Service Oriented Architecture

Personal Information

- 20 plus years in software development
- MS, Viterbi School of Engineering
- MBA, Marshall School of Business
- MS, Loyola Marymount University
- BS with Honors, DePaul University

Lecture Objectives

- Develop an understanding of how SOA is used in organizations
- Understand benefits & disadvantages of SOA
- Learn about governance paradigms and best practices
- Learn about emerging trends such as the use of SOA for embedded systems and Cloud Computing
- Internalize how critical *security* is but learn how to resolve security issues as well
- Get familiar with a toolbox of off-the-shelf applications that can help automate governance

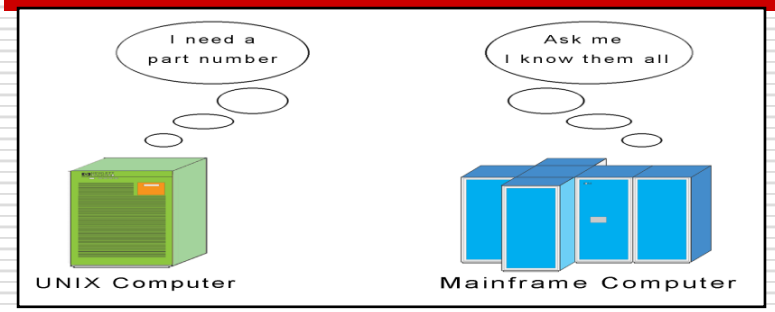
Course Outline

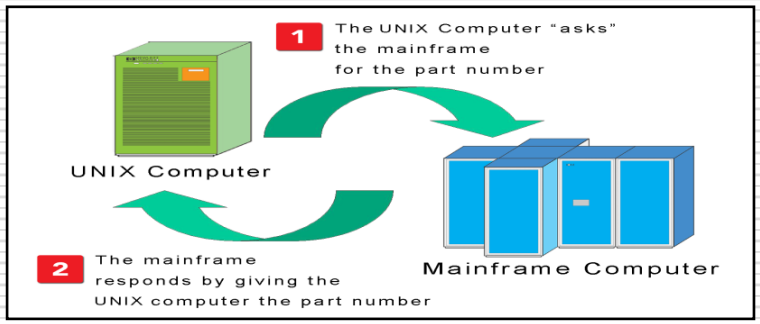
- Section 1 – Motivation for SOA
- Section 2 – Integrated SOA Governance
- Section 3 – Platform Independent Governance Automation
- Section 4 – SOA Best Practices
- Section 5 – Guide to SOA Implementation
- Section 6 – SOA Economics
- Section 7 – Summary

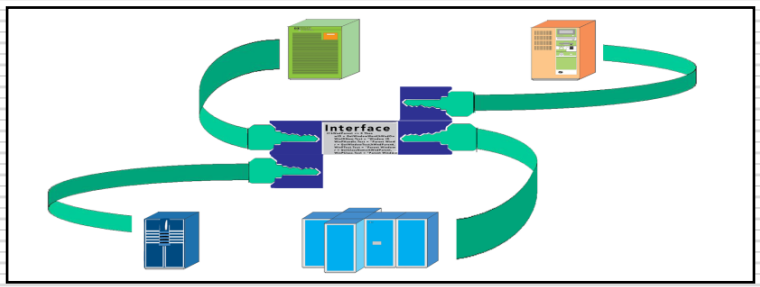
SECTION 1

MOTIVATION FOR SOA

What is SOA? First, Understand "Tight Coupling"







- Data and functionality typically reside on more than one system (and application)
- Applications need to be able to "talk to each other"
- Status quo: Proprietary or custom communication interfaces between applications

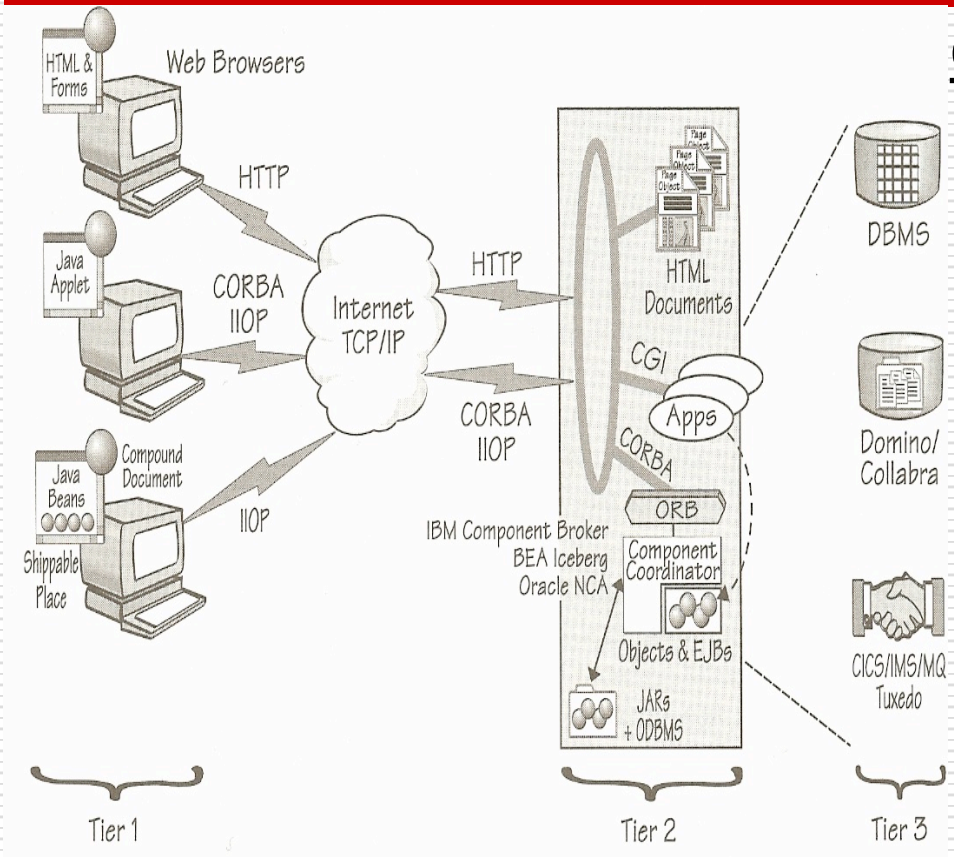
Source: H. Taylor, "*Service-Oriented Architecture (SOA) 101 'What's Hype, What's Real?'*", *Juniper Networks, Inc.,2007.*

Challenges with Tight Coupling

- While tight coupling is inherently sound, the following challenges are encountered in its implementation:
  - It's costly to maintain
  - Slow and costly to change
  - Cost and complexity compounded by multi-party scenarios such as B2B or integration with the public sector
  - Cost and complexity of managing and changing a tightly coupled architecture makes business agility difficult (IT can't keep up with business needs, but it's not their fault)
  - **Does not support reuse!**
- Recognized for many years as challenge industry wanted to solve
- Evolution of reuse solutions reflects industry's concerns
  - Header files, inheritance and polymorphism at the object level, frameworks
  - CORBA (Common Object Request Broker Architecture)
  - Microsoft COM (Component Object Model)
  - EAI (Enterprise Application Integration)
  - Web Services

Source: H. Taylor, "*Service-Oriented Architecture (SOA) 101 'What's Hype, What's Real?'*", *Juniper Networks, Inc.,2007.*

Challenges with Tight Coupling



The 3-Tier CORBA/Java Object Web.
Source: Client/Server Programming with JAVA and CORBA Second Edition by R. Orfali and D. Harkey, p. 45.

Overview of CORBA

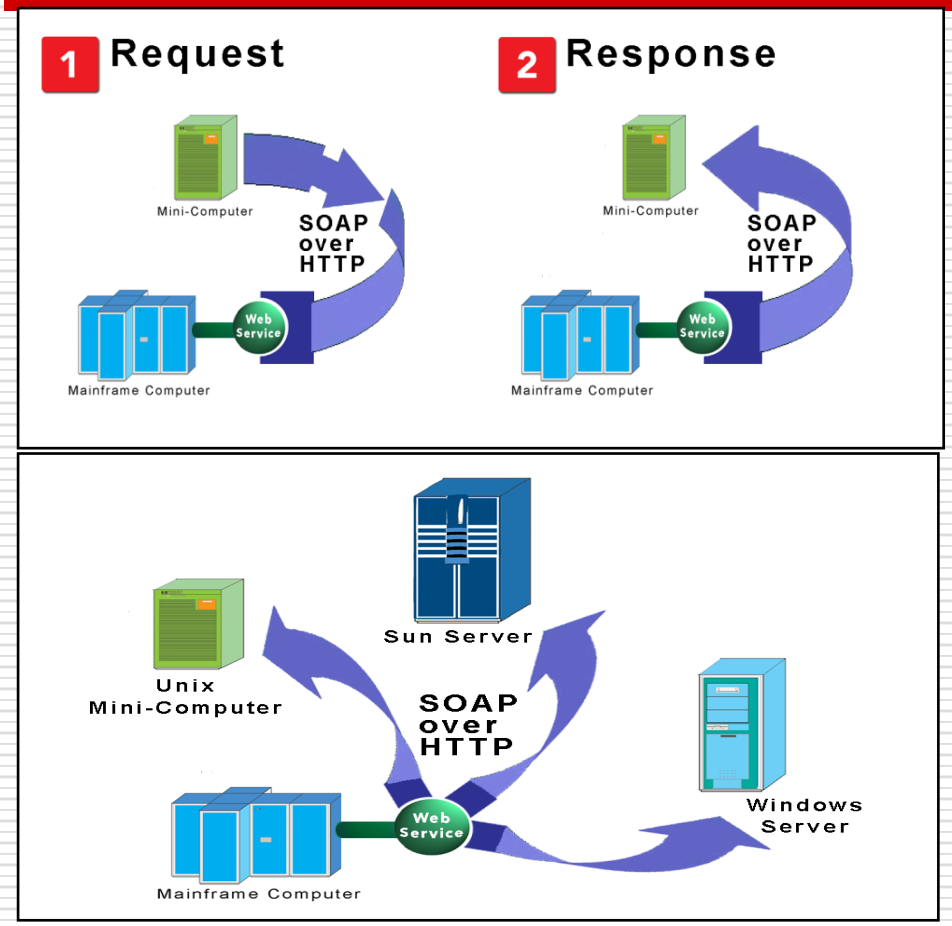
- Tier1 belongs to traditional Web browsers and Web-centric applications
- Tier 2 runs on any server that can support HTTP and CORBA clients
  - CORBA objects, like EJBs, encapsulate business logic
- Tier 3 consists of almost anything a CORBA object can access

Challenges with Tight Coupling

- Overview of COM
  - Microsoft COM (Component Object Model) enables software components to communicate
  - COM is used by developers to create re-usable software components, link components together to build applications, and take advantage of Windows services
  - COM objects can be created with a variety of programming languages including object-oriented languages such as C++
  - The family of COM technologies includes COM+, Distributed COM (DCOM) and ActiveX® Controls.
  - The .NET Framework provides bi-directional interoperability with COM, which enables COM-based applications to use .NET components and .NET applications to use COM components
- Reasons CORBA, COM, EAI and others did not work
  - Lack of open standards
  - Proprietary components

Source: http://www.microsoft.com/com

SOA: The Ideal of Open Interoperability (Loose Coupling)



SOA – A Definition

- An IT architecture composed of software that has been exposed as "Services" – i.e. invoked on demand using a standard communication protocol.
- "Web Services" – software available as a "service" using Internet protocols.
- One software application talking to another using a standards-based (i.e. non-proprietary) language over a standards-based communication protocol.
- Universal "Dial Tone" between software applications
- An IT architecture that enables "loose coupling" of applications

Source: H. Taylor, "*Service-Oriented Architecture (SOA) 101 'What's Hype, What's Real?'*", *Juniper Networks, Inc.,2007.*

Core SOA Definitions

- **XML** – Extensible Markup Language
- **SOAP** – Simple Object Access Protocol
- **WSDL** – Web Services Description Language
- **UDDI** - Universal Description, Discovery and Integration
- **ESB –** Enterprise Service Bus
- Key Concepts
  - Network Transparency
  - Virtualized endpoint
  - Self-describing software
  - Universally discoverable software
  - Universally understood software
  - Machine to machine interaction

Source: H. Taylor, "*Service-Oriented Architecture (SOA) 101 'What's Hype, What's Real?'*", *Juniper Networks, Inc.,2007.*

SOA Usage & Supporting Platforms

- SOA Usage
  - B2B
  - Enterprise Application Integration (EAI)
  - Application to Application
  - Government
- Major Players in SOA Space
  - IBM: WebSphere SOA Product Suite
  - BEA: Aqualogic (WebLogic)
  - Oracle: Fusion Middleware
  - Microsoft: .NET
  - SAP: NetWeaver

