Java EE Tasarım Örüntüleri (J2EE Design Patterns)

harezmi bilişim çözümleri

J2EE Tasarım Örüntüleri

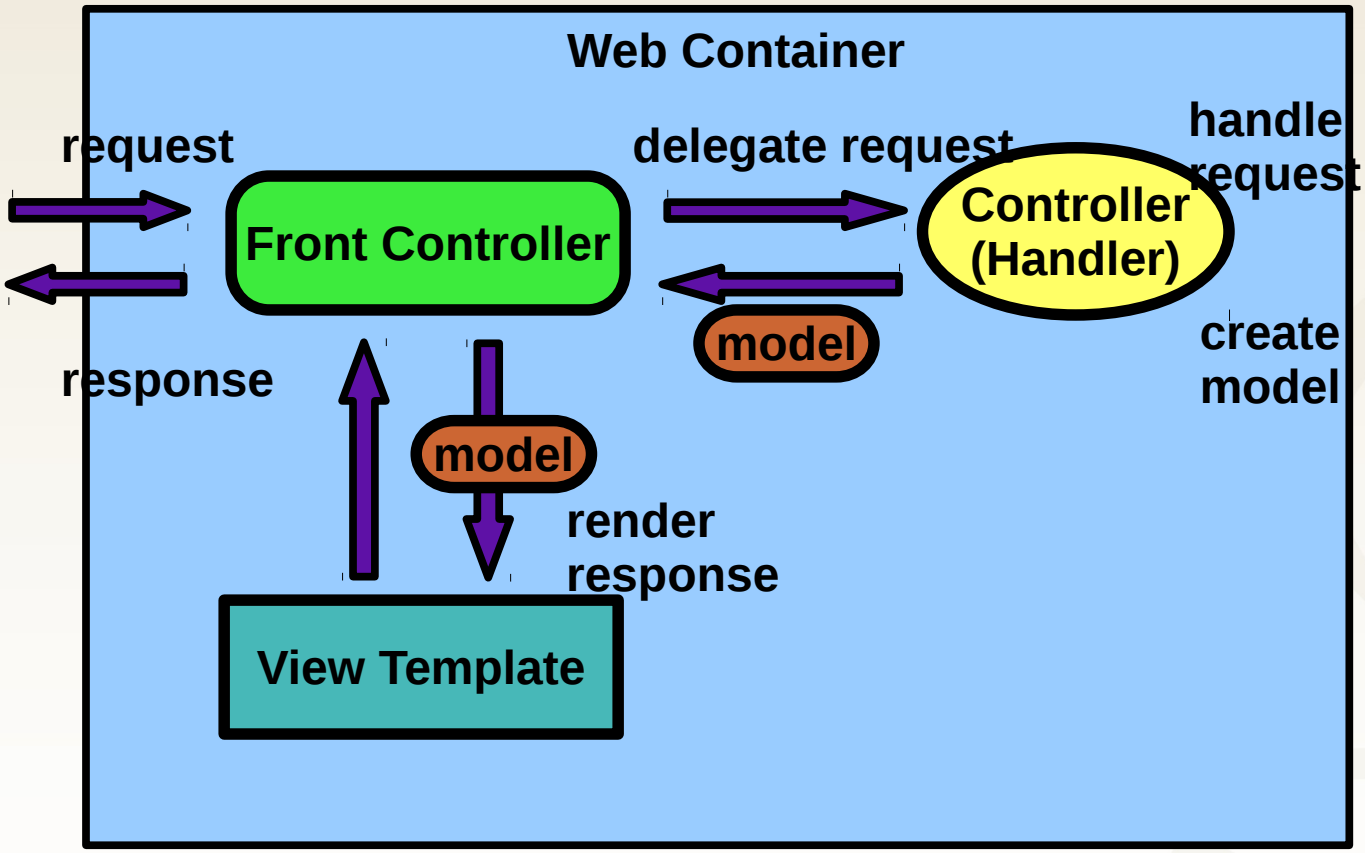
- **Core J2EE Design Patterns** (Alur Kurpi Malks) kitabı bu çalışmalara öncülük etmiştir
- 2000'lerin başında J2EE teknolojileri ile geliştirilen projelerdeki iyi pratikleri ve değişik tasarım örüntülerini kataloglamaya çalışmışlardır
- Diğer bir öncü kitap ise **EJB Design Patterns** (Marinescu) olmuştur
- Burada da EJB'lere odaklanılmış, özellikle www.theserverside.com sitesi hayata geçirilirken elde edilen tecrübeler ışığında bir takım örüntüler kataloglanmıştır

Front Controller

Web uygulamasına gelen bütün requestler Front Controller tarafından ele alınır

Bu sayede uygulamanın ihtiyaç duyduğu güvenlik, internationalization, view render Gibi işlemler tek ve merkezi bir noktadan halledilmiş olur

