Dr. Yüksel YurtaY

4.
Hafta

1

YAZILIM GELİŞTİRME METODOLOJİSİ VE YAŞAM DÖNGÜSÜ

METODOLOJİLER

- Metodoloji: Bir BT projesi ya da yazılım yaşam döngüsü aşamaları boyunca kullanılacak ve birbirleriyle uyumlu yöntemler bütünü.
- Bir metodoloji,
  - bir süreç modelini ve
  - belirli sayıda belirtim yöntemini içerir
- Günümüzdeki metodolojiler genelde Çağlayan ya da Helezonik modeli temel almaktadır

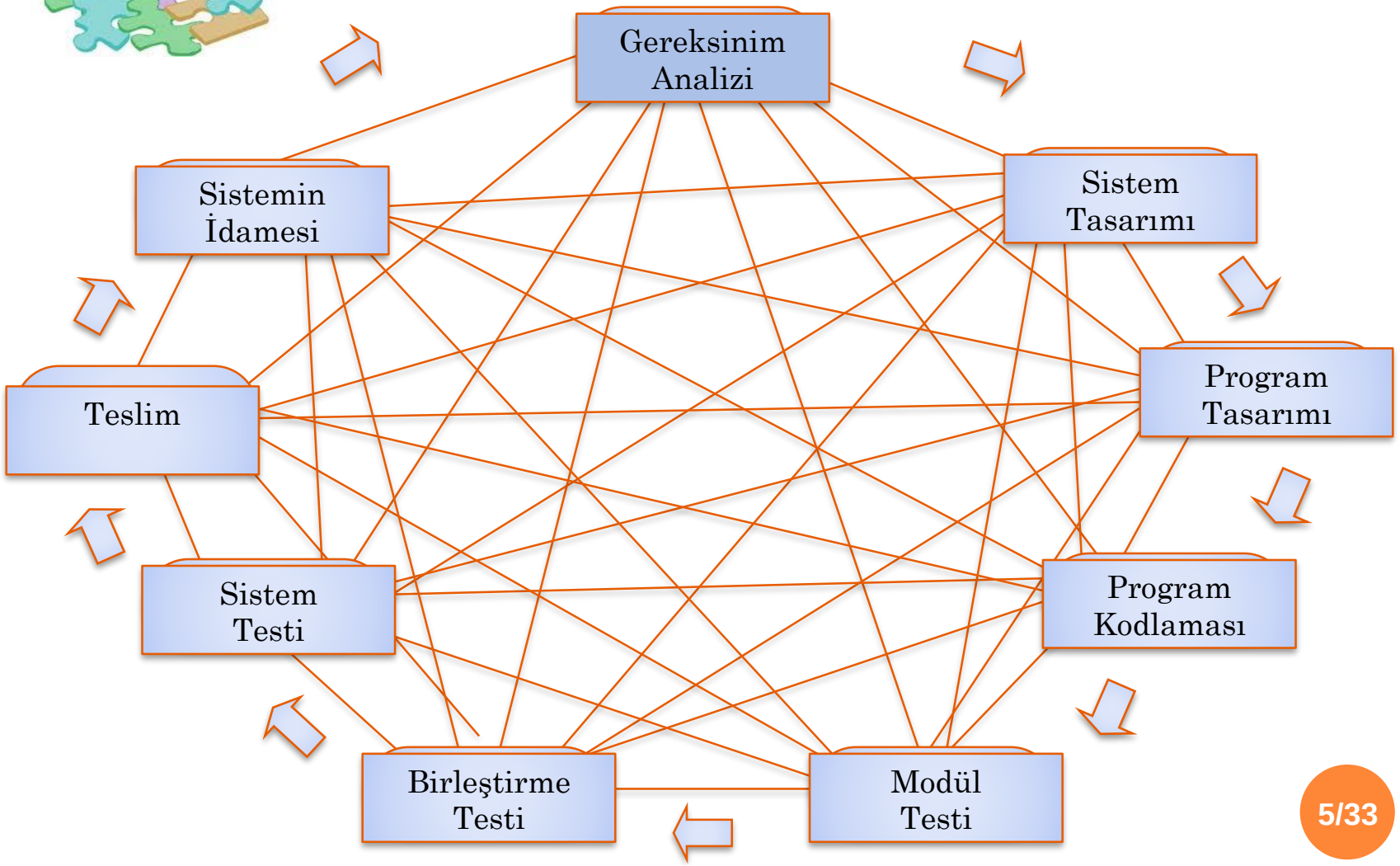
Bir Metodolojide Bulunması Gereken Temel Bileşenler (Özellikler)

