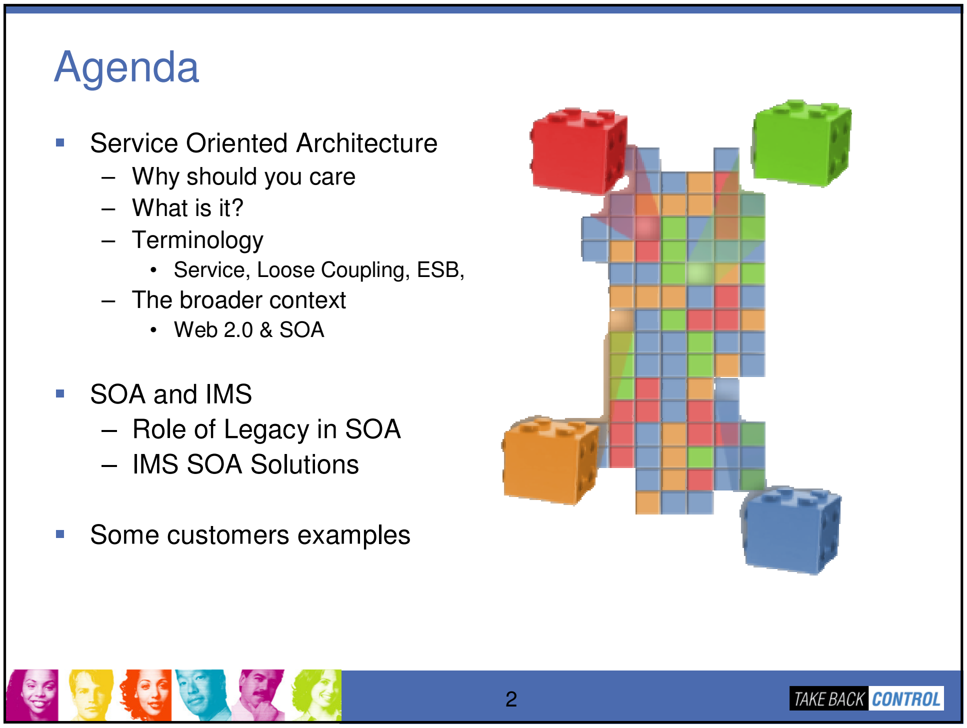
SOA What?

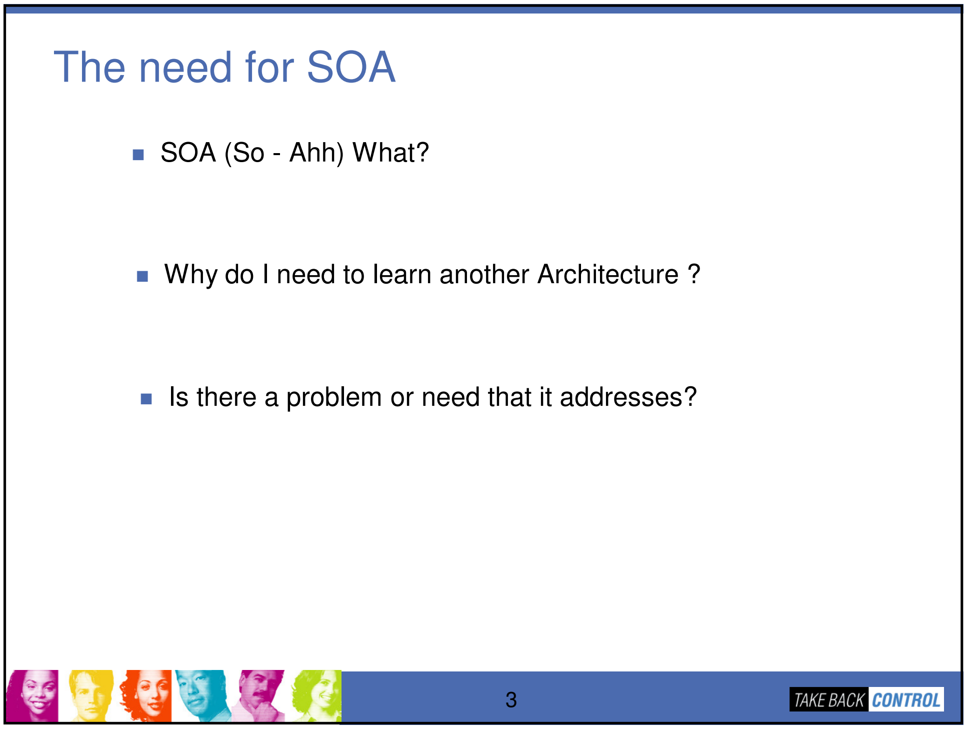
TAKE BACK CONTROL



Agenda

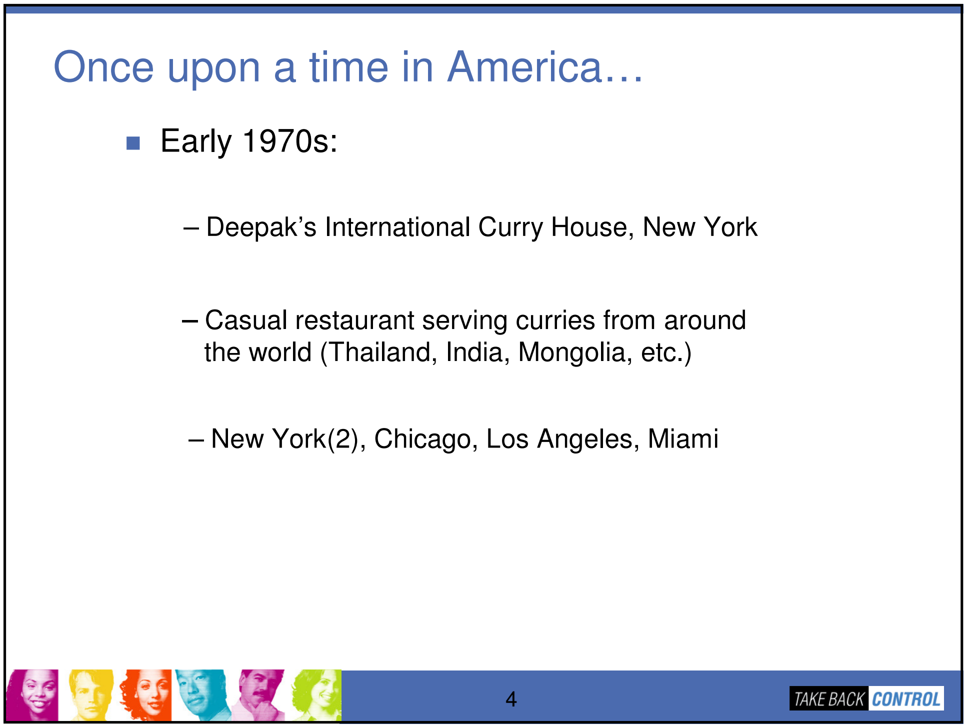
- Service Oriented Architecture
  - Why should you care
  - What is it?
  - Terminology
    - Service, Loose Coupling, ESB,
  - The broader context
    - Web 2.0 & SOA
- SOA and IMS
  - Role of Legacy in SOA
  - IMS SOA Solutions
- Some customers examples

TAKE BACK CONTROL



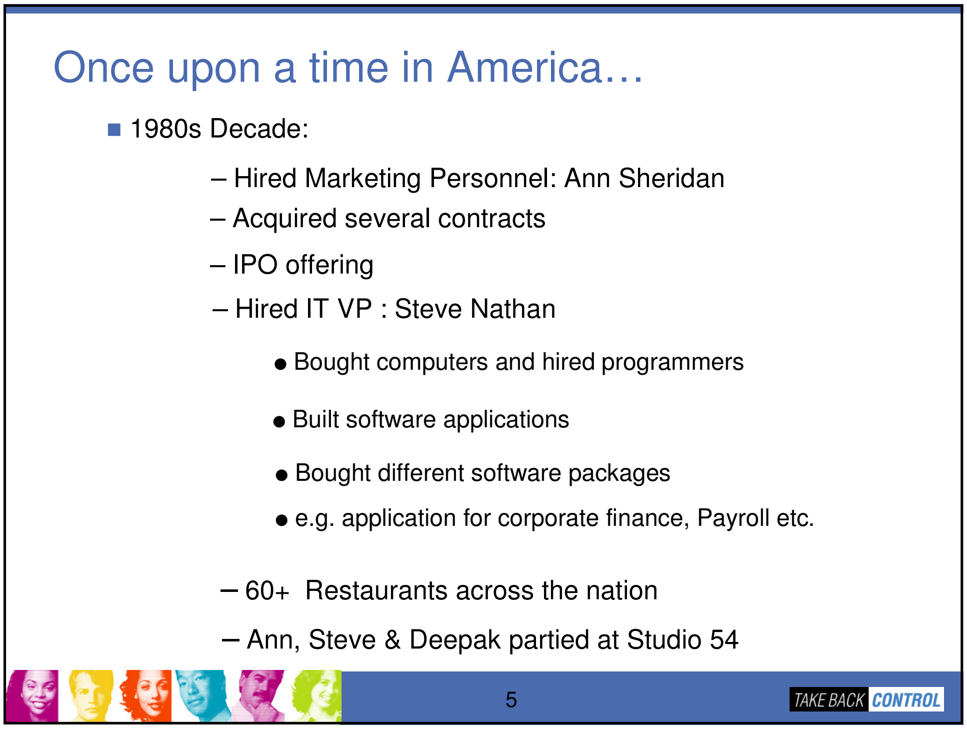
The need for SOA

- SOA (So - Ahh) What?
- Why do I need to learn another Architecture ?
- Is there a problem or need that it addresses?