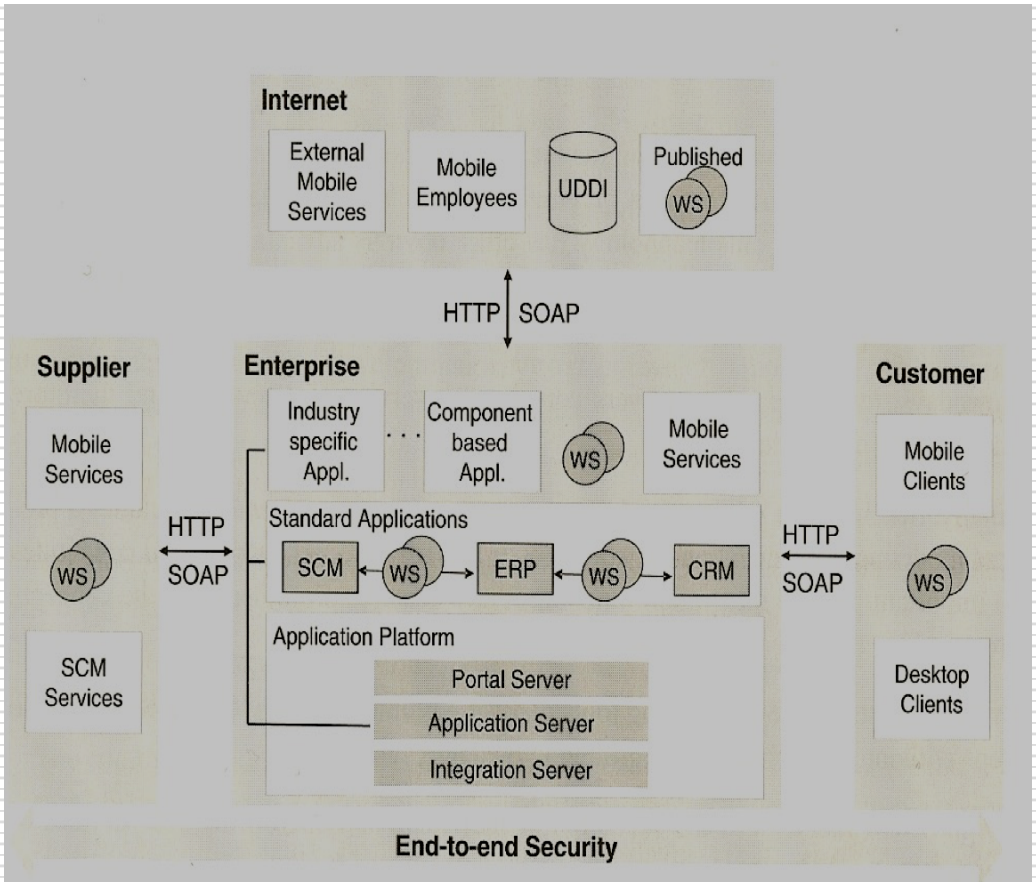
Source: H. Taylor, "*Service-Oriented Architecture (SOA) 101 'What's Hype, What's Real?'*", *Juniper Networks, Inc.,2007.*

What Makes Web Services Appealing?

	CORBA	JAVA RMI	ONC(SUN) RPC	WEB SERVICES
Data Encoding	Common Data Representation (CDR)	Serialized Java/CDR	Extended Data Representation (XDR)	XML (WS-I doc-literal, SOAP Encoding)
Message Format	IIOP (GIOP)	RMI Protocol/IIOP	RPC RMS	SOAP
Transport Protocol	TCP	TCP	UDP TCP	HTTP
Description Language	CORBA IDL	Java Interface/Class	RPC IDL	WSDL
Discovery Mechanism	COS Naming	RMI Registry	Undefined	UDDI
Invocation Method	CORBA RMI	Java RMI (method call)	RPC	Undefined

Source: See [56], Page 4.

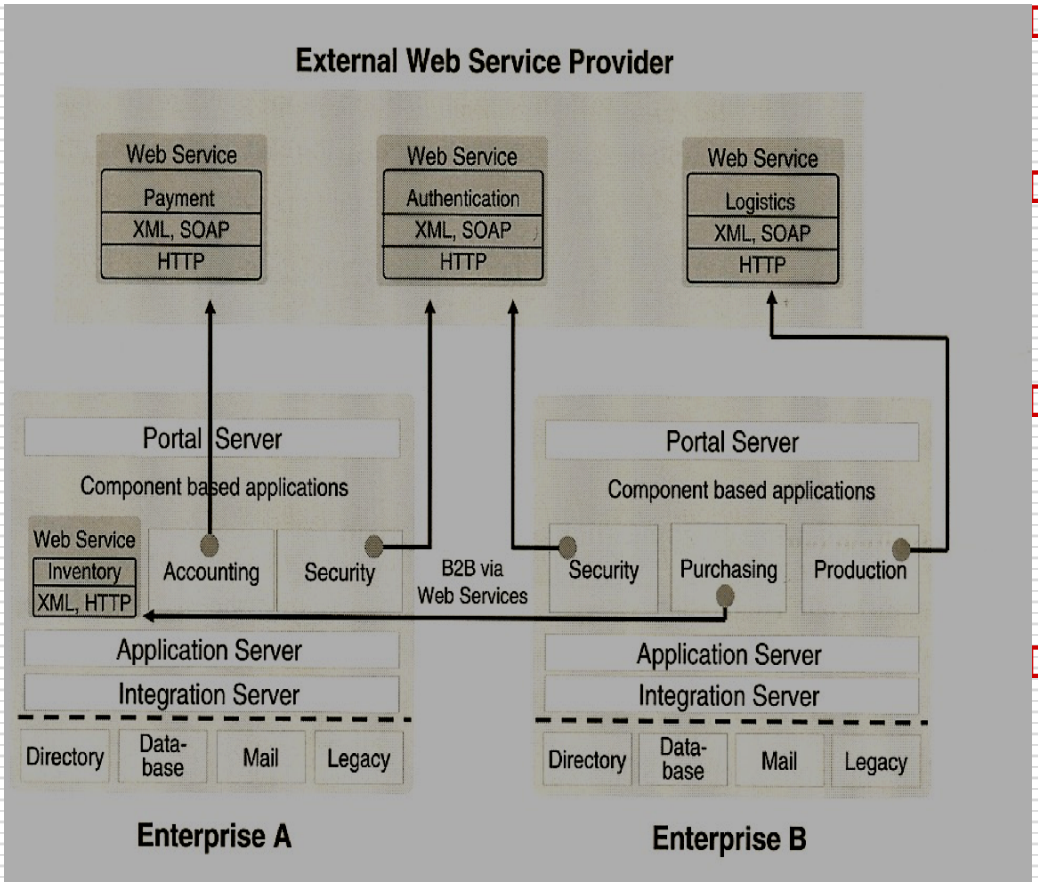
Cross-Enterprise Solution Architecture



- Figure illustrates tomorrow's e-business solution architecture
- 4 stakeholders communicate via Web services
  - Suppliers
  - Customers
  - Enterprise
  - Employees

Source: Mobility, Security and Web Services: Technologies and Service-Oriented Architectures for a new Era of IT Solutions by Gerhard Wiehler, p. 46.

e-Business Architecture for B2B



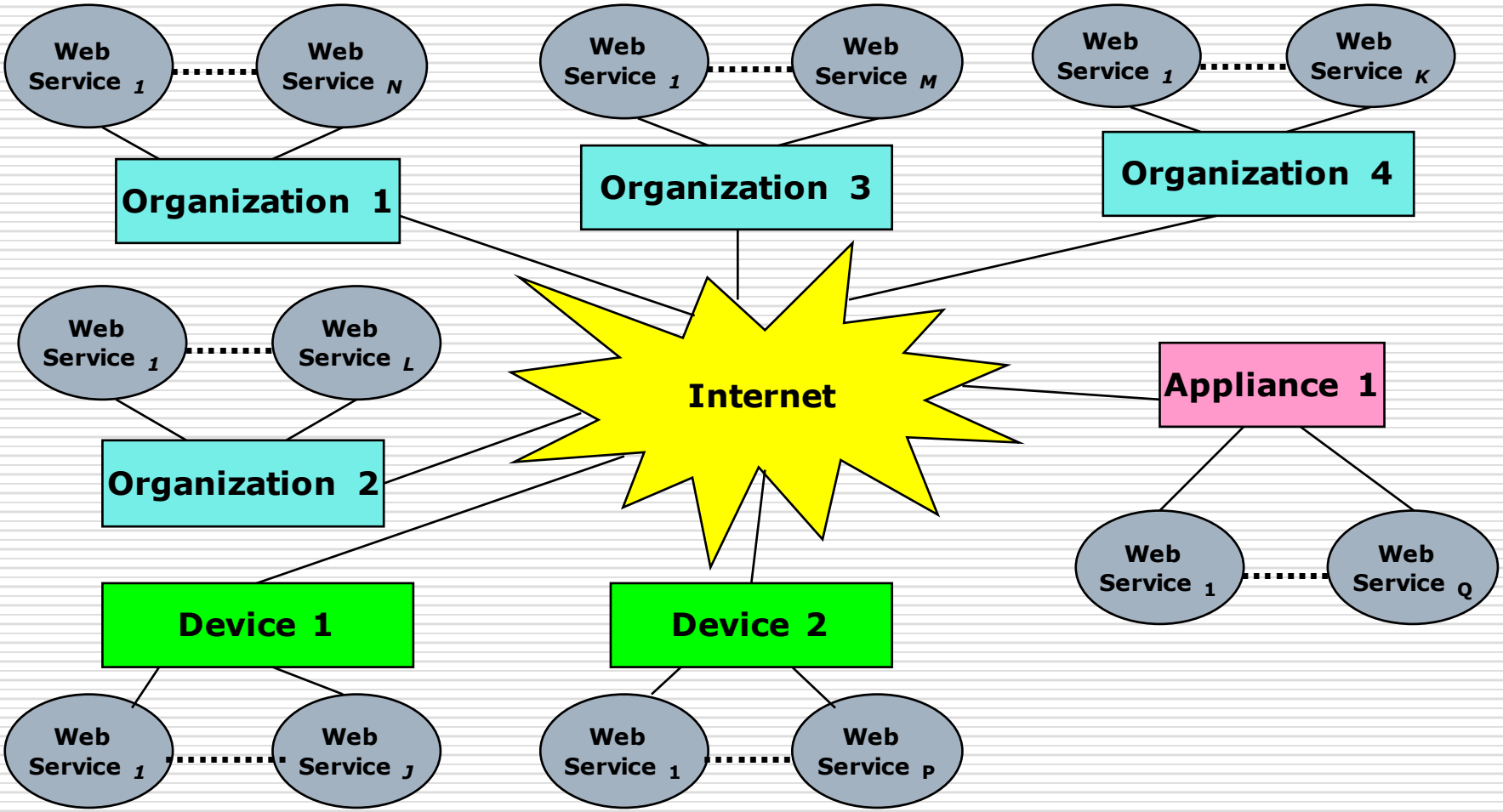
- 1 External Provider, 2 Enterprises: A & B
- Enterprise B **Purchasing** accesses Enterprise A's **Inventory** Web Service
- Both enterprises access an authentication Web service provided by an external provider
- Enterprises A & B separately access Web services that provide **Payment** and **Logistics** Web Services

Source: Mobility, Security and Web Services: Technologies and Service-Oriented Architectures for a new Era of IT Solutions by Gerhard Wiehler, p. 101.

SECTION 2

INTEGRATED SOA GOVERNANCE

Sample Web Service Topology



Why Governance?

- How do you develop Web Services in an organized and predictable way?
  - Is a Web Service being considered? How are you going about it?
  - Where is a Web Service in its life cycle?
    - Concept? Development? QA? Testing? Deployed?
- Questions if you have a complex *ecosystem* of services
  - How do you manage them operationally?
    - What services are up/down, for how long, etc.
    - Are the services load balanced?
  - What are policies for accessing the endpoints?
  - How about security?

Integrated SOA Governance

- Integrated SOA Governance ensures the applicability, integrity and usability of a wide range of assets through all their lifecycle stages
- Lifecycle stages range from asset identification through asset deprecation
- The full lifecycle is split into:
  1. Planning governance
  2. Development governance
  3. Operational governance
  4. Policy Governance

Source: *SOA Software, Inc., 2008.*

Planning Governance

- Idea is to **build the right things**
- New area for SOA
- Allows organizations to identify potential services in a planned and managed community
  - Enterprise Architects
  - Business Analysts
  - Portfolio Managers
- Recognized by industry as critical
  - Booz Allen Hamilton/US Government
  - Kaiser (Revitalized Claim Systems)
  - Consulting companies such as Infosys

Source: *SOA Software, Inc., 2008.*

Planning Governance Cont'd

- Key Task: Identification & Analysis
  - Define Services
  - Define Policy
  - Define Profiles
  - Define Process
  - Define Test Cases
  - Information Architecture
  - Identify other assets

Source: *SOA Software, Inc., 2008.*

Planning Governance Cont'd

- Typical Questions During Planning:
  - What capabilities should be exposed as Web Services?
  - What existing and planned applications would benefit from consuming shared services?
  - What services should be priority?
  - Who should access a specific service and how do we ensure appropriate access?
  - How about "Megaprogramming" [Boehm et al.] questions?

Source: *SOA Software, Inc., 2008.*

Planning Governance Cont'd

- Think about Megaprogramming Key Success Factors (KSF) & Natural Market Analogs [Boehm et al.]

