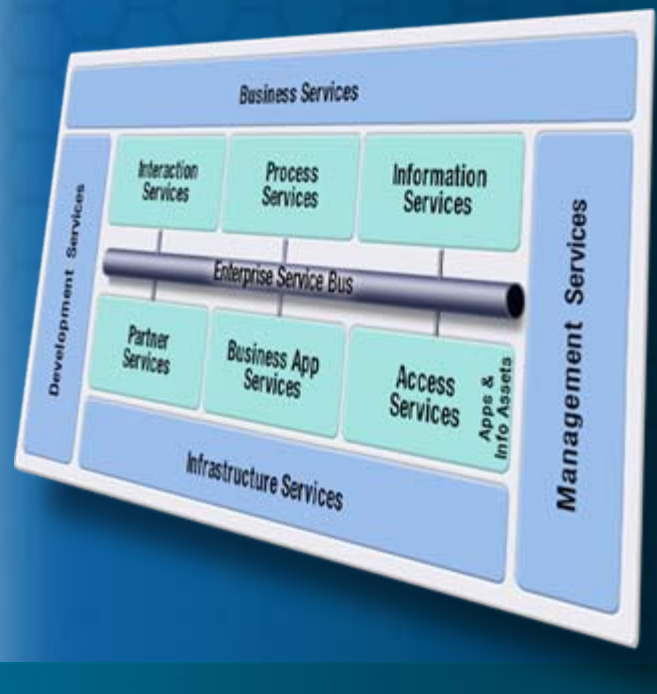
IBM SOA Architect Summit

SOA on your terms and our expertise

IBM SOA Architect Summit

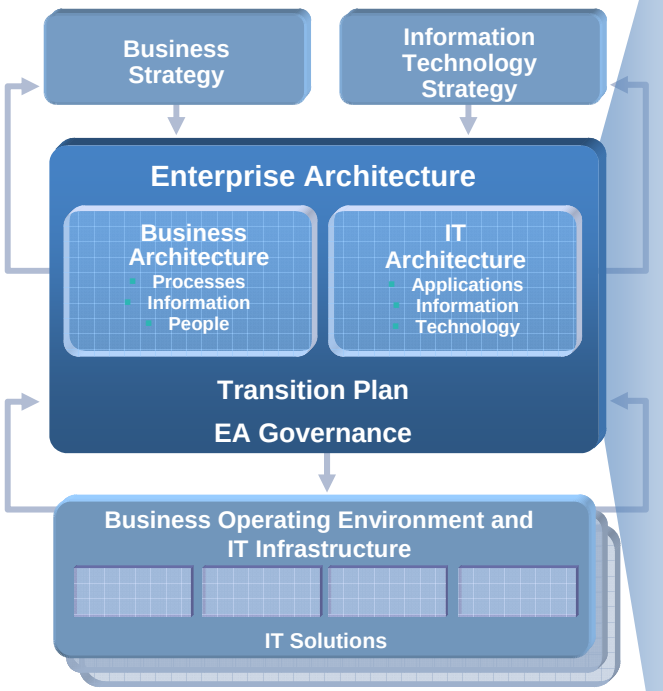
Application Architecture:

Reusing Existing Applications in SOA-Based Business Processes



Manuel Rodriguez
SOA Software IT Architect,
IBM Software Group

SOA Architect Summit Roadmap



What is the impact of SOA on current Enterprise Architectures?

- Alignment of Business and IT Architectures
- SOA Reference Models
- SOA Governance

How do you develop SOA with a business focus?

- Business Components
- SOA Design
- Business Process Management

How do you reuse applications in the context of SOA?

- Asset Discovery
- Application Reuse

How do you leverage information in an SOA?

- Information as a Service
- Master Data Management

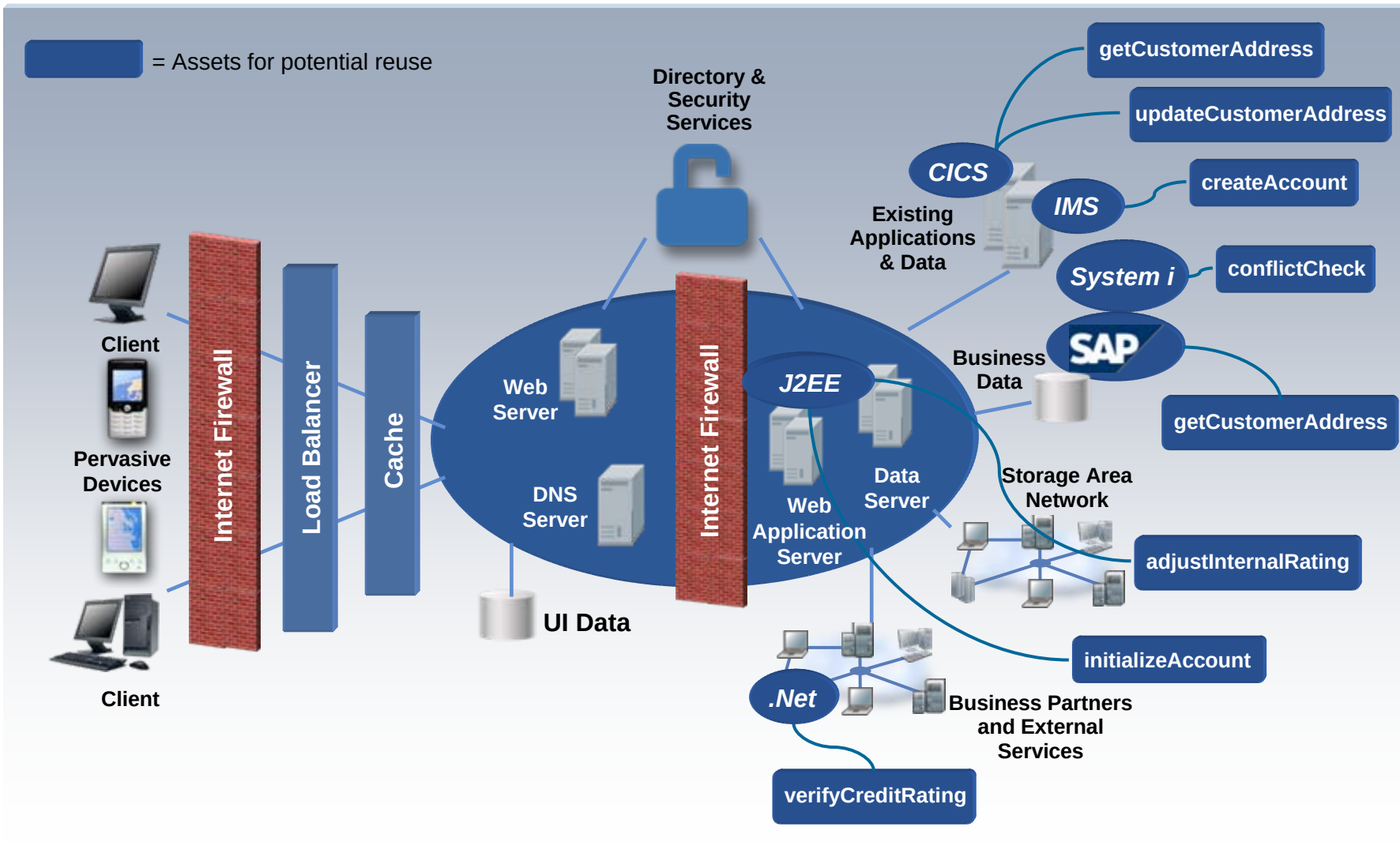
How does my infrastructure support SOA?

- Service Management / QoS
- Security

SOA and Application Architecture

- SOA Application Architecture Considerations
- SOA Application Architecture Best Practices
- IBM Capabilities to Support SOA Application Architecture
- Summary

SOA Enables Greater Reuse of Existing Assets



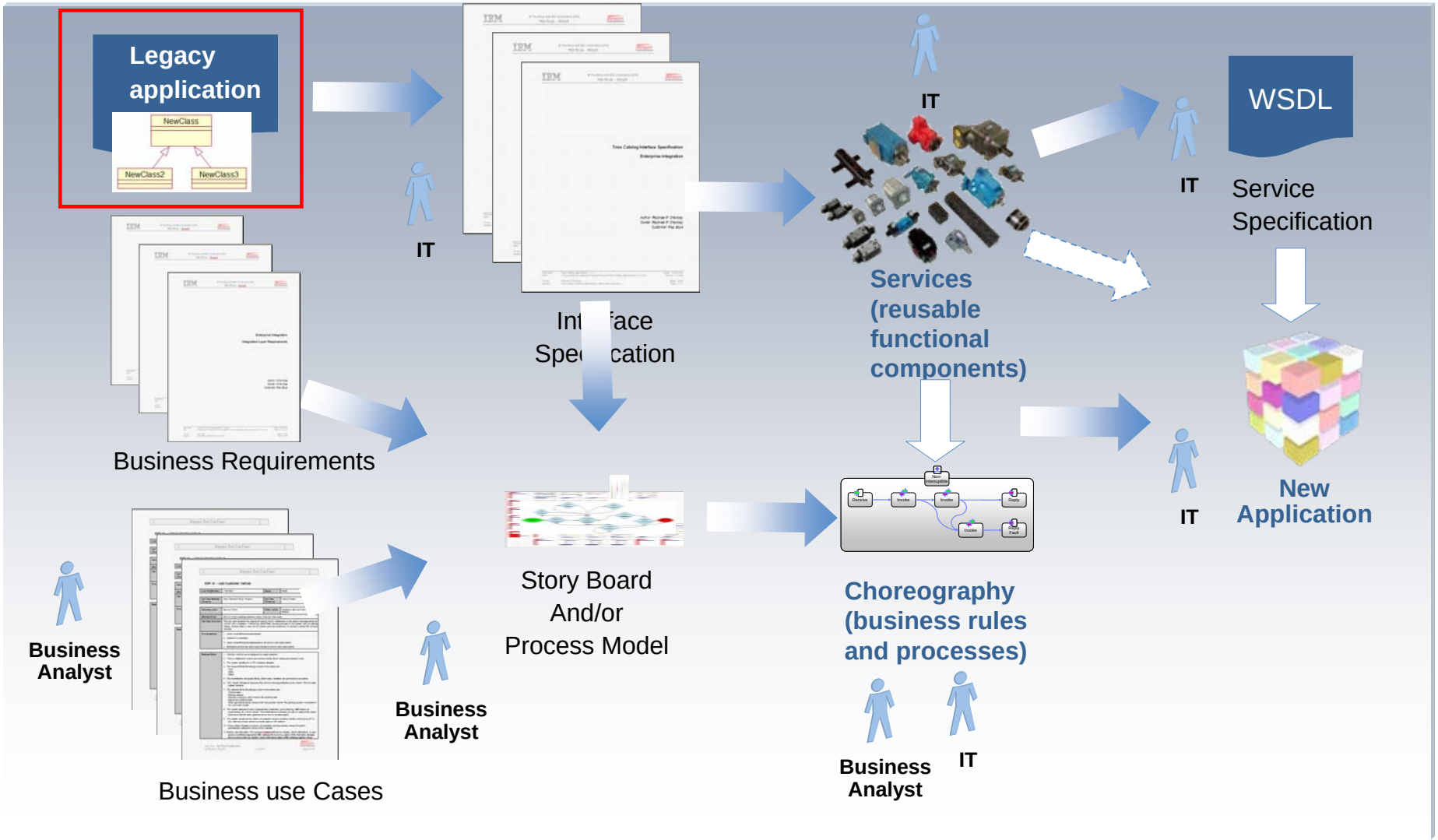
Application Architecture Considerations

- Analyze business processes to discover services
  - Identify services required to perform the individual tasks defined by a given business process
  - Analyze existing applications to identify service providers
- Creating services
  - Use best practices (defined as patterns) to create services from existing assets
    - Use tools and standards to service enable an asset
  - Use externally provide services to support commodity tasks
  - Fill in gaps by creating new services
- Connecting to service providers
  - Enable "any-to-any" linkage and communication between services inside and beyond the enterprise
  - Simplify connectivity by providing infrastructure that ensures Qualities of Service (QoS) including security, reliability, and scalability

SOA and Application Architecture

- SOA Application Architecture Considerations
- SOA Application Architecture Best Practices
  - **Asset Discovery Approaches**
  - Reuse Patterns
  - Service Connectivity Scenarios
- IBM Capabilities to Support SOA Application Architecture
- Summary

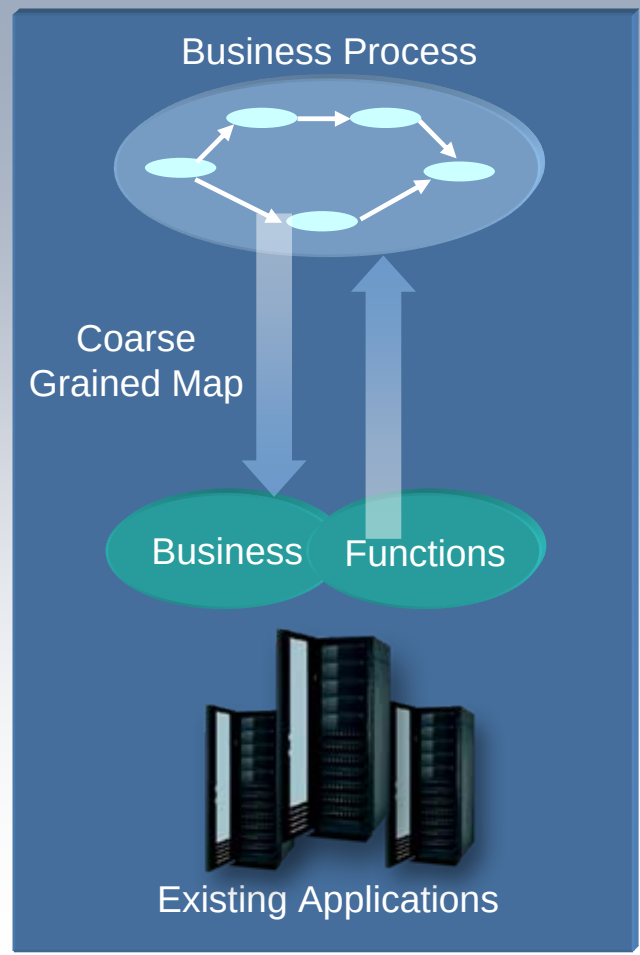
Bottom Up SOA Approach



Existing Asset Analysis

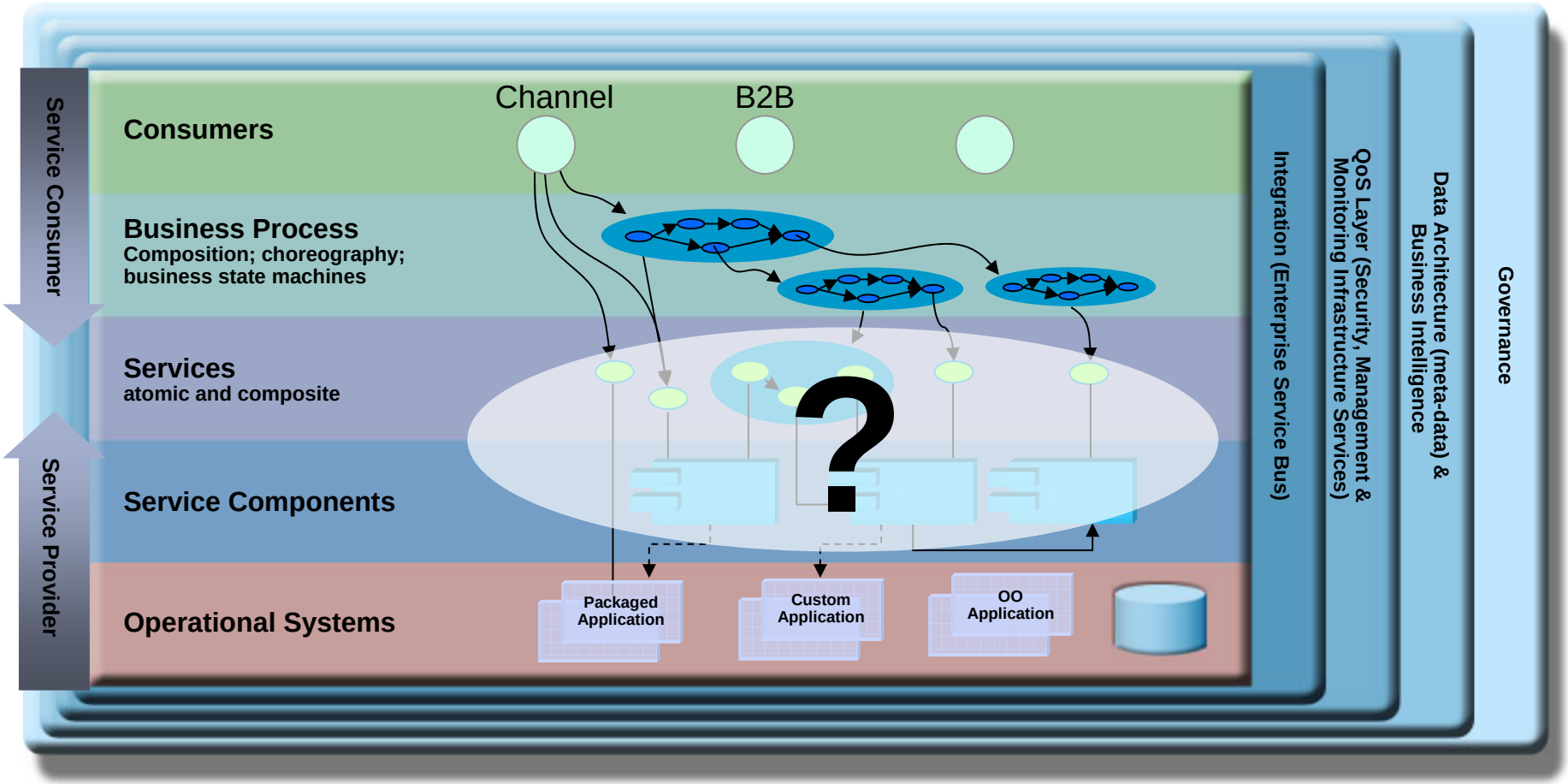
Coarse-Grained Mapping of Candidate Services to Existing Applications

