Erciyes Üniversitesi
Bilgisayar Mühendisliği Bölümü

BZ313 Yazılım Mühendisliği

2. Uygulamada Yazılım Geliştirme

Dersin Genel Amacı

Bilgisayar hakkında zaten çok şey bildiğinizi, makul bir şekilde programlayabildiğinizi ve çalışırken daha fazla bilgi edinebileceğinizi varsayıyorum.

Ancak yazılım geliştirmedeki başarı veya başarısızlık, iyi kod yazmanın ötesinde birçok faktöre bağlıdır.

Erciyes'ten ayrıldığınızda, başarı veya başarısızlığın milyonlarca dolara mal olabileceği üretim projelerinde çalışabilirsiniz.

Oldukça kısa süre içerisinde proje sorumlusu olabilirsiniz. Söz edilen bu para size ait olabilir.

Şimdi hatalarınızı yapmanızı ve hatalarınızdan ders çıkarmanızı istiyorum.

Dersin Teması: Mesleki Sorumluluk

Kuruluşlar yazılım geliştiricilerine güvenir:

- Yetkinlik: Etkili çalışmayan yazılımlar bir firmayı yok edebilir.
- Gizlilik: Yazılım geliştiricileri ve sistem yöneticileri çok gizli bilgilere (ör. ticari sırlar, kişisel veriler) erişebilir.
- Yasal ortam: Yazılım karmaşık bir yasal ortamda (ör. fikri mülkiyet) bulunur.
- Kabul edilebilir kullanım ve kötüye kullanım: Bilgisayarın kötüye kullanımı bir kuruluşu felç edebilir (örneğin, İnternet solucanı).

Akademik Dürüstlük ve Mesleki Uygulama

Yazılım Mühendisliği işbirlikçi bir faaliyettir. Birlikte çalışmak oldukça faydalıdır ancak...
Bazı görevler bireysel çalışma gerektirebilir.

- Her zaman elinizdeki kaynaklara ve işbirliği yaptığınız kişilere güvenin ve itibar edin.

Doğru ve profesyonel eylem: Başkalarının uzmanlığından yararlanmak ve önceki çalışmalara dayanarak, uygun atıflarda bulunmak.

Etik olmayan ve akademik intihal: Başkalarının çalışmalarını atıf yapmadan kullanmak.

Ders Teması: Yazılım Pahalıdır

Yazılım pahalıdır

En büyük maliyet genellikle geliştirme ekibinin maaşlarıdır

Parayı kim ödüyor?

Bu kişi veya kuruluş ne istiyor?

- Başarı nedir?
- Başarısızlık nedir?

Teknik insanlar, organizasyondan sorumlu kişilerden çok farklı başarı kriterlerine sahip olabilirler.

Dersin Teması: Çeşitlilik

Yazılım ürünleri çok çeşitlidir

Uygulamalar: web siteleri, akıllı telefonlar

Sistem yazılımı: işletim sistemleri, derleyiciler

İletişim: yönlendiriciler, telefon anahtarları

Veri işleme: telefon faturalandırması, emekli maaşları

Gerçek zamanlı: hava trafik kontrolü, otonom araçlar

Gömülü yazılım: aygıt sürücüleri, denetleyiciler

Mobil cihazlar: dijital kamera, GPS, sensörler

Bilgi sistemleri: veritabanları, dijital kütüphaneler

Ofisler: kelime işleme, video konferanslar Bilimsel: simülasyonlar, hava durumu tahmini

Grafik: film yapımı, tasarım

vb., vb., vb., ...

Çeşitlilik

Yazılım geliştirme sanatı

Yazılım ürünleri çok çeşitlidir

Müşteri gereksinimleri çok farklıdır

Yazılım mühendisliği için standart bir süreç yoktur

- En iyi dil, işletim sistemi, platform, veritabanı sistemi, geliştirme ortamı vb. Yoktur.

Yetenekli bir yazılım geliştiricisi çok çeşitli yaklaşımlar, yöntemler ve araçlar hakkında bilgi sahibidir. **Yazılım geliştirme sanatı**, her proje için uygun yöntemleri seçmek ve bunları etkili bir şekilde uygulamaktır.

Ders Teması: Client, Customer ve Kullanıcılar

Müşteri (Client)

Client (müşteri), yazılım geliştirme ekibinin yazılımı kendisi için oluşturduğu kişidir (veya bir grup insandır). Client ve Customer Türkçe'de müşteri olarak çevriliyor olsa da farklı anlamlara gelmektedir.

