Web Services Introduction

林光龍

Outline

- Web Services 範例
- Web Services 簡介
- 分散式元件技術的發展
- Web Services 主要技術簡介
- Web Services 的發展障礙

Web Services 就是 …

一種讓分散於不同平台上，由不同技術所實作出來的相關服務，得以互相運用的軟體程式整合技術，其目地在於結合各種相關服務，以提供使用者一更具效率性與便利性的全新服務

Payment Service

台灣電力公司

台灣省自來水公司

中華電信

台湾大哥大
TAIWAN CELLULAR CORP.

TAIPEI GAS

Universal Payment Service

台灣電力公司

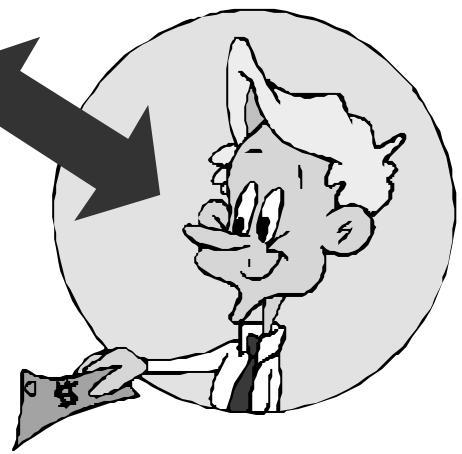
台灣省自來水公司

中華電信

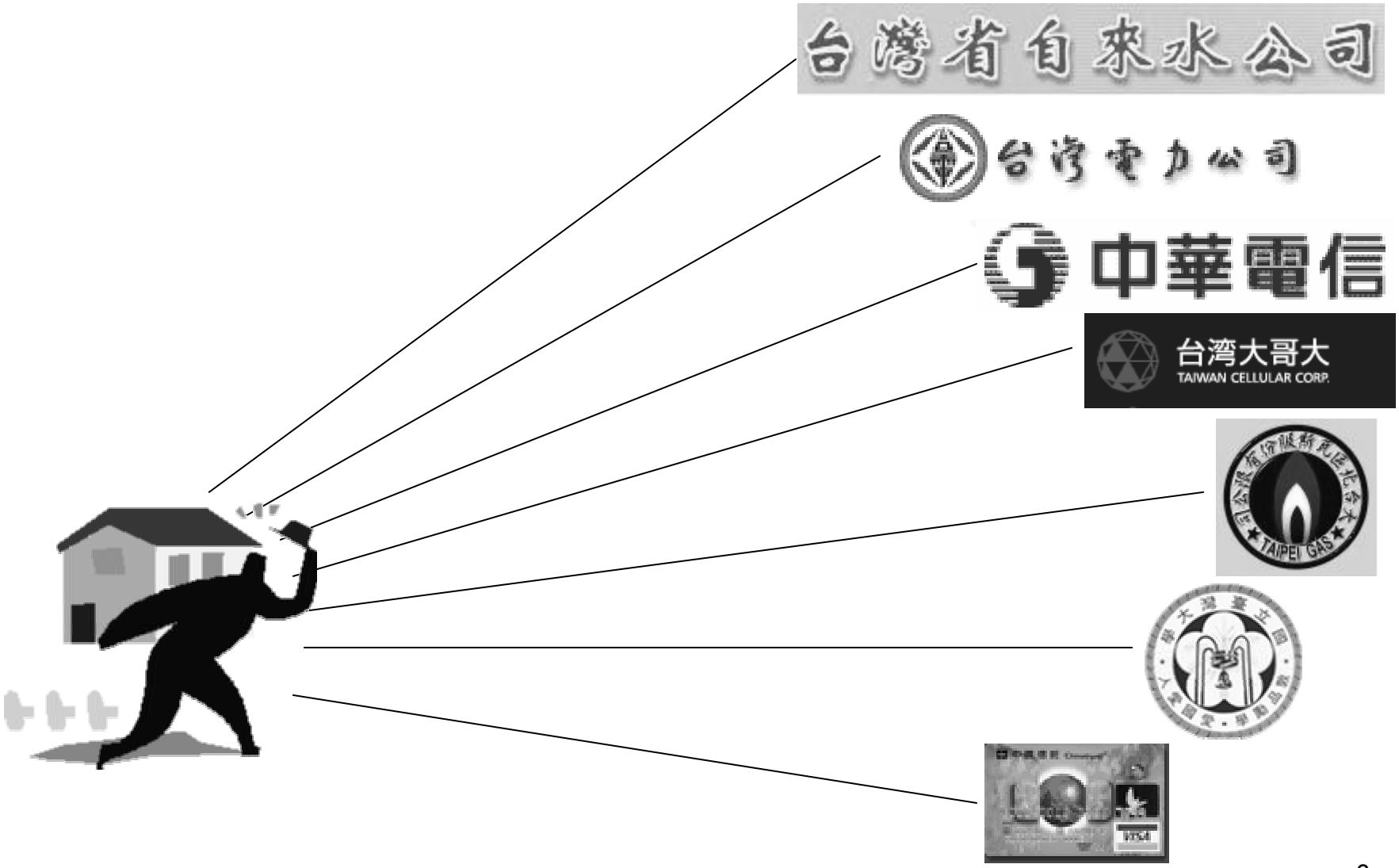
台湾大哥大
TAIWAN CELLULAR CORP.