- Examine assets (existing custom, packaged applications, and industry models, etc.) to determine what can be leveraged to realize service functionality
- Understand the business functions supported by each application
- Record attributes of existing applications in terms of technologies used, architectural styles, and so on
- Identify applications that perform common services

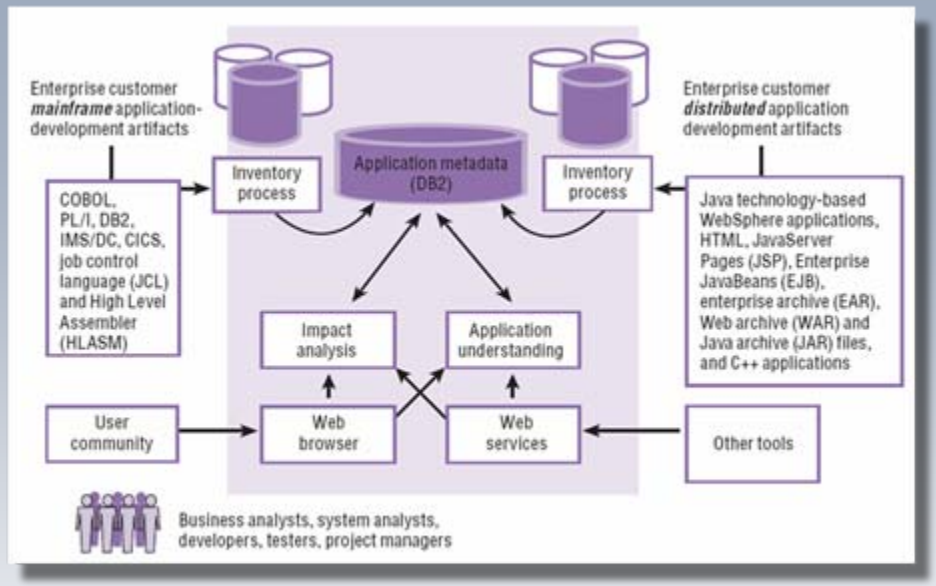


Identifying Services and Service Components

From Asset Analysis



Asset Analysis and Transformation Methods

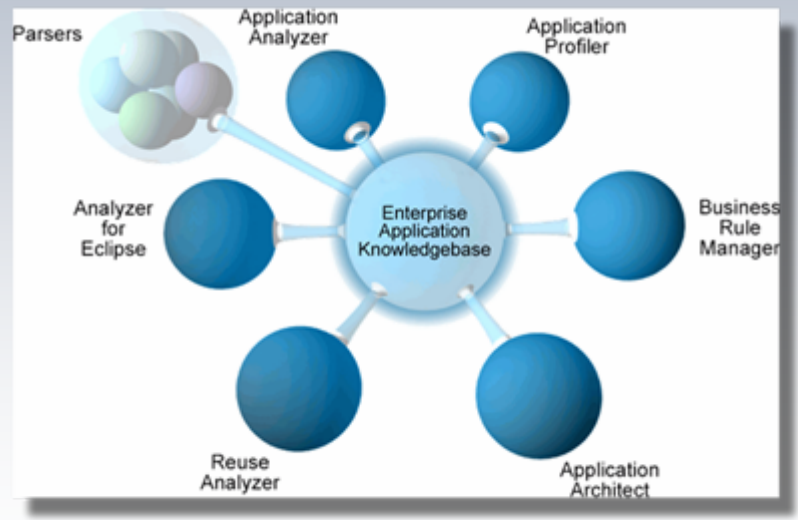


Asset Analysis

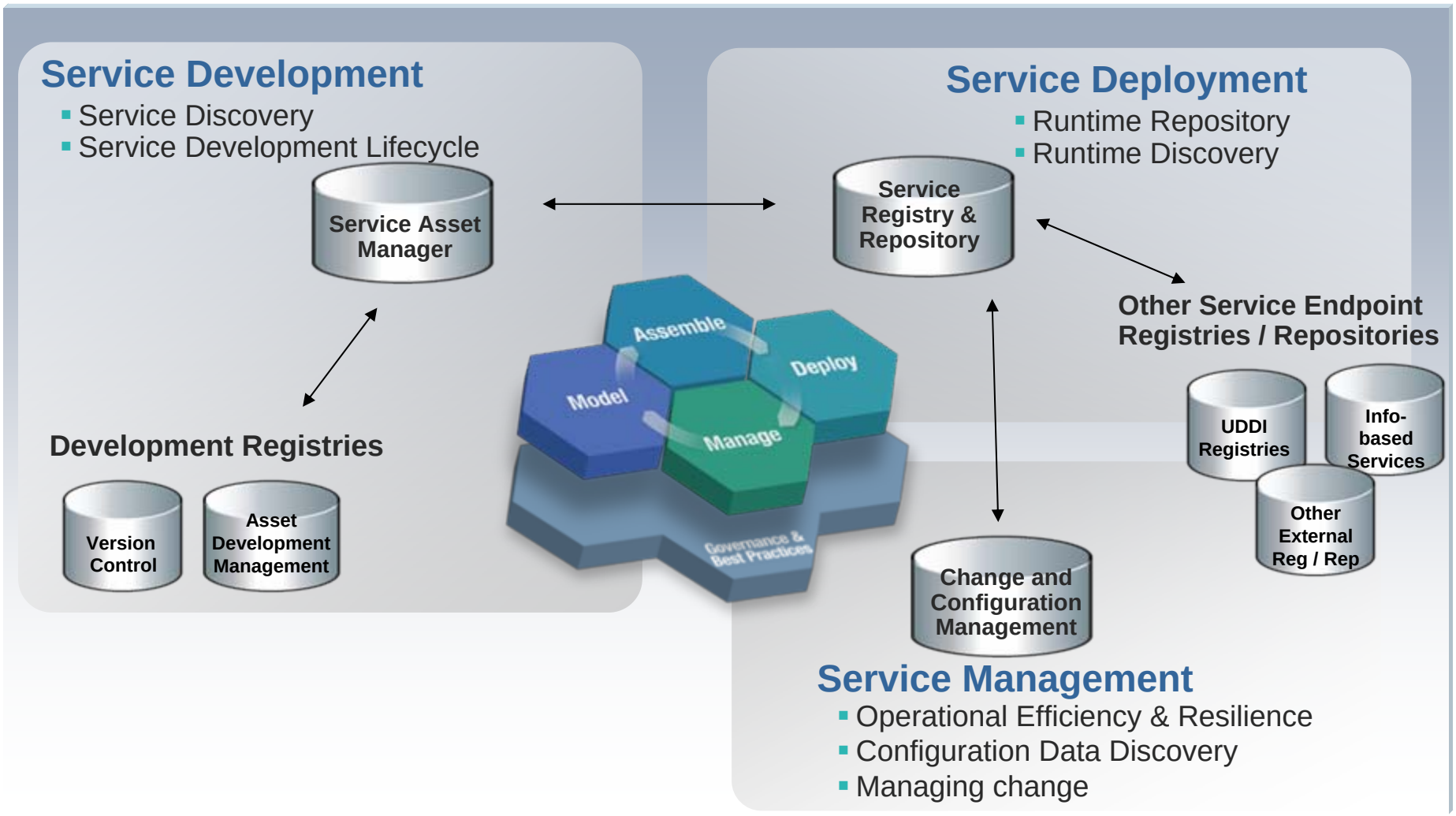
- Deep interactive analysis
- Find and manage business rules
- Create components from existing code
- Analyze code for SOA reuse

Asset Transformation

- Application discovery
- Application analysis
- Discover relationships between applications



Supporting Services Through The Lifecycle



Service-enable Mainframe Assets

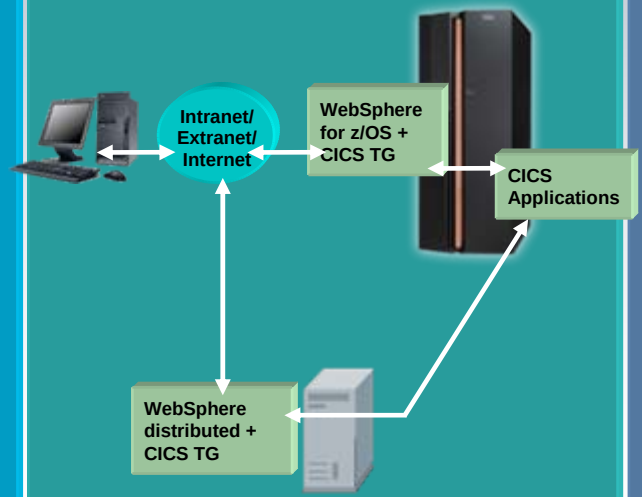
Make Better Use of CICS & IMS Investments

Enterprise Service Bus

- Messaging provides loosely coupled connectivity with assured delivery and reliability
- Advanced ESB Solutions can convert from any format (including SOAP) to COMMAREA format
- No changes required to existing application

Adapters

- J2EE to Mainframe adapters provide tightly coupled connectivity with two-phase commit support



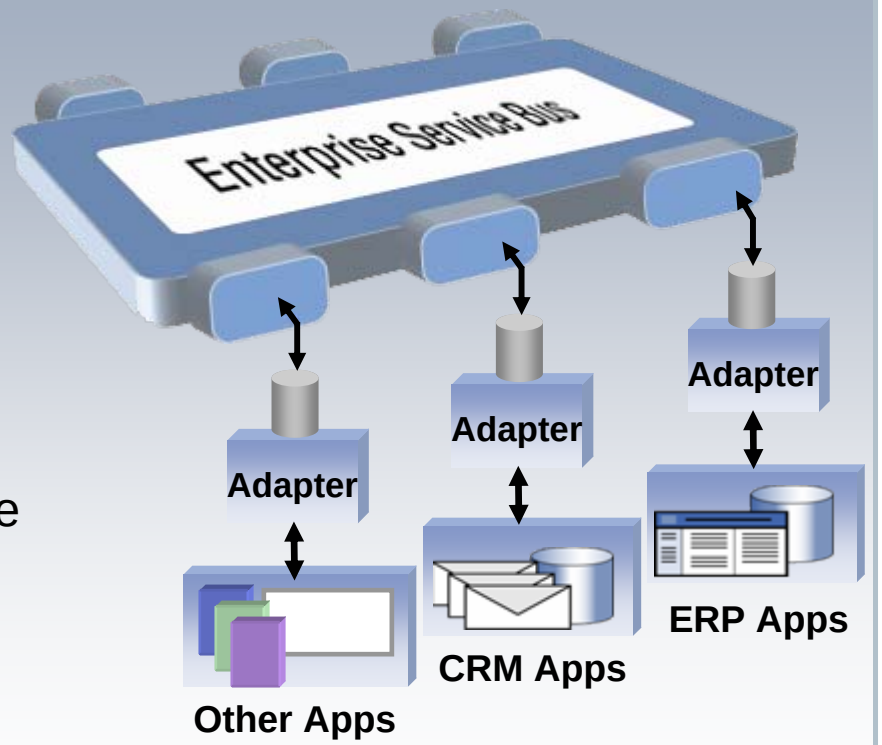
Native Web Services

- CICS & IMS can both expose transactions as native Web Services
- No other runtimes required

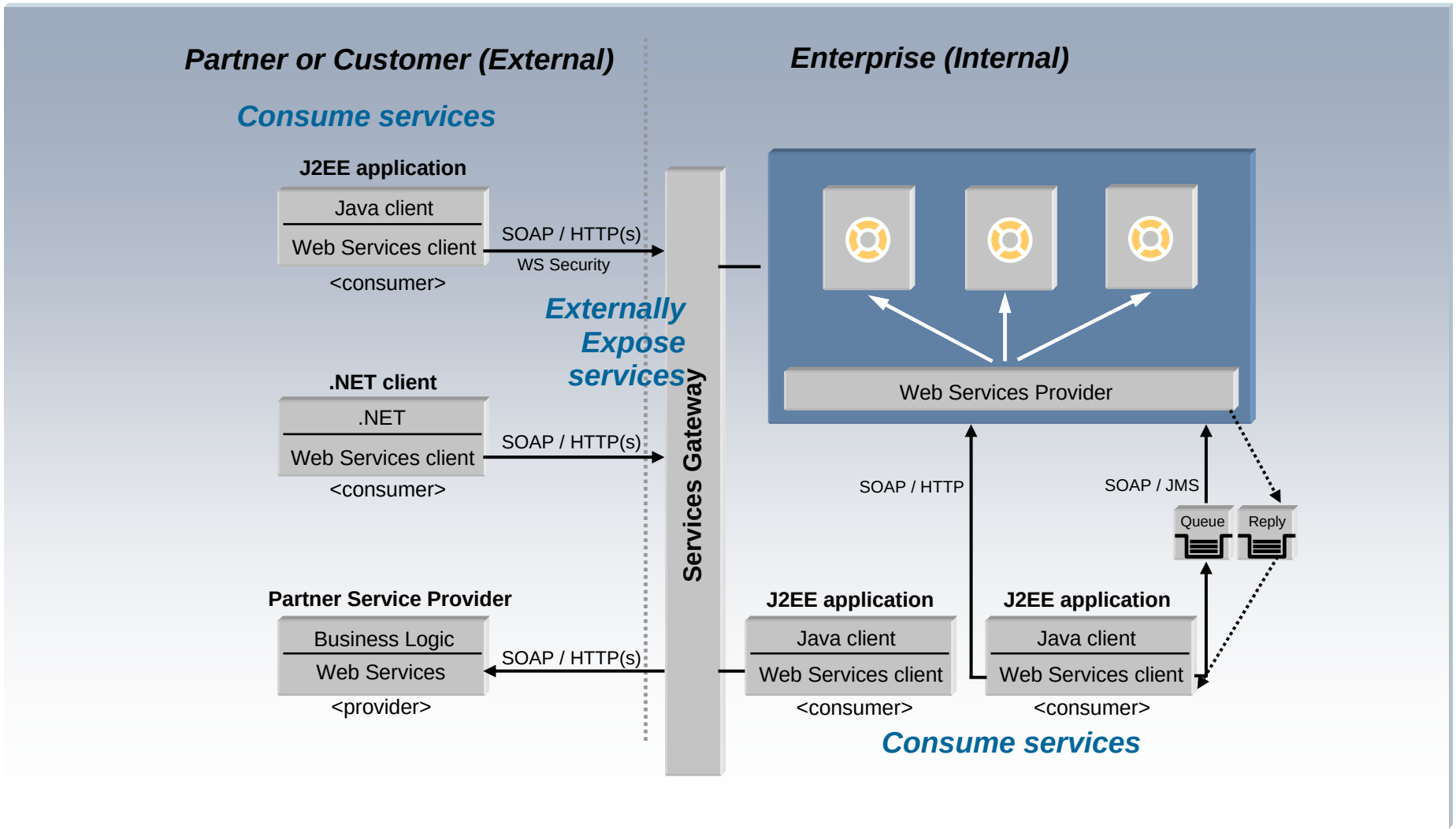
Service-enable Packaged Applications

Make Better Use of Commercial Software and Other Technology Investments

- **Service enablement** – provide a standard interface to proprietary implementations
- **Service discovery** – browse, select, and generate service descriptions from the application repository
- **Event capture** – detect and publish application events and control in-bound / out-bound information flows
- **Pre-built or build your own** – vast range of 'ready-to-go' adapters and toolkit to generate your own
- **Enterprise Quality of Service** – ensure mission-critical quality of service



