Introducing The ABCs of Windows Communication Foundation

Rick G. Garibay

azgroups

Introducing The ABCs of Windows Communication Foundation

- Agenda
  - A Brief History of Service Orientation
  - White-box vs Black-box Reuse
  - Services, SOA and Service Oriented Application Architecture
  - Tenets of Service Orientation
  - Introducing Windows Communication Foundation
  - Service Contract
  - Operation Contract
  - Hosting Options
  - Data Contracts

About Me

- Microsoft Certified Application Developer - .NET Framework
- Microsoft Certified Professional
- 8+ years professional experience developing Microsoft solutions for the retail and financial services industry
- Architect, Program Manager for ESS, a Microsoft Gold Partner ISV in Phoenix, AZ

Get this deck and code samples at http://rickgaribay.net

A Brief History of Service Orientation

- The historical relationship between software engineering and the reduced coupling between hardware, OS, objects, components and services.

Coupling

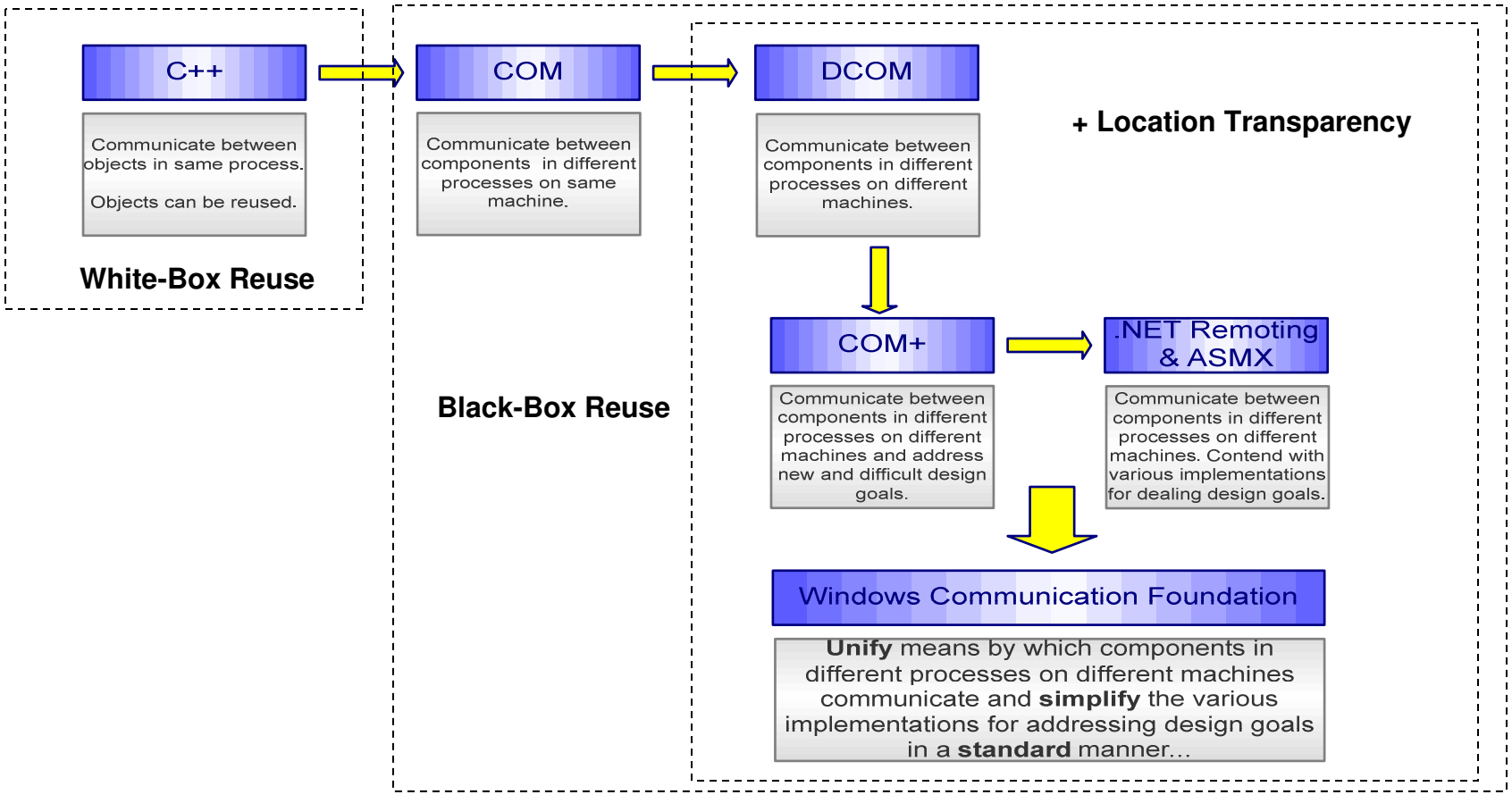
1940s & 1950s	1960s	1970s	1980s	1990s	Early 2000s
Electro-Mechanical Computers "Code" bound directly to hardware. Harvard Mark I Colossus ENIAC	General Purpose Computers Assembly language ties code directly to hardware. IBM 360 OS 360	The birth of microcomputers Procedural languages offer a nice layer of abstraction. Intel 4004, 8008, 80880, Altair 8800 Unix C, Pascal, Smalltalk	High level Compilation & Object Orientation Intel 80286, Macintosh,TRS-80, Commodore, Amiga MS-DOS, SunOS, Windows 1.0, Windows 2.0, OS400, BeOS C++, SQL, Perl	Object Orientation grows up (COM). Internet challenges current application architectures. Intel 486 Windows 3.1, Windows NT 3.1, Red Hat Linux, Windows 95,98 Visual Basic, Delphi, Java, JavaScript, Python, PHP	Globalization demands connectivity between computers, networks, systems and infrastructures. Windows 2000, XP, 2003, Mac OS X DCOM, COM+, Microsoft .NET Framework 1.0, 1.1, 2.0

