TOC

INDEX

VIEW

B07 Web Services and How DB2 Plays

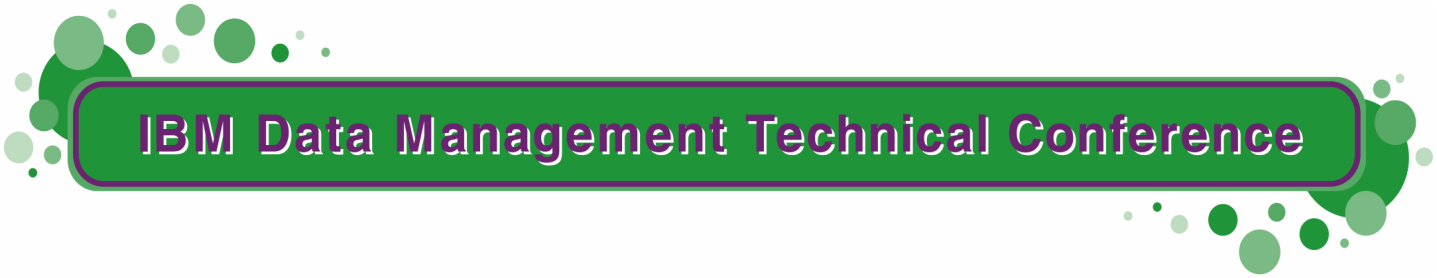
George Zagelow, XML System Manager, IBM

The Internet is evolving to a network of Web Services, where functions appear to potential partner applications as services, represented and described to enable automated interaction. Driving the evolution are requirements for easier Enterprise Application Integration and broader, more automated Business to Business interaction. Central to the discussion will be base technologies including XML, SOAP, WSDL and UDDI, as well as some very new additions - WS-Interoperability and WS-Security. The talk will include IBM's strategy for Web Services, and support in IBM products with emphasis on DB2 enablement. This presentation is meant to be a stand alone overview of the Web Services landscape, as well as an introduction to the more detailed talks scheduled later in the conference.

B07

Web Services and How DB2 Plays

George Zagelow
IBM Software Group



Anaheim, CA **Sept 9 - 13, 2002**

RETURN TO INDEX

Agenda

- **The Web Services model**
  - What are Web Serves and what problems do they solve?
  - SOAP - ~~Simple Object Access Protocol~~
  - WSDL - Web Services Description Language
- **Related Technologies**
  - UDDI - Publishing and Finding Web Services
  - WSFL - Web Services Workflow and Composition Language
  - Web Services and XML Schema
  - Web Services and Security
  - New Hot Items
- **IBM and Web Services**
  - IBM Web Services strategy
  - DB2, WebSphere, and other enabled products
  - Tools and Resources
- **Summary**

A New Web Model

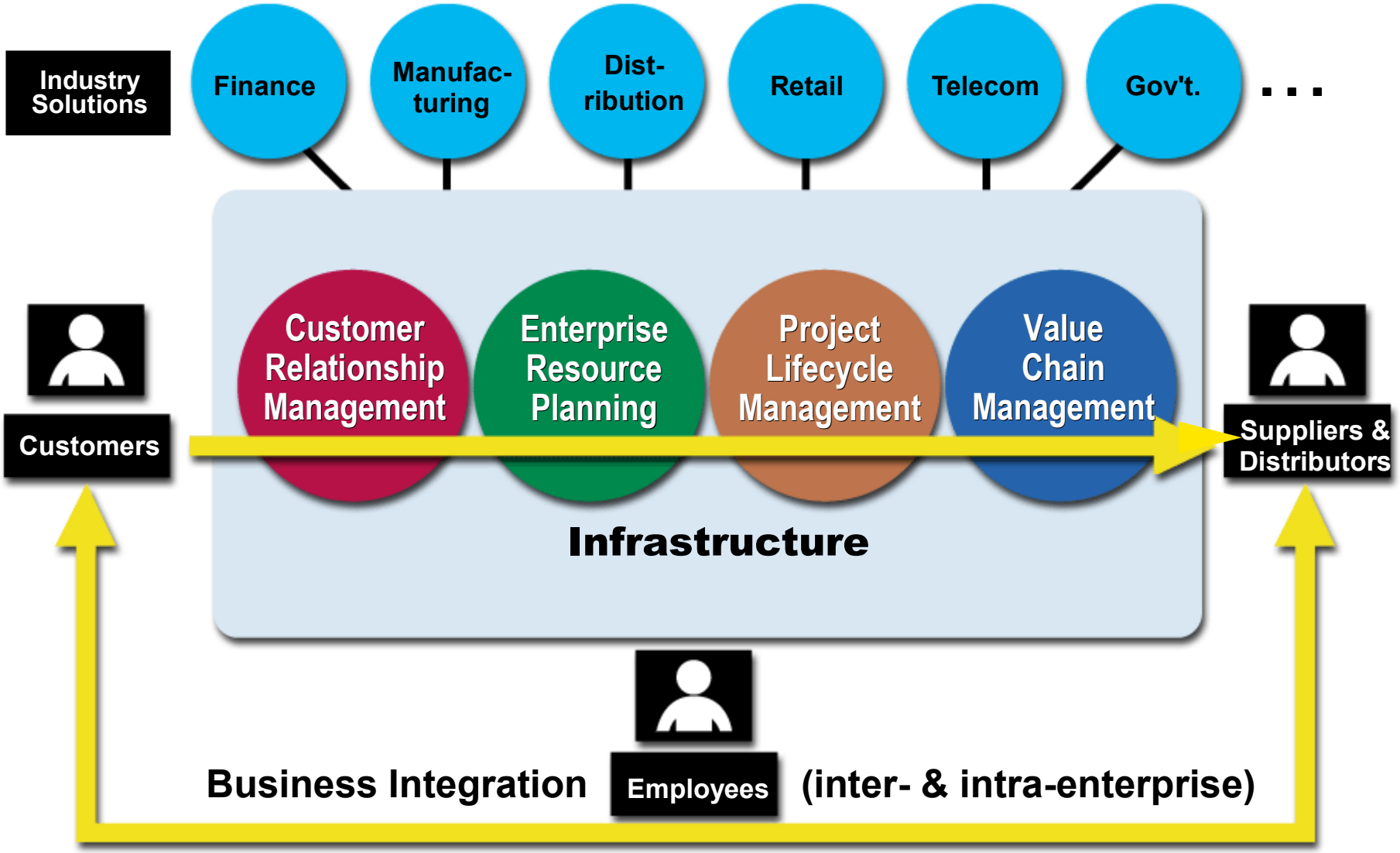
- Until now, the Web has provided for
  - browsing of linked documents
  - manually-initiated purchases and transactions
  - downloading files
    - all of this is manual, by way of a browser
- Web Services is a new model for using the Web
  - transactions initiated automatically by a program, not necessarily using a browser
  - can be described, published, discovered, and invoked dynamically in a distributed computing environment
  - new ways of using the web: intelligent agents, marketplaces, auctions
    - **all built on XML and other internet standards!**

Why Web services?

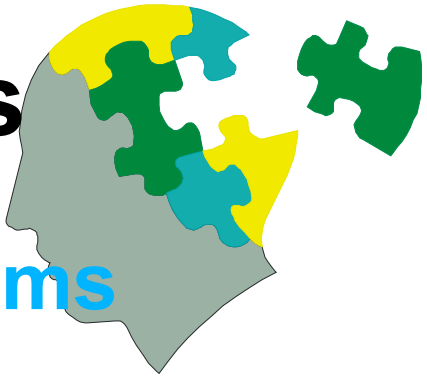
- We want and need:
  - to integrate systems regardless of their implementation
  - to move from monolithic, custom-coded apps to choreographed, scripted components.
  - agility and flexibility to reconfigure business functions to try new process models.
  - to move from tightly coupled systems to loosely coupled ones to deal with inevitable change.
  - a well-understood programming model for connecting businesses via the Internet.

End-to-end Integration

...from Demand through Delivery



EAI: Putting together the pieces



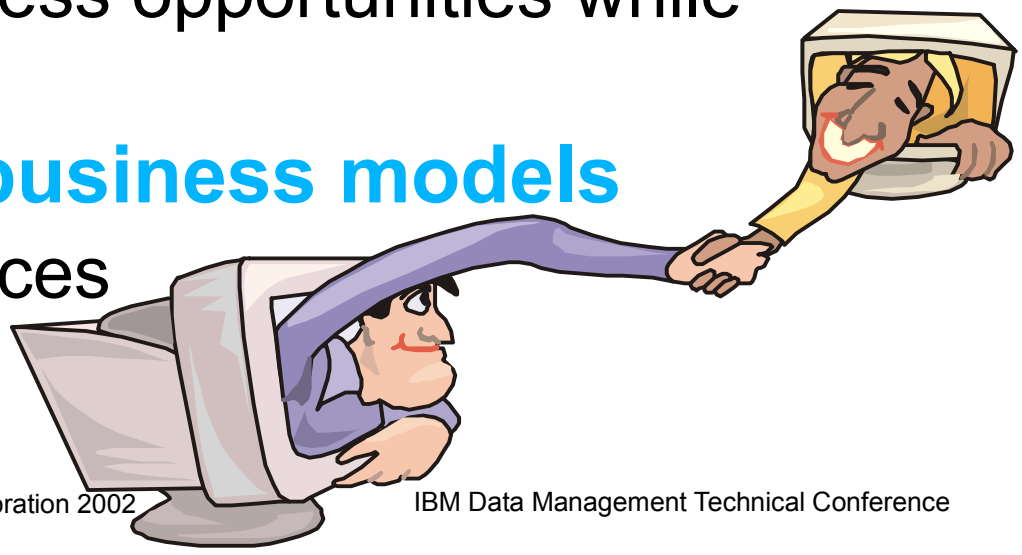
- **Legacy: heterogenous application systems**
  - Difficult to tie them together
- **Traditional integration (compiled-in API's and file formats)**
  - Leads to "brittle" systems that crash with the smallest of changes
  - Different systems, different techniques: no standards
- **Need fast integration for Mergers and Acquisitions**
  - An "integration-ready" company has much greater value
- **Need agility and flexibility in business processes**
  - Respond to business changes
  - Try out new business process ideas quickly and cheaply

How do Businesses Connect?

- Businesses connect to each other using a wide variety of methods.
- If you restrict yourself to a proprietary or platform-specific method, you are limiting your interactions to a relatively small community.
- On a global scale, on a good day we would say the connection methods are ad hoc.
- On a bad day, we might say that the range of connection methods might induce chaos.

B2B: better faster cheaper

- **Rapid and deep integration with business partners**
  - Harder than EAI: you don't control the software you need to integrate with!
- **Reduced cost of doing business through more efficient communication**
  - eliminate manual processes, paper communication
- **Find new business partners, integrate quickly**
  - new supplier when the old one can't deliver in time
  - respond to emerging business opportunities while they're hot
- **Participate in emerging business models**
  - industry-specific marketplaces
  - auctions



What is a Web Service?

"Web services are software components described via WSDL which are capable of being accessed via standard network protocols such as SOAP over HTTP."

Web Service

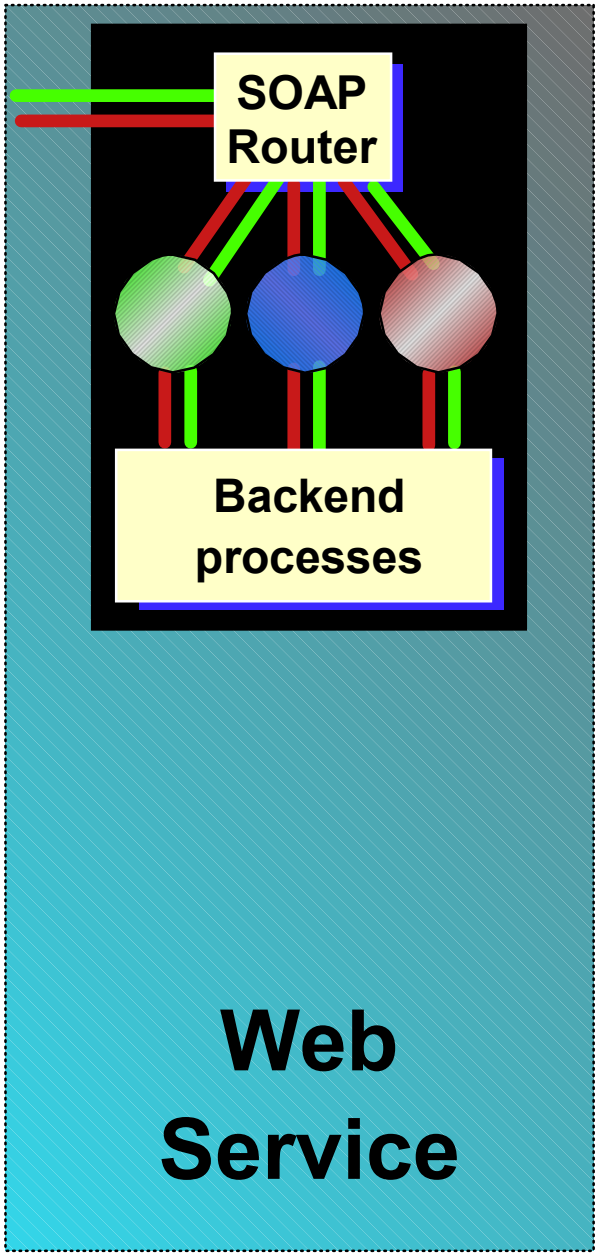
What is a Web Service?

"Web services are software components described via WSDL which are capable of being accessed via standard network protocols such as SOAP over HTTP."

Today, SOAP over HTTP is the common protocol for Web services.

For now, a SOAP interface connected to application processes can be thought of as a minimum...

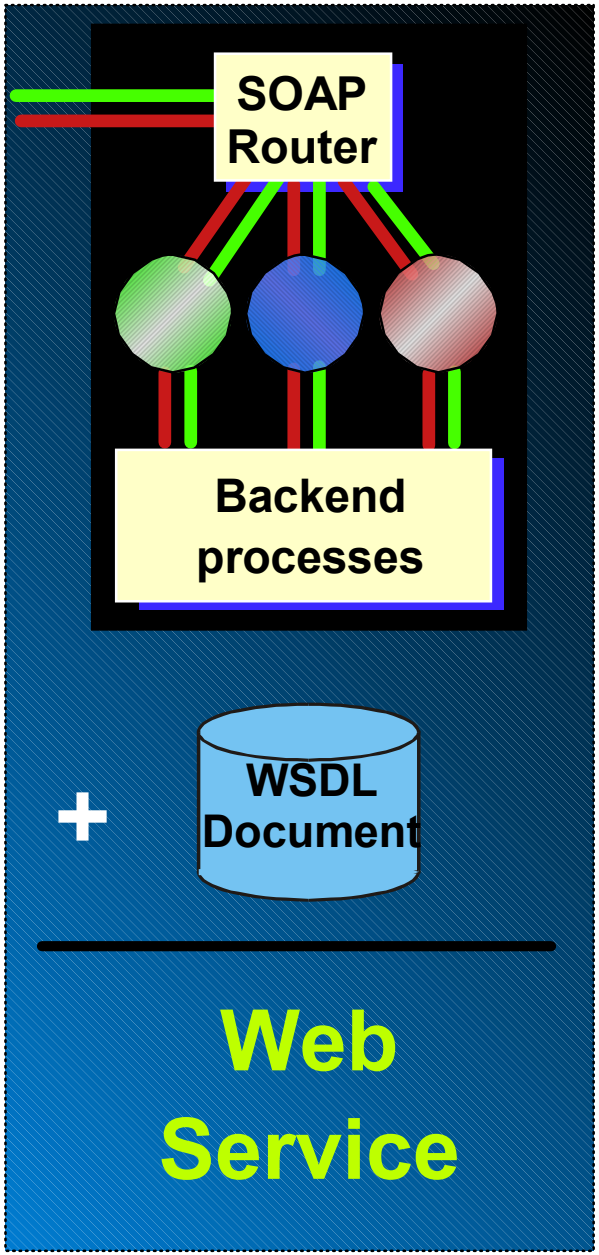
...but by itself does not address rapid integration.