Megaprogramming KSF	**Natural Market Analog KSF**
A. Architecture Determination	A. Product Line (market) Structuring
B. Architecture/Component Description	B. Product Line (market structure) description
C. Component construction	C. Producer
D. Component composition/assembly	D. Consumer
E. Component interchange	E. Brokerage

Planning Governance Cont'd

- Solutions require integration with:
  - Wide range of existing enterprise repositories
  - Application portfolio management
  - Enterprise architecture planning solutions

- Output from Planning Governance Process
  - **Candidates** for a suitable architecture
  - Set of **candidate services** that feed into the Development Governance process
  - Set of **candidate policies** that feed into the Policy Governance process

Source: *SOA Software, Inc., 2008.*

Development Governance

- Idea is to **build things right**
- Marshals an asset through the development process
- Development process typically spans:
  - Design
  - Development
  - Testing
  - Staging
- Development Governance includes:
  - Workflow mechanism to approve migration between phases
  - Policy compliance validation
  - Clear separation (logically, physically, or both) between lifecycle stages

Source: *SOA Software, Inc., 2008.*

Development Governance Cont'd

- Solution *depends* on Policy Governance for:
  - Compliance policy definition
  - Management, and validation
- Policies are used to determine:
  - Relevance and suitability of services at each lifecycle stage
  - Determine if assets meet enterprise standards and guidelines before they can promoted to the next stage of the lifecycle.
- ***Example***--For a service to move from design to development, the enterprise may require:
  - There is a design document in the repository
  - The service has a WSDL
  - The services are categorized appropriately
  - Registered consumers waiting for the service

Source: *SOA Software, Inc., 2008.*

Operational Governance

- Idea is to **ensure what's built behaves right**
- Controls the runtime aspects of SOA
- Typically includes
  - Web Service monitoring
  - Security and management
  - Runtime policy system
- Relies heavily on Policy Governance solution
  - Need to discover policies for implementation & enforcement

Source: *SOA Software, Inc., 2008.*

Operational Governance Cont'd

- Key goal of a well architected system is to fully abstract service consumers & providers from complexity
- Complexity includes:
  - ✓ Policy implementation
  - ✓ Enforcement
  - ✓ Service endpoint location
  - ✓ Transport
  - ✓ Standards
  - ✓ Message Exchange Pattern
  - ✓ Other impedances to operability
- Should provide:
  - ✓ Agents & delegates
  - ✓ Network resident intermediary for service virtualization

Source: *SOA Software, Inc., 2008.*

Policy Governance

- Key goal is to have a uniform policy for all governance areas
- Policy Governance does the following:
  - Defines and manages policies
  - Associates polices with assets
  - Validates and reports on policy compliance
- Policy types include:
  - Metadata compliance policies applied in Planning and Development Governance
  - Security, reliability, and service-level policies applied through an Operational Governance solution

Source: *SOA Software, Inc., 2008.*

Summary of Integrated SOA Governance

Source: "Integrated SOA Governance for Microsoft", *SOA Software, Inc., 2008.*

SECTION 3

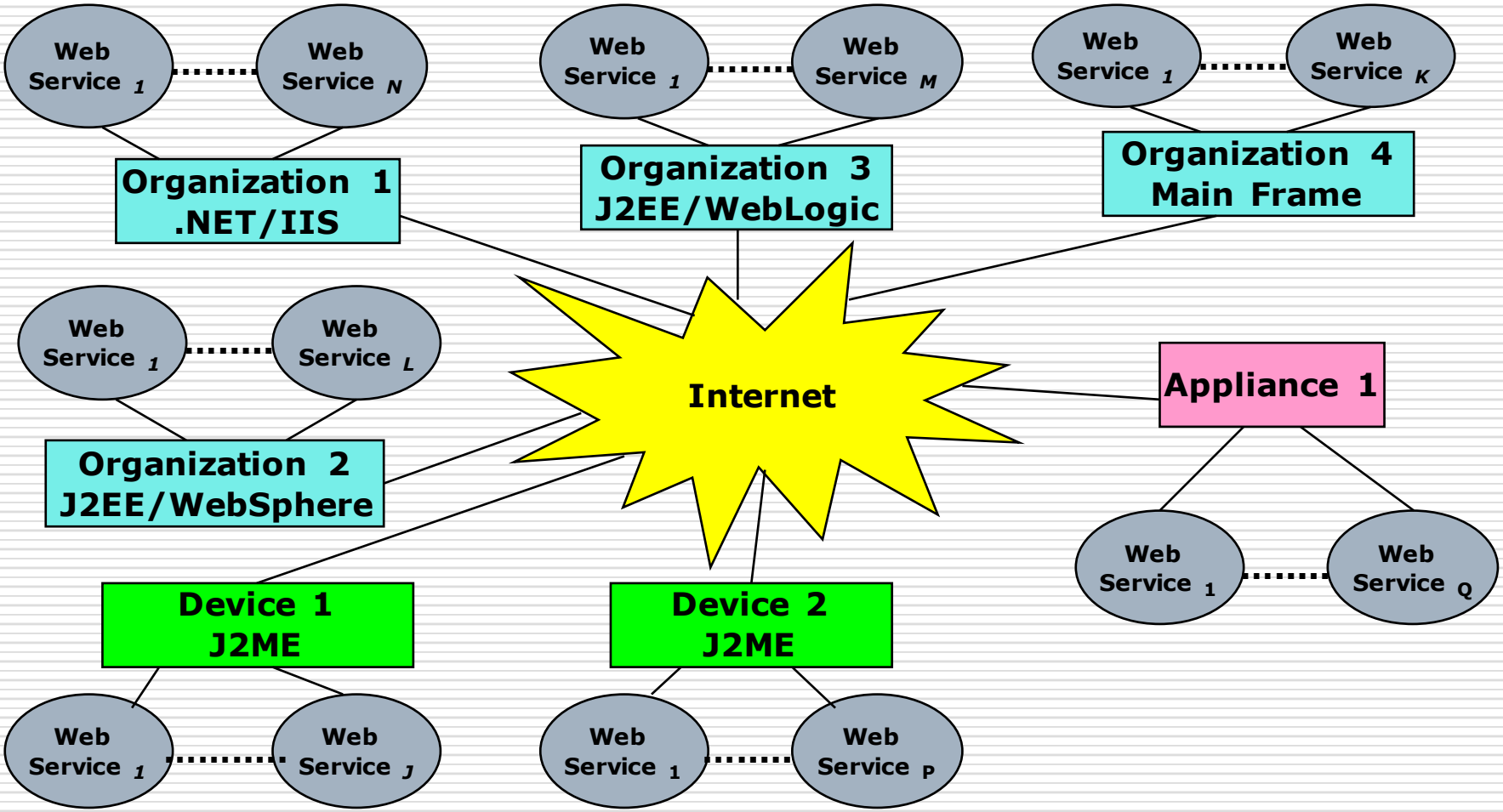
Platform Governance

Platform Governance

- Much of the benefit of SOA is derived from the promise of seamless interoperability between platforms
  - .NET
  - Windows Communications Foundation (WCF) consumer services exposed from COTS
  - Mainframe
  - Java applications
  - Embedded Systems
- **Core goal of SOA Governance is to ensure that services are relevant and consumable between platforms**

Source: *SOA Software, Inc., 2008.*

Platform Governance Cont'd



Platform Governance Cont'd

- Not all platforms are governable
- Platforms fall into one of 3 categories:
  - Ungoverned platforms, Self-governed platforms, Governed platforms
- Ungoverned Platforms
  - The purest form of Informal Governance
  - This often results in "Random SOA" or "Accidental SOA"
  - This includes any container that doesn't support policy enforcement natively or with an agent
- Self-Governed Platforms
  - A mixture of formal and informal
  - Some tasks and activities are governed, some are not
  - SOA Governance is as weak as the weakest link in the chain
  - **Example:** Containers that use their own tooling without policy integration with a centralized enterprise SOA Governance solution

Source: *SOA Software, Inc., 2008.*

Platform Governance Cont'd

- Governed Platforms
  - A real or virtual organization exists
    - Devoted to the promotion of SOA programs and causes
    - Programs & causes are accepted as a fundamental part of an SOA culture
  - Governed Service Platforms have:
    - Clear job titles / responsibility support SOA Governance activities
    - Supports clear separation between implementation activities and governance activities
    - Provides standards-based governance integration interfaces
- Governed Platforms fall into 2 categories:
  - Governed Service Platforms
  - Governed Development Platforms

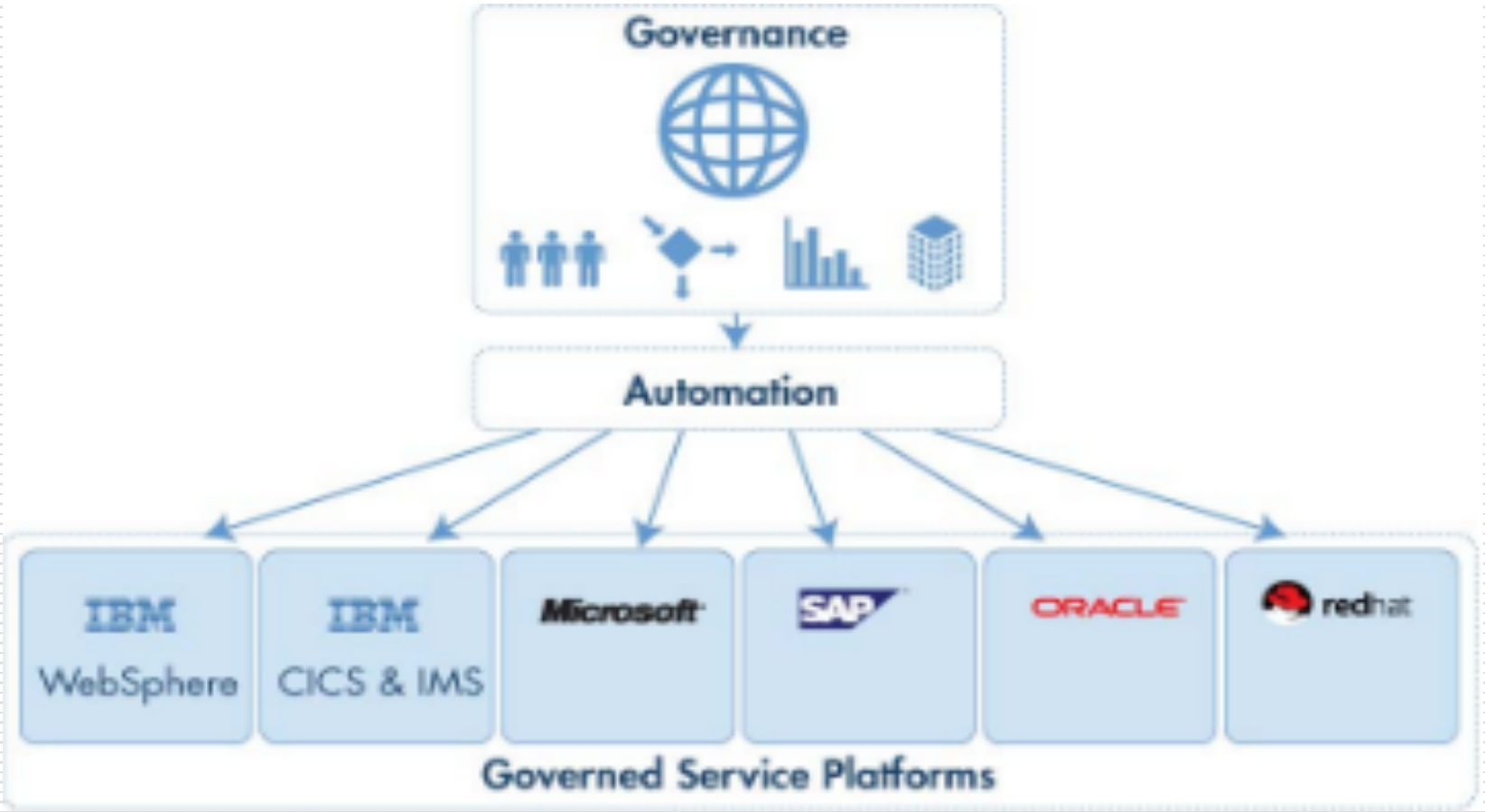
Source: *SOA Software, Inc., 2008.*

Platform Governance Cont'd

- Governed Service Platforms includes:
  - ✓ All applications that expose and consume services at runtime are service platforms
  - ✓ Application services like IBM WebSphere, Microsoft IIS, Oracle/BEA WebLogic, JBoss and others
  - ✓ ESBs from vendors including IBM, Microsoft Oracle/BEA, JBoss, TIBCO and others
  - ✓ Mainframe applications running in CICS and IMS
  - ✓ COTS applications like CICS
  - ✓ SaaS environments like Salesforce.com and Amazon.
- Governed Service Platforms offer standards-based governance integration interfaces
- They support the concepts of governance by an external enterprise governance system

Source: *SOA Software, Inc., 2008.*

Platform Governance Cont'd



Source: "Integrated SOA Governance for Microsoft", *SOA Software, Inc., 2008.*

Platform Governance Cont'd

- Governed Development Platforms
  - Most platform vendors provide:
    - An integrated development environment (IDE)
    - Source code management and version control system
    - Defect tracking/change request tooling
    - In many cases, a document management and/or asset management repository
  - An Integrated SOA Governance solution can provide:
    - Asset lifecycle management
    - Policy compliance capabilities
  - This ensures that developed software assets (such as services, components and applications) are:
    - Appropriate and relevant to the enterprise
      - They comply with applicable policies

Source: *SOA Software, Inc., 2008.*

SECTION 4

SOA BEST PRACTICES

Integrated SOA Best Practices

- **Governance Automation** ensures *scalability* of:
  - Enterprise processes implementing a lifecycle management workflow to implement development approval processes
  - Integrated provisioning and lifecycle management
  - Inter-departmental contract management and negotiation.

- **Uniform Policy Management** ensures *consistency* of the following through all stages of lifecycle and across all distributed and mainframe platforms:
  - Policy definition
  - Enforcement
  - Validation
  - Audit

Source: *SOA Software, Inc., 2008.*

Integrated SOA Best Practices Cont'd

- **Metadata Federation**
  - Provides seamless, heterogeneous SOA Governance and standards-based support for governance automation
    - UDDIv3
    - WS-MEX
    - WS-Policy
  - Ensures that governance processes are uniformly applied across all platform investments
  - Helps to expose the business value of a service (cost, usage, production issues) across the enterprise service lifecycle

Source: *SOA Software, Inc., 2008.*

Integrated SOA Best Practices Cont'd

- **Service Virtualization** provides:
  - Location-transparency
  - Service mobility
  - Impedance tolerance
  - Reliable service delivery
  - All of the above without requiring a re-platforming of existing platforms
  - All of the above without introducing yet another service platform to support the required solution architecture

Source: *SOA Software, Inc., 2008.*

Integrated SOA Best Practices Cont'd

- **Trust and Management Mediation**
  - Ensures interoperability across disparate partners and platforms
  - Ensures trust enablement and trust mediation complementing threat prevention systems
  - Provides last-mile security, metric collection and reporting, SLA monitoring and management
  - Ensures that services are governed, managed, and secured
  - Ensures that policy implementation and mediation to allow consumers to communicate with a wide range of mission critical business services are exposed from any platform.

