

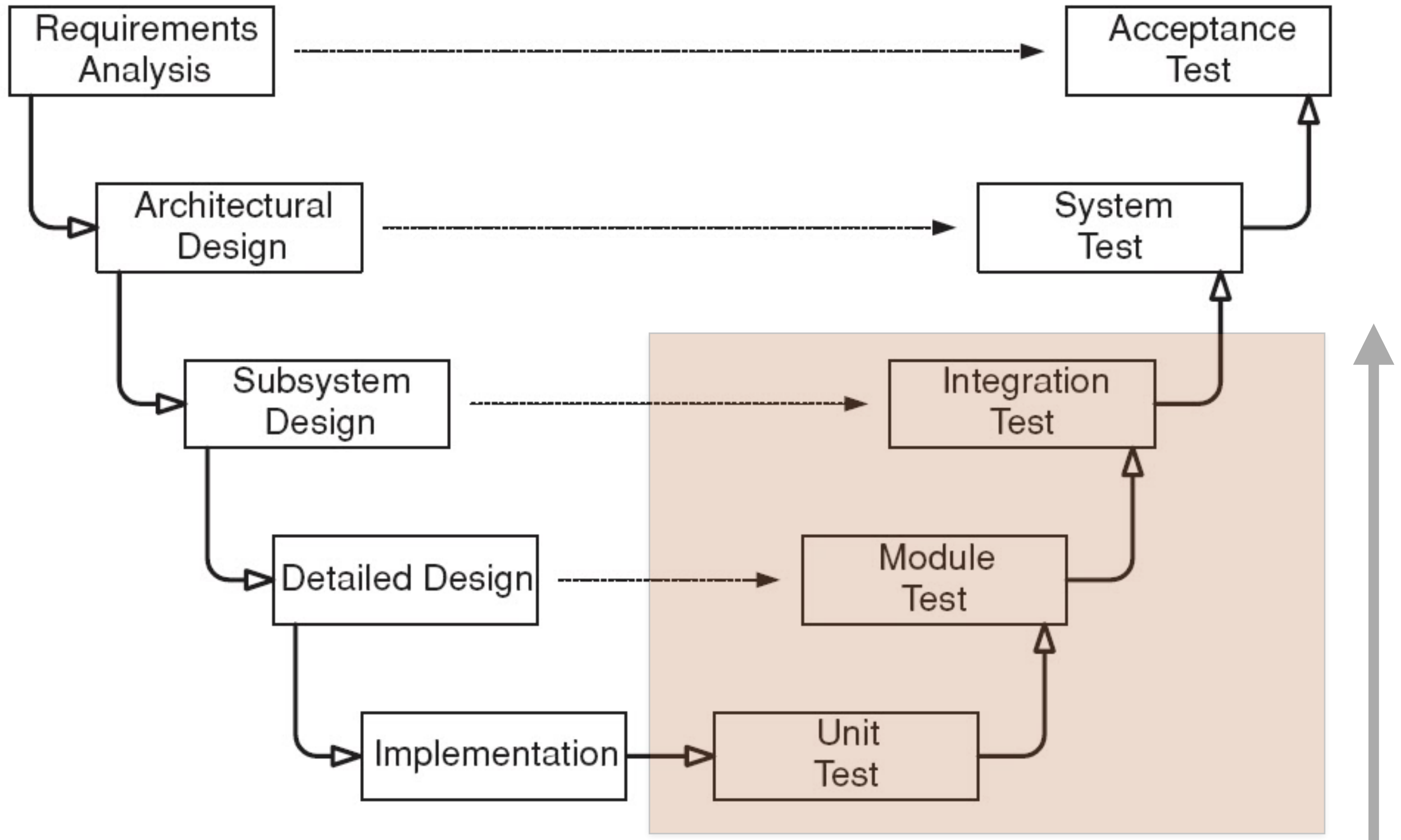
Spring Testing

Andrea Caracciolo
scg.unibe.ch/staff/caracciolo



^b
**UNIVERSITÄT
BERN**

Bottom-up approach



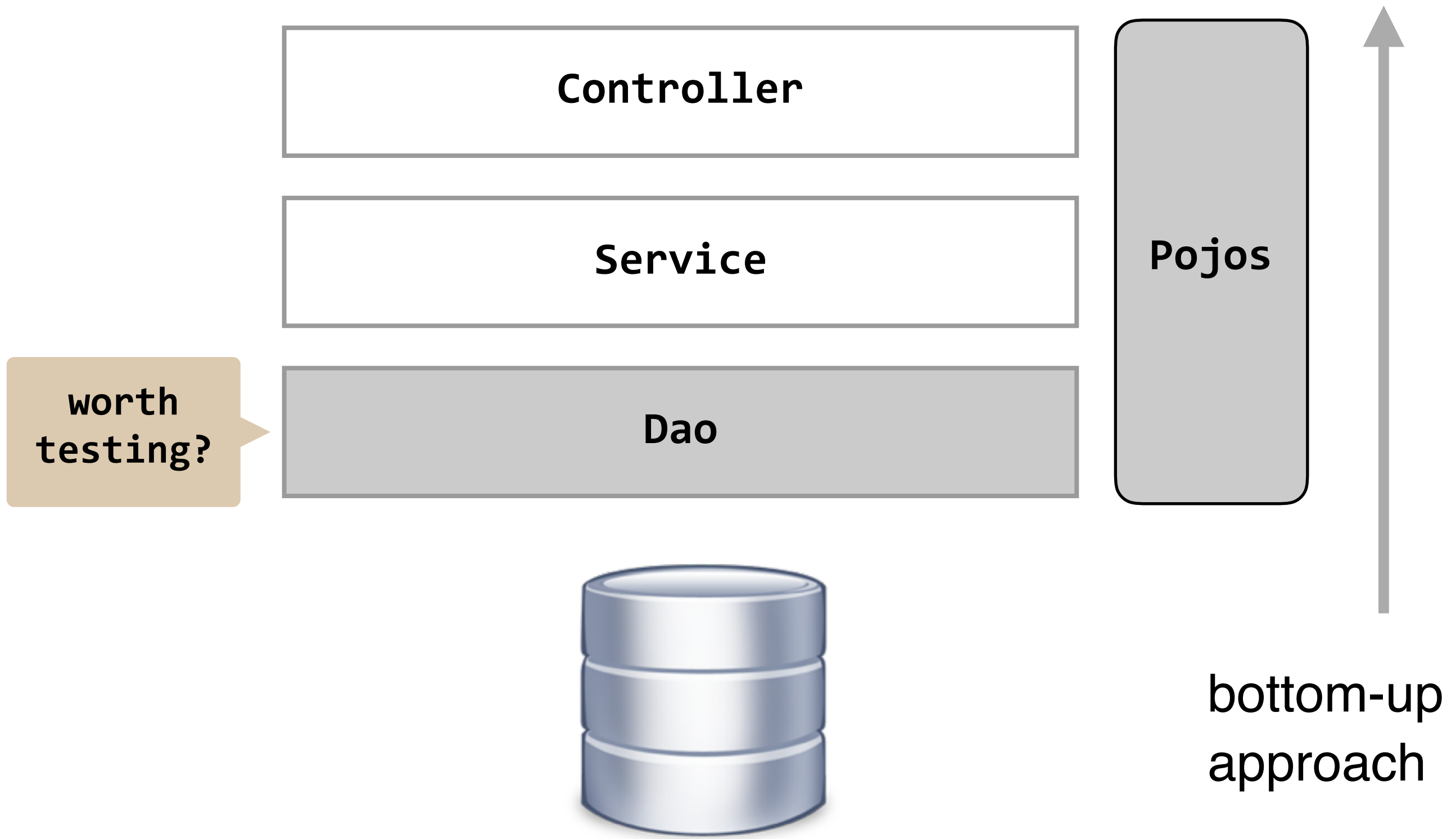
Controller

