YAZILIM MÜHENDİSLİĞİ – SOFTWARE ENGINEERING

YAZ16103
Yazılım Mühendisliğine Giriş

Dr. Öğr. Üyesi Bora ASLAN

Dersin Başarı Kriterleri

- 1 Kısa Sınav %10
- 1 Vize %20
- 1 Ödev %10
- 1 Kısa Sınav %10
- 1 Final %50

- Bağıl Sistem (Çan Eğrisi) Uygulanacak
- Final Sınavından 100 üzerinden en az 40 puan alınmalıdır.

Dersin Kitabı

- Yazılım Mühendisliği - Software Engineering

Yazar:

Ian Sommerville

Çeviri:

Prof. Dr. Yasemin Topaloğlu

İÇİNDEKİLER

KISIM 1

YAZILIM MÜHENDİSLİĞİNE GİRİŞ

1.BÖLÜM

GİRİŞ

Yazılım mühendisliği devletin, toplumun ve ulusal ve uluslararası iş dünyası ve kurumların işleyişi için gereklidir. Modern dünyayı yazılım olmadan yürütemeyiz. Ulusal altyapılar ve servisler bilgisayar tabanlı sistemler tarafından kontrol edilir ve elektrikli ürünlerin çoğu bilgisayar ve kontrol yazılımı içerirler.

Endüstriyel üretim ve dağıtım, iktisadi sistem gibi tamamen bilgisayarlaştırılmıştır. Eğlence, müzik endüstrisi dahil, bilgisayar oyunları, film ve televizyon, yazılım yoğunlukludur. Hâlâ çok sayıda kötü giden yazılım projesi ve “yazılım arıza”ları raporlanmaktadır. Yazılım mühendisliği modern yazılım geliştirme için yetersiz olmakla eleştirilmektedir.

Fakat kanımca, bir çok yazılım arızası iki etmenin sonucudur:

1. Artan sistem karmaşıklığı.

2. Yazılım mühendisliği yöntemlerini kullanmadaki başarısızlık.

Yazılım mühendisliğinin tarihçesi

Yazılım mühendisliği kavramı ilk olarak 1968'de o zaman yazılım krizi (Naur ve Randell 1969) olarak adlandırılan konuları tartışmak için düzenlenen bir konferansta önerilmiştir. Program geliştirmedeki birbirinden farklı yaklaşımların büyük ve karmaşık yazılım sistemlerinin geliştirilmesinde ölçeklenmediği anlaşılmıştı. Yazılım sistemleri güvenilir olmamakta, beklenenden daha fazla maliyetle geç teslim edilmekteydi.

1970'ler ve 1980'ler boyunca, yapısal programlama, bilgi saklama ve nesneye yönelik geliştirme gibi çeşitli yeni yazılım mühendisliği teknikleri ve yöntemleri geliştirilmiştir. Günümüzdeki yazılım mühendisliğinin temeli olan araçlar ve standart gösterimler geliştirilmiştir.

PROFESYONEL YAZILIM GELİŞTİRME

- Birçok kişi program yazar. İşteki insanlar işlerini kolaylaştırmak için hesap çizelgesi (spreadsheet) programları yazarlar; bilim adamları ve mühendisler deneysel verilerini işlemek için program yazarlar; amatörler kendi ilgileri ve merakları için program yazarlar.

- Ancak çoğu yazılım geliştirme, yazılımın iş amaçları için geliştirildiği, diğer cihazlara dahil edilmek üzere veya bilgi sistemleri ve bilgisayar destekli tasarım sistemleri gibi yazılım ürünleri olarak geliştirildiği profesyonel bir etkinliktir.

- Temel farklılıklar, profesyonel yazılımın geliştiricisinden ayrı bir kişi tarafından kullanılmasının amaçlanması ve genellikle yazılımın bireyler yerine takımlar tarafından geliştirilmesidir. Yazılım tüm yaşamı boyunca değiştirilir ve bakımı yapılır.

- Yazılım mühendisliğinin, bireysel programlamadan çok profesyonel yazılım geliştirmeyi desteklemesi amaçlanmıştır. Program spesifikasyonu, tasarımı ve evrimi gibi normal kişisel yazılım geliştirme için önemli olmayan teknikler içerir.

