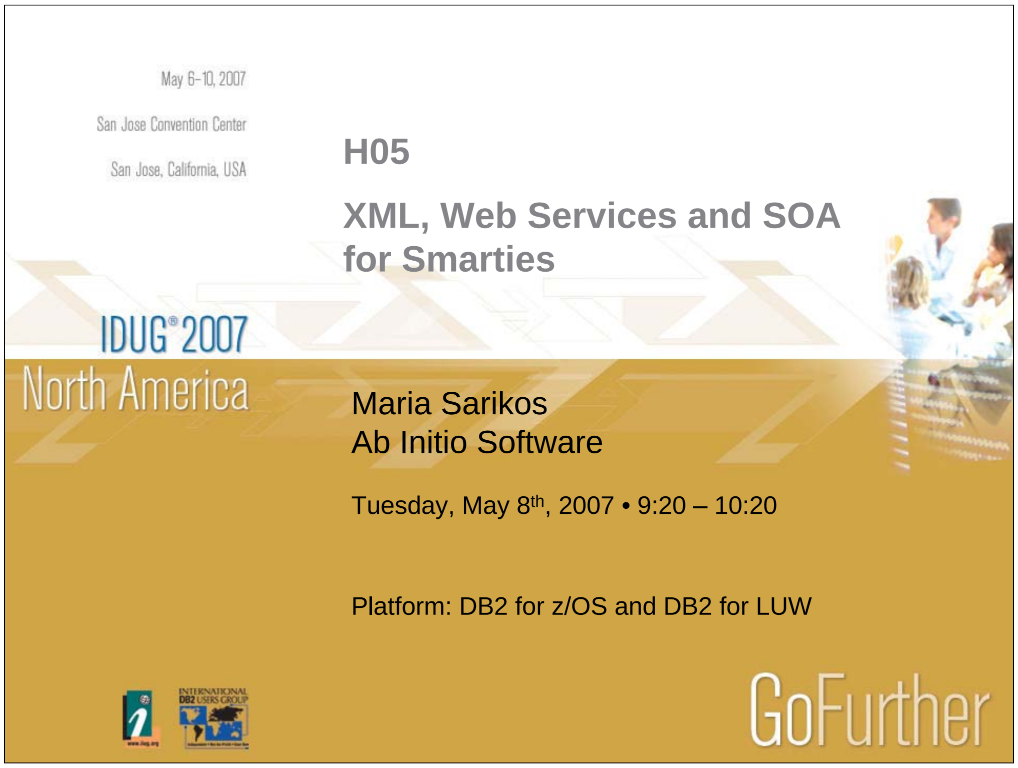


May 6–10, 2007

San Jose Convention Center

San Jose, California, USA

H05

XML, Web Services and SOA for Smarties

Maria Sarikos
Ab Initio Software

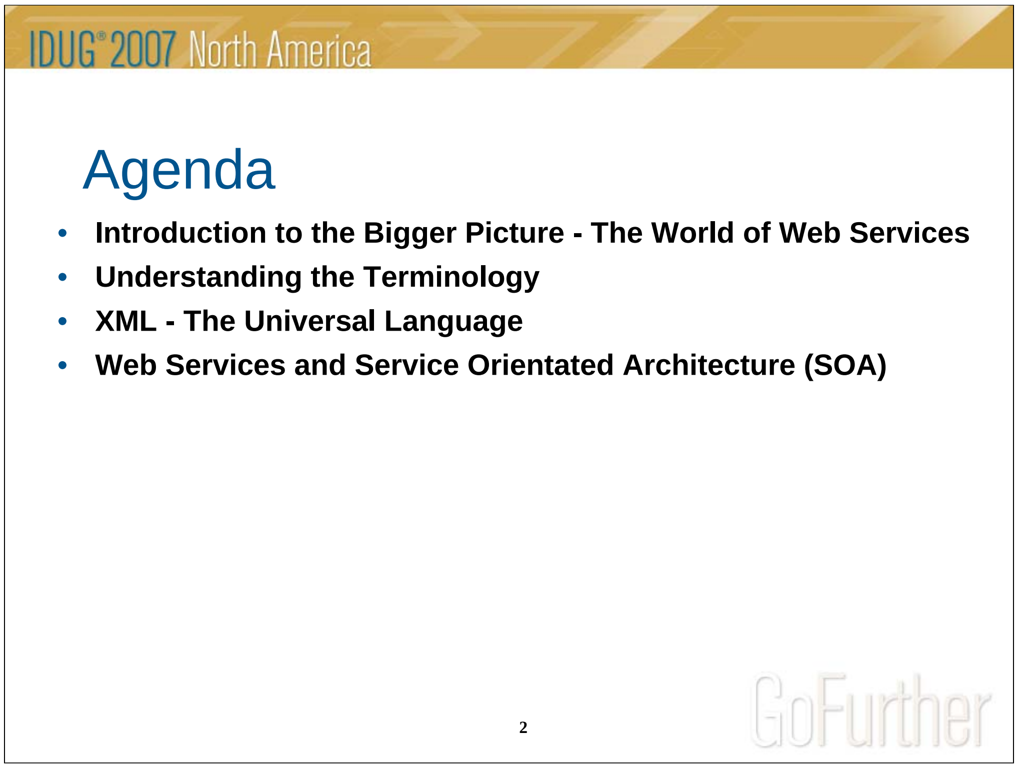
Tuesday, May 8th, 2007 • 9:20 – 10:20

Platform: DB2 for z/OS and DB2 for LUW

IDUG® 2007 North America

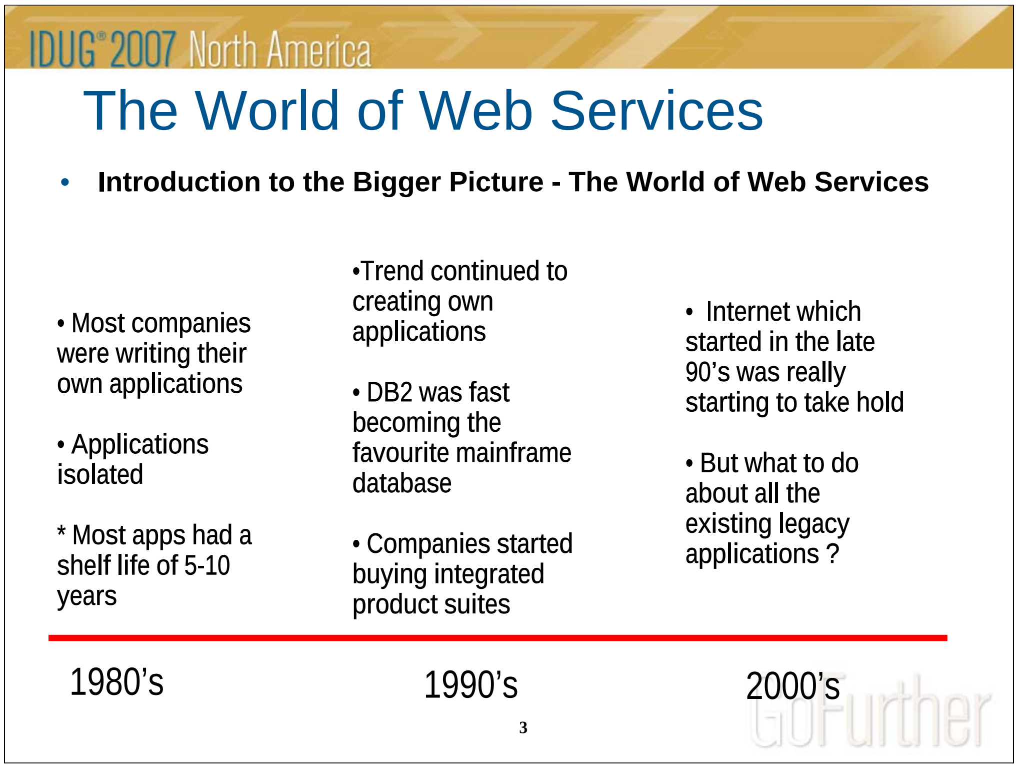
GoFurther

XML, Web Services and SOA (Service Orientated Architecture) are taking the world by storm! DB2 is fast becoming an XML database. Suddenly everyone is talking about Web Services. The second avalanche after the Web/Java/Internet ordeal is heading straight towards you. But what do all these concepts mean? How do they fit into the bigger company-wide picture and how do they affect you the DB2 professional? This presentation discusses the terminology, concepts and the relationships they have with DB2 in an in-depth and concise manner.



Agenda

- Introduction to the Bigger Picture - The World of Web Services
- Understanding the Terminology
- XML - The Universal Language
- Web Services and Service Orientated Architecture (SOA)



The World of Web Services

- **Introduction to the Bigger Picture - The World of Web Services**

1980's

• Most companies were writing their own applications

• Applications isolated

* Most apps had a shelf life of 5-10 years

1990's

•Trend continued to creating own applications

• DB2 was fast becoming the favourite mainframe database

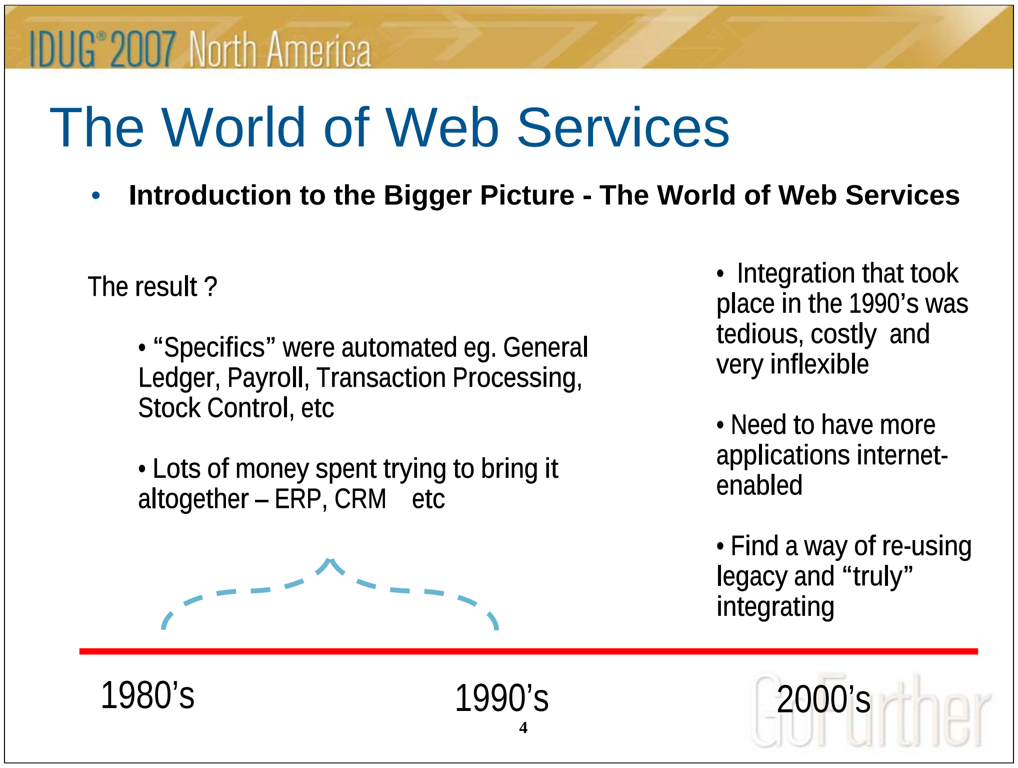
• Companies started buying integrated product suites

2000's

• Internet which started in the late 90's was really starting to take hold

• But what to do about all the existing legacy applications ?

GoFurther



IDUG® 2007 North America

The World of Web Services

- **Introduction to the Bigger Picture - The World of Web Services**

The result ?

• "Specifics" were automated eg. General Ledger, Payroll, Transaction Processing, Stock Control, etc

• Lots of money spent trying to bring it altogether – ERP, CRM etc

• Integration that took place in the 1990's was tedious, costly and very inflexible

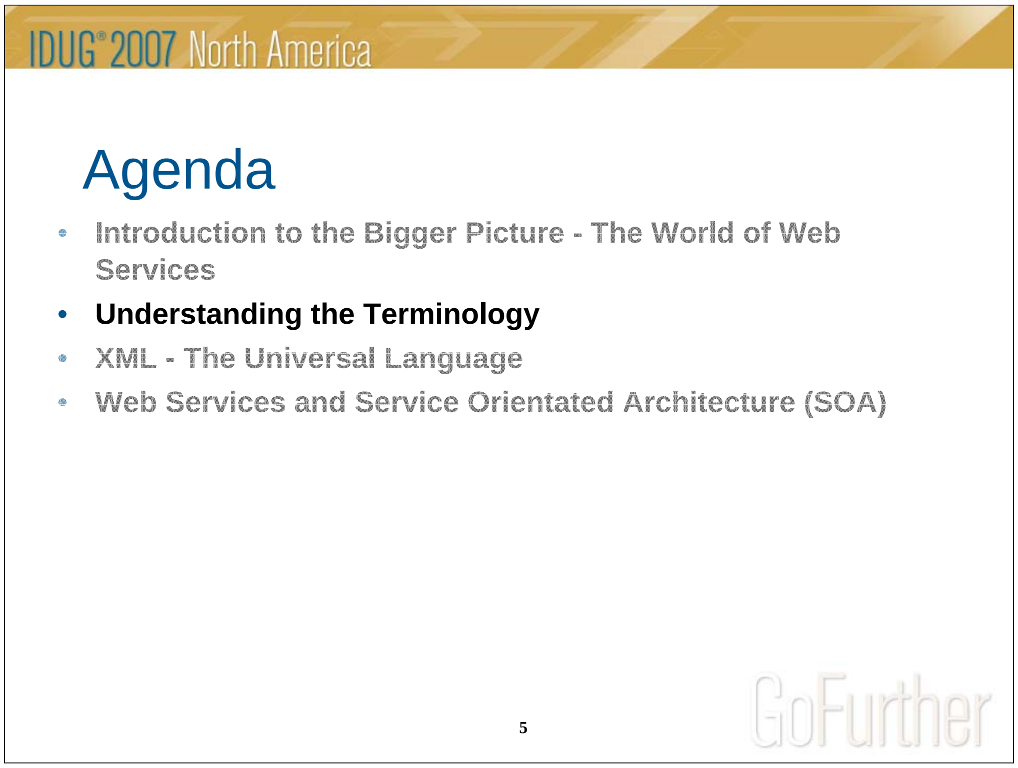
• Need to have more applications internet-enabled

• Find a way of re-using legacy and "truly" integrating

1980's 1990's 2000's

GoFurther

4

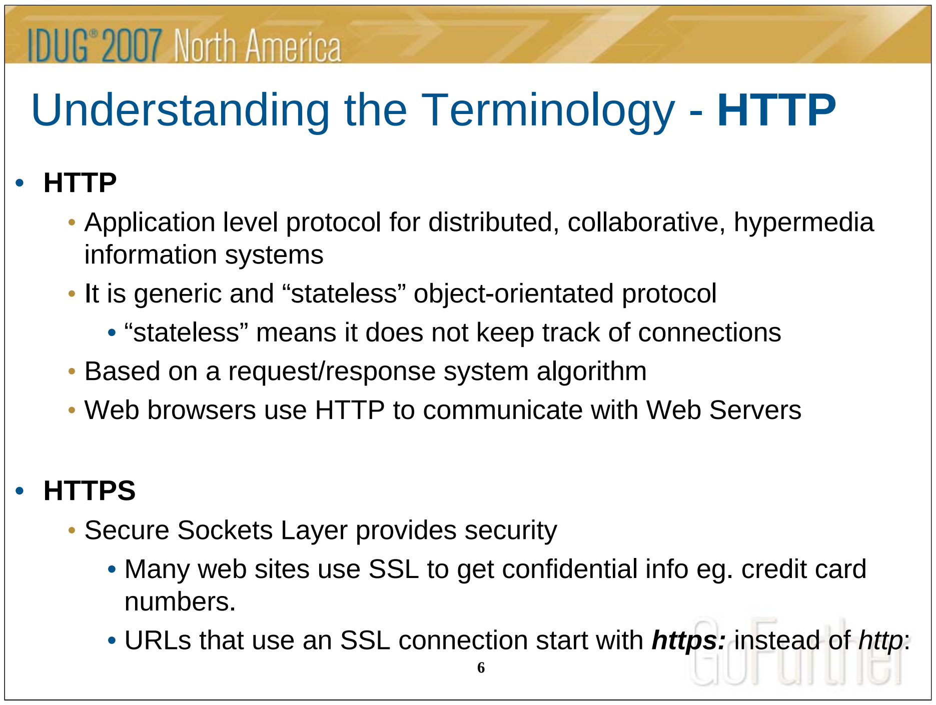

IDUG® 2007 North America

Agenda

- Introduction to the Bigger Picture - The World of Web Services
- **Understanding the Terminology**
- XML - The Universal Language
- Web Services and Service Orientated Architecture (SOA)

5

GoFurther


IDUG® 2007 North America

Understanding the Terminology - **HTTP**

- **HTTP**
  - Application level protocol for distributed, collaborative, hypermedia information systems
  - It is generic and "stateless" object-orientated protocol
    - "stateless" means it does not keep track of connections
  - Based on a request/response system algorithm
  - Web browsers use HTTP to communicate with Web Servers
- **HTTPS**
  - Secure Sockets Layer provides security
    - Many web sites use SSL to get confidential info eg. credit card numbers.
    - URLs that use an SSL connection start with ***https:*** instead of *http*:

6

GoFurther

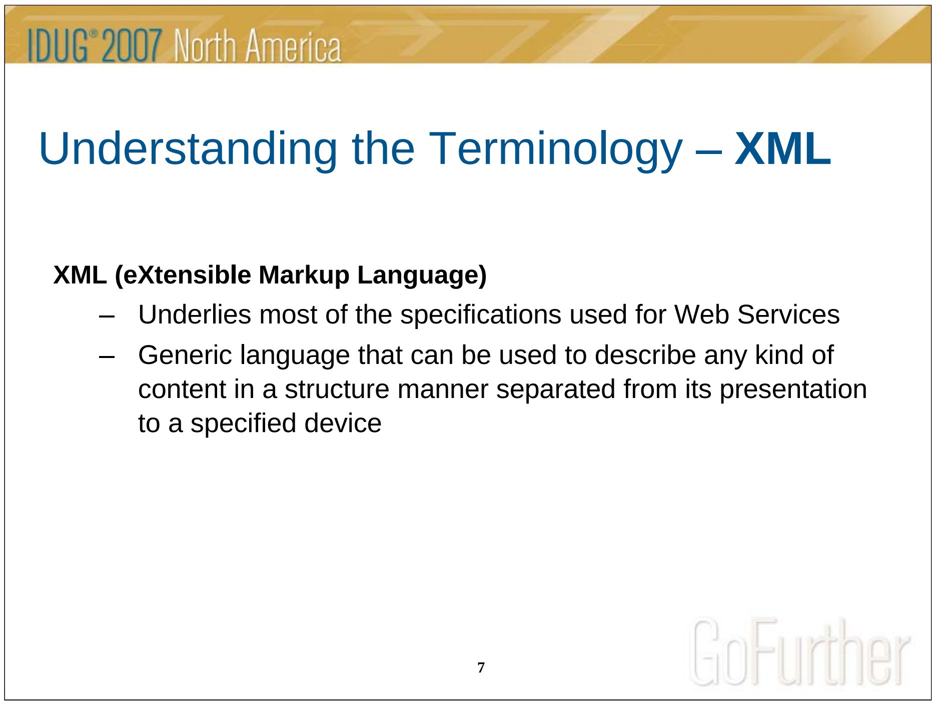
HTTP

HTTP is the RPC (remote procedure call) on top of TCP/IP. It is used to access and retrieve URLs. The HTTP RPC is stateless. The client establishes a connection to the remote server, then issues a request. The server then processes the request, returns a response, and closes the connection. A client (usually a Web browser) requests a hypertext page, then issues a sequence of requests to retrieve any images referenced in the document. Once the client has retrieved the images, the user will typically click on a hypertext link and move to another document.

SSL

Short for ***S**ecure **S**ockets **L**ayer,* a protocol developed by Netscape for transmitting private documents via the Internet. SSL works by using a public key to encrypt data that's transferred over the SSL connection. Both Netscape Navigator and Internet Explorer support SSL, and many Web sites use the protocol to obtain confidential user information, such as credit card numbers. By convention, URLs that require an SSL connection start with *https:* instead of *http*:.