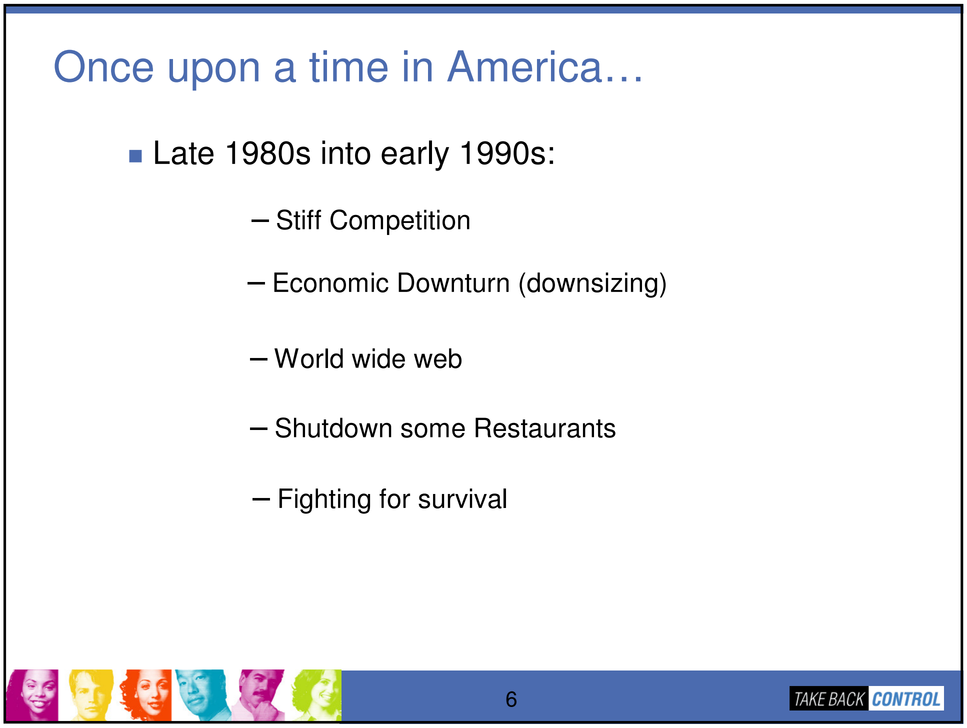
Once upon a time in America…

- Early 1970s:
  - Deepak's International Curry House, New York
  - Casual restaurant serving curries from around the world (Thailand, India, Mongolia, etc.)
  - New York(2), Chicago, Los Angeles, Miami



Once upon a time in America…

- 1980s Decade:
  - – Hired Marketing Personnel: Ann Sheridan
  - – Acquired several contracts
  - – IPO offering
  - – Hired IT VP : Steve Nathan
    - ● Bought computers and hired programmers
    - ● Built software applications
    - ● Bought different software packages
    - ● e.g. application for corporate finance, Payroll etc.
  - – 60+ Restaurants across the nation
  - – Ann, Steve & Deepak partied at Studio 54



Once upon a time in America…

- Late 1980s into early 1990s:
  - – Stiff Competition
  - – Economic Downturn (downsizing)
  - – World wide web
  - – Shutdown some Restaurants
  - – Fighting for survival

Once upon a time in America…

- Cost cutting
- New ways to make money:
  - Cyber Curry House in malls across America
  - Acquire a computer manufacturing company
- IT's (Steve Nathan's) response:

  "Woa Woa, Hold on Deepak. We have a problem.

  Acquired company's software has to tie into our software.

  Our day-to-day business functions, policies & information are locked inside our programs."
- Can't we integrate ?
- IT's (Steve Nathan's) response:

  "Yes, but it will take a very, very long time.
  May be 18 months to 2 years.
  And it will cost lots and lots of money"

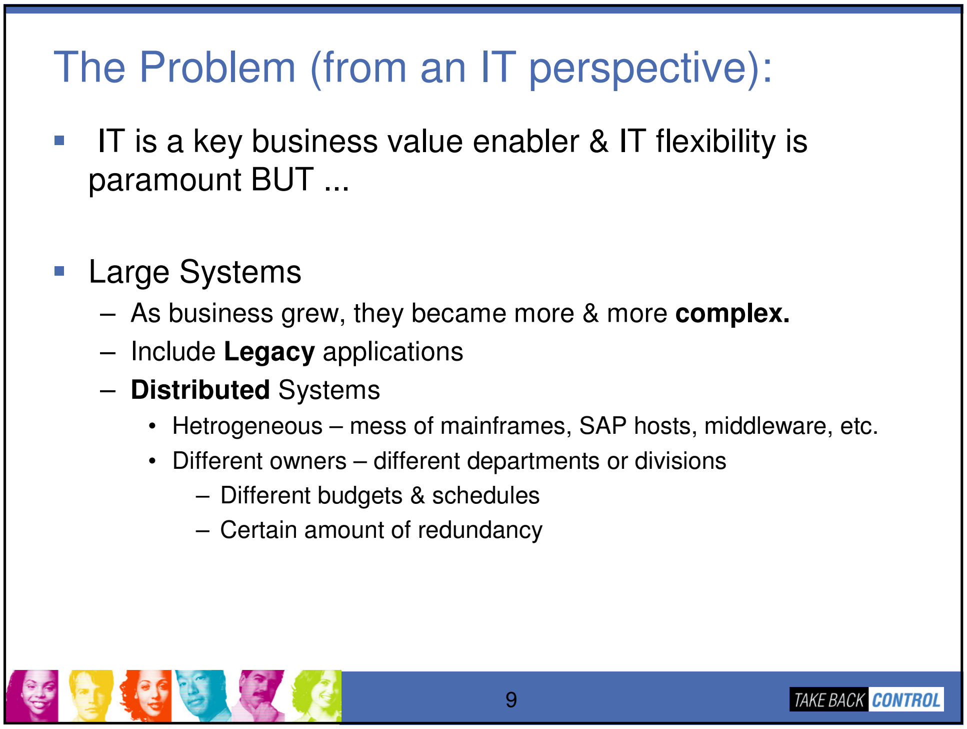
TAKE BACK CONTROL

The Problem (from a business perspective):

- Challenges of the Modern Business World
- Social market economy is being replaced by a Global market economy & the marketing guys rule the world
- The *Speed of Business* is rapidly increasing
  - Must respond to market changes **more quickly**
  - Must respond to market changes **more cheaply**
- Business environments have and continue to become more distributed
  - Assets and business information need to be accessible from everywhere
- Business environments are becoming more complex

  "Its not the strongest of the species that survive, nor the most intelligent, but the ones most responsive to change."

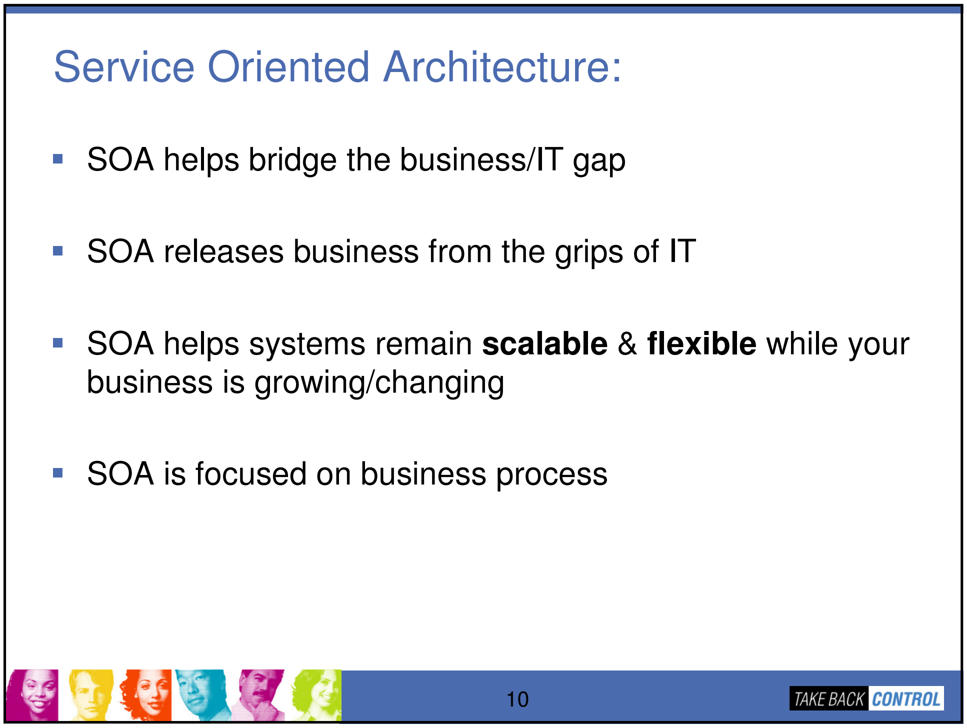
TAKE BACK CONTROL



The Problem (from an IT perspective):

- IT is a key business value enabler & IT flexibility is paramount BUT ...
- Large Systems
  - As business grew, they became more & more **complex.**
  - Include **Legacy** applications
  - **Distributed** Systems
    - Hetrogeneous – mess of mainframes, SAP hosts, middleware, etc.
    - Different owners – different departments or divisions
      - Different budgets & schedules
      - Certain amount of redundancy