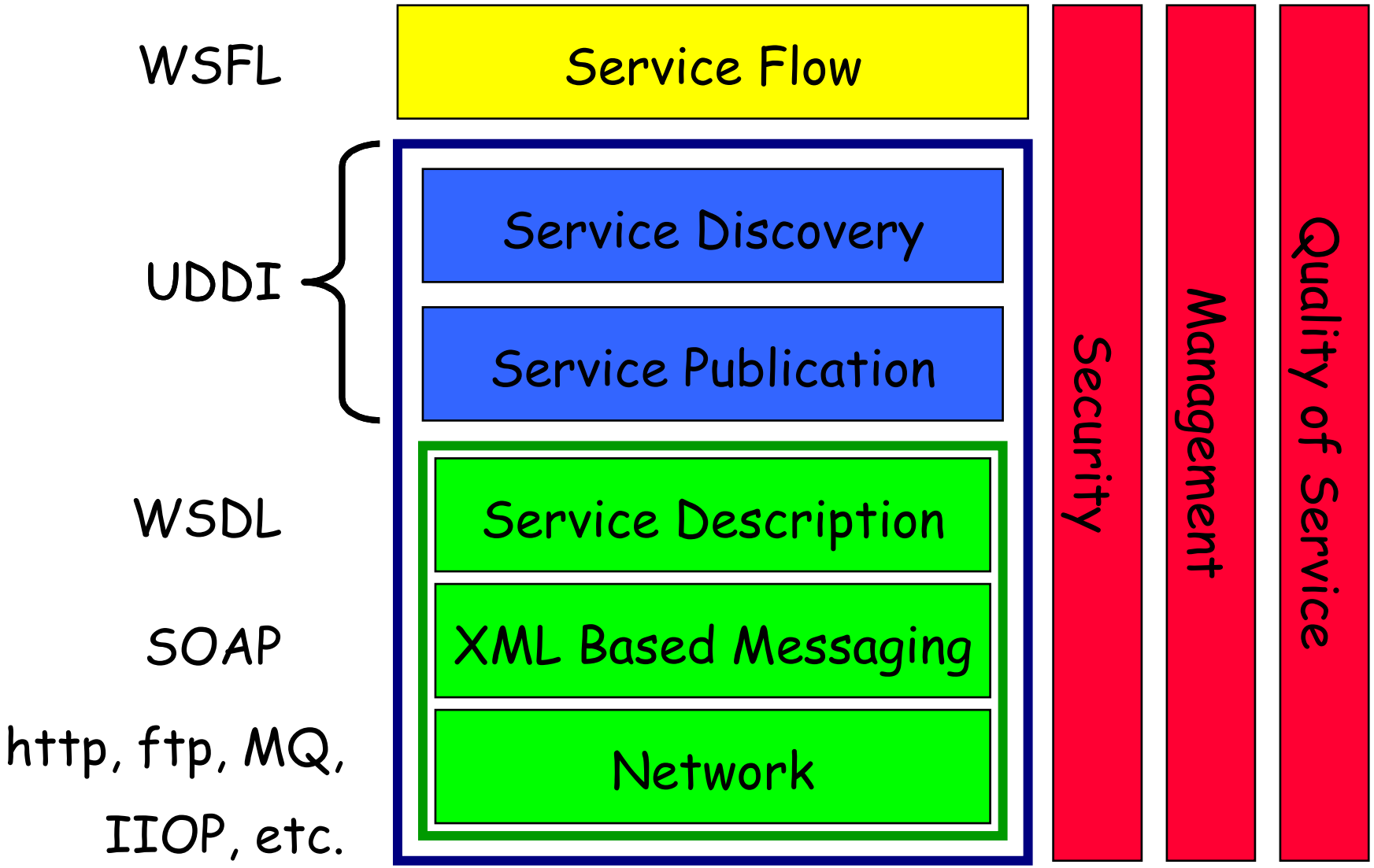
What is a Web Service?

""Web services are software components described via WSDL which are capable of being accessed via standard network protocols such as SOAP over HTTP."

- WSDL descriptions can be used to drive assembly tools, code generators, and other tools to speed integration.
- For now, SOAP+WSDL can be thought of as the base technologies for any Web service.
  - UDDI, other technologies can be considered optional, to add on as makes sense for the application



The Web Services "stack"



SOAP: ~~Simple Object Access Protocol~~

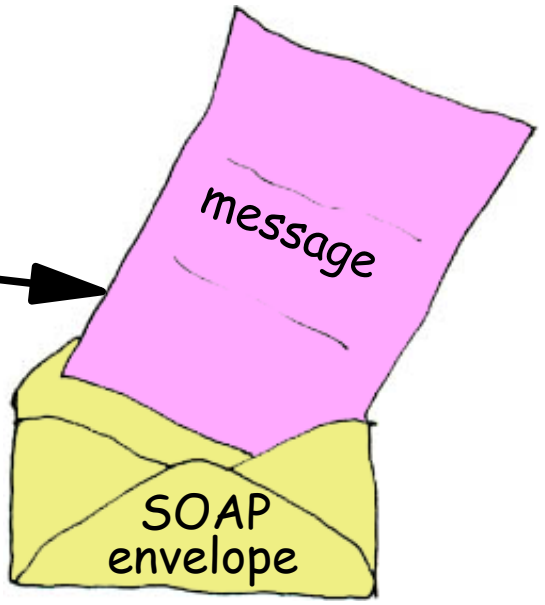
- SOAP 1.0: Userland, Microsoft, DevelopMentor
  - SOAP 1.0 was specific to COM and HTTP
- SOAP 1.1 (April 26, 2000) - includes contributions from IBM and Lotus
  - substitutable Transport bindings (not just HTTP)
  - substitutable Language bindings (e.g. Java)
  - substitutable Data encodings (pluggable)
  - completely vendor-neutral
  - independent of: programming language, object model, operating system, or platform
- SOAP 1.2 - working draft from w3.org "XML Protocol" working group, went to "last call" (June 26, 2002)
  - it will be called "SOAP 1.2", not "XML Protocol 1.0"

SOAP Message structure

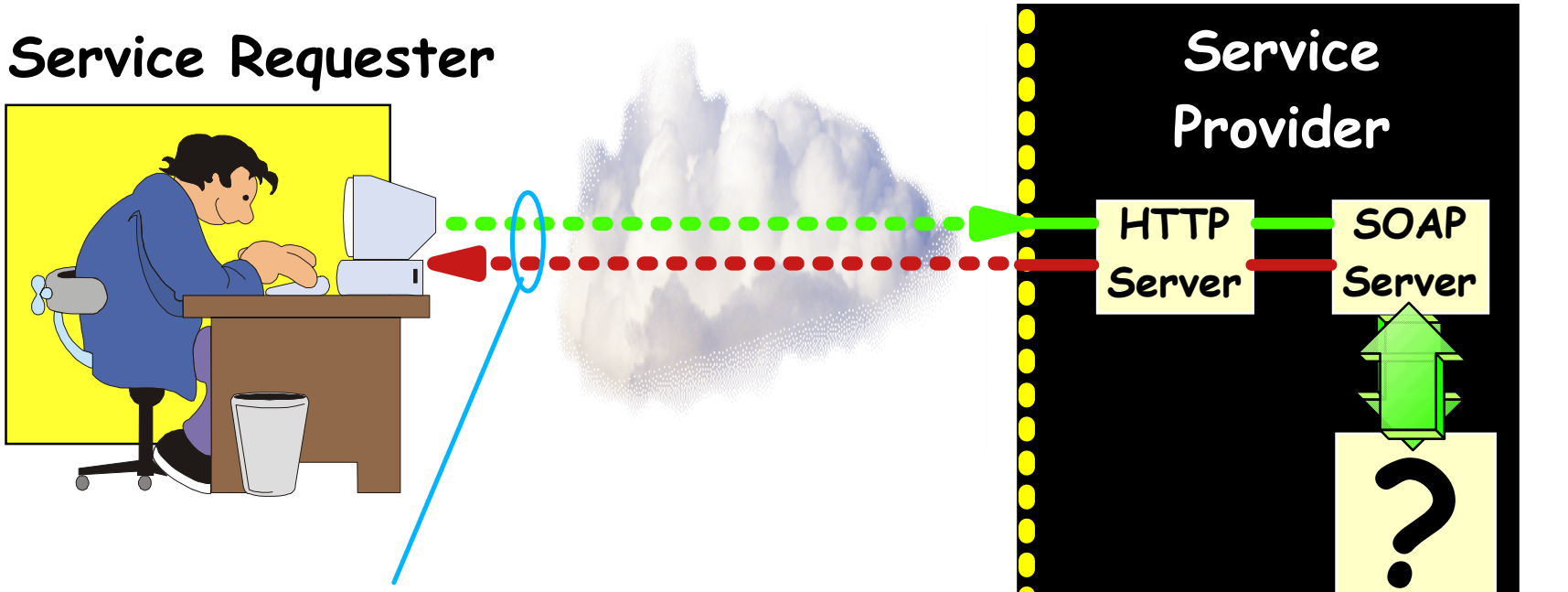
- Request and Response messages
  - Request invokes a method on a remote object
  - Response returns result of running the method
- SOAP specification defines an "envelope"
  - "envelope" wraps the message itself
  - message is a different vocabulary
  - namespace prefix is used to distinguish the two parts

application-specific message vocabulary

SOAP envelope vocabulary



SOAP hides the service implementation



Simple, standard XML messages

- ➡ we're only interested in the request and response messages
- ➡ we don't really want to know about the implementation details (less work for us!)

EJB? Corba? Cobol?

Why SOAP Will Succeed

Other distributed technologies failed on the Internet because they strongly coupled the endpoints:

- ✗ RMI - requires Java at each endpoint
- ✗ CORBA - requires compatible ORBs at each endpoint
- ✗ DCOM - requires Windows at each endpoint

- ✓ SOAP is the platform-neutral choice
  - ➡ simply an XML wire format
  - ➡ places no restrictions on the endpoint implementation technology choices
  - ➡ implementations are free, some are open-source

Apache SOAP 2.2

- History:
  - SOAP4J posted to IBM alphaWorks, April 2000
  - Contributed by IBM to the Apache Software Foundation, June 1, 2000
- SOAP from Apache:
  - Solid implementation of SOAP v1.1 Specification, supporting HTTP and SMTP protocols
  - platform-independent Java
  - Developed by IBM and others
  - Free download from xml.apache.org with source
- SOAP distribution includes:
  - User's Guide
  - API documentation
  - a tool for debugging SOAP
  - three samples

AXIS: SOAP 3.0

- New codebase implementing W3C SOAP 1.2 Specification
  - will feature full support for the spec
  - adds WSDL support
  - speed: SAX events for parsing SOAP messages
  - http://xml.apache.org/axis/index.html
- Alpha 3 release is available
  - adds JAX RPC, more WSDL support
- Implementation team spans many companies
  - including IBM (of course!)

SOAP Usage Models

- **RPC-like message exchange**
  - request message bundles up method name and parameters
  - response message contains method return value(s)
  - this is the commonly assumed model... but it isn't required by SOAP
- **The SOAP specification says nothing about the message content**
  - can be XML documents of any type
  - usage of message request and response is defined by the service provider
  - your application may not have RPC semantics, yet it can be a perfectly valid SOAP application
  - example:
    - send a purchase order document to the inbox of a B2B partner
    - expect to receive shipping and exceptions report as response

SOAP Resources

- **SOAP 1.1 Specification**
  - http://www.w3.org/TR/SOAP/
- **Apache SOAP4J: xml.apache.org**
  - SOAP4J version 2.2, stable, ready for use
  - AXIS (Alpha 3 available)
- **W3 standardization: w3.org/2000/xp**
  - SOAP 1.2 specification (draft)
  - XML Protocol working group requirements and charter
- **SOAP - WebServices Resource Center**
  - http://www.soap-wrc.com/webservices/default.asp
  - MANY resources - e.g., link to SOAP::Lite for Perl
- **Xmethods lists publicly-accessible web services**
  - http://www.xmethods.net
- **Articles and tutorials:**
  - http://ibm.com/developerworks/webservices

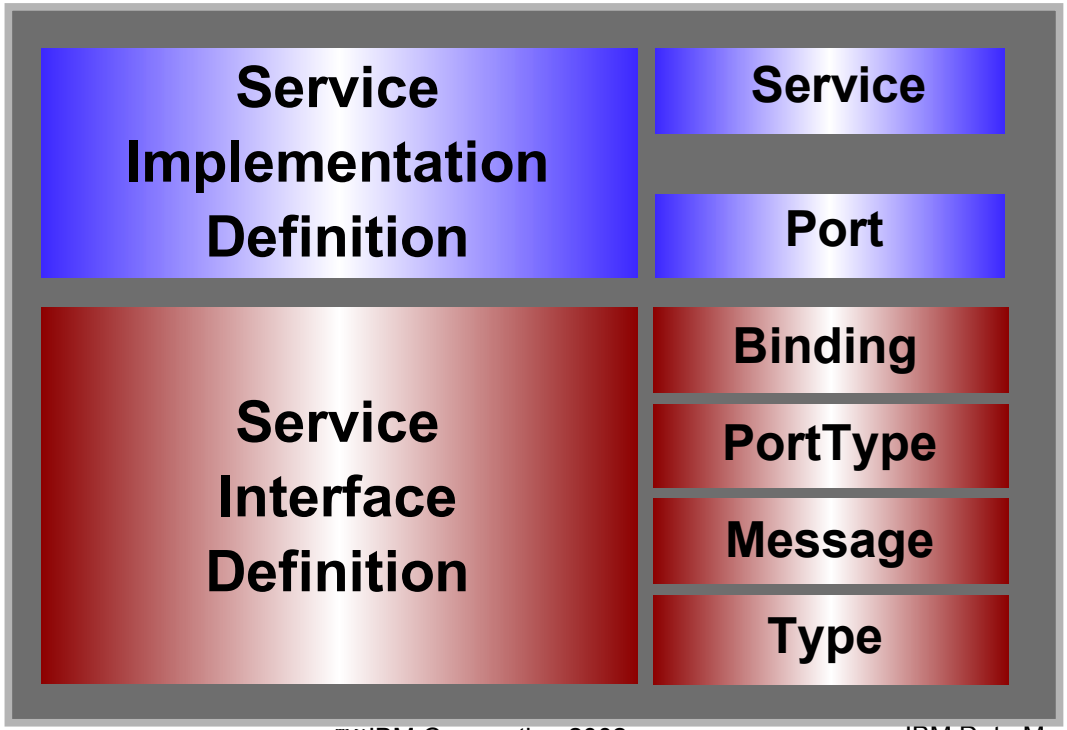
How do we define new web services?

