

# ADD 3.o: Etmenleri Yeniden Düşünmek ve Tasarım Sürecinde Kararlar\*

\*Rick Kazman Humberto Cervantes'in SATURN 2015 konuşmasından çevrilmiştir.

# İçerik

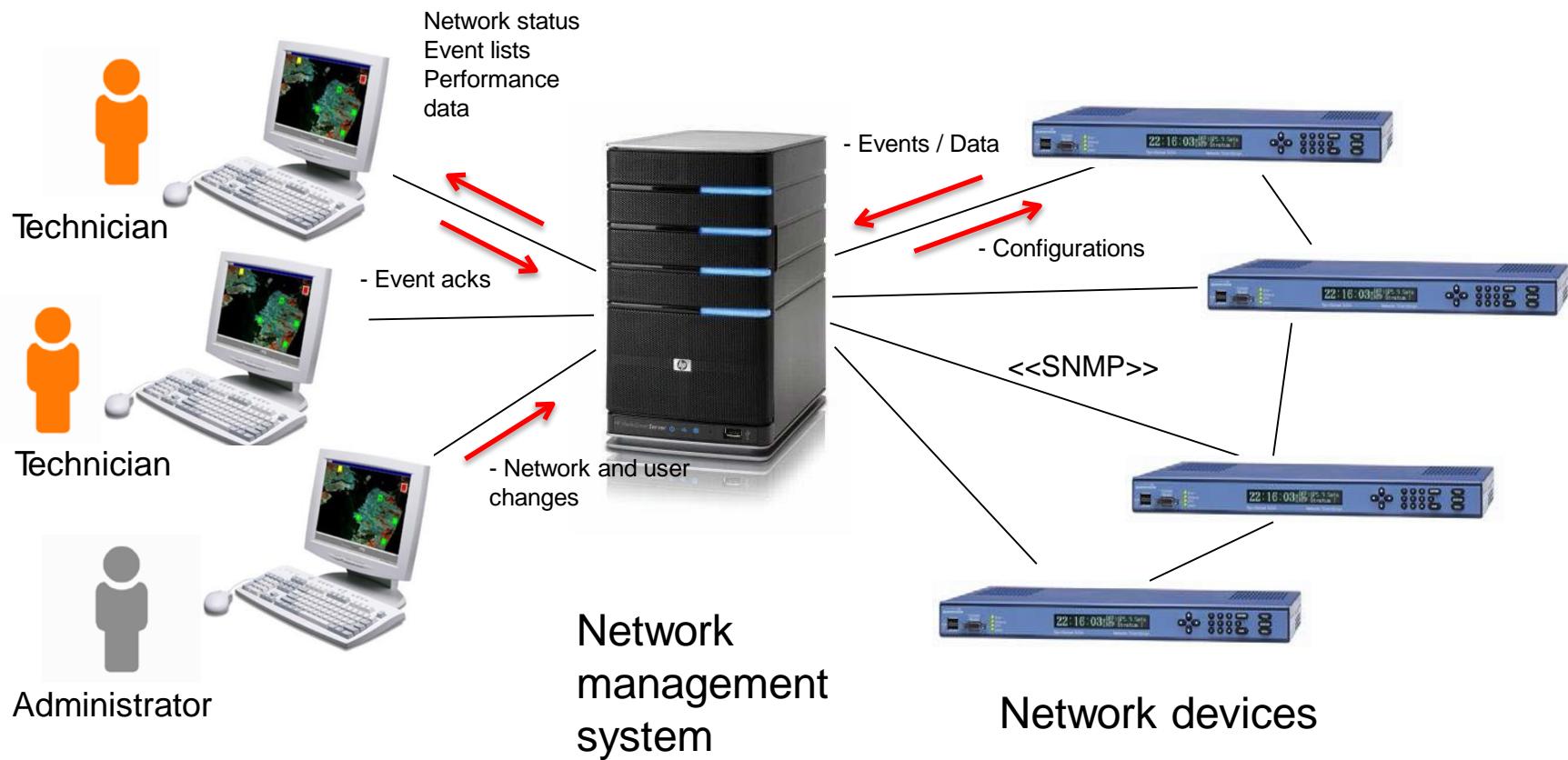
- Öğrenme Amaçları
- Mimari Tasarım ve Etmen Türleri
- Nitelik Odaklı Tasarım Yöntemi
- Tasarım kararları
- Örnek
- Sonuç

# Öğrenme Amaçları

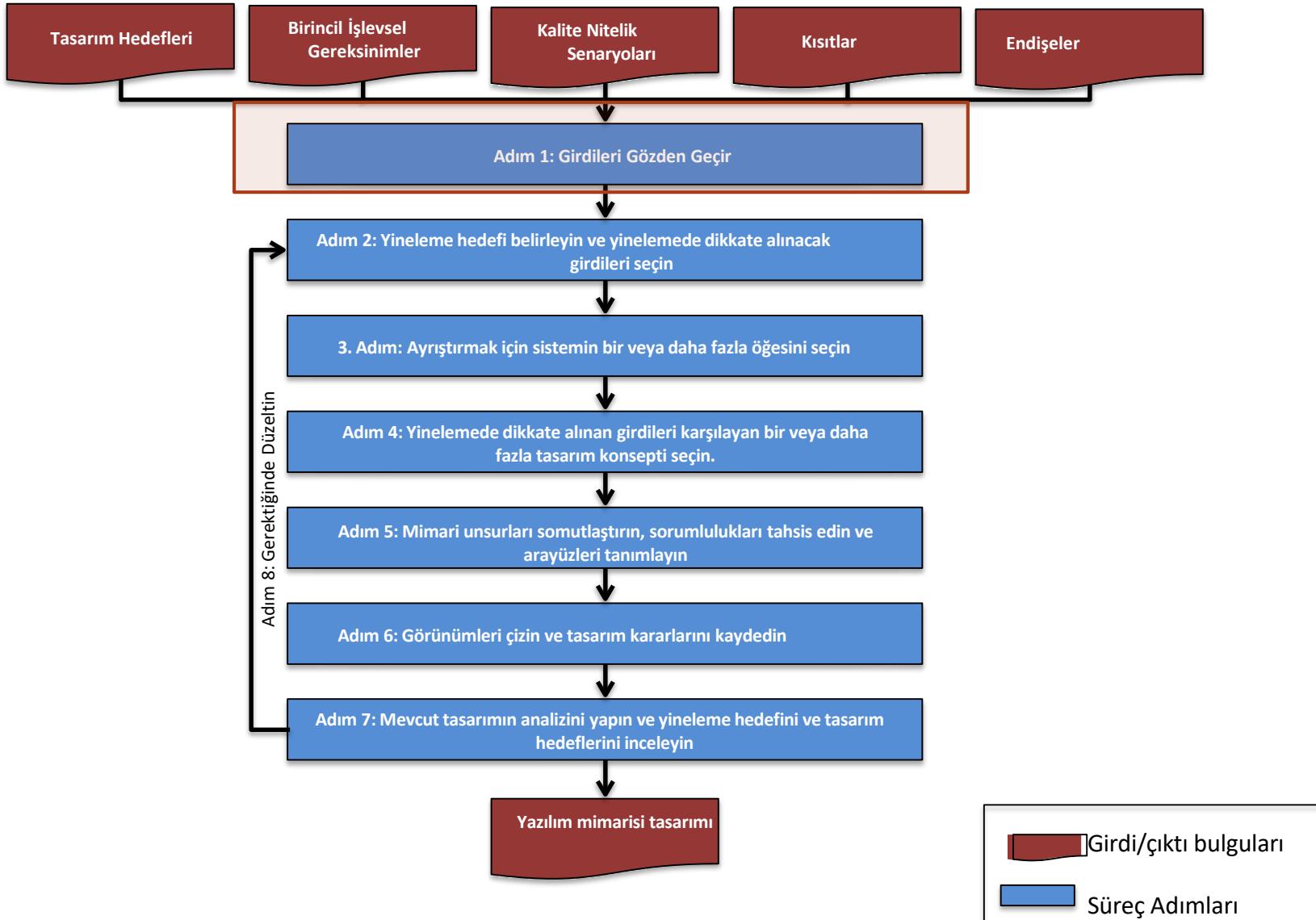
- Farklı mimari etmenler nelerdir?
- Tasarım kavramları ve bunların seçimi ile ilgili kararlar nelerdir
- ADD nedir ve bir mimari bu yöntem kullanılarak yinelemeli olarak nasıl tasarlanyır?

# Örnek

- Ağ Yönetim Sistemi “Marketecture” Diyagramı



# ADD 3.0

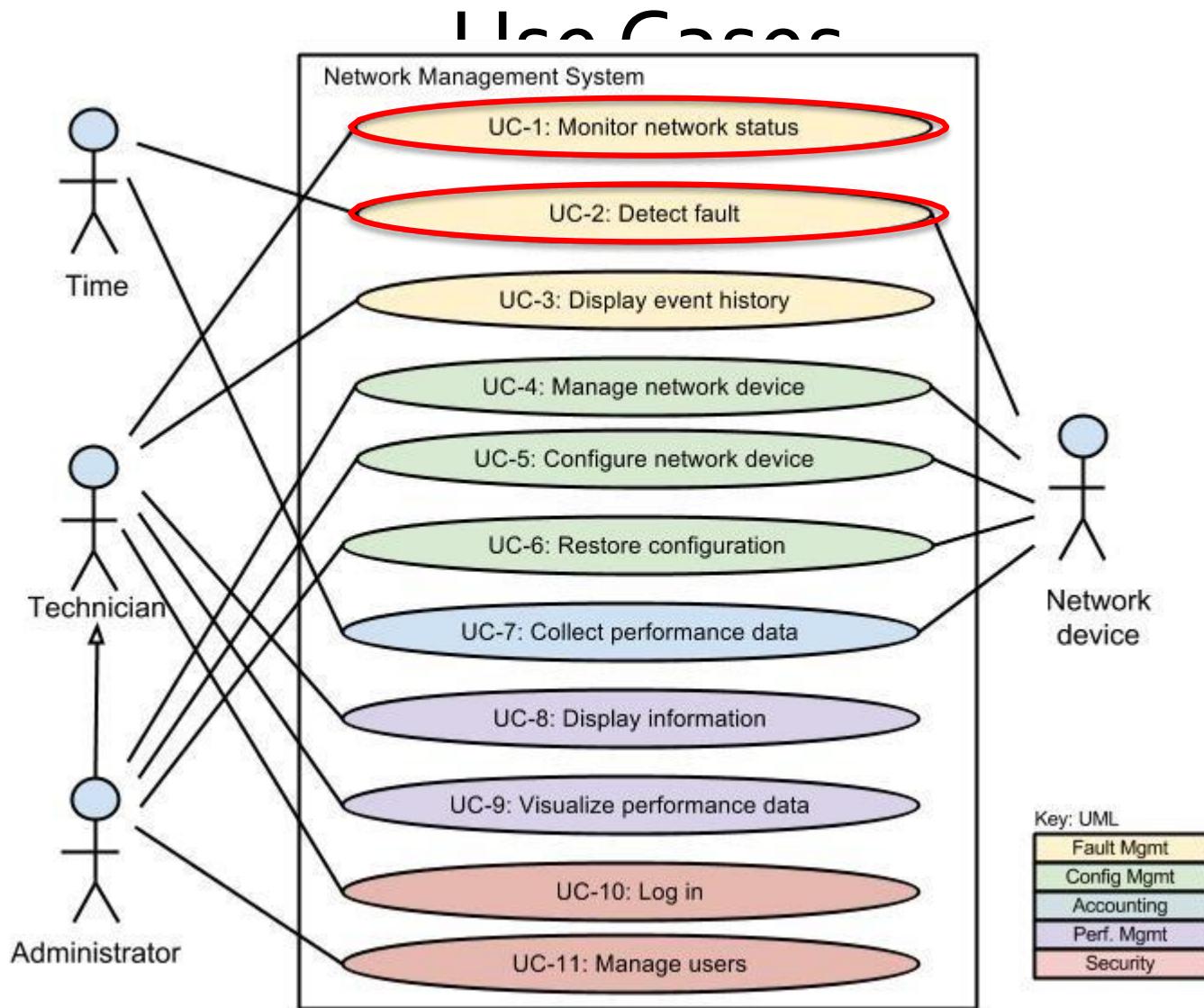


# Örnek

- **Sistem Türü:**
  - Olgun etki alanında sıfırdan proje
- **Amaç:**
  - Sistemin bir artımının inşası için hazırlık aşamasında tasarım
- **Endişeler**
  - Sistemi yapılandırin
  - Kod tabanının organizasyonu
  - Giriş doğrulama
  - İstisna yönetimi

# Sizin önerileriniz????

- Ağ Yönetim Sisteminde aktörler kimler olabilir?
  - ...
  - ...
  - ...
- Ağ Yönetim Sisteminde kullanım durumları ne olabilir?
  - ...
  - ...
  - ...
  - ....
  - ...
  - ...