Service Enable “Component” Applications



SOA and Application Architecture

- SOA Application Architecture Considerations
- SOA Application Architecture Best Practices
  - Asset Discovery Approaches
  - **Reuse Patterns**
  - Service Connectivity Scenarios
- IBM Capabilities to Support SOA Application Architecture
- Summary

Direct Access Pattern

Benefits:

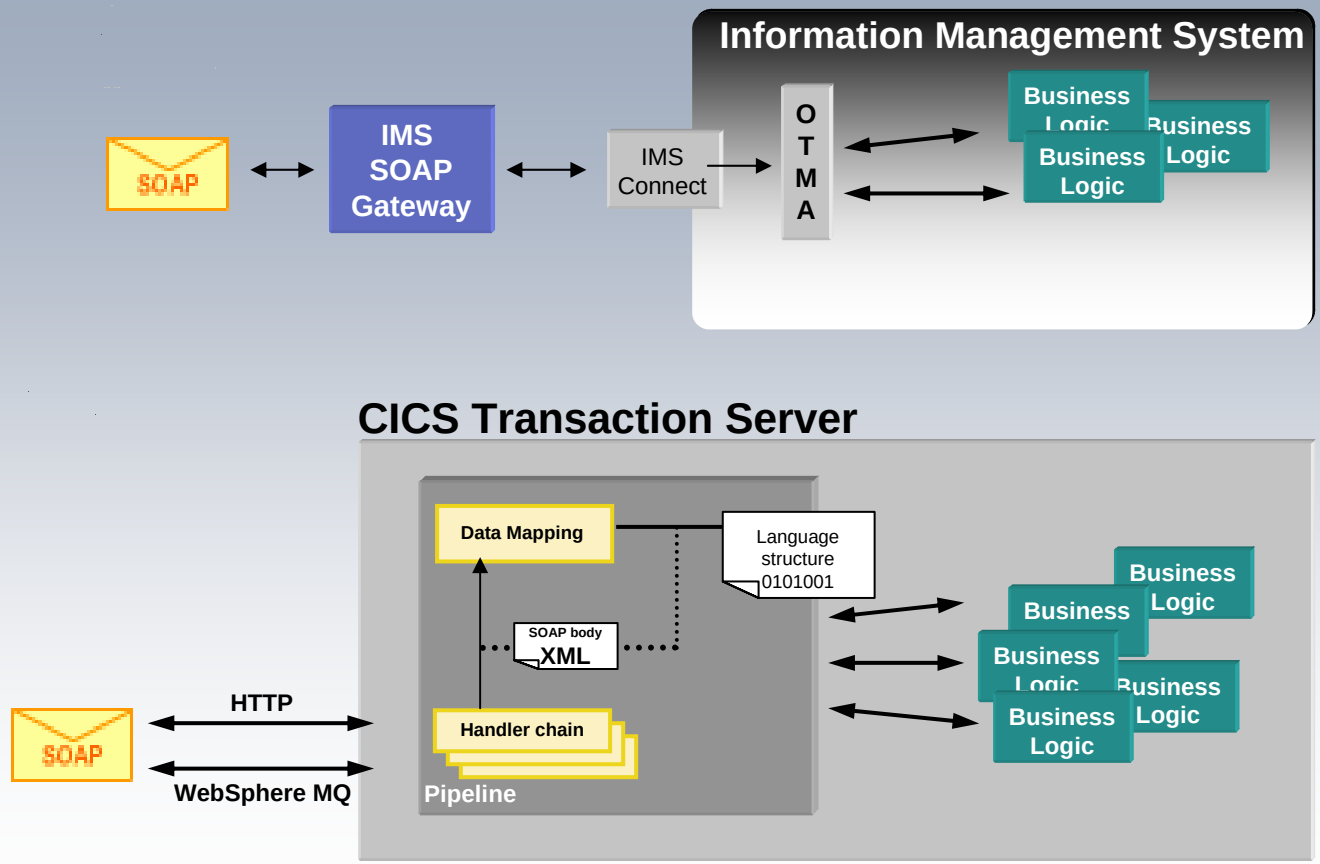
- Shorter deployment cycle … compared to indirect access
- The service interface is defined by the asset
  - No analysis required to determine the interface
- No knowledge of other runtimes (Java, Message Broker, etc.) is necessary
- Fewer platforms/moving parts

Issues:

- Consumers become coupled to the asset environment
  - Difficult to substitute the asset for an alternate
- Requires the asset runtime environment have support for service invocation
- Asset capability needs to match the service requirements
- Places an XML processing burden on the asset runtime
  - Systems that are often paid for on a “MIPS consumed” model

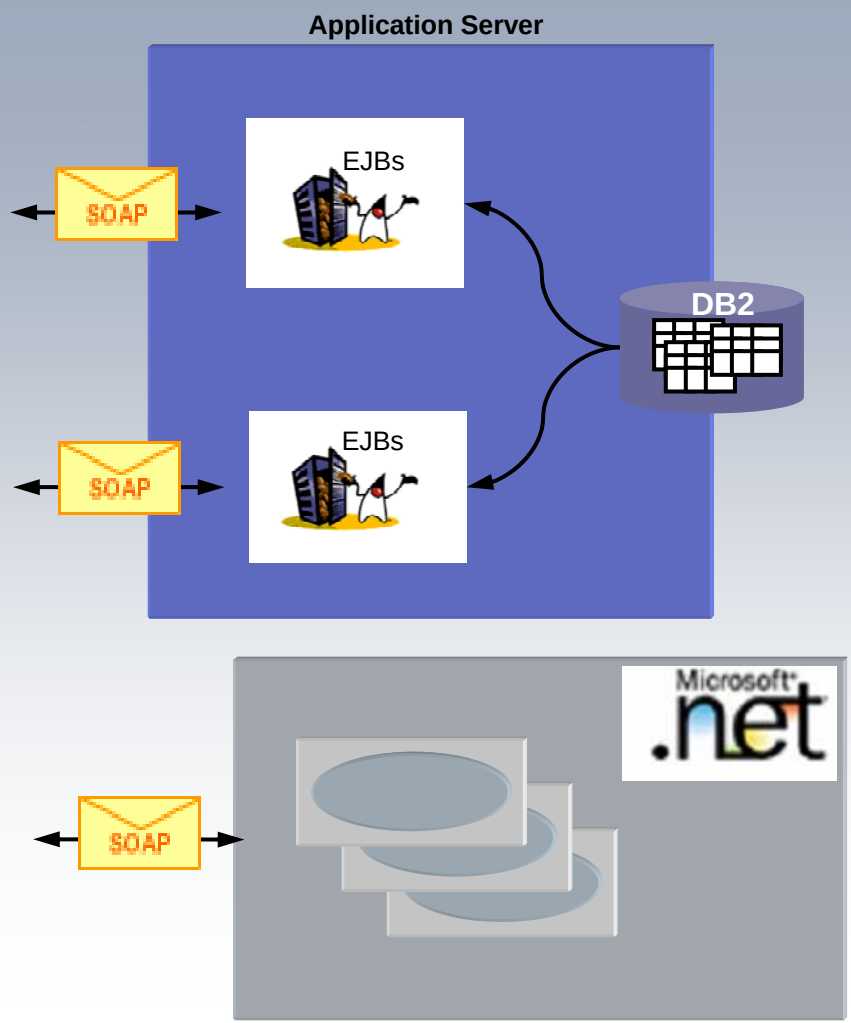
Direct Access Pattern Example

CICS and IMS Native Web Services



Direct Access Pattern Example

Java and .Net Applications



Indirect Access Pattern

Benefits:

- Business alignment is maintained
  - Service interface that suits/aligns with the business view and not with existing legacy assets
  - Service component maps between the two worlds
- Straightforward to substitute the asset
  - Service component may be replaced without impact on the consumer
- Offloads the XML processing burden
  - Many systems account for resources using a “MIPS consumed” model
- A service may be implemented using behavior from more than one asset
  - Service component aggregates the behavior to realize the service
  - Enables additional capability to be added

Issues:

- Longer deployment cycle than Direct Access
  - Consideration must be given to the definition of the service interface
  - Time spent developing the service component
- More complex than Direct Access
  - Generally involves the use of connector/adapter technology between the service component and the backend systems
  - Usually introduces a middle tier

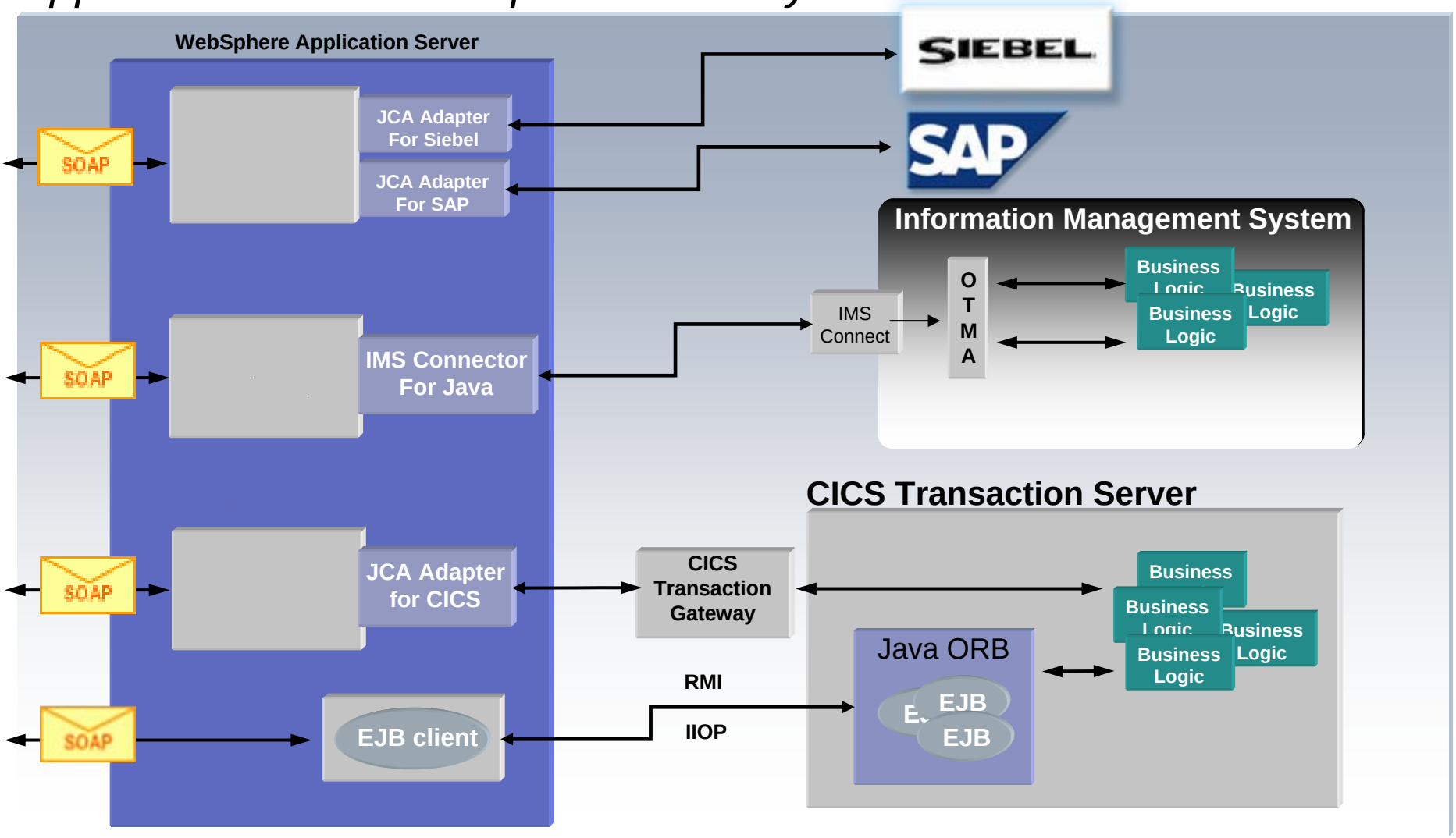
Indirect Pattern…

A service component serves as a front-end providing indirect access to the target asset

- Sub Pattern: Adapter, Gateway
  - Proxy based access to asset
  - Generally used to map standards based interface to asset based interface
- Sub Pattern: Application Server
  - Generalized capability for interacting with multiple target assets
  - Provides an environment for augmenting an asset's capabilities
- Sub Pattern: Enterprise Service Bus
  - Generalized capability for interacting with multiple target assets
  - Provides an environment for augmenting an asset's capabilities
- Sub Pattern: Terminal "Emulation" Environments
  - Encapsulating a sequence of screen interactions as a "macro"
  - Exposing a "macro" as a SOAP-based service