Source: *SOA Software, Inc., 2008.*

Integrated SOA Best Practices Cont'd

- **Continuous Compliance and Validation**
  - Ensures consistent policy implementation
  - Ensures enforcement across all stages of the lifecycle
  - Preserves the fidelity of the governance models, structures and mechanisms supporting enterprise SOA programs
  - Ensures relevance, applicability and suitability of services
- **Change Impact Mitigation**
  - Provides change management and impact analysis processes
  - Processes are integrated with the governance workflow to ensure that changes to services or other assets don't cause major outages by breaking the consumption model

Source: *SOA Software, Inc., 2008.*

Integrated SOA Best Practices Cont'd

- **Consumer Contract Provisioning**
  - Provides offer, request, negotiation and approval workflows for service access, capacity, SLA and policy contracts
  - Ensures that the service providers know which applications and users are consuming their services
  - Allows providers to treat different consumers with different priorities and service levels

Source: *SOA Software, Inc., 2008.*

SECTION 5

STEPS TO IMPLEMENT SOA

7 Steps to SOA

1. Create/Expose Services
2. Register Services
3. Secure Services
4. Manage (monitor) Services
5. Mediate and Virtualize Services
6. Govern the SOA
7. Integrate Services

Source: *SOA Software, Inc., 2008.*

1. Create & Expose Services

- Three primary choices
  - Rebuild existing applications using SOA paradigm
  - Expose existing application logic as a set of services
  - A combination of rebuild and expose
- Enterprises typically use a combination of rebuild & expose
  - Solutions exist that facilitate migration of mainframe applications such as CICS to Web Services
- ***Granularity*** is a key criterion for Web Service
  - Functionality must be sufficiently coarse-grained
  - If coarse-grained, potential to be useful to different applications

Source: *SOA Software, Inc., 2008.*

2. Register Services

- Application architects & developers need to know that a service exists
- Use a registry
  - UDDI compatibility important
  - Search and Browse capability
  - Facilitate quick and accurate discovery of services
  - Some vendors have extended registries to repositories

Source: *SOA Software, Inc., 2008.*

3. Secure Services

- May have inadvertently created gaping security holes
- My have exposed sensitive information
- 5 principles of security
  1. Authentication
     - Basic HTTP authentication, SAML, X.509 signature
  2. Authorization
     - Leverage solutions such as CA SiteMinder, IBM TAM
  3. Privacy
     - XML-Encryption
     - Key & certificate management & distribution capabilities
  4. Non-Repudiation
     - Requestor & Sender cannot deny activities
  5. Auditing
     - Accurate accounting of requests & responses

Source: *SOA Software, Inc., 2008.*

4. Manage Services

- Look for potential disaster
  - Too many applications consuming a service?
  - Is the load reasonable
  - Is there a degradation in performance?
- Need to be able to monitor for
  - Basic Availability
  - Performance
  - Throughput
  - SLA agreement

Source: *SOA Software, Inc., 2008.*

5. Mediate & Virtualize Services

- As SOA matures may need to:
  - Introduce new versions
  - Increase capacity by running multiple instances
  - Provision applications to use specific instances of services
- Solution is Virtualization
  - Virtual service is a new service
    - Own WSDL, network address, transport parameters
    - Doesn't implement business logic
    - Acts as proxy to one or more physical services
    - Routes, load-balances, transforms, mediates
- XML transformation can be used to allow consumers to use an old version of service that no longer exists
  - Request & response transformed

Source: *SOA Software, Inc., 2008.*

Mediate & Virtualize Services Cont'd

- Consumers can select specific operations from multiple different services & combine them into a single functional WSDL
- Consumers can provide different policy requirements for different classes of users
- Transport bridging can be provided
  - E.g. HTTP and JMS
- Meditation between different standards implementation or versions of standards
- Mediation between different messaging styles
  - RSS, SOAP, REST, Plain old XML (POX)
- Content-or-context-based routing to deliver advanced load-balancing and high-availability capabilities

Source: *SOA Software, Inc., 2008.*

6. Govern the SOA

- Use a governance framework
- Design Time Issues
  - What types of services can be published?
  - Who can publish them?
  - What types of schema and messages services can accept?
  - What are the rules for the services?
- Run Time Issues
  - Security
  - Reliability
  - Performance
  - Compliance with policies

Source: *SOA Software, Inc., 2008.*

Govern the SOA Cont'd

- Tools needed for active participants
- Service Developer needs tools to:
  - Publish, categorize, define meta-data, virtualize
  - Choose policy, participate in capacity planning & access workflow
- Service Consumer needs tools to:
  - Facilitate service discovery, selection & access workflow
- Operations Staff need to:
  - Monitor service performance
  - Troubleshoot problems, monitor dependencies
  - Version services, virtualization & proxy management

Source: *SOA Software, Inc., 2008.*

Govern the SOA Cont'd

- Security Staff needs tools to:
  - Manage policy, report policy, check compliance, audit security
- Enterprise Architect needs tools to:
  - Monitor application, manage relationships
  - Define & validate design policy
  - Assign services to proxy
  - Virtualize services
- Enterprise IT Management
  - Manage reuse metrics
  - Gather service reuse statistics
  - Gather SOA statistics

Source: *SOA Software, Inc., 2008.*

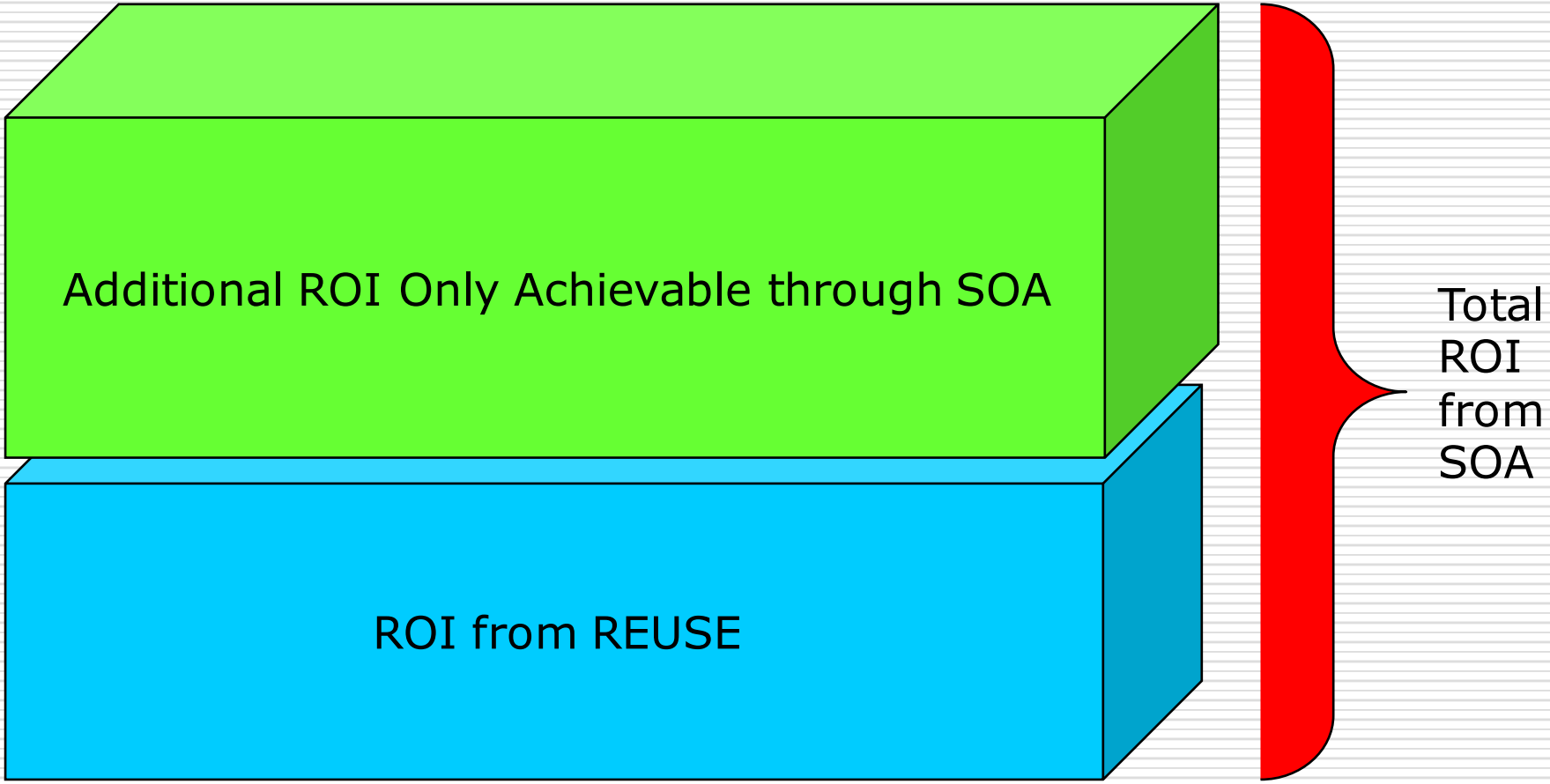
SECTION 6

ECONOMICS OF SOA

Overview of SOA ROI

- ROI for SOA is challenging for most organizations
  - Recall that ROI = Net Benefits/Investment
- Few organizations can provide ROI proofs, e.g. payback
- There is no single ROI model for SOA
- ROI realized at different phases of SOA implementation
- SOA is a long-term strategic investment
- A key area of research is to learn more about the economics of SOA including its **benefits, cost and cost justification model**
- A significant amount of research has been done on economics of **reuse**—benefits, cost and cost justification
  - See Lim [11], Boehm [2, 4] and Reifer [13, 15]
- **Reuse** is the underpinning of SOA

Total ROI: Reuse + Incremental SOA



Business Case Principles [See Reifer, Reference #13]

- Decisions are made relative to alternatives
- If possible, money should be used as a common denominator
- **Sunk costs** are irrelevant
- Investment decisions should recognize the **time value of money**
- Separable decisions should be considered separately
- Decisions should consider both quantitative and qualitative factors
- The risks associated with the decision should be **quantified** if possible
- The timing associated with making decisions is critical
- Decision processes should be **periodically assessed**
- Decision processes should be **continually improved**

Example Using ROI and Present Value

- Let us assume management is seriously considering funding your SOA proposal
- Your justification for the estimated expenditure is based on *shorter time to market*
- You believe you will save **$350,000.00 USD** a year if you invest **$250,000.00 USD** a year over **4** years
- Assume cost of money is **8%** per year
- Definitions
  - ROI = Net Benefits/Investment
  - Present Value (PV) = (Future Worth)/(1+ Rate/100)^N
    - N = Number of periods
  - Future Worth (FW) = (Present Value)*(1+ Rate/100)^N

Example Using ROI and Present Value

Year	SOA Implementation Investment	Benefits	Net Benefits	$1/(1 + i)^N$	Present Value of Net Benefits
1	$250,000,000	$350,000,000	$100,000,000	0.9259	$92,592,593
2	$250,000,000	$350,000,000	$100,000,000	0.8573	$85,733,882
3	$250,000,000	$350,000,000	$100,000,000	0.7938	$79,383,224
4	$250,000,000	$350,000,000	$100,000,000	0.7350	$73,502,985
	$1,000,000,000		**$400,000,000**		**$331,212,684**