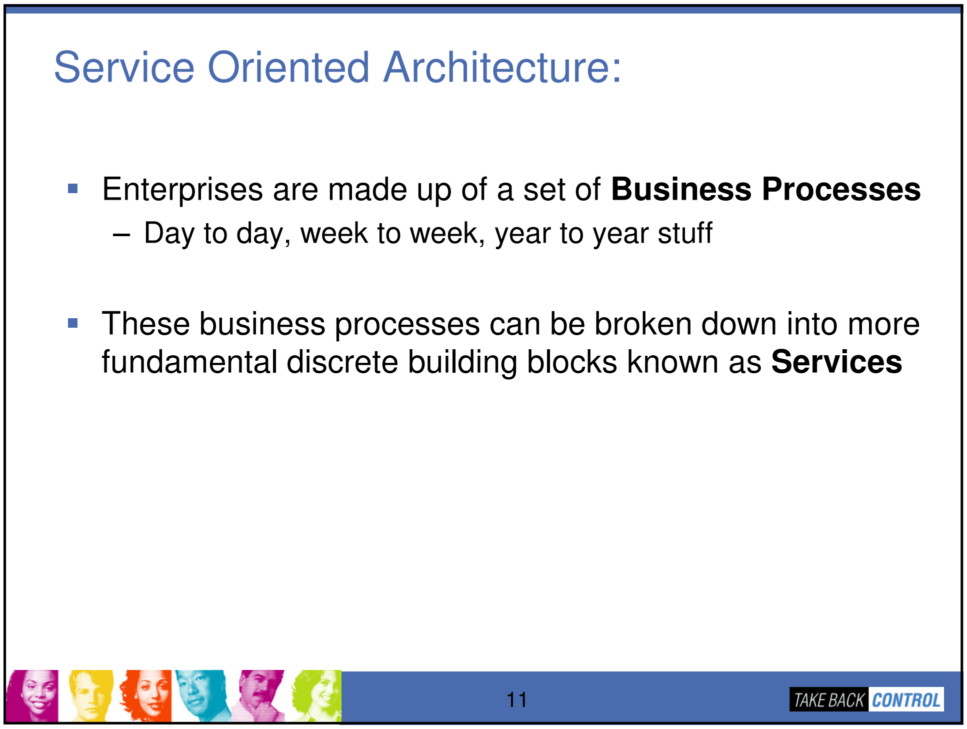
TAKE BACK CONTROL



Service Oriented Architecture:

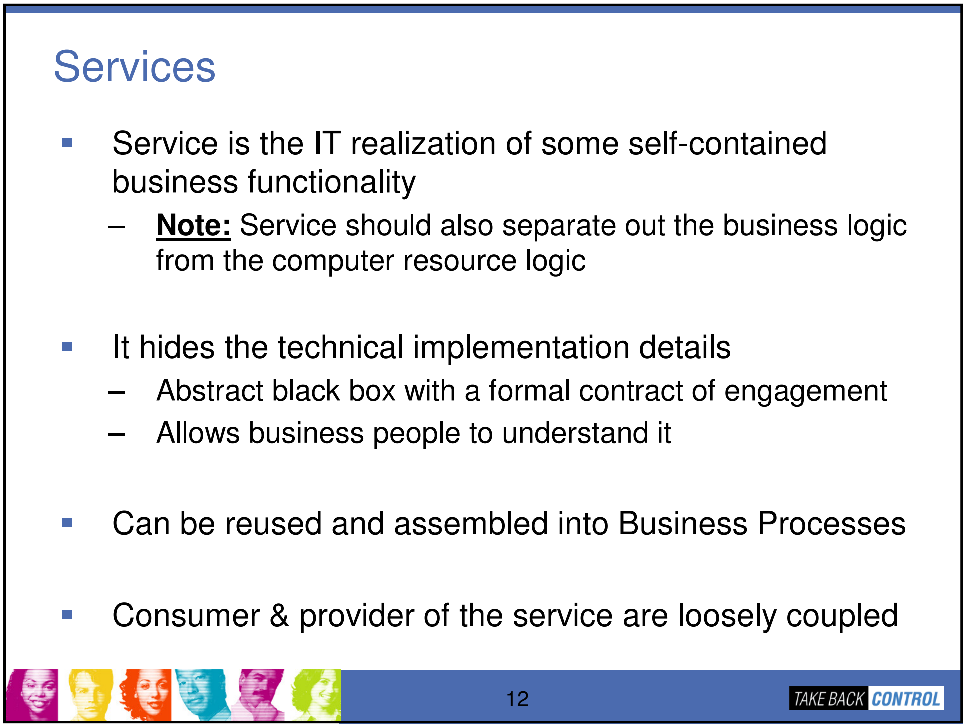
- SOA helps bridge the business/IT gap
- SOA releases business from the grips of IT
- SOA helps systems remain **scalable** & **flexible** while your business is growing/changing
- SOA is focused on business process

TAKE BACK CONTROL



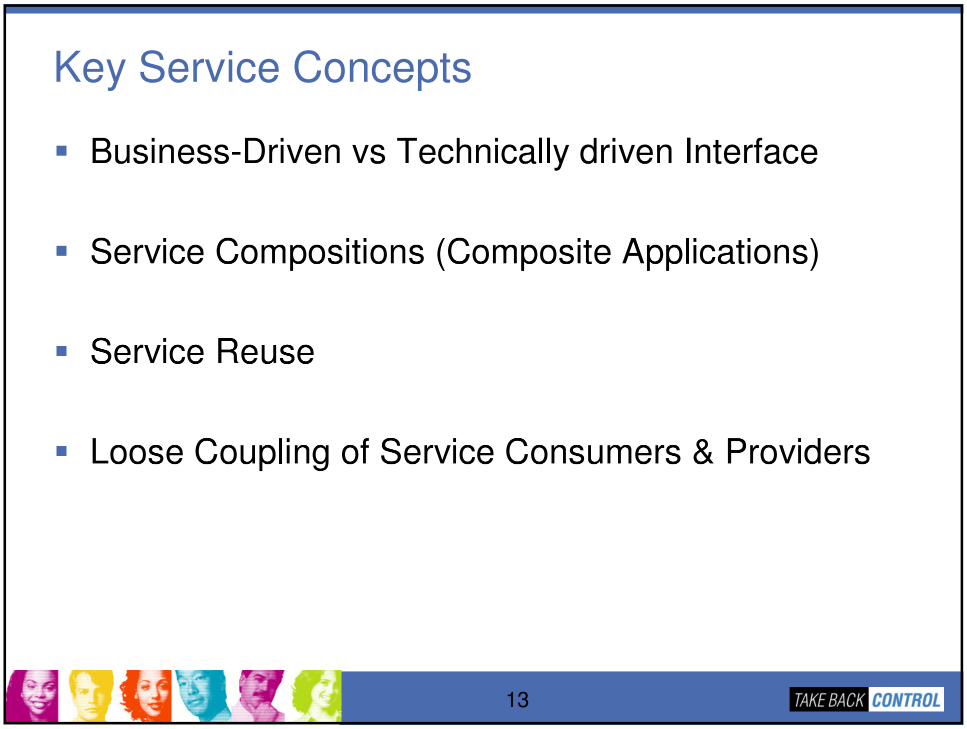
Service Oriented Architecture:

- Enterprises are made up of a set of **Business Processes**
  - Day to day, week to week, year to year stuff
- These business processes can be broken down into more fundamental discrete building blocks known as **Services**



Services

- Service is the IT realization of some self-contained business functionality
  - **Note:** Service should also separate out the business logic from the computer resource logic
- It hides the technical implementation details
  - Abstract black box with a formal contract of engagement
  - Allows business people to understand it
- Can be reused and assembled into Business Processes
- Consumer & provider of the service are loosely coupled

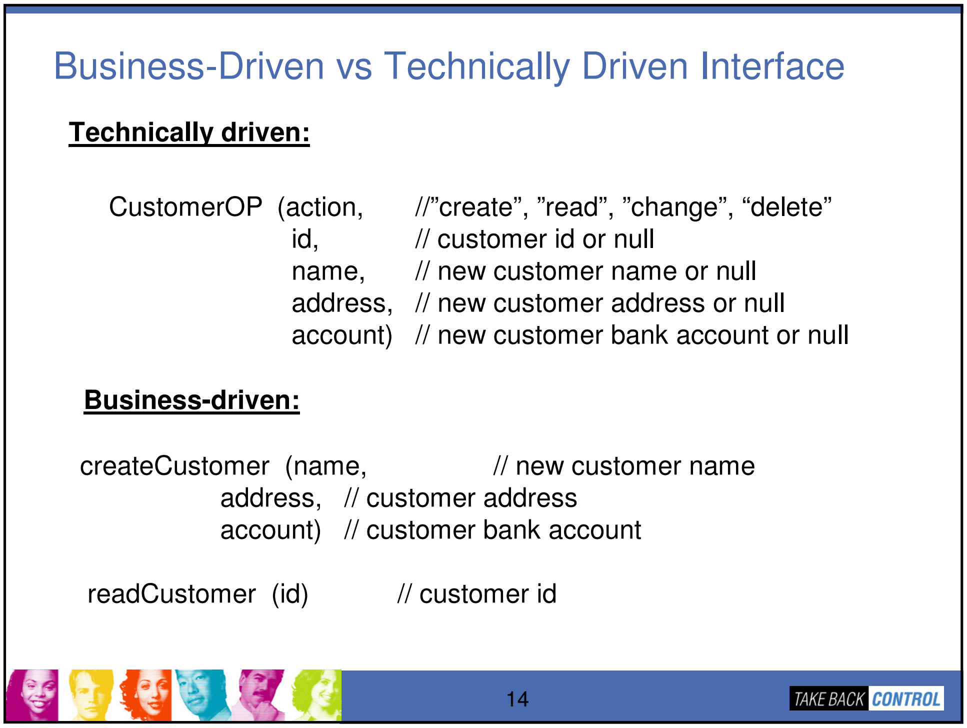


Key Service Concepts

- Business-Driven vs Technically driven Interface
- Service Compositions (Composite Applications)
- Service Reuse
- Loose Coupling of Service Consumers & Providers

13

TAKE BACK CONTROL



Business-Driven vs Technically Driven Interface

Technically driven:

```
CustomerOP  (action,      //”create”, ”read”, ”change”, “delete”
              id,          // customer id or null
              name,        // new customer name or null
              address,     // new customer address or null
              account)     // new customer bank account or null
```

Business-driven:

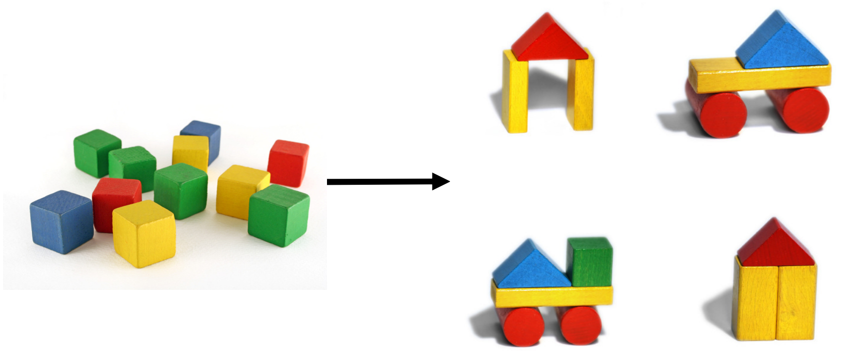
```
createCustomer  (name,                // new customer name
                 address,   // customer address
                 account)   // customer bank account

readCustomer  (id)           // customer id
```

14

TAKE BACK CONTROL

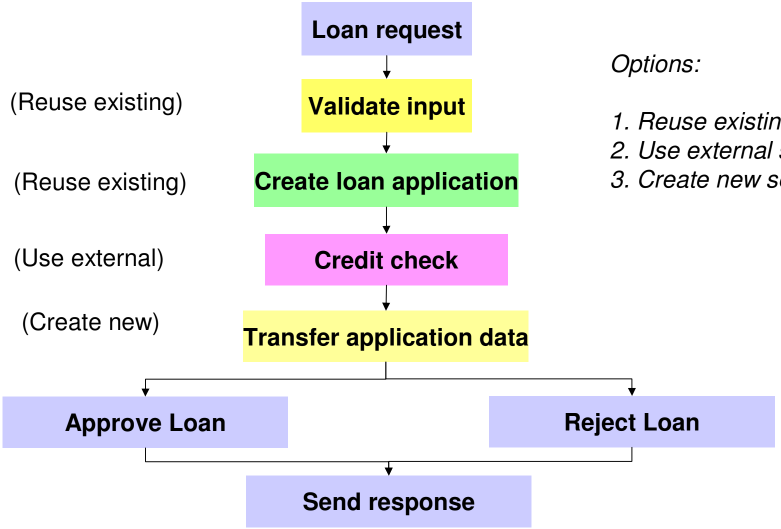
Service Composition (Composite Applications)

Business process building blocks



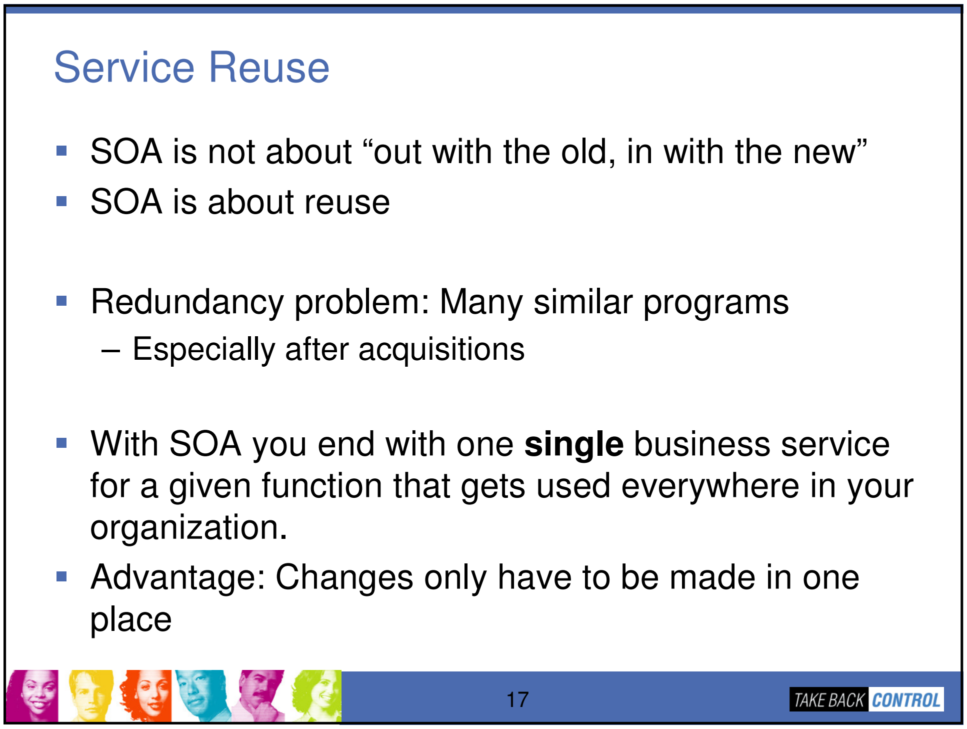
This is what Business wants! They want to play with their Lego and make lots and lots of money

Example: Mortgage Loan Application Process



Options:

1. *Reuse existing service*
2. *Use external service*
3. *Create new service*

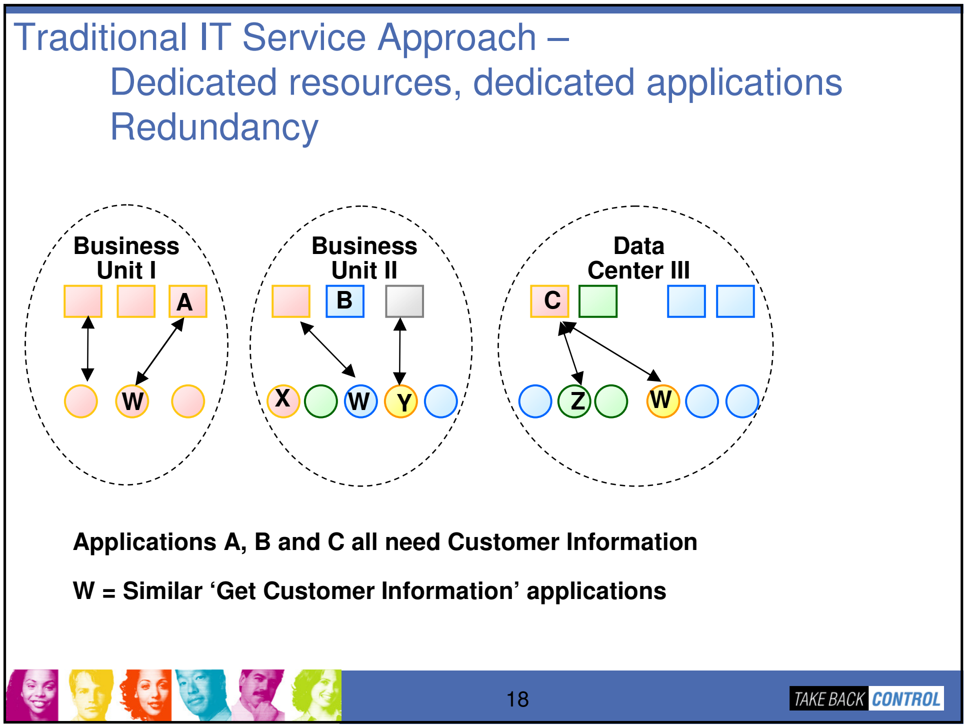


Service Reuse

- SOA is not about “out with the old, in with the new”
- SOA is about reuse

- Redundancy problem: Many similar programs
  - Especially after acquisitions

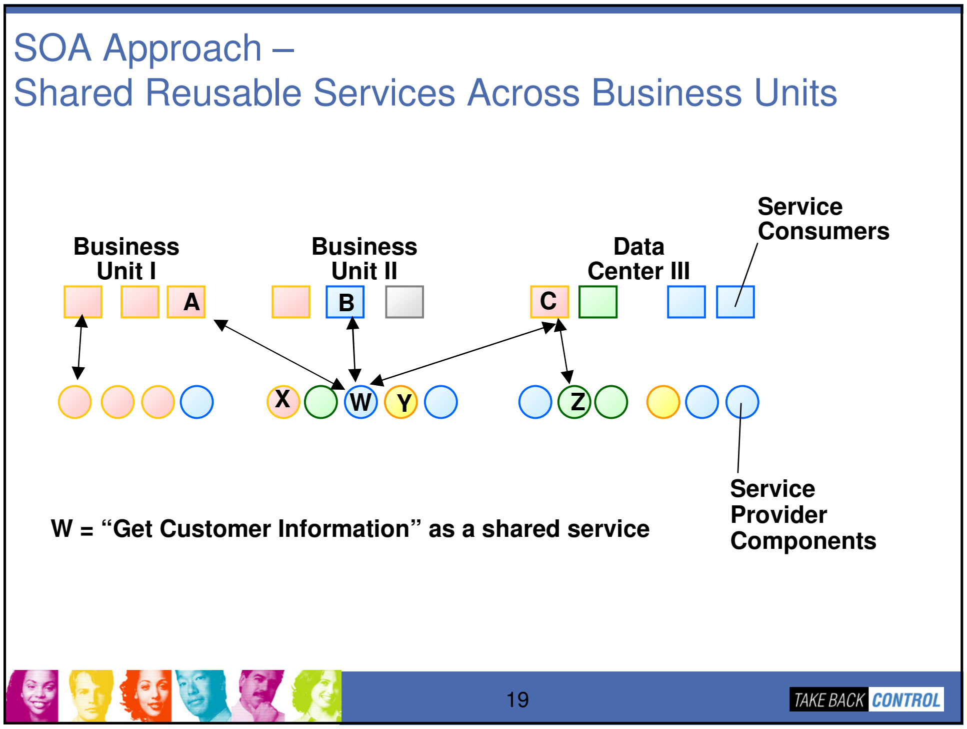
- With SOA you end with one **single** business service for a given function that gets used everywhere in your organization.
- Advantage: Changes only have to be made in one place



Traditional IT Service Approach – Dedicated resources, dedicated applications Redundancy