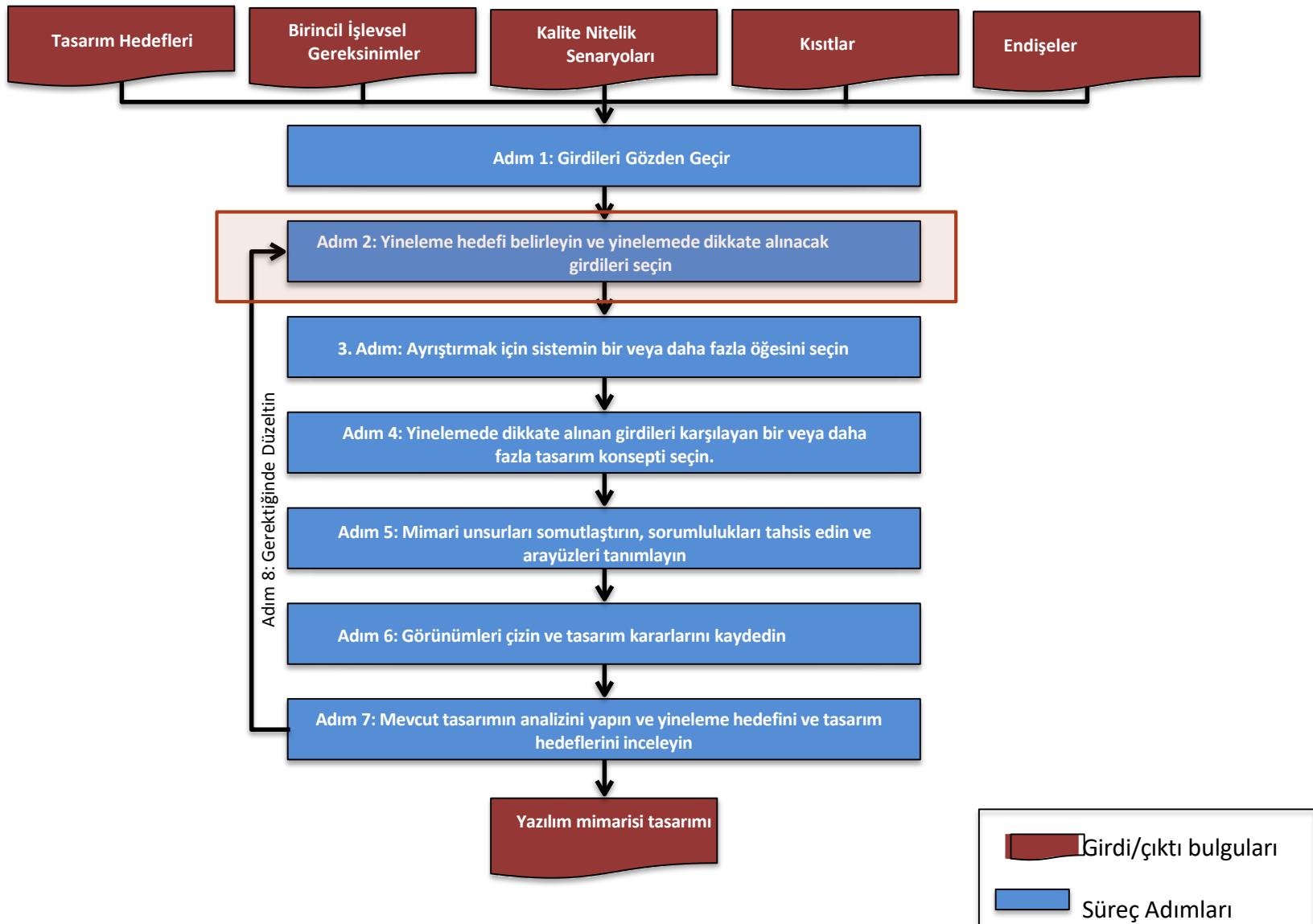
# Kalite Gereksinimleri Senaryoları

ID	Kalite Özelliği	Senaryo	İlişkili kullanım durumu	Öncelik
KG-1	Performans	Birçok ağ cihazı, en yüksek yükte yönetim sistemine tuzaklar gönderir. Tuzakların% 100'ü başarıyla işlendi ve saklandı.	Ağ cihazı hatasını tespit et (UC-2)	H, H
KG-2	Değiştirilebilirlik	Güncellemenin bir parçası olarak sisteme yeni bir ağ cihazı yönetimi protokolü eklenir. Protokol, sistemin temel bileşenlerinde herhangi bir değişiklik yapılmadan başarıyla eklenir.	Ağ cihazını yapılandırın (UC-5)	M, M
KG-3	Bulunurluk	İşletim sırasında yönetim sisteminde bir arıza meydana gelir. Yönetim sistemi 30 saniyeden daha kısa sürede çalışmaya devam eder.	Tamamı	H, H
KG-4	Performans	Yönetim sistemi, en yoğun yük sırasında bir ağ cihazından performans verilerini toplar. Yönetim sistemi, veri kaybını önlemek için tüm performans verilerini 5 dakika içinde toplar.	Performans verilerini toplayın (UC-7)	H, H
KG-5	Performans, Kullanılabilirlik	Bir kullanıcı, normal çalışma sırasında belirli bir ağ cihazının olay geçmişini görüntüler. Son 24 saatteki olayların listesi 1 saniye içinde görüntülenir.	Olay geçişini görüntüle (UC-3)	H, M
KG-6	Güvenlik	Bir kullanıcı, normal çalışma sırasında sistemde bir değişiklik yapar. Ameliyatı kimin ve ne zaman yaptığı% 100 bilmek mümkündür.	Tamamı	H, M

# Kısıtlar

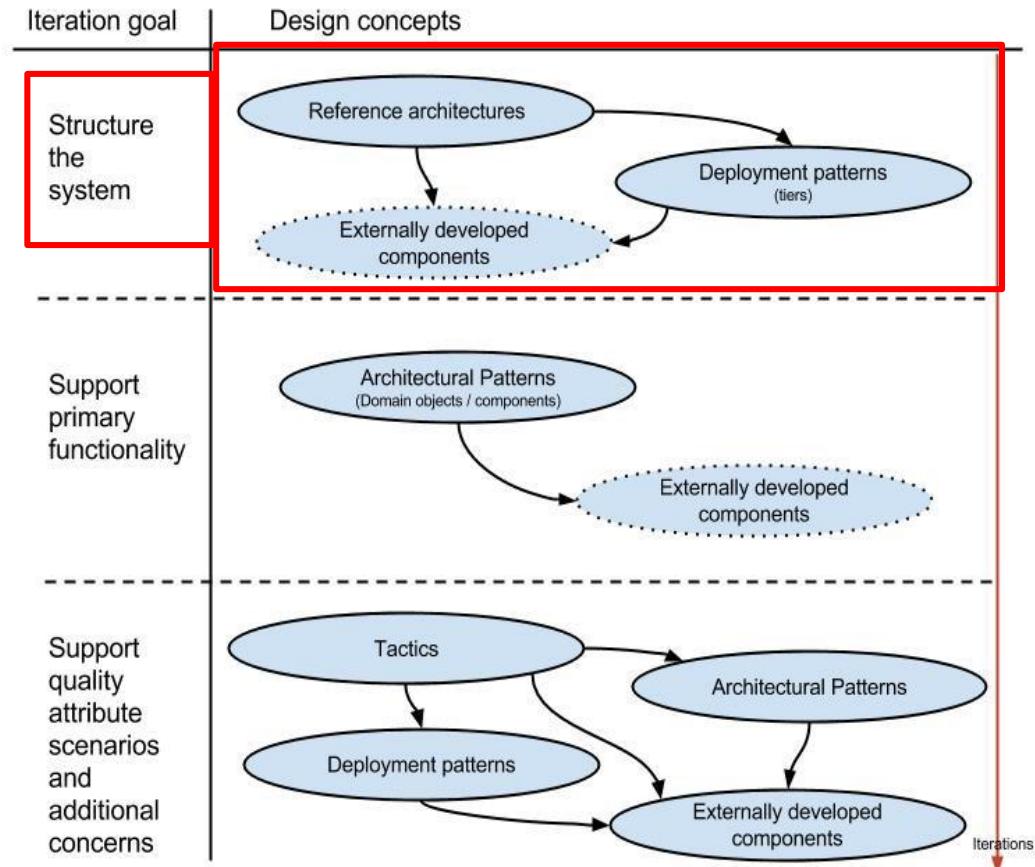
ID	Kısıt
CON-1	En az 50 eşzamanlı kullanıcı desteklenmelidir.
CON-2	Kullanıcı iş istasyonları aşağıdaki işletim sistemlerini kullanır: Windows, OSX ve Linux.
CON-3	Mevcut bir ilişkisel veritabanı sunucusu kullanılmalıdır.
CON-4	Kullanıcıların iş istasyonları ve sunucu arasındaki ağ bağlantısı güvenilir değil
CON-5	Mobil istemciler için ileride destek gereklili.
CON-6	En az 600 zaman sunucusu desteklenmelidir (İlk dağıtım 100 zaman sunucusu, 5 yıl boyunca yılda 100 daha fazla)
CON-7	Sunucu her seferinde ortalama 10 tuzak / saat gönderir.
CON-8	Daha yüksek aralıklar, zaman sunucularının verileri atmasına neden olduğundan, performans verilerinin 5 dakikadan fazla olmayan aralıklarla toplanması gereklidir.
CON-9	Son 30 gündeki olaylar saklanmalıdır
CON-10	Geliştirme ekibi Java teknolojilerine aşınadır

# ADD 3.0

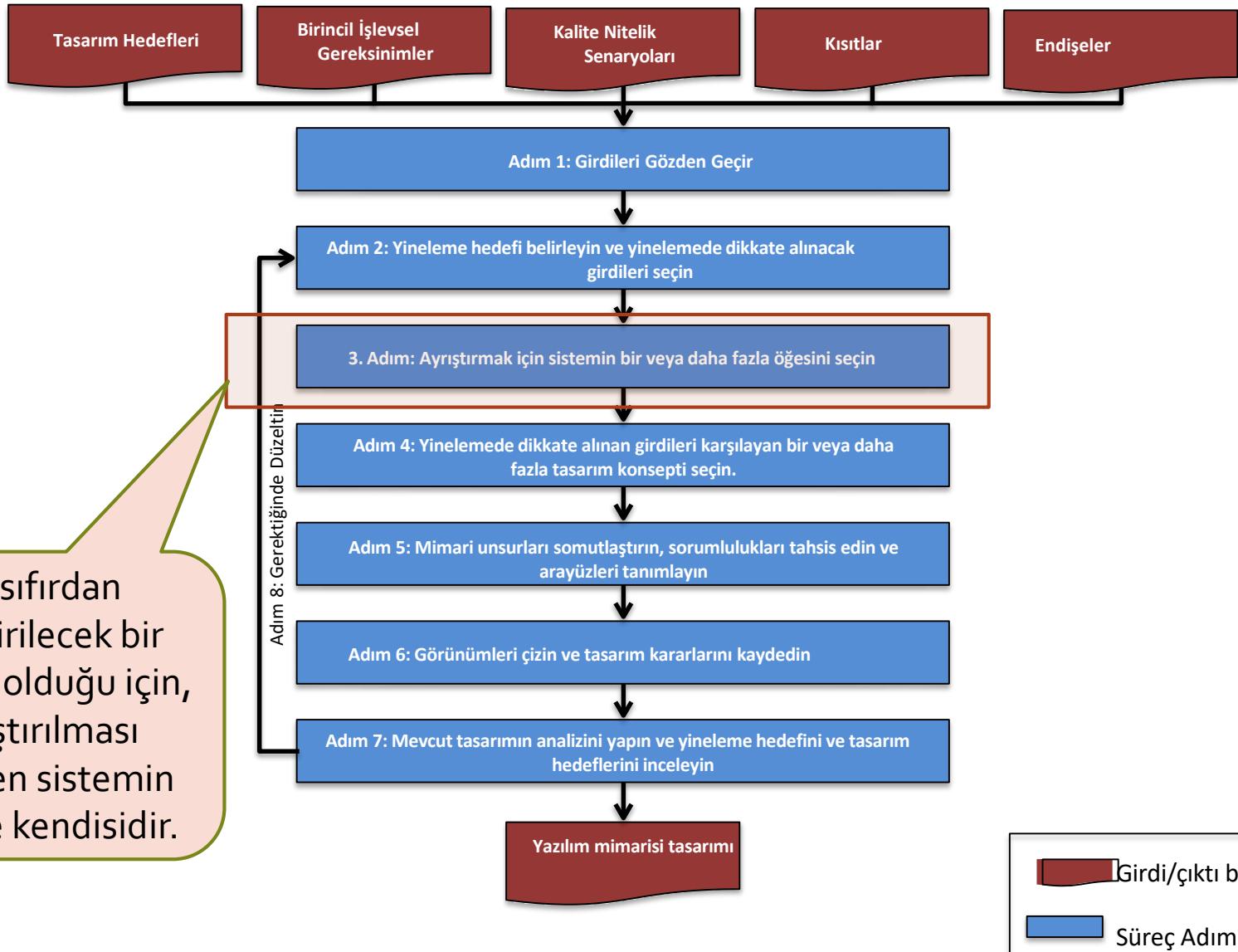


# Artırım Hedefleri ve Girdileri

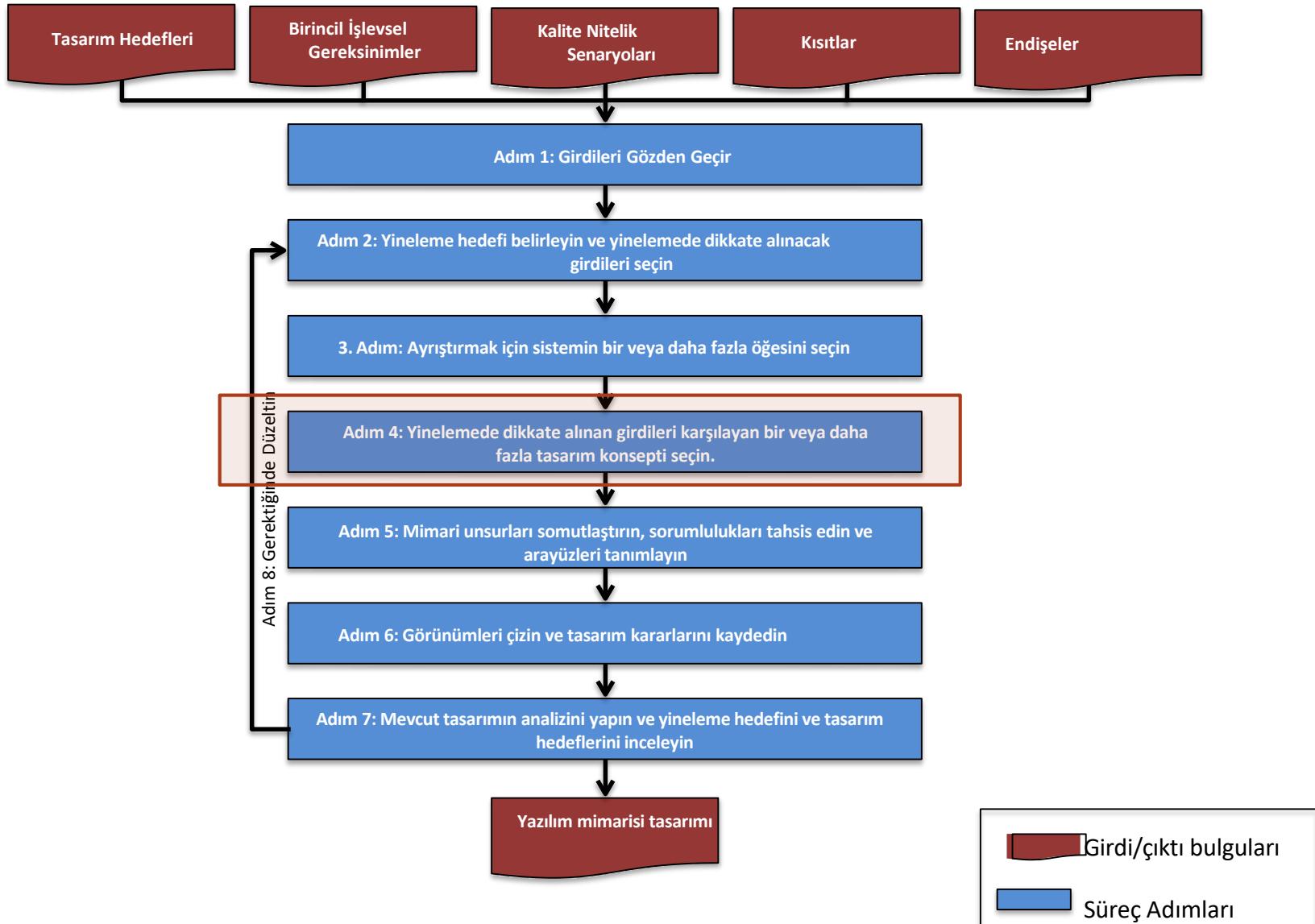
- Artırım Hedefi
  - Sistemin Genel Yapısını Oluşturun
- Dikkate alınması gereken girdiler
  - Hepsı



