Erciyes Üniversitesi
Bilgisayar Mühendisliği Bölümü

BZ 313 Yazılım Mühendisliği

13. Üç Popüler Sistem Mimarisi

Üç Popüler Mimari Stil

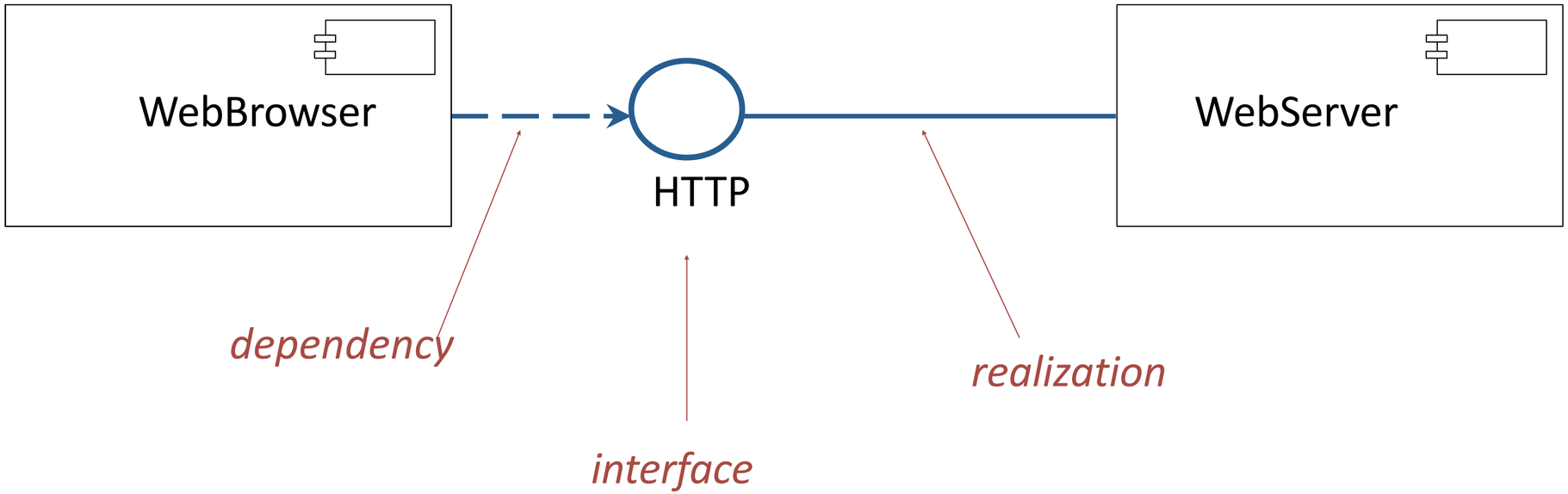
Örnek 1: Üç Katmanlı Mimari (Three Tier Architecture)

Bu mimari, istemci/sunucu modelinin bir uzantısıdır.

Küçük ve orta ölçekli web siteleri için standart mimaridir.

Çok Basit Bir Web Sitesi

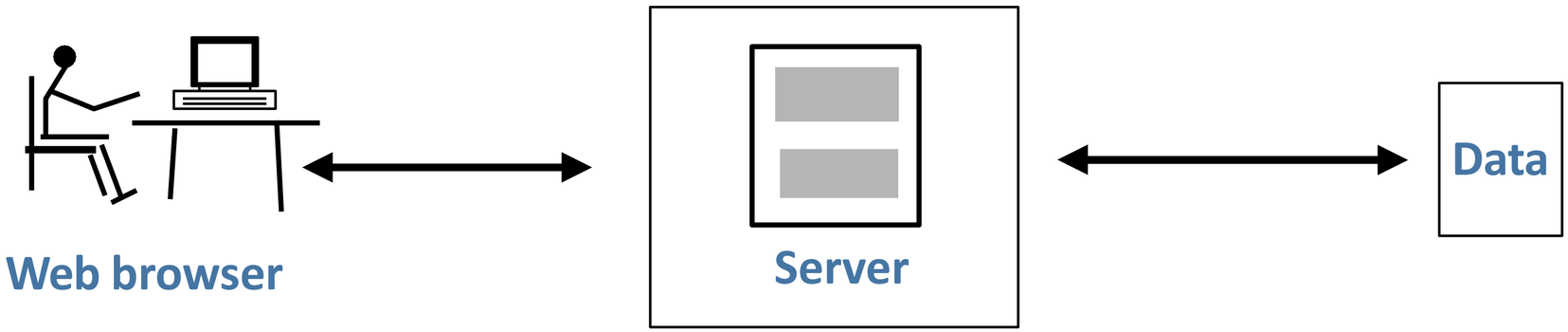
Bu çok temel web sistemi bir istemci/sunucu mimarisine sahiptir.



Veri Deposu (data store) ile Web Sunucusu

Temel istemci/sunucu web sitesi yalnızca sabit HTML sayfaları döndürür.

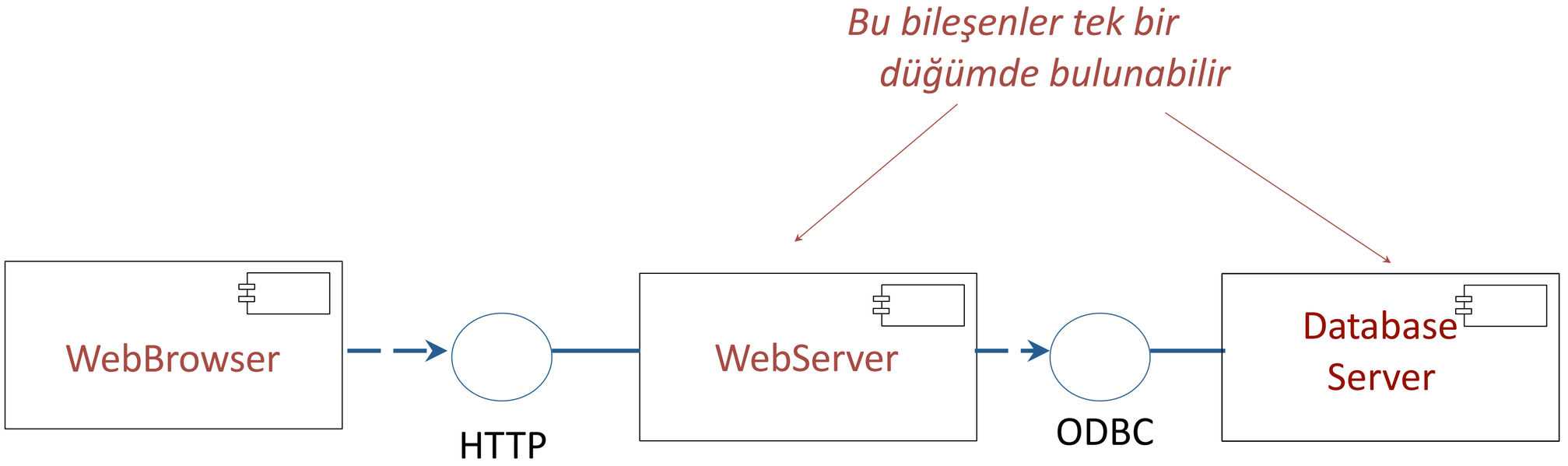
İstemciden gelen isteklere yanıt verebilmesi ve uygun içeriği döndürebilmesi için sunucuyu bir veri deposuna eklemek istiyoruz.



