.NET 軟體開發與實務演練

李學麟 (Polo Lee)
開發技術推廣經理
開發工具暨平台推廣處 / 台灣微軟

patterns & practices
Proven practices for predictable results

Agenda

- Patterns & Practices
- 軟體架構 與 Software Factory
- ASP.NET Web Part 實作

客戶的聲音, “我們需要”

客戶經驗	[illegible]全的架構 [illegible]的回饋
解決方案	微軟平台整合優化的解決方案
Patterns & practices	架構與設計準則 工具與開放源碼
文件	產品功能, 作業程序 疑難處理, 參考文件
平台	Windows Server System Visual Studio 開發平台與 管理工具

patterns & practices

Proven practices for predictable results

P&P Vision & Mission

Vision

- 讓企業的架構師與開發人員, 在微軟的平台上, 用最正確與最有效率的方法與工具, 得到最高品質與最可靠的軟體應用程式

  Enterprise architects and developers build trustworthy, quality enterprise applications agilely and effectively using the Microsoft platform. patterns & practices customer connection and experience improves future product releases.

Mission

- Drive enterprise customer, partner, and MS success today by delivering architectural guidance and code aligned with future product strategy outside product releases.

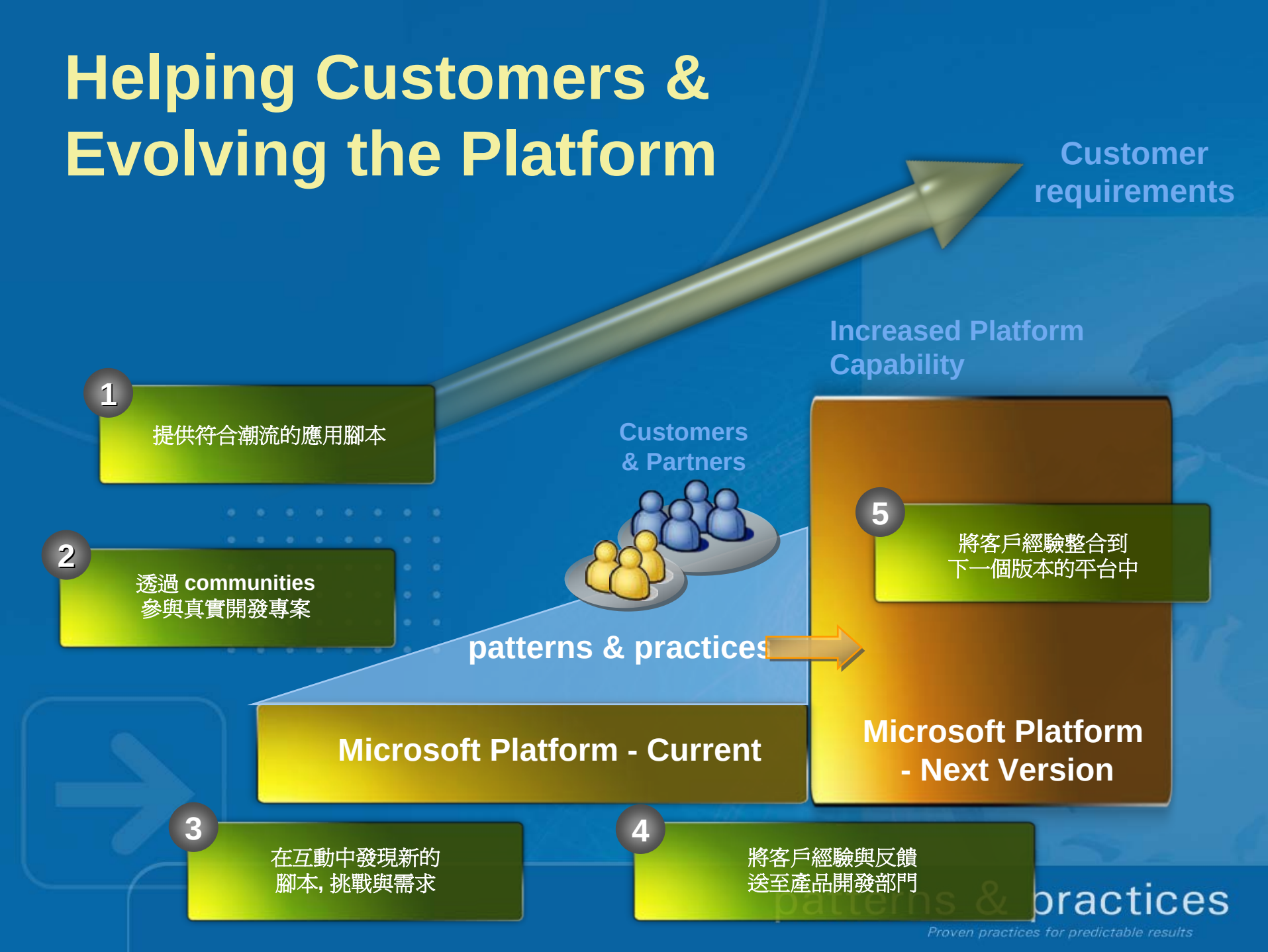
Guiding Principles

- Long-term customer success – now thru release after next
- Customer needs balanced with product strategy
- Quality over scope - no guidance is better than bad guidance
- Collaborative, transparent execution