Coupling by Generation

Time

A Brief History of Service Orientation

- A closer look at recent developments



White-box vs Black-box Reuse

- The primary motivation for moving from procedural programming to objects was all about reuse, and thus, improved productivity.
- As it turns out, most reuse was at the expense of productivity due to the white-box nature of classic objects.
- White Box Reuse
  - Developers must be intimate with object/class internals, often within all levels.
  - Client code is tightly bound to implementation.
  - Changes in a generalized class causes hidden (and sometimes severe) impact to client code.
- Black Box Reuse
  - Developers must simply understand the contract, or interface.
  - Interfaces provide separation between definition and implementation.
  - Client code knows absolutely nothing about the implementation.

Services, SOA and Service Oriented Applications

"Modern software engineering is ***the ongoing refinement of ever increasing degrees of decoupling***" – *Juval Lowy*

- In general, we all agree that:
  - Coupling is a bad thing.
  - But, coupling, to some extent is unavoidable.

- **Services** (Service Oriented Architecture, and Service Oriented Applications) are the next evolution from **components**, just as interfaced-based components where an evolution from **objects**.

- SOA is, to the best of our knowledge, the best way to build distributed applications that are maintainable, where the ***right*** *aspects are decoupled yielding:*
  - *Productivity*
  - *Maintainability*
  - *Extensibility*
  - ***Real*** *Reuse*

Services, SOA and Service Oriented Applications

Characteristics *within* services:

- Languages (i.e. C#)
- Technologies (i.e. .NET FX, J2EE)
- Platforms (Windows, etc)
- Versions (Major.Minor.Build.Revision)

Characteristics *between* services:

- Standards/Protocols (HTTP, MSMQ, IPC, etc)
- Policies (Transactions, Reliability, etc)
- Contracts (WSDL)
- Messages (SOAP)

