

Monolitten mikroservise

Ne zaman, neden, nasıl: sınırı bugün çizim, dağıtım
ölçülmüş bir sinyal gelince yapın.

Muhammet Şafak • 3 Ekim 2026



Sınır

Modüler monolit: tek codebase, tek deploy, tek veritabanı; içeride net modül sınırları.

Sınır nerede duruyor?

YAKLAŞIM	SINIR
Klasik monolit	Sınır yok. Her şey her şeye erişir, zamanla "big ball of mud".
Mikroservisler	Sınır var, ama sınır aynı zamanda bir network çağrısı.
Modüler monolit	Sınır var, ama sınır process içinde; bir method çağrısı.

ASIL MESELE

Sınır ile servis ayrı iki karardır.

Sınırı bugün çizin, dağıtımı gerçekten gerekince yapın.

SINIR KODDA

Modül başına tek public yüzey

Bir modül, başka bir modülün
yalnızca `*Api.php` sınıfını çağırır.
Başka modülün tablosuna doğrudan
`SELECT` atmaz, foreign key
bağlamaz.

app/Modules

```
1 app/
2   └─ Modules/
3       └─ Billing/
4           └─ Domain/          # modüle özel entity, value object
5           └─ Application/     # use-case 'ler
6           └─ Infrastructure/  # repository, dış servis adaptörü
7           └─ BillingApi.php   # modülün TEK public yüzeyi
8       └─ Catalog/
9       └─ Notification/
```

Beş somut sebep



Ucuz düzeltme

Yanlış çizilen sınır bir refactor ile düzelir.



Tek transaction

İki modül tek DB transaction'ında tutarlı güncellenir.



Az operasyon

Tek deploy, tek log akışı, izlenecek tek hedef.



Refactor araçları

Bir method çağrısını compiler doğrular.



Ucuz geçiş

Net sınırlı modülü servise çıkarmak mekanik bir iştir.

Sınırı ayakta tutan dört pratik

- ✓ Bağımlılık kuralı CI'da; ihlal build'i kırar (deptraç)
- ✓ Her modülün dışarı açık API'si tek bir sınıf
- ✓ Modüller arası ilişki foreign key değil, ID + public çağrı
- ✓ Review sorusu: bu değişiklik bir modül sınırını izinsiz geçiyor mu?



BÜTÜN DEĞER DİSİPLİNDE

CI zorlamıyorsa sınır yoktur.

Sınır CI tarafından zorlanmıyorsa, elinizde modüler monolit değil, sadece klasör isimleri düzgün bir monolit var.

Bedel

Mikroservis bir çözüm değil, bir takas.

TEK DEĞİŞİKLİK

Method çağrısı network'e döner.

Aldığınız şey bağımsız deploy ve bağımsız ölçekleme.
Karşılığında ödediğiniz şey uzun.

Mikroservisin faturası

	Mikroserviste	Monolitte
ÇAĞRI	Serializasyon, latency, partial failure	✓ In-process, ücretsiz
TUTARLILIK	Saga, outbox, eventual consistency	✓ Tek DB transaction'ı
İZLEME	Distributed tracing şart	✓ Tek log akışı
DEPLOY	Versiyonlu, geriye uyumlu kontratlar	✓ Tek deploy
REFACTOR	Kimsenin doğrulamadığı kontrat	✓ Compiler doğrular

Doğru hastalığa yanlış ilaç

GEREKÇE	ASIL ÇÖZÜM
"Kod tabanı çok büyük."	Modül sınırı, dosya düzeni, ölü kod temizliği
"Deploy yavaş ve korkutucu."	Hızlı pipeline, test, aşamalı yayın, rollback
"Bağımsız ölçeklemem lazım."	Monolit de yatay ölçeklenir: N kopya + load balancer
"Modern mimari bu."	Modernlik mimari gerekçe değil
"Netflix böyle yapıyor."	Problemi değil, yalnız operasyon yükü kopyalanır

KOD BÜYÜKLÜĞÜ

Aynı kod, araya network.

200 bin satırlık bir monolit ile 20 servise bölünmüş 200 bin satır aynı miktarda koddur; ikincisi sadece araya network koymuş hâlidir.

