3 Mart 2007 – Bilgi Üniversitesi Özgür Yazılım ve Açık Kaynak Günleri

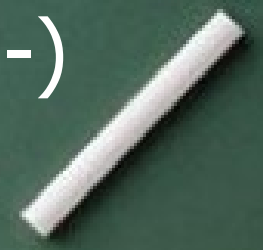
Başarılı Yazılım Projelerinin Sırrı:

Değişimi Kucaklamak

http://enveraltin.com/ - enver.altin@linux.org.tr

Doğru yok!

- Bu sunumda mutlak doğru herhangi bir bilgi yok, bu dahil. Beni sorumlu tutmayın!
- Evde deneyebilirsiniz.
- Kötü örneklerden edinilen tecrübe...
- Başarı ne demek?
- Bu konudaki ilk sunum denememem :-)



Sunum kimler için?

- Yazılım geliştiriciler
- Proje yöneticileri
- Yazılım geliştiricilerin işverenleri
- Yazılım projelerinin müşterileri

Özet

- Biraz psikoloji, biraz tarih, biraz teori
- Geleneksel yazılım geliştirme süreçleri
- Canlı yayın kötü örnek: birlikte üretelim
- Çevik geliştirme süreçleri
- İletişim ve işbirliği kültürü

Bir yazılımcının psikolojisi

- Sanatkâr mı, zanaatkâr mı?
- Mühendis mi, bilim adamı mı?
- Bir sonraki zor soruna meydan okumak
- Çay/kahve/alkol bağımlı, uykuya düşman, gözlüklü göbekli, insan ilişkilerinde sorunlu
- Sistem yöneticisinden ne farkı var?
- İlerleyen yaşlarda memuriyet