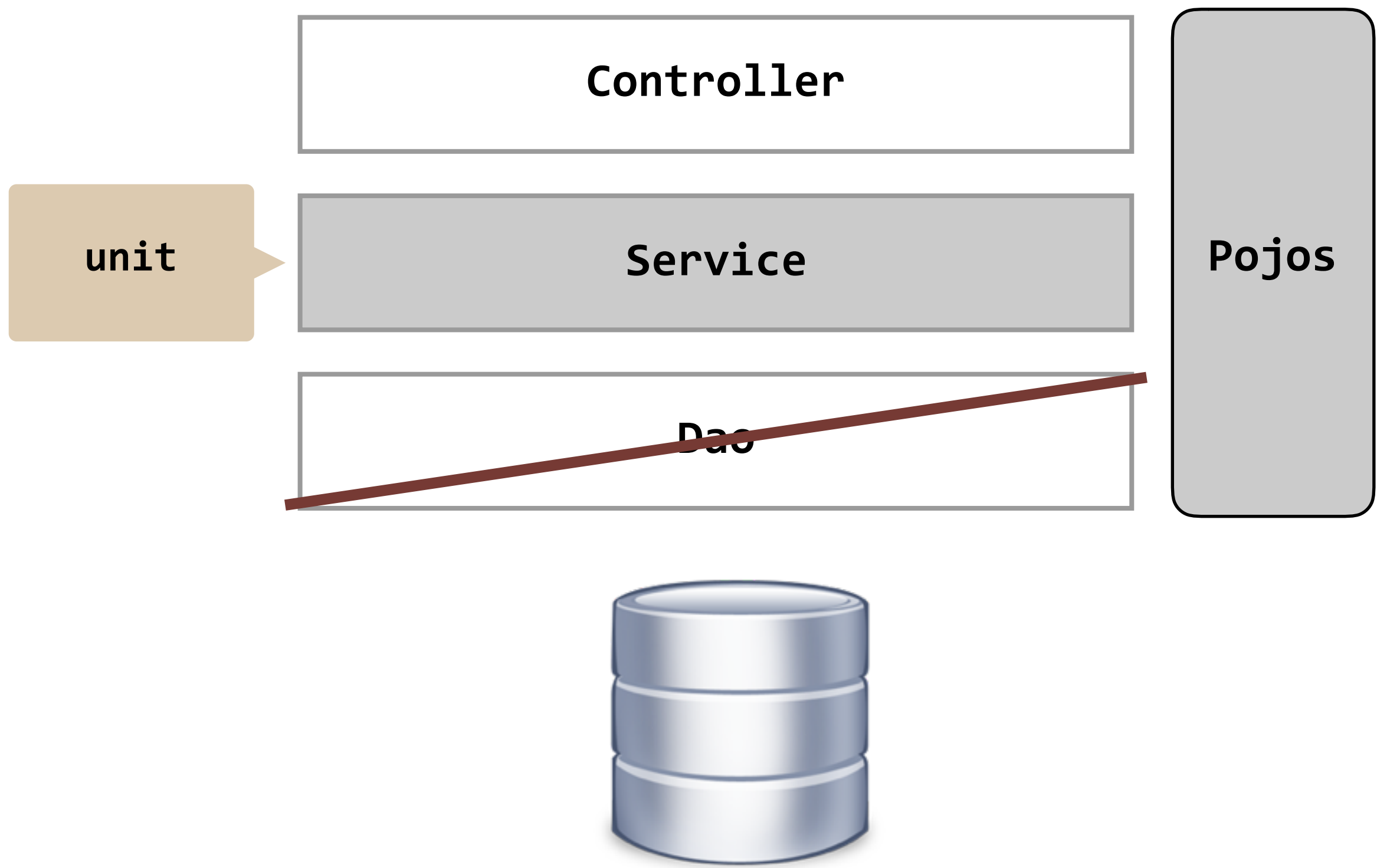
Service

Dao

Pojos







Test

```
@RunWith(SpringJUnit4ClassRunner.class)
@ContextConfiguration(locations={"file:src/main/webapp/WEB-INF/test.xml"})
public class ServiceTest {

    @Autowired private SampleService sampleService;
    @Autowired private UserDao userDao;

    @BeforeClass
    public static void oneTimeSetUp() {
        ..
    }

    @Test
    public void testSaveForm() {
        ..
    }
}
```