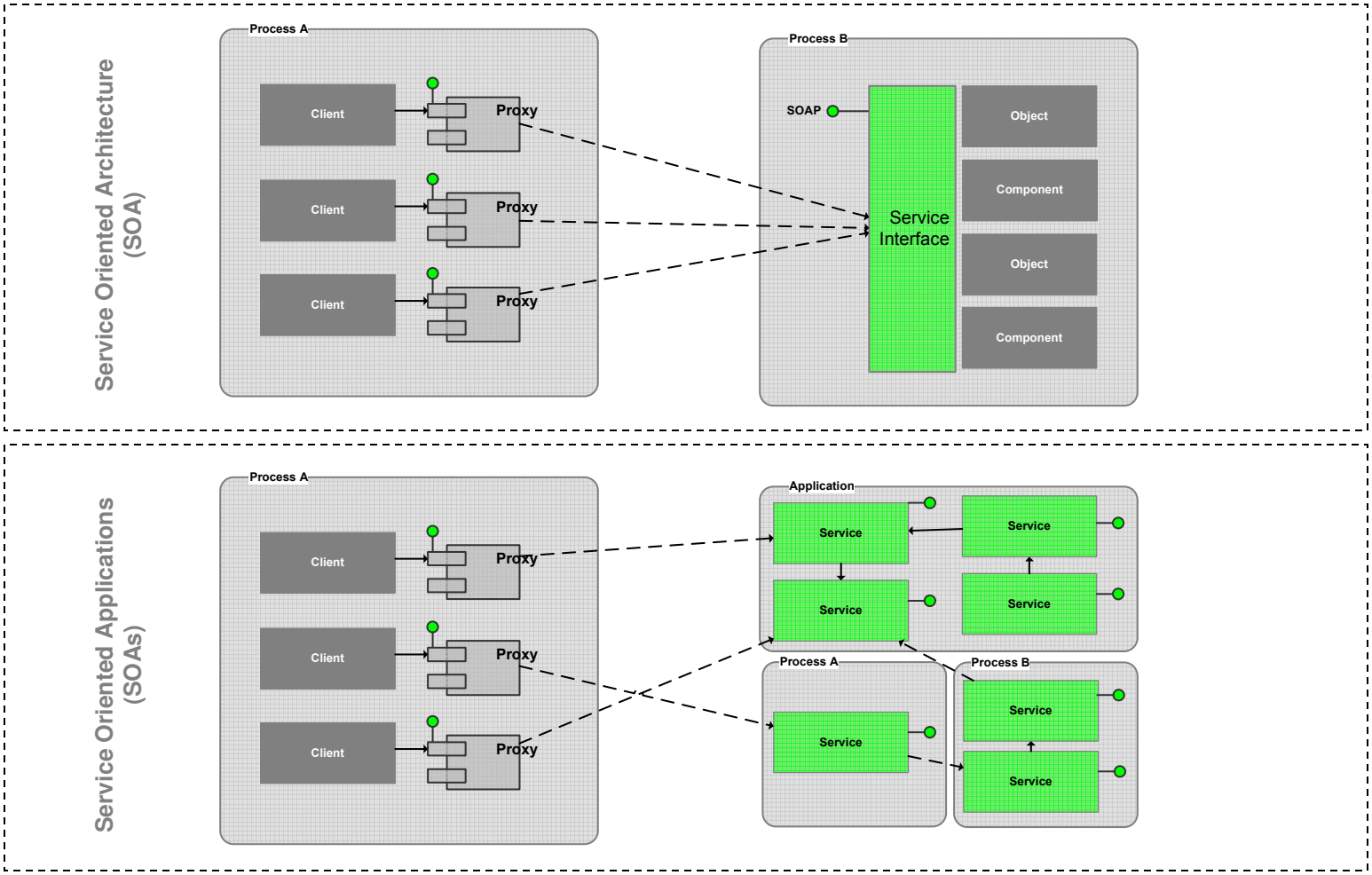
…But Web Services are Nothing New!

True, but…

- Can you really afford to forgo reliability in your mission critical services?
- Isn't there more to life than mere HTTP?
- Have you ever generated a WSDL on the .NET stack and had a non-.NET client consume it successfully the first go around, ever?
- Does anyone really understand the WS-* specifications down to the wire format? Do you really want/need to?
- Who thinks WSE is fun?

