ANKARA ÜNİVERSİTESİ ENFORMATİK BÖLÜMÜ
TEZSİZ YÜKSEK LİSANS PROGRAMI

Yazılım Mühendisliği

Doç. Dr. Recep ERYİĞİT

HEDEFLER

- ✓Yazılım, program ve algoritma kavramları anlar.
- ✓Yazılım ve donanım maliyetlerinin zamansal değişimlerini ve nedenleri hakkında yorum yapar.
- ✓Yazılım mühendisliği ile Bilgisayar mühendisliği kavramlarını karşılaştırabilir.
- ✓Yazılım yaşam döngüsünün çekirdek süreçlerini bilir.
- ✓Yazılım geliştirme sürecindeki maliyet ve hataları süreçlere dağıtabilir.

YAZILIM

Yazılım, elektronik aygıtların belirli bir işi yapmasını sağlayan programların tümüne verilen isimdir. Bir başka deyişle, var olan bir problemi çözmek amacıyla bilgisayar (programlama) dili kullanılarak oluşturulmuş anlamlı anlatımlar bütünüdür.

Yazılım için çeşitli programlama dilleri mevcuttur. Tarihsel gelişimlerine göre;

- Makine dili (0, 1)
- Assembly (put, move, save)
- Fortran (goto, if, do)
- C (printf, scanf)
- Java (metod, class, package)

Şeklinde özetlenebilirse de günümüzde yüzden fazla programlama dili olduğu söylenebilir. Bu kadar çok programlama dili olmasının nedeni, ilgilenilen problemin içerdiği kavramların problemin çözümünde kullanılacak programlama dilinde kolay tanımlanabilmesinin istenmesidir.

Programlama Dili

- ✓Programlama dili, yazılımcının bir algoritmayı ifade etmek amacıyla, bir bilgisayara ne yapmasını istediğini anlatmasının tektipleştirilmiş yoludur.

- ✓Programlama dilleri, yazılımcının bilgisayara hangi veri üzerinde işlem yapacağını, verinin nasıl depolanıp iletileceğini, hangi koşullarda hangi işlemlerin yapılacağını tam olarak anlatmasını sağlar.

Algoritma

Algoritma, belli bir problemi çözmek veya belirli bir amaca ulaşmak için tasarlanan yol. Matematikte ve bilgisayar biliminde bir işi yapmak için tanımlanan, bir başlangıç durumundan başladığında, açıkça belirlenmiş bir son durumunda sonlanan, sonlu işlemler kümesidir.

Örneğin klavyeden girilen iki sayının toplamını bulan ve sonucu ekrana yazdıran programın algoritması ve akış diyagramı,

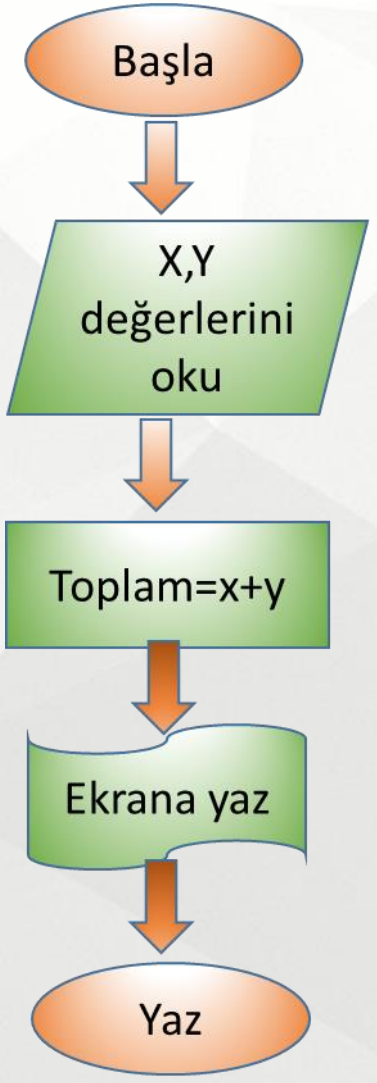
Değişkenler: x, y, toplam

Algoritma --> Gösterim(Akış diyagramı)

Algoritma

1. Başla
2. x değişkeninin değerini oku.
3. y değişkeninin değerini oku.
4. x ve y sayılarını topla sonucu toplam değişkenine aktar.
5. Toplam değerini ekrana yazdır.
6. Dur

Akış diyagramı



Yazılım Nedir ?

Yazılım, elektronik aygıtların belirli bir işi yapmasını sağlayan programların tümüne verilen isimdir. Bu tanımı içerdiği bileşenler cinsinden,

Yazılım= programlama dili + algoritma+ belge+ insan +....

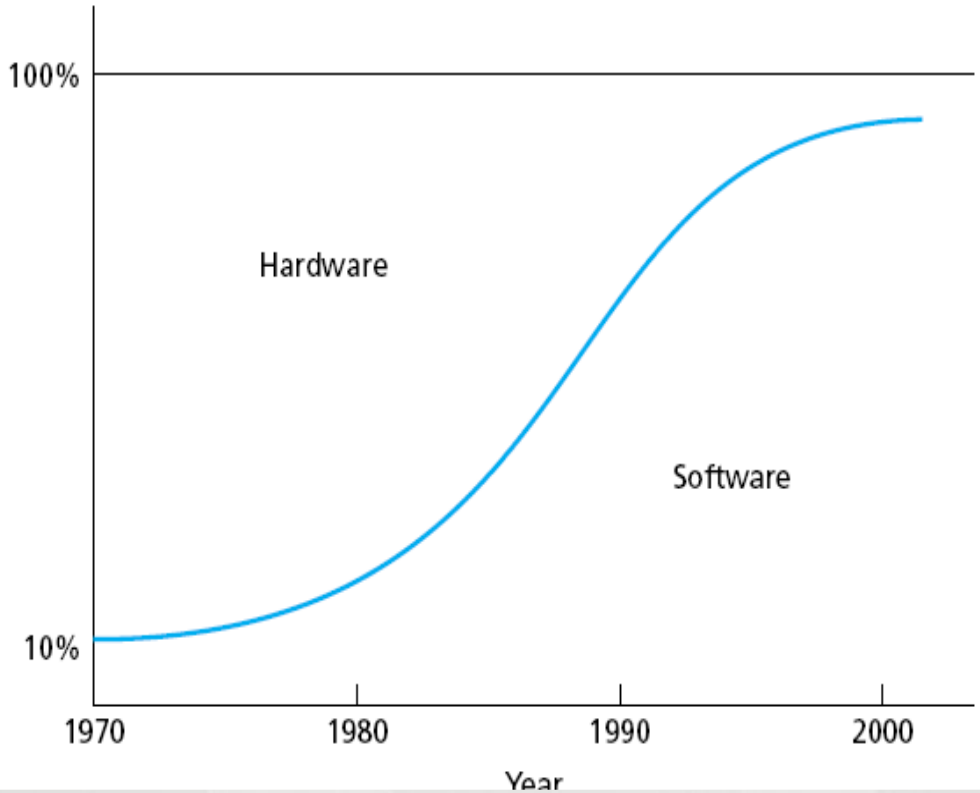
yazılabilir.

Yazılım Donanım Karşılaştırılması

Donanım, programlanabilir elektronik birim. Donanım bir kez geliştirildikten fabrikalarda çoğaltılırlar.

Yazılım her bir müşteri problemi için geliştirilmek zorundadır. (Paket Programlar nereye konacak!!!!!!!!)

Donanım – Yazılım fiyatlarının değişimi yan tarafta verilmiştir. Günümüzde yazılım geliştirme fiyatları toplam sistem fiyatının %95'lik bölümünü oluşturmaktadır.



Yazılım Türleri

Bilgisayar yazılımları genel olarak 2 ana grupta incelenebilir.

1- Sistem Yazılımları:

Bilgisayarın kendisinin işletilmesini sağlayan, işletim sistemi, derleyiciler (yazılan programı makine diline çeviren program) ve çeşitli yardımcı yazılımlardır.

Bütün sistem programları içinde en temel yazılım işletim sistemidir ki, bilgisayarın bütün donanım ve yazılım kaynaklarını kontrol ettiği gibi, kullanıcılara ait uygulama yazılımlarının da çalıştırılmalarını ve denetlenmelerini sağlar.

2- Uygulama Yazılımları:

Kullanıcıların işlerine çözüm sağlayan örneğin çek, senet, stok kontrol, bordro, kütüphane kayıtlarını tutan programlar, bankalardaki müşterilerin para hesaplarını tutan programlar vs. gibi yazılımlardır.

- ✓ Bazen bunlara ek olarak uygulama yazılımı geliştirmede kolaylık sağlamak amacı ile geliştirilmiş kütüphane yazılımları ek olarak verilebilmektedir.

Yazılım Mühendisliği Nedir?

Yazılım Mühendisliği; sistemli ve ölçülebilir bir yaklaşımın yazılımın geliştirilmesinde, işletilmesinde ve bakımında uygulanmasıdır. Özetle, mühendisliğin yazılıma uygulanmasıdır. Yazılım geliştirmek, dışarıdan bakıldığında "bilene kolay" gibi görünse de karmaşık yazılımların geliştirilmesi ve var olan sistemlere entegrasyonu mühendislik eğitimini gerekli kılmıştır.

Mühendislik, herhangi bir bilim alanındaki teoriyi sistematik olarak pratiğe geçirmeyi hedefler ve bilim ile matematiği kullanır. Yazılım da uygulanan mühendislik: Yapılacak işi planlamayla başlar ve geliştirilen sistemi kullanma esnasındaki bakıma kadar uzanan tüm etkinlikleri kapsar. Sadece teknik etkinlikle de değil, yönetim etkinliklerini de içerir.

Bilgisayar Mühendisliği – Yazılım Mühendisliği

Bilgisayar Mühendisliği

Algoritmalar

Hesaplama Kuramları

Derleyiciler

İşletim sistemleri

Yazılım Geliştirme

Yazılım Mühendisliği

Yazılım mimarisi

Proje Yönetimi

Teknik Planlama

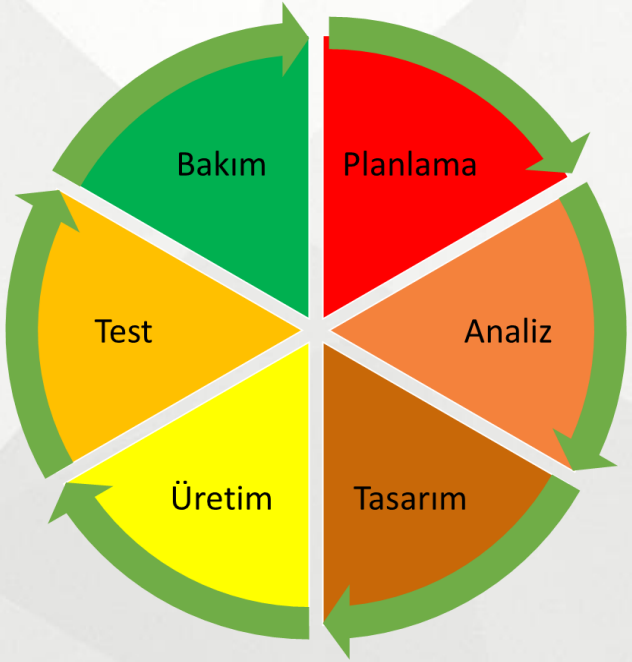
Risk Yönetimi

Yazılım Kalitesi ve Güvenliği

Yazılım Yaşam Döngüsü

Yazılım Mühendisliği, bilgisayar mühendisliğinin ilgi alanlarından biri olan yazılım geliştirme alanı üzerinde gelişmektedir. Yazılım Mühendisliğinde yazılım doğrusal değil döngüsel bir süreç olarak kabul edilir ve yazılım yaşam döngüsü kavramıyla ifade edilir.

Yazılım Yaşam döngüsü çoğunlukla 5 veya 6 çekirdek süreçten oluşturulur. Yazılım karmaşıklığı arttıkça üretim ile test süreçlerini birbirinden ayırma eğilimi artmaktadır.



Çekirdek Süreçler

Planlama: Temel ihtiyaçlar belirlenir, proje için **fizibilite** çalışmaları yapılır (maliyetlerin ve sistemin yararlarının tanımlanması) ve proje planlaması gerçekleştirilir.

Analiz: Sistemin işlevlerini ve kesin gereksinimleri açıklığa kavuşturarak dokümante etmektir. Bu çalışma müşteri, yazılım mühendisi, sistem analisti, iş analisti, ürün yöneticisi vb. rollerin bir araya geldiği gruplar tarafından yapılabilir. İhtiyaçların net olmadığı durumlarda yazılım mühendisi ve müşteri arasında iletişim ve birlikte çalışmanın çok daha fazla olması gerekir. Çeşitli yazılım geliştirme metodolojilerinde bu aşamada kullanım dokümanları ve test plan dokümanları da oluşturulabilir.

