VDM vs. Programming Languages Extensions or their Integration

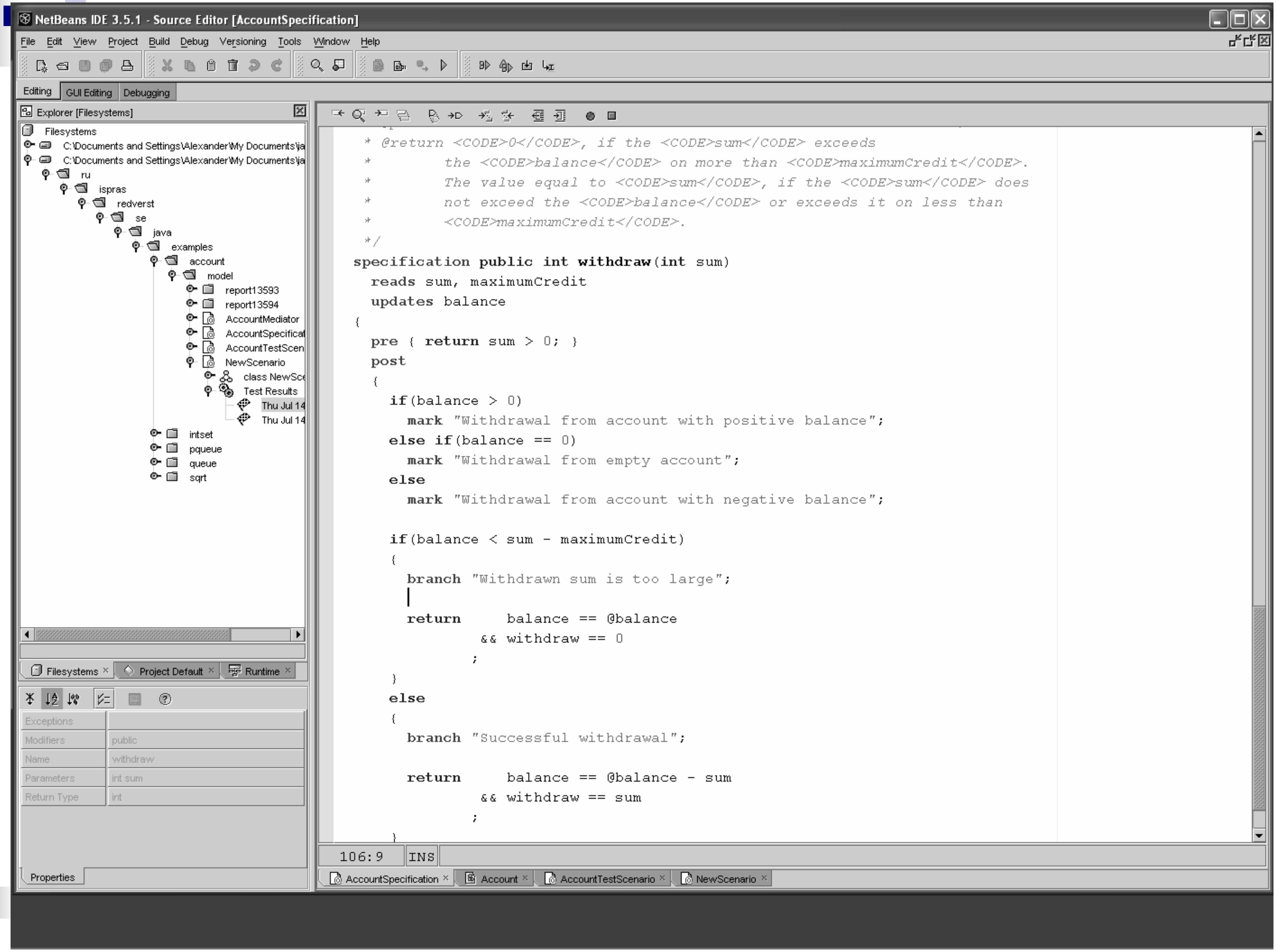
Alexander K. Petrenko
Institute for System Programming
Moscow, Russia
http://www.ispras.ru
http://unitesk.ispras.ru

Agenda

- UniTesK: experience of tool set development
- View on open semantic interfaces

UniTesK

- UniTesK is a methodology and family of tools for test development based on Design-by-Contract approach
- Language platforms: C, Java, C#
- Integrated into: NetBeans, MS VS, Eclipse (2005)
- Features:
  - Test oracle generation
  - Partition analysis (test coverage metrics generation)
  - Test scenario generation
  - Test traces analysis and visualization



