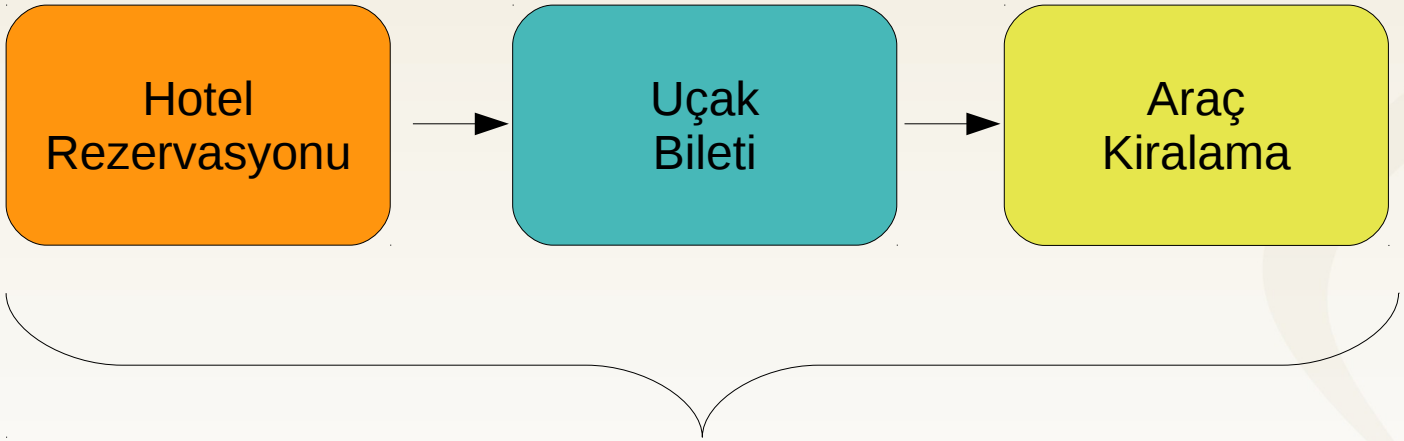
Conversation Senaryoları

CRUD Senaryolar

CRUD senaryolarda her işlem ayrı bir UOW'de gerçekleşir
UOW = TX

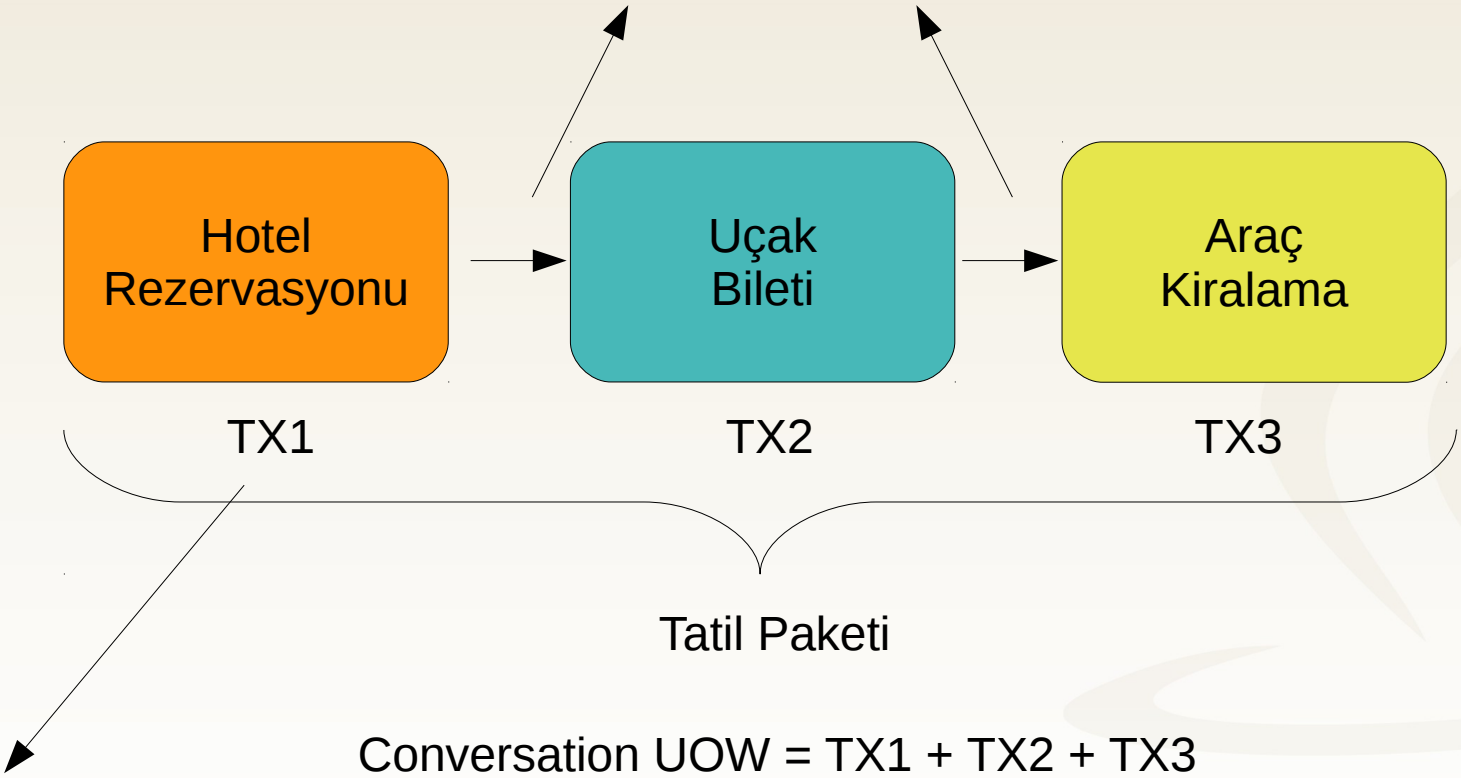
Conversational Senaryolar



Conversation UOW = TX1 + TX2 + TX3

Conversational Senaryolar

Kullanıcıların iş adımları arasındaki **düşünme süreleri uzun** olabilir. Fakat DB tx'ler de mümkün olduğunca **kısa** tutulmalıdır

