

Subvesion ile Versiyon Kontrolü

KurumsalJava.com
KurumsalJavaAkademisi.com

Bu PDF dosya herhangi bir server üzerine kopyalanarak, indirme sunulabilir.
Bunun için özel izin alınması gerekmez! Bilgi sadece paylaşılarak çoğalır. Lütfen
bu yazıyı faydalanacağını düşündüğünüz şahıslara gönderiniz.

Özcan Acar
Bilgisayar Mühendisi
<http://www.ozcanacar.com>
acar@unitedinter.net

Bir Kaosun Hikayesi ...

Anıl o gün işe gitmek için arabasına bindiğinde, akli hala, sabaha kadar düşünüp, cevap bulmadığı sorundaydı. Boğaz köprüsüne kadar geldiğinin farkına bile varmamış, İstanbul'un o tipik trafiğine takılıp kalmıştı. Anıl 29 yaşında, iyi denebilecek seviyede bir programcıydı. Birçok KOBİ'de programcı olarak tek kişilik projelerde çalışmış ve tecrübe edinmişti. Tek başına çalışmaya alışkın olduğu için, yazdığı programları 8 GB hafızalı USB Stick üzerinde iş yerinden eve, evden işe taşır ve böylece birden fazla bilgisayar üzerinde iş yerinde ve evde zaman buldukça çalışma fırsatı bulurdu. İş yerinde mesai saati bitimi hızlıca tüm projeyi USB Stick üzerinde çekerek kopyalar, kopyalama işlemi 3-5 dakika içinde tamamlandıktan sonra USB stick'i cebine atarak, evin yolunu tutardı. Birçok programcı gibi eve geldiğinde hemen bilgisayarın başına oturur ve önce maillerini kontrol eder, daha sonra USB stick üzerindeki programı bilgisayarına aktarır ve iş yerinde çözemediği problem üzerinde çalışmaya devam ederdi. Tabii eşi bu durumdan pek hoşnut değildi ☺. Yazdığı programları her zaman yanında bulundurmak ona güven verirdi. Herhangi bir bilgisayar üzerinde işine devam edebileceğini bilmek bilgisayar mühendisi olarak onun öz güvenini artırırdu.

Yıl 2008 senesine gelmiş ve Anıl hala birçok programcının beraber çalışmak zorunda olduğu bir projede yer alamamış ve bu yüzden bir takım içinde nasıl çalışılacağı hakkında bilgiye sahip olamamıştı. Diğer mühendis arkadaşlarıyla görüşmelerinde versiyon kontrolü, dal (branch) ve release yönetimi gibi terimler duymuş, lakin bu konuda detaylı araştırma ve uygulama yapma fırsatı bulamamıştı. Son zamanlarda yazılım sektöründe meydana gelen metot değişiklikleri Anıl'ı tedirgin etmeye başlamıştı. Oda her insan gibi değişikliklere karşı içgüdüsel bir karşı tavır almıştı, ama bilgisayar mühendisi olarak er yada geç bu yeni metotlarla tanışmak ve çalışmak zorunda olduğunu biliyordu.

İki hafta önce patronu onun yanına gelerek, çok büyük bir ihaleyi aldıklarını ve iyi bir programcı ekibi kurarak, yazılım sürecini başlatmaları gerektiğini söylemişti. Anıl'ın yeni görevi yeni programcı ekibi için elemanların işe alınması ve takım yöneticiliği olacaktı. Bu durum Anıl'ı tedirgin etmişti. Bir takım içinde kod paylaşımı USB stick'lerle mümkün olacaktı mıydı? Yoksa kodun merkezi bir server üzerinde bulundurulması mı gerekecekti? İşte bu ve bunun gibi sorular sabaha kadar beynini meşgul etmişti.

Boğaz köprüsünün üzerinde geldiğinde, boğazın mavi suları içini biraz rahatlatmış ve tedirginliğini azaltmıştı. Bugün yeni takım arkadaşlarıyla tanışacak ve yeni projenin temeli atılacaktı. Trafiğin yoğunluğundan dolayı ofise geç kalmıştı. Toplantı odasına girdiğinde patronu ve 3 yeni takım arkadaşını onu bekler durumda buldu. Tanışma faslının ardından proje hakkında konuşuldu. Patron kısa bir zaman sonra tüm ekibe başarılar dileyerek, kendi işinin başına döndü. Gelişmelerden Anıl aracılığıyla haberdar olmak istediğini ve kısa bir zaman içinde çalışır bir prototip oluşturulması için hemen çalışmalara başlanmasını istedi.

Anıl ve yeni çalışma arkadaşları Aydın, Müge ve Mustafa kolları sıvayarak işe başladılar. USB stick ile kod paylaşımının mümkün olmadığını bildikleri için, kodun merkezi bir server üzerinde bulundurulmasına karar verdiler. Sonuç itibariyle herkesin görevi belli idi ve kod paylaşımında belli kurallar takip edildiği sürece bir sorun çıkması beklenmiyordu. Takım mensubu her programcı akşam evine gitmeden önce, yaptığı program değişikliklerini servere kopyalanacak ve böylece kendi kodunu takım içinde paylaşmış olacaktı. Bu yöntemin ne kadar kötü sonuçlar doğuracağını daha sonra çok iyi anlayacaklardı.

İlk iki hafta herkes kendi modülü üzerinde çalışmalarını sürdürdü ve oluşturduğu kodu servere yükledi. Programcılar her sabah en aktüel kodu serverden edinerek, işlerine devam ettiler. Çalışmalar hızlı bir şekilde devam etti ve ikinci haftanın sonunda ilk prototip çalışır hale geldi.

Anıl patronun yanına giderek, ilk versiyonun tamamlandığını bildirdi. Anıl'ın patronu ve kalite kontrol bölümünden Veli bey prototip üzerinde incelemelerde bulundular. Prototip birçok modüllü ihtiva etmesede, çalışır durumda olan bir program parçasıydı. Patron, prototipin kendisinde iyi bir intiba bıraktığını söyleyerek, çalışmalara devam edilmesi direktifini verdi. Anıl için bu çalışma metodlarının ne kadar iyi olduğunun bir ispatıydı. Kendisi ve çalışma arkadaşlarıyla gurur duydu ve işinin başına döndü.

Üçüncü haftanın başında Müge rahatsızlanmış ve işe gelememişti. Bir hafta boyunca dinlenmesi ve iyileşmesi gerekiyordu. Yapılan proje planına ayak uydurulabilmesi için Müge'nin görevlerinin ekip içindeki diğer programcılara dağıtılması gerekiyordu. Kısa bir kriz toplantısının ardından Müge'nin üzerinde çalıştığı modüller Mustafa ve Aydın arasında paylaştırıldı. Bu durum Mustafa ve Aydın'ı tedirgin etmişti. İki hafta boyunca ekip içinde neredeyse kimse birbiriyle beraber çalışmadan kendi modülleri üzerinde çalışmıştı. Açıkcası Mustafa ve Aydın Müge'nin ne yaptığı hakkında bir bilgiye sahip değildiler. O gün Müge'nin kodlarını inceleyerek geçti.

Mustafa ve Aydın kendi modülleri yanısıra, Müge'den devraldıkları modül üzerinde ortak çalışmaya başladılar. Her programcı mesai bitiminde yaptığı tüm değişiklikleri servere aktarıyor ve mesai başlangıcında tüm projeyi server'den alıyordu. Zaman içinde Mustafa ve Aydın farkında olmadan aynı metodlar üzerinde değişiklik yapmaya başladılar. Bunun nasıl bir sonuç doğuracağı ortadaydı: en son programcının servere kopyaladığı programlar, diğer programcılar tarafından yapılan değişiklikleri yok ediyordu. Tabi bu durumun ortaya çıkması pek uzun bir zaman almadı. Mustafa ertesi gün çalıştığı en son sınıf kodunda oluşturduğu yeni metodu bulamadı. Bunun üzerine Aydın'ın yanına giderek, sınıf üzerinde bir değişiklik yapıp, yapmadığını sordu. Aydın sınıf üzerinde bir değişiklik yapmamıştı, ama dün mesai bitiminde kodları en son servere kopyalayan Aydın olduğu için, Mustafa'nın yaptığı değişiklikler yok olmuştu. Bir version kontrol sistemi kullanmadıkları için, herhangi bir dokümanın belirli bir tarihteki durumunu elde etmeleri de mümkün değildi.