Front Controller



Front Controller genellikle bir Servlet olur, uygulamaya gelen bütün request'leri handle eder

Request'in hangi komuta karşılık geldiğini çözümlemesi için request'in path'inden, parametrelerinden vs yararlanılır

Data Access Object (DAO)

Mimarisel bir örüntüdür

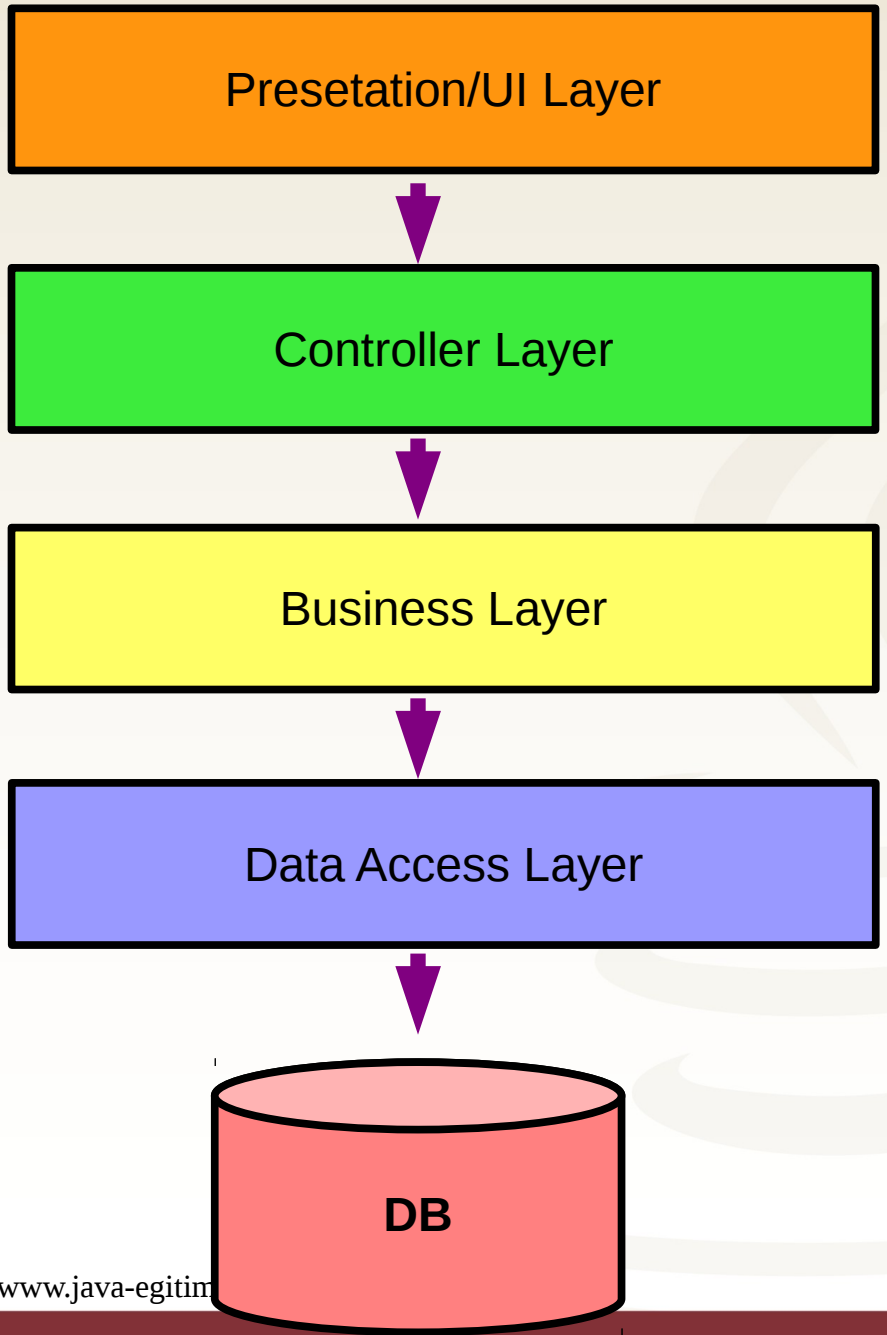
Veri erişim işlemlerini uygulamanın
Diğer katmanlarından izole eder

Bu sayede veri erişim yöntemleri
Değiştiğinde diğer kısımlar
etkilenmemiş olacaktır

Veri erişim işlemlerinin gerçekleştiği
Nesneler DAO nesneleri olarak
Adlandırılır