patterns & practices

Proven practices for predictable results

Helping Customers & Evolving the Platform



導入狀態


Downloads
百萬
2.0
1.8
1.6
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0
62,700
85,140
350,000
573,000
989,000
1,370,000
1,900,000 (projected)
FY00
FY01
FY02
FY03
FY04
FY05
FY06

四種類的 Guidance

Guides

提供 概念性的架構 & 設計準則，並盡可能的搭配 How To 的實作議題

Example: Guidance Explorer

Application Blocks

共用平台架構下, 可重複使用的程式碼元件

Example: Enterprise Library

Reference Implementations

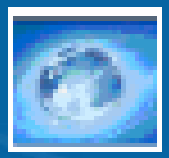
提供驗證過的應用程式範例當作實作基本準則，完整的程式碼寫作方式

Example: Integration Baseline Architecture

Software Factories

讓架構師能夠自我定義軟體架構模板與包裝既有資產的最佳演練

Example: Web Service Software Factory , GAX / GAT



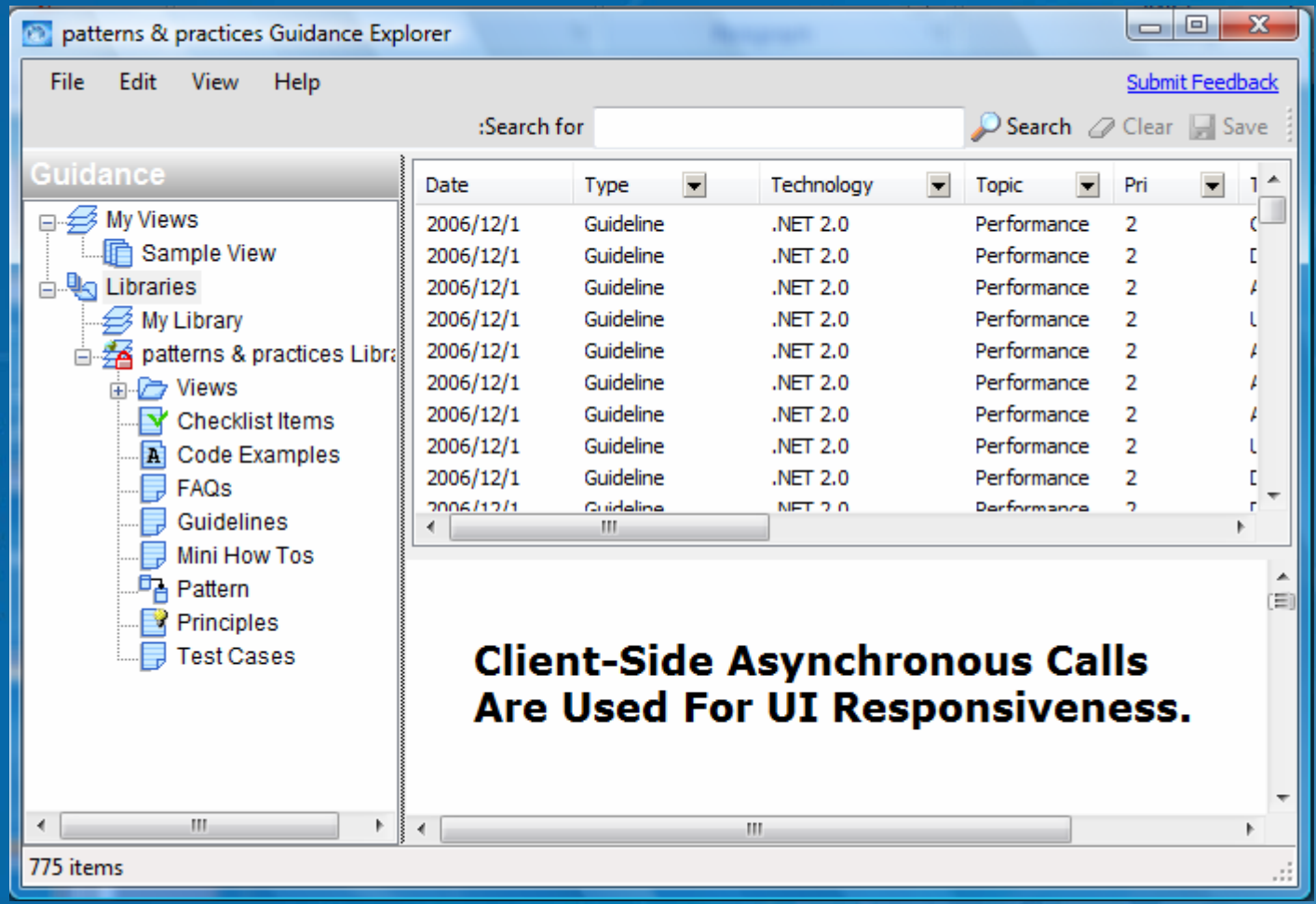
Guidance Explorer

patterns & practices

Proven practices for predictable results

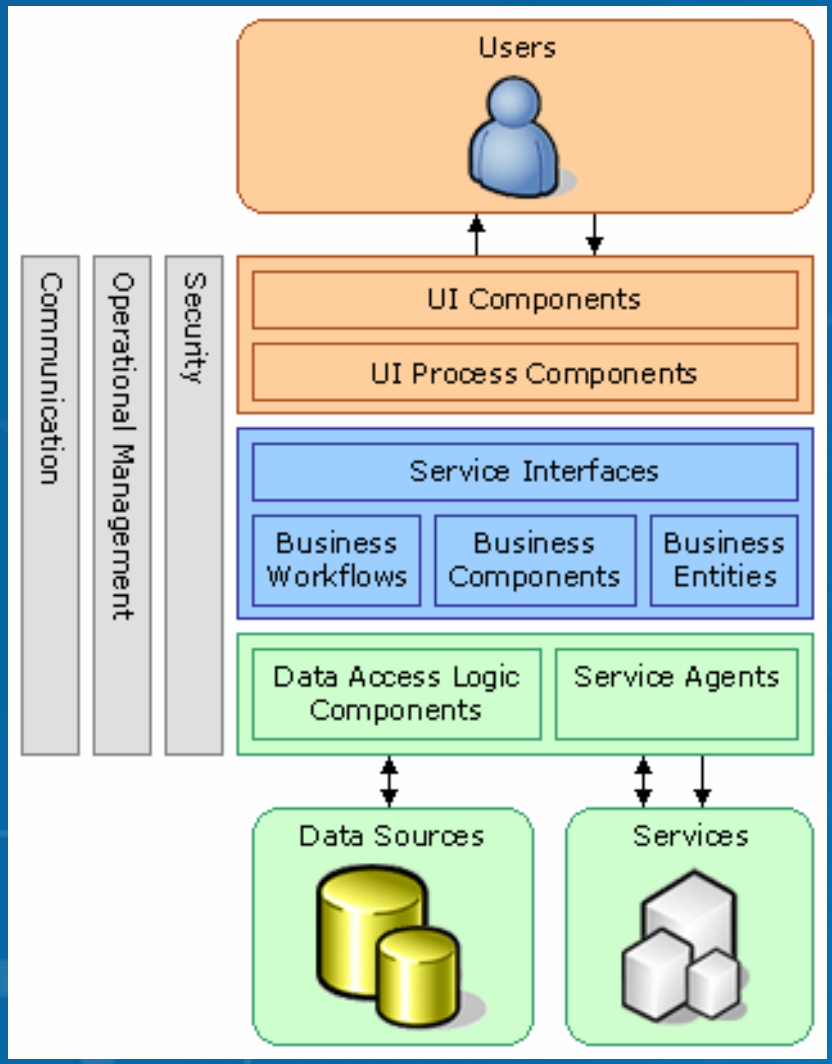
Guidance Explorer

- 軟體協同開發知識庫 (微軟團隊 + Open Group + 您)



