YAZILIM MÜHENDİSLİĞİ – Şubat 2012
Yrd.Doç.Dr. Yunus Emre SELÇUK

YAZILIM GELİŞTİRME SÜREÇ (MODEL)LERİ

1

YAZILIM GELİŞTİRME SÜREÇLERİ

- Yazılım geliştirme bir süreçtir
  - Süreç: Önceden belirlenmiş adımlardan oluşan iş akışı.
- Süreç modelleri, yazılım geliştirme sürecinin yapısını ve adımlarını belirler.
  - Önceden ve iyi planlanmış bir süreç, zamanında ve 'kaliteli' bir 'ürün' elde edilmesini sağlar.
- Çeşitli modellerin kendine özgü avantaj ve dezavantajları vardır.
  - Gerçeklenecek projeye uygun modelin seçilmesi gerekir.

2

YAZILIM GELİŞTİRME SÜREÇLERİ

ŞELALE MODELİ

- Ardışıl Model / Şelale Modeli (Sequential / Waterfall)
  - Adımlar: Analiz (Çözümleme) – Tasarım – Kodlama – Sınama – Bakım.
  - Bir adımın tamamlanmasından sonra diğerine geçilir.
  - Eksiklikler veya hatalar farkedilirse bir önceki adıma geçilir.
- Artılar:
  - En eski model, yaygın kullanımda.
  - İyi tanımlanmış adımlar.
- Eksiler:
  - Son ürünün eldesi uzun süreceğinden müşteri sabırlı olmalıdır.
  - Adımları geride bıraktıkça, ilerleyen aşamalarda karşılaşılan hataların düzeltilmesi üstel olarak zorlaşmaktadır.
  - Bir çok 'müşteri' de gereksinimleri eksiksiz ve kesin belirtmekte zorlanmaktadır.
- Sonuç: Hiç model kullanmamaktan iyidir!

3

YAZILIM GELİŞTİRME SÜREÇLERİ

ÖN ÜRÜN MODELİ

- Ön ürün modeli / Prototip modeli
  - Adımlar: Müşteriyi dinle – Ön ürün oluştur – Müşteri ön ürünü dener –

- Artılar :
  - Kullanıcı gereksinimlerinin daha iyi elde edilmesi.
  - Kullanıcının erkenden ürünü değerlendirmeye başlayabilmesi.
- Eksiler :
  - Ön ürün mükemmel değildir.
  - Eksik ürün zaman ve maliyet kısıtlamaları nedeniyle olgunlaşmadan canlı kullanıma alınabilmektedir.
- Sonuç: Prototip oluşturmayı başlı başına bir model olarak kullanmamalı, daha olgun bir modelin analiz aşamasında kullanılacak bir araç olarak ele almalı ve prototip ürünü silip atmalı.

4

YAZILIM GELİŞTİRME SÜREÇLERİ

HIZLI UYGULAMA GELİŞTİRME (RAD: Rapid Application Development)

- Kısa geliştirme çevrimleri üzerinde duran artımsal bir model.
- Gereksinimler:
  - Uygulamanın yaklaşık/ortalama 3 aylık bölümlere ayrılabilmesi,
  - Yeterli sayıda bölümün eşzamanlı ilerlemesinin sağlanabilmesi,
  - Yazılımın bileşenlerden kurulabilmesi.
- Artılar:
  - Bu sürece uygun yazılım projelerinde verimliliğin artması.
- Eksiler:
  - Büyük ölçekli çalışmalarda yeterli sayıda bölümü eşzamanlı ilerletebilecek sayıda çalışanın bulunamaması.
  - Çalışanlar hıza uyum sağlayabilmelidirler.
  - Yüksek teknik risklere uygun değil.
- Sonuç:
  - Prototip geliştirmede kullanılması veya ana fikirlerinin diğer süreçlere uygulanması yerinde olacaktır.

5

YAZILIM GELİŞTİRME SÜREÇLERİ

BİLEŞEN TABANLI (Component Based) UYGULAMA GELİŞTİRME

- Uygulamanın hazır yazılım bileşenlerinden oluşturulmasını öngörür.
- Aşamaları:
  - Konu alanı mühendisliği (Domain Engineering)
  - Aday bileşenlerin sınıflandırılması ve seçilmesi (Qualification)
  - Seçilen bileşenlerin kendi yazılımımıza uyarlanması (Adaptation)
  - Bileşenlerin bir araya getirilmesi (Composition)
- Artılar:
  - Yeniden kullanımın özendirilmesi (azalan giderler)
- Eksiler:
  - Uygun bileşenlerin bulunması gerekliliği
  - Bileşenlerin uyarlanması gerekliliği
- Sonuçlar:
  - Özellikle hızlı uygulama geliştirme olmak üzere, ana fikirleri çeşitli süreçlere uygulanabilir.

