

# Multilanguage Programming with Ada in .NET



Martin C. Carlisle  
Associate Professor  
US Air Force Academy

# .NET Framework

- Microsoft's answer to Java
  - Common Language Runtime
    - Seeks language independence– provides large library of core routines
  - C# is flagship programming language
    - Syntax very similar to Java
  - Platform independence
    - Java has advantage here, but Linux Mono version of .NET in works

# RAPTOR



About Raptor



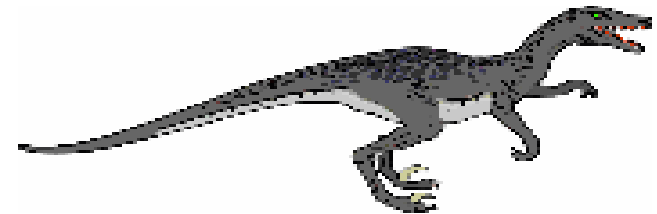
## Raptor

Rapid Algorithmic Prototyping Tool for  
Ordered Reasoning



Terry A. Wilson  
Martin C. Carlisle  
Jeffrey W. Humphries

Version 2.00  
December 1, 2003



OK

# RAPTOR

- Simple problem solving tool
- Used in CS 110, required of all cadets (including non-majors)
- Create programs visually
- Very little syntax

Raptor - test\_collapse.rap

File Edit Scale View Run Window Help

100%

**Symbols**

- Assignment
- Call
- Input
- Output
- Selection
- Loop

main

Start

*Sample program that demonstrates +/- on structures*

$i \leftarrow 1$

$a \leftarrow 0$

$long\_variable \leftarrow 10$

Decision:  $long\_variable > 0$

- Yes:  $m \leftarrow 1$  then  $m \leftarrow 2$  (highlighted)
- No:  $m \leftarrow 2$  then  $m \leftarrow 3$

Loop

$a \leftarrow a + 1$

Decision:  $i = 10$

- Yes: Loop back to start of loop
- No:  $m \leftarrow 2$  then Loop back to start of loop

End

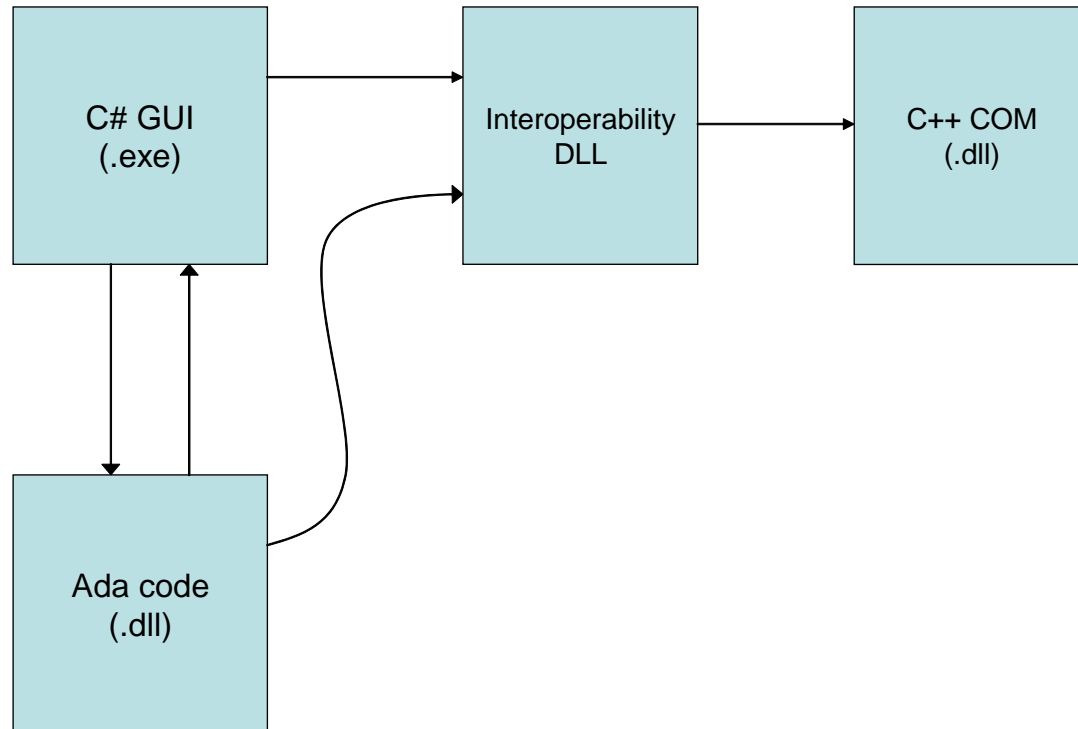
A: 0  
G: 1  
I: 1  
LONG\_VARIABLE: 10

```
graph TD
    Start([Start]) --> I1["i ← 1"]
    I1 --> A0["a ← 0"]
    A0 --> LV10["long_variable ← 10"]
    LV10 --> D1{"long_variable > 0"}
    D1 -- Yes --> M1["m ← 1"]
    M1 --> M2["m ← 2"]
    D1 -- No --> M3["m ← 2"]
    M3 --> M4["m ← 3"]
    M2 --> Loop([Loop])
    M4 --> Loop
    Loop --> Aplus["a ← a + 1"]
    Aplus --> D2{"i = 10"}
    D2 -- Yes --> Loop
    D2 -- No --> M5["m ← 2"]
    M5 --> Loop
    Loop --> End([End])
```

