Mock Objects and Test Driven Development

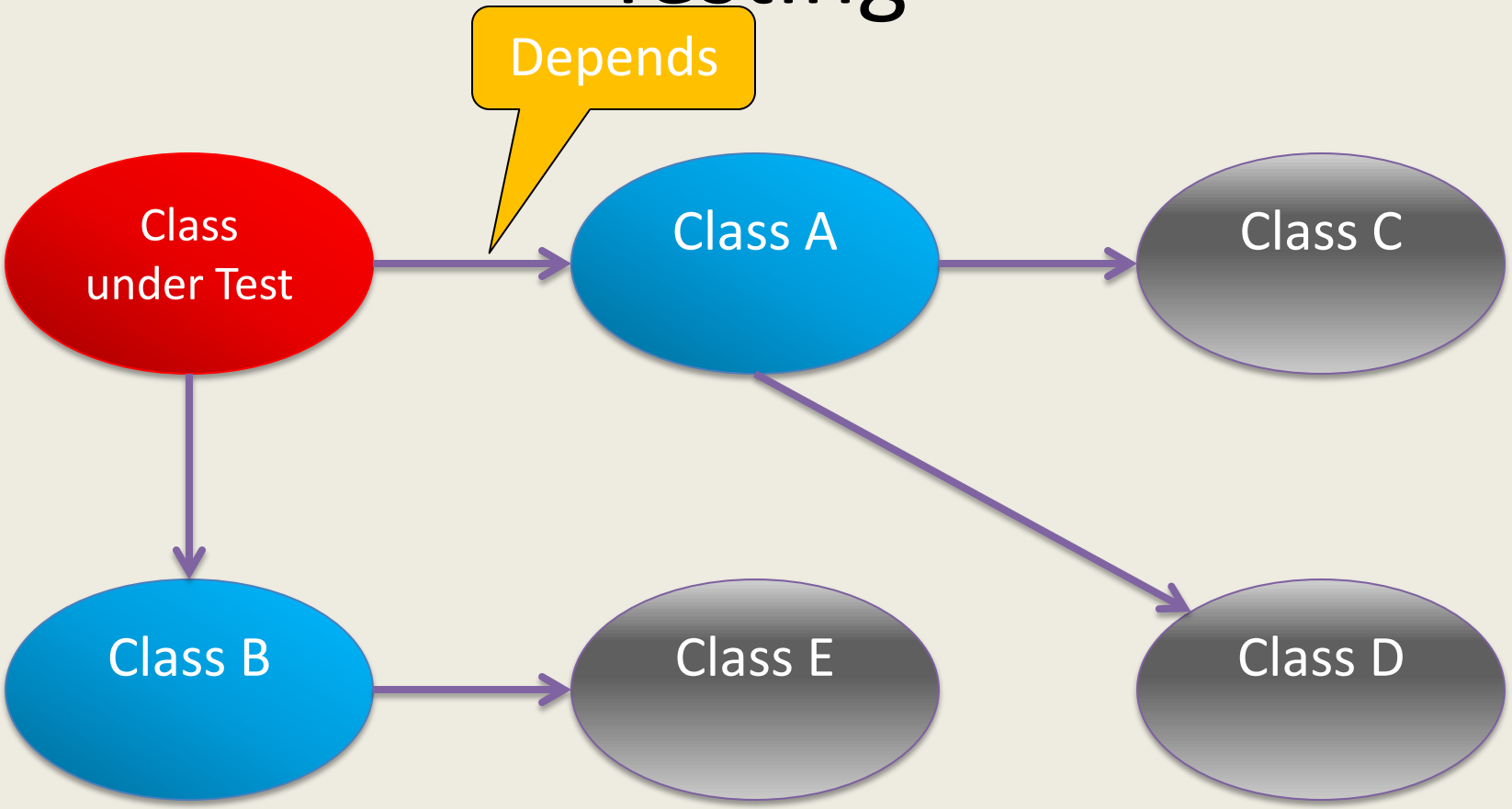
Your friendly assistant

(<name>@inf.ethz.ch)

