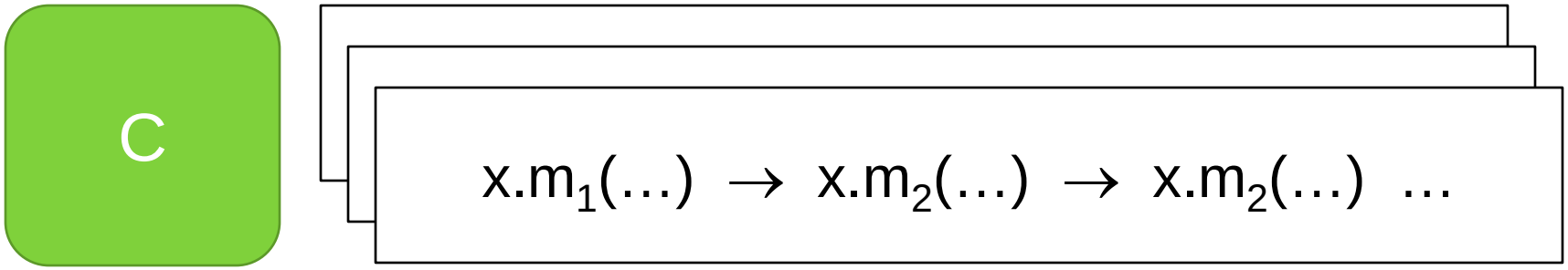
Specification-based Testing of Object-oriented Programs with T2

Wishnu Prasetya (Utrecht Univ.)
Tanja Vos (Univ. Politectica de Valencia)

Email: wishnu@cs.uu.nl
URL: www.cs.uu.nl/~wishnu
T2 site: t2framework.googlecode.com

T2 : automated Testing Tool

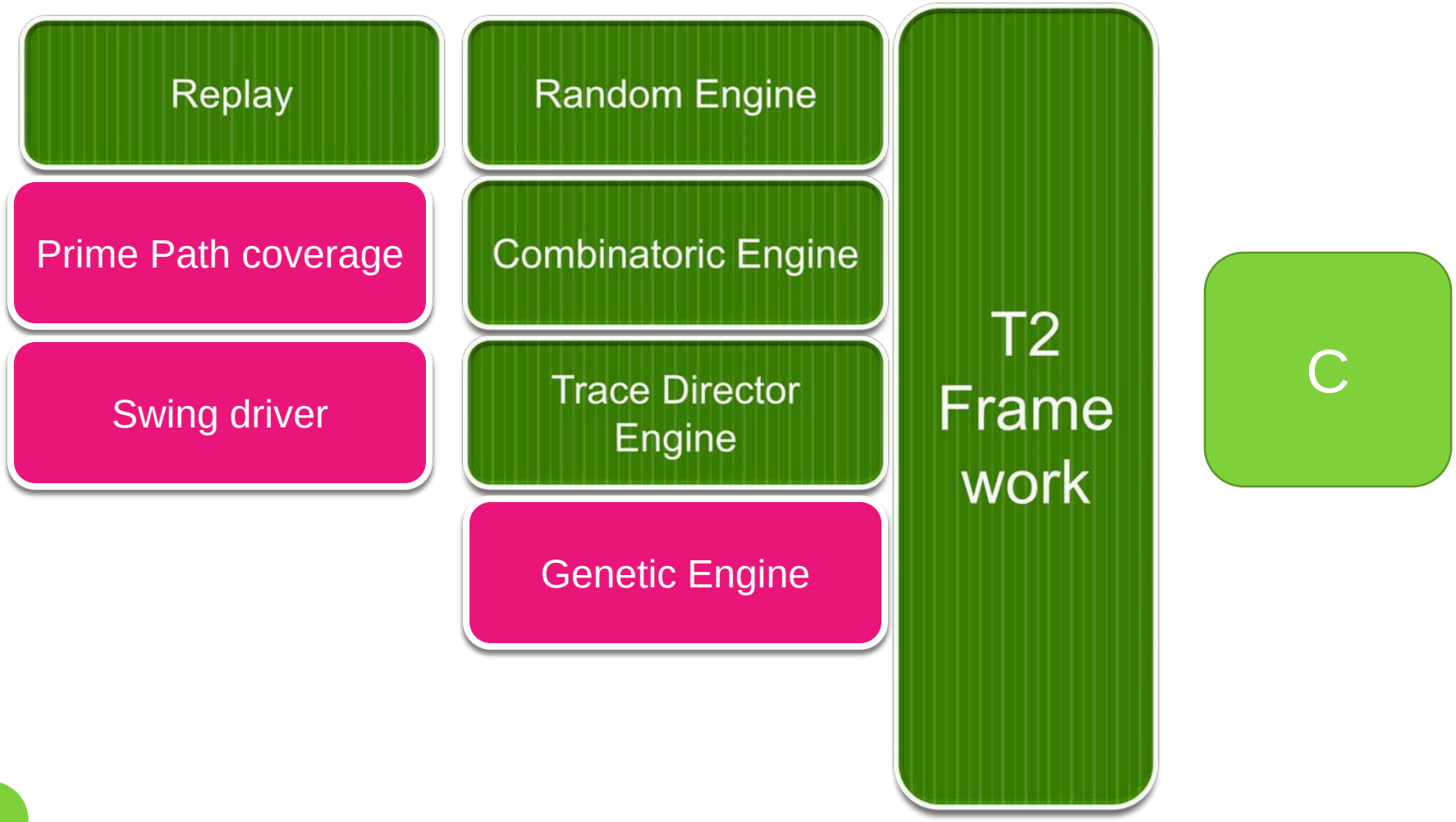
- Light weight, open source, target Java
- Out of the box, for Unit Testing → class level



