SOA Concepts

Service Oriented Architecture
Johns-Hopkins University

Lecture 2 Goals

To learn the basic concepts behind SOA

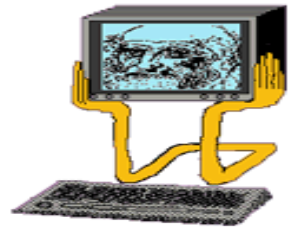
- The roots of SOA: the history from XML to SOA, and the continuing evolution of SOA
- Understand the primitive SOA framework: services, service descriptions and Messaging
- What is an IT architecture?
- What is a distributed architecture?
- Understand the paradigm behind SOAP messaging
- Learn to build a consumer for a web service

Session #2 Today's Agenda

Today's Lecture

- Introducing SOA (Ch 3) continued from lecture 1
- Evolution of SOA (Ch 4)

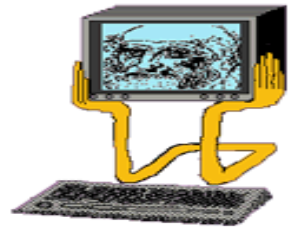
Ch 3 Introducing SOA (review)

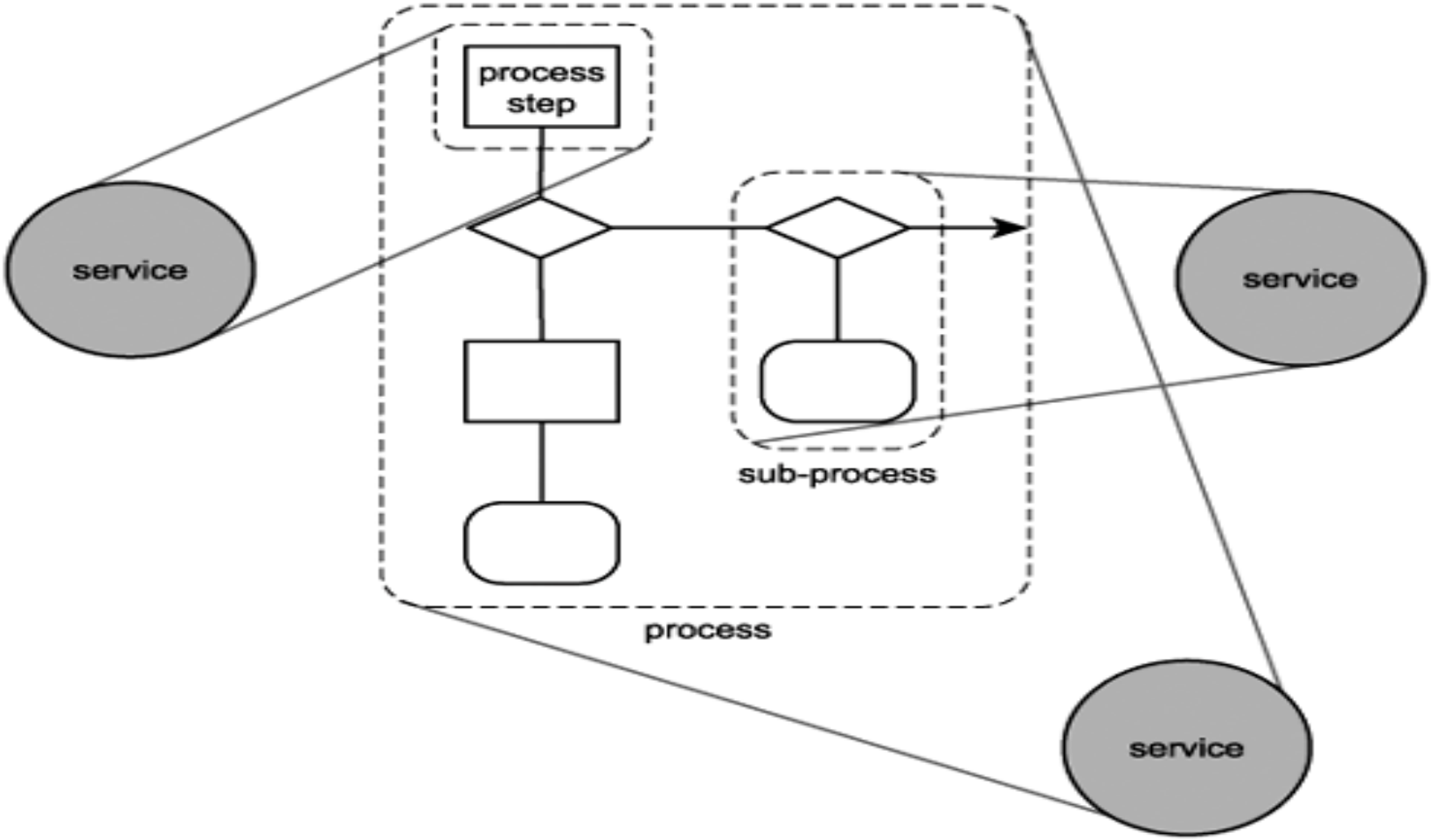


Section 3.1 Fundamental SOA

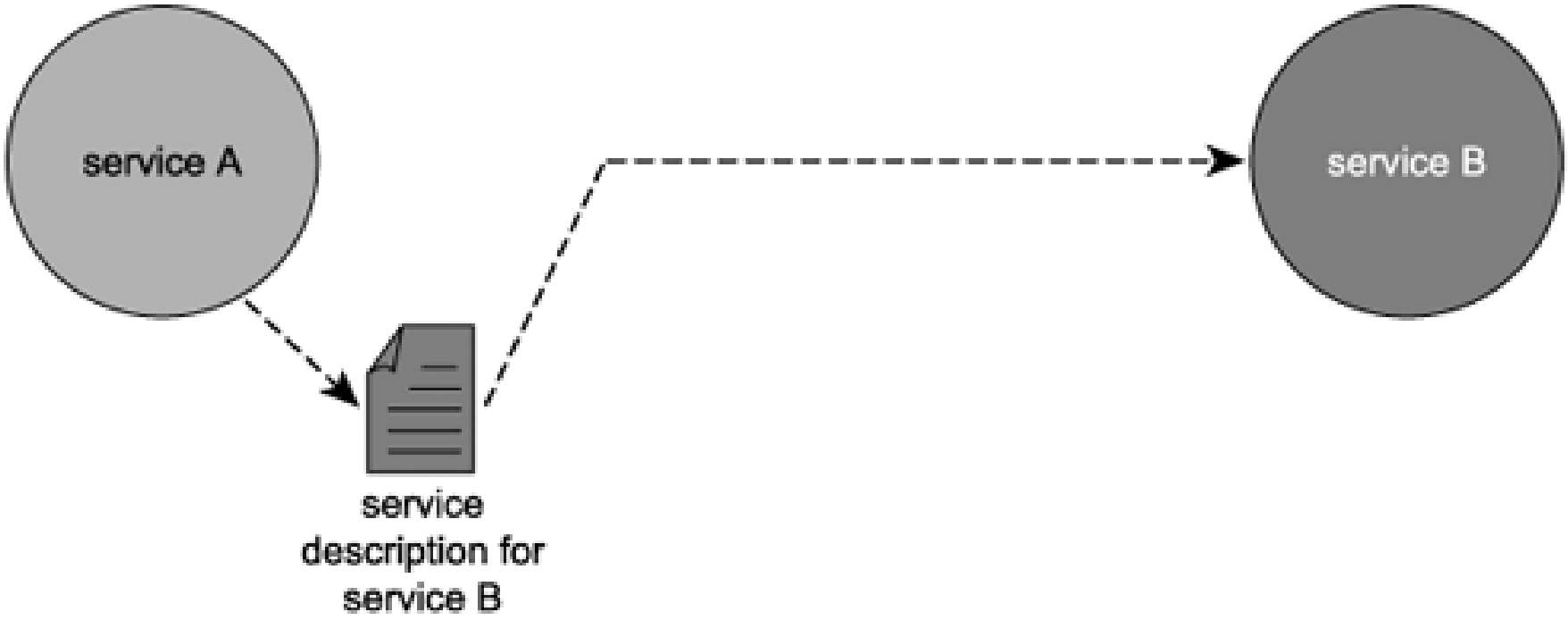
- Services encapsulate logic within a distinct context
  - Context == business task or business entity or any logical grouping
- Varied scales == a process step, set of steps or the entire process
- Services are related via relationships
- Services communicate via messages which are independent units of communication
- Service design promotes: loose coupling, contract driven composability, autonomy, discoverable stateless reusable components pp 37
- To this point we have described "primitive SOA" pp 39
- Case Study pp38

Services can encapsulate varying amounts of logic.