Çekirdek Süreçler

Tasarım: Yazılım ürün tasarımı, müşterinin gereksinim ve isteklerini karşılamak üzere yazılım ürününün özellikleri, yetenekleri, ve arayüzlerinin belirlenmesi etkinliğidir. İki tür tasarımdan bahsetmek mümkündür (Yüksek düzeyde tasarım – Mimari tasarım ve Detay tasarım). Mimari tasarım, yazılım modüllerinin genel yapıları ve organizasyon içerisindeki etkileşimleri ile ilgilenir. Detay tasarım aşamasında Mimari tasarım dokümanları genelde revize edilirler. Tasarım ve analiz aşamalarının ayrımı "Problem **Ne**?/Problem **Nasıl** Çözülür?" sorularının kullanımı ile ilgilidir. Gereksinimlerin belirlendiği analiz aşaması problemin ne olduğu ile ilgilidir.

Çekirdek Süreçler

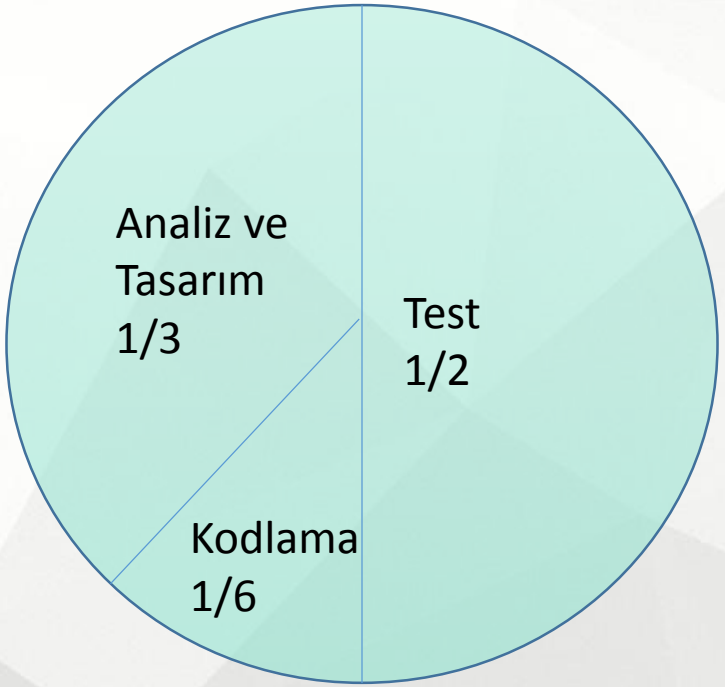
Gerçekleştirim (Kodlama ve Test) Tasarım aşamasının belirli bir olgunluğa ulaşmasıyla birlikte **Kodlama** aşaması başlar. Müşteriye teslim edilecek ürünü programlama aşamasıdır.

Bakım Teslim ile birlikte bakım aşaması da başlar. Hata giderici, önleyici, altyapıyı iyileştirici, ürüne yeni özellikler ekletici gibi farklı bakım faaliyetleri mevcuttur.

Yazılım Geliştirme Süreçlerinin Maliyet Oranları

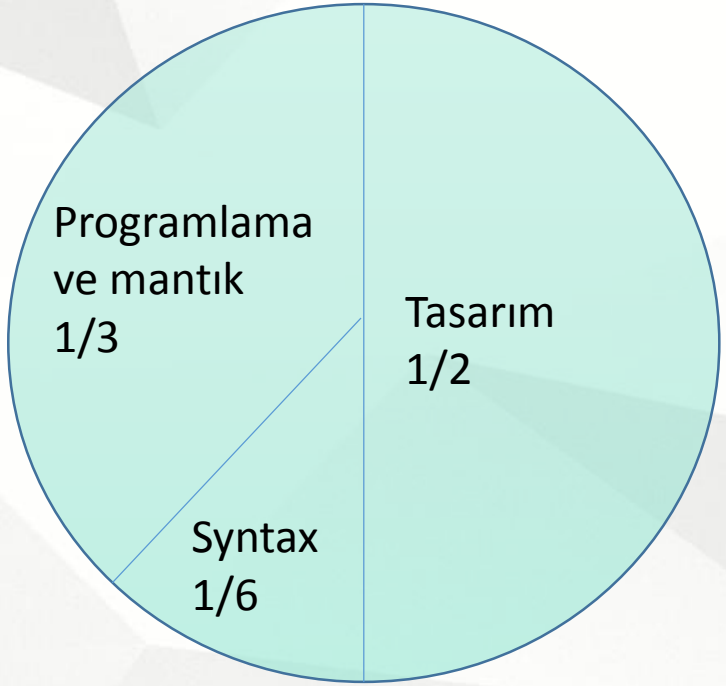
Yazılımın yaşam döngüsünün beş yıllık toplam maliyetinin %70'e yakınını bakım süreci oluşturmaktadır.

Yazılım geliştirme sürecine ait alt süreçlerin maliyet oranları yan tarafta verilmiştir.



Yazılım Geliştirmede Hatalaların Süreçlere Dağılımı

✓Geliştirmede yapılan hataların süreçlere dağılımı



✓Hata düzeltme maliyet oranları

Sorular

1. Yazılım ile programlama kavramlarını tanımlayınız.
2. Algoritma nedir? Bir algoritmanın gerçeklemesinde kullanılacak dilin seçimi neden önemlidir?
3. Yazılım Mühendisi ile bilgisayar mühendislerinin temel çalışma alanlarını yazınız.
4. Büyük çaplı bir yazılım projesinin geliştirilmesinde yalnızca yazılım mühendislerinin bulunması yeterli olur mu?
5. Yazılı yaşam döngüsünün çekirdek süreçleri nelerdir?
6. Analiz ve tasarım süreçleri arasındaki fark nedir?
7. Yazılım + donanım sistemi ediniminin yıllara göre maliyet değişimi nasıldır? Grafiği açıklayınız.
8. Analiz ve tasarım süreçleri yazılım geliştirmenin ne kadarlık zamanında yapılır.

Önümüzdeki Hafta

Süreç modellleri

1. Barok Model
2. Şelale modeli
3. V-süreç modeli
4. Helezonik Model
5. Artımsal Geliştirme modeli
6. Artımsal geliştirme modeli

Metodoloji Nedir?

Metodolojiler ile süreç modellerinin karşılaştırılması

Yazılım Mühendisliği
Süreç Modelleri

KONULAR

- ✓ Yazılım Yaşam Döngüsü (Çekirdek Süreçler)
- ✓ Belirtim Yöntemleri –Süreç Modelleri
- ✓ Yazılım Süreç Modelleri
- ✓ Gelişigüzel Model
- ✓ Barok Model
- ✓ Çağlayan Modeli
- ✓ V Modeli
- ✓ Spiral Model
- ✓ Evrimsel Geliştirme Süreç Modeli
- ✓ Artırımsal Geliştirme Süreç Modeli
- ✓ Araştırma Tabanlı Süreç Modeli
- ✓ Metodolojiler

Yazılım Yaşam Döngüsü

1. Planlama

   Personel ve donanım gereksinimlerinin çıkarıldığı, fizibilite çalışmasının yapıldığı ve proje planının oluşturulduğu aşamadır.

2. Analiz

   Sistem gereksinimlerinin ve işlevlerinin ayrıntılı olarak çıkarıldığı aşama. Var olan işler incelenir, temel sorunlar ortaya çıkarılır.

   mantıksal; önerilen sistemin yapısı anlatılır,

3. Tasarım

   Belirlenen gereksinimlere yanıt verecek yazılım sisteminin temel yapısının oluşturulduğu aşamadır.

   mantıksal; önerilen sistemin yapısı anlatılır,

   fiziksel; yazılımı içeren bileşenler ve bunların ayrıntıları.

4. Gerçekleştirim

   Kodlama, test etme ve kurulum çalışmalarının yapıldığı aşamadır.

5. Bakım

   Hata giderme ve yeni eklentiler yapma aşaması.

Belirtim Yöntemleri

Bir çekirdek sürece ilişkin fonksiyonları yerine getirmek veya çekirdek süreçler arası geçişlerin belirtilmesinde kullanılan yöntemler, Belirtim Yöntemleri olarak adlandırılır.

Süreç Akışı İçin Kullanılan Belirtim Yöntemleri

Süreçler arası ilişkilerin ve iletişimin gösterildiği yöntemler (Veri Akış Şemaları, Yapısal Şemalar, Nesne/Sınıf Şemaları).

Süreç Tanımlama Yöntemleri

Süreçlerin iç işleyişini göstermek için kullanılan yöntemler (Düz Metin, Algoritma, Karar Tabloları, Karar Ağaçları, Anlatım Dili).

Veri Tanımlama Yöntemleri

Süreçler tarafından kullanılan verilerin tanımlanması için kullanılan yöntemler (Nesne İlişki Modeli, Veri Tabanı Tabloları, Veri Sözlüğü).

Yazılım Süreç Modelleri

- ✓ Yazılım yaşam döngüsündeki çekirdek süreçlerin geliştirme aşamasında hangi sırada uygulanacağını tanımlayan modellere Süreç Modelleri denir.
- ✓ Yazılım üretim işinin yapılma düzenine ilişkin rehberlerdir.

Süreçlere ilişkin ayrıntılarla ya da süreçler arası ilişkilerle ilgilenmezler.

1. Gelişigüzel Model
2. Barok Modeli
3. Çağlayan (Şelale) Modeli
4. V Modeli
5. Spiral Model
6. Evrimsel Model
7. Artırımsal Model
8. Araştırma Tabanlı Model

Gelişigüzel Model

- ✓Herhangi bir model ya da yöntem yok.
- ✓Geliştiren kişiye bağlı.
- ✓İzlenebilirliği ve bakımı oldukça zor.
- ✓Genellikle tek kişilik üretim ortamı.
- ✓Karmaşık olmayan yazılım sistemleri.

Barok Model

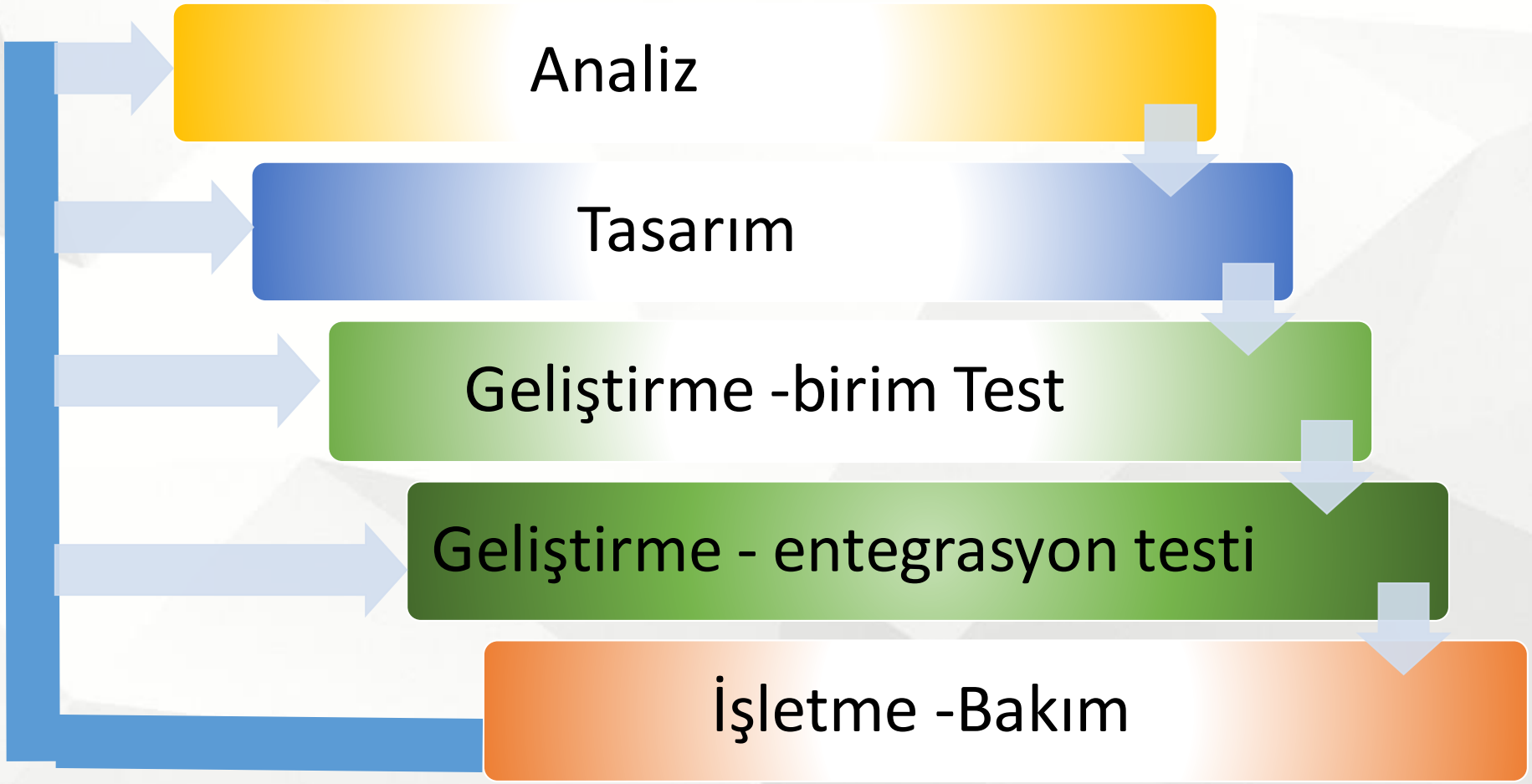
- ✓ Analiz
- ✓ Tasarım
- ✓ Kodlama
- ✓ Modül Testleri
- ✓ Altsistem Testleri
- ✓ Sistem Testi
- ✓ Belgeleme
- ✓ Kurulum

Yaşam döngüsü temel adımlarının doğrusal bir şekilde geliştirildiği model.