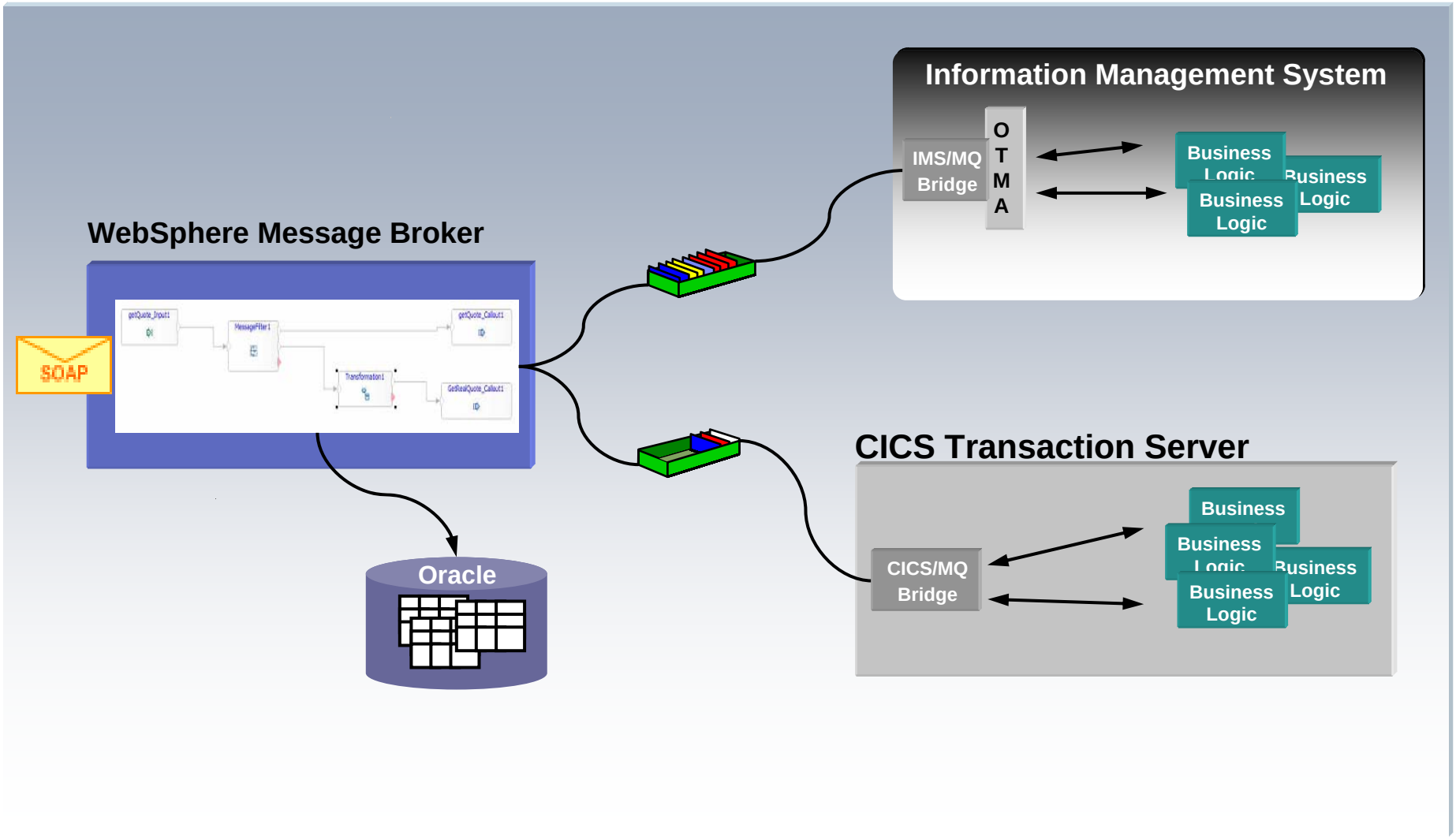
Indirect Pattern Example

Application Server / Adapter / Gateway

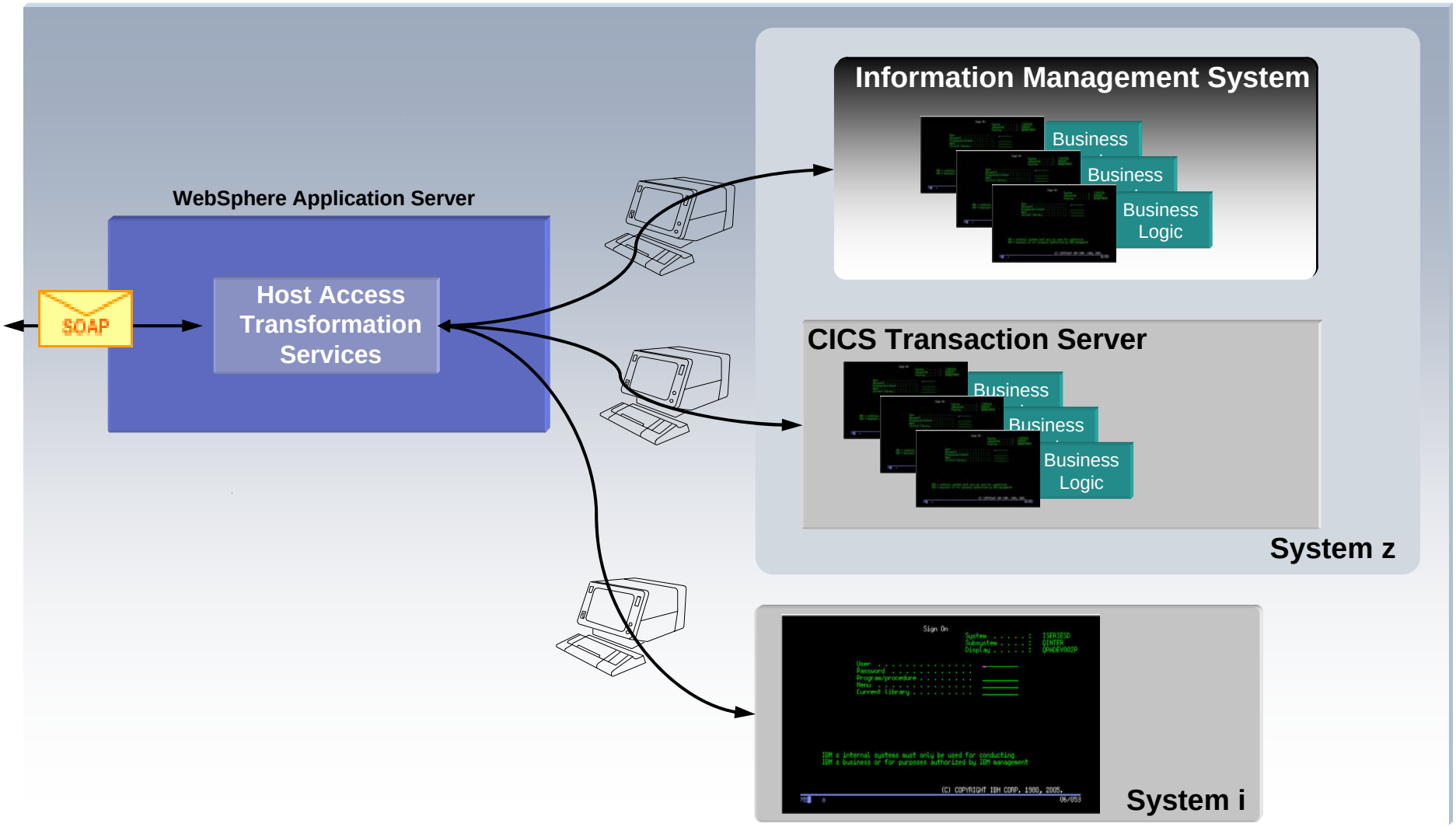


Indirect Pattern Example

Enterprise Service Bus

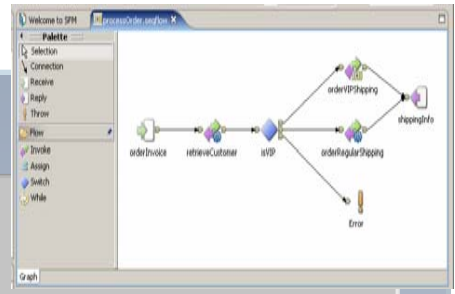


Indirect Pattern Example
Terminal Emulation



Combinations of Direct and Indirect Patterns

CICS Service Flow Modeler



Rational Developer for System z

CICS Transaction Server

SOAP

CICS Transaction Gateway

Service Flow

Output

CICS Commarea Application

CICS Commarea Application

CICS Commarea Application

CICS Terminal Interaction

CICS Terminal Interaction

CICS Terminal Interaction

CICS Application

Patterns Selection Guide

Comparison of Indirect vs Direct Access

Decision Criteria	Indirect	Direct
Implementing an existing or business driven service definition	■	
Many requestors or requestors outside of providers domain	■	
Aggregation or business logic applied across multiple existing functions	■	
Need to enable service provider/implementation replacement	■	
Return subset of information available in the existing function	■	
Cost of MIPS on existing platform is key concern	■	
Several assets to be aggregated together	■	
Skills only available on existing platform		■
Existing platform is strategic platform		■
Expediency is key driver		■

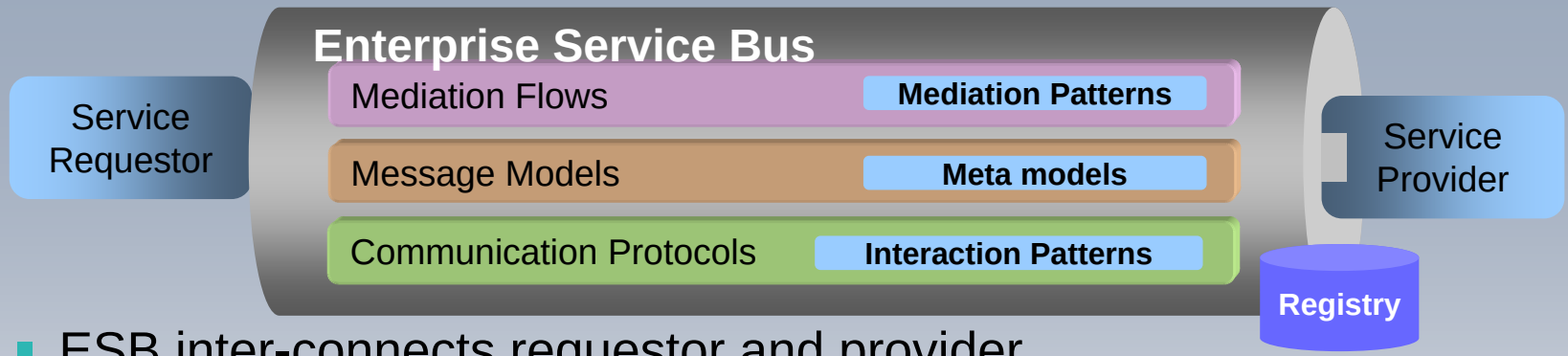
Reuse Pattern Architectural Decisions

- Common deployment decisions
  - If indirect pattern, location of proxy component relative to consumer and provider
  - Placement of service boundary
- Transaction management
  - Handling rollback and compensation
- Identity management
  - Mapping to the security requirements of the asset
- End to end management
  - Providing visibility across composite application
- Likelihood of changes to the service interface
- Chargeback for use of an asset

SOA and Application Architecture

- SOA Application Architecture Considerations
- SOA Application Architecture Best Practices
  - Asset Discovery Approaches
  - Reuse Patterns
  - **Service Connectivity Scenarios**
- IBM Capabilities to Support SOA Application Architecture
- Summary

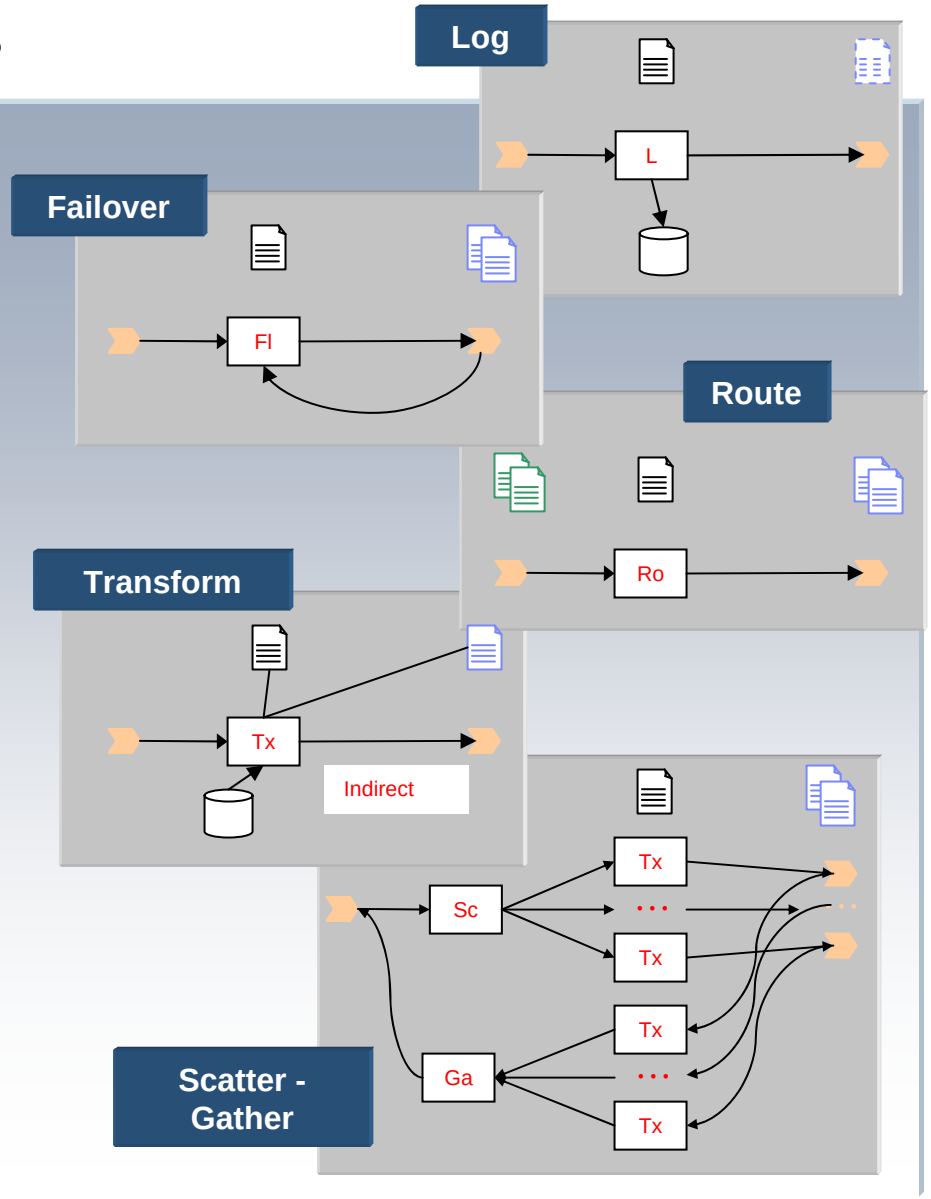
Core Principles of the ESB Architectural Pattern



- ESB inter-connects requestor and provider
  - Handle multiple communication protocols supporting interaction patterns
  - Flexibility to support message content models based on meta models
  - Enable interactions through defined mediation flows to process request messages and correlated results using defined patterns
- ESB provides ***Service Virtualization*** of:
  - *Identity* via routing
  - *Protocol* via conversion
  - *Interface* via transformation
- ESB also enables ***Aspect Oriented Connectivity***
  - To handle security, management, logging, auditing, etc.

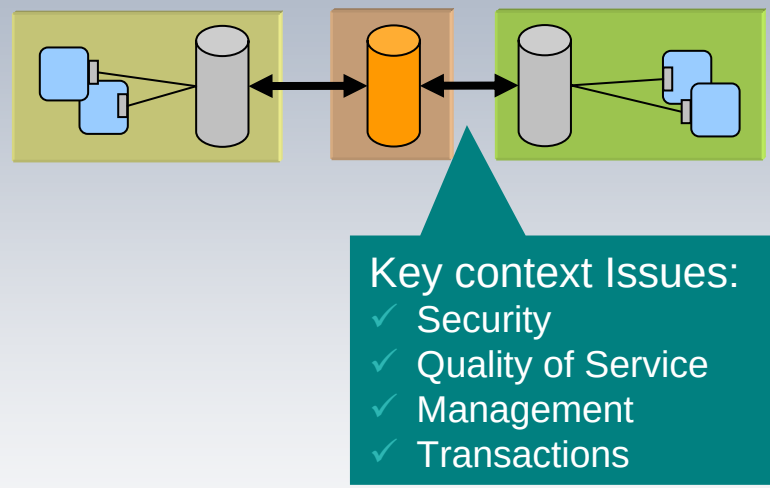
ESB Mediation Base Patterns

- Aspect oriented connectivity
  - Monitoring, logging and access to the services registry
  - Security and management policy definition points
- Service virtualization
  - Route messages dynamically based on defined service metadata derived from registry
  - Handle differing QoS for services
- Compositions
  - Complex event processing, e.g. handling failover
  - Logging the event and processing results



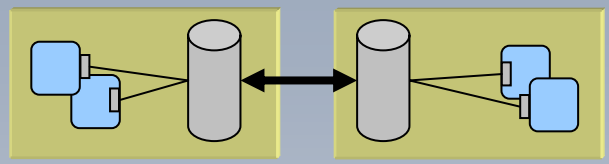
ESB Gateway Pattern

- Variant of routing or protocol switch pattern which maps service endpoints, possibly providing security functions (authorization and access control) and logging or auditing capabilities
- May incorporate transform and monitor mediations to provide encryption and logging, or auditing. It may also aggregate and disaggregate messages in a one-to-many relationship
- Example: Service portals which act as a single point of contact for multiple services and hide the details of “internal” services



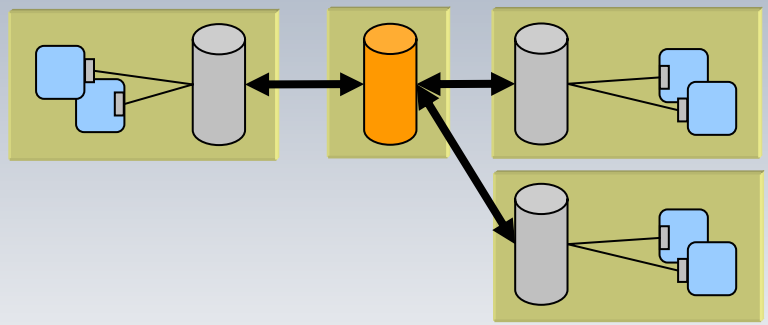
ESB Integration Topology Patterns

Direct



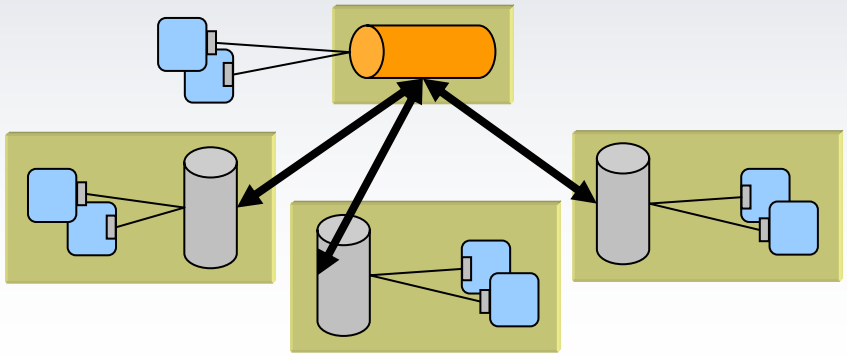
- Multiple namespaces and administration domains
- Namespace mapping in each ESB
- Services are likely to be applicable throughout the enterprise

Brokered



- Multiple namespaces and administration domains
- Namespace mapping in gateway facilitates service interaction
- Subset of services applicable throughout the enterprise