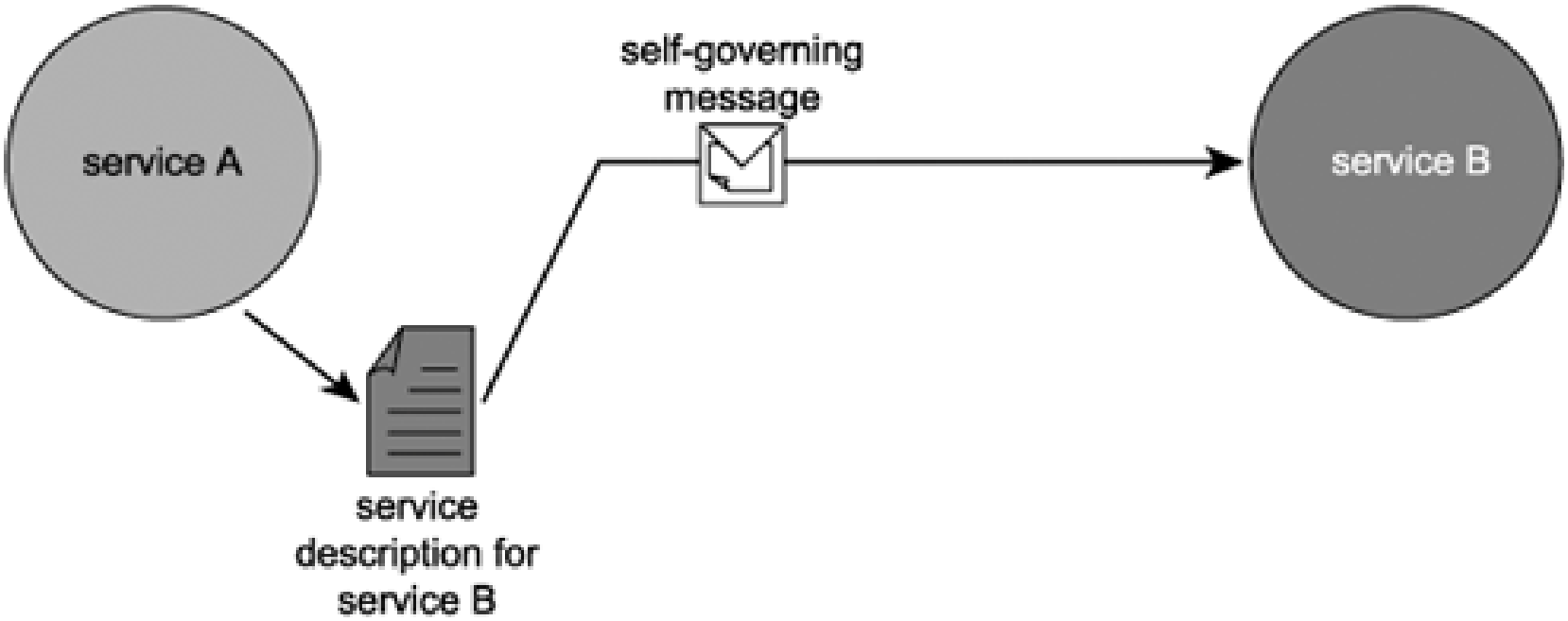




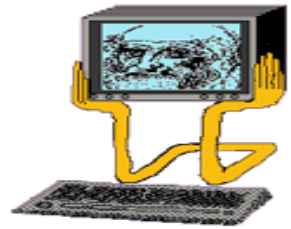
Because it has access to service B's service description, service A has all of the information it needs to communicate with service B.

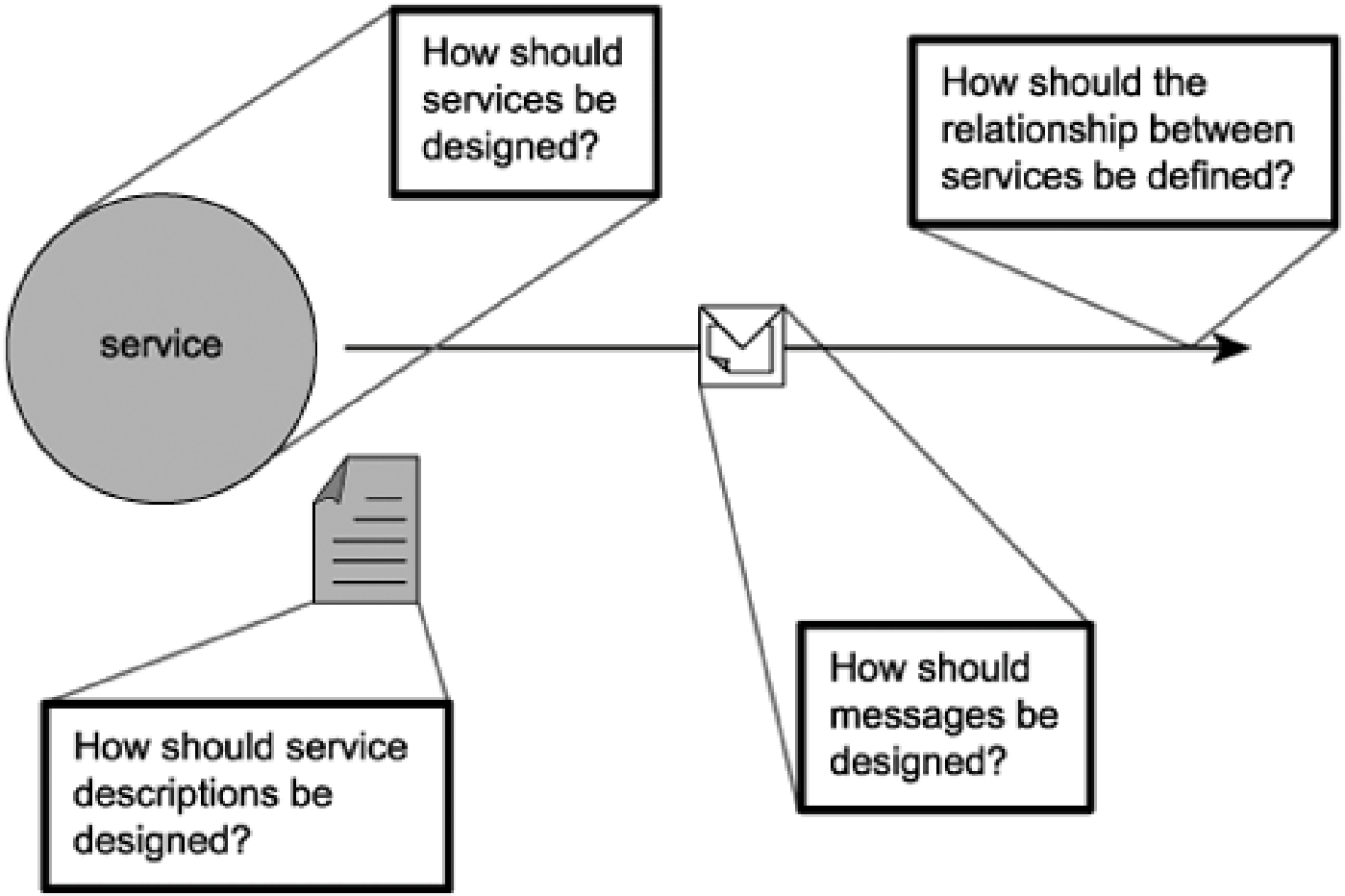


A message existing as an independent unit of communication



Service-orientation principles address design issues





Principles

Loose coupling Services maintain a relationship that minimizes dependencies and only requires that they retain an awareness of each other.