TAIPEI GAS

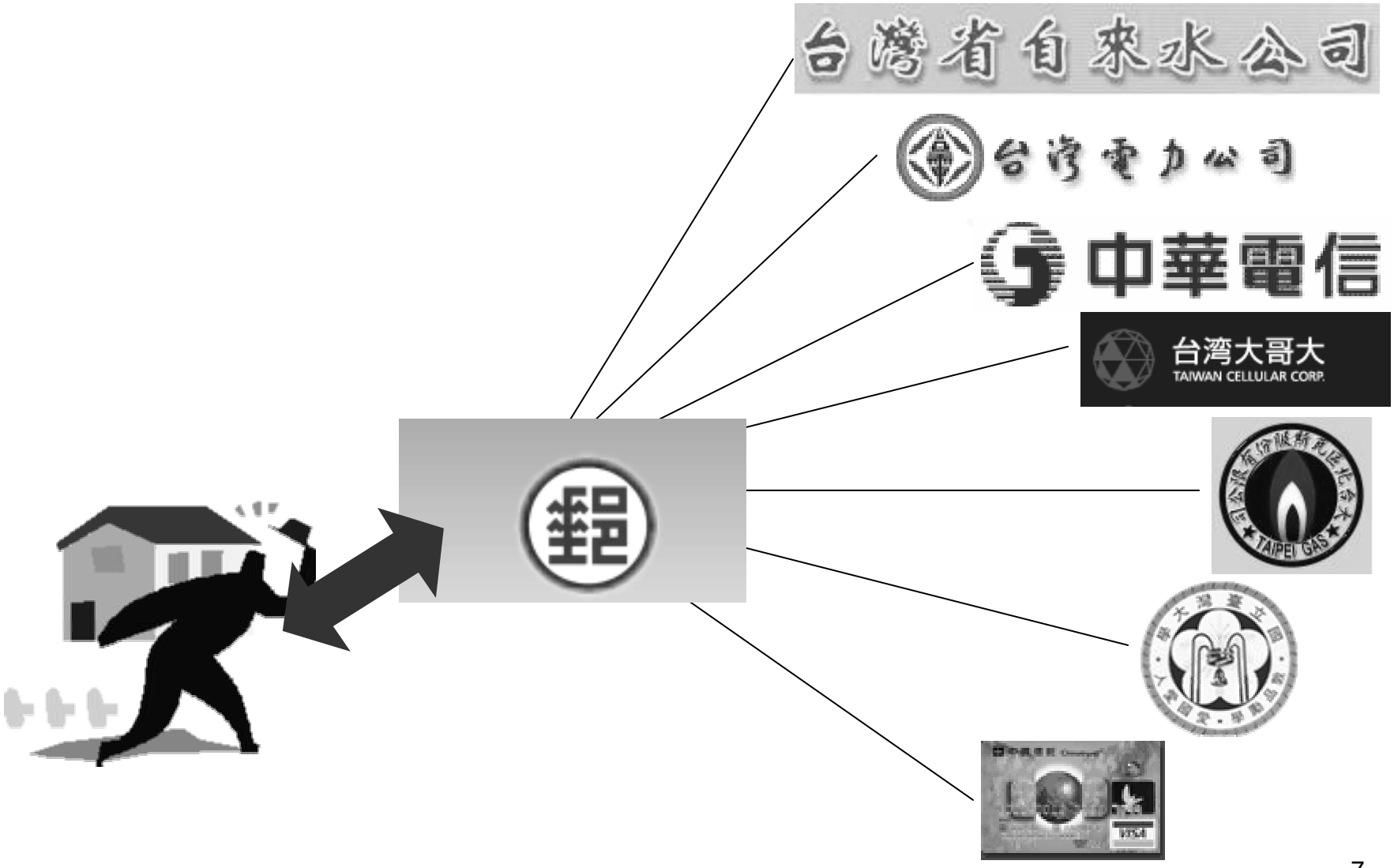
ELEVEN



Address Notification

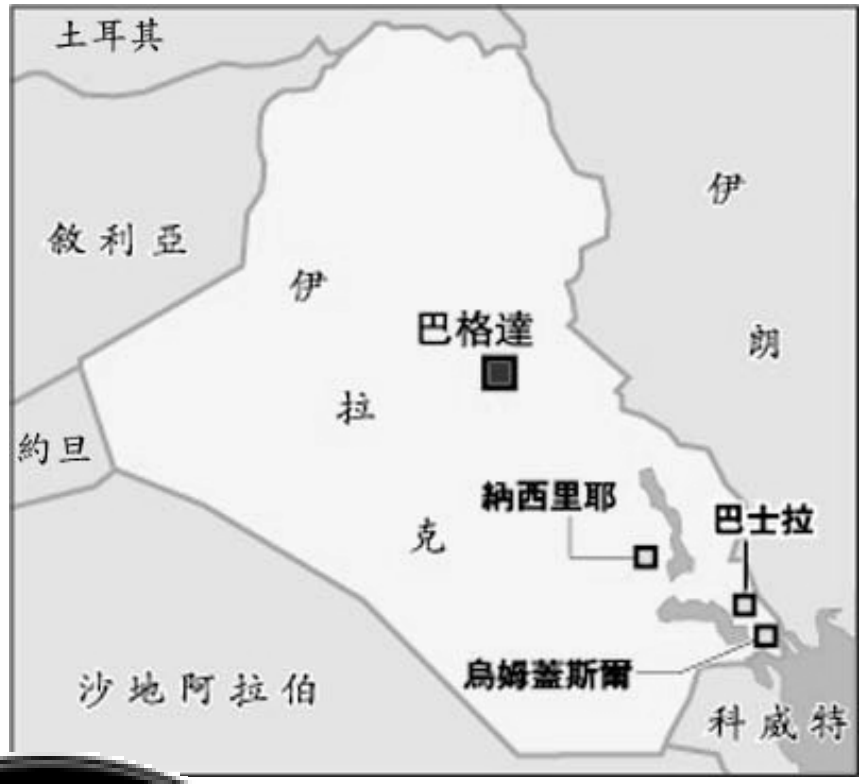


Universal Address Announce Service



Time Coordinate

Universal Coordinated Time Service

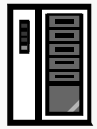


Travel Agent System

觀光地點: 夏威夷自助旅行
交通工具: 搭飛機、開車
時間: 7月份五天三夜

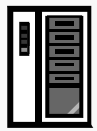
交通工具: 華航CI0018
交通工具: Hertz 福特 Escape
飯店: Holiday Hotel
時間: 7月20日五天三夜