Avantaj

Sunucu tarafı kodu, verilere erişerek, sayfaları yapılandırarak, bilgileri doğrulayarak vb. kullanıcı isteklerine yanıt verebilir.

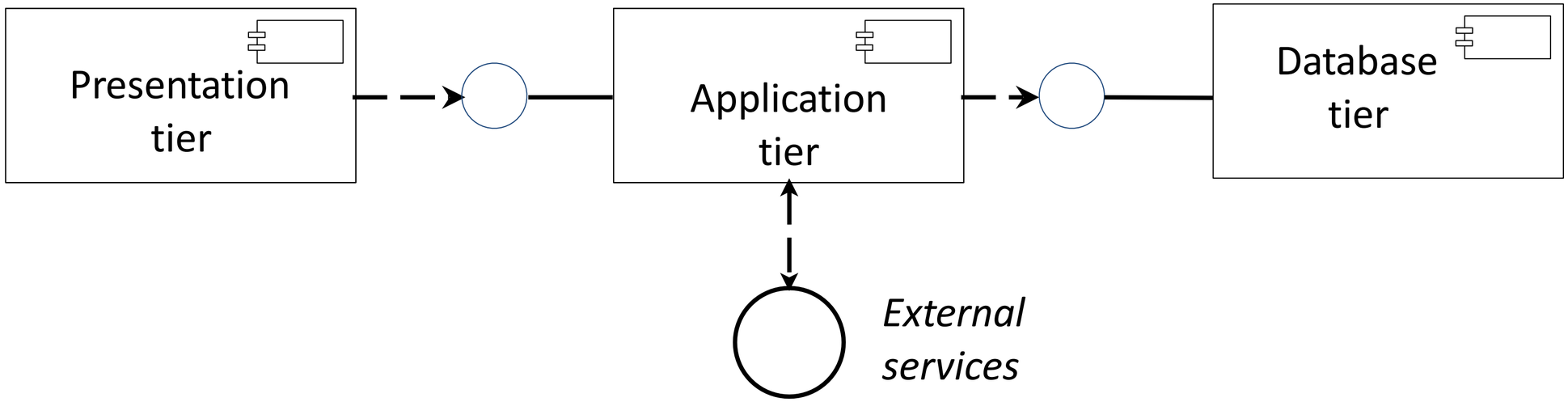
Bileşen (Component) Diyagramı



Bileşenler değiştirilebilir binary elemanlar olduğundan:

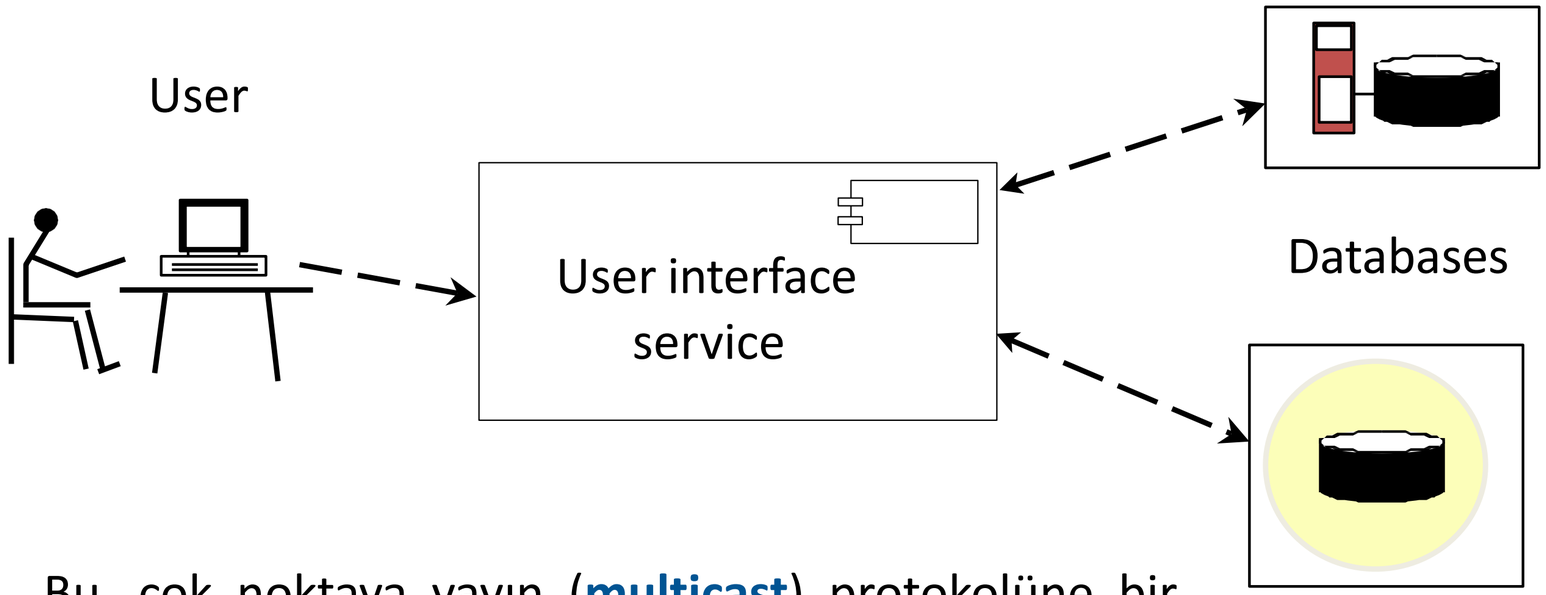
- Herhangi bir web tarayıcısı web sitesine erişebilir.
- Veritabanı sunucusu, aynı arayüzü destekleyen başka bir veritabanıyla değiştirilebilir.

Mimari Stil: Üç Katmanlı Mimari



Katmanların her biri, aynı arayüzleri uygulayan diğer bileşenlerle değiştirilebilir.