- Ayrıntılandırılmış bir süreç modeli
- Ayrıntılı süreç tanımları
- İyi tanımlı üretim yöntemleri
- Süreçlerarası arayüz 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 ve ilişki 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.
- Bir kuruluşun kendi metodolojisini geliştirmesi oldukça kapsamlı, zaman alıcı ve uzmanlık gerektiren bir faaliyet olup, istatistikler yaklaşık 50 kişi/ay'lık bir iş gücü gerektirdiğini göstermektedir.

Gerçek Hayatta Program Geliştirme



Yazılım Geliştirme Yaşam Döngüsü

- Yazılımın hem üretim, hem de kullanım süreci boyunca geçirdiği tüm aşamalar yazılım geliştirme yaşam döngüsü olarak tanımlanır.
- Yazılım işlevleri ile ilgili gereksinimler sürekli olarak değiştiği ve genişlediği için, söz konusu aşamalar sürekli bir döngü biçiminde ele alınır.
- Döngü içerisinde her hangi bir aşamada geriye dönmek ve tekrar ilerlemek söz konusudur.
- Yazılım yaşam döngüsü tek yönlü ve doğrusal değildir.

Yazılım Yaşam Döngüsü Temel Adımları

1. **Analiz**

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

2. **Planlama(yol haritası)**

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

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 (teslimden sonra).

Yazilim Yaşam Döngüsü Temel Adimlari

- Yaşam döngüsünün temel adımları çekirdek süreçler (core processes) olarak da adlandırılır.

- Bu süreçlerin gerçekleştirilmesi amacıyla
  - Belirtim (specification) yöntemleri - bir çekirdek sürece ilişkin fonksiyonları yerine getirmek amacıyla kullanılan yöntemler
  - Süreç (process) modelleri - yazılım yaşam döngüsünde belirtilen süreçlerin geliştirme aşamasında, hangi düzen ya da sırada, nasıl uygulanacağını tanımlayan modeller kulanılır.

Belirtim Yöntemleri

- **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 Süreci Modelleri

- Yazılım üretim işinin genel 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. Helezonik (Spiral) Model
6. Evrimsel Model
7. Artırımsal Model
8. Araştırma Tabanlı Model

GELİŞİGÜZEL MODEL

- Herhangi bir model ya da yöntem yok.
- Geliştiren kişiye bağımlı (belli bir süre sonra o kişi bile sistemi anlayamaz ve geliştirme güçlüğü yaşar).
- İzlenebilirliği ve bakımı oldukça zor.
- 60'lı yıllarda.
- Genellikle tek kişilik üretim ortamı.
- Basit programlama.

Barok Modeli

- İnceleme
- Analiz
- Tasarım
- Kodlama
- Modül Testleri
- Alt sistem Testleri
- Sistem Testi
- Belgeleme
- Kurulum

Yaşam döngüsü temel adımlarının doğrusal bir şekilde geliştirildiği model.

70'li yıllar.