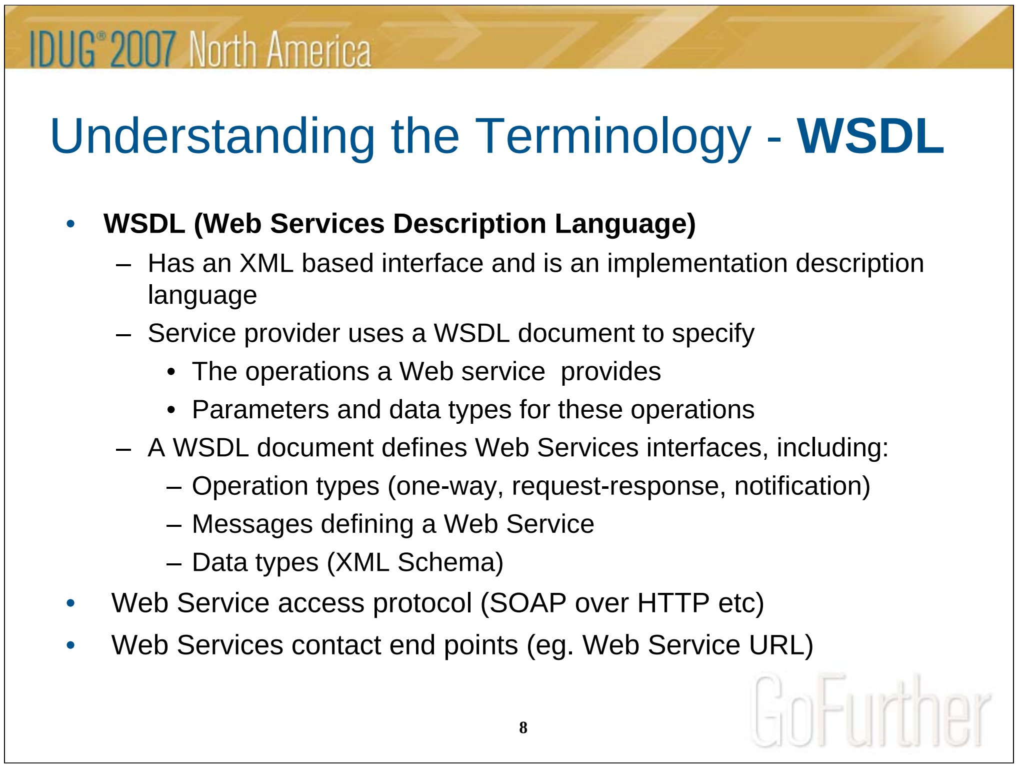
Another protocol for transmitting data securely over the World Wide Web is *Secure HTTP (S-HTTP)*. Whereas SSL creates a secure connection between a client and a server, over which any amount of data can be sent securely, S-HTTP is designed to transmit individual messages securely. SSL and S-HTTP, therefore, can be seen as complementary rather than competing technologies. Both protocols have been approved by the Internet Engineering Task Force (IETF) as a standard.



Understanding the Terminology – **XML**

XML (eXtensible Markup Language)

- Underlies most of the specifications used for Web Services
- Generic language that can be used to describe any kind of content in a structure manner separated from its presentation to a specified device



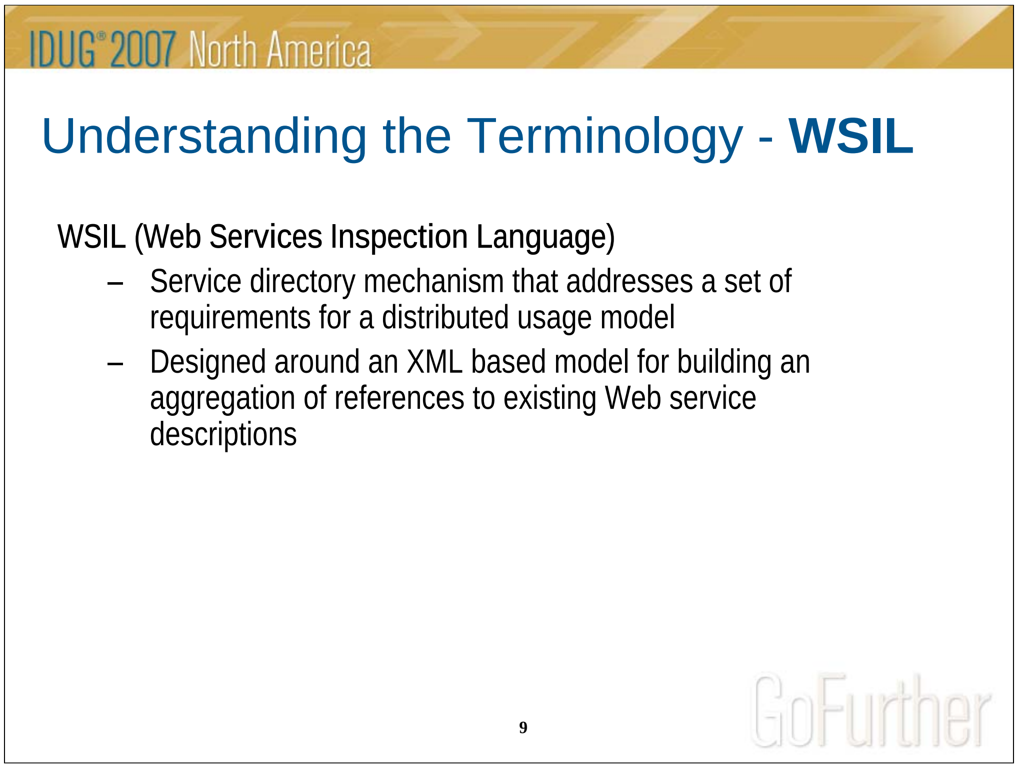
IDUG® 2007 North America

Understanding the Terminology - **WSDL**

- **WSDL (Web Services Description Language)**
  - Has an XML based interface and is an implementation description language
  - Service provider uses a WSDL document to specify
    - The operations a Web service provides
    - Parameters and data types for these operations
  - A WSDL document defines Web Services interfaces, including:
    - Operation types (one-way, request-response, notification)
    - Messages defining a Web Service
    - Data types (XML Schema)
- Web Service access protocol (SOAP over HTTP etc)
- Web Services contact end points (eg. Web Service URL)

8

GoFurther

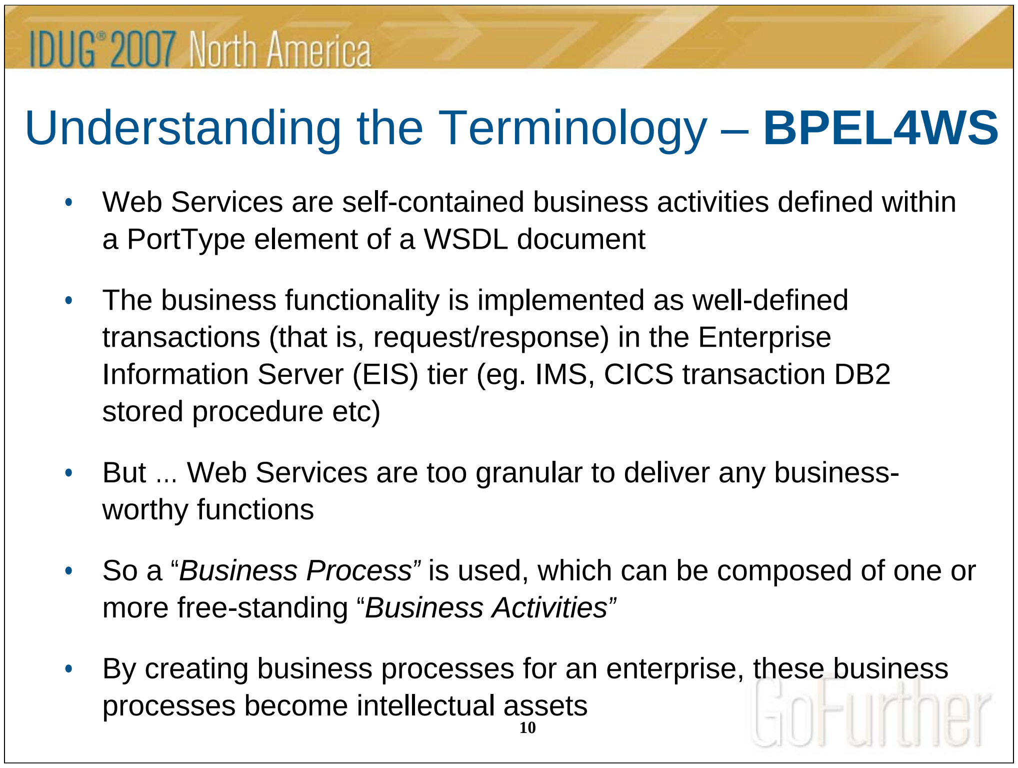


Understanding the Terminology - **WSIL**

WSIL (Web Services Inspection Language)

- Service directory mechanism that addresses a set of requirements for a distributed usage model
- Designed around an XML based model for building an aggregation of references to existing Web service descriptions

GoFurther

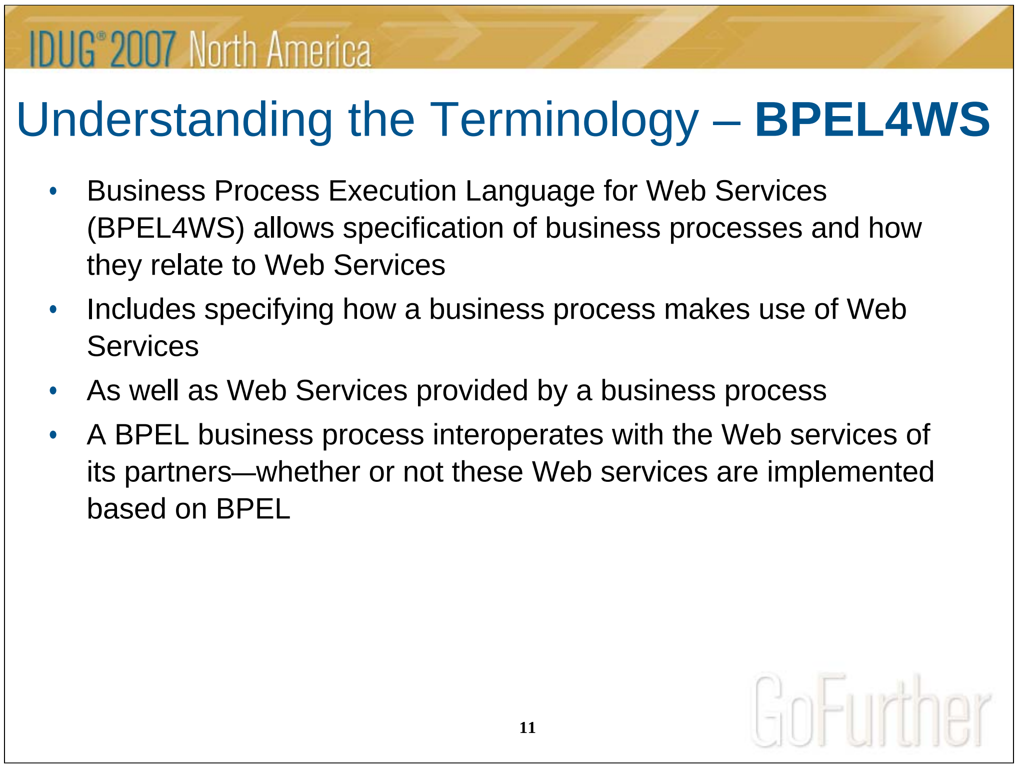

IDUG® 2007 North America

Understanding the Terminology – **BPEL4WS**

- Web Services are self-contained business activities defined within a PortType element of a WSDL document
- The business functionality is implemented as well-defined transactions (that is, request/response) in the Enterprise Information Server (EIS) tier (eg. IMS, CICS transaction DB2 stored procedure etc)
- But ... Web Services are too granular to deliver any business-worthy functions
- So a “*Business Process”* is used, which can be composed of one or more free-standing “*Business Activities*”
- By creating business processes for an enterprise, these business processes become intellectual assets

GoFurther

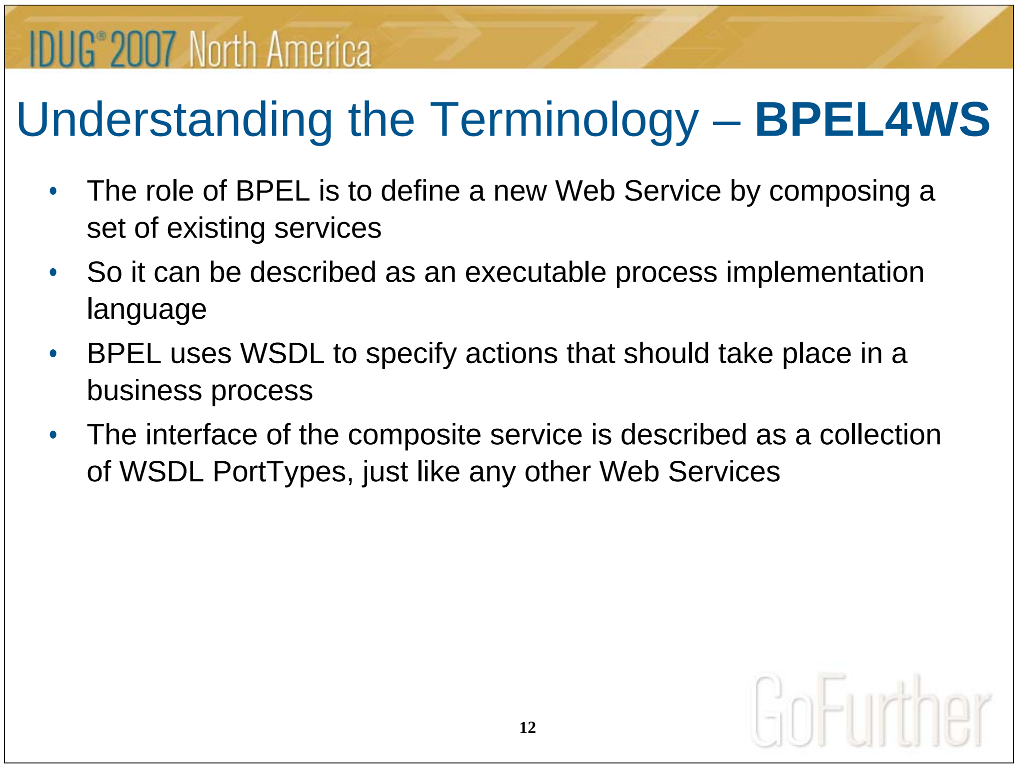
10



Understanding the Terminology – **BPEL4WS**

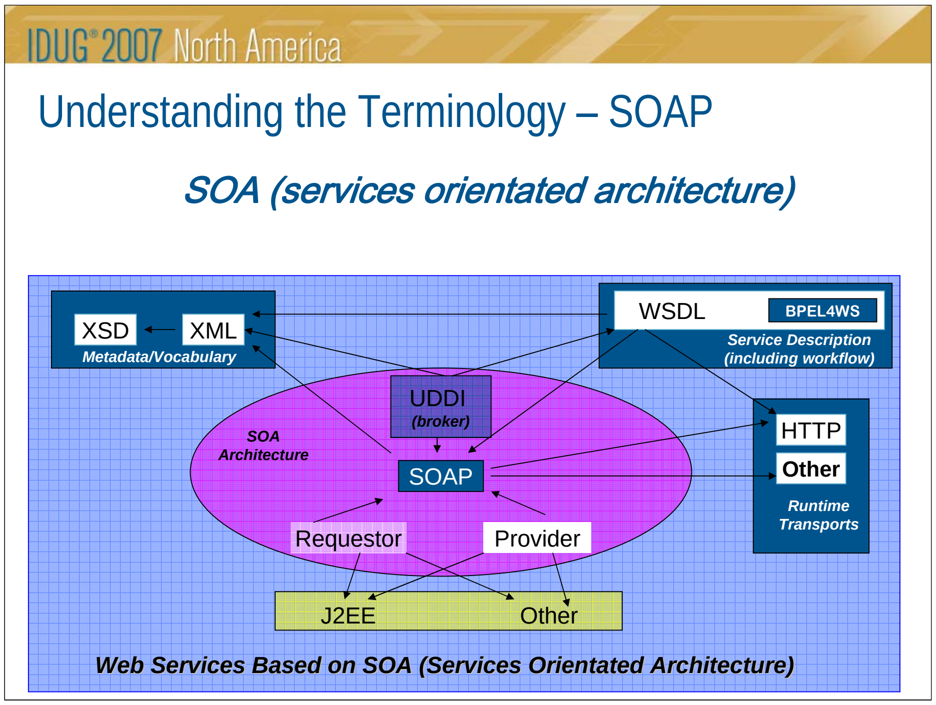
- Business Process Execution Language for Web Services (BPEL4WS) allows specification of business processes and how they relate to Web Services
- Includes specifying how a business process makes use of Web Services
- As well as Web Services provided by a business process
- A BPEL business process interoperates with the Web services of its partners—whether or not these Web services are implemented based on BPEL

GoFurther



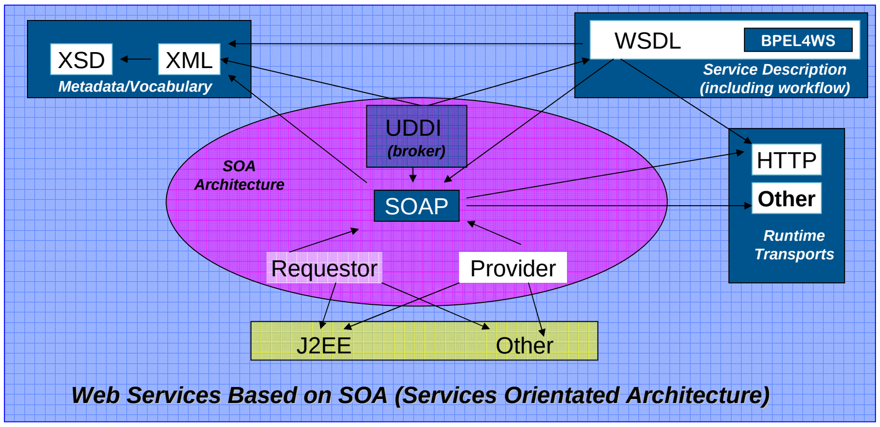
Understanding the Terminology – **BPEL4WS**

- The role of BPEL is to define a new Web Service by composing a set of existing services
- So it can be described as an executable process implementation language
- BPEL uses WSDL to specify actions that should take place in a business process
- The interface of the composite service is described as a collection of WSDL PortTypes, just like any other Web Services


IDUG® 2007 North America

Understanding the Terminology – SOAP

SOA (services orientated architecture)

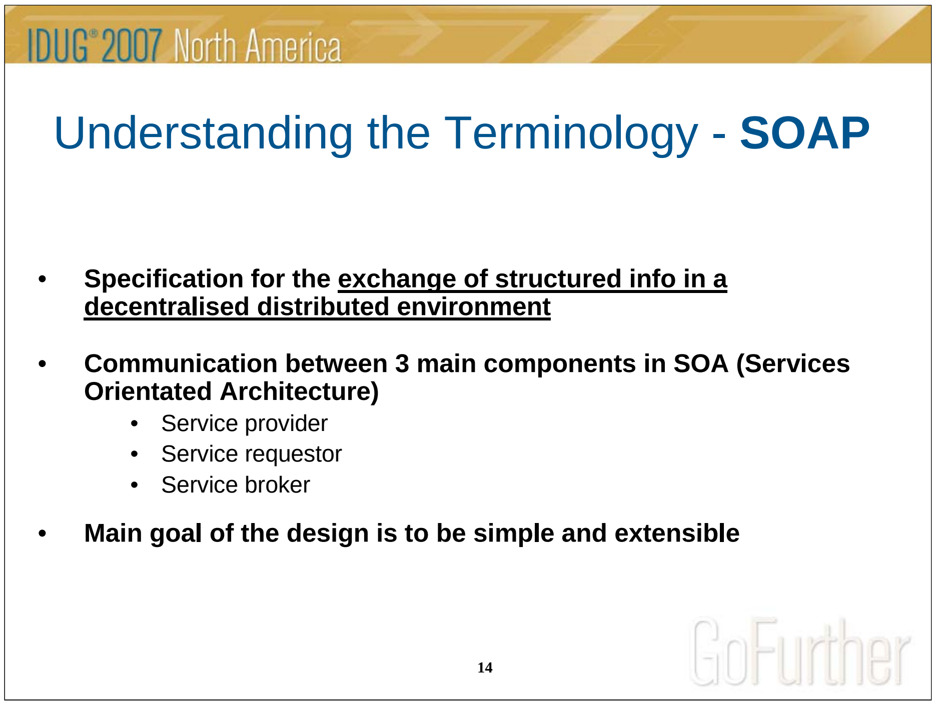


XSD

- XSD (XML Schema Definition), a Recommendation of the World Wide Web Consortium (W3C), specifies how to formally describe the elements in an Extensible Markup Language (XML) document. This description can be used to verify that each item of content in a document adheres to the description of the element in which the content is to be placed.

