# The Overture AST and Plug-in Development

# Kenneth Lausdahl

Department of Engineering, Aarhus University, Denmark

# 28 August 2012 / 10th International Workshop Overture/VDM

The Overture AST Creating Plug-ins For Overture

# Outline







Creating Plug-ins For Overture

# Features



# **Tool Overview**

# Core Tools

- AST
- Parser
- Type Checker
- Interpreter
- Proof Obligation Generator

### IDE

- Core Plug-in
  - Resource Management
  - Parser & TC Framework
- UI Plug-in
  - Editor
  - Outline
  - Wizards
- Language Specific Plug-ins

# **Tool Overview**

# Core Tools

- AST
- Parser
- Type Checker
- Interpreter
- Proof Obligation Generator

### IDE

- Core Plug-in
  - Resource Management
  - Parser & TC Framework
- UI Plug-in
  - Editor
  - Outline
  - Wizards
- Language Specific
   Plug-ins

# Outline









# AST Generation



The Overture AST and Plug-in Development







# AST Generation

Java Output

# Package View

- ⊿ 🚈 target/testData/simple
  - Image: Image:
  - a 🌐 org.overture.ast.analysis.intf
    - IAnalysis.java
    - IAnswer.java
    - IQuestion.java
    - IQuestionAnswer.java
  - ▲ ⊕ org.overture.ast.expressions
    - AAndBinop.java
    - D ABinaryExp.java
    - AOrBinop.java
    - EBinop.java
    - EExp.java
    - D PBinop.java
    - D PBinopBase.java
    - PExp.java
    - PExpBase.java

# Outline of ABinaryExp

🖻 📴 💘 🐒 💿 👷 ' 🗄 Outline 🕺 org.overture.ast.expressions 18 import declarations \* serialVersionUID : long Ieft : PExp op : PBinop \_right : PExp C ABinaryExp() G<sup>C</sup> ABinaryExp(PExp, PBinop, PExp) apply(IAnswer<A>) <A> : A apply(IQuestion<Q>, Q) <Q> : void apply(IOuestionAnswer<0, A>, 0) <0, A> ; A clone() : ABinaryExp o clone(Map<INode, INode>) : ABinaryExp ● ▲ equals(Object) : boolean getChildren(Boolean) : Map<String, Object> getLeft() : PExp getOp(): PBinop getRight() : PExp kindPExp() : EExp removeChild(INode) : void setLeft(PExp) : void setOp(PBinop) : void setRight(PExp) ; void

● \_ toString() : String

### AST Generation AST Source File Hierarchy

```
Abstract Syntax Tree
exp {-> package='org.overture.ast.expressions' }
    = #Binary
        . . .
#Binary {-> package='org.overture.ast.expressions'}
      {plus}
    =
        {and}
        . . .
Aspect Declaration
%exp->#Binary = [left]:exp [op]:LexToken [right]:exp
```

### AST Generation AST Source File Hierarchy



### AST Generation Java Output Features

### Node

- INode parent()
- INode getAncestor(Class<INode> classType)
- Enumerations
  - NodeEnum kindNode() return NodeEnum.EXP
  - ... EExp kindPExp() return EExp.BINARY

# AST Generation

# Analysis

# Analysis - Default visitor

• void apply(IAnalysis analysis)

Question

• <Q> void apply(IQuestion<Q> caller, Q question)

## Answer

• <A> A apply(IAnswer<A> caller)

# Question Answer

 <Q, A> A apply( IQuestionAnswer<Q, A> caller, Q question)

### AST Generation Analysis Depth First Search



# Outline







# Creating a plug-in



Creating a plug-in Development Environment

**Requirements:** 

- Java SDK
- Eclipse Classic
  - Extended with the Overture core feature

The Overture AST Creating Plug-ins For Overture

### Creating a plug-in Development Environment - Setup

## Download Eclipse Classic - www.eclipse.org





# Creating a plug-in Development Environment - Setup

## Install New Software

🕘 Java - Eclipse SDK					- C - X -
File Edit Navigate Search Project Run	Window Help				
📬 • 🔛 🕾 🛆   🗶   🅸 • 💽 • 🍳	L -   🕸 (🚳	Welcome	⇒ <del>-</del>	Quick Access	😰 🔡 Java
Il Peckage Deplorer II 😐 🛛 🖂 🥸 🤝	0	Help Contents Search Dynamic Help Key Atziek Chef Shetts Cheef Shetts Cheef Shetts Cheef Shetts Cheef Shetts About Eclipse SDK		An e	batine is net available.
	0 items	s avadoc 👊 Declaration			- 0
	Description	^ Re	source Path	Location	Туре