# Languages Used

- Ada (A#)
  - Parsing input, execution
- C#
  - GUI
- C++
  - Legacy COM object of AdaGraph routines

# RAPTOR Interoperability



# Environments Used

- AdaGIDE
  - Used to build .NET DLL via MGNAT (A#)
- Visual Studio 7.0
  - Used to build COM object (DLL)
- Visual Studio .NET
  - Included A# DLL and COM object as references
  - Created .NET EXE



# Implementing Interoperability

- Visual Studio .NET
  - Add COM object and Ada DLL as references
  - Can then call Ada and COM procedures/functions as if written in C#
- AdaGIDE
  - Use MSIL2Ada to generate Ada specs
  - Can call C# methods as if written in Ada

# Implementing Interoperability

## ■ Strings

- C# strings are like Ada's Unbounded\_String
- Conversion is done with unary "+" operator
- Ada routines calling .NET routines will automatically typecast Standard.String to C#'s System.String

# Additional A# Syntax

- Object.method syntax added (proposed for Ada 0Y)
- Pragmas for importing .NET types

# Language Advantages

- A# (language)
  - Enumeration types
  - Generics
  - Subtypes
  - Modular types
  - Decimal types
  - Can call routine stored in separate EXE

# C# enumerations

- Really named constants

```
public enum FontStyle {Bold, Italic, Underline...};
```

```
System.Drawing.FontStyle fs;
```

```
fs = FontStyle.Bold;
```

```
// Writes Bold
```

```
Console.WriteLine(Enum.GetName(fs.GetType(),fs));
```

```
fs = FontStyle.Bold | FontStyle.Italic;
```

```
// Writes a blank line
```

```
Console.WriteLine(Enum.GetName(fs.GetType(),fs));
```

# Ada enumerations

- Adding to an enumeration type cause compiler to report error on all case statements that need a new “when” clause
- Also, all assignments to arrays indexed by enumeration types are flagged.

# Example use of enumeration

```
type Token is (...
  -- adagraph procedures
  Clear_Window,Close_Graph_Window,Display_Number,
  Display_Text,Draw_Arc,
  Draw_Box,Draw_Circle,Draw_Ellipse,Draw_Line,
  Flood_Fill,Open_Graph_Window,Wait_For_Key,
  Wait_For_Mouse_Button,
  Get_Mouse_Button,
  Set_Font_Size,
  Put_Pixel,Set_Window_Title,...);

subtype Adagraph_Procedure is Token range
  Clear_Window..Set_Window_Title;
```

# Example enumeration (pt 2)

```
type Parameter_Count_Array is array(Adagraph_Procedure) of Natural;  
Parameter_Count : constant Parameter_Count_Array :=  
  (Clear_Window => 1,  
   Close_Graph_Window => 0,  
   Display_Number => 4,  
   Display_Text => 4,  
   Draw_Box => 6,  
   Draw_Arc => 9,  
   Draw_Circle => 5,  
   Draw_Ellipse => 6,  
   Draw_Line => 5,  
   Flood_Fill => 3,  
   Get_Mouse_Button => 3,  
   Open_Graph_Window => 2,  
   Wait_For_Key => 0,  
   Wait_For_Mouse_Button => 1,  
   Put_Pixel => 3,  
   Set_Font_Size => 1,  
   Set_Window_Title => 1);
```



# Example enumeration (pt 3)

```
Kind : Adagraph_Procedure;  
begin  
  case Kind is  
    when Clear_Window =>  
      Dotnetgraph.Clear_Window(  
        Color_Type.ValueType'Val(  
          Numbers.Integer_Of(  
            Check_Parameter_Number(  
              This.Param_List.  
                Parameter)))));
```

# Language Advantages

- C# (environment)
  - More sophisticated GUI Design Tool (VS .NET vs RAPID)
  - Automatic import of COM objects
  - Sophisticated auto-completion