Soru	Cevap
Yazılım nedir?	Bilgisayar programları ve ilişkili belgeleme. Yazılım ürünleri belirli bir müşteri için geliştirilebilir veya genel bir pazar için geliştirilebilir.
İyi yazılımın özellikleri nelerdir?	İyi yazılım kullanıcıya gereken fonksiyonelliği ve performansı sağlamalı ve bakımı yapılabilir, güvenilebilir ve kullanılabilir olmalıdır.
Yazılım mühendisliği nedir?	Yazılım mühendisliği yazılım üretiminin başlangıçtaki ilk fikirden işletim ve bakıma kadar olan tüm yönleriyle ilgilenen bir mühendislik disiplinidir.
Temel yazılım mühendisliği etkinlikleri nelerdir?	Yazılım spesifikasyonu, yazılım geliştirme, yazılım doğrulama ve yazılımın evrimi.
Yazılım mühendisliği ve bilgisayar bilimleri arasındaki fark nedir?	Bilgisayar bilimleri teori ve temeller üzerine odaklanır; yazılım mühendisliği ise kullanışlı yazılım üretim ve tesliminin uygulamalarıyla ilgilenir.
Yazılım mühendisliği ve sistem mühendisliği arasındaki fark nedir?	Sistem mühendisliği donanım, yazılım ve süreç mühendisliği dahil olmak üzere bilgisayar tabanlı sistem geliştirmenin tüm yönleriyle ilgilenir. Yazılım mühendisliği bu daha genel sürecin parçasıdır.
Yazılım mühendisliği ile ilgili esas zorluklar nelerdir?	Artan çeşitlilikle başa çıkma, daha kısa teslim süresi talepleri ve güvenilir yazılım geliştirme.
Yazılım mühendisliğinin maliyetleri nelerdir?	Yazılım maliyetlerinin kabaca %60'ı geliştirme maliyetleridir, %40'ı sınama maliyetleridir. Özel ısmarlama yazılım için, evrim maliyetleri genellikle geliştirme maliyetlerini aşar.
En iyi yazılım mühendisliği teknikleri ve yöntemleri nelerdir?	Tüm yazılım projelerinin profesyonel olarak yönetilmesi ve geliştirilmesi gerekir ama farklı sistem türleri için farklı teknikler uygundur. Örneğin oyunlar her zaman bir dizi prototip kullanılarak geliştirilmeliyken, hayati tehlike arz eden kontrol sistemleri tam ve analiz edilebilir bir spesifikasyonun geliştirilmesini gerektirirler. Her şey için iyi olan yöntemler ve teknikler yoktur.
İnternet yazılım mühendisliğinde ne değişiklik yapmıştır?	İnternet sadece çok büyük, çok dağıtık servis-tabanlı sistemlerin geliştirilmesine yol açmamıştır, aynı zamanda yazılımın ekonomisini değiştiren mobil cihazlar için bir uygulama (*app*) endüstrisi yaratılmasını desteklemiştir.

Şekil 1.1 Yazılım mühendisliği hakkında sık sorulan sorular.

Yazılım Mühendisliği

Yazılım mühendisliği sistem spesifikasyonunun ilk aşamalarından sistemin kullanıma verildikten sonraki bakımına kadar yazılım üretiminin tüm yönleriyle ilgilenen bir mühendislik disiplinidir. Bu tanımda iki anahtar ifade vardır:

a) Mühendislik Disiplini
b) Yazılım Üretiminin Tüm Yönleri

Yazılım mühendisliği iki nedenden dolayı önemlidir:

1. Kişiler ve toplum, gittikçe daha fazla gelişmiş yazılım sistemlerine güvenmektedirler. Emniyetli ve güvenilir sistemleri ekonomik ve hızlı olarak üretebilmeliyiz.

2. Genellikle, uzun vadede, profesyonel yazılım sistemleri için yazılım mühendisliği yöntemlerini ve tekniklerini kullanmak, kişisel bir programlama projesiymiş gibi sadece programlar yazmaktan daha ucuzdur. Yazılım mühendisliği yöntemini kullanmamak sınama, kalite güvence ve uzun dönemli bakım için daha yüksek maliyetlere neden olur.