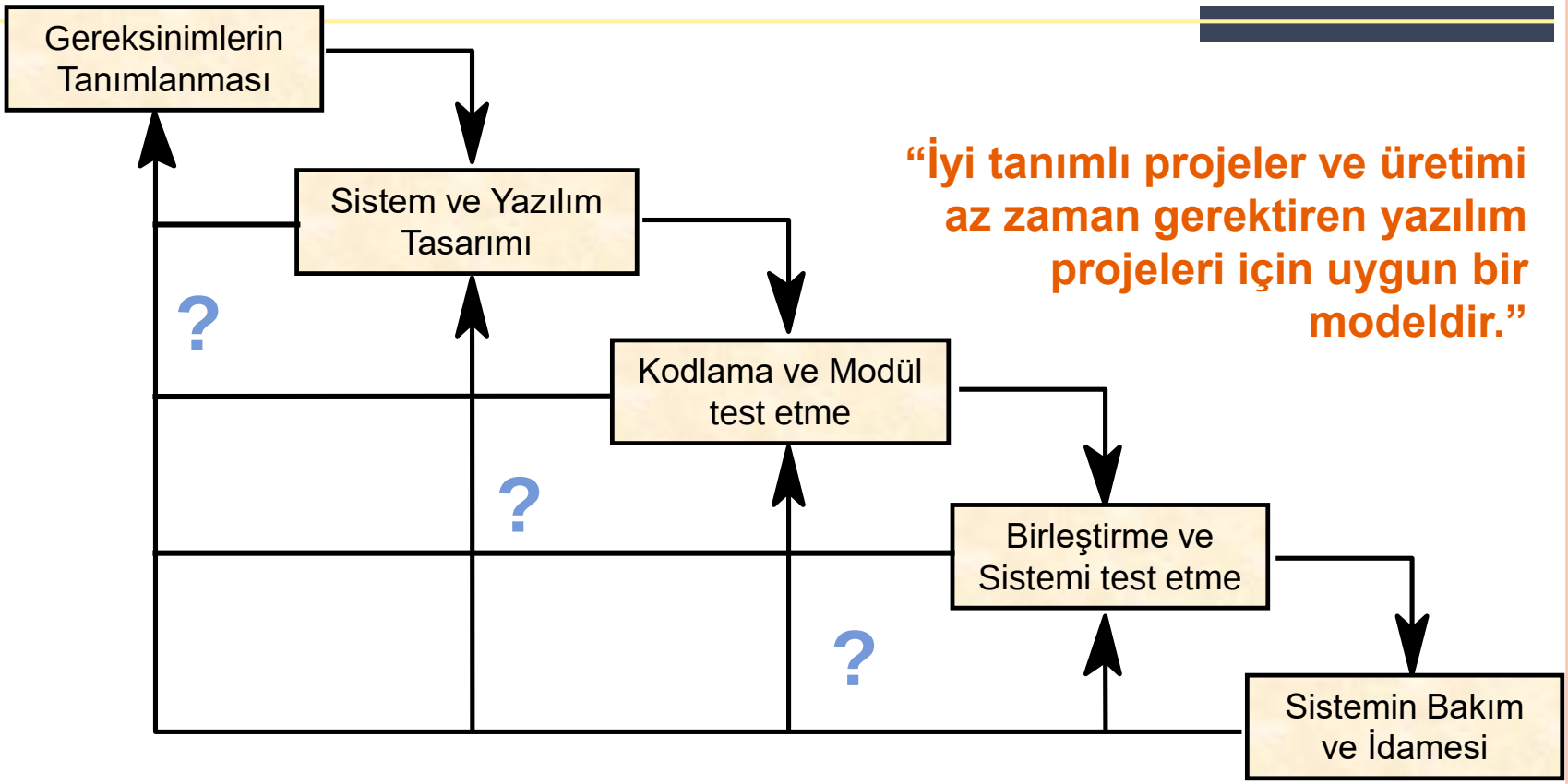
Belgelemeyi ayrı bir süreç olarak ele alır, ve yazılımın geliştirilmesi ve testinden hemen sonra yapılmasının öngörür.

