IBM

IBM SOA Architect Summit

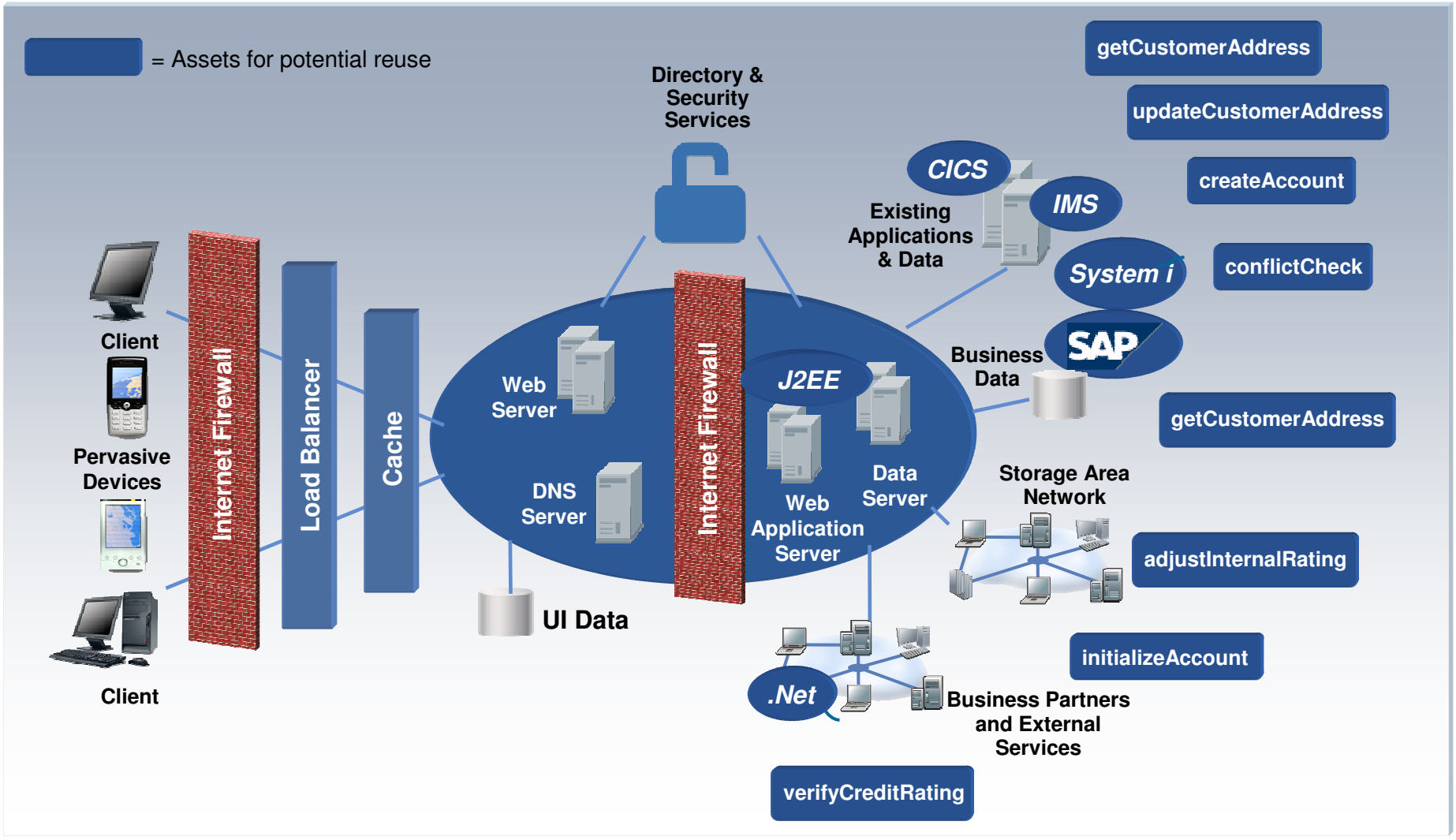
GET PRACTICAL HELP TO MEET THE DEMANDS OF YOUR BUSINESS.

