30.04.2022
Ömer Faruk Birgül

3. Software Architecture

- Karmaşıklığı en aza indirmek yazılım mimarisindeki en önemli hedef olmalıdır,
- Daha karmaşık sistemi çözmesi, sonradan değiştirmesi daha zor ve daha pahalıdır,

Sistem kalite özellikleri

Responsiveness: Sistem, sonucu makul bir sürede veriyor mu? (cevaplanabilirlik)

Reliability: Sistem, kullanıcılar ve geliştiriciler için beklendiği gibi mi davranıyor? (güvenilirlik)

Availability: Sistem, kullanıcılar talep ettiğinde hizmet sağlayabiliyor mu? (kullanılabilirlik)

Security: Sistem kendini izinsiz girişlerden ve saldırılardan koruyabiliyor mu? (güvenlik)

Usability: Sistem kullanıcıların ihtiyacı olan özellikleri hızlıca ve hata vermeden çalıştırıyor mu? (kullanılabilirlik)

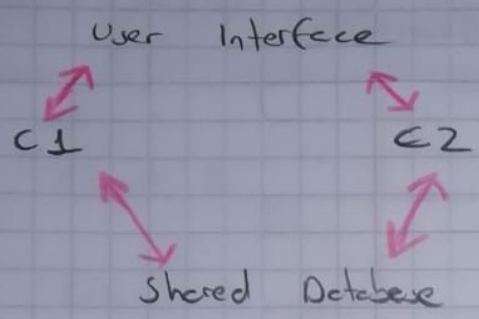
Maintainability: Sistem güncellenebilir ve yeni eklenen özellikleri maliyet artışı olmadan sağlayabiliyor mu? (sürdürebilirlik)

Resilience: Sistem saldırılarda yıkım çekmeden hizmet sağlamaya devam ediyor mu? (dayanıklılık)

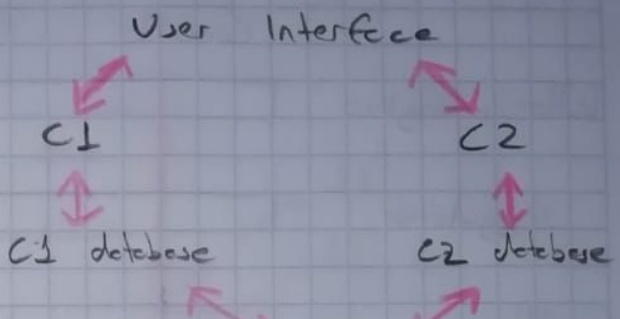
Merkezi Güvenlik Mimarisi

- Merkezi güvenlik mimarisinin faydaları; korume ise etmesi daha kolaydır, ve bilgilere ulaşmak daha etkilidir.
- Eğer tüm bilgileri tek bir yerde depolanırsa ve bu yer ele geçirilirse herşey kaybedilir.

- Eğer bilgiler değiştirilirse ele geçirilme daha zor dur ve maliyet artar.



1



2

- 1. şema'da iki bileşen tek bir veritabanını paylaştığı görülür.

Eğer C1'in yavaş çalıştığını kabul edersek C1'i hızlandırmanın tek yolu veritabanını değiştirmektir, Bunun anlamı C2'yi de değiştirmektir ki bu da cevap süresini etkiler.

- 2. şema'da ise bir bileşen değişmesi gerekse bile bu diğer bileşeni etkilemez.

Trade off:

Bir üründe her özelliği optimize etmek imkansızdır. O yüzden bazı özellikler arasında seçimler yapılır.