Test

```
@RunWith(SpringJUnit4ClassRunner.class)
@ContextConfiguration(locations={"file:src/main/webapp/WEB-INF/test.xml"})
public class ServiceTest {

    @Autowired private SampleService sampleService;
    @Autowired private UserDao userDao;

    @BeforeClass
    public static void oneTimeSetUp() {
        ..
    }

    @Test
    public void testSaveForm() {
        ..
    }
}
```

**Mock Data
Autowire Service**

Test

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans" ...>

    <!-- MOCK data access classes (DAO) -->
    <bean class="org.mockito.Mockito" factory-method="mock">
        <constructor-arg value="org.sample.model.dao.UserDao"/>
    </bean>
    <bean class="org.mockito.Mockito" factory-method="mock">
        <constructor-arg value="org.sample.model.dao.AddressDao"/>
    </bean>

    <!-- AUTOWIRE services -->
    <bean class="org.sample.controller.service.SampleServiceImpl" />

</beans>
```


Service

@Service

```
public class SampleServiceImpl implements SampleService {
```

```
    @Autowired    UserDao userDao;
```

```
    @Transactional
```

```
    public User saveFrom(SignupForm signupForm) throws InvalidUserException{
```

```
        ...
```

```
        User user = new User();
```

```
        user.setFirstName(signupForm.getFirstName());
```

```
        user.setEmail(signupForm.getEmail());
```

```
        user.setLastName(signupForm.getLastName());
```

```
        user.setAddress(address);
```

```
        user = userDao.save(user);
```

```
        return user;
```

```
    }
```

```
}
```

Test


```
@RunWith(SpringJUnit4ClassRunner.class)
@ContextConfiguration(locations={"file:src/main/webapp/WEB-INF/test.xml"})
public class ServiceTest {

    @Autowired private Service;
    @Autowired private ...

    @BeforeClass
    public static void ...
    {
        ..
    }

    @Test
    public void testSaveForm() {
        ..
    }

}
```



@BeforeClass
@AfterClass
@Before
@After

Service

```
@Service
public class SampleServiceImpl implements SampleService {

    @Autowired    UserDao userDao;

    @Transactional
    public SignupForm saveFrom(SignupForm signupForm) throws InvalidUserException{
        ...

        User user = new User();
        user.setFirstName(signupForm.getFirstName());
        user.setEmail(signupForm.getEmail());
        user.setLastName(signupForm.getLastName());
        user.setAddress(address);

        user = userDao.save(user);

        return user;
    }
}
```

Test

Mock DAO

```
@Test
public void testSaveForm() {
    // GIVEN
    when(userDao.save(any(User.class))).thenReturnFirstArg();

    // WHEN
    SignupForm signupForm = new SignupForm();
    signupForm.setLastName("formLast");
    signupForm.setFirstName("formFirst");
    signupForm.setEmail("form@test.com");

    User user = sampleService.saveFrom(signupForm);