Halbuki, günümüzde belgeleme yapılan işin doğal bir ürünü olarak görülmektedir.

Aşamalar arası geri dönüşlerin nasıl yapılacağı tanımlı değil.

" Gerçekleştirim aşamasına daha fazla ağırlık veren bir model olup, günümüzde kullanımı önerilmemektedir. "

Çağlayan (Şelale) Modeli



"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."

" Yazılım yaşam döngüsü adımları baştan sona en az bir kez izlenir. "

Çağlayan (Şelale) Modeli

- 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 (Şelale) Modeli Sorunları

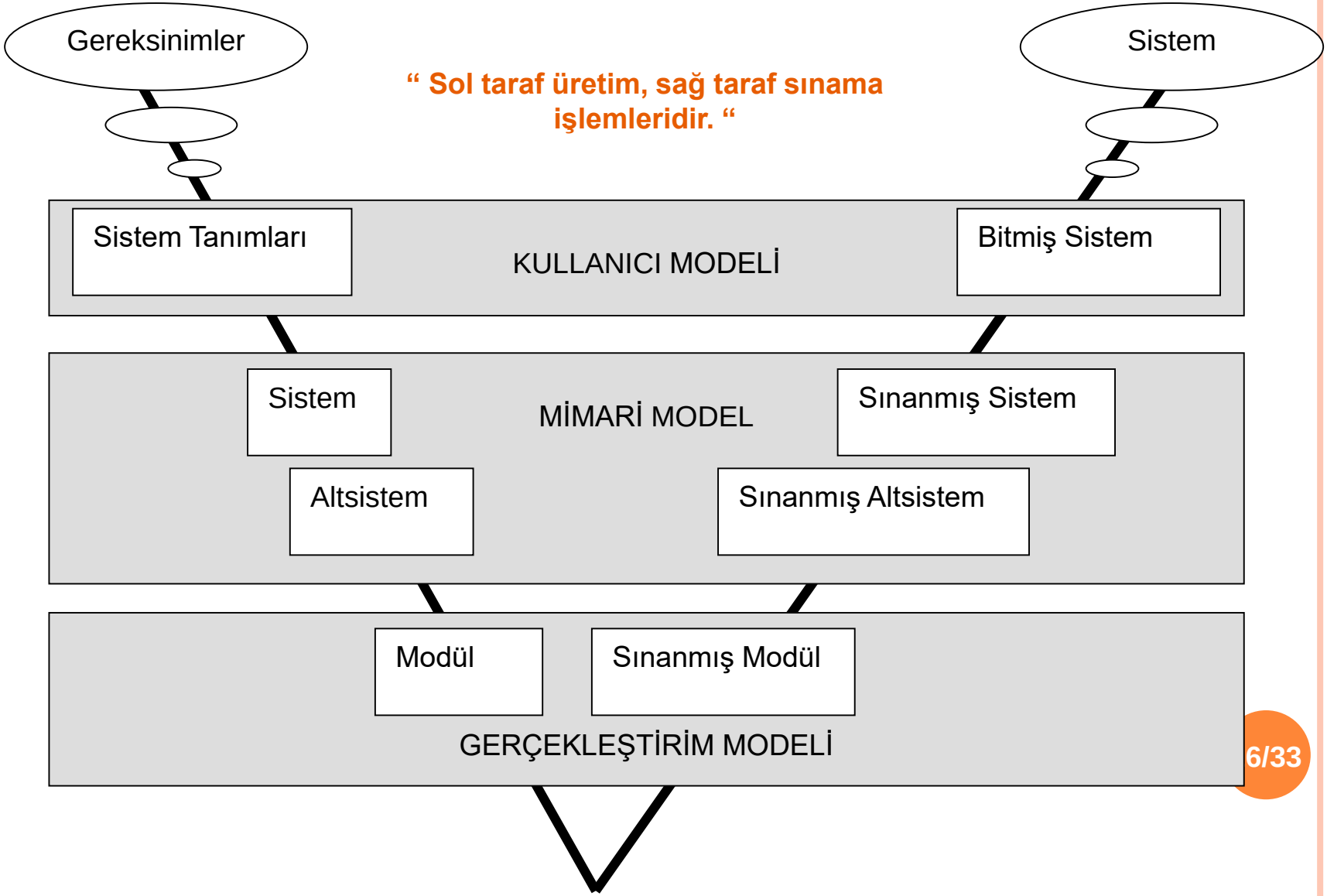
?