In general, a schema is an abstract representation of an object's characteristics and relationship to other objects. An XML schema represents the interrelationship between the attributes and elements of an XML object (for example, a document or a portion of a document). To create a schema for a document, you analyze its structure, defining each structural element as you encounter it. For example, within a schema for a document describing a Web site, you would define a Web site element, a Web page element, and other elements that describe possible content divisions within any page on that site. Just as in XML and HTML, elements are defined within a set of tags.

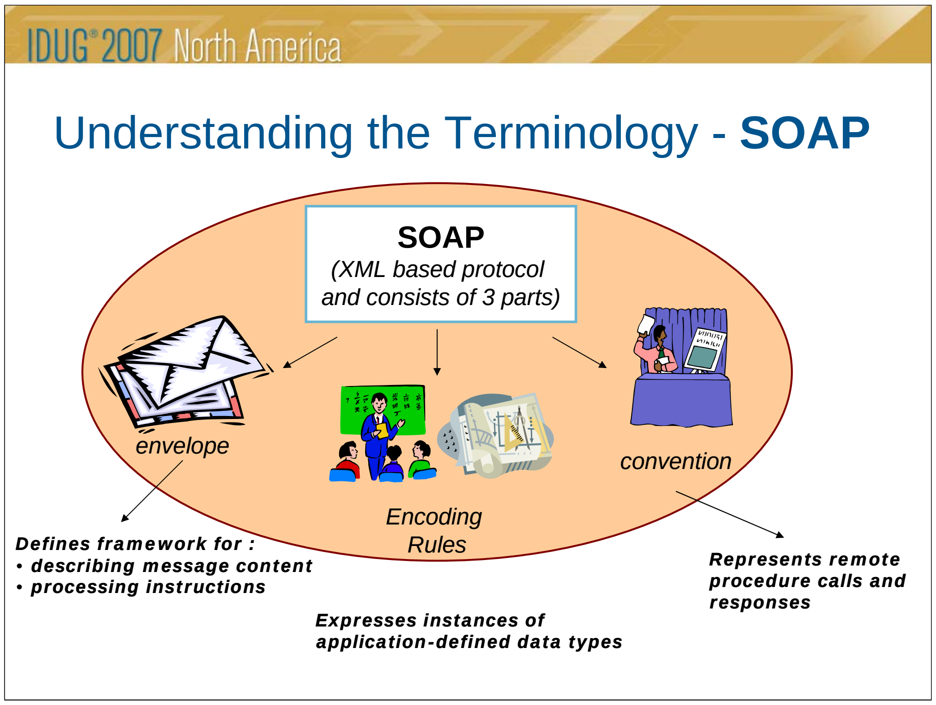
XSD has several advantages over earlier XML schema languages, such as document type definition (DTD) or Simple Object XML (SOX). For example, it's more direct: XSD, in contrast to the earlier languages, is written in XML, which means that it doesn't require intermediary processing by a parser. Other benefits include self-documentation, automatic schema creation, and the ability to be queried through XML Transformations (XSLT). Despite the advantages of XSD, it has some detractors who claim, for example, that the language is unnecessarily complex.



Understanding the Terminology - **SOAP**

- **Specification for the <u>exchange of structured info in a decentralised distributed environment</u>**
- **Communication between 3 main components in SOA (Services Orientated Architecture)**
  - Service provider
  - Service requestor
  - Service broker
- **Main goal of the design is to be simple and extensible**

GoFurther

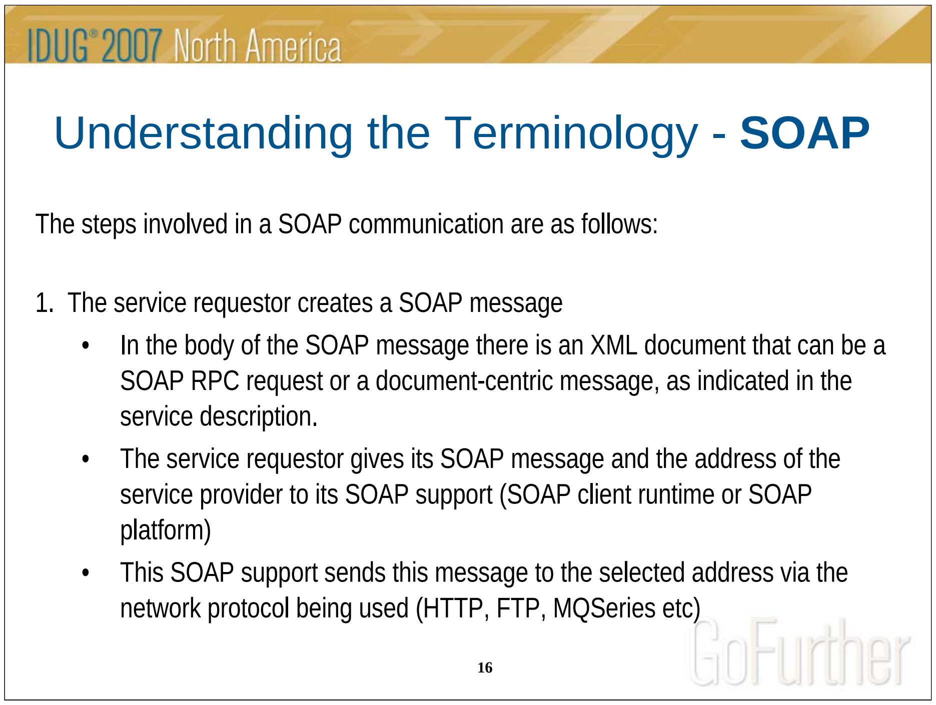


Understanding the Terminology - **SOAP**

Defines framework for :
- ***describing message content***
- ***processing instructions***

Expresses instances of application-defined data types

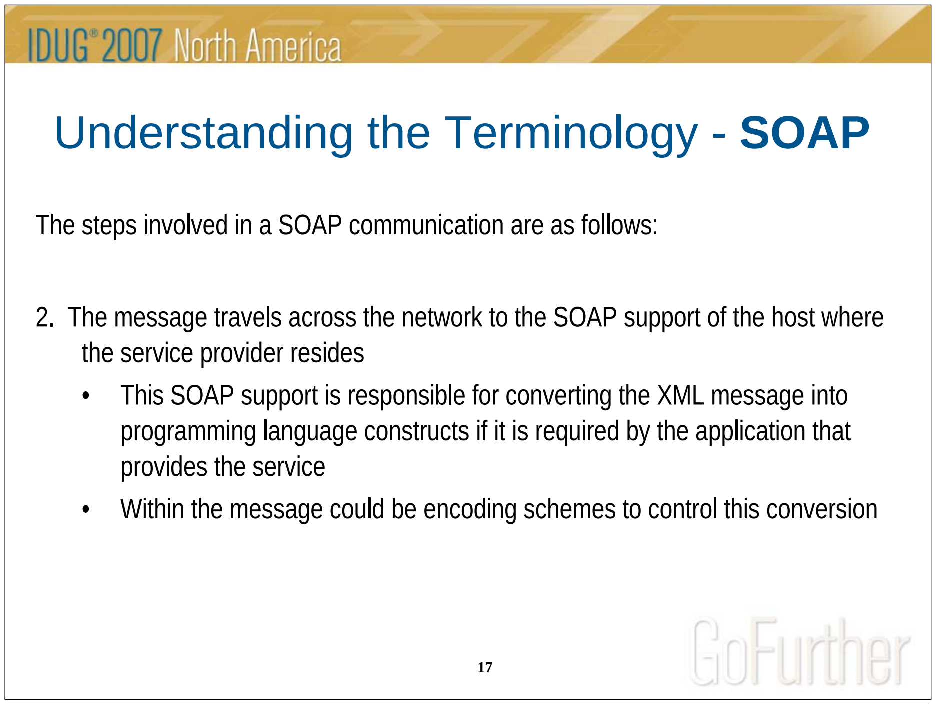
Represents remote procedure calls and responses



Understanding the Terminology - **SOAP**

The steps involved in a SOAP communication are as follows:

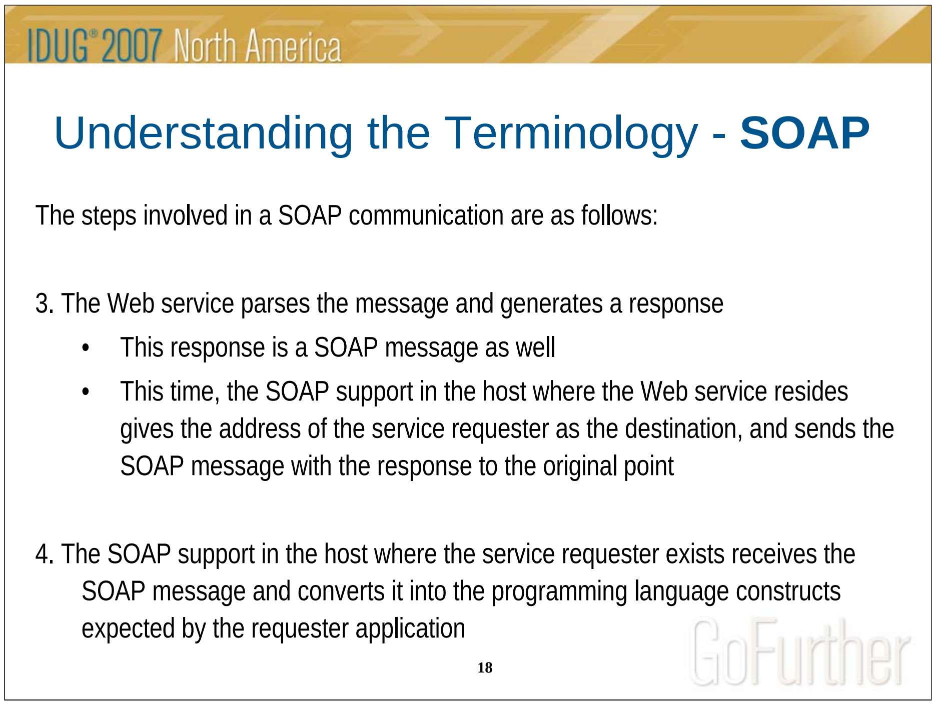
1. The service requestor creates a SOAP message
   - In the body of the SOAP message there is an XML document that can be a SOAP RPC request or a document-centric message, as indicated in the service description.
   - The service requestor gives its SOAP message and the address of the service provider to its SOAP support (SOAP client runtime or SOAP platform)
   - This SOAP support sends this message to the selected address via the network protocol being used (HTTP, FTP, MQSeries etc)



Understanding the Terminology - **SOAP**

The steps involved in a SOAP communication are as follows:

2. The message travels across the network to the SOAP support of the host where the service provider resides
   - This SOAP support is responsible for converting the XML message into programming language constructs if it is required by the application that provides the service
   - Within the message could be encoding schemes to control this conversion

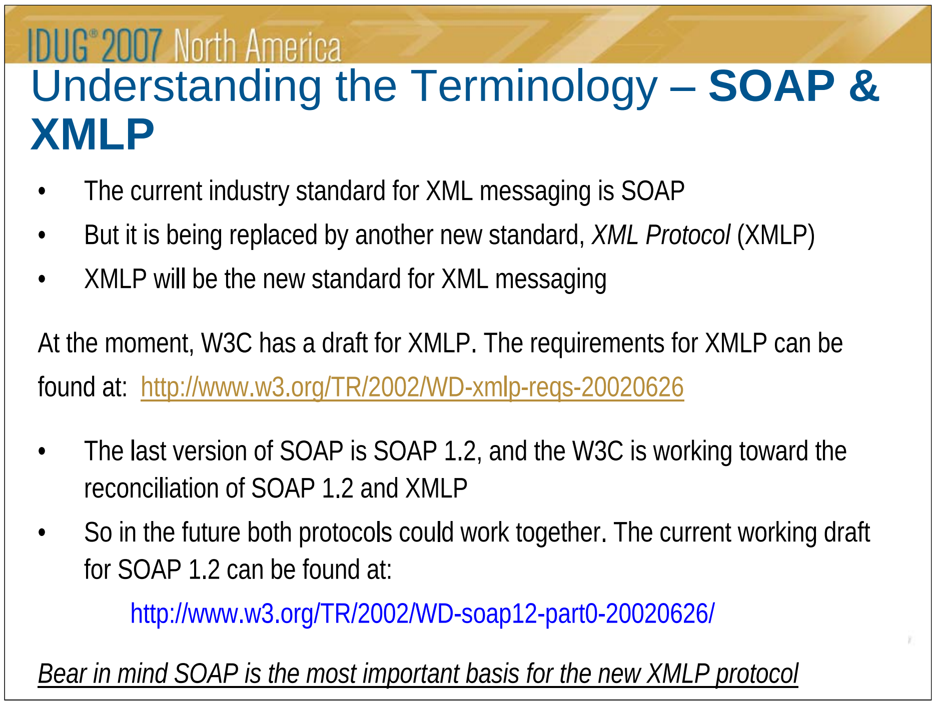


Understanding the Terminology - **SOAP**

The steps involved in a SOAP communication are as follows:

3. The Web service parses the message and generates a response
   - This response is a SOAP message as well
   - This time, the SOAP support in the host where the Web service resides gives the address of the service requester as the destination, and sends the SOAP message with the response to the original point

4. The SOAP support in the host where the service requester exists receives the SOAP message and converts it into the programming language constructs expected by the requester application



IDUG® 2007 North America

Understanding the Terminology – **SOAP & XMLP**

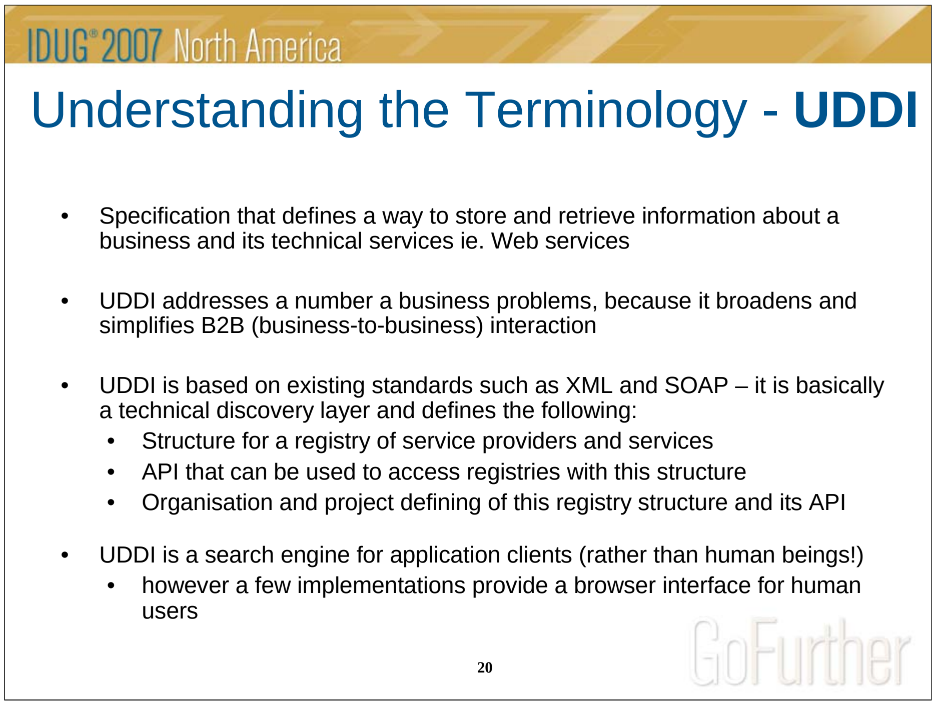
- The current industry standard for XML messaging is SOAP
- But it is being replaced by another new standard, *XML Protocol* (XMLP)
- XMLP will be the new standard for XML messaging

At the moment, W3C has a draft for XMLP. The requirements for XMLP can be found at: http://www.w3.org/TR/2002/WD-xmlp-reqs-20020626

- The last version of SOAP is SOAP 1.2, and the W3C is working toward the reconciliation of SOAP 1.2 and XMLP
- So in the future both protocols could work together. The current working draft for SOAP 1.2 can be found at:

  http://www.w3.org/TR/2002/WD-soap12-part0-20020626/

Bear in mind SOAP is the most important basis for the new XMLP protocol


IDUG® 2007 North America

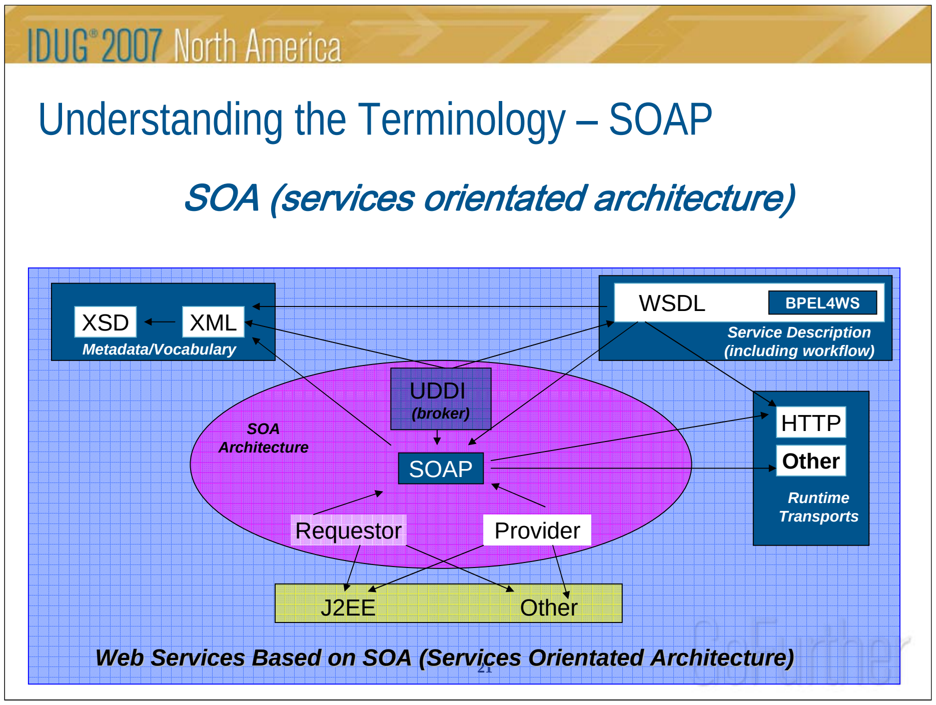
Understanding the Terminology - **UDDI**

- Specification that defines a way to store and retrieve information about a business and its technical services ie. Web services
- UDDI addresses a number a business problems, because it broadens and simplifies B2B (business-to-business) interaction
- UDDI is based on existing standards such as XML and SOAP – it is basically a technical discovery layer and defines the following:
  - Structure for a registry of service providers and services
  - API that can be used to access registries with this structure
  - Organisation and project defining of this registry structure and its API
- UDDI is a search engine for application clients (rather than human beings!)
  - however a few implementations provide a browser interface for human users

20

GoFurther

•Think of a car manufacturer who has to create many relationships with various suppliers, each with their own set of standards and protocols – UDDI provides a highly flexible description of services using almost any interface


IDUG® 2007 North America

Understanding the Terminology – SOAP

SOA (services orientated architecture)

XSD

- XSD (XML Schema Definition), a Recommendation of the World Wide Web Consortium (W3C), specifies how to formally describe the elements in an Extensible Markup Language (XML) document. This description can be used to verify that each item of content in a document adheres to the description of the element in which the content is to be placed.