Belgelemeyi ayrı bir süreç olarak ele alır, ve yazılımın geliştirilmesi ve testinden sonra yapılmasının öngörür.

Aşamalar arası geri dönüşlerin nasıl yapılacağı tanımlı değil.

Çağlayan Modeli



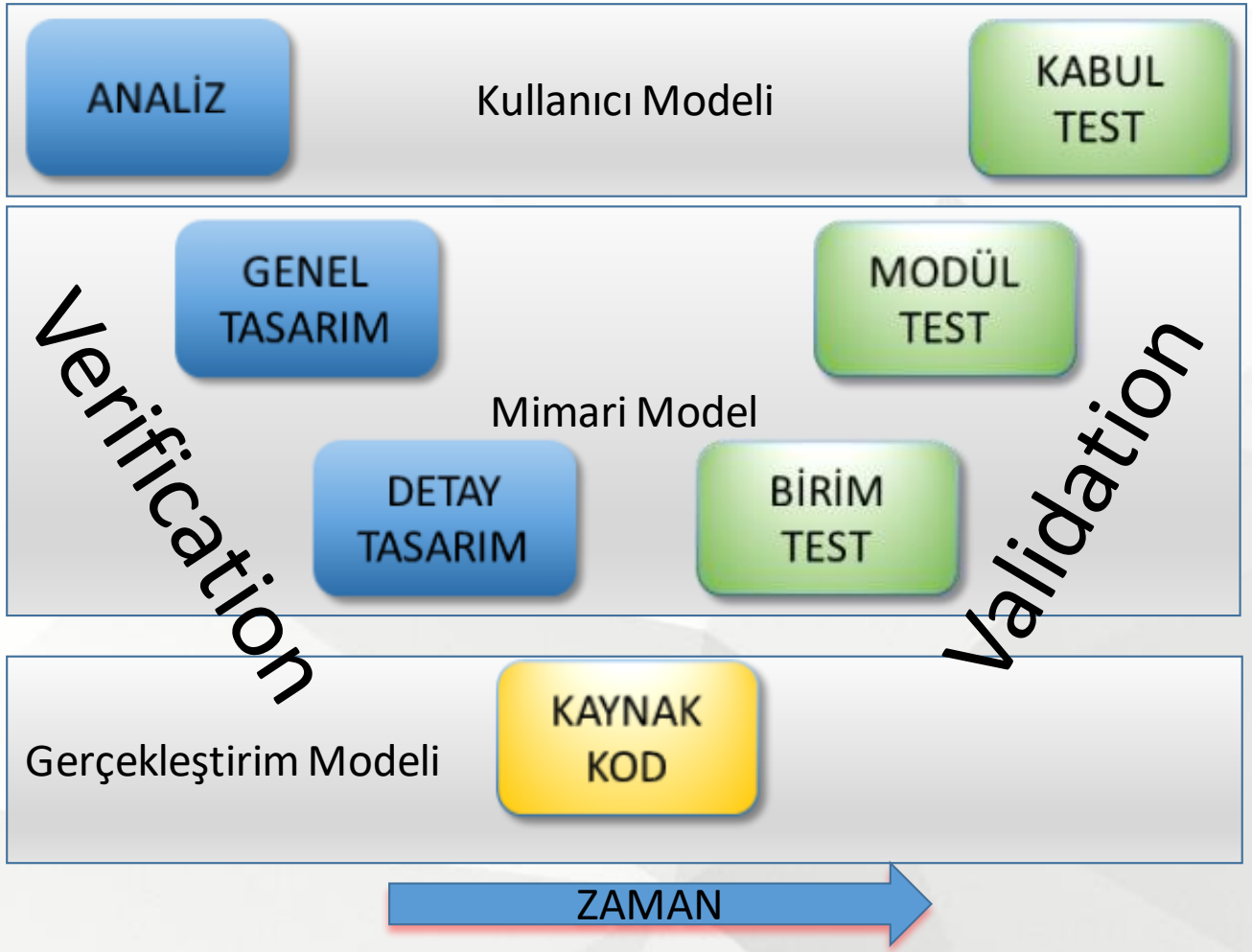
Çağlayan Modeli

- ✓Yaşam döngüsü temel adımları baştan sona en az bir kez izleyerek gerçekleştirilir.
- ✓İyi tanımlı projeler ve üretimi az zaman gerektiren yazılım projeleri için uygun bir modeldir.
- ✓Geleneksel model olarak da bilinen bu modelin kullanımı günümüzde giderek azalmaktadır.
- ✓Barok modelin aksine belgeleme işlevini ayrı bir aşama olarak ele almaz ve üretimin doğal bir parçası olarak görür.
- ✓Barok modele göre geri dönüşler iyi tanımlanmıştır.
- ✓Yazılım tanımlamada belirsizlik yok (ya da az) ise ve yazılım üretimi çok zaman almayacak ise uygun bir süreç modelidir.

Çağlayan Modeli

- ✓Gerçek yaşamdaki projeler genelde yineleme gerektirir.
- ✓Genelde yazılımın kullanıcıya ulaşma zamanı uzundur.
- ✓Gereksinim tanımlamaları çoğu kez net bir şekilde yapılamadığından dolayı, yanlışların düzeltilme ve eksiklerin giderilme maliyetleri yüksektir.
- ✓Yazılım üretim ekipleri bir an önce program yazma, çalıştırma ve sonucu görme eğiliminde olduklarından, bu model ile yapılan üretimlerde ekip mutsuzlaşmakta ve kod yazma dışında kalan (ve iş yükünün %80'ini içeren) kesime önem vermemektedirler.
- ✓Yöneticilerin ürünü görme süresinin uzun oluşu, projenin bitmeyeceği ve sürekli gider merkezi haline geldiği düşüncesini yaygınlaştırmaktadır

V Süreç Modeli



V Süreç Modeli

Sol taraf üretim, sağ taraf sınama işlemleridir.

V süreç modelinin temel çıktıları;

Kullanıcı Modeli

Geliştirme sürecinin kullanıcı ile olan ilişkileri tanımlanmakta ve sistemin nasıl kabul edileceğine ilişkin sınama belirtimleri ve planları ortaya çıkarılmaktadır.

Mimari Model

Sistem tasarımı ve oluşacak altsistem ile tüm sistemin sınama işlemlerine ilişkin işlevler.

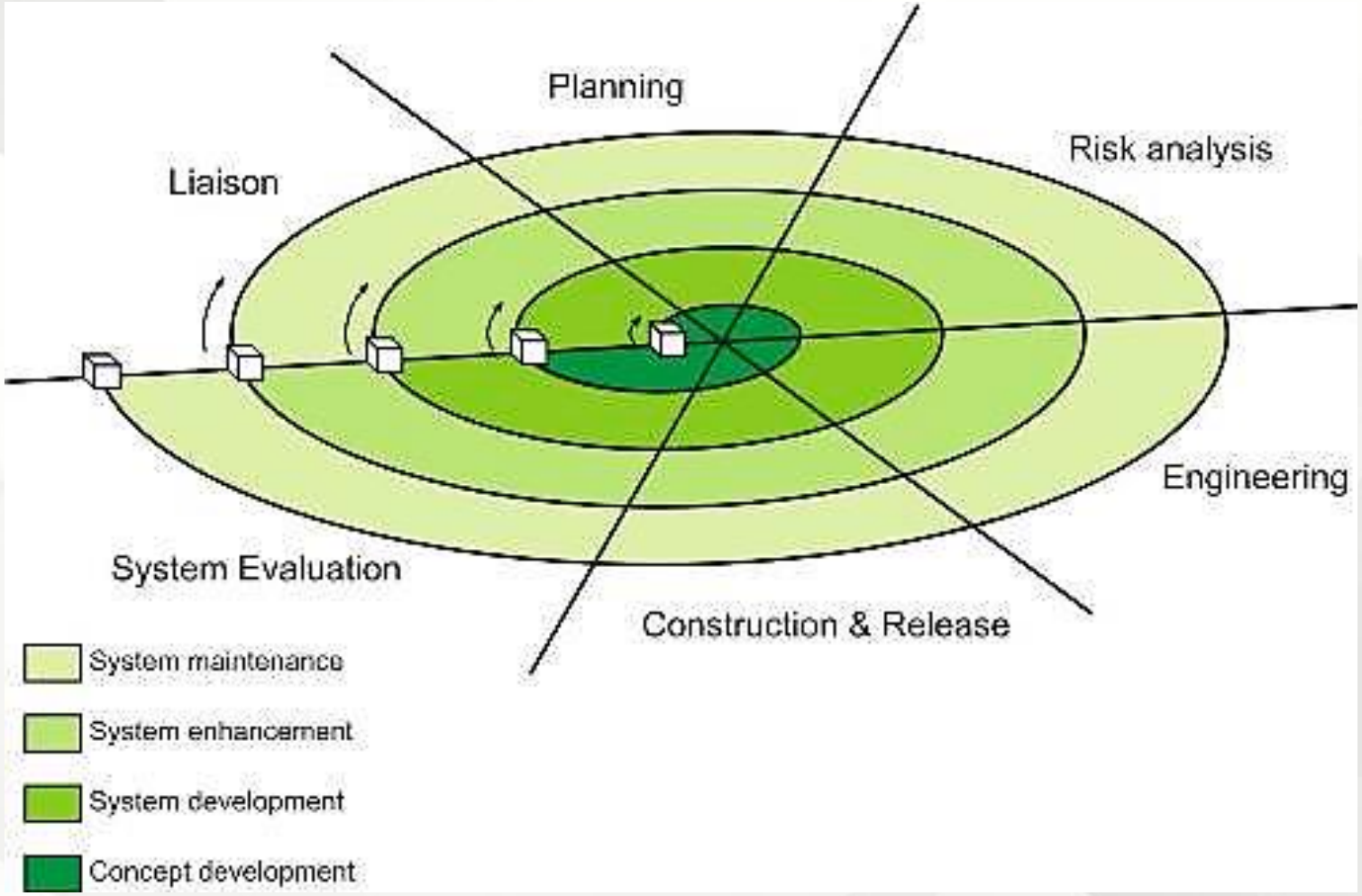
Gerçekleştirim Modeli

Yazılım modüllerinin kodlanması ve sınanmasına ilişkin fonksiyonlar.

V Süreç Modeli

- ✓Belirsizliklerin az, iş tanımlarının belirgin olduğu BT projeleri için uygun bir modeldir.
- ✓Model, kullanıcının projeye katkısını arttırmaktadır.
- ✓BT projesinin iki aşamalı olarak ihale edilmesi için oldukça uygundur:
  - İlk ihalede kullanıcı modeli hedeflenerek, iş analizi ve kabul sınamalarının tanımları yapılmakta,
  - İkinci ihalede ise ilkinde elde edilmiş olan kullanıcı modeli tasarlanıp, gerçekleşmektedir.

Spiral Model



Spiral Model

Tasarımı doğrusal bir süreç olarak gören diğer modellerin aksine, bu model spiral bir süreç olarak görür. Bu, yineleyici tasarım döngülerini genişleyen bir spiral olarak temsil ederek yapılır.

Genellikle iç çevrimler, gereksinim tanımının rafine edilmesi için prototipleme ile birlikte ihtiyaç analizinin erken evresini ve dış spiraller yazılım tasarımını aşamalı olarak temsil eder.

Her helezonda, tasarım çabalarını ve bu yineleme için ilgili riski değerlendirmek için bir risk değerlendirme aşaması vardır. Her spiralin sonunda, mevcut spiralin gözden geçirilebilmesi ve bir sonraki aşamanın planlanabilmesi için gözden geçirme aşaması vardır.

Spiral Model

Her tasarım sarmalının altı ana faaliyeti altı temel görevle temsil edilmektedir:

1. Müşteri İletişimi
2. Planlama
3. Risk Analizi
4. Yazılım Tasarımı
5. Üretim-dağıtım
6. Müşteri onayı

Avantajları

1. Risk analizi yapmaktadır.
2. Bu yazılım tasarım modeli, büyük yazılım projelerini tasarlamak ve yönetmek için daha uygundur.