Chair of Programming Methodology

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Testing


Depends
Class under Test
Class A
Class C
Class B
Class E
Class D

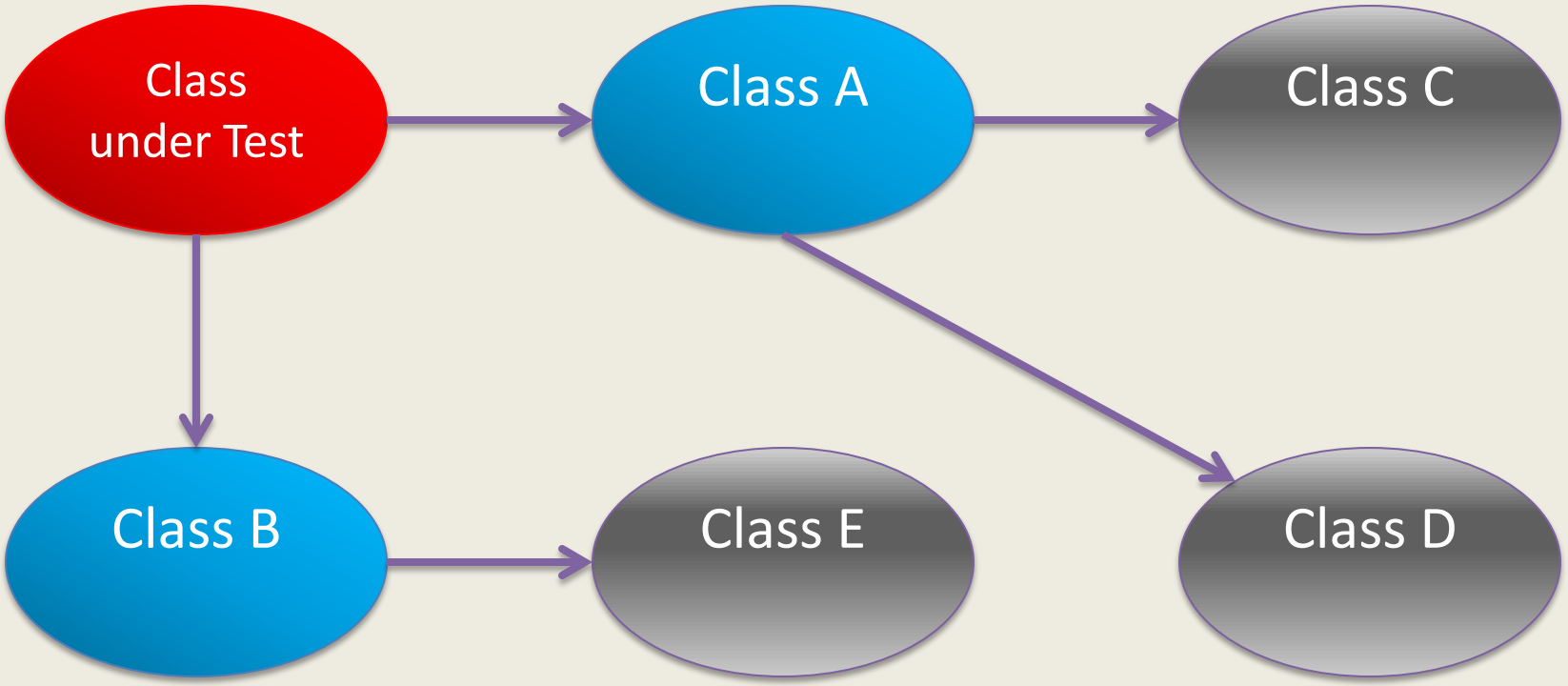
Motivation

- How do we test an object that depends on objects that do not exist yet?
- How do we test an object that depends on objects on which invocation is too time-consuming to pass the tests quickly? (e.g. databases)
- How do we test objects that depend on components that we don't have access to? (e.g. code developed by some third party)