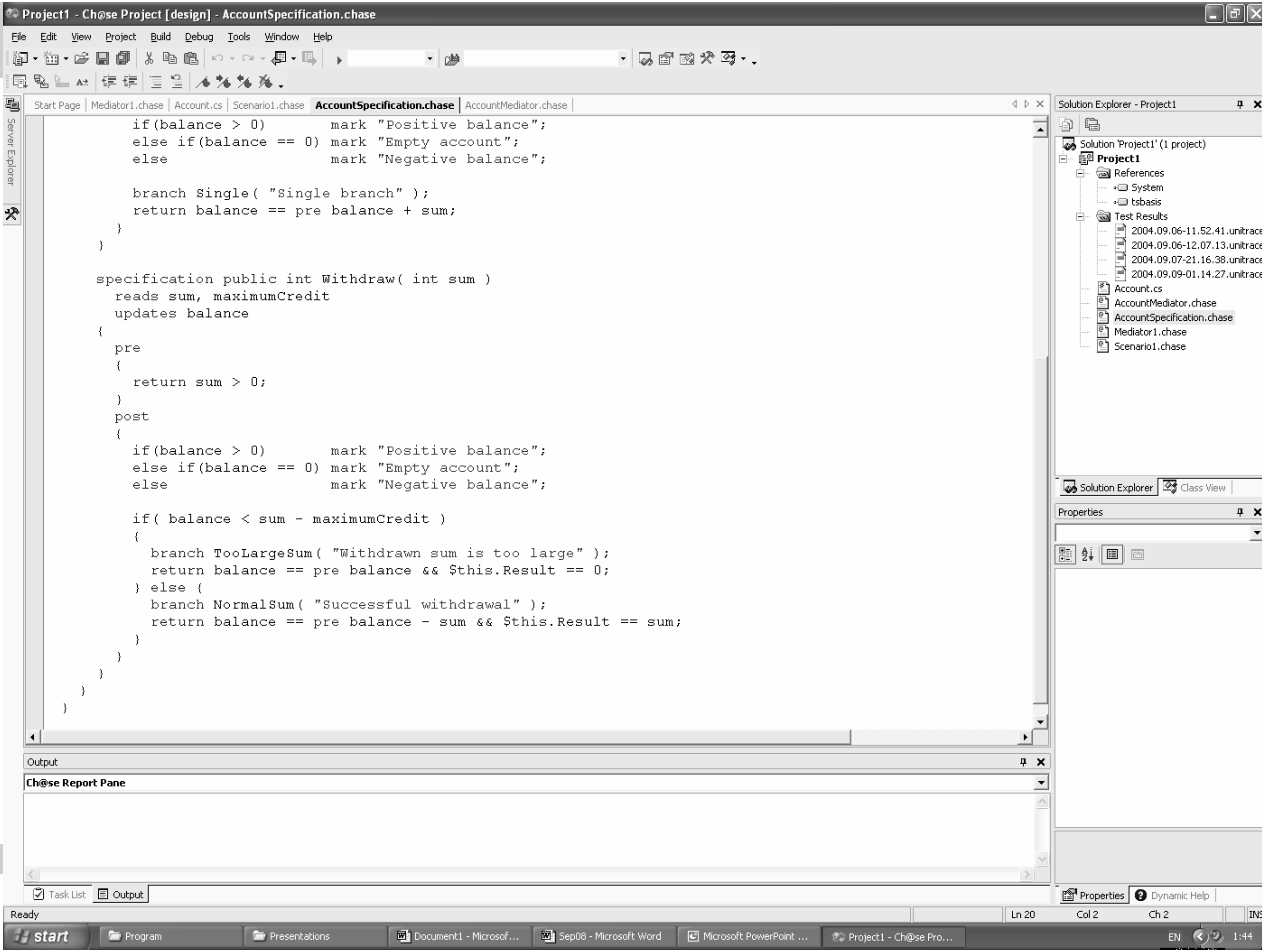
Exceptions	
Modifiers	public
Name	withdraw
Parameters	int sum
Return Type	int

```
 * @return <CODE>0</CODE>, if the <CODE>sum</CODE> exceeds
 *         the <CODE>balance</CODE> on more than <CODE>maximumCredit</CODE>.
 *         The value equal to <CODE>sum</CODE>, if the <CODE>sum</CODE> does
 *         not exceed the <CODE>balance</CODE> or exceeds it on less than
 *         <CODE>maximumCredit</CODE>.
 */
specification public int withdraw(int sum)
  reads sum, maximumCredit
  updates balance
{
  pre { return sum > 0; }
  post
  {
    if(balance > 0)
      mark "Withdrawal from account with positive balance";
    else if(balance == 0)
      mark "Withdrawal from empty account";
    else
      mark "Withdrawal from account with negative balance";

    if(balance < sum - maximumCredit)
    {
      branch "Withdrawn sum is too large";

      return     balance == @balance
              && withdraw == 0
             ;
    }
    else
    {
      branch "Successful withdrawal";

      return     balance == @balance - sum
              && withdraw == sum
             ;
    }
```