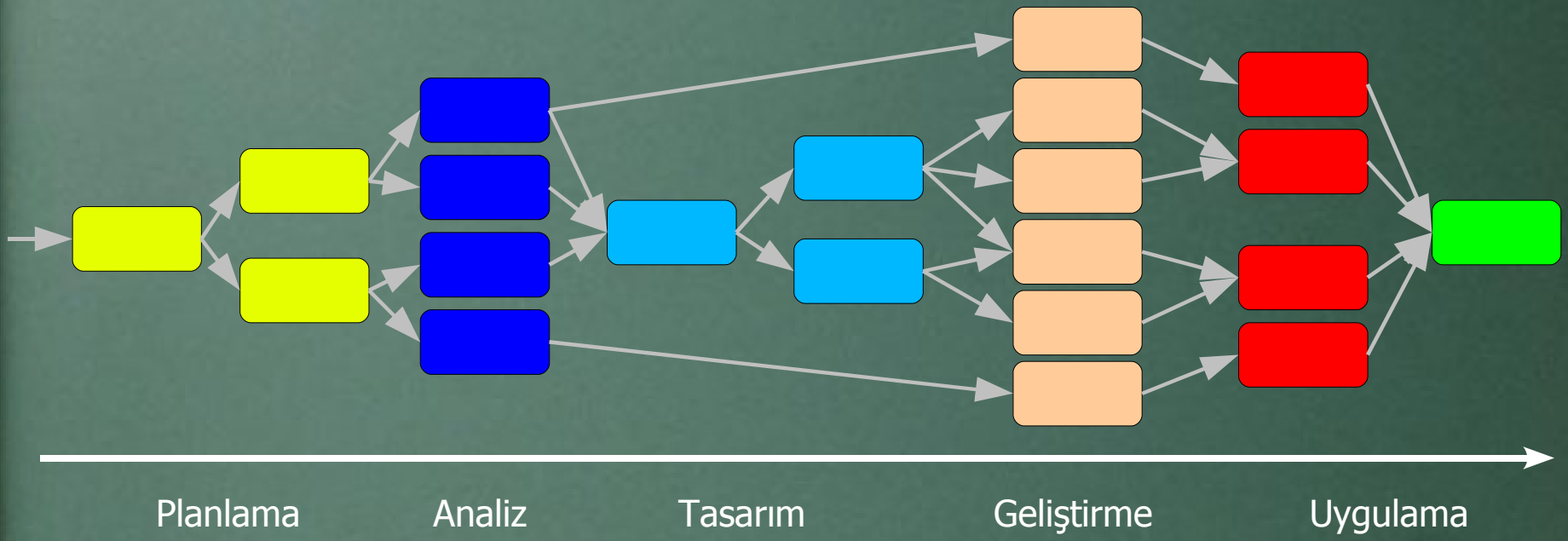
Biraz tarih

- Yarım asırlık yazılım seceresi
- NATO Yazılım Mühendisliği Konferansı'nda "yazılım krizi" tanımı (1968)
- Yazılım geliştirme süreçlerini mühendislik disiplinine benzetme çabaları
- İnsanoğlu entropiye direniyor mu?

Proje planlama

- Yeni bir terim, 2. dünya savaşından bu yana kullanılıyor.
- Türkiye'de ilk defa (bugün de en etkin kullanan) 1950'lerde inşaat sektöründe kullanılmış; pek değişmemiş.
- Önce plan, plana göre faaliyet.

Geleneksel süreç: Şelale modeli



Şelale modeli (2)

- Projede sürpriz olmayacağından eminsek, işin tamamını biliyorsak, değişme ihtimali yoksa uygulanması makul.
- Paralel yürüyen çok fazla proje varsa, personel projeler arasında sürekli yer değiştiriyorsa uygulanması makul.

Spiral model

Spiral model (2)

- 1988'de Barry Boehm tarafından geliştirildi
- Daha uzun süreli, gereksinimlerin belirli ölçüde kontrol altında olduğu ve değişikliklerin sınırlı olduğu projeler için
- Proje ekibi hedef ürün hakkında bilgi ve tecrübeye sahip değilse makul
- İlk yinelemeli geliştirme süreci

Canlı yayın kötü örnek

- Bir müşteri bulalım.
- Hedefleri ve gereksinimleri belirleyelim.
- Gerçek dünya etkenleri :-)
- Tuz, biber ekleyelim.
- Sonuçları değerlendirelim.
- Aynı hataları tekrarlıyor muyuz?

Gerçek dünya etkenleri

- Gereksinimler değişir
- Müşteri değişir, müşterinin fikirleri değişir
- Proje ekibi veya ekipteki bireyler değişir
- Teknoloji ve araçlar değişir

Başarı nedir?

- Bir yazılım projesinin başarılı olup olmadığının temelde 2 ölçüsü var:
  - **Yönetim başarısı:** Ürünü zamanında, bütçeye uygun maliyetle ve kabul edilebilir kalite ile teslim etmek.
  - **Ürün başarısı:** Ürünün müşterinin hedeflerine olan etkisinin beklentileri karşılaması.

Kıvrak (Agile) Süreçler

- **Bireyler ve etkileşimler,** süreçler ve araçlardan;
- **Çalışan yazılım,** kapsamlı belgeden;
- **Müşteri ile işbirliği,** sözleşme pazarlığından;
- **Değişimi karşılamak,** bir planı izlemekten daha değerlidir.