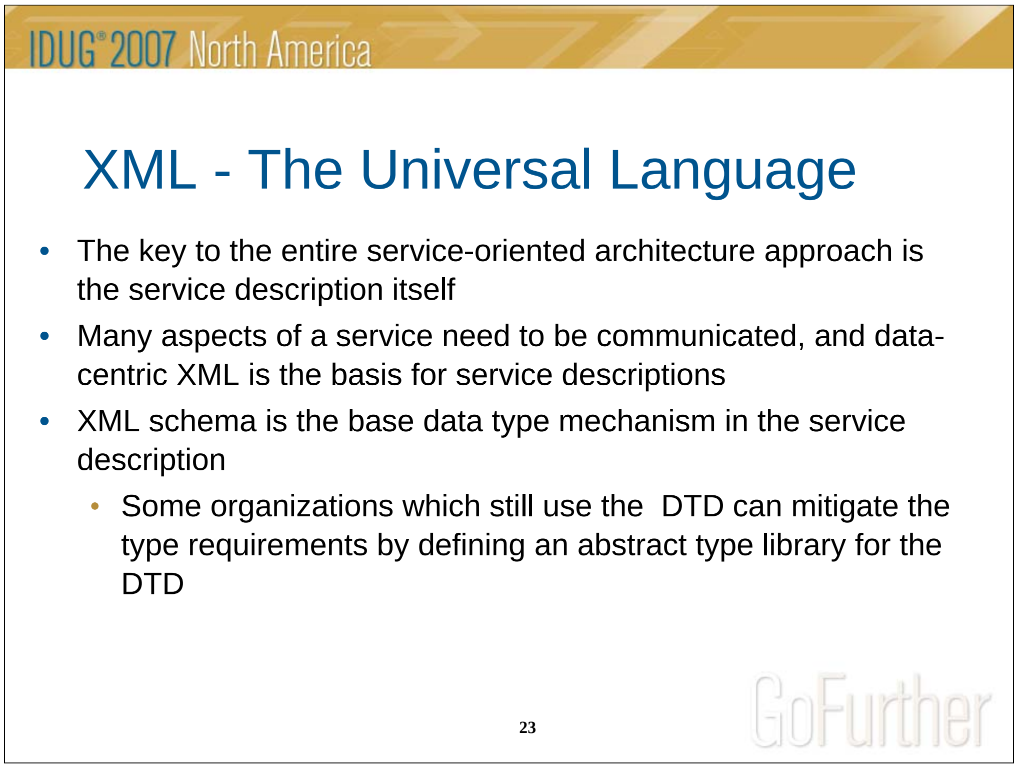
In general, a schema is an abstract representation of an object's characteristics and relationship to other objects. An XML schema represents the interrelationship between the attributes and elements of an XML object (for example, a document or a portion of a document). To create a schema for a document, you analyze its structure, defining each structural element as you encounter it. For example, within a schema for a document describing a Web site, you would define a Web site element, a Web page element, and other elements that describe possible content divisions within any page on that site. Just as in XML and HTML, elements are defined within a set of tags.

XSD has several advantages over earlier XML schema languages, such as document type definition (DTD) or Simple Object XML (SOX). For example, it's more direct: XSD, in contrast to the earlier languages, is written in XML, which means that it doesn't require intermediary processing by a parser. Other benefits include self-documentation, automatic schema creation, and the ability to be queried through XML Transformations (XSLT). Despite the advantages of XSD, it has some detractors who claim, for example, that the language is unnecessarily complex.

Agenda

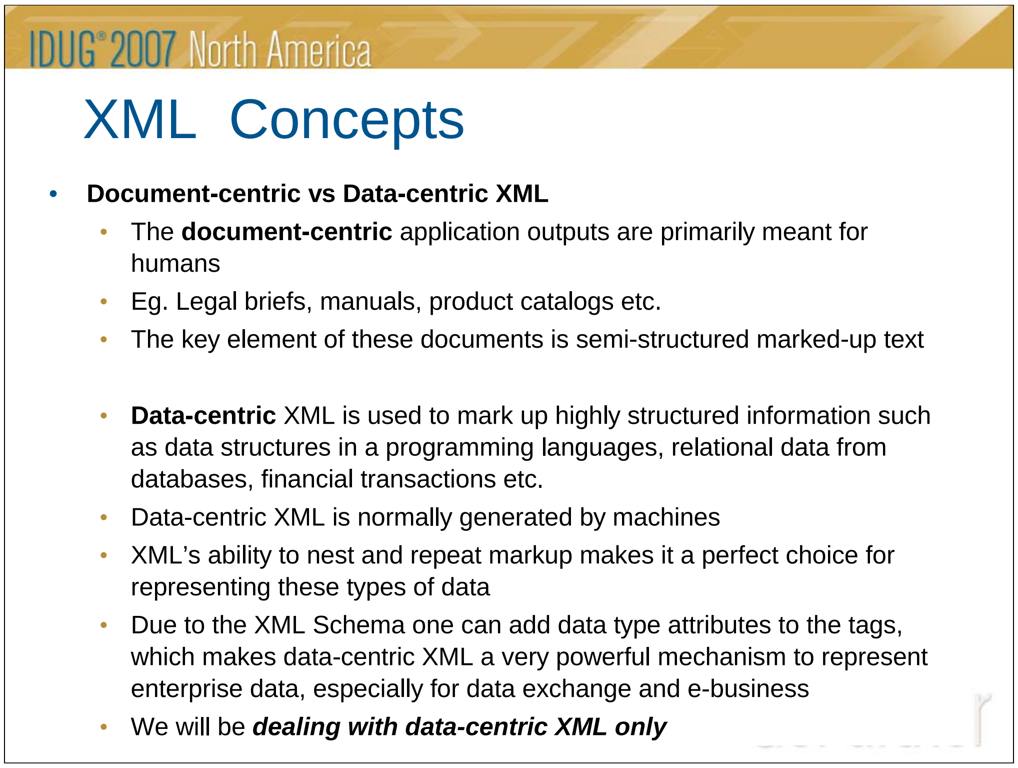
- Introduction to the Bigger Picture - The World of Web Services
- Understanding the Terminology
- **XML - The Universal Language**
- Web Services and Service Orientated Architecture (SOA)

GoFurther



XML - The Universal Language

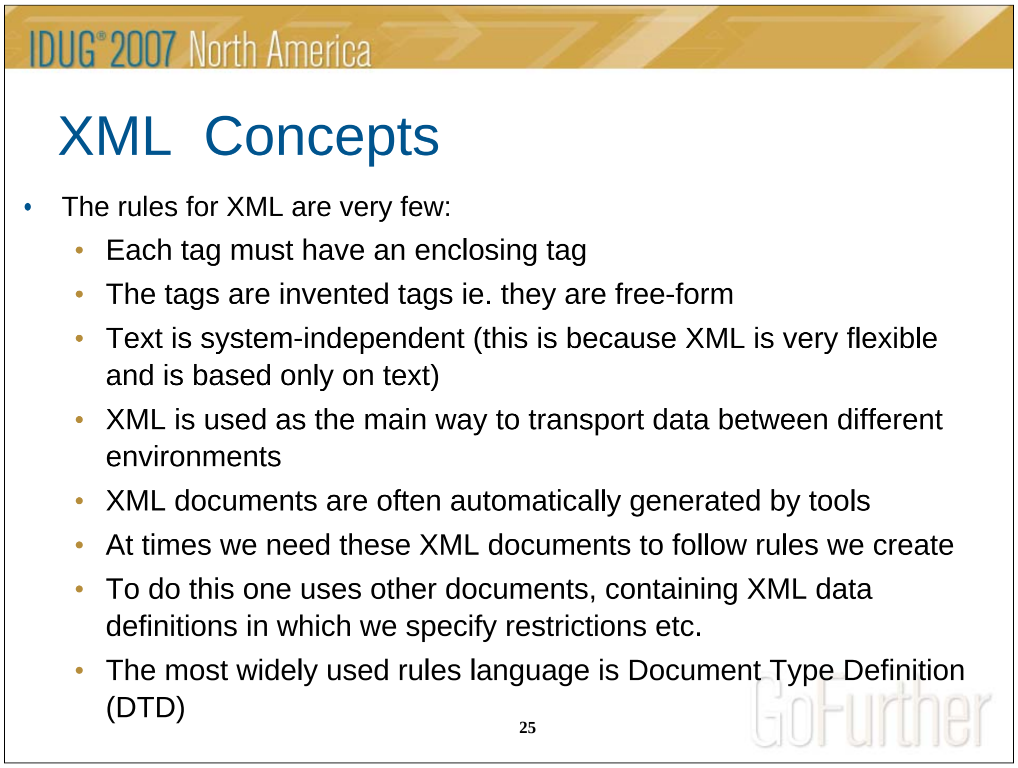
- The key to the entire service-oriented architecture approach is the service description itself
- Many aspects of a service need to be communicated, and data-centric XML is the basis for service descriptions
- XML schema is the base data type mechanism in the service description
  - Some organizations which still use the DTD can mitigate the type requirements by defining an abstract type library for the DTD



XML Concepts

- **Document-centric vs Data-centric XML**
  - The **document-centric** application outputs are primarily meant for humans
  - Eg. Legal briefs, manuals, product catalogs etc.
  - The key element of these documents is semi-structured marked-up text

  - **Data-centric** XML is used to mark up highly structured information such as data structures in a programming languages, relational data from databases, financial transactions etc.
  - Data-centric XML is normally generated by machines
  - XML's ability to nest and repeat markup makes it a perfect choice for representing these types of data
  - Due to the XML Schema one can add data type attributes to the tags, which makes data-centric XML a very powerful mechanism to represent enterprise data, especially for data exchange and e-business
  - We will be ***dealing with data-centric XML only***



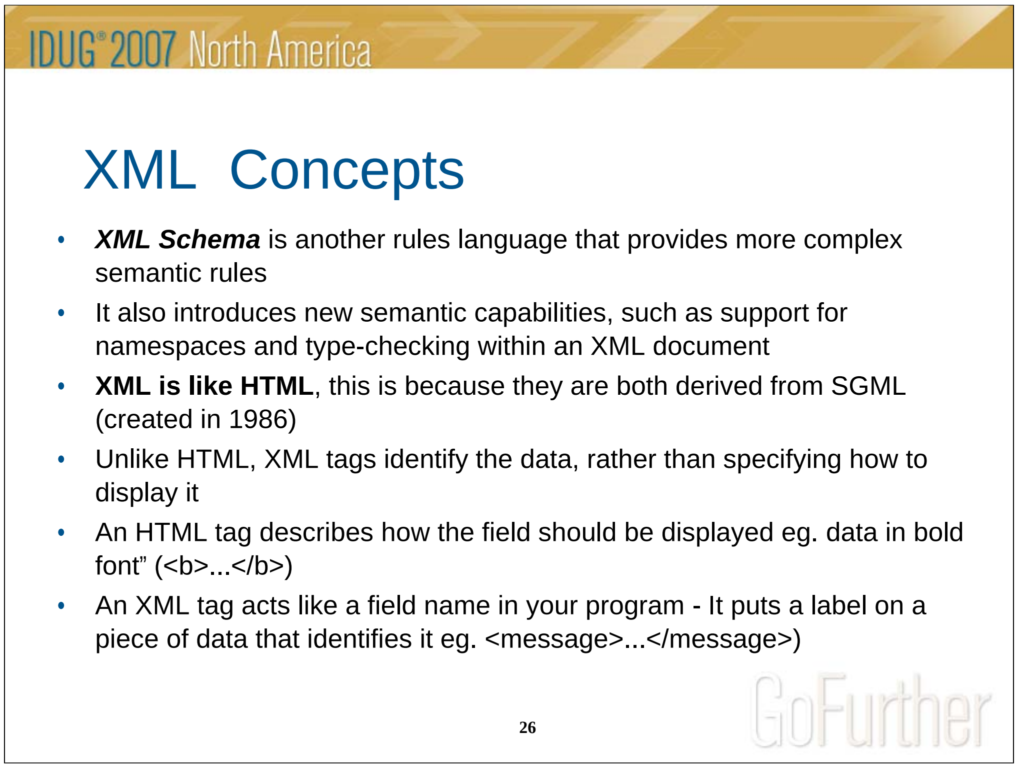
IDUG® 2007 North America

XML Concepts

- The rules for XML are very few:
  - Each tag must have an enclosing tag
  - The tags are invented tags ie. they are free-form
  - Text is system-independent (this is because XML is very flexible and is based only on text)
  - XML is used as the main way to transport data between different environments
  - XML documents are often automatically generated by tools
  - At times we need these XML documents to follow rules we create
  - To do this one uses other documents, containing XML data definitions in which we specify restrictions etc.
  - The most widely used rules language is Document Type Definition (DTD)

25

GoFurther

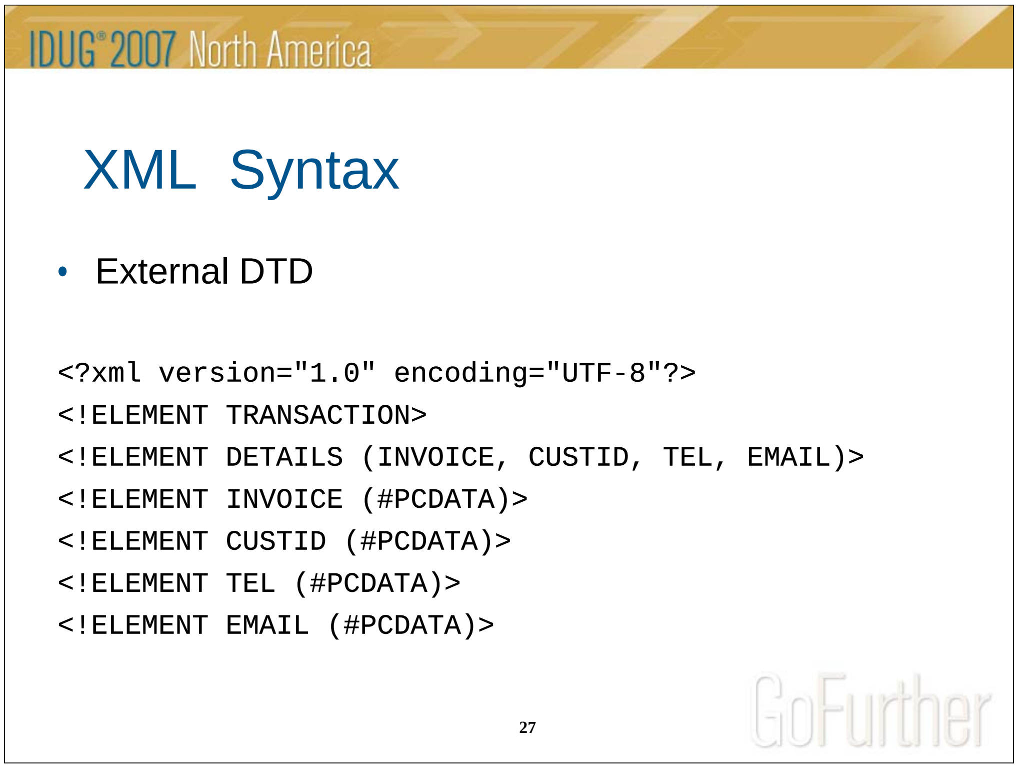

IDUG® 2007 North America

XML Concepts

- ***XML Schema*** is another rules language that provides more complex semantic rules
- It also introduces new semantic capabilities, such as support for namespaces and type-checking within an XML document
- **XML is like HTML**, this is because they are both derived from SGML (created in 1986)
- Unlike HTML, XML tags identify the data, rather than specifying how to display it
- An HTML tag describes how the field should be displayed eg. data in bold font” (<b>...</b>)
- An XML tag acts like a field name in your program - It puts a label on a piece of data that identifies it eg. <message>...</message>)

26

GoFurther



IDUG® 2007 North America

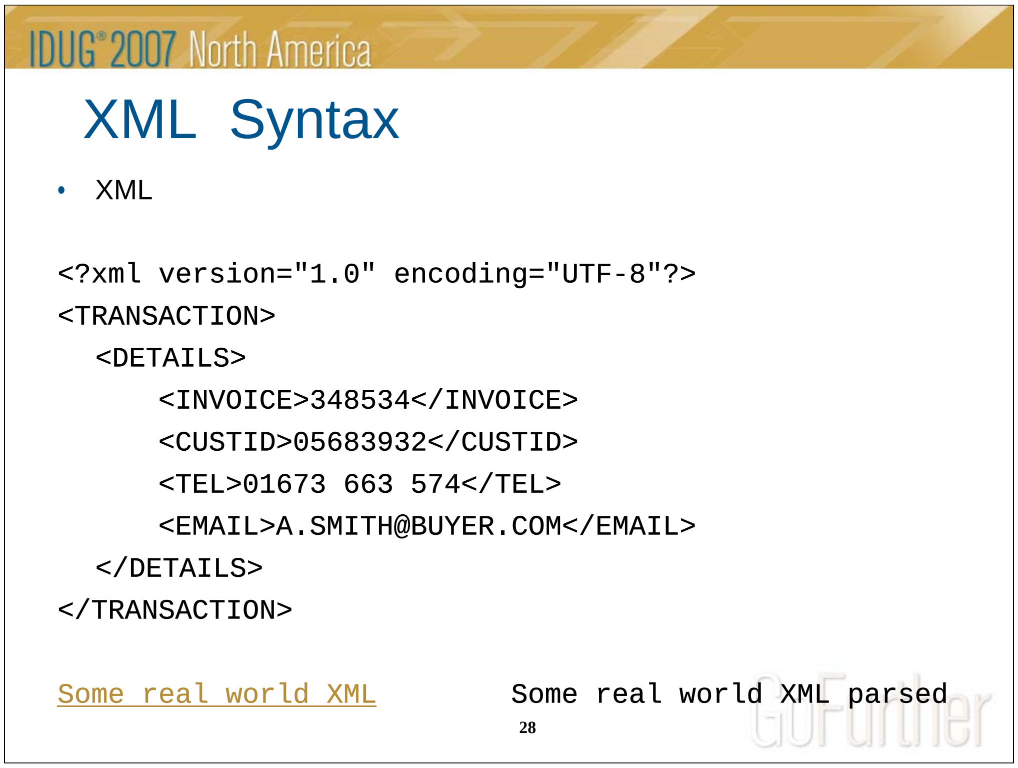
XML Syntax

- External DTD

```
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT TRANSACTION>
<!ELEMENT DETAILS (INVOICE, CUSTID, TEL, EMAIL)>
<!ELEMENT INVOICE (#PCDATA)>
<!ELEMENT CUSTID (#PCDATA)>
<!ELEMENT TEL (#PCDATA)>
<!ELEMENT EMAIL (#PCDATA)>
```

27

GoFurther



IDUG® 2007 North America

XML Syntax

- XML

```
<?xml version="1.0" encoding="UTF-8"?>
<TRANSACTION>
  <DETAILS>
      <INVOICE>348534</INVOICE>
      <CUSTID>05683932</CUSTID>
      <TEL>01673 663 574</TEL>
      <EMAIL>A.SMITH@BUYER.COM</EMAIL>
  </DETAILS>
</TRANSACTION>
```

Some real world XML   Some real world XML parsed

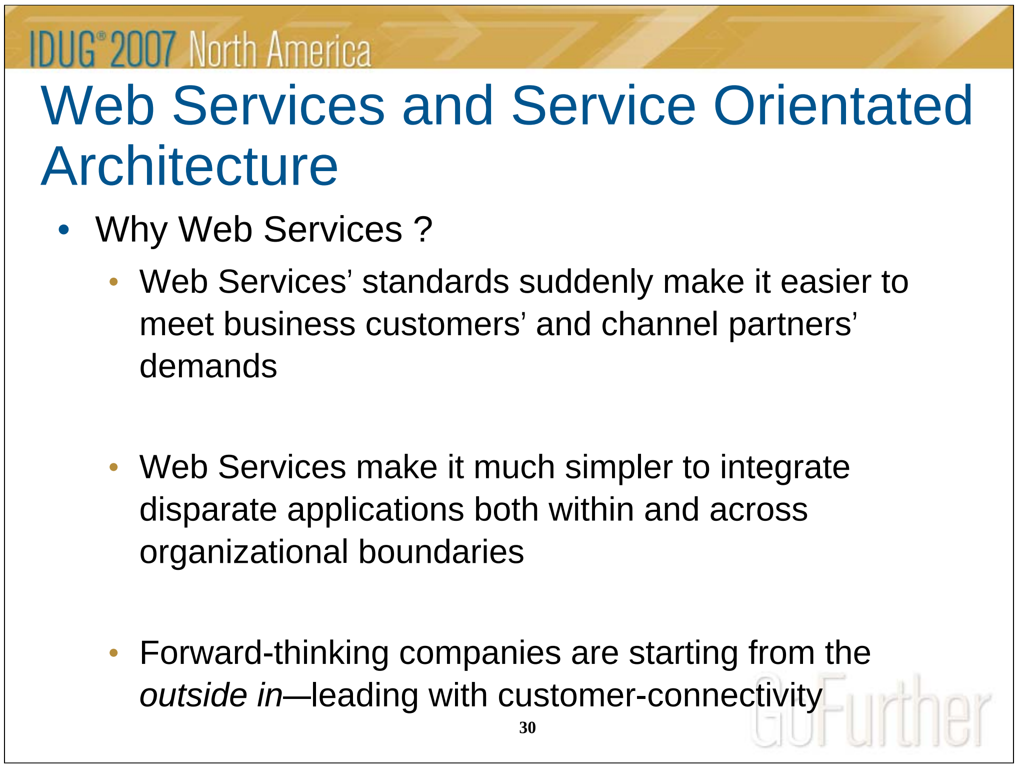
28

GoFurther

Agenda

- Introduction to the Bigger Picture - The World of Web Services
- Understanding the Terminology
- XML - The Universal Language
- **Web Services and Service Orientated Architecture (SOA)**