  - On the fly → fast
  - Tets scope

- Integration & application level

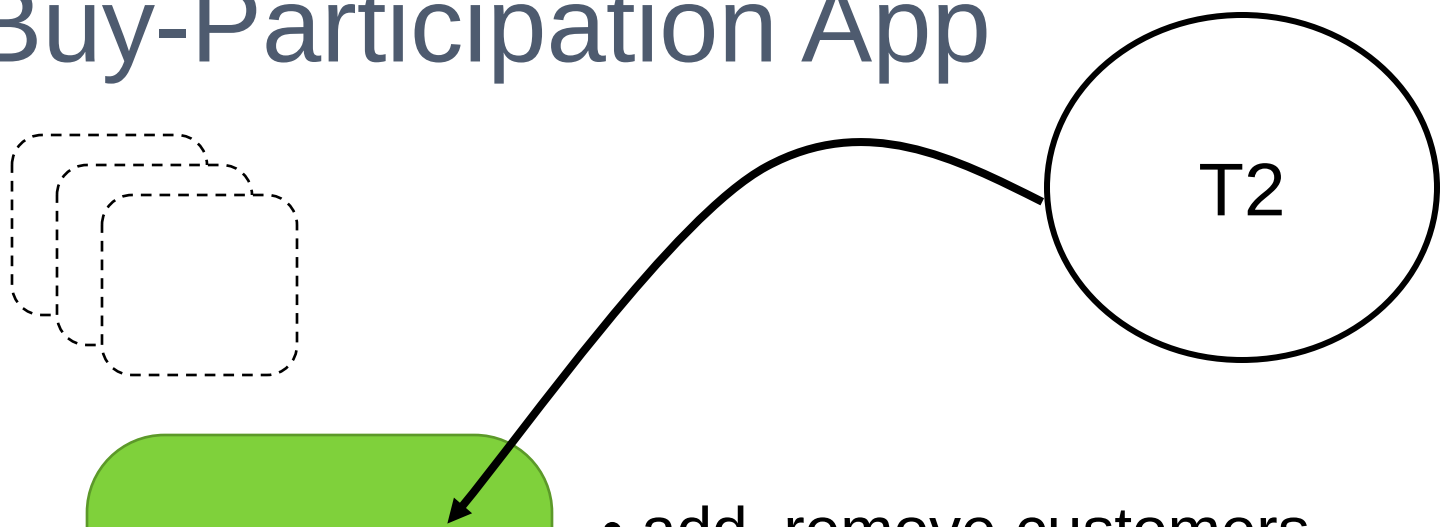
Architecture



T2 is for specification-based testing