航空公司



訂票系統

旅館

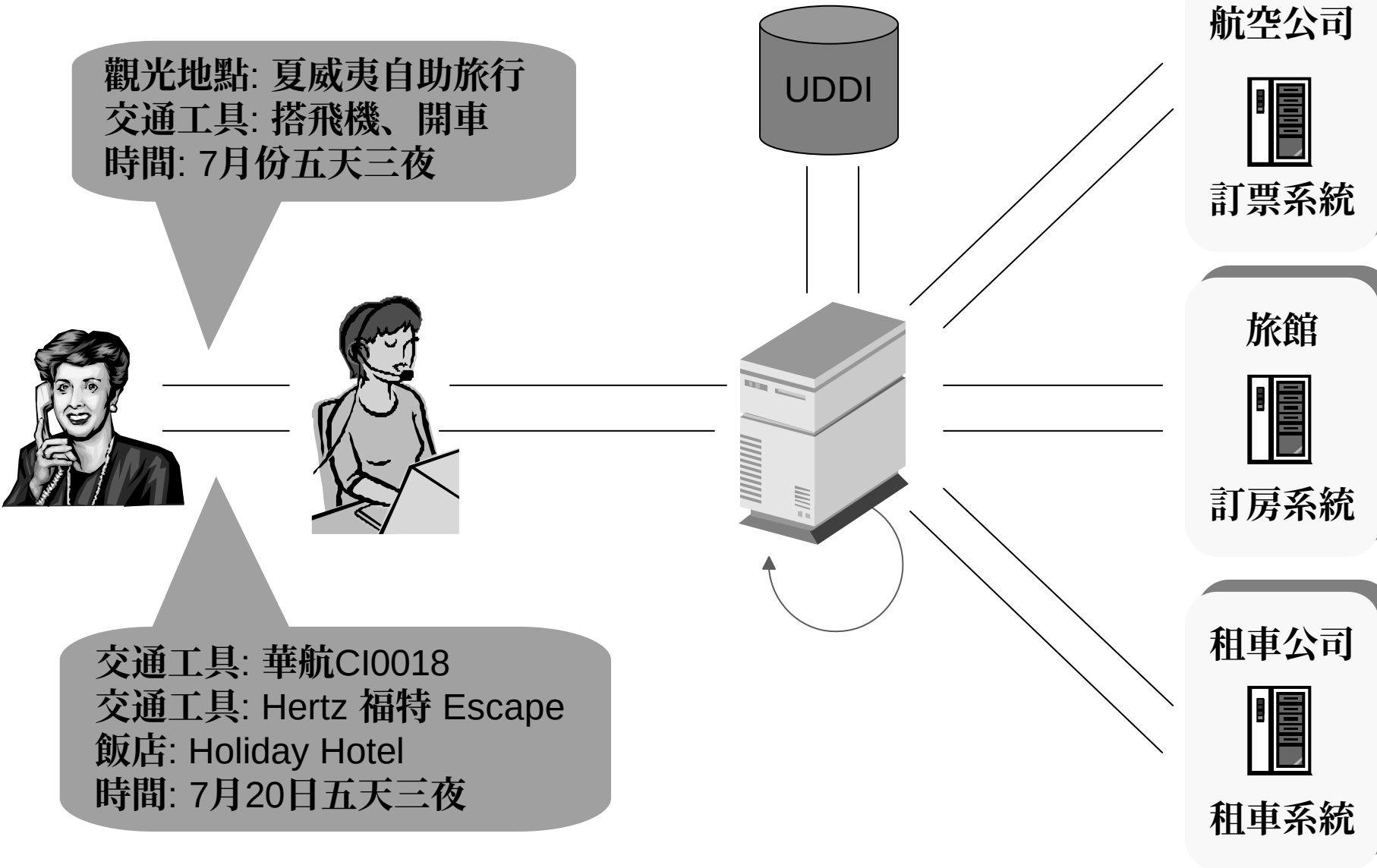


訂房系統

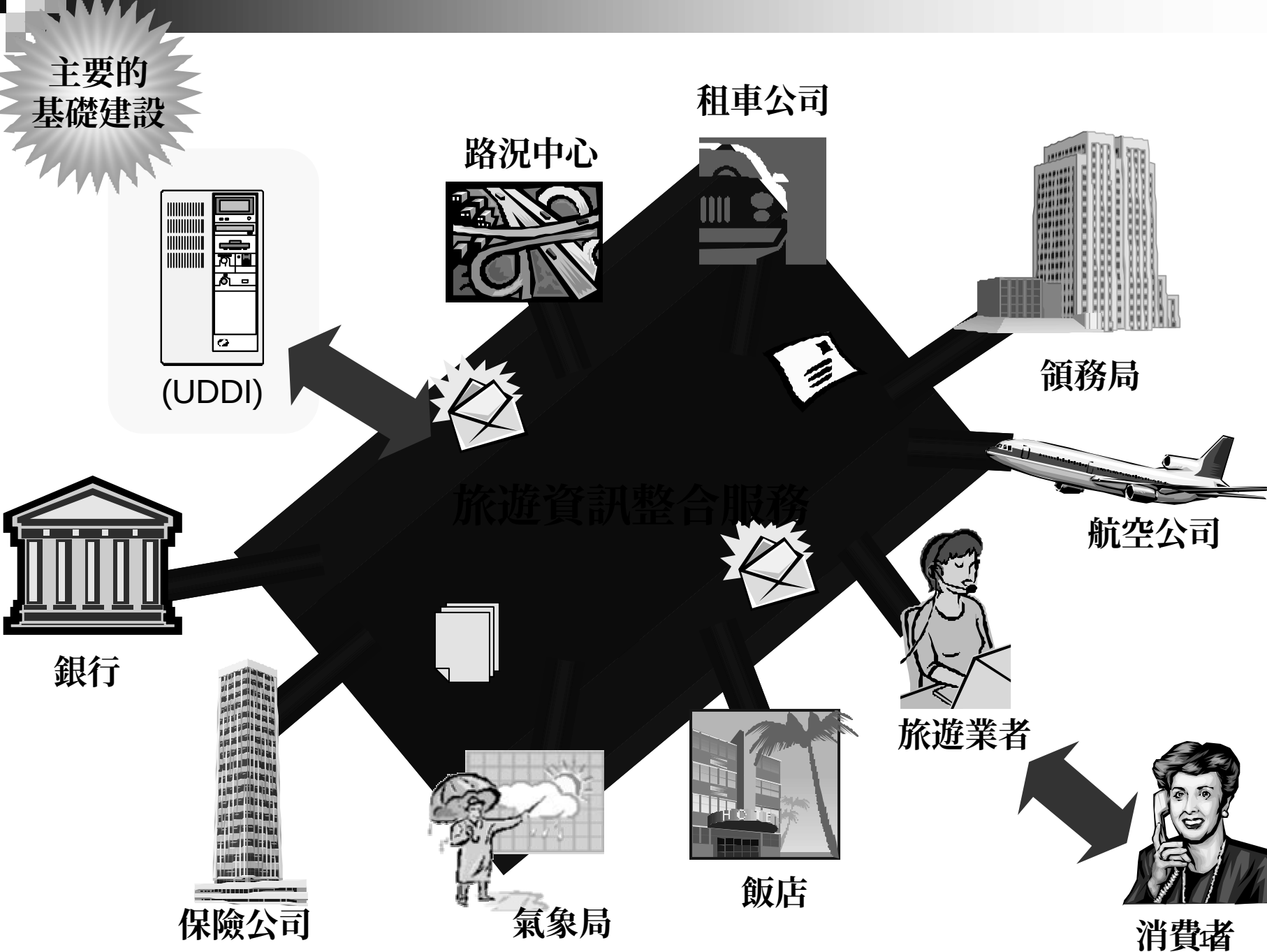
租車公司

租車系統

Travel Agent Web Services



主要的
基礎建設



Outline

- Web Services 範例
- Web Services 簡介
- 分散式元件技術的發展
- Web Services 主要技術簡介
- Web Services 的發展障礙

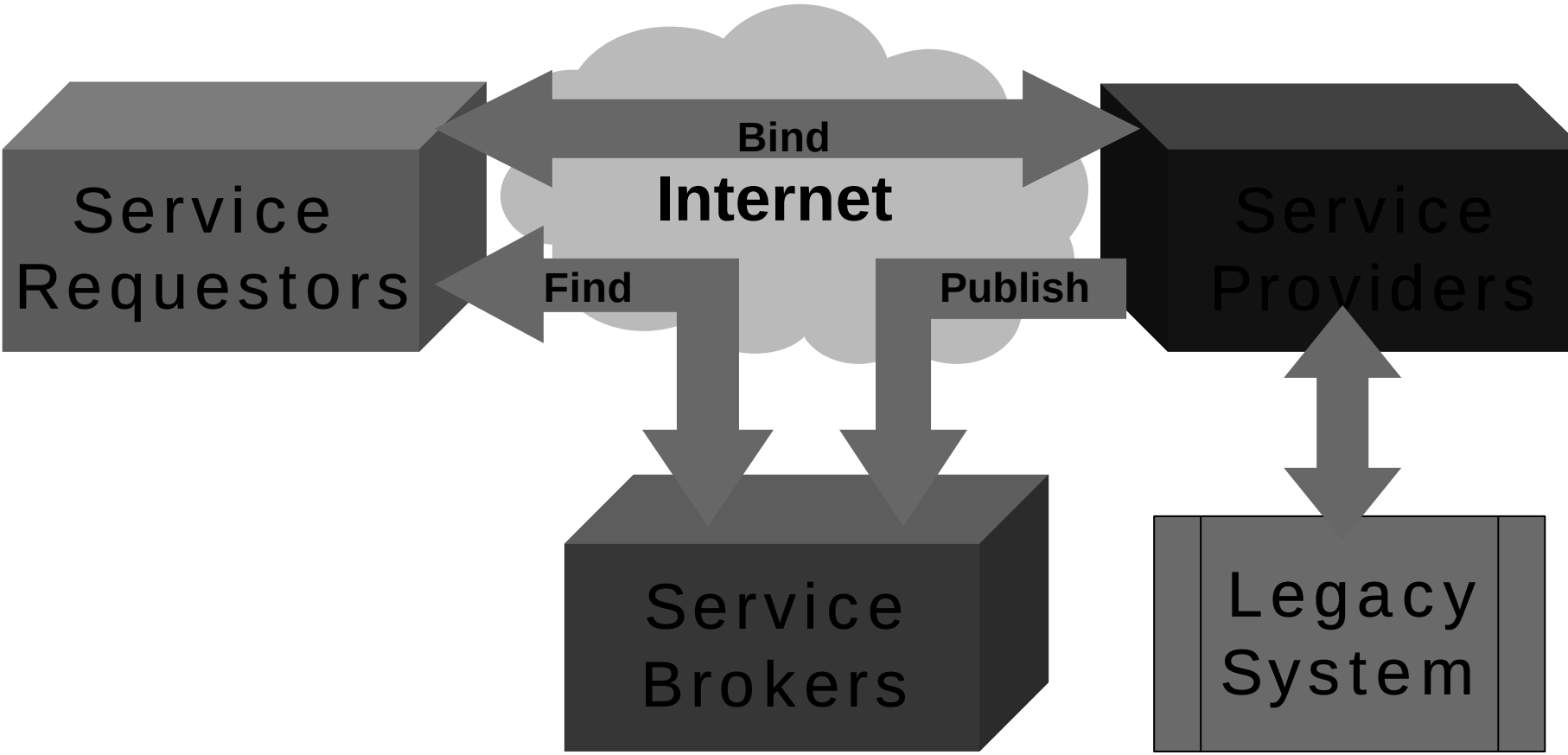
Web Services 簡介

- 什麼是 Web Services?
- Web Services 與 Services of Web 的差異
- 為什麼需要 Web Services？
- Web Services 的重要技術及標準
- Web Services 的流程
- Web Services 與 Web Application 的差異
- Web Services 的特色
- Web Services 應釐清的幾點觀念