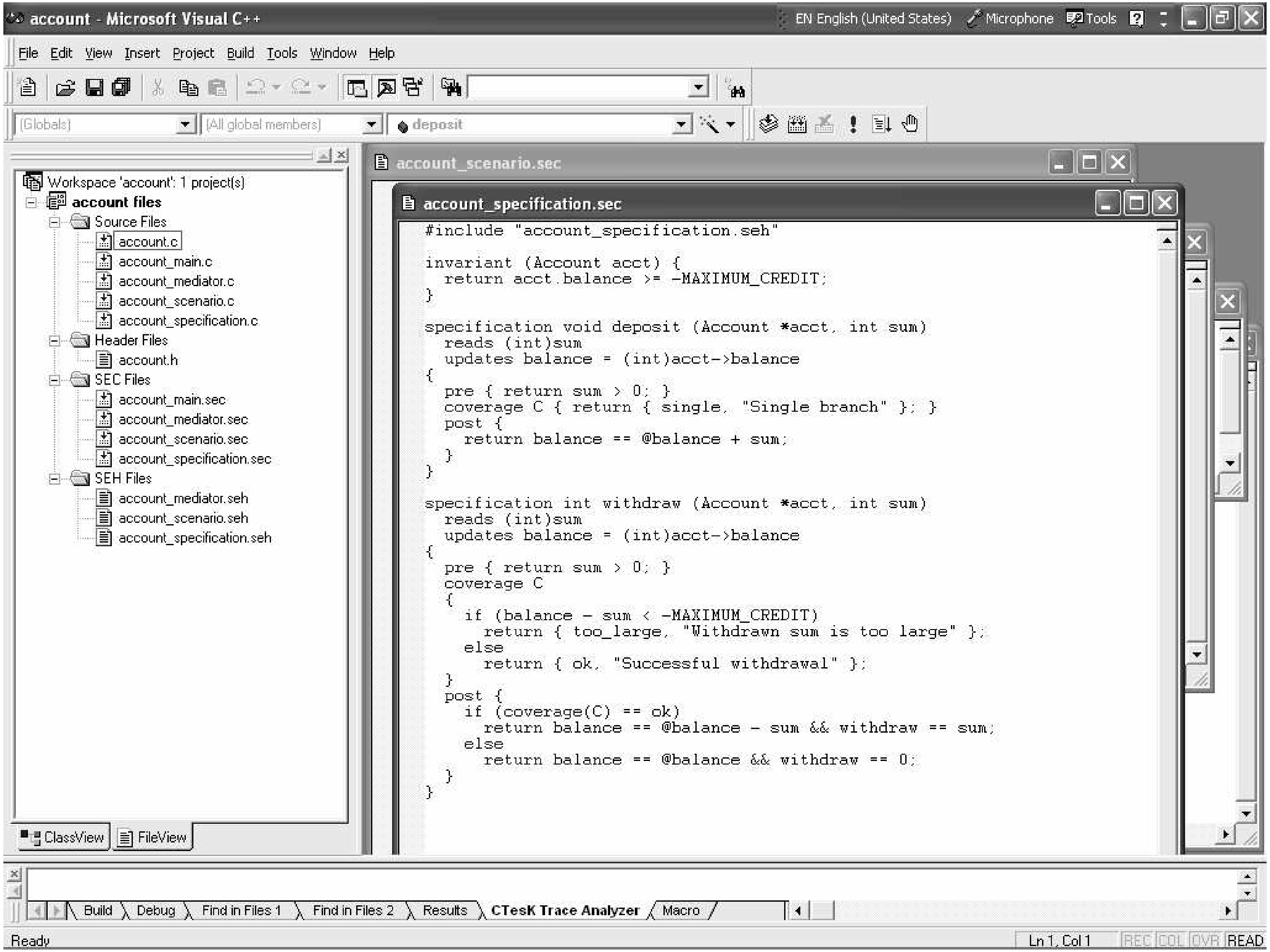

Project1 - Ch@se Project [design] - AccountSpecification.chase
File Edit View Project Build Debug Tools Window Help
Start Page | Mediator1.chase | Account.cs | Scenario1.chase | AccountSpecification.chase | AccountMediator.chase
Solution Explorer - Project1
Solution 'Project1' (1 project)
Project1
References
System
tsbasis
Test Results
2004.09.06-11.52.41.unitrace
2004.09.06-12.07.13.unitrace
2004.09.07-21.16.38.unitrace
2004.09.09-01.14.27.unitrace
Account.cs
AccountMediator.chase
AccountSpecification.chase
Mediator1.chase
Scenario1.chase
Solution Explorer
Class View
Properties
Output
Ch@se Report Pane
Task List
Output
Properties
Dynamic Help
Ready
Ln 20
Col 2
Ch 2
INS
start
Program
Presentations
Document1 - Microsof...
Sep08 - Microsoft Word
Microsoft PowerPoint ...
Project1 - Ch@se Pro...
EN
1:44

```
        if(balance > 0)        mark "Positive balance";
        else if(balance == 0) mark "Empty account";
        else                   mark "Negative balance";

        branch Single( "Single branch" );
        return balance == pre balance + sum;
      }
    }

    specification public int Withdraw( int sum )
      reads sum, maximumCredit
      updates balance
    {
      pre
      {
        return sum > 0;
      }
      post
      {
        if(balance > 0)        mark "Positive balance";
        else if(balance == 0) mark "Empty account";
        else                   mark "Negative balance";

        if( balance < sum - maximumCredit )
        {
          branch TooLargeSum( "Withdrawn sum is too large" );
          return balance == pre balance && $this.Result == 0;
        } else {
          branch NormalSum( "Successful withdrawal" );
          return balance == pre balance - sum && $this.Result == sum;
        }
      }
    }
  }
}
```