- Müşteri para gibi kaynaklar sağlar ve karşılığında bir ürün bekler.
- Müşterinin iş başarısı, yazılım projesinin başarısına bağlı olabilir.

Müşteri memnuniyeti, bir yazılım projesinde başarının birincil ölçüsüdür.

Genel amaçlı bir paketin müşterisi kimdir (örneğin, Microsoft Excel)?

Client, Customer ve Kullanıcılar

Müşteri (Customer)

Müşteri, yazılımı satın alan veya bir kuruluş tarafından kullanılmak üzere seçen kişidir. Client doğrudan bir hizmet için üreticiyle iletişime geçerken customer var olan bir ürünü satın alan kişidir.

Kullanıcı

Kullanıcı, yazılımı gerçekten kullanan bir kişidir.

- Kişisel yazılımda, kullanıcı müşteri ile aynı kişi olabilir.
- Firmalarda, müşteriler ve kullanıcılar genellikle farklıdır.

Erciyes Üniversitesi yönetimindeki birçok kişi Microsoft Excel kullanıyor. Müşteri kimdir? Kullanıcılar kimlerdir?

Dersin Teması: Risk

Birçok (muhtemelen çoğu) yazılım geliştirme projesi çeşitli zorluklarla karşılaşır.

Sorunlar:

- Beklendiği gibi çalışmıyor (İŞLEV)
- Bütçeyi aştı (MALİYET)
- Geç teslimat (ZAMAN)

Rekabet Eden Hedefler

Her yazılım projesinin **işlev**, **maliyet** ve **zaman** arasında bir denge vardır.

Her bir ekstra işlev geliştirme test, bakım gibi ekstra maliyete neden olur.

Ödeme yapan kişi için önemli olan nedir?

Risk

Yazılım geliştirme projelerinin başarısızlıkları şirketleri iflas ettirebilir.

Bir yazılım geliştirme projesinin başarısızlıkları genellikle üst düzey yöneticilerin işlerine mal olur.

Örnek: Apple'ın harita uygulaması.

Risk

Yazılımların projelerinin çoğu başarısız olur (belki de %50'si hiç kullanılmaz).

Yazılım geliştiricileri yanlış yazılım oluşturduğu için birçok yazılım projesi başarısız olur.

Yazılım geliştirme ekibi şunları yapmalıdır:

- Müşterinin yazılımdan ne beklediğini anlamalı.
- Müşterinin (Client) firmasının bu müşteriden ne beklediğini anlayın.
- Müşterilerin (Customer) ve kullanıcıların yazılımdan ne beklediğini anlayın.

Yazılım geliştirme ekibi genellikle teknik içgörüler ve öneriler ekleyecektir, ancak unutmayın ki:

Müşteri (Client) memnuniyeti, bir yazılım projesinde başarının birincil ölçüsüdür.

Riski En Aza İndirgeme: Müşteri (Client) ile İletişim

- **Fizibilite çalışmaları** (Bir projeye başlayıp başlamama).
- **Gereksinimlerin** (müşterinin ne istediği) **tasarımdan** ayrılması

  (geliştiricilerin gereksinimleri nasıl karşıladığı).
- **Kilometre taşları** (geliştiricilerin müşterilere (client) ilerlemeyi nasıl bildirdiği veya gösterdiği) ve **yayımlama (release)**.
- **Kabul** (istemcinin yazılımın gereksinimleri karşıladığını nasıl test ettiği) ve **kullanıcı testi**.
- **Devir teslim** (Müşterinin uzun bir süre boyunca çalıştırılabilen ve bakımı yapılabilen bir paket almasını sağlamak).

Riski En Aza İndirme: Görünürlük

Görünürlük

Sorumluluk alan insanlar, nelerin gerçekleştiğini bilmelidir

(Yönetici tarafından görülen) bir problem

Çeşitli zorluklar veya ilerleme raporları için başkalarına güvenmelidir.

... ancak *yazılım geliştiricileri*

Süreci takip etmede zorluk yaşayabilir.

- İlerleme konusunda genellikle iyimserdirler.
- Raporlamayı boşa harcanan zaman olarak değerlendirirler.

vs.

Projeleriniz hakkında düzenli ilerleme raporları hazırlamanız gerekecektir.

Çalışan bir yazılım mükemmel bir görünürlük sağlar.

Riski En Aza İndirme: Kısa Geliştirme Döngüleri

Risk, çalışan yazılımların sık sık teslim edilmesiyle en aza indirilebilir (aylar yerine haftalar).