Applications A, B and C all need Customer Information

W = Similar ‘Get Customer Information’ applications



SOA Approach – Shared Reusable Services Across Business Units

Business Unit I

Business Unit II

Data Center III

Service Consumers

A

B

C

X

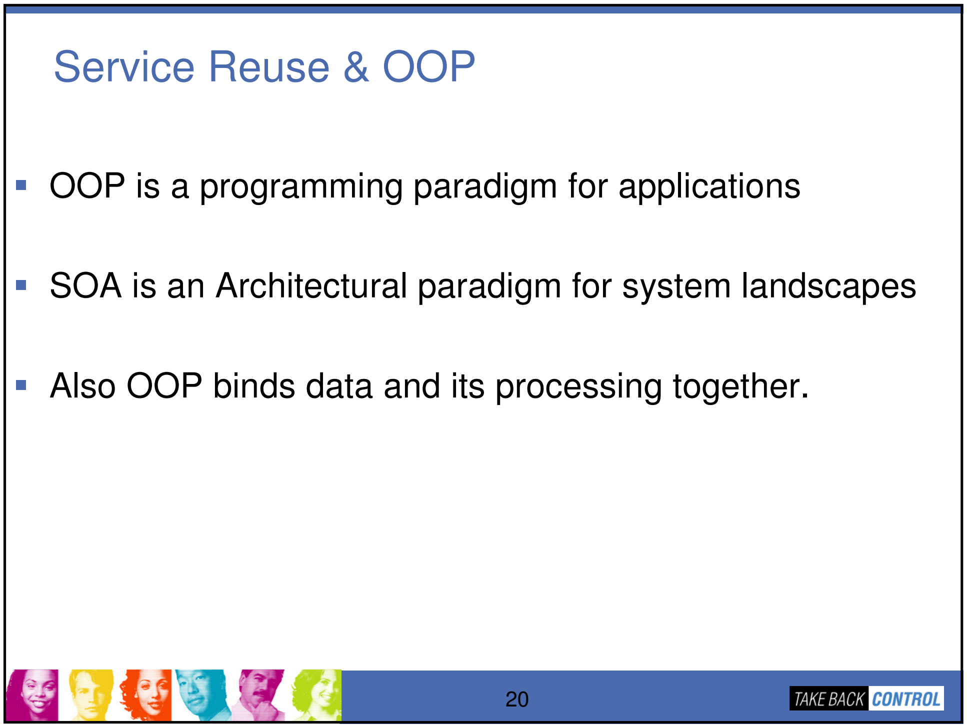
W

Y

Z

Service Provider Components

W = “Get Customer Information” as a shared service



Service Reuse & OOP

- OOP is a programming paradigm for applications
- SOA is an Architectural paradigm for system landscapes
- Also OOP binds data and its processing together.

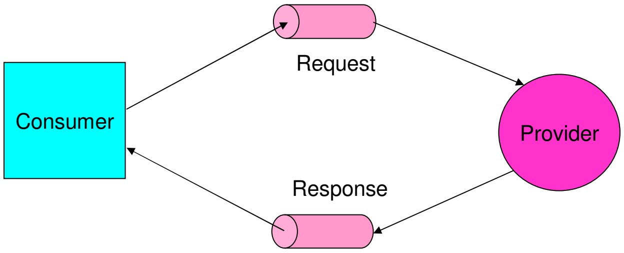
Loose Coupling of Service Providers & Consumers

- Aim of loose coupling is to minimize dependencies
- When there are fewer dependencies, modifications to or faults in one system will have fewer consequences on other systems.

Possible forms of loose coupling in SOA:

	Tight coupling	**Loose coupling**
Physical Connections	Point-to-point	Via mediator
Communication style	Synchronous	Asynchronous
Data Model	Common Complex Type	Simple common types only
Type System	Strong	Weak
Interaction Pattern	Navigate through complex object trees	Data-centric, self-contained message
Control of process logic	Central control	Distributed control
Binding	Statically	Dynamically

Asynchronous Communication

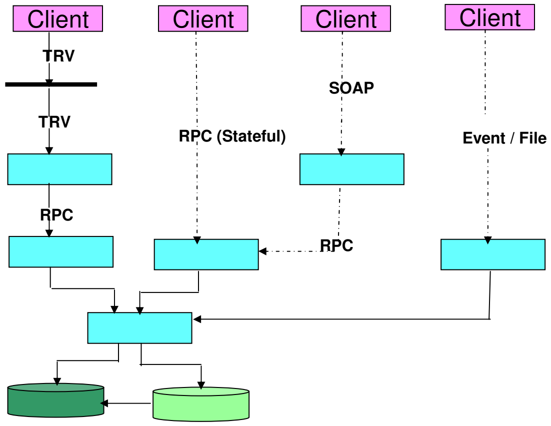


- How do you associate the response to the request (correlation ID)
- Processing the response requires knowledge of initial state & context
- What if you send multiple async requests the responses are not in order?
  - Testing and debugging this can be complicated

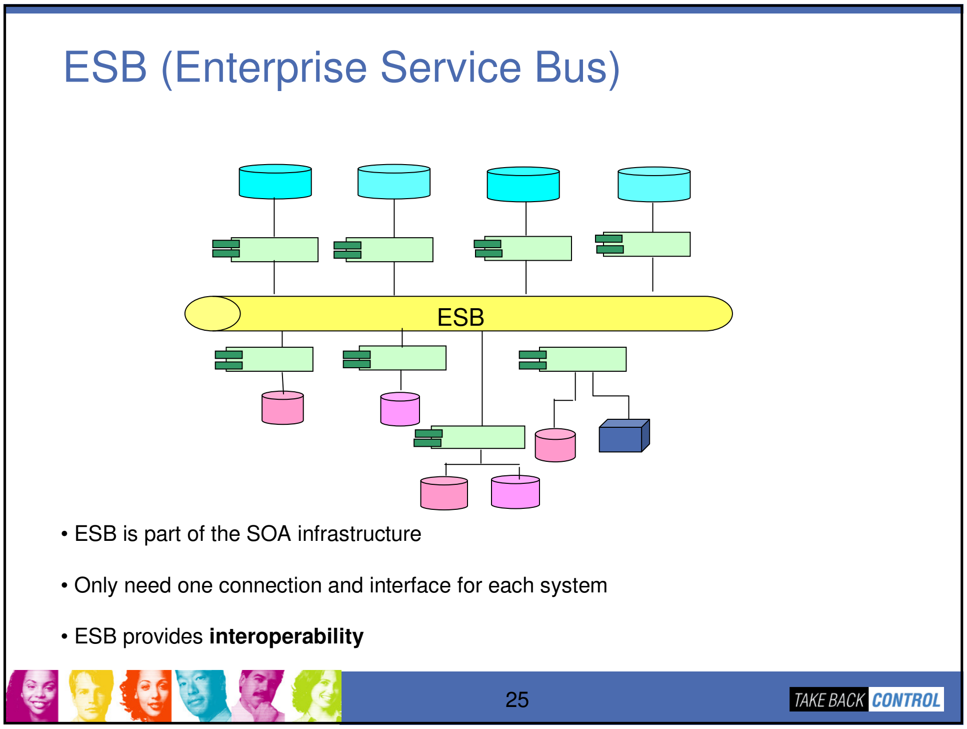
More Concepts:
Business Service Registry and Repository

- Publish and Find Services
- Govern
  - Full access control for management of who can access which services
- Enrich
  - Enable ESB to find best -fitting endpoint when request is received
- Manage
  - Enable policy enforcement and conduct impact analysis

The Reason for ESB (Enterprise Service Bus)

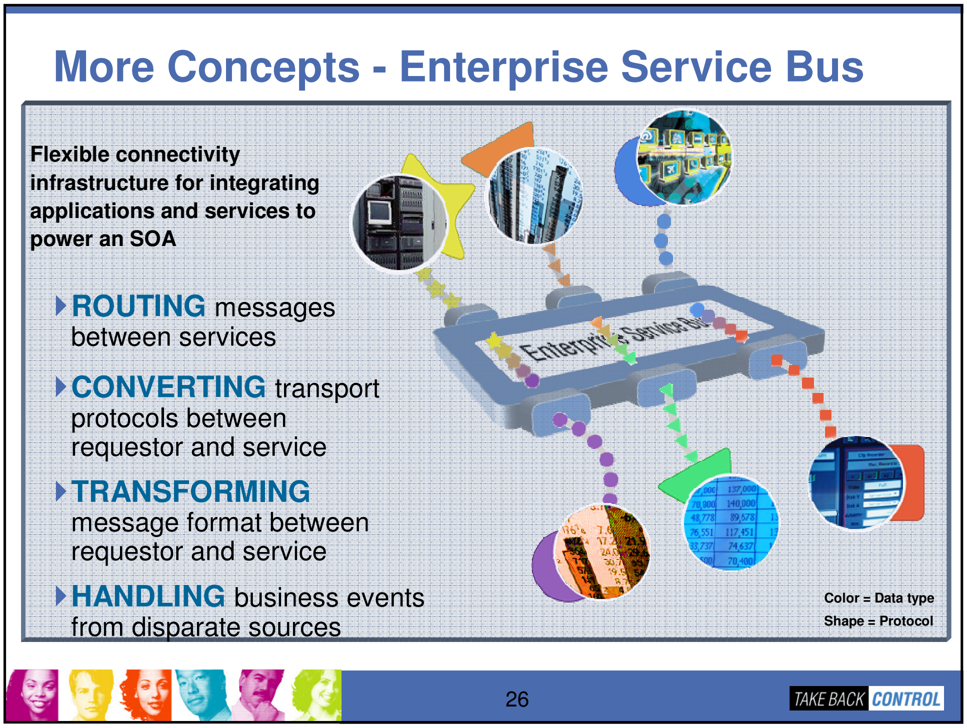


- n Systems require n x (n-1)/2 connections



ESB (Enterprise Service Bus)

- ESB is part of the SOA infrastructure
- Only need one connection and interface for each system
- ESB provides **interoperability**