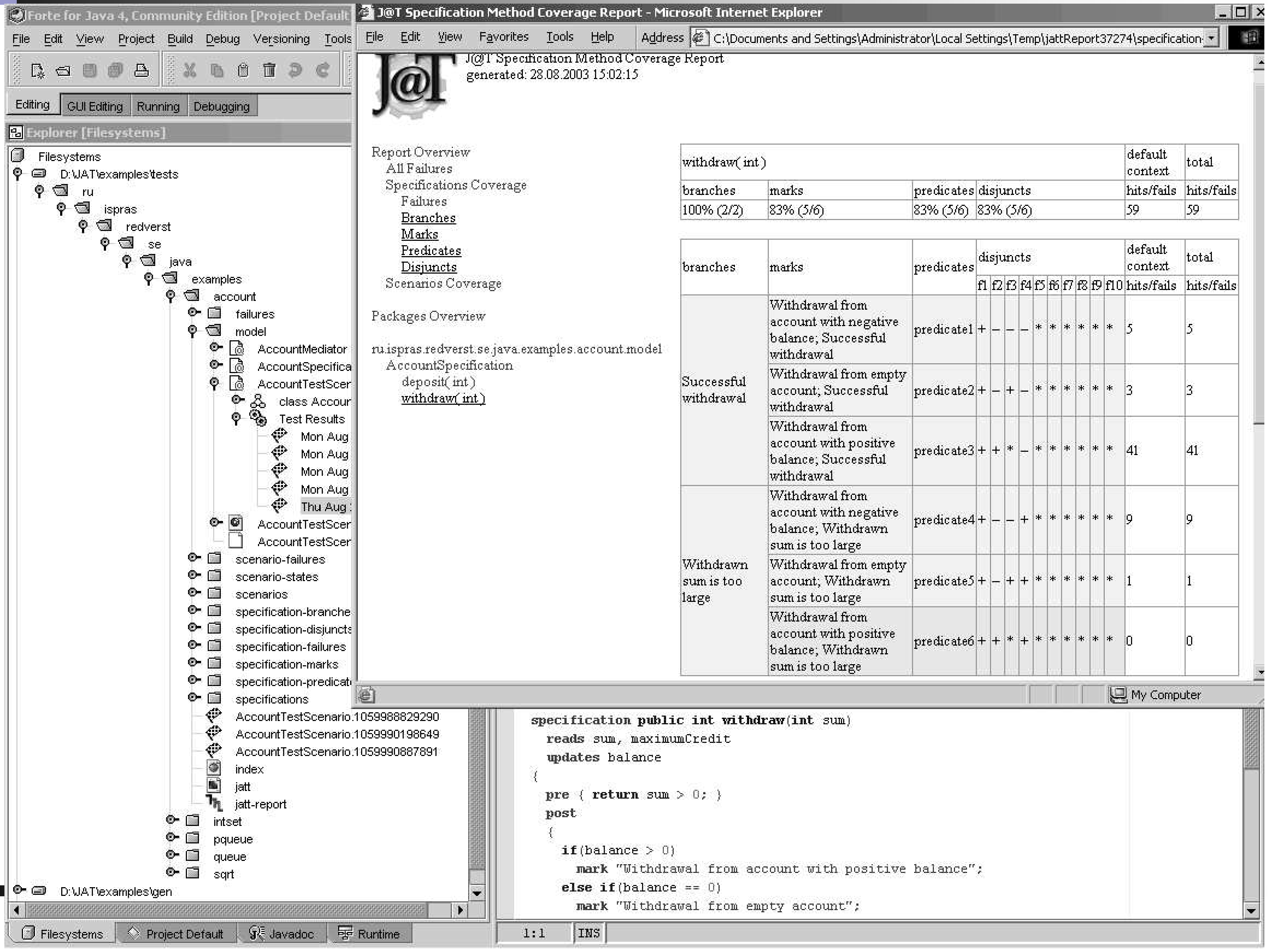


```
#include "account_specification.seh"

invariant (Account acct) {
  return acct.balance >= -MAXIMUM_CREDIT;
}

specification void deposit (Account *acct, int sum)
  reads (int)sum
  updates balance = (int)acct->balance
{
  pre { return sum > 0; }
  coverage C { return { single, "Single branch" }; }
  post {
    return balance == @balance + sum;
  }
}

specification int withdraw (Account *acct, int sum)
  reads (int)sum
  updates balance = (int)acct->balance
{
  pre { return sum > 0; }
  coverage C
  {
    if (balance - sum < -MAXIMUM_CREDIT)
      return { too_large, "Withdrawn sum is too large" };
    else
      return { ok, "Successful withdrawal" };
  }
  post {
    if (coverage(C) == ok)
      return balance == @balance - sum && withdraw == sum;
    else
      return balance == @balance && withdraw == 0;
  }
}
```