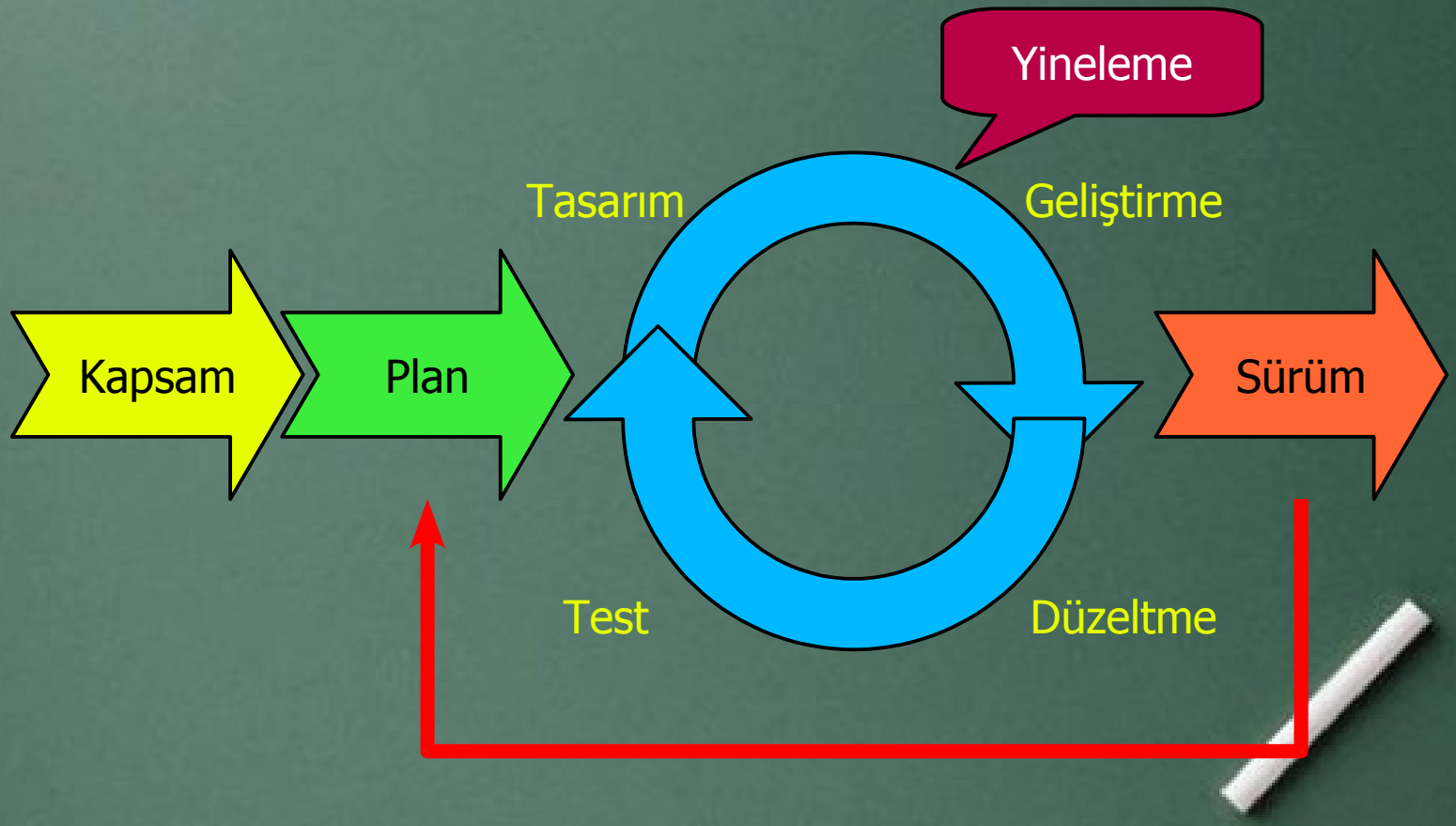
Değişimi kucaklamak

- Değişimi kabullenmek ve bununla yaşamaya alışmak önemli.
- Bugün için tasarlamak ve geliştirmek.
- Değişimin sonuçları.
- Herkes haberdar olmalı.

Değişimle yaşamak

- En kolay çözüm: yinelemeli süreçler
- Kendi içinde analiz, tasarım, geliştirme, test ve uygulama süreçlerini barındıran daha kısa süreli (1 hafta-1 ay) adımlar
- Sıklıkla hazırlanan ve yalnızca tamamlanmış işlevleri içeren sürümler.
- Sürekli iletişim ve geri besleme.

