaman içinde yapılan her değişikliği kapsar. Dosya ekleme, silme ve içerik değişiklikleri yapılan değişiklik örnekleridir. Dallanma ve birleştirme sayesinde ekip üyelerinin aynı anda aynı kod üzerinde çalışmasına imkan sağlar. Alt sürümler oluşturularak yazılım üzerinde farklı çalışmaları yürütülüp, daha sonra ana yazılıma bu entegre edilebilir. Yazılımda yapılan her değişikliği takip edilmesine ve bunların proje yönetimine bağlanabilinmesine imkan sağlar. Yazılım üzerindeki sorunların, sürümlerle ilişkilendirilebilmesini ve takip edilebilmesini sağlar.

Versiyon Kontrol Sistemi Türleri:

Local versiyon kontrol sistemleri en eski olan versiyon kontrol sistemi yaklaşımıdır. Üzerinde çalışılan proje ve yapılan değişiklikler kullanıcının makinası üzerindeki veritabanında tutulur. Her yapılan commit bir versiyon olarak tutulur ve commit değerine hash ataması yapılarak her versiyon birbirinden ayırt edilir. Versiyon görüntüleme imkanını sağlar. Fakat bu sistemde sadece bir kullanıcı etkin bir şekilde çalışma sağlayabilir.

Merkezi versiyon kontrol sistemleri bir yazılım ekibin proje üzerinde etkin çalışmasını sağlar. CVS, SVN örnek verilebilir. Bu sistemde yapılan proje ortak bir respository'de tutulur ve birden fazla geliştirici bu respository üzerinden checkout ve commit işlemlerini gerçekleştirir.

Dağıtık versiyon kontrol sistemlerinde merkezi bir respository yoktur, proje üzerinde çalışan her geliştirici, projenin kopyasını kendi yerel bilgisayarında tutmaktadır. Bu sayede sunuculardan biri çökerse, geliştiricilerden birinin projeyi sunucuya geri yükleyerek sistem kurtarılabilir. Git, Mercurial, BitKeeper örnek verilebilir. Aynı proje üzerinde farklı geliştiriciler, farklı biçimlerde çalışmalar yürüterek, farklı iş akışları ile rahatlıkla çalışabilmesini sağlar.

GIT:

Git, en popüler versiyon kullanım sistemleri araçlarından biridir ve hem Windows hem de macOS üzerinde kullanılabilir. Git, dağıtılmış versiyon kullanım sistemi kapsamına girer, yani her geliştiricinin yerel bir kopyası bulunur ve değişiklikleri paylaşmak için sunucu tabanlı bir yapıya ihtiyaç duyulmaz. Git'le her commit yapıldığında ya da projenin durumu kaydedildiğinde, tüm dosyaların anlık görünümünün referansını saklar. Eğer dosyalar değişmediyse Git dosyaları tekrar saklamaz, onun yerine o dosyaların olan referansına bağlantı verir. Verileri anlık görünüm akışı olarak tutar. Merkezi bir respository yoktur ve çoğu işlem çalışmak için yalnızca yerel dosyalara ve kaynaklara ihtiyaç duyar. Git'deki veriler saklanmadan önce sağlaması yapılarak ve ondan sonra da o sağlama referans alınarak gösterilir. Böylece aktarım yaparken bilgi kaybı olmaz.

GIT Avantajları/Zorlukları:

Git'in avantajları; hızlı ve verimli performans sağlar, kod değişikleri temiz ve kolay takip edilir, çapraz platform desteği sunar, kolay sürüm yönetimi mevcuttur, ekip çalışmasını kolaylaştırır, dalları (branches) ve birleştirmeleri (merges) etkili bir şekilde yönetme imkanı sağlar. Git'in zorlukları; başlangıçta bazı kavramları anlamak ve kullanımını öğrenmek zaman alabilir, karmaşık proje yapılarında veya büyük ekiplerle çalışırken, dalların ve birleştirmelerin yönetimi bazen zor olabilir.

