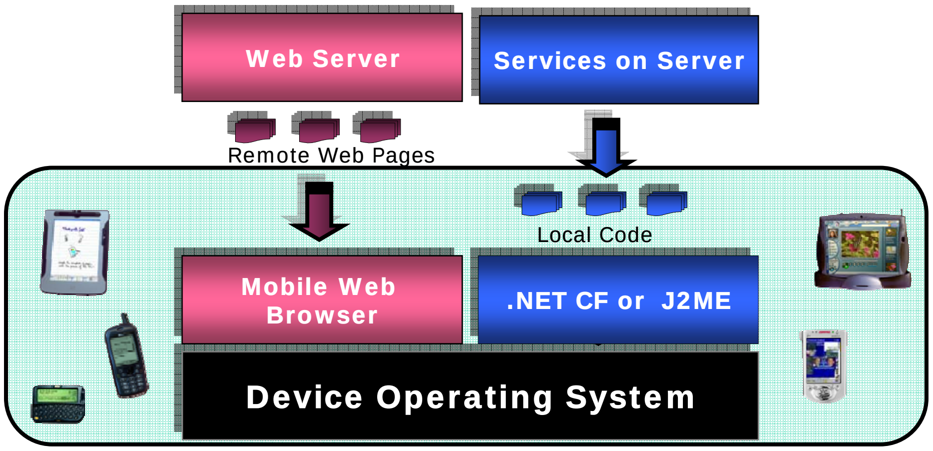
.NET CF Development

David Tipper
Associate Professor
Department of Information Science and Telecommunications
University of Pittsburgh

tipper@tele.pitt.edu
http://www.sis.pitt.edu/~dtipper/2936.html
Slides 9

Thin Client Vs. Smart Client



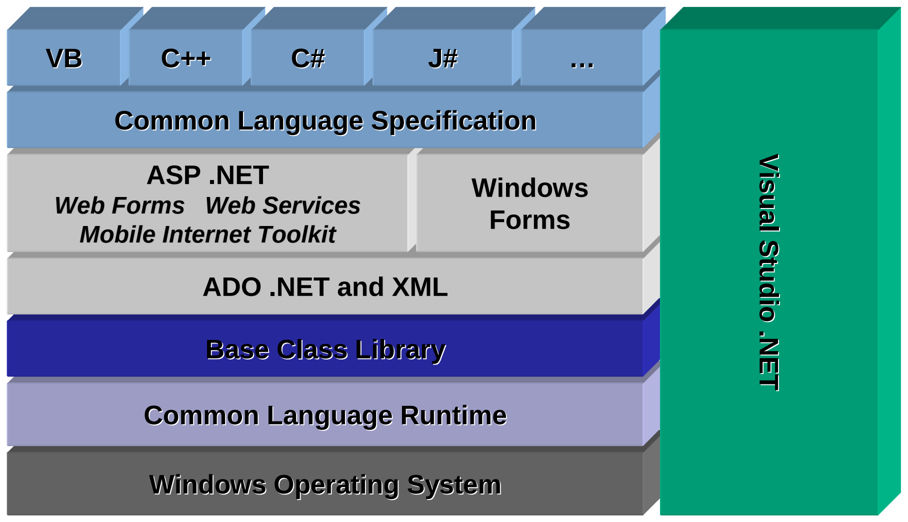
What is Microsoft's .NET ?

- .NET Framework is set of products and technologies primarily aimed at developing and deploying XML based web services
  - '.NET is Microsoft's platform for a new computing model built around XML Web Services'
    Microsoft Corporation Annual Report, 2001
- A core feature of .NET Framework is Microsoft's Common Language Infrastructure (CLI) standard
  - Source code and complied binaries in Microsoft Intermediate Language (MSIL) can run across CLI-based heterogeneous devices
  - Microsoft's Common Language Runtime (CLR), like Sun's JVM, has the objective of platform independence