Software Factories

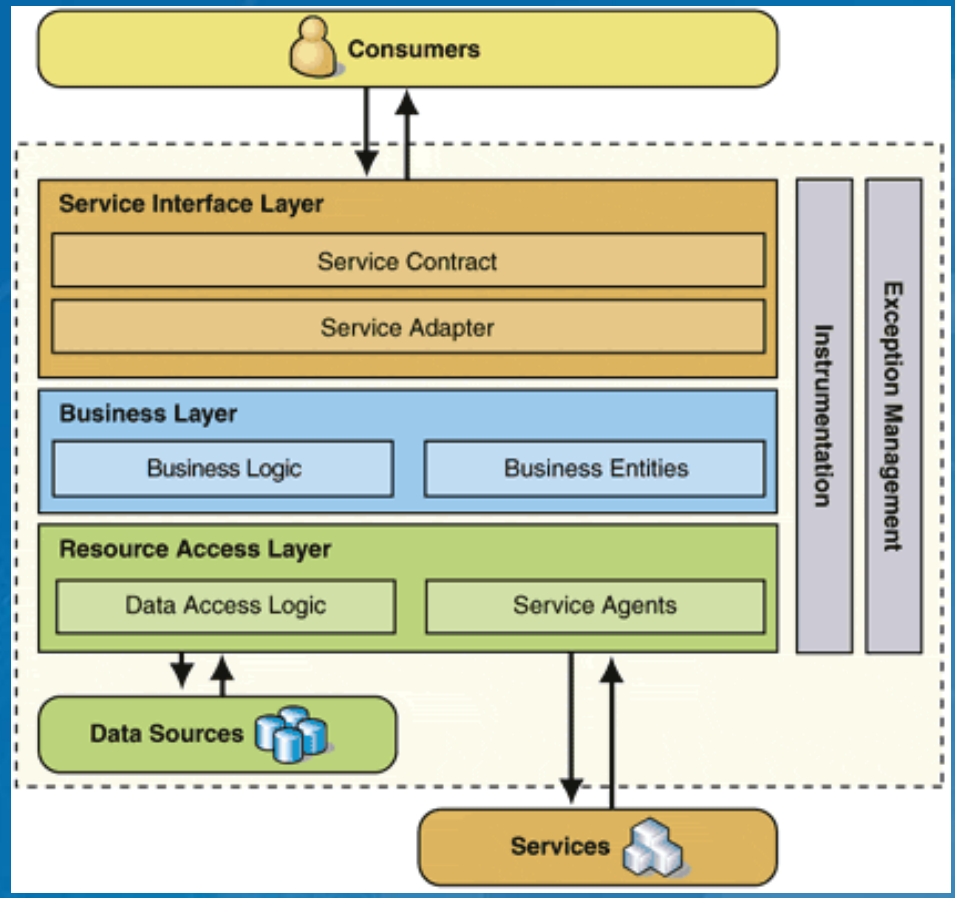
Application Architecture for .NET



- .NET 平台軟體優化架構
  - UI: 使用者介面
  - SI: 服務介面
  - DAL / Agents : 資料處理

Web Services Software Factory

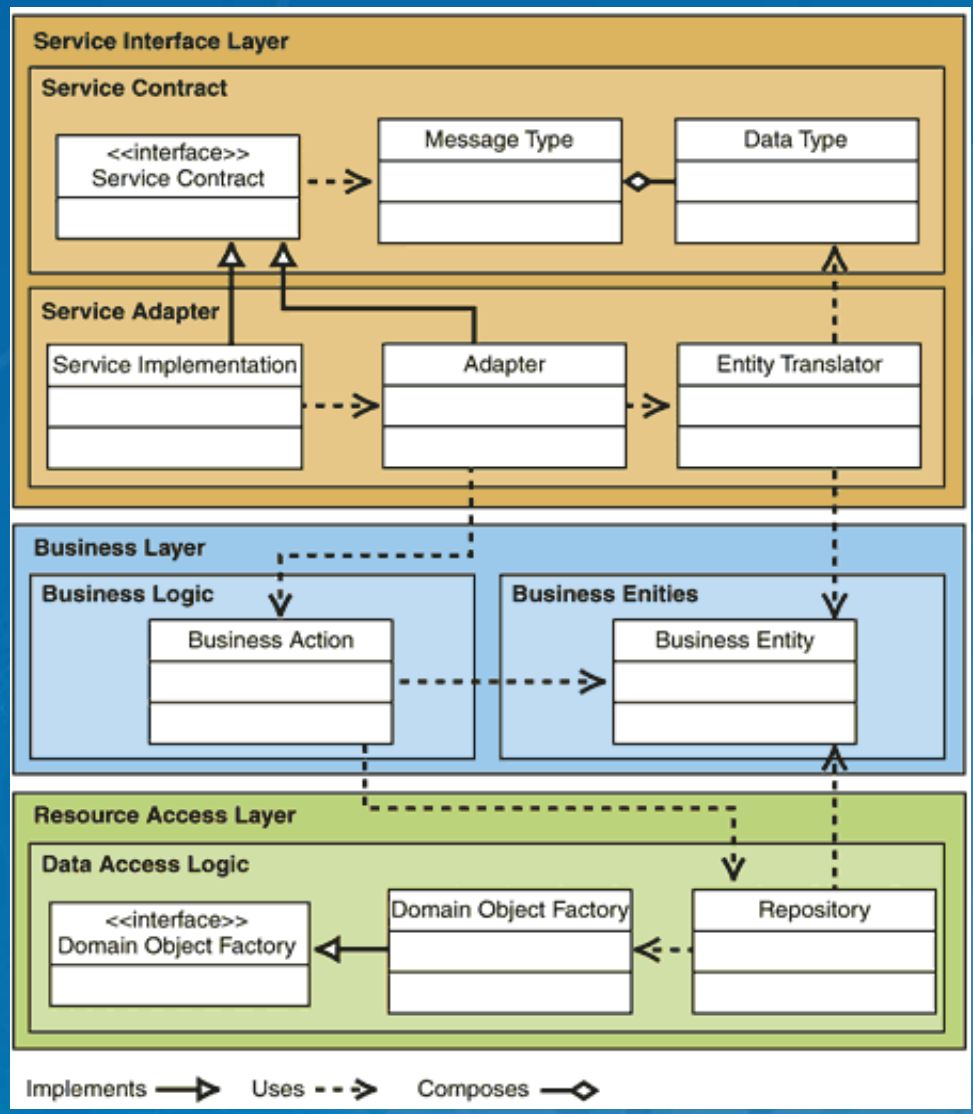
- Message and Service Design
- Entity Translation
- Data Access Layer creation
- Web Service Security
- ASMX and WCF guidance