# ADD: Artırım I

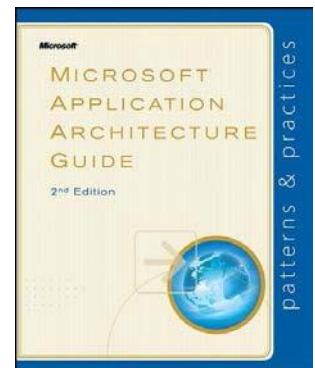


# ADD: Artırım I



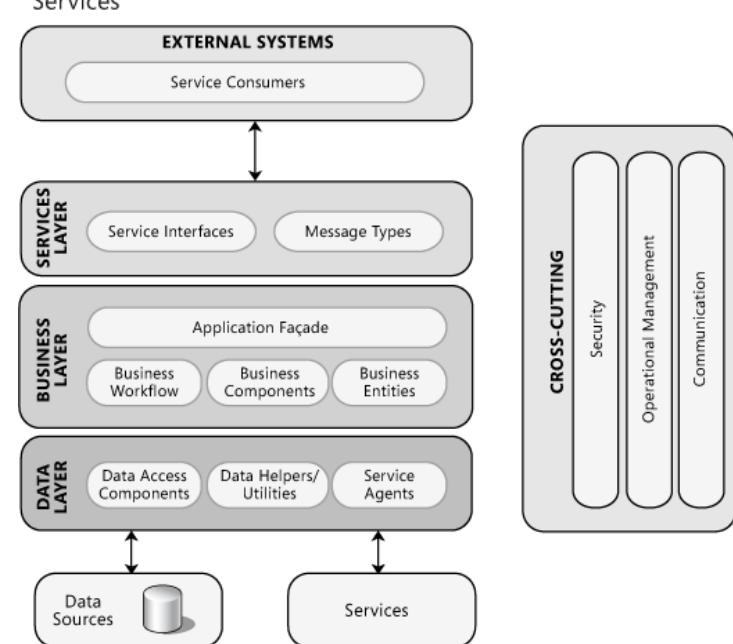
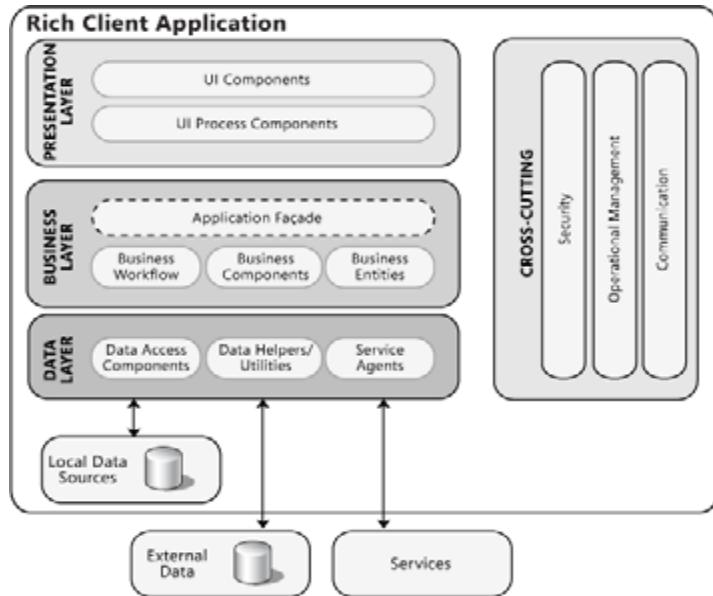
# Tasarım Konseptlerinin Seçimi

- Referans mimari alternatifler
  - Mobil uygulamalar
  - Zengin istemci uygulamaları
  - Zengin internet uygulamaları
  - Servis Uygulamaları
  - internet uygulamaları
- Dağıtılmış dağıtım modeli alternatifleri
  - 2 katman
  - 3 katman
  - 4 katman



# Tasarım Kararları

- İki adet referans mimarisi seçilmiştir

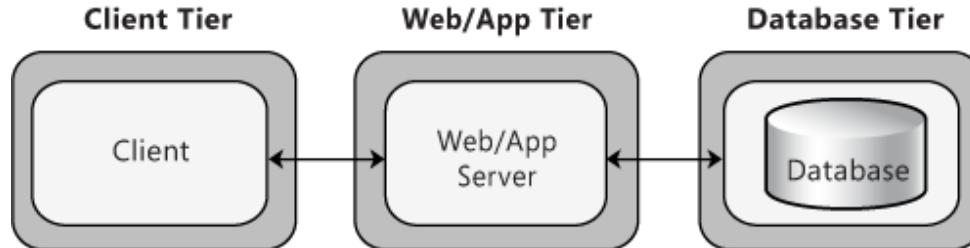


Tasarım kararı	Gerekçe
İstemci tarafında zengin istemci uygulaması	<ul style="list-style-type: none"><li>Zengin kullanıcı arayüzü destekler</li><li>Taşınabilirlik (CON-2)</li></ul>
Sunucu tarafında servis uygulaması	- Gelecekte mobil istemicileri destekleyin (CON-5)

# Tasarım Kararları

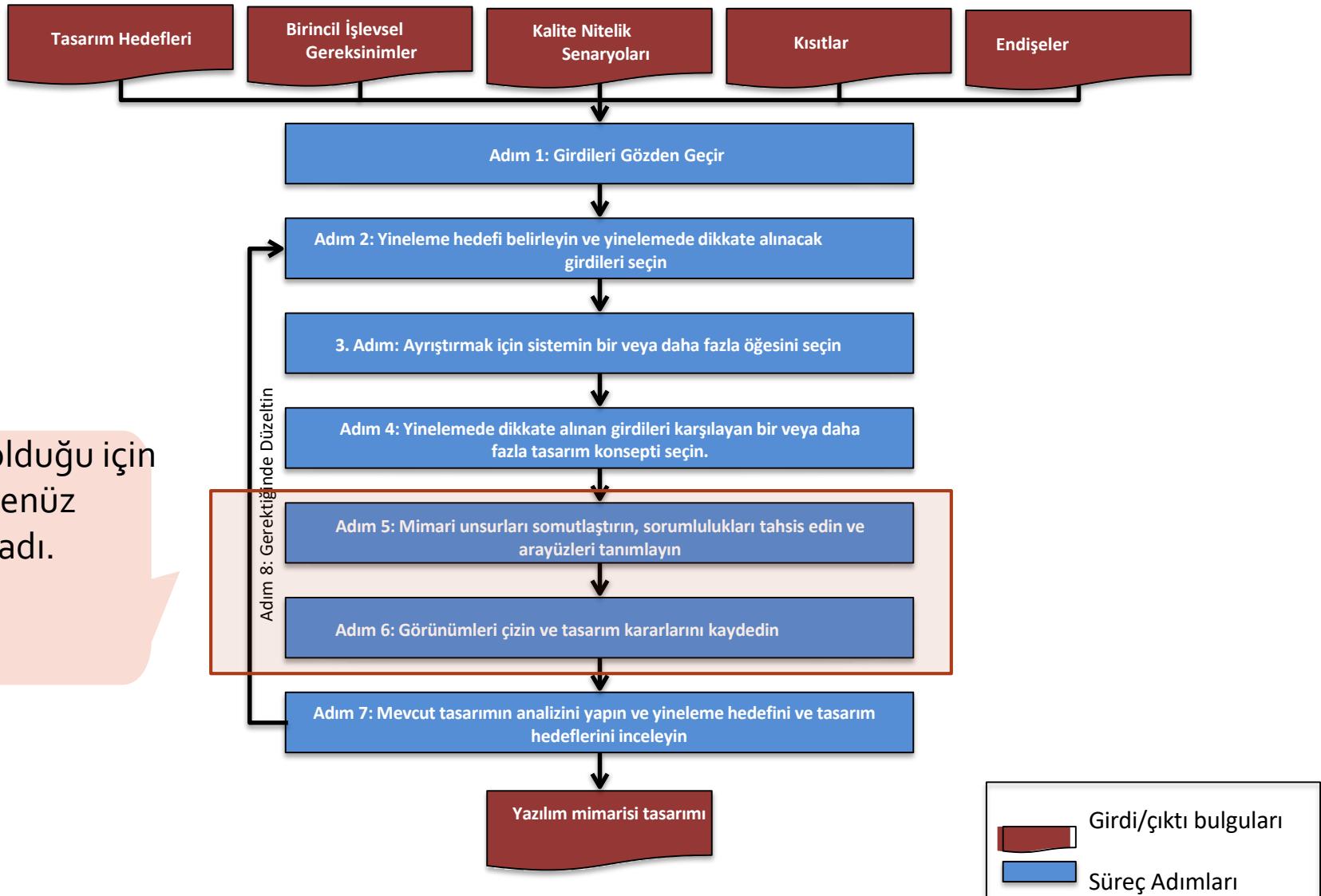
Tasarım kararı	Gerekçe
İstemci tarafında zengin istemci uygulaması	<ul style="list-style-type: none"><li>Zengin kullanıcı arayüzü destekler</li><li>Taşınabilirlik (CON-2)</li></ul>
Sunucu tarafında servis uygulaması	- Gelecekte mobil istemcileri destekleyin (CON-5)
3 Katmanlı uygulama	- Mevcut veritabanı sunucusu (CON-3)
...	...

- Dağıtılmış dağıtım modeli alternatifleri
  - 3 katman

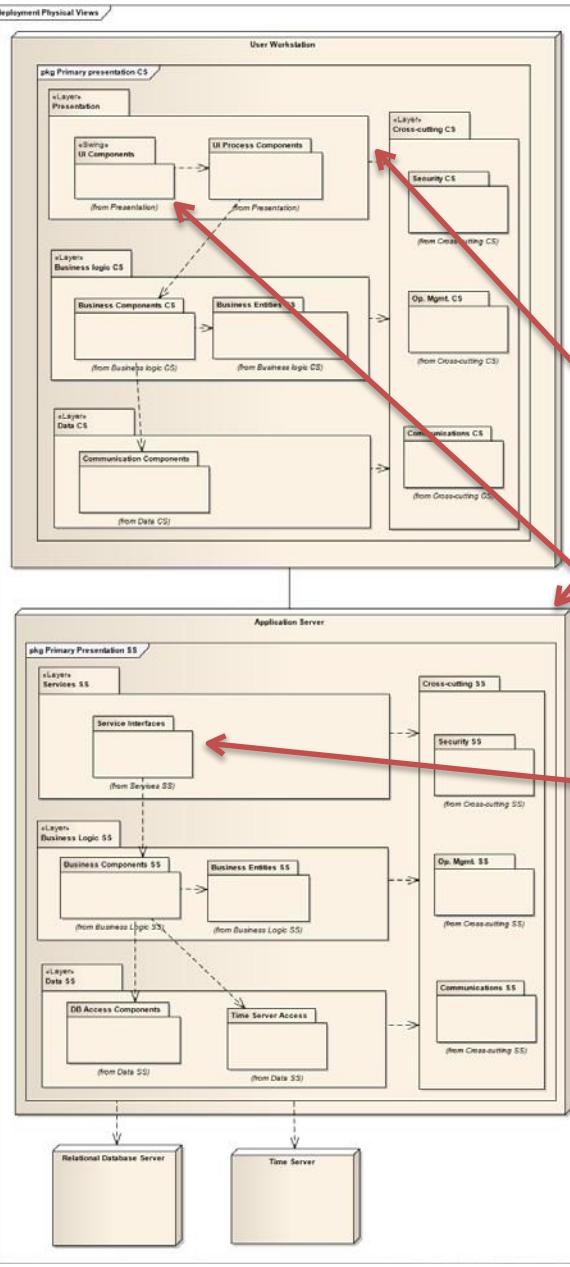


# ADD: Artırım I

İlk artırım olduğu için  
arayüzler henüz  
tanımlanmadı.