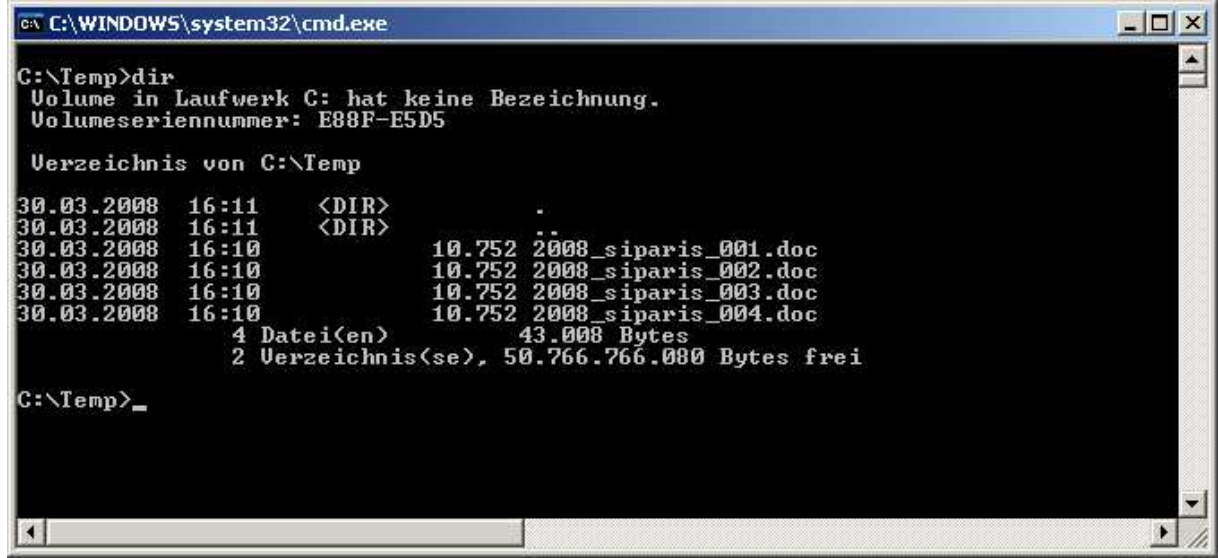
Bu hikayenin sonucunu büyük bir ihtimalle kestirebiliyorsunuz. Hayır! Arkadaşlar işlerinden olmadılar, ama patrone büyük bir azar işittikten sonra, versiyon kontrol sistemi hakkında verilen bir seminerin yolunu tuttular ☺.

Doküman varsa versiyon vardır...

İnsanoğlu okuma, yazmayı icat etmeden önce mağara duvarlarına resimler yaparak düşüncelerini şekillendirmeye başladı. Araştırmalara göre ilk yazının Sümer'liler tarafından İsa'dan önce 3500 civarında icat edildiği söylenmektedir. O devrin insanları yazı benzeri işaretler kullanarak ilk dokümanları oluşturmuş ve bu dokümanları iletişim aracı olarak kullanmışlardır. Günümüzde latin alfabesinde yer alan kelimeleri kullanarak bilgisayarda dijital dokümanlar oluşturuyoruz, arkadaşlarımıza email gönderiyoruz yada elimize kağıt ve kalem alarak mektup yazıyoruz. Bu işlemlerin sonucunda dijital olan yada olmayan bir doküman oluşuyor.

Her doküman oluşturulduğu ilk saniyeden itibaren bir versiyon ihtiva eder. Bir versiyon, dokümanın geleceğe doğru olan yolculuğunda durak yaptığı istasyonlardan birisidir. Zaman içinde doküman değişikliğe uğrar. Her değişikliğin ardından dokümanın yeni bir versiyonu oluşur. Dokümanın herhangi iki versiyonu arasındaki fark, bu iki versiyonun oluşumu için geçen zaman diliminde, doküman üzerinde yapılan tüm değişiklikleri ihtiva eder.

Bir dokümanın versiyonları arşivlenmediği sürece, o doküman üzerinde yapılan değişiklikler takip edilemez.



```
C:\WINDOWS\system32\cmd.exe
C:\Temp>dir
Volume in Laufwerk C: hat keine Bezeichnung.
Volumeseriennummer: E88F-E5D5

Verzeichnis von C:\Temp

30.03.2008  16:11    <DIR>          .
30.03.2008  16:11    <DIR>          ..
30.03.2008  16:10             10.752  2008_siparis_001.doc
30.03.2008  16:10             10.752  2008_siparis_002.doc
30.03.2008  16:10             10.752  2008_siparis_003.doc
30.03.2008  16:10             10.752  2008_siparis_004.doc
               4 Datei(en)               43.008 Bytes
               2 Verzeichnis(se), 50.766.766.000 Bytes frei

C:\Temp>_
```

Resim 16.1 2008_siparis.doc isimli dosyanın değişik versiyonları

Resimde görüldüğü gibi 2008 senesi için yapılan siparişler 2008_siparis_xxx.doc isiminde bir doküman içinde tutulmaktadır. Her yeni sipariş için dokümanın yeni bir versiyonu oluşturulmuş ve böylece bu dokümanın tarihçesi arşivlenmiştir. 2008_siparis_004.doc en son versiyondur ve bu doküman bünyesinde meydana gelen tüm değişiklikleri ihtiva etmektedir. Dokümanın değişik versiyonları arşivlendiği için, değişik versiyonlar arasında kıyaslama (compare) yapılarak, meydana gelen değişiklikler takip (track) edilebilir.

Versiyon kontrolü nedir?

Bir dokümanın oluşum sürecini ve değişik versiyonların takibi ve arşivlenmesi için kullanılan metot ve sistemlere versiyon kontrolü adı verilir. Genelde yazılım sektöründe projelerin yönetimi için versiyon kontrol sistemleri kullanılır. Birden fazla programcının kod paylaşımı ve yapılan değişikliklerin takibi için bir versiyon kontrol sisteminin kullanımı kaçınılmazdır. Daha sonra yakından inceleyeceğimiz gibi oluşturulan bir yazılım ürününün (program) değişik versiyonlarının oluşturulması ve bu versiyonlardaki hataların (bug) giderilmesi için kullanılan versiyon kontrol sistemi değişik araçlar ve yöntemler ihtiva etmektedir. Bu metotlar kullanılarak yazılım süreci desteklenir.

Bir versiyon kontrol sistemini depo gibi düşünebiliriz. Oluşturulduğumuz her doküman bu depoya gönderilir. Depo, dokümanların ve değişik versiyonlarının nasıl yönetileceğini bilir. Programcı olarak deponun içinde neler olup, bittiğini bilmek zorunda değiliz. Sadece bir dokümana gerek duyduğumuz zaman depoya başvurarak, gerekli dokümanı ediniriz. Bu doküman üzerinde değişiklik yapabilir ve değiştirilen dokümanı tekrar depoya gönderebiliriz. Bu esnada dokümanın değişik versiyonları oluşur. Oluşan değişik versiyonlar üzerinde de kafa yormamıza gerek yoktur, çünkü depo bu versiyonların yönetimi üstlenir. Depoya danışarak, bir dokümanın tarihçesini edinebiliriz. Depo hangi dokümanın kim tarafından ne zaman değiştirildiğini her zaman bilir. Kayıtlarda iz bırakmadan bir doküman üzerinde değişiklik yapmak mümkün değildir.

Versiyon kontrol sistemlerinde kullanılan depolara **repository** ismi verilir. Çoğu versiyon kontrol sistemi bünyelerinde bilgibankaları kullanarak repository oluşturmaktadırlar. Örneğin Subversion'da Berkley DB bilgibankasını kullanarak bir repository oluşturulabilir. Bunun yanı sıra versiyon kontrol sistemi tarafından normal sistem dosyaları (file repository) kullanılarak da repository'ler oluşturulabilir. Bu gereksinimlere bağlı olarak yapılması gereken bir seçimdir.

Versiyon kontrol sisteminin merkezinde bir repository olduğu için, bu repository'nin güvenli bir bilgisayar üzerinde olmasına dikkat edilmelidir. Bu bilgisayar erişilemez durumda olduğunda yada en kötü ihtimalle bozulduğunda, tüm versiyon kontrol sistemi çalışmaz hale gelecektir. Mutlaka düzenli aralıklarla repository'nin kopyası (backup) alınmalıdır.