Service contract Services adhere to a communications agreement, as defined collectively by one or more service descriptions and related documents.

Autonomy Services have control over the logic they encapsulate.

Abstraction Beyond what is described in the service contract, services hide logic from the outside world.

Reusability Logic is divided into services with the intention of promoting reuse.

Composability Collections of services can be coordinated and assembled to form composite services.

Statelessness Services minimize retaining information specific to an activity.

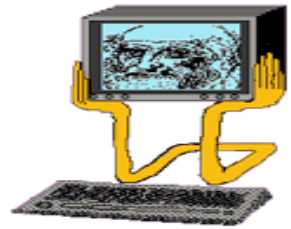
Discoverability Services are designed to be outwardly descriptive so that they can be found and assessed via available discovery mechanisms.

Section 3.2: Defining SOA (review)

Section 3.2 Common Characteristics of Contemporary SOA

3.2.20 Defining SOA page 54

Common characteristics of contemporary SOA



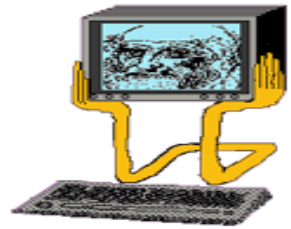
Contemporary SOA is at the core of the service-oriented computing platform.

Contemporary SOA represents an architecture that promotes service-orientation through the use of Web services.

Contemporary SOA increases quality of service.

- The ability for tasks to be carried out in a secure manner, protecting the contents of a message, as well as access to individual services.
- Allowing tasks to be carried out reliably so that message delivery or notification of failed delivery can be guaranteed.

Common characteristics of contemporary SOA



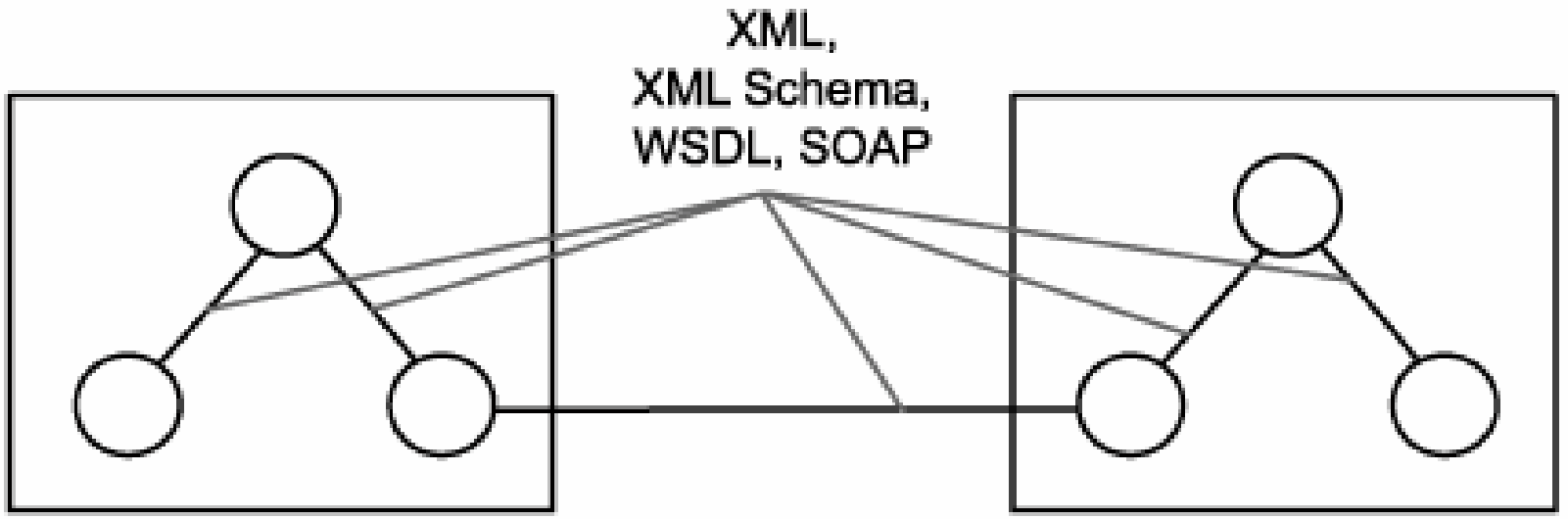
- Performance requirements to ensure that the overhead imposed by SOAP message and XML content processing does not inhibit the execution of a task.
- Transactional capabilities to protect the integrity of specific business tasks with a guarantee that should the task fail, exception logic is executed.

Contemporary SOA is fundamentally autonomous.

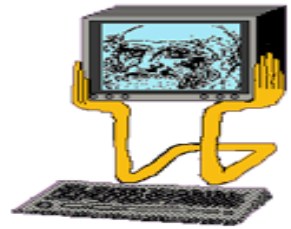
The service-orientation principle of autonomy requires that individual services be as independent and self-contained as possible with respect to the control they maintain over their underlying logic.

Common characteristics of contemporary SOA

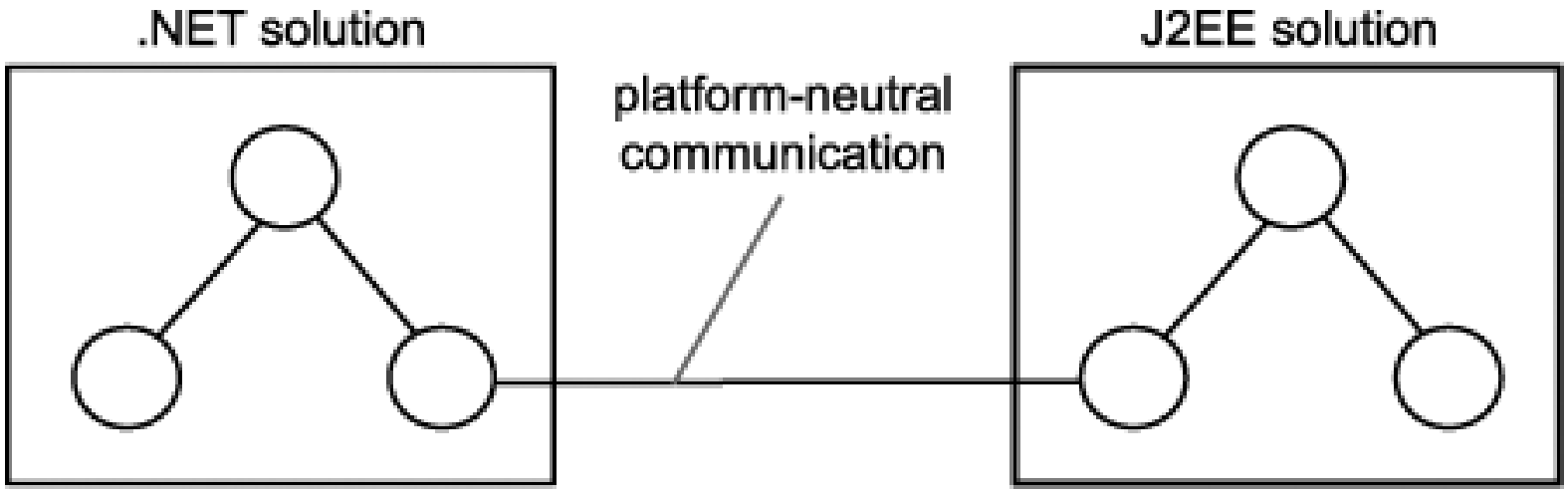
Contemporary SOA is based on open standards.