- 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.
- Üst düzey yönetimlerin ürünü görme süresinin uzun oluşu, projenin bitmeyeceği ve sürekli gider merkezi haline geldiği düşüncesini yaygınlaştırmaktadır.

V Süreç Modeli

"Belirsizliklerin az iş tanımlarının belirgin olduğu bilişim teknolojileri projeleri için uygun bir modeldir. "

" Sol taraf üretim, sağ taraf sınama işlemleridir. "



V Süreç Modeli

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

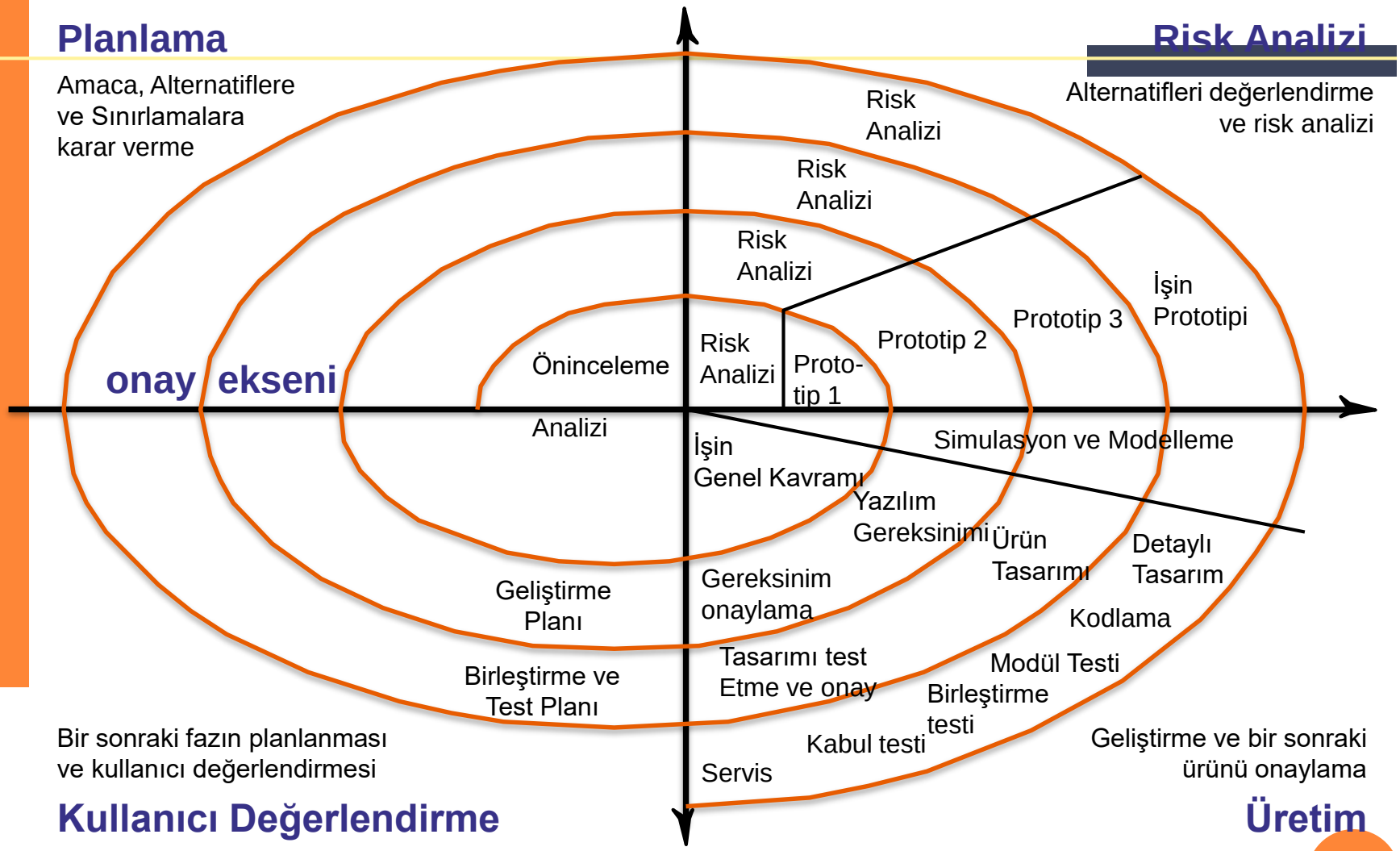
  Sistem tasarımı ve oluşacak alt sistem ile tüm sistemin sınama işlemlerine ilişkin işlevler.