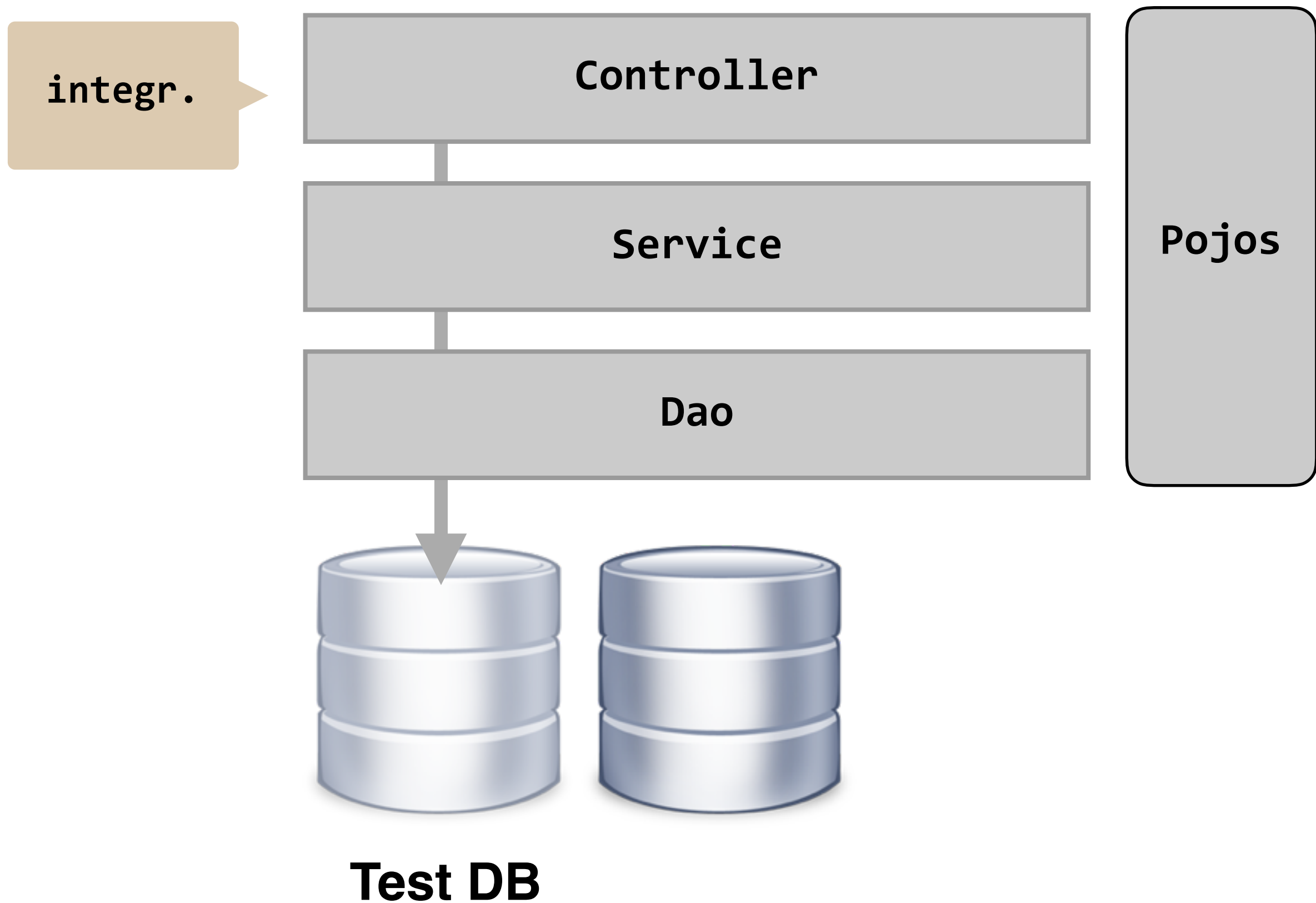
    // THEN
    assertEquals("formLast", user.getLastName());
}
```

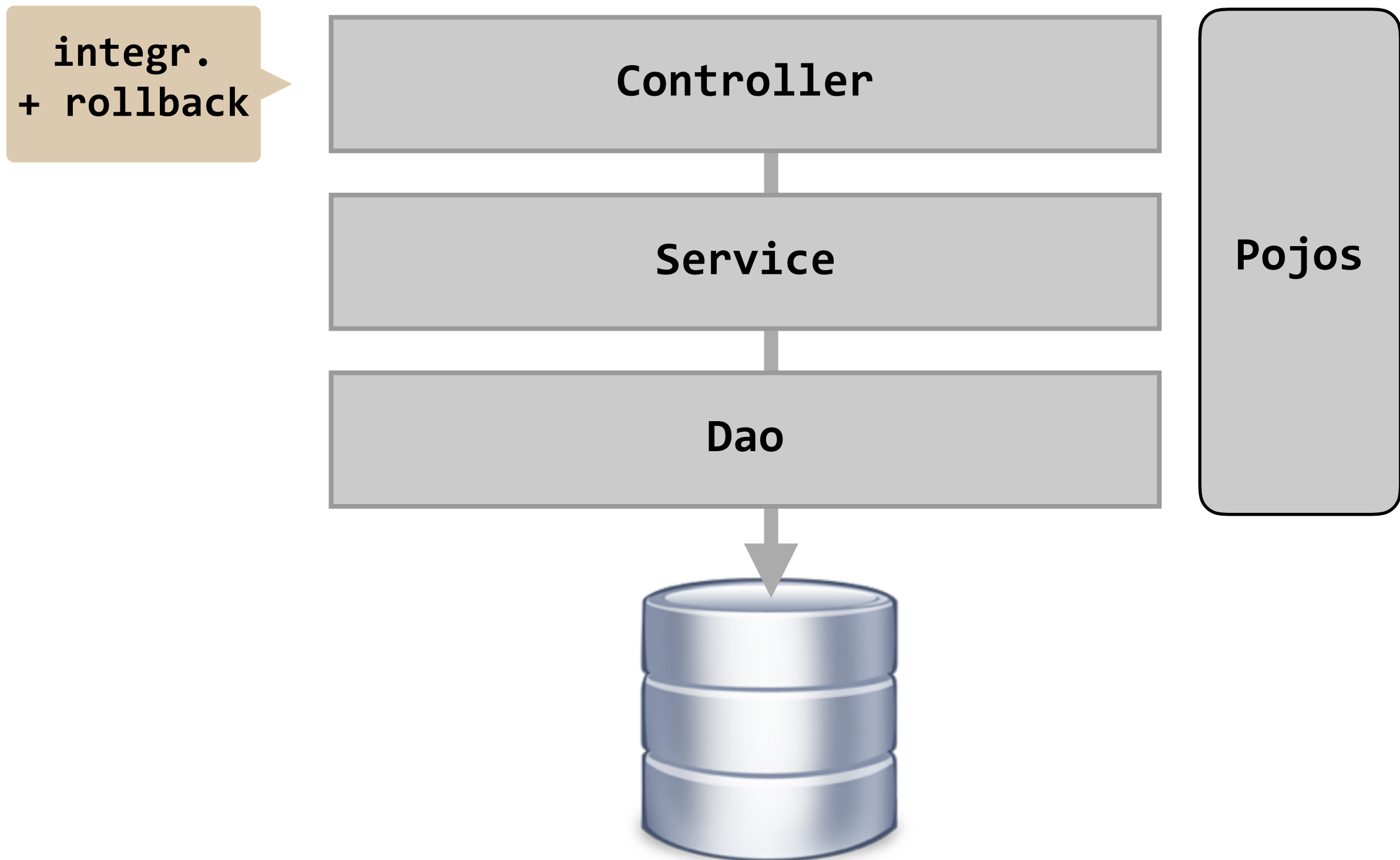
Test

Exceptions

