LMU

Abschlussvortrag Projektarbeit

Extending the Fragment Explorer with Test Cases and a Graphical Viewer

Written by: Joséphine Nitschké
Supervised by: Thomas Fritz (UBC), Prof. Gail Murphy, PhD (UBC), Prof Dr. Heinrich Hußmann(LMU)
SoSe 2009

Structure

- Background Information
- Problem
- Extension of the FE with Test Cases
- Extension of the FE with a new view
- Summary

Background Information

What is the Fragment Explorer?

- Plugin for Jazz
- Provides a new view
- Combines information on
  - Change Sets
  - Work Items
  - Java Elements
  - Team Members
- Implemented in Java

Problem

- Distributed Information in todays development environments

  → different views for: program code, bugs, change sets...

- ... Test Cases

  → difficult to answer questions like:

  - „Which code caused the test case to fail?“
  - „Who is responsible for the test case to fail?“
  - „Is a change responsible for the failure of the test case?“
  - ...

Problem with the JUnit view I

- Get an answer to the question:

  „Which code causes the Test Case to fail?“

  - Several steps are needed:

    (Step 1: run the Test Cases)

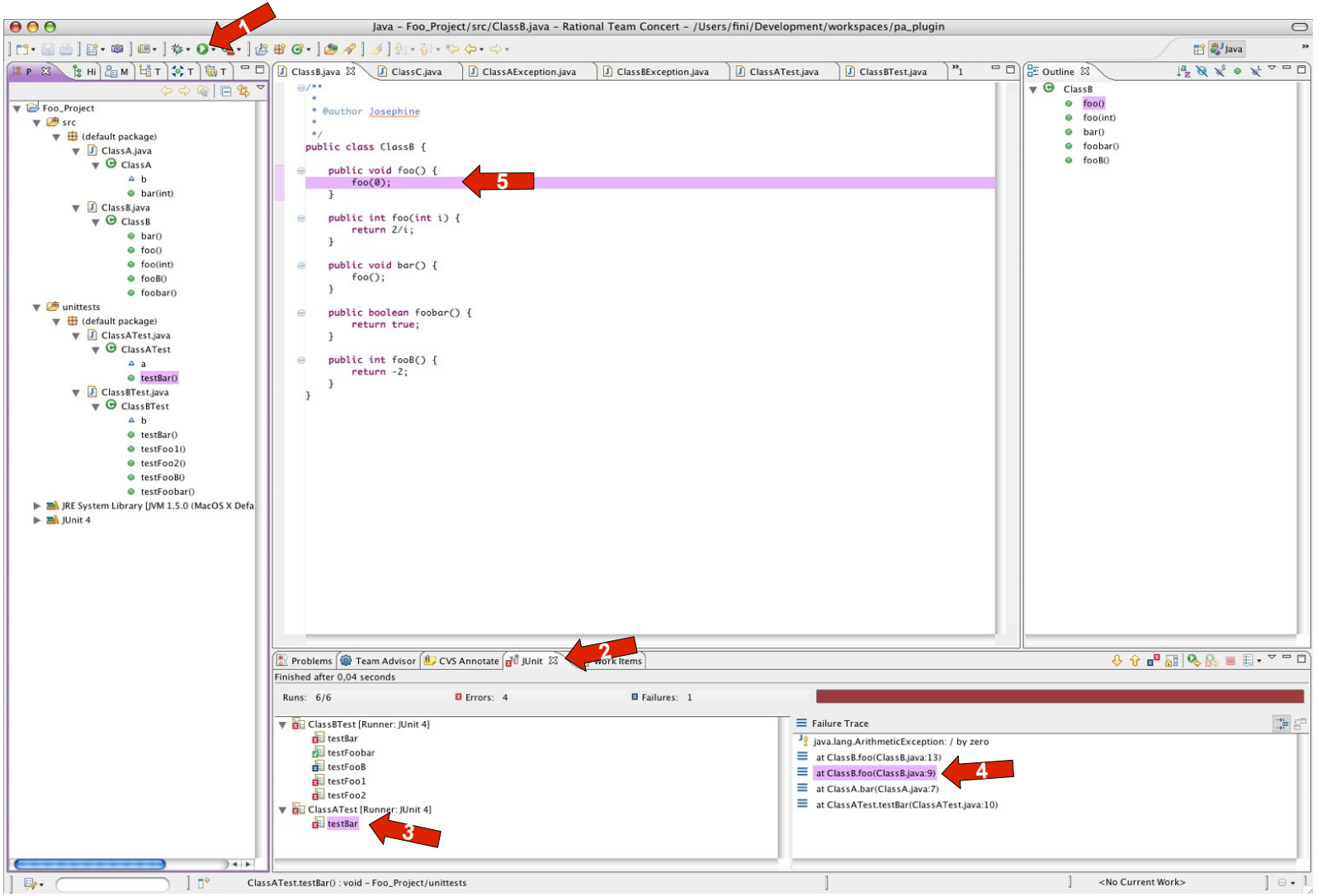
    Step 2: open the JUnit view

    Step 3: select the Test Case of interest

    Step 4: get information out of the Error Trace

    Step 5: open the editor to find out why the Test Case failed

Problem with the JUnit view II



Problem with the JUnit view III

More complex scenarios are conceiveable

- e.g. „Which task caused a test case to fail?“

  (Step 1: run the test cases)

  Step 2: open the JUnit view

  Step 3: select the Test Case of interest

  Step 4: open the Error Trace

  Step 5: open the Work Items view

Problem with the JUnit view IIII

Step 6: get the linked Change Sets

Step 7: get the linked Java Elements

Step 8: compare the Java Elements with the ones from the Error Trace

a lot of work for a simple information!

→ **combine information from different views in one view**

Extending the FE with Test Cases

Example of Use

- A software engineer writes:
  - Two classes (ClassA.java, ClassB.java)
  - Two test classes (ClassATest.java, ClassBTest.java)
- He wants to know:

„Is my code doing what it is supposed to do and if not, where is the failure?“

Example of Use

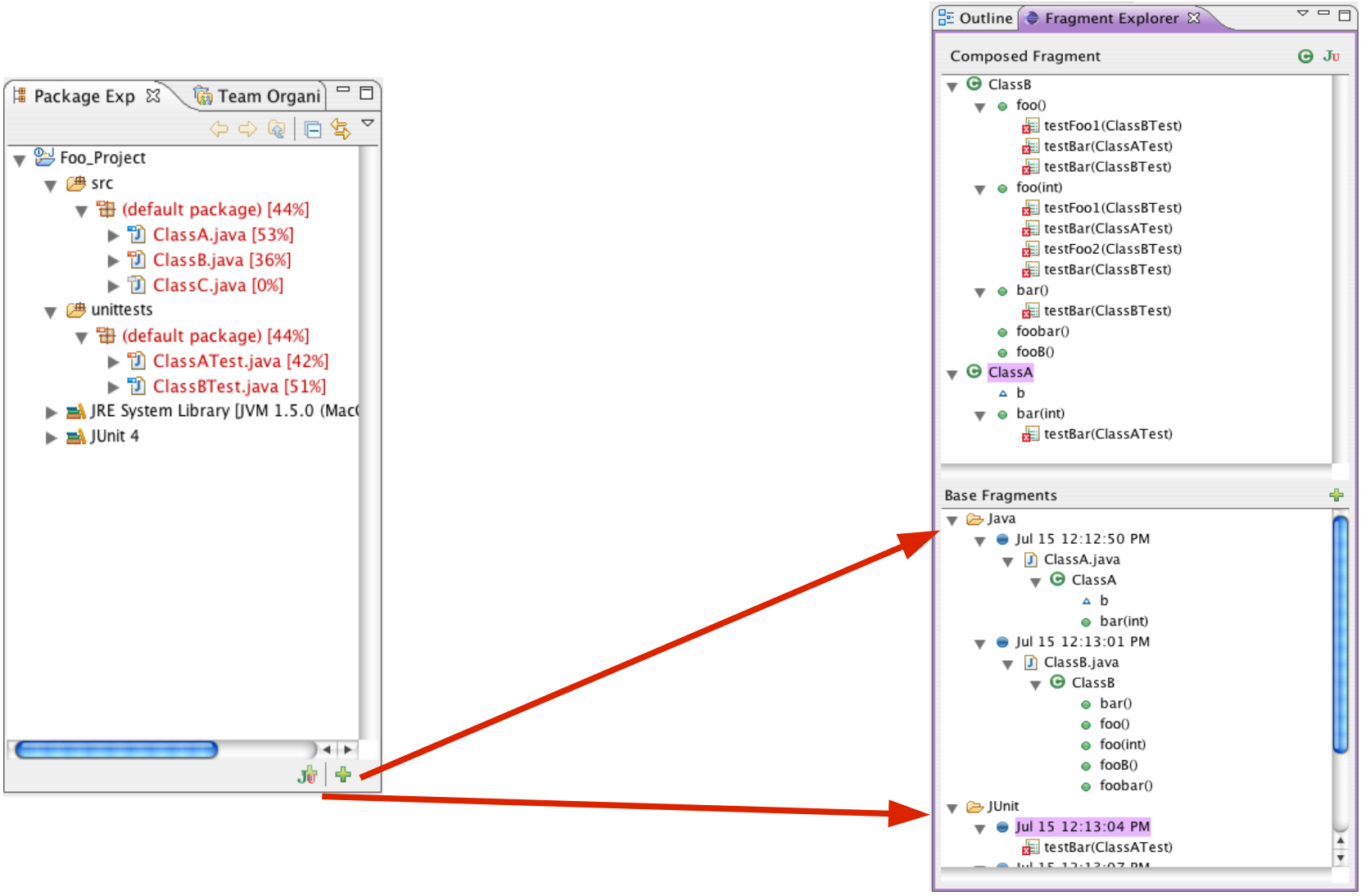
With the help of the Fragment Explorer

Step 1: select the test cases in the Package Explorer (→ appear in the base viewer of the FE)

Step 2: select the “normal“ Java classes (→ appear in the base viewer of the FE)

Step 3: compose the test cases and normal classes (→ appear in the composed viewer of the FE)

Example of Use I



Implementation of the example classes

```
public class ClassA {

	ClassB b= new ClassB();

	public int bar(int i) {
		b.foo();
		return i+1;
	}
}
```

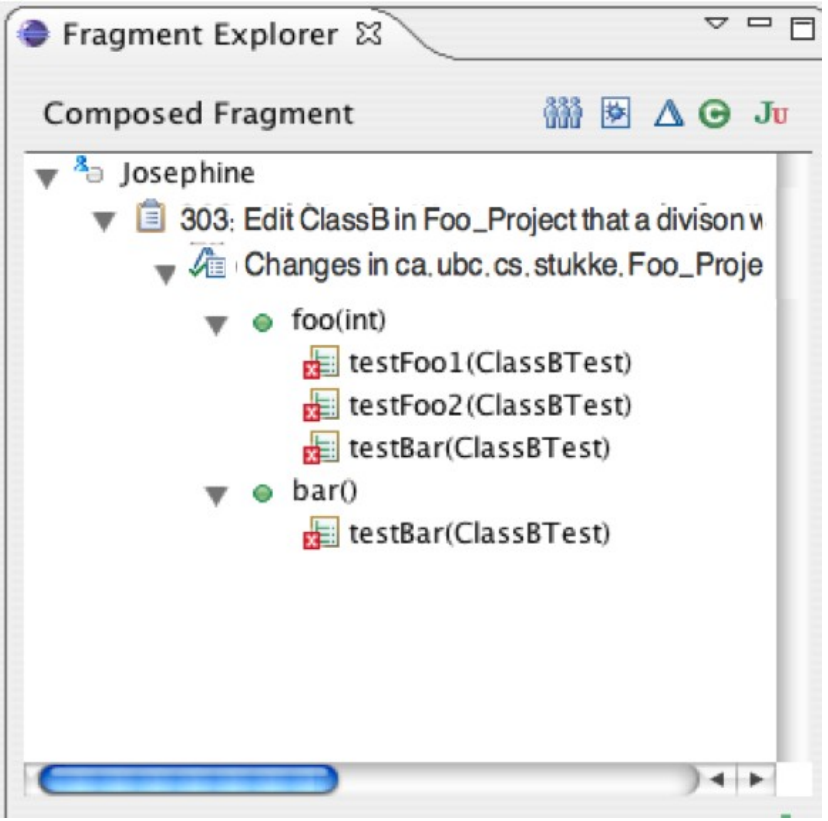
```
public class ClassB {



	public void foo() {
		foo(0);
	}

	public int foo(int i) {
		return 2/i;
	}

	public void bar() {
		foo();
	}

	public boolean foobar() {
		return true;
	}

	public int fooB() {
		return -2;
	}

}
```

```
public class ClassATest extends TestCase {


	ClassA a= new ClassA();

	public void testBar() {
		int i=a.bar(-1);
		assertNotSame(0, i);
	}

}
```

```
public class ClassBTest extends TestCase {

	ClassB b= new ClassB();


	public void testFoo1() {
		b.foo();
	}

	public void testFoo2() {
		b.foo(0);
	}

	public void testBar() {
		b.bar();
	}

	public void testFoobar() {
		assertEquals(true, b.foobar());
	}

	public void testFooB() {
		assertTrue(b.fooB() > 0);
	}

}
```