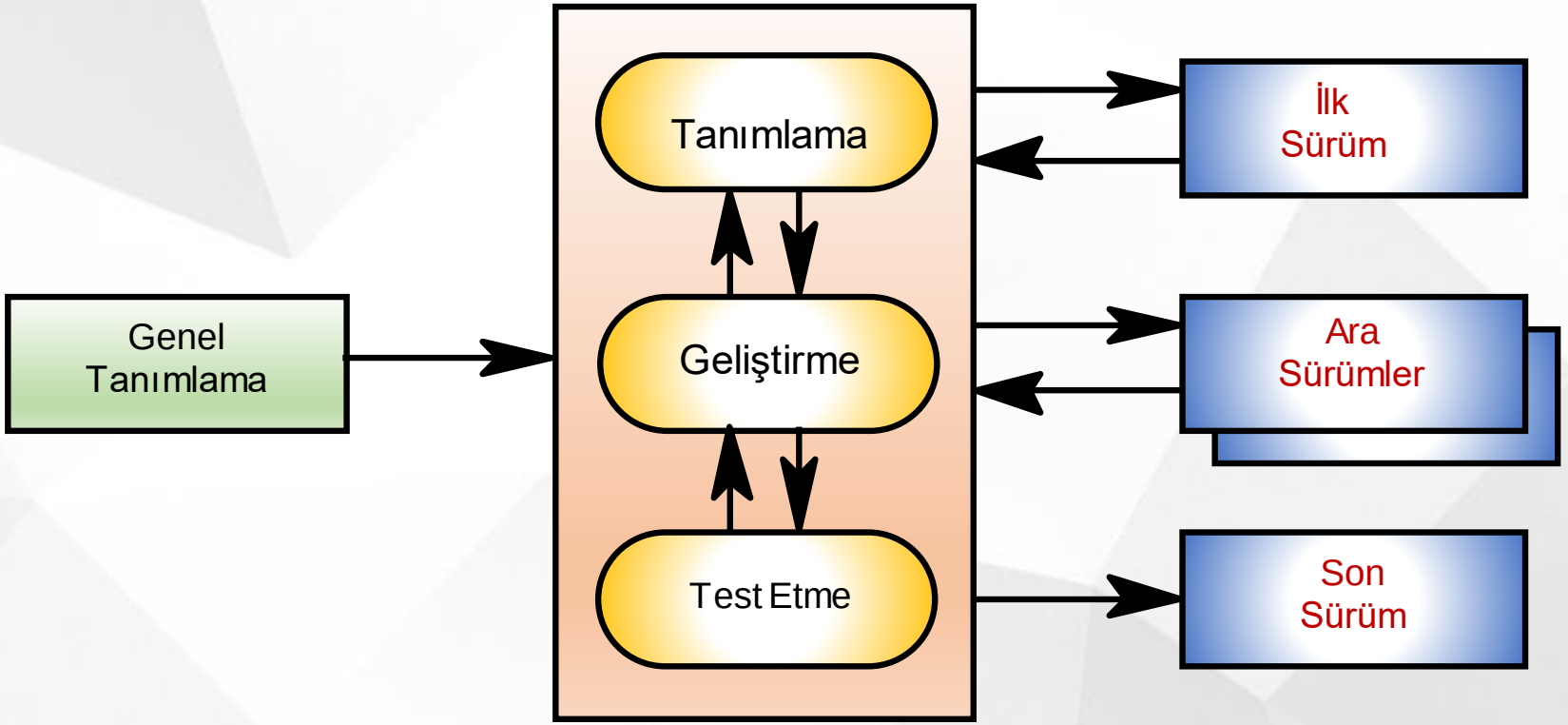
Dezavantajları

1. Risk analizi yüksek uzmanlık gerektirir.
2. Kullanması pahalı model
3. Küçük projeler için uygun değildir.

Evrimsel Geliştirme Süreç Modeli

- İlk tam ölçekli modeldir.
- Coğrafik olarak geniş alana yayılmış, çok birimli organizasyonlar için önerilmektedir.
- Her aşamada üretilen ürünler, üretildikleri alan için tam işlevselliği içermektedirler.
- Pilot uygulama kullan, test et, güncelle diğer birimlere taşı.
- Modelin başarısı ilk evrimin başarısına bağımlıdır

Evrimsel Geliştirme Süreç Modeli



Evrimsel Geliştirme Süreç Modeli

- ✓Çok birimli banka uygulamaları.
- ✓Önce sistem geliştirilir ve Şube-1'e yüklenir.
- ✓Daha sonra aksaklıklar giderilerek geliştirilen sistem Şube-2'ye yüklenir.
- ✓Daha sonra geliştirilen sistem Şube-3'e,…. yüklenir.
- ✓Belirli aralıklarla eski şubelerdeki güncellemeler yapılır.

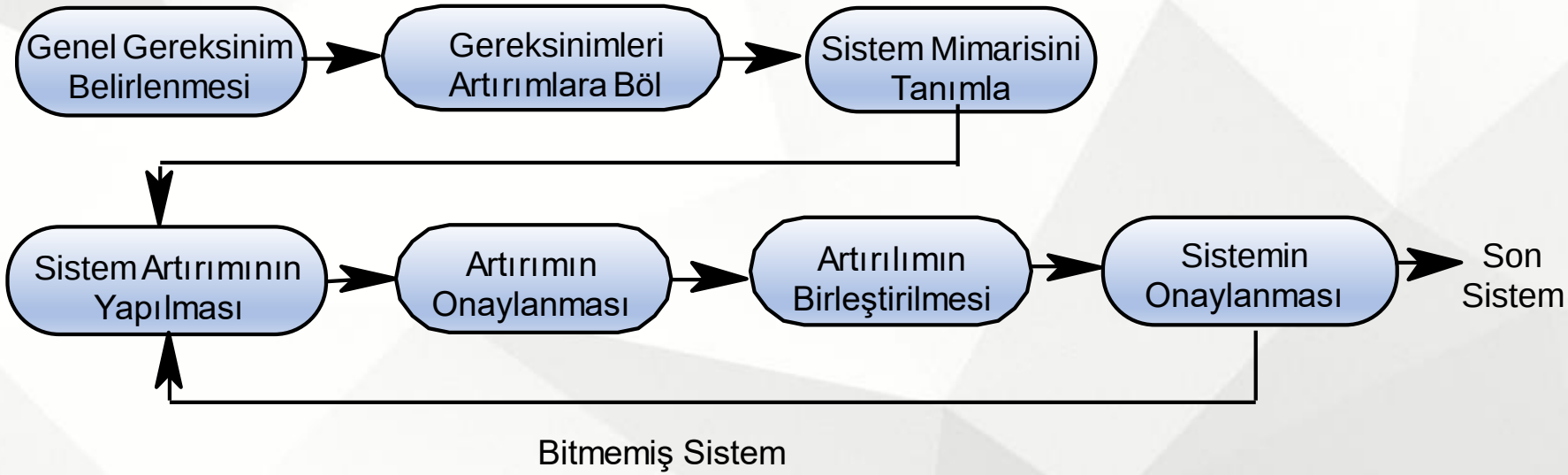
EKSİKLERİ

- ✓ Değişiklik denetimi
- ✓ Konfigürasyon Yönetimidir
  - Sürüm Yönetimi
  - Değişiklik Yönetimi
  - Kalite Yönetimi

Artırımsal Geliştirme Süreç Modeli

- ✓Üretilen her yazılım sürümü birbirini kapsayacak ve giderek artan sayıda işlev içerecek şekilde geliştirilir.
- ✓Öğrencilerin bir dönem boyunca geliştirmeleri gereken bir programlama ödevinin 2 haftada bir gelişiminin izlenmesi (bitirme tezleri).
- ✓Uzun zaman alabilecek ve sistemin eksik işlevlikle çalışabileceği türdeki projeler bu modele uygun olabilir.
- ✓Bir taraftan kullanım, diğer taraftan üretim yapılır

Artırımsal Geliştirme Süreç Modeli



Araştırma Tabanlı Süreç Modeli

- ✓Yap-at prototipi olarak ta bilinir.
- ✓Araştırma ortamları bütünüyle belirsizlik üzerine çalışan ortamlardır.
- ✓Yapılan işlerden edinilecek sonuçlar belirgin değildir.
- ✓Geliştirilen yazılımlar genellikle sınırlı sayıda kullanılır ve kullanım bittikten sonra işe yaramaz hale gelir ve atılır.
- ✓Model-zaman-fiyat kestirimi olmadığı için sabit fiyat sözleşmelerinde uygun değildir.

Metodolojiler

Metodoloji: Bir BT projesi ya da yazılım yaşam döngüsü aşamaları boyunca kullanılacak birbirleriyle uyumlu yöntemler bütünü.

Bir metodoloji,

- bir süreç modelini ve
- belirli sayıda belirtim yöntemini içerir

Günümüzdeki metodolojiler; çağlayan, V ya da spiral süreç modellerini temel almaktadır.

Bir Metodolojide Bulunması Gereken Temel Bileşenler

- ✓ Ayrıntılı bir süreç modeli
- ✓ Ayrıntılı süreç tanımları
- ✓ İyi tanımlı üretim yöntemleri
- ✓ Süreçler arası ara yüz tanımları
- ✓ Ayrıntılı girdi tanımları
- ✓ Ayrıntılı çıktı tanımları
- ✓ Proje yönetim modeli

- Konfigürasyon yönetim modeli
- Maliyet yönetim modeli
- Kalite yönetim modeli
- Risk yönetim modeli
- Değişiklik yönetim modeli
- Kullanıcı arayüz modeli
- Standartlar

Bir Metodolojide Bulunması Gereken Temel Bileşenler

- ✓Metodoloji bileşenleri ile ilgili olarak bağımsız kuruluş (IEEE, ISO, vs.) ve kişiler tarafından geliştirilmiş çeşitli standartlar ve rehberler mevcuttur.

- ✓Kullanılan süreç modelleri ve belirtim yöntemleri zaman içinde değiştiği için standart ve rehberler de sürekli güncellenmektedir.

Yourdon Yapısal Sistem Tasarım Metodolojisi

Aşama	Kullanılan Yöntem ve Araçlar	Ne için Kullanıldığı	Çıktı
Planlama	Veri Akış Şemaları, Süreç Belirtimleri, Görüşme, Maliyet Kestirim Yöntemi, Proje Yönetim Araçları	Süreç İnceleme Kaynak Kestirimi Proje Yönetimi	Proje Planı
Analiz	Veri Akış Şemaları, Süreç Belirtimleri, Görüşme, Nesne ilişki şemaları Veri	Süreç Analizi Veri Analizi	Sistem Analiz Raporu
Analizden Tasarıma Geçiş	Akışa Dayalı Analiz, Süreç belirtimlerinin program tasarım diline dönüştürülmesi, Nesne ilişki şemalarının veri tablosuna dönüştürülmesi	Başlangıç Tasarımı Ayrıntılı Tasarım Başlangıç Veri tasarımı	Başlangıç Tasarım Raporu
Tasarım	Yapısal Şemalar, Program Tasarım Dili, Veri Tabanı Tabloları	Genel Tasarım Ayrıntılı Tasarım Veri Tasarımı	Sistem Tasarım Raporu

ANKARA ÜNİVERSİTESİ ENFORMATİK BÖLÜMÜ
TEZSİZ YÜKSEK LİSANS PROGRAMI

Yazılım Mühendisliği
Temel Süreçler - PLANLAMA

Doç. Dr. Recep ERYİĞİT

HEDEFLER

- ✓ Proje kaynakları

  İnsan, donanım ve yazılım kaynakları

Planlama

Proje planlama aşamasında yapılan işlemler

- Kaynakların Belirlenmesi
- Maliyetlerin Kestirilmesi
- Proje Ekip Yapısının Oluşturulması
- Ayrıntılı Proje Planının Yapılması
- Projenin İzlenme Yönteminin Belirlenmesi

Çıktı: Proje Planı

Proje planı tüm proje süresince sürekli olarak kullanılacak, güncellenecek ve gözden geçirilecek bir belgedir

Proje Kaynakları

- ✓İnsan Kaynakları
- ✓Donanım Kaynakları
- ✓Yazılım Kaynakları

Planlama; bu kaynakların tanımını yapar ve zaman kullanımı, görev süreleri, edinilme zamanlarını planlar

İnsan Kaynakları

- ✓Planlama; hangi tür elemanların, hangi süre ile ve projenin hangi aşamalarında yer alacağını belirler

Proje Yöneticisi	Donanım Ekip Lideri
Yazılım Ekip Lideri	Donanım Mühendisi
Web Tasarımcısı	Ağ Uzmanı
Sistem Tasarımcısı	Yazılım Destek Elemanı
Programcı	Donanım Destek Elemanı
Sistem Yöneticisi	Eğitmen
Veri Tabanı Yöneticisi	Denetleyici
Kalite Sağlama Yöneticisi	Çağrı Merkezi Elemanı

Donanım Kaynakları

- ✓ Donanım Kaynakları:
  - Ana Bilgisayarlar
  - Sunucular (Web, E-posta, Veri Tabanı)
  - Kullanıcı Bilgisayarları (PC)
  - Yerel Alan Ağı (LAN) Alt Yapısı
  - Geniş Alan Ağı (WAN) Alt Yapısı
- ✓ Yazılımın geliştirileceği ortam, gerçek kullanım ortamı dışında olmalıdır.
- ✓ Öte yandan, geliştirme ve uygulama ortamlarının aynı konfigürasyonda olmaları, ileride kurulum sırasında ortaya çıkabilecek taşıma sorunlarını büyük ölçüde giderecektir.