- **Refer to web services that others have defined and agreed upon (tModels).**
- **...but what if I want to publish my own web services**
  - e.g. simple functions useful to others
  - or complete business processes that I offer
- **To make it practical... and easy to integrate to many such services, we need a standard way of describing web services.**

The key to interoperability

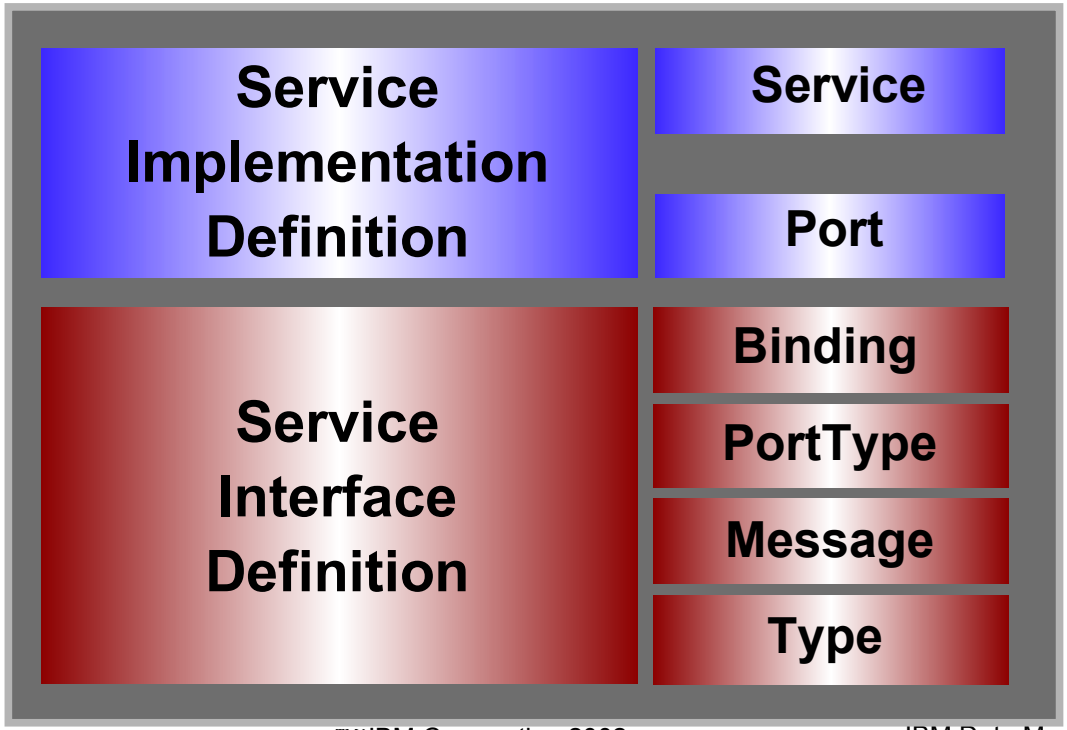
- **How does the Service Requestor know the format of**
  - the expected request message(s)?
  - the response message(s)?

- **By the Service Description (WSDL document)**

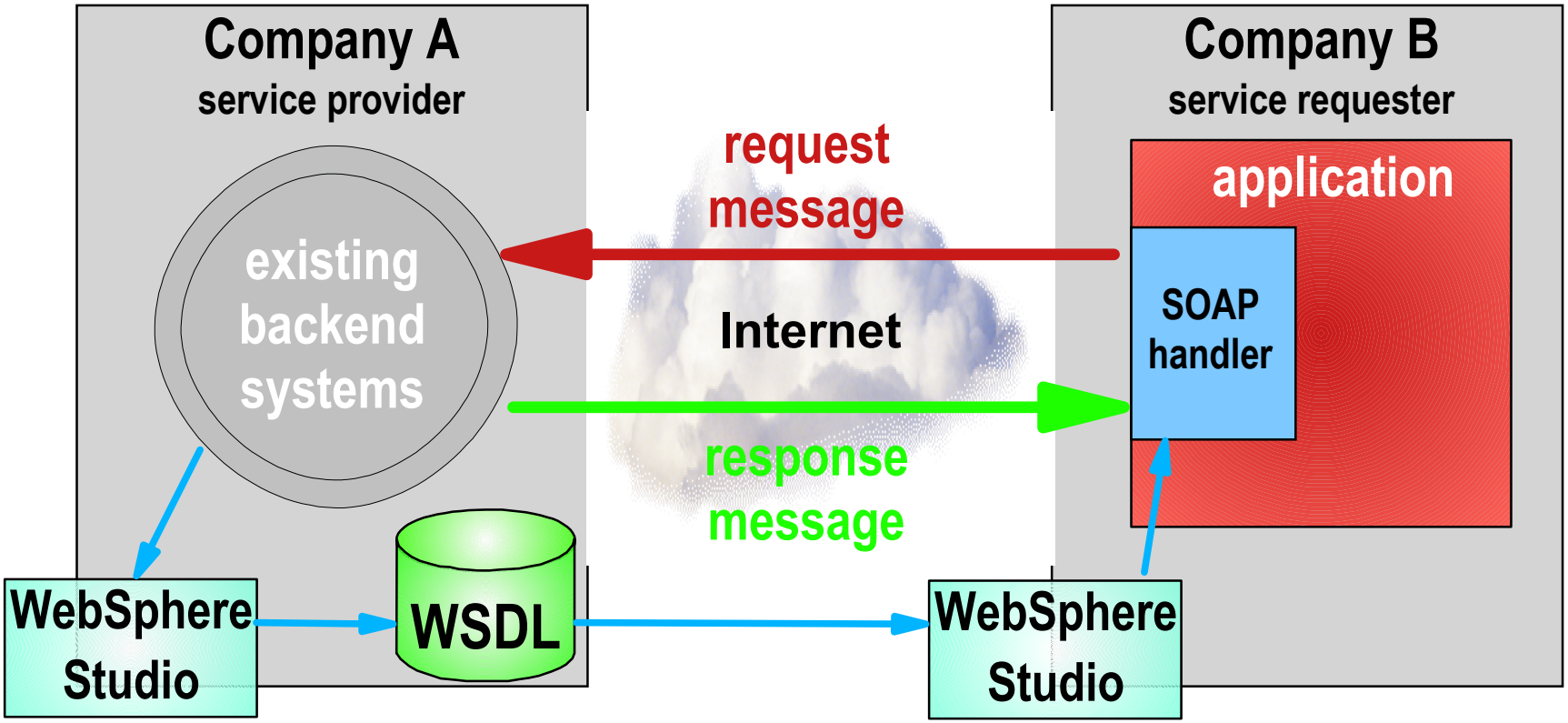


WSDL: Web Services Description Language

- **WSDL describes operational information**
  - where the service is located (service implementation definition)
  - what the service does (service interface definition)
  - machine readable, generated and used by IDEs
  - similar in purpose to IDL, but in XML form



WSDL: Simplifying and Speeding Integration



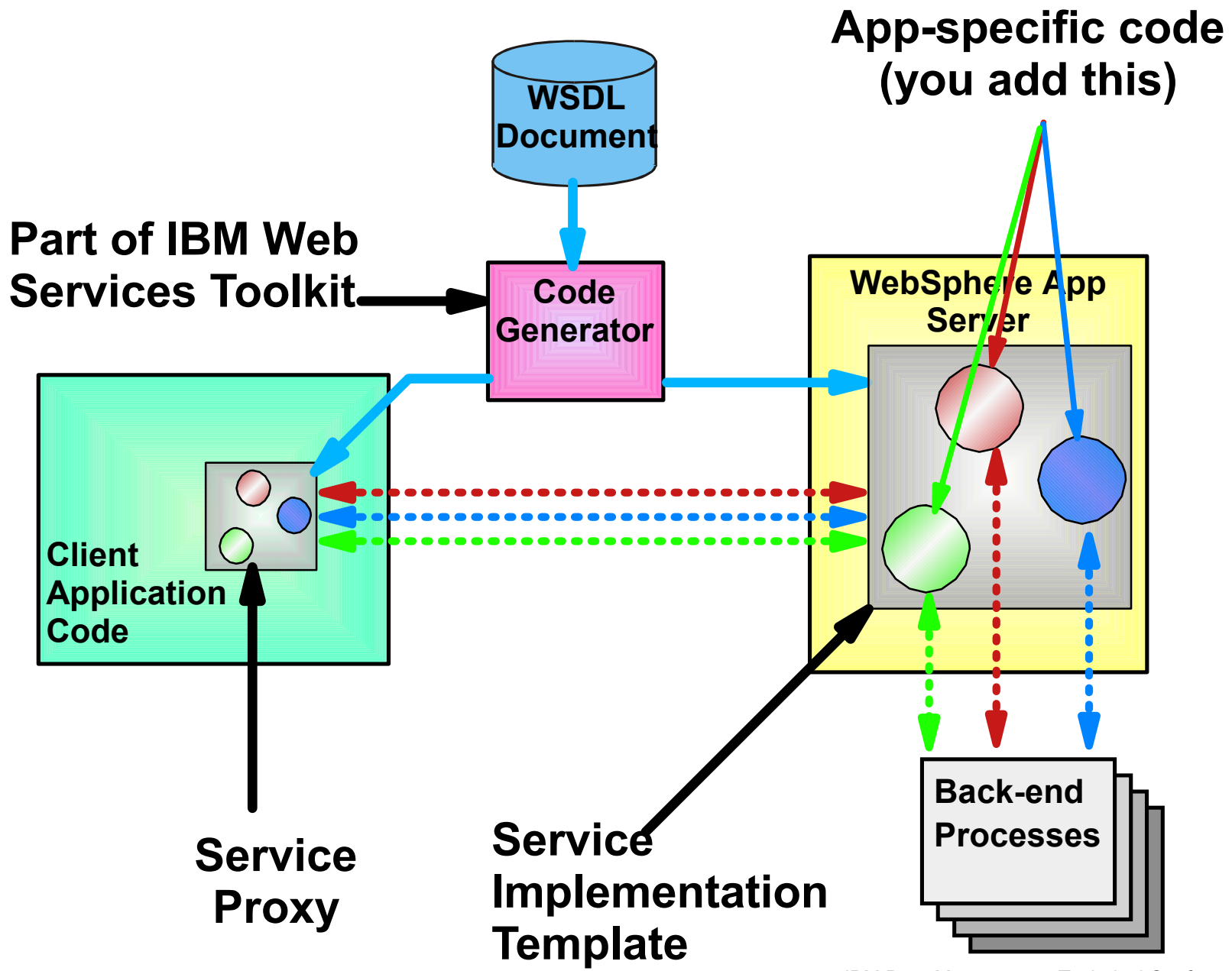
WSDL Description is generated automatically by inspecting SOAP interface layer

Code for handling messages is generated automatically from WSDL description

WSDL Resources

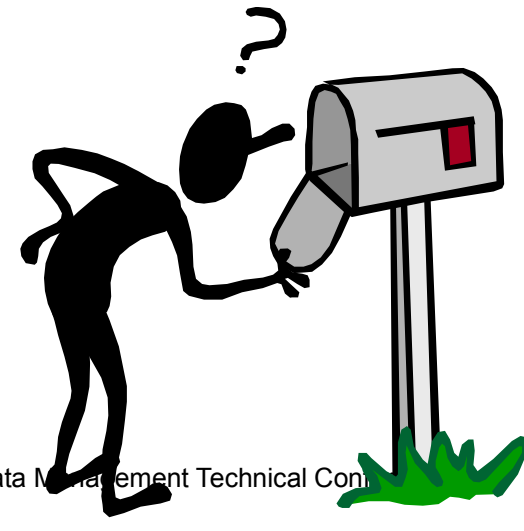
- **WSDL 1.1 Specification**
  - http://w3.org/TR/wsdl
- **WSDL4J**
  - http://oss.software.ibm.com/developerworks/projects/wsdl4j
- **WSDL Toolkit (part of WSTK)**
  - http://ibm.com/alphaworks (look under xml on left)
- **WS Application Developer (beta available soon):**
  - http://ibm.com/software/webservers/studio/preregister.html
- **WSDE (early version of WSAD available now):**
  - http://ibm.com/alphaworks (look under xml on left)
- **Articles and tutorials:**
  - http://ibm.com/developerworks/webservices

WSDL Speeds Implementation



How does the Requestor get the WSDL?

- **What are the ways a requestor an get the WSDL?**
  - WSDL (or its URL) can be emailed to requestor
  - find WSDL for available services at repository sites like xmethods.net or www.salcentral.com
  - ...or use UDDI "find" methods to look it up in the UDDI Business Registry

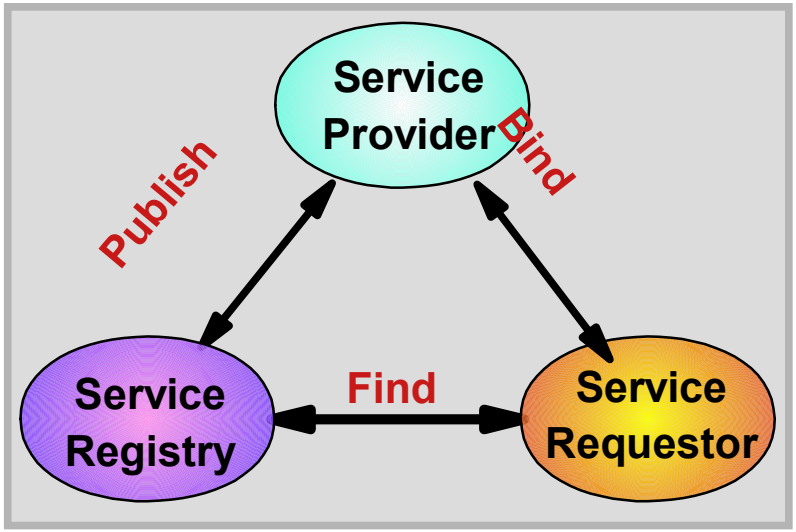


What is UDDI?