Forte for Java 4, Community Edition [Project Default
File Edit View Project Build Debug Versioning Tools
Editing GUI Editing Running Debugging
Explorer [Filesystems]
Filesystems
D:\JAT\examples\tests
ru
ispras
redverst
se
java
examples
account
failures
model
AccountMediator
AccountSpecifica
AccountTestScer
class Accour
Test Results
Mon Aug
Mon Aug
Mon Aug
Mon Aug
Thu Aug
AccountTestScer
AccountTestScer
scenario-failures
scenario-states
scenarios
specification-branche
specification-disjuncts
specification-failures
specification-marks
specification-predicat
specifications
AccountTestScenario.1059988829290
AccountTestScenario.1059990198649
AccountTestScenario.1059990887891
index
jatt
jatt-report
intset
pqueue
queue
sqrt
D:\JAT\examples\gen
Filesystems Project Default Javadoc Runtime
J@T Specification Method Coverage Report - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address C:\Documents and Settings\Administrator\Local Settings\Temp\jattReport37274\specification
J@T
My Computer
1:1 INS

J@T Specification Method Coverage Report
generated: 28.08.2003 15:02:15

Report Overview
- All Failures
- Specifications Coverage
  - Failures
  - Branches
  - Marks
  - Predicates
  - Disjuncts
- Scenarios Coverage

Packages Overview

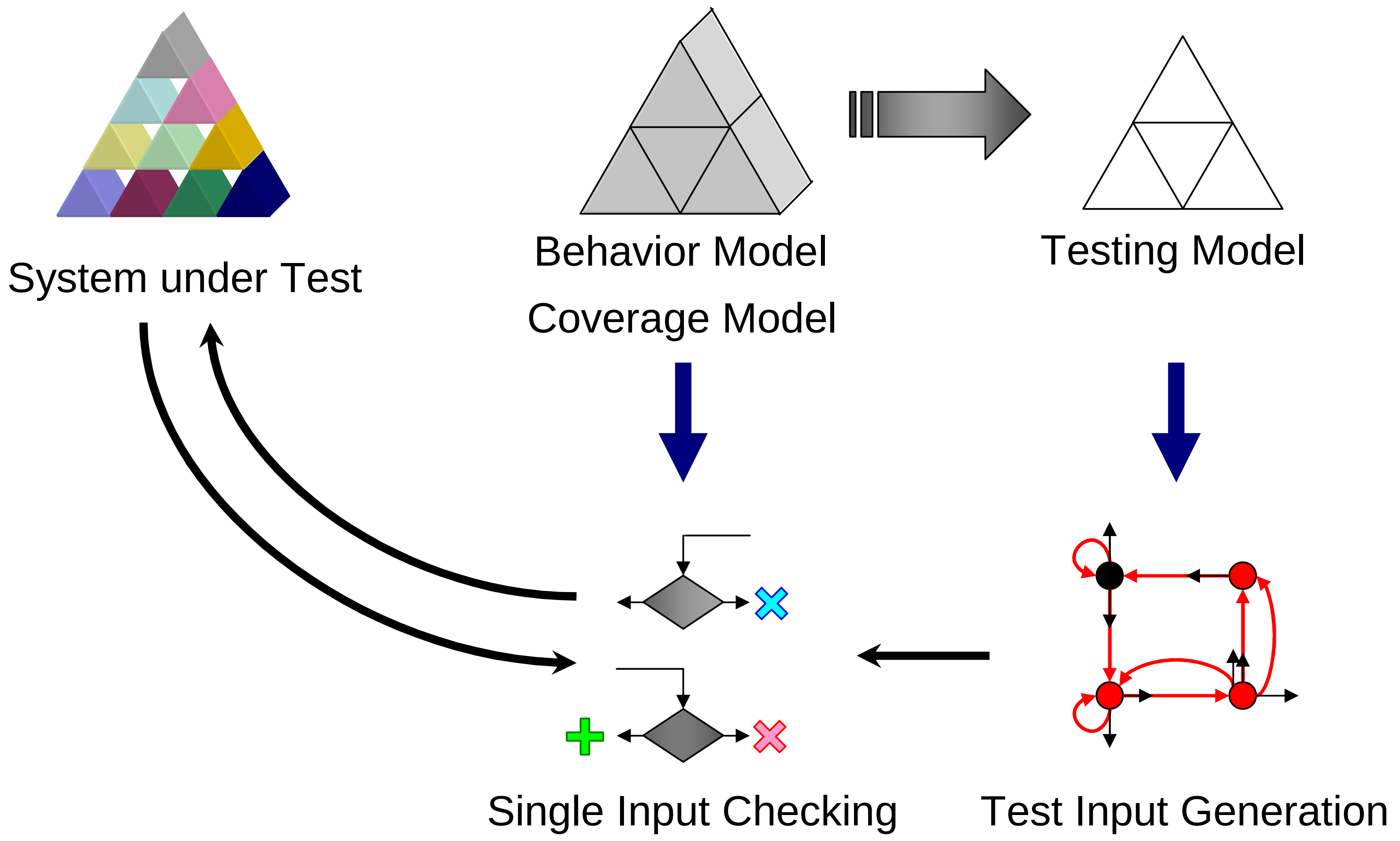
ru.ispras.redverst.se.java.examples.account.model
- AccountSpecification
  - deposit(int)
  - withdraw(int)

withdraw(int)				default context	total
branches	marks	predicates	disjuncts	hits/fails	hits/fails
100% (2/2)	83% (5/6)	83% (5/6)	83% (5/6)	59	59