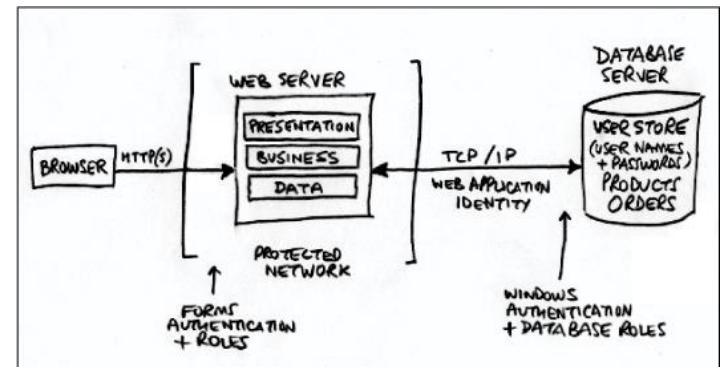
# Adım 5



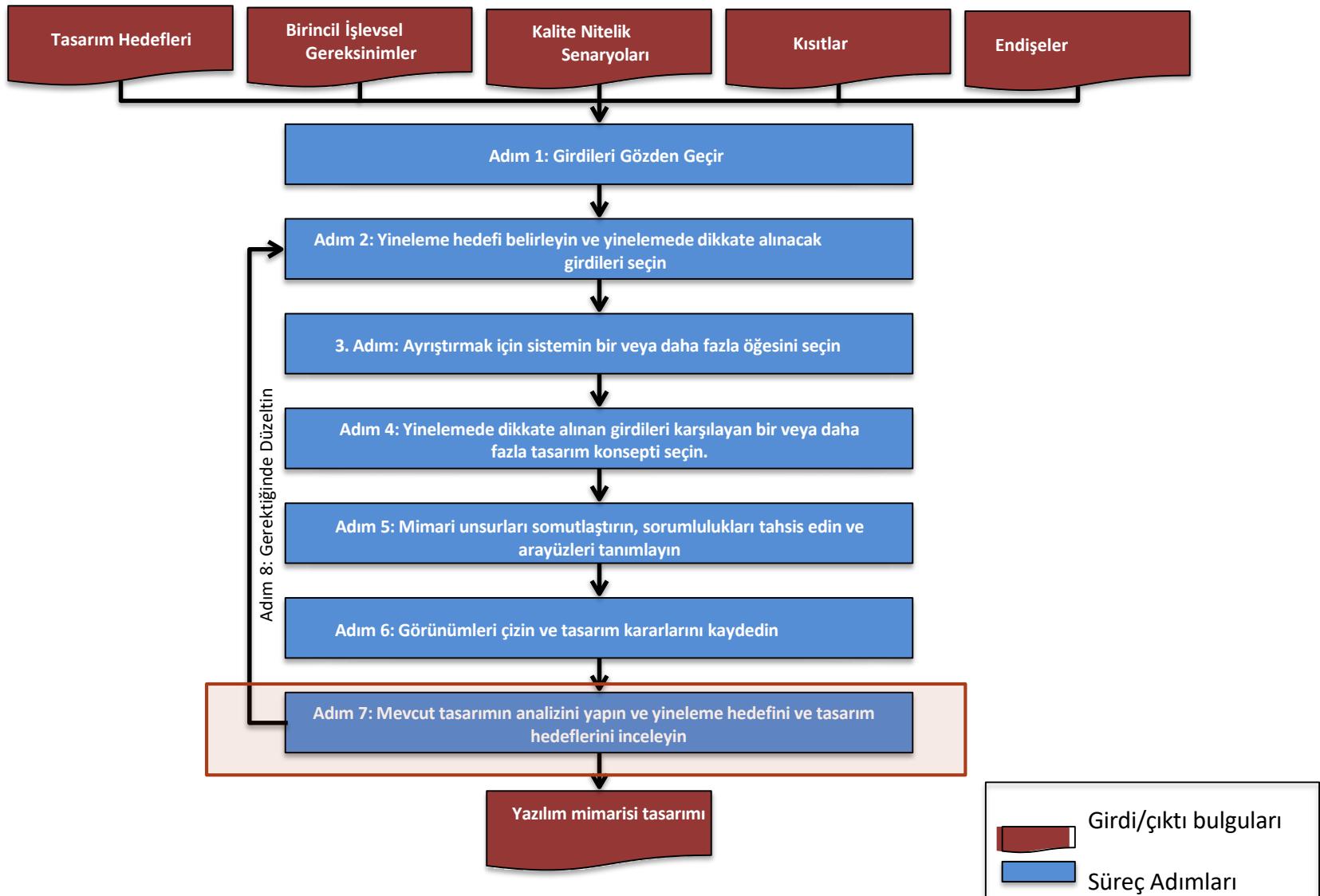
Eleman	Sorumluluk
<b>Uygulama sunucusu</b>	<ul style="list-style-type: none"> <li>• Servis referans mimarisinin öğelerini barındırır</li> <li>• Ağ cihazlarına bağlanır</li> <li>• Veritabanı sunucusuna bağlanır</li> </ul>
<b>Sunum katmanı (İstemci tarafı)</b>	Bu katman, kullanıcı etkileşimiini kontrol eden ve durum kontrol akışını kullanan bileşenleri içerir.
<b>UI Bileşenleri (İstemci tarafı)</b>	Bunlar, kullanıcı arayüzü oluşturulan ve kullanıcı etkileşimi alan bileşenlerdir.
<b>Servis Arayüzleri</b>	Bunlar, istemciler tarafından tüketilen hizmetleri ortaya çıkarılan bir grup bileşendir

# Tasarım Sırasında Belgeleme

- Tasarım konseptlerini somutlaştırırken genellikle eskizler oluşturursunuz . Bunlar, mimariniz için ilk belgelerdir.
  - onları yakala ve sonra et et
  - gayri resmi notasyon kullanırsanız tutarlı olun
  - Öğelere atadığınız sorumlulukları ve verdığınız ilgili tasarım kararlarını yazmanın bir disiplini geliştirmek
- Tasarım sırasında yazıya dökmeniz, daha sonra hatırlamak zorunda kalmayacağınız anlamına gelir ...