- **Universal Description, Discovery, and Integration**
- **A project to speed interoperability and adoption for web services**
  - Standards-based specifications for service description and discovery
- **A set of Internet-based implementations**
  - UDDI Business Registry
  - Interoperating to share registrations
- **Partnership among industry and business leaders**
  - Initiated by IBM, Microsoft, and Ariba
  - 300+ UDDI community members
  - Specification work transferred to OASIS - July 2002
- **UDDI has two pieces:**
  - the UDDI Business Registry (hosts the data)
  - the API and data model (provides access to the data)

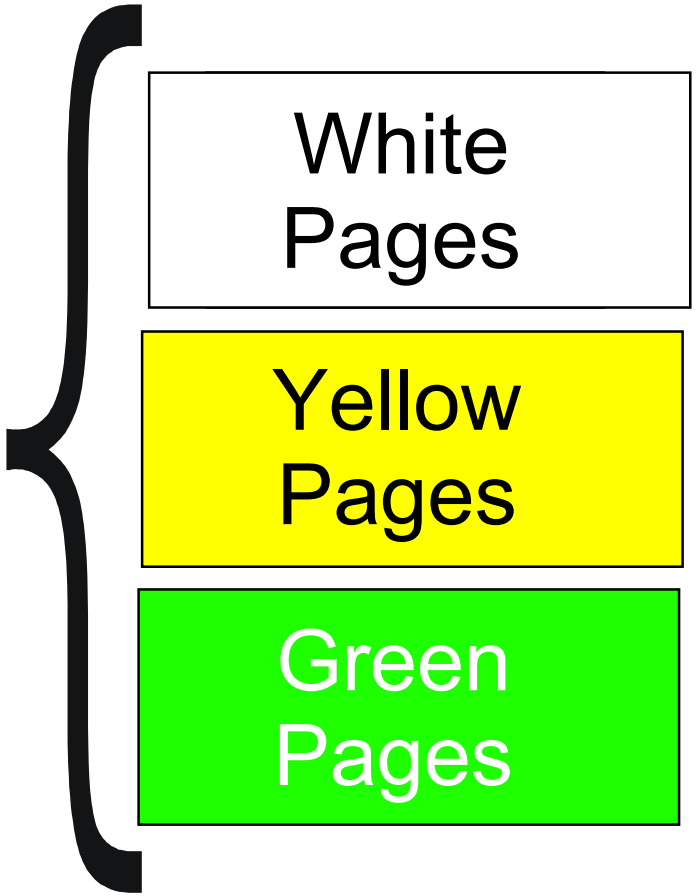
UDDI Roles and Operations

- **Service Registry**
  - provides support for publishing and locating services
  - like telephone yellow pages
- **Service Provider**
  - provides e-business services
  - **PUBLISHES** availability of these services through a registry
- **Service Requestor**
  - **FINDS** required services via the Service Broker
  - **BINDS** to services via Service Provider

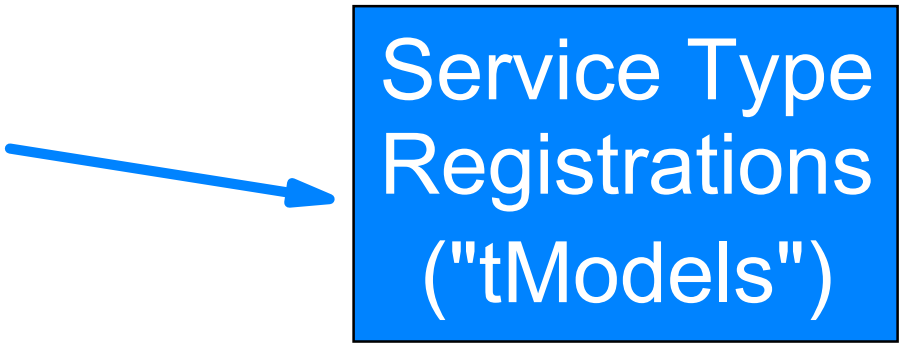


Registry Data

Businesses register public information about themselves

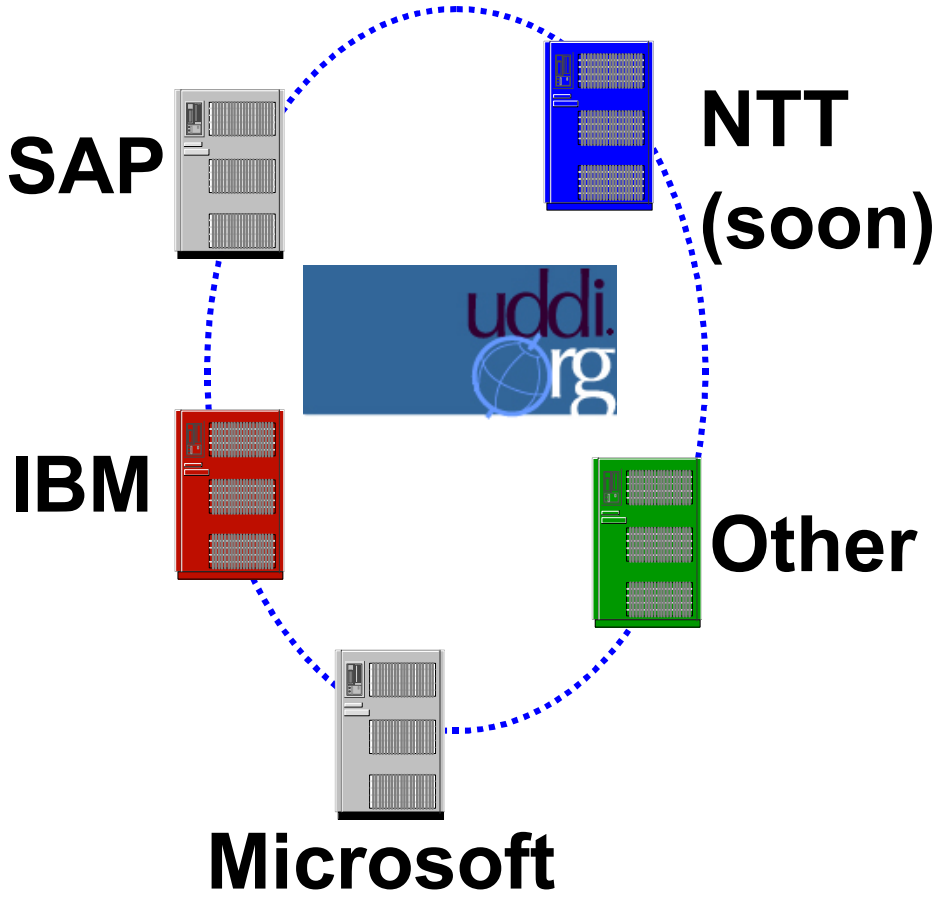


Standards bodies, programmers, businesses register information about their service types

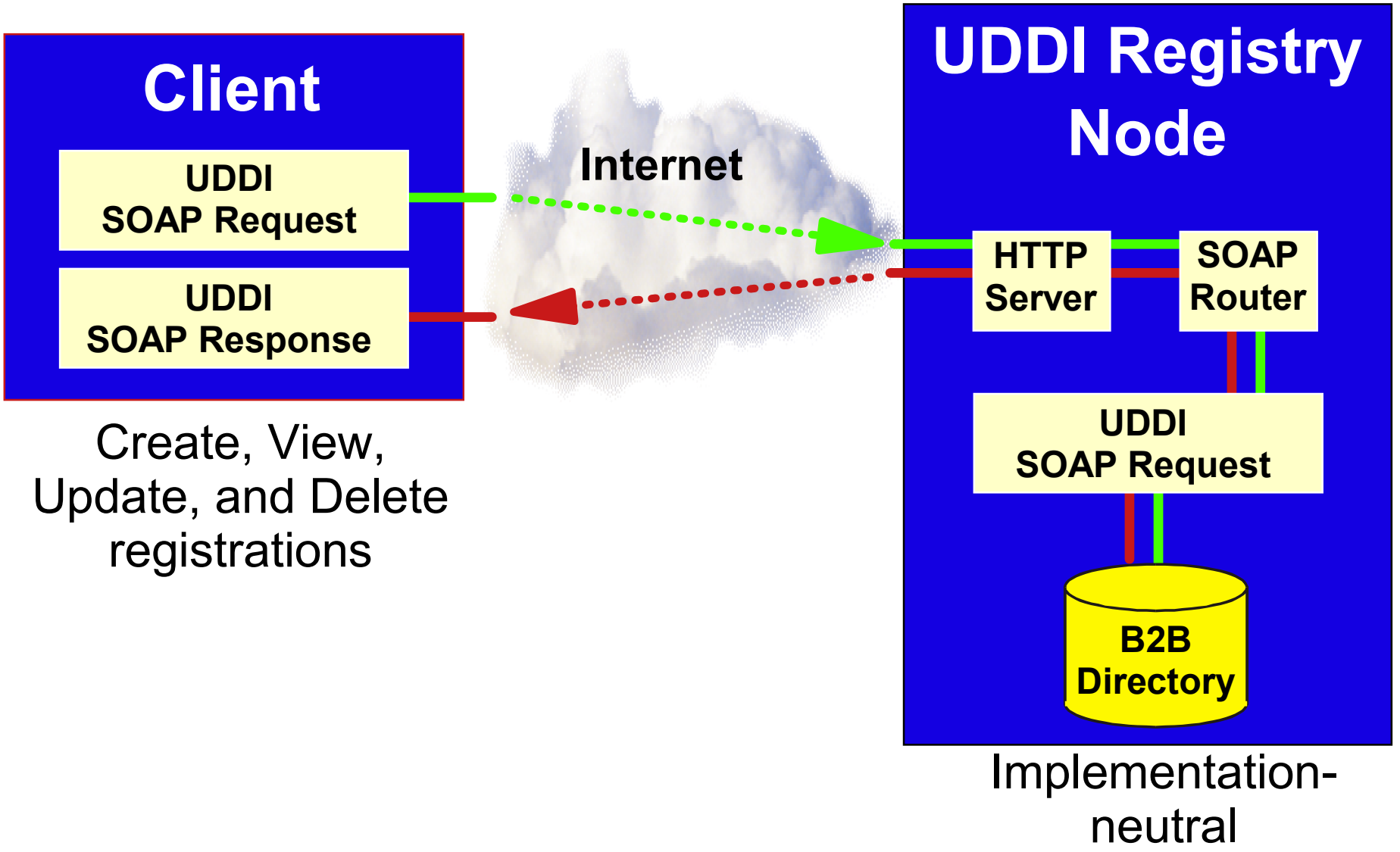


Registry Operation

- **Peer nodes (websites)**
  - Companies register with any node
  - Registrations replicated hourly
  - Complete set of “registered” records available at all nodes
- **Common set of SOAP APIs supported by all nodes**
- **Compliance enforced by business contract**



UDDI and SOAP



Create, View, Update, and Delete registrations

Implementation-neutral

The UDDI Inquiry APIs (SOAP messages)

- **Find things**
  - find_business
  - find_service
  - find_binding
  - find_tModel
  - find_relatedBusinesses*
- **Get Details about things**
  - get_businessDetail
  - get_serviceDetail
  - get_bindingDetail
  - get_tmodelDetail
  - get_registeredInfo
  - get_publisherAssertions*
  - get_assertionStatusReport*

* New with UDDI version 2

The UDDI Publisher's API (SOAP messages)

- **Save things**
  - save_business
  - save_service
  - save_binding
  - save_tModel
  - set_publisherAssertions*
  - add_publisherAssertions*
- **Delete things**
  - delete_business
  - delete_service
  - delete_binding
  - delete_tModel
  - delete_publisherAssertions*
- **Security**
  - get_authToken
  - discard_authToken

* New with UDDI version 2

UDDI Version 2.00 - June 2001

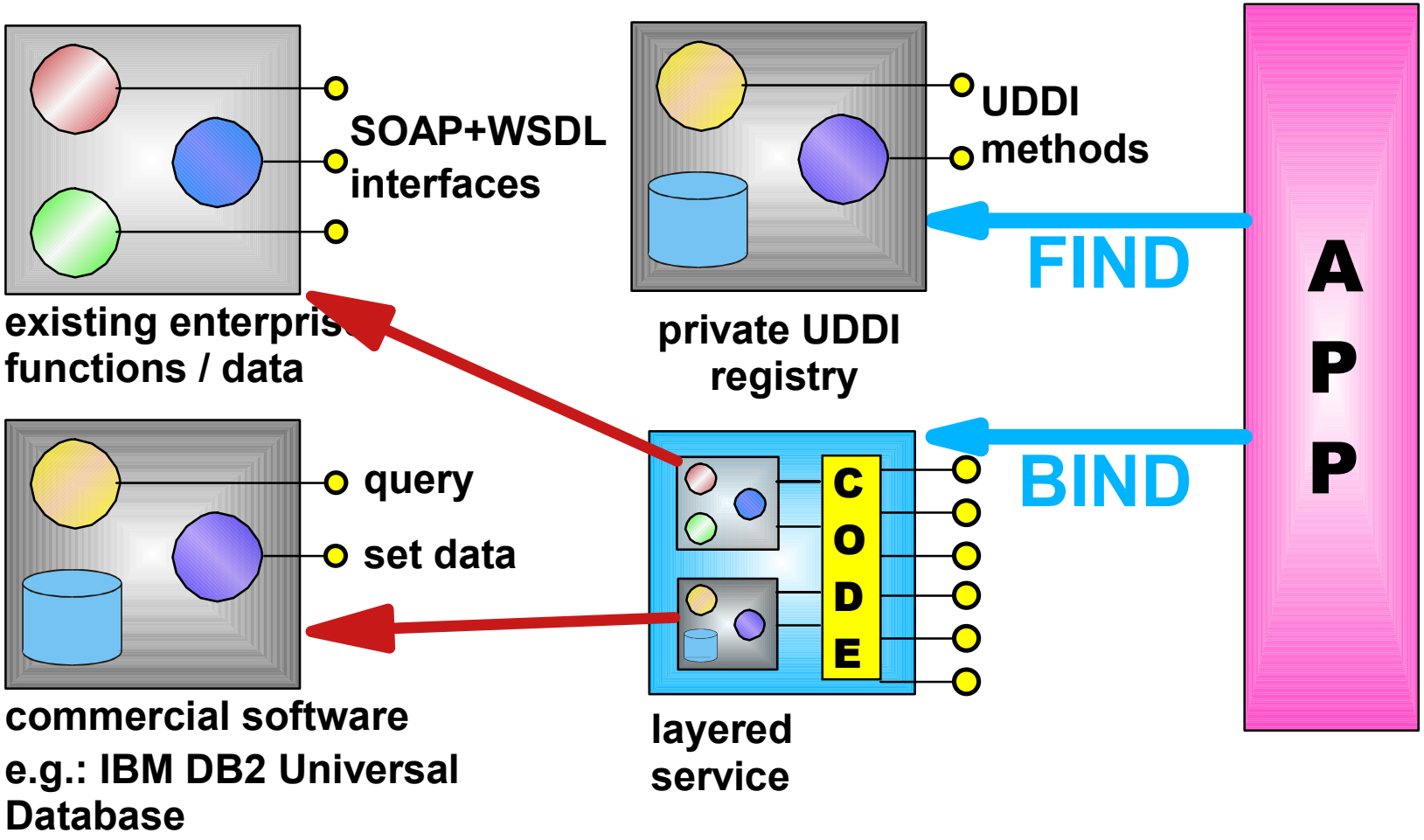
- **Updated specifications on uddi.org**
- **Significant functionality added to UDDI:**
  - Description of Complex Organizations - business units, departments, divisions, and subsidiaries
  - Additional categorization and identifier schemes
    - register 'checked' and 'unchecked' taxonomies
  - Richer searching options: more expressive query parameters, using more fields and complex combinations of fields
  - Better internationalization for describing businesses and services in multiple languages
- **Peer based replication for improved scaling**
- **UDDI version 2.00 beta registries went to "production" July 23, 2002**

UDDI Version 3.00 - July 2002

- **Security**
  - Support for Digital Signatures
- **Multiple Registry Support**
  - Topologies of public and private registries
- **Advanced Data Management**
  - Enhanced search capability
  - Better interpretation of query results
  - More meaningful descriptions of businesses and services
  - Easier management of existing data.
- **Internationalization**
  - Enhanced support for multinational corporations to describe their global operations across international business units
  - Addressing localization of UDDI data and services.