- WCF supports all relevant specifications via bindings:
  - WsHttpBinding and WsDualProfileBinding
    - WS-Secure Conversation
    - WS-Reliable Messaging
    - WS-Atomic Transactions
    - WS-Coordination
  - More on bindings:
    - http://windowssdk.msdn.microsoft.com/en-us/library/ms730294.aspx

Service Oriented Architecture & SO Applications



(Generally Agreed Upon) Tenets of Service Orientation

Don Box's Canonical Tenets of Service Oriented Architecture (SOA):

- Boundaries are Explicit.
- Services are Autonomous.
- Services share Schema and Contract, not Class.
- Compatibility is Based on Policy.

Additional SO Principles

Additional SO principles (you should adhere to simply because it is the right thing to do) :

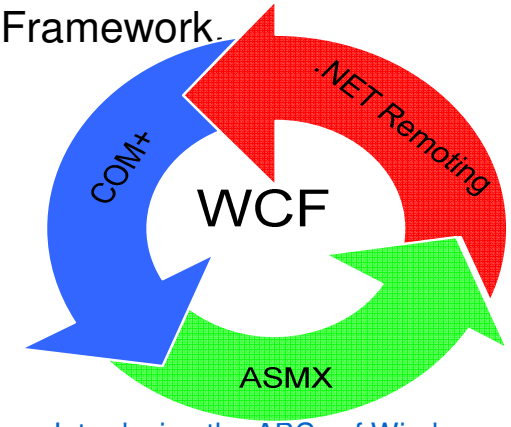
- Services are secure.
- Services leave the system in a consistent state.
- Services are thread-safe.
- Services are reliable & robust.
  - Interoperable
  - Scaleable
  - Available
  - Responsive
  - Disciplined

From the IDesign WCF Standard Guidelines & Best Practices

Introducing Windows Communication Foundation

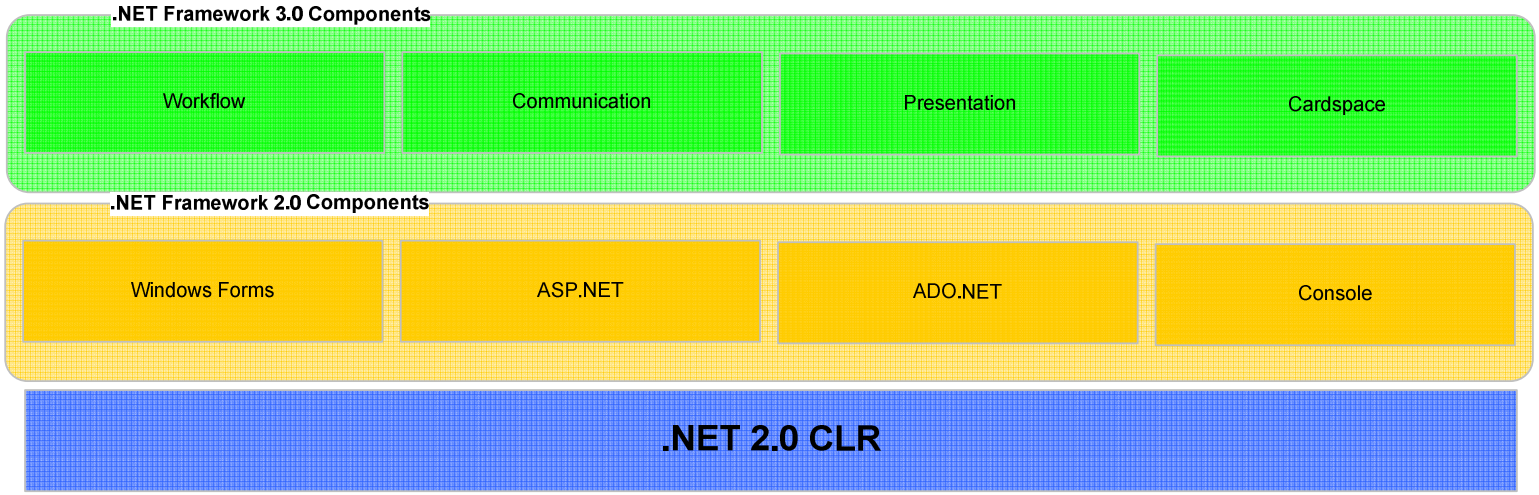
What is Windows Communication Foundation (WCF)?