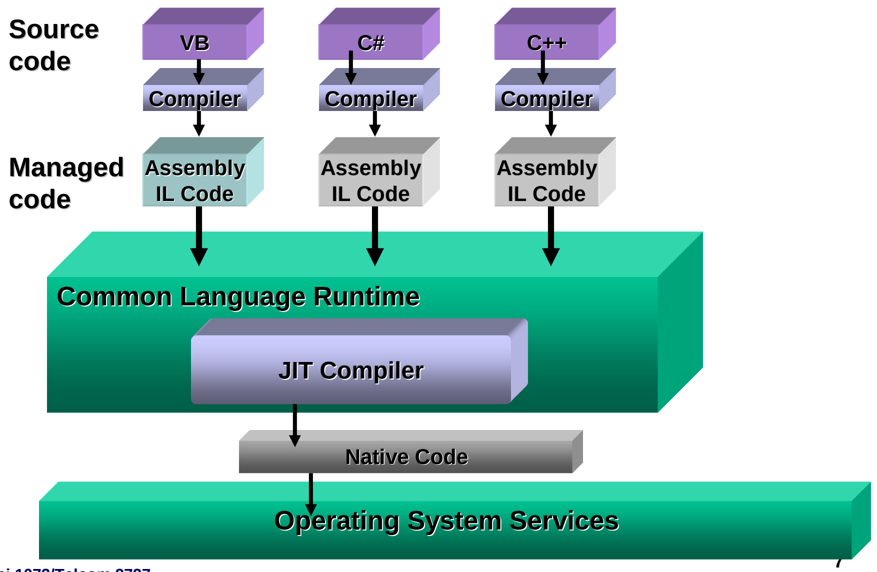
.NET Technologies

- Common Language Runtime (CLR) Manag code
  - Threading, Memory management
  - Auto-versioning
  - Code access + Role-based security
  - Integrated with underlying OS
- .NET Framework provides a set of tools for developers to build a variety of applications
  - Supports Visual Basic.NET, C#, J#, C++, … can use any language
  - Source code is compiles into MSIL within an assembly
  - Assemblies contain meta data and are primary units of deployment
  - MSIL is compiled into native code and executed by CLR
- Primary .Net development tool
  - Visual Studio.Net

Visual Studio .NET and .NET Framework

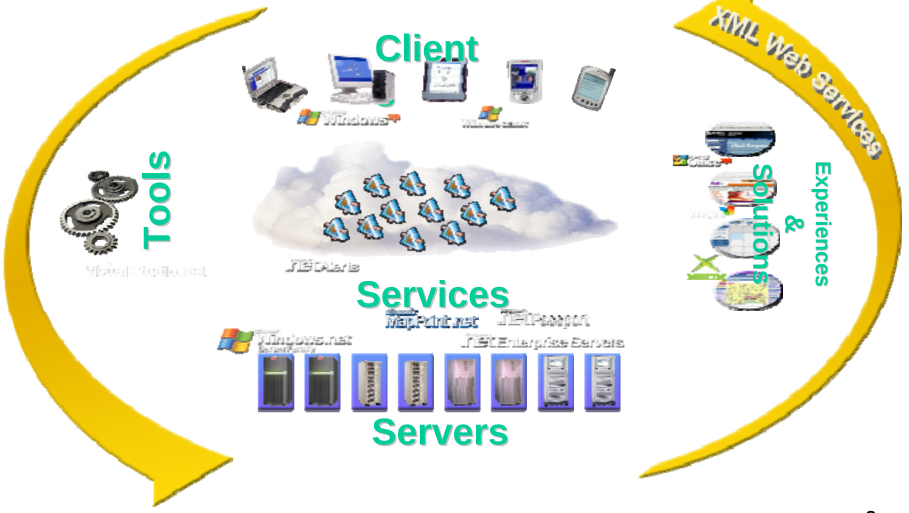


.Net Execution Model



Microsoft .NET Vision

Web services support across the Microsoft platform



.NET CF Design Goals

- Target mobile and embedded devices
- Portable subset of .NET Framework
  - Visual Basic .NET and C# compiler support in v1
  - Framework size 1.35MB (ROM) on Windows CE .NET Device
  - Typical application sizes 5 - 100 KB
- Leverage Visual Studio .NET
  - (CF is add-in for Visual Studio .NET 2005)
  - Run managed .EXEs and .DLLs directly
  - Debug with Visual Studio .NET – develop just as desktop app.
- Peacefully co-exist with host OS
  - Run on native threads, P/Invoke to call native code
- Use standardized Internet protocols
  - XML-based Simple Object Access Protocol (SOAP)
  - Web Service Description Language (WSDL)