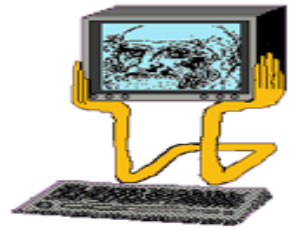
Common characteristics of contemporary SOA



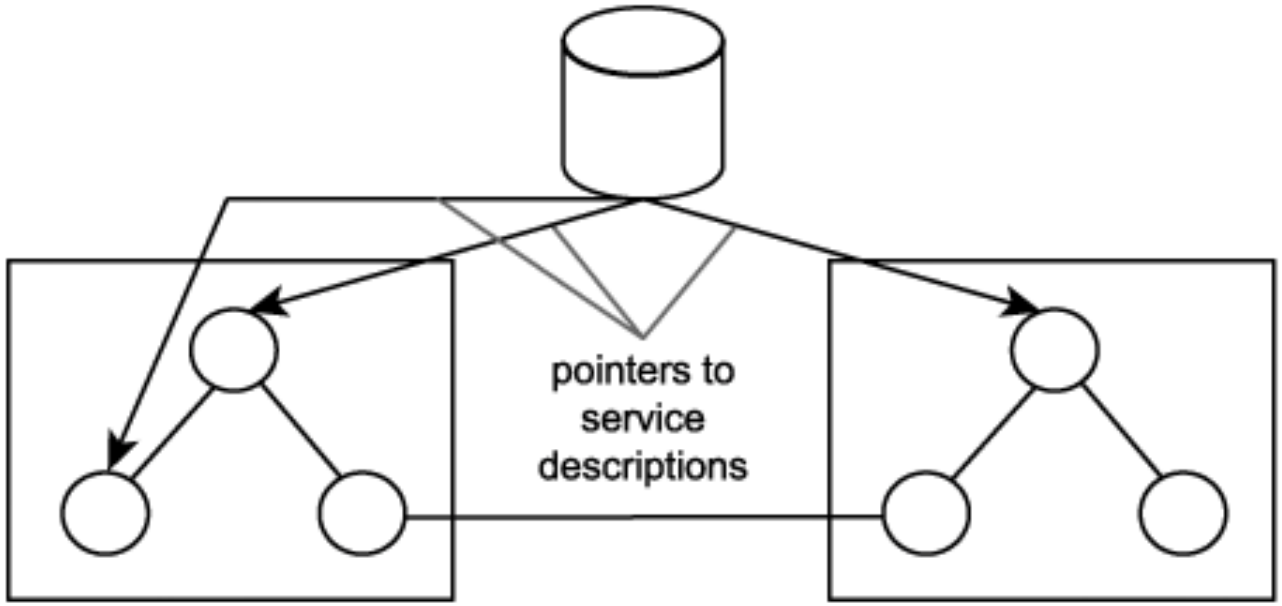
Contemporary SOA supports vendor diversity.



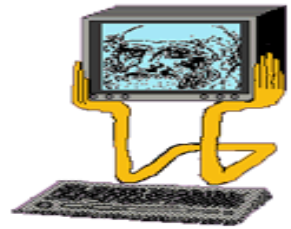
Common characteristics of contemporary SOA



Contemporary SOA promotes discovery
- Registries enable a mechanism for the discovery of services.

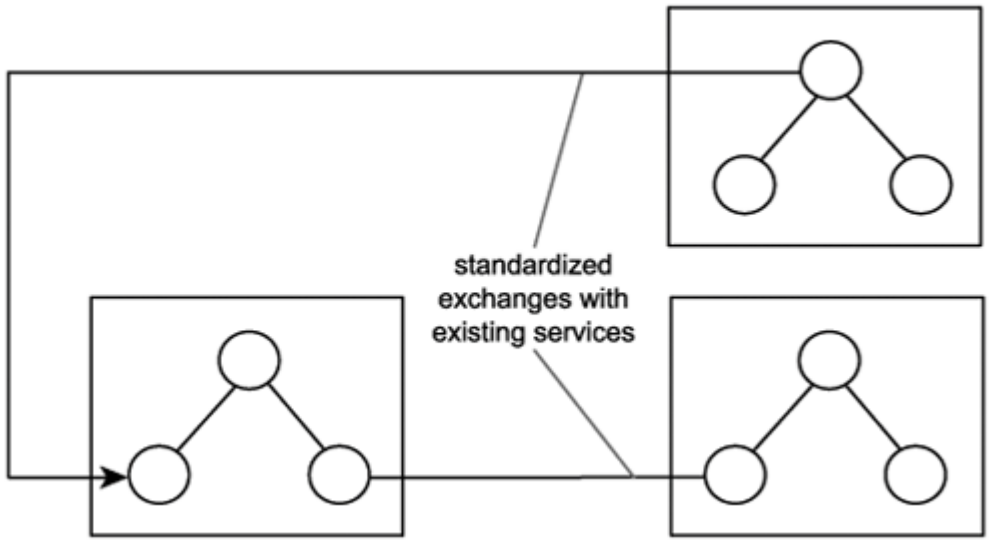


Common characteristics of contemporary SOA

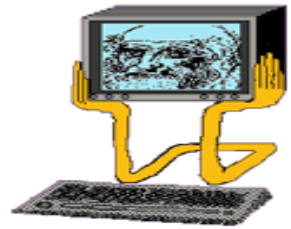


Contemporary SOA fosters intrinsic interoperability.

Intrinsically interoperable services enable unforeseen integration opportunities.

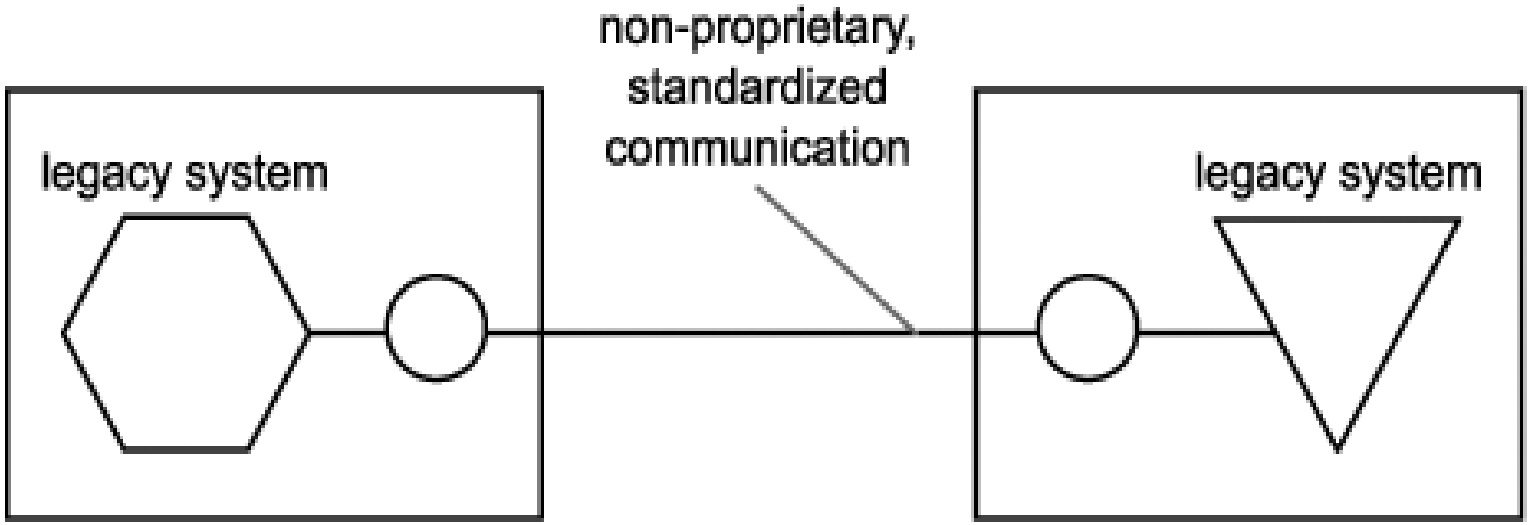


Common characteristics of contemporary SOA



Contemporary SOA promotes federation.