Web Services: Inside the Enterprise

SOAP+WSDL+UDDI is useful for an application or data integration strategy: offers loose coupling and late binding



UDDI4J: an Open-source Java API

- **Open-source Java bindings for UDDI messages**
  - Creates SOAP messages via Java method calls with an API that maps to UDDI message elements
  - Other housekeeping chores to make your UDDI implementation work easier
  - works with any UDDI Registry
- **UDDI4J source and binaries available**
  - oss.software.ibm.com- IBM's open source software site
  - OSI-approved open-source licence
  - version 2 included in IBM WSTK 3.1 (ibm.com/alphaworks)
- **Read Doug Tidwell's "UDDI4J: Matchmaking for Web services" to get started**
  - ibm.com/developerWorks/library/ws-uddi4j.html

UDDI Resources

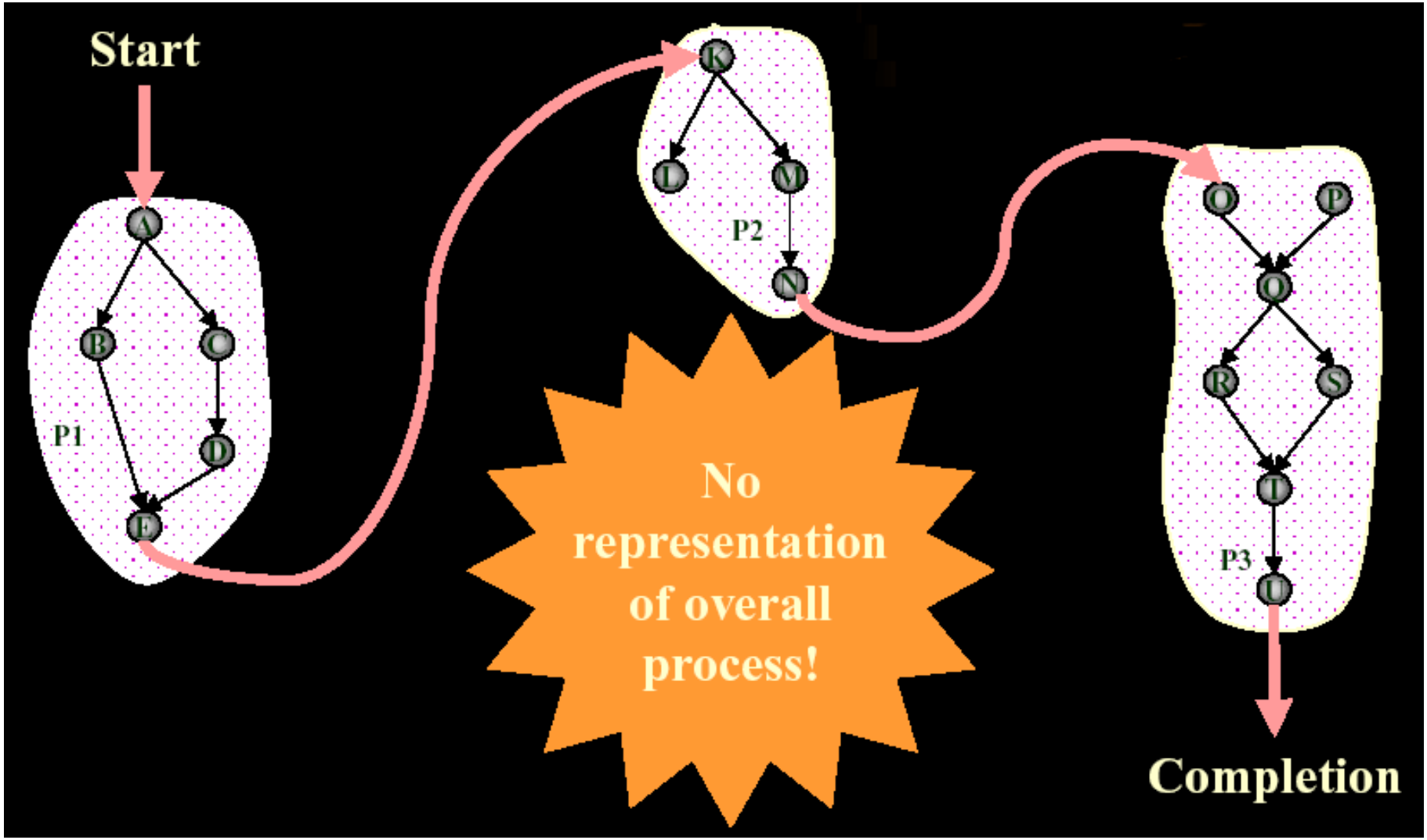
- **White papers, product offerings**
  - http://www.ibm.com/webservices
- **Software:**
  - UDDI4J - open-source Java API to access UDDI
    - code: http://oss.software.ibm.com
  - Private UDDI preview for developers edition
  - http://www7b.software.ibm.com/wsdd/downloads/UDDIregistry.html
  - Web Services ToolKit (WSTK)
    - http://www.alphaworks.ibm.com/tech/webservicestoolkit
- **Articles, tutorials: http://ibm.com/developerworks/webservices**
  - Steve Graham: Role of private UDDI nodes in Web services
    - Part 1: Six species of UDDI
    - Part 2: Private nodes and operator nodes
  - Doug Tidwell: Introduction to UDDI4J
    - ibm.com/developerWorks/library/ws-uddi4j.html

More UDDI.org papers

- White Papers:
  - Executive White Paper
  - Technical White Paper
- Best Practices papers:
  - Using WSDL in a UDDI Registry
  - Providing a Taxonomy for use in UDDI version 2.00

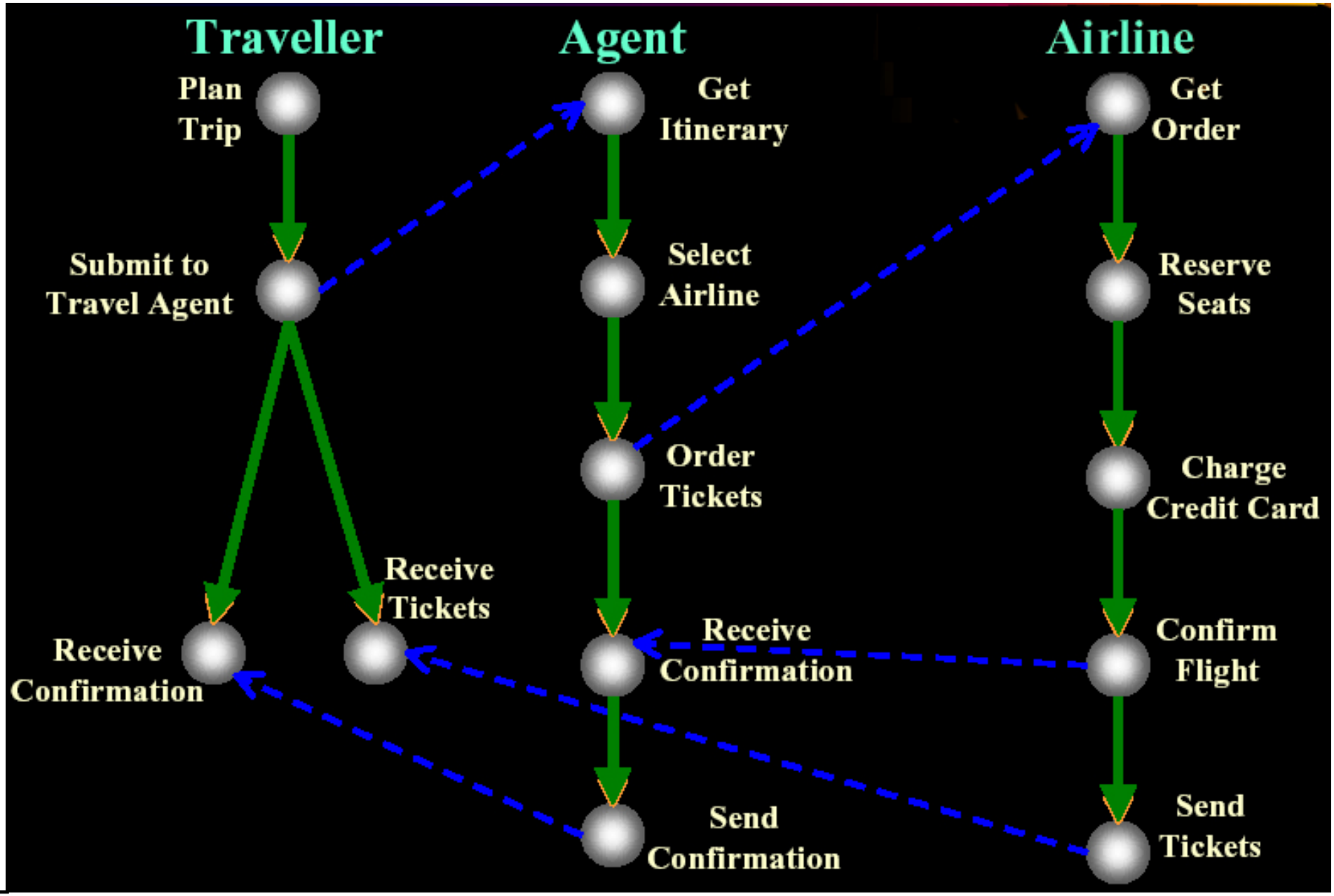
Layered Web Services

With Web Services technology, we can imagine the combination of various business entities teaming to provide a composite, or layered, web service...



IBM®

Layered service example: automated travel management



WSFL: Web Services Flow Language

An XML language to describe Web Services compositions. Two types:

1. Usage pattern of a collection of Web Services
   - describes how to achieve a particular business goal as a business process
   - flow composition, orchestration, or choreography
   - defines the flow of control and data
2. Interaction pattern of a collection of Web Services
   - describes the overall partner interactions
   - no specification of an execution sequence is provided

- **WSFL has extensive support for the recursive composition of services**
  - support for top-down progressive refinement design
  - support for bottom-up aggregation

WSFL Resources

Specification:
ibm.com/software/solutions/webservices/pdf/WSFL.pdf

Introductory articles: visit ibm.com/developerworks/webservices, search for "Snell" for four articles by James Snell ("Web Services Insider" series):

- Introducing WSFL
- Business process modeling with WSFL
- Implementing roles in WSFL
- WSFL and recursive composition

XML Schema

- XML Schema 1.0 (w3.org) provides specifications to allow comprehensive automatic data validation by XML parsers
  - rich type definition for data type validation
  - composite data types
  - also supports most features of DTD (albeit with a different syntax)
- Validation is less important for B2B with a regular partner
  - in some cases you're better turning off validation to get better performance because of the frequency of XML messages
- With Web Services
  - we potentially have very large numbers of partners of varying duration of business
  - XML messages come from all manner of systems - various B2B programs, browser-operated manual entry, maybe hand-coded!
  - thus validation is crucial: need quick determination of whether we can process the data, or need to "return to sender"

One great thing about SOAP...

- By using HTTP port 80 you can send requests right thru the firewall
- This easing of security considerations is really **convenient**

One bad thing about SOAP...

- By using HTTP port 80 you can send requests right thru the firewall
- Some consider this circumvention of security really **dangerous**
  - What it really does is move the security problem
  - If we circumvent the security offered by the firewall, we need to address security in the SOAP handling

Seven Aspects of Security

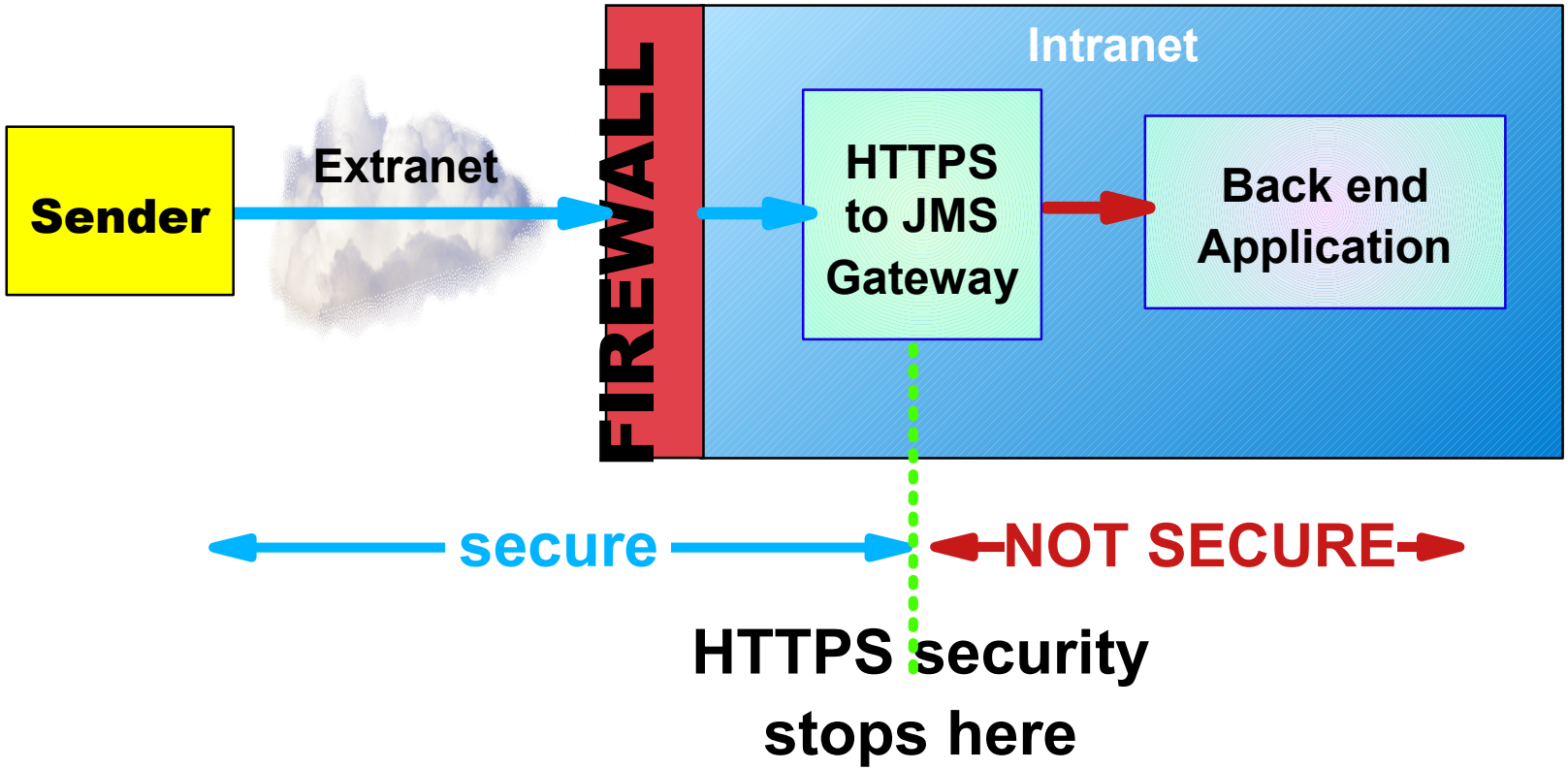
- **identification**: who are you?
- **authentication**: how do I know your identity is true?
- **authorization**: are you allowed to perform this transaction?
- **integrity**: is the data you sent the same as the data I received?
- **privacy**: are we sure that nobody read the data you sent me?
- **auditing**: record of all transactions so we can look for security problems after the fact
- **non-repudiation**: both sender and receiver can prove to a third party that
  - the sender did send the transaction, and
  - the receiver received the identical transaction

What HTTPS/SSL offers (at the protocol level)

- ✓ **identification:** who are you?
- ✓ **authentication:** how do I know your identity is true?
- ✗ **authorization:** are you allowed to perform this transaction?
- ✓ **integrity:** is the data you sent the same as the data I received?
- ✓ **confidentiality:** are we sure that nobody read the data you sent me?
- ✗ **auditing:** record of all transactions so we can look for security problems after the fact
- ✗ **non-repudiation:** both sender and receiver can provide legal proof to a third party that
  - the sender did send the transaction, and
  - the receiver received the identical transaction

Why isn't HTTPS enough?