Kod paylaşımını kolaylaştırmak için çoğu modern versiyon kontrol sistemleri bilgisayar ağı üzerinden kullanılabilir şekilde çalıştırılabilir (network enabled). Bu şekilde programcının bulunduğu yer önemini yitirmekte ve bilgisayar ağı mevcut olduğu sürece, yerden bağımsız olarak versiyon kontrol sistemi kullanılabilir. Programcı bilgisayar ağı üzerinden versiyon kontrol sisteminin barındırdığı repository'ye bağlanarak, kod paylaşımını gerçekleştirebilir.

Peki bir programcı bilgisayar ağı üzerinden erişim olmadığı durumlarda çalışmasını nasıl devam ettirebilir? Modern versiyon kontrol sistemleri devre dışı mod (disconnected yada offline mode) olarak tabir edebileceğimiz çalışma tarzını desteklemektedirler. Programcı iş yerinden ayrılmadan önce üzerinde çalışmak istediği projeyi repository'den alır ve bilgisayarına yükler. Programcı böylece kendi bilgisayarında projenin bir kopyasına sahip olur (working copy). Gün içinde programcı proje üzerinde çalışarak, bir takım değişiklikler yapar. Programcı ertesi gün tekrar ofise gelerek, yaptığı değişiklikleri repository ile senkronize eder.

Bir versiyon kontrol sistemi seçilirken, programcı ekibinin gereksinimleri göz önünde bulundurulmalıdır. Her versiyon kontrol sistemi istenilen esnekliği sağlayamayabilir. Edindiğim tecrübeler doğrultusunda Subversion versiyon kontrol sisteminin, programcı ekibin gereksinimlerine büyük oranda cevap verebileceğini söyleyebilirim.

Bir Başarı Hikayesi ...

Cem 34 yaşında, Türkiye'nin en iyi Java programcıları arasında yer alan, tecrübeli bir bilgisayar mühendisi idi. ODTÜ bilgisayar mühendisliği fakültesini bitirdikten ve askerden geldikten sonra Türkiye'nin en başarılı İnternet firmalarında çalışmış ve takım tecrübesi edinmişti. Bir projenin başarıya ulaşması için gerekli tüm metot ve araçlara hakimdi.

Her zaman olduğu gibi o günde erkenden ofise gelmiş ve çalışmaya başlamıştı. Cem ilk önce kendi bilgisayarında yer alan kodları merkezi versiyon kontrolü sistemi aracılığıyla aktualize (update) etti. Bir gün önceki mesai bitiminde üzerinde çalıştığı kodların çalışır durumda olduğunu kontrol ettikten sonra merkezi versiyon kontrolü sistemine eklemişti (commit). Takımda yer alan her programcı her sabah aynı işlemi yapardı. Bu bir alışkanlık haline gelmişti. Her programcı kendi bilgisayarında kodların bir kopyasına (local copy) sahipti. Programcılar sahip oldukları kodları her sabah aktualize ettikten sonra çalışmalarına devam ederlerdi.

Cem o sabah aktualize işlemi sırasında, takım arkadaşı olan Sevgi'nin bazı sınıflar üzerinde değişiklikler yaptığını fark etti. Sevgi Siparis sınıfını da değiştirmiş ve yeni değişkenler eklemişti. Cem'in üzerinde çalıştığı sınıflar Siparis sınıfı ile bağlantılı olduklarından, Sevgi'nin yanında giderek, yapılan değişiklikler hakkında bilgi almayı düşündü, lakin Sevgi o gün müşteri görüşmesindeydi. Cem tekrar bilgisayarının başına döndü. Merkezi versiyon kontrol sistemi proje bünyesinde oluşan her dokümanı ve bu dokümanların değişiklikler sonunda meydana gelen versiyonlarını bünyesinde barındırıyordu. Cem, Siparis sınıfında yapılan değişiklikler listesini aldığı anda, Sevgi tarafından eklenen en son yorumu gördü:

Siparis sınıfına, siparişin gönderileceği adres için gerekli değişkenleri ekledim.

Yapılan değişiklikler merkezi versiyon kontrol sistemine eklenmeden önce, her programcı yaptığı değişiklikleri açıklayıcı bir yorum eklemek zorundaydı. Böylece bu yorumları okuyarak, ne gibi değişikliklerin yapıldığını anlamak kolaydı. Ayrıca merkezi versiyon kontrolü sisteminin bünyesinde barındırdığı bazı araçlar aracılığıyla, bir dokümanın değişik versiyonlarını kıyaslamak ve yapılan değişiklikleri bu şekilde görmekte mümkündü. Her doküman kayıtsız şartsız merkezi versiyon kontrolü sisteminin himayesindeydi ve hiç kimse iz bırakmadan bir doküman üzerinde değişiklik yapamazdı. Kimin ne zaman ne yaptığı her zaman merkezi versiyon kontrolü sistemine danışılarak takip edilebilirdi.

Cem yeni Siparis sınıfını göz önünde bulundurarak, kendi kodu üzerinde gerekli değişiklikleri yaptı ve diğer takım arkadaşları ile paylaşmak için tüm değişiklikleri merkezi versiyon kontrolü sistemine ekledi. Bu işlemin ardından üzerinde çalıştığı tüm dokümanların yeni versiyonları merkezi versiyon kontrolü sisteminde yerini almıştı. Bu şekilde kodun aktualize edilmesi ve tekrar merkezi versiyon sistemi üzerinden paylaşımı tüm takımın efektif bir şekilde çalışmasını sağlıyordu.

Öğleden sonra Sevgi'den telefon geldi. Sevgi müşteride olduğunu ve orada kullanılan versiyonda bir hata bulduklarını söyledi. Bu hatanın giderilmesi gerekiyordu, çünkü müşteri için program kullanılmaz hale gelmiş ve hatalı hesaplar üretiyordu. Cem bu hatayı ofiste düzeltebilir ve müşteriye oluşturulan yeni versiyonu gönderebilirdi. Bunun için bilmesi gereken iki nokta vardı: 1.) program hatası nerede oluşuyordu? 2.) Müşteri, programın hangi sürümünü kullanıyordu. Sevgi hatanın Fatura sınıfında olduğunu ve KDV'nin yanlış

hesaplandığını söyledi. İkinci sorunun cevabı ise v.3.5 idi, yani müşterinin kullandığı sürümün numarası 3.5 idi. Müşteri için bir sürüm oluşturulmadan önce merkezi versiyon kontrol sisteminde müşteri için bir etiket (tag) oluşturulurdu. Bu etiket v.3.5 ismini taşıyordu. Bu etiket ile programın o sürümünde bulunan tüm sınıf ve dosyalarına ulaşmak mümkündü.

Cem v.3.5 etiketini taşıyan sürümü, kodları incelemek için versiyon kontrol sisteminden kendi bilgisayarına indirdi. Bir etikete sahip dosya üzerinde değişiklik yapmak mümkün değildi. Aynı etikete sahip dosyalardan herhangi birisi üzerinde değişiklik yapmak, etiketin belirli bir tarihte belirli bir amaç için oluşturulma ilkesine ters düşmekteydi. Yeni bir dalın (branch) oluşturulması gerekiyordu. Cem v.3.5 etiketini baz alarak yeni bir dal oluşturdu ve hatayı bu dal içinde giderdi. Programı test ettikten sonra v.3.6 isminde yeni bir etiket kullanarak, programın yeni sürümünü oluşturdu. Cem bu sürümü CD üzerinde kurye ile, müşteride olan Sevgi'ye gönderdi.

Bu hikaye devam eder, ama görüldüğü gibi arkadaşlar işlerine hakimler ve gözümüz arkada kalmadan onların yanından ayrılabiliriz ☺.