Yinelemeli süreçler



Çevik süreçler: Temel kavramlar

- Görevleri değil, işlevleri tamamlıyoruz.
- Değişimi engellemek yerine ona uyum sağlıyoruz.
- Yazılımın çalışması, belgelenmesinden daha önemli.
- Yineleme adımlarını kararların zamanında verilmesini ve doğru önceliklendirmeyi sağlamak için kullanıyoruz.
- Planlama ve süre tahmini takım işidir, bir proje yöneticisi bu işleri tek başına yapmamalıdır.
- Geri besleme projenin gelişimi boyunca sürekli olarak sağlanmalı, projenin sonunda gelen yorumlar yararlı değil!
- Hiçbir süreç iyi insanların yerini alamaz.

Extreme Programming

- Öncelikli olarak yazılım geliştiriciyi hedefleyen tek çevik geliştirme süreci.
- Geliştirici odaklı; test bağımlı.
- Bir XP takımında herkes geliştirici olabilir (müşteri, analist, test ekibi, yönetici...)
- İkili/çift programlama.
- Sürdürülebilir hız.
- Müdahil müşteri; küçük ve bir arada çalışan takımlar.

Scrum

- Yönetim odaklı, belirsiz ve deneysel süreç.
- Geliştirme sürecinin hedef ürünü etkileyen tüm yönlerini müşteriye görünür kılıyoruz.
- Müşterinin ürünü sıklıkla ve düzenli olarak incelemesini olanaklı kılıyoruz.
- İncelemelerde kabul edilebilir sınırların dışında bir sapma bulursak müşterinin veya projenin yeni duruma uyması gerek.
- Geliştiricilerin günlük işlerine karışmıyor.
- Sıklıkla XP ile birlikte kullanılıyor.

Diğer popüler metodolojiler

- Crystal/Crystal Clear
- Feature-Driven Development
- Adaptive Software Development
- Lean Software Development
- Dynamic Systems Development Method

Ne zaman çevik olmalı?

- Proje hedefi kritik değilse (insan hayatı)
- Geliştiriciler tecrübeli ise
- Gereksinimler sık sık değişiyorsa
- Takım çok büyük değilse (<20)
- Kaos altında daha iyiye giden bir kurum kültürü varsa, düşünmeyin bile!

Ne zaman geleneksel yola gitmeli?

- Proje hedefi insan hayatını etkiliyorsa
- Geliştiriciler tecrübeli değilse
- Gereksinimler büyük ölçüde belliyse ve pek değişmiyorsa
- Takım büyükse (>20)
- Kurum kültürü düzen emrediyorsa... geleneksel yoldan gidiyormuş gibi yapmak uygun olabilir :-)

Nasıl çevik olunur?

- Kaynak kod yönetim sistemi kullanarak.
- Yapılacak işleri ortak bir yere kaydederek.
- Kaynak kodun derlenmesi ve sürüm hazırlanması işlemlerini otomatik hale getirerek.
- Yazılımın bir test sistemi üzerine kurulması işlemini otomatik hale getirerek.
- Geliştiricilerin çalıştıkları kod parçalarını düzenli olarak bir araya getirerek.

Nasıl çevik olunur? (2)

- Basit tasarım.
- Düzenli temizlik: refactoring.
- İkili/çift programlama.
- Kod tabanına ortaklaşa sahip çıkmak.
- İşlevler veya işlev grupları için küçük takımlar oluşturmak.
- Birim testleri kullanmak.

Test-güdümlü geliştirme

- Alışması biraz zor ;-)
- İşlevleri tasarlamadan önce, nasıl test edileceklerini belirlemek ve işlevi test edecek kodları yazmak gerek.
- Tasarımda basitlik, yeniden kullanılabilir kod tabanı getiriyor.
- Özellikle çalışanların sık sık değiştiği takımlarda hayat kurtarabilir.

Çevik olmayan yönetim

- Şelale yaklaşımında proje oldukça detaylı ve kapsamlı bir planlama ile başlıyor.
- Proje yöneticisi plan ve projenin gerçek durumu arasındaki farkı kapatabilmek için bolca zaman harcar, sonunda vazgeçer.
- Plan zıvanadan çıkar, işe yaramaz eski bir döküman haline gelir.

