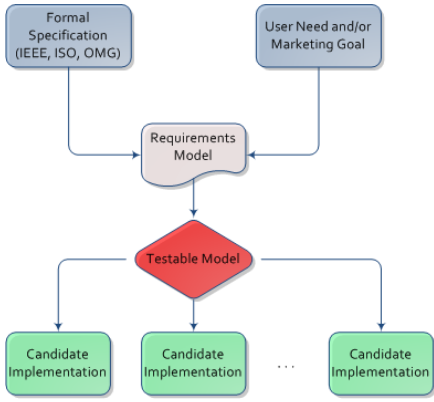
 **Carleton**
UNIVERSITY

Scenario-Based Validation Beyond the User Requirements Notation

Dave Arnold, Jean-Pierre Corriveau, and Wei Shi
Carleton University, Ottawa, Canada


April 8th 2010

Model-Based Testing



```
graph TD; A[Formal Specification (IEEE, ISO, OMG)] --> B[Requirements Model]; C[User Need and/or Marketing Goal] --> B; B --> D{Testable Model}; D --> E[Candidate Implementation]; D --> F[Candidate Implementation]; D --> G[...]; D --> H[Candidate Implementation];
```

The diagram illustrates the Model-Based Testing process. It starts with two inputs: 'Formal Specification (IEEE, ISO, OMG)' and 'User Need and/or Marketing Goal'. Both inputs feed into a 'Requirements Model'. From the 'Requirements Model', the process moves to a 'Testable Model' (represented by a red diamond). From the 'Testable Model', the process branches into three paths, each leading to a 'Candidate Implementation' (represented by a green rounded rectangle). The paths are separated by an ellipsis (...).

 **Carleton**
UNIVERSITY

© Dave Arnold 2010 Slide - 2

Testable Models



- We require a testable model capable of automatically generating executable checks.
- Such a testable model must support:
 - The capture of functional and non-functional requirements
 - Testability of the requirements model
 - Executability of the generated checks
 - Semantics rooted in the notions of responsibilities and scenarios
 - Abstraction of the testable model over several possible implementations
- Current approaches to validation typically do not offer a testable requirements model with the above characteristics...



© Dave Arnold 2010

Slide - 3

Validation Framework (Administrator)

File Edit View Tools Window Help

Start Page

Validation Framework

Version 1.1

Welcome to the Validation Framework

To create a new contract project, select New Project from the File menu.

Installed Modules

- Validation Framework Core Runtime Version 1.1: 1.1.0.0
- Contract Intermediate Language Version 1.1: 1.1.0.0
- Another Contract Language Version 3.1: 1.1.0.0
- .NET/C++ Binding Version 1.1: 1.1.0.0
- Contract Evaluation Report Version 1.1: 1.1.0.0

Validation Framework Version 1.1.0.0 (Feb 2009)
 Uses Microsoft® Cassini Web Server 3.5.
 Uses Microsoft® Platform Software Development Kit Codenamed "Phoenix" Preview Edition (April 2008 Release).
 © Dave Arnold 2006-2009.

Output

Show output from: General

```

VSIP: Developer edition disabled, normal security checks initiated.
VSIP: Third party package 'DaveArnold.VF.Languages.ACL.Project.ACLProjectPackage, DaveArnold.VF.Languages.ACL.Project, Ver
ACL Project Support - Version 1.1.0.0 (Feb 2009) - Loaded.
VSIP: Third party package 'DaveArnold.VF.Languages.CIL.Project.CILProjectPackage, DaveArnold.VF.Languages.CIL.Project, Ver
CIL Project Support - Version 1.1.0.0 (Feb 2009) - Loaded.
VSIP: Third party package 'DaveArnold.VF.UI.Core.CorePackage, DaveArnold.VF.UI.Core, Version=1.1.0.0, Culture=neutral, Pub
Core Validation Framework Package - Version 1.1.0.0 (Feb 2009) - Loaded.
  