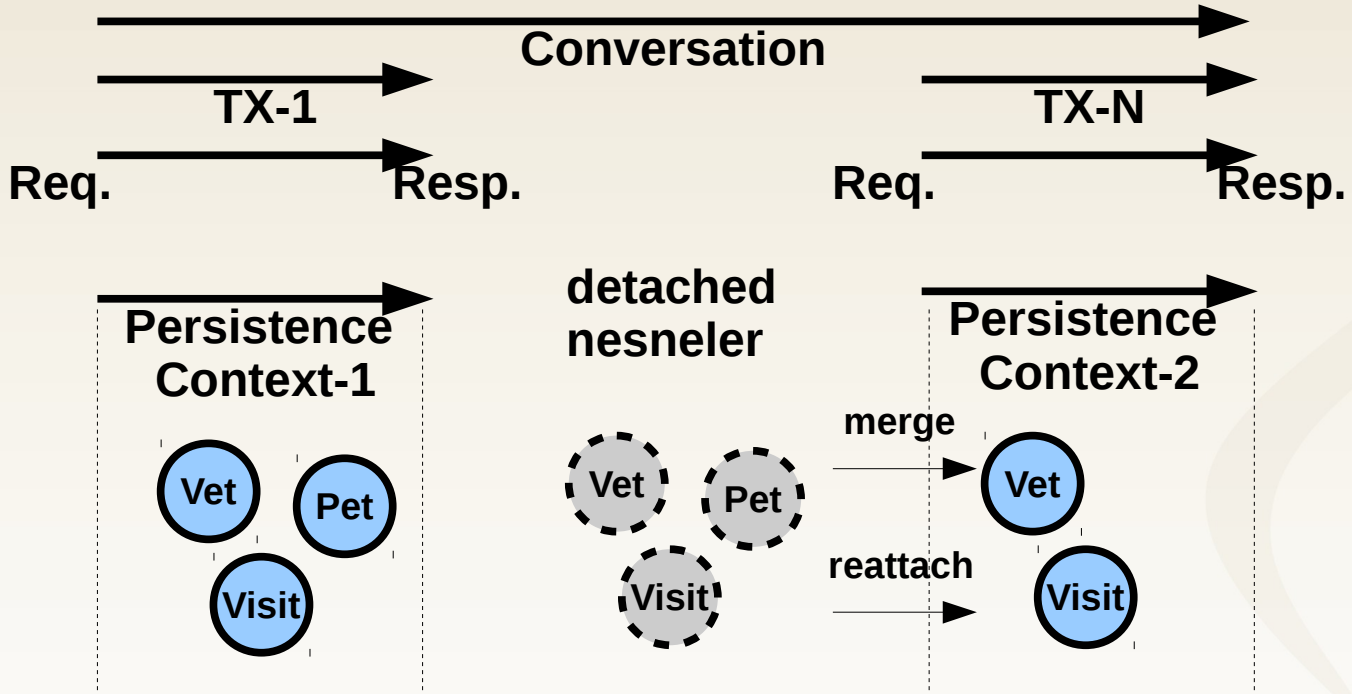


- Herhangi bir anda kullanıcı conversation'ı **yarıda kesebilir**
- Bu durumda o ana kadar yapılmış **değişiklikler rollback edilmelidir**

Conversational Senaryolar