- Gerçekleştirim Modeli

  Yazılım modüllerinin kodlanması ve sınanmasına ilişkin fonksiyonlar.

Model, kullanıcının projeye katkısını arttırmaktadır.

HELEZONİK(SPİRAL) MODELİ



Helezonik Model

1. **Planlama**

   Üretilecek ara ürün için planlama, amaç belirleme, bir önceki adımda üretilen ara ürün ile bütünleştirme

2. **Risk Analizi**

   Risk seçeneklerinin araştırılması ve risklerin belirlenmesi

3. **Üretim**

   Ara ürünün üretilmesi

4. **Kullanıcı Değerlendirmesi**

   Ara ürün ile ilgili olarak kullanıcı tarafından yapılan sınama ve değerlendirmeler

HELEZONİK MODELİN AVANTAJLARI

1. Kullanıcı Katkısı

   Üretim süreci boyunca ara ürün üretme ve üretilen ara ürünün kullanıcı tarafından sınanması temeline dayanır.

   Yazılımı kullanacak personelin sürece erken katılması ileride oluşabilecek istenmeyen durumları engeller.

2. Yönetici Bakışı

   Gerek proje sahibi, gerekse yüklenici tarafındaki yöneticiler, çalışan yazılımlarla proje boyunca karşılaştıkları için daha kolay izleme ve hak ediş planlaması yapılır.