Federated



- Multiple namespaces and administration domains
- Namespace mapping in Federated ESB facilitates service interaction with multiple implementations
- Subset of services applicable throughout the enterprise

SOA and Application Architecture

- SOA Application Architecture Considerations
- SOA Application Architecture Best Practices
- **IBM Capabilities to Support SOA Application Architecture**
- **Summary**

Mapping to the IBM Products

Rational Developer for System z

WebSphere Development Studio Client for System i

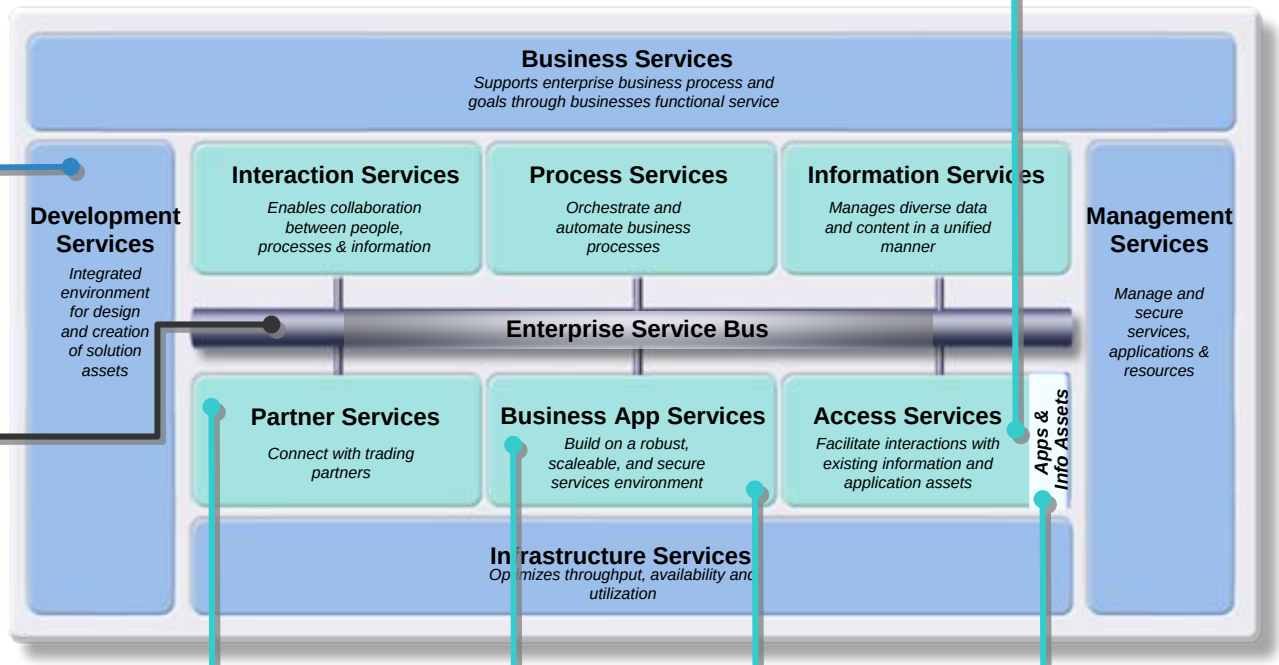
WebSphere Studio Asset Analyzer

Rational Transformation Workbench

WebSphere Adapters

CICS Transaction Gateway

WebSphere Host Access Transformation Services



WebSphere ESB

WebSphere Message Broker

WebSphere DataPower Integration Appliance

WebSphere Service Registry & Repository

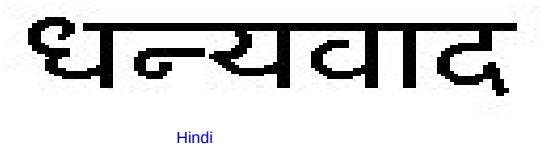
WebSphere Partner Gateway

CICS IMS

WebSphere Application Server

Summary

- There is significant value in reusing existing assets
  - Faster time to value
  - Cheaper to re-use than to re-write
  - Existing assets are tried and trusted
- Well defined approaches to discovering high-value assets for reuse
  - Analysis done as part of service design methodology (e.g. SOMA)
  - Existing asset analysis through tools
- Two primary architecture patterns for reusing existing applications
  - Indirect access to target asset through service component
  - Direct access to target asset through service interface
- Need capabilities to support connecting and using existing assets:
  - Enterprise Service Bus provides main capabilities to connect and use existing assets
  - Service registries and repositories to support service through lifecycle



Hindi

Traditional Chinese

Thai

Russian

Spanish

Arabic

Thank You

Slovenian

Italian

German

French

Simplified Chinese

Hungarian

Korean

Japanese

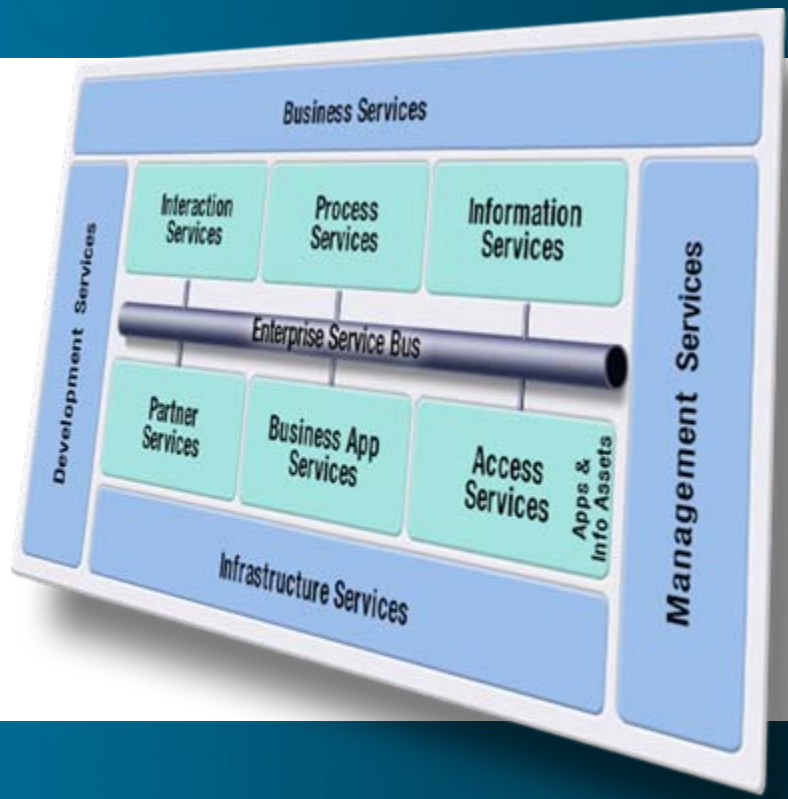
IBM SOA Architect Summit

SOA on your terms and our expertise

IBM SOA Architect Summit

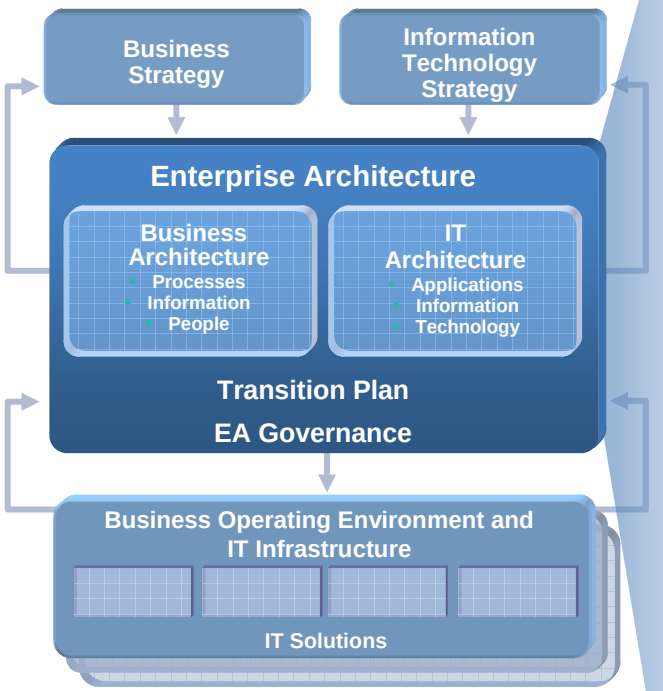
Infrastructure Architecture:

Architecting the Right SOA Infrastructure



Ignacio Izaga
IT Consultant, Global Technology Services

SOA Architect Summit Roadmap



What is the impact of SOA on current Enterprise Architectures?

- Alignment of Business and IT Architectures
- SOA Reference Models
- SOA Governance

How do you develop SOA with a business focus?

- Business Components
- SOA Design
- Business Process Management

How do you reuse applications in the context of SOA?

- Asset Discovery
- Application Reuse

How do you leverage information in an SOA?

- Information as a Service
- Master Data Management

How does my infrastructure support SOA?

- Service Management / QoS
- Security

Agenda

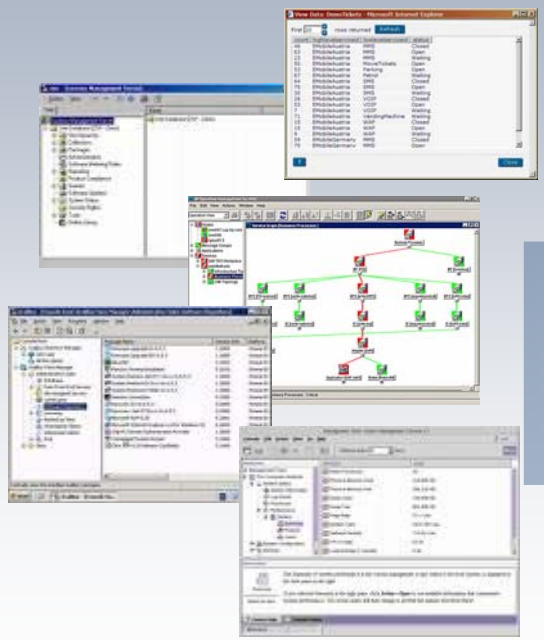
- SOA requiere una nueva forma de pensar en las Infraestructuras
- Consideraciones de Infraestructuras en SOA
  - Rendimiento
  - Disponibilidad
  - Gestión de Servicio
  - Seguridad
  - Virtualización
- Capacidades de IBM para dar soporte a Arquitecturas de Infraestructuras SOA
- Conclusiones

SOA representa un cambio notable en las prioridades de TI

Y requiere una nueva forma de pensar

Old Thinking

IT *maintains* IT **resources** that support the business



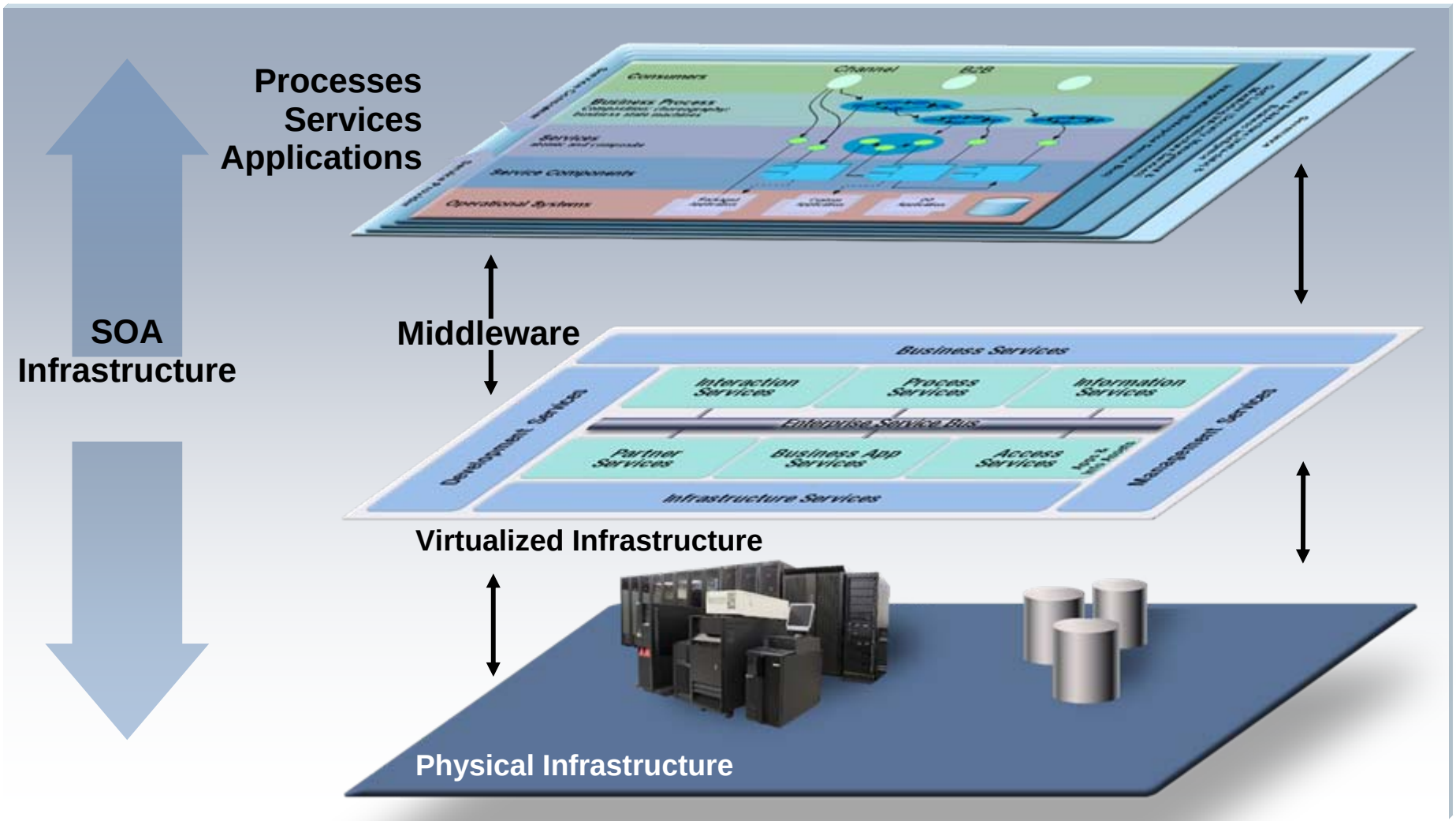
From Silos …

New Thinking

IT *delivers* **services** designed to meet business **goals**

… to Services

Niveles de Abstracción en SOA



¿Cómo afecta SOA a la Gestión e Infraestructuras de Sistemas?