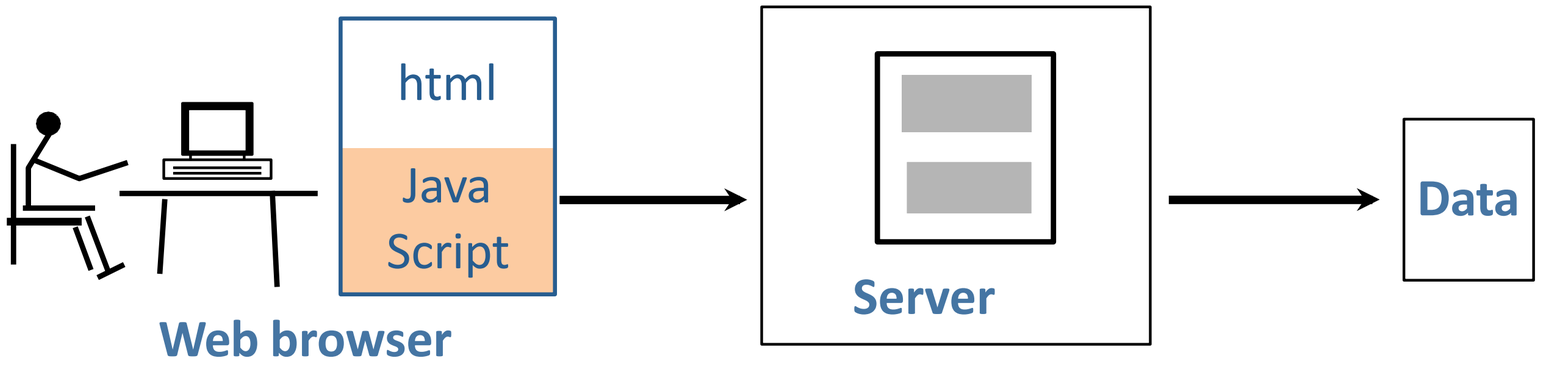
Üç Katmanlı Mimari: Yayın (broadcast) Arama



Bu, çok noktaya yayın (**multicast**) protokolüne bir örnektir.

Birincil zorluk, bir sitede tüm sistemi bozan sorunlardan kaçınmaktır (örneğin, her işlem bir sistemin zaman aşımına uğramasını bekleyemez).

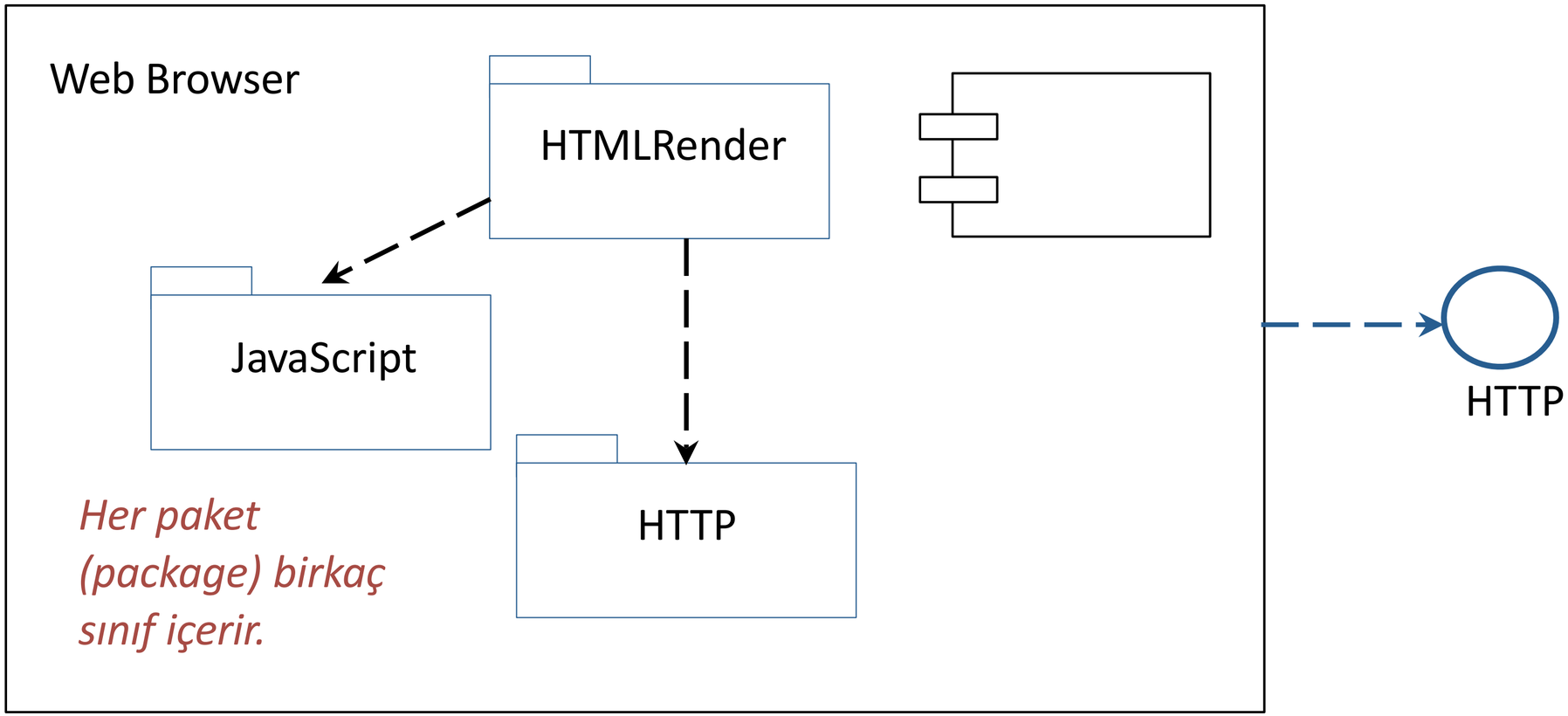
İndirilebilen Yürütülebilir Kodla Web'i Genişletme



JavaScript gibi bir betik dilinde yürütülebilir kod sunucudan indirilebilir.

Bu, üç katmanlı mimari içinde yapılabilir.

JavaScript ile Web Browser



Sunum Katmanı daha karmaşık hale geldi. Yine de aynı arayüzü desteklediğinden değiştirilebilir bir binary bileşen olarak adlandırılabilir.

Örnek 2: Ana Dosya Güncellemesi (Master File Update)

Bu mimari, repository modeline bir alternatiftir.

Veri işleme sistemlerinde çok yaygın olarak kullanılmaktadır.

Master File Update

Organization	Application	Master file	Transactions
Erciyes Üniversitesi	kayıt	öğrenciler	ekle/sil, not kaydet
KASKİ	hizmet faturalandırma	Ev sahipleri	Fatura öde, sayaç oku
Ertok Bank	bankacılık	Müşteri	para yatırma, çekme

Özellikler

- Geri dönüşler (düzeltmeler) dahil olmak üzere birçok karmaşık işlem (transaction) türü.
- Ana dosyanın doğruluğu, kuruluşun başarısı için hayati önem taşır.
- Ana dosyanın toplu olarak güncellenmesi kabul edilebilir (örneğin, bir gecede).
- Tam güncellemeyi beklemeden müşteri sorularını yanıtlayabilmelidir.