Çevik Süreçlerde Versiyon Kontrolü

Çevik süreçlerin olmazsa olmazlarından birisi versiyon sistemi kullanımıdır. İteratif ve inkrementel yapıya sahip olan çevik süreçlerde versiyon kontrol sistemler merkezi bir rol oynar. Bir çevik projede versiyon kontrol kullanımını gerekli kılan nedenler şöyledir:

- Her iterasyonun fiziksel olarak diğer iterasyonlardan ayırt edilebilmesi için, o iterasyon bünyesinde oluşan ve değişikliğe uğrayan dosyaların etiketlenmesi gerekir. Her iterasyon sonunda müşteriye çalışır bir sistem sunulur. Sistem iterasyonun etiketini taşır. Böylece hangi sürümün müşteri tarafından kullanımda olduğu anlaşılır.
- Birden fazla iterasyonda oluşan sistem entegre edilerek, bir versiyon numarasıyla etiketlenir. Bu versiyon müşteriye kullanım için verilir. Versiyon numarası ile hangi sürümün kullanımda olduğu anlaşılır.
- XP tekniklerinden ortak sorumluluğu (collective ownership) uygulayabilmek için programcılar arası kod paylaşımını kolaylaştırmak gerekmektedir.
- XP tekniklerinden sürekli entegrasyonu (continuous integration) gerçekleştirebilmek için merkezi bir versiyon kontrol sistemine ihtiyaç duyulmaktadır.

Subversion

Subversion¹ açık kaynaklı (open source) versiyon kontrol sistemidir. Bazı özellikleri şöyledir:

- Subversion CVS² örnek alınarak yapılmıştır. Amaç CVS'de daha iyi bir versiyon kontrol sistemi oluşturmaktır. Bu yüzden Subversion birçok CVS özelliğine sahiptir.
- Subversion dizinlerin de normal dosyalar gibi versiyonlarını oluşturur.
- Kopyalama, silme ve isim değiştirme işlemlerinde Subversion tarafından yeni versiyonlar oluşturulur.

¹ Bakınız: <http://subversion.tigris.org>

² Bakınız: <http://www.cvshome.org>

- Subversion’da yapılan işlemler ya hep ya hiç prensibiyle gerçekleşir, yani commit’ler atomiktir (atomic commits).
- Dal (branch) ve etiket (tag) oluşturulması copy işlemi kullanılarak gerçekleştirildiği için kısa sürer.
- File locking mekanizması kullanılarak, dosyaların üzerinde değişiklik yapılması engellenebilir.
- Subversion bir Apache webserver üzerinde erişilebilir hale getirilebilir.
- Svnserve komutuyla Subversion, versiyon kontrol serveri olarak görev yapabilir.

Bunun yanı sıra daha birçok özelliği olan Subversion çevik projelerde vazgeçilmez bir araç haline gelmiştir.

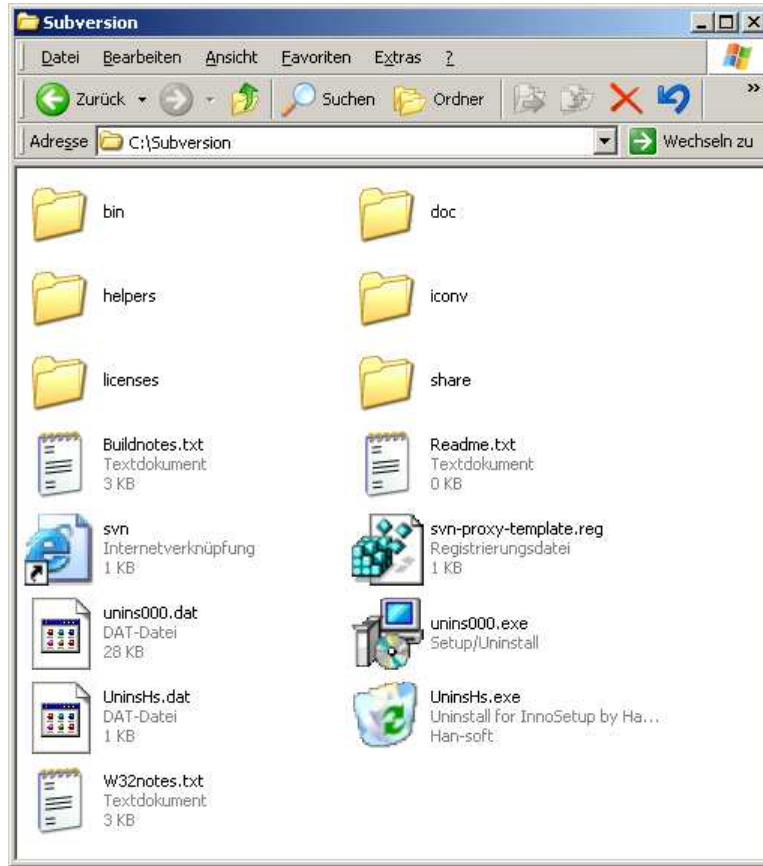
Subversion Windows Kurulumu

Bu bölümde Subversion’ın Windows işletim sistem üzerinde kurulumunu inceleyeceğiz. Kitabın yazılımı esnasında aktüel olan Subversion versiyonu 1.4.6’dır.



Resim 16.2 Subversion kurulumu

Kurulum tamamlandıktan sonra seçilen dizine bağlı olarak aşağıdaki dizin yapısı oluşacaktır.



Resim 16.3 Subversion dizin yapısı

Subversion'ın grafiksel arayüzü yoktur. İşlemleri **bin** dizininde bulunan komutlar aracılığıyla bir Dos Shell altında gerçekleştirmemiz gerekiyor.

Bin dizininde yer alan svn komutu ile kurulumu kontrol edebiliriz:

```
C:\WINDOWS\system32\cmd.exe

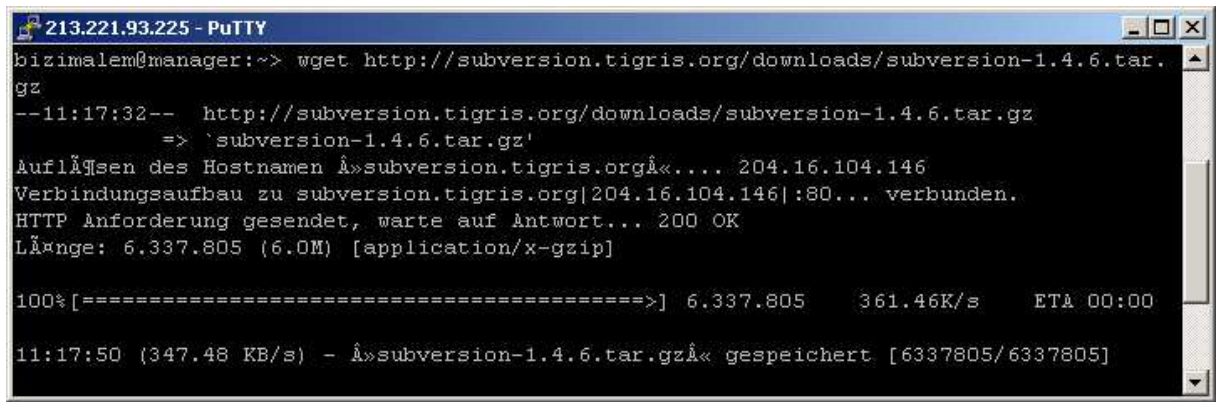
C:\Subversion\bin>svn --version
svn, Version 1.4.6 (r28521)
übersetzt Dec 20 2007, 16:19:22

Copyright (C) 2000-2007 CollabNet.
Subversion is open source software, see http://subversion.tigris.org/
This product includes software developed by CollabNet (http://www.Collab.Net/).
```

Subversion Linux / Unix Kurulumu

Subversion'ı Red Hat, FreeBSD ve Solaris gibi değişik Linux / Unix işletim sistemlerinde kullanmak mümkündür. Çoğu zaman belli bir işletim sistemi için hazırlanmış kurulum paketini kullanılabilir.