K. Lausdahl

The Overture AST and Plug-in Development

## Creating a plug-in Development Environment - Setup

## Add the Overture repository -

http://build.overturetool.org/builds/overtureAst2/repository/

Add Re	pository	X
<u>N</u> ame:	overture	L <u>o</u> cal
Location:	://build.overturetool.org/builds/overtureAst2/repository/	<u>A</u> rchive
?	ОК	Cancel

## Creating a plug-in Development Environment - Setup

Uncheck Group items by Category
 Select Overture Core and install

🕘 install			
Available Software Check the items that yo	u wish to install.		
Work with: overture -	ittp://build.overturetool.org/build: Find	/overtureAst2/repository/	<u>A</u> dd
Our transmission	1110	note solutione by working men an	protococca
overtare core			<i>U.</i>
Name		version 0.2.0.201	208221540.7577
Select All Des	dect All 1 item selected		
Details			
The core features of Ove	rture Tool		c More
Show only the latest ve	rsions of available software	Hide items that are alread	ady installed
Group items by catego	7	What is already installed	
Show only software ap	plicable to target environment		
☑ <u>C</u> ontact all update site	a during install to find required soft	ware	
?		< Back Next >	Einish Cancel

#### The Overture AST and Plug-in Development

# Tutorial

## Simple Class Analysis Plug-in

- For each Class count:
  - Values
  - Instance Variables
  - Functions
  - Operations

 For each Operation and Function count the total number of expressions and statements Tutorial Overview

What do we need to do?

- Create a plug-in project
- Add dependencies to Eclipse and Overture
- Add a command to invoke the functionality
- Add a menu to show the command in the UI
- Add a Handler to invoke the implementation when the command is called
- Add a view to show the result
- Add the Overture-specific VDM analysis code

# Tutorial Create plug-in project



# Tutorial Add dependencies

O 称 (3
ta Imported Packages Specify packages on which this plug-in depends without explicitly identifying t
e Add. Remove Remove Remove Properties
Turke

K. Lausdahl

#### The Overture AST and Plug-in Development

# Tutorial Add dependencies

### 📚 Dependencies

#### **Required Plug-ins**

Specify the list of plug-ins required for the operation of this plug-in.



Add	
Remove	
Up	
Down	
Properties	

a,

#### Imported Packages

Specify packages on which this plug-in depo originating plug-in.

# Tutorial Add a command

# Tutorial Add a menu

```
<extension
         point="org.eclipse.ui.menus">
 <menuContribution
    allPopups="false"
    locationURI="popup:org.overture.ide.ui.VdmExplorer">
  <menu
        label="Tutorial 1">
     <command
           commandId="overture.tutorial1.commandAnalysis"
           label="Analysis"
           style="push">
     </command>
  </menu>
 </menuContribution>
</extension>
```

# Tutorial Add a menu



# Tutorial Add a menu



The Overture AST and Plug-in Development

# Tutorial Add a Handler

```
<extension
    point="org.eclipse.ui.handlers">
    <handler
        class="org.overture.ide.analysis.AnalysisHandler"
            commandId="overture.tutoriall.commandAnalysis">
        </handler>
    </extension>
```

```
public class AnalysisHandler extends AbstractHandler
 public Object execute(ExecutionEvent event) {
    ISelection selection = HandlerUtil.getCurrentSelection(event);
    IStructuredSelection structuredSelection = (IStructuredSelection)
    Object firstElement = structuredSelection.getFirstElement();
    IProject p = ((IResource)firstElement).getProject();
    IVdmProject vdmProject =
        (IVdmProject) p.getAdapter(IVdmProject.class);
    a = new VdmAnalysis(vdmProject, HandlerUtil.getActivePart(event),
                              HandlerUtil.getActiveShell(event));
    a.runAnalysis();
```