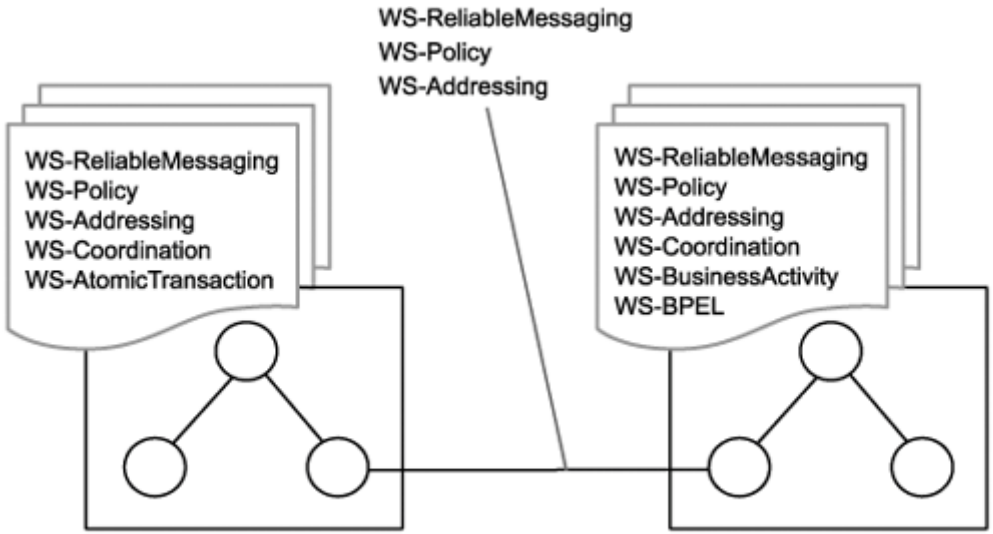
- Services enable standardized federation of disparate legacy systems.



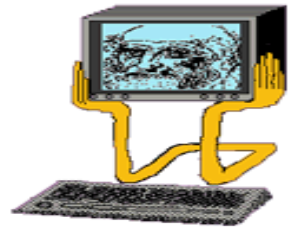
Common characteristics of contemporary SOA

Contemporary SOA promotes architectural composability.

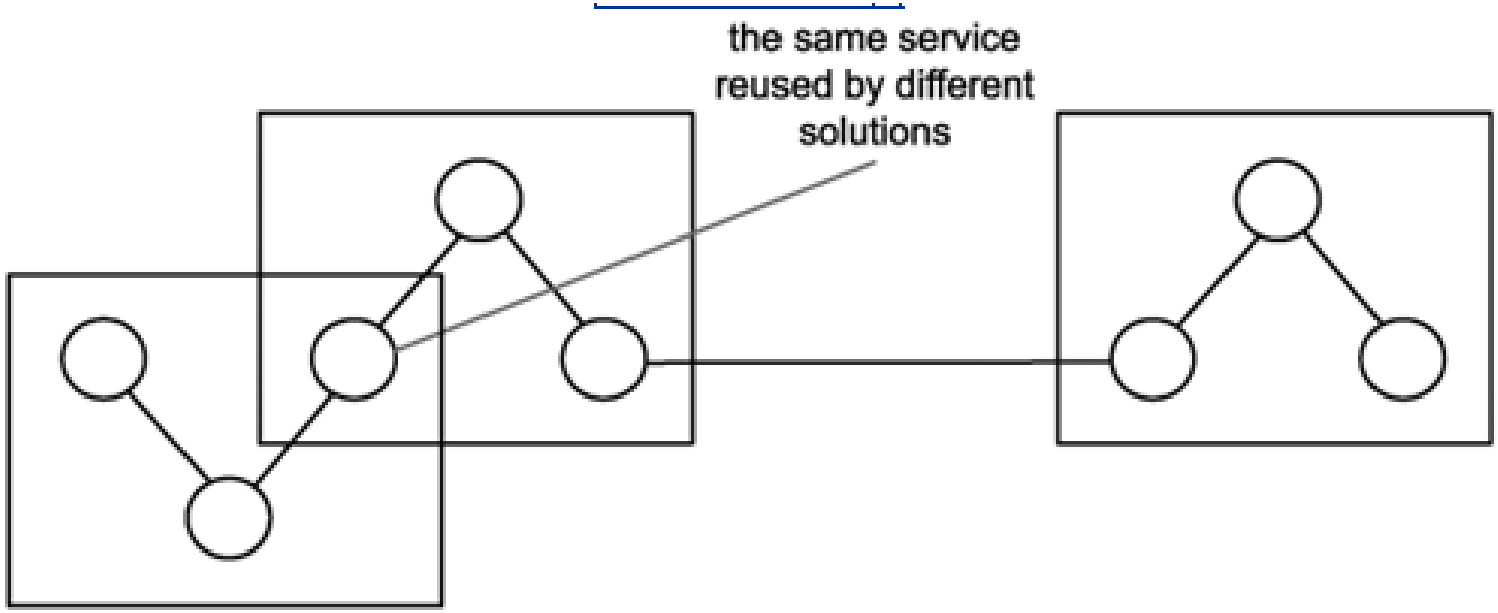
- Different solutions can be composed of different extensions and can continue to interoperate as long as they support the common extensions required.



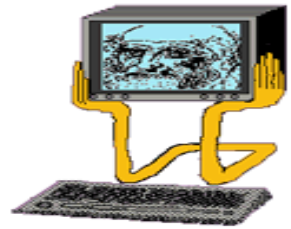
Common characteristics of contemporary SOA



- Contemporary SOA fosters inherent reusability.
- Inherent reuse accommodates unforeseen reuse opportunities.



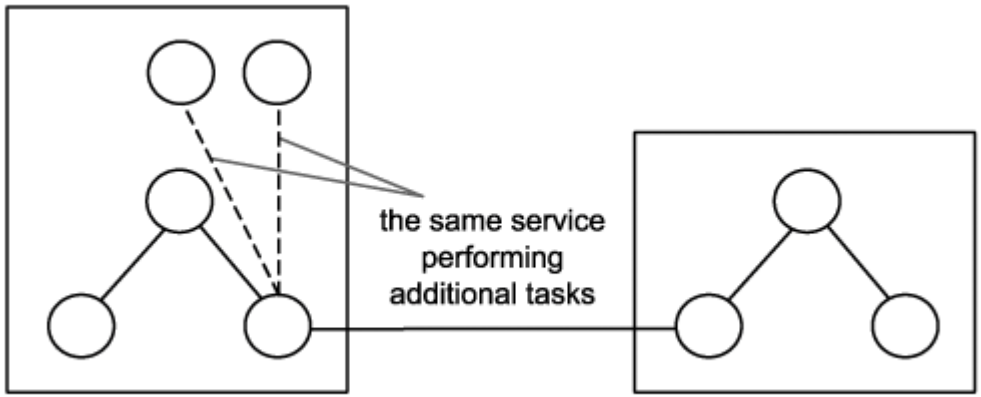
Common characteristics of contemporary SOA



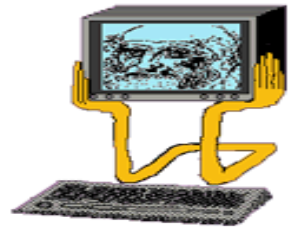
Contemporary SOA emphasizes extensibility.

Extensible services can expand functionality with minimal impact.

Contemporary SOA represents an open, extensible, federated, composable architecture that promotes service-orientation and is comprised of autonomous, QoS-capable, vendor diverse, interoperable, discoverable, and potentially reusable services, implemented as Web services.