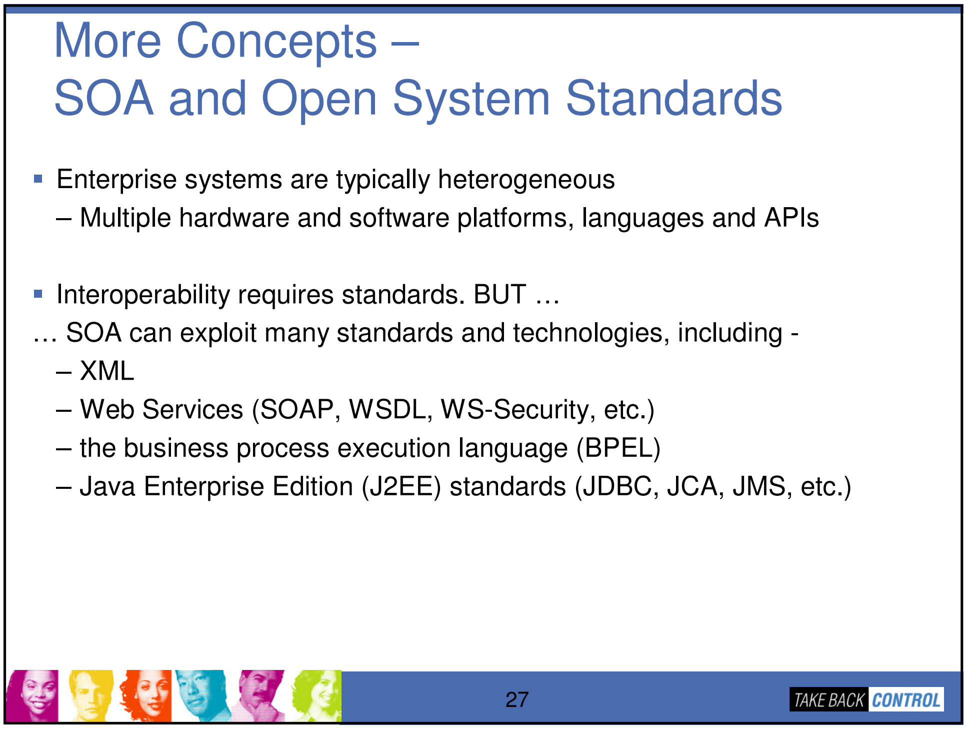
More Concepts - Enterprise Service Bus

Flexible connectivity infrastructure for integrating applications and services to power an SOA

- **ROUTING** messages between services
- **CONVERTING** transport protocols between requestor and service
- **TRANSFORMING** message format between requestor and service
- **HANDLING** business events from disparate sources

Color = Data type

Shape = Protocol



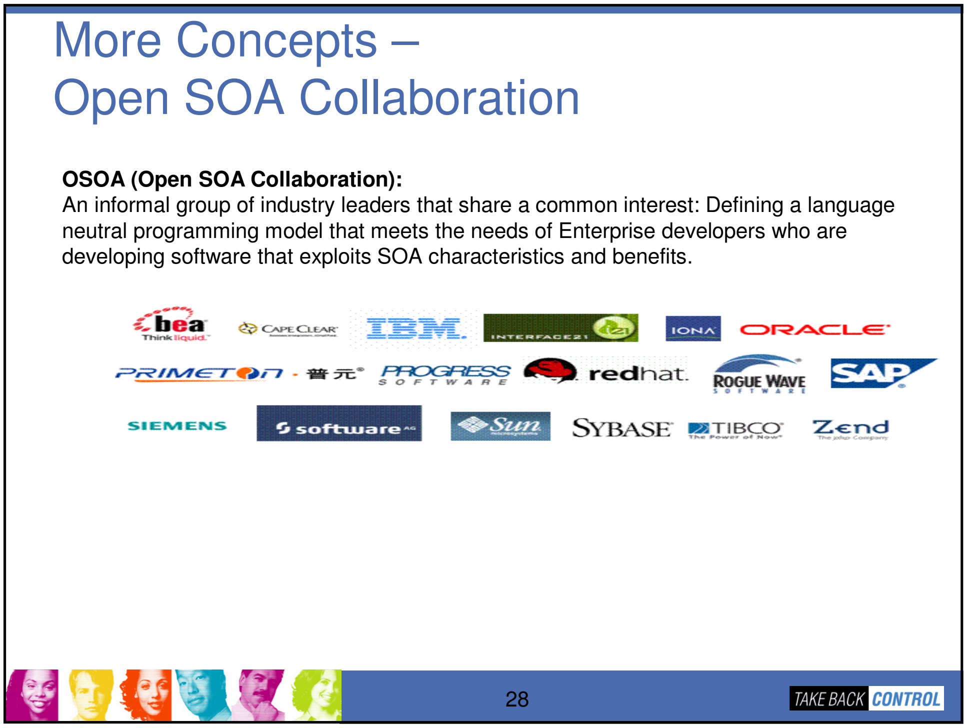
More Concepts – SOA and Open System Standards

- Enterprise systems are typically heterogeneous
  - – Multiple hardware and software platforms, languages and APIs

- Interoperability requires standards. BUT …

… SOA can exploit many standards and technologies, including -

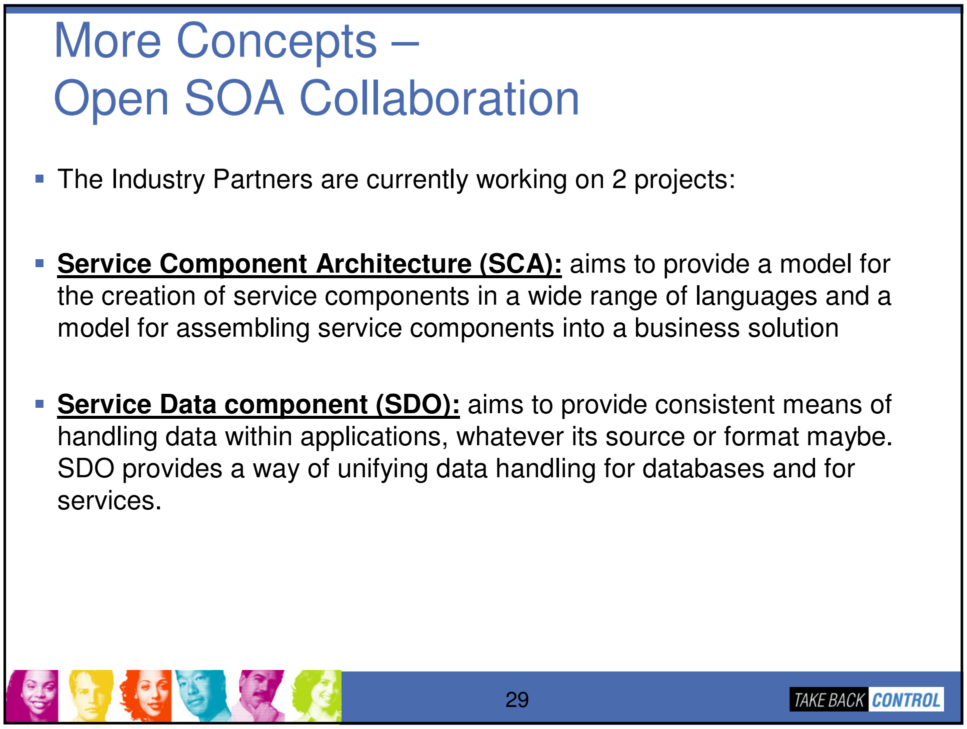
- – XML
- – Web Services (SOAP, WSDL, WS-Security, etc.)
- – the business process execution language (BPEL)
- – Java Enterprise Edition (J2EE) standards (JDBC, JCA, JMS, etc.)



More Concepts – Open SOA Collaboration

OSOA (Open SOA Collaboration):
An informal group of industry leaders that share a common interest: Defining a language neutral programming model that meets the needs of Enterprise developers who are developing software that exploits SOA characteristics and benefits.

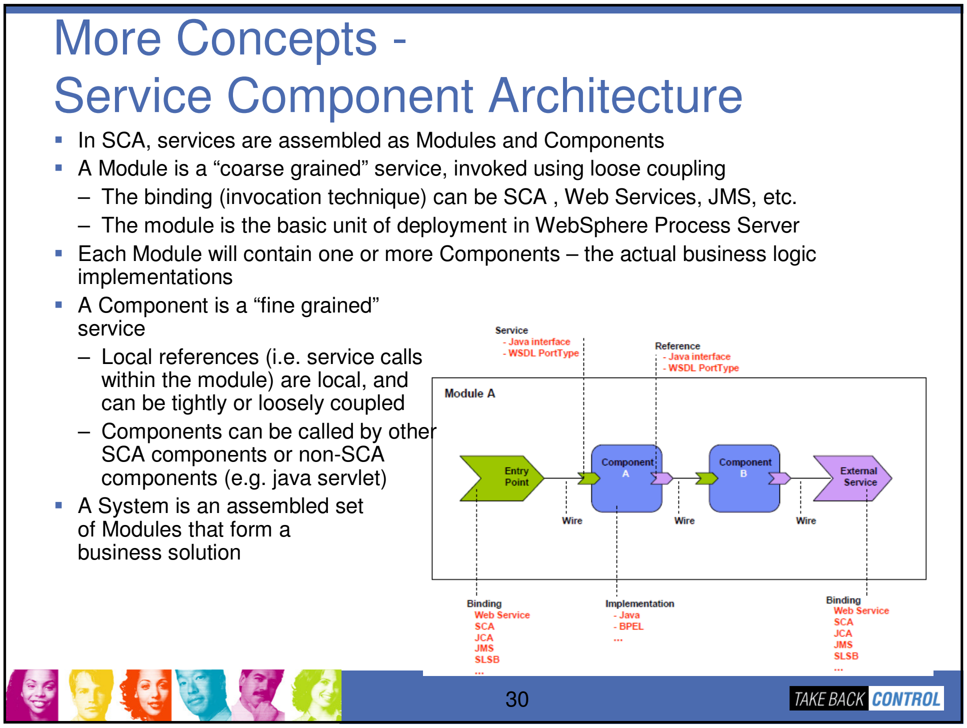
bea Think liquid.
CAPE CLEAR
IBM
INTERFACE21
IONA
ORACLE
PRIMETON · 普元
PROGRESS SOFTWARE
redhat.
ROGUE WAVE SOFTWARE
SAP
SIEMENS
software AG
Sun microsystems
SYBASE
TIBCO The Power of Now
Zend The PHP Company



More Concepts – Open SOA Collaboration

- The Industry Partners are currently working on 2 projects:
- **Service Component Architecture (SCA):** aims to provide a model for the creation of service components in a wide range of languages and a model for assembling service components into a business solution
- **Service Data component (SDO):** aims to provide consistent means of handling data within applications, whatever its source or format maybe. SDO provides a way of unifying data handling for databases and for services.