Extracting Value From What is Already There

Potential Issues and Expected Value From Reusing Existing Applications

Richard G Brown

SOA Enables Greater Reuse of Existing Assets



Application Architecture Considerations

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

Service-enable Mainframe Assets

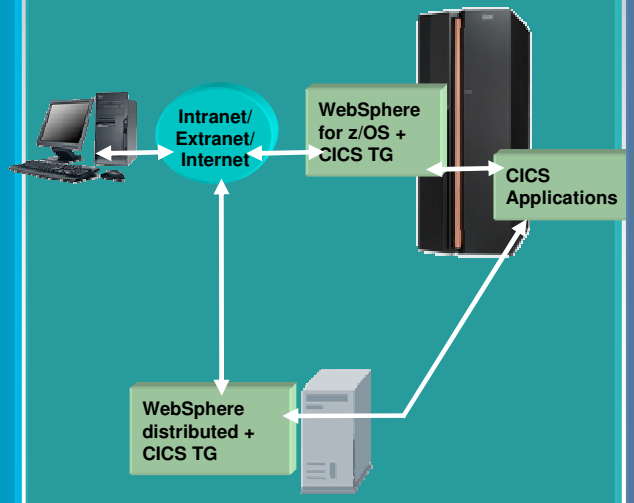
Make Better Use of CICS & IMS Investments

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

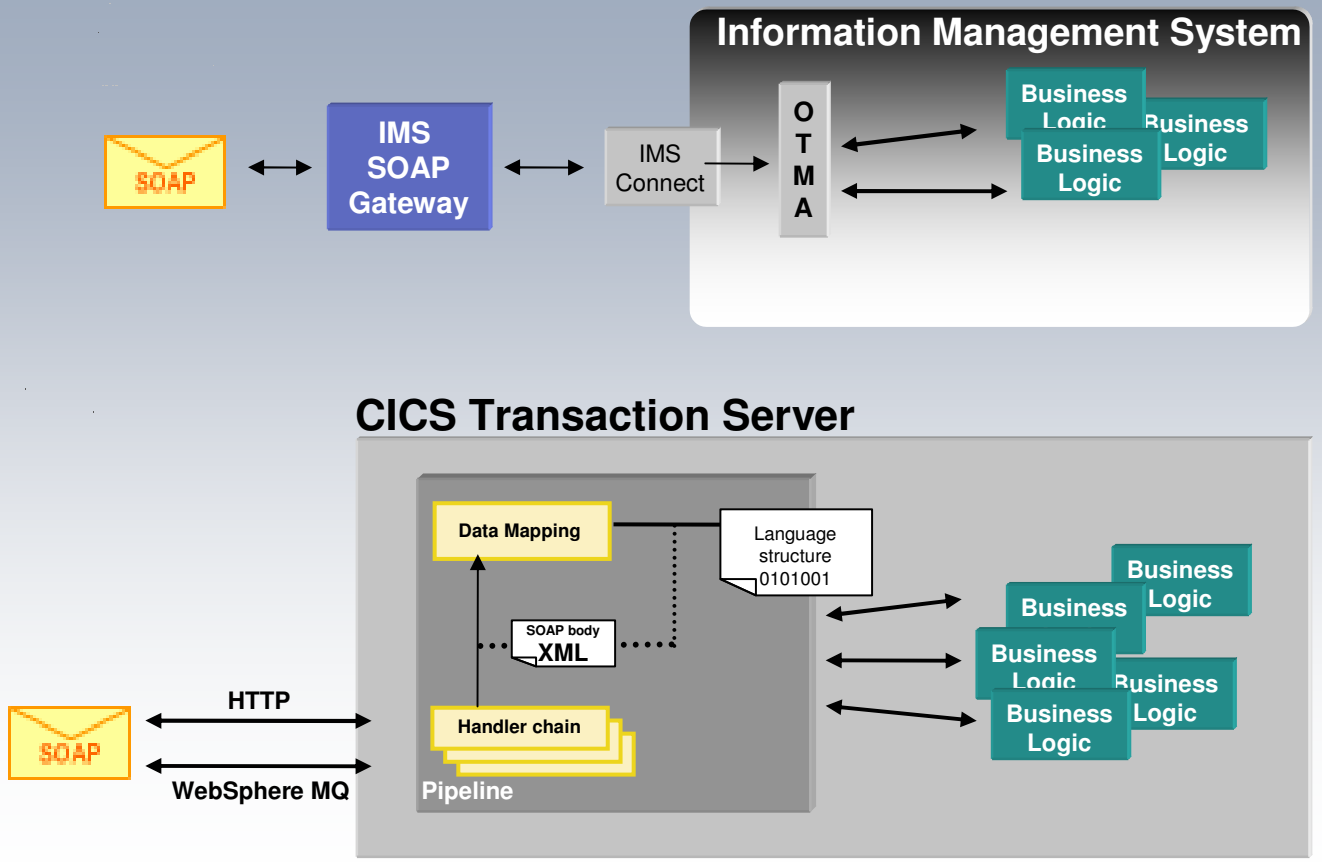
- CICS & IMS can both expose transactions as native Web Services
- No other runtimes required