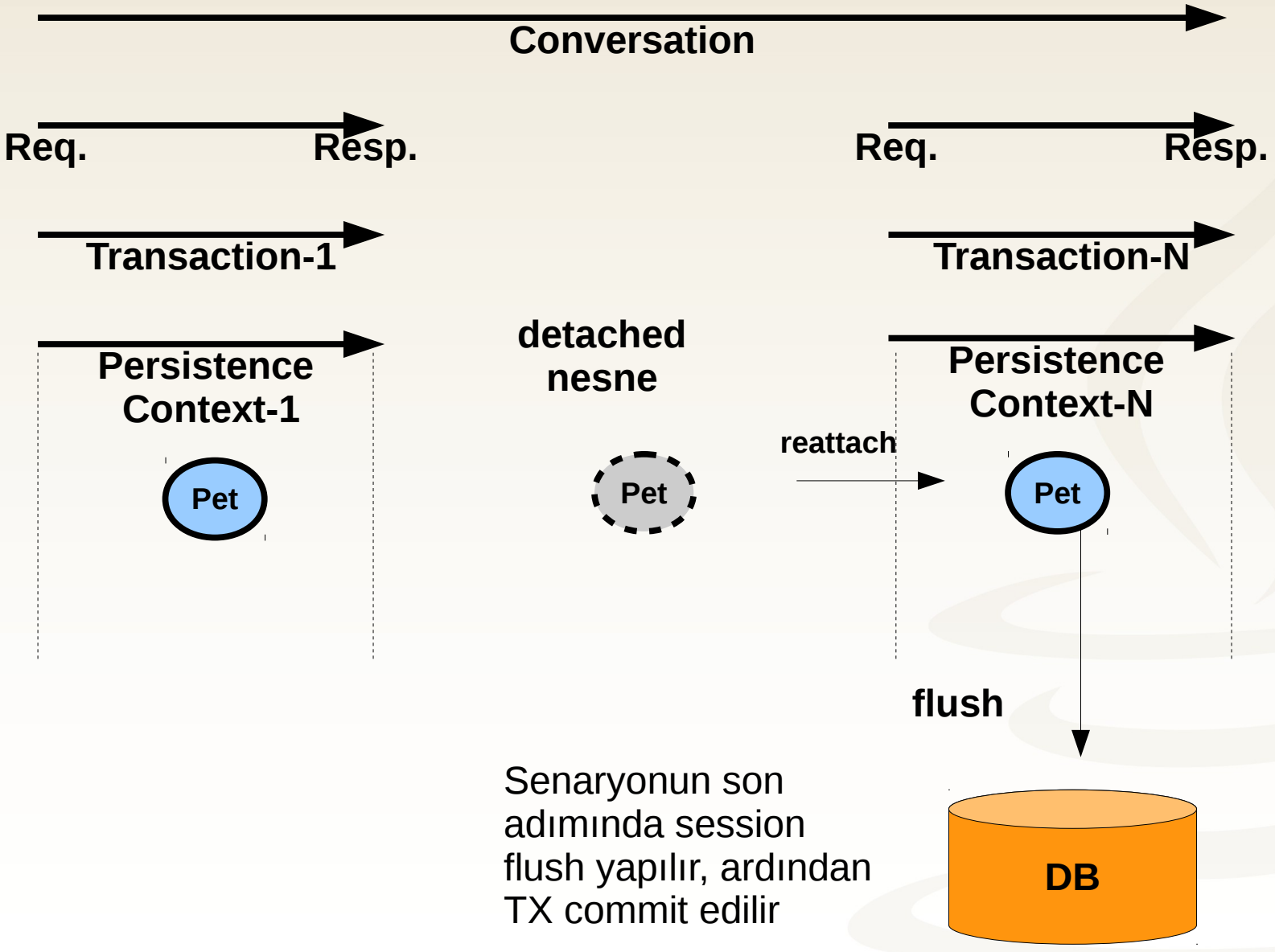
- Conversational senaryoları implement etmek için **iki yaklaşım** söz konusu
  - Detached nesneler ile
  - Persistence context'in ömrünü uzun tutarak
- Her ikisinin de kendine özgü **artı ve eksileri** var