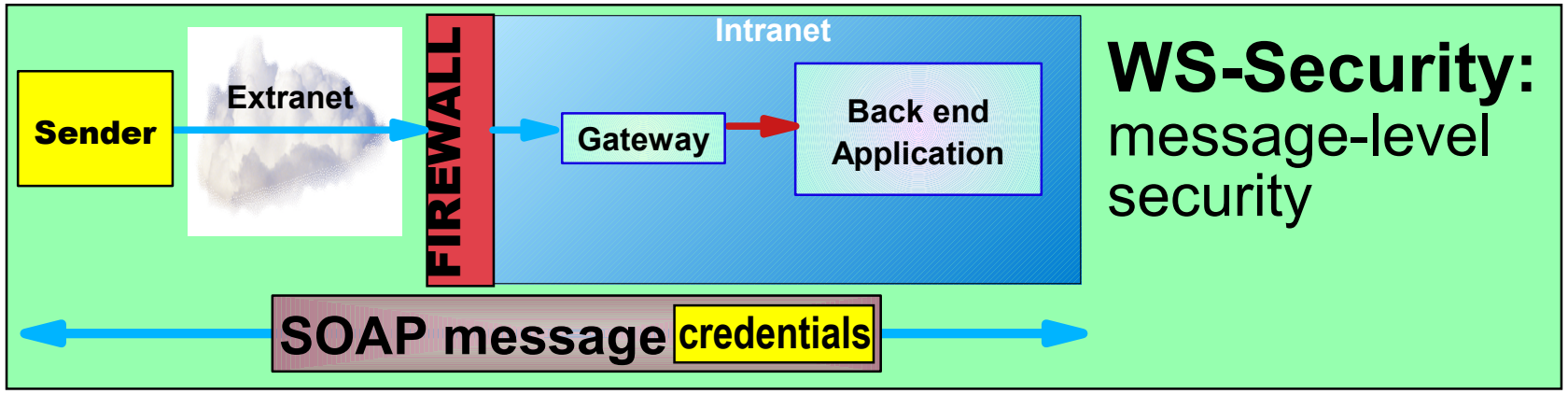
- **Limitation 1**:
  - no authorization, auditing, non-repudiation
- **Limitation 2**: Protocol translation
  - identification, authentication, integrity, confidentiality stop at HTTPS end point



Why isn't HTTPS enough?

- **Limitation 3**: Signature and non-repudiation
  - we want an integrity signature to persist... all the way to a database used for audit trail
  - prove message has not been modified
  - HTTPS has no signature (that can be used for non-repudiation)
- **Limitation 4**: Element-wise encryption
  - decryption is necessary to route the message
    - HTTPS encrypts everything...
    - so you have to decrypt everything to route it
  - we may need certain data (credit card #) to remain encrypted all the way to endpoint

Message-level security



- **Message-level security**
  - credentials persist end-to-end
  - allows non-repudiation
  - element-wise encryption
- **Now interoperable for Web services:**
  - Kerberos, PKI, X.509, HTTPS/SSL
  - W3C XML Signature, Encryption, XKMS
  - OASIS SAML, XACML

WS-Security 1.0

- **Proposed Web Services security standards road map**
  - Announced by IBM, Microsoft and Verisign on April 11, 2002
  - Presents our strategy for addressing security issues within a Web Services environment
  - Consists of one defined specification (WS-Security 1.0), and several planned composable specifications along with example scenarios
  - Addresses same-domain and cross-domain secure messaging
  - The proposed specifications builds upon foundational technologies such as SOAP, WSDL, XML Digital Signatures, XML Encryption and SSL/TLS
  - Brings together formerly incompatible security technologies such as public key infrastructure, Kerberos, and others

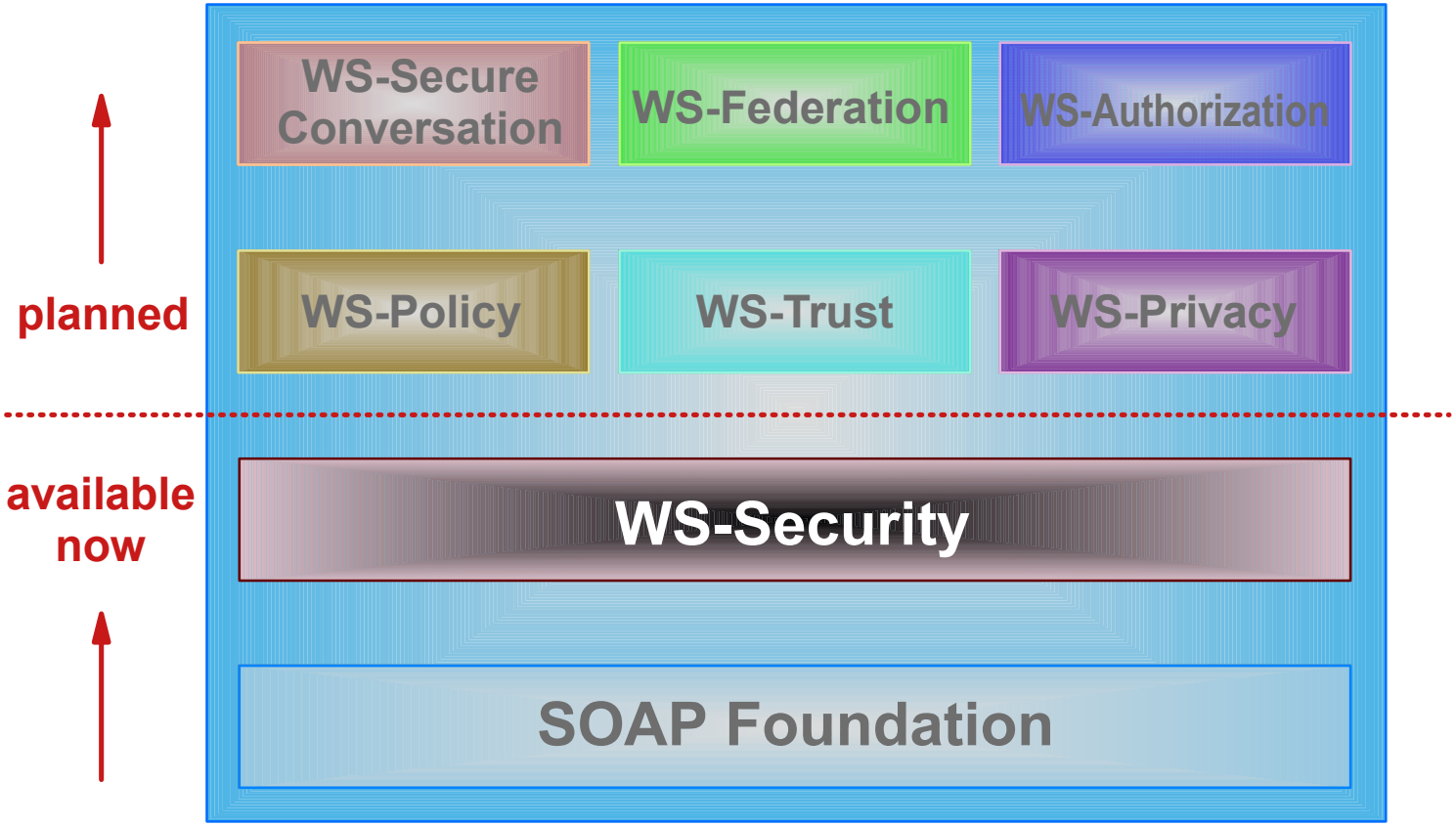
WS-Security Roadmap - Scenarios

To make the issues and solutions discussed in the roadmap as concrete as possible, we include several scenarios that reflect current and anticipated applications of Web services.

- Direct Trust using Username/Password and Transport-Level Security
- Direct Trust using Security Tokens
- Security Token Acquisition
- Firewall Processing
- Issued Security Token
- Enforcing Business Policy
- Privacy
- Smart Clients
- Web Clients
- Mobile Clients
- Enabling Federation
- Validation Service
- Supporting Delegation
- Access Control
- Auditing

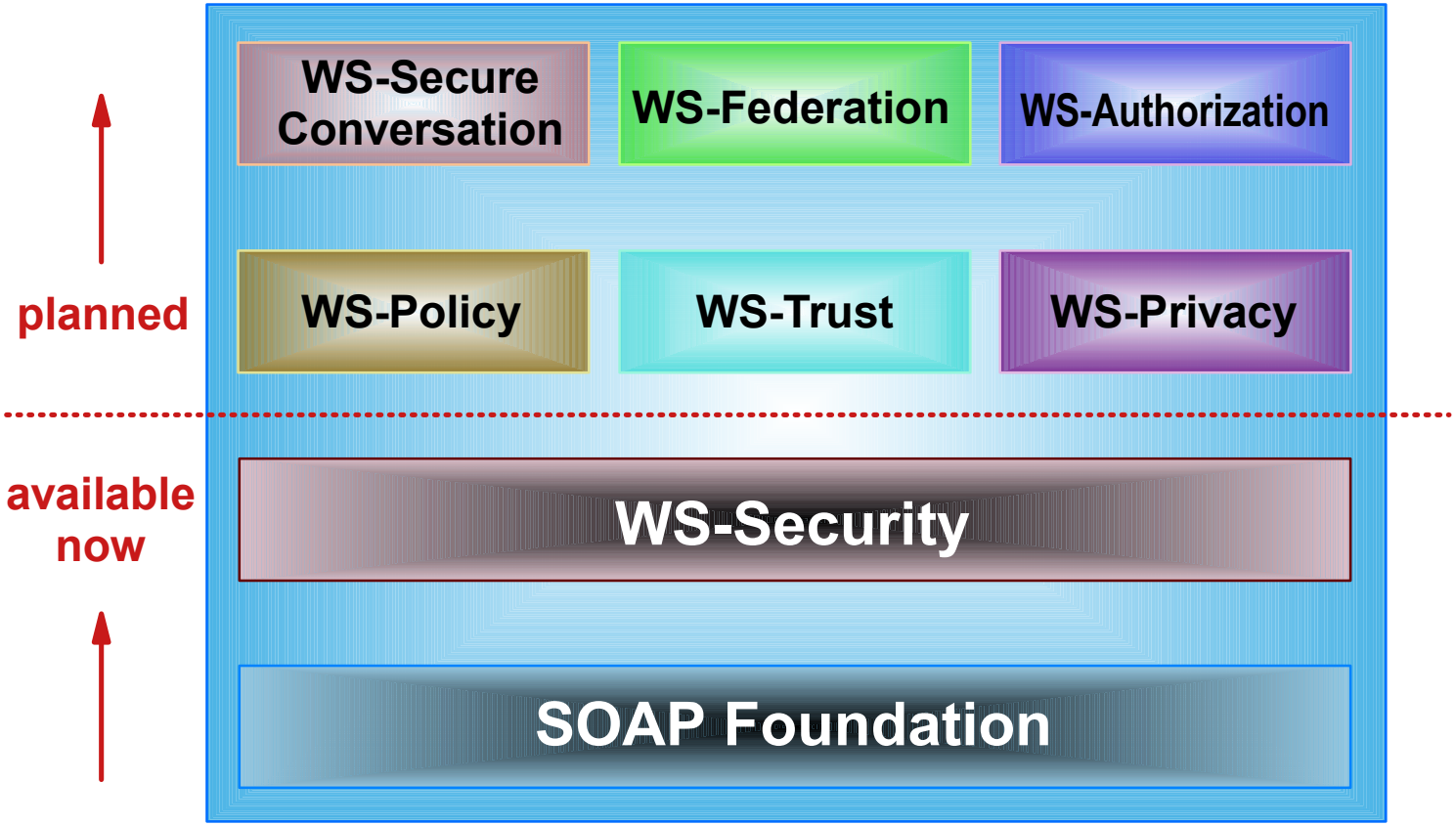
WS-Security 1.0 Specification

Available now. Describes SOAP extensions for secure messaging, provides foundation for other building blocks.



