YASİN BERA ŞAHİN
220601069

VERSİYON KONTROL SİSTEMLERİ (VKS) TARİHÇESİ

Versiyon Kontrol Sistemleri (VKS), yazılım geliştirme sürecinde dosya ve kod değişikliklerini yönetmek için kullanılan bir araçtır. VKS'nin tarihçesi, yazılım geliştirme sürecindeki ihtiyaçların ve teknolojik gelişmelerin sonucunda gelişmiştir. VKS 1970'lerde ortaya çıkan "versiyon kontrolü" kavramına dayanır. Bu dönemde, yazılım geliştirme sürecinde yapılan değişikliklerin yönetilmeye ve izlenmeye ihtiyacı vardır. O zamanlar, VKS'nin temelleri daha basit ve sınırlı özelliklere sahiptir. 1980'lerde, Merkezi Versiyon Kontrol Sistemleri (MVKS) ortaya çıkmıştır. Bu sistemler, projenin merkezi bir sunucuda depolandığı ve kullanıcıların sunucuya erişerek değişikliklerini kaydettiği bir yapı niteliğindedir. Ayrıyeten MVKS'nin merkezi sunucuya bağımlı olması, ağ bağlantısı kesildiğinde çalışmayı durduracağı için geliştirilmeye ihtiyacı vardır. 1990'ların ortalarında, Dağıtık Versiyon Kontrol Sistemleri (DVKS) ortaya çıkmıştır. Bu sistemlerde, projenin her bir kullanıcının lokalinde bir kopyası bulunur ve değişiklikler yerel olarak kaydedilir. Kullanıcılar, değişiklikleri paylaşmak ve senkronize etmek için bir merkezi sunucuya ihtiyaç duymazlar. Merkezi bir sunucu olmadığı için MVKS'deki gibi bağlantı sorunu olduğunda sadece kişinin erişimi durur ve diğer kullanıcılar çalışmaya devam edebilir. Ama herkesin ayrı ayrı yapması değişikliklerin yönetimi zor hale getirebilir ve olası çakışmalar yaşanabilir. 2000'lerde, açık kaynaklı VKS'ler popülerleşmiş ve çeşitli VKS araçları geliştirilmiştir. Bu araçlar arasında en yaygın kullanılanlar Git, Subversion (SVN), Mercurial, ve Perforce'dır. Bu araçlar, kullanıcı dostu arayüzler, gelişmiş işbirliği özellikleri ve güçlü versiyon kontrol yetenekleri sunmaktadırlar. Günümüzde, VKS'ler yazılım geliştirme sürecinin vazgeçilmez bir parçası haline gelmiştir. Büyük ölçekli projelerde, ekiplerin paralel olarak çalışmasını sağlar ve değişikliklerin takibi ve yönetimi konusunda kolaylık sağlar. Ayrıca, bulut tabanlı VKS hizmetleri de geliştirilmiştir. Bu bulut tabanlı VKS, ekiplerin projelerini çevrimiçi ortamda depolamasını ve erişimini sağlar.

VERSİYON KONTROL SİSTEMİ (VKS) NEDİR?

Versiyon Kontrol Sistemleri (VKS) yazılım geliştirme süreçlerinde çok önemli bir yer kaplar. Bu sistemler, bir projenin dosya ve kodlarının değişimlerini izler, kaydeder ve yönetir. VKS, bir ekip veya bireyin aynı projede çalışırken dosyaları senkronize etmelerini, değişiklikleri geri alabilmelerini, farklı sürümleri karşılaştırabilmelerini ve birlikte çalışabilmelerini sağlar. VKS'nin merkezi bileşeni, projenin ana depo veya sunucusudur. Bu depoda, projenin dosyaları, sürüm geçmişi ve değişiklik kayıtları saklanır. Proje dosyaları ve geçmişi sunucuda saklanırken, kullanıcılar bu dosyaları kendi bilgisayarlarına indirerek üzerinde çalışır. Kullanıcılar, yerel çalışma kopyaları aracılığıyla bu depoya erişebilir, dosyalar üzerinde değişiklik yapabilir ve değişiklikleri sunucuya gönderebilir. Böylece her bir değişiklik kaydedilir ve projenin geçmişi oluşturulur. Bu geçmişte her bir değişikliğin ne zaman yapıldığı, kim tarafından yapıldığı ve ne amaçla yapıldığı takip edilebilir.

VKS KULLANIMININ AVANTAJLARI NELERDİR?

- VKS, ekip üyelerinin aynı projede birlikte çalışmasını kolaylaştırır. Her kullanıcı, bağımsız olarak çalışabilir ve değişiklikleri diğerleriyle paylaşabilir. Böylece hızlı ve geliştirici şekilde projeyi ilerletebilirler.