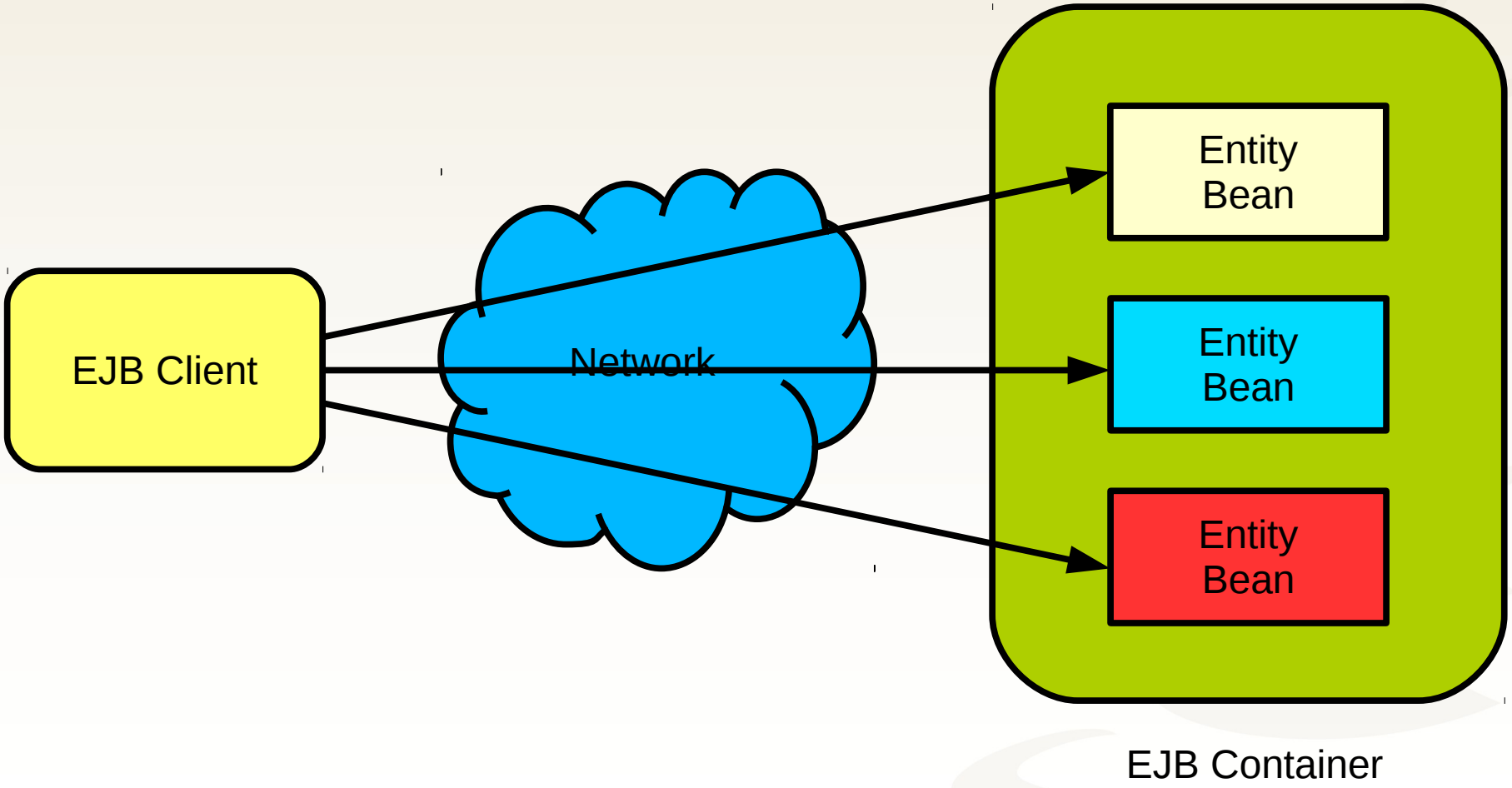
Bir interface implement ederler

Servis katmanı bu interface'e
Depend eder



Session Façade

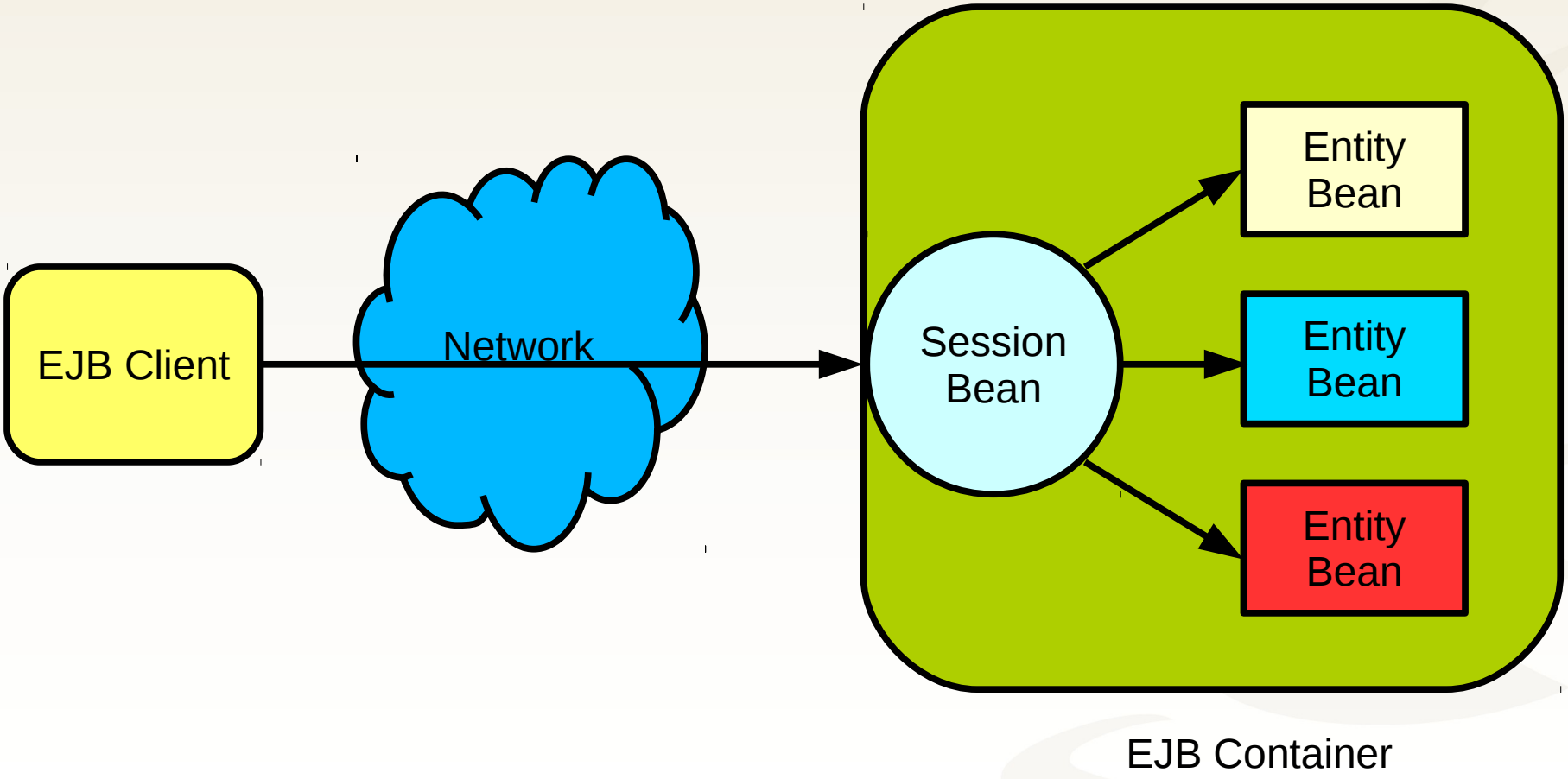
Client'in bir senaryoyu hayata geçirmek için network üzerinden teker teker entity Bean instance'larına erişimesi performans açısından oldukça kötü sonuçlar doğuracaktır