Bu bölümde Subversion'ın kaynak kodlarını kullanarak, nasıl kurabileceğimizi inceleyeceğiz. Eğer Linux / Unix işletim sistemli bilgisayarınızın İnternet bağlantısı varsa, wget komutu ile kaynak kodun yer aldığı paketi edinebilirsiniz:



```
213.221.93.225 - PuTTY
bizimalem@manager:~> wget http://subversion.tigris.org/downloads/subversion-1.4.6.tar.
gz
--11:17:32--  http://subversion.tigris.org/downloads/subversion-1.4.6.tar.gz
=> `subversion-1.4.6.tar.gz'
Aufl sen des Hostnamen  subversion.tigris.org .... 204.16.104.146
Verbindungsaufbau zu subversion.tigris.org[204.16.104.146]:80... verbunden.
HTTP Anforderung gesendet, warte auf Antwort... 200 OK
L nge: 6.337.805 (6.0M) [application/x-gzip]

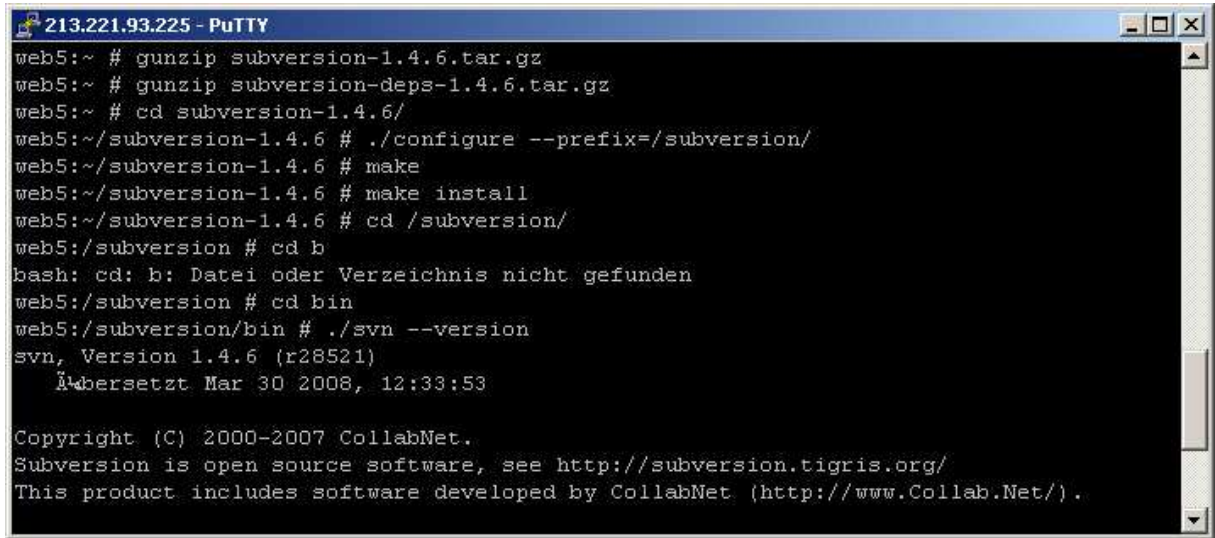
100%[=====] 6.337.805  361.46K/s  ETA 00:00

11:17:50 (347.48 KB/s) -  subversion-1.4.6.tar.gz  gespeichert [6337805/6337805]
```

Resim 16.5 wget ile kurulum paketini <http://subversion.tigris.org/downloads/subversion-1.4.6.tar.gz> lokasyonundan edinebilirsiniz

Subversion kurulumu i in kaynak kodların bulunduėu iki paketin indirilmesi gerekmektedir:

1. subversion-1.4.6.tar.gz
<http://subversion.tigris.org/downloads/subversion-1.4.6.tar.gz>
2. subversion-deps-1.4.6.tar.gz
<http://subversion.tigris.org/downloads/subversion-deps-1.4.6.tar.gz>



```
213.221.93.225 - PuTTY
web5:~ # gunzip subversion-1.4.6.tar.gz
web5:~ # gunzip subversion-deps-1.4.6.tar.gz
web5:~ # cd subversion-1.4.6/
web5:~/subversion-1.4.6 # ./configure --prefix=/subversion/
web5:~/subversion-1.4.6 # make
web5:~/subversion-1.4.6 # make install
web5:~/subversion-1.4.6 # cd /subversion/
web5:/subversion # cd b
bash: cd: b: Datei oder Verzeichnis nicht gefunden
web5:/subversion # cd bin
web5:/subversion/bin # ./svn --version
svn, Version 1.4.6 (r28521)
 bersetzt Mar 30 2008, 12:33:53

Copyright (C) 2000-2007 CollabNet.
Subversion is open source software, see http://subversion.tigris.org/
This product includes software developed by CollabNet (http://www.Collab.Net/).
```

Resim 16.6 Subversion kurulumu

Kurulum i n atılması gereken adımlar  yledir:

- gunzip subversion-1.4.6.tar.gz
- gunzip subversion-deps-1.4.6.tar.gz
- ve subversion-1.4.6
- ./configure --prefix=/subversion/
- make
- make install

Bu i lemlerin ardından /subversion/bin dizininde Subversion komutları yer alır.

Subversion Komutları

Kurulum i lemi ardında bin dizininde a ağıda yer alan programlar bulunacaktır.

svn

Programcılar tarafından kullanılan client programdır. Repository’de bulunan bir dok mana ula mak yada bir dok man  zerinde yapılan deėi iklikleri repository’ye g ndermek i in kullanılır.

```
C:\WINDOWS\system32\cmd.exe

C:\Subversion\bin>svn.exe --help
Aufruf: svn <Unterbefehl> [Optionen] [Parameter]
Subversion-Kommandozeilenclient, Version 1.4.6.
Geben Sie »svn help <Unterbefehl>« ein, um Hilfe zu einem Unterbefehl
zu erhalten.
Geben Sie »svn --version« ein, um die Programmversion und die Zugriffsmodule
oder »svn --version --quiet«, um nur die Versionsnummer zu sehen.

Die meisten Unterbefehle akzeptieren Datei- und/oder Verzeichnisparameter,
wobei die Verzeichnisse rekursiv durchlaufen werden. Wenn keine Parameter
angegeben werden, durchläuft der Befehl das aktuelle Verzeichnis rekursiv.

Verfügbare Unterbefehle:
  add
  blame <praise, annotate, ann>
  cat
  checkout <co>
  cleanup
  commit <ci>
  copy <cp>
  delete <del, remove, rm>
  diff <di>
  export
  help <?, h>
  import
  info
  list <ls>
  lock
  log
  merge
  mkdir
  move <mv, rename, ren>
  propdel <pdcl, pd>
  propedit <pedit, pe>
  propget <pget, pg>
  proplist <plist, pl>
  propset <pset, ps>
  resolved
  revert
  status <stat, st>
  switch <sw>
  unlock
  update <up>

Subversion ist ein Programm zur Versionskontrolle.
Für weitere Informationen, siehe: http://subversion.tigris.org/
C:\Subversion\bin>
```

Resim 16.7 svn komutu

svnadmin

Repository'lerin oluşturulması ve yönetiminde kullanılır. Bu program sadece Subversion serveri üzerinde çalıştırılabilir. Bilgisayar ağı üzerinden repository'lerin yönetimi mümkün değildir.

svnlook

Repository'lerin kontrolü için kullanılır. svnadmin gibi server üzerinde kullanılabilir. Bu program repository bilgilerini sadece okuyabilir, değiştiremez.

svnversion

Bu program ile üzerinde çalışılan dokümanların revizyon numarası öğrenilebilir.

svnservice

Kullanılan repository'leri bilgisayar ağı üzerinde erişilir hale getirmek için svnservice programı kullanılır. Svnservice ile bir Subversion serveri kurulmuş olur. IP Adresi üzerinden bu serverde bulunan repository'lere erişilir.

svndumpfilter

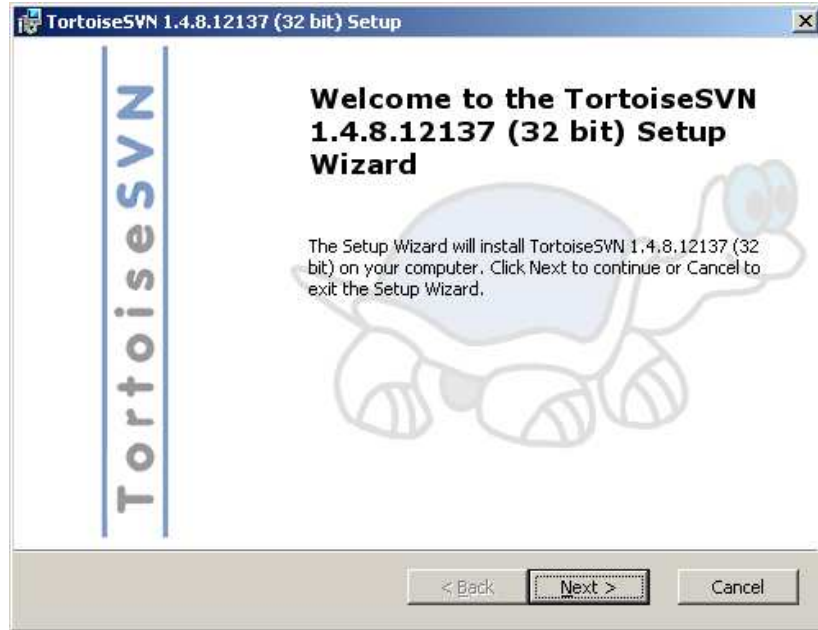
Bu filtre programı ile, svnadmin komutuyla oluşturulmuş repository yığınları (dump) içinde belirli kriterlerde arama yapılabilir.

svnsync

Bu program aracılığıyla herhangi bir repository'nin sadece okunabilir ama değiştirilemez bir kopyası oluşturulabilir. Oluşturulan kopya, ana repository ile, bu repository üzerinde yapılan değişiklikleri yansıtacak şekilde senkron tutulur.