```

Loaded Plug-ins

- Core
 - Dynamic Checks
 - UniqueValue(Boolean)
 - UniqueValue(Char)
 - UniqueValue(Double)
 - UniqueValue(Int32)
 - UniqueValue(String)
 - Metric Evaluators
 - AvgMetric(List1)
 - MaxMetric(List1)
 - MinMetric(List1)
 - Static Checks
 - HasMemberOfType(TypeSymb
 - HasMemberOfType(TypeSymb

Loaded Plug-ins Solution Explorer

Properties

Ready Ln 6 Col 22 Ch 22 INS

Bindings

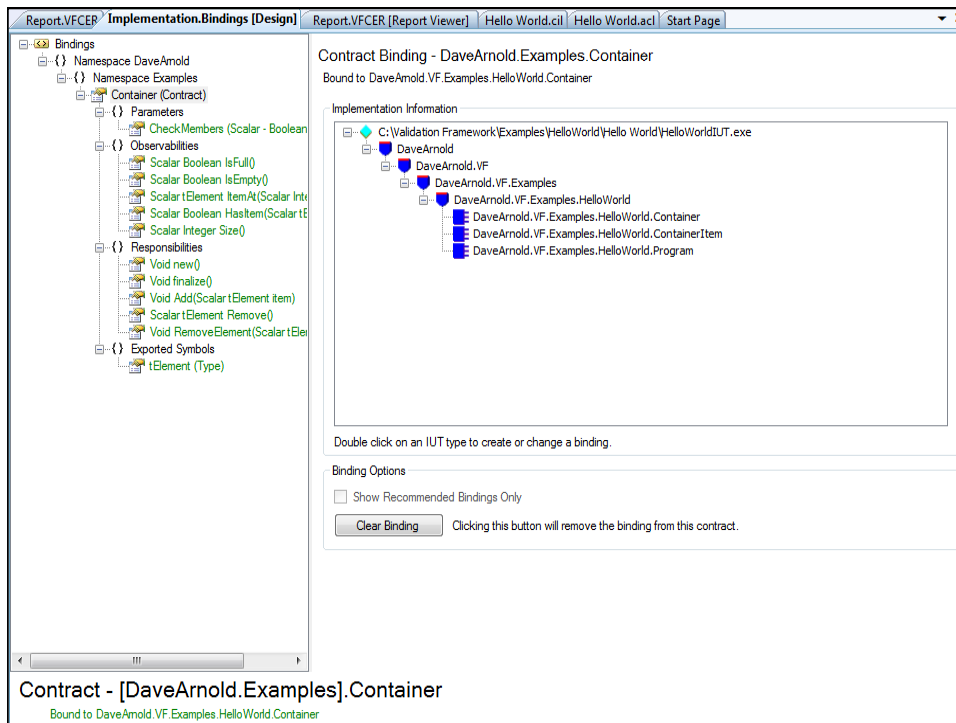
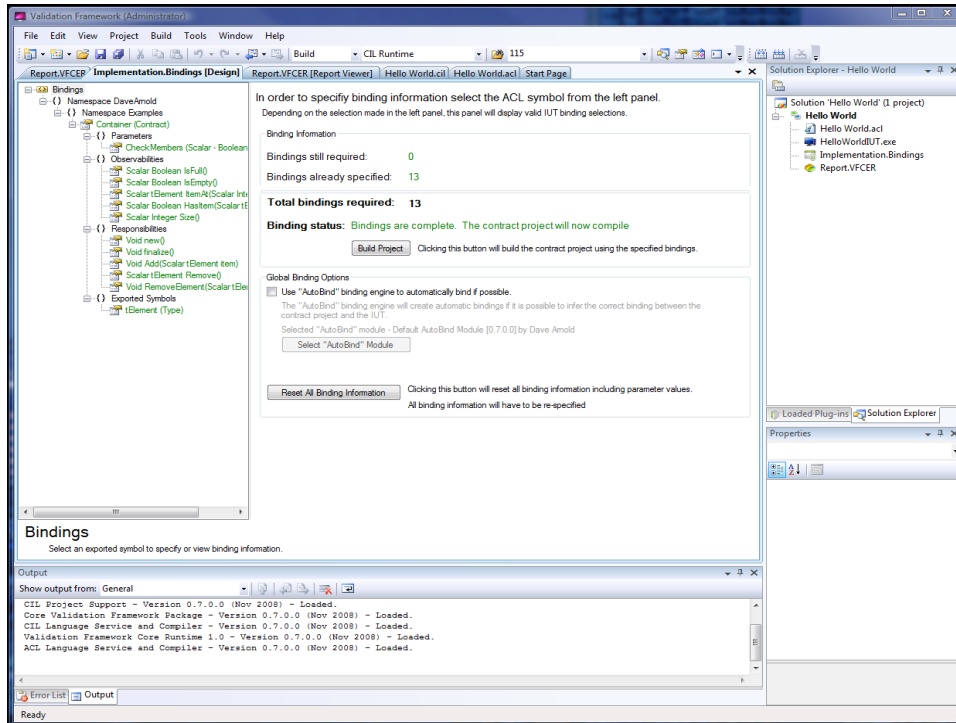


