Software Engineering Final Notları

~ $Yunus\ Emre\ AK$ ©

Döküman Renklendirme Yapısı

PDF Başlığı

Ana Başlıklar

Alt Başlıklar

İç Başlıklar

En İç Başlıklar

Tablo	Başlığı

Bağlantılar

`Değişmez ifadeler`

$Formüller$

Önemli notlar

Terimsel ifadeler

> Yorum satırları

İçerikler

- Temel Terimler
- Softawre Process Involves
- Incremental Development
- Reuse Oriented Development
- Testin Sateges
- System Evolution Steps
- Prototypes
- Boehms Spiral Model
- The Rational Unified Process (RUP)
- Agile Development (Çevik Geliştirme)
    - Extreme Programming (XP)
- Gereksinimlerin Açıklamaları
    - Fonksiyonel Olmayan (Non-Functional) Gereksinimler
- Yazılım Testi
- Test Seviyeleri
- Gereksinim Kontrolleri
- Verification vs Validation
    - Onaylama & Doğrulama Sisteminin Amaçları (V & V Confidence)
    - Static Verification
- Güvenlik Kavramları
    - Güvenilirlik Özellikleri (Dependability Properties)
- Gelişim ve Servis Modeli (Evolution & Service)
    - Gelişim ve Geliştirme (Evolution and Development) Modeli
    - Yazılımın Değiştirilme Sebepleri
    - Yazılım Bakımları
- Yeniden Yapılandırma (Re-Engineering) Süreci
- Yazılım Projelerinin Yönetimi (Software Project Management)
    - Yönetim Faaliyetleri
    - Risk Yönetimi
    - Yazılım Projelerindeki Riskler
- Harici Bağlantılar

Temel Terimler

Terim	Açıklama
Software	programs + related documentation
Software process	structured set of actiivitie to develop software
Software process model	Abstract representation process

Softawre Process Involves

- Specification
- Development
- Validation
- Evolution

Incremental Development

- Eylemler iç içe geçmiş şekilde ilerler (Activities are interleaved)
- Geliştirme süreci bir çok başlangıç versiyonu ile ilerler (Development proceeds with many intermadiate versions)
- Değişiklik gereksinimlerini karşılamak daha kolaydır (Changing requirements are easier to accomodate)
- Daha hızlı geliştirme imkanı vardır (More rapid delivery is possible)
- Müşteri bilgilerini almak sorunlu değildir (Getting customer feedback is not problematic)

Reuse Oriented Development

- Bileşenlerin analizi (Component analysis)
- Modifikasyon gereksinimleri (Requirement modification)
- Sistemin tasarlanması ve yeniden kullanılması (System design with reuse)
- Geliştirme ve entegrasyon (Development & integration)

Testin Sateges

- Bileşen testi (Component testing)
- Arayüz testi (Interface testing)
- Sistem testi (System testing)

System Evolution Steps

- Yeni gereksinimleri tanımlama (Define new requirement)
- Mevcut sistemi değerlendirme (Assets existing systems)
    - Sistemi doğrulamaz (validation)
- Değişikilk önerileri (Purpose changes)
- Sistemi değiştirme (Modify the system)

Prototypes

Prototiplerin atılma sebepleri:

- Dökümanlaştırılmamış (Undocumented)
- Ayarlaması çok zor (Very difficult to tune)
- Hızlı bozulan yapı (Quickly degrading structure)

> Test edilmesinin zor olması atılma sebeplerinden biri değildir.

Boehms Spiral Model

- İşlem yapısı spiral olarak gösterilir (Process represented as a spiral)
- Her bir döngü bir aşamayı temsil eder (Each loop represent a phase)
- Aşama sayısı sınırlı değildir (Not limited phases)
- Riskler açıkça ele alınır ve çözülür (Risk are explicitly assesed and resolved)

The Rational Unified Process (RUP)

Organizasyon içindeki sorumlulukların detaylı olarak benimsenmesini ve disiplinli yazılım gel iştirilmesini amaçlar.