Assumptions

1. Cost of money = 8%
2. 4-year Investment

ROI Over 4 Years: **40%**

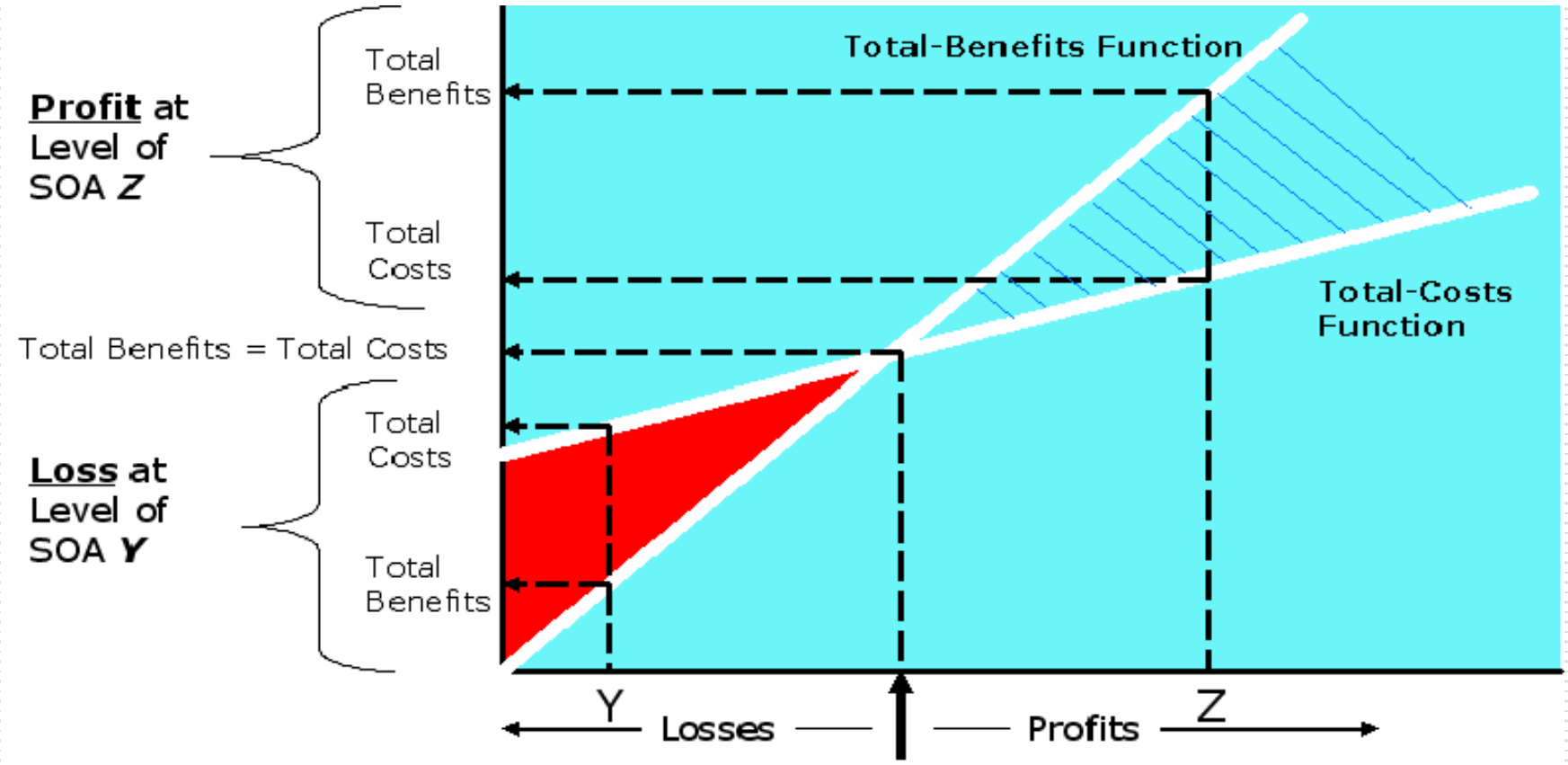
Annual ROI: **10%**

Discussion on ROI and Present Value

- Should the CEO/CFO/CIO invest with an annual ROI of 10%?
  - Why?
  - Why Not?
  - What return is acceptable? Does it depend on the corporate policy?

- What are your interpretations of the Present Value (PV)?
- What costs would you quantify as an SOA investment?
- What benefits would you quantify?
- What about some of the qualitative benefits?

Breakeven Analysis



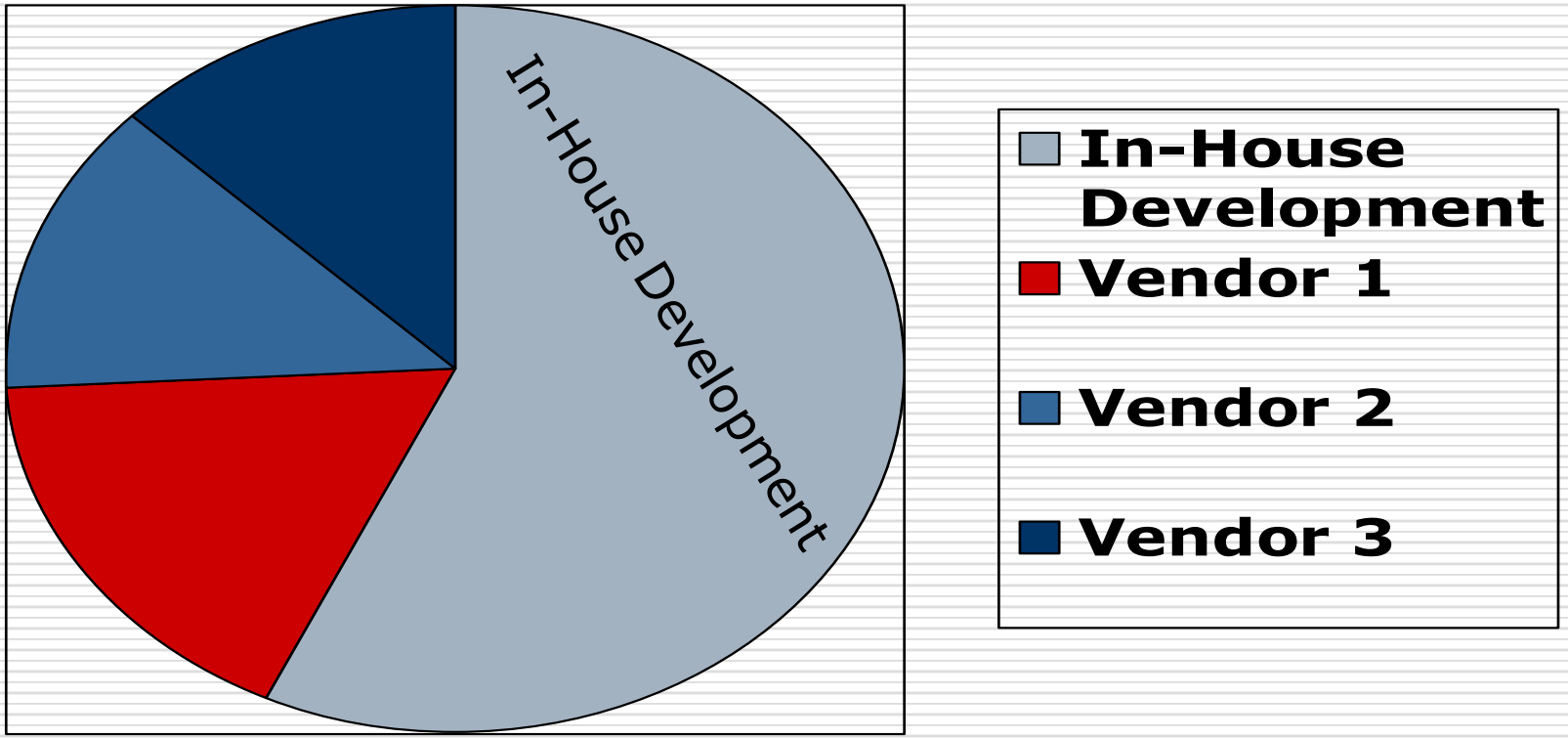
Source: Adapted from C. T. Horngren & G. Foster, *Cost Accounting: A Managerial Emphasis*, Prentice Hall, 1987, p. 51

Breakeven Analysis Cont'd

- Idea behind break even analysis is **volume**
- For an SOA environment, what **volume metric** makes sense?
  - # Individual Web Services implemented and consumed by at least ***N*** consumers?
    - What is a good number for ***N:*** 3, 4, 99?
- Is **volume** enough?
- What about **domain** coverage?
  - This is subjective
  - Is there a healthy ecosystem of WS that would create a trend towards a decline in number of lines of code?
- How do we construct the **Total-Benefits** & **Total-Costs** functions?

Build VS Buy Analysis

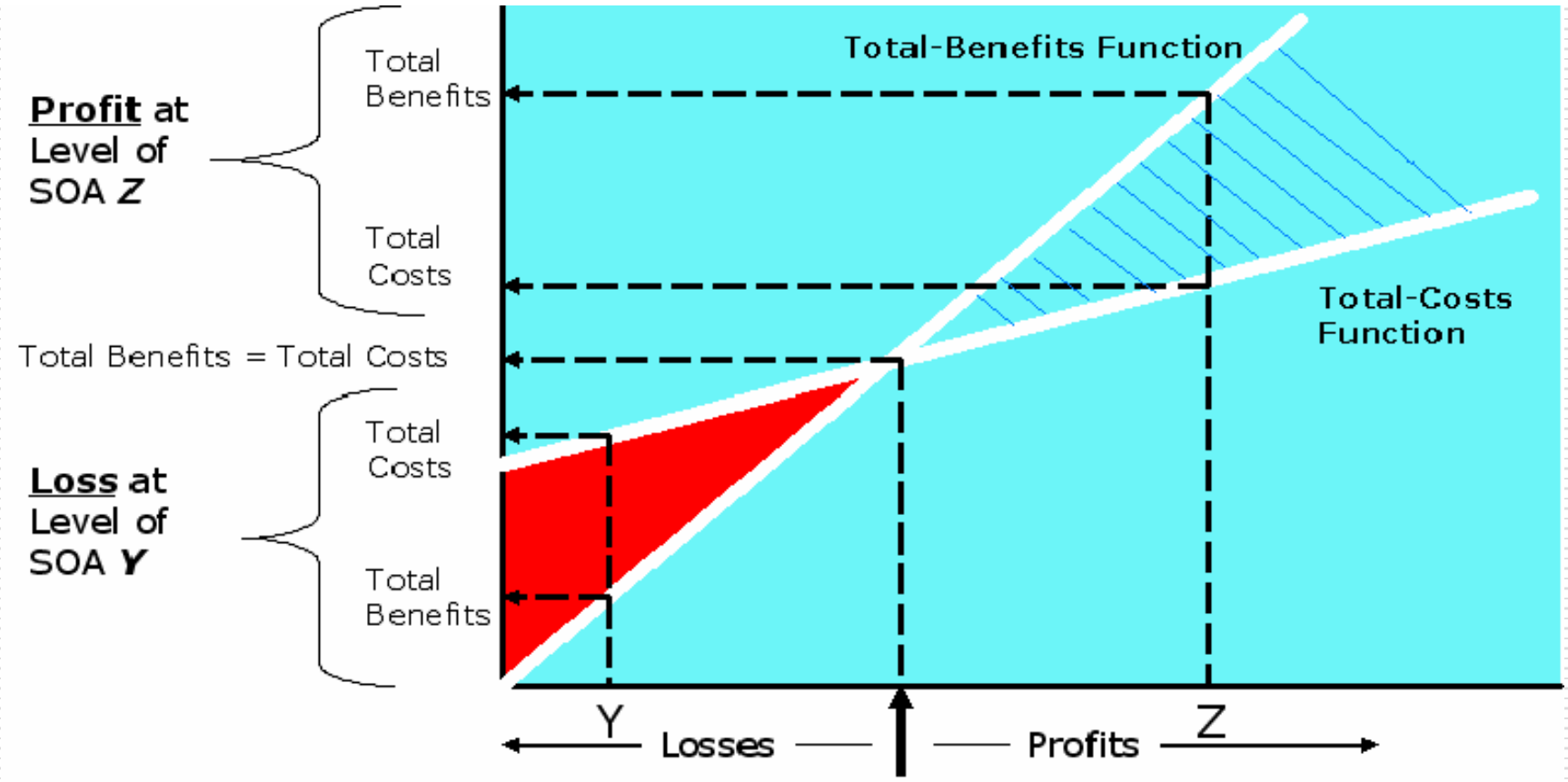
- Full domain could involve vendors who provide Web Services



Build VS Buy Analysis Cont'd

- Should think about extending set of available services through buying services (COTS)
- **Xignite** is a great example of a company that develops Web Services for Financial Services domain
  - Market Data
  - Company Data
  - Tools
  - xignite Financial Web Services
- **strikeiron** is another great example of a company that develops Web Services for various domains
  - Communications, CRM, Data Enhancement
  - Financial, Government, Lead Verification
  - Marketing, Other, Utilities, eCommerce
  - strikeiron Web Services

Breakeven Analysis Revisited

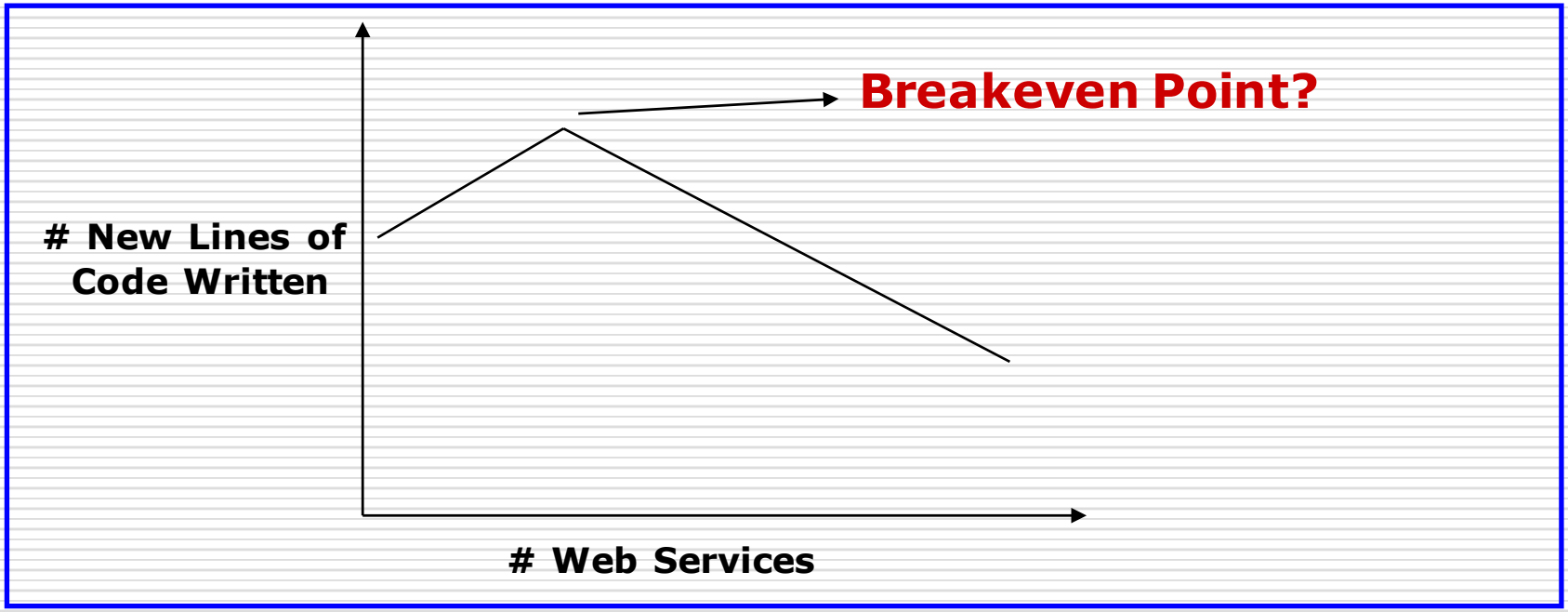


 # Web Services that provides volume required for Breakeven Level

Source: Adapted from C. T. Horngren & G. Foster, *Cost Accounting: A Managerial Emphasis*, Prentice Hall, 1987, p. 51