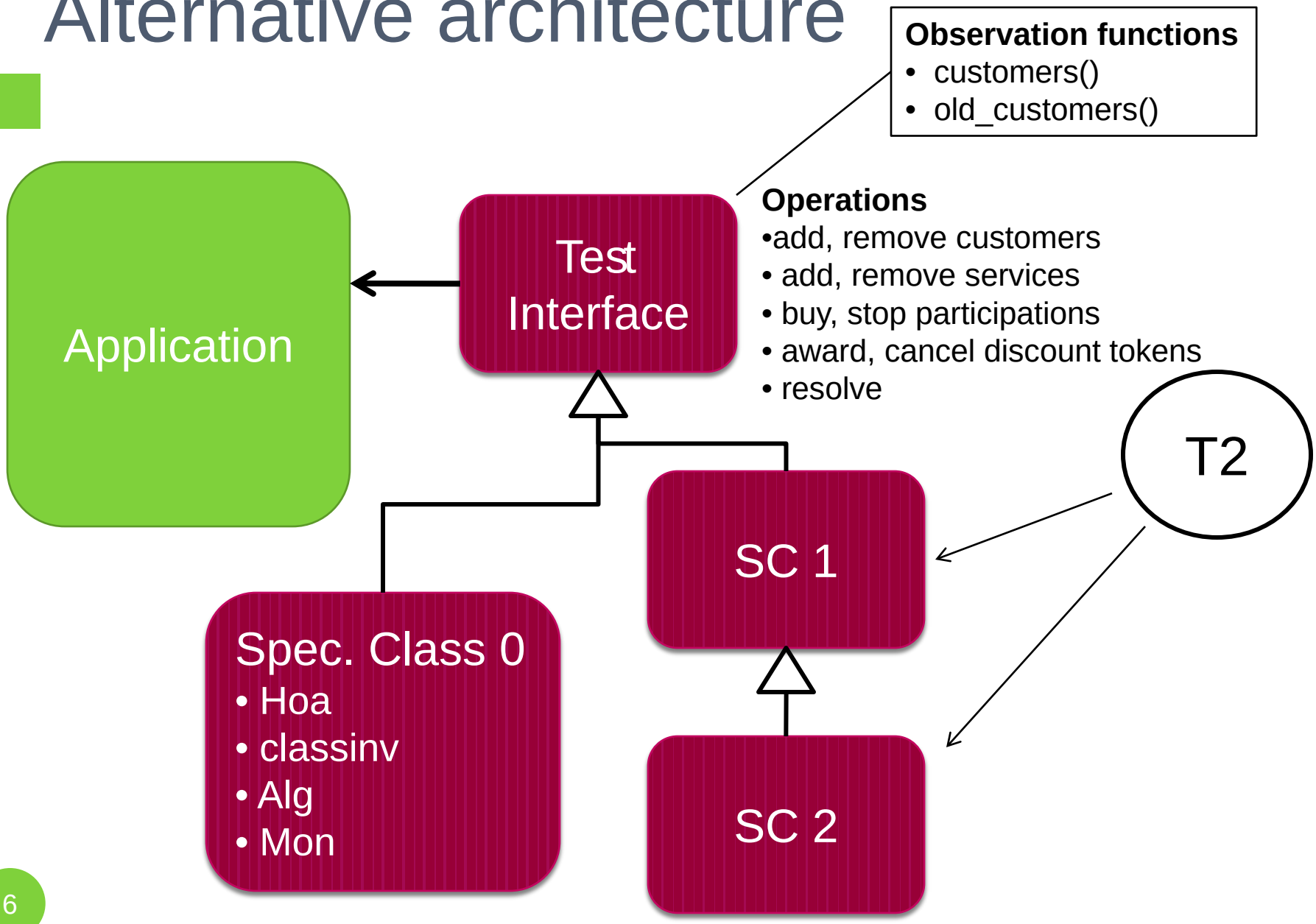
- T2 checks unexected exception
- But then it can also check “assertions”
- Class invariant
- In-code specifications.