Yazılım Kaynakları

- ✓Büyük ölçekte otomatik hale getirilmiş ve bilgisayar destekli olarak kullanılmaktadır.
- ✓Bilgisayar Destekli Tasarım (CAD) ve Bilgisayar Destekli Mühendislik (CASE) araçları olarak bilinmektedirler

✓Test araçları

- Yazılımı doğrulama ve geçerleme işlemlerinde kullanılır. Test verisi üreticiler, otomatik test yordamları, ...

✓Prototipleme ve simülasyon araçları

- Geliştirmenin erken aşamalarında kullanıcıya, sonuç ürünün çalışması ile ilgili fikir veren ve yönlendiren araçlar.

✓Bakım araçları

- Programın bakımını kolaylaştıran, bir kaynak koddan program şemalarının üretilmesini, veri yapısının ortaya çıkarılmasını sağlayan araçlar.

✓Destek araçları

- İşletim sistemleri, ağ yazılımları, e-posta ve ortam yönetim araçları.

Proje Maliyetleri

✓Maliyet kestirimi; bir bilgi sistemi ya da yazılım için gerekebilecek iş gücü ve zaman maliyetlerinin üretimden önce belirlenebilmesi için yapılan işlemlerdir.

✓ Kullanılan Unsurlar

- Geçmiş projelere ilişkin bilgiler
- Proje ekibinin deneyimleri
- İzlenen geliştirme modeli

birden çok kez uygulanabilir

Maliyet yönetimi sayesinde;

- ✓Gecikmeler önlenir
- ✓Bilgi sistemi geliştirme süreci kolaylaştırılır
- ✓Daha etkin kaynak kullanımı sağlanır
- ✓İş zaman planı etkin olarak gerçekleştirilir
- ✓Ürün sağlıklı olarak fiyatlandırılır
- ✓Ürün zamanında ve hedeflenen bütçe sınırları içerisinde bitirilir

Gözlemlenebilecek değerler

- ✓Projenin toplam süresi
- ✓Projenin toplam maliyeti
- ✓Projede çalışan eleman sayısı, niteliği, çalışma süresi
- ✓Toplam satır sayısı
- ✓Bir satırın maliyeti (ortalama)
- ✓Bir kişi/ay'da gerçekleştirilen satır sayısı
- ✓Toplam işlev sayısı
- ✓Bir işlevin maliyeti
- ✓Bir kişi/ay'da gerçekleştirilen işlev sayısı
- ✓Bir kişi/ay'da maliyeti

Maliyet Kestirim Yöntemleri

1. Projenin boyut türüne göre
   - Proje büyüklüğünü kestiren yöntemler
   - Proje zaman ve işgücünü kestiren yöntemler
2. Projelerin büyüklüğüne göre
   - Makro yöntemler (büyük boyutlu projeler 30 kişi-yıl)
   - Mikro Yöntemler (orta ve küçük boyutlu projeler)
3. Uygulanış biçimlerine göre
   - Çok yalın düzeyde
   - Orta ayrıntılı düzeyde
   - Çok ayrıntılı düzeyde

Maliyet Kestirim Yöntemleri

. Değişik aşamalarda kullanılabilirlik

- Planlama ve analiz aşamasında kullanılabilen
- Tasarım aşamasında kullanılabilen
- Gerçekleştirim aşamasında kullanılabilen yöntemler

5. Yöntemlerin yapılarına göre

- Uzman deneyimine gereksinim duyan
- Önceki projelerdeki bilgileri kullanan yöntemler

İşlev Noktaları Yöntemi

- ✓İşlev noktaları geliştirmenin erken aşamalarında (analiz aşamasında) saptanan bir değerdir.
- ✓Sistemin oluşturulduğu ortamdan bağımsız elde edilir.
- ✓Problem tanımı girdi olarak alınarak üç temel adım izlenir:
  - Problemin bilgi ortamının incelenmesi
  - Problemin teknik karmaşıklığının incelenmesi
  - İşlev noktası hesaplama

Problemin bilgi ortamının incelenmesi

- ✓Kullanıcı Girdileri: personel sicil bilgileri, personel izin bilgileri gibi
- ✓Kullanıcı Çıktıları: her türlü mantıksal çıktı; raporlar, ekran çıktıları, hata iletileri,...
- ✓Kullanıcı Sorguları: personel sicil bilgilerinin sorgulaması, personel maaş bilgilerinin sorgulaması
- ✓Dosyalar: Her türlü mantıksal bilgi yığını, tablolar, veri tabanları
- ✓Dışsal arayüzler: Başka programlarla veri iletimi. import/export

Bunların ağırlık faktörleriyle çarpımları toplanarak, Ayarlanmamış İşlev Nokta (AİN) sayısı hesaplanır.

Problem Bilgi Ortamı Bileşenleri

Ölçüm Parametresi	Sayı	Ağırlık Faktörü				
		Yalın	Ortalama	Karmaşık		
Kullanıcı Girdi sayısı	?	3	4	6	=	
Kullanıcı Çıktı sayısı	?	4	5	7	=	
Kullanıcı Sorgu Sayısı	?	3	4	6	=	
Kütük Sayısı	?	7	10	15	=	
Dışsal Araryüz Sayısı	?	5	7	10	=	
Toplam Sayı					=	

Problemin teknik karmaşıklığının incelenmesi

1. Uygulama, güvenilir yedekleme ve kurtarma gerektiriyor mu?
2. Veri iletişimi gerektiriyor mu?
3. Dağıtılmış İşlemler var mı?
4. Performans kritik mi?
5. Girdiler, çıktılar, dosyalar ya da sorgular karmaşık mı?
6. İçsel işlemler karmaşık mı?
7. Tasarlanacak kod yeniden kullanılabilir mi?
8. Dönüştürme ve kurulun tasarımda dikkate alınacak mı?

*****

Cevaplar 0 ile 5 arasında puanlandırılır

Bunlar hesaplanıp toplanarak Teknik Karmaşıklık Faktörü (TKF) elde edilir.

İşlev noktası sayısı hesaplama

✓İN=AİN*(0,65*0,01*TKF)

Değişik amaçlarla kullanılabilir

- **Üretkenlik** = İN / Kişi-Ay
- **Kalite** = Hatalar / İN
- **Maliyet** = $ / İN

Satır Sayısı Kestirimi

Assembly	300
Cobol	100
Fortran	100
Pascal	90
C	90
Ada	70
Nesne Kökenli Diller	30
4. Kuşak Dilleri	20
Kod Üreticiler	15

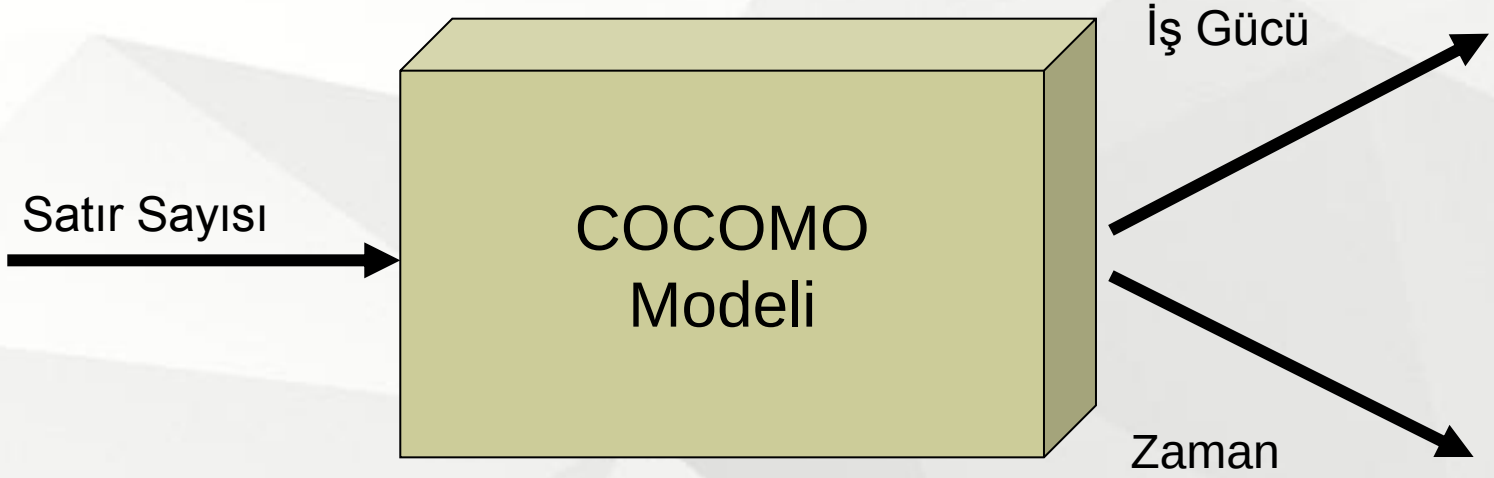
İN=300 ise ve Nesne Tabanlı bir dil (SmalTalk) kullanılıyor ise

Satır Sayısı=300*30

olarak hesaplanır

Etkin Maliyet Modeli

- ✓COCOMO 1981 Boehm
- ✓Mikro maliyet kestirim modeline örnektir.
- ✓Kullanılacak ayrıntı düzeyine göre üç ayrı model biçiminde yapılabilir:
  - Temel Model
  - Ara Model
  - Ayrıntı Model



COCOMO formülleri

- ✓İş Gücü (K) $K=a*S^b$
- ✓Zaman (T) $T=c*K^d$

a,b,c,d : her bir model için farklı katsayılar

S : bin türünden satır sayısı

Proje Sınıfları

✓Ayrık Projeler:

- Boyutları küçük,
- Deneyimli personel tarafından gerçekleştirilmiş
- LAN üzerinde çalışan insan kaynakları yönetim sistemi gibi

✓Yarı Gömülü:

Hem bilgi boyutu hem donanım sürme boyutu olan projeler

✓Gömülü Projeler:

Donanım sürmeyi hedefleyen projeler (pilotsuz uçağı süren yazılım - donanım kısıtları yüksek)

Temel Model

- ✓Küçük-orta boy projeler için hızlı kestirim yapmak amacıyla kullanılır
- ✓Dezavantajı: Yazılım projesinin geliştirileceği ortam ve yazılımı geliştirecek ekibin özelliklerini dikkate almaz
- ✓Avantajı: Hesap makinesi ile kolaylıkla uygulanabilir

- Ayrık Projeler
  - İş Gücü $K=2.4*S^{1,05}$
  - Zaman $T=2.5*K^{0,38}$
- Yarı Gömülü Projeler
  - İş Gücü $K=3,0*S^{1,12}$
  - Zaman $T=2.5*K^{0,35}$
- Gömülü Projeler
  - İş Gücü $K=3,6*S^{1,20}$
  - Zaman $T=2.5*K^{0,32}$

Ara Model

- ✓Temel modelin eksikliğini gidermek amacıyla oluşturulmuştur.
- ✓Bir yazılım projesinin zaman ve iş gücü maliyetlerinin kestiriminde;
  - Proje ekibinin özelliklerini,
  - Proje geliştirmede kullanılacak araçları, yöntem ve ortamı dikkate alır.
- ✓Üç Aşamadan oluşur:
  - İş gücü hesaplama
  - Maliyet çarpanı hesaplama
  - İlk iş gücü değerini düzeltme

İş Gücü Hesaplama

- ✓Ayrık Projeler $K=3.2*S^{1,05}$
- ✓Yarı Gömülü Projeler $K=3,0*S^{1,12}$
- ✓Gömülü Projeler $K=2.8*S^{1,20}$

Maliyet Çarpanı Hesaplama

- ✓Maliyet Çarpanı 15 maliyet etmeninin çarpımı sonucudur.

$$C = C1*C2*C3*\ldots*C15$$

Maliyet Etmenleri

Maliyet etmeni		Seçenekler					
		Çok Düşük	Düşük	Normal	Yüksek	Çok Yüksek	Oldukça Yüksek
Ürün Özellikleri	RELY	0,75	0,88	1,00	1,15	1,40	-
	DATA	-	0,94	1,00	1,08	1,16	-
	CPLX	0,70	0,85	1,00	1,15	1,30	1,65
Bilgisayar Özellikleri	TIME	-	-	1,00	1,11	1,30	1,66
	STOR	-	-	1,00	1,06	1,21	1,56
	VIRT	-	0,87	1,00	1,15	1,30	-
	TURN	-	0,87	1,00	1,07	1,15	-
Personel Özellikleri	ACAP	1,46	1,19	1,00	0,86	0,71	-
	AEXP	1,29	1,13	1,00	0,91	0,82	-
	PCAP	1,42	1,17	1,00	0,86	0,70	-
	VEXP	1,21	1,10	1,00	0,90	-	-
	LEXP	1,14	1,07	1,00	0,95	-	-
Proje Özellikleri	MODP	1,24	1,10	1,00	0,91	0,82	-
	TOOL	1,24	1,10	1,00	0,91	0,83	-
	SCED	1,23	1,08	1,00	1,04	1,10	-

Ürün Özellikleri

- ✓Rely: Yazılımın güvenirliği
- ✓Data: Veri Tabanının Büyüklüğü.
  Burada program büyüklüğüne oranı dikkate alınır.
- ✓Cplx: Karmaşıklığı.