Características SOA

- Applications reused in new dynamic ways
- Services combined from multiple sources
- Rapid deployment
- Services route to any available resource
- Distributed access

Consideraciones Fundamentales en Infraestructuras y su Gestión

Agenda

- SOA requiere una nueva forma de pensar en las Infraestructuras
- **Consideraciones de Infraestructuras en SOA**
  - Rendimiento
  - Disponibilidad
  - Gestión de Servicio
  - Seguridad
  - Virtualización
- Capacidades de IBM para dar soporte a Arquitecturas de Infraestructuras SOA
- Conclusiones

SOA Introduce Retos en el Rendimiento

- Measuring performance across organizational boundaries can be more difficult than in siloed applications
- Response time estimation is more challenging in a more distributed environment
  - Performance costs can be difficult to predict
  - Performance testing an SOA application requires the use of new techniques
- Increased requirement for XML processing may impact performance

El Rendimiento no es una consideración a posteriori

Debería estar considerado desde el diseño de la solución

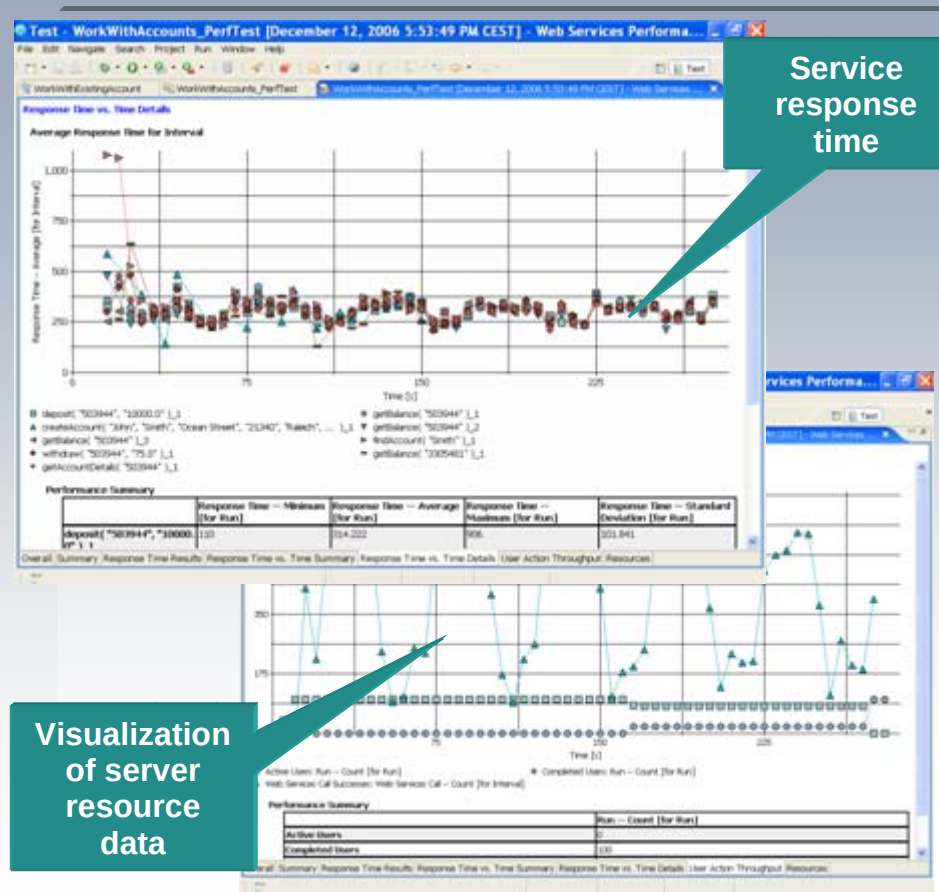
- Performance in SOA systems should be a combination of performance engineering and performance management
- SOA-based applications can change the way an infrastructure performs
  - XML message transformation, location, message size, frequency
  - More complex applications and transactions
- Each of the components should be used to build a performance budget, transaction models and use cases
- Middleware and server sizing need to be done with the application teams
  - How many, how available, virtualized, system platform
- Don’t forget about security overhead
  - Authentication, Authorization, Encryption

Guía de Pruebas de Rendimiento SOA

- Test SOA applications throughout the development lifecycle
- Test tools must be separate and external to the SOA environment
- Use multiple diverse datasets that are representative of an SOA workload
- Stress test the solution to detect latency issues
- Run tests in a comparable environment to the deployment environment
- Use multiple test tools – similar results from multiple test tools using identical data sets validates the tests

Pruebas de Rendimiento SOA y Herramientas de Determinación de Problemas

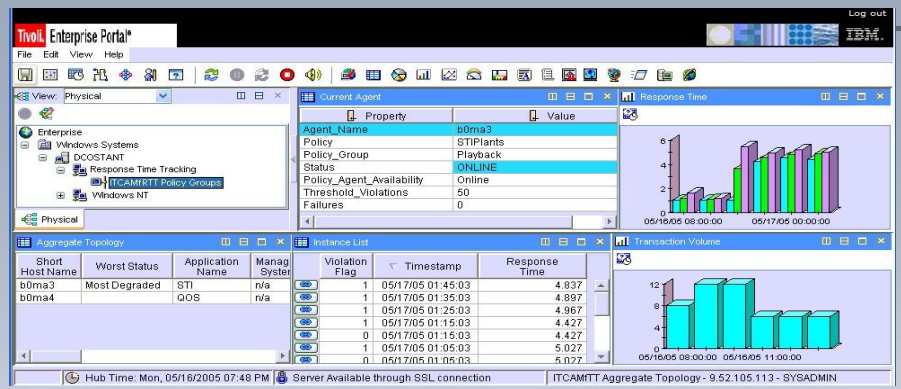
- Validate system scalability
  - Workload modeling for automated generation of test clients
  - Automated generation of performance tests
  - Real-time reporting of server response time and throughput
- Isolate performance bottlenecks and resolve problems
  - Monitoring support for services across multiple platforms
  - Collection and visualization of server resource data – root cause analysis

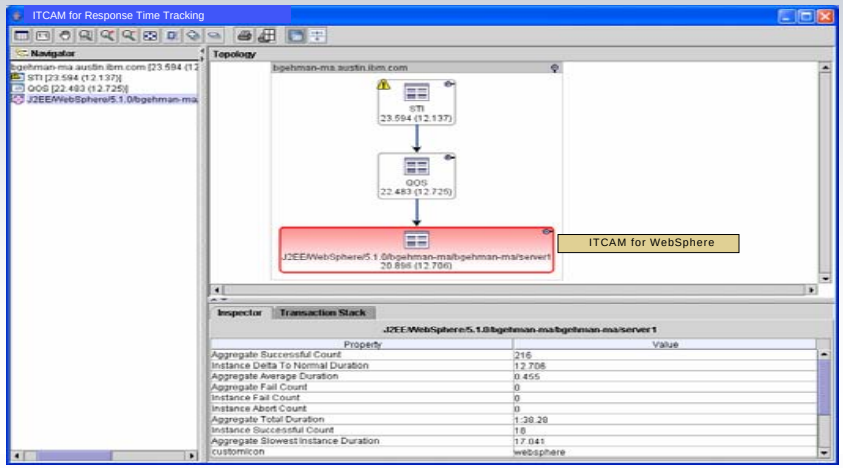


Monitorización de Transacciones en SOA

Medidas del tiempo de respuesta en entornos distribuídos

- Composite applications span technology and platform boundaries
- Can be difficult to identify and isolate performance bottlenecks
- Use lightweight instrumentation that can be dynamically configured to proactively identify performance problems
- Use industry-standard ARM-based instrumentation to isolate the problem

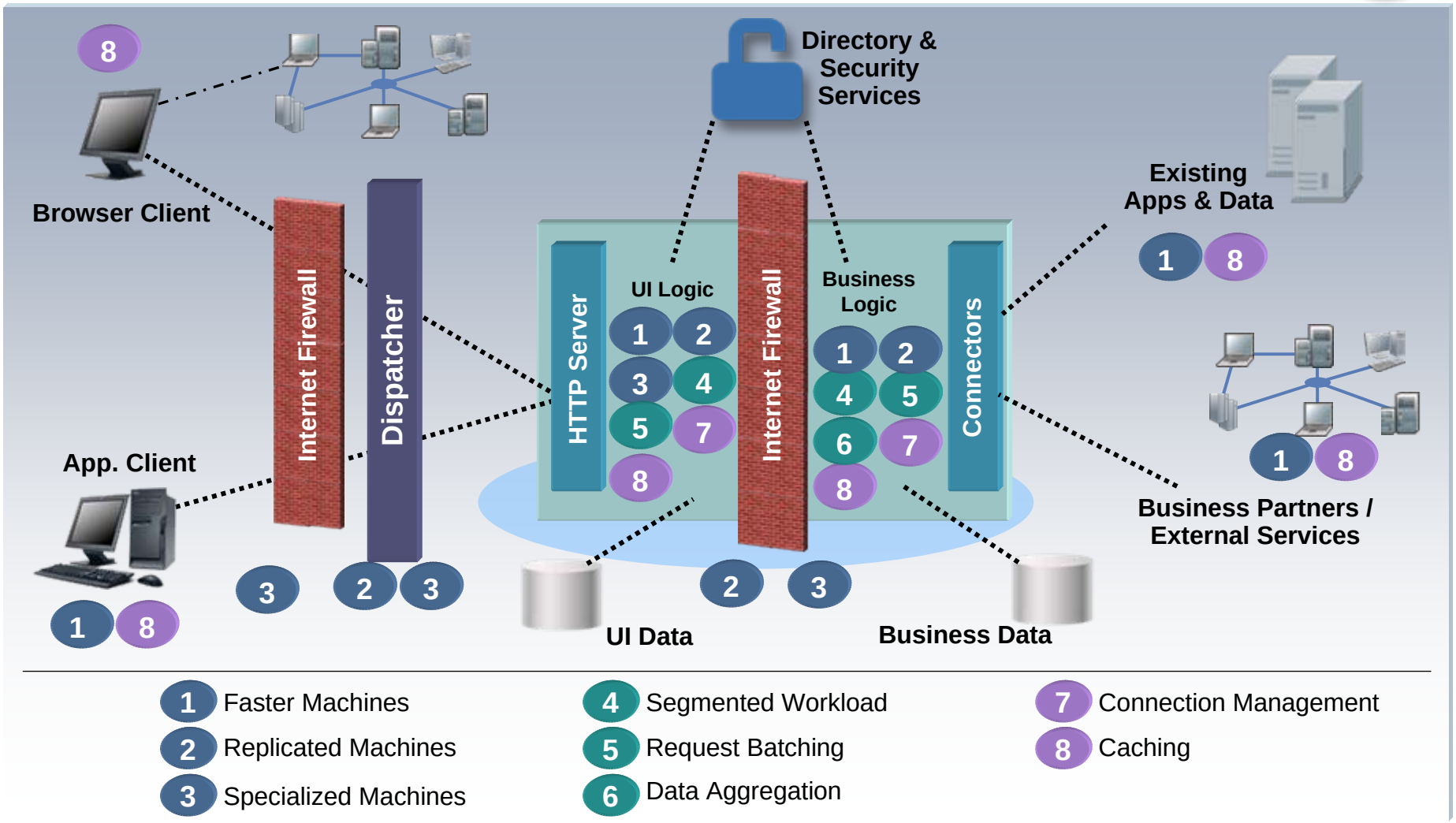




Mejores Prácticas para el Rendimiento en SOA

- The SOA performance model should be created and maintained throughout the lifecycle as the application is built
- Performance testing needs to obtain sufficient metrics to validate that services meet performance expectations
- Use established techniques to meet SOA performance requirements
- Design, test, and retest to confirm that non-functional requirements are met
- Implement an integrated solution that will automatically monitor, analyze and resolve response time problems
- Consider dedicated network appliances to optimize and accelerate XML parsing and security processing

Técnicas de Alta Disponibilidad y Escalabilidad

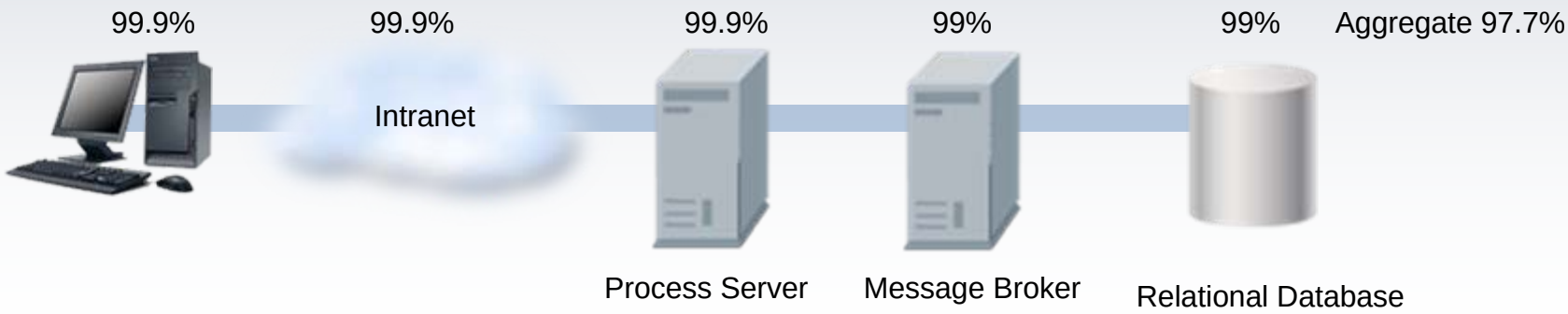


Alta Disponibilidad en el mundo SOA

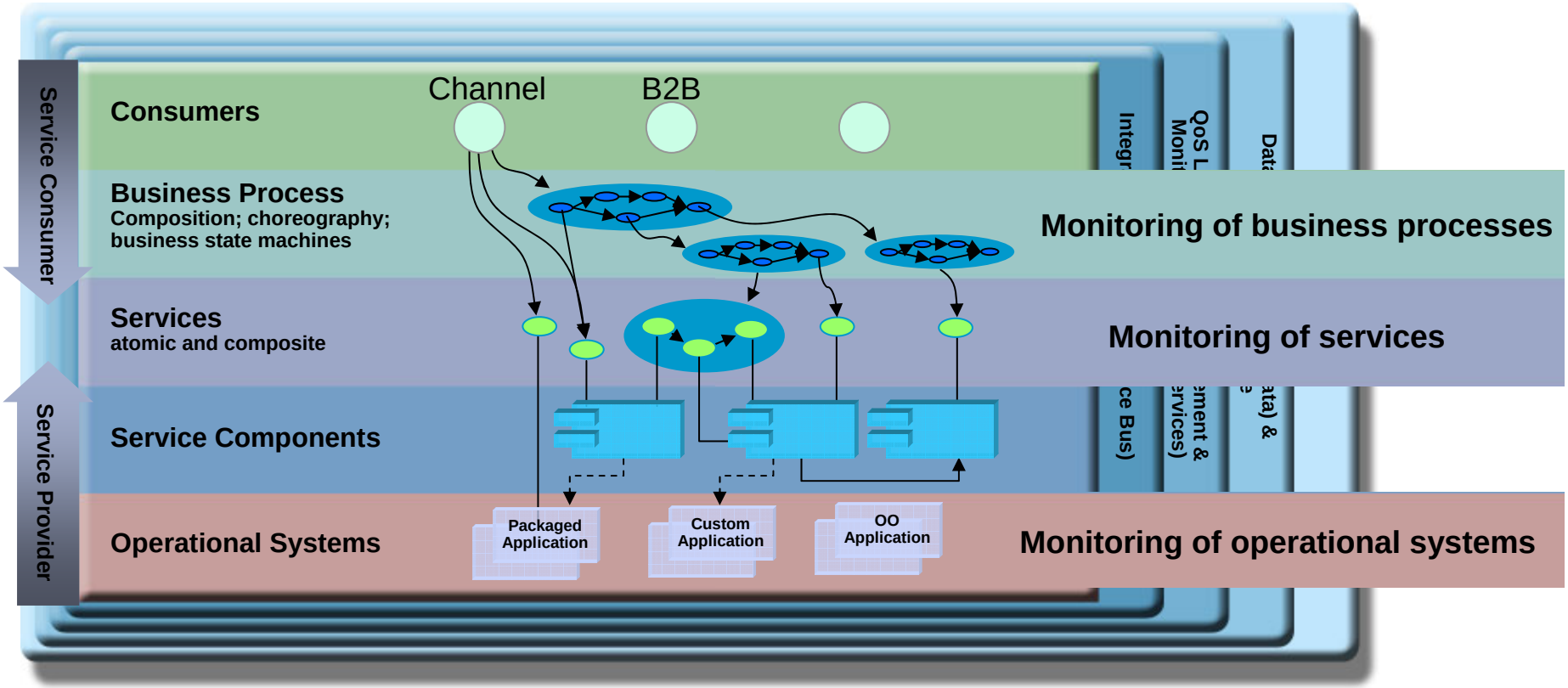
- An application may exist on multiple servers in different locations
  - Applications need to be “availability” aware in case a service within the workflow is unavailable
- SOA applications impact service availability levels
  - SOA introduce new application dependencies, including externally provided services
  - Need to understand the end-to-end view
- Monitoring, management and reporting is required to achieve predictable availability in an SOA environment
- Plan for the unexpected
  - What are the non-functional requirements? What systems are you using? Distributed? Mainframe? Where are they located? How will they be accessed?
  - The more components in the transaction, the greater the risks for failure or human error

Mejores Prácticas para Alta Disponibilidad en SOA

- There are an increased number of components in an SOA infrastructure, so test rigorously for availability
- Create failover plans based on criticality of applications and services
- Take advantage of established availability techniques
  - Each component requires its own availability architecture
  - Leverage capabilities like Workload Management, High-Availability Manager, Deployment Manager, etc.
- Some components may require both hardware and software clustering
  - Databases, enterprise messaging infrastructure, SOA appliances



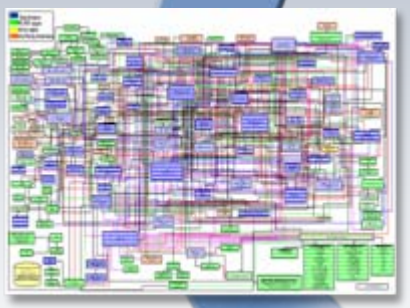
Retos de la Gestión bajo SOA



Gestión de Servicios en SOA

Requiere una visión cíclica

How does this relate to the business service?



What's happening with the infrastructure?

What actions do we take to correct the problems?

IBM Service Management

What's happening with the infrastructure?