Buy-Participation App

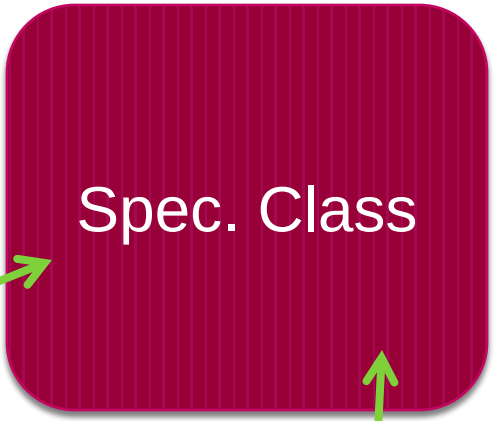


- add, remove customers
- add, remove services
- buy, stop participations
- award, cancel discount tokens
- resolve

Alternative architecture



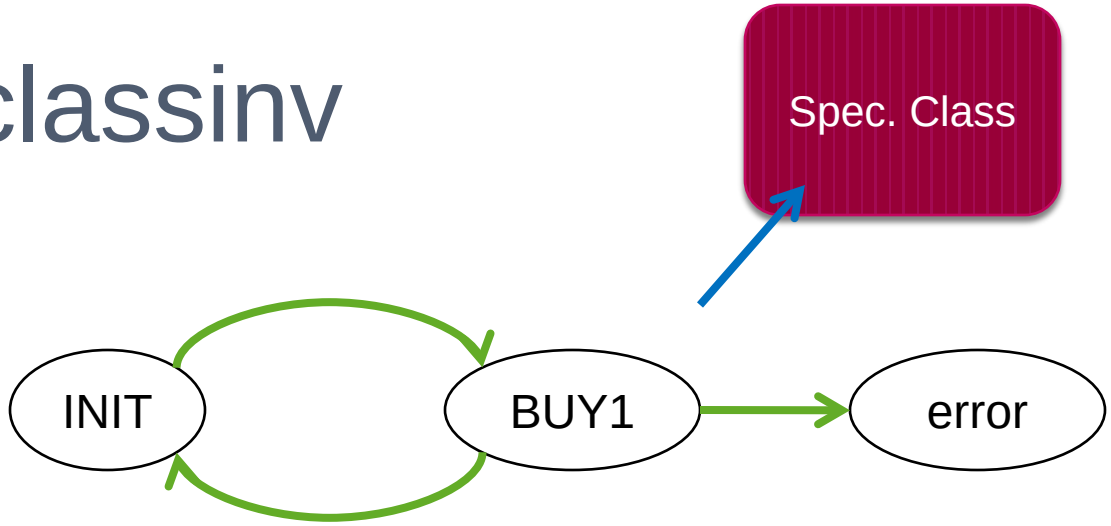
Class invariant




```
customerOk = new Predicate() {

    Boolean check(Customer C) {
        return   C.participationValue() == expectedPartVal (C)
                 &&
                 C.getDiscountValue() == expectedDiscVal(C) ;  }
    }
}
```

```
boolean classinv() { return forall(customers(),customerOk)  ; }
```