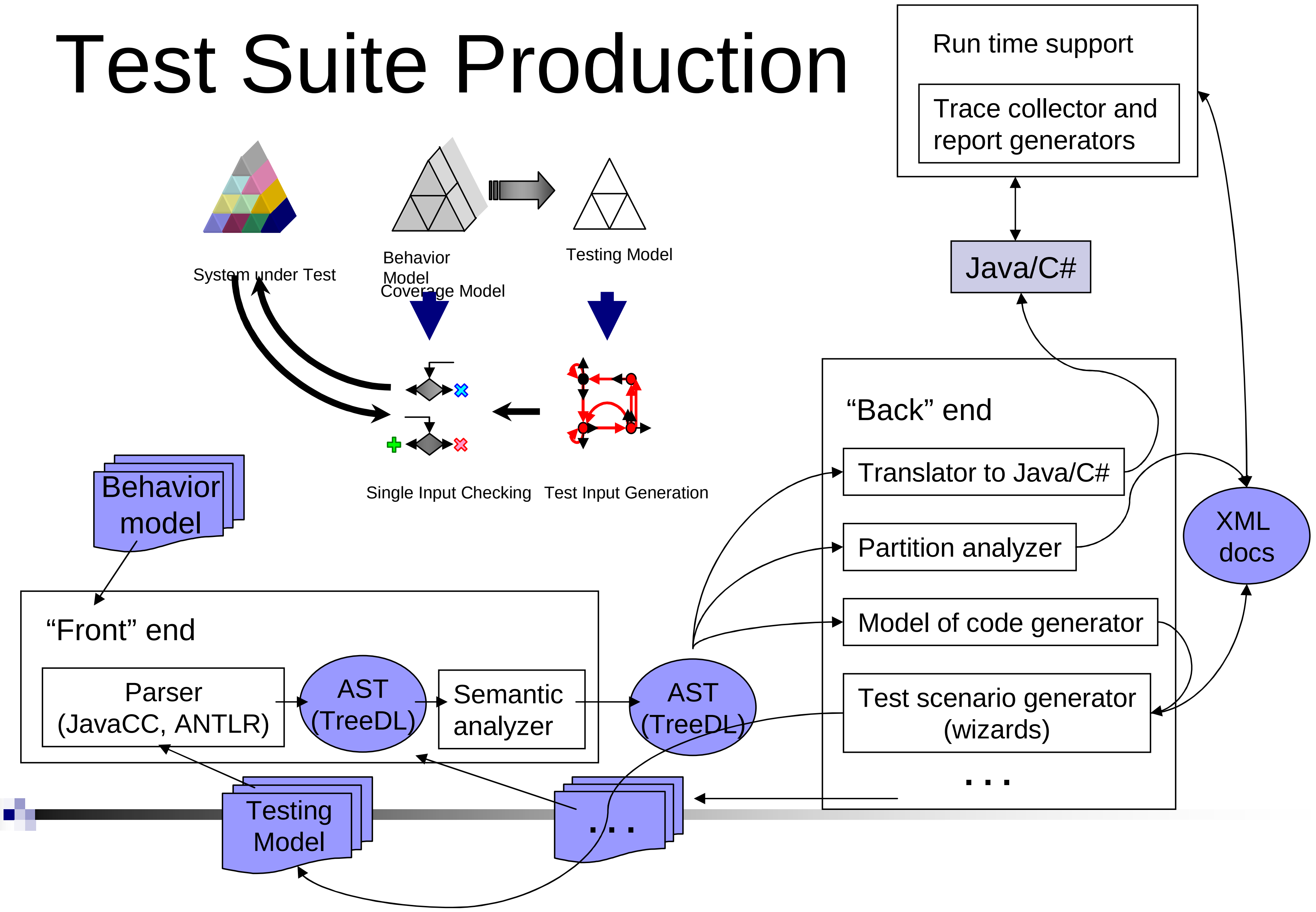
branches	marks	predicates	disjuncts										default context	total
			f1	f2	f3	f4	f5	f6	f7	f8	f9	f10	hits/fails	hits/fails
Successful withdrawal	Withdrawal from account with negative balance; Successful withdrawal	predicate1	+	–	–	–	*	*	*	*	*	*	5	5
	Withdrawal from empty account; Successful withdrawal	predicate2	+	–	+	–	*	*	*	*	*	*	3	3
	Withdrawal from account with positive balance; Successful withdrawal	predicate3	+	+	*	–	*	*	*	*	*	*	41	41
Withdrawn sum is too large	Withdrawal from account with negative balance; Withdrawn sum is too large	predicate4	+	–	–	+	*	*	*	*	*	*	9	9
	Withdrawal from empty account; Withdrawn sum is too large	predicate5	+	–	+	+	*	*	*	*	*	*	1	1
	Withdrawal from account with positive balance; Withdrawn sum is too large	predicate6	+	+	*	+	*	*	*	*	*	*	0	0

```
specification public int withdraw(int sum)
  reads sum, maximumCredit
  updates balance
{
  pre { return sum > 0; }
  post
  {
    if(balance > 0)
      mark "Withdrawal from account with positive balance";
    else if(balance == 0)
      mark "Withdrawal from empty account";
```

The Whole Picture



Test Suite Production



The Best Intermediate Representation?