. . . .





```
IVdmProject vdmProject =
   (IVdmProject) p.getAdapter(IVdmProject.class);
```

```
a.runAnalysis();
```

```
. . . .
```



(IVdmProject) p.getAdapter(IVdmProject.class);

a = new VdmAnalysis(vdmAroject,HandlerUtil.getActivePart(event), HandlerUtil.getActiveShell(event)); a.runAnalysis();

.... Adapt selection to a VDM Project

# Tutorial View

```
<extension
    point="org.eclipse.ui.views">
    <view
        class="org.overture.ide.analysis.AnalysisViewPart1"
        id="overture.tutorial1.view1"
        name="Analysis View"
        restorable="true">
        </view>
        </extension>
```

The implementation takes a data object and displays it in a table view. (Not shown here)

# Tutorial View



Analysis View	×	
Class	Name	Count
A	Values	1
Α	Instance Variables	1
A	Operations	1
A	Functions	1
A	fun1	3
A	op1	4
В	Values	1
В	Instance Variables	0
В	Operations	1
В	Functions	1
В	fun1	6
В	op1	7
C	Values	1
C	Instance Variables	0
C	Operations	1
C	Functions	0
С	opTest1	6

# Tutorial Status

🚱 VDM - tutor	ial1/A	.vdr	mpp - Overture Tools	-					
File Edit Na	ivig		New	•		_	_		
i 🗈 • 🗔 🖗	1		Go Into		§1 - *> <> -	,		E 🚱	VDM
VDM Explo			Сору	Ctrl+C				ine 🗖 Analysis V	
	i i		Paste	Ctrl+V			U_ Out		
D. T. L.	. >	٤.	Delete	Delete			Class	Name	Count
A tim Tutona	nu .		Move						
	AN		Rename	F2					
8	В.v 🖻	19	Import						
8	C.\ 🔛	3	Export						
	5	2	Refresh	F5					
			Close Project		it				
			Close Unrelated Projects						
			Debug As	+					
			Run As	+					
			Debug As	•					
			Team	•					
			Compare With	•					
			Restore from Local History						
	P	D	Proof Obligations	+			II		
	R.	×	Latex	÷.		Ŧ			
	_		Tutorial 1	•	Analysis	4			
0° 6	5 t		Build Path	•		_			) 🖁 🖉 🖾 🛛

# Tutorial Overture Analysis Implementation

# Check the model and run the Type Checker:

```
final IVdmModel model = project.getModel();
if (model != null && model.isParseCorrect())
{
    if (!model.isTypeCorrect())
    {
        VdmTypeCheckerUi.typeCheck(shell, project);
    }
}
```

# Run the analysis:

```
if(model.isTypeCorrect())
{
   AnalysisData data = analyse(model.getRootElementList());
}
```

# Tutorial Overture Analysis Implementation 2

```
private AnalysisData analyse(List<INode> rootElementList) {
   AnalysisVisitor visitor = new AnalysisVisitor();
   for (INode node : rootElementList)
   {
     try
     {
        node.apply(visitor);
     } catch (AnalysisException e) {
     }
   }
   return visitor.data;
}
```

# Tutorial Overture Analysis Implementation 3 Visitor

```
class AnalysisVisitor extends DepthFirstAnalysisAdaptor
  void inAClassClassDefinition (AClassClassDefinition node) {
    data.initClass(node.getName().name);
  void inAExplicitFunctionDefinition(...) {
    expCount = 0;
  void outAExplicitFunctionDefinition(... node) {
    data.fun.get(data.activeClass).put(node.getName().name, expCount);
  void defaultInPExp(PExp node) {
    expCount++;
```