# ADD Artırımı I



# Adım 7: Kanban Panosu Tasarlayın

Ele Alınmadı

Kısmen Ele Alındı

Ele Alındı

Sistemi Yapılandırin

UC-1

UC-2

QA-1

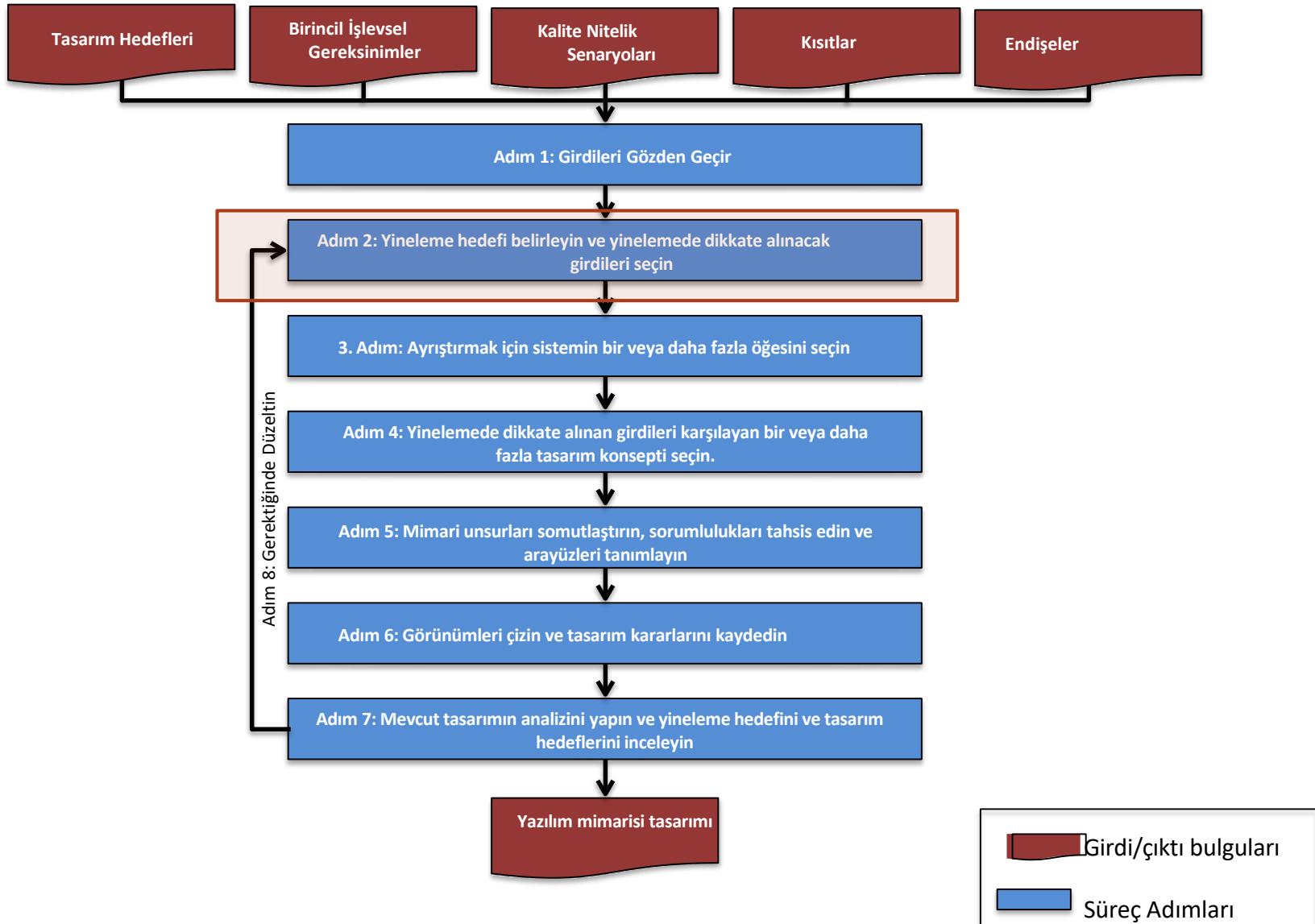
QA-3

QA-4

CON-3

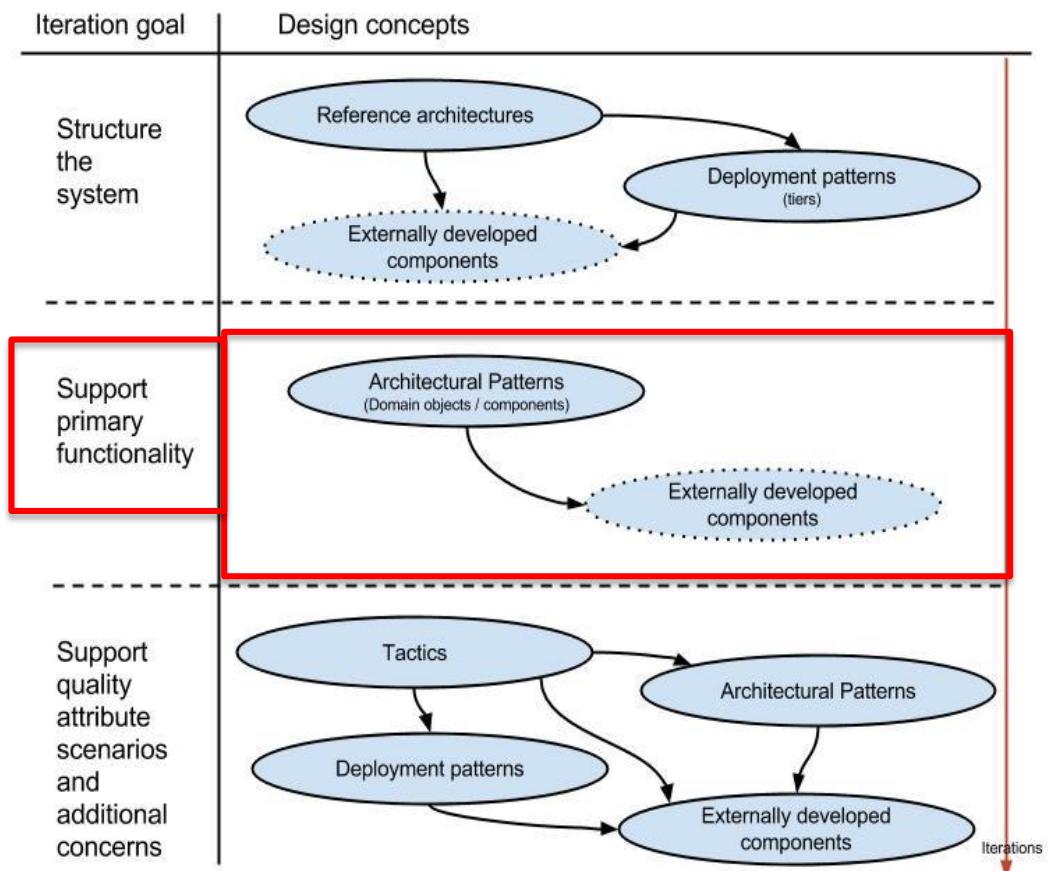
Tasarım Birikimi

# ADD Artırım 2

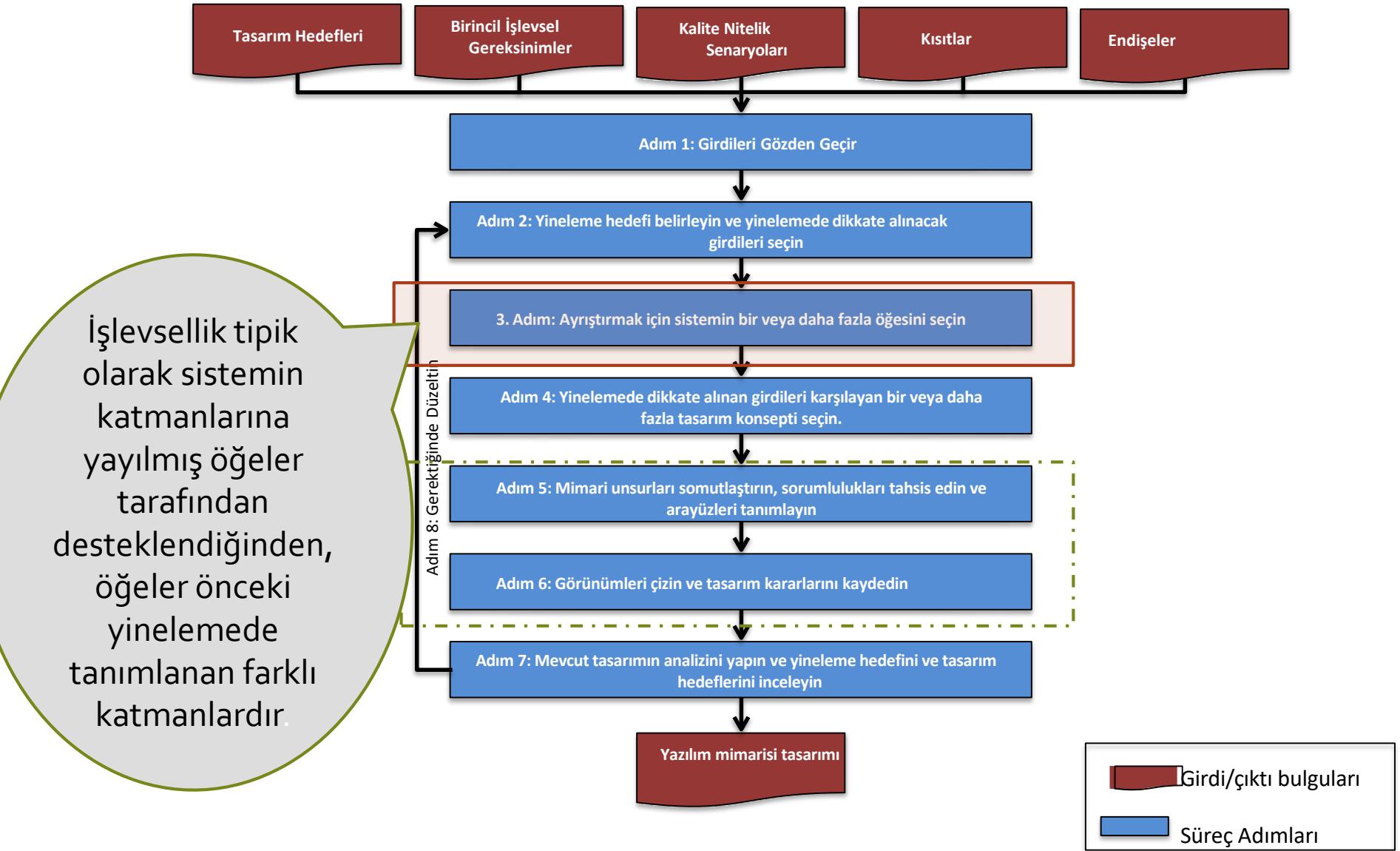


# Artırım Hedefleri ve Girdiler

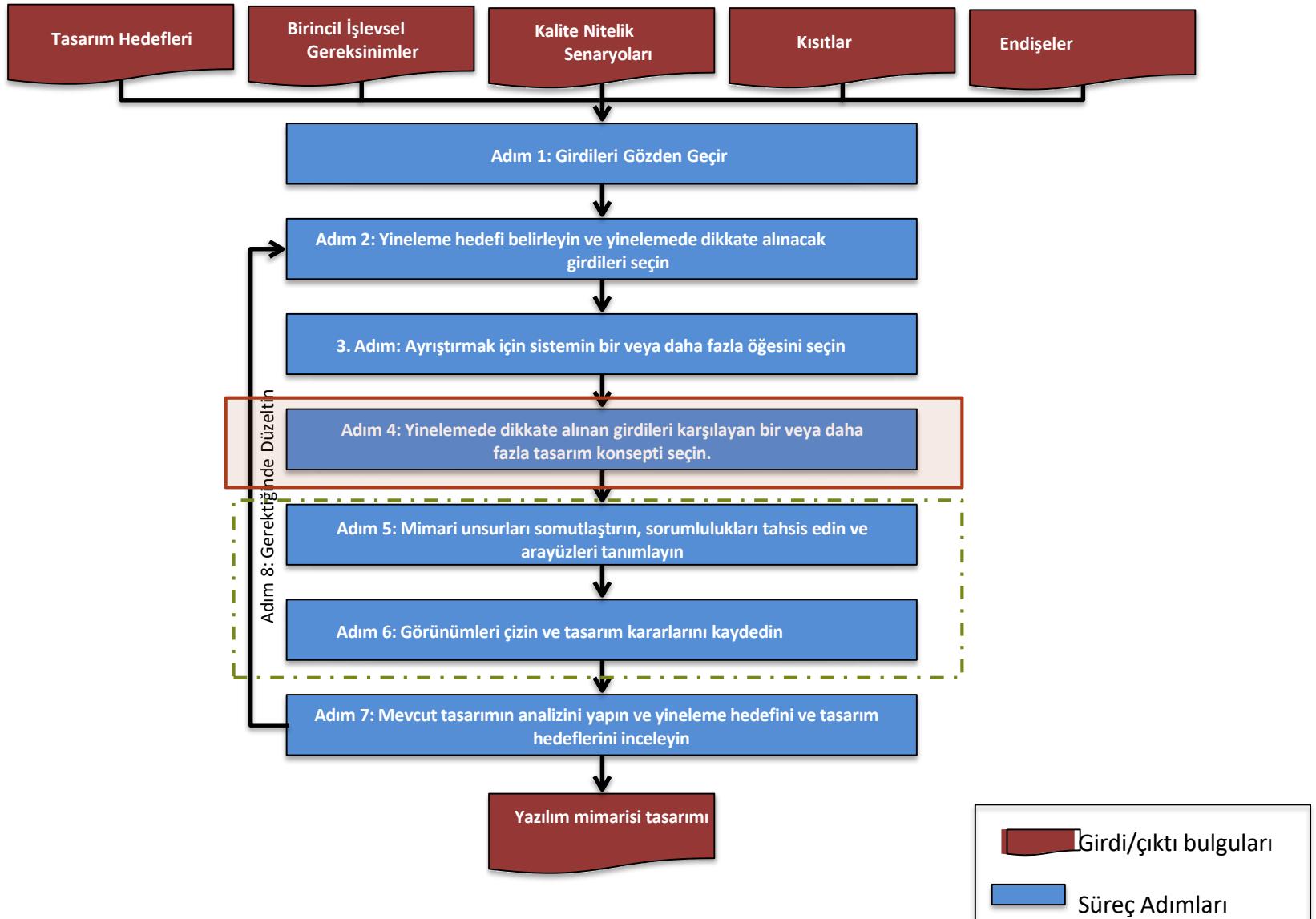
- Yineleme hedefi:
  - Birincil işlevselligi destekleyen öğeleri tanımlayın
- Girişler
  - Birincil Kullanım Durumları



# ADD: Artırım 2



# ADD Artırım 2



# Tasarım Konseptlerinin Seçimi

- Tasarım konseptleri
  - Etki alanı nesneleri
  - Dışarıdan geliştirilen bileşenler

Tasarım kararı	Gerekçe
Katmanları ayırtmak için etki alanı nesnelerini (ör. Bileşenler) kullanın	Katmanların ayrırtılması, iş atamasını ve bileşenleri bileşenlerle ilişkilendirmeyi kolaylaştırır
UI için Swing Framework'ü kullanın	Geliştiriciler bu çerçeveye aşinadır (CON-10)
VEYA Eşleme için Hazırda Bekletme'yi kullanın	<ul style="list-style-type: none"><li>• CON-10</li><li>• Yeni Veritabanı (eski değil)</li></ul>

208 From Mud To Structure

**Domain Object \*\***

When realizing a DOMAIN MODEL (182), or its technical architecture in terms of LAYERS (185), MODEL-VIEW-CONTROLLER (188), PRESENTATION-ABSTRACTION-CONTROL (191), MICROKERNEL (194), REFLECTION (197), PIPES AND FILTERS (200), SHARED REPOSITORY (202), or BLACKBOARD (205) ...

... a key concern of all design work is to decouple self-contained and coherent application responsibilities from one another.

The parts that make up a software system often expose manifold collaboration and containment relationships to one another. However, implementing such interrelated functionality without care can result in a design with a high structural complexity.

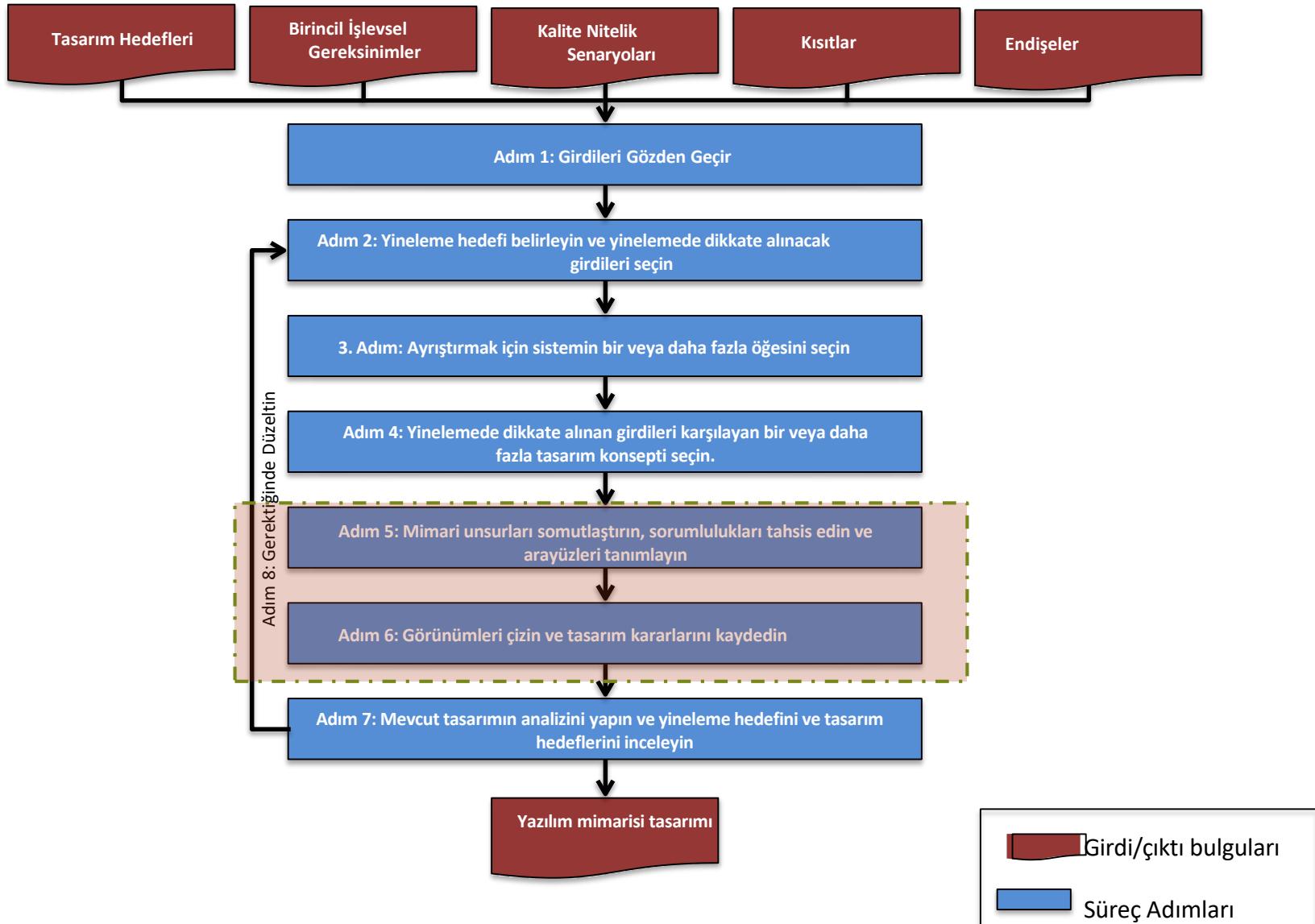
Separation of concerns is a key property of well-designed software. The more decoupled are the different parts of a software system, the better they can be developed and evolved independently. The fewer relationships the parts have to one another, the smaller the structural complexity of the software architecture. The looser the parts are coupled, the better they can be deployed in a computer network or composed into larger applications. In other words, a proper partitioning of a software system avoids architectural fragmentation, and developers can better maintain, evolve and reason about it. Yet despite the need for clear separation of concerns, the implementation of and collaboration between different parts in a software system must be effective and efficient for key operational qualities, such as performance, error handling, and security.