6

YAZILIM GELİŞTİRME SÜREÇLERİ

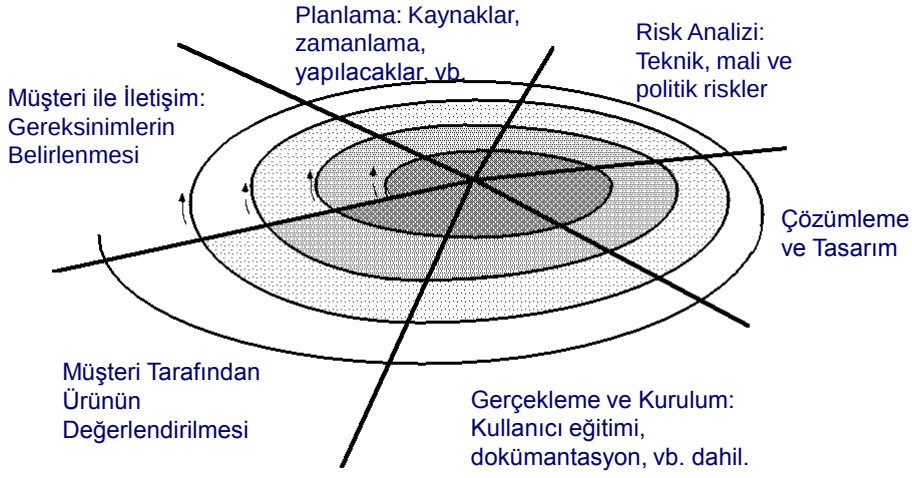
ARTIMSAL / YİNELEMELİ MODELLER

- Artımsal / Yinelemeli Modeller (Incremental / Iterative)
  - Adımlar: Analiz – Tasarım – Kodlama – Sınama – → Bakım
  - Gereksinimler önemlerine ve birbirine bağımlılıklarına göre sıralanarak her yinelemede bunların bir kısmı tamamlanır.
- Artılar :
  - Ön ürün modeli ve ardışıl modelin güçlü yönlerini kendinde toplayarak dezavantajlarını geride bırakmıştır.
  - Nesneye yönelik programlama metodolojisi ile uyum içerisindedir.
  - Eksiler : Yazılımın küçük artımlarına fazla yoğunlaşmak, sistemin geneline bakıldığında kolayca görülebilecek sorunların gözden kaçmasına neden olabilir.
- Sonuçlar:
  - Sistemin genelini göz ardı etmemek şartıyla güçlü bir modeldir.
- Örnekler: Spiral Model ve Kazan-Kazan Modeli

7

YAZILIM GELİŞTİRME SÜREÇLERİ

SARMAL (Spiral) MODEL



8

KLASİK YİNELEMELİ SÜREÇLER

Kazan-Kazan Modeli (WINWIN Model)

- Paydaş: Yazılımın başarısı ve başarısızlığının etkileyeceği kişi ve kurumlar.

9

YAZILIM GELİŞTİRME SÜREÇLERİ

ÇEVİK (Agile) SÜREÇLER

- Değişen gereksinimler, teknik riskler gibi önceden belirlenemeyen durumlara ve yazılım ürününü etkileyebilecek her tür değişikliğe karşı esneklik sağlayan süreçlerdir.

- Bireyler ve etkileşimler
- Çalışan yazılım
- Müşterinin sürece katılımı
- Değişikliklere uyum sağlamak
- Süreçler ve gereçler
- Ayrıntılı belgeler
- Sözleşme pazarlığı
- Bir planı izlemek

- Çevik süreçler, sağ taraftaki maddelerin yararını kabul etmekle birlikte, sol taraftaki maddelere daha çok önem vermektedir.
- Bir ilerleme olmaksızın yalnızca sürekli uyum sağlamak başarı değildir.
  - Yazılımın artımsal gelişimi
  - Müşteriye erken ve sık ürün teslimi
  - Başarımın birincil ölçütü çalışan yazılımdır.