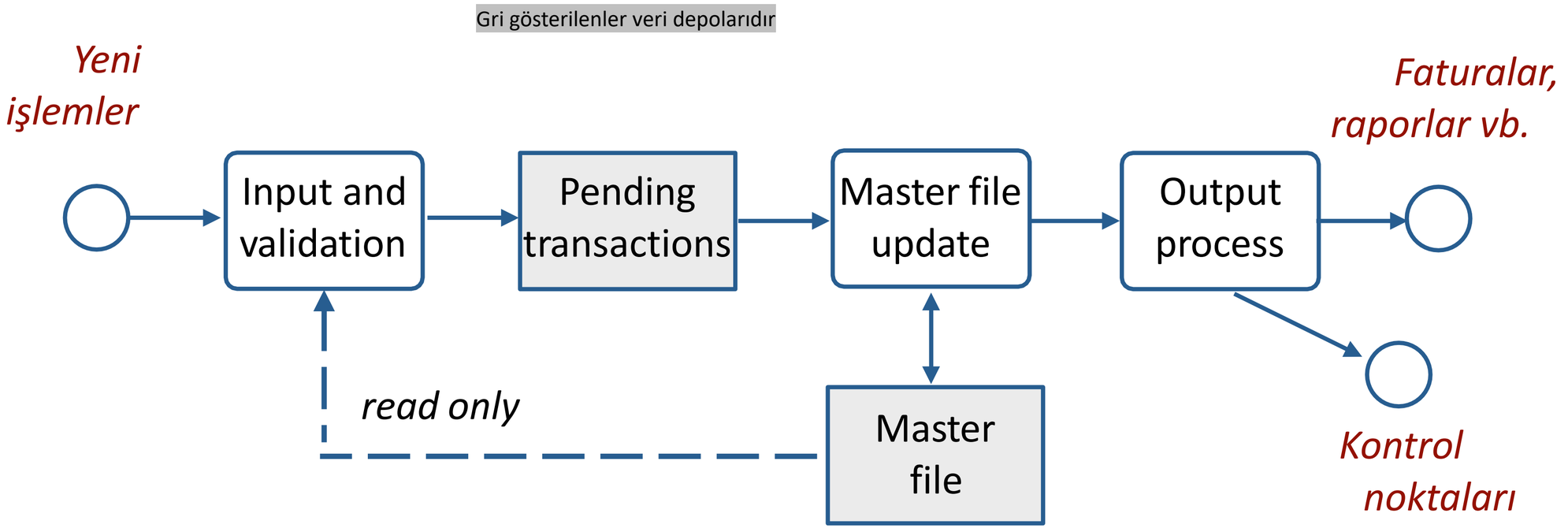
Master File Update: Transactions

Örnek: Elektrik Faturalandırması

Gereksinim analizi sırasında birçok işlem (transaction) türü tanımlanacaktır:

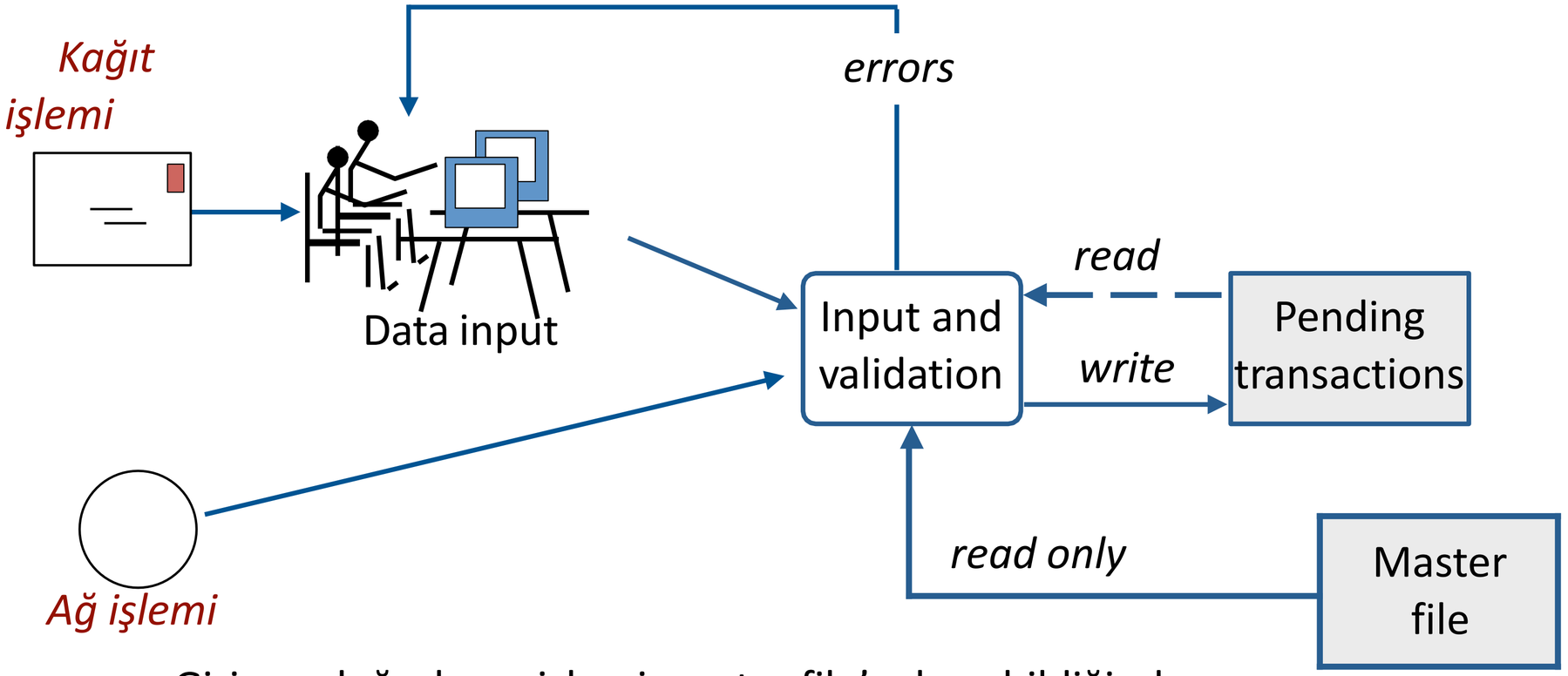
- Hesap oluştur / hesabı kapat
- Sayaç okuma
- Ödeme alındı
- Diğer krediler / borçlar
- Çek karşılandı / karşılıksız çek
- Hesap sorgusu
- Hata düzeltme
- vs., vs., vs.,

Master File ile Toplu İşleme: Veri Akışı Modeli



- Giriş ve doğrulama süreci gün boyunca çalışır. Yeni işlem geldiğinde işler.
- Ana dosya güncelleme programı (master file update) günde bir kez çalışır (genellikle geceleri).
- Çıktı işlemi (output process), ana dosya güncellemesi tamamlandıktan sonra çalıştırılır.