Session Façade

Client senaryoyu işletmek için session bean'a bir metot çağrısı yapar

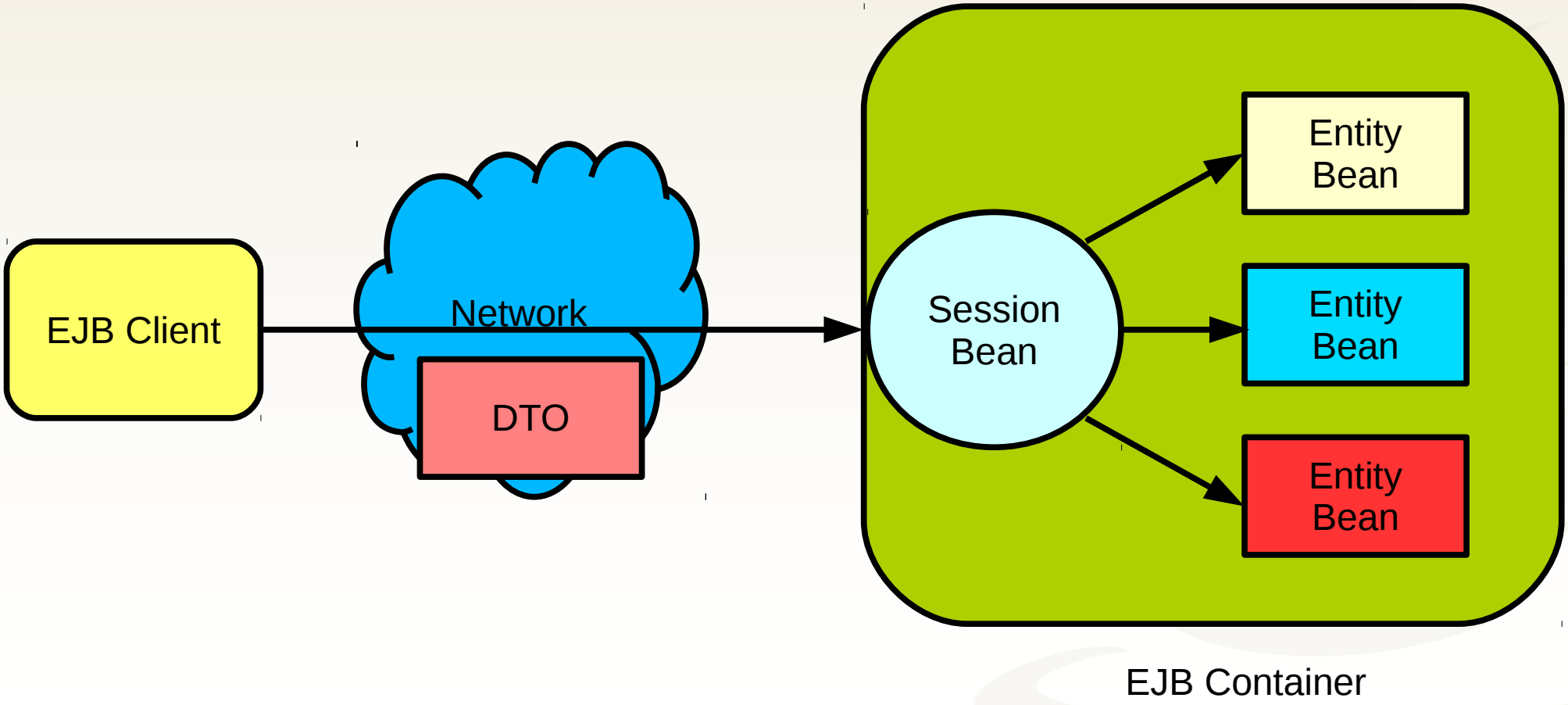
Session Bean arka tarafta