- a'la AST
- XML based
- Language sensitive
- Common language features (call/return, threads, basic types, etc.)

Infrastructure for “compiler compiling”

- TreeCC
  Projects: Portable .NET, DotGNU;
  http://www.southern-storm.com.au
- TreeDL
  Project: UniTesK
  http://treedl.sf.net
- Compiler testing tools (see http://unitesk.ispras.ru):
  - BNF driven test generator (positive and negative tests)
  - Static semantic driven test generator (alpha version)

Open semantic interfaces

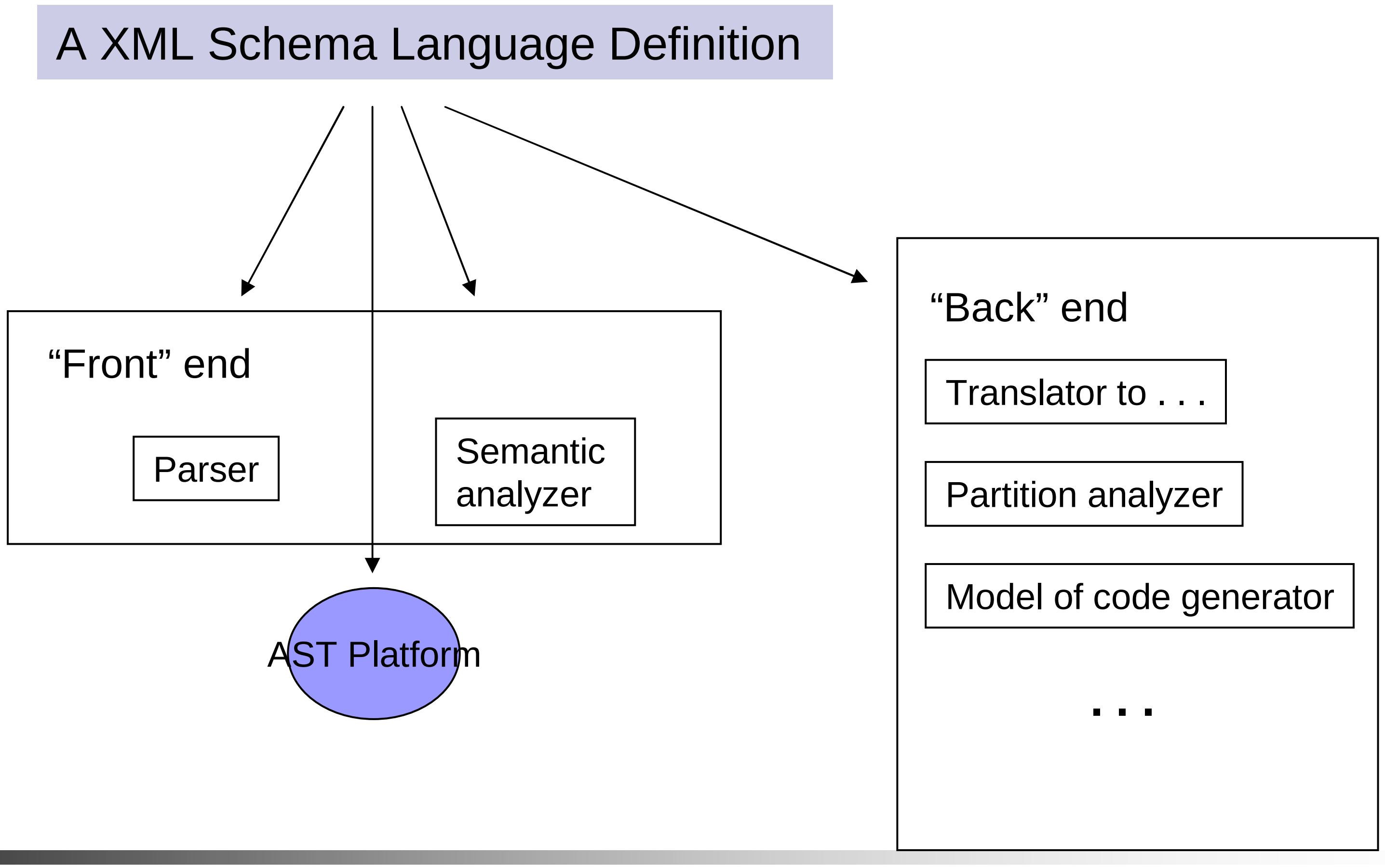
Practical View

- Overture should be positioned in industrial SW development processes, hence we should be able to answer:
  - How to combine Overture models with models in other languages
  - How to apply tools for other modeling and programming languages to Overture models
- Candidates (JML tool set, PathFinder, Spec# tools set, UniTesK, etc.)