3. Yazılım Geliştirici (Mühendis) Bakışı

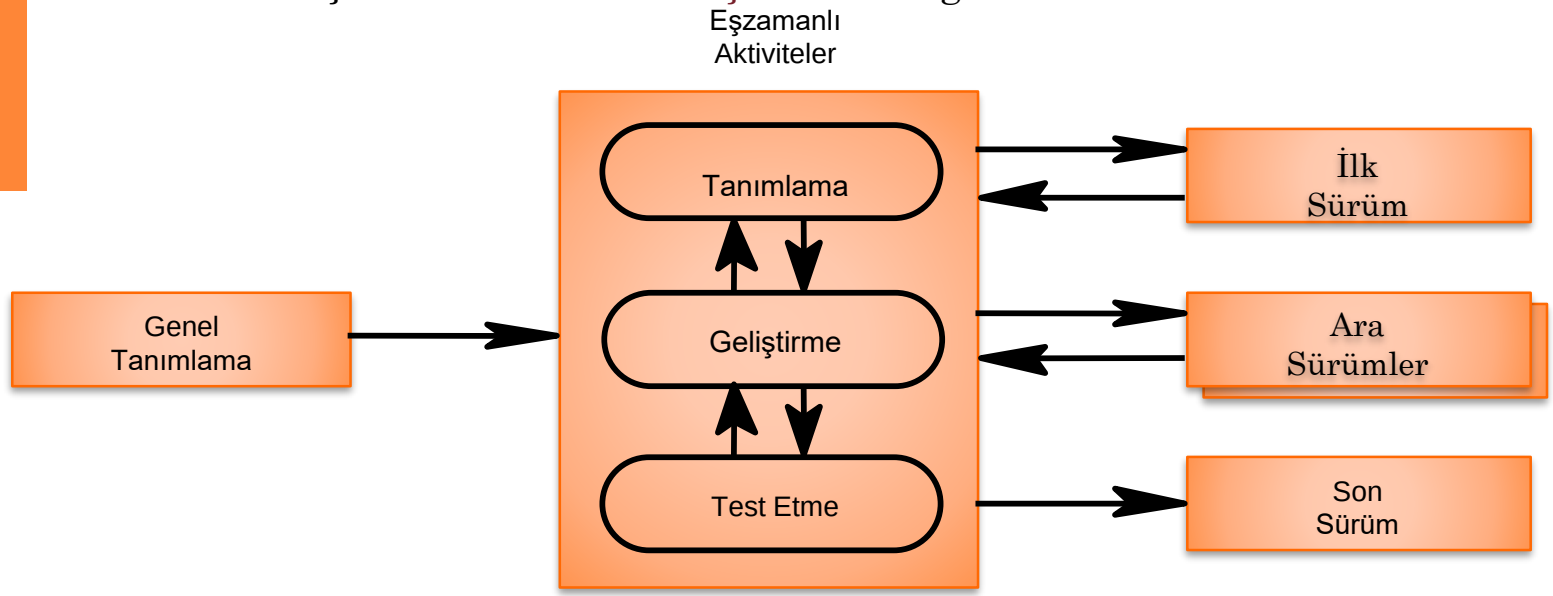
   Yazılımın kodlanması ve sınanması daha erken başlar.

Helezonik Model

- Risk Analizi Olgusu ön plana çıkmıştır.
- Her döngü bir fazı ifade eder. Doğrudan tanımlama, tasarım,... vs gibi bir faz yoktur.
- Yinelemeli artımsal bir yaklaşım vardır.
- Prototip yaklaşımı vardır.

Evrimsel Geliştirme Süreç Modeli

- İlk tam ölçekli modeldir.
- Coğrafik olarak geniş alana yayılmış, çok birimli organizasyonlar için önerilmektedir (banka uygulamaları).
- 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

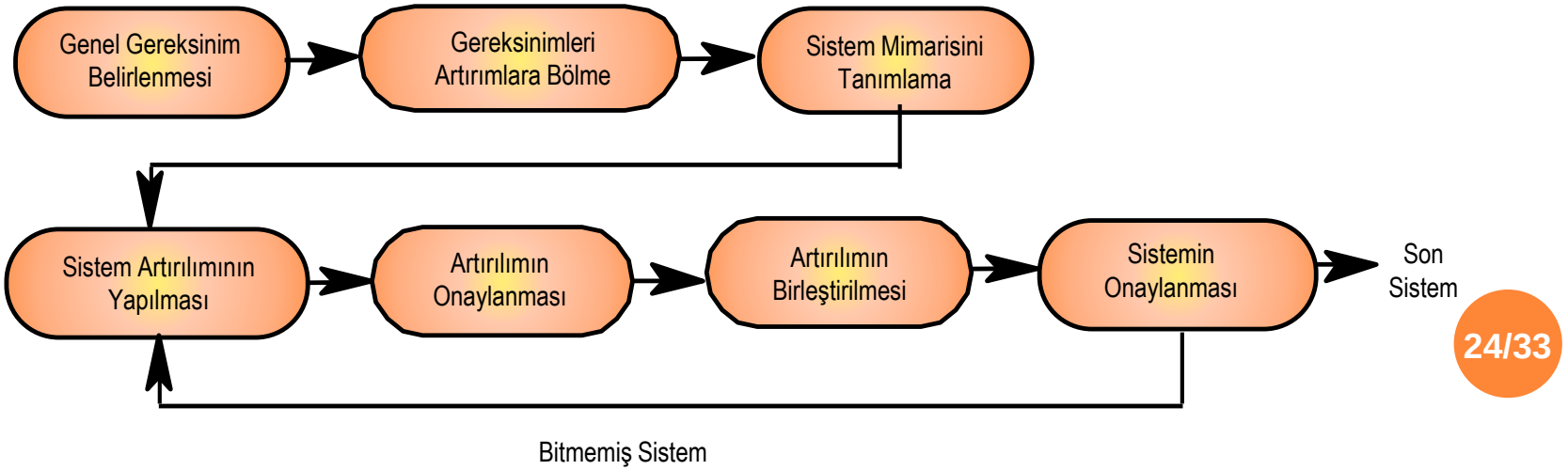
- Banka ve şubeleri olan işletme uygulamaları.
- Önce sistem geliştirilir ve adım adım dağıtılır.
- Her adımda geri besleme ile düzeltmeler yapılır.