TAKE BACK CONTROL

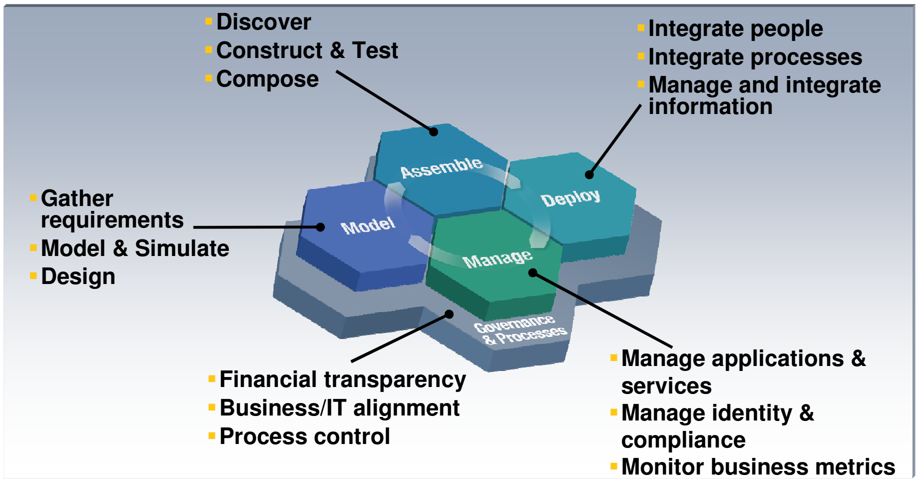


More Concepts - Service Component Architecture

- In SCA, services are assembled as Modules and Components
- A Module is a "coarse grained" service, invoked using loose coupling
  - The binding (invocation technique) can be SCA , Web Services, JMS, etc.
  - The module is the basic unit of deployment in WebSphere Process Server
- Each Module will contain one or more Components – the actual business logic implementations
- A Component is a "fine grained" service
  - Local references (i.e. service calls within the module) are local, and can be tightly or loosely coupled
  - Components can be called by other SCA components or non-SCA components (e.g. java servlet)
- A System is an assembled set of Modules that form a business solution

TAKE BACK CONTROL

SOA is an evolution



- Discover
- Construct & Test
- Compose

- Integrate people
- Integrate processes
- Manage and integrate information

- Gather requirements
- Model & Simulate
- Design

- Financial transparency
- Business/IT alignment
- Process control

- Manage applications & services
- Manage identity & compliance
- Monitor business metrics

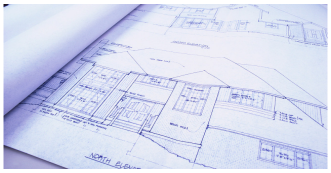
What SOA Isn’t

- A product that can be purchased
- The need to rewrite existing applications
- The need to overhaul IT infrastructure
- Web services (more about this later)
- The need to replace legacy systems

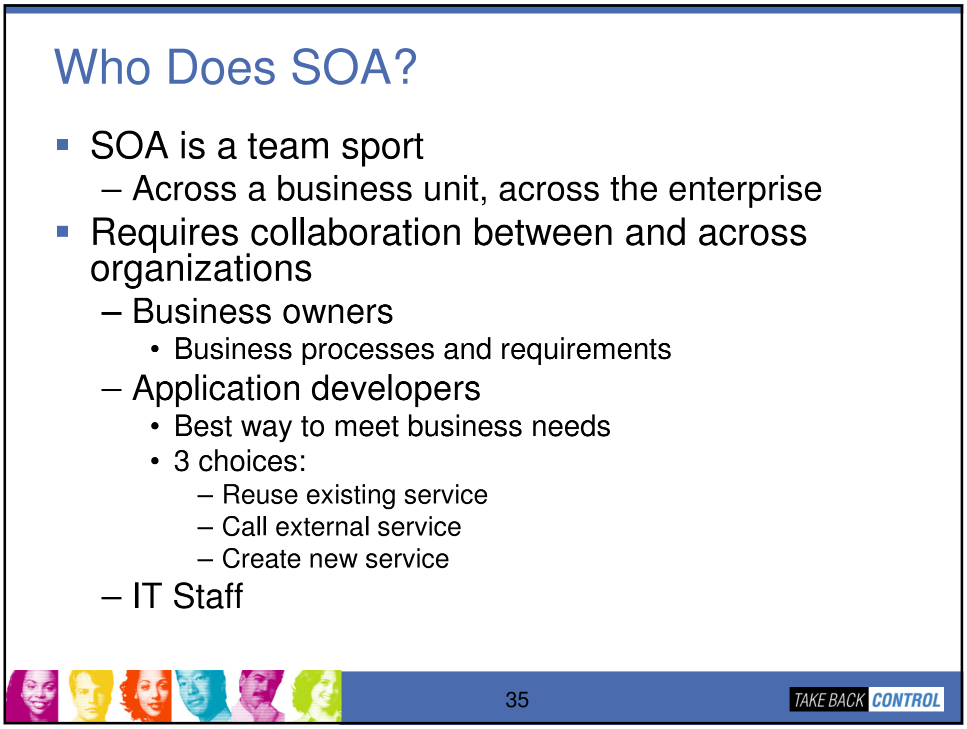
SOA

- SOA is an architecture but not a concrete architecture
- SOA is a style, paradigm, concept, philosophy
- It is a way of thinking, a value system that leads to certain concrete decisions when designing a concrete software architecture.
- There is no tool or recipe you can use use to make everything work.
- While applying this paradigm, you have to make decisions that are appropriate to your circumstances.

Benefits of a SOA

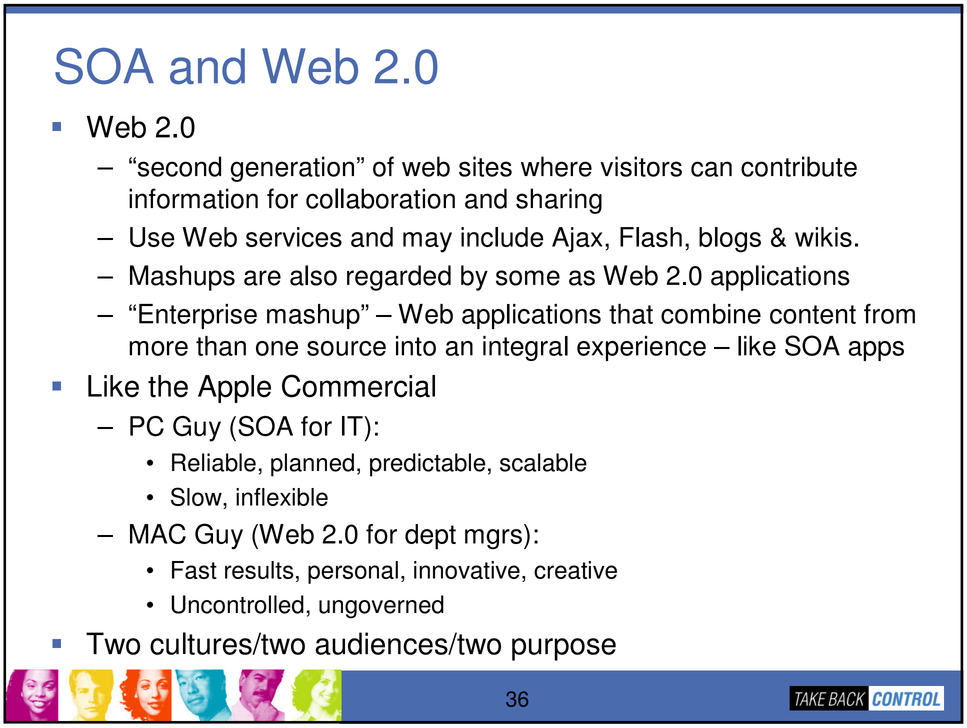


- Improved Adaptability and Agility
  - *Provide more function faster*
- Functional Reusability
  - *Reuse what you already have*
  - *Eliminate the need for large scale rip and replace*
- Independent Change Management
  - *Compose services from parts vs. writing new services*
  - *Focus on configuration rather than programming*
- Interoperability instead of point-to-point integration
  - Loosely-coupled framework, services in network
- Orchestrate rather than integrate
  - Configuration rather than development to deliver business needs
  - Easier to change



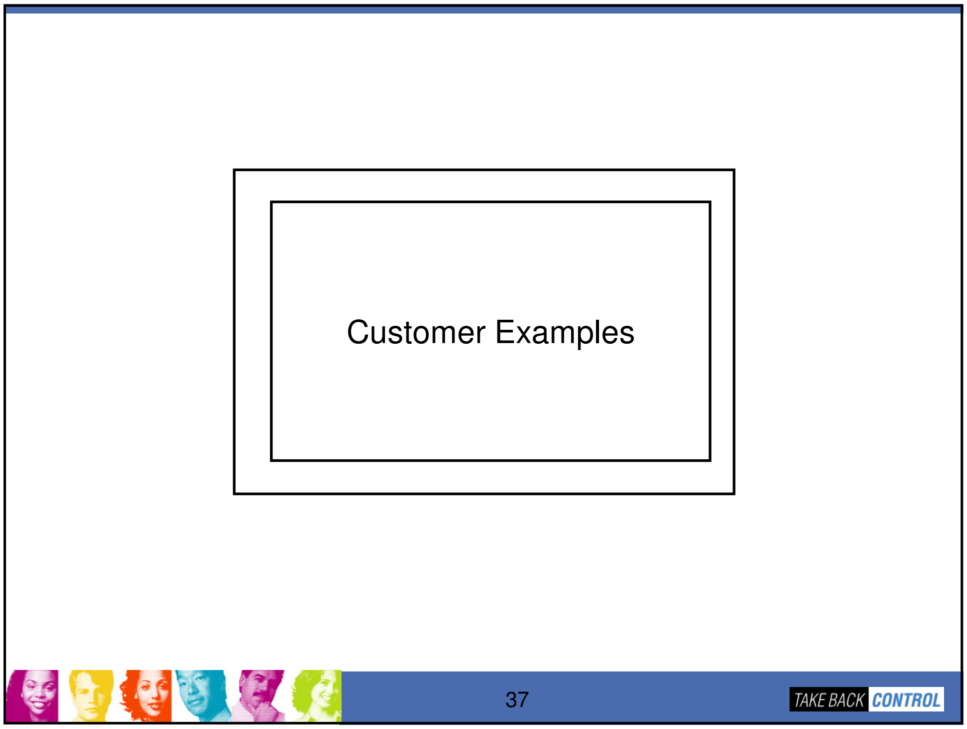
Who Does SOA?