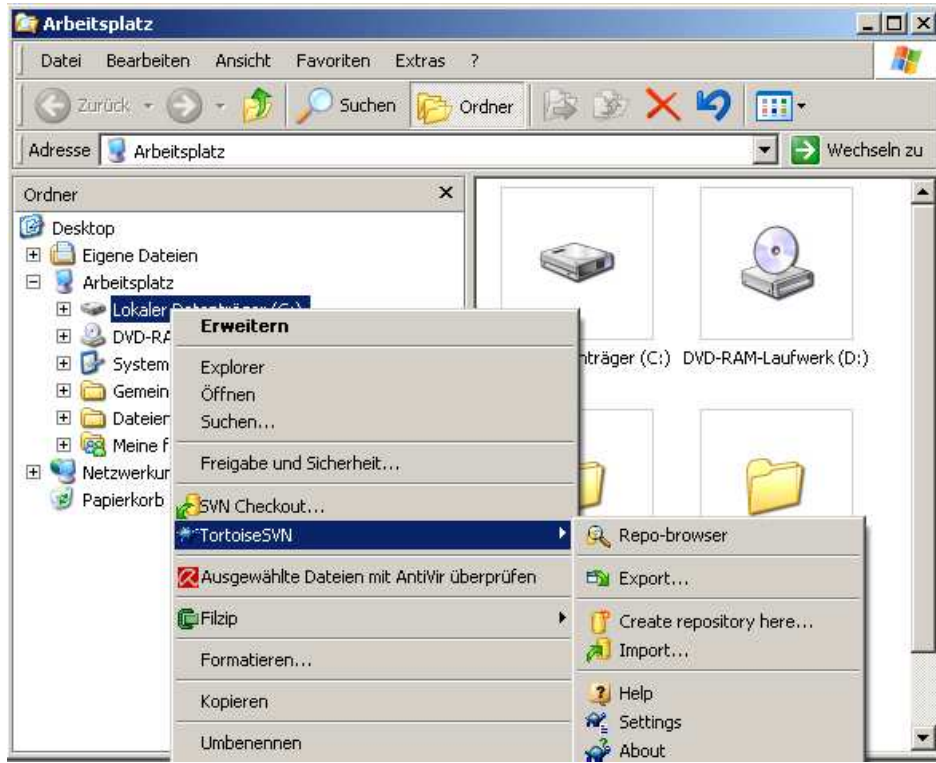
Subversion Client TortoiseSVN

Windows işletim sistemi altında Subversion ile çalışmayı kolaylaştırmak için grafiksel arayüze sahip olan TortoiseSVN programı geliştirilmiştir. Bu programı <http://tortoisesvn.tigris.org/> adresinden temin edebilirsiniz. TortoiseSVN bir kullanıcı (client) program olarak düşünülebilir. Subversion server modunda çalıştırıldığı takdirde, TortoiseSVN ile servere bağlantı kurularak, dokümanlar üzerinde işlem yapılabilir.



Resim 16.8 TortoiseSVN kurulumu

Kurulma işleminin ardından Windows Explorer içinde herhangi bir dizin üzerinde gidilerek sağ tuşa tıklandığı takdirde TortoiseSVN açılan kontekst menüsünde görünecektir.



Resim 16.9 TortoiseSVN kontekst menü

Subversion ile çalışırken hem bin dizininde bulunan svn komutu ile hemde TortoiseSVN gibi grafiksel bir client programı ile çalışabilirsiniz. Bu konudaki seçimi çalışma tarzınız uyumlu olacak şekilde yapabilirsiniz.

Repository (Depo)

Subversion bünyesinde tüm dokümanlar ve bu dokümanların değişik versiyonları repository ismini taşıyan bir depoda tutulur. Kurulum esnasında nasıl bir tip repository kullanılması gerektiği belirlenir. Subversion'ın ilk sürümleri Berkeley DB bilgibankasını repository olarak kullanırdı. Subversion'in yeni sürümlerinde FSFS ismini taşıyan ve dokümanları dosya sisteminde (filesystem) bir dosya içinde tutabilen alternatif bir repository türü bulunmaktadır. Bu iki repository türü arasında seçim yapılabilir. Hangi tür repository'nin kullanılması gerektiği gereksinimler doğrultusunda belirlenmelidir.

Bir repository, dokümanların yer aldığı bir dizin ağacı olarak düşünülebilir. Her ağaç, barındırdığı doküman ve dizinlerin belli bir zamanda yapılan değişiklikler sonucu oluşan versiyonlarını ihtiva eder. Bu versiyonlar kullanıcıların yaptığı işlemler sonrasında oluşurlar ve Subversion jargonunda revizyon (revision) olarak isimlendirilir.

Revizyon (Revision)

Revizyonun ne olduğunu bir örnek kullanarak açıklamak istiyorum. HelloWorld.java isminde bir doküman oluşturup, repository'ye eklediğimizi düşünelim. Bu andan itibaren repository'de 1 nolu revizyon oluşur. Bu revizyon numarası tüm repository genelinde geçerlidir. Aynı repository içinde birden fazla proje yer alabilir. Repository'ye değişik projelerden değişik dokümanlar eklense bile, genel revizyon numarası bir artırılabacaktır.

Subversion her dokümanın yeni versiyonu için o dokümana bir versiyon numarası atamaz. Subversion için her değişiklik yeni bir revizyon oluşturulması anlamına gelir. Örneğin 1. revizyondaki HelloWorld.java isimli dokümanın versiyon numarası yoktur. HelloWorld.java yapılan değişikliklerden sonra en son versiyonuyla 1 nolu revizyonda yerini alır.

Revizyon 1:
HelloWorld.java

Bir nolu revizyon içinde HelloWorld.java dokümanı yer almaktadır. Akabinde Test.java isminde ikinci bir doküman oluşturup, tekrar repository'ye ekliyoruz. Bu işlemin ardından revizyon 2 oluşuyor.

Revizyon 2:
HelloWorld.java
Test.java

2 nolu revizyon, bünyesinde HelloWorld.java ve Test.java dokümanlarını barındırmaktadır. HelloWorld.java üzerinde hiçbir değişiklik yapmamış olmamıza rağmen, Subversion otomatik olarak bu dokümanı da göz önünde bulundurarak yeni revizyonu oluşturdu. Bu andan itibaren 2 nolu revizyon söz konusu olduğunda, bu revizyonda yer alan iki dokümanın en son versiyonları karşımıza çıkacaktır.

Siparis.java ismini taşıyan bir dosya daha ekleyerek, 3. revizyonu oluşturuyoruz. Bu arada HelloWorld.java üzerinde bazı değişiklikler yaptık ve böylece bu dokümanın yeni bir versiyonu oluşmuş oldu.