Toplu İşleme: Giriş ve Doğrulama



Giriş ve doğrulama işlemi master file'ı okuyabildiğinden, işlemin master file ile uyumlu olup olmadığını kontrol edebilir, örneğin dosyada müşteri kaydı yoksa tekrar veri girişine gönderilir.

Master File Update ile Toplu İşlemenin Faydaları

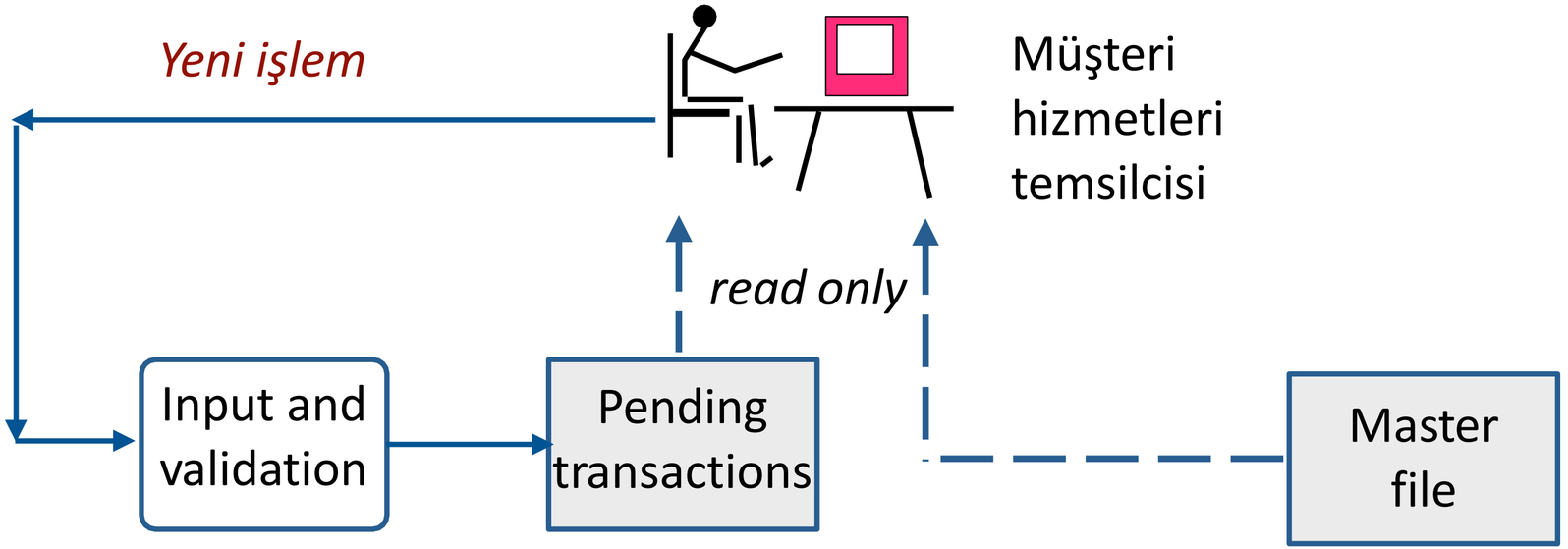
Avantajlar:

- Yedekleme ve kurtarmanın sabit kontrol noktaları vardır.
- Operasyonların daha iyi bir yönetim kontrolü vardır.
- Personel ve donanımın verimli kullanımı sağlanır.
- Hata algılama ve düzeltme basitleştirilmiştir.

Dezavantajlar:

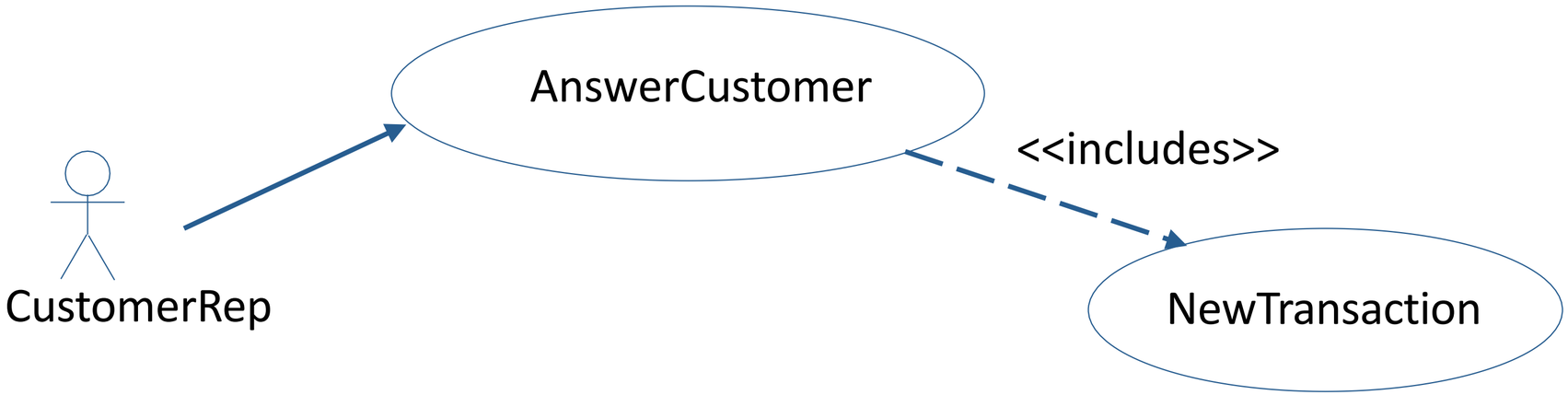
- Master File'daki bilgiler hemen güncellenmez.
- Müşteri sorularını yanıtlamanın iyi bir yolu değildir.

Online Sorgulama



- Müşteri kuruluşu arar ve bir müşteri hizmetleri temsilcisiyle konuşur.
- Temsilci ana dosyayı ve bekleyen dosyayı okuyabilir, ancak bunları değiştiremez.
- *Opsiyonel*. Müşteri, ana dosyanın ve bekleyen dosyanın seçilen bölümlerini müşteri hizmetlerine gitmeden okuyabilir.

Online Sorgulama: Use Case



- Temsilci ana dosyayı ve bekleyen dosyayı okuyabilir, ancak bunlarda değişiklik yapamaz.
- Temsilci ana dosyadaki bilgileri değiştirmek isterse, yeni bir işlem oluşturulur ve giriş ile doğrulama sürecine gönderilir.
- Temsilci, örneğin bir müşteriyle yapılan telefon görüşmesinin notu gibi bir ek açıklama oluşturabilir. Bu yeni bir işlem olarak değerlendirilir.

Üç Popüler Mimari Stil

Örnek 3: Model/View/Controller (MVC)

Bu mimari, karmaşık bir kullanıcı arayüzünü kontrol etmek için kullanılır.