GoFurther



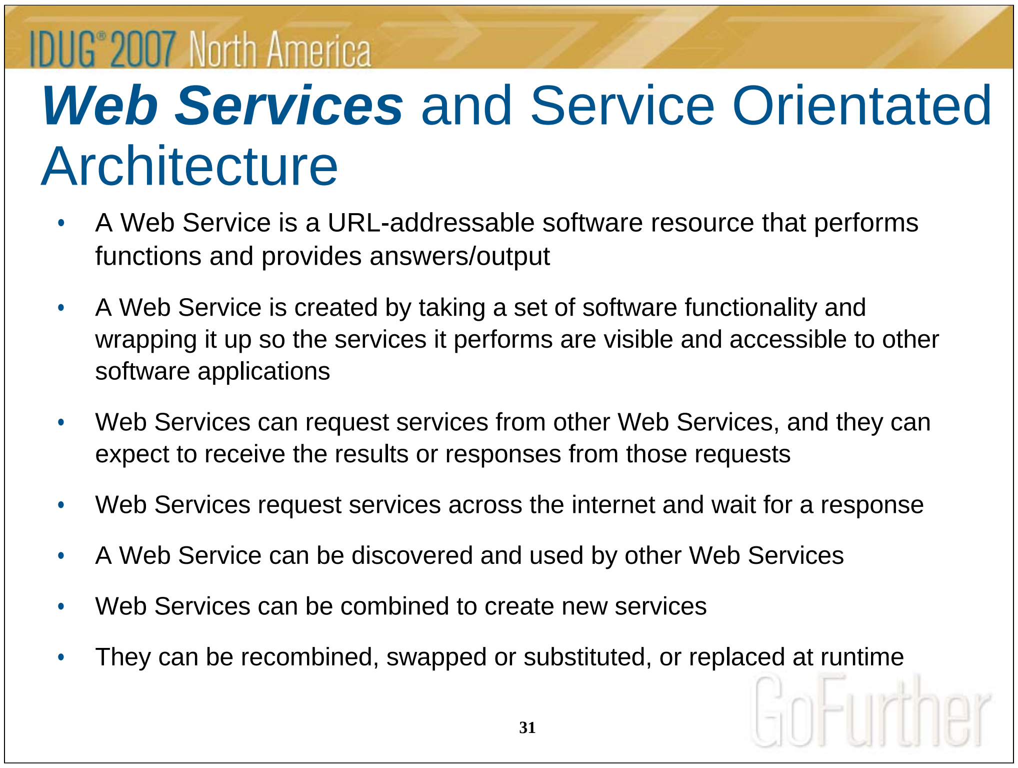
IDUG® 2007 North America

Web Services and Service Orientated Architecture

- Why Web Services ?
  - Web Services' standards suddenly make it easier to meet business customers' and channel partners' demands
  - Web Services make it much simpler to integrate disparate applications both within and across organizational boundaries
  - Forward-thinking companies are starting from the *outside in*—leading with customer-connectivity

GoFurther

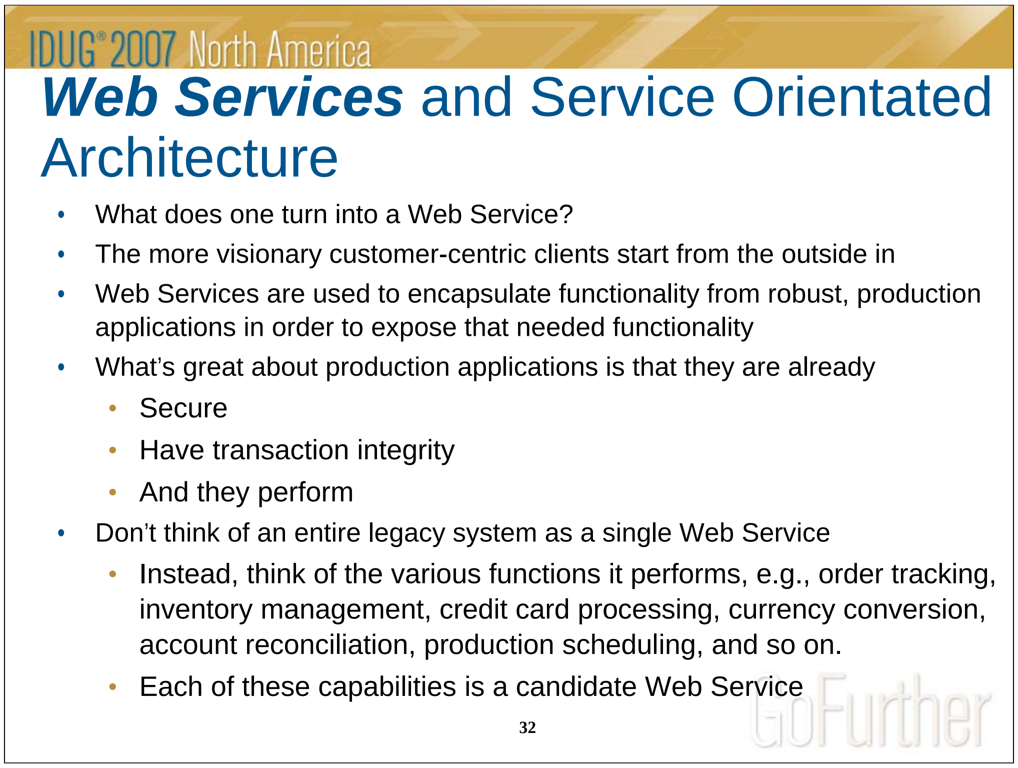
30



Web Services and Service Orientated Architecture

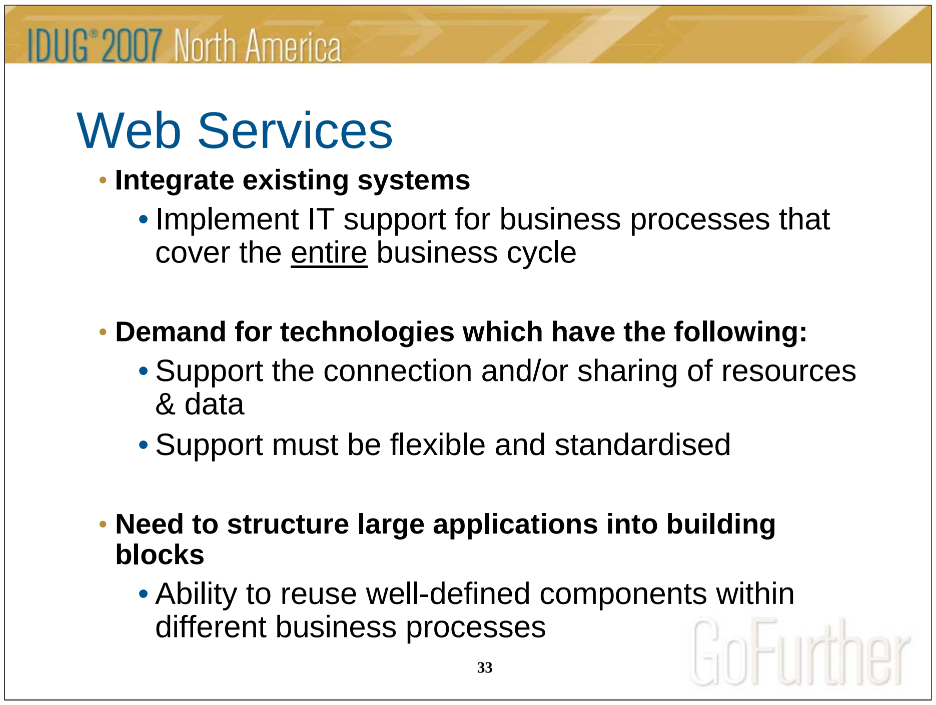
- A Web Service is a URL-addressable software resource that performs functions and provides answers/output
- A Web Service is created by taking a set of software functionality and wrapping it up so the services it performs are visible and accessible to other software applications
- Web Services can request services from other Web Services, and they can expect to receive the results or responses from those requests
- Web Services request services across the internet and wait for a response
- A Web Service can be discovered and used by other Web Services
- Web Services can be combined to create new services
- They can be recombined, swapped or substituted, or replaced at runtime

GoFurther



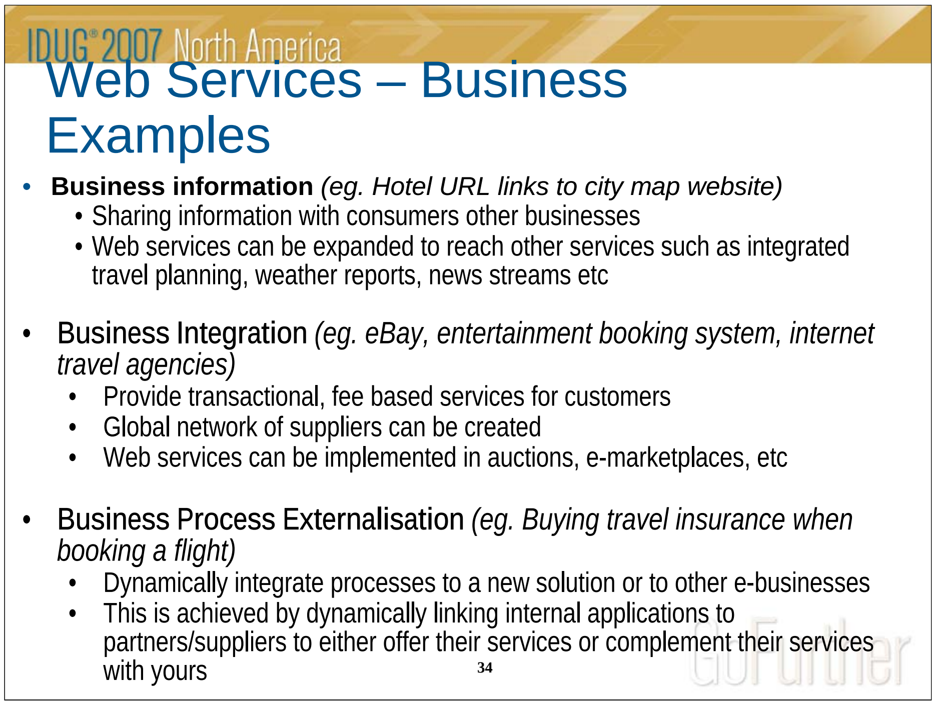
Web Services and Service Orientated Architecture

- What does one turn into a Web Service?
- The more visionary customer-centric clients start from the outside in
- Web Services are used to encapsulate functionality from robust, production applications in order to expose that needed functionality
- What's great about production applications is that they are already
  - Secure
  - Have transaction integrity
  - And they perform
- Don't think of an entire legacy system as a single Web Service
  - Instead, think of the various functions it performs, e.g., order tracking, inventory management, credit card processing, currency conversion, account reconciliation, production scheduling, and so on.
  - Each of these capabilities is a candidate Web Service



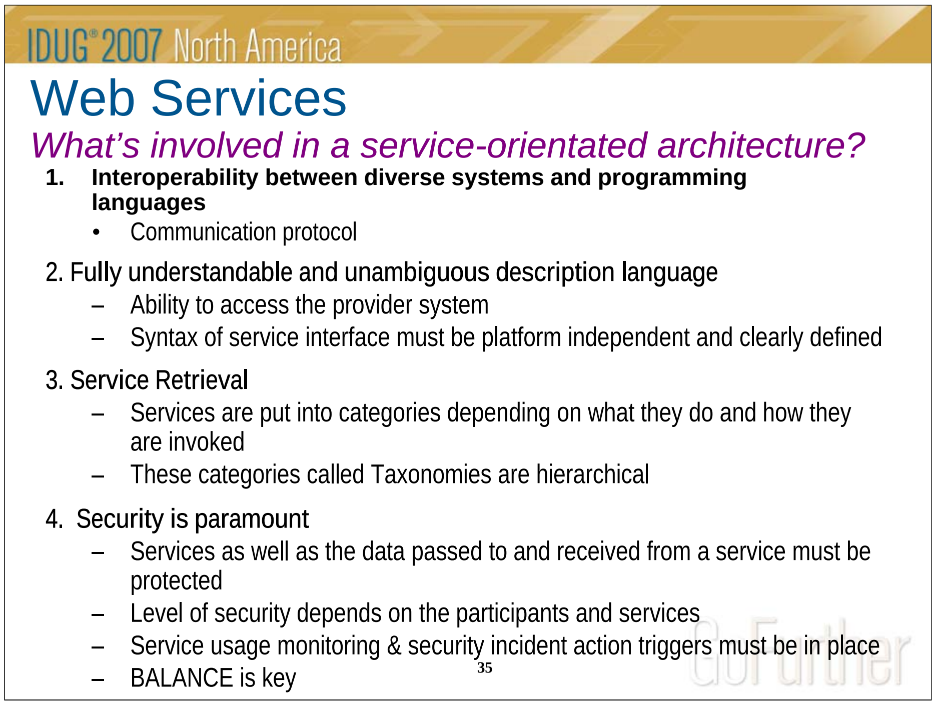
Web Services

- **Integrate existing systems**
  - Implement IT support for business processes that cover the entire business cycle
- **Demand for technologies which have the following:**
  - Support the connection and/or sharing of resources & data
  - Support must be flexible and standardised
- **Need to structure large applications into building blocks**
  - Ability to reuse well-defined components within different business processes



Web Services – Business Examples

- **Business information** *(eg. Hotel URL links to city map website)*
  - Sharing information with consumers other businesses
  - Web services can be expanded to reach other services such as integrated travel planning, weather reports, news streams etc
- Business Integration *(eg. eBay, entertainment booking system, internet travel agencies)*
  - Provide transactional, fee based services for customers
  - Global network of suppliers can be created
  - Web services can be implemented in auctions, e-marketplaces, etc
- Business Process Externalisation *(eg. Buying travel insurance when booking a flight)*
  - Dynamically integrate processes to a new solution or to other e-businesses
  - This is achieved by dynamically linking internal applications to partners/suppliers to either offer their services or complement their services with yours



Web Services

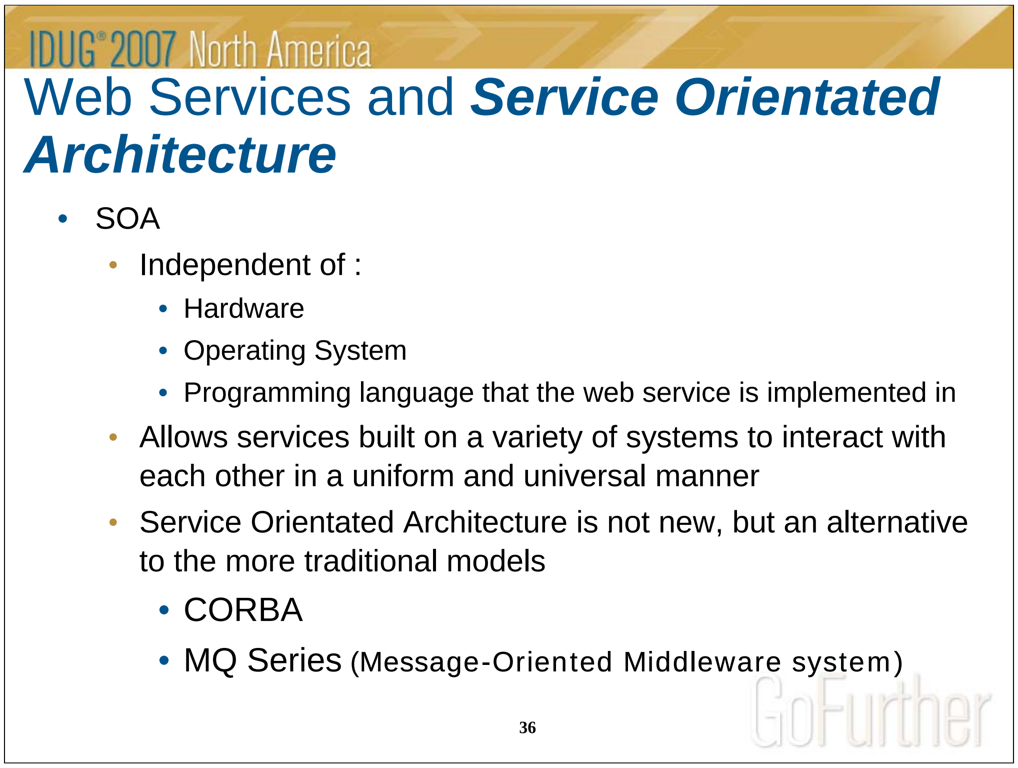
What's involved in a service-orientated architecture?