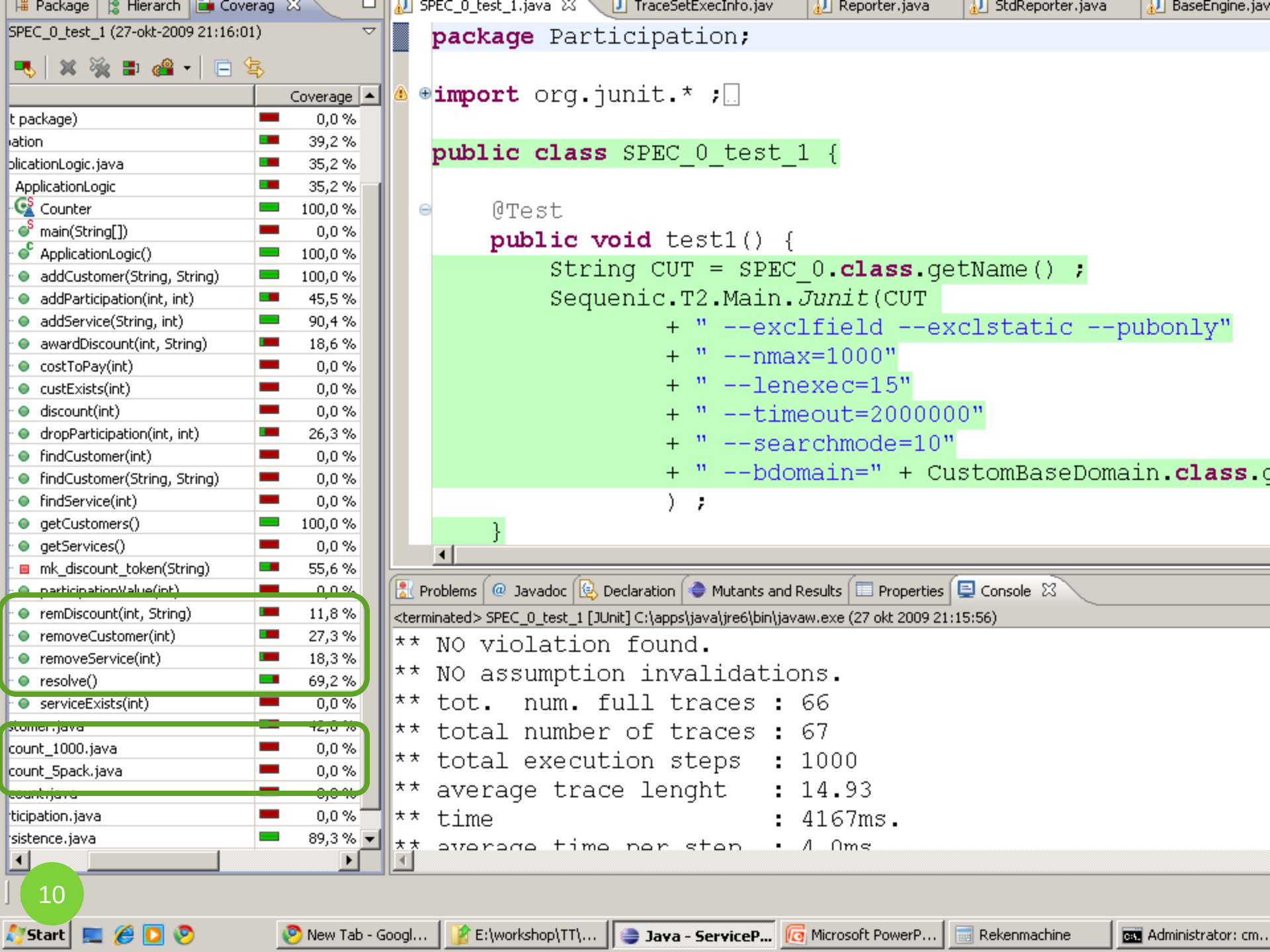
Monitor → classinv



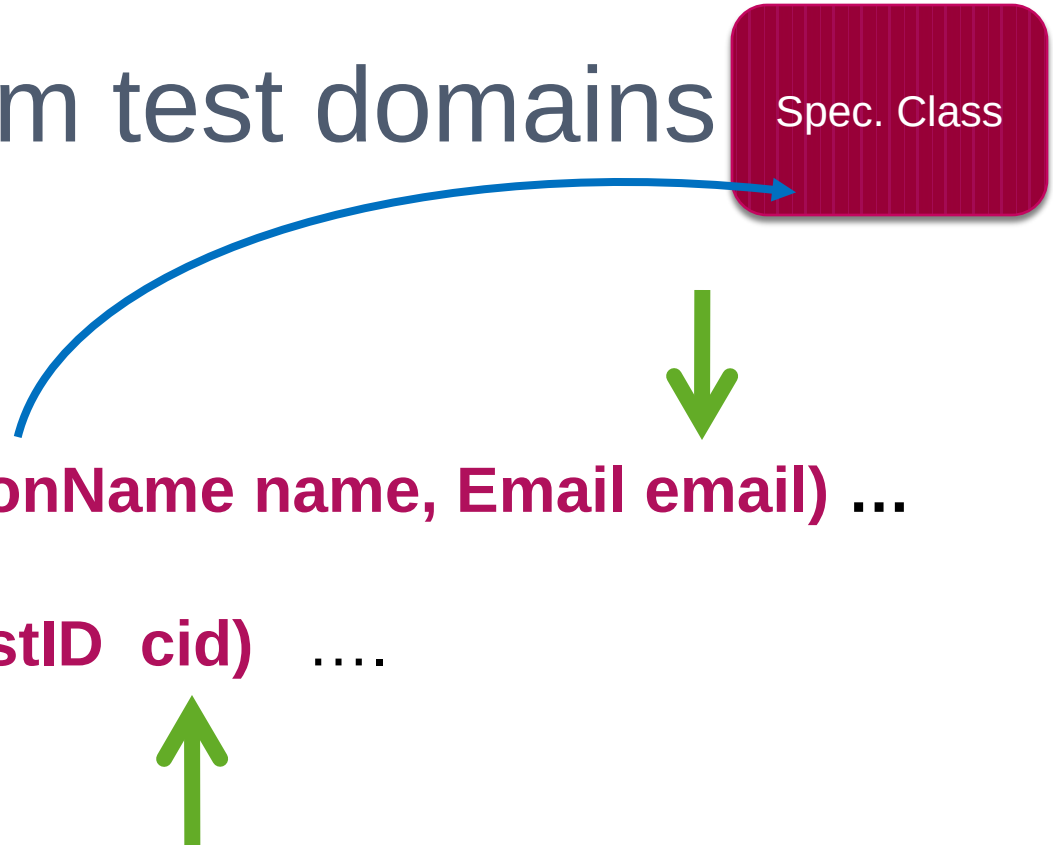

```
public class Monitor  {
      int state = INIT ;
      Customer c ;

      public void exec(int buyer) {

         if (state==INIT && buyer==c.id) {   state=BUY1 ; return ; }

         if (buyer!= c.id) {  state=INIT ;  return ;  }

         if (state==BUY1 && buyer== c.id) {  assert false ;  return ;  }
      }
}
```

‘algebraic’ properties

```
class  SPEC_alg  extends SPEC_0 {

    public void property1(name,email) {
        addCust(name,email) ;
        int N = customers().size() ;
        addCust(name,email) ;
        assert   customers().size() == N ;
    }

    public void property2 …

    public void property3…
```


SPEC_0_test_1 (27-okt-2009 21:16:01)
Coverage
0,0 %
39,2 %
35,2 %
35,2 %
100,0 %
0,0 %
100,0 %
100,0 %
45,5 %
90,4 %
18,6 %
0,0 %
0,0 %
0,0 %
26,3 %
0,0 %
0,0 %
0,0 %
100,0 %
0,0 %
55,6 %
11,8 %
27,3 %
18,3 %
69,2 %
0,0 %
0,0 %
0,0 %
89,3 %
package Participation;
import org.junit.* ;
public class SPEC_0_test_1 {
@Test
public void test1() {
String CUT = SPEC_0.class.getName() ;
Sequenic.T2.Main.Junit(CUT
+ " --exclfield --exclstatic --pubonly"
+ " --nmax=1000"
+ " --lenexec=15"
+ " --timeout=2000000"
+ " --searchmode=10"
+ " --bdomain=" + CustomBaseDomain.class.
) ;
}
Problems
Javadoc
Declaration
Mutants and Results
Properties
Console
<terminated> SPEC_0_test_1 [JUnit] C:\apps\java\jre6\bin\javaw.exe (27 okt 2009 21:15:56)
** NO violation found.
** NO assumption invalidations.
** tot. num. full traces : 66
** total number of traces : 67
** total execution steps : 1000
** average trace lenght : 14.93
** time : 4167ms.