Therefore:

Encapsulate each distinct functionality of an application in a self-contained building-block—a domain object.

```
graph LR; DOI[Domain Object Interface] --- FA[Function A]; FA --- DO1[Domain Object 1]; subgraph DO1 [Domain Object 1]; FA --- FB[Function B]; FB --- FC[Function C]; end; subgraph DO2 [Domain Object 2]; FB --- FC --- FZ[Function Z]; end; subgraph DO3 [Domain Object 3]; FX[Function X] --- FY[Function Y]; end; subgraph DI [Domain Object Implementation]; DOI --- FA; DOI --- DO1; end;
```

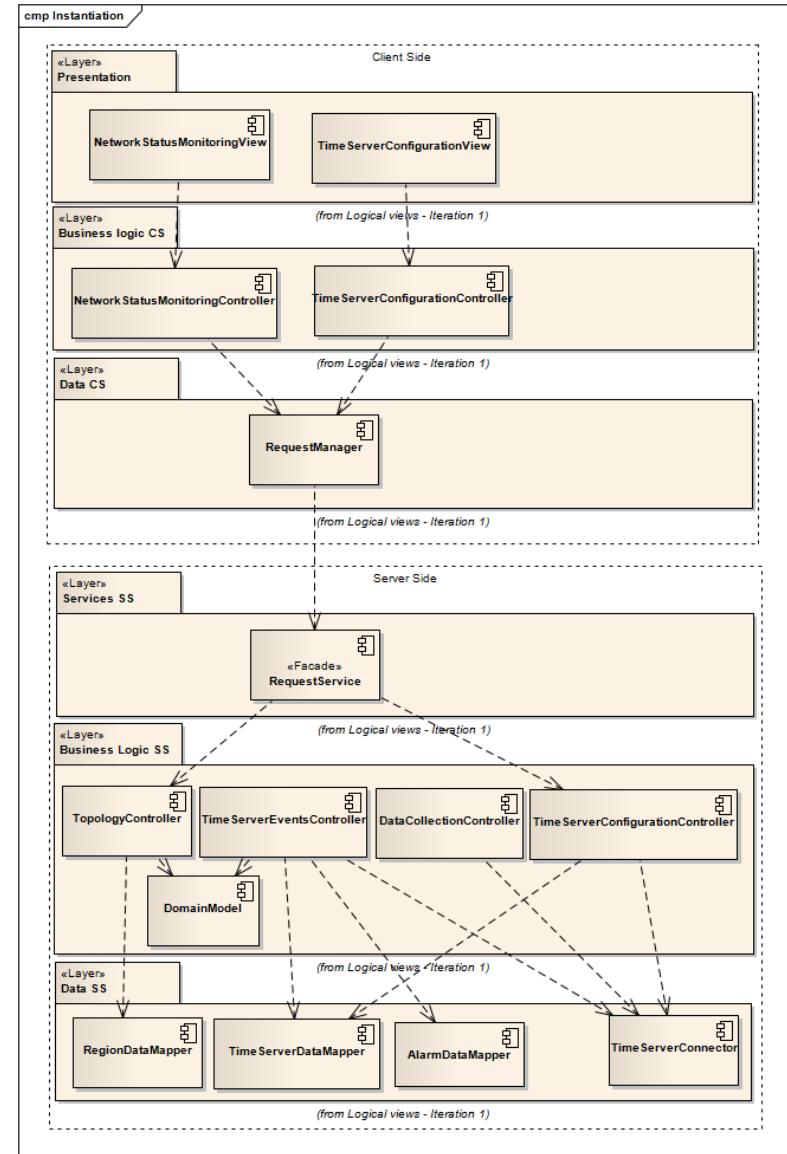
# ADD Artırım 2



# Adım 5

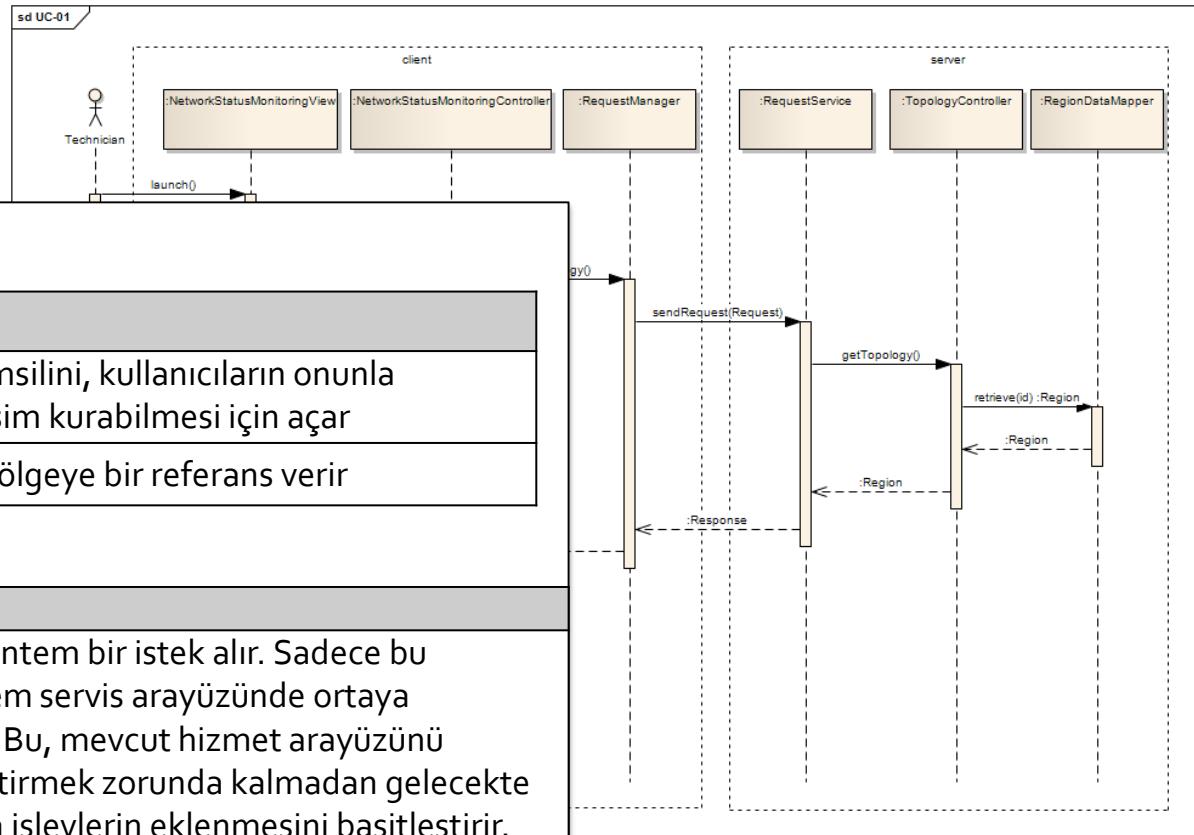
## • Mantıksal Görünüm

Eleman	Sorumluluk	Özellikleri
NetworkDeviceConnector	Ağ cihazlarıyla iletişim kurun ve sistemin geri kalanını belirli protokollerden ayıran	
NetworkDeviceEventContr oller	Ağ cihazlarından alınan olayları işleme	
Topoloji Denetleyicisi	Ağ topolojisi bilgilerine ve içindeki değişikliklere erişim sağlar	Tür = durumsuz
Bölge Veri Eşleştirmecis i	Bölgelerin kalıcılığını yönetin	Çerçeve = Hazırda Bekletme
NetworkStatusMonitoring View	Cihazlarda meydana gelen ağ topolojisini ve olayları görüntüleyin	Çerçeve = Salıncak

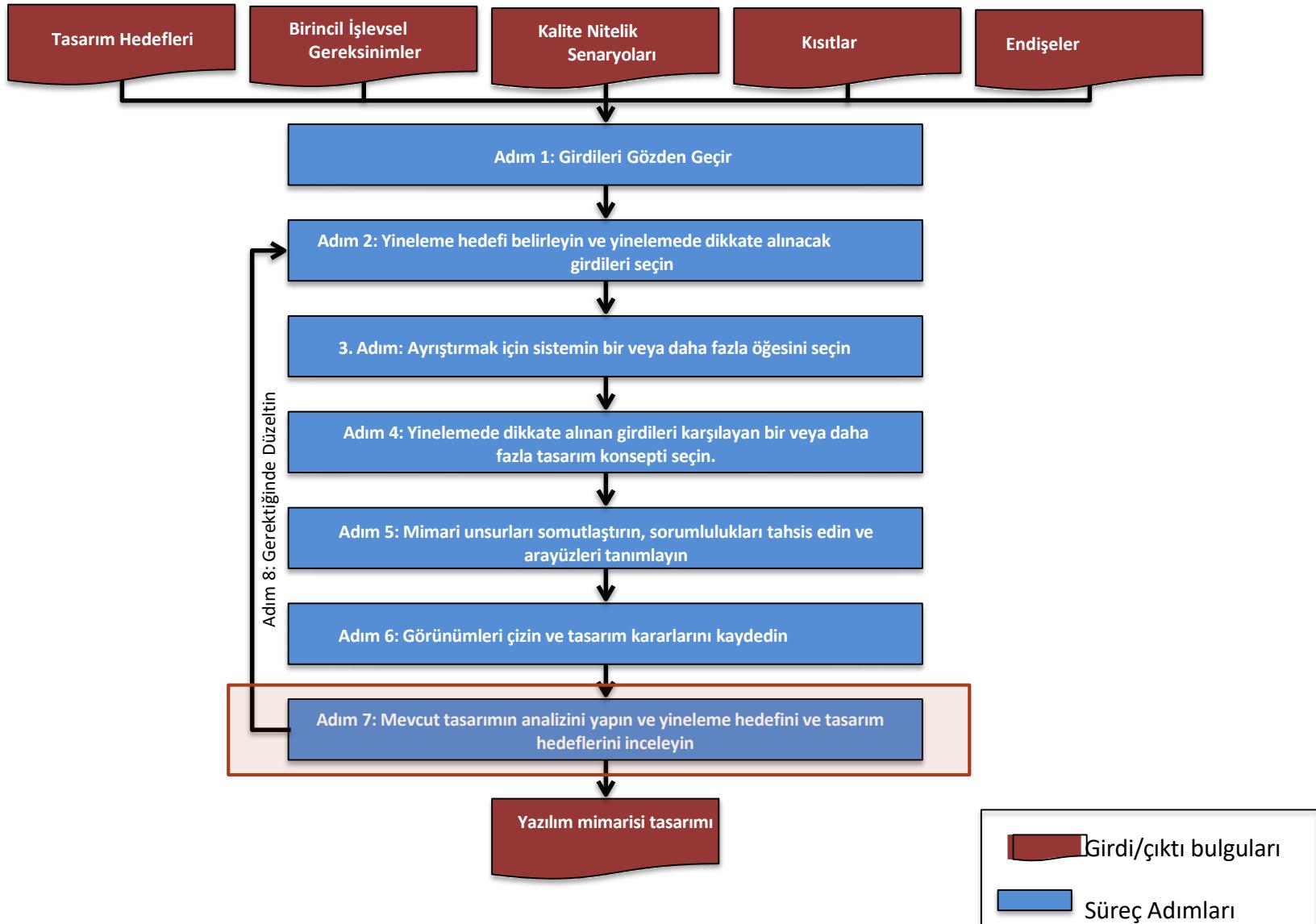


# Adım 5: Arayüzler

- Öğeler tanımlandıktan sonra, dinamik analiz arayüzlerin tanımlanmasına izin verir
  - UC-1