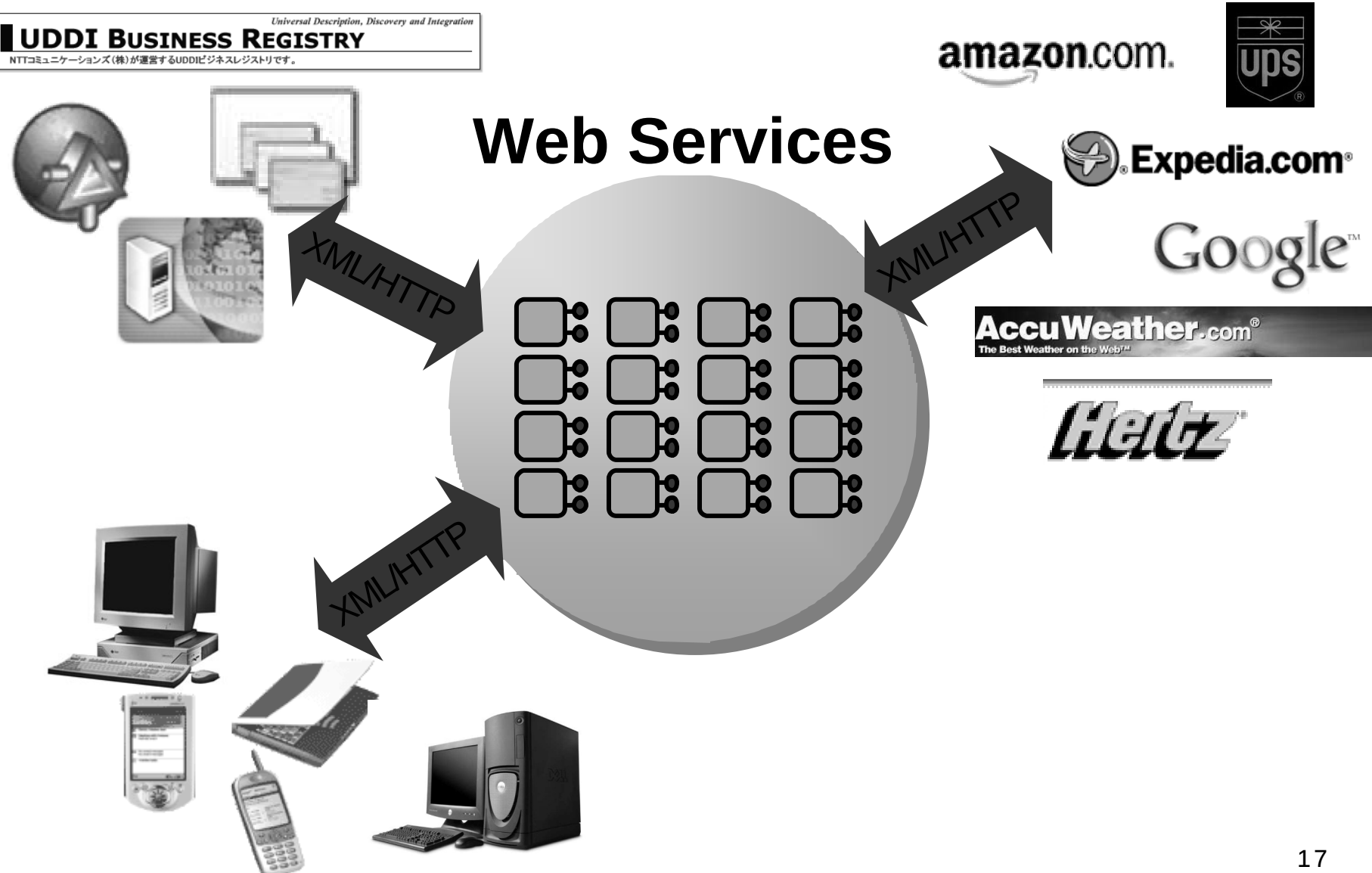
什麼是 Web Services?

- 一種新的程式開發環境
  - 建構於服務導向的應用程式架構
- 結合元件開發的優點及 Web 的優點
  - 一種鬆散偶合的網際網路應用程式架構
- 一種全新的網路經濟模式
  - Web Services 讓不同的電腦系統可以在不以人為介入的情況底下彼此溝通，進而幫企業節省成本；也可因此創造出新的服務付費客戶，以賺取更多的錢

服務導向的應用程式架構



鬆散偶合的網際網路應用程式架構

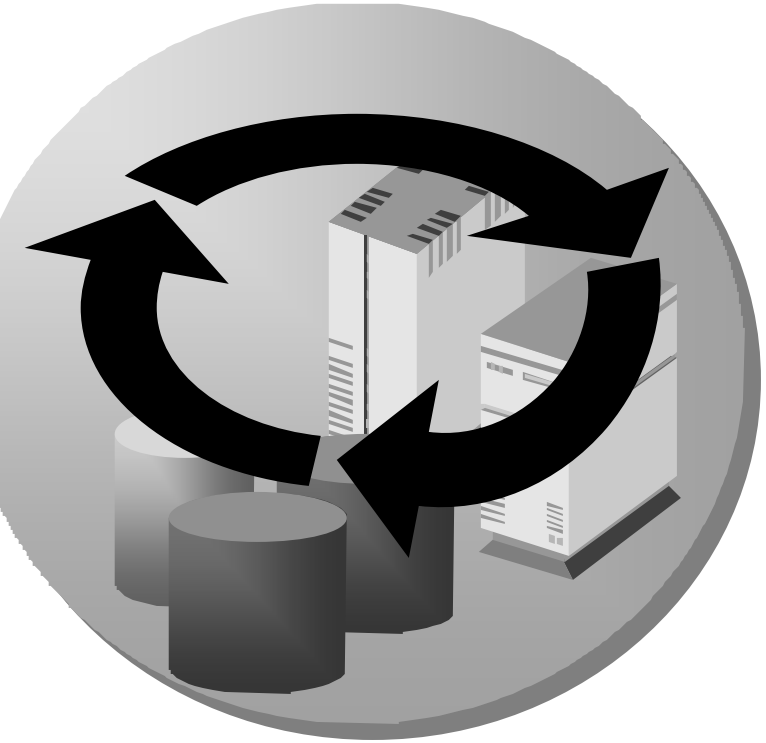


Web Services ≠Services of Web

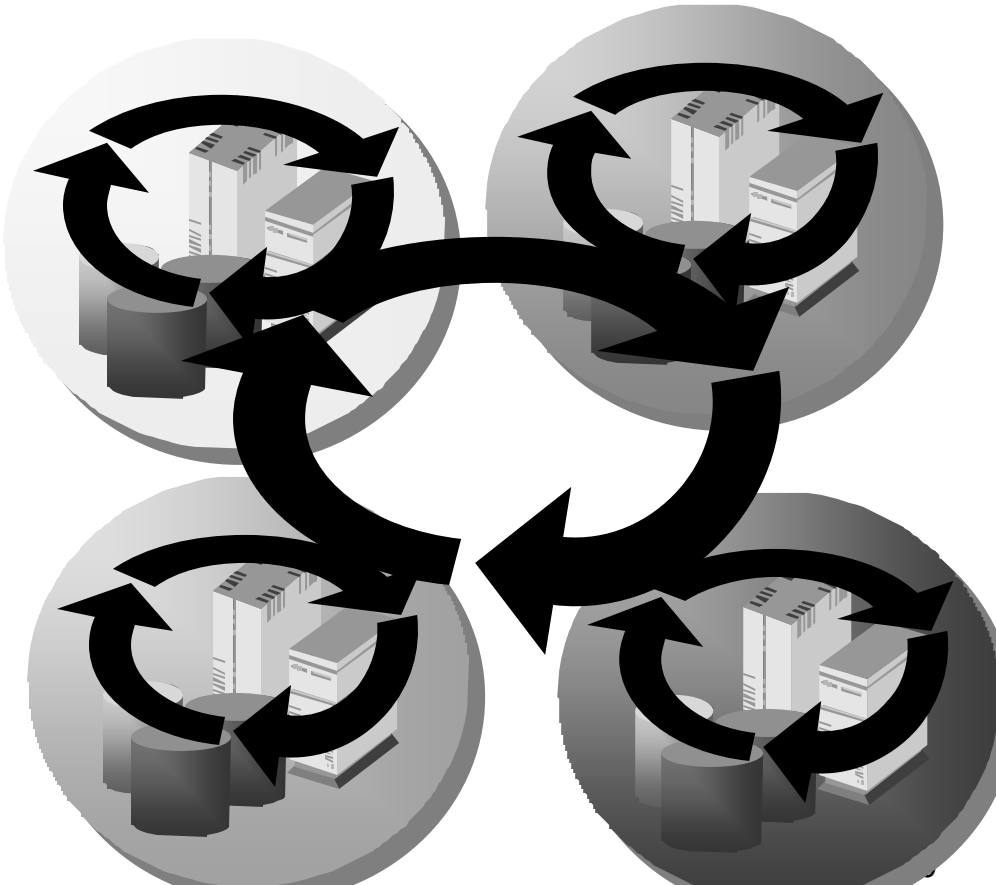
- Web Services（W3C的定義）
  - 透過 URI 方式存取的軟體程式，可透過 XML 定義、描述或搜尋其介面與結合方式（binding），同時也藉由以網際網路為基礎的通訊協定，以 XML 訊息與其它軟體程式溝通
- Web Services 並非只是利用瀏覽器介面提供服務，其運作未必需要瀏覽器，可發生在任何兩個程式之間
- Web Services 是軟體工業發展的新架構，以「服務」的形式提供動態整合，應用程式間可透過網際網路取用軟體元件，並能彼此透過標準相互連結