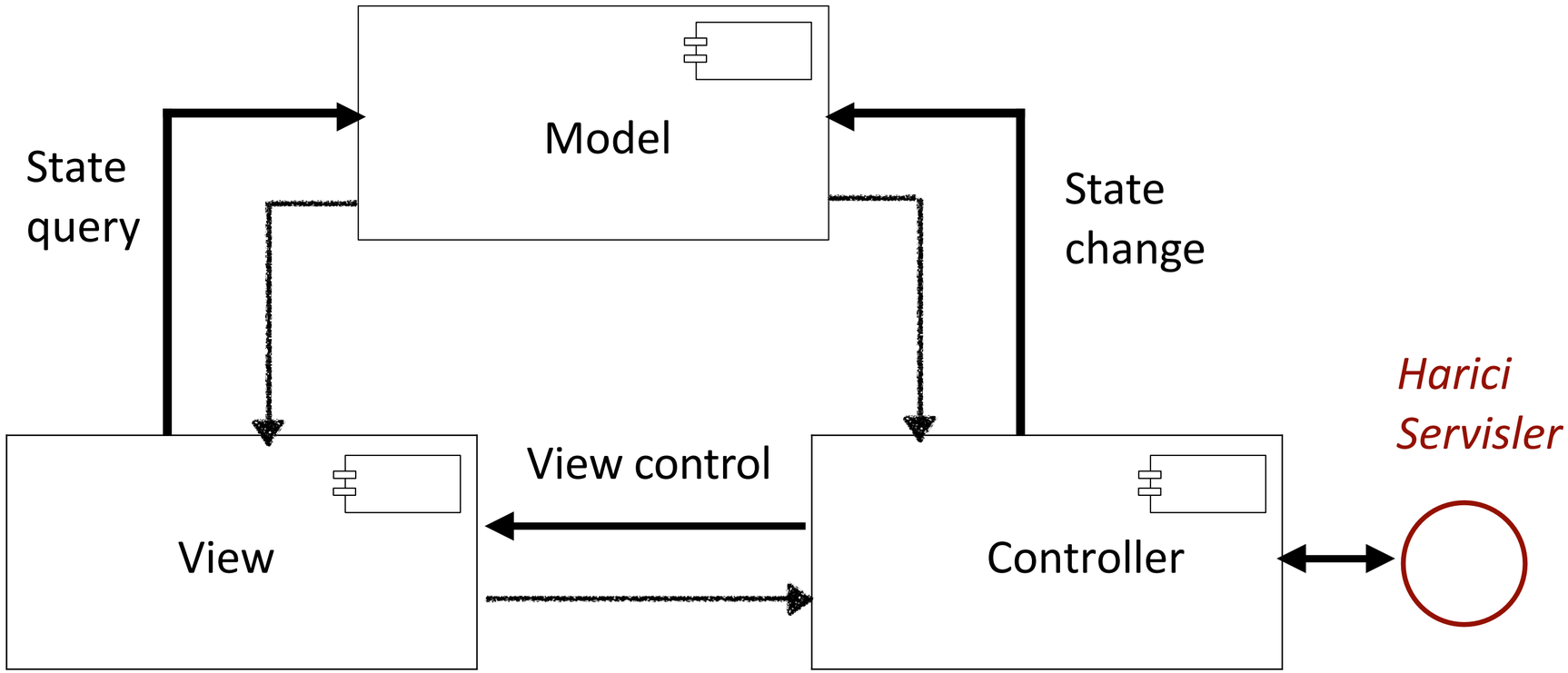
Mobil uygulamalar için standart mimaridir ve robotikte yaygın olarak kullanılır.

Model/View/Controller (MVC)

Model/View/Controller (MVC) tanımı bir değişim halindedir. Terim, bir dizi mimariyi ve tasarımı tanımlamak için kullanılır.

- Bazı varyantlar modelin, view ve controller'ın ayrı bileşenler olduğu sistem mimarileridir.
- Bazıları model, view ve controller adı verilen sınıflarla program tasarımlarıdır.

Sistem Mimarisi Olarak Model/View/Controller



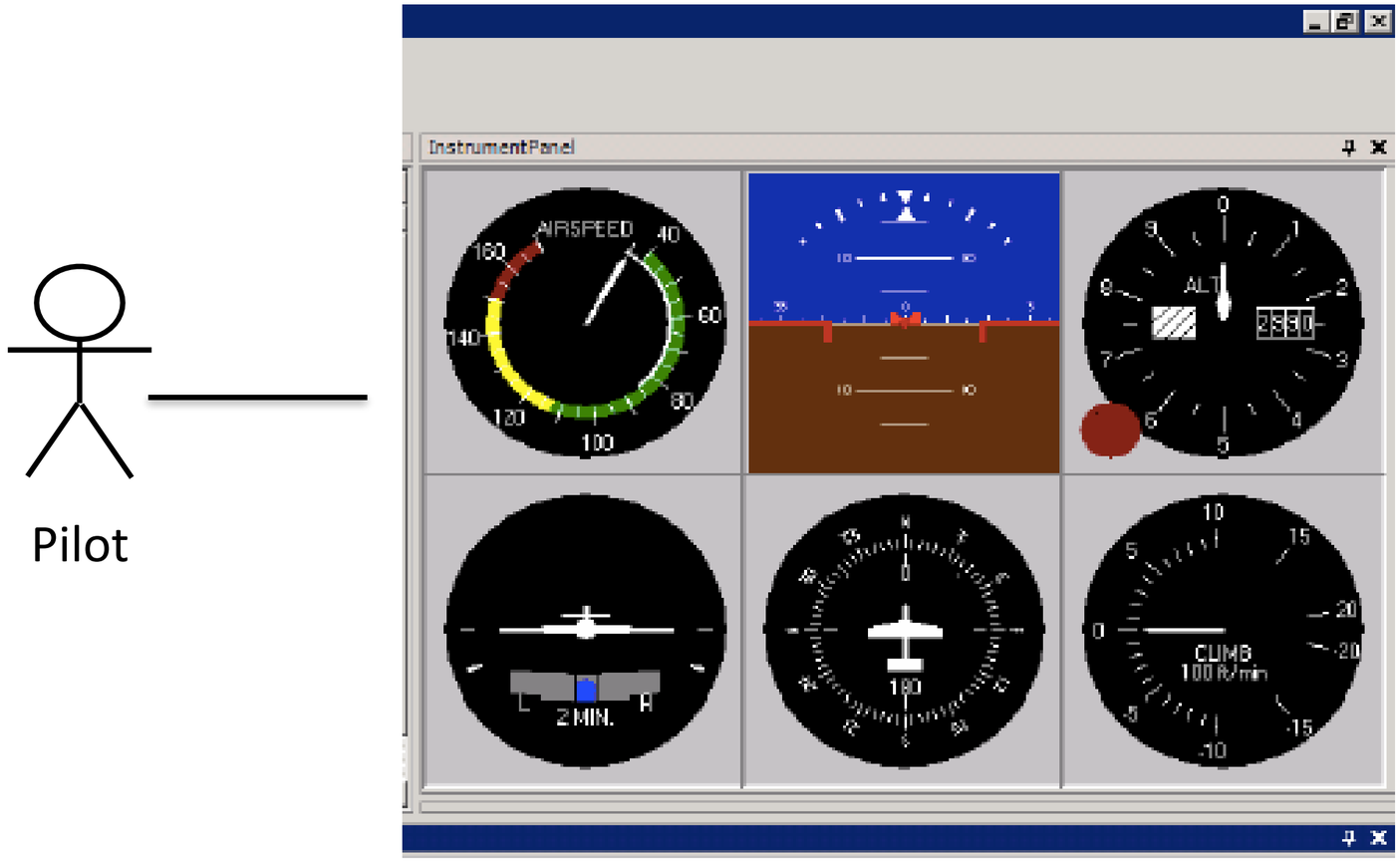
Örnek: bir drone

Drone küçük bir insansız uçaktır.