Bilgisayar Özellikleri

- ✓Time: İşletim zamanı kısıtı
- ✓Stor: Ana Bellek Kısıtı
- ✓Virt: Bilgisayar Platform Değişim Olasılığı.

  Bellek ve Disk kapasitesi artırımı,

  CPU Upgrade
- ✓Turn: Bilgisayar İş Geri Dönüş Zamanı.

  Hata düzeltme süresi.

Personel Özellikleri

- ✓Acap: Analist Yeteneği:

  Deneyim, Birlikte çalışabilirlik.
- ✓Aexp: Uygulama Deneyimi.

  Proje ekibinin ortalama tecrübesi.
- ✓Pcap: Programcı Yeteneği.
- ✓Vexp: Bilgisayar Platformu Deneyimi.

  Proje ekibinin geliştirilecek platformu tanıma oranı.
- ✓Lexp: Programlama dili deneyimi.

Proje Özellikleri

- ✓Modp: Modern Programlama Teknikleri.
  - Yapısal programlama,
  - Görsel programlama,
  - Yeniden kullanılabilirlik.
- ✓Tool: Yazılım Geliştirme araçları kullanımı.
  - CASE araçları
  - Metin düzenleyiciler
  - Ortam yönetim araçları
- ✓Sced: Zaman Kısıtı.

İlk İşgücü değerini Düzeltme

✓Kd= K * C Kd= Düzeltilmiş İşgücü

* Temel Formüldeki Zamanla formülü kullanılarak zaman maliyeti hesaplanır.

Ayrıntı modeli

Temel ve ara modele ek olarak iki özellik taşır.

- ✓Aşama ile ilgili işgücü katsayıları: her aşama için (planlama, analiz, tasarım, geliştirme, test etme) farklı katsayılar, karmaşıklık belirler
- ✓Üç düzey ürün sıra düzeni: yazılım maliyet kestiriminde
  - Modül
  - Altsistem
  - Sistem

Sıra düzenini dikkate alır

Proje Ekip Yapısı Oluşturma

- ✓PANDA proje Ekip yapısı temel olarak her proje biriminin doğrudan proje yönetimine bağlı olarak çalışması ve işlevsel bölümlenme esasına göre oluşturulur. Temel bileşenler
  - Proje Denetim Birimi
  - Proje Yönetim Birimi
  - Kalite Yönetim Birimi
  - Proje Ofisi
  - Teknik Destek Birimi
  - Yazılım Üretim Eşgüdüm Birimi
  - Eğitim Birimi
  - Uygulama Destek Birimi

Yüklenici Proje Ekip Yapısı

- ✓Proje Denetim Birimi: En üst düzey yönetimlerin proje ile ilgisinin sürekli sıcak tutulması ve onların projeye dahil edilmesi
- ✓Proje Yönetim Birimi: Proje yönetiminden en üst düzeyde sorumlu birim.proje boyutuna göre bir yada daha çok yöneticiden oluşur.
- ✓Kalite Yönetim Birimi: Projenin amacına uygunluğunu üretim süreci boyunca denetler ve onaylar
- ✓Proje Ofisi: Her türlü yönetimsel işlerden(yazışma, personel izleme) sorumlu birimdir.

Yüklenici Proje Ekip Yapısı

- ✓Teknik Destek Birimi: Donanım, İşletim sistemi, Veri tabanı gibi teknik destek
- ✓Yazılım Üretim Eşgüdüm Birimi: Yazılım Üretim Ekiplerinden oluşur(4-7 kişilik sayı fazla artmaz). Eğer birden fazla yazılım Üretim Ekibi varsa Ortak uygulama yazılım parçalarının geliştirilmesinden sorumlu Yazılım Destek Ekibi de olur.
- ✓Eğitim Birimi: Proje ile ilgili her türlü eğitimden sorumludur.
- ✓Uygulama Destek Birimi: Uygulama anında destek. (mesela telefonla)

İş Sahibi Proje Ekip Yapısı

- ✓Proje Eşgüdüm Birimi
- ✓Kalite Yönetim Birimi
- ✓Proje Ofisi
- ✓Teknik Altyapı izleme birimi
- ✓Yazılım Üretim İzleme Birimi
- ✓Eğitim İzleme Birimi
- ✓Kullanıcı Eşgüdüm Birimi

Yazılım Mühendisliği
Temel Süreçler - *Sistem Analizi*

HEDEFLER

- ✓Planlama raporu içeriği
- ✓Yazılım Yaşam Döngüsü
- ✓Analiz- Greksininim nedir? Gereksinim türleri
- ✓Gereksinim Verisi Toplama Yöntemleri
- ✓Kullanıcı Arayüz Prototipleme (KAP)
- ✓Sistem Analiz Raporu

Proje Planı(Faaliyet-Zaman-Maliyet Çizelgesi)

Proje Kaynakları-

1. İnsan kaynakları : Proje şamalarında görev alacak personelin nitelikleri ve çalışma zamanları
2. Sistemin geliştirilmesinde ve nihai sistemde kullanılacak donanım kaynaklarının edinilme zaman çizelgesi
3. Sistem geliştirme sunumunda kullanılacak yazılım kaynaklarının edinilme tarihleri

Bu aşamanın en önemli görünür çıktısı projenin çıktılarına ait zaman çizelgesidir.

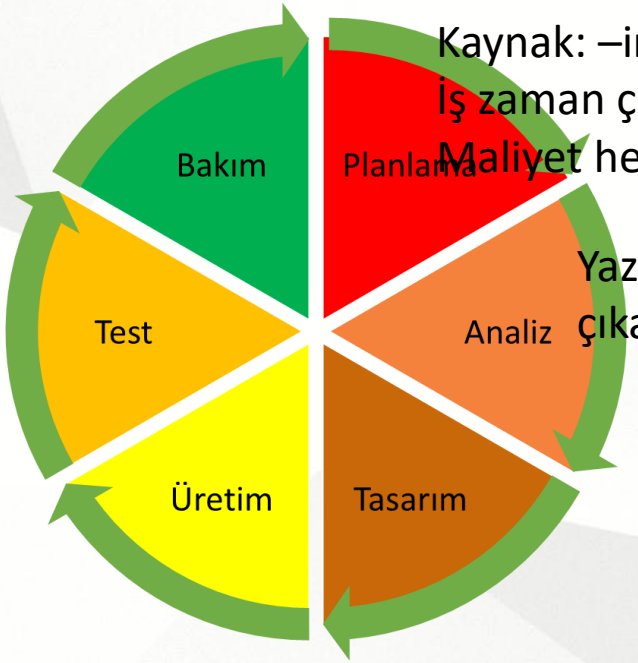
İlk maliyet hesaplama bu aşamada olmasına karşın proje planı raporunda genellikle yer almaz.

Projenin başlangıç tarihi : .../.../ 20...

20... Yılı Fiyatlarıyla

Faaliyet	I. Yıl				II. Yıl				III. Yıl	Maliyet (Bin TL)
	1-3. ay	4-6. ay	7-9. ay	10-12. ay	13-15. ay	16-18. ay	19-21. ay	22-24. ay	...	
1.										
1.1.										
1.2.										
2.										
2.1.										
2.2.										
2.3										
3.										
4.										
5.										
Toplam Tutar										

Yazılım Yaşam Döngüsü



Kaynak: –insan, donanım-yazılım
İş zaman çizelgesi
Maliyet hesabı

Yazılıma konu olan işlerin algoritmasının çıkarılması

Analiz (Çözümleme)

Amaç: Sistemin işlevlerini ve kesin gereksinimleri belirlemek ve dokümante etmek.

- ✓Analiz çalışması; müşteri, yazılım mühendisi, sistem analisti, iş analisti, ürün yöneticisi vb. rollerin bir araya geldiği gruplar tarafından yapılabilir. İhtiyaçların net olmadığı durumlarda yazılım mühendisi ve müşteri arasında iletişim ve birlikte çalışmanın çok daha fazla olması gerekir. Çeşitli yazılım geliştirme metodolojilerinde bu aşamada kullanım dokümanları ve test planları bu aşamada belirlenir.

Gereksinim Nedir?

Sistemin amaçlarını yerine getirme yeteneği olan bir özellik ya da belirtim olarak tanımlanmaktadır.

✓Gereksinim sistemin yada işlevlerinin nasıl yerine getirileceği ile ilgili değildir. Ne olduğu ile ilgilidir.

Geliştirilecek yazılımın yapması gerekenler, sistem gereksinimleri olarak adlandırılır.

İşlevsel Gereksinim

- ✓İşlevsel gereksinim (Kullanıcı gereksinimi); sistem ile çevresi arasındaki iletişimi belirleyen gereksinimlerdir.
- ✓Sistemin herhangi bir durum karşısındaki davranışını belirler.
  - bordronun ne zaman alınacağı
  - hangi verilerin alınacağı
  - çıktı formatı

İşlevsel Olmayan Gereksinimler

✓İşlevsel olmayan gereksinimler(sistemsel gereksinimler), kullanıcının sorunundan bağımsız olarak çözülmesi gereken işlemlerdir.

✓Sistem Kısıtları olarak ta adlandırılabilir

- kullanılacak bilgisayarın türü
- yazılım geliştirme ortamı
- kullanılacak veri tabanı yönetim sistemi

Gereksinim Türleri

- ✓Fiziksel Çevre
- ✓Arayüzler
- ✓Kullanıcı ve İnsan etmeni
- ✓İşlevsellik
- ✓Belgeleme
- ✓Veri
- ✓Kaynaklar
- ✓Güvenlik
- ✓Kalite Güvencesi

Fiziksel Çevre

- ✓İşlevlerin geliştirileceği, işletileceği aygıtlar nerededir.
- ✓Sistem tek bir yerde mi olacak? birden çok ve fiziksel olarak birbirinden ayrılmış yerler söz konusu mu?
- ✓Sıcaklık nem oranı veya manyetik etkileşim gibi çevresel kısıtlamalar var mı?

Arayüzler

- ✓Girdiler bir mi yoksa birden çok sistemden mi geliyor?
- ✓Çıktılar bir mi yoksa birden çok sisteme mi gidiyor?
- ✓Verilerin nasıl biçimlendirileceğine ilişkin bir yol var mı?
- ✓Verilerin kullanılacağı önerilen bir ortam var mı?

Kullanıcı ve İnsan etmeni

- ✓Sistemi kim kullanacak?
- ✓Farklı tiplerde kullanıcılar olacak mı?
- ✓Her bir kullanıcı tipinin yetenek düzeyi nedir?
- ✓Her kullanıcı tipi için ne tür eğitimler gerekli?
- ✓Bir kullanıcının sistemi kötü amaçlı kullanması ne ölçüde zordur?

İşlevsellik

- ✓Sistem ne yapacak?
- ✓Sistem bunu ne zaman gerçekleştirecek?
- ✓Sistem nasıl ve ne zaman değiştirilebilir ve/veya güçlendirilebilir?
- ✓Çalışma hızı, yanıt süresi ya da çıktı üzerinde kısıtlayıcı etmenler var mı?

Belgeleme

- ✓Ne kadar belgeleme gereklidir?
- ✓Belgeleme hangi kullanıcı kitlesini hedeflemektedir?

Veri

- ✓Hem giriş hem çıkış için verinin biçimi ne olmalıdır?
- ✓Bu veri ne sıklıkla alınacak veya gönderilecektir?
- ✓Bu verinin doğruluk (kesinlik) ölçüsü ne olmalıdır?
- ✓Hesaplamalar hangi duyarlık derecesine kadar yapılandırılacaktır?
- ✓Sistemde ne kadar veri akışı olacaktır?
- ✓Veri belirli bir zaman süresince kaynağında saklanacak mı?

Kaynaklar

- ✓Sistemi kurmak, kullanmak ve bakımını yapmak için ne kadar malzeme, personel ve diğer kaynaklara ihtiyaç var?
- ✓Geliştiriciler hangi yeteneklere sahip olmalı?
- ✓Sistem ne kadar fiziksel yer kaplayacak?
- ✓Güç, ısıtma ve soğutma için kısıtlar nelerdir?
- ✓Geliştirim için tavsiye edilen bir zaman çizelgesi var mı?

Güvenlik

- ✓Sisteme ya da bilgiye erişim denetlenmeli midir?
- ✓Bir kullanıcının verisi diğerinden nasıl ayrılacaktır?
- ✓Kullanıcı programları, diğer program ve işletim sisteminden nasıl ayrı tutulacaktır?
- ✓Sistem hangi sıklıkla yedeklenecektir?
- ✓Yedek kopyaları başka yerde saklanacak mıdır?
- ✓Yangın ve hırsızlığa karşı ne tür önlemler alınacaktır?
- ✓Internet erişimi var mı? Güvenlik kullanılıyor mu?