Service Factory Demonstration

Demo

- Step 1.
  - 定義 Data Type
- Step 2.
  - 定義 Message Type
- Step 3.
  - 定義 Service 合約介面
  - 決定 Service 合約實作物件
- Step 4.
  - 定義商業物件(Business Entity)
- Step 5.
  - 定義商業物件資料邏輯存取物件
- Step 6.
  - 對外公開介面



Software factory 導入效益

- 針對每個 Developer 做好 guidance
- 幫助你架構出整體的解決方案
- 採用被驗證過的軟體經驗 (提高品質 quality)
- 所有設計可被收斂與一致性 (提高可預測性 predictability)
- 用更少的時間 (提高產能productivity)

ASP.NET Web Part

活用與整合

Web Parts

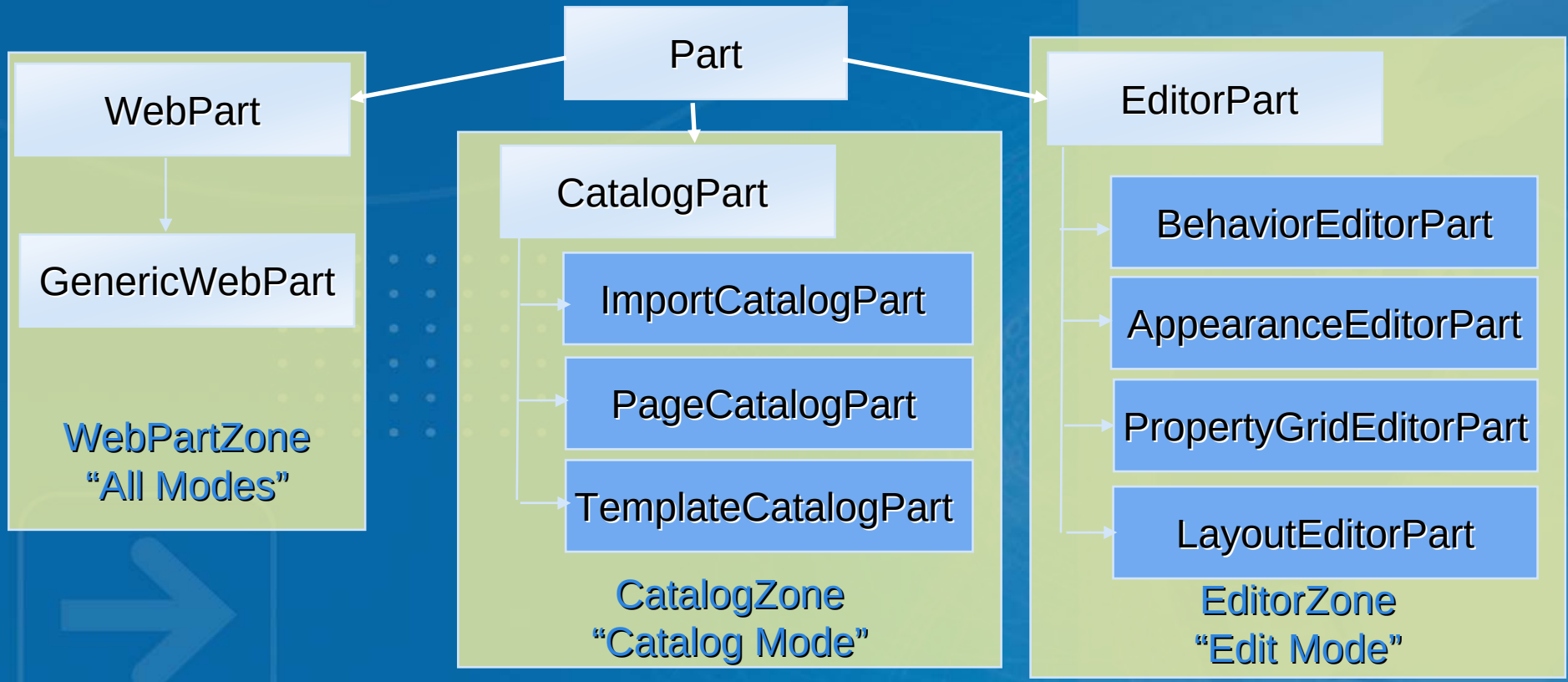
- 開發入口網站基礎元件技術框架
  - 可以應用在自行開發的 ASP.NET 2.0 網站
  - 可以整合在 SharePoint V3 / Office Server 2007 中
  - 元件架構在 System.Web.UI.WebControls.WebParts
- 目標在 “用少量的程式碼, 創造出更豐富的使用者介面”
  - 支援網頁即時 drag-and-drop
  - 支援網頁即時 屬性與外觀異動
- 支援個人化
- 支援 Web Parts 元件互動 ("connections")