Sorunlar :

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

Artirimsal 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.



Araştırma Tabanlı Süreç Modeli

- Araştırma ortamları bütünüyle belirsizlik üzerine çalışan ortamlardır.
- Yap-at prototipi olarak ta bilinir.
- 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.

- En Hızlı Çalışan asal sayı test programı!
- En Büyük asal sayıyı bulma programı!
- Satranç programı!
- …

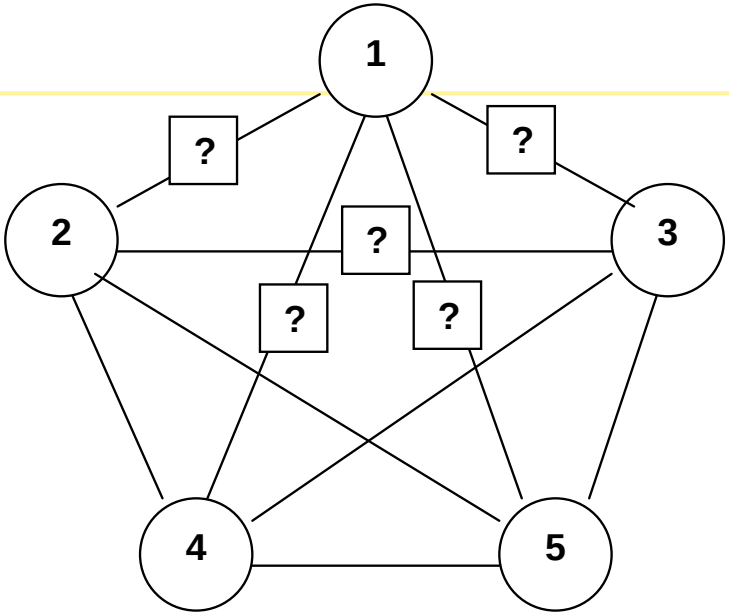
Yourdon Yapisal Sistem Tasarim Metodolojisi

Kolay uygulanabilir bir model olup, günümüzde oldukça yaygın olarak kullanılmaktadır.

Çağlayan modelini temel almaktadır.

Bir çok CASE aracı tarafından doğrudan desteklenmektedir.

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



1: Düşük Maliyetli Gerçekleştirim
2: Bakım Kolaylığı
3: Zamanında Teslim
4: Güvenirliği Yüksek Yazılım
5: Uzmanca Yazılım Geliştirme

Uygulama Yazılımı Geliştirmede Çelişen Tercihler

Pilot uygulama yapmaya üç nedenle başvurulur:

Kaynak :

www.mehmetduran.com
G.YILMAZ - Yazılım Geliştirme Yaşam Döngüsü