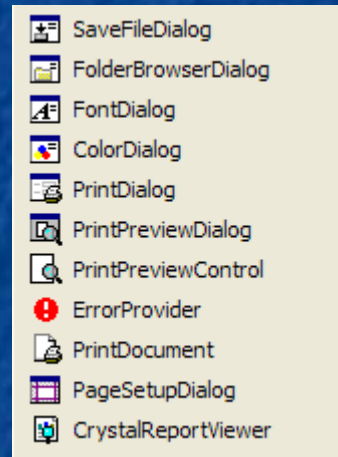
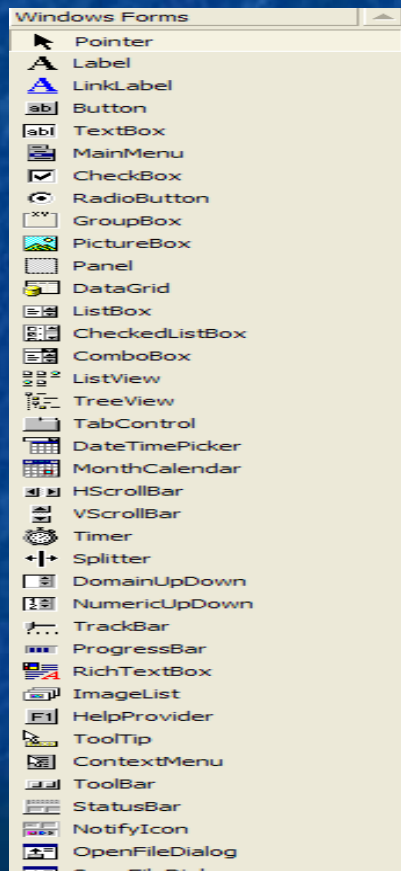
# VS GUI Designer

The image displays the Visual Studio GUI Designer interface. On the left, a form titled 'MasterConsole' is shown in design mode. The form has a menu bar with 'File', 'Font Size', 'Edit', and 'Help'. The main area is mostly empty, with a 'Clear' button at the bottom right. A status bar at the bottom left shows 'mainMenu1'. On the right, the 'Properties' window is open, showing the properties for the 'MasterConsole' control. The control is of type 'System.Windows.Forms.Form'. The properties list includes:

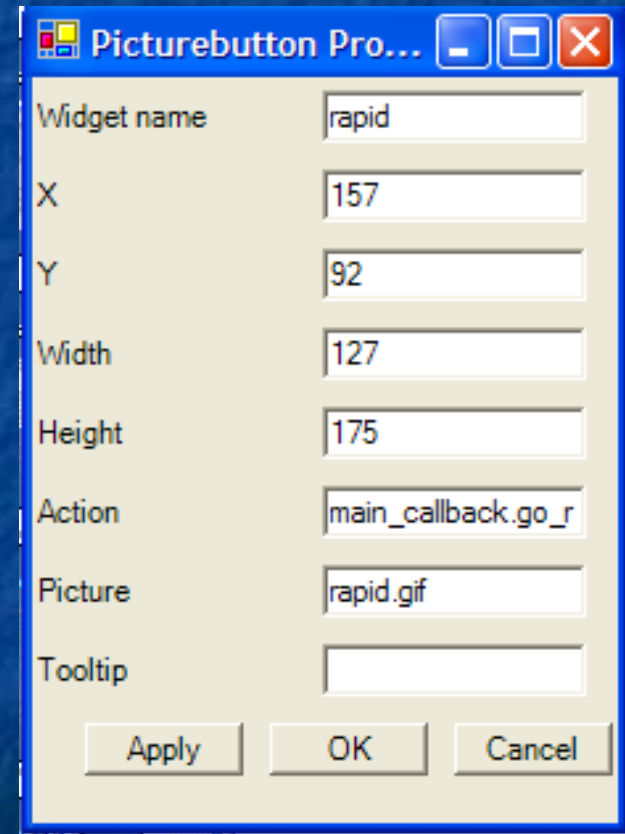
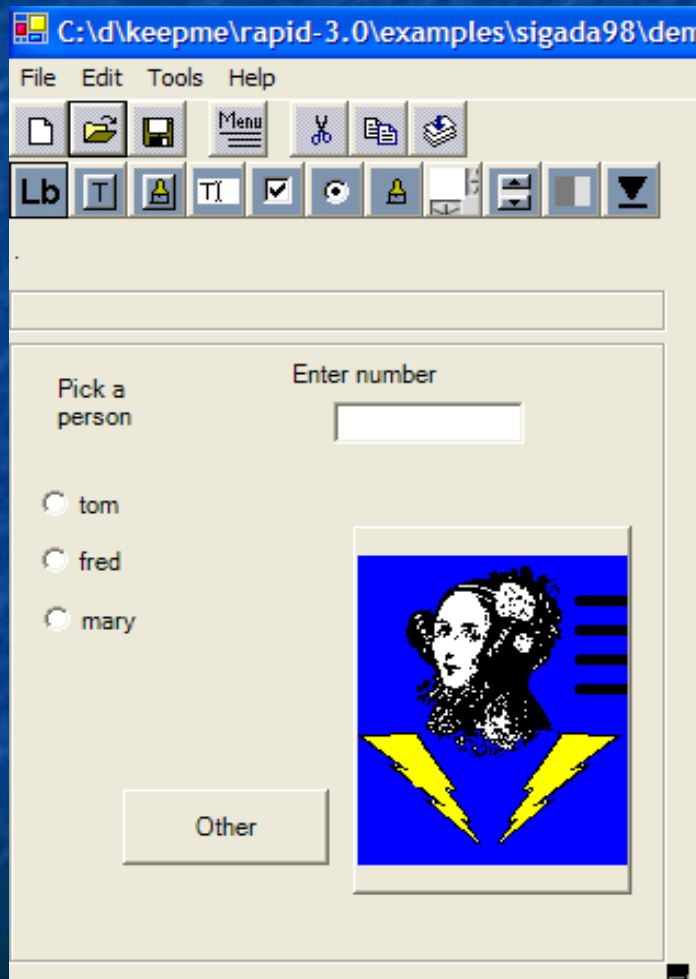
Property	Value
(DataBindings)	
(DynamicProperties)	
(Name)	MasterConsole
AcceptButton	(none)
AccessibleDescription	
AccessibleName	
AccessibleRole	Default
AllowDrop	False
AutoScale	True
AutoScroll	True
AutoScrollMargin	0, 0
AutoScrollMinSize	0, 0
BackColor	Control
BackgroundImage	(none)
CancelButton	(none)
CausesValidation	True
ContextMenu	(none)
ControlBox	True
Cursor	Default
DockPadding	
DrawGrid	True
Enabled	True
Font	Microsoft Sans Serif, 8.25pt
ForeColor	ControlText
FormBorderStyle	Sizable
GridSize	8.8

Below the properties list, the 'Text' property is described as 'The text contained in the control.' The Properties window also includes a 'Dynamic Help' button.