-VKS, değişikliklerin geri alınabilmesini sağlar. Hatalı bir değişiklik yapıldığında veya bir önceki sürüme dönülmesi gerektiğinde, geçmişteki versiyonlara geri dönülebilir. Ayrıca, farklı sürümler karşılaştırılarak değişikliklerin izlenmesi ve çakışmaların tespit edilmesi kolaylaşır.

-Her bir değişiklik kaydedildiği için, projenin geçmişi izlenebilir. Hangi değişikliklerin ne zaman ve kim tarafından yapıldığı takip edilebilir.

-VKS, paralel olarak farklı özellikler üzerinde çalışmayı destekler. Dallar oluşturarak farklı ekip üyeleri farklı sürümleri geliştirebilir ve değişiklikleri birleştirerek ana projeye entegre edebilir.

VKS KULLANIMININ ZORLUKLARI NELERDİR?

- VKS'nin karmaşıklığı ve farklı komutları veya arayüzleri öğrenmek biraz zaman alabilir. Kullanıcıların VKS'nin doğru şekilde kullanmayı öğrenmeleri , pratik yaparak ve proje geliştirerek zamanla kullanmayı pekiştirir.

- Birden fazla kullanıcının aynı dosyalar üzerinde çalışması durumunda çakışmalar ortaya çıkabilir. VKS, çakışmaları yönetmek için bazı araçlar ve stratejiler sunar ancak çakışmaların çözümlenmesi bazen zor olabilir.

- VKS, büyük projeler veya büyük dosya boyutlarıyla performans sorunları yaşayabilir. Özellikle dosyaların ve geçmişin büyüklüğü arttıkça VKS'nin hızı ve verimliliği azalabilir.

VKS ARACI ÖRNEKLERİ

Git: Git, en popüler ve güçlü dağıtık versiyon kontrol sistemlerinden biridir. Linus Torvalds tarafından 2005 yılında Linux çekirdek geliştirme projesi için geliştirilmeye başlamış olan Git, zamanla açık kaynaklı bir proje haline gelmiş ve geniş bir kullanıcı kitlesi tarafından yoğun ilgi görmüştür. Git; hızlı, esnek, güvenilir ve ölçeklenebilir olmasıyla bilinir. Git'in temel amacı, projelerin kod tabanındaki değişikliklerin takibini yapmak, sürümleri yönetmek ve işbirliğini kolaylaştırmaktır. Git'in diğer VKS'lere kıyasla avantajlarından biri dağıtık yapısıda olmasıdır. Kullanıcılar, projenin tam bir kopyasına sahip olur ve bu kopya üzerinde değişiklikler yapabilir. Bu şekilde kullanıcılar bağımsız olarak çalışmış olur ve yaptıkları değişiklikleri diğer kullanıcılarla paylaşılmasını sağlar. Git'in kullanım avantajlarından biri, hızlı ve verimli olmasıdır. Git, değişiklikleri takip ederken dosyaların tamamını değil, sadece değişikliklerin farklarını depo eder. Bu da dosya boyutunu küçültür ve işlemleri hızlandırır. Git, yerel bir veritabanına sahip olduğu için internet bağlantısı gibi gereklilik olmaz ve çevrimdışı çalışabilme olanağı sağlar. Git'in güçlü bir dallanma ve birleştirme (branching ve merging) sistemi vardır. Projede farklı özellikleri veya versiyonları ayrı dallarda geliştirebilir ve daha sonra bu dalları birleştirebilirsiniz. Böylece ekip çalışmasını kolaylaştırır ve aynı anda birden fazla özelliğin üzerinde çalışılmasına olanak tanır. Git, dağıtık bir yapıya sahip olduğu için projenin her katılımcısı kendi yerel bölgesinde tam bir kopyaya sahiptir. Bu da bağımsız

çalışmayı ve projenin farklı noktalarında değişiklikler yapmayı mümkün kılar. Ayrıca Git'in güvenliği de yüksektir. Ama Git'in kullanımında bazı zorluklar da bulunabilir. İlk olarak Git, diğer versiyon kontrol sistemlerinden öğrenilmesi daha zorlu olabilir. Komutların çeşitliliği ve karmaşıklığı, yeni kullanıcılar için başlangıçta zor olabilir. Ancak zamanla Git'in temel komutlarını öğrendikçe ve pratik yaptıkça kullanımına alışılacaktır. Ayrıca birden fazla kullanıcının aynı dosyalar üzerinde çalışması durumunda çakışmalar ortaya çıkabilir. Git, çakışmaları tespit etmek için bazı araçlar sunar ama çözümlemek kullanıcılara ait bir sorumluluktur. Git, hem Windows hem de macOS işletim sistemlerinde kullanılabilir. Windows için Git, Git Bash adlı bir komut satırı arayüzü sunar. Ayrıca, Grafik Kullanıcı Arayüzü (GUI) olan GitKraken, SourceTree veya TortoiseGit gibi araçlar da kullanılabilir. macOS için Git, Terminal veya Xcode gibi yerleşik araçlarla kullanılabilir. Ayrıca, Grafik Kullanıcı Arayüzü olarak GitHub Desktop veya Tower gibi uygulamalar da kullanılabilir. Sonuç olarak Git, yazılım geliştirme süreçlerinde önemli bir yere sahip olan ve büyük bir kullanıcı kitlesi tarafından benimsenen bir VKS'dir. Git hem Windows hem de macOS işletim sistemlerinde kullanılabilir. Esneklik, güvenilirlik ve işbirliği özellikleriyle projelerin yönetimini kolaylaştırır ve yazılım geliştirme sürecini daha verimli hale getirir.