為什麼需要 Web Services？

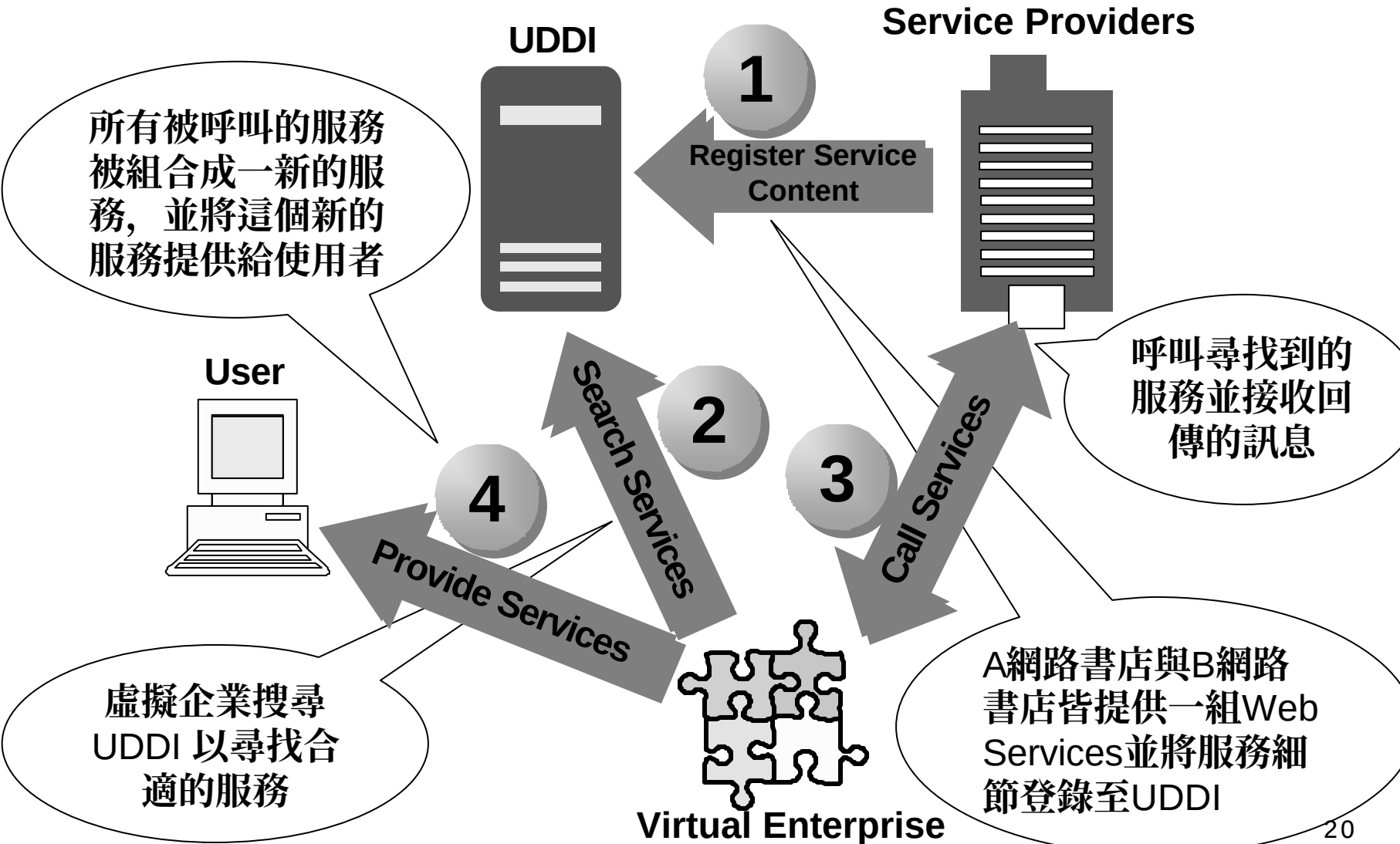
企業內應用程式的互用性（Interoperability）：企業應用系統整合（EAI）



企業間應用程式的互用性：企業與企業間的整合（B2Bi）企業間供應鏈管理自動化與透明化



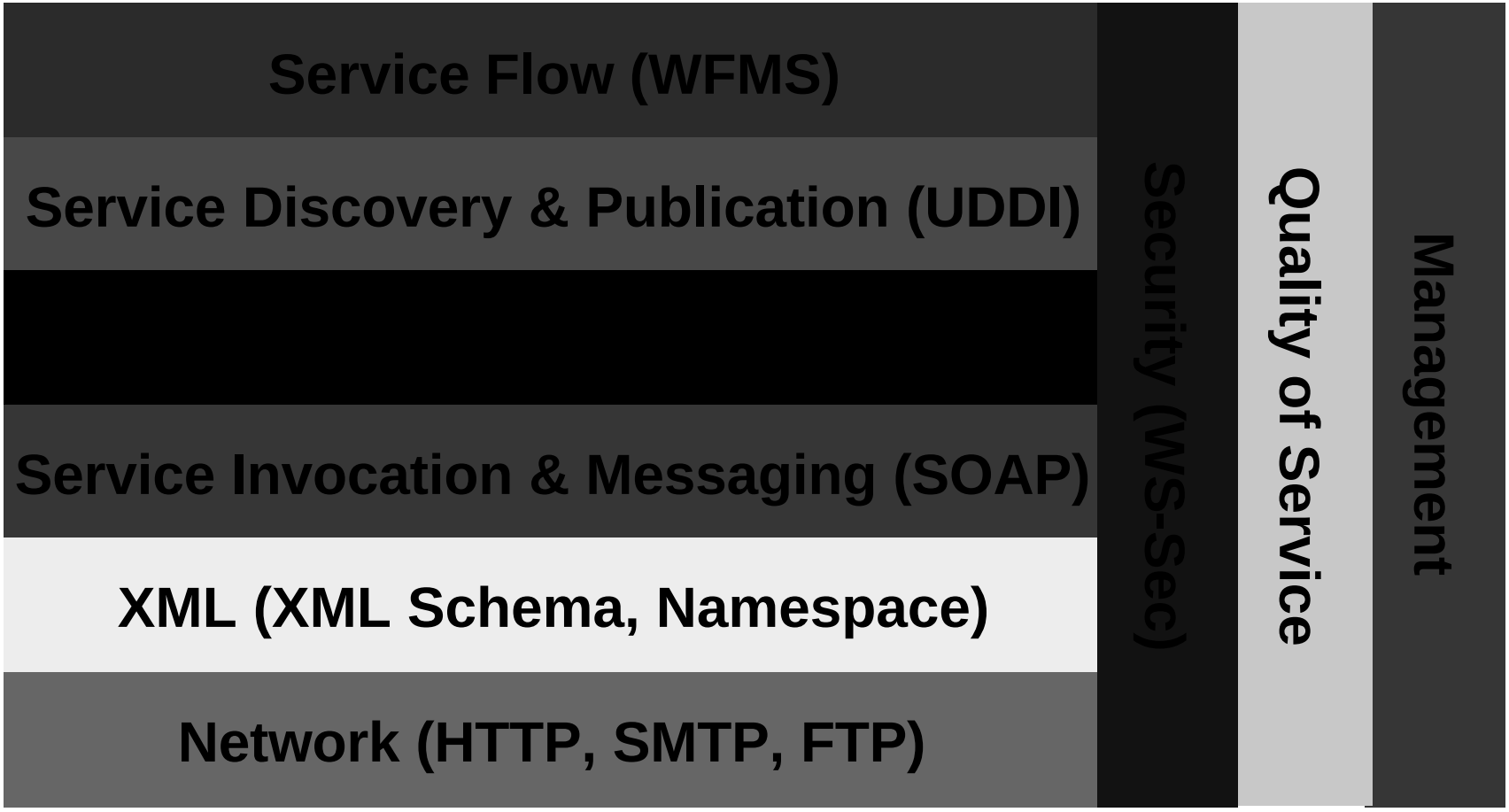
從 B2Bi 的角度來看 Web Services 的應用 – 虛擬企業（Virtual Enterprise）