1. **Interoperability between diverse systems and programming languages**
   - Communication protocol

2. Fully understandable and unambiguous description language
   - Ability to access the provider system
   - Syntax of service interface must be platform independent and clearly defined

3. Service Retrieval
   - Services are put into categories depending on what they do and how they are invoked
   - These categories called Taxonomies are hierarchical

4. Security is paramount
   - Services as well as the data passed to and received from a service must be protected
   - Level of security depends on the participants and services
   - Service usage monitoring & security incident action triggers must be in place
   - BALANCE is key



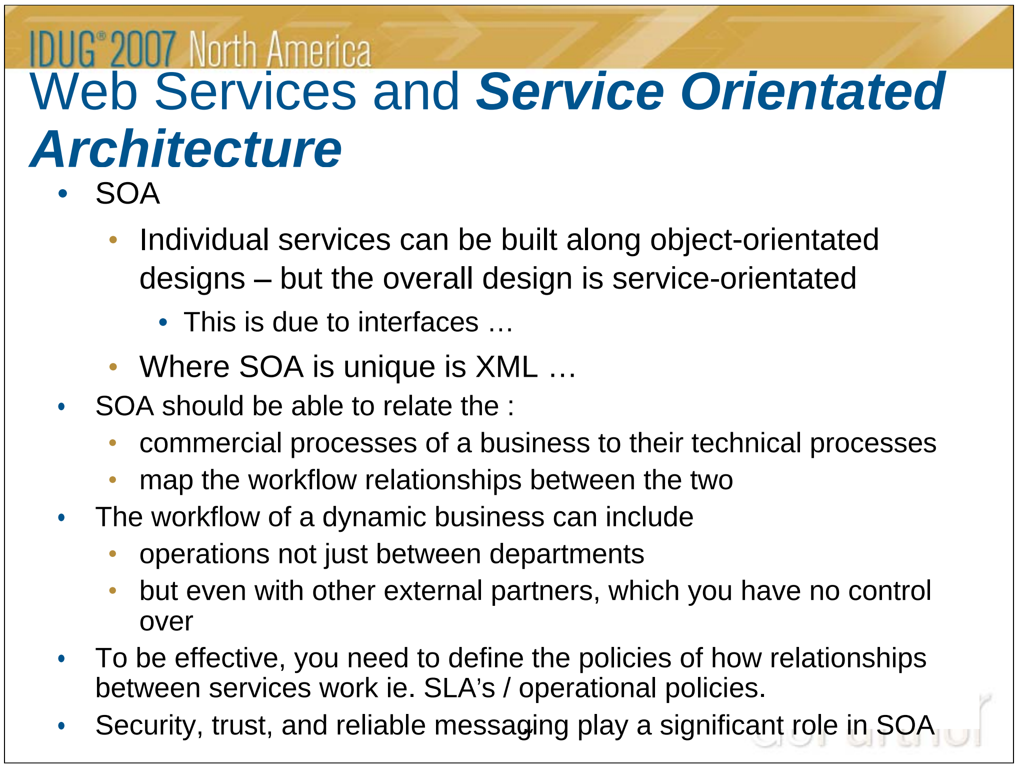
IDUG® 2007 North America

Web Services and ***Service Orientated Architecture***

- SOA
  - Independent of :
    - Hardware
    - Operating System
    - Programming language that the web service is implemented in
  - Allows services built on a variety of systems to interact with each other in a uniform and universal manner
  - Service Orientated Architecture is not new, but an alternative to the more traditional models
    - CORBA
    - MQ Series (Message-Oriented Middleware system)

36

GoFurther


IDUG® 2007 North America

Web Services and ***Service Orientated Architecture***

- SOA
  - Individual services can be built along object-orientated designs – but the overall design is service-orientated
    - This is due to interfaces …
  - Where SOA is unique is XML …
- SOA should be able to relate the :
  - commercial processes of a business to their technical processes
  - map the workflow relationships between the two
- The workflow of a dynamic business can include
  - operations not just between departments
  - but even with other external partners, which you have no control over
- To be effective, you need to define the policies of how relationships between services work ie. SLA's / operational policies.
- Security, trust, and reliable messaging play a significant role in SOA

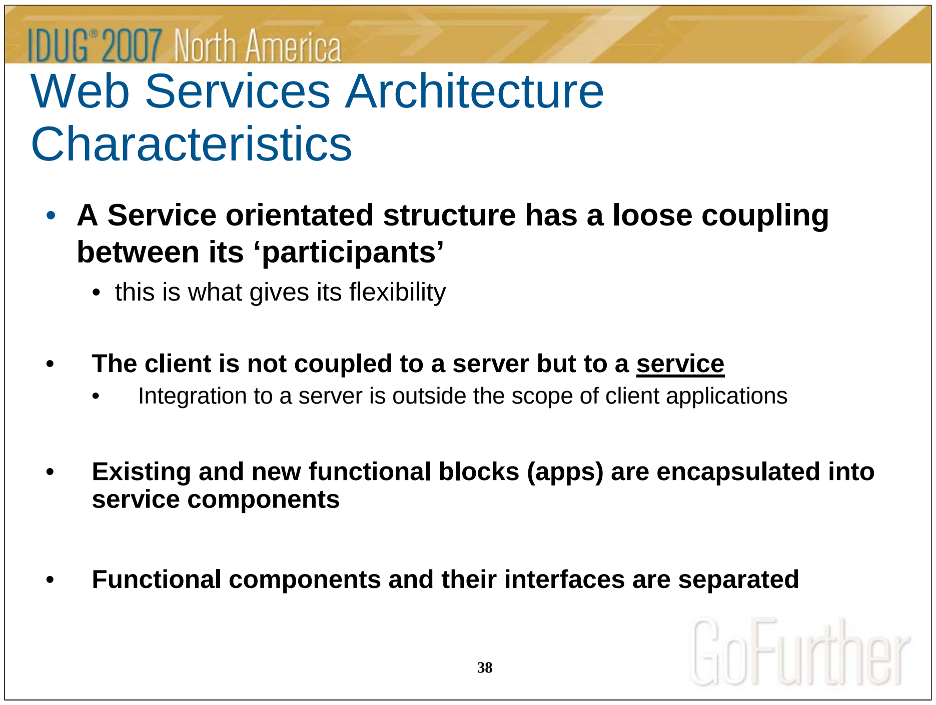
Service-oriented architecture is not new, but is an alternative model to the more traditionally tightly-coupled object-oriented models that have emerged in the past decades. While SOA-based systems do not exclude the fact that individual services can themselves be built with object-oriented designs, the overall design is service-oriented. Since it allows for objects within the system, SOA is *object-based*, but it is not, as a whole, *object-oriented*. The difference lies in the interfaces themselves. A classic example of a proto-SOA system that has been around for a while is the Common Object Request Broker Architecture (CORBA), which defines similar concepts to SOA.

However, the SOA of today is different in that it relies on a more recent advance based upon the eXtensible Markup Language (XML). By describing interfaces in an XML-based language called *Web Services Definition Language* (WSDL), services have moved to a more dynamic and flexible interface system than the older Interface Definition Language (IDL) found in CORBA.

Web services aren't the only way to implement SOA. CORBA, as just explained earlier is one other way and so are Message-Oriented Middleware systems such as the IBM MQ Series. But to become an architecture model, you need more than just a service description. You need to define how the overall application performs its workflow between services. More so, you need to find the transformation point between the operations of the business versus the operations of the software used in the business. Thus, an SOA should be able to relate the commercial processes of a business to their technical processes, and map the workflow relationships between the two. For example, the act of paying a supplier is a business process, while updating your parts database to include the newly supplied shipment is a technical one. Thus workflow also plays a significant role in the design of SOA.

Furthermore, the workflow of a dynamic business can include operations not just between departments, but even with other external partners, which you have no control over. Thus to be effective, you need to define the policies of how relationships between services should transpire, often in the form of service-level agreements and operational policies.

Finally, all this has to work in an environment of trust and reliability to carry out the processes as expected according to agreed terms. So, security, trust, and reliable messaging should play a significant role in any SOA.



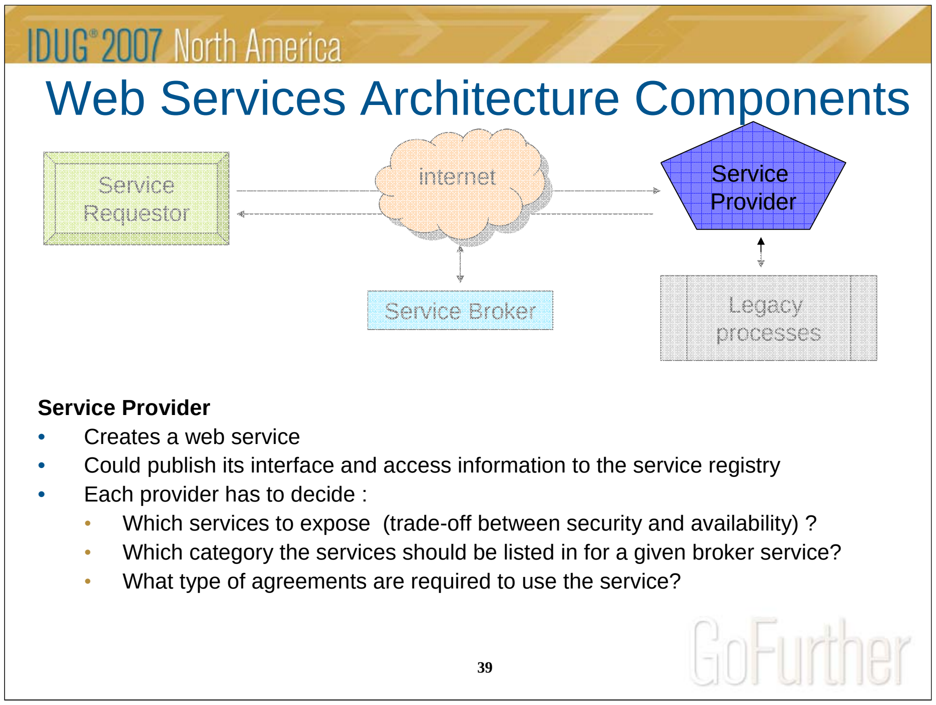
IDUG® 2007 North America

Web Services Architecture Characteristics

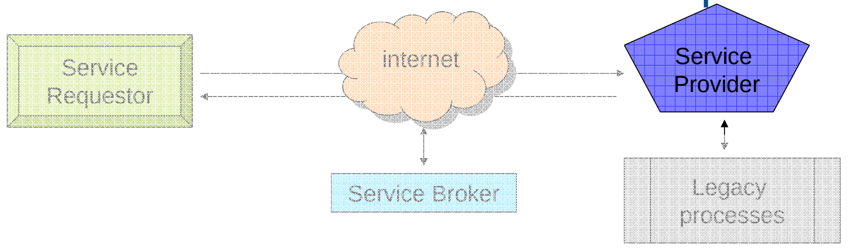
- **A Service orientated structure has a loose coupling between its 'participants'**
  - this is what gives its flexibility
- **The client is not coupled to a server but to a <u>service</u>**
  - Integration to a server is outside the scope of client applications
- **Existing and new functional blocks (apps) are encapsulated into service components**
- **Functional components and their interfaces are separated**

38

GoFurther



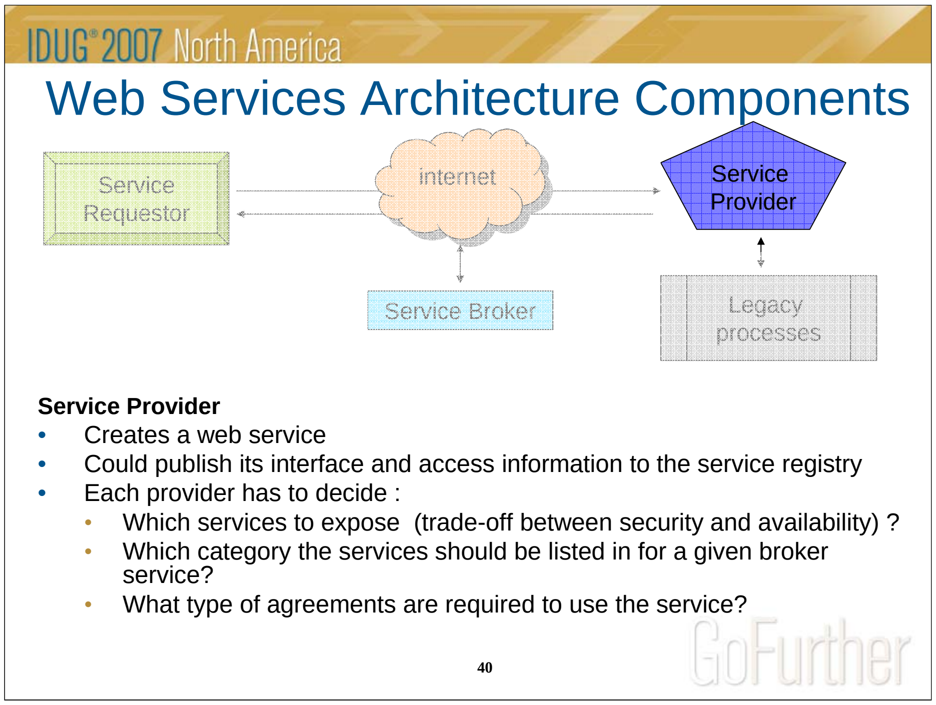
Web Services Architecture Components



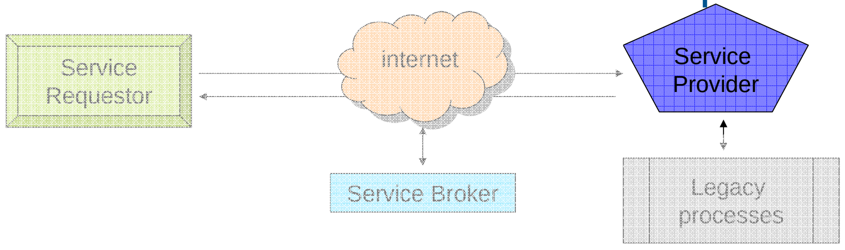
Service Provider

- Creates a web service
- Could publish its interface and access information to the service registry
- Each provider has to decide :
  - Which services to expose (trade-off between security and availability) ?
  - Which category the services should be listed in for a given broker service?
  - What type of agreements are required to use the service?

GoFurther



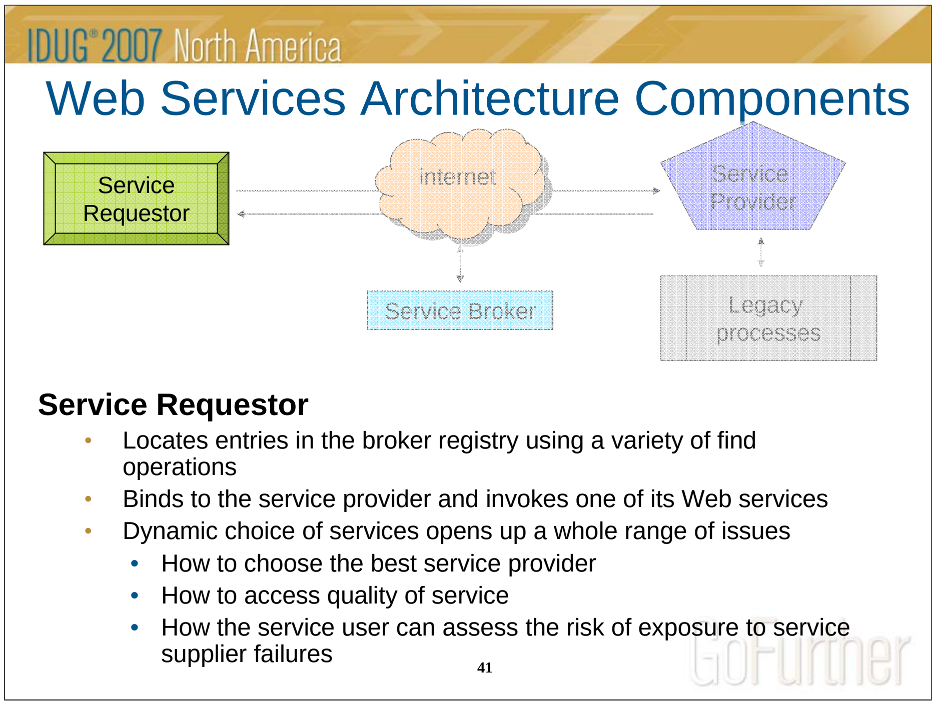
Web Services Architecture Components



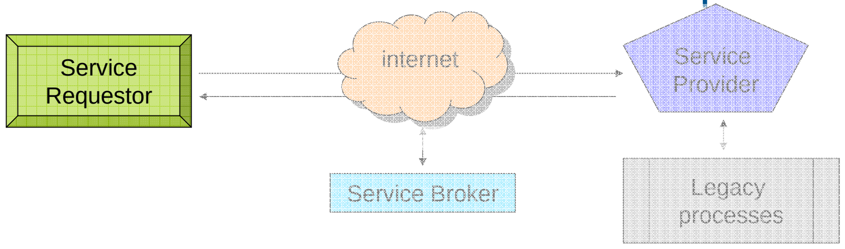
Service Provider

- Creates a web service
- Could publish its interface and access information to the service registry
- Each provider has to decide :
  - Which services to expose (trade-off between security and availability) ?
  - Which category the services should be listed in for a given broker service?
  - What type of agreements are required to use the service?

GoFurther


IDUG® 2007 North America

Web Services Architecture Components

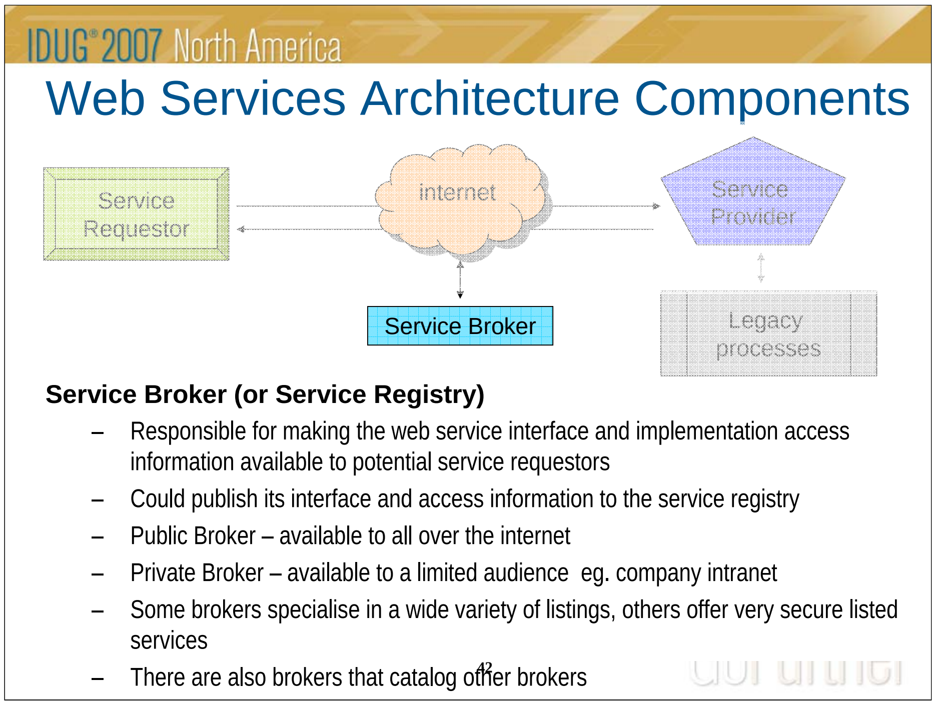


Service Requestor

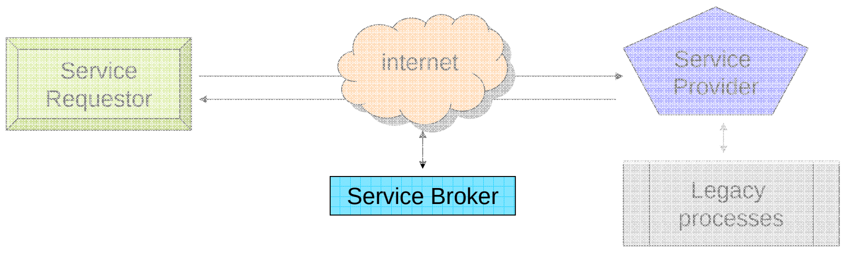
- Locates entries in the broker registry using a variety of find operations
- Binds to the service provider and invokes one of its Web services
- Dynamic choice of services opens up a whole range of issues
  - How to choose the best service provider
  - How to access quality of service
  - How the service user can assess the risk of exposure to service supplier failures

41

GoFurther

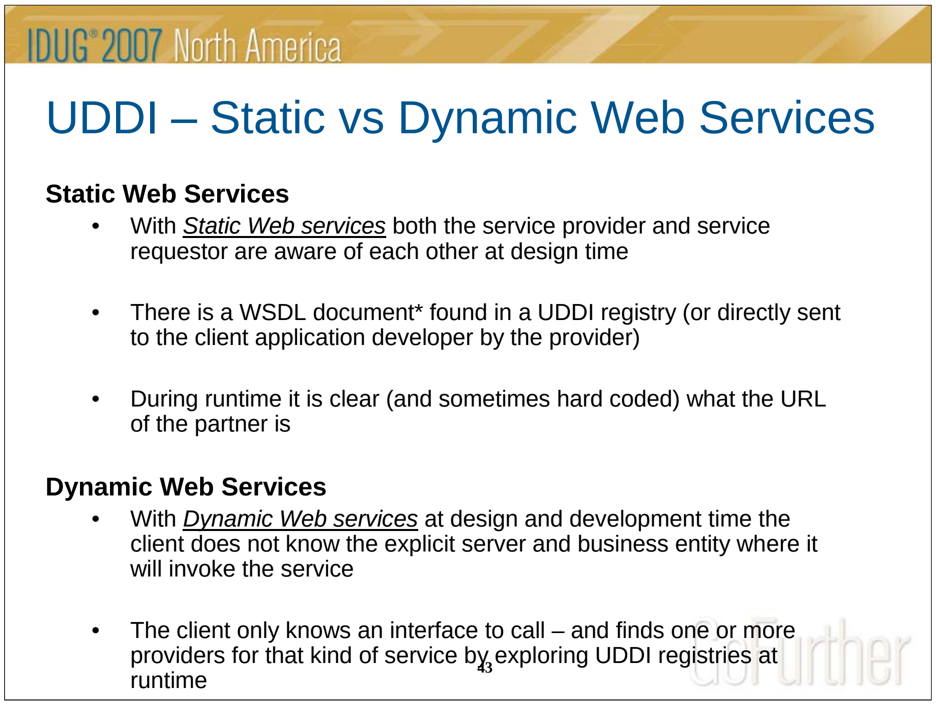


Web Services Architecture Components



Service Broker (or Service Registry)

- Responsible for making the web service interface and implementation access information available to potential service requestors
- Could publish its interface and access information to the service registry
- Public Broker – available to all over the internet
- Private Broker – available to a limited audience eg. company intranet
- Some brokers specialise in a wide variety of listings, others offer very secure listed services
- There are also brokers that catalog other brokers


IDUG® 2007 North America

UDDI – Static vs Dynamic Web Services

Static Web Services

- With *Static Web services* both the service provider and service requestor are aware of each other at design time
- There is a WSDL document* found in a UDDI registry (or directly sent to the client application developer by the provider)
- During runtime it is clear (and sometimes hard coded) what the URL of the partner is

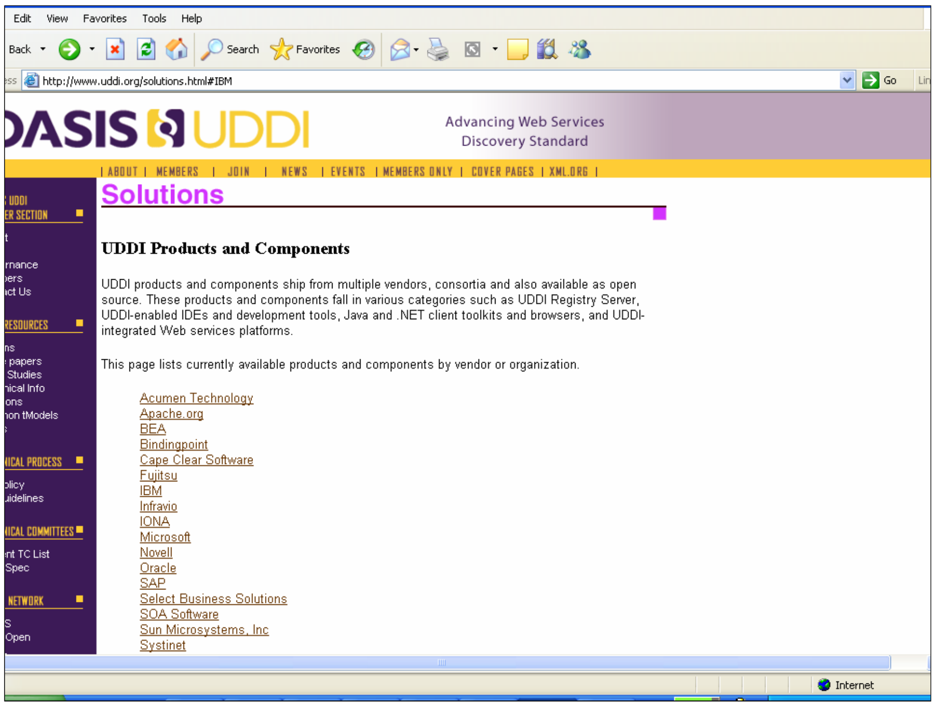
Dynamic Web Services

- With *Dynamic Web services* at design and development time the client does not know the explicit server and business entity where it will invoke the service
- The client only knows an interface to call – and finds one or more providers for that kind of service by exploring UDDI registries at runtime

43

GoFurther

A WSDL document specifies the location of the Web service, possible binding protocols and formats for operations & messages.


