* Putem sa avem nevoie sa folosim un bean de tip Service unde prin clase ce nu sunt de tip @Component, de ex in metoda main(), si daca vom incerca sa folosim beanul de tip Service, vom primi eroare ca beanul este de tip Proxy, desi clasa noastra Service se cheama cu totul altfel.
* In main() nu vom putea folosi acest bean nicidecum, de aceea unica solutie este de a crea un bean cu @Component si acolo sa cream un obj de tip Service cu @Autowired si mai jos de @Component sa punem:

@EnableTransactionManagement(proxyTargetClass = true)

@Component("utility")  
@EnableTransactionManagement(proxyTargetClass = true)  
public class Utility {  
 @Autowired  
 private ServiceImpl service;  
  
 public ServiceImpl getService() {  
 return service;  
 }  
  
 public void setService(ServiceImpl service) {  
 this.service = service;  
 }  
}

public class Test {  
 public static void main(String[] args){  
  
 ClassPathXmlApplicationContext context = new ClassPathXmlApplicationContext("applicationContext.xml");  
 Utility utility = context.getBean("utility",Utility.class);  
  
 Teacher student = utility.getService().getTeacher(11);  
 System.*out*.println(student);  
 }  
}

Daca vom incerca sa obtinem beanul de tip Service in main si sa il folosim, nu vom putea, caci doar @Service,@Repositry si @Controller pot folosi transaction manager sau @Component cu acea anotatie de mai sus. Altfel nicidecum.