# VS GUI Designer

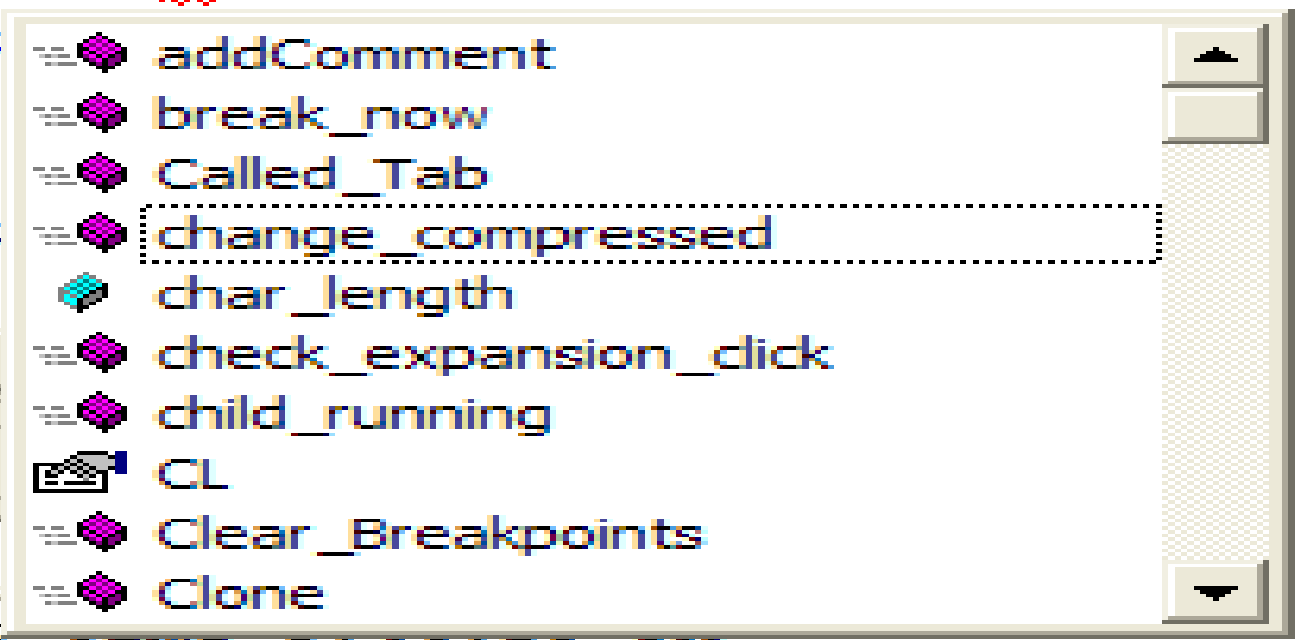


# RAPID GUI Designer



# VS .NET Autocompletion

```
get  
{  
    Start. ~  
    ret  
}  
  
lic Syst  
started at
```



- addComment
- break\_now
- Called\_Tab
- change\_compressed
- char\_length
- check\_expansion\_click
- child\_running
- CL
- Clear\_Breakpoints
- Clone

# VS .NET Hints

```
Start.check_expansion_click(|  
bool Component.check_expansion_click (int x, int y) ol1.SelectedTab) .panel1;
```

# Language Advantages

- C++
  - Umm, well, ... it was already written ☺
  - Interface Definition Language (IDL) was awkward to use, especially when passing strings
  - Did provide relatively easy mechanism for calling native Win32 methods



# A# tools

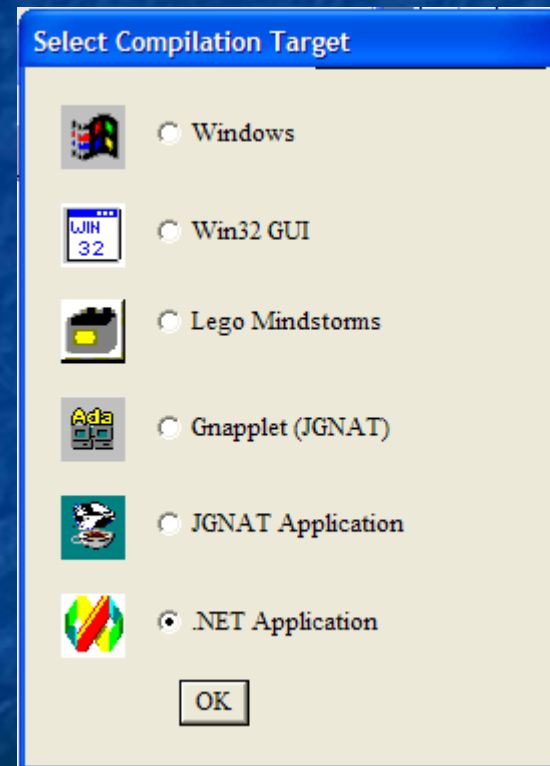
- MGNAT
  - Based on JGNAT, Ada to JVM compiler
  - Generates Microsoft Intermediate Language (MSIL)
  - Not full implementation of Ada 95
    - E.g. no representation clauses, nested controlled types not working

# A# tools

- MSIL2Ada
  - Takes in MSIL and generates Ada specification files
- RAPID
  - Multi-platform, multi-implementation GUI Design Tool
    - .NET, Tcl/Tk, Gtk and JVM implementations
    - Recompile with different library to port to different implementation

# A# tools

- AdaGIDE
  - Target button allows selection of different compiler (GNAT for Windows, JGNAT, MGNAT)



# Download Locations

- MGNAT (Ada compiler for .NET)
- MSIL2Ada (creates Ada specs for .NET files)
  - [http://www.usafa.af.mil/dfcs/bios/mcc\\_html/a\\_sharp.html](http://www.usafa.af.mil/dfcs/bios/mcc_html/a_sharp.html)
- AdaGIDE
  - [http://www.usafa.af.mil/dfcs/bios/mcc\\_html/adagide.html](http://www.usafa.af.mil/dfcs/bios/mcc_html/adagide.html)
- RAPID
  - <ftp://ftp.usafa.af.mil/pub/dfcs/carlisle/usafa/rapid/index.html>

# European Mirror Sites

- AdaGIDE:

- <ftp://sunsite.informatik.rwth-aachen.de/pub/mirror/ftp.usafa.af.mil/pub/dfcs/carlisle/adagide/>

- A#

- <ftp://sunsite.informatik.rwth-aachen.de/pub/mirror/ftp.usafa.af.mil/pub/dfcs/carlisle/asharp/>

- RAPID

- <ftp://sunsite.informatik.rwth-aachen.de/pub/mirror/ftp.usafa.af.mil/pub/dfcs/carlisle/usafa/rapid>