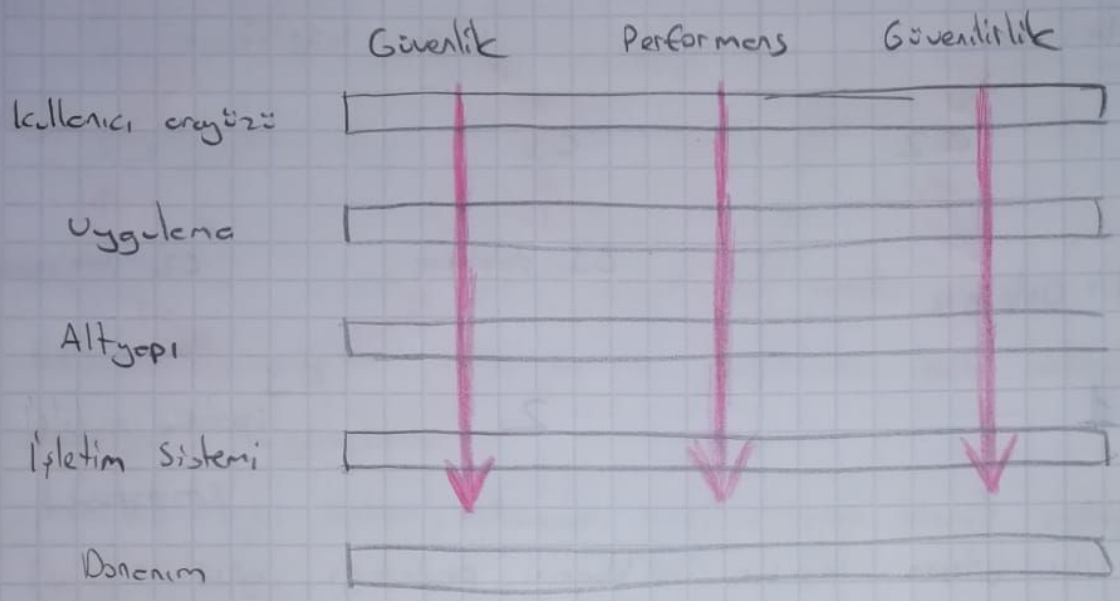
Architectural Complexity

Localize relationship: A ve B bileşenleri arasında ilişki varsa, A ve B'yi aynı modülde tanımlamak daha kolaydır.

Reduce shared dependies: A ve B aynı bir bileşene bağlı ise karmaşıklık artar çünkü bağlı bileşen değişince bunun A ve B'yi nasıl etkileyeceğini hesaplamamız gerekir.

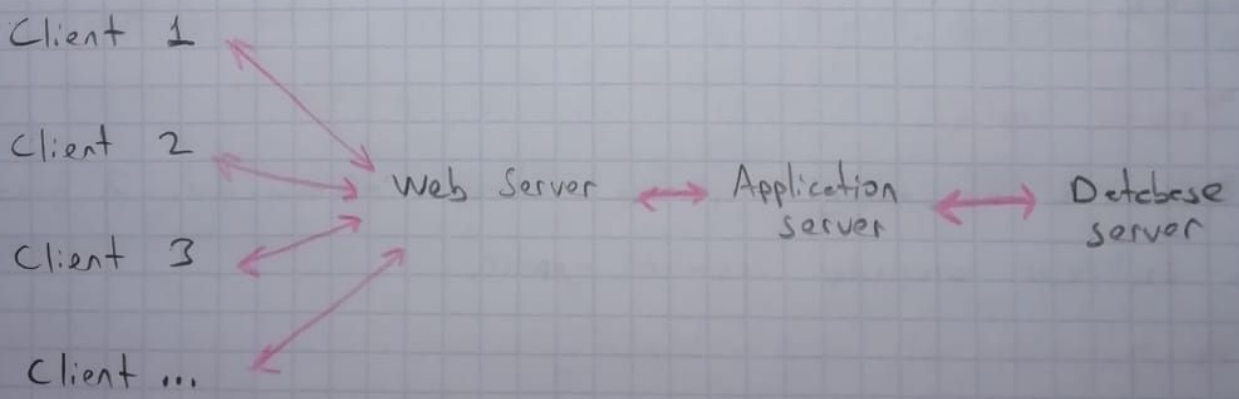
Cross-Cutting

Cross-Cutting endişeler sistemik, tüm sistemi etkileyen endişelerdir.



Client - Server Architecture

Multi-tier Client - Server



Service - oriented

Durum bileşenleridir yani çoğaltılabilir ve bir bilgisayardan ötekine göç edebilir. Talep arttıkça ölçeklendirmek daha kolaydır ve beğenisizliğe karşı direnç yüksektir.