Edit View Favorites Tools Help
Back Search Favorites
http://www.uddi.org/solutions.html#IBM
Go
OASIS UDDI
Advancing Web Services Discovery Standard
| ABOUT | MEMBERS | JOIN | NEWS | EVENTS | MEMBERS ONLY | COVER PAGES | XML.ORG |
Internet

Solutions

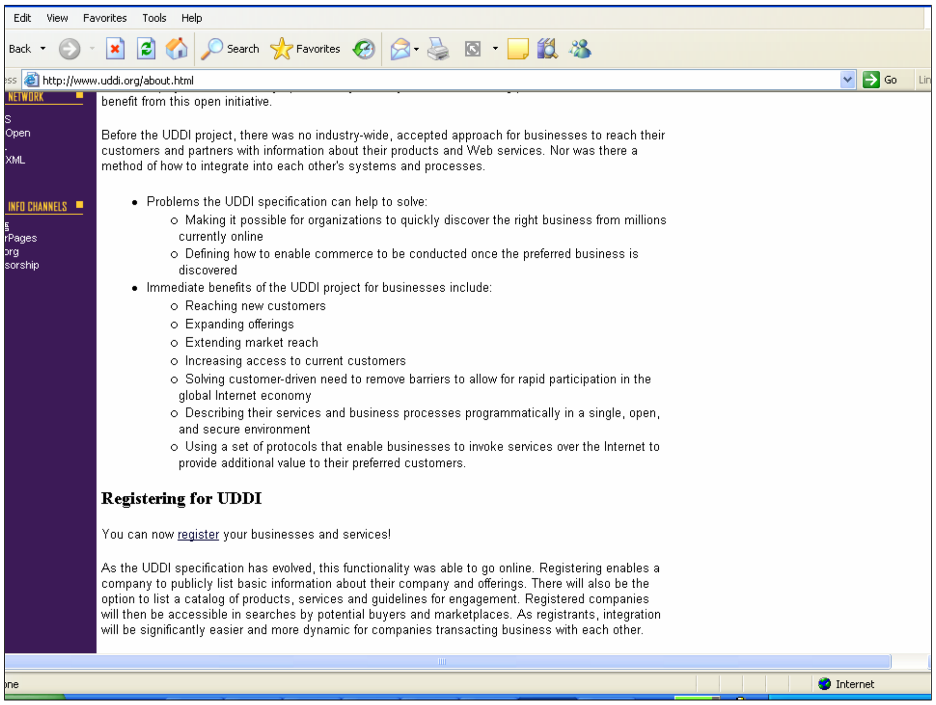
UDDI Products and Components

UDDI products and components ship from multiple vendors, consortia and also available as open source. These products and components fall in various categories such as UDDI Registry Server, UDDI-enabled IDEs and development tools, Java and .NET client toolkits and browsers, and UDDI-integrated Web services platforms.

This page lists currently available products and components by vendor or organization.

- Acumen Technology
- Apache.org
- BEA
- Bindingpoint
- Cape Clear Software
- Fujitsu
- IBM
- Infravio
- IONA
- Microsoft
- Novell
- Oracle
- SAP
- Select Business Solutions
- SOA Software
- Sun Microsystems, Inc
- Systinet

A WSDL document specifies the location of the Web service, possible binding protocols and formats for operations & messages.


Edit View Favorites Tools Help
Back
Search
Favorites
http://www.uddi.org/about.html
Go
NETWORK
INFO CHANNELS
Internet

benefit from this open initiative.

Before the UDDI project, there was no industry-wide, accepted approach for businesses to reach their customers and partners with information about their products and Web services. Nor was there a method of how to integrate into each other's systems and processes.

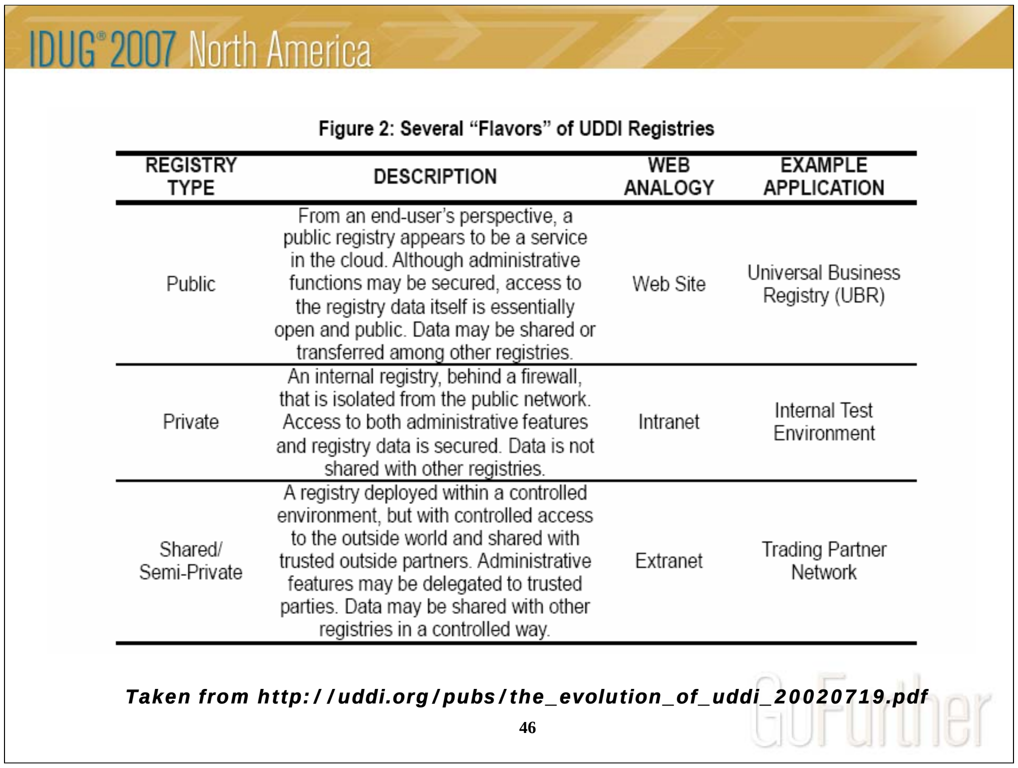
- Problems the UDDI specification can help to solve:
  - Making it possible for organizations to quickly discover the right business from millions currently online
  - Defining how to enable commerce to be conducted once the preferred business is discovered
- Immediate benefits of the UDDI project for businesses include:
  - Reaching new customers
  - Expanding offerings
  - Extending market reach
  - Increasing access to current customers
  - Solving customer-driven need to remove barriers to allow for rapid participation in the global Internet economy
  - Describing their services and business processes programmatically in a single, open, and secure environment
  - Using a set of protocols that enable businesses to invoke services over the Internet to provide additional value to their preferred customers.

Registering for UDDI

You can now register your businesses and services!

As the UDDI specification has evolved, this functionality was able to go online. Registering enables a company to publicly list basic information about their company and offerings. There will also be the option to list a catalog of products, services and guidelines for engagement. Registered companies will then be accessible in searches by potential buyers and marketplaces. As registrants, integration will be significantly easier and more dynamic for companies transacting business with each other.

A WSDL document specifies the location of the Web service, possible binding protocols and formats for operations & messages.



IDUG® 2007 North America

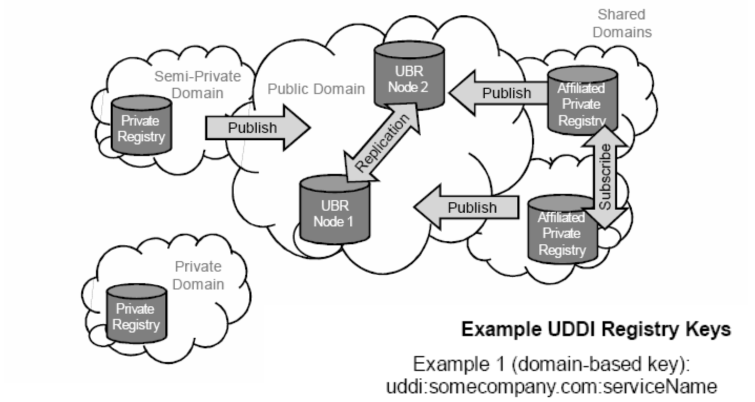
Figure 2: Several "Flavors" of UDDI Registries

REGISTRY TYPE	DESCRIPTION	WEB ANALOGY	EXAMPLE APPLICATION
Public	From an end-user's perspective, a public registry appears to be a service in the cloud. Although administrative functions may be secured, access to the registry data itself is essentially open and public. Data may be shared or transferred among other registries.	Web Site	Universal Business Registry (UBR)
Private	An internal registry, behind a firewall, that is isolated from the public network. Access to both administrative features and registry data is secured. Data is not shared with other registries.	Intranet	Internal Test Environment
Shared/ Semi-Private	A registry deployed within a controlled environment, but with controlled access to the outside world and shared with trusted outside partners. Administrative features may be delegated to trusted parties. Data may be shared with other registries in a controlled way.	Extranet	Trading Partner Network

Taken from http://uddi.org/pubs/the_evolution_of_uddi_20020719.pdf

46

GoFurther

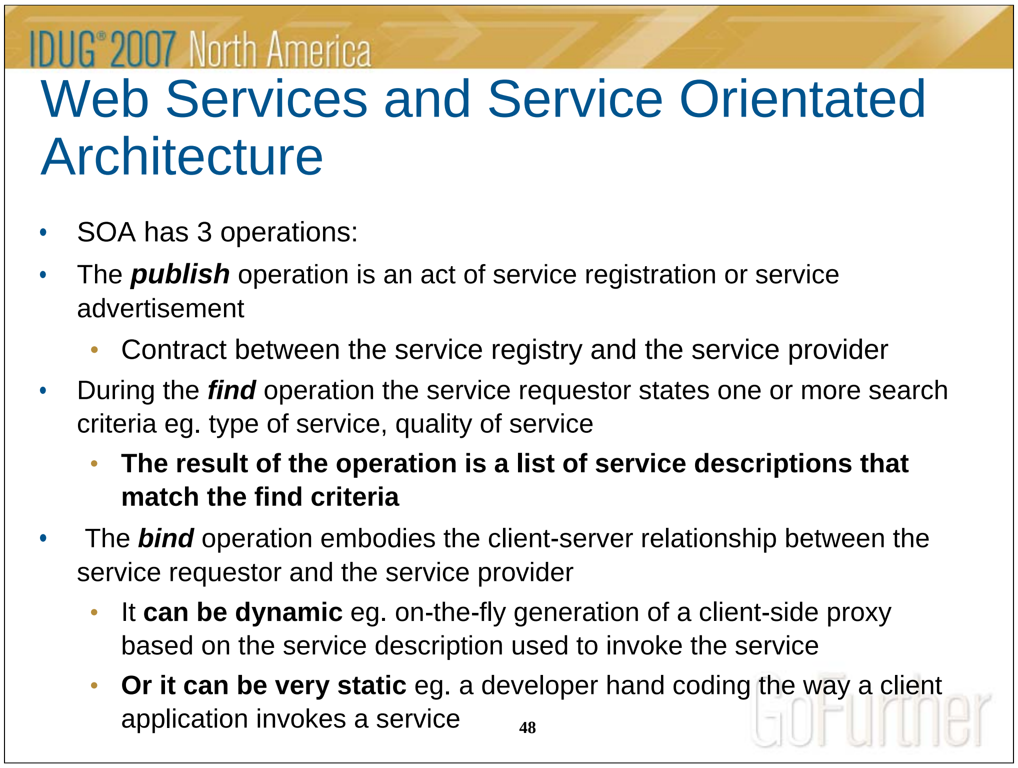


Example UDDI Registry Keys

Example 1 (domain-based key):
uddi:somecompany.com:serviceName

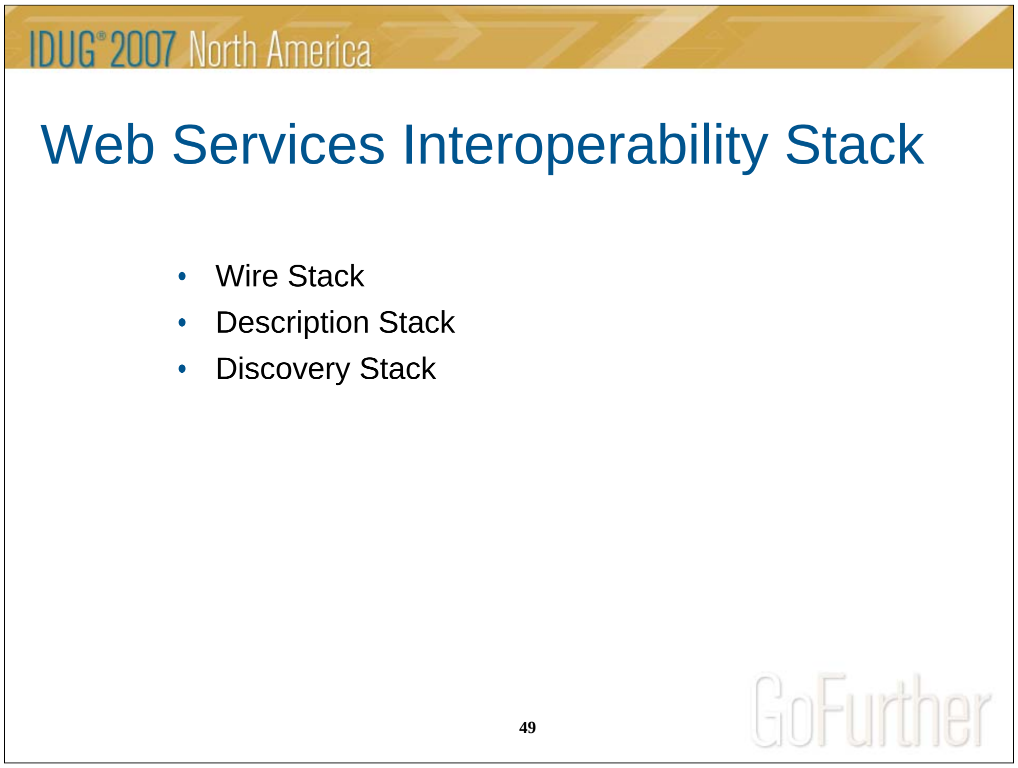
Example 2 (globally unique identifier):
uddi:4CD7E4BC-648B-426D-9936-443EAAC8AE23

Taken from http://uddi.org/pubs/the_evolution_of_uddi_20020719.pdf



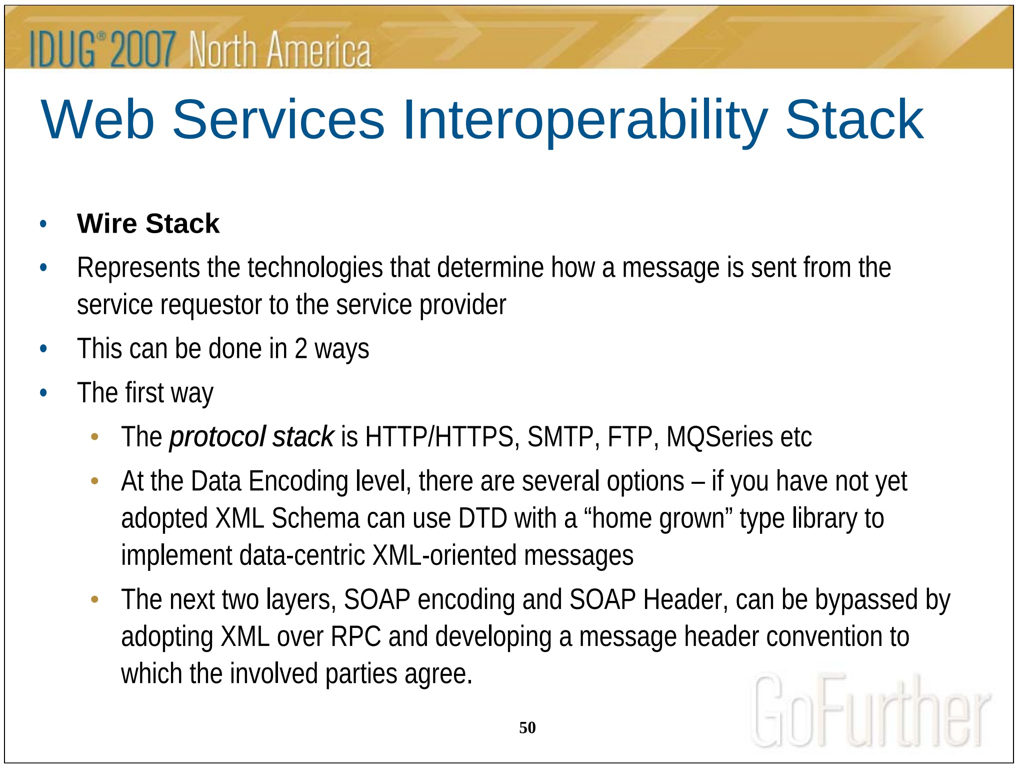
Web Services and Service Orientated Architecture

- SOA has 3 operations:
- The ***publish*** operation is an act of service registration or service advertisement
  - Contract between the service registry and the service provider
- During the ***find*** operation the service requestor states one or more search criteria eg. type of service, quality of service
  - **The result of the operation is a list of service descriptions that match the find criteria**
- The ***bind*** operation embodies the client-server relationship between the service requestor and the service provider
  - It **can be dynamic** eg. on-the-fly generation of a client-side proxy based on the service description used to invoke the service
  - **Or it can be very static** eg. a developer hand coding the way a client application invokes a service



Web Services Interoperability Stack

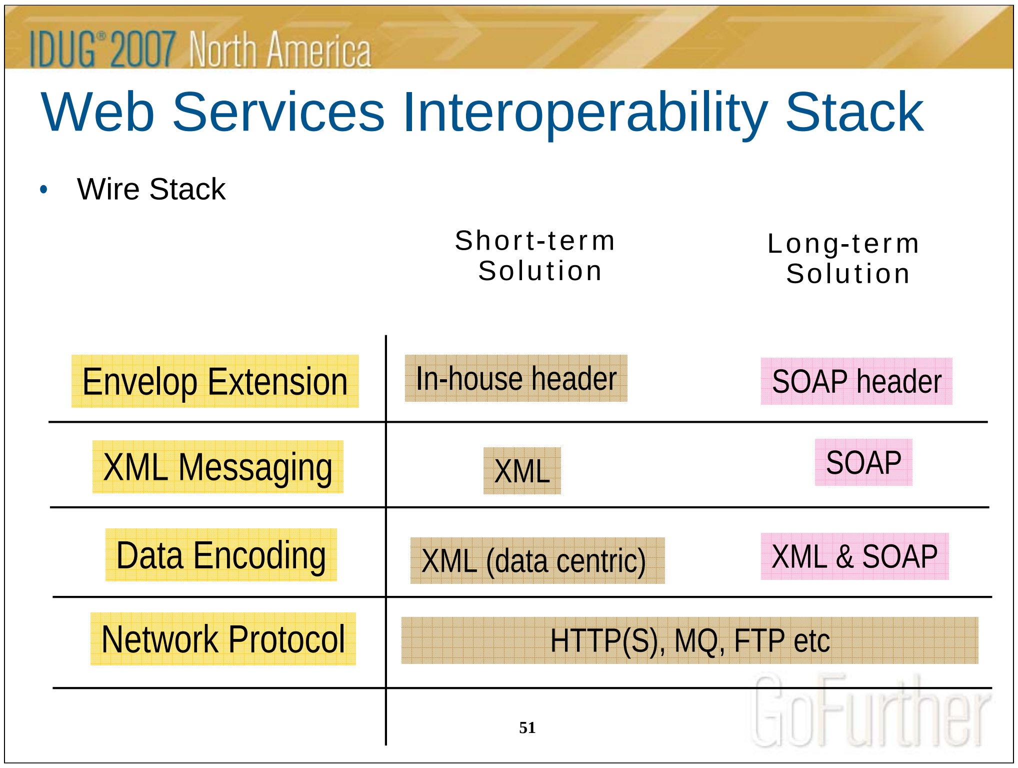
- Wire Stack
- Description Stack
- Discovery Stack



Web Services Interoperability Stack

- **Wire Stack**
- Represents the technologies that determine how a message is sent from the service requestor to the service provider
- This can be done in 2 ways
- The first way
    - The *protocol stack* is HTTP/HTTPS, SMTP, FTP, MQSeries etc
    - At the Data Encoding level, there are several options – if you have not yet adopted XML Schema can use DTD with a "home grown" type library to implement data-centric XML-oriented messages
    - The next two layers, SOAP encoding and SOAP Header, can be bypassed by adopting XML over RPC and developing a message header convention to which the involved parties agree.