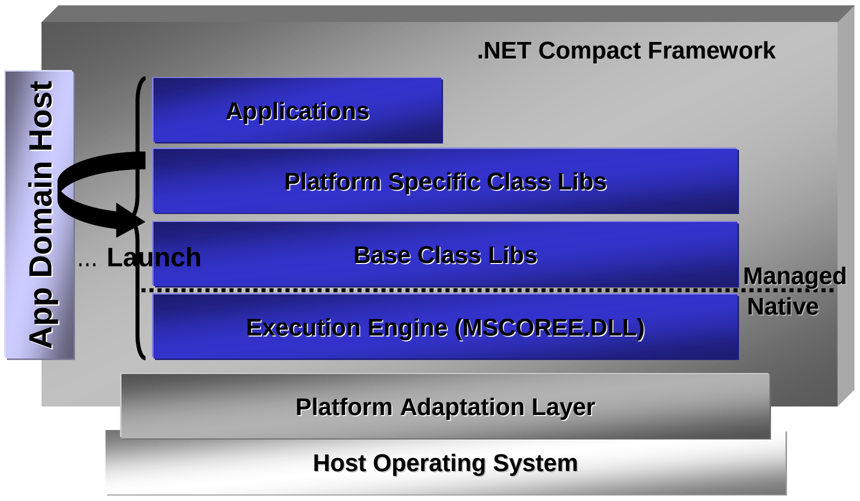
Slide 8

PJW1 Crop out the white square and "premium" from around the arrows. Leave the white in the oval inside the arrows.

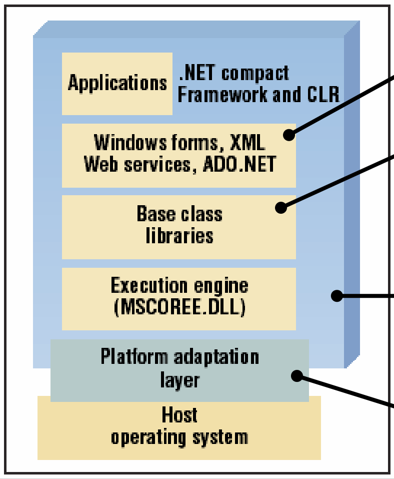
Make the arrows wrap around the outside of the big XML oval.

We need to show the 4 areas of the .NET nicely. One idea I had was to do the aperture idea we're currently using for Deborah Tom and swirl 4 quadrants in at the center to show they all get swooshed in to the centeral .net connection. Whatever we do on this slide, needs to be supported on teh next slide.
v-paulaw, 6/12/2002

Architecture



.NET CF Architecture

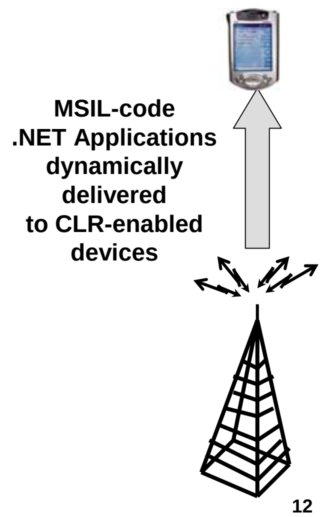

Applications
.NET compact Framework and CLR
Windows forms, XML Web services, ADO.NET
Base class libraries
Execution engine (MSCOREE.DLL)
Platform adaptation layer
Host operating system

- Available to applications
  - Windows forms library
  - XML web services
  - ADO.NET (for remote data-access)
- CLI-compliant base class libraries provide building block functionality for all applications (basic file I/O, networking, XML)
- CLR runs MSIL and uses a just-in-time (JIT) compiler to convert MSIL to native code
- Platform adaptation layer (PAL) tailored for a specific platform