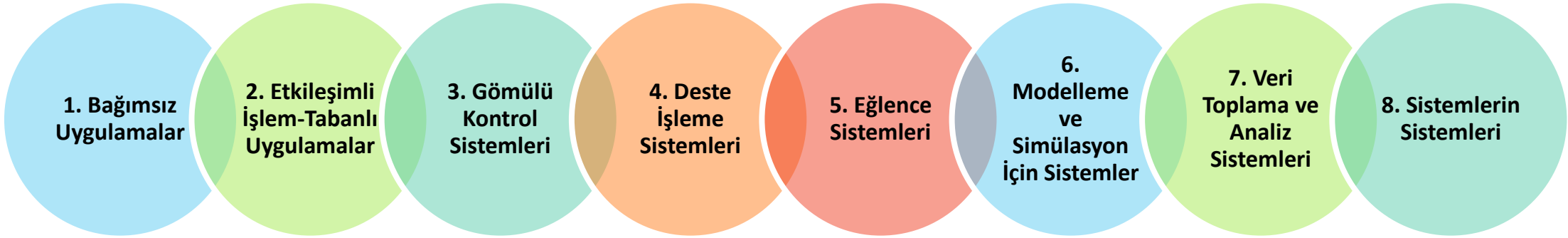
Ürün özelliği	Tanım
Kabul edilebilirlik	Yazılım tasarlandığı kullanıcı tipi tarafından kabul edilebilir olmalıdır. Bunun anlamı, anlaşılabilir, kullanılabilir ve kullandıkları diğer sistemlerle uyumlu olması gerektiğidir.
Güvenilebilirlik ve güvenlik	Yazılım güvenilebilirliği, güvenirlik, güvenlik ve emniyet dahil olmak üzere bir dizi özelliği içerir. Güvenilebilir yazılım bir sistem arızası durumunda fiziksel veya ekonomik bir zarara neden olmamalıdır. Yazılım zararlı kullanıcıların sisteme erişemeyeceği veya zarar veremeyeceği şekilde emniyetli olmalıdır.
Verimlilik	Yazılım bellek ve işlemci döngüleri gibi sistem kaynaklarını gereksiz yere kullanmamalıdır. Bu nedenle verimlilik, çabuk yanıt verme yeteneği, işlem zamanı, kaynak kullanımı vb. içerir.
Bakım kolaylığı	Yazılım müşterilerin değişen gereksinimlerini karşılamak için evrilebileceği şekilde geliştirilmelidir. Bu hayati bir özelliktir çünkü yazılım değişimi, değişen bir iş ortamının kaçınılmaz bir gereksinimidir.

Şekil 1.2 İyi yazılımın gerekli özellikleri.

Yazılım Mühendisliği Çeşitliliği

Yazılım mühendisliği, yazılım üretiminde yazılım müşterilerinin ve üreticilerinin gereksinimleri kadar pratikteki maliyet, zamanlama ve güvenilirlik konularını da dikkate alan sistematik bir yaklaşımdır. Kullanılan yöntemler, araçlar ve teknikler yazılımı geliştiren kuruluşa, yazılımın türüne ve geliştirme sürecine katılan kişilere bağlıdır.

Herhalde hangi yazılım mühendisliği yöntemlerinin ve tekniklerinin en önemli olduğunun belirlenmesinde en önemli faktör geliştirilen uygulamanın türüdür. Çok farklı uygulama türleri vardır:



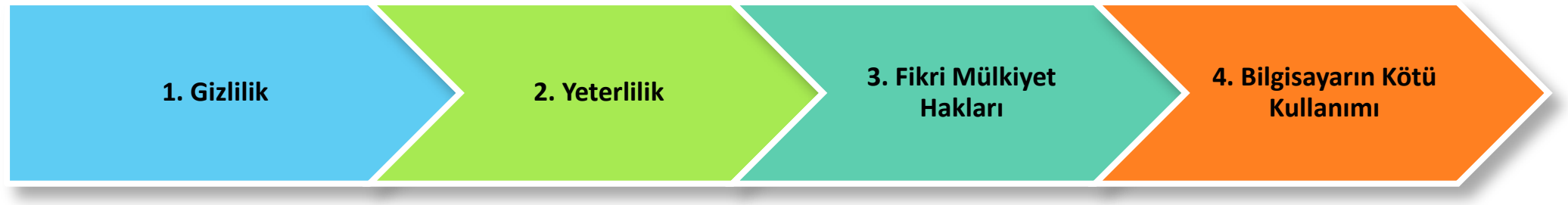
İnternet Yazılım Mühendisliği

- İnternet'in ve World Wide Web'in geliştirilmesinin hepimizin yaşamlarında derin etkileri olmuştur. Başlangıçta, web evrensel olarak erişilebilen bir bilgi deposuydu ve yazılım sistemleri üzerinde küçük bir etkisi olmuştu.
- Bu sistemler yerel bilgisayarda çalıştı ve sadece bir organizasyondan erişilebilirdi. 2000 yılı civarında web değişmeye başladı ve tarayıcılara gittikçe daha fazla fonksiyonellik eklendi.
- Bunun anlamı, özel amaçlı bir kullanıcı arayüzü yerine, bu sistemlerin bir web tarayıcı aracılığıyla erişilebildiği yerde web tabanlı sistemler geliştirilebilirdi.
- Bu durum web üzerinden erişilen ve yenilikçi servisler sunan çok büyük çeşitlilikte yeni sistemlerin geliştirilmesine yol açtı. Bunlar genellikle kullanıcının ekranında görüntülenen reklamlarla desteklendi ve kullanıcılardan doğrudan ödeme içermedi.