Web Services 的重要技術及標準

- Web Services 使用 XML（eXtensible Markup Language）作為它的資料架構
- Web Services 使用 SOAP（Simple Object Access Protocol）作為它的通訊協定
- Web Services 使用 WSDL（Web Service Description Language）作為它描述服務細節的標準
- Web Services 使用 UDDI（Universal Description Discovery and Integration）的標準作為它的服務搜尋工具

Web Services Technology Stack



Web Services 與 Web Application的差異

	Web Services	Web Application
提供者與使用者關係	程式對程式	人類對程式
描述語言	XML	HTML
服務的檢索	利用UDDI查詢	利用搜尋引擎查詢
主要應用領域	B2B	B2C
通訊協定	SOAP+HTTP/HTTPS/SMTP	HTTP/HTTPS

Web Services 的特色

- **元件化服務**（Services as Components）
  - 以元件化的方式開發各項服務，並達到服務元件的再利用
- **開放性平台**（Platform-Free）
  - 只要遵循標準，任何裝置或作業系統均可互相連通
- **公開標準**（Open Standard）
  - 制訂有SOAP、UDDI、WSDL等產業公開標準
- **動態整合**（Dynamic Integration）
  - 使用者隨時依需求取得需要的服務，或商業伙伴間透過事先建立的服務元件進行流程整合
- **互通性**（Interoperability）
  - 打破平台之間的籓籬，透過自動化服務進行連通
- **漸進式部署**（Incremental Deployment）
  - 可由現有功能的重新包裝開始，未必要重新開發

對於 Web Services 應釐清的觀念

- Web Services 強調互通性（interoperability），而不是可攜性（portability）
  - 將 legacy 的應用程式於原始平台上重新包裝，而非移植至新平台
- Web Services 不是整合問題的唯一解決方案
  - 資訊應用整合不一定非 Web Services 不可，但對於有此需求的應用， Web Services 則是一項值得投入的技術
- Web Services 的價值在於商業經營觀點的思考
  - 經由新服務模式的創造，以開創出新的商業模式

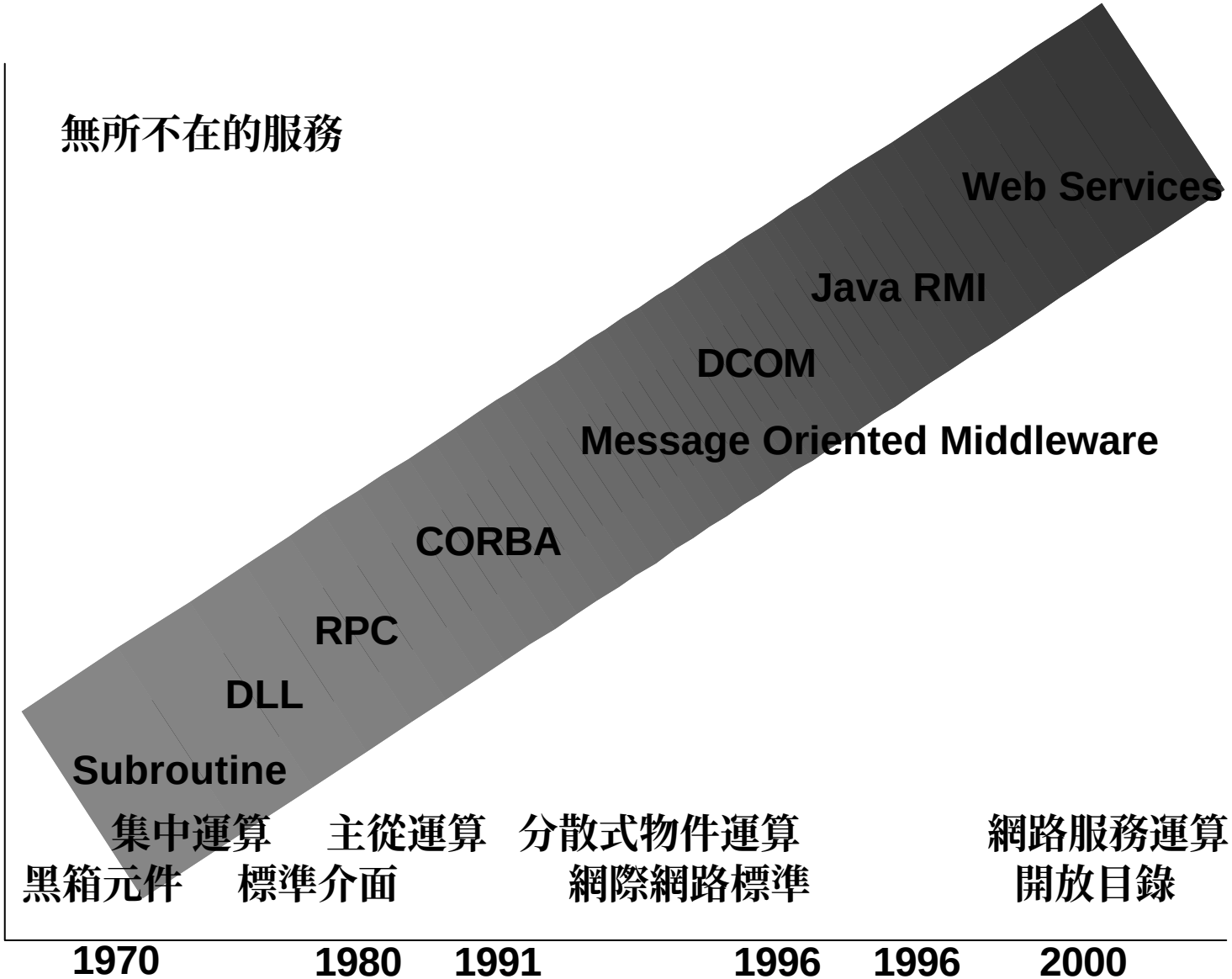
Outline

- Web Services 範例
- Web Services 簡介
- 分散式元件技術的發展
- Web Services 主要技術簡介
- Web Services 的發展障礙

分散式元件技術的發展

- 軟體工程的演進
- 分散式計算環境的評估
- 分散式元件技術之比較

軟體工程的演進


無所不在的服務
Web Services
Java RMI
DCOM
Message Oriented Middleware
CORBA
RPC
DLL
Subroutine
集中運算
主從運算
分散式物件運算
網路服務運算
黑箱元件
標準介面
網際網路標準
開放目錄
1970
1980
1991
1996
1996
2000

分散式計算環境的評估

- 透明性（Transparency）
- 互通性（Interoperability）
- 可擴充性（Scalability）
- 可攜性（Portability）
- 安全性（Security）
- 可靠性（Reliability）、可用性（Availability ）及效能（Performance）