farklı entity bean instance'lara erişerek istenen fonksiyonaliteyi Yerine getirir



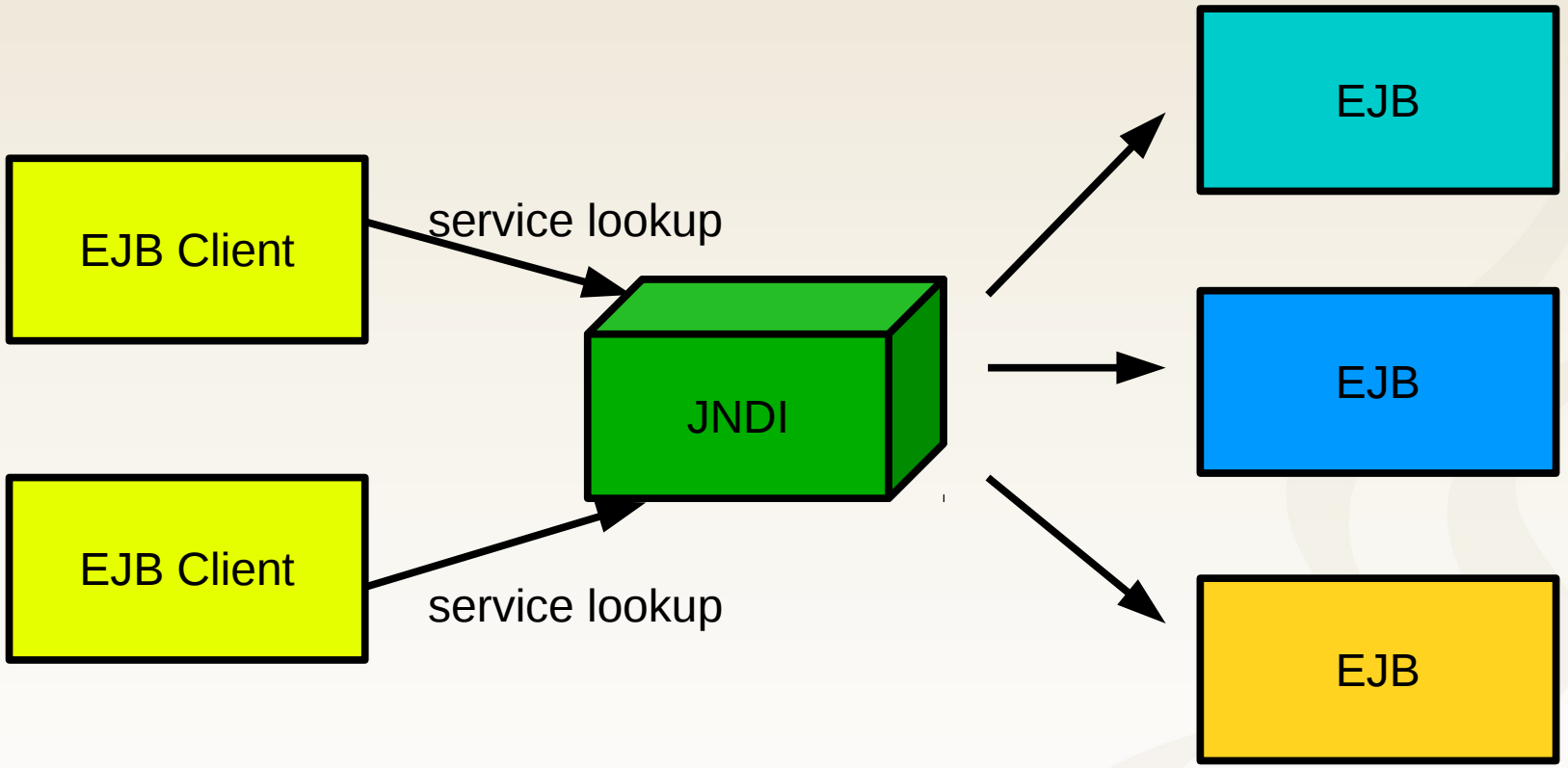
Data Transfer Object (DTO)