Mercurial: Mercurial, yazılım geliştirme sürecinde kullanılan bir dağıtık versiyon kontrol sistemidir. Mercurial, Matt Mackall tarafından geliştirilen ve 2005 yılında piyasaya sürülen açık kaynaklı bir projedir. Mercurial, hızlı, basit ve kullanıcı dostu olmayı hedefleyen bir versiyon kontrol sistemidir. Başlangıçta Linux çekirdek geliştirme projesi için tasarlanmış olan Mercurial, daha sonra geniş bir kullanıcı kitlesi tarafından benimsenmiştir. Mercurial'in kullanım avantajlarından biri, kullanıcı dostu bir arayüze sahip olmasıdır. Komut satırı tabanlı bir arayüze sahip olmasına rağmen grafiksel kullanıcı arayüzleri de mevcuttur ve kullanıcıların tercihine göre seçilebilir. Mercurial, basit ve anlaşılır komutlarla çalışır. Bu da kullanıcıların hızlı bir şekilde öğrenmesini ve projelerini yönetmesini sağlar. Ayrıca, Mercurial'in güçlü dallanma ve birleştirme (branching ve merging) sistemi vardır. Bu sayede, farklı özellikleri veya versiyonları ayrı dallarda geliştirebilir ve daha sonra bu dalları birleştirebilirsiniz. Böylece paralel geliştirme ve işbirliğini kolaylaştırır. Mercurial'in hızlı ve verimli çalışması da önemli bir avantajdır. Değişikliklerin sadece farklarını kaydeder. Bu da tüm dosya yerine sadece değişiklik olduğundan az yer kaplar ve daha hızlı çalışır. Büyük projelerde bile performans yüksektir ve kullanıcıya hızlı yanıtlar sunar. Mercurial'in kullanımında bazı zorlukları da vardır. İlk olarak diğer VKS'lerle kıyaslandığında daha az popüler olması nedeniyle, geniş bir kullanıcı topluluğuna sahip değildir. Bu yüzden bazı kaynakların ve toplulukların daha sınırlı olabileceği söylenebilir. Ayrıca Mercurial'in bazı özellikleri ve komutları Git gibi diğer VKS'lerden farklılık gösterebilir. Bu nedenle bir alışma süreci gereklidir. Ancak Mercurial'in kullanıcı dostu kaynaklarıyla alışılmasına kolaylık sağlanır. Mercurial hem Windows hem de macOS işletim sistemlerinde kullanılabilir. Windows için, Mercurial'in resmi dağıtımı olan TortoiseHg adlı bir grafiksel kullanıcı arayüzü mevcuttur. Ayrıca, komut satırı tabanlı kullanıcılar, Windows Command Prompt veya PowerShell aracılığıyla Mercurial'i kullanabilirler. macOS için de Mercurial, Terminal üzerinden veya grafiksel arayüzlerle kullanılabilir. Mac kullanıcıları, MacPorts veya Homebrew gibi paket yöneticileri aracılığıyla Mercurial'i yükleyebilirler. Sonuç olarak Mercurial; basitlik, kullanım kolaylığı, güçlü dallanma ve birleştirme sistemi gibi özellikleri

sayesinde projelerin etkili bir şekilde yönetilmesini sağlar. Mercurial'in kullanımı kolaydır ve kullanıcı dostu arayüzleri mevcuttur. Mercurial hem Windows hemde macOS işletim sistemlerinde kullanılabilir. Mercurial, yazılım geliştirme sürecinde verimliliği artıran bir araç olarak kullanıcılar tarafından tercih edilmektedir.