- We need to connect the testable requirements model (in ACL) to the Implementation Under Test (IUT)
 - This is accomplished through the notion of bindings
 - Bindings are a mapping between an ACL element and a IUT element:
 - Contracts → Types (Classes, Structs)
 - Observabilities → A single method or property
 - Responsibilities → One or more methods

Bindings



- In order to reduce the dependency on manual binding
 - We use binding extension modules to infer as many bindings as possible
 - Modules can be written by third-party developers.
 - When a binding cannot be inferred, a short list of possible bindings is presented, and the user is asked to make a selection
 - Each of our two sample modules achieve 95% of the required bindings
- Each binding extension module can target different implementation styles or development methodologies.



Report.VFCER Implementation.Bindings [Design] Report.VFCER [Report Viewer] Hello World.cil Hello World.acl Start Page

Observability Binding - Scalar Boolean IsFull()
Bound to method get_IsFull : bool()

Implementation Information

- DaveArnold.VF.Examples.HelloWorld.Container
 - method HasItem : bool(DaveArnold.VF.Examples.HelloWorld.ContainerItem)
 - prop IsEmpty : instance bool()
 - prop IsFull : instance bool()

Double click on an IUT method to create or change a binding.

Binding Options

Show Recommended Bindings Only

Clear Binding Clicking this button will remove the binding from this observability.

Observability - Scalar Boolean IsFull()
Bound to method get_IsFull : bool()

Report.VFCER Implementation.Bindings [Design] Report.VFCER [Report Viewer] Hello World.cil Hello World.acl Start Page

Responsibility Binding - Void Add(Scalar tElement item)
Bound to method Add : void(DaveArnold.VF.Examples.HelloWorld.ContainerItem)

Implementation Information

- DaveArnold.VF.Examples.HelloWorld.Container
 - method .ctor : void()
 - method Add : void(DaveArnold.VF.Examples.HelloWorld.ContainerItem)
 - method Finalize : void()
 - method HasItem : bool(DaveArnold.VF.Examples.HelloWorld.ContainerItem)
 - method ItemAt : DaveArnold.VF.Examples.HelloWorld.ContainerItem(int32)
 - method Remove : DaveArnold.VF.Examples.HelloWorld.ContainerItem()
 - method RemoveElement : void(DaveArnold.VF.Examples.HelloWorld.ContainerItem)
 - prop IsEmpty : instance bool()
 - prop IsFull : instance bool()

Double click above to create or change a binding.

method Add : void(DaveArnold.VF.Examples.HelloWorld.ContainerItem)

Move Up
Move Down
Remove
Remove All

Binding Options

Show Recommended Bindings Only

Set Binding Clicking this button will set the responsibility binding.

Clear Binding Clicking this button will remove the binding from this responsibility.

Responsibility - Void Add(Scalar tElement item)
Bound to method Add : void(DaveArnold.VF.Examples.HelloWorld.ContainerItem)

Report.VFCER [Report Viewer] | Hello World.cil | Hello World.acl | Start Page

Report.VFCER [Design] Implementation.Bindings [Design]

Bindings

- Namespace DaveArnold
 - Namespace Examples
 - Container (Contract)
 - Parameters
 - CheckMembers (Scalar - Boolean)
 - Observabilities
 - Scalar Boolean IsFull()
 - Scalar Boolean IsEmpty()
 - Scalar tElement ItemAt(Scalar Int)
 - Scalar Boolean HasItem(Scalar tElement)
 - Scalar Integer Size()
 - Responsibilities
 - Void new()
 - Void finalize()
 - Void Add(Scalar tElement item)
 - Scalar tElement Remove()
 - Void RemoveElement(Scalar tElement)
 - Exported Symbols
 - tElement (Type)

Type Binding - DaveArnold.Examples.Container::tElement
Bound to DaveArnold.VF.Examples.HelloWorld.ContainerItem

Implementation Information

- C:\Validation Framework\Examples\HelloWorld\Hello World\HelloWorldIUT.exe
 - DaveArnold
 - DaveArnold.VF
 - DaveArnold.VF.Examples
 - DaveArnold.VF.Examples.HelloWorld
 - DaveArnold.VF.Examples.HelloWorld.Container
 - DaveArnold.VF.Examples.HelloWorld.ContainerItem
 - DaveArnold.VF.Examples.HelloWorld.Program

Double click on an IUT type to create or change a binding.