Remote çağrıların sayısı ve network üzerinde taşınan parametre sayısı ne kadar çok olursa gerçekleşecek olan serialization/deserialization maliyeti de o kadar çok olacaktır

Bu maliyeti azaltmak için çağrı ve parametre sayısını azaltmak için DTO nesneleri kullanılır



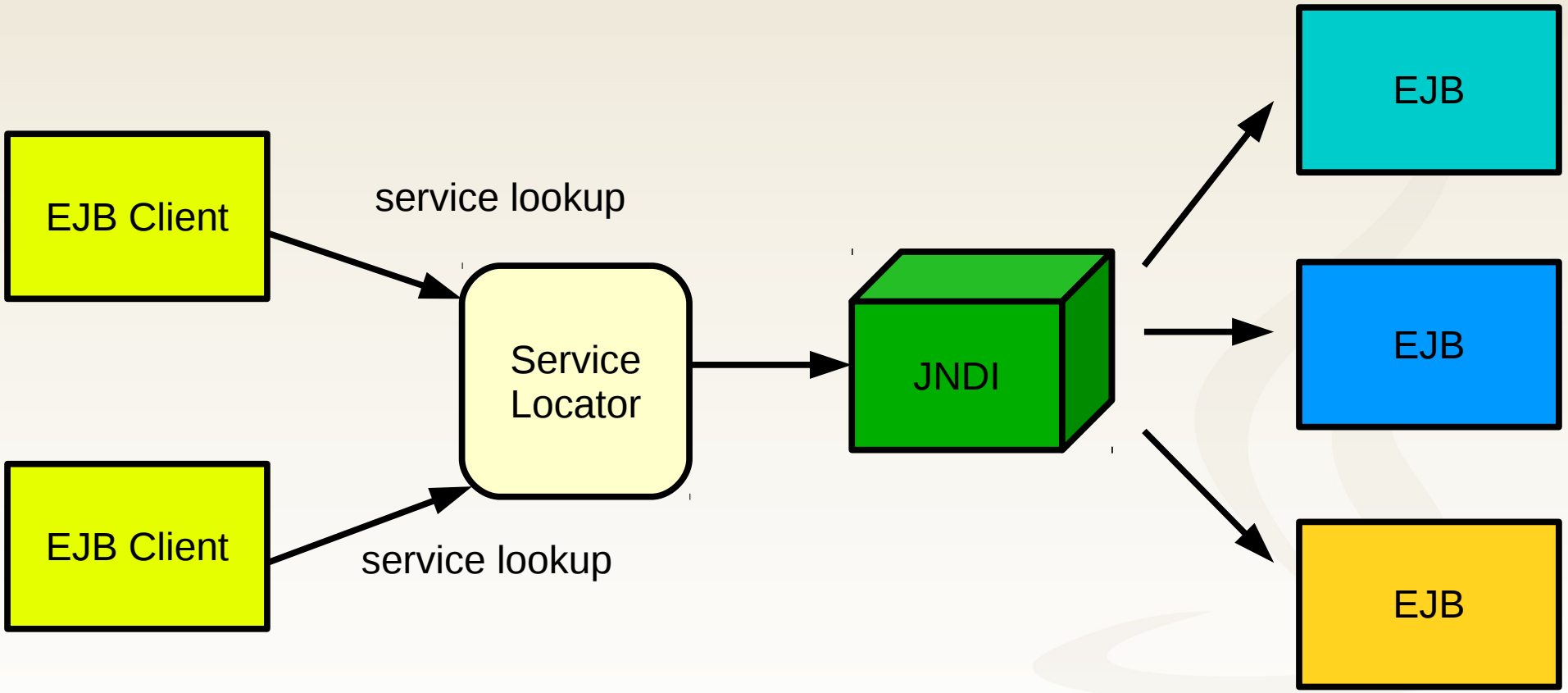
Service Locator



Eğer EJB client'lar her bir servis için lookup işlemlerini kendi kendilerine yaparlarsa Farklı EJB client'ların aynı EJB için birden fazla defa lookup işlemi yapması söz Konusu olabilir

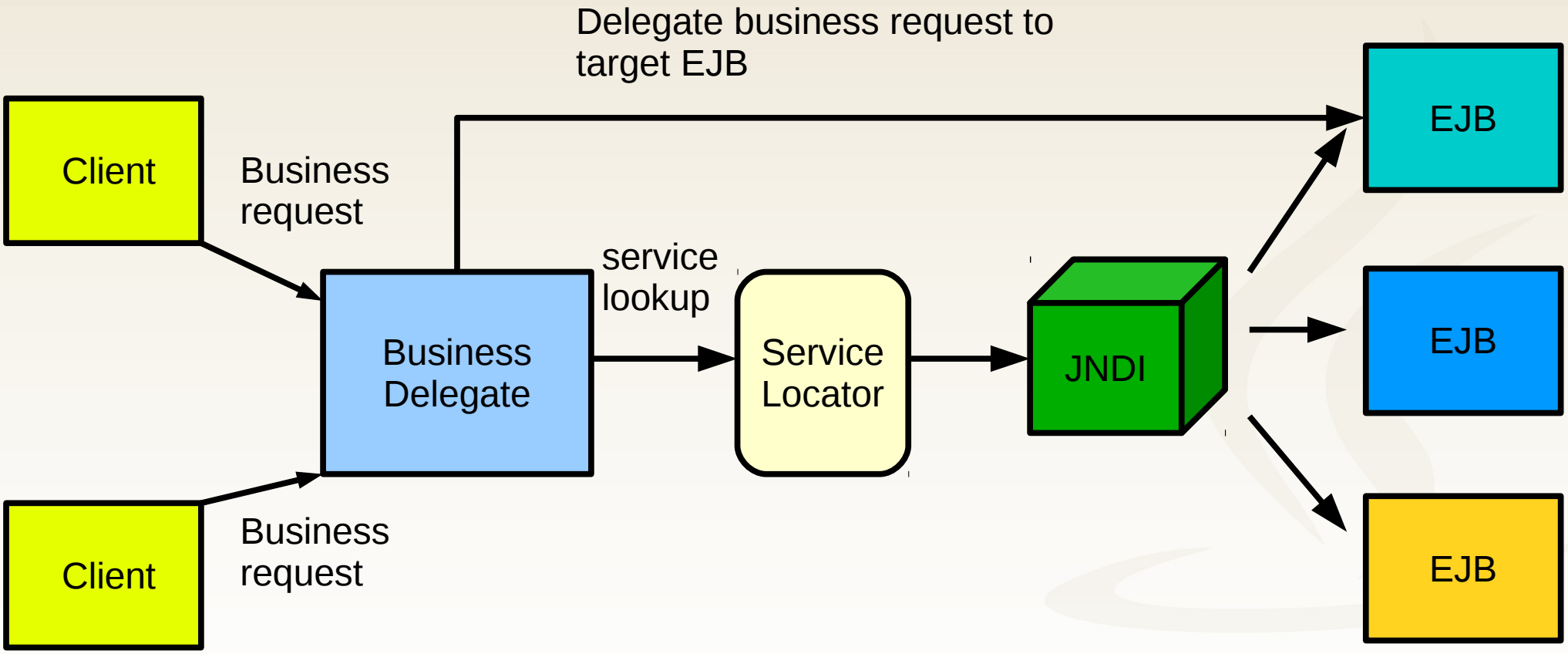
Bu gereksiz yere bir network trafiği ve performans kaybı yaratacaktır

Service Locator



EJB lookup işlemlerinin tek bir nokta üzerinden gerçekleşmesi durumunda aynı Instance'a lookup sonucu dönen nesne cache'lenebilecek, bu sayede gereksiz lookup Işlemleri ortadan kaldırılmış olacaktır

