SPRING TRANSACTIONS

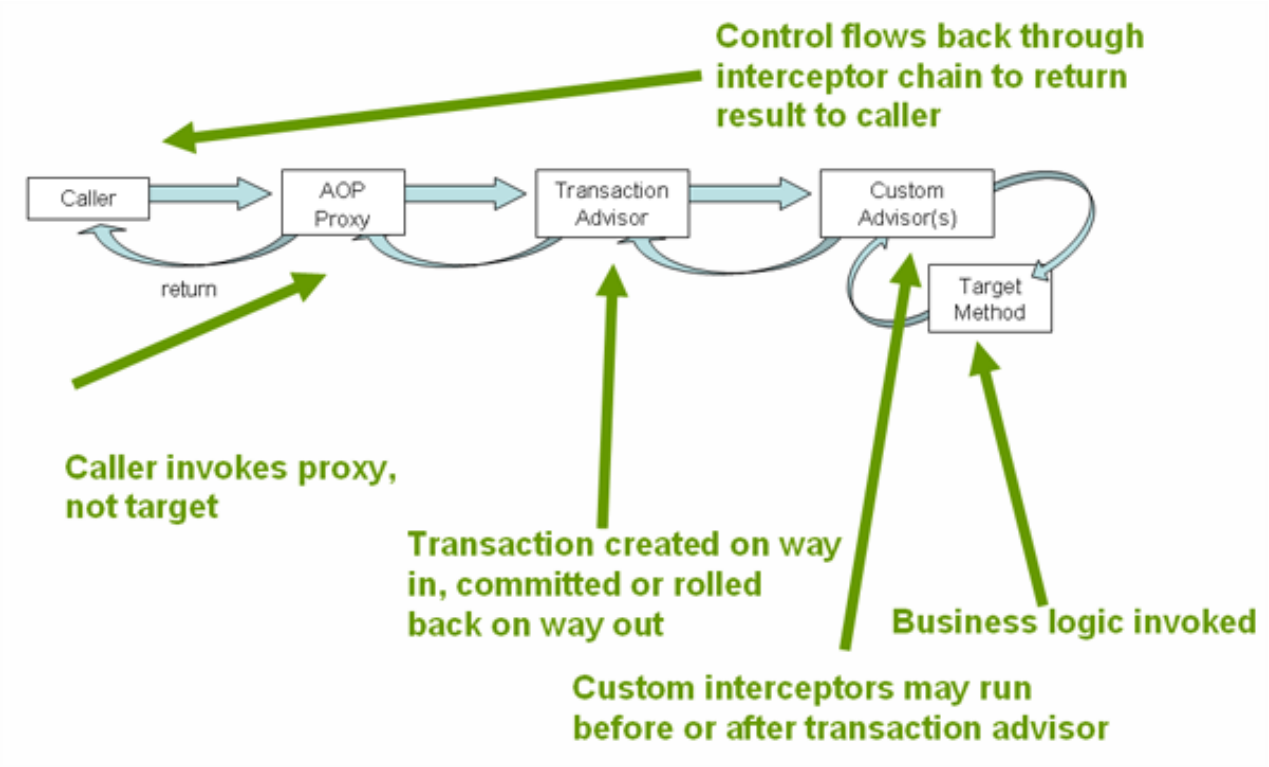
DECLARATIVE TRANSACTION MANAGEMENT

MARCH, 2015

CMT vs Spring Declarative Transactions

	CMT	Spring Transactions
Tied to specific transactions model	Yes	No
Types of classes supported	EJB	Any
Custom rollback behavior	No	Yes, using AOP
Transaction propagation for remote calls	Yes	No

Transaction management flow



Declarative transactions configuration

```
<tx:advice id="txAdvice" transaction-manager="txManager">
    <tx:attributes>
      <!-- all methods starting with 'get' are read-only -->
      <tx:method name="get*" read-only="true"/>
      <!-- other methods use the default transaction settings -->
      <tx:method name="*"/>
    </tx:attributes>
  </tx:advice>

  <!-- ensure that the above transactional advice runs for any
       execution
       of an operation defined by the FooService interface -->
  <aop:config>
  <aop:pointcut id="fooServiceOperation"
          expression="execution(* x.y.service.FooService.*(..))"/>
  <aop:advisor advice-ref="txAdvice"
                            pointcut-ref="fooServiceOperation"/>
  </aop:config>
```

Customizing <tx:method/>

Attribute	Required?	Default
name	Yes	
propagation	No	REQUIRED
isolation	No	DEFAULT
timeout	No	-1
read-only	No	false
rollback-for	No	
no-rollback-for	No	

Annotation based transaction management

```
<bean id="fooService" class="x.y.service.DefaultFooService"/>

  <!-- enable transactional behavior based on annotations -->
  <tx:annotation-driven transaction-manager="txManager"/>

  <!-- a PlatformTransactionManager is still required -->
  <bean id="txManager“
   class="org.springframework.jdbc.datasource.DataSourceTransactionManager">
    <property name="dataSource" ref="dataSource"/>
  </bean>
```

```
@Transactional(readOnly = true)
public class DefaultFooService implements FooService {

  // these settings have precedence for this method
  @Transactional(readOnly = false, propagation = Propagation.REQUIRES_NEW)
  public void updateFoo(Foo foo) {
    // do something
  }
}
```

Reuse common configuration

```
@Target({ElementType.METHOD, ElementType.TYPE})
@Retention(RetentionPolicy.RUNTIME)
@Transactional(value = “orderTxManager“, readOnly=false)
public @interface OrderTx {
}

class TransactionalService {
    @OrderTx
    public void setSomething(String name) { ... }
}
```

<epam>

THANK YOU!

MAKSYM_GOVORISCHEV@EPAM.COM

MARCH, 2015