Revizyon 3:
HelloWorld.java
Test.java
Siparis.java

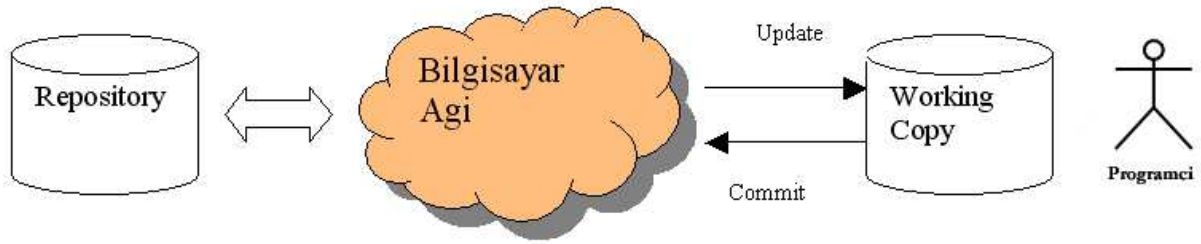
İlk bakışta 3 nolu revizyonda yer alan HelloWorld.java dokümanı üzerinde yapılan değişiklikleri görmemiz mümkün değildir, çünkü bu doküman bir versiyon numarası taşımamaktadır. Sadece en son versiyonun yer aldığı 3 nolu revizyonu görmekteyiz.

Subversion bir dokümanın tarihçesine (history) bakmamızı mümkün kılmaktadır. 3. revizyonda bulunan HelloWorld.java dokümanın tarihçesine bakarak, kim tarafından hangi değişikliklerin yapıldığını görmemiz mümkündür.

1. ve 3. revizyon kıyaslandığında Test.java ve Siparis.java dokümanlarının yeni eklendiğini ve HelloWorld.java dokümanı üzerinde değişiklikler yapıldığını tespit ederiz. Subversion repository içinde revizyon numaraları üzerinde her dokümanı takip ettiği için, yapılan değişiklikleri kolaylıkla gösterecektir.

Working Copy (Üzerinde Çalışılan Kopya)

Bir proje bünyesinde oluşan tüm dokümanlar Subversion tarafından oluşturulan repository içinde yer alır. Bir programcının kod üzerinde çalışabilmesi için, repository’de bulunan dokümanların bir kopyasına sahip olması gerekmektedir. Programcının sahip olduğu bu dokümanlara working copy adı verilir. Programcı yaptığı her değişikliği repository’ye gönderebilir (commit) ve en son yenilikleri repository’den alabilir (update).



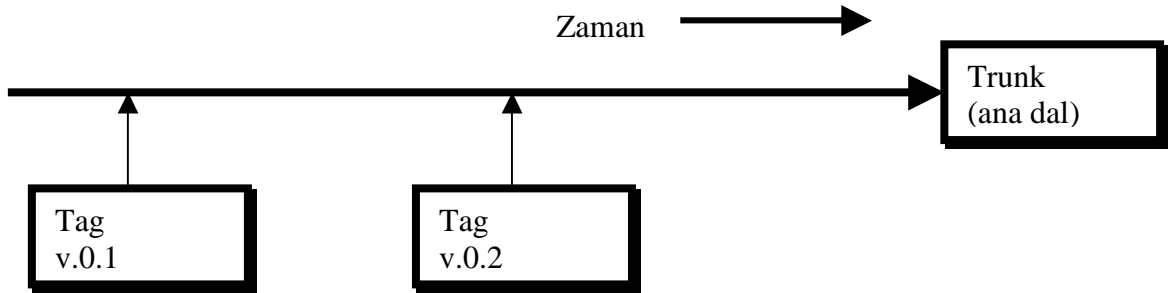
Resim 16.10 Working copy

Working copy oluşturma işlemine Subversion dilinde checkout (çıkış yapmak) ismi verilir. Programcı genelde ufak ve orta boyllu projelerde, projeye ait tüm dokümanların birer kopyasını kendi bilgisayarına almış olur. Proje eğer büyük bir yapıya sahipse, projenin alt modülleri yada belirlenmiş kısımları working copy olmak üzere checkout yapılır.

Tag (Etiket)

Subversion’da değişik versiyonların revizyon numarası üzerinden yönetildiğini gördük. Sonuç itibariyle revizyon numarası bir rakam olduğu için akılda kalıcı olmayabilir. Programcılar arasında değişik program versiyonları için version1, release2 gibi isimlerin kullanılması daha yaygındır.

Subversion ile projenin belirli bir aşamasında, o zamana kadar yapılmış tüm çalışmalarını bir etiket altında toplamak amacıyla tag konsepti uygulanabilir. Herhangi bir isim seçilerek projenin belirli bir aşamaya ulaştığını ifade etmek için etiket (tag) oluşturulur. Bu bir fotoğrafın çekilmesi gibi o anki anlık görüntünün alınması anlamına gelir.

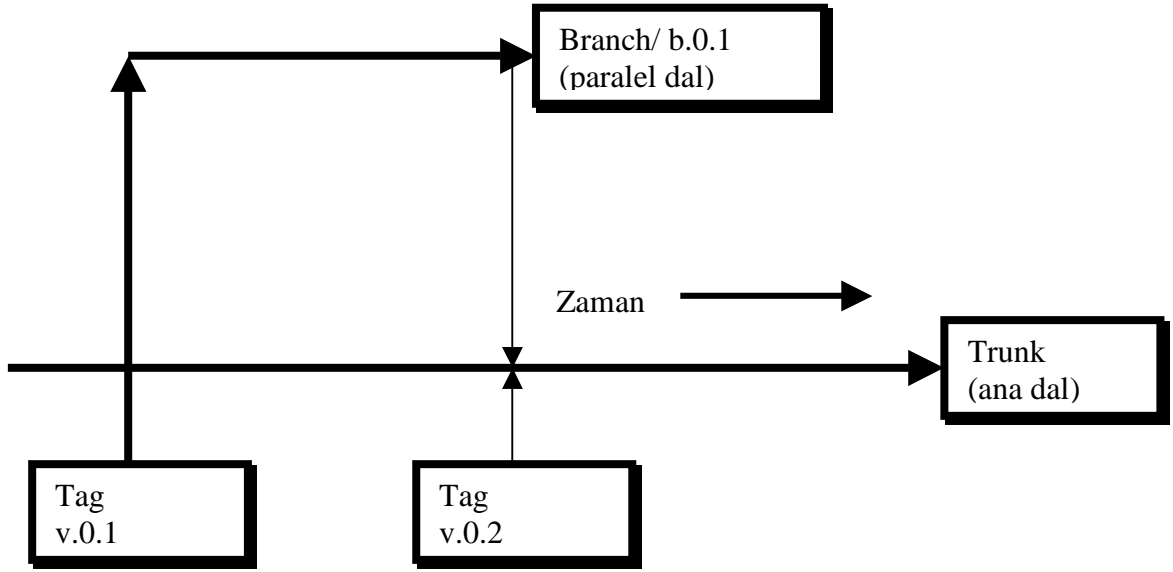


Resim 16.11 Tag konsepti

Etiket ismi kullanılarak, projenin bu aşamadaki durumu daha sonra repository’den edinilebilir. Tag kullanılarak oluşturulmuş versiyonlar repository’den sadece okuma (readonly) amacıyla alınmalıdır. Yapılacak herhangi bir değişiklik yine aynı tag altında kayıtlandığı için, etiket ile oluşturulan anlık görüntü bozulmuş olur.

Branch (Dal)

Bir iterasyon sonunda müşteriye yeni bir sürüm oluşturarak teslim ettiğimizi düşünelim. Müşteri bu sürümü kullanırken, bizde yeni bir iterasyonda projeye devam ediyor olalım. Sürüm için yeni bir versiyon tagı (etiket) kullandık, örneğin v.0.1. Müşteri programı kullanırken bir hatanın oluştuğunu bize bildirmiş olsun. Bu durumda üzerinde çalıştığımız ana kod üzerinde hatayı gidererek, müşteriye yeni bir sürüm veremeyiz, çünkü üzerinde çalıştığımız ana kod çalışır ve stabil durumda değil yada yeni implementasyonlar tamamlanmadı. Bu durumda yapılabilir tek şey: v.0.1 versiyonunu baz alarak, yeni bir dal oluşturmamız ve hatayı o dal üzerinde gidermemiz gerekiyor. Yeni bir dal oluşturduğumuz zaman ana koda paralel olarak başka bir versiyonu baz alan yeni bir yazılım kanadı oluşur.