Subversion (SVN): Subversion (SVN), yazılım geliştirme sürecinde kullanılan bir merkezi versiyon kontrol sistemidir. Subversion, CollabNet Inc. tarafından geliştirilen ve 2000'li yılların başından beri kullanılan bir versiyon kontrol sistemidir. Subversion, açık kaynaklı bir proje olup, geniş bir kullanıcı kitlesine sahiptir. Subversion'ın temel amacı, bir merkezi sunucu üzerindeki depoda projenin tüm dosyalarını ve sürüm geçmişini saklamaktır. Kullanıcılar, kendi yerel bilgisayarlarında projenin kopyalarını alarak değişiklik yapabilir, bu değişiklikleri sunucuya yükleyebilir ve diğer kullanıcılarla senkronize işlemler yapabilirler. Subversion'ın kullanım avantajlarından biri, merkezi bir yapıya sahip olmasıdır. Projede yapılan değişiklikler, merkezi sunucuda yönetildiği için projeye erişimi olan herkes en güncel sürüme sahip olabilir ve diğer kullanıcıların değişikliklerini görebilir. Böylece ekip çalışması kolaylaşır ve işbirliğini geliştirir. Bununla birlikte Subversion, kullanıcıların dosya ve klasörlerde gerçekleştirdiği değişiklikleri yönetmek için bir dizi komut seti sunar. Kullanıcılar; dosyaları güncelleme, ekleme, silme ve taşıma gibi işlemlerle değişiklikleri kontrol edebilirler. Bu işlemler, projenin sürüm geçmişine kaydedilir ve gerektiğinde geri alınıp kullanılabilirler. Böylece istenilen vakit istenilen versiyona geçiş sağlanabilir. Subversion'ın kullanımının bazı zorlukları da vardır. İlk olarak merkezi yapıya sahip olması nedeniyle ağ bağlantısı gerektirir. Sunucuya erişimin olmadığı durumlarda çalışmak mümkün olmayabilir. Yani ağ bağlantısıyla çevrimiçi olmak gerekmektedir. Ayrıca projedeki dosyaların senkronize edilmesi bazen zaman alabilir. Özellikle büyük projelerde ve yavaş ağ bağlantılarıyla çalışıldığında performansta düşüklük olması kaçınılmaz bir durumdur. Subversion hem Windows hem de macOS işletim sistemlerinde kullanılabilir. Windows için, TortoiseSVN adlı bir grafiksel kullanıcı arayüzü ve komut satırı tabanlı araçlar mevcuttur. TortoiseSVN, dosya gezgini entegrasyonu sağlayarak kullanıcıların dosyaları doğrudan Windows Gezgini üzerinden yönetmelerini sağlar. macOS için, Komut Satırı Aracı (Command Line Tools) kullanılarak Subversion komutları kullanılabilir. Sonuç olarak, Subversion, merkezi bir versiyon kontrol sistemi olarak yazılım geliştirme süreçlerinde kullanılan güvenilir bir araçtır. Projelerin sürüm geçmişini takip etmek, değişiklikleri yönetmek ve işbirliğini kolaylaştırmak için kullanılır. Subversion hem Windows hem de macOS işletim sistemlerinde kullanılabilir. Subversion, kullanıcı dostu arayüzleriyle kullanıcıların projelerini daha rahat ve daha anlaşılır olmasından ötürü kolay bir şekilde yönetmelerini sağlar. Subversion, büyük bir kullanıcı topluluğuna sahiptir ve geniş çapta kullanılmaktadır.

Verdiğimiz örneklere göre en son Git, Mercurial dağıtık versiyon kontrol sistemine örnektir. Ama Subversion merkezi versiyon kontrol sistemine örnektir. Aralarındaki fark merkezi versiyon kontrol sisteminde bir merkeze bağlanıp sunucuda tüm kullanıcılar değişiklik yapabilir ve birbirleriyle paylaşabilirler. Dağıtık versiyon kontrol sisteminde ise bu şekilde değildir. Bu sistemde ise her kullanıcı projenin kopyasını alır ve kendi yerel bilgisayarından değişiklikler yapar ve depoya yüklenir. Bu yüzden çakışmaların olma ihtimali vardır. Tüm hepsini genel olarak alırsak bir ekip veya bireyin aynı projede çalışırken dosyaları senkronize etmelerini, değişiklikleri geri alabilmelerini, farklı sürümleri karşılaştırabilmelerini ve birlikte

çalışabilmelerini sağlar. Bu sistemler yazılım geliştirme süreçlerinde çok önemli yere sahiplerdir.

KAYNAKÇA

https://tr.wikipedia.org/wiki/S%C3%BCr%C3%BCm_kontrol_sistemi

https://git-scm.com/book/tr/v2/Ba%C5%9Flang%C4%B1%C3%A7-Versiyon-Kontrol

https://koraypeker.com/2018/05/11/versiyon-kontrol-sistemleri-vcs/

GitHub==> YasinBeraSahin

Medium==> Yasin_Bera_Şahin