```
@Test(expected = InvalidUserException.class)
public void testInvalidUserException() {
    // WHEN
    SignupForm signupForm = new SignupForm();
    signupForm.setLastName("formLast");
    signupForm.setFirstName("ESE");
    signupForm.setEmail("form@test.com");

    sampleService.saveForm(signupForm); // throws ex
}
```





```
@RunWith(SpringJUnit4ClassRunner.class)
@WebAppConfiguration
@ContextConfiguration(locations = {
    "file:src/main/webapp/WEB-INF/config/springMVC.xml",
    "file:src/main/webapp/WEB-INF/config/springData.xml" })
@Transactional
@TransactionConfiguration(defaultRollback = true)
public class ControllerTest {

    @Autowired private WebApplicationContext wac;

    private MockMvc mockMvc;

    @Before
    public void setup() {
        this.mockMvc = MockMvcBuilders.webAppContextSetup(this.wac).build();
    }

    @Test
    public void testGetSignupForm() throws Exception {
        ...
    }
}
```


User input

```
this.mockMvc
    .perform(
        post("/create").param("email", "<error>")
            .param("firstName", "<error>")
            .param("lastName", "<error>"))
    .andExpect(status().isOk())
    .andExpect(forwardedUrl("/pages/index.jsp"))
    .andExpect(model().attributeHasFieldErrors("signupForm", "email"))
```

Outcome