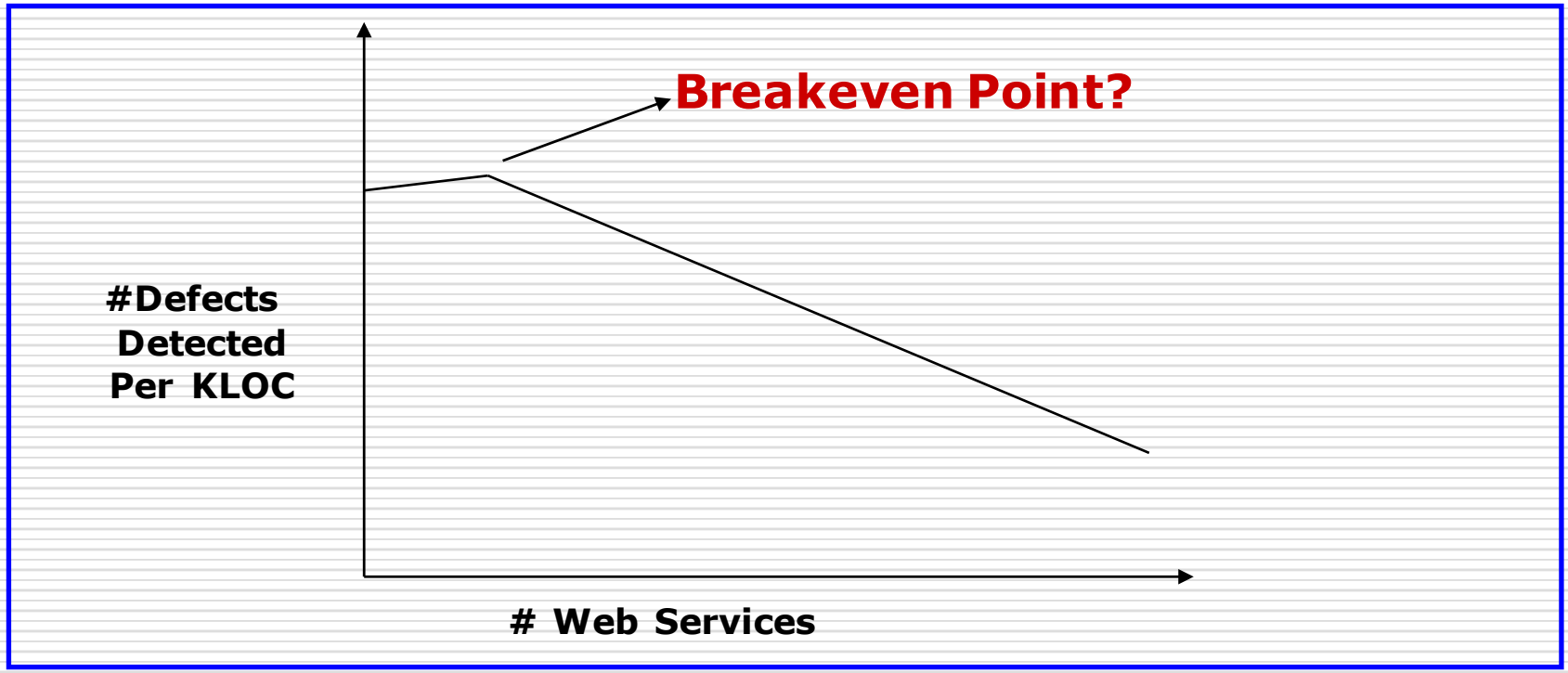
Trend Analysis: New Lines of Code

- Is there a healthy ecosystem of Web Services that would create a trend that indicates a consistent decline in **new** number of lines of code?
- Maybe after the breakeven point?



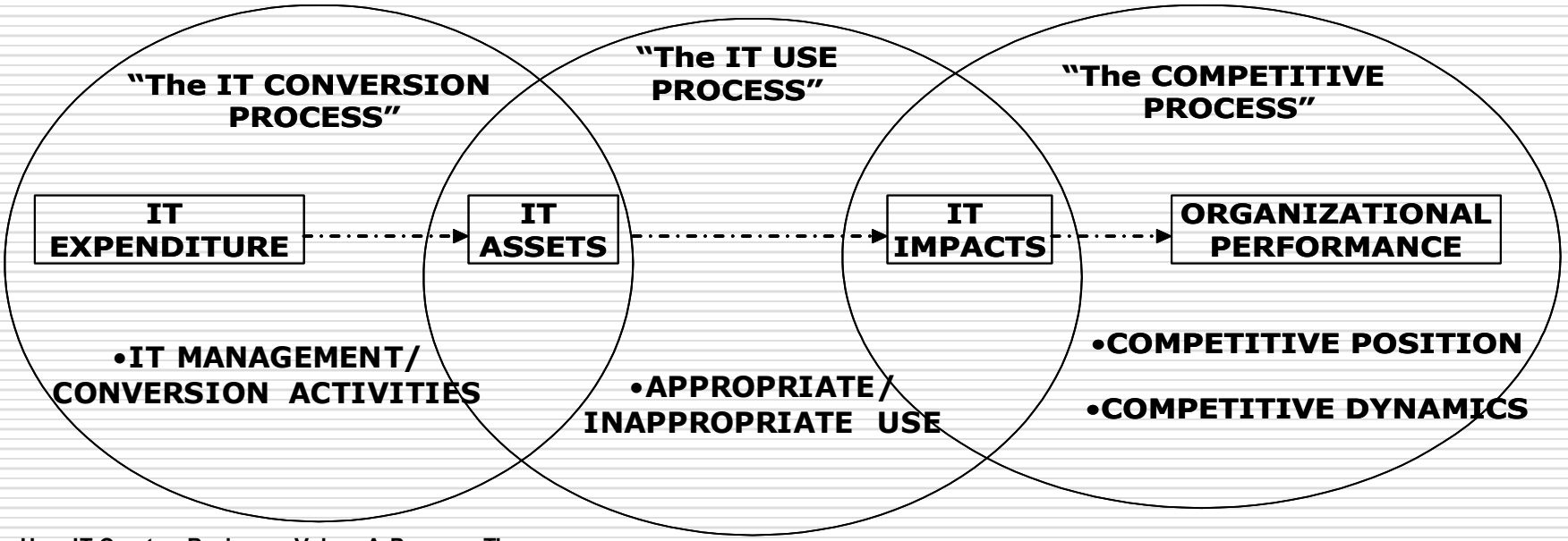
Trend Analysis: Quality

- What about **quality**?
- Is there a healthy ecosystem of WS that would create a trend that indicates consistent improvement in **quality**?



A Paradigm for SOA ROI

- Eric Marks and Michael Bell provide some insights into the complexity of ROI for SOA [52]
- They leverage the work of Soh and Markus to derive a value model for SOA



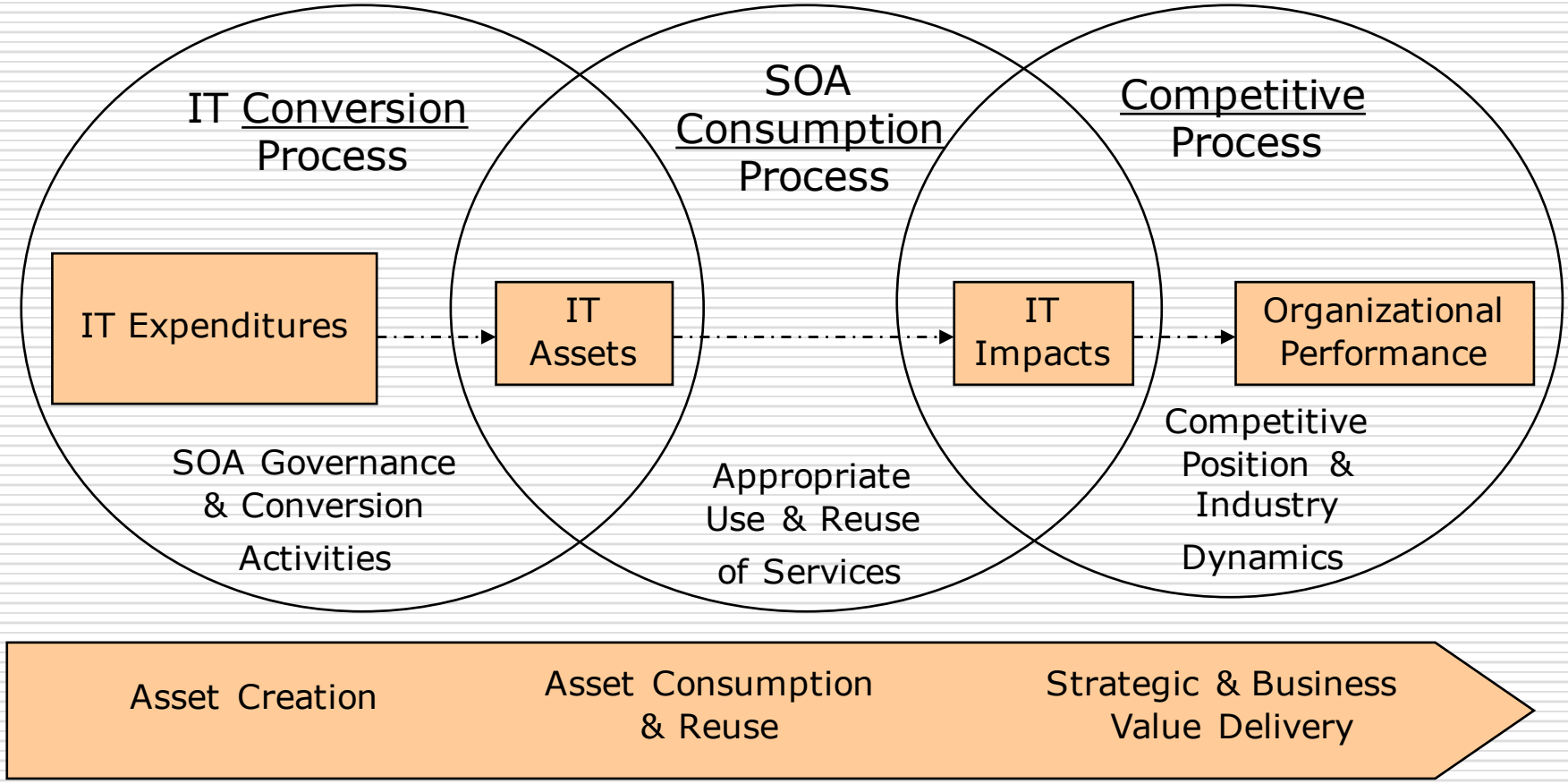
How IT Creates Business Value: A Process Theory

Source: *How IT Creates Business Value: A Process Theory Synthesis, Soh and Markus, pp. 29-41*

A Paradigm for SOA ROI Cont'd

- Marks' and Bell's Main Idea
  - Process of *conversion* creates SOA assets, or services
  - Assets or services are *consumed* by developers, analysts, customers & suppliers
  - This impacts the organization's *competitive* advantage
  - There are 3 broad value-creating processes:
    - *Conversion* value
    - *Consumption* value
    - *Competitive* value
  - Marks and Bell posit that there are ***multiple*** ROIs associated with each of the processes mentioned earlier