分散式元件技術之比較

	CORBA	DCOM	Java RMI	SOAP
RPC通訊協定	IIOP	RPC	IIOR or JRMP	HTTP
訊息格式	CDR	NDR	Java Ser. Format	XML
描述語言	OMG IDL	IDL	Java	WSDL
探索機制	Naming Service	Windows Registry	RMI Registry	UDDI

Outline

- Web Services 範例
- Web Services 簡介
- 分散式元件技術的發展
- Web Services 主要技術簡介
- Web Services 的發展障礙

Web Services 主要技術簡介

- XML（eXtensible Markup Language）
  - 以SGML為基礎的一種標示語言（markup language），可透過自我定義標籤，定義文件格式與結構化資訊
- SOAP（Simple Object Access Protocol）
  - Web Services 的訊息傳輸與資料存取協定，作為訊息收送雙方溝通的標準
- WSDL（Web Service Description Language）
  - Web Services 的描述語言，以 XML 語法描述 Web Services 的資料、命令與各元件所提供的功能
- UDDI（Universal Description Discover and Integration）
  - Web Services 的註冊與搜尋機制，服務提供者可將所提供的服務至 UDDI 資料庫進行註冊，服務要求者則可至其中取用已註冊的各項服務

Web Services 主要技術簡介

Standard	WSDL	SOAP	UDDI
Maintain by	W3C	W3C	OASIS
Recent status	1.2	1.2	3.0

Web Services 主要技術簡介

- SOAP
- WSDL
- UDDI

SOAP 的發展歷史

- 1998 年，UserLand 公司的執行總裁 Dave Winer 提出透過 XML 讓 RPC 的通訊方式透過 HTTP 協定在網際網路上執行
  - XML-RPC
- 經由微軟公司加以改良，提出了實際可行的 Simple Object Access Protocol (SOAP)通訊協定
- 2000 年由IBM、Microsoft、UserLand 和 DevelopMentor 共同提交給 W3C

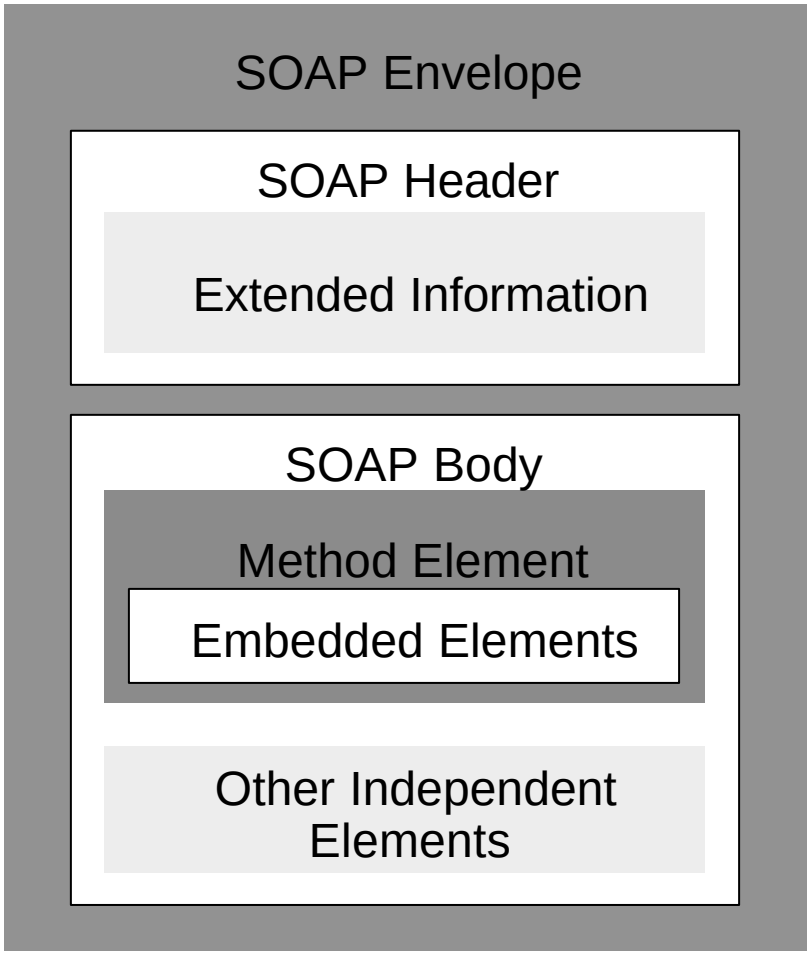
SOAP 標準中所定義的事項

- Envelop
  - 定義了整個訊息架構與如何處理訊息的指示
- Encoding Rules
  - 定義如何將應用程式資料表現於SOAP訊息內的方法
- RPC Representation
  - 定義了用SOAP來表示遠端程序叫用時所應遵尋的規範

SOAP 的技術規格

- SOAP是利用所謂的「訊息」（Message）為溝通的基本單位。而一個標準的SOAP訊息，就是一個制式的XML文件，並包含了以下幾個基本元素：
  - 一個SOAP封套（Envelop），用來定義整個SOAP訊息的內容。
  - 一個SOAP訊息標頭紀錄（Header），此部份為選用，包含了所有標頭所應登錄的資料。
  - 一個SOAP訊息主體（Body），其中包括了所有的呼叫描述與回應內容。

標準的 SOAP 訊息結構圖



SOAP Message 的命名空間

http://schemas.xmlsoap.org/soap/envelope/

http://schemas.xmlsoap.org/soap/encoding/

Example：Hello Service

```
public interface Hello {
  public String sayHelloTo(String
name);
}
```

Simplified SOAP Request

```
<?xml version="1.0"?>
<Hello>
    <sayHelloTo>
        <name>John</name>
    </sayHelloTo>
</Hello>
```

Simplified SOAP Response

```
<?xml version="1.0"?>
<Hello>
    <sayHelloToResponse>
        <message>
            Hello John, How are you?
        </message>
    </sayHelloToResponse>
</Hello>
```