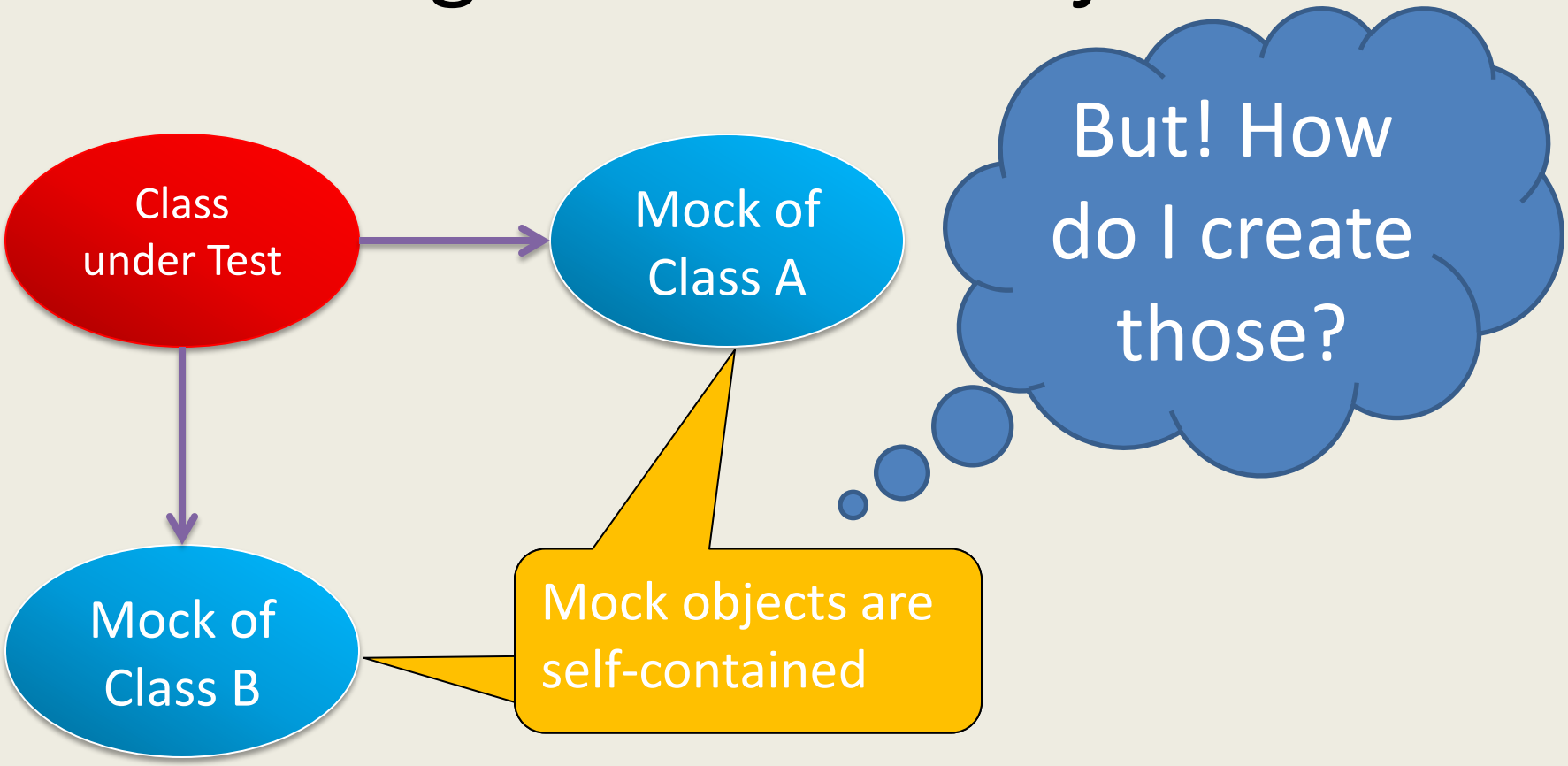
Mock Objects

- Objects simulating behaviour of concrete (real) objects in a controlled way
- Used when the real object is not available or when it is impractical to use
  - E.g. databases, interfaces, mocks of remote objects for local testing
- Speed up tests.

Testing with mock objects



Testing with mock objects



Mocking Frameworks

- jMock 2
  - http://jmock.org/
- EasyMock
  - http://easymock.org/
- Mockito
  - http://code.google.com/p/mockito
- JMockit
  - https://jmockit.dev.java.net
- …

JMock 2

- Most recent versions are:
  - Unstable: 2.6.0-RC2
  - Stable: 2.5.1
- Easy to use and more expressive than e.g. Easy Mock
- Simple library using loads of reflection
- Allows: creation of mock objects, expectations set-up, interaction execution (expect-run-verify)

JMock 2 – Mockery

- Serves as a context of a tested object (i.e. it contains objects that the object under test is dependent on/ communicates with)

```
import org.jmock.Mockery;

public class TestClassTest {
        // we create the context
        Mockery context = new Mockery();

        final ClassA mockClassA =

        context.mock(ClassA.class);

        final ClassB mockClassB =

        context.mock(ClassB.class);
        …
}
```

Objects that the class under test communicates with

JMock 2 – Expectations

- For each mock object we need to define the behaviour that we expect from it.
- This allows us to simulate behaviour of the real object in a controlled way.

```
import org.jmock.Expectations;

…

final ClassA mo
            conte

Object result =

context.checking(new Expecta    ns() {{
    oneOff(mockClassA).doSth();
        will(returnValue(result));
}});
```

We anticipate that doSth() will return object result when invoked

Method doSth() is called on mockClassA only once

JMock 2 – Expectations (cont'd)

- We can introduce arbitrary many expectations in the expectation block.
- There can be an arbitrary number of expectation block in a test and they are appended to each other sequentially.
- For more constructs that can appear in an expectation block, see:
  - http://www.jmock.org/cheat-sheet.html

Example (taken from http://www.jmock.org/getting-started.html)

```
import org.jmock.Mockery;
import org.jmock.Expectations;

class PublisherTest extends TestCase {
   Mockery context = new Mockery();

   public void testOneSubscriberReceivesAMessage() {
      // set up
      final Subscriber subscriber = context.mock(Subscriber.class);

      Publisher publisher = new Publisher();
      publisher.add(subscriber);

      final String message = "message";

      // expectations
      context.checking(new Expectations() {{
         oneOf (subscriber).receive(message);
      }});

      // execute
      publisher.publish(message);

      // verify
      context.assertIsSatisfied();
   }
}
```

```
interface Subscriber {
   void receive(String message);
}
```