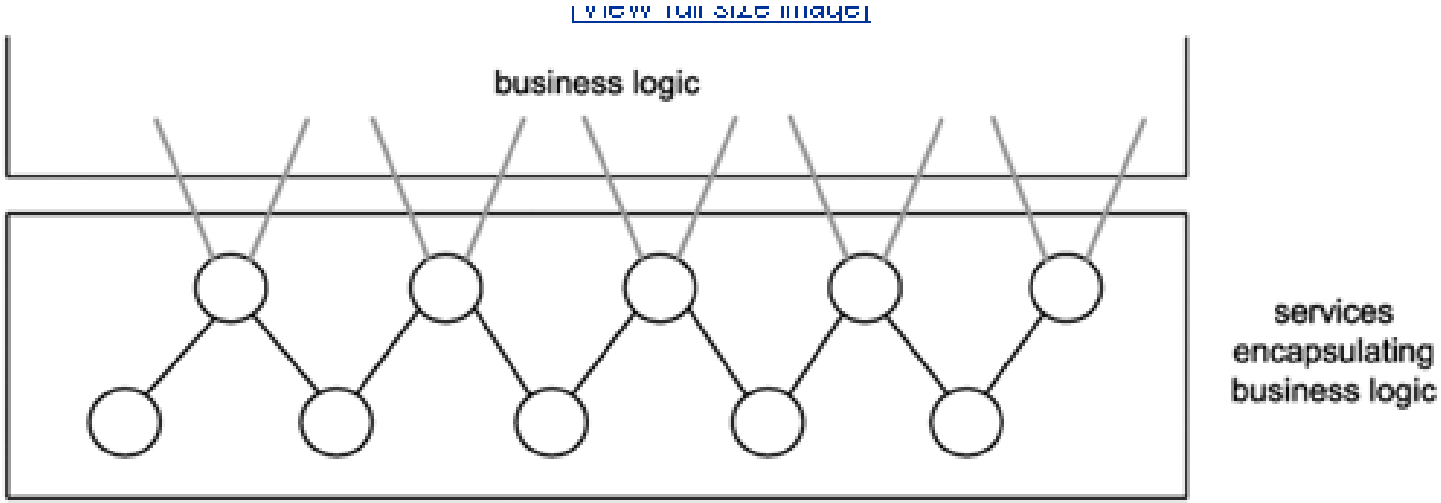


Common characteristics of contemporary SOA

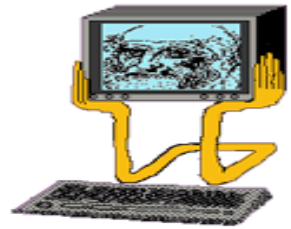


Contemporary SOA supports a service-oriented business modeling paradigm.

A collection (layer) of services encapsulating business process logic.

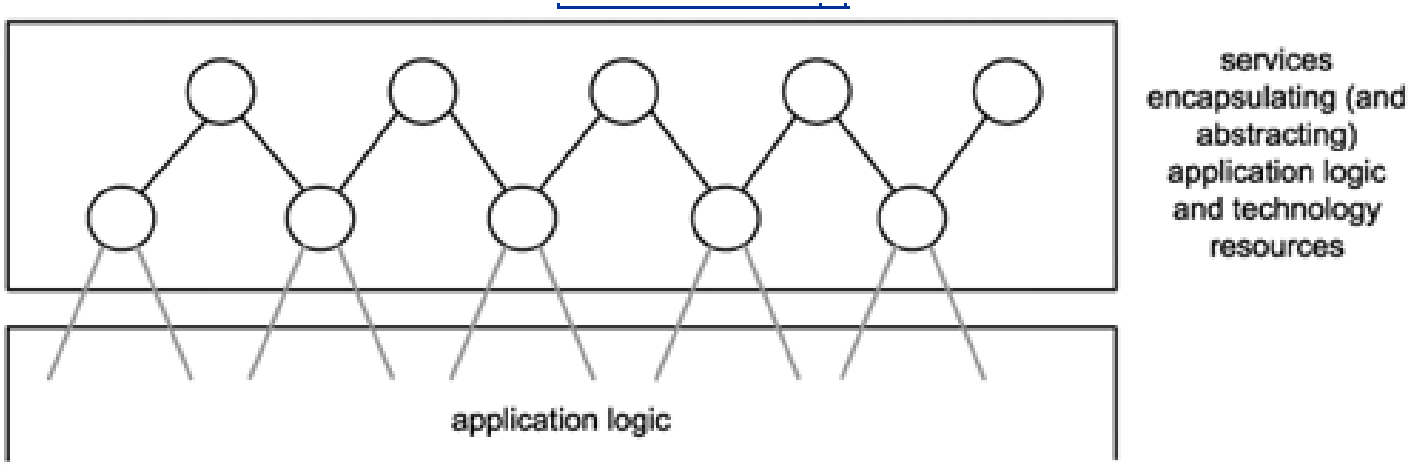


Common characteristics of contemporary SOA

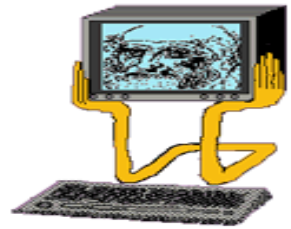


Contemporary SOA implements layers of abstraction.

Application logic created with proprietary technology can be abstracted through a dedicated service layer.

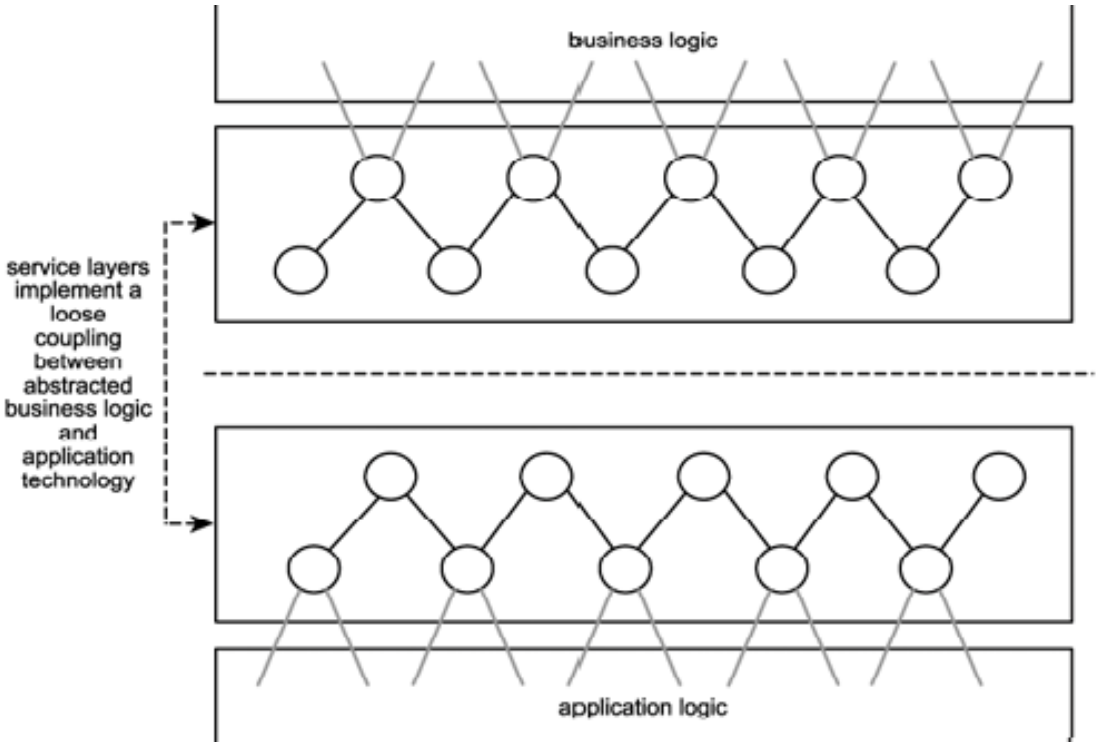


Common characteristics of contemporary SOA

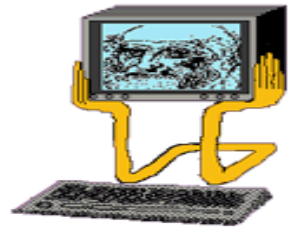


Contemporary SOA promotes loose coupling throughout the enterprise.

Through the implementation of service layers that abstract business and application logic, the loose coupling paradigm can be applied to the enterprise as a whole.