YAZILIM MÜHENDİSLİĞİ ETİĞİ

Diğer mühendislik disiplinleri gibi, yazılım mühendisliği, bu alanda çalışan kişilerin özgürlüğünü sınırlayan sosyal ve yasal bir çerçevede yürütülür. Bir yazılım mühendisi olarak, mesleğinizin sadece teknik becerilerin uygulanmasından daha geniş sorumluluklar içerdiğini kabul etmelisiniz.

Yeteneklerinizi ve becerilerinizi dürüst olmayan bir şekilde veya yazılım mühendisliği mesleğine itibarsızlaştıracak şekilde kullanmamalısınız. Ancak kabul edilebilir davranışların standartlarının yasalara bağlı olmadığı fakat daha zayıf mesleki sorumluluk kavramına bağlı olduğu alanlar vardır. Bunların bazıları:



Yazılım Mühendisliği Etik kurallar ve Profesyonel uygulamalar

ACM/IEEE-CS Yazılım Mühendisliği Etiği ve Profesyonel Uygulamalar Ortak Çalışması

ÖNSÖZ

Kuralların kısa sürümü istekleri yüksek bir soyutlama düzeyinde özetler; tam sürümde eklenen ifadeler örnekler verir ve bu isteklerin profesyonel yazılım mühendisleri olarak davranışımızı nasıl değiştirdiğini ayrıntılandırır. Bu istekler olmadan, ayrıntılar yasaya benzeyebilir ve bıktırıcı olabilir; ayrıntılar olmadan istekler kulağa iyi ama boş gelebilir; istekler ve ayrıntılar birlikte uyumlu kurallar oluştururlar.

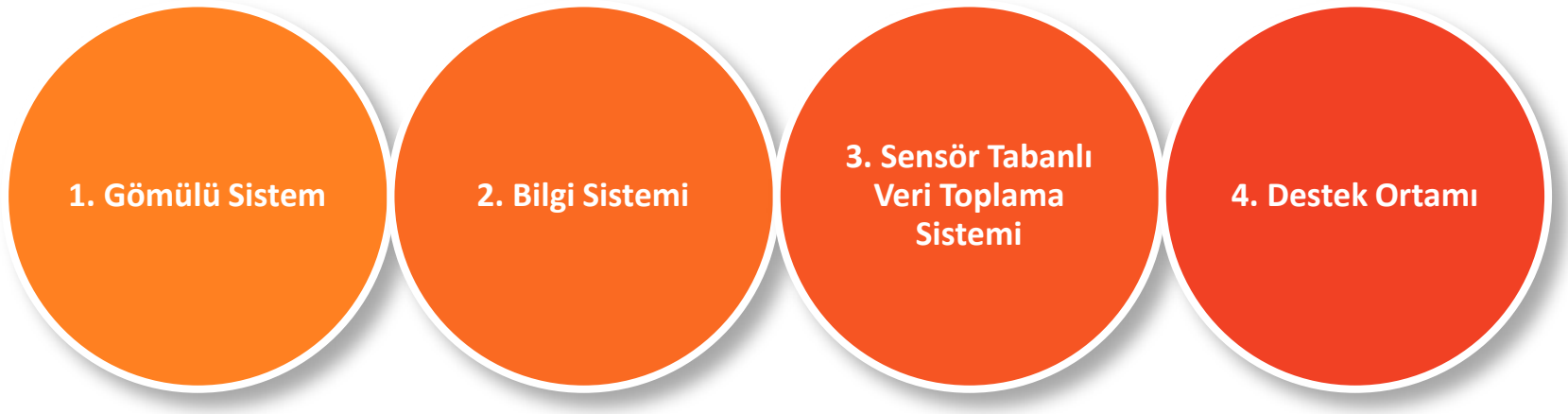
Yazılım mühendisleri yazılımın analiz, spesifikasyon, tasarım, geliştirme, test ve bakımını yararlı ve saygı duyulan bir meslek olarak yapma sorumluluğunu üstlenirler. Bu sorumlulukla uyumlu olarak, toplumun sağlığı, güvenliği ve refahı için yazılım mühendisleri aşağıdaki sekiz ilkeye sadık kalacaklardır:

1. TOPLUM – Yazılım mühendisleri toplumun ilgisiyle tutarlı hareket edeceklerdir.
2. MÜŞTERİ ve İŞVEREN – Yazılım mühendisleri toplumun ilgisiyle tutarlı olarak müşterilerin ve işverenin ilgisine en uygun şekilde davranacaklardır.
3. ÜRÜN – Yazılım mühendisleri ürünlerinin ve ürünlerle ilgili değişikliklerin mümkün olan en yüksek mesleki standartları karşılamasını sağlarlar.
4. DEĞERLENDİRME – Yazılım mühendisleri kendi mesleki değerlendirmelerinde bütünlüğü ve bağımsızlığı sağlayacaklardır.
5. YÖNETİM – Yazılım mühendisliği yöneticileri ve liderleri yazılım geliştirme ve bakımının yönetimine etik bir yaklaşıma uyacak ve onu daha da geliştireceklerdir.
6. MESLEK – Yazılım mühendisleri toplum ilgisiyle tutarlı olarak mesleğin bütünlüğünü ve itibarını geliştireceklerdir.
7. MESLEKTAŞLAR – Yazılım mühendisleri meslektaşlarına karşı adil ve destekleyici olacaklardır.
8. KENDİSİ – Yazılım mühendisleri mesleklerinin uygulamalarını göz önüne alarak yaşam boyu öğrenmeye katılacak ve mesleklerinin uygulanmasında etik bir yaklaşım izleyeceklerdir.

Şekil 1.3 ACM/IEEE Etik Kurallar. (ACM/IEEE-CS Yazılım Mühendisliği Etiği ve Profesyonel Uygulamalar Ortak Çalışması, kısa sürüm. http://www.acm.org/about/se-code) (© 1999 ACM, Inc. ve IEEE, Inc.)

DURUM ÇALIŞMALARI

Yazılım mühendisliği kavramlarını örnekle açıklamak için dört farklı tür sistemden örnekler kullanıyorum. Bu kitaptaki anahtar mesajlardan biri yazılım mühendisliği uygulamasının üretilen sistemin türüne bağlı olması olduğu için kasten tek bir durum çalışması kullanmadım. Bu nedenle emniyet, güvenilirlik, sistem modelleme, yeniden kullanım vb. gibi kavramları tartışırken uygun bir örnek seçiyorum. Durum çalışmaları olarak kullandığım sistem türleri:

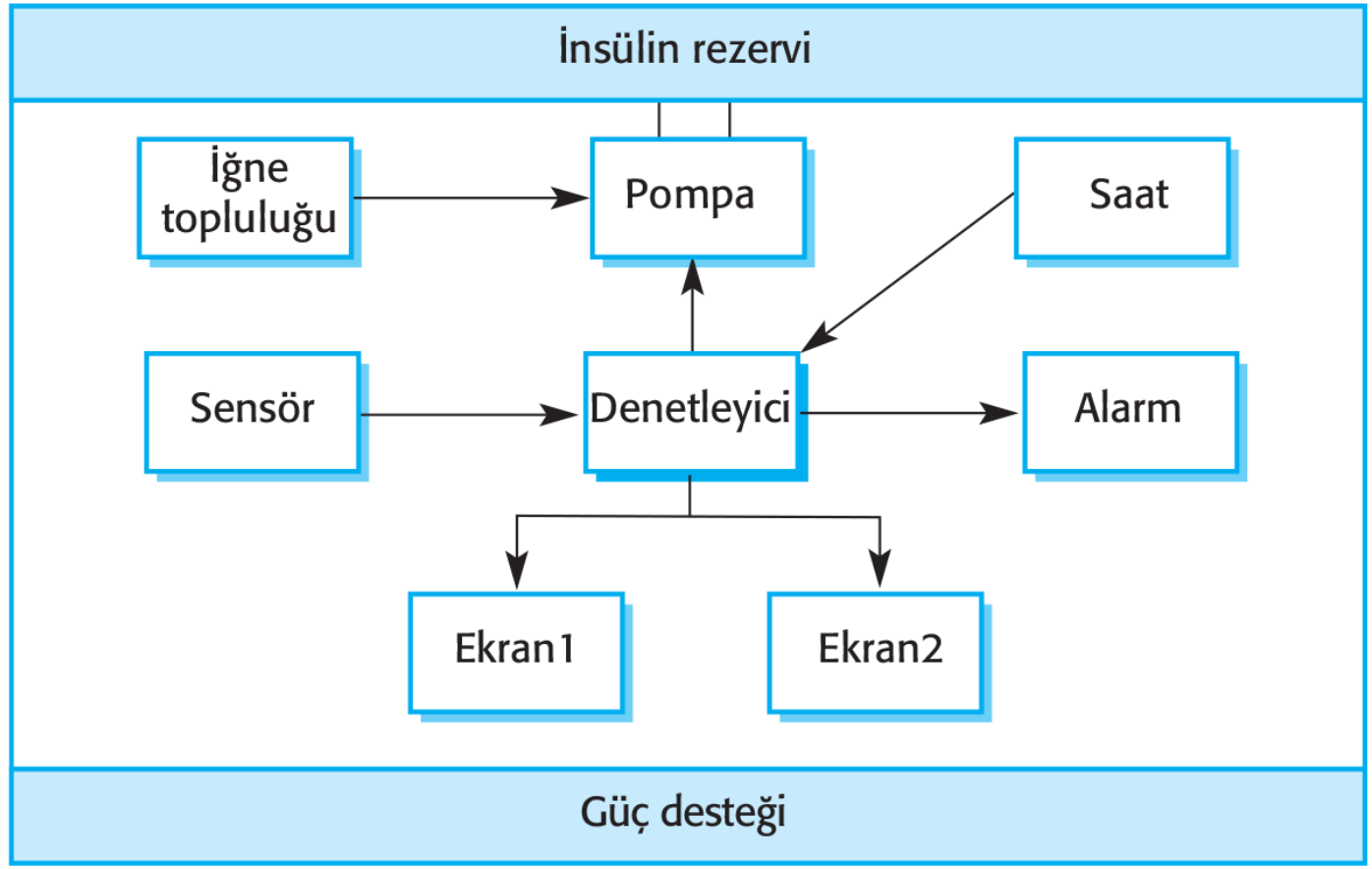


Bir İnsülin Pompası Kontrol Sistemi

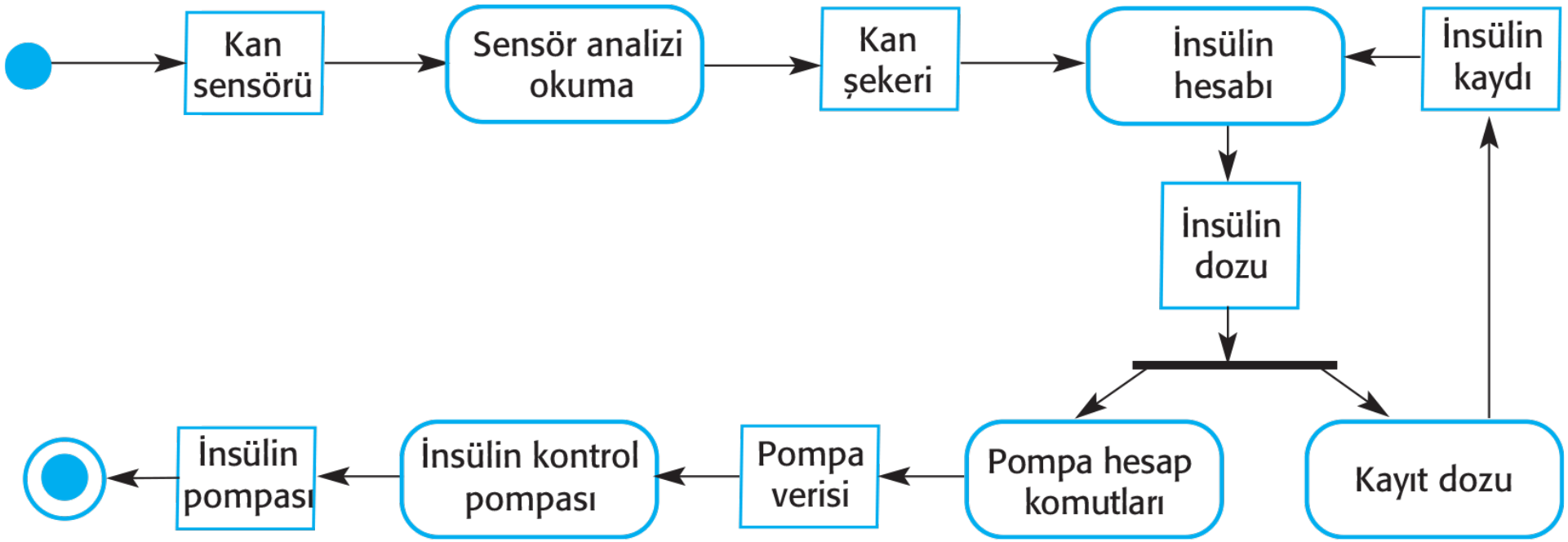
İnsülin pompası pankreasın (bir iç organ) işleyişini taklit eden medikal bir sistemdir. Bu sistemi kontrol eden yazılım bir sensörden bilgi toplayan ve kullanıcıya kontrollü olarak insülin dozu veren bir pompayı yöneten bir gömülü sistemdir. Diyabetli kişiler bu sistemi kullanırlar.