So many options…

… let's focus on patterns

Direct Access Pattern Example

CICS and IMS Native Web Services



Direct Access Pattern

Benefits:

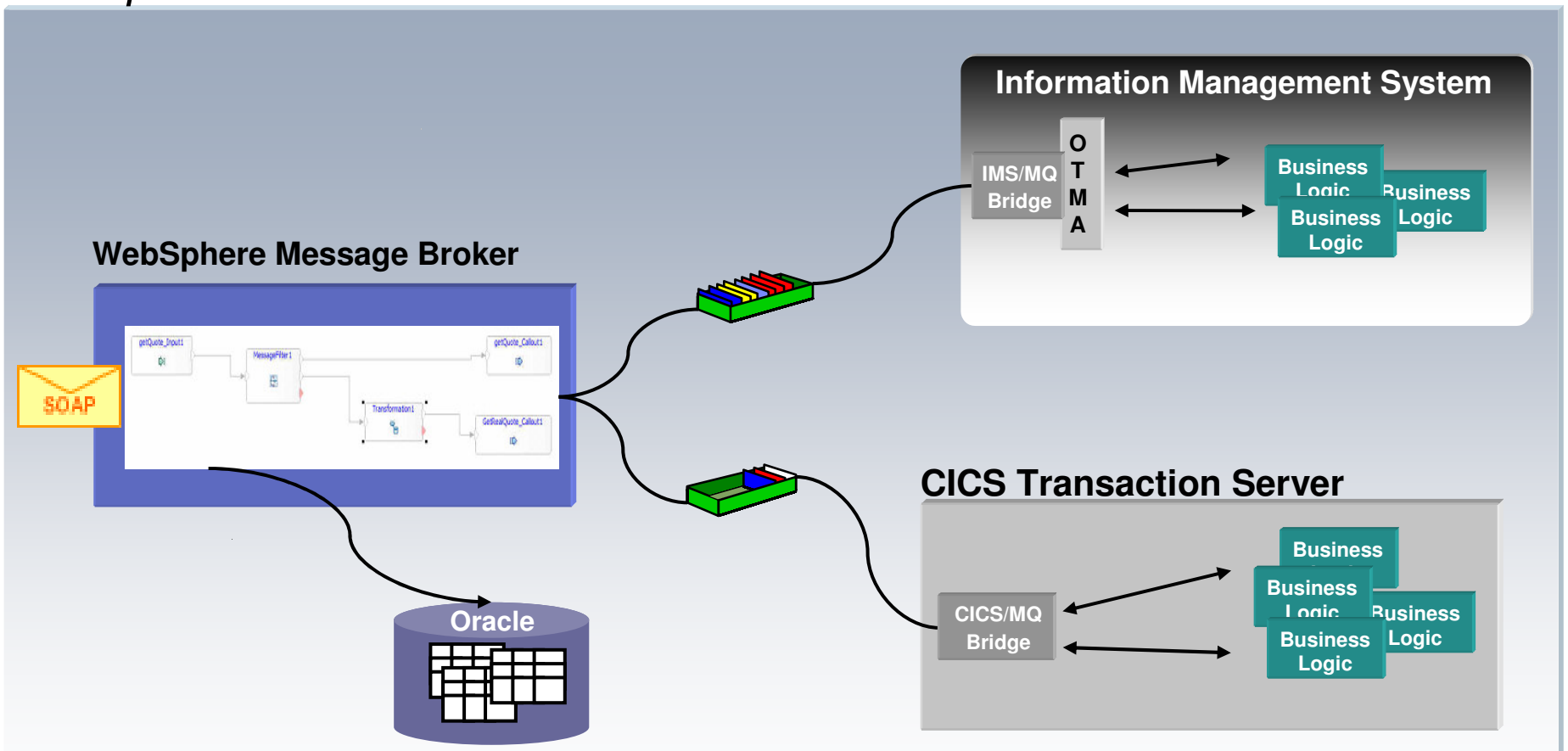
- Shorter deployment cycle … compared to indirect access
- The service interface is defined by the asset
  - No analysis required to determine the interface
- No knowledge of other runtimes (Java, Message Broker, etc.) is necessary
- Fewer platforms/moving parts