Sinyal

Şu sinyallerden en az biri ölçülmüş olmalı.

Geçiş hakkı çıkaran dört sinyal



Ölçeklenme profili

Bir parça kaynakları geri kalanından bambaşka tüketiyor.



Sahiplik

Ekipler birbirinin deploy'unu bekliyor; sınır organizasyonel.



Runtime

Ölçülmüş darboğaz: iş yükü Go ya da Rust'ta belirgin ucuz.



Hata izolasyonu

Bir modülün çöküşü sistemi düşürmemeli; process içinde sağlanamıyor.

EŞİK

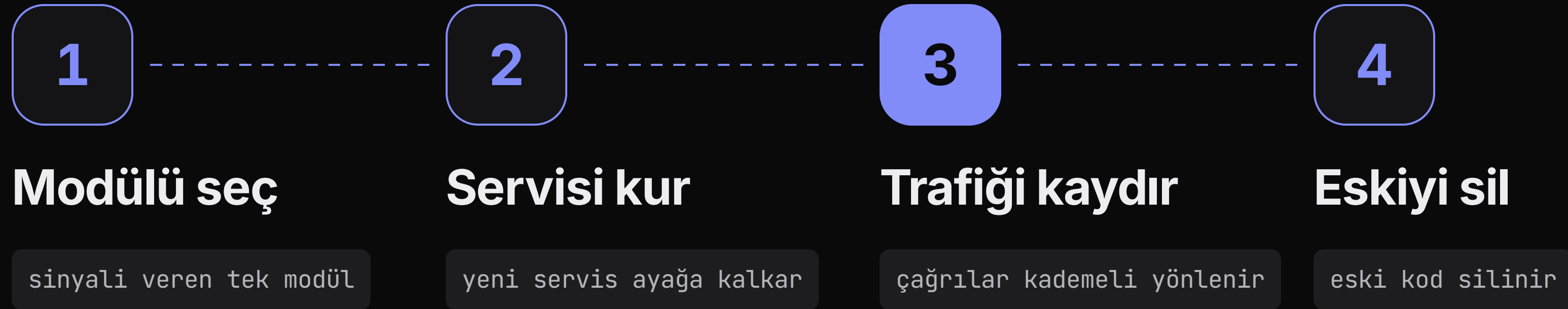
Grafikte yoksa sinyal değildir.

Sinyal bir grafikte görünmüyorsa, henüz sinyal değildir.

Geçiş

Hepsini değil, sinyali veren tek modülü ayırın.

Tek modülü dışarı almak



Ne zaman bölünür, nasıl bölünmez?



BİG-BANG

Her şeyi aynı anda bölmek en pahalı ve en sık başarısız göçtür.



EKİP

Birden çok ekip aynı deploy biriminde sıkıştığında.



KAYNAK

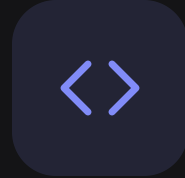
Bir parçanın kaynak profili sistemin geri kalanını rehin aldığı anda.



ÇÖKÜŞ

Bir bileşenin çöküşü düzenli olarak tüm sistemi düşürdüğünde.

Karmaşıklık bütçesi sınırlı



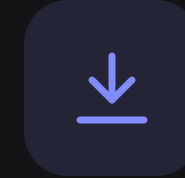
PHP + Laravel

ya da Symfony; uygulama katmanı.



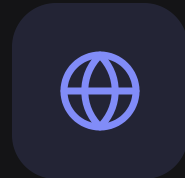
PostgreSQL

İlişkisel veritabanı.



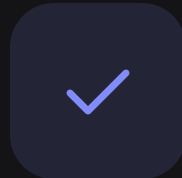
Redis

Bellek içi veri deposu.



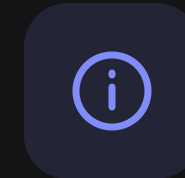
Nginx

Web sunucusu.



Linux

systemd + Supervisor ile süreç yönetimi.



Yeni araç

Benim eşiğim: boring sayılması için 12–18 ay üretimde aktif kullanım.

Teşekkürler

sade.dev

Sınırları erken çizin, dağıtımı geç yapın. Sinyali bekleyin; kararı moda değil, ölçüm versin.