Çevik yönetim

- Başlarken basit ve üstü kapalı bir plan.
- İlk sürüm için işaretlenmiş işlev listesi.
- Her yinelemeden önce takım kendisine bir miktar iş seçer. Bu seçim sırasında geliştirme hızı hesabı kullanılır.
- Projenin yaşam süreci boyunca planlama, değerlendirme ve ölçme faaliyetleri.
- Bir işlev ya tamamdır, yada değildir. ("70% bitti" için tebrik yok)

İşlevlerin yönetimi

- Tüm işlevler listelenir, 4 seviye (mutlak gerekli, eklenmesi iyi olur, mümkünse yapalım, önemsiz) kullanılarak önceliklendirilir ve sıralanır.
- İşlevsel olmayan gereksinimler (teknik ve sanatsal işler) de birer işlev gibi önceliklendirilir ve plana eklenir.

Süre tahmini

- Bir iş için süre tahminini mutlaka işi yapacak olan kişi yapmalı.
- Süre tahminleri saklanmalı ve işler tamamlandıktan sonra olası sapmalar ölçülmeli.
- Ölçme sayesinde projede çalışanlar zamanla daha belirgin tahminler yapabilmeye başlıyorlar.

Bitmiş işin tanımı

- Birim testleri olan ve bu testleri geçen
- Varsa kullanıcı kabul testlerini karşılayan
- Kurulmaya ve kullanılmaya hazır
- Mümkünse/gerekliyse belgelenmiş

Bir projeye başlarken

- Projenin görevi tanımlanmalı.
- Proje ürününün sağlayacağı işlevler listesi
- İlk sürüm planı

İşlev	Hedef süre	Yineleme	Sürüm
Geliştirme ortamı	5 gün	1	1
Görev listesi ekranı	2 gün	1	2
...	...	...	...

Çevik müşteri

- İş alanını iyi bilmeli.
- Geliştirilecek sistemin amacını hem bir kullanıcı olarak, hem de stratejik ve organizasyonel olarak anlamalı.
- Proje takımı müşteriye kolaylıkla erişebilmeli, birlikte çalışabilmeli, yeterince zamanı olmalı.
- Karar vermekte istekli ve yetkili olmalı, verdiği kararların sorumluluğunu almalı.
- Mükemmeliyetçi olmamalı.
- Bir de politik beceri sahibi olsa harika olur!

İletişim kültürü oluşturmak

- Kararları fikir birliği ile vermek.
- Birlikte ve yakın çalışma ortamı.
- Önemli bilgileri görünür bir yere asmak.
- Tepeden tırnağa açık fikirlilik.
- Az konuşup çok dinlemek.
- Çok konuşup az e-posta göndermek.
- İnsanlara çalışacak zaman vermek.
- Günlük veya günaşırı *kısa* toplantılar (kötü haber, hiç haber gelmemesinden iyidir).

Belgeleme

- Gerekliliğini sorgulamak gerek.
- Mümkün olan en basit çözüm, en iyisidir.
- Bir kere okunup unutulacaksa eğitim önermek.
- Kurumsal onay/imza mekanizmaları...
- Belgeleme işlerini de bir işlev gibi proje planına eklemek.
- Javadoc veya Wiki?
- Belgeyi isteyenleri belgeleme işine ortak etmek.
- Başka birine yaptırmak? Hmm...

Çevik projeler için İK süreci

- İletişim ve işbirliği yapabilme becerisi
- Dürüstlük.
- Alçakgönüllülük
- Merak
- Öğrenme arzusu

Kaynaklar

- Integrating Agile Development in the Real World, Peter Schuh.
- Agile Software Development, Alistair Cockburn.
- Agile Project Management with Scrum, Ken Schwaber.
- http://www.agilealliance.org/