# ADD Artırım 2



# Adım 7: Kanban Panosu Tasarla

Ele Alınmadı

Kısmen Ele Alındı

Ele Alındı

UC-1

UC-2

QA-1

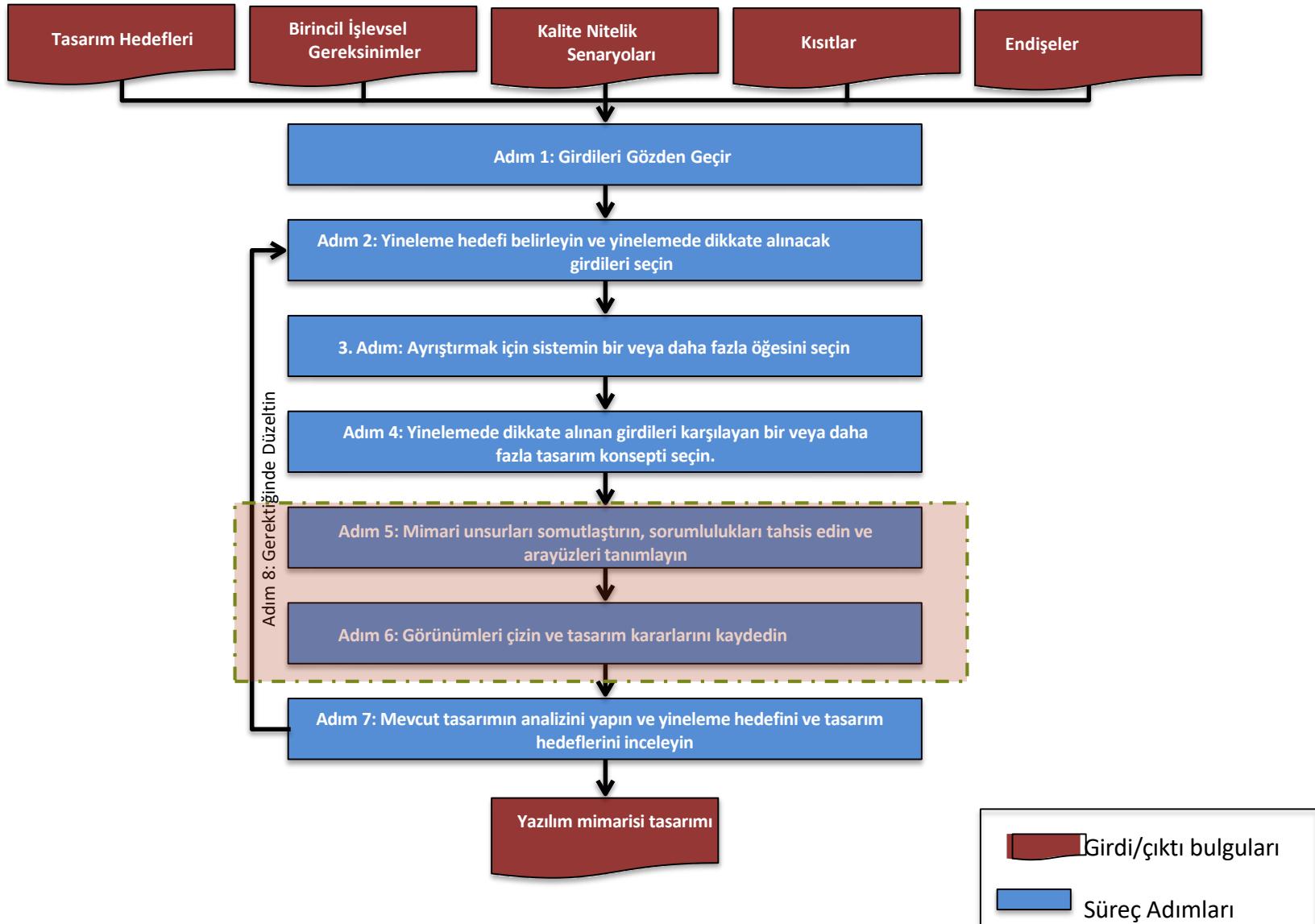
QA-3

QA-4

Sistemi Yapılandırır

CON-3

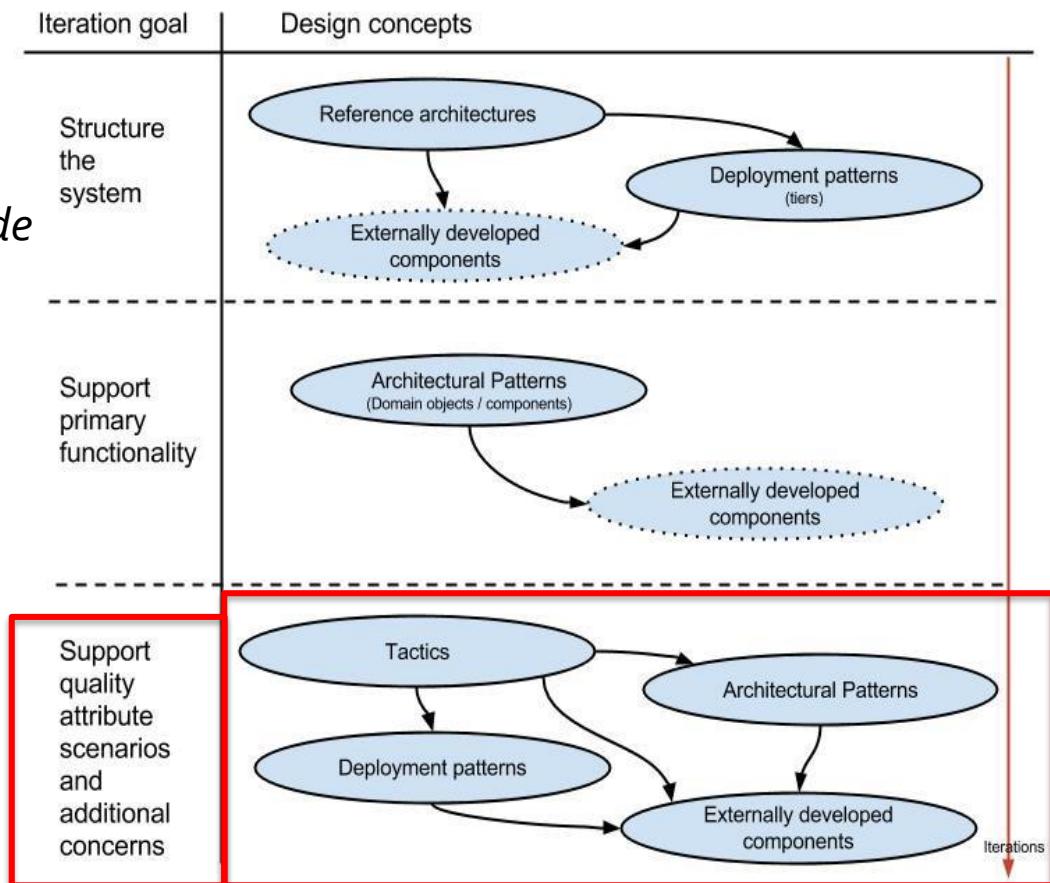
# ADD Artırım 2



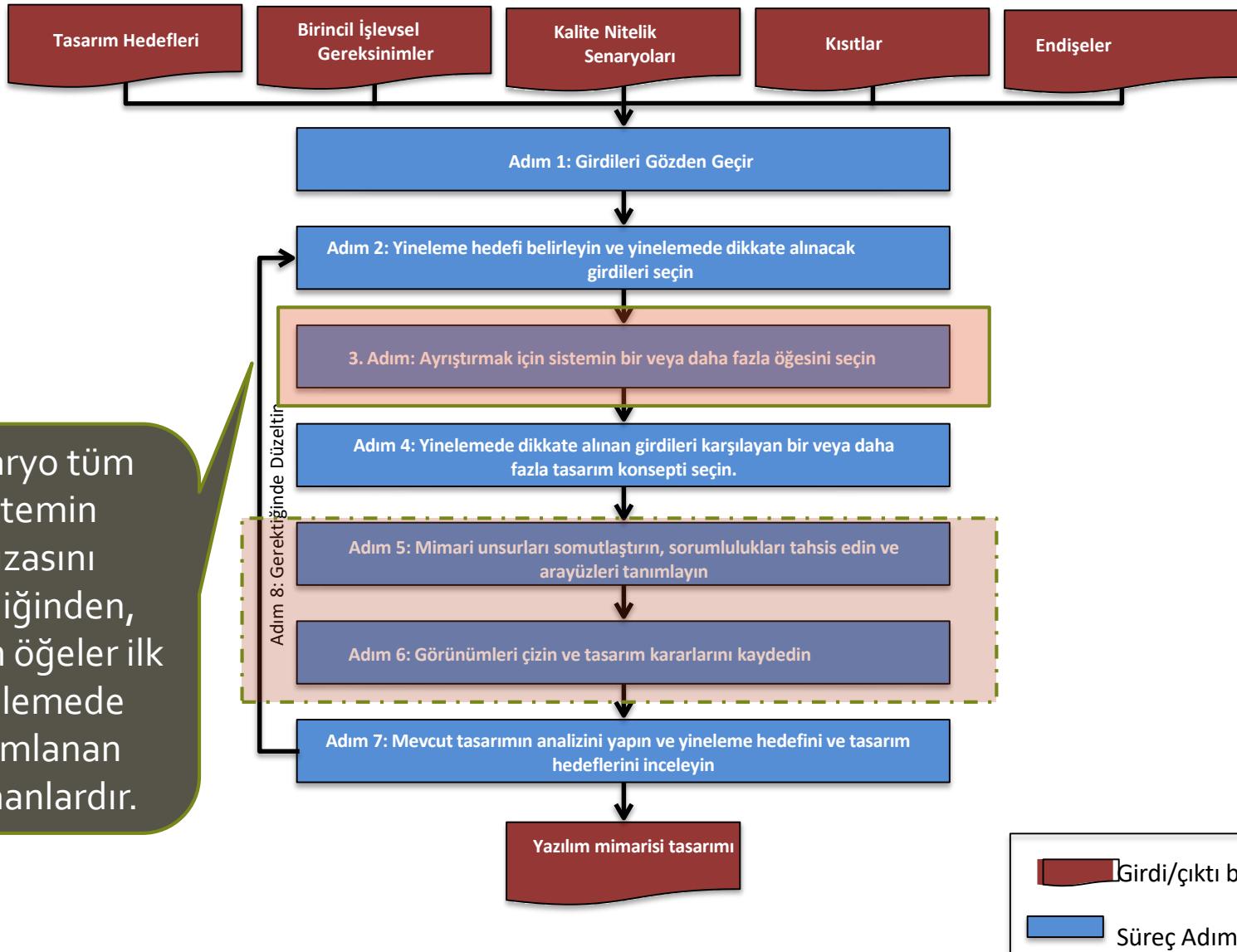
# Artırım Hedefi ve Girdiler

Yineleme hedefi:

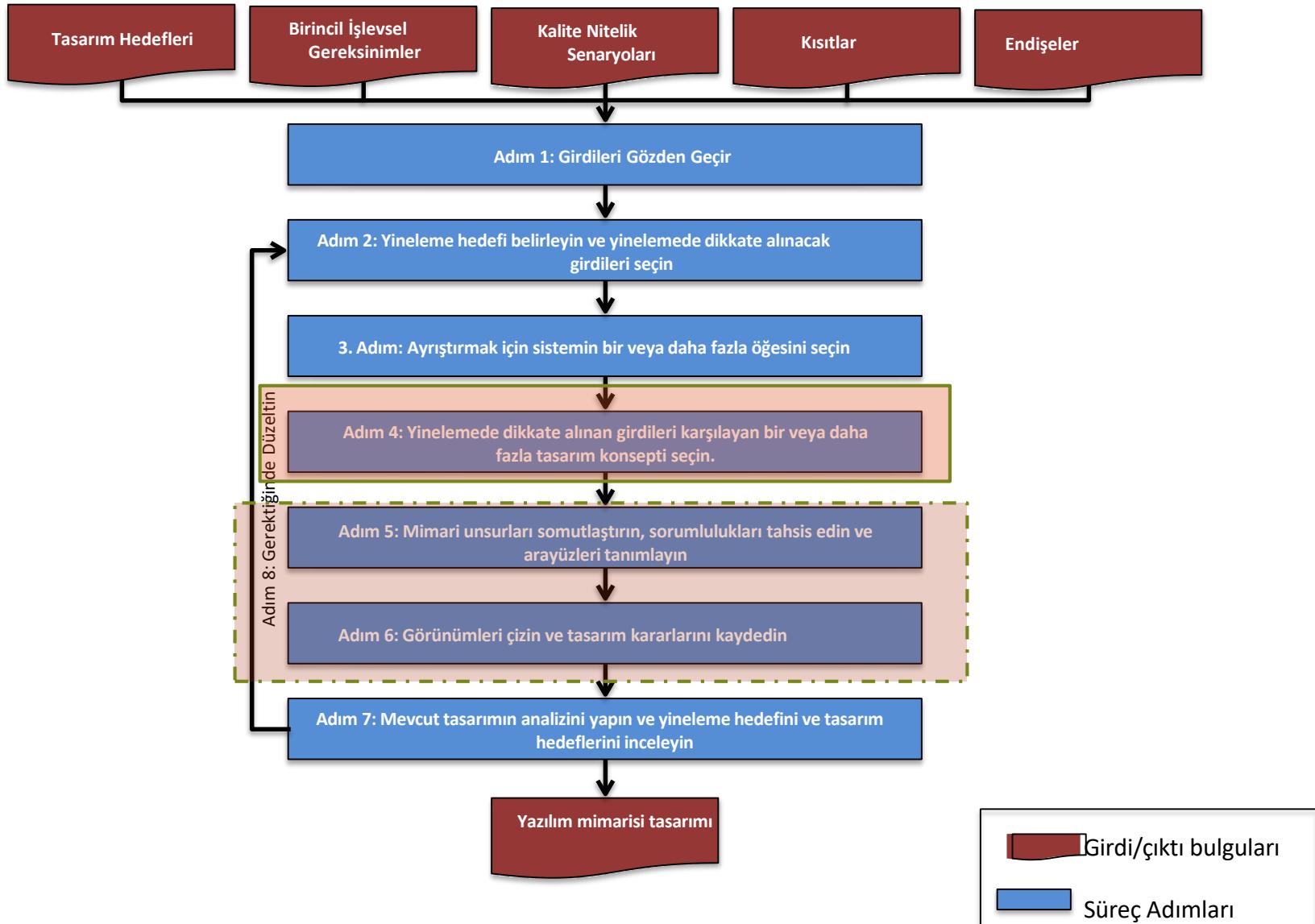
- Kalite Gereksinimi KG3'ü adresler:  
*İşletim sırasında ağ yönetim sisteminde bir arıza meydana gelir. Sistem, 30 saniyeden daha kısa sürede çalışmaya devam eder.*