Birleşik Rasyonal İşlem yapsının **bakış açısı**:

- Dinamik (Dynamic)
- Statik (Static)
- Uygulama (Practive)

> Davranışsal (behavioral) bakış açısına sahip değildir.

RUP için güzel örnekler:

- Yinelemeli olarak yazılım geliştirme (Develop sofware iteratively)
- Gereksinimleri yönetme (Manage requirements)
- Bileşen tabanlı mimarilerin kullanımı (Use component-based architectures)
- Görsel model yazılımı (Visually model software)

> Yazılımı indirgemek gibi bir amacı yoktur. (Reduce software)

Agile Development (Çevik Geliştirme)

İnsana ve sürece odaklı modeldir.

- En başarılı bulunan sistemlerden biridir
- Süreç anında tartışılarak fikir üretilir
- Scrum & Kanban gibi framwork (metodolojileri) kullanır
    - ToDo - In progress - ... yapısı
- Başarı oranını %80'lere çıkarabilmektedir

Neleri ön planda tutar:

- Bireyler ve etkilişime > Süreçler ve açlar (Rapid development over plans and tools)
- Çalışan yazılım > Kapsamlı dökümantasyon (Working software over detailed documentation)
- Müşteri ile iş birliği > Sözleşme (Customer collaboration over contract negotiation)
- Değişime karşılık verme > Plana bağlı kalmak (Responding to change over following a plan)

Extreme Programming (XP)

- Bir günde birden fazla versiyon çıkabilir
- Tüm sürümleri test edip yayınlamak mümkündür
- Kod düzenleme (refactoring)
    - Tekrarları fonksiyon haline getirme
    - Güzel adlandırma
    - Önce yazma sonra düzenleme

Gereksinimlerin Açıklamaları

Gereksinim (requirement)	Açıklama
Kullanıcı	Doğal dille müşteri ve kullanıcılar için yazılır
Sistem	Detaylı gereksinimler, teknik ekip ve sözleşme için detaylı UML
Fonksiyonal	Kullanıcıdan istediği fonksiyonların alınması
Fonksiyonal Olmayan	Yazılımdaki özellikler veya koşullar (zaman, güvenlik vs)

Fonksiyonel Olmayan (Non-Functional) Gereksinimler

- Hız (Speed)
- Boyut (Size)
- Kullanılabilirlik (Usability)
- Güvenilirlik (Reliability)

> Test edilebilirlik bu gereksinimlerden biri değildir.

Yazılım Testi

Program testleri:

- Validation Testing (Doğrulama)
- Defect Testing (Kusur)

Test etme işleminin yapılma sebepleri:

- Hataları bulmak ya da göstermek (Reveal or find errors)
- Programın istenilenleri karşılaması (Meets customer requierements)
- Doğru çalışması (Working correctly)

Test Seviyeleri

Test Seviyesi	Açıklama
Alpha	Geliştirici ekip ile yazılımın testi
Beta	Kullanıcı tarafından test edilip, geliştiricilere bildiriliyor

Gereksinim Kontrolleri

- Geçerlilik (Validity)
- Tutarlılık (Consistency)
- Tamlık (Completeness)
- Doğrulanabilirlik (Verifiability)

> Doğruluk (accuarcy) kontrollerden biri değildir, doğrulanabilirlik kontrol edilir.

Verification vs Validation

Terim	Açıklama
Verification (Onaylama)	İşi düzgün yapma
Validation (Doğrulama)	Doğru işi yapma

Onaylama & Doğrulama Sisteminin Amaçları (V & V Confidence)

- Software purpose (Yazılımın amacı)
- User expectations (kullanıcı beklentileri)
- Marketing environment (Pazarlama ortamı)

Static Verification

- Kaynak kodunu inceleme
- Derleme gerektirmek
- Hata bulmak için etkili teknikler

> Fonksiyonal olmayan (Non-Functional) özellikleri test etmez.