Web Parts

Demo

Web Part Concepts
Parts

- 不同種類的 Web Parts



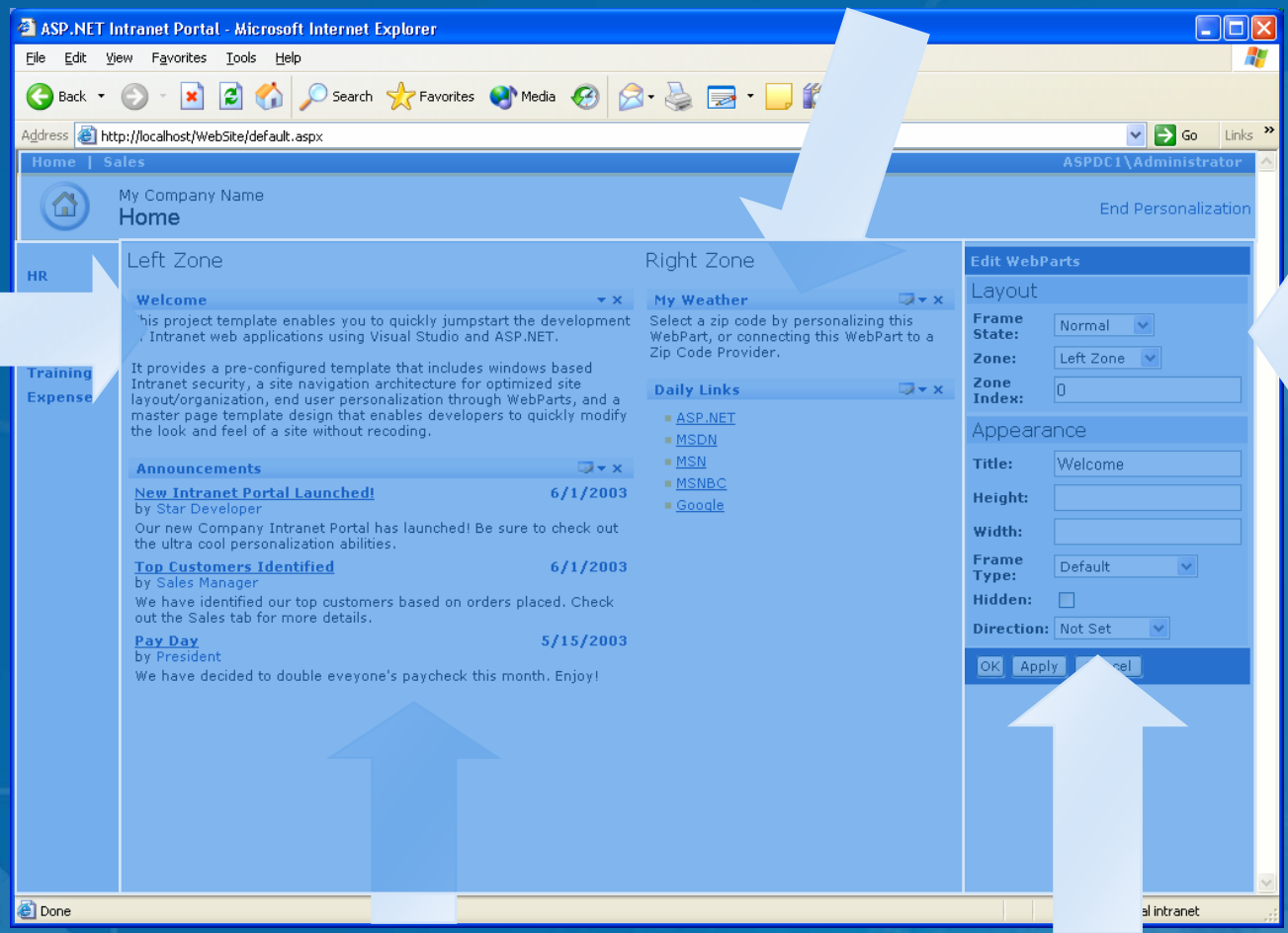
Web Part Concepts
WebPartManager

```
<asp:WebPartManager runat="server" />
```