Kalite Güvencesi

- ✓Güvenirlilik için gereksinimler nelerdir?
- ✓Sistemin özellikleri insanlara nasıl aktarılmalıdır?
- ✓Sistem çökmeleri arasında öngörülen zaman aralığı nedir?
- ✓Kaynak kullanımı ve yanıt süresine ilişkin verimlilik ölçütleri nelerdir?

Gereksinim Özellikleri

Gereksinimler üç amaca hizmet eder

- ✓Geliştiricilerin, müşterilerin sistemin nasıl çalışmasını istediklerini anlamalarını sağlar.
- ✓Gereksinimler, sonuç sistemin ne özellikte ve işlevsellikte olacağını söyler.
- ✓Gereksinimler sınama ekibine, kullanıcıyı, sunulan sistemin istenen sistem olduğuna ikna etmek için neler göstermeleri gerektiğini söyler.

Doğrulama Süreci

1. Gereksinimler doğru oluşturulmuş mu?
2. Gereksinimler tutarlı mı?
3. Gereksinimler tam mı? (Dışsal tamlık / İçsel tamlık)
4. Gereksinimler gerçekçi mi?
5. Her gereksinim kullanıcı tarafından istenen bir şeyi mi tanımlamaktadır?
6. Gereksinimler doğrulanabilir mi?
7. Gereksinimler izlenebilir mi?

Sistem Çözümleme Çalışması

✓Geliştirilecek bilgi sistemi yada yazılımla ilgili olarak;

- tüm gereksinimlerin araştırılması,
- tanımlanması,
- ortaya çıkarılması ve
- bir gösterim biçimi ile açıklanması

çalışmasıdır.

Mevcut sistemin incelenmesi

- ✓Amaç: Yazılım geliştirilecek sistemin tanınmasıdır.
- ✓Girdi, İşlev ve çıktı analizi yapılır.
- ✓Kanun, yönerge ve yönetmenlikler incelenir.
- ✓Elde yürütülen işlerde kullanılan form, defter ve yazışma örnekleri incelenir.

Önerilen Sistemin Modellenmesi

- ✓Önerilen sistemin işlevsel yapısını, veri yapısını ve kullanıcı arayüzünü oluşturur.
- ✓Bu model daha çok bilgi sistemini geliştirecek teknik personele yöneliktir.
- ✓Mantıksal model olarak ta tanımlanır.

Gereksinim Verisi Toplama Yöntemleri

- ✓Gereksinim Verisi Toplama Yöntemleri
  - Sorma
  - Karşılıklı görüşme (Anket)
  - Psikolojik türetme
  - İstatiksel teknikler
- ✓Veri Modelleme Yöntemleri
  - Nesne İlişki şemaları (1-1,1-N, M-N)
  - Veri Sözlüğü
- ✓Süreç/İşlem Modelleme yöntemleri

Sorma Yöntemi

- ✓Amaçlar, resmi olmayan yöntemler, duygular ve düşünceler araştırılır.
- ✓Yönlendirici sorular (bence.....) ve iki nesneli sorulardan kaçınılmalıdır (ne zaman ve nasıl...?).

Anket Yöntemi

- ✓Kullanıcı sayısının fazla olduğu durumlarda eğilimleri ve davranış biçimlerini saptamak için kullanılır.
- ✓Anket değerlendirilirken gerçekçi olmayan değerlendirmeler çıkarılmalıdır.

Psikolojik Türetme Teknikleri

- ✓Özellikle belirsizliğin fazla olduğu ve zayıf yapılı ortamlarda, bilgi edinebilmek amacıyla insan psikolojisine dayalı teknikler kullanılır.

İstatistiksel Teknikler

- ✓Veri yoğun ve veri hacmi yüksek ortamlarda verinin özelliklerini belirlemek amacıyla kullanılır.

Örnekleme yöntemi ve PIRA yöntemi.

Kullanıcı Arayüz Prototipleme (KAP)

- ✓Ekran tasarımı için kullanıcıdan onay alınması esastır.
- ✓Geleneksel yaklaşımlarda bilgi sistemi girdi ve çıktılarının tanımları el ile kağıt üzerinde yapılır ve kullanıcılardan bu biçimiyle onay alınmaya çalışılır.
- ✓Gereksinimlerin kesinleştirilmesini kolaylaştırır.

KAP Özellikleri

- ✓Ayrılan zaman sistem analizi için ayrılan zamanın %5'ini aşmamalıdır.
- ✓Her özellik bir kez gösterilmelidir.
- ✓Hiç bir içsel işlem içermemelidir.

KAP Raporları

- ✓Raporların bir kod numarası olmalıdır.
- ✓Her rapor için örnek çıktı yapısı ayarlanır. Word dokümanında örnek yapı hazırlanır. İlgili çıktı gönderilirken bu çıktı gönderilir.

Sistem Analiz Raporu

- ✓Sistem analiz çalışması sonucunda alınan rapordur. Söz Konusu rapor çalışmanın tüm ayrıntılarını içerir.
  - Giriş
  - Mevcut sistemin incelenmesi
  - İstenen sistem mantıksal modeli
  - Arayüz gerekleri
  - Belgeleme gerekleri

Yazılım Mühendisliği
Temel Süreçler - *Tasarım*

HEDEFLER

Tasarım, Sistem Analizi çalışması sonucunda üretilen Mantıksal Modelin Fiziksel Modele dönüştürülme çalışmasıdır.

- ✓ Tasarım kavramları: Soyutlama, İyileştirme ve Modülerlik olmak üzere 3 çeşittir.
- ✓ Yapı Tasarımı, arayüz tasarımı ve süreç tasarımından önce yapılması gereken ilk tasarım veri tasarımıdır.
- ✓ Veri Akışları Üç parçada incelenebilir: Girdi Akışı, Çıktı Akışı ve İşlem Akışı
- ✓ Süreç tasarımı; veri, yapı ve ara yüz tasarımından sonra yapılır.
- ✓ Program Akış Diyagramları: Tekrarlı, ardışıl ve koşullu şeklindedir.
- ✓ Tasarlanması Gereken Ortak Alt Sistemler

  - Yetkilendirme altsistemi
  - Güvenlik altsistemi
  - Yedekleme altsistemi
  - Veri transferi altsistemi
  - Arşiv altsistemi
  - Dönüştürme altsistemi

HEDEFLER

- ✓Tasarım Kavramları
- ✓MODÜL-işlevsel Bağımsızlık
- ✓Veri Tasarımı
- ✓Tasarlanması Gereken ortak alt sistemler
- ✓Tasarım Kalitesi
- ✓Bağlaşıklık -Yapışıklık

TASARIM

✓Tasarım, Sistem Analizi çalışması sonucunda üretilen Mantıksal Modelin Fiziksel Modele dönüştürülme çalışmasıdır.

✓Fiziksel Model geliştirilecek yazılımın;

- hangi parçalardan oluşacağını,
- bu parçalar arasındaki ilişkilerin neler olacağını,
- parçaların iç yapısının ayrıntılarını,
- gerekecek veri yapısının fiziksel biçimini (dosya, veri tabanı, hash tablosu, vektör, vs)

tasarımını içerir

TASARIM KAVRAMLARI

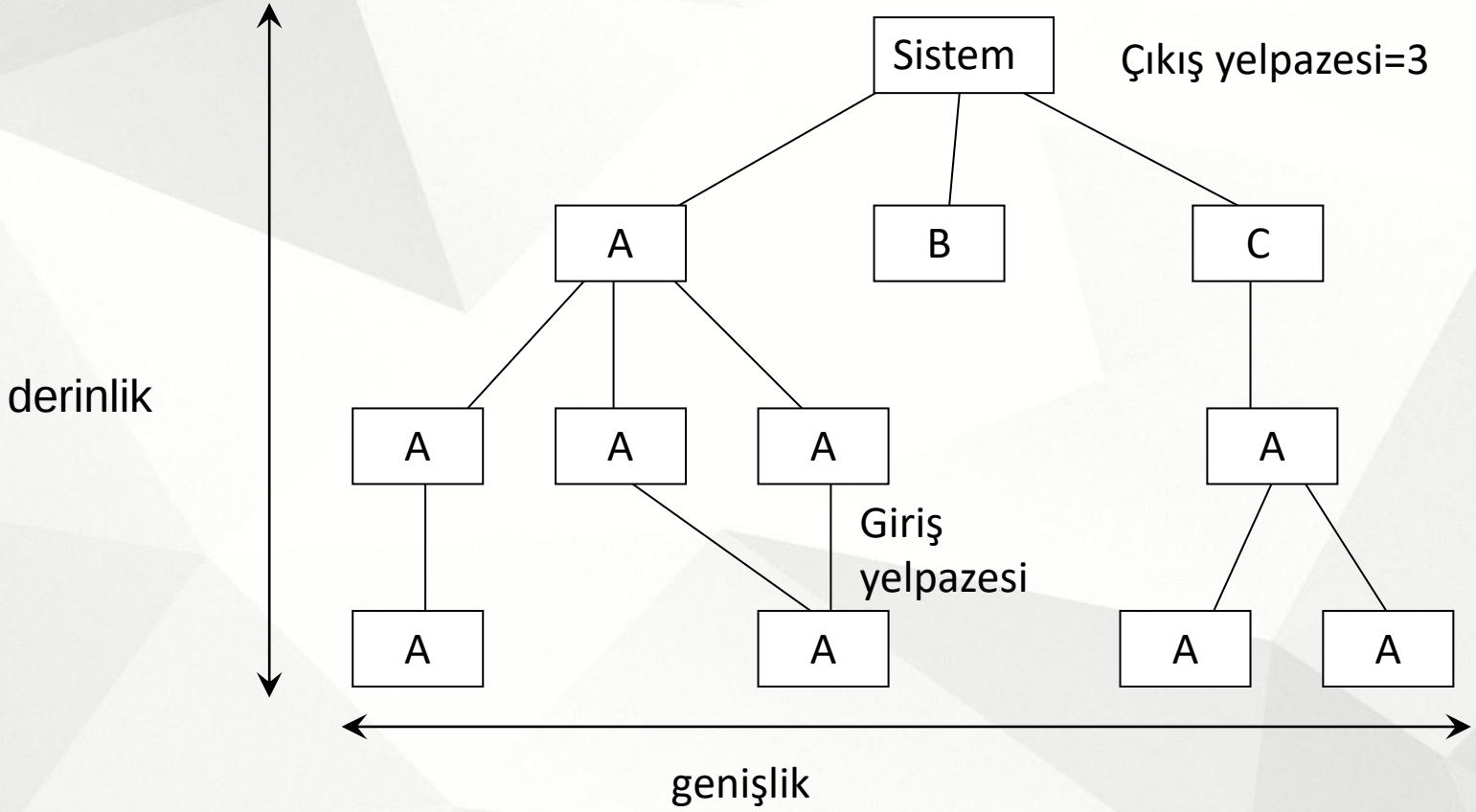
- ✓Soyutlama (abstraction): Detayları gizleyerek yukarıdan bakabilme imkanı sağlanır.
- ✓İyileştirme (enhancement): Soyutlama düzeyinde irdeleme bittikten sonra, daha alt seviyelere inilerek tanımlamalarda ayrıntı, bazen de düzeltme yapılarak tasarımın daha kesinlik kazanması sağlanır.
- ✓Modülerlik (modularity): Sistemi istenen kalite faktörleri ışığında parçalara ayrıştırma sonucu elde edilir.

MODÜL

- ✓Bütün karmaşıklığın tek bir modülde toplanması yerine, anlaşılabilir ve dolayısıyla projenin zihinsel kontrol altında tutulması için sistem bir çok modüle ayrılır.

- ✓Modüller, isimleri olan tanımlanmış işlevleri bulunan ve hedef sistemi gerçekleştirmek üzere tümleştirilen birimlerdir.

MODÜL


Sistem
Çıkış yelpazesi=3
A
B
C
A
A
A
A
A
A
A
A
derinlik
Giriş
yelpazesi
genişlik

İşlevsel Bağımsızlık

- ✓Modüllere parametre ile veri gönderilir ve sonuç değer alınır. Bu modülü çağıran program parçası sadece bu sonucu kullanabilir. Çağrılan modülün işlevsel olarak yaptıkları ile ilgili değildir.

Veri Tasarımı

- ✓Yapı Tasarımı, arayüz tasarımı ve süreç tasarımından önce yapılması gereken ilk tasarım veri tasarımıdır.

- ✓Bilgi saklama ve soyutlama bu işlem için önemli kavramlardır.

Veri tasarımında dikkat edilecek konular

- ✓Değişik veri yapıları değerlendirilmelidir.
- ✓Bütün veri yapıları ve bunlar üzerinde yapılacak işlemler tanımlanmalıdır.
- ✓Alt düzeyde tasarım kararları tasarım süreci içerisinde geciktirilmelidir.
- ✓Bazı çok kullanılan veri yapıları için bir kütüphane oluşturulmalıdır.
- ✓Kullanılacak programlama dili soyut veri tiplerini desteklemelidir.