# ADD Artırım 3

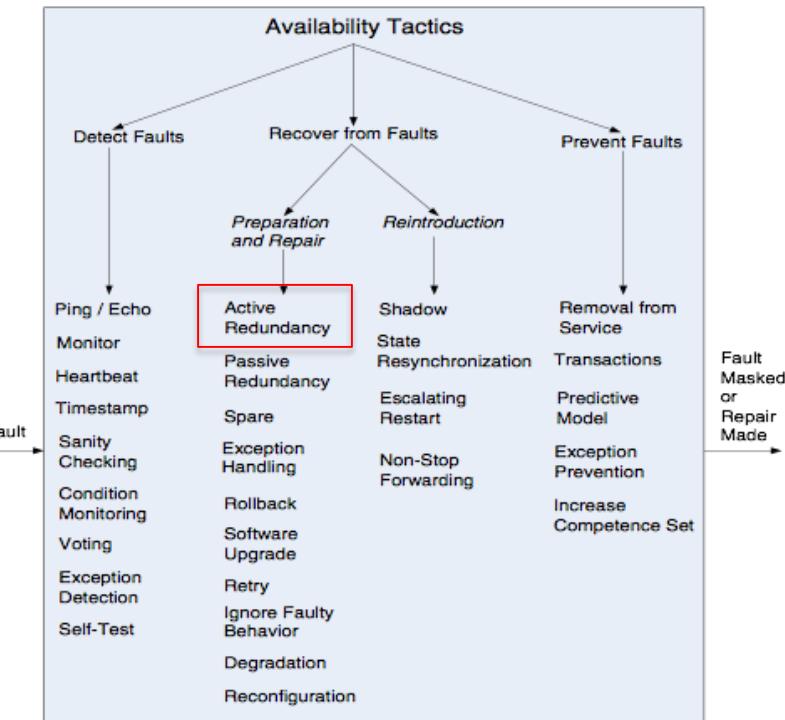


# ADD Artırım 3



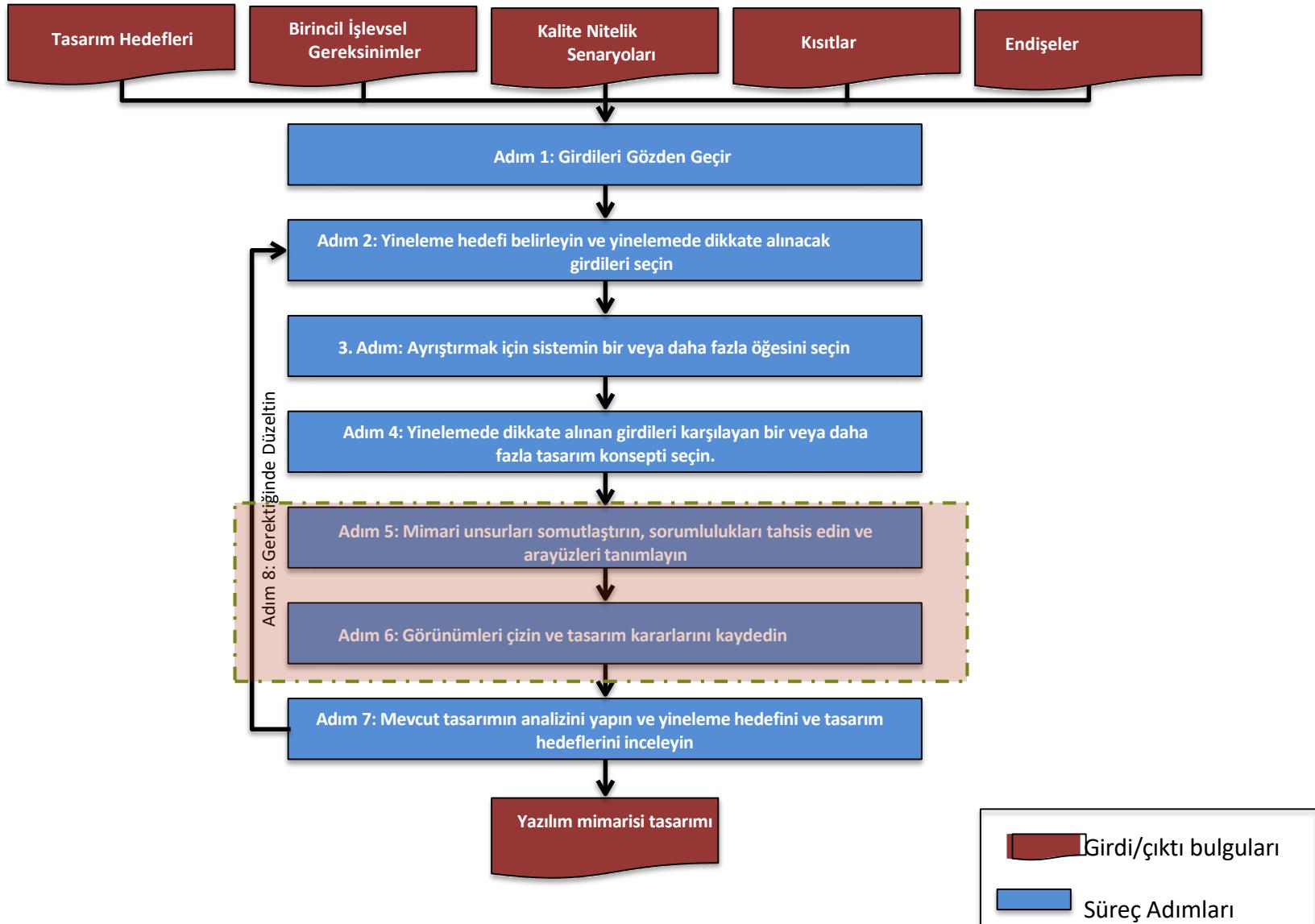
# Adım 4

- Taktiklerle başlamanız ve oradan kalıplara veya teknolojilere gitmeniz önerilir.

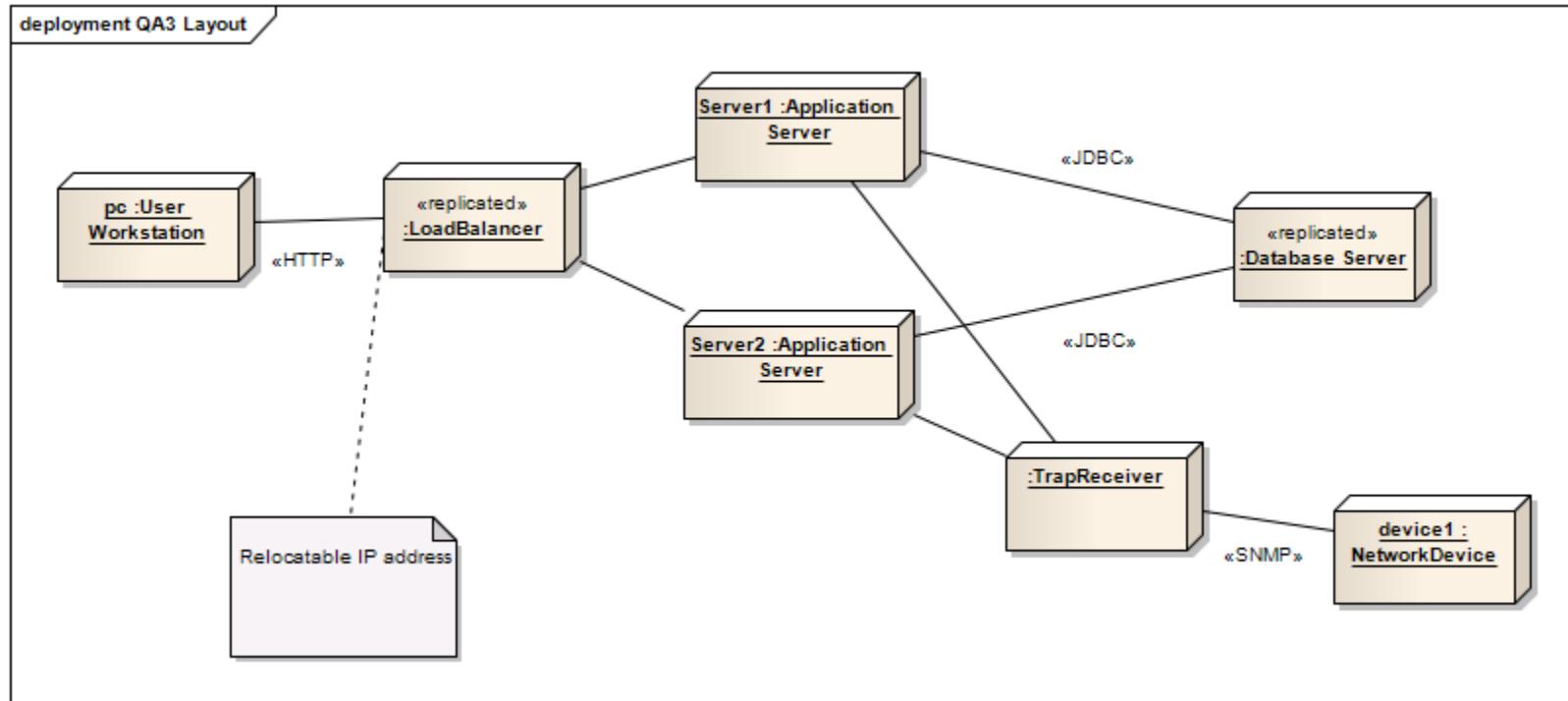


Tasarım kararı	Gerekçe
Uygulama sunucusunu ve veritabanı gibi diğer kritik bileşenleri çoğaltarak aktif yedeklilik sağlayın	Sistem, kritik bileşenleri çoğaltarak, işlevselliği etkilemeden çoğaltılan öğelerden birinin arızasına dayanabilir.
Apache + mod_proxy kullanarak artıklık uygulayın	Apache giriş noktası olarak hizmet verir, mod_proxy yük dengeleyici görevi görür

# ADD Artırım 3

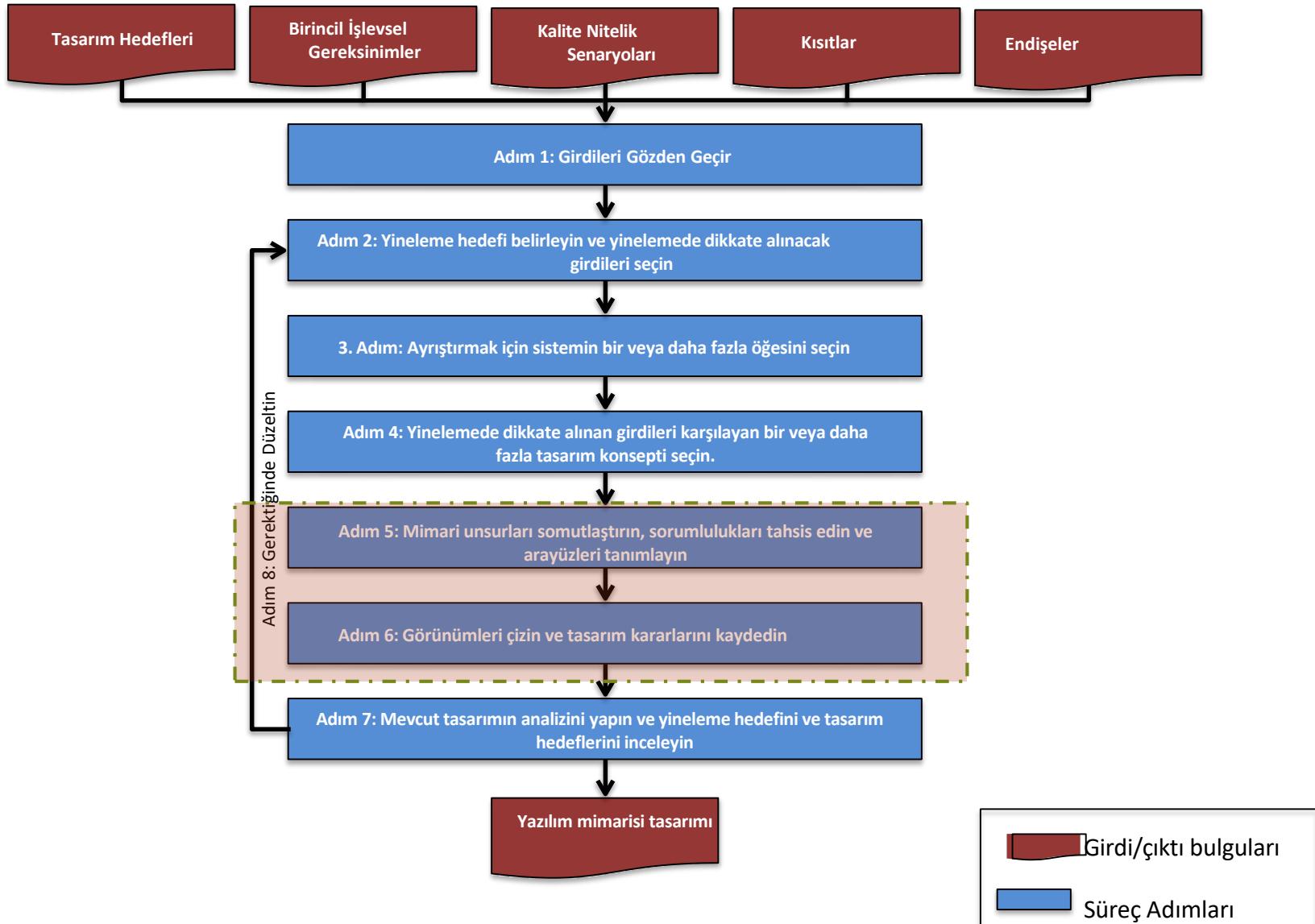


# Adım 5



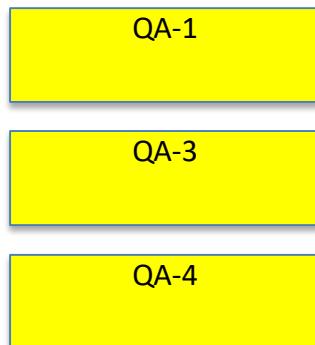
Eleman	Sorumluluk	Özellikleri
TrapReceiver	Ağ cihazlarından tuzakları alın, bunları olaylara dönüştürün ve bu olayları bir mesaj kuyruğuna koyn	Çerçeve = SNMP4J
...	...	...

# ADD Artırım 3



# Adım 7: Kanban Panosu Tasarlayın

Not addressed



Partially addressed

Addressed

Structure the system

UC-1

UC-2

More decisions need  
to be made

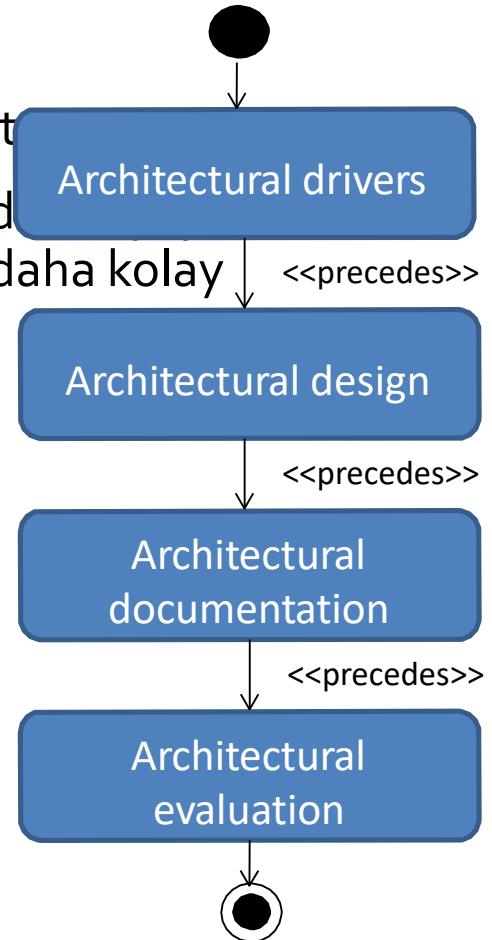
CON-3

# Tasarım Süreci Sonlandırma Kriterleri

- Tasarım süreci birkaç yinelemeyle devam eder:
  - tüm etmen mimarisi gereksinimleri için tasarım kararları alınana kadar (tasarım hedefine ulaşıldı); veya
  - en önemli teknik riskler azaltılıncaaya kadar; veya
  - mimari tasarım için ayrılan süre tüketilene kadar (çok arzu edilen bir şey değil!).

# Ek hususlar

- Tasarım sürecinin bir parçası olarak prototip oluşturmak, tasarım adımları sistematik olarak gerçekleştirildiğinde daha kolay gerçekleştirilebilir.
- Dokümantasyon oluşturma ve mimari değerlendirmenin tasarım adımları sistematik olarak gerçekleştirildiğinde daha kolay gerçekleştirilebilir.



# Özet

- Mimari tasarım, etkenleri (etmenleri) yapılara dönüştürür.
- Mimari etkenler, işlevsel gereksinimleri, kalite özelliklerini ve kısıtlamaları, aynı zamanda hedefleri, endişeleri ve sistemin türünü içerir.
- ADD, mimari tasarıımı sistematik olarak gerçekleştirilebilecek şekilde yapılandıran bir yöntemdir.
- Tasarım konseptleri, tasarımın oluşturduğu yapı taşlarıdır. Birkaç önemli tür vardır: Referans Mimariler, Dağıtım kalıpları, Mimari Modeller, Taktikler ve çerçeveler gibi dışarıdan geliştirilmiş bileşenler.
- ADD, tasarımdaki ilerlemeyi izlemek için ilk dokümantasyon (eskizler) ve bir tasarım kanban panosu kullanılarak çevik bir şekilde gerçekleştirilebilir

# Teşekkürler

- **Sorular?**