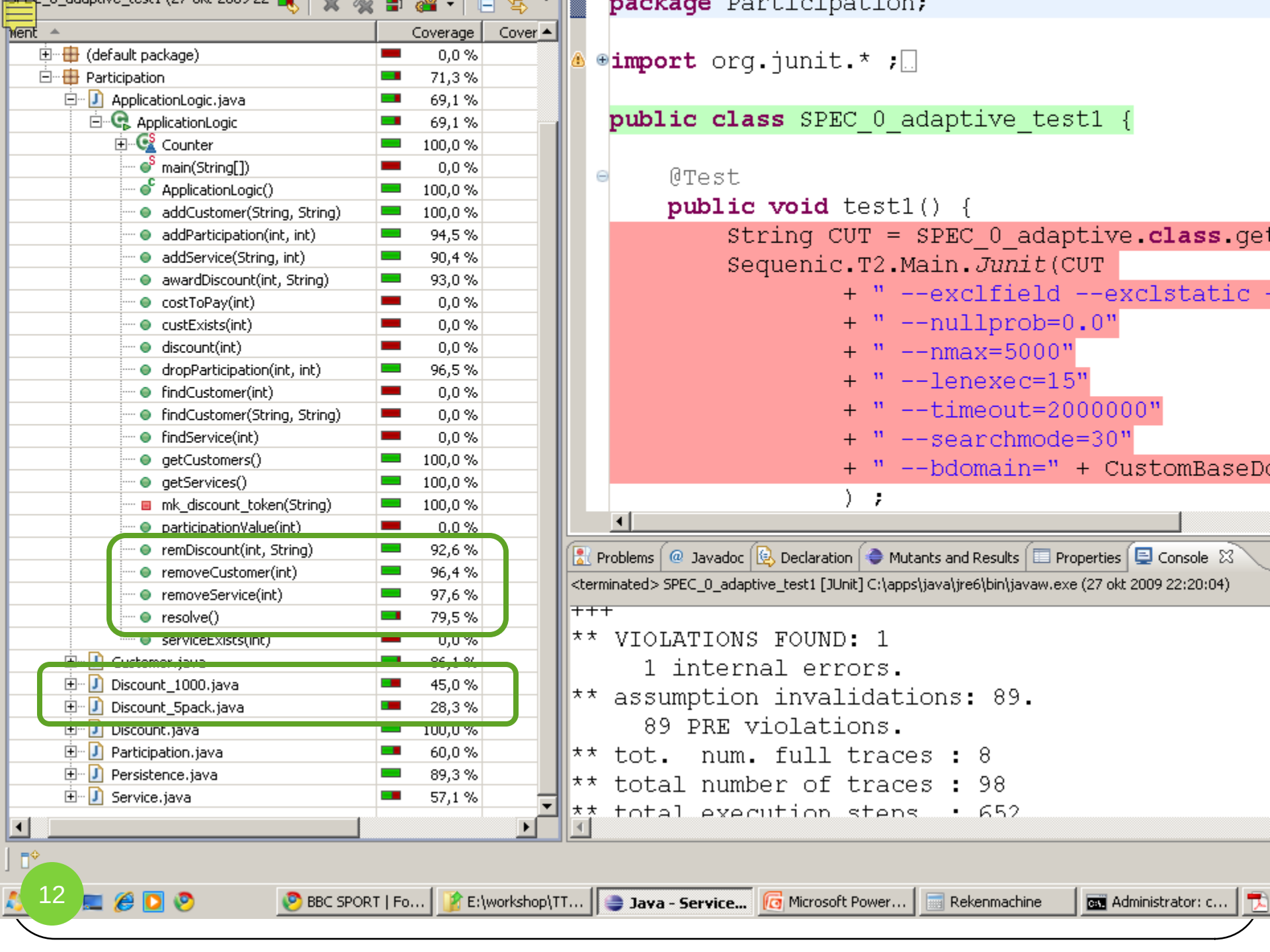
Injecting custom test domains



public int **addCust(PersonName name, Email email)** **…**

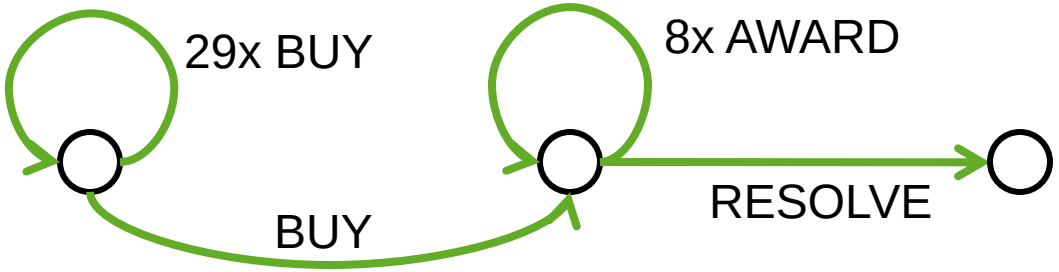
public void **remCust(CustID cid)** ….

```
static class CustID {
    String id ;

    public CustId(int k)  … // deterministically maps int to valid ID
}
```


Coverage
(default package) 0,0 %
Participation 71,3 %
ApplicationLogic.java 69,1 %
ApplicationLogic 69,1 %
Counter 100,0 %
main(String[]) 0,0 %
ApplicationLogic() 100,0 %
addCustomer(String, String) 100,0 %
addParticipation(int, int) 94,5 %
addService(String, int) 90,4 %
awardDiscount(int, String) 93,0 %
costToPay(int) 0,0 %
custExists(int) 0,0 %
discount(int) 0,0 %
dropParticipation(int, int) 96,5 %
findCustomer(int) 0,0 %
findCustomer(String, String) 0,0 %
findService(int) 0,0 %
getCustomers() 100,0 %
getServices() 100,0 %
mk_discount_token(String) 100,0 %
participationValue(int) 0,0 %
remDiscount(int, String) 92,6 %
removeCustomer(int) 96,4 %
removeService(int) 97,6 %
resolve() 79,5 %
Discount_1000.java 45,0 %
Discount_5pack.java 28,3 %
Participation.java 60,0 %
Persistence.java 89,3 %
Service.java 57,1 %
package Participation;
import org.junit.* ;
public class SPEC_0_adaptive_test1 {
@Test
public void test1() {
String CUT = SPEC_0_adaptive.class.get
Sequenic.T2.Main.Junit(CUT
+ " --exclfield --exclstatic
+ " --nullprob=0.0"
+ " --nmax=5000"
+ " --lenexec=15"