WS-Security - Roadmap

An overview of the plan for a set of comprehensive security specifications intended to secure Web services



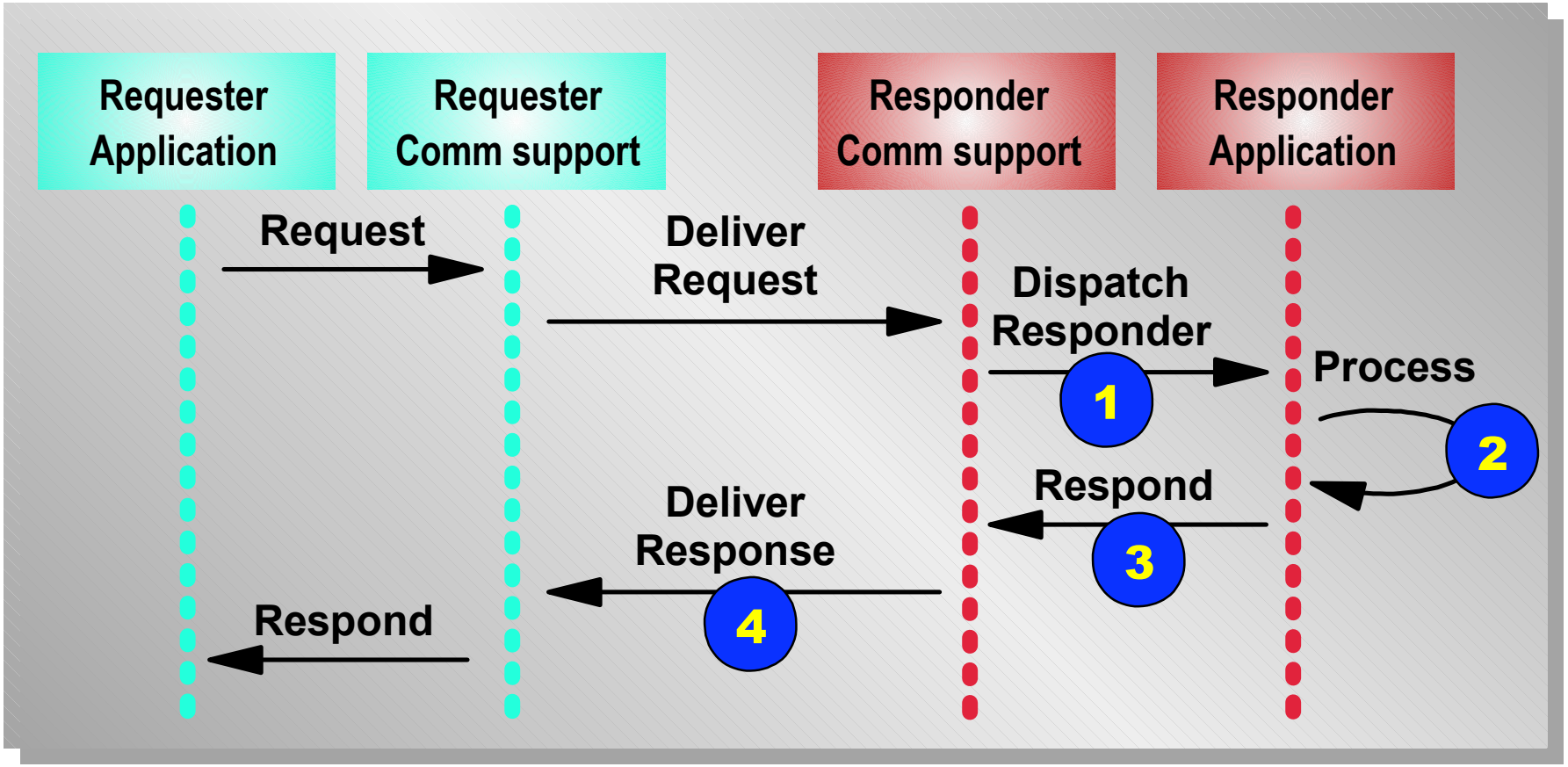
Resources: Security

- **See full-length talk on this subject**
  - "SOAP and Security, Issues and Solutions"
  - ibm.com/developerworks/speakers/colan
  - description and links for related technologies - XML Signature, XML Encryption, SAML, XACML, XKMS, ...
- **Several papers on Web services security**
  - WS-Security 1.0 Specification
  - WS-Security 1.0 Roadmap
  - various whitepapers (Hondo and Snell, etc)
  - all available on ibm.com/developerworks/webservices

Reliability

- **The delivery of messages over a reliable protocol is an essential component for middleware in e-business systems**
- **"Reliable" means:**
  - The message is delivered exactly once, OR
  - We reliably get an "undeliverable" report
- **Fortunately, HTTP is a reliable protocol!**
- **...unless, of course, something goes wrong**

SOAP on HTTP: Failures - status in doubt



1. request not delivered to responder application
2. processed, connection dropped, transaction rolled back
3. processed, reply waiting to be delivered
4. reply lost, responder app doesn't know or can't rollback

Towards reliability on HTTP 1.1

- **The basic technique:**
  - send a message repeatedly until acknowledged
  - message contains same identifier for all copies
  - receiver drops duplicate messages (same identifier)
- **Requirement: reliability in the protocol layer**
  - don't want apps to have to do any of the work
- **Not as easy as it sounds. Solution involves:**
  - persisting at each step along processing
  - sender needs to send the message and update its record of the transmission in a single transaction
  - This is quite a difficult task to perform efficiently.
- **Thus, we have some suggestions for doing this in the HTTP protocol.**

HTTPR protocol

- **A new enhancement to the HTTP protocol proposed by IBM (July 2001)**
  - Small and simple in scope...
- **Provides reliable one-time delivery of a message**
  - It will arrive (or we will know it did not)
  - one message will be delivered
  - duplicate messages will be prevented
- **Of obvious use for SOAP messaging**
- **For more information:**
  - A Primer to HTTPR
    - ibm.com/developerworks/webservices/library/ws-phtt/
  - HTTPR Specification
    - ibm.com/developerworks/webservices/library/ws-phtt/httprspecV2.pdf
  - HTTPR Demo in Web Services Toolkit v3.1
    - ibm.com/alphaworks

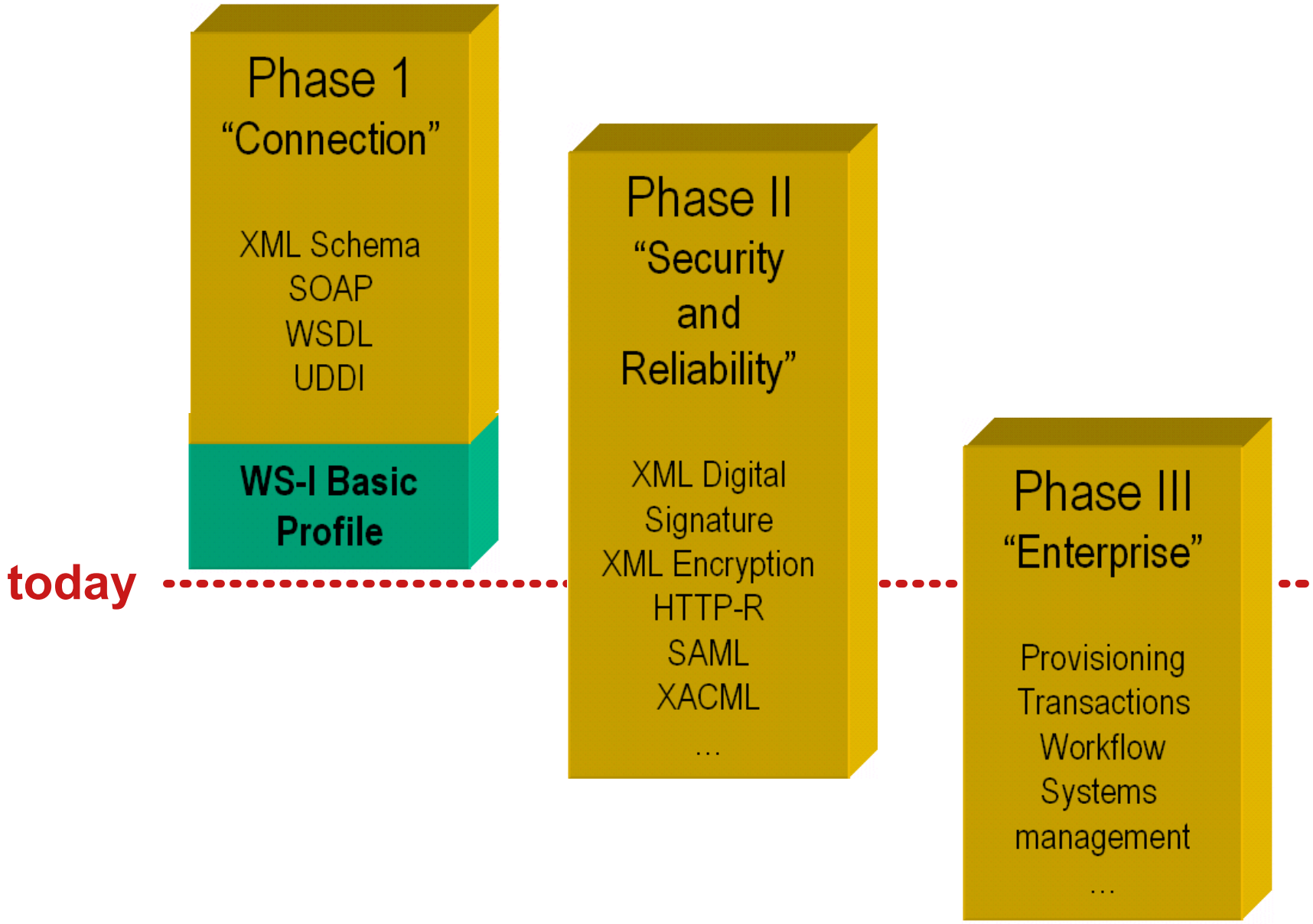
Interoperability

- **Web services standards and technologies enable interoperability**
  - **But, they do not guarantee it.**
- **WS-I.org - the Web Services Interoperability Organization**
  - Formed February 6, 2002
  - IBM, Microsoft, Oracle, HP, Intel, SAP, Fujitsu, Accenture, BEA and 46 other companies
- **WS-I will provide clarity and guidance for**
  - developers who wish to build Web services that will use the underlying standards "correctly" and according to industry conventions
  - CIOs, CTOs and others making investment decisions who need to understand when tools, runtimes, and Web services themselves are compatible.
- **WS-I membership currently at 125+**

WS-I.org

- **Deliverables**:
  - **Profiles** - named groups of specifications at given version levels with conventions about how they work together
  - **Implementation Scenarios** - based on customer requirements
  - **Common or best practices**
  - **Testing software**
  - **Testing materials**

Specifications and Standards



IBM and Web Services

IBM's Web Services Strategy

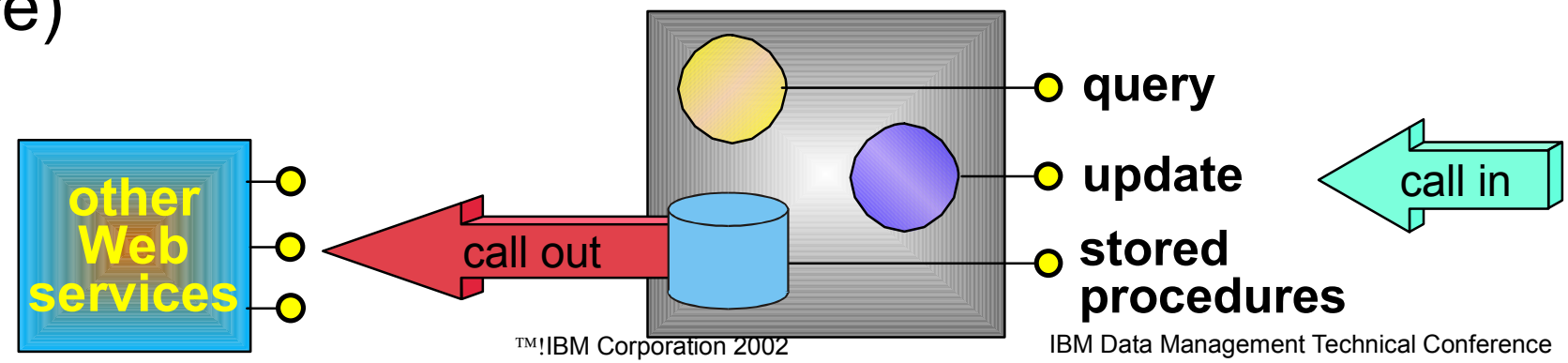
Mission: Deliver Web Services solutions that will help our customers and business partners build, deploy, and manage e-business applications.

We are doing this by

- **Ensuring strong, open standards**
- **Enabling our entire product line for SOAP, UDDI, WSDL, and emerging Web Services technologies**
- **Building e-business solutions**

DB2 Universal Database / XML Extender

- **XML Extender:**
  - XML import and decompose across tables
  - XML export and recomposition
  - XPath to SQL query transformation
  - flexible configuration using XML-syntax Document Access Definition (DAD)
- **Web Services features:**
  - query and update functions available via Web services
  - stored procedures called as Web services
  - stored procedures call out to other Web services (future)



DB2 Web Services

- **Document Access Definition Extension (DADX) runtime:**
  - retrieveXML
  - storeXML
  - SELECT
  - INSERT/UPDATE/DELETE
  - CALL
- **XML tools:**
  - XML-RDB Mapper for Document Access Definition (DAD)
  - SQL Builder
  - SQL to XML Wizard (DADX)

Web Services in WebSphere Application Server 4.0

- **WAS4 is the industry's premier production-ready Web app server for deploying Web Services solutions for dynamic e-business**

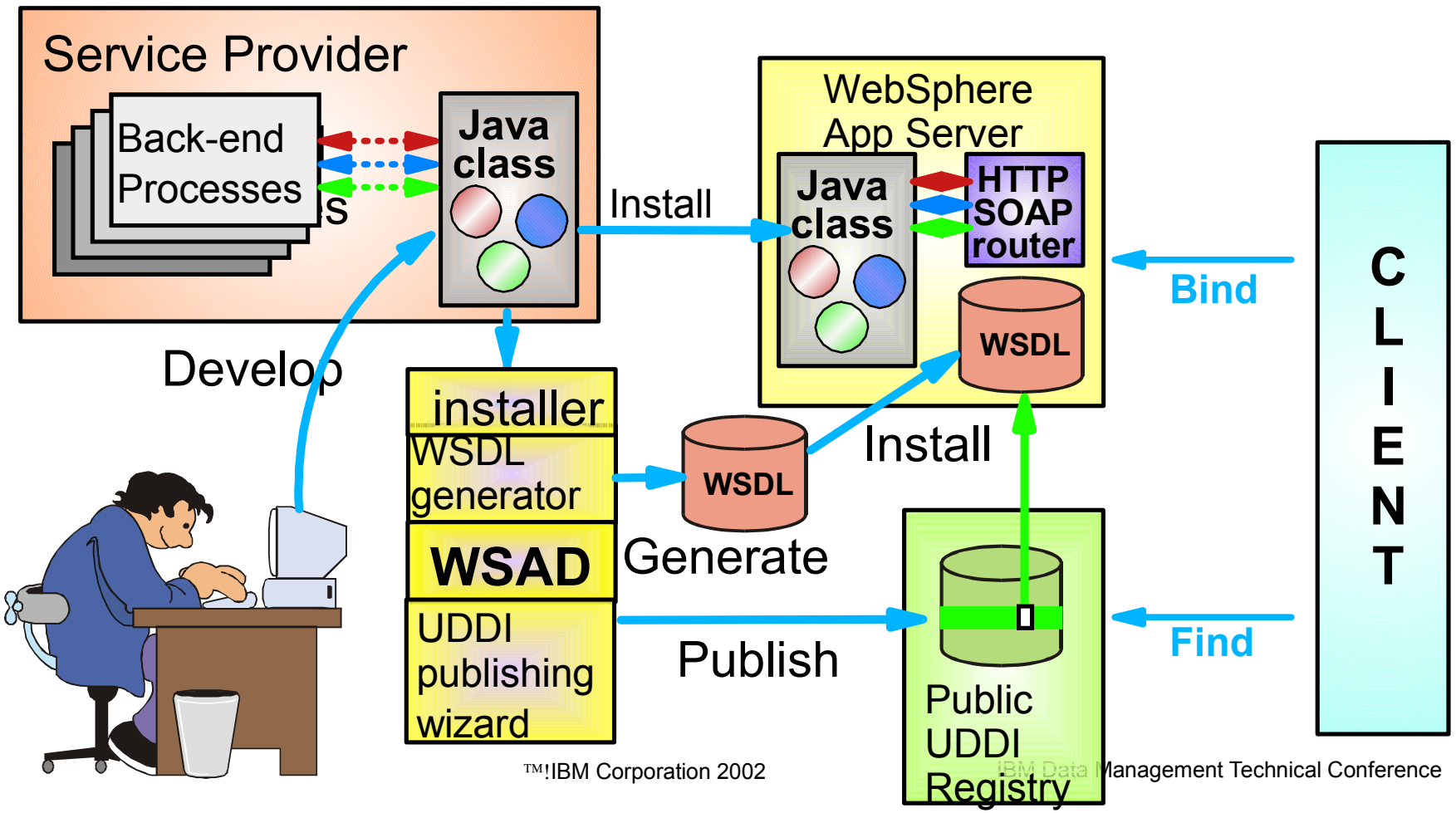
- **Integrated support for Web Services**
  - SOAP
  - UDDI - Universal Description, Discovery, Integration
  - WSDL - Web Services Description Language
  - enables powerful interoperability between Web Services and J2EE applications
- **Security:**
  - HTTPS support
  - Implementations of XML Signature and Encryption