GoFurther



IDUG® 2007 North America

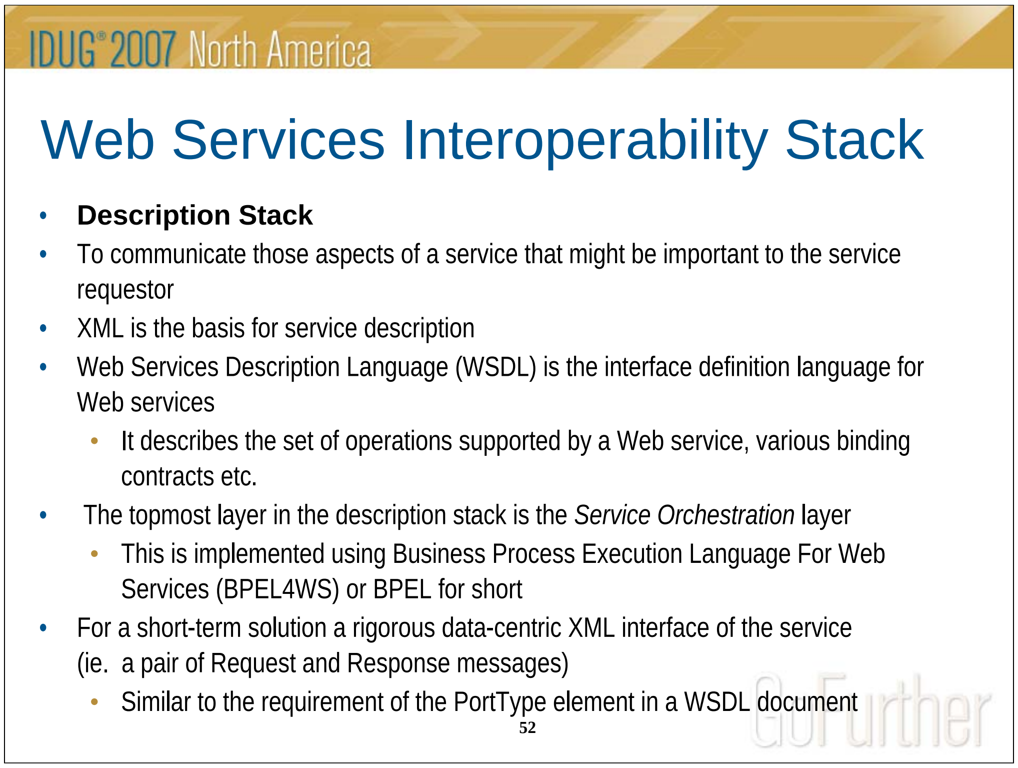
Web Services Interoperability Stack

- Wire Stack

	Short-term Solution	Long-term Solution
Envelop Extension	In-house header	SOAP header
XML Messaging	XML	SOAP
Data Encoding	XML (data centric)	XML & SOAP
Network Protocol	HTTP(S), MQ, FTP etc	

51

GoFurther



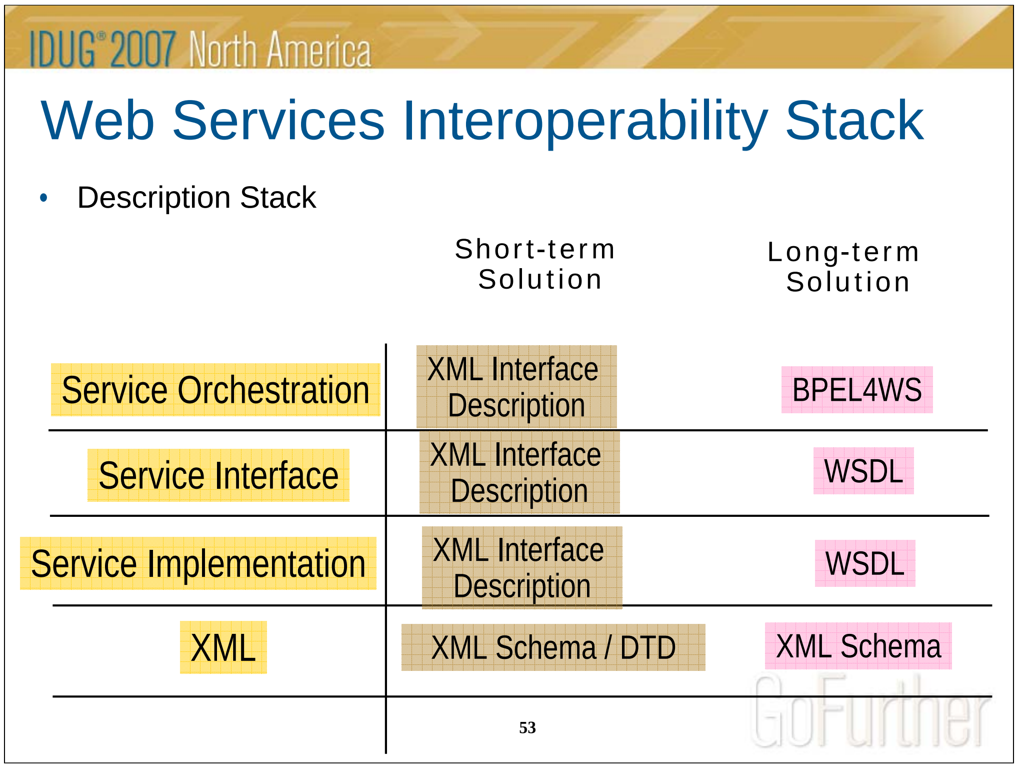
IDUG® 2007 North America

Web Services Interoperability Stack

- **Description Stack**
- To communicate those aspects of a service that might be important to the service requestor
- XML is the basis for service description
- Web Services Description Language (WSDL) is the interface definition language for Web services
  - It describes the set of operations supported by a Web service, various binding contracts etc.
- The topmost layer in the description stack is the *Service Orchestration* layer
  - This is implemented using Business Process Execution Language For Web Services (BPEL4WS) or BPEL for short
- For a short-term solution a rigorous data-centric XML interface of the service (ie. a pair of Request and Response messages)
  - Similar to the requirement of the PortType element in a WSDL document

52

GoFurther



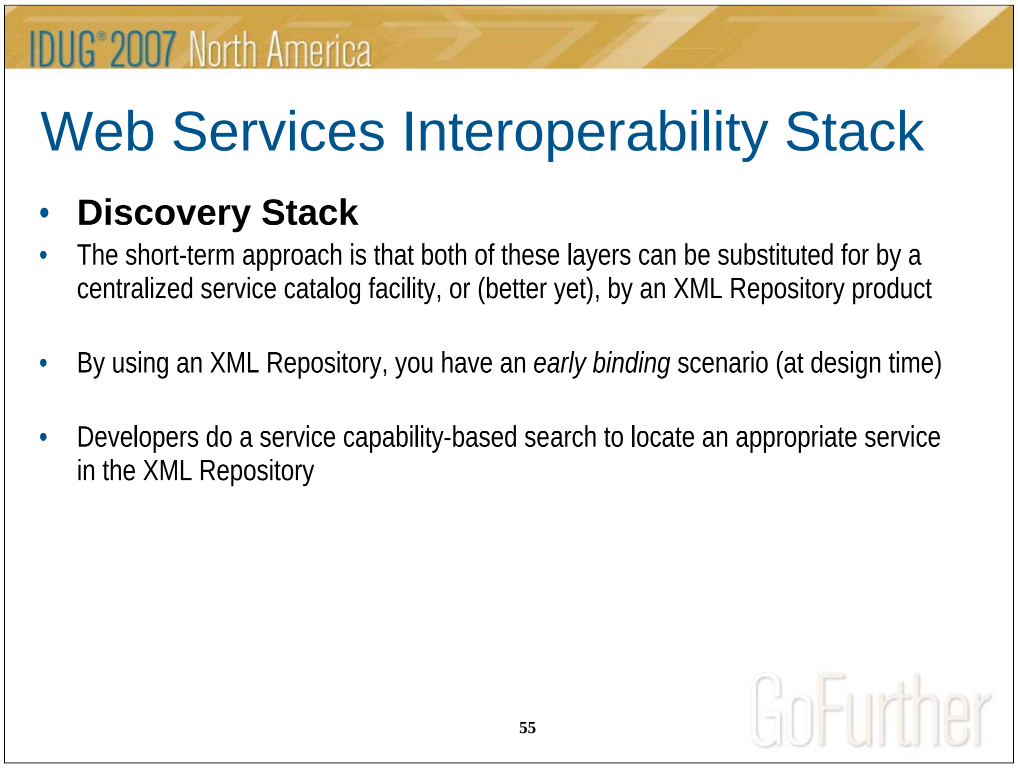
Web Services Interoperability Stack

- Description Stack

	Short-term Solution	Long-term Solution
Service Orchestration	XML Interface Description	BPEL4WS
Service Interface	XML Interface Description	WSDL
Service Implementation	XML Interface Description	WSDL
XML	XML Schema / DTD	XML Schema

Web Services Interoperability Stack

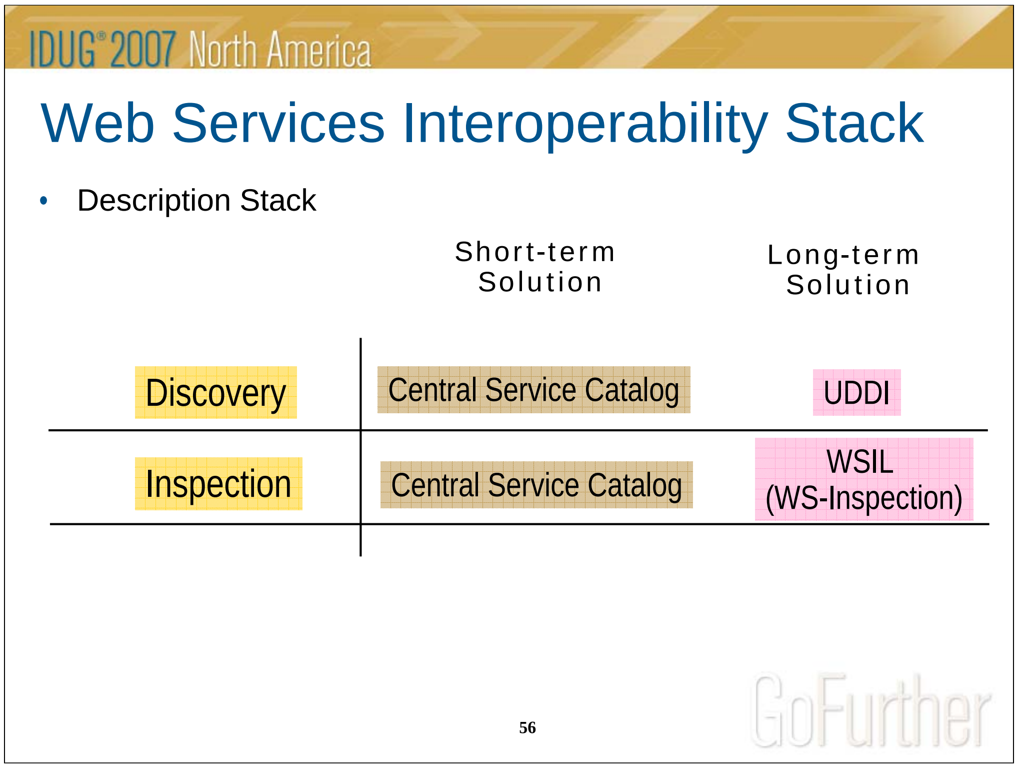
- **Discovery Stack**
- Organize technologies associated with Web services discovery
- The first level of the stack represents a simple inspection level
  - Inspection is a technique for discovering the service description given that the details about the service are already known
  - The Universal Description, Discovery, and Integration (UDDI) specification addresses a subset of the overall requirements by using a centralized service discovery model
  - The Web Services Inspection Language (WS-Inspection) is another service discovery mechanism that addresses a different subset using a distributed usage model
  - The WS-Inspection specification is designed around an XML-based model for building an aggregation of references to existing Web service descriptions
- The discovery level represents the capability of discovering Web services and service providers using a capability-based lookup



Web Services Interoperability Stack

- **Discovery Stack**
- The short-term approach is that both of these layers can be substituted for by a centralized service catalog facility, or (better yet), by an XML Repository product
- By using an XML Repository, you have an *early binding* scenario (at design time)
- Developers do a service capability-based search to locate an appropriate service in the XML Repository

GoFurther


IDUG® 2007 North America

Web Services Interoperability Stack

- Description Stack

	Short-term Solution	Long-term Solution
Discovery	Central Service Catalog	UDDI
Inspection	Central Service Catalog	WSIL (WS-Inspection)

56

GoFurther

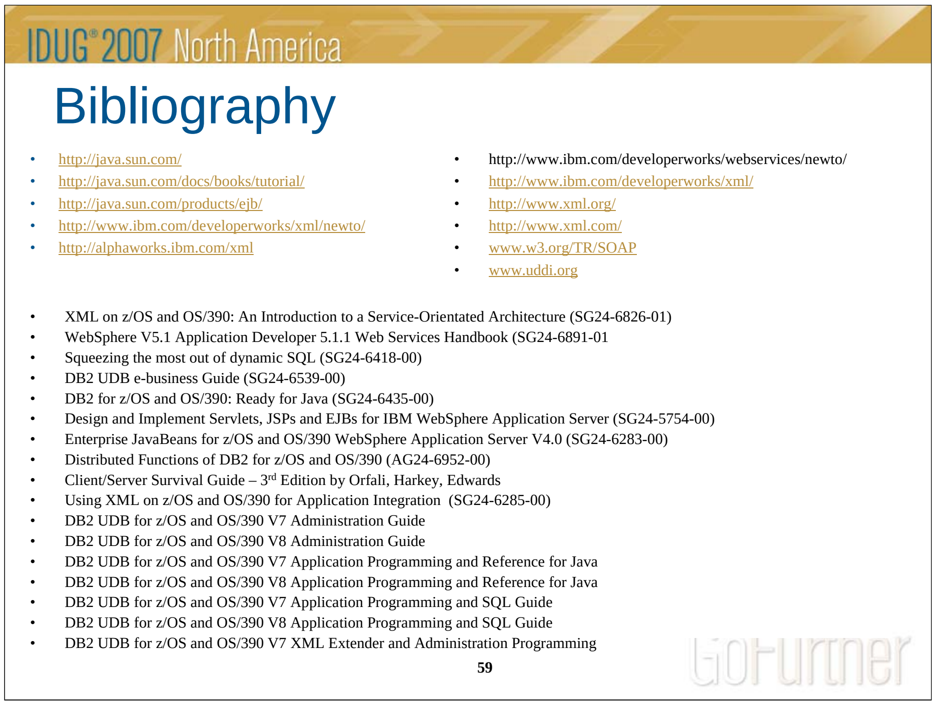
Agenda

- Introduction to the Bigger Picture - The World of Web Services
- Understanding the Terminology
- XML - The Universal Language
- Web Services and Service Orientated Architecture (SOA)

GoFurther

Conclusion

- Web Services are more likely to use existing application software than to replace it
- Business processes comprised of Web Services will be much easier to adapt to changing customer needs and business climates than are today's home-grown or purchased applications
- The difference between using functionality on a Web site and using a Web Service is the fact that a person doesn't have to be the one interacting with the Web-accessible resource.
  - Instead, any software program can invoke and use (or consume) the service by requesting it over the Internet



IDUG® 2007 North America

Bibliography

- http://java.sun.com/
- http://java.sun.com/docs/books/tutorial/
- http://java.sun.com/products/ejb/
- http://www.ibm.com/developerworks/xml/newto/
- http://alphaworks.ibm.com/xml
- http://www.ibm.com/developerworks/webservices/newto/
- http://www.ibm.com/developerworks/xml/
- http://www.xml.org/
- http://www.xml.com/
- www.w3.org/TR/SOAP
- www.uddi.org

- XML on z/OS and OS/390: An Introduction to a Service-Orientated Architecture (SG24-6826-01)
- WebSphere V5.1 Application Developer 5.1.1 Web Services Handbook (SG24-6891-01
- Squeezing the most out of dynamic SQL (SG24-6418-00)
- DB2 UDB e-business Guide (SG24-6539-00)
- DB2 for z/OS and OS/390: Ready for Java (SG24-6435-00)
- Design and Implement Servlets, JSPs and EJBs for IBM WebSphere Application Server (SG24-5754-00)
- Enterprise JavaBeans for z/OS and OS/390 WebSphere Application Server V4.0 (SG24-6283-00)
- Distributed Functions of DB2 for z/OS and OS/390 (AG24-6952-00)
- Client/Server Survival Guide – 3rd Edition by Orfali, Harkey, Edwards
- Using XML on z/OS and OS/390 for Application Integration (SG24-6285-00)
- DB2 UDB for z/OS and OS/390 V7 Administration Guide
- DB2 UDB for z/OS and OS/390 V8 Administration Guide
- DB2 UDB for z/OS and OS/390 V7 Application Programming and Reference for Java
- DB2 UDB for z/OS and OS/390 V8 Application Programming and Reference for Java
- DB2 UDB for z/OS and OS/390 V7 Application Programming and SQL Guide
- DB2 UDB for z/OS and OS/390 V8 Application Programming and SQL Guide
- DB2 UDB for z/OS and OS/390 V7 XML Extender and Administration Programming

59

GoFurther

Session H05

Session Title : XML, Web Services and SOA for Smarties

Maria Sarikos

Ab Initio

msarikos@abinitio.com