- Drone, yerdeki bir bilgisayarın başında oturan bir **pilot** tarafından kontrol edilir
- Drone, yer istasyonundan gelen radyo **sinyalleri** ile kontrol edilir (örneğin, gaz kelebeği (throttle), kanatçıklar, kanatlar vb. değişiklikler).,
- Uçaktaki **sensörler** yer istasyonuna radyo sinyalleri gönderir (örn. hava hızı, GPS koordinatları, kontrol yüzeyi ayarları vb.).

Drone: View

Pilot, hafif bir uçağın kokpit kontrollerini temsil eden bir kullanıcı arayüzü ile yerdeki bir bilgisayarda oturur. Bu view katmanıdır.



Drone: Model

Drone için model:

- dronun durumunun, örneğin hız, yakıt gibi bir kaydını tutar.
- gelecekteki durum değişikliklerini modeller, örneğin dönüş hızı, hız değişiklikleri.
- Controller'dan gelen verilere dayalı olarak durumu günceller, örneğin kontrol ayarlarındaki değişiklikler.
- Görünümü, kullanıcı arayüzüne görüntülenecek bilgilerle güncelleştirir.

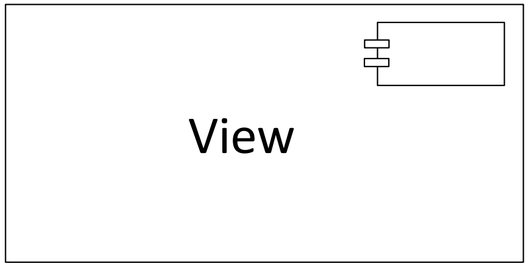
Genel olarak, model farklı bir drone için güncellenecektir ancak view ve controller ayrı bileşenlerdir ve muhtemelen değişmeyecektir.

Drone: Controller

Senaryo: Pilot, kanat ayarlarını 20 dereceye değiştirmek istiyor. Bu, iniş için hızı azaltacaktır.

- View, controller'a bir mesaj gönderir, setFlaps(20).
- Controller uçağa bir mesaj gönderir, setFlaps(20).
- Uçak kanatları ayarlar ve controllar'a bir mesaj gönderir, flapsAreSet(20).
- Controller mesajı modele iletir, flapsAreSet(20).
- Model, durum bilgilerini günceller ve durma hızını yeniden hesaplar.
- Model, view'a, flapsAreSet(20) ve ayrıca durma hızında (stall) tahmin edilen değişikliği bildirir.
- View, pilotun konsolunda değiştirilen ayarları görüntüler.

View

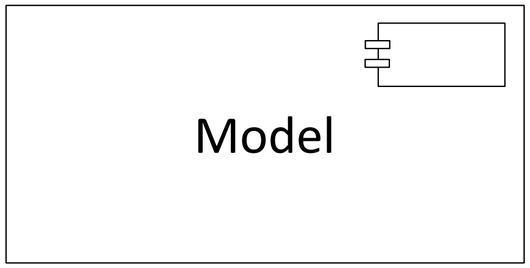


View, arayüz durumunu kullanıcıya sunar. Durumu değiştiren olayları bildiren modele abone olur.

- Kullanıcı arayüzü için modelden veri işler
- Metin alanları vb. gibi özellikler için editörler sağlar.
- modelden güncelleştirmeleri alır
- Controllar'a kullanıcı girdilerini gönderir

Belirli bir model, alternatif view seçimini destekleyebilir.

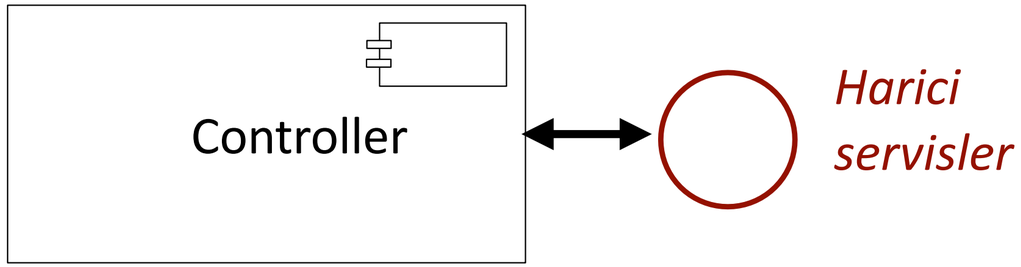
Model



Model, uygulamanın durumunu kaydeder ve aboneleri bilgilendirir. Controller veya view'a bağlı değildir.

- Uygulamanın durumunu uygun veri yapılarında veya veri tabanlarında saklar
- Durum bilgilerini değiştirme talimatlarına yanıt verir
- Durumu değiştiren olayları abonelere bildirir
- bilgilerin doğrulanmasından sorumlu olabilir

Controller



Controller, uygulama içinde kullanıcı girişini ve navigasyonu yöneten sistemin bir parçasıdır.