GIT Windows Üzerinde Kullanımı:

Temel Git Komutları:

1. git config :
Bu komut ile kullanıcı adı ve e-posta adresi yapılandırılır ve sonraki tüm projelerde bu kullanıcı adı ve e-posta adresi kullanılır.

2. git init:
Bu komut kullanılarak bulunulan dizin boş bir git repository'si haline getirilir ve .git isimli bir dizin oluşturulur.

3. git clone:
Bu komut var olan bir Git repository'sinin kopyalanması için kullanılır. Git clone komutu ile öncelikle yerel bir repository işaret edilerek bu repository'nin bir kopyası, yeni bir dizine kopyalanır.

4. git add:
Bu komut kullanılarak işaret edilen dosya veya tüm proje, çalışılan dizine eklenir.

5. git commit:
Bu komut kullanılarak çalışılan dizinde bulunan dosyalar paketlenerek .git klasörü içindeki head isimli kısıma eklenir. Daha sonra gelen pencerede bir commit mesajı girilir.

6. git status:
Bu komut kullanılarak üzerinde çalışılan repository'nin o anki durumu görüntülenir. Üzerinde değişiklik yapılan dosyalar, yeni eklenmiş dosyalar ve commit komutu uygulanmamış dosyalar konsol üzerinde listelenir.

7. git checkout –b:
Bu komut kullanılarak yeni bir branch oluşturulur ve o branch üzerine geçilir.

8. git pull:
Bu komut kullanılarak uzak bir repository'deki değişiklikler üzerinde çalışılan dizine getirilir ve bu dizin ile birleştirilir.

CVS(Concurrent Versions System):

CVS(Concurrent Versions System) eski bir versiyon kontrol sistemidir ve yazılım geliştirme süreçlerinde kullanılan bir araçtır. CVS, projelerin kaynak kodlarını, dosyalarını ve değişikliklerini izlemek ve yönetmek için kullanılır. Merkezi versiyon kontrol sistemidir.

CVS(Concurrent Versions System) Avantajları/Zorlukları:

CVS(Concurrent Versions System)'nin avantajları; iyi bir çapraz platform desteği sunar, sağlam ve tam özellikli komut satırı istemcisi sayesinde güçlü komut dosyası oluşturulur, kaynak kod deposunda iyi bir web taraması sunar, çok eski, iyi bilinen ve anlaşılan bir araçtır. CVS(Concurrent Versions System)'nin zorlukları; paralel geliştirme sürecini desteklese de, çakışmaların yönetimi ve birleştirmelerin yapılması bazen karmaşık olabilir. Bu yüzden birden fazla geliştiricinin aynı dosyayı değiştirdiği durumlarda çakışmalar ortaya çıkabilir. Büyük ve karmaşık projelerde performans sorunları yaşayabilir.

CVS(Concurrent Versions System) Windows üzerinde kullanımı:

1. cvs co test:

Kodları sunucudan çalışma dizimine çeker.

2. vi test.c:

Çalışma dizinindeki kodlarda değişiklik yapar.

3. cvs commit:

Kodları sunucuya yükler

4. cvs update:

Çalışma dizinimindeki kodları günceller.

5. vi yeni.c - cvs add yeni.c - cvs commit:

Yeni dosya eklenir.

6. rm silinecek.c - cvs delete silinecek.c - cvs commit:

Mevcut dosya silinir.

7. cvs update - cvs tag rel-0-0-1:

Bir dağıtım ertesinde yeni sürüme geçilir.

Kaynakça:

- https://git-scm.com/book/en/v2/Getting-Started-About-Version-Control
- https://www.geeksforgeeks.org/version-control-systems/
- https://about.gitlab.com/topics/version-control/
- https://git-scm.com/book/en/v2/Getting-Started-What-is-Git%3F
- https://www.nongnu.org/cvs/