- WCF is Microsoft's implementation of industry standards to provide a communication subsystem enabling applications on one machine (process boundary) or across multiple machines to communicate.
- WCF is a core component of the .NET Framework 3.0 (formerly WinFX) which is included with Windows Vista and will support Windows 2003 and Windows XP platforms as well as the future version of Windows Server.
- The WCF API unifies ASMX Web Services, .NET Remoting, distributed transactions and messaging into a single programming model that makes true service orientation tenable (while guaranteeing interoperability).
- Fundamental to .NET Framework.



A Word about .NET 3.0 (formerly WinFx)

- .NET 3.0 is additive to .NET 2.0.
- It uses (and requires) the .NET 2.0 CLR exclusively
- Really bad branding decision



Introducing Windows Communication Foundation

- The ABCs of Windows Communication Foundation:

An **Address** uniquely identifies a service. Provides the **transport** protocol, name of target **machine** (host) and **port** if applicable. Expressed as an explicit **path** or URI:

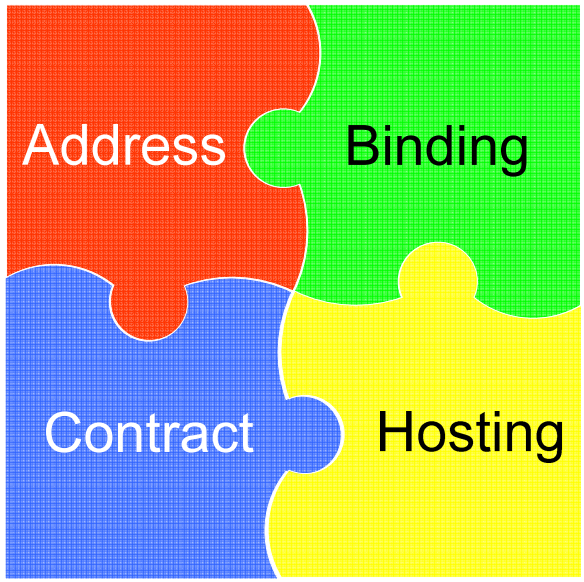
[**transport**]://[**machine**][:optional **port**]

```
http://localhost
http://localhost:8081
http://localhost:8081/Service
net.tcp://localhost:8082/Service
net.pipe://localhost/Pipe
```

All services expose a **Contract.**

WCF uses 5 types of contracts:

Service Contract – Exposes the service.
Operation Contract- Exposes the service members.
Data Contract – Describes service parameters.
Fault Contracts – Defines error handling semantics.



Bindings provide "canned" policies that implement the WCF features required to support the design goals of the service.

Some common bindings include:

```
BasicHttpBinding
NetTcpBinding
WSHttpBinding
NetMsqmqBinding
```