- 並非視覺化的元件
- 是 Web Part 框架的核心
  - 負責管理所有畫面上的 parts, zones, connections
  - 與 Personalization Provider 整合提供 個人化的 Web Part 架構
- 每一個存放 Web Part 的網頁都必須有一個 獨立 的元件實體存在

Parts

Web Part Control



Literal

LayoutEditorPart

User Control

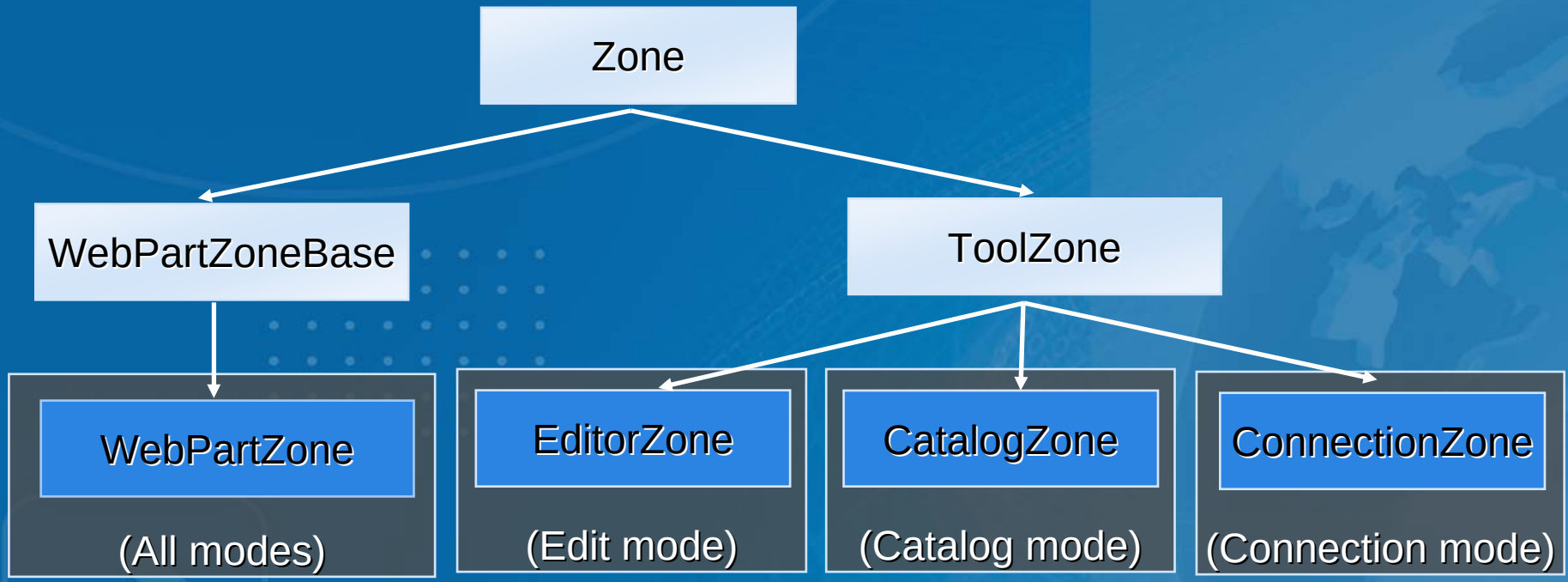
AppearanceEditorPart

Creating Web Parts

Demo

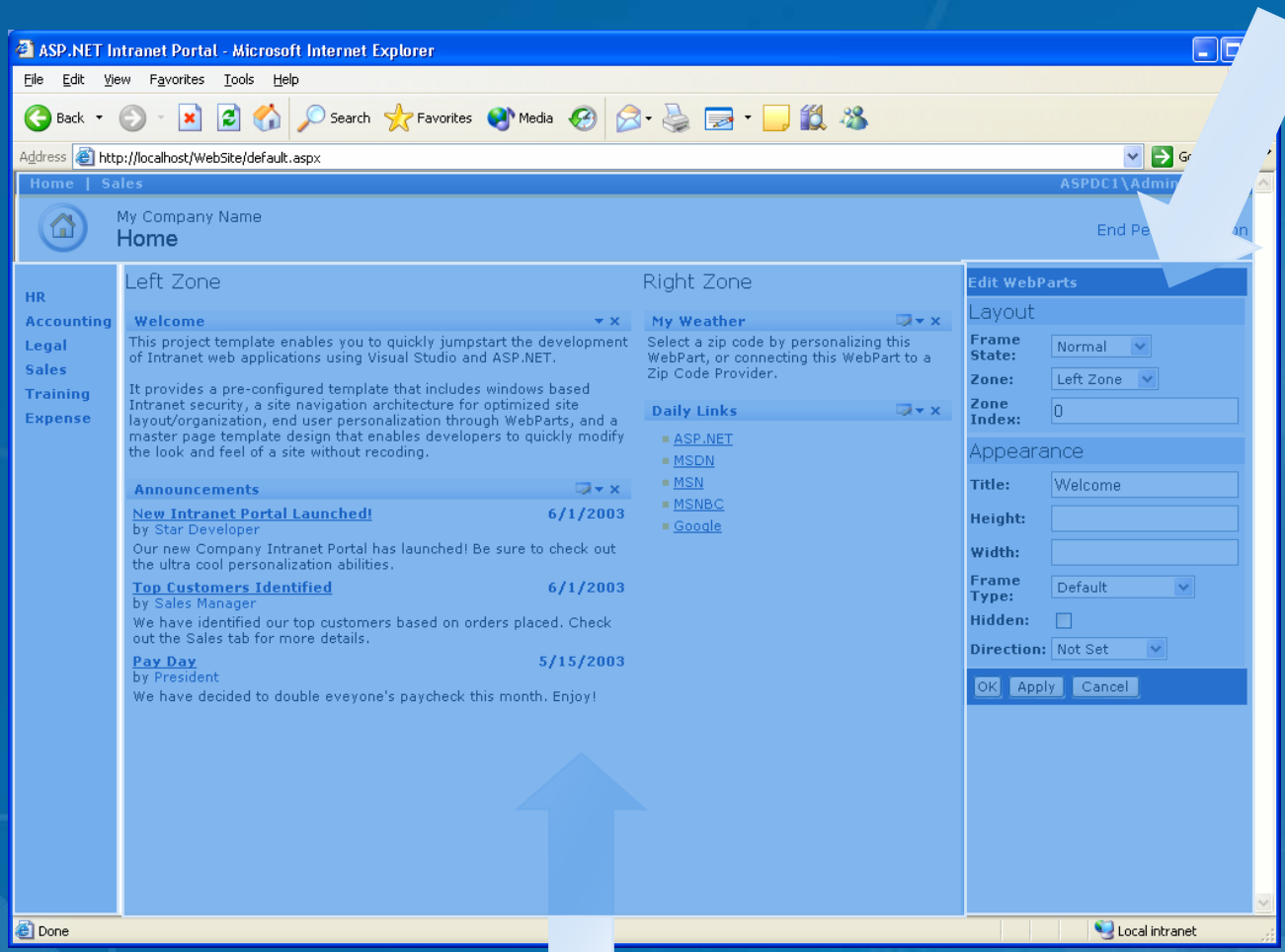
Web Part Concepts
Zones

- Contain Parts and show up conditionally



Zones

EditorZone



WebPartZone

Creating Web Part Editors

Demo

The Web Part Connection Model

- The Connection
  - Managed by the WebPartManager
- The Connection Points
  - One from the Provider
  - One from the Consumer
- Communication can be established with more than one other part

The Connection

```
<StaticConnections>
    <asp:WebPartConnection
        ID="ZipCodeConnection"
        ProviderID="Weather1"
        ProviderConnectionPointID="ZipCodeProvider"
        ConsumerID="News1"
        ConsumerConnectionPointID="ZipCodeConsumer">
    </asp:WebPartConnection
</StaticConnections>
```

The Connection Point

- Must be defined in the provider control
- The provider control must implement the Contract Interface

```
public partial class WeatherNew : System.Web.UI.UserControl, IZipCode
{
        :
        :
}
```

The Provider Contract Interface

- A function that returns an instance of the provider class