Summary of Value Creating Processes



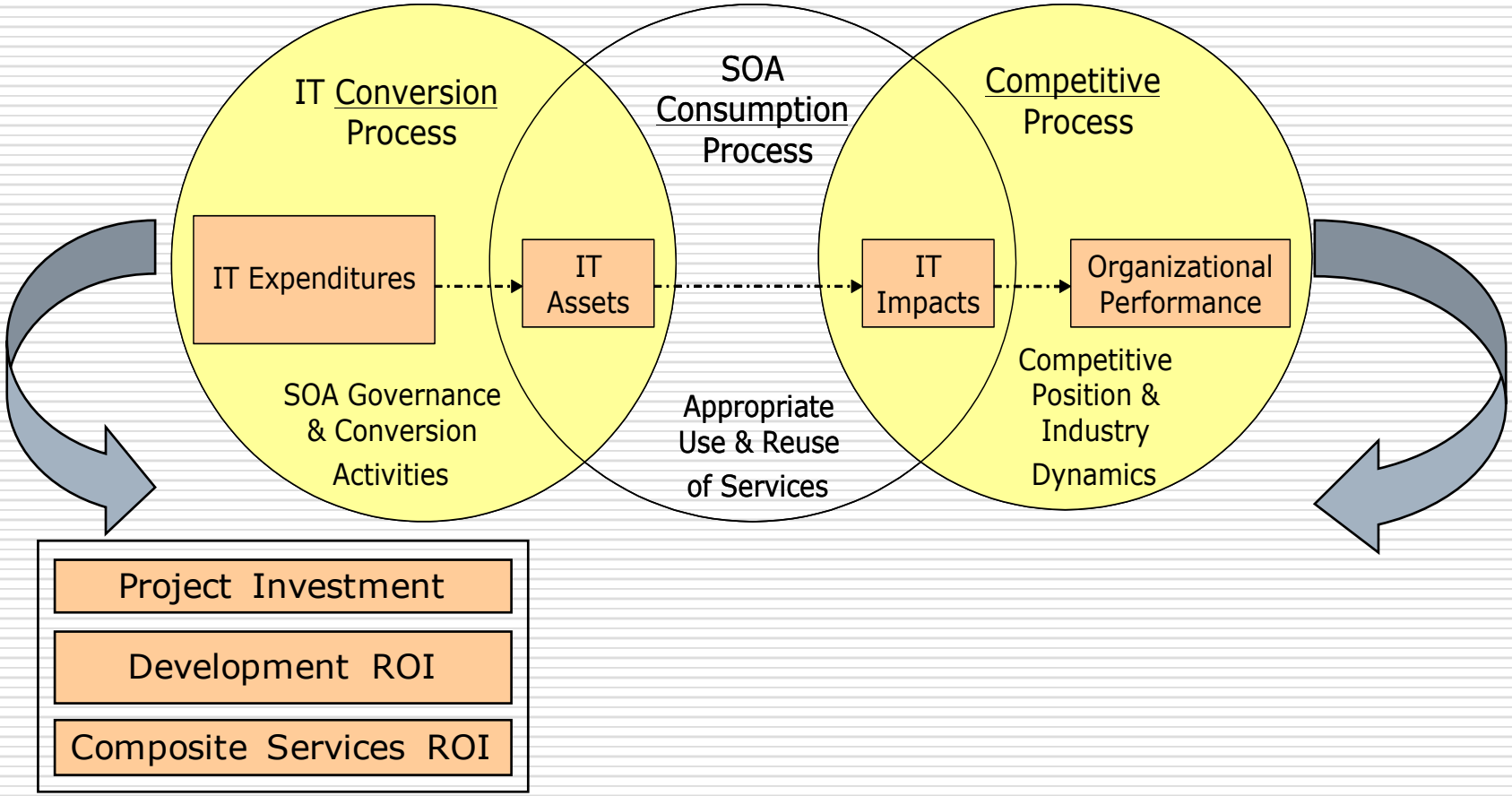
Process Approach to Creating SOA Value

Source: *Service Oriented Architecture: A Planning and Implementation Guide for Business and Technology, pp. 328*

Conversion Process ROI

- Project ROI
  - Reduced cost, reduced development time for a specific project
- Development ROI
  - Reduced development time
  - Better software quality
- Composite Services ROI
  - Faster development time using building block services
- Reuse ROI
  - Attained in subsequent iterations when enough services are able to be reduced
  - Initially this ROI may be small

Conversion Process ROI Summary



SOA ROI Threshold Model: Conversion Value

Source: *Service Oriented Architecture: A Planning and Implementation Guide for Business and Technology, pp. 338*

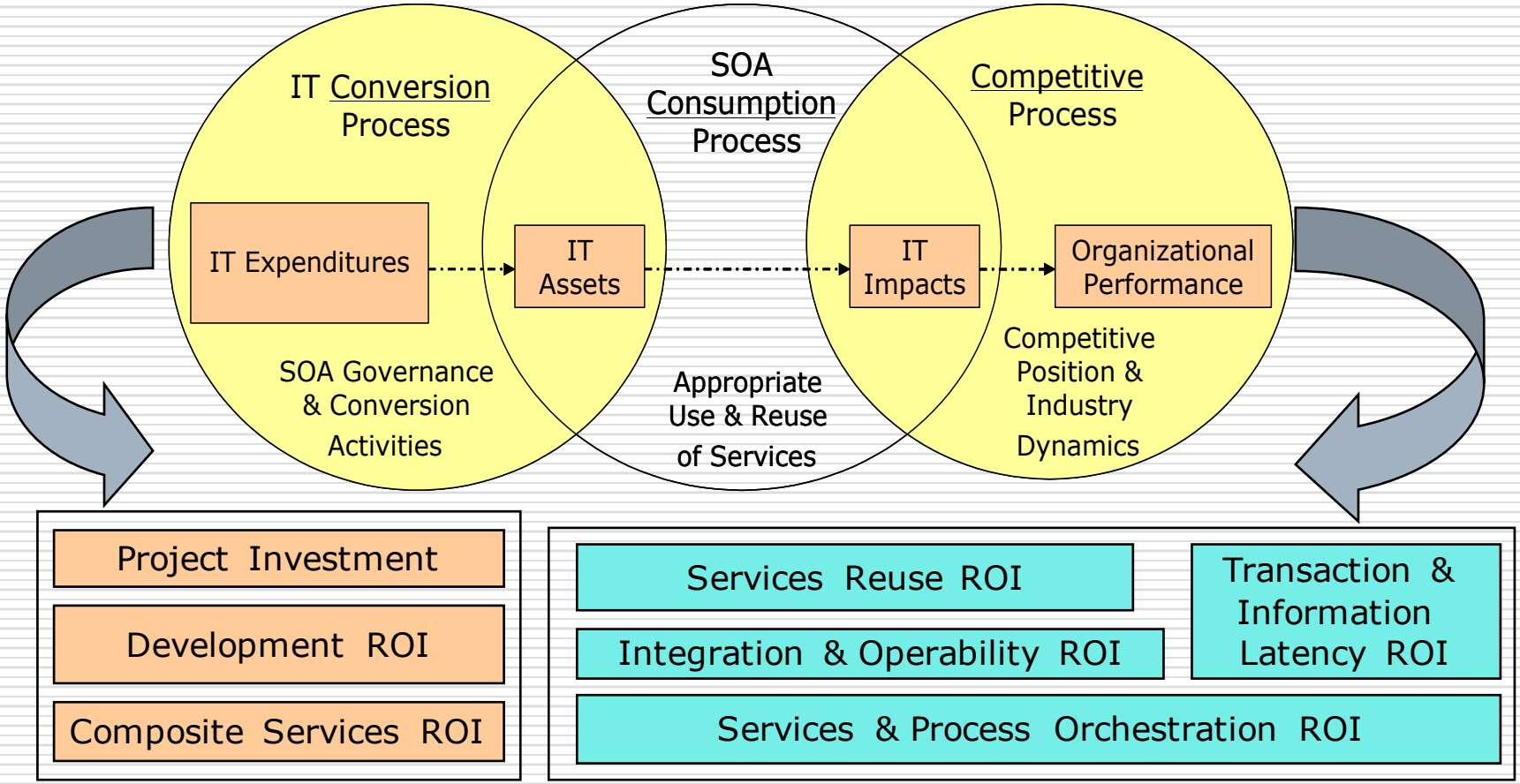
Consumption Process ROI

- Services Reuse ROI
  - Cost avoidance savings when services are reused
- Integration and Operability ROI
  - Cost avoidance from:
    - Implementing standards-based services rather than proprietary integration strategies
    - Reduced need to purchase licenses for proprietary integration middleware
    - Leveraging reuse of pre-built interoperable services to avoid point-to-point integrations common in the pre-SOA generation of IT
  - In an SOA, services are already integrated
  - No incremental integration tasks are required to make them interoperate

Consumption Process ROI Cont'd

- Services and process orchestration ROI
  - Benefits from orchestrating:
    - Composite services and applications
    - Business processes within enterprise
  - Additional benefits include:
    - Faster time to market for IT solutions and business initiatives
    - Lower development costs & reduced development time
    - Reduced maintenance of applications due to reuse
    - Additional levels of service reuse
- Transaction & Information Latency ROI
  - Includes benefits of removing stale info from business processes
  - Allows implementation of event-based services
    - Replaces batch-driven processes
    - Allows real time access to information

Consumption Process ROI Summary



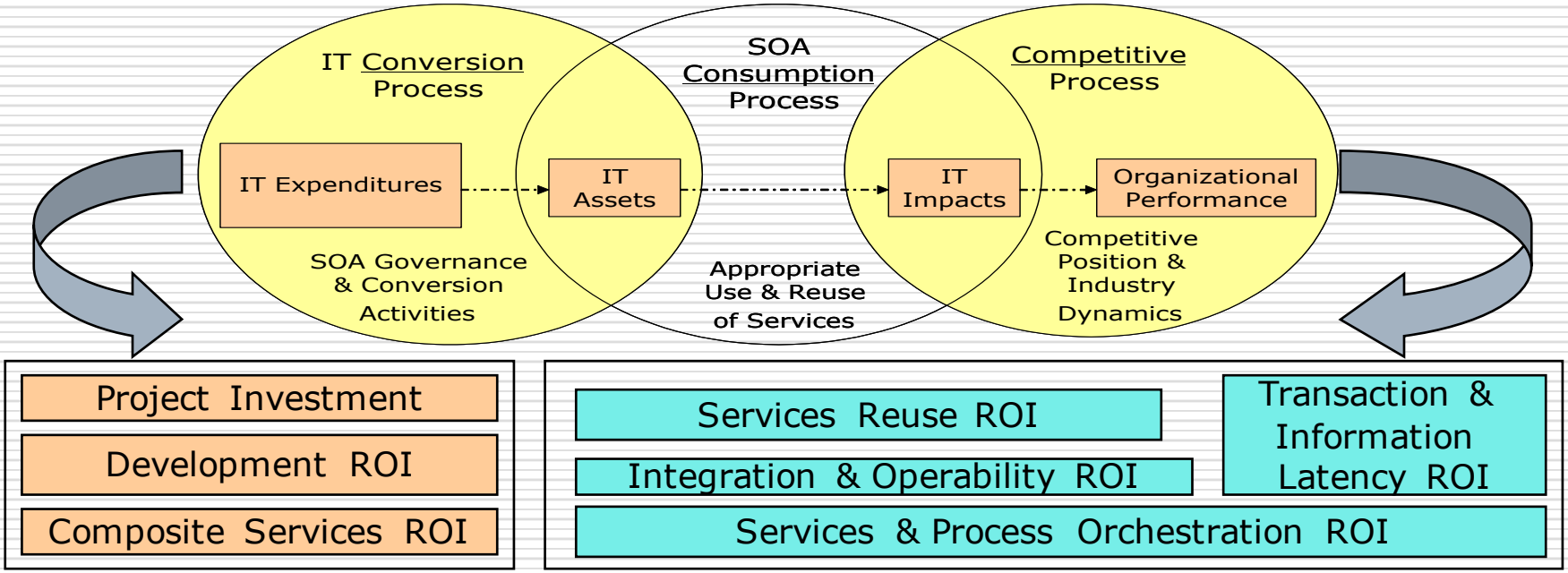
SOA ROI Threshold Model: Consumption Value

Source: *Service Oriented Architecture: A Planning and Implementation Guide for Business and Technology, pp. 328*

Competitive Value—SOA ROI

- Business Agility
- Faster Time to Market
- Cost Reductions
- Customer Satisfaction
- Revenue Growth
- Greater Productivity
- IT Flexibility
- Services & Software Reuse
- Faster M & A

Competitive Value—SOA ROI



Competitive Value in the SOA ROI Threshold Model

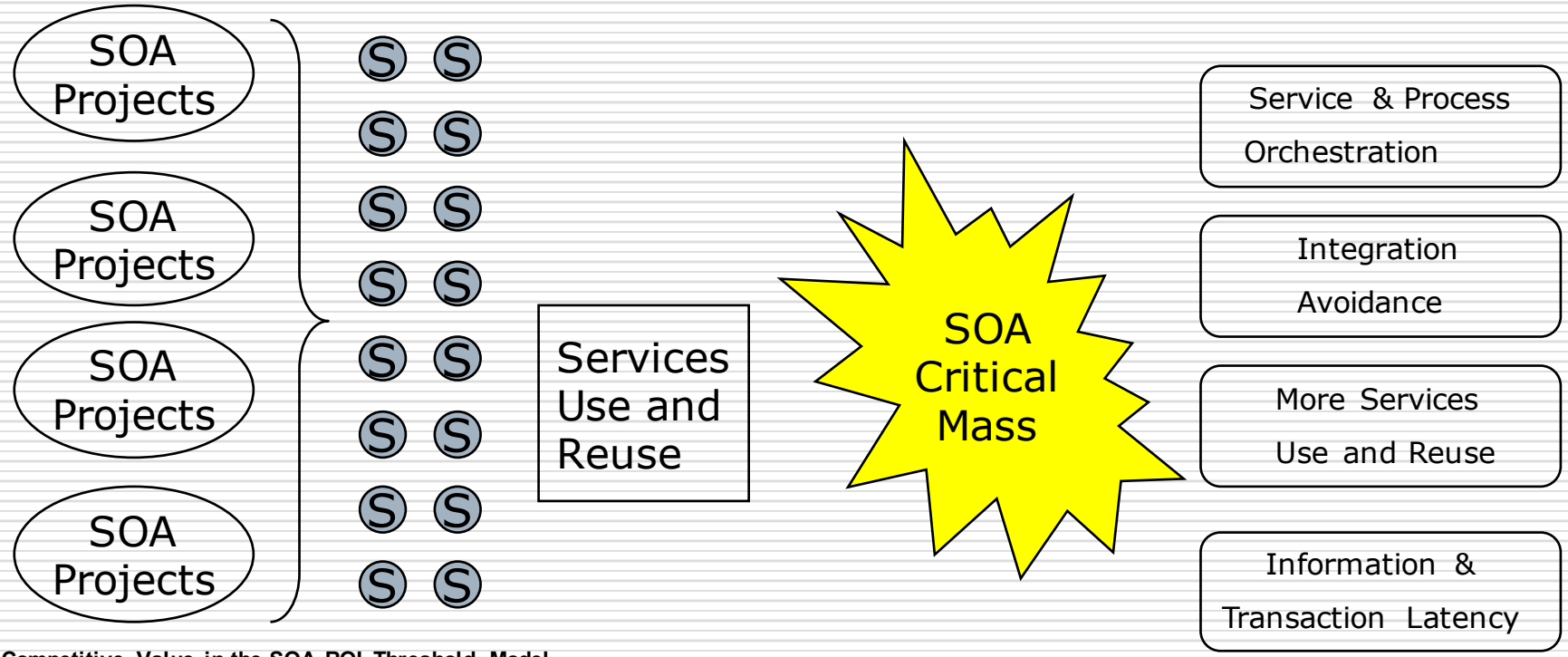
Source: *Service Oriented Architecture: A Planning and Implementation Guide for Business and Technology, pp.339*

SOA Needs Critical Mass

- Value of individual service is low until there are enough consumers (reuse) to accelerate return on services
- Value of an SOA increases as the volume of services and consumers increase
- Volume needs to hit critical mass
- SOA *network effects* kick in at that time
- SOA critical mass is:
  - Point where there are enough available reusable services
  - Such that one business process can be orchestrated from them

SOA Needs Critical Mass

SOA projects over time… result in many services…… that are used and reuse… until SOA critical mass is achieved… which accelerates SOA business value…



Competitive Value in the SOA ROI Threshold Model

Source: *Service Oriented Architecture: A Planning and Implementation Guide for Business and Technology, pp.339*

SECTION 7

SUMMARY

SOA: All Hype?