- Böylece client, customer ve kullanıcılar geliştiricilerin çalışmalarını değerlendirebilir.
- Değişen koşullara uyum sağlama fırsatları sunar.

Bu, çevik yazılım geliştirmenin temel ilkelerinden biridir.

Dersin Teması: Ölçekleme

Büyük ve daha büyük sistemlerin farklı ihtiyaçları bulunmaktadır.

Büyük bir proje 100 ila 10.000+ kişi yılı sürebilir.

Her büyük sistem, sürekli değişen birçok insan tarafından geliştirilmiştir.

- Büyük bir proje tamamlanmadan önce gereksinimler birçok kez değişmiştir.
- Hiçbir büyük sistem hiçbir zaman tamamlanmamıştır.
- Herkes projenin yalnızca bir kısmına hakimdir.

BZ313 Kapsamında Projeler

- BZ313 kapsamında gerçekleştirilecek projeler yaklaşık olarak 0.3 insan/yıldır.

Bu, bir üretim çevik sürecindeki bir veya iki sprint boyutuna denk gelir.

Ders Teması: Ekipler (Teams)

Çoğu yazılım geliştirme **bir ekip** tarafından gerçekleştirilir.

- Takımın etkinliği başarıyı belirler.

Çoğu büyük yazılım projesi **eski projeler** üzerine kuruludur. Yeni bir sistemi sıfırdan başlatmak oldukça nadir görülür.

- Başkalarının çalışmaları üzerine inşa etmek, yazılım geliştirme süreçleriyle sağlanan temel bir beceridir.
- Birçok yazılım artırımlarla (increment) oluşturulmuştur ve artımlardan farklı ekipler sorumludur.

Büyük Projelerin Yönetimi

Büyük çaplı projeler yetenekli yönetim ekibi tarafından gerçekleştirilir

- Proje yönetimi
- Personel yönetimi
- Ekonomi, hukuki süreç ve çevre faktörleri
- Geliştirme süreci

Yazılım Geliştirme Süreçleri

Temel Varsayım:

Sürecin iyi yönetilmesi ile iyi yazılımlar ortaya çıkar

Sürecin iyi yönetilmesi ile riskler azalır

Sürecin iyi yönetilmesi ile görünürlük iyileşir

Sürecin iyi yönetilmesi takım çalışmasını olanaklı kılar

Çeşitli Yazılım Süreçleri

Yazılım ürünleri oldukça çeşitlidir...

Bu nedenle, tüm yazılım geliştirme projeleri için standart bir süreç yoktur.

ANCAK başarılı yazılım geliştirme projelerinin hepsinin benzer sorunları ele alması gerekir.

Bu, tüm yazılım projelerinin bir parçası olması gereken bir dizi adım ile gerçekleştirilir.

Tüm Yazılım Geliştirme Süreçlerinde İşlem Adımları

Fizibilite ve planlama

Gereksinimler

Kullanıcı arayüzü tasarımı

Sistem tasarımı

Program geliştirme (tasarım ve kodlama)

Kabul ve yayınlama

İşletme ve bakım

Bu adımlar geliştirme döngüsü boyunca birçok kez tekrarlanabilir.

Bu adımlar arasında keskin bir ayrım yapmak ve herhangi bir anda ne yapacağınızı açıkça belirtmek önemlidir.

Not

- Yukarıdaki adımların tamamı test, güvenlik ve performans gibi süreçleri barındırır.

Fizibilite

Bir **fizibilite çalışması** henüz projeye başlamadan gerçekleştirilir.

- Önerilen projenin kapsamı nedir?
- Proje teknik olarak uygulanabilir mi?
- Öngörülen faydalar nelerdir?
- Maliyetler nelerdir, zaman çizelgesi nasıl olmalı?
- Kullanılabilir mevcut kaynak var mı?
- Riskler nelerdir ve nasıl yönetilebilir?

Bir fizibilite çalışması sonrasında projeye devam edip etmeyeceğiniz **kararını** verirsiniz.

Ders 4'te fizibilitenin ayrıntılarından söz edeceğiz.

Gereksinimler

Gereksinimler, sistemin işlevini **müşterinin bakış açısından** tanımlar.

Gereksinimler, müşteriye (client), müşterilere (customer) ve kullanıcılara (user) danışarak sistemin işlevselliğini, kısıtlamalarını ve hedeflerini belirler.