Diyabet, kişinin pankreasının insülin adı verilen hormonu yeterli miktarlarda üretemediği bir durumdur. İnsülin kandaki glikozu (şeker) metabolizma için kullanır.

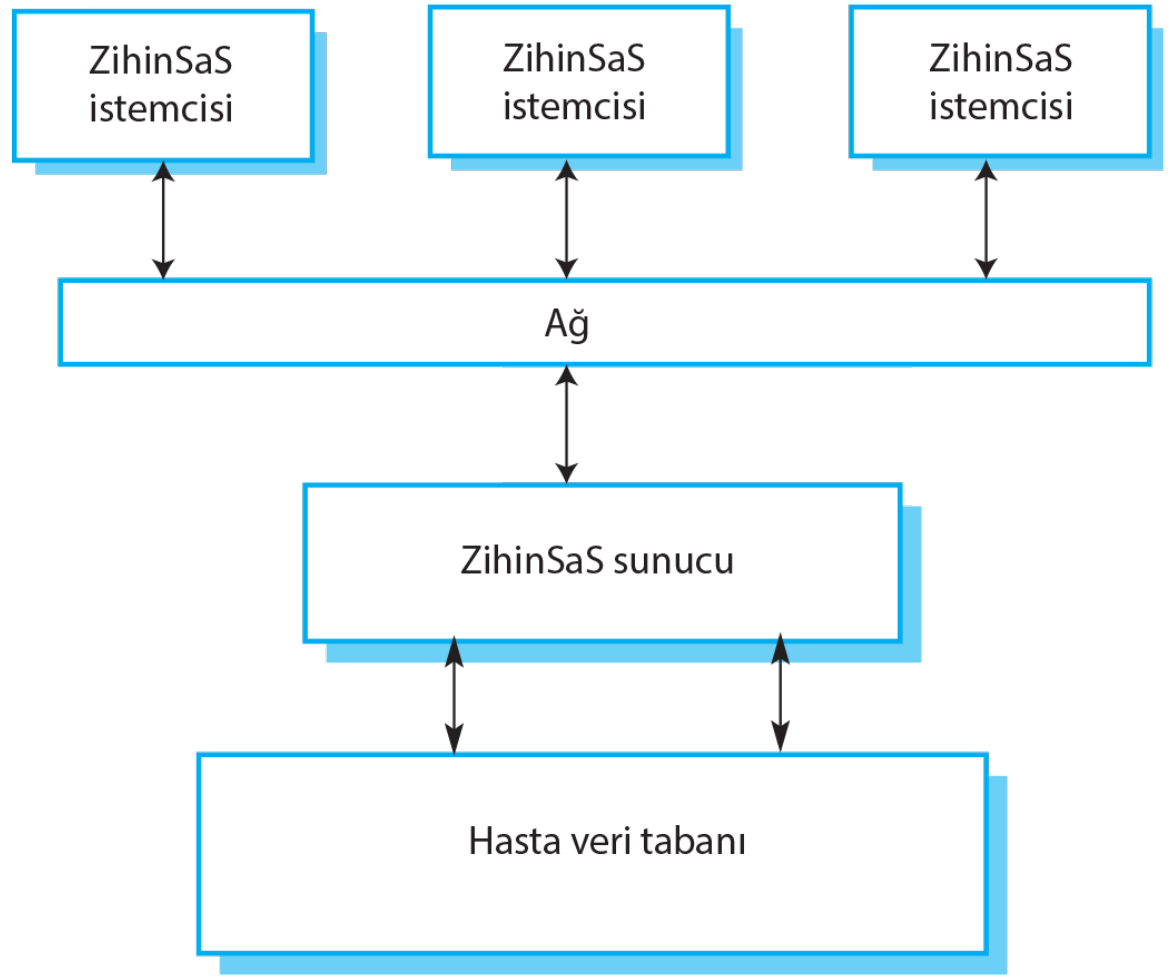
Diyabetin geleneksel tedavisi genetik olarak üretilmiş insülinin düzenli enjeksiyonlarını içerir. Diyabetliler düzenli olarak kan şekeri düzeylerini dışsal bir ölçü kullanarak ölçerler ve enjekte etmeleri gereken insülin dozunu tahmin ederler.