# Creating a plug-in Launching the Debugger

Debug Configurations	X						
Create, manage, and run configurations Create a configuration to launch an Eclipse application in debug mode.							
Yes     Yes </th <th>Name       Overture         Main       69- Arguments)       Plug-ins)       Econfiguration       Tracing)       Environment       Common         Workspace Data       Storkspace (JoC).//runtime-New_configuration       Image: Configuration       Image: Configuration       Image: Configuration         Clear:       Storkspace (JoC).//runtime-New_configuration       Image: Configuration       Image: Configuration       Image: Configuration         Clear:       Synchropace (Joc).//runtime-New_configuration       Image: Configuration       Image: Configuration       Image: Configuration         Program to Run       Configuration before clearing       Configuration       Image: Configuration       Image: Configuration         Run as groduct:       Orgoverture ide platform.product       Image: Configuration       Image: Configuration       Image: Configuration         Java Runtime Environment       Java Runtime Environment       COL-10/Foundation-1.0 (m2)       Enginomments.       Enginomments.         Bacution environment       COL-10/Foundation-1.0 (m2)       Image: Configuration       Image: Configuration         Bacuton environment       Col-10/Foundation-1.0 (m2)       Image: Configuration       Image: Configuration         Bacuton environment       Col-10/Foundation-1.0 (m2)       Image: Configuration       Image: Configuration       Image: Configuration     </th>	Name       Overture         Main       69- Arguments)       Plug-ins)       Econfiguration       Tracing)       Environment       Common         Workspace Data       Storkspace (JoC).//runtime-New_configuration       Image: Configuration       Image: Configuration       Image: Configuration         Clear:       Storkspace (JoC).//runtime-New_configuration       Image: Configuration       Image: Configuration       Image: Configuration         Clear:       Synchropace (Joc).//runtime-New_configuration       Image: Configuration       Image: Configuration       Image: Configuration         Program to Run       Configuration before clearing       Configuration       Image: Configuration       Image: Configuration         Run as groduct:       Orgoverture ide platform.product       Image: Configuration       Image: Configuration       Image: Configuration         Java Runtime Environment       Java Runtime Environment       COL-10/Foundation-1.0 (m2)       Enginomments.       Enginomments.         Bacution environment       COL-10/Foundation-1.0 (m2)       Image: Configuration       Image: Configuration         Bacuton environment       Col-10/Foundation-1.0 (m2)       Image: Configuration       Image: Configuration         Bacuton environment       Col-10/Foundation-1.0 (m2)       Image: Configuration       Image: Configuration       Image: Configuration						
Filter matched 11 of 11 items	Apply. Revert						
?	Debug Close						

# Creating a plug-in

B VDM - tutorial1/Avdmpp - Overture Tools							
<u>F</u> ile <u>E</u> dit <u>N</u> avigate Se <u>a</u> rch <u>P</u> roject <u>R</u> un <u>W</u> indow <u>H</u> elp							
C1 ▼ 🖩 🛍 🖄 ▼ O ▼ Q₄ ▼ 🥐 🛷 ▼ ½ ▼ ⅔ ▼ 🏷 ⇔ ▼ ⇔ ▼ 🛛 🔡 😥 VOM							
🚷 VDM Explorer 🛛 🗌 🛙	E Out	ine 🗖 Analysis Vie					
🖻 🔄 🏹	class A	^	Class	Name	Count		
🔺 🏠 Tutorial1	values		Α	Values	1		
🔺 🖽 tutorial1	vi . Int - 1,		Α	Instance Variables	1		
🛞 A.vdmpp	instance variables		Α	Operations	1		
B.vdmpp	ints1 : int := 2;		Α	Functions	1		
😪 C.vdmpp			Α	fun1	3		
	functions		Α	op1	4		
			В	Values	1		
	<pre>private fun1 : int -&gt; int fun1 (a) ata:</pre>		В	Instance Variables	0		
	funit (a) ata;		В	Operations	1		
			В	Functions	1		
	operations		В	fun1	6		
	private op1 : () ==> int		В	op1	7		
	op1 () == return 1+1;		с	Values	1		
	end A		С	Instance Variables	0		
			С	Operations	1		
			С	Functions	0		
			С	opTest1	6		
		-					
	4	F.					
				e 🍳 🖁	2 🖉 🗉 🔍		

Creating a plug-in Tutorial 2: Extension

Try to extend the example given in tutorial 1 by:

- Count expressions in initialization of:
  - Values
  - Instance Variables
- Enable the analysis for Modules
- ... or what you have in mind

# Thanks

# You can download the tutorial at:

### http://tinyurl.com/overture10tutorial

# Wiki Overture Workshop 10

The Overture AST and Plug-in Development