Resim 16.12 Dal konsepti

Resim 16.12 de dal konsepti gösterilmektedir. Trunk x (zaman) ekseninde ilerleyen ana kod dalıdır. Oluşan hataları gidermek için trunka paralel olarak yeni bir dal (branch) oluşturmamız gerekir. Hatalar bu dal üzerinde giderildikten sonra yeni bir versiyon (v.0.2) oluşturduktan sonra paralel dal üzerinde yaptığımız değişiklikleri trunka aktarırız (merge). Bu işlemin ardından hem ana dal rahatsız edilmeden hata giderilerek yeni bir sürüm oluşturulur, hemde hataların giderilmesi için yapılan değişiklikler merge metoduyla ana dala aktarılır.

Trunk (Ana dal)

Trunk repository’de bulunan ana kod dalıdır. Yapılan tüm değişiklikler trunk içinde yer alır. Tüm etiket ve dallar trunk üzerinden gerçekleştirilir.

Merge (Birleştirme)

Eğer belirli bir sürüm için hata gidermemiz gerekiyorsa, trunka paralel yeni bir dal (branch) oluşturmamız gerekiyor. Burada yapılan değişikliklerin tekrar trunka aktarılmasına merge adı verilir.

Subversion Server

Programcılar arasında kod paylaşımını sağlayabilmek için Subversion serverin kurulması gerekmektedir. svnserve.exe programıyla Subversion’ı server modunda çalıştırabiliriz.

```
C:\WINDOWS\system32\cmd.exe - svnserve.exe -d
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Dokumente und Einstellungen\acar>cd c:\_open-source
C:\_open-source>cd Subversion
C:\_open-source\Subversion>cd bin
C:\_open-source\Subversion\bin>svnserve.exe -d
-
```

Resim 16.13 svnserve.exe

Resim 16.13 de görüldüğü gibi svnserve.exe -d (daemon) Subversion serverini aktif hale getirir. Subversion serverin aktif hale gelmesi, daha önce bir repository oluşturulmamışsa bir anlam taşımaz. Bu sebepten dolayı ilk önce bir Subversion repository'sinin oluşturulması gerekir.

```
C:\WINDOWS\system32\cmd.exe

C:\_open-source\Subversion\bin>svnadmin.exe create c:/repo
```

Resim 16.14 svnadmin.exe

Resim 16.13 de svnadmin.exe programı kullanılarak c:\repo dizininde bir repository oluşturulmaktadır.



Resim 16.15 Repository dizin yapısı

Subversion server ayarları, kurulan yeni repository'nin conf dizininde bulunan svnserve.conf dosyası üzerinden yapılır.

Kod 15.1 svnserve.conf

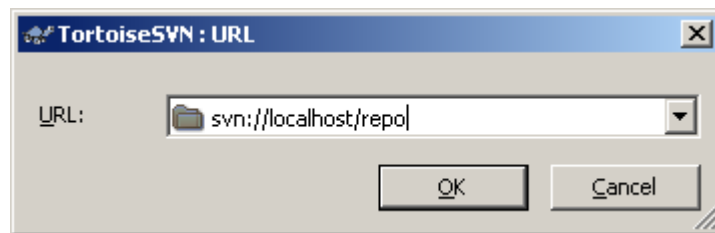
```
[general]  
password-db = userfile  
realm=Shop-Projesi  
auth-access = write
```

Bu dosyanın en basit hali kod 16.1 de yer almaktadır. password-db anahtarıyla kullanıcıların isim ve şifrelerinin yer aldığı bir dosya tanımlanır. Bu dosyanın içeriği kod 16.2 de yer almaktadır.

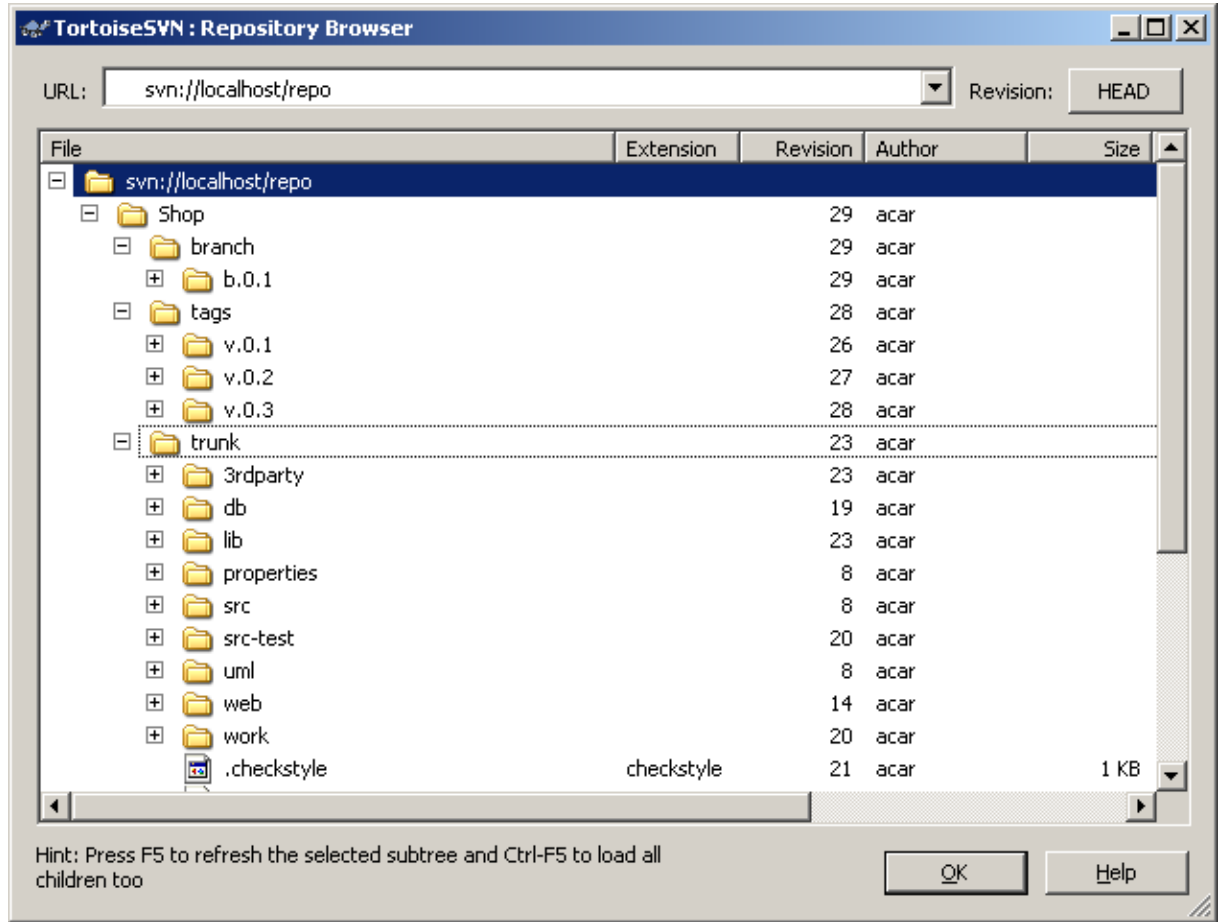
Kod 15.2 userfile

```
[users]  
acar=acar
```

Bu işlemlerin ardından herhangi bir Subversion client programı ile repository'ye bağlanabiliriz.



Resim 16.16 Repository adresi



Resim 16.17 Repo browser

Subversion Proje Dizin Yapısı

Resim 16.17 de genel proje dizin yapısı yer almaktadır. Her proje için aşağıdaki dizin yapısı tavsiye edilmektedir:

- Proje_ismi/trunk
- Proje_ismi/branch
- Proje_ismi/tags

trunk dizini içinde üzerinde aktif çalışılan kodlar yer alır. Programcılar değişiklikleri trunk üzerinden birbirleriyle paylaşırlar.

branch dizininde değişik versiyonlar baz alınarak oluşturulmuş ve değişik versiyonlarda oluşan hataları gidermek için kullanılan trunka paralel kod dalları yer alır. Örneğin v.0.1 de oluşan bir hatayı düzeltmek için b.0.1 isiminde bir branch oluşturulur. Hata bu branch içinde giderildikten sonra yapılan değişiklikler trunk ile merge edilir ve b.0.1 silinir.

tags dizininde değişik zamanlarda etiket kullanılarak oluşturulan program versiyonları yer alır. Bu versiyonlar müşteriye gönderilen sürümlerdir. Müşteri tarafından tespit edilen sürüm hataları, tags dizininde yer alan bir versiyon baz alınarak oluşturulan branch içinde giderilir.