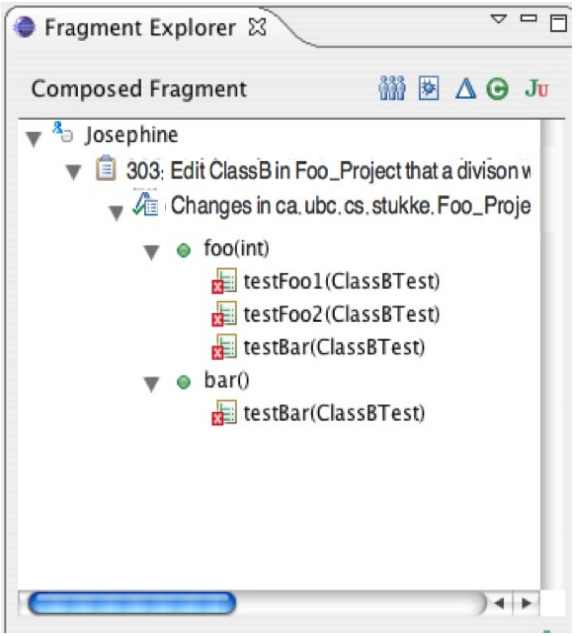
Example of use II

„Who is responsible for the test case to fail?“



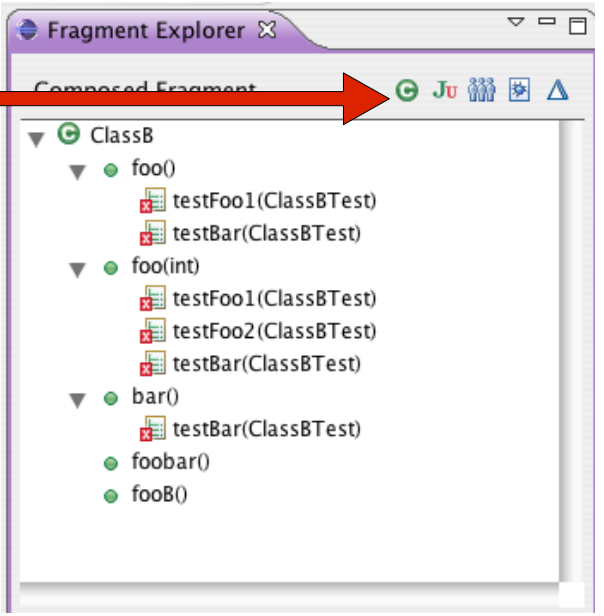
Extending the FE with a new view

- Disadvantage of a tree viewer:
  - Presentation of the information depends on the order, e.g.:



Different orders

→ not every information is displayed at all times



New view

- Idea:
  - Present the information as a graph (with the help of zest)

    → elements as nodes

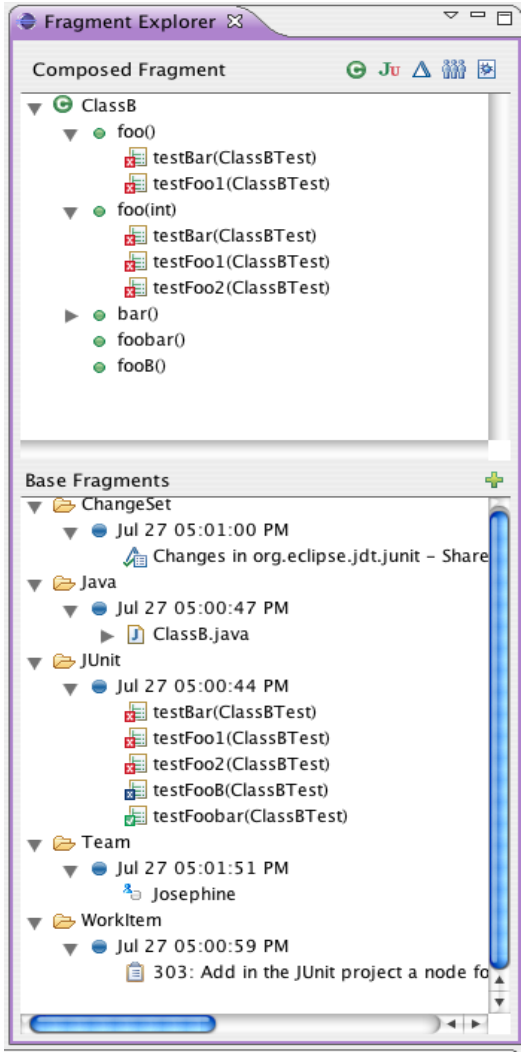
    → cohesions as edges

→ every information is available at all times

Example of Use I

- User wants the connection between:
  - A team member
  - A change set
  - A work item
  - A Java class
  - Some test cases

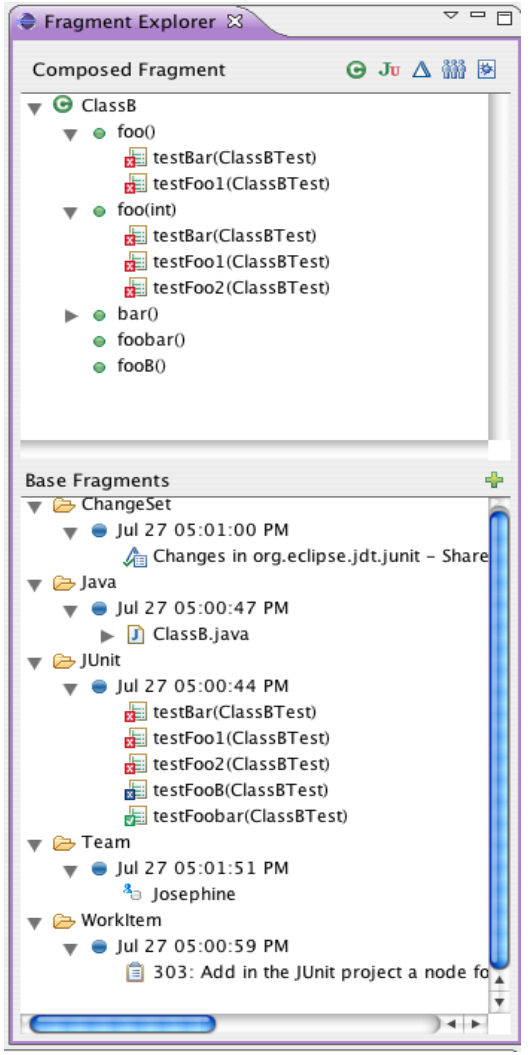
his choice:



Example of Use I

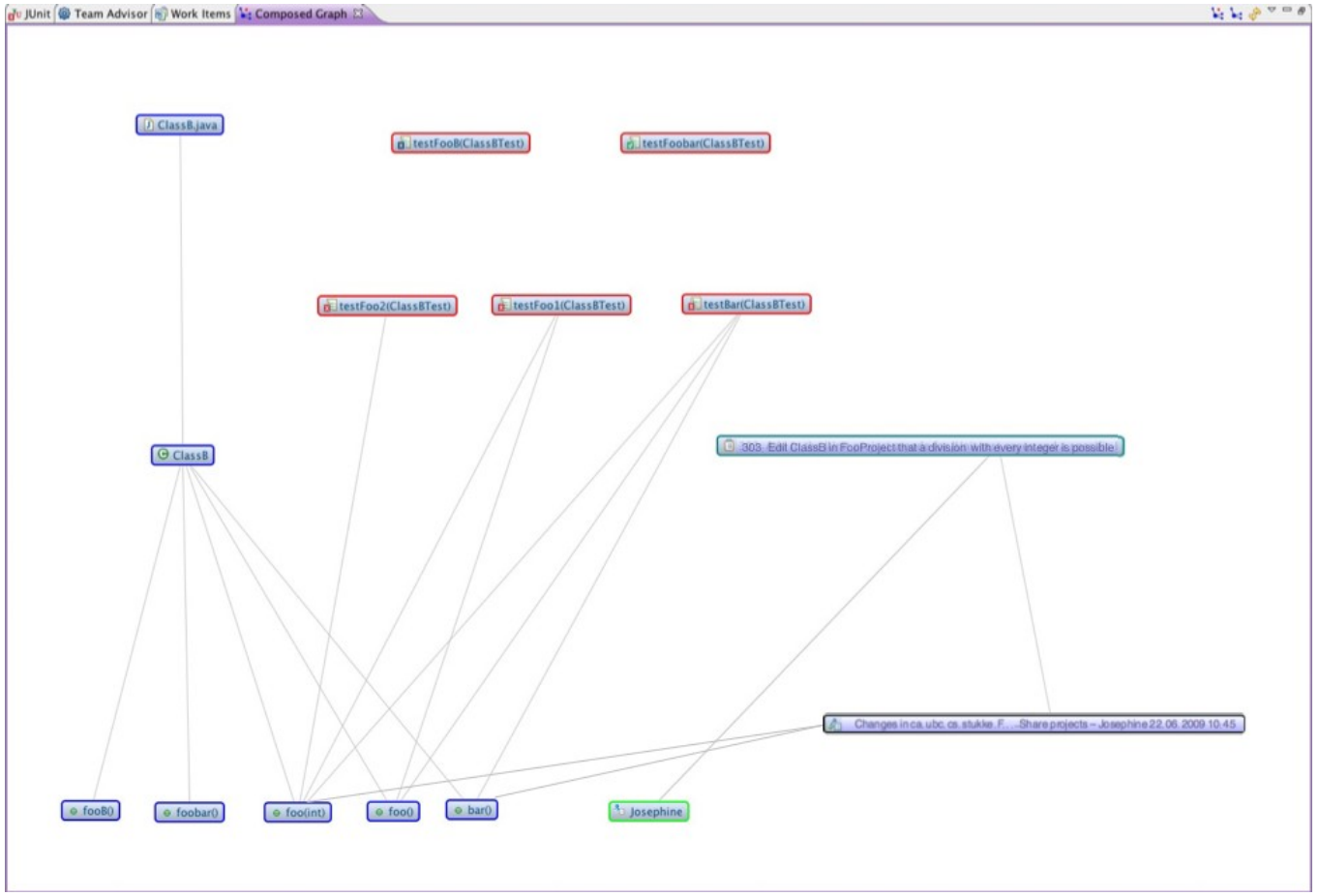
- User wants the connection between:
  - A team member
  - A change set
  - A work item
  - A Java class
  - Some test cases

his choice:



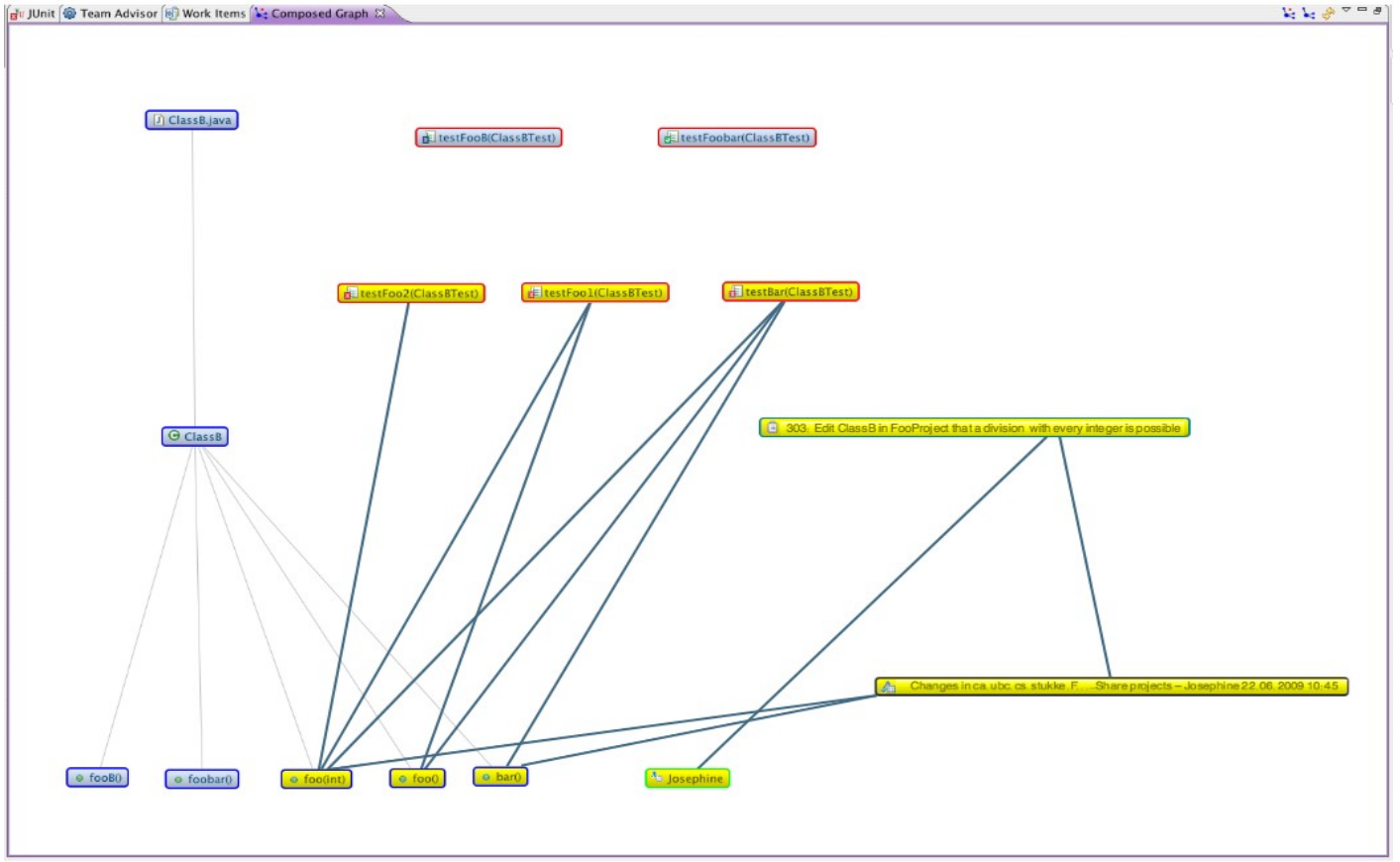
Example of Use II

Composed Graph View:



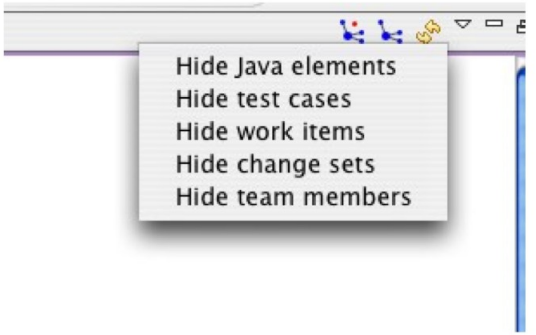
Features I

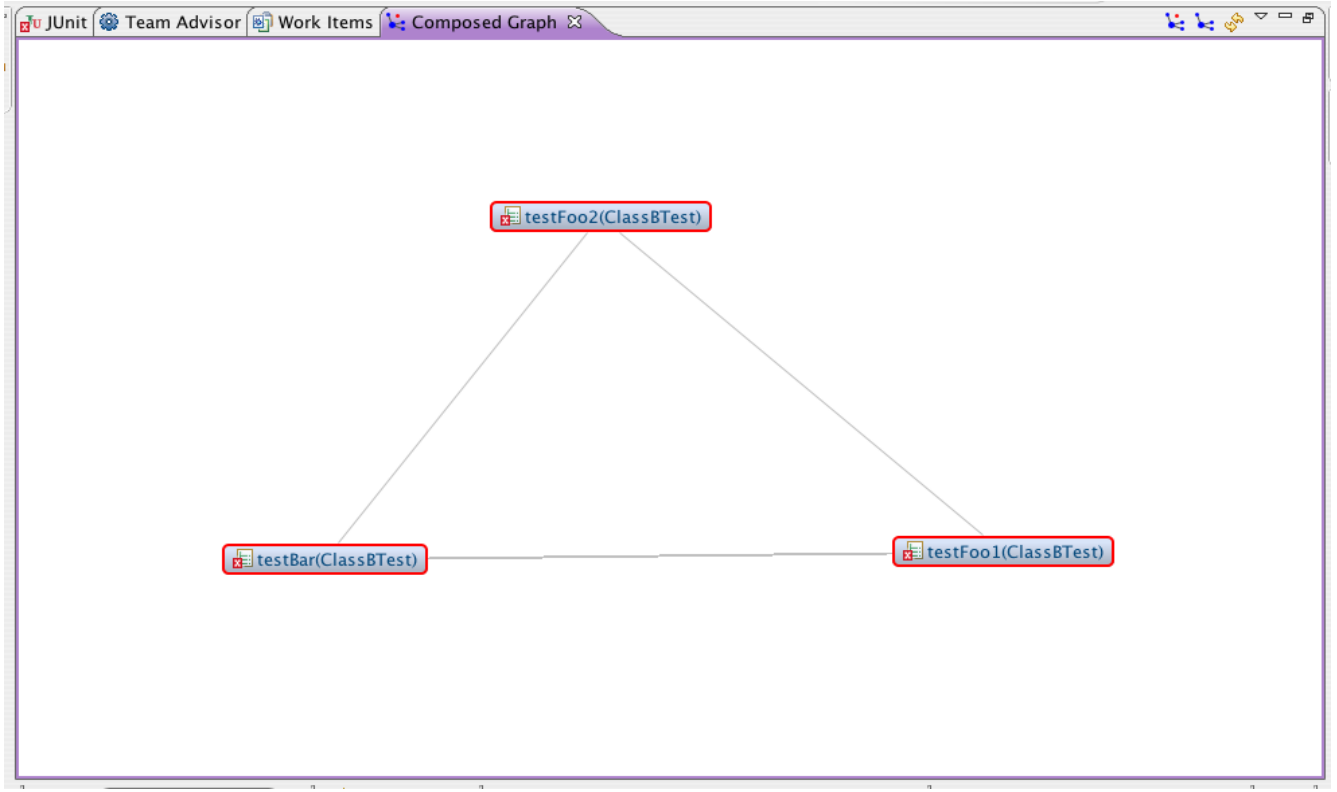
- Select a node:



Features II

- Hide Nodes





Summary

- FE extended with Test Cases
- FE extended with a new view to represent the information as a graph

Thanks

- Questions???

References

- http://jazz.net/
- http://www.eclipse.org/gef/zest/