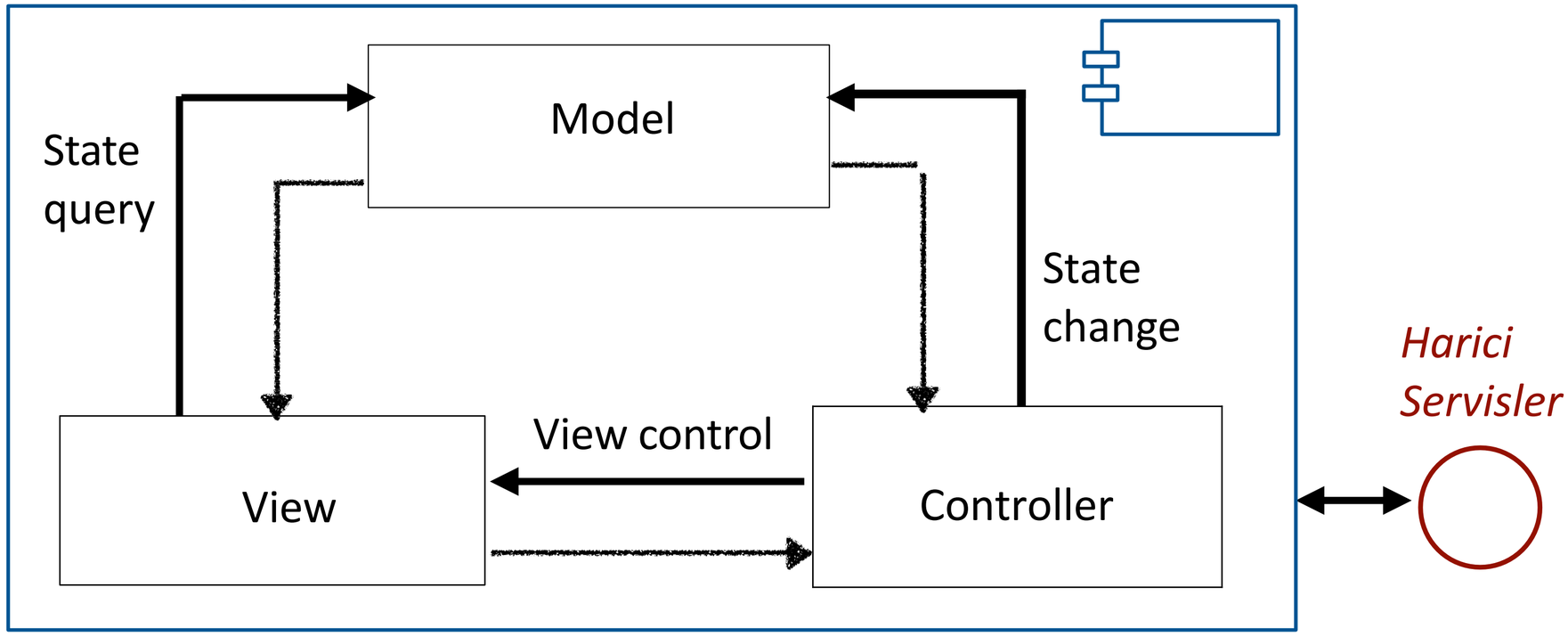
- Uygulama davranışını tanımlar
- Kullanıcı eylemlerini modelin durumundaki değişikliklerle eşleştirir
- API'ler aracılığıyla harici hizmetlerle etkileşime girer
- bilgilerin doğrulanmasından sorumlu olabilir

Farklı frameworkler controller'i farklı şekillerde ele alır. Özellikle, sorumlulukları model ve controller arasında bölmenin birkaç yolu vardır, örneğin veri doğrulama nerede yapılır veya harici API'lar sisteme nerede bağlanır?

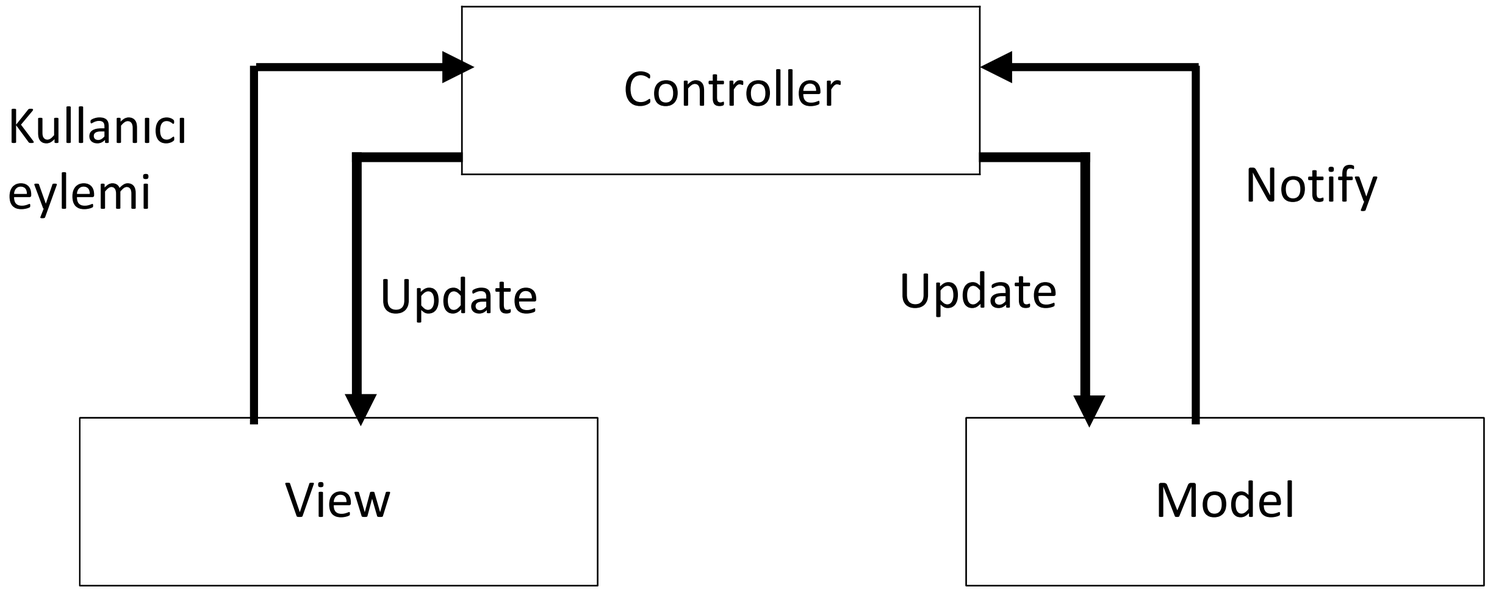
Program Tasarımı olarak Model/View/Controller

Mobil uygulamalar için MVC tek bir bileşendir. Model, view ve controller birer sınıflardır.

Programlar genellikle her biri bir API'ye (ör. konum, doğrulama) sahip bulut tabanlı harici hizmetler kullanır. Bunlar genellikle controller tarafından yönetilir.



Apple'ın Model/View/Controller Versiyonu



MVC bir program tasarımıdır (bir sistem mimarisi değildir).

- Model, view ve controller birer sınıflardır (bileşenler değil).
- Tüm mesajlar controller'dan geçer.
- Çok ekranlı bir uygulamada aynı modeli paylaşan birkaç view ve controller bulunur.

Mimari Stiller ve Tasarım Desenleri

Ortak mimari tarzların birçok çeşidi vardır. Bu kursta anlatılandan farklı bir çeşidiyle karşılaşırsanız şaşırmayın.

Bu derste mimari stiller ve tasarım desenleri arasında dikkatli bir ayrım yapacağız.

Mimari stiller sistem tasarımının bir parçasıdır. Alt sistemler, bileşenler ve dağıtım (deployment) açısından tanımlanırlar.

Tasarım desenleri program tasarımının bir parçasıdır. Sınıflar açısından tanımlanırlar.

Erciyes Üniversitesi
Bilgisayar Mühendisliği Bölümü

BZ 313 Yazılım Mühendisliği

13. Üç Popüler Sistem Mimarisi

Ders Sonu