Detached Nesneler İle Conversation



- Kullanıcı düşünme süresi boyunca nesneler **detached** state'dedir
- Bir sonraki adımda nesneler Session'a **reattach** veya **merge** yapılır

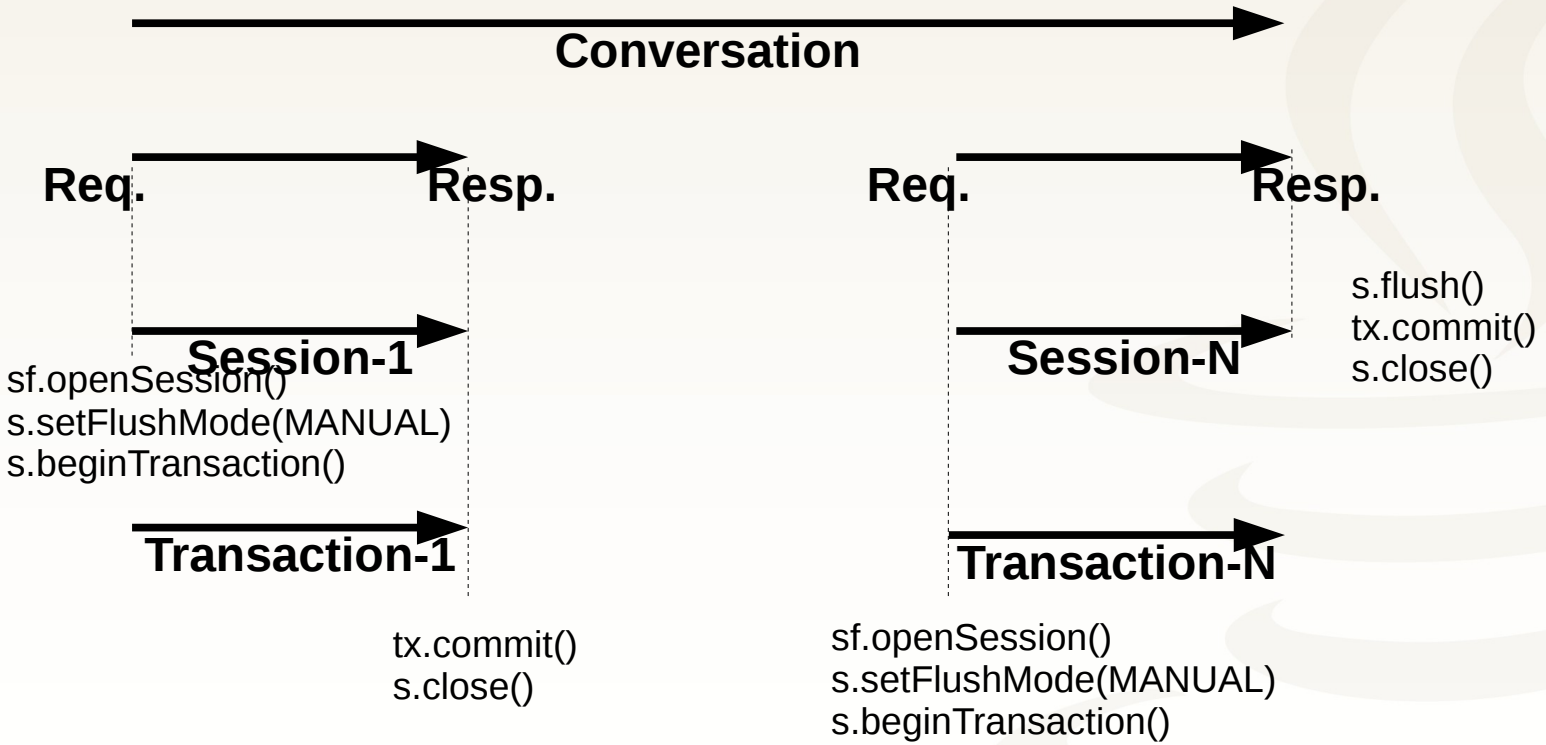
Detached Nesneler İle Conversation



Senaryonun son adımında session flush yapılır, ardından TX commit edilir

Threaded Session Yönetimi

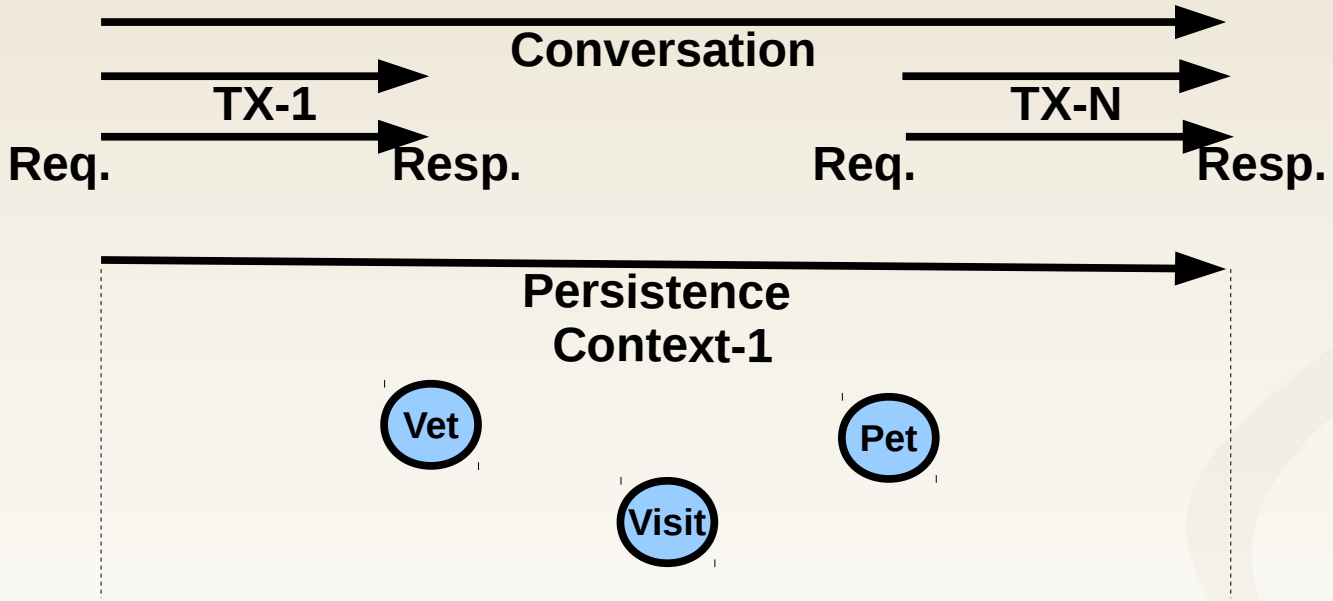
- Request boyunca aynı Session'a erişim sağlamak için **ThreadLocal değişken** üzerinden contextual session kabiliyeti kullanılmaktadır



Threaded Session Yönetimi

- SessionFactory.**getCurrentSession()** çağrısı ile yeni bir Session açılarak ThreadLocal değişkene konur
- Müteakip getCurrentSession() çağrıları ThreadLocal değişkendeki **aynı session'ı** döner
- TX **commit/rollback** ile bu **Session kapatılır** ve Thread'den de **unbind** edilir
- Bu kabiliyetin devreye girmesi için **hibernate.current_session_context_class** attribute değeri **thread** olmalıdır