Create a mock Subscriber object

Register a mock Subscriber with a Publisher

We set up our expectations

The method is expected to be called once

verify that the mock Subscriber was called as expected

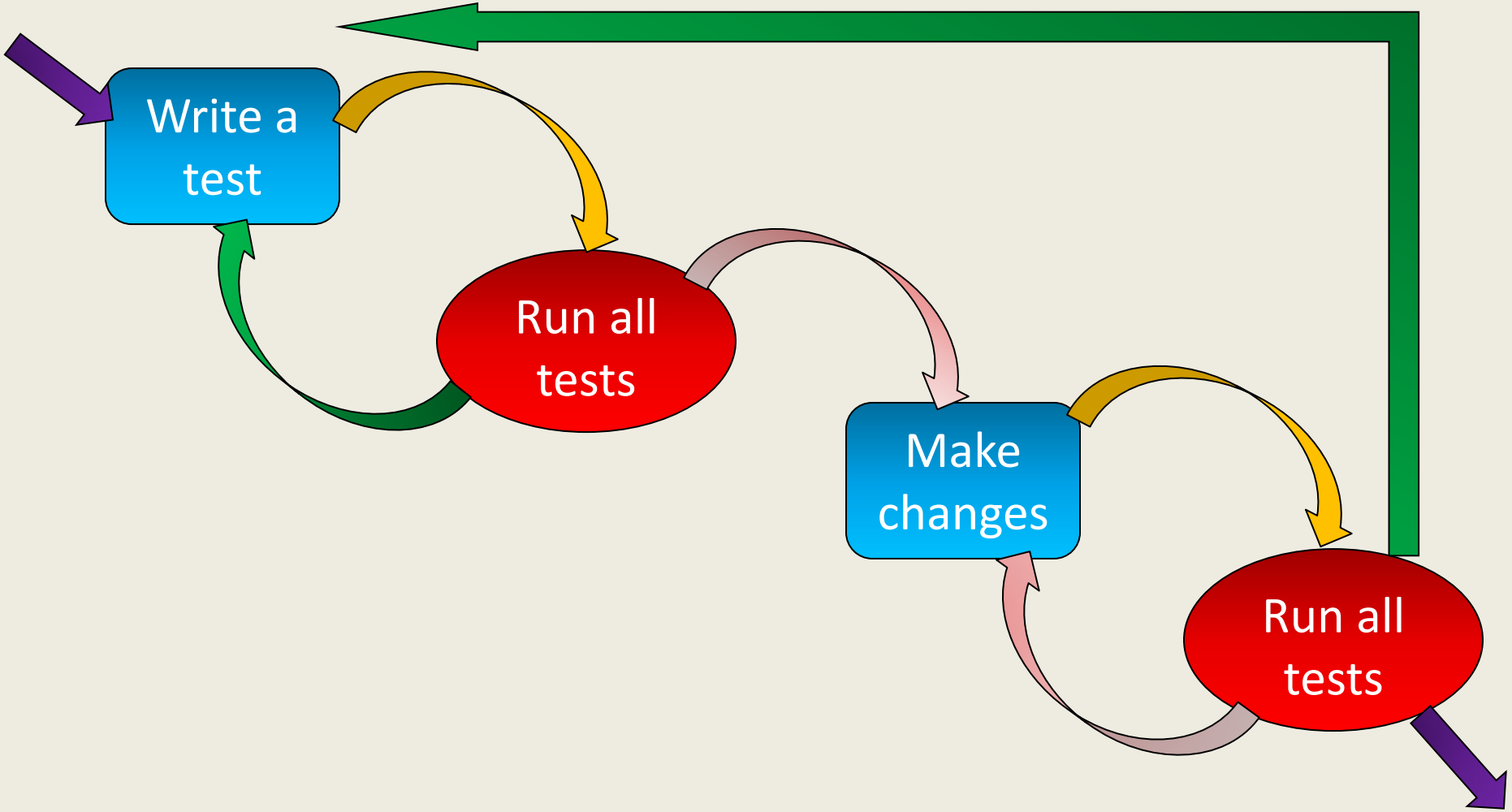
Test Driven Development (TDD)

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Test Driven Development (TDD)

- We do not start directly with the implementation.
- Write tests first that test some piece of functionality.
- Write code that implements this functionality and keep refining it until all the tests pass.
- Complies to extreme programming – test first and short development cycles.

TDD – Work Flow



DEMO