+ " --timeout=2000000"
+ " --searchmode=30"
+ " --bdomain=" + CustomBaseDo
) ;
Problems
Javadoc
Declaration
Mutants and Results
Properties
Console
<terminated> SPEC_0_adaptive_test1 [JUnit] C:\apps\java\jre6\bin\javaw.exe (27 okt 2009 22:20:04)
+++
** VIOLATIONS FOUND: 1
1 internal errors.
** assumption invalidations: 89.
89 PRE violations.
** tot. num. full traces : 8
** total number of traces : 98
BBC SPORT | Fo...
E:\workshop\TT...
Java - Service...
Microsoft Power...
Rekenmachine
Administrator: c...

Trace Director

Spec. Class




```
List<Method> customTraceDirector() {

    nextsteps.clear() ;

    if (N_ < 30) { nextsteps.add(BUY) ; return nextsteps ; }
    if (N_ < 38) { nextsteps.add(AWARD) ; return nextsteps ; }
    nextsteps.add(RESOLVE) ;

    return nextsteps ;
}
```

Results

Participation System	
App Logic	99 %
Discount_1000	100 %
Discount_5pack	92 %
Java.util.LinkedList	97 %
BinarySearchTree	92 %
Hypotheek Applet	
App Logic	100 %
InkomenBel2009.rekenArbeidsKorting	70 %
EigenwoningForfait2007.reken	68 %

Conclusion

- Specification classes are not always elagant, but are practical and expressive.
- Instrument to organize your specs.
- Substantial innitial effort is needed to setup test domains and test strategies.
- But after that automated tool like T2 will deliver.

Future work

- Improving, e.g. combinatorial engine
- Stabilizing the ‘red’ components.
- Moving to Internet and mobile applications