Extended Persistence Context İle Conversation

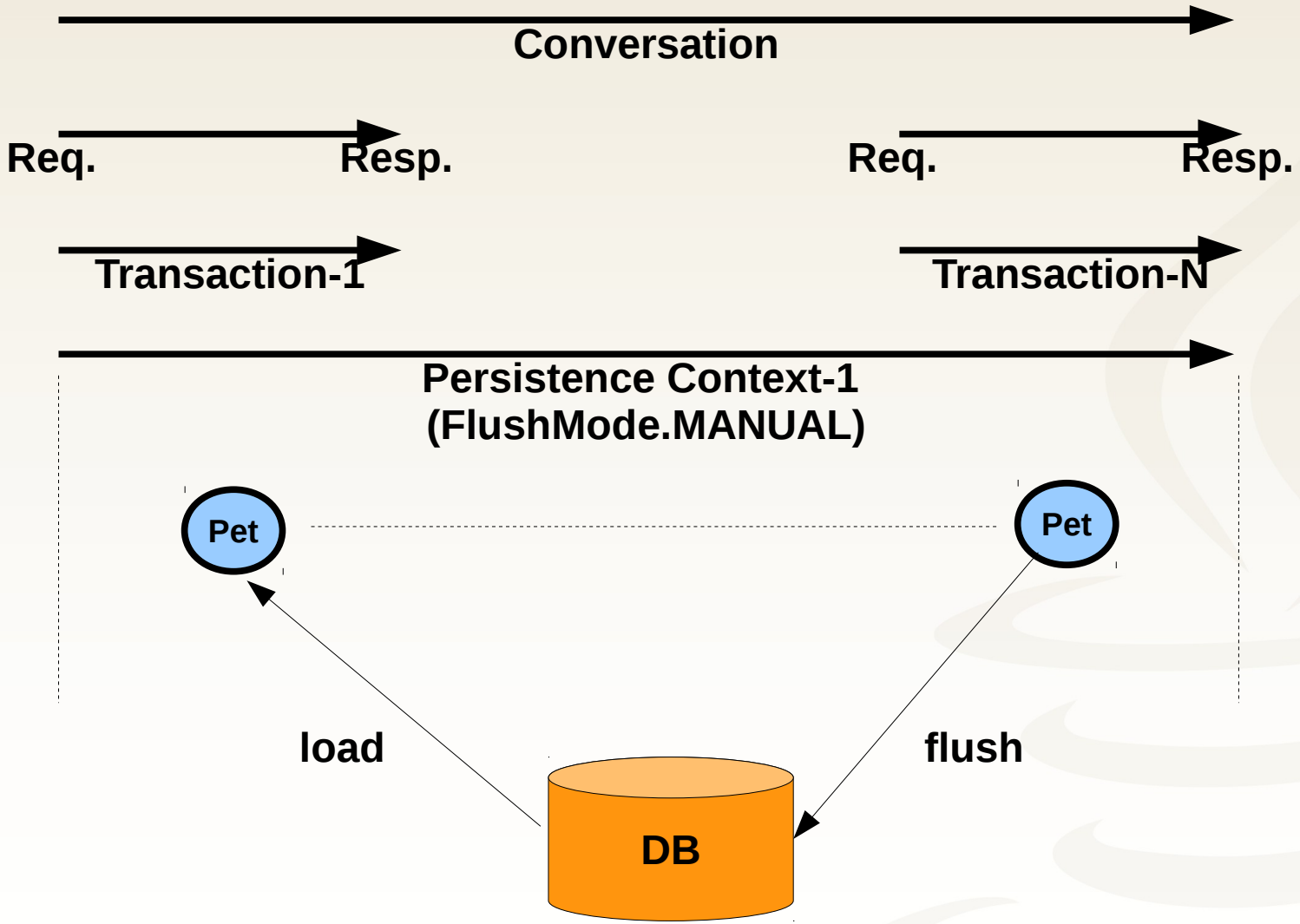


- **Persistence context** bütün UOW boyunca aktiftir
- **session-per-conversation** pattern olarak da bilinir