Binding Options

Show Recommended Bindings Only

Clear Binding Clicking this button will remove the binding from this exported type.

Type - [DaveArnold.Examples].Container::tElement
Bound to DaveArnold.VF.Examples.HelloWorld.ContainerItem

Contract Evaluation Report



- The CER provides information on the IUT's execution:
 - Static evaluation results
 - For each object instance
 - Information pertaining to any dynamic checks
 - Information regarding the pass/fail of observabilities, responsibilities, and scenarios
 - Preconditions
 - Post-conditions
 - Invariants
 - Beliefs
 - Dynamic Checks
 - The result of metric analysis

Validation Framework Contract Evaluation Report
Depending on the selection made in the left panel, this panel will display different elements of the evaluation report.

Global Information

Contract Project: **ACL Project**
 Implementation Under Test: **HelloWorldIUT.exe**
 Validation Framework Runtime: **Runtime 1.0 - Version 0.7.0.0 (Nov 2008)**
 Report Date: **Sunday, November 30, 2008 at 11:41:08 PM**

Number Of Interactions: **0 Passed, 0 Failed**
 Number Of Contracts: **1 Passed, 0 Failed**
 Number Of Static Checks: **0 Passed, 0 Failed**
 Number Of Dynamic Checks: **0 Passed, 0 Failed**

Overall Result: Pass

Clicking this button will build and evaluate the current contract project.

Execution Report for SizeCheck
Displays information regarding an execution of the above invariant.

Global Information

Contract Name:
 Overall Result: **Pass** Number Of Dynamic Checks: **0 Passed, 0 Failed**
 Execution Type: **Entry** Number Of Built-in Checks: **2 Passed, 0 Failed**

Beliefs Dynamic Checks Checks

Check	Result
context.size >= 0	Pass
context.size == Size()	Pass

Additional Features



- The VF is also able to support
 - Contract refinement/inheritance
 - Atomic / parallel scenario blocks
 - Support for execution against web applications

- The VF consists of 1,355 classes totaling over 260,000 lines of C# and C++ source code

An Example Contract



```

Import Core;

Namespace DaveArnold.Examples.School
{
  MainContract University
  {
    Parameters
    {
      [1-100] Scalar Integer InstanceBind UniversityCourses;
      Scalar Integer MaxCoursesForFTStudents = 4;
      Scalar Integer MaxCoursesForPTStudents = 2;
      Scalar Integer PassRate = 70;
      [1-12] Scalar Integer InstanceBind NumTermsToComplete;
    }

    Observability List tCourse Courses();
    Observability List tStudent Students();
  }
}

```


An Example Contract



```

Responsibility new() {
    Post(Courses().Length() == 0);
    Post(Students().Length() == 0);
}
Responsibility finalize() {
    Pre(Courses().Length() == 0);
    Pre(Students().Length() == 0);
}
Responsibility tCourse CreateCourse(String name, Integer cap) {
    once Scalar Integer oldSize;
    oldSize = PreSet(Courses().Length());
    Post(value.bindpoint.Name() == name);
    Post(value.bindpoint.CapSize() == cap);
    Post(Courses().Length() == oldSize + 1);
    Post(Courses().Contains(value) == true);
}
Responsibility RegisterStudentForCourse(tStudent student, tCourse course);

```



© Dave Arnold 2010 Slide - 17

An Example Contract



```

Scenario Term {
    Trigger(new(),
    (
        CreateCourse()[Parameters.UniversityCourses],
        TermStarted(),
        fire(TermStarted),
        LastDayToDrop(),
        fire(LastDayToDrop),
        TermEnded(),
        fire(TermEnded),
        observe(MarksRecorded)[Parameters.UniversityCourses],
        CalculatePassFail(),
        DestroyCourse()[Parameters.UniversityCourses],
        fire(TermComplete)
    )+,
    Terminate(finalize());
}

```



© Dave Arnold 2010 Slide - 18

An Example Contract



```
Exports
{
  Type tCourse conforms Course
  {
    Student::tCourse;
  }
  Type tStudent conforms Student
  {
    Course::tStudent;
  }
}
```