Güvenlik Kavramları

Kavram	Açıklama
Safety (Güvenlik)	Yazılımın kullanıcıya, ortama veya verilere zarar vermemesi
Security (Koruma)	Yazılıma karşı yetkisiz işlemlerin engellenmesi

Güvenilirlik Özellikleri (Dependability Properties)

- Düzeltilebilirlik (Repairability)
- Yeni gereksinimlere uyarlanabilir (Maintainability)
- Saldırı karşısında hizmete devam edebilme (Survivability)
- Hataları önleyebilme (Error tolerance)
- Kurtarılabilirlik (Recoverability) güvenlik prensiplerinden değildir.

Gelişim ve Servis Modeli (Evolution & Service)

Model	Açıklama
Gelişim (Evolution)	Fonksiyonellik arttırılır, hatalar çözülür, yeni ortama hazırlanır
Sunma (Servicing)	Sadece hata çözümü olur, fonksiyonallik arttırılmaz
Son aşama (Phase-out)	Fonksiyonellik arttırımı, hata düzeltmesi ve bakım olmaz

Gelişim ve Geliştirme (Evolution and Development) Modeli

- Belirleme (Specification)
- Tanımlama (Implemention)
- Onaylama (Validation)
- Gerçekleştirme (Operation)

Yazılımın Değiştirilme Sebepleri

- Yeni gereksinimler (New requirements)
- Hatalar (Errors)
- Yeni ortamlar (New platforms)
- Performans zaafiyeti (Poor performances)

> Döküman için yazılım değiştirilmez, yazılım için döküman değiştirilir.

Yazılım Bakımları

- Yazılım hatalarını onarma (Maintenance software faults)
- Yazılımı yeni uygulamaya uyarlamak için bakım (Maintenance to adapt software to a new platform)
- Sistem fonksiyonalliğini geliştirme veya ona bakım (Maintenance to add or modify the system's functionality)

> Kod güzelleştirme ve güvenilirliği arttırma (Maintenance to refactor code and improve dependability) gibi sebeplerle yazılım bakımı yapılmaz.

Yeniden Yapılandırma (Re-Engineering) Süreci

Var olan bir programı veya yazılımı tekrardan ele alma ve geliştirme sürecidir. Temel hedefleri:

- Tersine mühendislik (Reverse Engineering) ile yeniden düzenleme
- Program yapısını geliştirme (Program structure improvement)
- Modüler yapı oluşturma (Program modularisation)
- Verilerin yeniden yapılandırılması (Data reengineering)

> Doğrulama (validation) gibi işlemlerle ilgilenmez, temel amaç geliştirmektir.

Yazılım Projelerinin Yönetimi (Software Project Management)

Yazılımın yönetiminin odaklandığı kavramlar:

- Zamanında bitirme (On time)
- Sahip olunan maaliyetle tammalanması (On bugdet)
- Müşteri isteklerini karşılaması (Meeting customer expectation)

Yönetim Faaliyetleri

- Proje planlaması (Project planning)
- Raporlama (Reporting)
- Risk analizi (Risk management)
- İnsan yönetimi (People management)

> Değişim yönetimi (Change management) ile ilgilenmez.

Risk Yönetimi

- Risk tanımlaması (Risk identification)
- Risk analizi (Risk analysis)
- Risk planlama (Risk planning)
- Risk takibi (Risk monitoring)

> Risk azaltma (risk reduciton) durumuna odaklanmaz, var olan risklere çözüm bulunur.

Yazılım Projelerindeki Riskler

- İşe alım sorunları (Recruitment problems)
- Müşteri ihtiyaçlarının değişmesi (Changing customer needs)
- Personal hastalığı (Staff illness)
- Arızalı bileşenler (Defective componenets)

> Personel miktarı (Staff quantity) bir risk değildir, risk sorun ihtimallerini kapsar.

Harici Bağlantılar

- Agile nedir
- Yazılım Mimarisinin Kalite Gereksinimleri - Yazılım Güvenilirliği