C. Neable, "The .NET Compact Framework," *IEEE Pervasive Computing*, Vol. 1, No. 4, Oct.-Dec. 2002, pp. 84-87.

.NET CF Functionality

- Platform independence
  - The same .NET MSIL-code application can be downloaded and executed by CLR enabled devices
- Data Transformation
  - XML transformed automatically to HTML, XHTML, cHTML, or WML (at the server end)
- Mobile Data access
  - ADO. NET supports mobile applications accessing Microsoft SQL Server on remote servers or access a SQL Server CE locally on the device
- Disconnected operations
  - Data caching, pre-fetching, and synchronization available using SQL Server CD



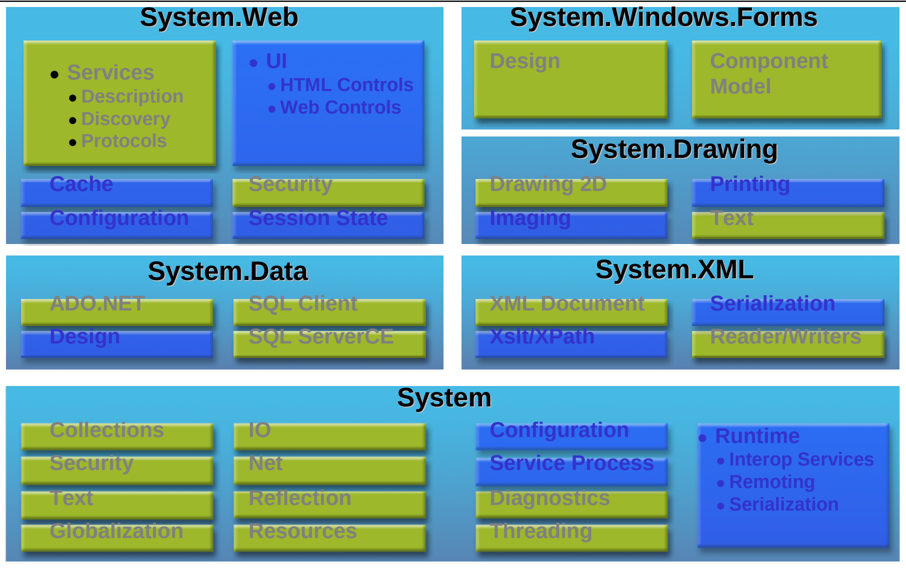
.NET CF vs. NET

- Common Base Classes
  - IO, collections, reflection, math, drawing
- Connectivity
  - Networking, HTTP classes, calling XML Web services
- Data Access
  - ADO.NET, SQL Server CE, SQL Server
- XML, XmlDocument, XmlReader/Writer
- Windows Forms
- Verifiable execution
- JIT compilation

.NET CF vs. NET

- Class libraries are a subset of .NET (~25%)
- Different Size and scalability characteristics
- .CF Additions (
  - IrDA support,
  - Device specific controls
  - SQL Server CE managed classes
  - Telephony functions
    - For example SMSINVOKE – will access SMS transmission on device

.NET Compact Framework

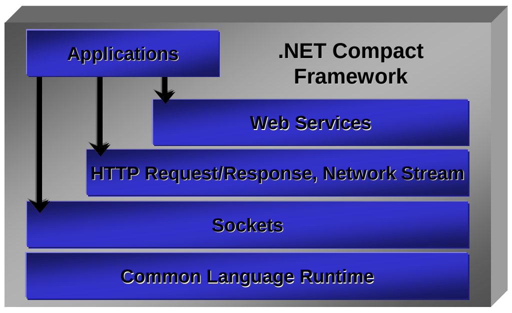


Data and Networking

- Base data types are the same as the desktop
  - Formatting
  - StringBuilder
    - More efficient when string length changes
  - Arrays
  - Value types (Int16, Int32, Int64, UInt16, etc.)
  - Floats and doubles
- Collections
  - Classes for storing sets of objects
  - Arraylists and Hashtables
- Networking
  - Sync and asynch sockets and streams, HTTP