- With the attribute *ConnectionProvider*
- For provider that implements only one provider interface there is no need to specify the contract interface

```
// ZipCodeProvider is the ProviderConnectionPointID
[ConnectionProvider("Zip Code", "ZipCodeProvider")]
public IZipCode GetZipCodeInterface()
{
      return (IZipCode)this;
}
```

The Provider Contract Interface

- For provider that implements more than one provider interface specification of the contract interface is needed

```
[ConnectionProvider("Zip Code", "ZipCodeProvider“ typeof(IZipCode))]
public IZipCode GetZipCodeInterface()
{
      return (IZipCode)this;
}
```

The Consumer Contract Interface

- Create a function that receives the provider instance
- Has the attribute *ConnectionConsumer*

```
// ZipCodeConsumer is the ConsumerConnectionPointID
[ConnectionConsumer("Zip Code", "ZipCodeConsumer")]
public void GetIZipCodeInterface(IZipCode provider)
{
      string zip = provider.GetZipCode();
            :
            :
}
```

Creating Web Part Connection

參考資料

- Patterns & Practices
  - http://msdn.microsoft.com/practices/
- http://blogs.msdn.com/polo_lee/
- MSDN 開發人員論壇

  http://www.microsoft.com/taiwan/forum
- 微軟程式開發技術訓練短期課程
  - http://www.microsoft.com/taiwan/vstudio/events/training/
- MSDN 開發技術研討會
  - http://www.microsoft.com/taiwan/msdn/events/

FAQ