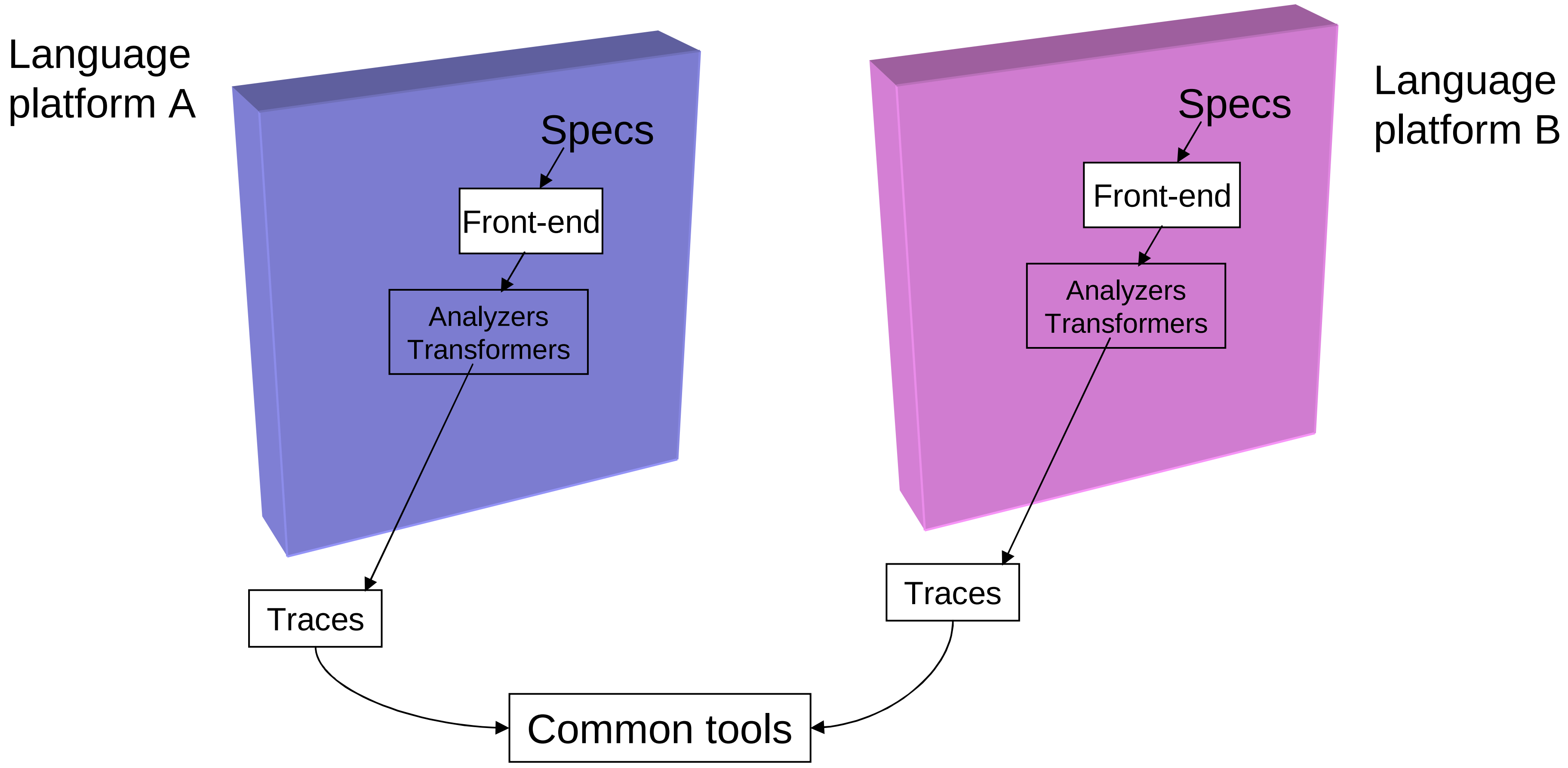
A Way for Re-use in Tool Set Development

- XML-based language definition (syntax, static semantics, “additions”)
- Generation of infrastructure and templates for particular languages
  - parser,
  - semantic analyzer,
  - model of code generator,
  - partition analyzer,
  - test generators, etc.

Overture 2: Tool Set Production



Realistic way of tool sets integration



ISPRAS Contribution

- Tools for testing parsers and other toolkit component.
- Technique of extendable semantics definitions in XML Schema
- XML docs validation techniques based on extended XML Schema
- Bridge between XML schema and TreeDL

References

1.	http://unitesk.ispras.ru	UniTesK testing tools
2.	http://treedl.sf.net	Infrastructure for AST
3.	http://www.southern-storm.com.au	TreeCC