Base: Networking

- Sockets
  - Synchronous and asynchronous
  - Multiple protocols
- Streams
  - Built on top of sockets
  - Synchronous and asynchronous
- HTTP request and response
  - Use stream model
  - Requires no user knowledge of HTTP



Data Choices

- Remote data
  - XML Web Services, ADO.NET (.NET Data Providers), Networking
- On Device data
  - Handle with XML, ADO.NET (DataSet)
  - Cache for use offline with SQL CE, ADO.NET (DataSet persistence as XML)
- Intelligent synchronization of data when connected
  - SQL CE Synchronization, ActiveSync

.CF Development

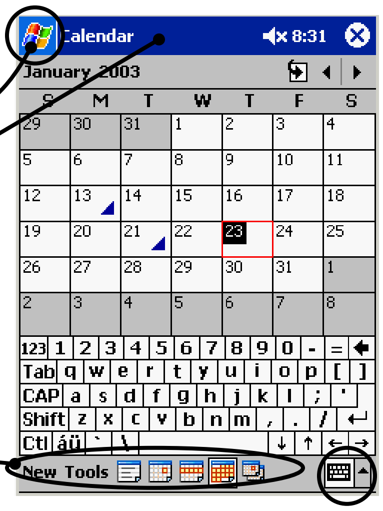
- Applications start with an initial thread
- Applications can start new threads
- Using threads
  - Responsive UI
  - Program function segregation
- Thread synchronization primitives provided
- App domains exist until all threads exit
- Managed ➔ native (P/Invoke)
  - Calls into existing native code

Windows Forms Support

- Layout
- Drawing
  - Polygons, lines, arcs, ellipses, rectangles
  - JPEG, BMP images
- Text and images
  - TrueType bitmap fonts on Mobile
- Most desktop controls
- Designer support

Example PocketPC UI Form

- System-level and navigation actions at the top of the screen
  - Start Menu
  - Navigation Bar (top)
- Application-level and editing actions at the bottom of the screen
  - Menu bar (also called command bar)
  - Input Panel Button



Supported Controls

- **Supported controls**

Button
CheckBox
ComboBox
ContextMenu
DataGrid
DomainUpDown
FileOpenDialog
HScrollBar
ImageList
Label
ListBox
ListView
TreeView
FileSaveDialog
MainMenu
NumericUpDown
Panel
PictureBox
ProgressBar
RadioButton
StatusBar
TabControl
TextBox
Timer
ToolBar
VScrollBar

- Unsupported controls

GroupBox
Printing Controls
RichTextBox
NotificationBubble (PPC)

- Unsupported controls – not available in CE

CheckedListBox
ColorDialog
ErrorProvider
HelpProvider
LinkLabel
NotifyIcon
ToolTip
Splitter
FontDialog

.NET CF Application Development

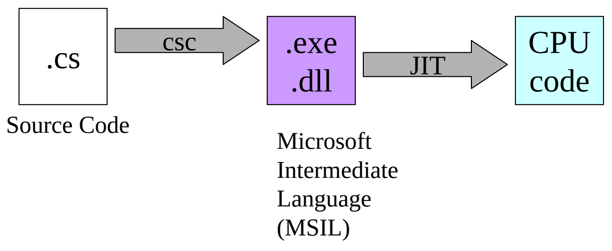
- Create server-side Web applications XML –-> thin clients
- Smart Clients
  - Use Microsoft C# .NET or Microsoft Visual Basic .NET
- C#
  - Derived from C++ and Java
  - Only runs on Windows machines!
  - Development environments
    - Visual Studio .NET, Borland XEmacs
  - Uses the libraries from the .NET Framework
    - Threading, Windows Forms, XML, ADO, etc.
    - For handhelds refer to .CF Framework libraries

C#

- C# is “component oriented” language in the C/C++ family
- Component concepts are :
  - Properties, methods, events
  - Design-time and run-time attributes
  - Integrated documentation using XML
  - No Header files
- Syntax similarities to Java and C++

C# vs. The World

- Compilation Process
- Common Language Runtime (CLR)
  - Provides an execution engine for developers code



C# Language

- Hello world Program

```
using System;

class HelloWorld {
  public static void Main() {
    Console.WriteLine(“Hello World!");
  }
}

>csc HelloWorld.cs
>Hello World!
```