WebSphere Studio Application Developer

- **Extensible development environment for**
  - XML development
  - Web Services Development
- **Built on Eclipse open-source tooling platform**
  - add third-party tools, or write your own!
- **Tight integration with WebSphere App Server**
- **For Web Services:**
  - a set of tools to speed the deployment of a Web service
  - a set of tools to help you find and speed the integration of a Web service

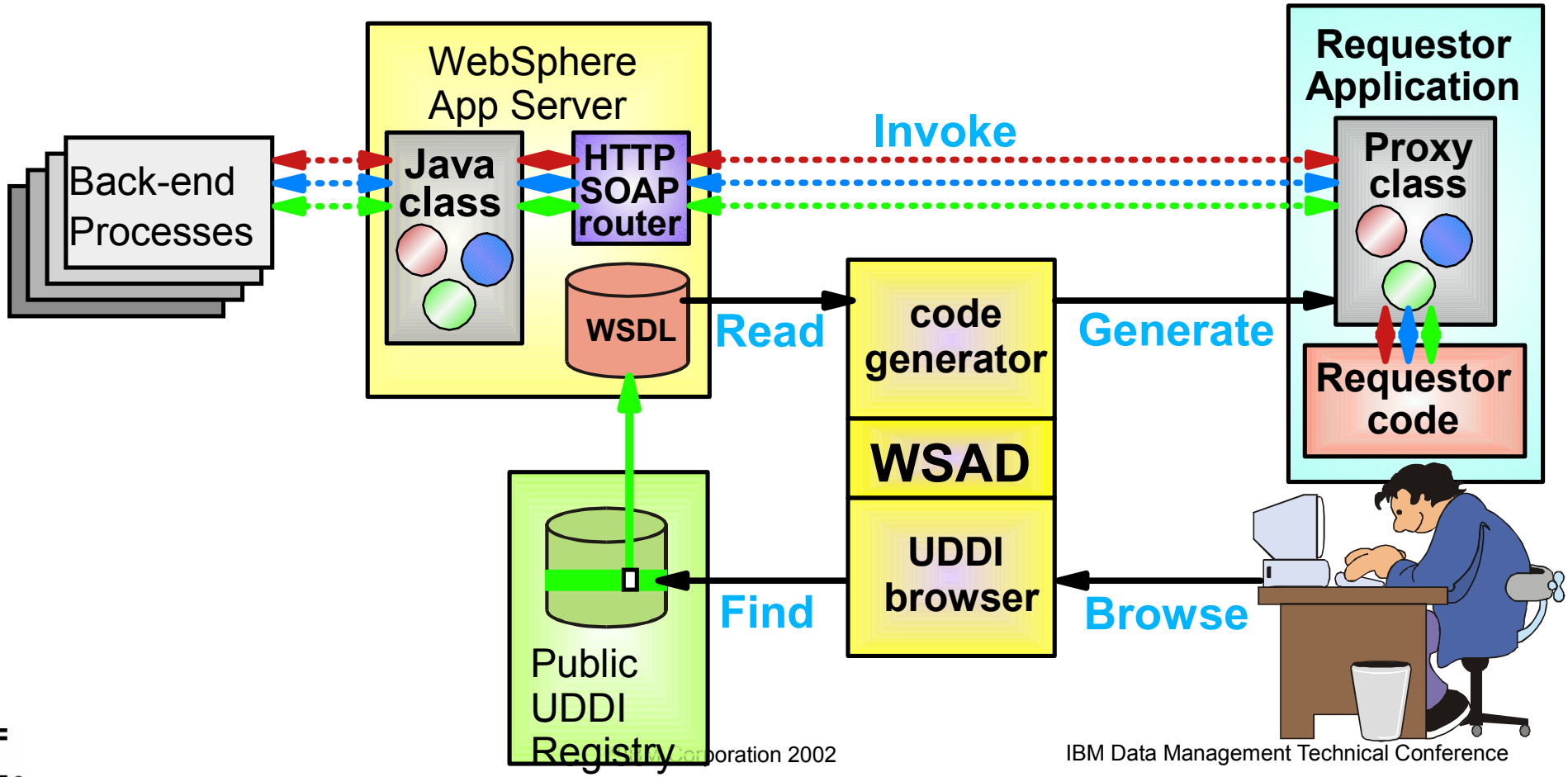
WebSphere Studio App Developer: speeding deployment of Web Services

1. You **develop** a Java class for the service provider to be deployed
2. WSAD **generates** a service description by introspecting your class
3. WSAD **installs** code and WSDL description on the server
4. WSAD wizard **publishes** the availability of the service to UDDI
5. Client **finds** your service via UDDI then **binds** to your code



WebSphere Studio App Developer: speeding integration of Web Services

1. You use UDDI browser in WSAD to **find** the service you want
2. WSAD **reads** the service description and sets up environment
3. WSAD **generates** a Web service proxy class for local use
4. You **call methods** on the service proxy class just like local code
5. Service proxy class **invokes** the service for you via SOAP messages



Web Services in other IBM Products

- **WebSphere MQ Series**
  - technology preview of SOAP in WSTK 2.3
  - integrated into next release of the product
- **Lotus:**
  - enable Domino Services as Web Services
  - incorporate SOAP interfaces, XML-based messaging
  - other products will explore web services features for collaborative products like instant messaging
- **Tivoli:**
  - management and security for Web Services deployment

jStart Engagement Model

Business Qualification

Project Definition

Project Readiness

Customer Commitment

Solution Building

We use a "jumpstart" approach to help customers successfully build e-business solutions using XML and Web Services, starting with education and ending with services.

Engage → Deploy → Promote

ibm.com/software/jstart

jstart@us.ibm.com

IBM alphaWorks

http://ibm.com/alphaWorks

- **Hundreds of tools for Web Services, XML, Java**
  - early versions of features that may be in products
  - some are solid production-code (XML4J, LotusXSL)
  - some are experimental, prototypes
  - free download and use

- **Some recent Web Services downloads:**
  - Web Services Toolkit 3.0 and demos
  - Web Services Hosting Technology
  - Web Services Process Management Toolkit
  - Web Services Invocation Framework
  - Web Services Gateway
  - WSDL Toolkit

IBM's Web Services Toolkit version 3.1

- **An implementation of the Web Services architecture for creating, locating and invoking web services**
  - **New!** WS-Security implementation
  - Utility business services
  - HTTPR Demo
  - Integrated W3C Digital Signatures and Encryption
  - XKMS prototype
  - WSDL Toolkit
  - IBM MQSeries transport for SOAP (technology preview)
  - UDDI4J -- support for UDDI version 2
  - UDDI4B (UDDI for Browser plugin)
  - COM object support
  - Lotus Domino enablement kit
- **Needs only a JDK to run**
- **Versions available for Windows and Linux**
- **Available for free download from http://ibm.com/alphaworks**

UDDI4J: an Open-source Java API

- **Open-source Java bindings for UDDI messages**
  - Creates SOAP messages via Java method calls with same "API" as UDDI messages
  - Other housekeeping chores to make your UDDI implementation work easier
- **Read Doug Tidwell's "UDDI4J: Matchmaking for Web services" to get started**
  - ibm.com/developerWorks/library/ws-uddi4j.html
- **UDDI4J source and binaries available**
  - oss.software.ibm.com
    (IBM's open source software site)
  - OSI-approved open-source licence
  - originally part of the WSTK; now available on our open source site
  - now supports UDDI version 2 API

WSDL4J: an Open-source Java API

- **Allows the creation, representation, and manipulatio of WSDL documents describing services**
- **Service descriptions can be treated by a client in a uniform manner, regardless of the origin of the description:**
  - parsing a WSDL document
  - constructed programmatically by direct invocation of the AP
  - built using information provided by a user via a command-l or graphical interface
  - built using information retrieved from a network source
- **Reference implementation of JSR110**
- **Visit:**
  - oss.software.ibm.com/developerworks/projects/wsdl4j

XML Security Suite

- Complete implementation of W3C specifications:
  - Digital Security 1.0 proposed recommendation
  - Encryption 1.0 working draft
  - Experimental access control implementation
- Integrated into Websphere App Server 4.0
- Included in Web Services Toolkit

Schema support in XML Parsers

- **Apache Xerces-J 2.0.1**
  - Open Source from xml.apache.org, free redistribution
  - Developed by IBM and others in the open source community
  - The same code we use in Websphere Application Server and other IBM products
  - Complete, conformant implementation of XML Schema 1.0 recommendation from W3C
- **Apache Xerces-C 1.7.0**
  - XML Parser for C++ (portable subset)
  - Partial implementation of XML Schema 1.0

developerWorks

Web services zone

Web services
- Tools and products
- Education
- Articles
- FAQs
- News
- Forums
- Events
- Related links

More technologies
Java Go

Product domains
IBM developer Go

Web service invocation sans SOAP

Web Services Invocation Framework creates an interface that is independent of the transport mechanism used by a service. It allows the developer to invoke Web services by using the Web Services Description Language directly, thus completely hiding the transport layer interactions. (Articles)

- **Using WSDL in a UDDI registry, Part 2**: We continue this series with an introspective on the various programming scenarios of using WSDL in a UDDI registry environment. (Articles)
- **Using WSDL in a UDDI registry, Part 1**: This paper discusses the specifics of dealing with WSDL in UDDI registry environments to allow services to search for each other. It expands the guidelines given by UDDI.org on how WSDL can work with UDDI with more concrete detail. (Articles)
- **SOAP security extensions: digital signature**: Satoshi Hada explains how Digital Signatures, SSL, and SOAP can work together in a cohesive, complementary, and standards-based system. (Articles)
- **Web services and short messaging**: This is a case study on the development of a Web services-enabled implementation of the Short messaging service used in cellphones, 2-way pagers, and wireless PDAs. (Articles)
- **Web services and XML technologies CD**: This recently updated CD offers articles, tutorials and tools to keep you up to date with the latest XML and Web services developments from developerWorks and alphaWorks. (Articles)

dW theme: Code reuse: Be the master of your code.

Discussion forums

- **Web services technical**: Get answers to questions on designing, implementing, and managing vendor-independent Web services. (Forums)

Columns

Web services architect, Part 3 by Dan Gisolfi
The Web services architect examines the structural differences between Web services and CORBA.

The Web services insider, Part 9 by James Snell and Maryann Hondo
The Insider defines the questions we should be asking about Web services security. **New!**

The Web services (r)evolution, Part 4 by Graham Glass
In this installment, Graham explains WSDL, how to describe the core properties of a Web service, and introduces tools that leverage WSDL to accelerate your development process.

The Python Web services developer, Part 4 by Uche Ogbuji and Mike Olson
This conclusion to the series on Web services software repository explains how WSDL plays its part in describing the packages.

September 26, 2001

- Tutorials
- Submit content
- Find industry jobs
- Ask the experts
- IBM developers store

News

More news

- Firms wait for UDDI proof (ZDNet UK)
- IBM serves up Web services technology (EarthWeb.com)
- Flamenco Networks offers plug-and-play Web services (InfoWorld)

alphaWorks code

ibm.com/webservices

IBM

Search

Home | Products & services | Support & downloads | My account

Select a country

IBM software

Dynamic e-business

About

Offerings

News

Partners

Developer links

Resources

Case studies

Glossary

Events

Get started

Dynamic e-business

The next stage of e-business and Web services

→ About dynamic e-business
Learn about dynamic e-business and how IBM is providing the software and services you need to build Web services, how you can start simple and with what you have, getting benefits right away. And with IBM you can do this in a single, open, integrated platform.

→ Offerings
Be a part of the e-business evolution. Work with IBM to build and deploy Web services in your business. Read about current and future product offerings from our different brands.

→ News
Read what's being written about dynamic e-business.

→ Partners
Learn about our partners and the Web services we are building together.

→ Developer links
Tools, resources and information to help developers move forward with dynamic e-business.

→ Resources
White papers, product roadmaps, FAQs, thought leadership papers and more to help you take the next step.

→ Case studies
See how dynamic e-business is changing the marketplace. Read about real world examples of our solutions at work.

→ Get started
Take the next step in working with the jStart team to build, deploy and manage Web services in your business.

What's new

Lotus to offer Web services kit to developers

Web services battle at centerstage at JavaOne

Highlights from the dynamic e-business announcement

Demo: Wrap an existing application with Web Services

Register

Register now

About IBM | Privacy | Legal | Contact

IBM

How our customers are using Web services today

- **Customers are already realizing the cost-savings and flexibility offered by Web services**
  - visit ibm.com/software/jstart for some case studies
- **There are two basic categories:**
  - EAI (Web services inside the firewall)
  - Improving existing business partner integration (B2B)
- **Public UDDI is seen as a future growth**
  - focus is on current business partnerships
  - advertising on UDDI for new business comes later
  - Private UDDI is hot now

Cost savings, Flexibility, Agility

- **Web services reduces costs**
  - "By 2005, the aggressive use of Web services will drive a 30% increase in the efficiency of IT development projects"
- **Flexibility and agility in EAI**
  - "more than 40 percent of enterprises' first experience with Web services will be an internal deployment...enterprises will begin to realize immediate benefits"
- **Improving efficiency in existing B2B partnerships**
  - "Web services will offer some B2B benefits early on as well...established trading partners will seek to drive down the costs of interconnection"
  - we have customers doing this now
- **These cost savings are not speculative investments!**
  - in a recovering economy, you can still make more money by reducing your costs

All quotes on this page: Gartner Inc, "The Hype Is Right: Web Services Will Deliver Immediate Benefits", 9 Oct 2001

Web Services: Summary

- **Software evolution, Business revolution**
  - leverage existing software as highly-integratable objects
  - integrate systems internally, or with business partners
  - new business opportunities abound
- **Open standards is a requirement**
  - Web Services build on existing standards
  - IBM leads the industry in development of new standards
- **Get started**
  - DB2 has enablement today, and more on the way
  - WebSphere 4.0 fully supports Web Services applications
  - IBM Software products will continue to release Web Services features
  - Web Services Toolkit, Development Environment available now
  - jStart Web Services team skilled at helping your development team with a limited-scope project

Questions?

George Zagelow - zagelow@us.ibm.com