Bu süreç bazen parçalara ayrılabilir:

- Gereksinim analizi
- Gereksinim tanımı
- Gereksinim spesifikasyonu

Gereksinimler üzerinde anlaşmaya varılamaması veya bunların yeterince tanımlanamaması, yazılım projelerinin başarısız olmasının en büyük nedenlerinden biridir.

Ders 6, 7 ve 8'de Gereksinimler ayrıntılı anlatılacaktır.

Kullanıcı Arayüzü Tasarımı

Kullanılabilirlik birçok modern uygulama ve yazılım sisteminde büyük önem taşımaktadır. Kullanılabilirlik, iyi bir kullanıcı arayüzü tasarımı ile sağlanabilir**.**

Kullanıcı arayüzleri kullanıcıların geri dönüşleri ile değerlendirilebilir. Bu değerlendirme süreci iteratif olarak gerçekleştirilir:

- Kullanıcı arayüzü tasarlama
- Kullanıcılar ile test etme
- Arayüzü revize etme
- *Tekrar*

9, 10, 11 numaralı derslerde Kullanıcı Deneyimi ayrıntılandırılacaktır.

Sistem Tasarımı

Tasarım sistemin **yazılım geliştiricisi bakış açısıyla** tanımlanmasıdır**.**

Sistem tasarımı:

Hem donanım hem de yazılım olmak üzere gereksinimleri karşılayan bir sistem mimarisi oluşturmayı hedeflemektedir.

Güvenlik ve performans da sistem tasarımının önemli parçalarıdır.

12'den 15'e kadar olan dersler Sistem Tasarımı ile ilgilidir.

Model:

Modeller genellikle gereksinimleri, sistem tasarımını ve program tasarımını temsil etmek için kullanılır. Bu ders içerisinde Birleştirilmiş Modelleme Dili'nin (Unified Modeling Language , UML) temel kavramları öğretilmektedir.

Program Geliştirme

Program geliştirme:

Sistem tasarımı ve kullanıcı arayüzü tasarımı harmanlanarak gereksinimleri karşılayan programlar oluşturulur.

Program geliştirme bazen iki bölümde ele alınır:

- Program tasarımı.
- Uygulama (Implementation) (kodlama).

Genellikle bu iki parça eş zamanlı yürütülür veya birleştirilir.

Ders 16, 17 ve 18'de Program Geliştirme detaylandırılmaktadır.

Kabul ve Yayımlama (Release)

Kabul testi

Sistem, client tarafından sağlanan gereksinimlere karşı seçilmiş olan müşteri (customer) ve kullanıcılarla birlikte test edilir.

Teslimat ve yayımlama

Başarılı bir kabul testinden sonra, sistem client'a teslim edilir ve yayımlanır veya müşterilere (customer) pazarlanır.

İşletme (Operation) ve Bakım (Maintenance)

İşletim (Operation):

Sistem pratik olarak kullanıma sunulur.

Bakım (Maintenance):

Hatalar (error) ve sorunlar (problem) tanımlanır ve düzeltilir.

Gelişim (Evolution):

Sistem, gereksinimler değiştikçe, yeni işlevler eklemek veya değişen teknik ortama uyum sağlamak için zaman içinde gelişir.

Aşamalı olarak kullanımdan kaldırma (Phase out):

Sistemin hizmetten çekilmesidir.

Buna bazen **Yazılım Yaşam Döngüsü** denir.

Tüm Yazılım Geliştirme Süreçlerinde Kalite Kontrol

- **Gereksinimleri** doğrulama
- Sistem ve program **tasarımının** doğrulanması

- **Kullanılabilirlik** testi
- **Program** testi
- **Kabul** testi

- Hata düzeltme ve bakım

Bu adımlardan bazıları sistemin ömrü boyunca birçok kez tekrarlanacaktır

Test Kategorileri

"**Test**" terimi, kolayca karıştırılabilen birkaç farklı bağlamda kullanılmaktadır:

Kullanıcı testi

Kullanıcı arabiriminin sürümleri kullanıcılar tarafından sınanır. Kullanıcı deneyimleri, gereksinimlerde veya tasarımda değişikliklere yol açabilir.

Program testi

Geliştirme ekibi, hataları vb. bulmak için bileşenleri tek tek (birim testi) veya kombinasyon halinde (sistem testi) tasarıma göre test eder.

Kabul testi

Client, sistemin son sürümünü veya sistemin parçalarını gereksinimlere göre test eder.

Ders 19, 20 ve 21 güvenilirlik ve test konularını incelemektedir.

BZ313 Yazılım Mühendisliği

2. Uygulamada Yazılım Geliştirme

Ders Sonu