- Infrastructure and application discovery
- Server monitoring
- Storage monitoring
- Network monitoring
- Data monitoring
- Application monitoring
- Service monitoring

How does this relate to the business service?

- Dashboard
- Application dependency mapping
- Business service management
- Service level management

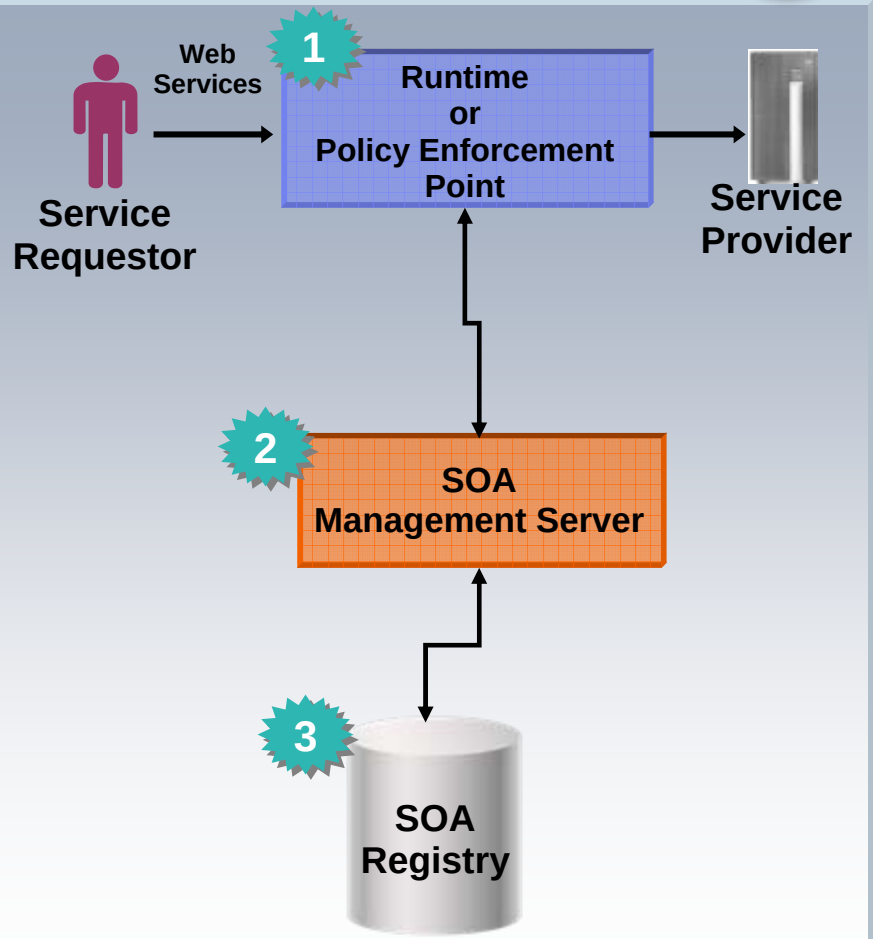
What actions do we take?

- System reconfiguration
- Data restore
- User identity provisioning
- System and application restart
- Infrastructure deployment
- Service mediation

Elementos Clave para la Gestión de Servicios

There are 3 key components in services management:

1. The runtime environment – this is where messages are routed, secured, transformed, filtered and logged
2. The management server – aggregates the data from all of the endpoints and runtimes and sends configuration changes based on policy
3. The registry – stores meta data about services and policies



Soporte Dinámico de los Servicios

Modificación dinámica de dependencias entre servicios y aplicaciones

Support Change Process

- Initial state
- Use to validate that planned changes were executed and that the results are as expected

Pre-Change Validation

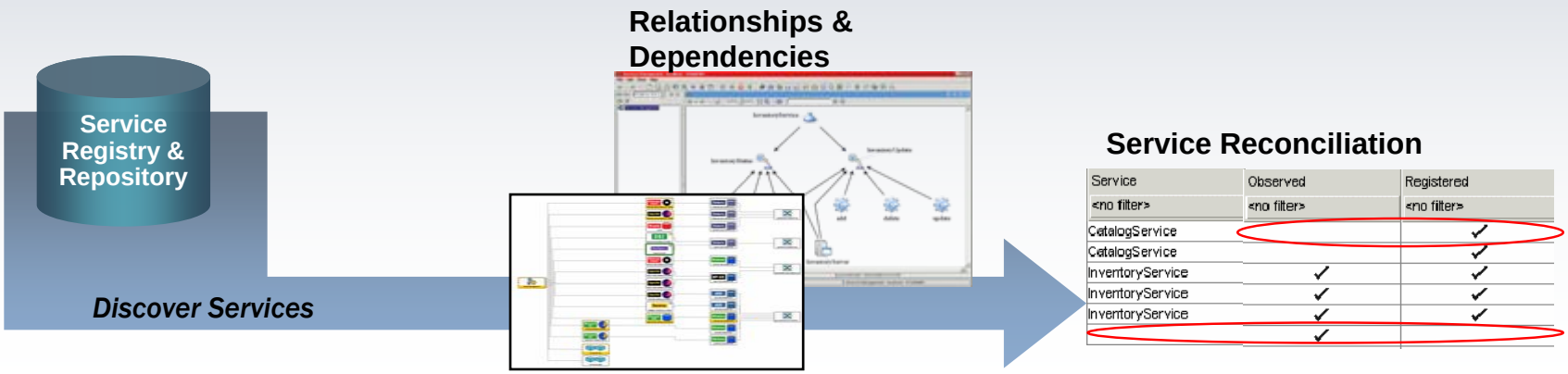
- What applications does a component support?
- Reduce unintended consequences

Improve MTTR (Mean Time to Resolution)

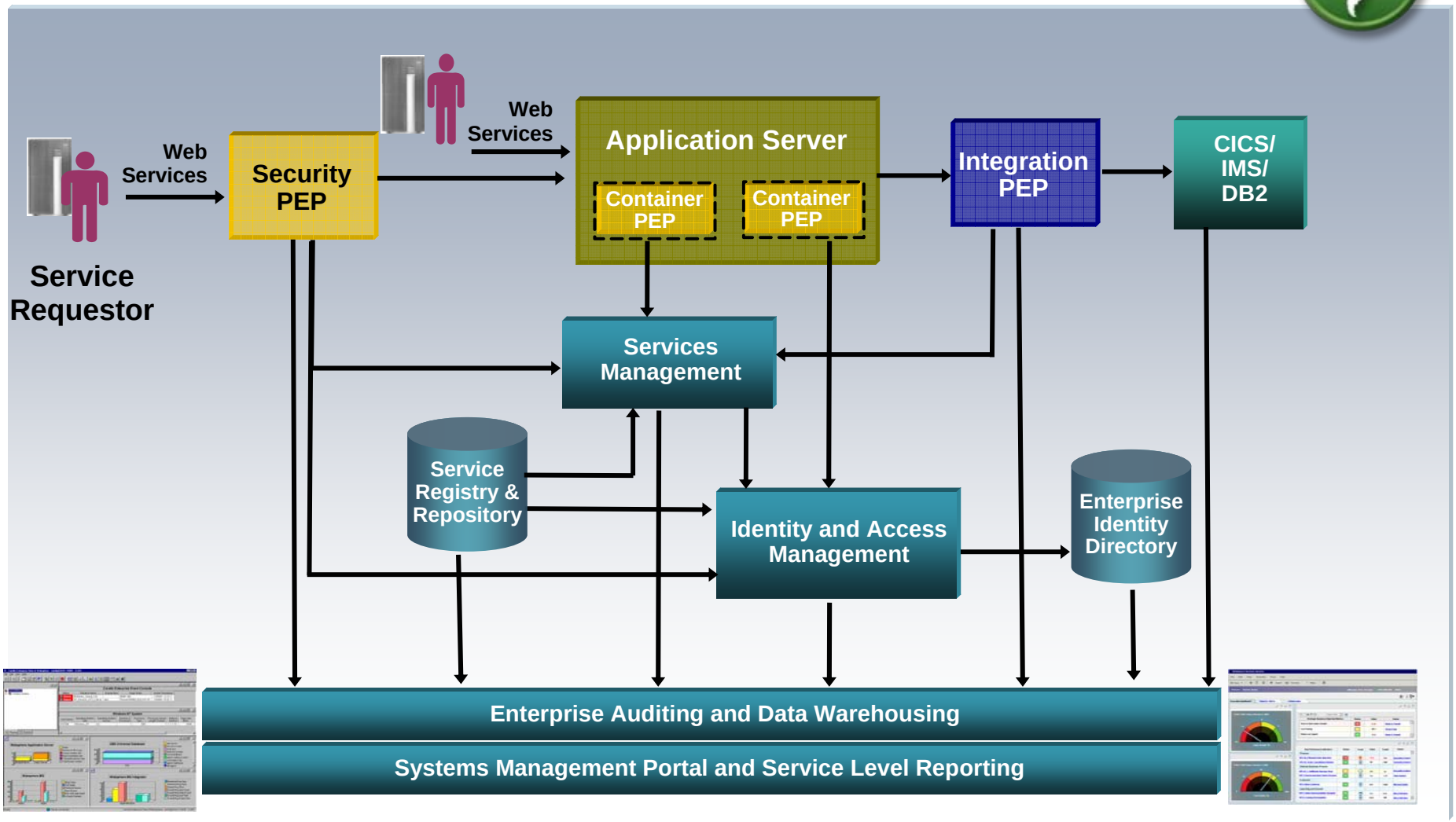
- Accurate application maps show you what is important
- Find the “Last Change” before a problem shows up
- Simplify impact analysis

Configuration Drift

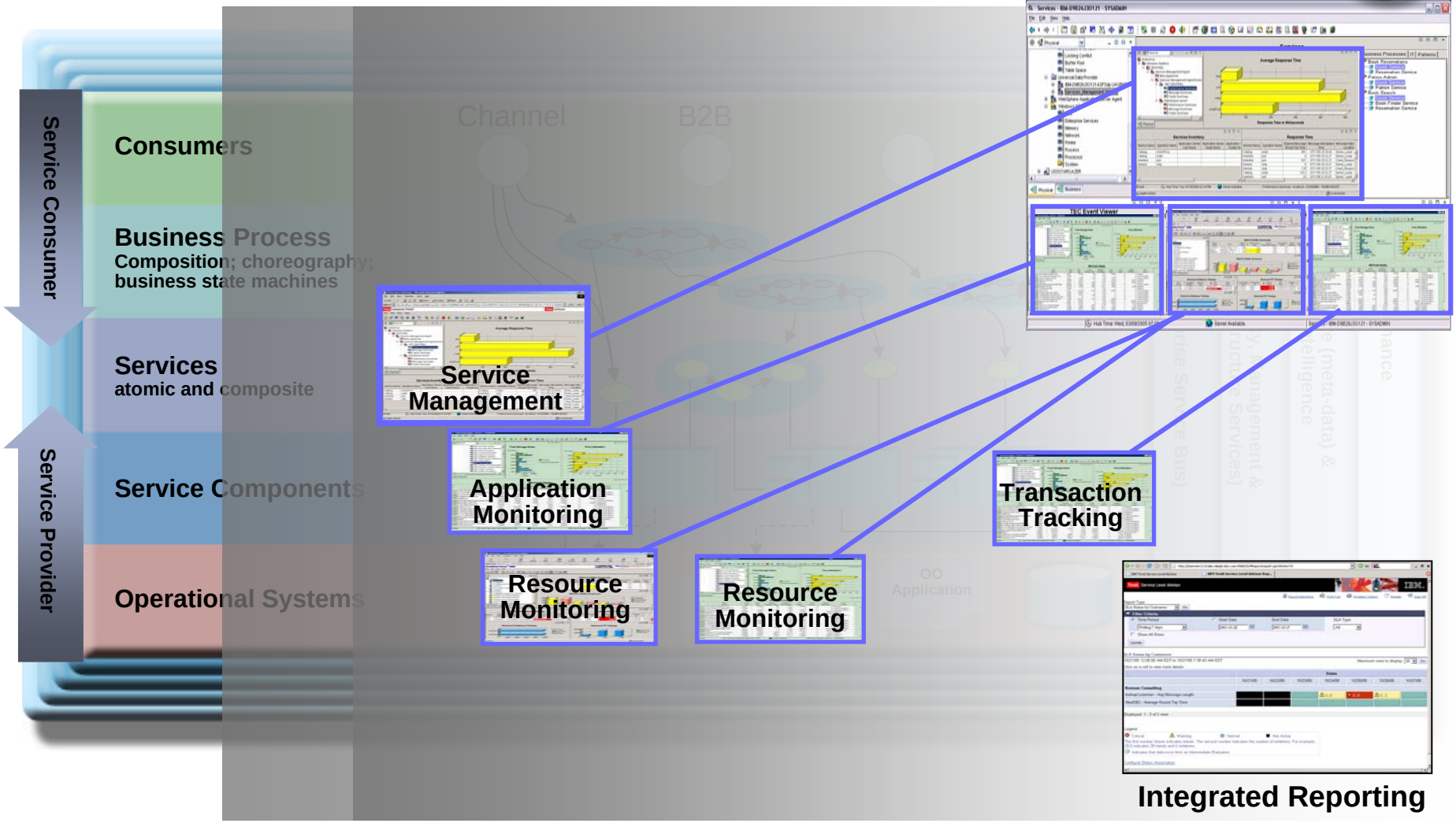
- Notice when Configurations change and notify operations
- Keep “bit-rot” from impacting operational readiness



Componentes de una Solución de Gestión en SOA



Visión Integrada de los Componentes de Gestión SOA



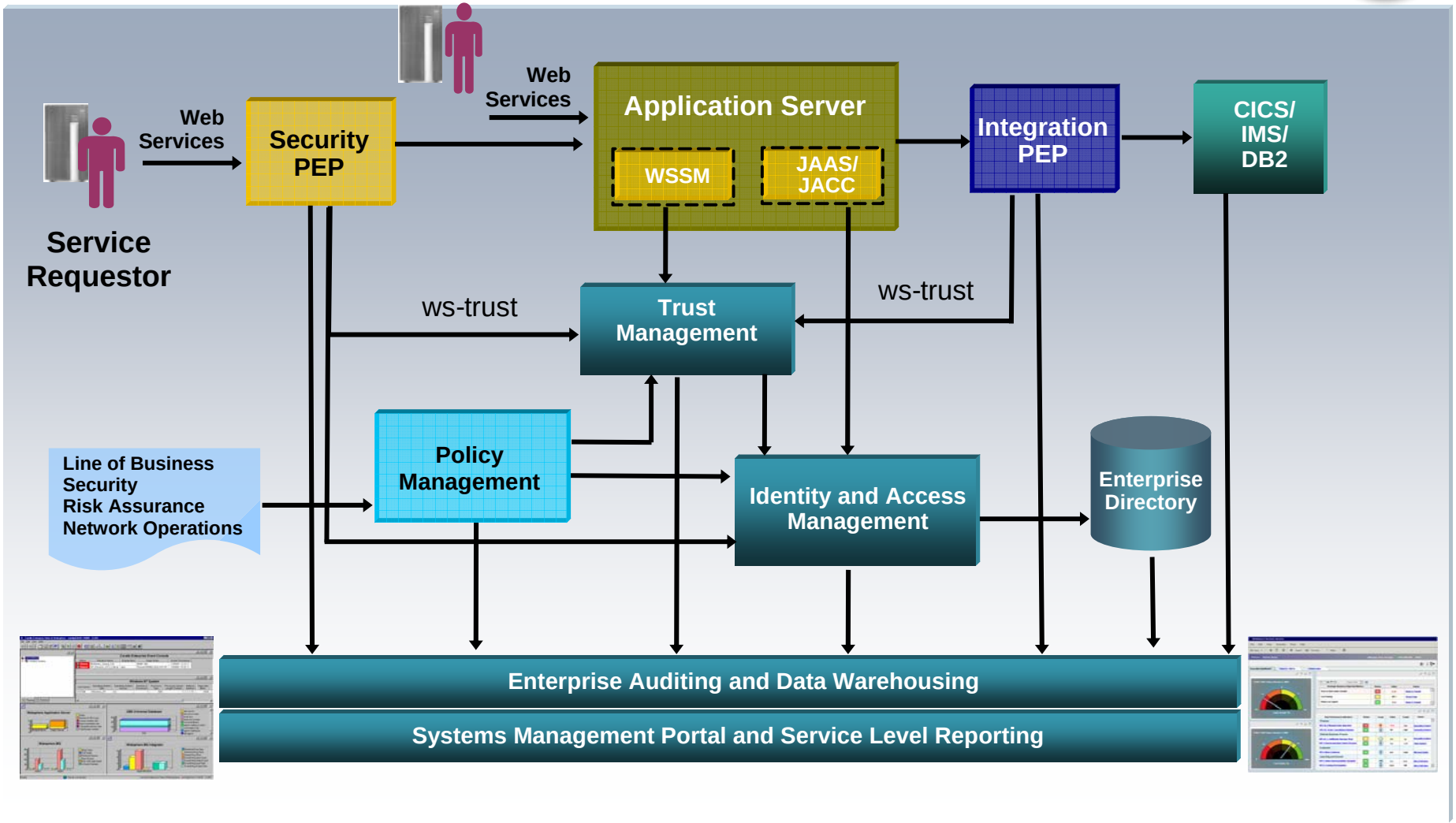
Mejores Prácticas de *Service Management*