10

YAZILIM GELİŞTİRME SÜREÇLERİ

ÇEVİK (Agile) SÜREÇLER

- Çevik süreci yürütecek ekibin özellikleri:
  - Yüz yüze görüşme, en etkili bilgi aktarım yoludur.
  - Takım üyeleri çevik yaklaşım hakkında eğitilmelidir.
  - Ekip üyelerinin ortak amacı, çalışan yazılım üreterek müşteriye zamanında teslim etmek olmalıdır.
  - Ekip üyeleri birbirleriyle ve müşteriyle işbirliği içinde olmalıdır.
  - Ekip üyeleri karşılıklı saygı ve güven içerisinde olmalıdır.
  - Ekipler hem teknik, hem de tüm proje hakkında kararlar verebilmelidir.
  - Boşuna harcanan çaba yoktur: Çözülen bir sorun gereksizleşse bile, çözüm sürecinde edilen deneyim ekibe ileri aşamalarda yararlı olabilir.
  - Kendi kendini düzenleme:
    - Ekibin kendisini yapılacak işe göre uyarlaması,
    - Ekibin kullanacağı süreci yerel ortama uyarlaması,
    - Üstünde çalışılan artımsal yazılım parçasını teslim etmek için gerekli çalışma zamanlamasını ekibin kendisinin belirlemesi.
- Çevik Süreç Örnekleri:
  - Aşırı Programlama (XP: Extreme Programming)
  - Sürü (Scrum)

11

YAZILIM GELİŞTİRME SÜREÇLERİ

ÇEVİK (Agile) SÜREÇLER

- Aşırı Programlama (XP)
  - Adımlar: Planlama – Tasarım – Kodlama – Sınama → Artımsal Ürün

- Planlama:
  - Müşteri, kullanıcı öyküleri oluşturur.
    - Müşteri, öyküleri önemine göre derecelendirir.
    - Yaklaşık 3 haftada gerçeklenemeyecek öyküler varsa, ekip müşteriden bunları alt öykülere bölmesini ister.
  - Ekip ve kullanıcı, öykülerin sıradaki artımsal ürüne nasıl ekleneceğine karar verir:
    - Ya önce yüksek riskli öyküler gerçeklenir,
    - Ya da önce yüksek öncelikli öyküler gerçeklenir.
    - Her olasılıkta tüm öyküler kısa sürede (birkaç hafta) gerçeklenmelidir.

12

YAZILIM GELİŞTİRME SÜREÇLERİ

ÇEVİK (Agile) SÜREÇLER

- Aşırı Programlama (XP)
  - Adımlar: Planlama – Tasarım – Kodlama – Sınama Artımsal Ürün

- Planlama (Devam):
  - İlk artımsal ürün projenin hızını ölçme amacıyla değerlendirilir:
    - Eldeki artımın hızına göre sonraki artımların teslim tarihleri belirlenir.
    - Aşırı sözler verildiği ortaya çıkarsa artımsal ürünlerin içeriği de yeniden kararlaştırılabilir.
  - Süreç ilerledikçe müşteri yeni öyküler ekleyebilir, eski öykülerin önceliğini değiştirebilir, öyküleri farklı şekillerde bölüp birleştirebilir, bazı öykülerden vazgeçebilir.
    - Bu durumda ekip kalan artımları ve iş planlarını uygun biçimde değiştirir.

13

YAZILIM GELİŞTİRME SÜREÇLERİ

ÇEVİK (Agile) SÜREÇLER

- Aşırı Programlama (XP)
  - Adımlar: Planlama – Tasarım – Kodlama – Sınama Artımsal Ürün

- Tasarım:
  - Basit tasarım karmaşık gösterimden üstündür. (KISS: Keep It Simple, Stupid!)
  - CRC (Class-Resposibility-Collaboration) kartları ile yazılımın sınıf düzeyinde incelenmesi.
  - Karmaşık bir tasarımdan kaçınılamazsa işlevsel bir ön gerçekleme yapılır (Spike solution).
  - Refactoring teşvik edilir.
  - Bu aşamanın ürünleri CRC kartları ve ön gerçeklemelerdir (başka ürün yok).

14

YAZILIM GELİŞTİRME SÜREÇLERİ

ÇEVİK (Agile) SÜREÇLER

- Örnek CRC kartı:

Sınıf adları

Sınıf: Satış	
Kasada yapılan ödemeyi simgeleyen sınıf.	
Üst Sınıf(lar): Yok	
Alt Sınıf(lar): Yok	
Sorumluluk:	**İşbirlikçi:**
Satışın yapıldığı tarih ve saati saklamak	
Yapılan ödeme tutarını saklamak	Ödeme
Satılan malların listesine erişim	Mal

15

YAZILIM GELİŞTİRME SÜREÇLERİ

ÇEVİK (Agile) SÜREÇLER

- Aşırı Programlama (XP)
  - Adımlar: Planlama – Tasarım – Kodlama – Sınama — Artımsal Ürün

- Kodlama:
  - Önce birim sınamaları hazırlanır.
    - Programcı tarafından yapılan, sınıfların (NYP'de; yapısal'da fonksiyonlar, vb.'lerin) temel işlevselliklerini sınama amaçlı kod.
    - Sadece sınavı geçmeye yarayan kod yazılır (KISS).
  - Çift kişi ile kodlama:
    - Bir programcı eldeki sorunu çözerken diğeri çözümün genel tasarıma uygunluğunu gözetir ve kodlamanın takımın karar verdiği ölçütlere (kalite, vb.) uygunluğunu denetler.