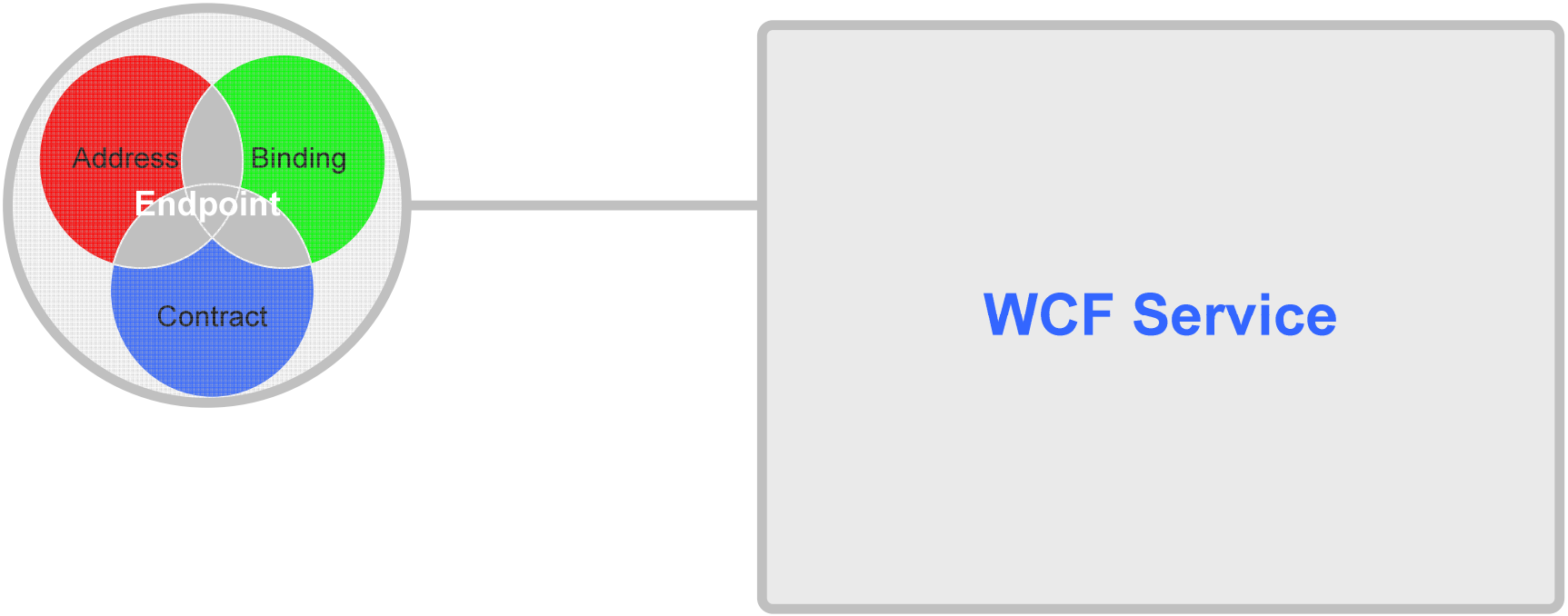
WCF services must be hosted by a Windows Process (host process).

Hosting options include:

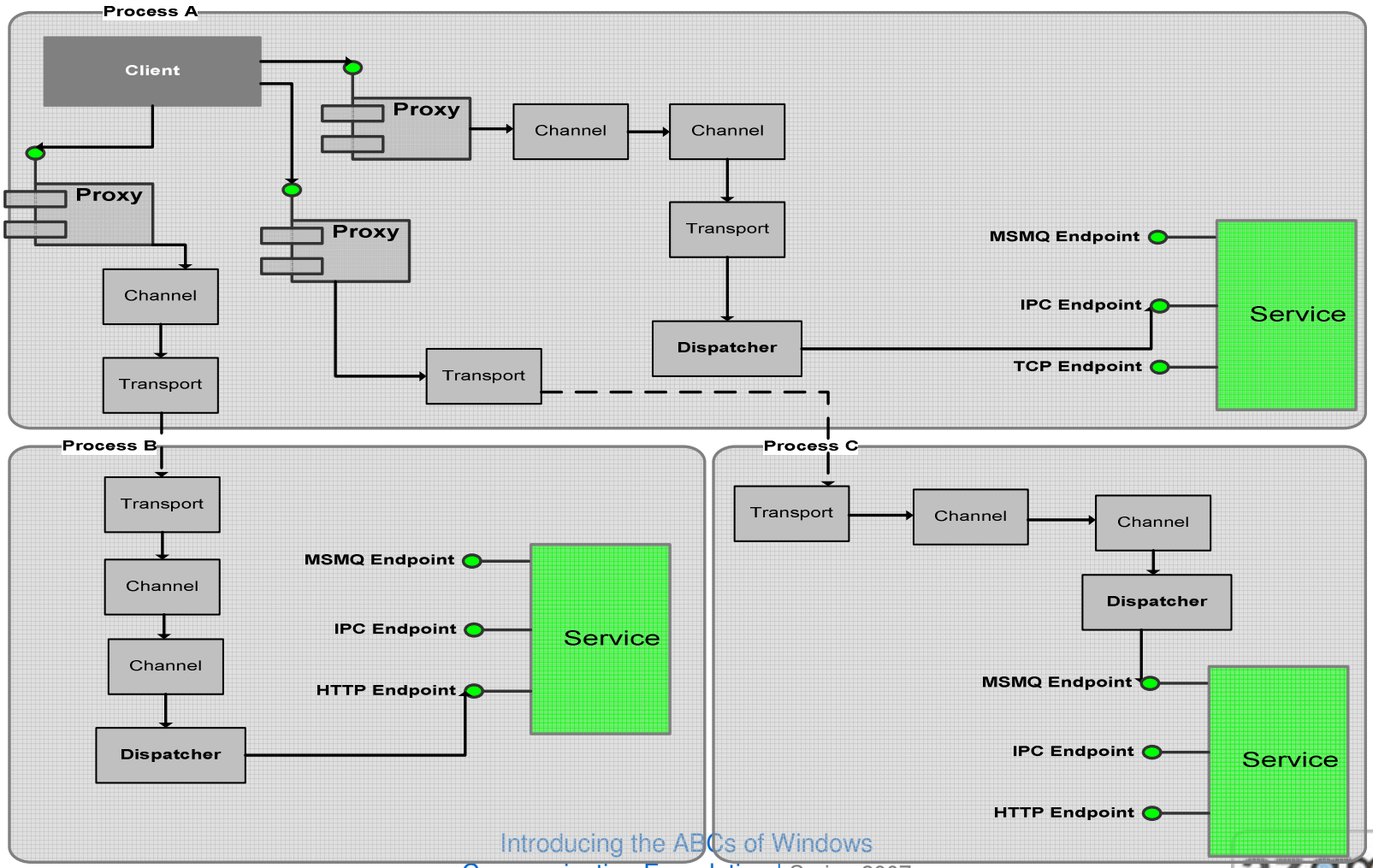
IIS 5 & 6
IIS 7 & Windows Activation Service (WAS)
SmartClient, Console or Windows Service (also called "Self-Hosting")

Introducing Windows Communication Foundation

- WCF Services are deployed, discovered and consumed as endpoints.



Architecture



Service Contracts

- .NET 3.0 WCF uses common .NET interface along with the ServiceContractAttribute class to explicitly define a contract.
- Service Contracts are implicitly public (access modifiers are a .NET notion).

```
using System;
using System.ServiceModel;

[ServiceContract()]
interface IPersonnelAction
{
  // Members

}
```

Operation Contracts

- An Operation Contract uses the OperationContractAttribute class to opt-in a method to participate in a Service Contract.

```
using System;
using System.ServiceModel;

[ServiceContract()]
interface IPersonnelAction
{
      [OperationContract]
      double AdjustSalary(int employeeId, double baseSalary, double adjustmentPercent);

      // Not a part of the contract
      bool Terminate(int employeeId);
}
```