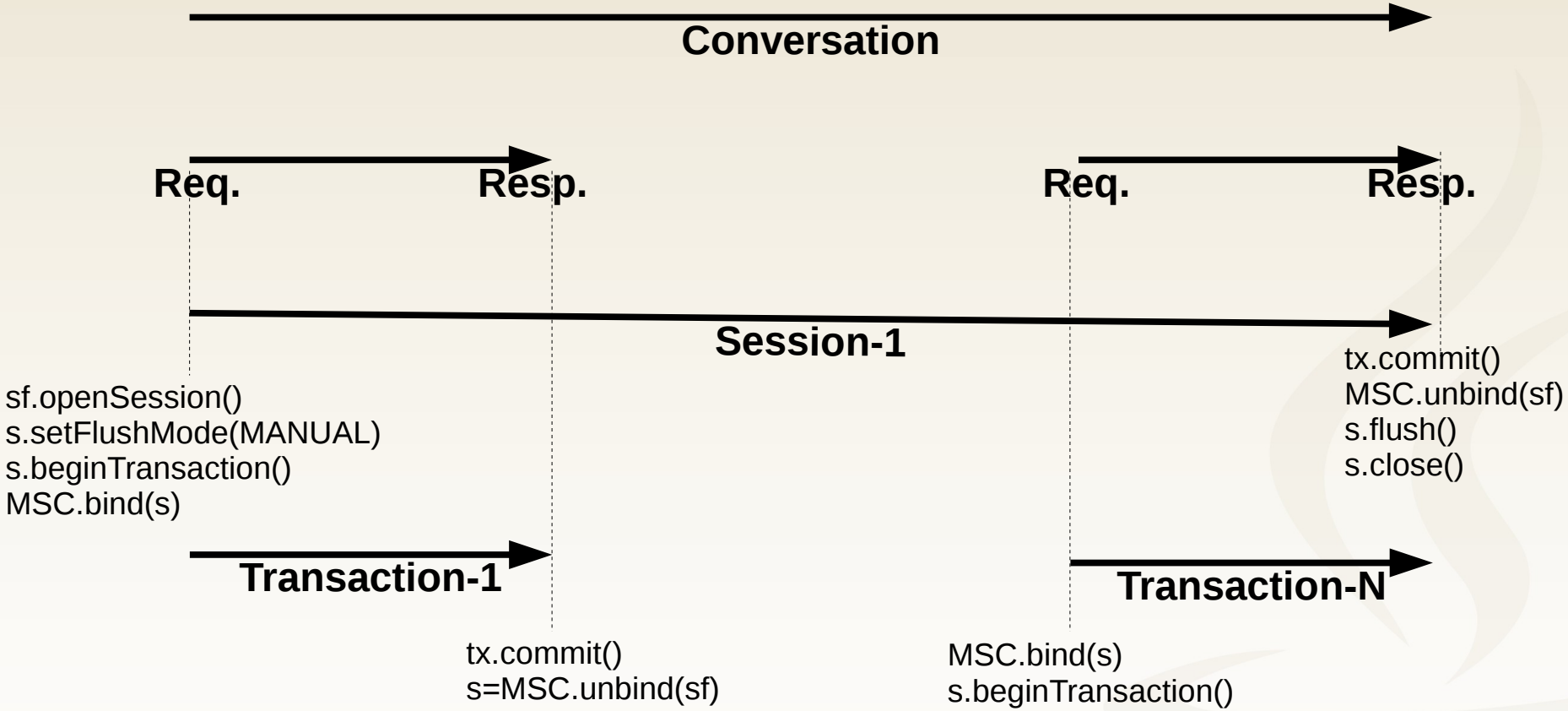
Extended Persistence Context İle Conversation

- Conversation boyunca **tek bir persistence context** ile çalışılmaktadır
- Ancak kullanıcı düşünme süresi boyunca **DB bağlantısı** söz konusu değildir
- Bir sonraki işlem zamanında gerçekten **ihtiyaç olduğunda DB bağlantısı** tekrar kurulur
- **DB senkronizasyonu** (session flush) conversation sonunda yapılır

Extended Persistence Context İle Conversation



Managed (Extended) Session Yönetimi



- **hibernate.current_session_context_class** konfigürasyon property değeri **managed** olmalıdır
- Request boyunca Session **ManagedSessionContext** içerisinde tutulur, request'ler arasında ise kullanıcı oturumunda saklanabilir (Örneğin Http Session)
- Session açıldıktan sonra **sf.getCurrentSession()** thread'e bind edilen Session nesnesini döner

İletişim

- Harezmi Bilişim Çözümleri
- Kurumsal Java Eğitimleri
- http://www.java-egitimleri.com
- info@java-egitimleri.com