Şekil 1.4 İnsülin pompasının donanım mimarisi



Şekil 1.5 İnsülin pompasının etkinlik diyagramı



Şekil 1.6 ZihinSaS sistem organizasyonu

Ruh Sağlığı İçin Bir Hasta Bilgi Sistemi

- Ruh sağlığını desteklemek için bir hasta bilgi sitemi (ZihinSaS sistemi), ruh sağlığı problemleri olan hastalar ve aldıkları tedaviler hakkında bilgi tutan bir tıbbi bilgi sistemidir.

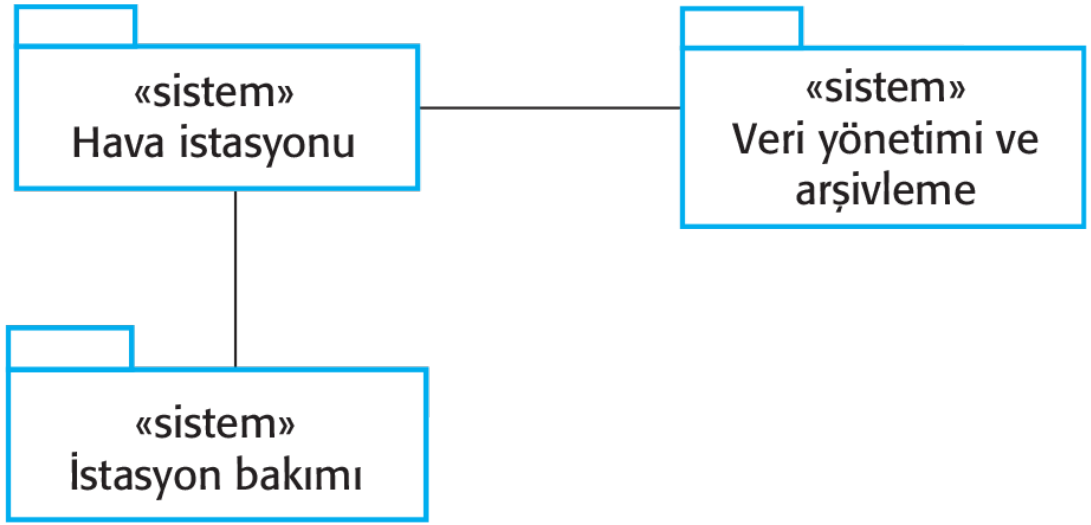
- ZihinSaS sistemi (Şekil 1.6) kliniklerde kullanımı amaçlanmış bir hasta bilgi sistemidir. Merkezileştirilmiş bir hasta bilgi veri tabanı kullanır fakat bir dizüstü bilgisayarda çalışabilecek şekilde tasarlanmıştır, bu nedenle güvenli ağ bağlantısı bulunmayan sitelerden erişilip kullanılabilir.

- Sistem tam bir tıbbi kayıt sistemi değildir ve bu nedenle diğer tıbbi koşullar hakkında bilgi tutmaz. Fakat diğer klinik bilgi sistemleriyle etkileşebilir ve veri değişimi yapabilir. Sistemin iki amacı vardır:

1. Sağlık servis yöneticilerinin yerel ve devlet hedeflerine göre performansının değerlendirebilmesi için yönetim bilgisi oluşturmak.
2. Hastaların tedavisini desteklemek için tıbbi personele zamanında veri sağlamak.

Bir Kır Hava İstasyonu

Kır hava istasyonları, hava istasyonlarından veriler toplayan ve işlenmesi için diğer sistemlere sunulan daha büyük bir hava bilgi sistemin (Şekil 1.7) parçasıdırlar. Şekil 1.7'deki sistemler şunlardır:



Şekil 1.7 Hava istasyonunun çevresi

Okullar İçin Sayısal Bir Öğrenme Ortamı

Birçok öğretmen eğitimi desteklemek için etkileşimli yazılım sistemlerinin kullanılmasının hem artırılmış öğrenci motivasyonu hem de öğrencilerde daha derin bir bilgi düzeyi ve anlayış oluşturabileceğini iddia ederler.

Fakat bilgisayar destekli öğrenme için en iyi strateji konusunda genel bir anlaşma yoktur ve öğretmenler öğrenmeyi desteklemek için pratikte bir çok farklı etkileşimli, web tabanlı araç kullanırlar. Sistem, tüm sistem bileşenlerinin değiştirilebilir servisler olarak düşünüldüğü bir servis tabanlı sistemdir.

Sistemde üç tür servis vardır:

1. Yardımcı servisler,
2. Uygulama servisleri,
3. Konfigürasyon servisleri.

Şekil 1.8 Sayısal bir öğrenme ortamının (eOgren) mimarisi