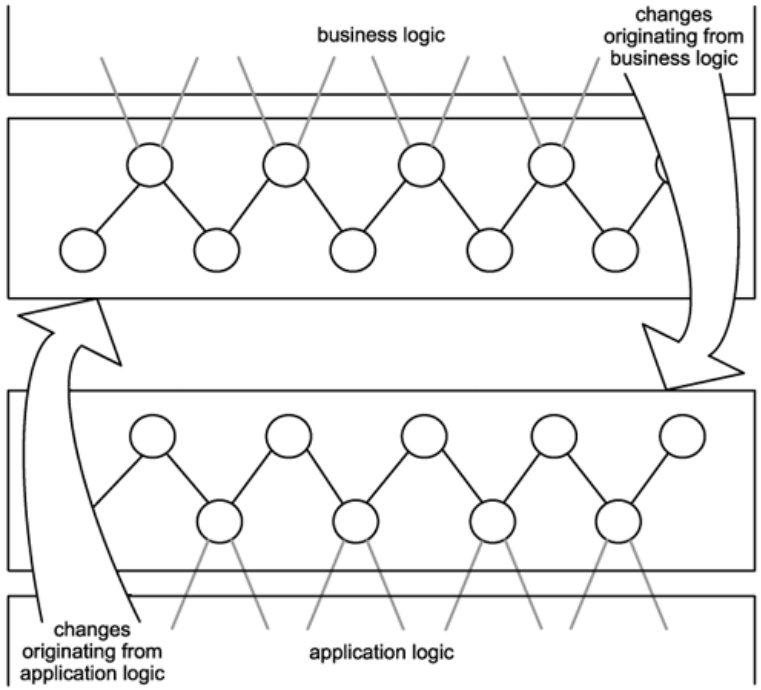


Common characteristics of contemporary SOA

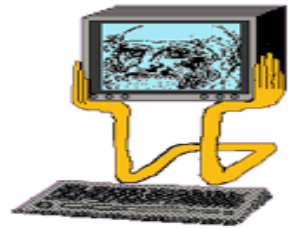


Contemporary SOA promotes organizational agility.

A loosely coupled relationship between business and application technology allows each end to more efficiently respond to changes in the other.



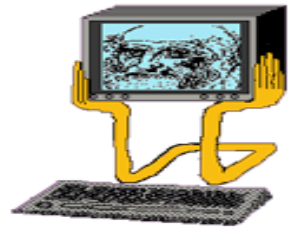
Common characteristics of contemporary SOA



Contemporary SOA is a building block.

SOA can establish an abstraction of business logic and technology that may introduce changes to business process modeling and technical architecture, resulting in a loose coupling between these models. These changes foster service-orientation in support of a service-oriented enterprise.

Common characteristics of contemporary SOA

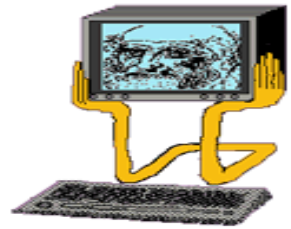


Contemporary SOA is an evolution.

SOA defines an architecture that is related to but still distinct from its predecessors.

- It differs from traditional *client-server* and *distributed* environments in that it is heavily influenced by the concepts and principles associated with service-orientation and Web services.
- It is similar to previous platforms in that it preserves the successful characteristics of its predecessors and builds upon them with distinct design patterns and a new technology set.
- For example, SOA supports and promotes reuse, as well as the componentization and distribution of application logic.
- These and other established design principles that are commonplace in traditional distributed environments are still very much a part of SOA.

Common characteristics of contemporary SOA



Contemporary SOA is still maturing.

Contemporary SOA is an achievable ideal.

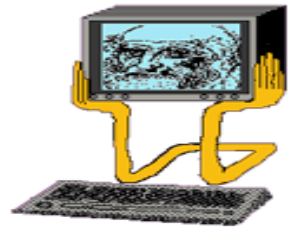
Defining SOA

Now that we've finished covering characteristics, we can finalize our formal definition.

Contemporary SOA represents an open, agile, extensible, federated, composable architecture comprised of autonomous, QoS-capable, vendor diverse, interoperable, discoverable, and potentially reusable services, implemented as Web services.

SOA can establish an abstraction of business logic and technology that may introduce changes to business process modeling and technical architecture, resulting in a loose coupling between these models.

Defining SOA



- SOA is an evolution of past platforms, preserving successful characteristics of traditional architectures, and bringing with it distinct principles that foster service-orientation in support of a service-oriented enterprise.
- SOA is ideally standardized throughout an enterprise, but achieving this state requires a planned transition and the support of a still evolving technology set.
- SOA is a form of technology architecture that adheres to the principles of service-orientation. When realized through the Web services technology platform, SOA establishes the potential to support and promote these principles throughout the business process and automation domains of an enterprise.

Section 3.3 SOA Misperceptions

3.3.1 thru 3.3.3: Major Misperception #1: SOA is NOT really something new NOT

Some validity, but the combination of "Old" ideas, as well as some new ideas synthesizes a new approach: the whole is greater then the sum of its parts.

Section 3.3 SOA Misperceptions

3.3.5 thru 3.3.6: SOA is Web Services, or WS with some extensions…NOT

Those are just technologies used to implement SOA

Section 3.3 SOA Misperceptions

3.3.7 SOA makes everything interoperable NOT

If you design and build an SOA system properly, it's pieces are interoperable with similar "good" SOA systems

Section 3.3 SOA Misperceptions

3.3.4: SOA Simplifies…: NOT

The point is not to simplify the solution to the problem, but to simplify performing individual steps in solving he problems, and in the integration of those resultant steps into the complete solution.

Section 3.4 SOA Benefits and the start of a Business Case for SOA

3.4.1 thru 3.4.3: Component Based Software Engineering benefits:

- Integration of components, reuse of components, streamlined architecting of components into solutions.
- Business Case is reduction of effort, solving multiple problems with single work products.

Section 3.4 SOA Benefits and the start of a Business Case for SOA

3.4.4 Leveraging the Legacy Systems

Retaining and (optionally) controlling the evolutionary replacement of legacy systems

Business Case is extending usable life of past investments

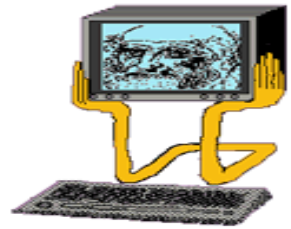
Section 3.5 Common Pitfalls in Adopting SOA

Designing an SOA as though it were a traditional distributed architecture

Question: What kinds of systems should not be designed as SOA?

- Proliferation of RPC-style service descriptions (leading to increased volumes of fine-grained message exchanges).
- Inhibiting the adoption of features provided by WS-* specifications.
- Improper partitioning of functional boundaries within services.
- Creation of non-composable (or semi-composable) services.
- Further entrenchment of synchronous communication patterns.
- Creation of hybrid or non-standardized services.

Section 3.5 Common Pitfalls in Adopting SOA



- Not having an XML foundation
  - What problem is it that XML solves?
- Not understanding SOA Performance and Security
  - What does SOA change in designing for Performance requirements?
  - Testing the message processing capabilities of your environments prior to investing in Web services.
  - Stress-testing the vendor supplied processors (for XML, XSLT, SOAP, etc.) you are planning to use.
  - Exploring alternative processors, accelerators, or other types of supporting technology, if necessary. For example, the XML-binary Optimized Packaging (XOP) and SOAP Message Transmission Optimization Mechanism (MTOM) specifications developed by the W3C. (For more information, visit www.w3c.org.)

Section 3.5 Common Pitfalls in Adopting SOA

What does SOA change in designing for Security requirements?

The WS-Security framework establishes an accepted security model supported by a family of specifications that end up infiltrating service-oriented application and enterprise architectures on many levels.

One of the more significant design issues you may face when WS-Security hits your world is the potential introduction of centralized security. With this approach, the architecture abstracts a large portion of security logic and rules into a separate, central layer that is then relied upon by service-oriented applications

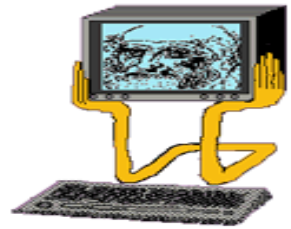
Chapter 4: The Evolution of SOA

Section 4.1 An SOA Timeline

XML

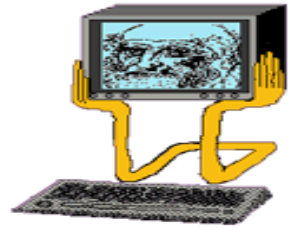
- Initially a publishing tool
- Then a data format tool
- Now a service oriented version of a remote procedure call protocol
- XML Schema Definition Language (XSD) and the XSL Transformation Language (XSLT)