- In a profound sense, the industry hype about SOA is actually true.
  - It does work
  - It is being used in major deployments
  - It does cut costs and enable agility
  - It's an incremental shift that is possible to adopt without scrapping earlier IT efforts

Source: H. Taylor, "*Service-Oriented Architecture (SOA) 101 'What's Hype, What's Real?'*", *Juniper Networks, Inc.,2007.*

SOA Is Not a Silver Bullet

- Assumes costs and challenges inherent in reuse
- SOA does not make politics go away
- Your IT organization still has to master it
- Governance is a major challenge
- Security can be a big issue
- Vendors may not necessarily cooperate in an effort that commoditizes their products
- Vendors may be embedded in your organization, rendering some of the theoretical benefits of SOA moot
- Getting started with SOA may require longer and more expensive project cycles the first time around
  - Need high reuse **potential** & reuse **aptitude**
- Some SOA standards are still immature, leading to confusion and vendor-driven proprietary creep

Source: H. Taylor, "*Service-Oriented Architecture (SOA) 101 'What's Hype, What's Real?'*", *Juniper Networks, Inc.,2007.*

Benefits & Limitations of SOA

- Benefits
  - Flexibility in new software design
  - Reuse of business components in networks
  - Interoperability and integration capability
  - Ease of assembling new business processes
- Limits and Open Issues
  - Not a universal remedy for today's mix and match architectures
  - It is not a solution for all upcoming challenges
  - Not best practice for long-running asynchronous processes
  - Natural strengths in real-time request-response exchanges (asynchronous and synchronous)
  - SOA requires an environmental framework
    - .NET, SAP NetWeaver, IBM WebSphere, BEA WebLogic
    - Platform independence not yet achieved

Benefits & Limitations of SOA Cont'd

- Limits and Open Issues Cont'd
  - Most critical issues are pending **security issues**
    - Physical Network
      - Need Intra- and Inter-organizational security
    - SOAP Messages
      - Need to protect content of SOAP Messages
    - Endpoint (Web Service) Security
      - Need Intra- and Inter-organizational security
    - Extensive security framework worked out
    - Applicable products on the market
    - For mission critical processes, security measures still an issue
  - Not valuable for applications whose business logic components are in a closed application domain
  - Not valuable if there is no intention for reuse

SECTION 8

REFERENCES

References

1. B.W. Boehm and C. Gacek, "*Composing Components: How Does One Detect Potential Architectural Mismatches?*," USC Center for Software Engineering, Los Angeles, CA, 1999.
2. B.W. Boehm and W.L. Scherlis, "*Megaprogramming*," Proceedings of the DARPA Software technology Conference, April 1992 (available via USC Center for Software Engineering, Los Angeles, CA, 90089-0781).
3. B.W. Boehm, *Software Engineering Economics*, Prentice Hall, Englewood Cliffs, N.J., 1981.
4. B.W. Boehm, C. Abts, A.W. Brown, S. Chulani, B.K. Clark, E. Horowitz, R. Madachy, D. Reifer, and B. Steece, *Software Cost Estimation with COCOMO II*, Prentice Hall, Upper Saddle River, N.J., 2000.
5. Carnegie Mellon University, Software Engineering Institute (M.C. Pauk, C.V. Weber, B. Curtis, and M.B. Chrissis). *The Capability Maturity Model: Guidelines for Improving the Software Process*, Addison-Wesley, 1999.
6. F. DeRemer and H.H. Kron. "*Programming –in-the-large versus programming-in-the-small*," IEEE Transactions on Software Engineering, SE-2(2), June 1976, pp. 80-86.
7. D. Garlan, R. Allen, and J. Ockerbloom. "*Architectural Mismatch: Why Reuse Is So Hard*," *Software*, IEEE Computer Society, November 1995, pp. 17-26.
8. E.M. Hall, *Managing Risk: Methods for Software Systems Development*, Addison-Wesley, 1998.
9. W.S. Humphrey, *Managing the Software Process*, Addison-Wesley, 1989.
10. I.M. Jacobson and P. Jonsson, *Software Reuse: Architecture, Process, and Organization for Business Success*, Addison-Wesley, 1997.
11. W.C. Lim, *Managing Software Reuse*, Prentice Hall, 1998
12. S. Vinoski, "*CORBA: Integrating Diverse Applications Within Distributed Heterogeneous Environments*," IEEE Transactions on Software Engineering, 1996.
13. D. J. Reifer, *Making the Software Business Case*, Addison-Wesley, Upper Saddle River, N.J., 2000.
14. J.S. Poulin, *Measuring Software Reuse: Principles, Practices, and Economic Models*, Addison-Wesley, 1997.
15. D.J. Reifer, *Practical Software Reuse*, John Wiley & Sons, 1997.
16. C.K. Prahalad and G. Hatmel "*The Core Competence of the Corporation*," Harvard Business Review, May-June, 1990.
17. B.W. Boehm, "*Software Risk Management: Principles and Practices*," IEEE Transactions on Software Engineering, January 1991, pp. 32-41.
18. M. Shaw and D. Garlan, *Software Architecture: Perspectives On An Emerging Discipline*, Prentice Hall, Upper Saddle River, N.J., 1996.
19. C.J. Date, *An Introduction to Database Systems Volume II*, Addison-Wesley, 1983.
20. R.N. Taylor, N. Medvidovic, K.M. Anderson, E. J. Whitehead Jr., J.E. Robbins, K.A. Nies, P. Oreizy, and D. L. Dubrow, "*A Component-and Message-Based Architectural Style for GUI Software*." (Not published in any paper)
21. D. E. Perry, "*Generic Architecture Descriptions for Product Lines,*" Bell Laboratories, Murray Hill, N.J. (Not published in any paper)

References

22. R. T. Fielding, "*Software Architectural Styles for Network-based Applications*," University of California, Irvine, July 15, 1999.
23. B.W. Boehm and T.A. Standish, "*Software Technology in the 1990's: using an evolutionary paradigm*," Computer, vol. 16, pp. 30-7, 1983.
24. "*Failed technology projects(The Standish Group Report)*," in Investor's Business Daily. Los Angeles, Jan. 1995, pp. A8.
25. B. Bongard, B. Gronquist, and D. Ribot, "*Impact of Reuse on Organizations*," Cap Gemini Innovation, Esprit project REBOOT, Grenoble, France, Sept. 4, 1992.
26. N. Buxton and R. Malcolm, "*Software technology transfer*," *Software Engineering journal,* vol. 6, no.1, pp. 17-23, Jan., 1991.
27. G. Caldiera, "*Domain Factory and Software Reusability*," Software Engineering Symposium: New Frontiers for Software Maintenance, May, 1991.
28. T. Durek, "*Strategies and Tactics for Software Reuse Tutorial*," presented at Improving the Software Process and Competitive Position via Software Reuse and Reengineering, Alexandria, V A, 1991.
29. R. Holibaugh, s. Cohen, K. Kang, and S. Peterson, "*Reuse: where to begin and why*," Proceedings. TRI-Ada '89, pp. 266-77, Oct. 23-26,1989.
30. R. Joos, "*Software Reuse in an Industrial Setting*," 4th Annual Workshop on Software Reuse, Nov.18-22, 1991.
31. D. Parkhill, "*Object-oriented technology transfer: techniques and guidelines for a smooth transition*," Object Magazine, pp. 57-59, May/June, 1992.
32. R. Prieto-Diaz, "*Making software reuse work: an implementation model*," SIGSOFT Software Engineering Notes, vol. 16, no.3, pp. 61-8, July, 1991.
33. "*Reuse adoption guidebook*," Software Productivity Consortium, Hemdon, VA, SPC-92051-CMC, Version 01.00.03, Nov., 1992.
34. "*Software Reuse Guidelines*," U.S. Army Information Systems Engineering Command (USAISE), ASQB-GI-90-015, Apr ., 1990.
35. B. Whittle, W. Lam, and T. Kelly, " *A pragmatic approach to reuse introduction in an industrial setting,*" Systematic Reuse: Issues in Initiating and Improving a Reuse Program. Proceedings of the International Workshop on Systematic Reuse, pp. 104-15, 1996.
36. T. J. Biggerstaff, " *An Assessment and Analysis of Software Reuse* " in *Advances in Computers,* vol. 34, M. C. Yovits, Ed., New York, N. Y .: Academic Press, 1992
37. T. Davis, "*Adopting a policy of reuse*," *IEEE Spectrum,* vol. 31, pp. 44-8, June 1994.
38. W. B. Frakes and C. J. Fox, "*Sixteen questions about software reuse*," Communications of the ACM, vol. 38, pp. 75-87, 112, June 1995.
39. A. J. Incorvaia, A. M. Davis, and R. E. Fairley, "*Case studies in software reuse*," presented at Fourteenth Annual International Computer Software and Applications Conference (Cat. No.90CH2923-1), 1990.
40. R. M. Sonnemann, "*Exploratory study of software reuse success*," Ph.D. dissertation, Dep. Engineering, George Mason University, Fairfax, VA, 1996.
41. E.M. Rogers, *Diffusion of Innovations*, Simon & Schuster, 1995.

References

41. B.W. Boehm, *Competing on Schedule, Cost, and Quality : The Role of Software Models*, USC Center for Software Engineering, August, 2001.
42. C.J. Date, *An Introduction to Database Systems Volume II*, Addison-Wesley, 1983.
43. A. S. Tanenbaum, *Distributed Operating Systems*, Prentice Hall, 1995.
44. T. B. Bollinger and S. L. Pfleeger, "*The Economics of Software Reuse*," Contel Technology Center, Chantilly, VA, Tech. Report CTC-TR-89-014, Dec. 13, 1989
45. J. E. Gaffney and T. A. Durek, "*Software Reuse-Key to Enhanced Productivity: Some Quantitative Models*," Software Productivity Consortium, Herndon, V A, Tech. Report SPC- TR-88-015, Apr., 1988.
46. E. Guerrieri, L. A. Lashway, and T. B. Ruegsegger, "*An Acquisition Strategy for Populating a Software Reuse Library*," National Conference on Software Reusability, July 19-20, 1989.
47. W. C. Lim, "*A Cost-Justification Model for Software Reuse*," 5th Annual Work- this shop for Institutionalizing Software Reuse, Oct. 26-29, 1992.
48. R. A. Malan and K. Wentzel, "*Economics of Reuse Revisited*," Hewlett Packard Laboratories, Palo Alto, CA, Technical Report, HPL-93-31, Apr., 1993.
49. J. S. Poulin and J. M. Caruso, "*A reuse metrics and return on investment model*," Proceedings Advances in Software Reuse. Selected Papers from the Second International Workshop on Software Reusability (Cat. No. 93THO495- 2), pp. 152-66, Mar. 24-26, 1993.
50. B. Bloom, *Deploying and Managing Microsoft .NET Web Farms*, Sams Publishing, 2001.
51. H. Taylor, *"Service-Oriented Architecture (SOA) 101 'What's Hype, What's Real?", Juniper Networks, Inc.,2007.*
52. Wiehler, Gerard. Mobility, Security and Web Services: Technologies and Service-Oriented Architectures for a new Era of IT Solutions. Publicis Corporate Publishing, 2004.
53. Pulier, Eric and Hugh Taylor. Understanding Enterprise SOA. Manning Publications Co., 2006.
54. G. B. Machado, F. Siqueira, R. Mittmann, C. A. Vieira e Vieira. *"Embedded Systems Integration Using Web Services∗,"* Proceedings of the International Conference on Networking, International Conference on Systems and International Conference on Mobile Communications and Learning Technologies (ICNICONSMCL'06),0-7695-2552-0/06,IEEE Computer Society, 2006.
55. S. Karnouskos, O. Baecker, L. M. S. de Souza, and P. Spie. *"Integration of SOA-ready Networked Embedded Devices in Enterprise Systems via a Cross-Layered Web Service Infrastructure,"* 12th IEEE Conference on Emerging Technologies and Factory Automation, September 25-28, 2007, Patras, Greece.
56. S. Furr. *"Implementing Web Services in Embedded Systems,"* Embedded Systems Conference Boston Class #206, QNX Software Systems, 2004.
57. K.C. Thramboulidis, G. Doukas, G. Koumoustos. *"An SOA-Based Embedded Systems Development Environment for Industrial Automation,"* EURASIP Journal on Embedded Systems, Volume 2008, Article ID 312671.
58. J. Canosa, *"Embedded System Design,"* Embedded.com, 02/01/02.
59. R. Heffner, *"Embedded SOA Management Solutions,"* Forrester Research, Inc., October 22, 2007.

References

60. C. T. Horngren & G. Foster, *Cost Accounting: A Managerial Emphasis*, Prentice Hall, 1987.