- Sınama:
  - Birim sınamalarının otomatik çalıştırılması.
  - Müşterinin artımsal ürünü denemesi.

16

YAZILIM GELİŞTİRME SÜREÇLERİ

ÇEVİK (Agile) SÜREÇLER

- Scrum:
  - Adımlar: Görev Listesi – Koşu – İşlev Gösterimi

- Görev Listesi = Kullanıcı öyküleri.
  - Önceliklendirilmiştir.
- Koşu:
  - Görev listesinin maddelerinden biri seçilir ve önceden belirlenmiş kısa bir süre içerisinde (Ör. 1-4 hafta) gerçeklenir.
  - Koşu süresince ekibin her gün yaptığı kısa (Ör. 15dk) toplantılar:
    - Proje lideri yönetir.
    - Cevaplanmaya çalışılan üç ana soru:
      - Son toplantıdan bu yana ne yaptınız?
      - Karşılaştığınız engeller nelerdir?
      - Yarınki toplantıda neleri başarmayı hedefliyorsunuz?
- İşlev Gösterimi: Müşterinin en yeni işlevi veya o ana dek gerçeklenen tüm işlevleri sınaması.

17

YAZILIM GELİŞTİRME SÜREÇLERİ

ÇEVİK (Agile) SÜREÇLER

- Çevik Modelleme
  - Bir amaç için modelleme yapın:
    - Neyi, kime, hangi düzeyde anlatmak istiyorsunuz?
    - Buna göre uygun modelin ve ayrıntılandırmanın seçimi .
  - İçerik sunumdan daha önemlidir.
    - Gerekli bilgiyi içermeyen hatasız model işe yaramaz!
  - Kullandığınız modelleme yolunun özünü ve modellerinizi oluşturmak için kullanacağınız araçları iyi öğrenin.
- DİKKAT: Önemli olan dengeyi korumaktır.
  - Çevik çalışacağız diye serseri programcı olmayın.
  - Disiplinli çalışacağız diye sırtınızda tuğla çuvalı taşımayın.

18

YAZILIM GELİŞTİRME SÜREÇLERİ

SÜREÇ SERTİFİKASYONU

- Olgunlaşmış bir yazılım geliştirme sürecine sahip olmayan bir yazılım firması, projelerini başarı ile sonuçlandıramaz.
- Bir yazılım firması, süreçlerinin yeterliliğini bağımsız kurumlara onaylatmayı seçebilir.
- Gerekli olduğu durumlar:
  - Bazı büyük müşteriler sertifikalı yazılım firmaları ile çalışmayı şart koşarlar.
- Gereksiz olduğu durumlar:
  - Çok küçük şirketler ve/veya projeler için ek yük olarak görülebilir.
- Güncel standartlar:
  - CMMI: Capability Maturity Model Integration
    - SEI tarafından önerilmiştir (Software Engineering Institute of Carnegie-Mellon University)
  - PMI: Genel amaçlı bir proje yönetimi yaklaşımı
  - ISO 9001:2000 standartları (Genel)
    - ISO/IEC 90003:2004 (Yazılım geliştirmeye özel)
  - Genel vs. Özel (Peynir mi üretiyoruz?)

19

YAZILIM GELİŞTİRME SÜREÇLERİ

CMMI DÜZEYLERİ

- CMMI düzeyleri:
  - 1. Düzey: Giriş düzeyi (Level 1: Initial). İş şansa ve anahtar kişilere kalmış.
  - 2. Düzey: Yinelenebilir (Repeatable). Temel planlama ve izleme yöntemleri kullanılarak, önceki projelerdeki başarılar yeni projelerde tekrarlanılabilir.
  - 3. Düzey: Tanımlanmış (Defined). Kişi ve risk yönetimi ile projenin yönetimi iyileştirilir.
    - Büyük müşteriler en azından bu düzeyde yazılım evleri ile çalışmak ister.
  - 4. Düzey: Yönetilen (Managed). Süreç ve yazılım ölçütleri kullanılarak kalite yönetimine geçilir. İlerleme sürekli izlenir, bütçe ve zaman hedeflerinden sapmalar erkenden belirlenerek gerekli önlemler alınır.
  - 5. Düzey: İyileştirilmiş (Optimized). Süreç yönetimi geçmiş deneyimlerin ışığında sürekli iyileştirilir.
- 700'den fazla sayfaya sahip dokümanı için: http://www.sei.cmu.edu/cmmi/

20