C# Programming

- Parameter Passing
  - By value or References
- Boxing /Unboxing
  - Allows value types to be converted to and from objects automatically
- Pointers
  - Not recommended for use
- Versioning
  - C# requires developers to clearly state their intent
    - Use of the keyword ‘new’ and ‘override’

```
public static void Swap(ref int x, ref int y) {
  int z = x;
  x = y;
  y = z;
}
```

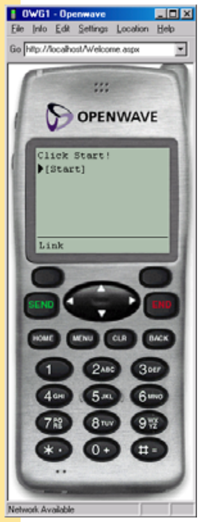
C# vs. The World

Comparison of C# syntax with Java and C++

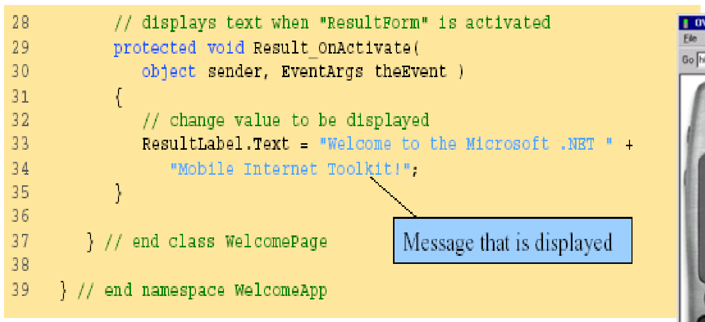
- Similarities
  - Single rooted class hierarchy
  - Similar keywords (derived from C++)
  - Virtual Machine & IL/CLR
  - Garbage Collection
  - No global methods
  - Interface, no Multiple inheritance
  - Exception handling
  - Easy to learn

Example

```
// Welcome.cs


namespace WelcomeApp
{
   using System;
   using System.Web;
   using System.Web.UI;
   using System.Web.UI.MobileControls;

   // inherit from System.Web.UI.MobileControls.MobilePage
   public class WelcomePage :
      System.Web.UI.MobileControls.MobilePage
   {
      protected System.Web.UI.MobileControls.Form Result;
      protected System.Web.UI.MobileControls.Label ResultLabel;

      // changes current form when "Start" is clicked
      protected void StartCommand_OnClick(
         object sender, EventArgs theEvent )
      {
         // change the current form to "ResultForm"
         ActiveForm = Result;

      } // end StartButton_OnClick


```

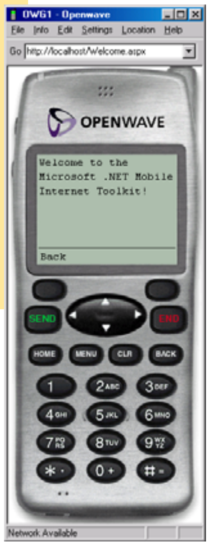


Example



```
      // displays text when "ResultForm" is activated
      protected void Result_OnActivate(
         object sender, EventArgs theEvent )
      {
         // change value to be displayed
         ResultLabel.Text = "Welcome to the Microsoft .NET " +
            "Mobile Internet Toolkit!";
      }

   } // end class WelcomePage

} // end namespace WelcomeApp
```

Message that is displayed

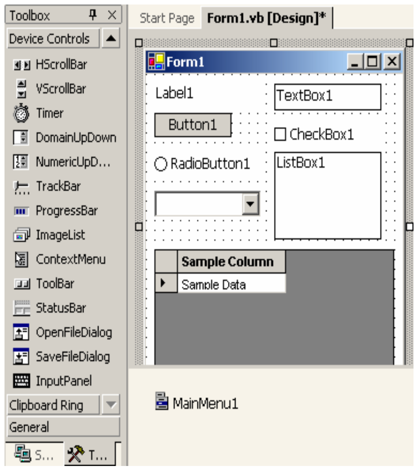

OWG1 - Openwave
File Info Edit Settings Location Help
Go http://localhost/Welcome.aspx
OPENWAVE
Back
SEND
END
HOME
MENU
CLR
BACK
Network Available

Welcome to the Microsoft .NET Mobile Internet Toolkit!