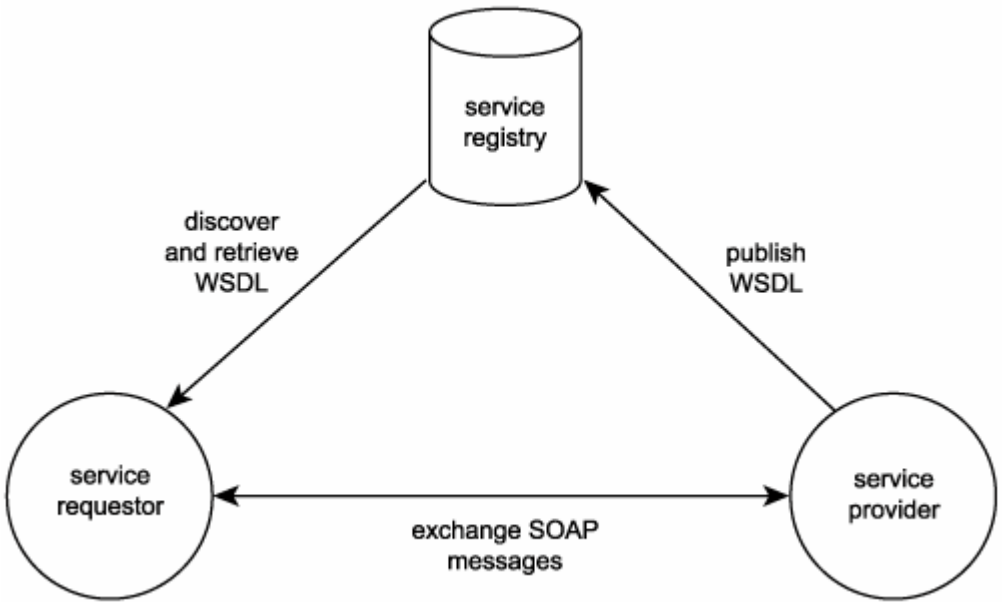
Chapter 4: The Evolution of SOA



- Web Services
  - Initially port 80 tunneling
  - Initially Simple Object Access Protocol
  - Now (with WS-*) much more complex
    - Security
    - Dependable delivery
    - Orchestration
    - UDDI
- SOA
  - Forced evolution of XML and Web Services

An early incarnation of SOA





An early incarnation of SOA

First-generation Web services standards fulfilled this model as follows:

- WSDL described the service.
- SOAP provided the messaging format used by the service and its requestor.
- UDDI provided the standardized service registry format.
- business process definition languages, WS-BPEL

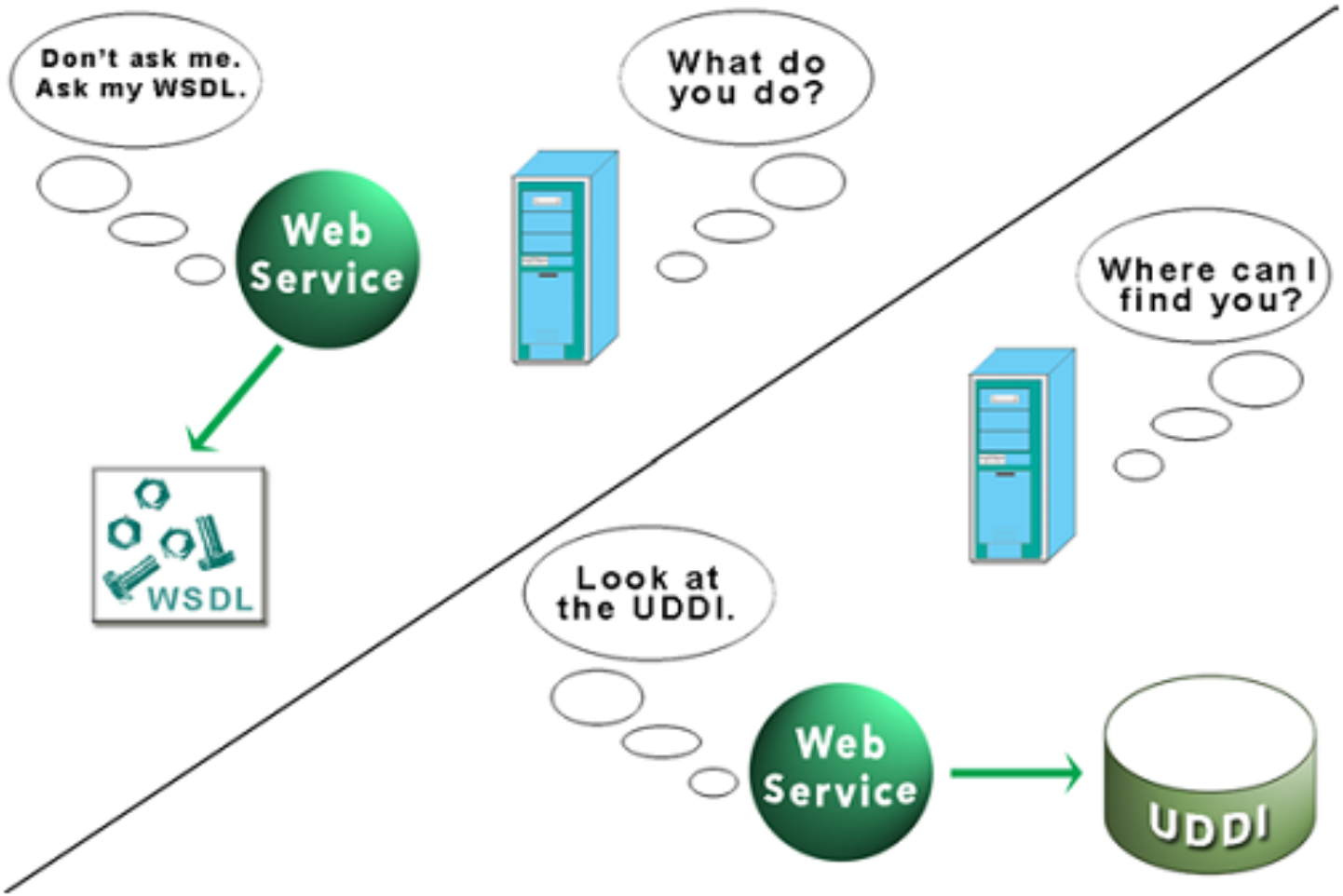


Figure 2.4 When it comes to describing its functionality and location, a web service is in and of itself essentially "dumb." To describe itself to potential requestors, the web service relies on its WSDL document, which provides a detailed explanation of the web service's functionality and how to access it. For location, the web service relies on its listing in a UDDI registry to enable potential requestors to find it.

Section 4.2: Continuing Evolution of SOA

Standards, Specifications, and Extensions

- Standard An accepted industry standard. All of the first-generation Web services specifications are considered standards, as are a number of XML specifications.
- Specification A proposed or accepted standard, described in a specification. XML standards, first-generation Web services standards, and WS-* extensions all exist within specifications.
- Extension An extension typically represents a WS-* specification or a feature provided by a WS-* specification.

Section 4.2: Continuing Evolution of SOA

Standards Organizations

- The World Wide Web Consortium (W3C)
- SOAP , WSDL ,
- Choreography Description Language (WS-CDL),
- Web Services Architecture

Section 4.2: Continuing Evolution of SOA

- Organization for the Advancement of Structured Information Standards (OASIS)
- WS-BPEL specification
- ebXML (a specification that aims to establish a standardized means of B2B data interchange)
- Security Assertion Markup Language (SAML)

Section 4.2: Continuing Evolution of SOA

- Extensible Access Control Markup Language (XACML)

Section 4.2: Continuing Evolution of SOA

- SOA Vendors
  - Distributed Tool and System Vendors
    - Web Service infrastructure
    - Web Service Development Tools
      - Integrated Development Environment (IDE)
  - "Forward Thinking, Imaginative" Vendors
    - Not a good thing, notice the quotation marks above
- Microsoft, IBM, BEA Systems, Sun Microsystems, Oracle, Tibco, Hewlett-Packard, Canon, Commerce One, Fujitsu, Software AG, Nortel, Verisign, and WebMethods

Section 4.3: Roots of SOA

- A distributed integration/communication protocol
- Can be synchronous or asynchronous
  - HTTP basis of most SOAP implementations forces some call and response behavior
- Can be used to wrap non Service Oriented components, integrating their functionality in to an SOA