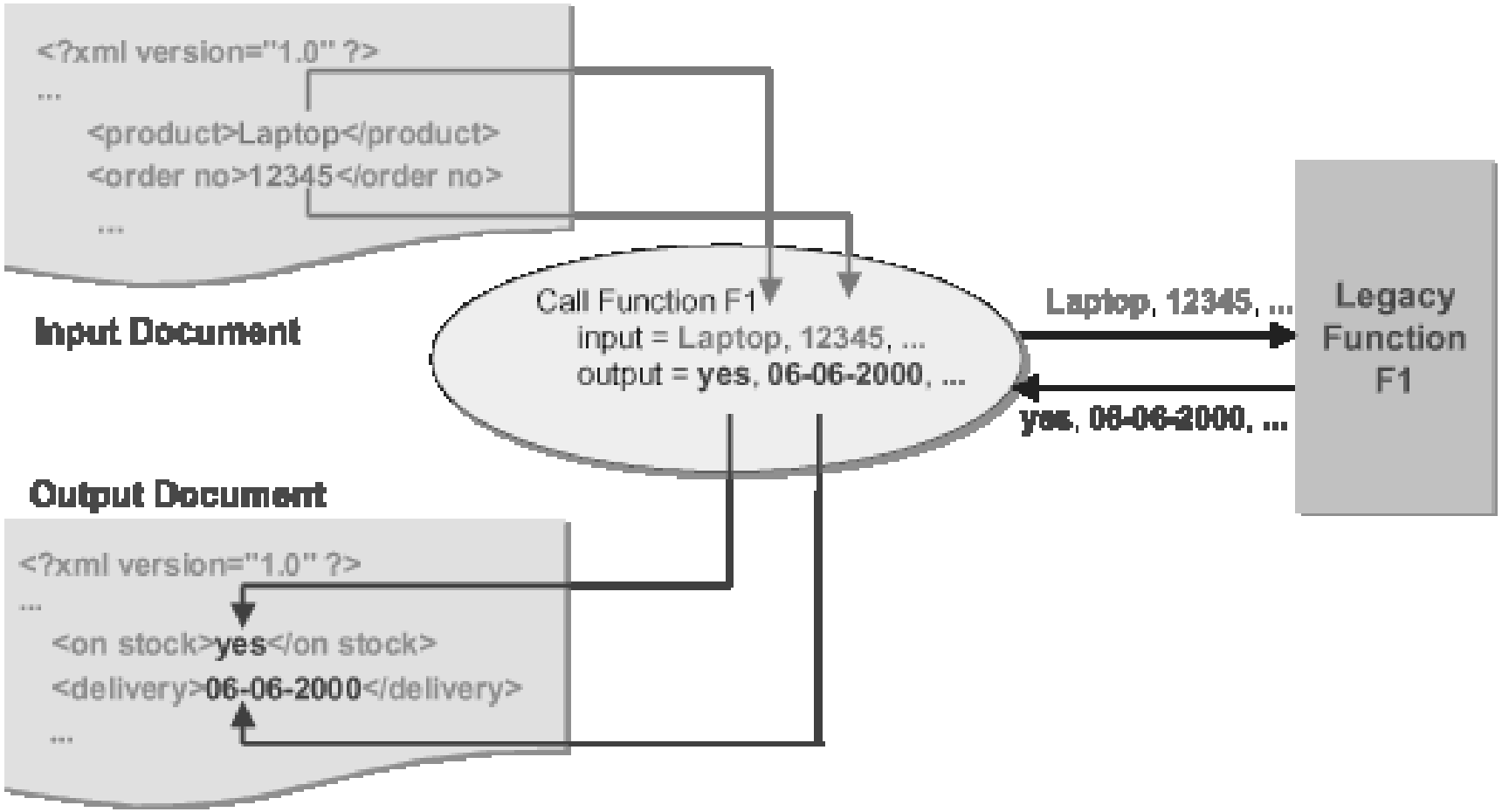
SOAP Request

```
<SOAP-ENV:Envelope
xmlns:SOAP-
ENV="http://schemas.xmlsoap.org/soap/envelope/
"
xmlns:xsd="http://www.w3.org/1999/XMLSchema">
  <SOAP-ENV:Header> </SOAP-ENV:Header>
  <SOAP-ENV:Body>
      <ns1:sayHelloTo xmlns:ns1="Hello"
        SOAP-ENV:encodingStyle=

"http://schemas.xmlsoap.org/soap/encoding/">
          <name type="xsd:string">John</name>
      </ns1:sayHelloTo>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

SOAP Respone

```
<SOAP-ENV:Envelope xmlns:SOAPENV="
http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/1999/XMLSchema">
  <SOAP-ENV:Header> </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    <ns1:sayHelloToResponse xmlns:ns1="Hello"
      SOAP-ENV:encodingStyle=

"http://schemas.xmlsoap.org/soap/encoding/">
      <return type="xsd:string">
        Hello John, How are you doing?
      </return>
    </ns1:sayHelloToResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

從 SOAP 的角度來看 Web Services 的應用 – 中介軟體（Middleware）



SOAP Message 的優點

- 不需特定的技術才能傳送
  - SOAP 利用 XML 純文字的特性，提供一套機制，以 XML 來包裝方程式呼叫和訊息。由於採用了 XML，SOAP 順理成章地繼承了許多 XML 的優點，可輕易透過 HTTP、SMTP 等網路上最常使用、極為耐操的通信管道來來挾帶，更能穿越企業的 firewalls，還可透過 SSL、S/MIME 等機制加密，提供安全性
- 不需特定的平台才能接收
  - 透過 XML 來傳訊，還有一項更大的優點，是 CORBA、Java RMI，及 DCOM 這些以專屬 binary 格式傳送資料所不及之處，那就是對程式語言、作業平台的獨立性——由於是純文字 XML 格式，SOAP 訊息可由任何一種程式語言所產生，被任何程式語言、甚至肉眼所解讀。這個 late-binding 的特性，正是 Web-services 時代所迫切需要的

SOAP Message 的缺點

- 利用 XML Parser 來解釋 SOAP 訊息封套中的 XML 資訊是很耗費時間的
- 在 SOAP 訊息中並沒有提供服務執行先後順序的資訊
- 因為 SOAP 的編碼規則，使得在所有的 SOAP 訊息封套中都必須包含一些額外的型別資訊。因此會多佔用一些儲存空間，且增加系統編碼及解碼的額外處理時間

Web Services 主要技術簡介

- SOAP
- WSDL
- UDDI

WSDL 所提供的資訊

- 描述 Service Provider 所提供的 Web Server 的 operations
- 描述 Service Requester 如何和 Web Server 的 operations 溝通，內容包括傳輸協定、格式及參數

WSDL檔的主要部份

<definitions name="Weather_Service">
  <message> ... 定義通訊傳輸所需使用的訊息
  </message>
  <portType> ... Operations 和 operations 的 messages
  </portType>
  <binding> ... *How* the operations are invoked
  </binding>
  <service> ... *Where* the service is located
  </service>
</definitions>

Web Services 主要技術簡介

- SOAP
- WSDL
- UDDI

UDDI 的註冊資料

- White pages
  - 包含企業基本資料及一組識別碼
- Yellow pages
  - 包含企業的分類
- Green pages
  - 包含業務上所提供服務的技術資訊，例如如何計價等。另外也包含商業流程、服務規格，以及連結資訊等

White Pages

- Business Name
- Text Description
  - list of multi-language text strings
- Contact info
  - names, phone numbers, fax numbers, web sites…
- Known Identifiers
  - list of identifiers that a business may be known by – D-U-N-S (UDDI registry generated unique number for each business)

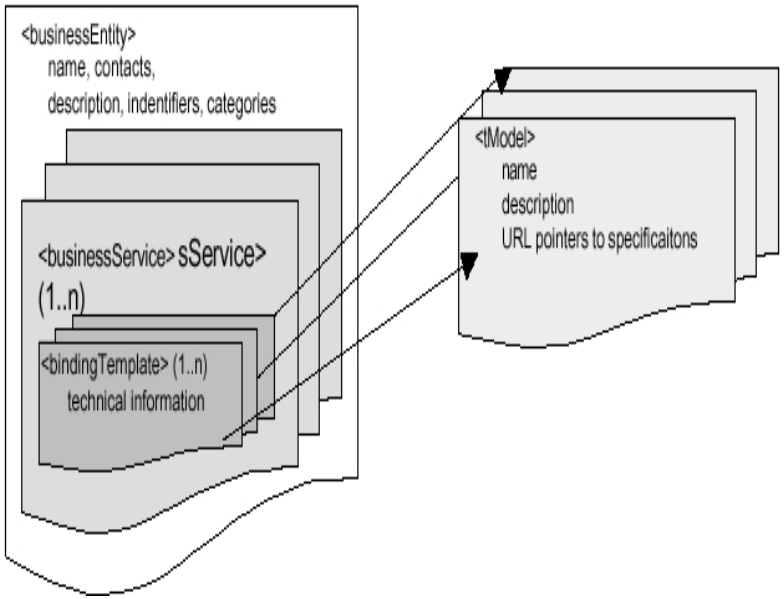
Yellow Pages

- Business categories
  - 5 standard taxonomies in Version 2.0
    - Industry: NAICS (Industry codes - US Govt.)
    - Product/Services: Standard Industrial Classification, USPSC
    - Location: Geographical taxonomy (GGC, ISOGT)
  - Implemented as name-value pairs to allow any valid taxonomy identifier to be attached to the business white page

Green Pages

- New set of information businesses use to describe how to “do e-commerce” with them
  - Nested model
    - Business processes
    - Service descriptions
    - Binding information
  - Programming/platform/implementation independent
  - Services can also be categorized

UDDI 的資料結構



- All XML based
- API implementations
  - Inquire
  - Publish
- Close to 15 structures
- Main Data Structures
  - businessEntity
  - serviceEntity
  - tModel
  - bindingTemplate

Outline

- Web Services 範例
- Web Services 簡介
- 分散式元件技術的發展
- Web Services 主要技術簡介
- Web Services 的發展障礙

Web Services 的發展障礙

- 程式除錯的困難性
- UDDI 的互通性
- 服務的可信任性
- 服務的身份辨識性
- 分散式交易的交付
  - Transaction Timing
  - Rollback