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

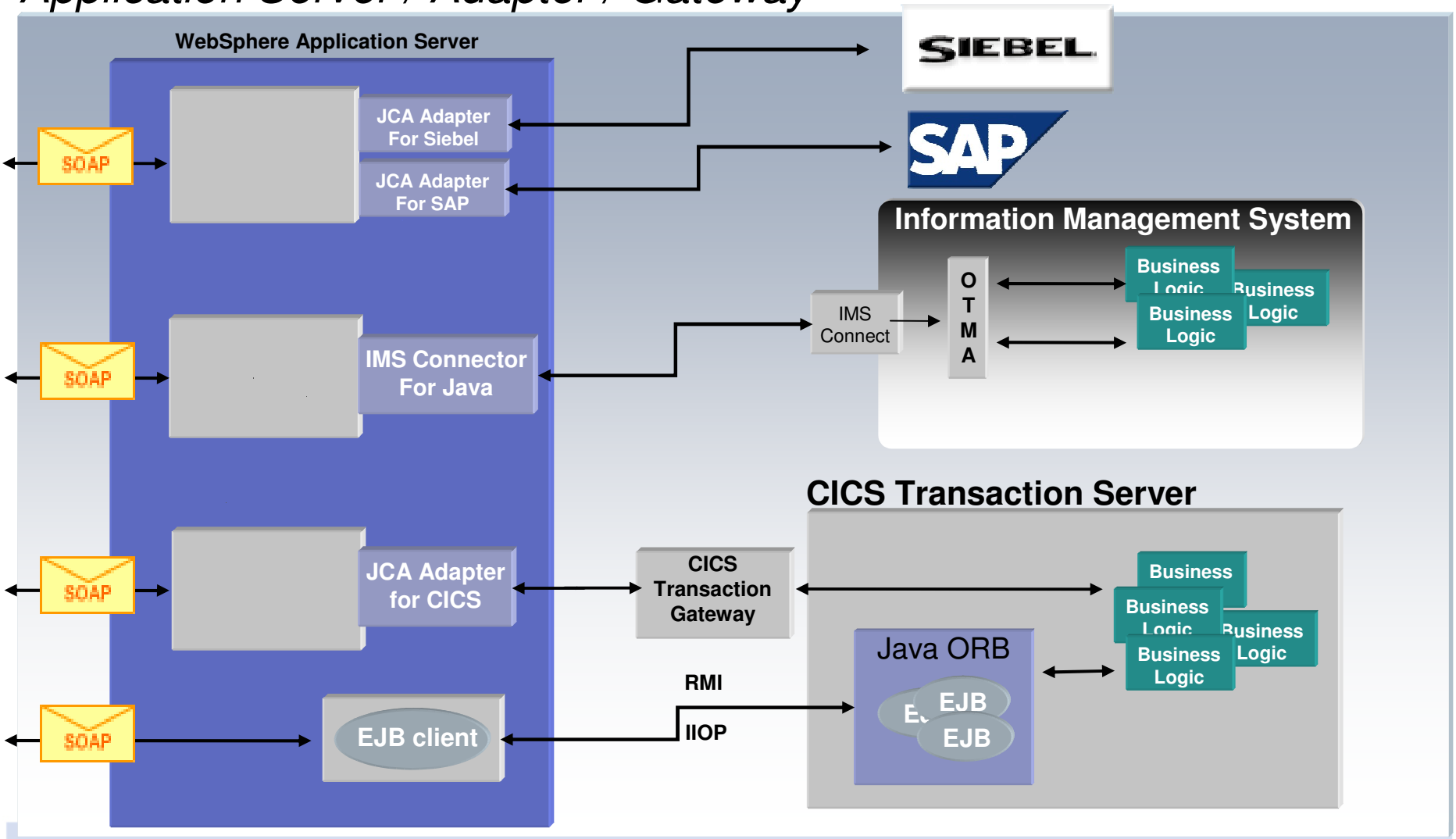
Indirect Pattern Example

Enterprise Service Bus



Indirect Pattern Example

Application Server / Adapter / Gateway



IBM

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

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		■

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
  - *Isn't that spaghetti really 40 years of accumulated knowledge?*

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
- (Next Presentation)