Implementing a Service Contract

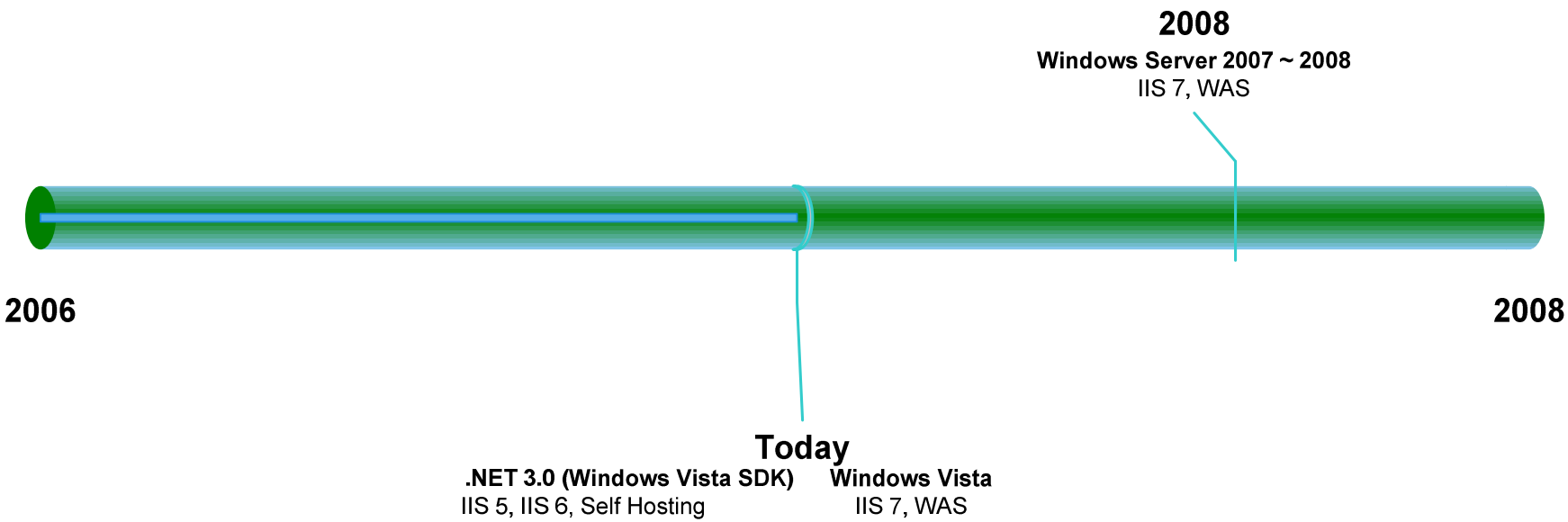
- Disciplined developers will maintain the separation of contract from implementation (nothing changes here).

```
using System;
using System.ServiceModel;

class HRManagerService : IPersonnelAction
{…}

// Avoid!!!
[ServiceContract()]
class HRManagerService
{
      [OperationContract]
      double AdjustSalary(int employeeId, double baseSalary, double adjustmentPercent)
      {…}

      // Not a part of the contract
      bool Terminate(int employeeId)
      {…}

}
```

Demo - Service Contract, Operation Contracts & Configuration

http://www.netfx3.com/

Hosting Options



Demo - IIS Hosting & Deployment

azgroups

Data Contracts

- A Data Contract marks a business entity (class) as a participant in a Service Contract and Service Operation. Unlike using the SerialziableAttribute, all entities and members are strictly opt-in.

```
using System;
using System.Runtime.Serialization;

[DataContract()]
public class Employee : Person
{
            private string m_Name = string.Empty;
            private int m_EmployeeId = string.Empty;

           [DataMember]
            public int EmployeeId
           {
            get
               {
                 return m_EmployeeId;
               }
            set
               {
                 m_EmployeeId = value;
               }
           }
 }
```

Demo - Data Contracts & Operation Overloading

Fault Contracts

- A Fault Contract is the definition of how errors are raised and handled by WCF clients.
- There is no concept of an exception in WCF (this is a .NET only idea).
- Fault Contracts describe error conditions and behavior for managing these conditions.

Only the Tip of the Iceberg…

- Advanced topics
  - Bindings
  - Fault Handling
  - Instance Management
  - Security
  - Transactions
  - Concurrency Management
  - Queued Services

Questions?

- References

  - IDesign WCF Master Class, Microsoft Silicon Valley Campus, San Jose, CA. June 2006.
  - IDesign WCF Standard
  - Programming WCF Services, Juval Lowy, 2007 O'Reilly Press.
  - Timeline of Computing, Wikipedia: http://en.wikipedia.org/wiki/Timeline_of_computing. July 2006.
  - S. Someasegar, Someasegar's Blog: http://blogs.msdn.com/somasegar/archive/2006/06/09/624300.aspx. July 2006