- Establish operational and business-focused management and monitoring perspectives
- Monitor the end-to-end solution to isolate and fix problems
- Automate provisioning and control of services to meet SLAs
- Make use of tools to improve application availability
- Track/predict change to reduce costs and downtime

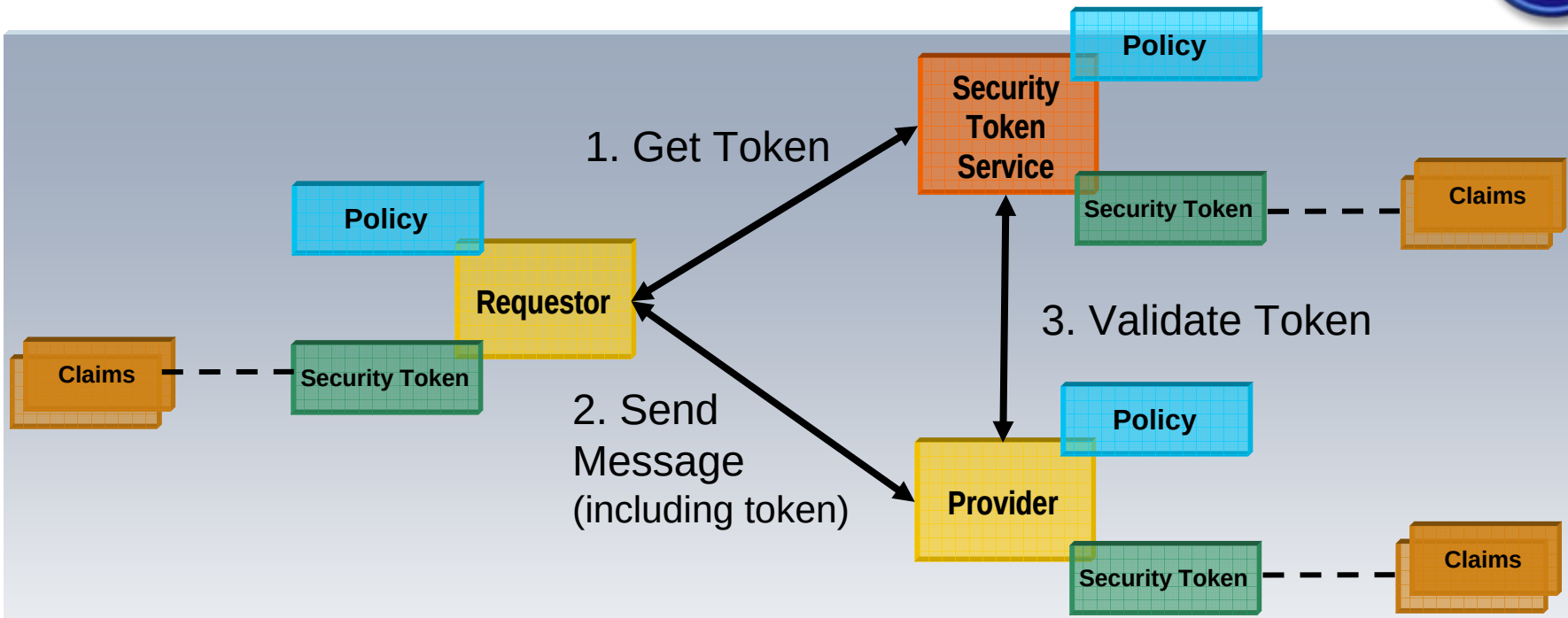
Consideraciones de Seguridad bajo SOA

- SOA introduces additional security issues
  - How do we identify and authenticate the service requester?
  - How to we identify and authenticate the source of the message?
  - Is the client authorized to send this message?
  - Can we ensure message integrity & confidentiality?
  - How do we audit the access to services?
  - How do we leverage Web services security standards?
  - How do we propagate identities with trusted service providers?
- XML Web services may expose backend systems in unintended ways
- SOA security may require multiple layers of enforcement – perimeter, gateway, app server, application
- Traditional security devices do not secure XML/SOAP

Componentes Lógicos de Seguridad en SOA



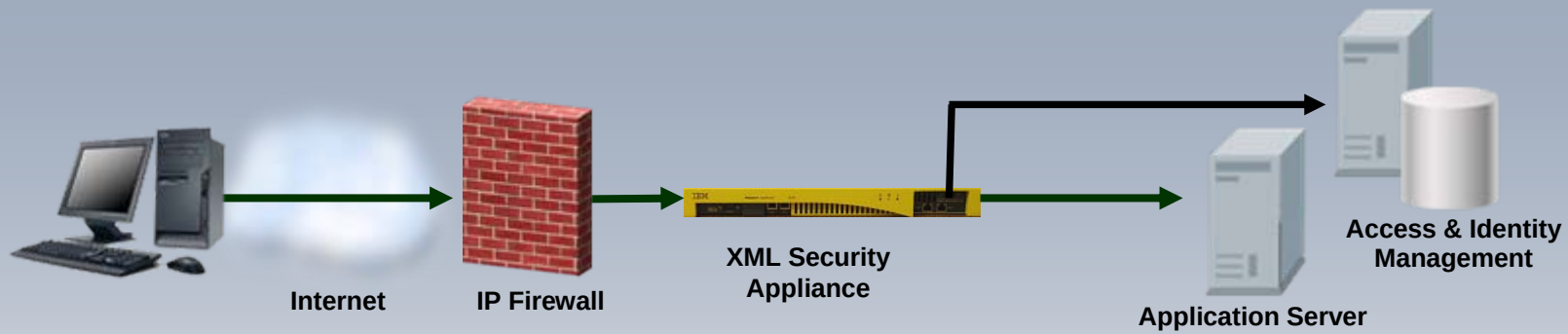
Seguridad SOA – Modelo de confianza



- Identity Federation and Web Services requires trust
  - This trust is based on agreements between partners & expressed as policies
- Trust can be enabled by technology
  - Trust requirements expressed as infrastructure policies and requirements
  - Security tokens include identity information; Cryptographic keys used to sign Security Tokens
- Technology needs to be standards based
  - Standard ways to express and exchange policies that reflect trust relationships
  - Agreed token format, information content, signing and encryption methods

Dispositivos de Seguridad XML

Pueden Simplificar y Acelerar la seguridad en SOA

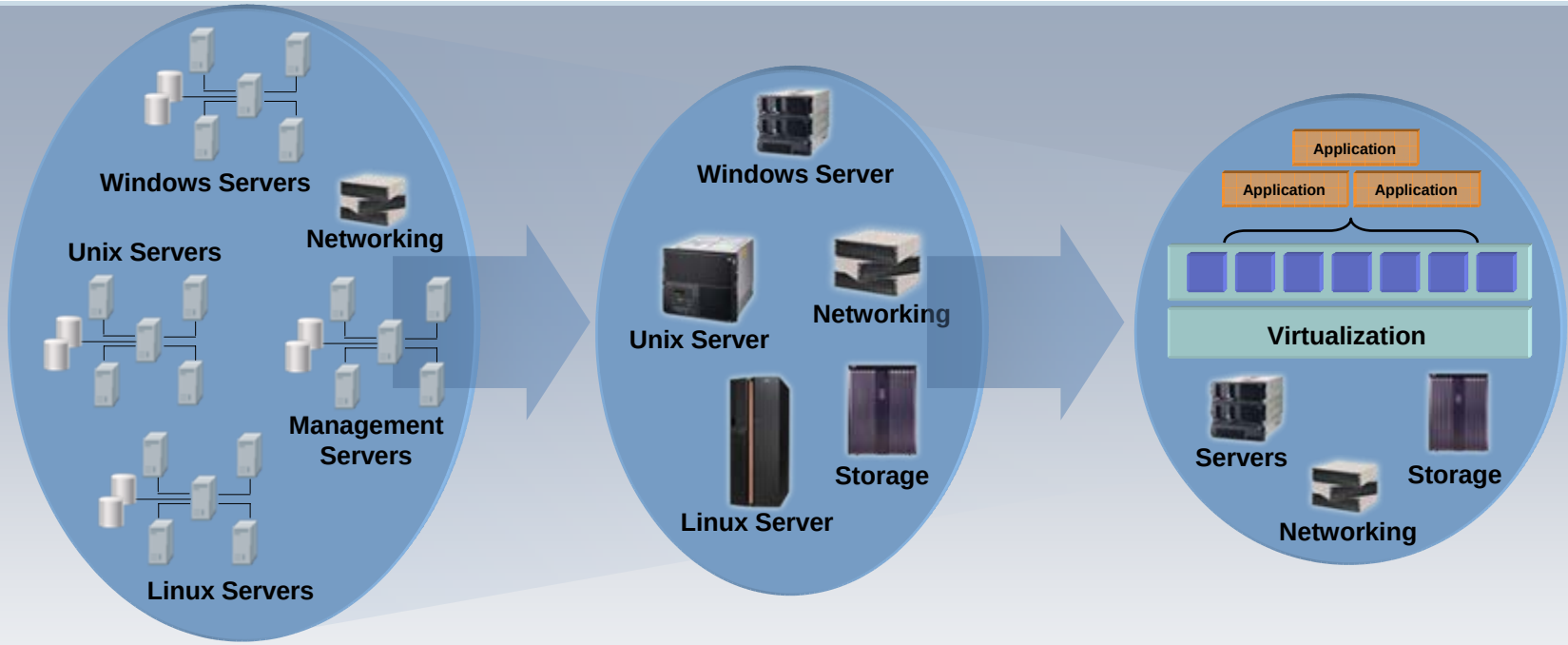


- XML/SOAP firewall enables filtering on any content, metadata or network variables
- Incoming and outgoing XML and SOAP is validated at wire speed
- Security can be performed at the field level
  - WS-Security
  - Encrypt & sign individual fields
  - Non-repudiation
- Provides XML/Web services access control

Mejores Prácticas de Seguridad en SOA

- Security authorization needs to be granular at the service level
- Understand existing corporate security policies (especially approval and audit process) and apply them in the SOA environment
- Work with the SOA application teams to understand the requirements
- Understand the trade-offs of security, performance and cost
- Choose policy-based over programmatic approaches to allow security decisions to be implemented at service invocation
- Evaluate performance implications of security implementations
- Consider XML appliances to accelerate security processing

Virtualization como elemento para liberar Aplicaciones de la Infraestructura



Complex

- Islands of computing and data
- Physical resources are bound to applications
- Disparate management tools
- Manual provisioning

Consolidated

- Fewer devices and licenses
- Increased utilization
- Physical resources still bound to applications
- disparate management tools
- Labor intensive provisioning

Virtualized

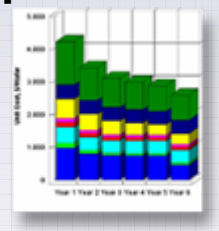
- Pools of resources
- Logic and physical resources decoupled
- Standardized, automated infrastructure management
- Automated provisioning

Optimización & Virtualización de Infraestructura

Optimize infrastructure investment and prioritize applications and users in a mission-critical manner

- Provide high availability and redundancy for business-critical applications
- Increase server utilization to optimize capital & administration costs
- Ensure that the most important applications and users are given priority according to business and IT policies
- Flexibly respond to unforeseen application demand

Resource Optimization



Utilization

Application Prioritization

Importance

High Availability

Assurance

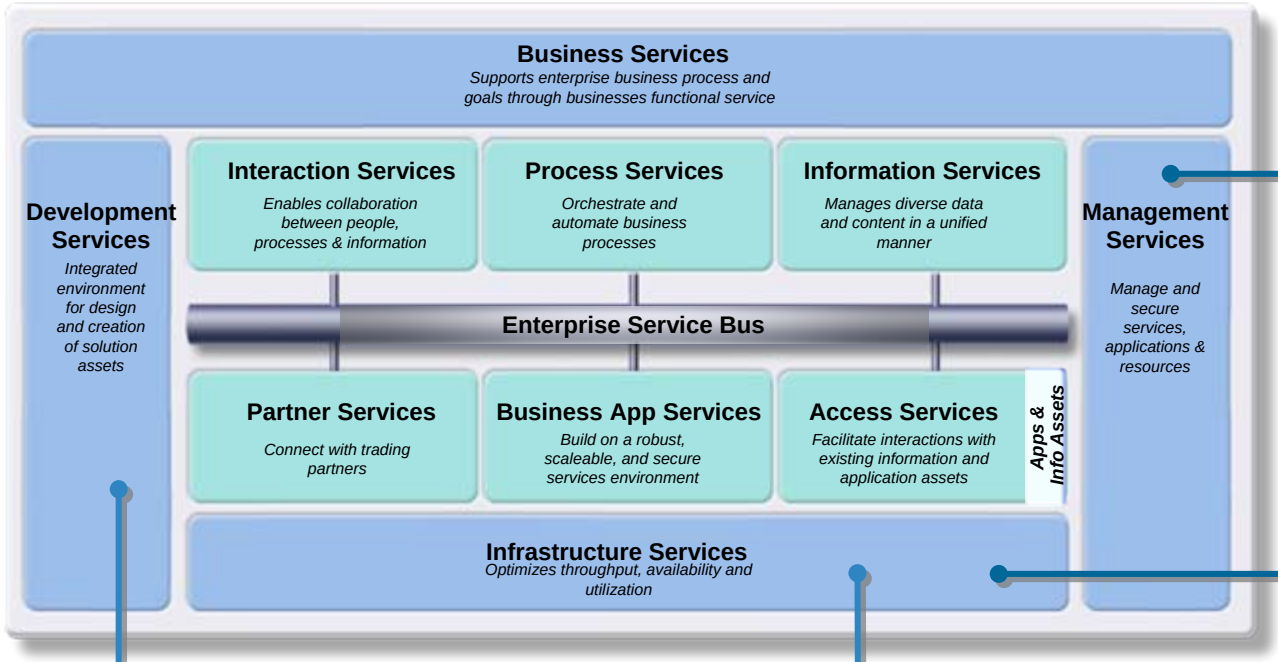
Mejores Prácticas para Virtualization bajo SOA

- Consolidate servers, storage & network assets for greater efficiency & reduced complexity
- IT resources should be used across applications without regard to where they physically reside
- Replace error-prone manual tasks & repetitive IT resource/capacity management tasks with automated capabilities
- Dynamically allocate IT capacity to meet business goals for increased infrastructure agility and readiness for growth

Agenda

- SOA requiere una nueva forma de pensar en las Infraestructuras
- Consideraciones de Infraestructuras en SOA
  - Rendimiento
  - Disponibilidad
  - Gestión de Servicio
  - Seguridad
  - Virtualización
- **Capacidades de IBM para dar soporte a Arquitecturas de Infraestructuras SOA**
- **Conclusiones**

Software IBM para la mejora de Infraestructuras SOA



Tivoli Composite Application Managers

Tivoli OMEGAMON XE for Messaging

Tivoli Federated Identity Manager

Tivoli Access Manager

Tivoli Provisioning Manager

Tivoli Intelligent Orchestrator

Rational Tester for SOA Quality

Rational Performance Tester Extension for SOA Quality

WebSphere XD

WebSphere DataPower SOA Appliances

Claves de diseño de Infraestructuras TI preparadas para SOA

- In the real-world, SOA-based applications put a lot of stress on a typical infrastructure
- From a business view, the application layer is geared towards simplification but the infrastructure can become complex
- The IT Infrastructure/Middleware Architect cannot let the SOA application become a "*black box*" within the infrastructure
- Visibility of quality of service metrics within the SOA application is crucial to achieving performance and availability goals
- As an IT Infrastructure Architect, one needs to know what is in the toolbox and how to build the best infrastructure for the SOA application

Hindi

Traditional Chinese

Thai

Russian

Spanish

Arabic

Thank You

Slovenian

Italian

German

French

Simplified Chinese

Hungarian

Korean

Japanese

IBM SOA Architect Summit

SOA on your terms and our expertise