Telephony functions in .dll
e.g. smsinvoke

How to Create a Smart Device Application

- Create a New Smart Device Application Project
- Choose the platform and type of project
- Add additional forms, controls, and code

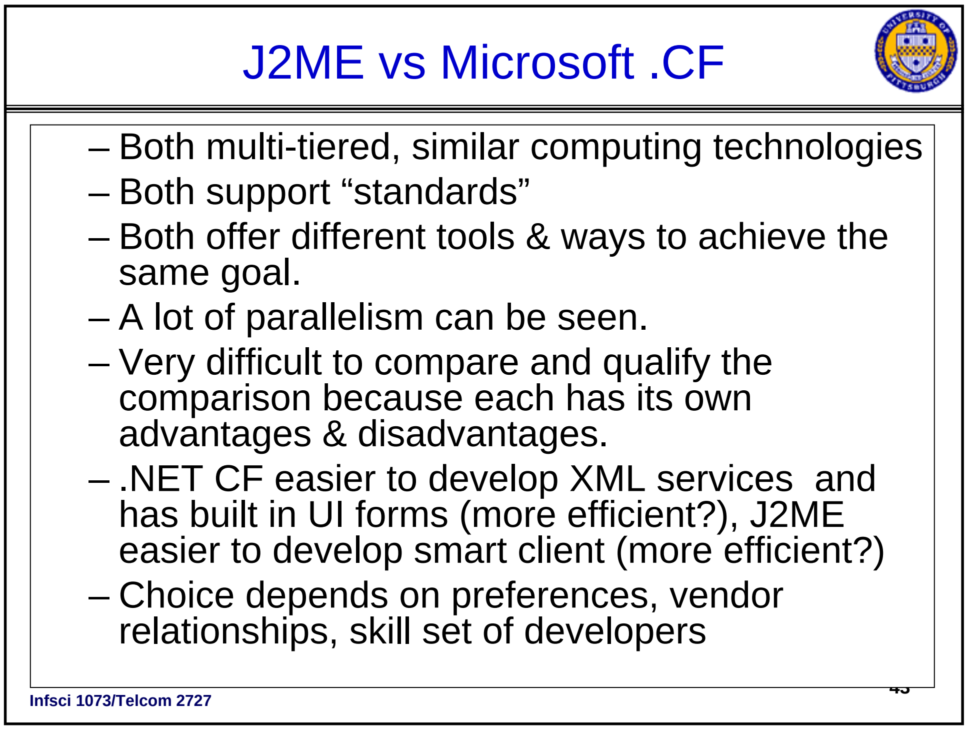


How to Test a Smart Device Application

- Visual Studio .NET 2005 includes device emulators that let you test your application
  - Pocket PC and SmartPhone
  - Windows CE .NET 4.1
  - etc
- You should also test with an actual device
- Debugging
  - Set breakpoints
  - Step through executing code in emulators or on device

How to Deploy a Smart Device Application

- You can use Microsoft ActiveSync from a desktop computer to manually deploy applications
- You can also use automated distribution mechanisms such as:
  - Downloading CAB files from a Web site
  - Microsoft Systems Management Server (SMS)
- For Thin Client Development
  - Develop XML pages use XSLT to convert to appropriate format (i.e., WML, cHTML, etc)
  - Mobile Internet Toolkit
    - Web site Design



J2ME vs Microsoft .CF

- Both multi-tiered, similar computing technologies
- Both support “standards”
- Both offer different tools & ways to achieve the same goal.
- A lot of parallelism can be seen.
- Very difficult to compare and qualify the comparison because each has its own advantages & disadvantages.
- .NET CF easier to develop XML services and has built in UI forms (more efficient?), J2ME easier to develop smart client (more efficient?)
- Choice depends on preferences, vendor relationships, skill set of developers

Infsci 1073/Telcom 2727 43