Yapısal Tasarım

- ✓Yapısal Tasarımın ana hedefi modüler bir yapı geliştirip modüller arasındaki kontrol ilişkilerini temsil etmektir.
- ✓Ayrıca yapısal tasarım bazen de veri akışlarını gösteren biçime dönüştürülebilir.
- ✓Veri Akışları Üç parçada incelenebilir
  - Girdi Akışı
  - Çıktı Akışı
  - İşlem Akışı

AYRINTI TASARIM- Süreç Tasarımı

- ✓Süreç tasarımı; veri, yapı ve arayüz tasarımından sonra yapılır.
- ✓İdeal şartlarda bütün algoritmik detayın belirtilmesi amaçlanır.
- ✓Ayrıca süreç belirtiminin tek anlamı olması gerekir, değişik şahıslar tarafından farklı yorumlanmamalıdır.
- ✓Doğal diller kullanılabilir (açıklamalarda, çünkü doğal dil tek anlamlı değildir)
- ✓Süreç Tanımlama Dili (PDL)

Yapısal Program Yapıları

- ✓Yapısal programlamanın temel amacı;
  - program karmaşıklığını en aza indirmek,
  - program anlaşılabilirliğini artırmaktır.
- ✓Bu amaçla şu yapılar kullanılır;
  - Ardışıl İşlem yapısı
  - Koşullu işlem yapısı
  - Döngü yapısı

Karar Tabloları

- ✓Bazen karmaşık koşul değerlendirmeleri yapmak gerekir. Bunların düzenli bir gösterilimi karar tablolarında yapılabilir.
- ✓Öncelikle, bütün işlemler saptanmalı, sonra ön koşullar belirlenmelidir.
- ✓Belirli işlemler ile belirli koşulları birleştirerek tablo oluşturulur.
- ✓Alt tarafta ise işlemler benzer satırlar olarak gösterilir.

Program Tanımlama Dili

- ✓Program Tanımlama Dilleri süreç belirtiminde doğal dillerin programlama dili ile sentezlenmesi şeklinde ortaya çıkmıştır.

- ✓Hangi programlama dilinin kullanılacağından bağımsız özellikler bulunmalıdır.

```
DO
        Hesap Numarasını Oku
        IF (hesap numarası geçerli değil) başlangıca dön
            işlem türünü iste
        IF (para yatırma islemi) { para_yatir(); Başlangıca dön}
        IF (yeterli bakiye yok) başlangıca dön
WHILE
```

Tasarlanması Gereken Ortak Alt Sistemler

- ✓Yetkilendirme altsistemi
- ✓Güvenlik altsistemi
- ✓Yedekleme altsistemi
- ✓Veri transferi altsistemi
- ✓Arşiv altsistemi
- ✓Dönüştürme altsistemi

Yetkilendirme Alt Sistemi

- ✓Özellikle kurumsal uygulamalarda farklı kullanıcıların kullanabilecekleri ve kullanamayacakları özellikleri ifade eder.
  - İşlev bazında yetkilendirme
  - Ekran bazında yetkilendirme
  - Ekran alanları bazında yetkilendirme

Güvenlik Alt Sistemi

- ✓ Yapılan bir işlemde, işlemi yapan kullanıcının izlerinin saklanması işlemleri.
- ✓ LOG files (Sistem günlüğü)

Yedekleme Alt Sistemi

- ✓Her bilgi sisteminin olağandışı durumlara hazırlıklı olmak amacıyla kullandıkları veri tabanı (sistem) yedekleme ve yedekten geri alma işlemlerinin olması gerekmektedir.

- ✓Coğrafi olarak dağıtılmış hizmet birimlerinde çalışan makineler arasında veri akışının sağlanması işlemleri
- ✓Çevirim içi veri iletimi (real-time)
- ✓Çevirim dışı veri iletimi (disketler, teypler)

Arşiv Alt Sistemi

- ✓Belirli bir süre sonrasında sık olarak kullanılmayacak olan bilgilerin ayrılması ve gerektiğinde bu bilgilere erişimi sağlayan alt sistemlerdir.

- ✓Aktif veri tabanı.

Dönüştürme Alt Sistemi

- ✓Geliştirilen bilgi sisteminin uygulamaya alınmadan önce veri dönüştürme (mevcut sistemdeki verilerin yeni bilgi sistemine aktarılması) işlemlerine ihtiyaç vardır.

Dönüştürme Alt Sistemi

- ✓Geliştirilen bilgi sisteminin uygulamaya alınmadan önce veri dönüştürme (mevcut sistemdeki verilerin yeni bilgi sistemine aktarılması) işlemlerine ihtiyaç vardır.

Kullanıcı Arayüz Tasarımı

- ✓Kullanıcı ile ilişkisi olmayan arayüzler
  - Modüller arası arayüz
  - Sistem ile dış nesneler arası arayüz

- ✓Kullanıcı arayüzleri
  - Kullanım kolaylığı ve öğrenim zamanı esastır
  - Program=arayüz yaklaşımı vardır

Genel Prensipler

- ✓Veri giriş formlarının tutarlı olması
- ✓Önemli silmelerde teyit alınmalı
- ✓Yapılan çoğu işlem geri alınabilmeli
- ✓Hataların affedilmesi, yanlış girişte kırılmama
- ✓Komut isimlerinin kısa ve basit olması
- ✓Menülerin ve diğer etkileşimli araçların standart yapıda kullanımı

Bilgi Gösterimi

- ✓Yalnızca içinde bulunulan konu çerçevesi ile ilgili bilgi gösterilmeli
- ✓Veri çokluğu ile kullanıcı bunaltılmamalı, grafik ve resimler kullanılmalı
- ✓Tutarlı başlık, renkleme ve kısaltma kullanılmalı
- ✓Hata mesajları açıklayıcı ve anlaşılır olmalı
- ✓Değişik tür bilgiler kendi içinde sınıflandırılmalı
- ✓Rakamsal ifadelerde analog görüntü verilmeli (%89 değil)

Veri Girişi

- ✓Kullanıcı hareketleri en aza indirilmeli
- ✓Gösterim ve girdi sahaları birbirinden ayrılmalı (renk)
- ✓Kullanıcı uyarlamasına izin verilmeli, kullanıcı bazı özellikleri tanımlayabilmeli
- ✓Kullanılan konu ile ilgili gereksiz komutlar deaktifleştirilmeli
- ✓Bütün girdiler için yardım kolaylıkları olmalı

Kullanıcı Arayüz Prototipi

- ✓Tasarım çalışması sonucunda, daha önceden gereksinim çalışması sırasında hazırlanmış olan kullanıcı arayüz prototipi, ekran ve rapor tasarımları biçimine dönüşür. Ekranlar son halini alır, raporlar kesinleşir. Kullanıcıya gösterilerek onay alınır.

- ✓Tüm programın tek elden çıktığının ifade edilebilmesi açısından tüm ekranların aynı şablon üzerine oturtulması önerilmektedir.
  - **Menü Çubuğu**
  - **Araç Çubuğu**
  - **Gövde (Değişebilir)**
  - **Durum Çubuğu**

Başlangıç Tasarım Gözden Geçirme

✓Yapılan tasarım çalışmasının bir önceki geliştirme aşaması olan analiz aşamasında belirlenen gereksinimleri karşılayıp karşılamadığının belirlenmesidir.

- Sistem gereksinimlerine yardımcı olan kullanıcılar
- Sistem analizini yapan çözümleyiciler
- Sistemin kullanıcıları
- Tasarımcılar
- Yönlendirici
- Sekreter
- Sistemi geliştirecek programcılar

dan oluşan bir grup tarafından yapılır.

Ayrıntılı Tasarım Gözden Geçirme

✓Başlangıç tasarımı gözden geçirme çalışmasının başarılı bir biçimde tamamlanmasından sonra, tasarımın teknik uygunluğunu belirlemek için Ayrıntılı Tasarım Gözden Geçirme çalışması yapılır. Bu çalışmada;

- Çözümleyiciler
- Sistem Tasarımcıları
- Sistem Geliştiriciler
- Sekreter

den oluşan bir ekip kullanılır.

Tasarım Kalite Ölçütleri

✓Bağlaşım (Coupling)

Tasarımı oluşturan modüller arası ilişki ile ilgilidir.

✓Yapışıklık (Cohesion)

Modüllerin iç yapısı ile ilgilidir.

Bağlaşım

- ✓Modüller arası bağlılığın ölçülmesi için kullanılan bir ölçüttür.
- ✓Yüksek kaliteli bir tasarımda bağlaşım ölçümü az olmalıdır.
- ✓Bağlaşımın düşük olması
  - Hatanın dalgasal yayılma özelliğinin azaltılması
  - Modüllerin bakım kolaylığı
  - Modüller arası ilişkilerde karmaşıklığın azaltılması

  nedenleri ile istenmektedir

Yalın Veri Bağlaşımı

- ✓Herhangi iki modül arası iletişim yalın veriler (tamsayı, karakter, boolean, vs) aracılığı ile gerçekleştiriliyorsa bu iki modül yalın veri bağlaşımlıdır şeklinde tanımlanır.

Karmaşık Veri Bağlaşımı

- ✓Herhangi iki modül arasındaki iletişimde kullanılan parametrelerin karmaşık veri yapısı (kayıt, dizi, nesne, vs) olması durumunda modüller karmaşık veri paylaşımlı olarak tanımlanır.

Denetim Bağlaşımı

- ✓İki Modül arasında iletişim parametresi olarak denetim verisi kullanılıyorsa bu iki modül denetim bağlaşımlı olarak tanımlanır.

Ortak Veri Bağlaşımı

- ✓Eğer iki modül ortak bir alanda tanımlanmış verilere ulaşabiliyorsa bu iki modül ortak veri bağlaşımlı olarak tanımlanır.

- ✓Verilerin ortak veri bağlaşımlı olmaları şu nedenlerden dolayı fazla istenmez;
  - Ortak veri alanını izlemek zordur.
  - Ortak veri kullanan modüllerde yapılan değişiklikler diğer modülleri etkiler.
  - Ortak veri üzerinde yapılacak değişikliklerde bu veriyi kullanacak bütün modüller göz önüne alınmalıdır.

Ortak Veri Bağlaşımı

- ✓Eğer iki modül ortak bir alanda tanımlanmış verilere ulaşabiliyorsa bu iki modül ortak veri bağlaşımlı olarak tanımlanır.

- ✓Verilerin ortak veri bağlaşımlı olmaları şu nedenlerden dolayı fazla istenmez;
  - Ortak veri alanını izlemek zordur.
  - Ortak veri kullanan modüllerde yapılan değişiklikler diğer modülleri etkiler.
  - Ortak veri üzerinde yapılacak değişikliklerde bu veriyi kullanacak bütün modüller göz önüne alınmalıdır.

İçerik Bağlaşımı

- ✓Modüllerin iç içe tasarlanması sonucu, bir modülün başka bir modül içerisinde tanımlanmış veri alanına erişebilmesi olanaklaşır ve bu durum içerik bağlaşımına yol açar.

Yapışıklık

- ✓Bir modülün kendi içindeki işlemler arasındaki ilişkilere ilişkin bir ölçüttür. Modül gücü olarak ta tanımlanır.
- ✓Tasarımda yapışıklık özelliğinin yüksek olması tercih edilir.
- ✓Yapışıklık ile Bağlaşım ters orantılıdır.

İşlevsel Yapışıklık

- ✓İşlevsel Yapışık bir modül, tek bir iş problemine ilişkin sorunu çözen modül olarak tanımlanır.

- ✓Maas_Hesapla, Alan_Hesapla gibi

Sırasal Yapışıklık

- ✓Bir modülün içindeki işlemler incelendiğinde, bir işlemin çıktısı, diğer bir işlemin girdisi olarak kullanılıyorsa bu modül sırasal yapışık bir modül olarak adlandırılır.

Ham_Veri_Kaydını_Düzelt
Duzeltilmis_Ham_Veri_Kaydini_Dogrula
Dogrulanmis_Kaydi_Gonder

İletişimsel Yapışıklık

- ✓Bir modülün içindeki farklı işlemler aynı girdi ya da çıktıyı kullanıyorlarsa bu modül iletişimsel yapışık bir modül olarak adlandırılır.

Sicil_No_yu_Al
Adres_Bilgisini_Bul
Telefon_Bilgisini_Bul
Maas_Bilgisini_Bul

Yordamsal Yapışıklık

- ✓Yordamsal Yapışık modüldeki işlemler arasında denetim ilişkisi bulunmaktadır.
- ✓İşlemlerin birbirleri ile veri ilişkisi yoktur, ancak işlem sırası önemlidir.

Ekran_Goruntusunu_Yaz
Giris_Kaydini_Oku

Mantıksal Yapışıklık

- ✓Mantıksal olarak aynı türdeki işlemlerin bir araya toplandığı modüller mantıksal yapışık olarak adlandırılır.

Dizilere değer atama işlemleri

Gelişigüzel Yapışıklık

- ✓İşlemler arasında herhangi bir ilişki bulunmaz.

Ara_Kayit_Oku
B_dizisine_baslangic_deger_ata
Stok_kutugu_oku
Hata_iletisi_yaz