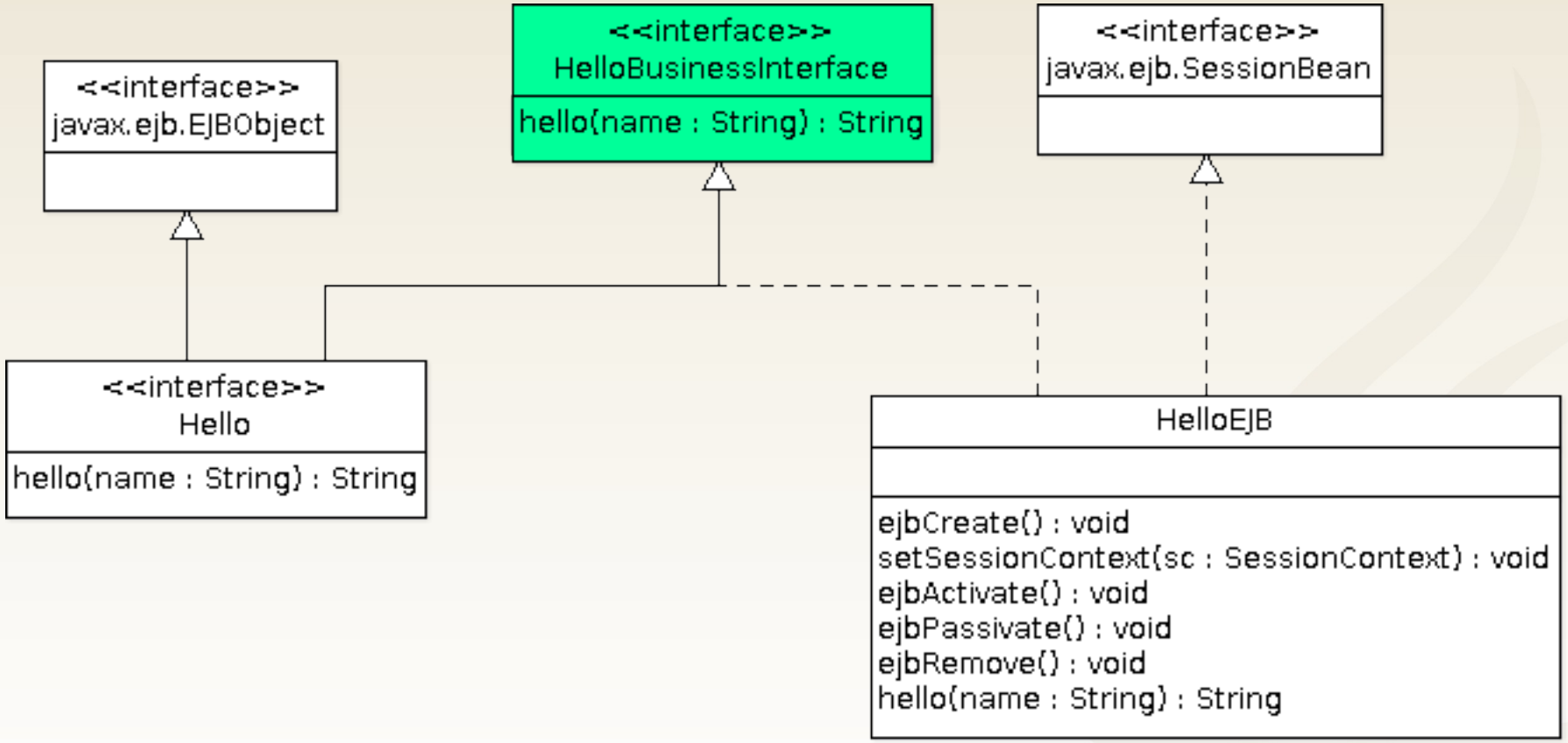
Business Delegate



BusinessDelegate'in amacı client'ı EJB teknolojisinden tamemen soyutlamaktır

Bu sayede business fonksiyonalitesi EJB dışında bir teknoloji ile implement edilecek olursa Bundan client'ın herhangi bir şekilde haberi olmasına gerek kalmayacaktır

Business Interface



EJBObject'in ve EJB sınıfının aynı metot tanımlarına sahip olmaları gerekir

Ancak EJB sınıfının EJBObject'den extend etmesine imkan yoktur

BusinessInterface ile iki tarafta da olması gereken metotlar tanımlanır, bu sayede business Metotlarında bir değişiklik yapıldığında EJBObject ve EJB sınıfının birbirleri ile sürekli Uyumlu olmaları sağlanır

İletişim

www.harezmi.com.tr

www.java-egitimleri.com

info@harezmi.com.tr

info@java-egitimleri.com

@HarezmiBilisim

@JavaEgitimleri