- SOA is a team sport
  - Across a business unit, across the enterprise
- Requires collaboration between and across organizations
  - Business owners
    - Business processes and requirements
  - Application developers
    - Best way to meet business needs
    - 3 choices:
      - Reuse existing service
      - Call external service
      - Create new service
  - IT Staff

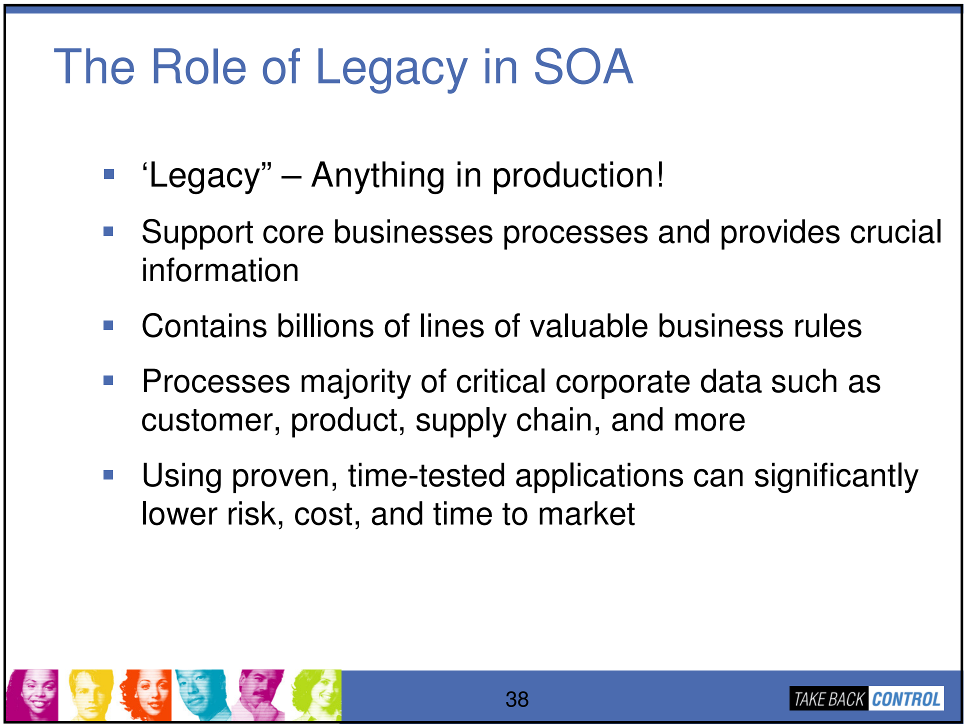


SOA and Web 2.0

- Web 2.0
  - “second generation” of web sites where visitors can contribute information for collaboration and sharing
  - Use Web services and may include Ajax, Flash, blogs & wikis.
  - Mashups are also regarded by some as Web 2.0 applications
  - “Enterprise mashup” – Web applications that combine content from more than one source into an integral experience – like SOA apps
- Like the Apple Commercial
  - PC Guy (SOA for IT):
    - Reliable, planned, predictable, scalable
    - Slow, inflexible
  - MAC Guy (Web 2.0 for dept mgrs):
    - Fast results, personal, innovative, creative
    - Uncontrolled, ungoverned
- Two cultures/two audiences/two purpose

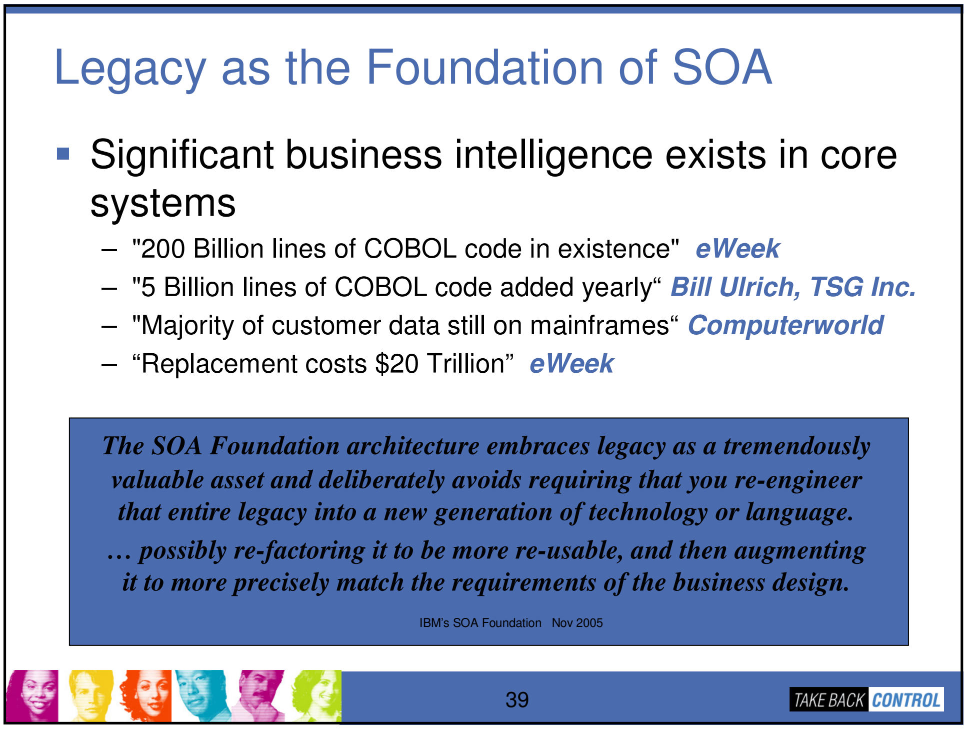


Customer Examples



The Role of Legacy in SOA

- ‘Legacy” – Anything in production!
- Support core businesses processes and provides crucial information
- Contains billions of lines of valuable business rules
- Processes majority of critical corporate data such as customer, product, supply chain, and more
- Using proven, time-tested applications can significantly lower risk, cost, and time to market

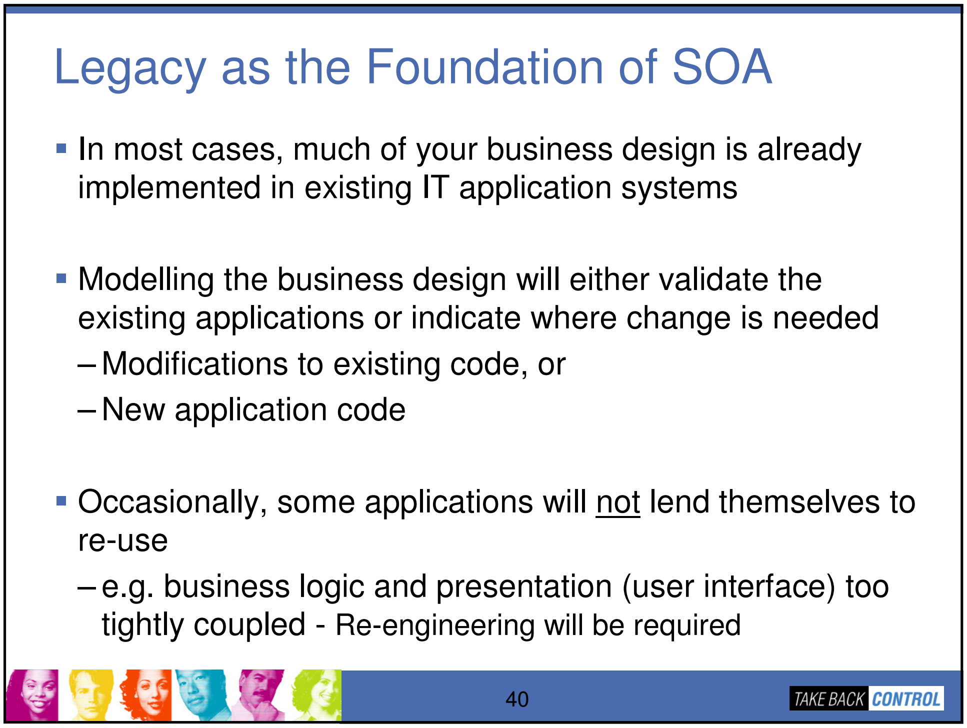


Legacy as the Foundation of SOA

- Significant business intelligence exists in core systems
  - "200 Billion lines of COBOL code in existence" ***eWeek***
  - "5 Billion lines of COBOL code added yearly“ ***Bill Ulrich, TSG Inc.***
  - "Majority of customer data still on mainframes“ ***Computerworld***
  - “Replacement costs $20 Trillion” ***eWeek***

> ***The SOA Foundation architecture embraces legacy as a tremendously valuable asset and deliberately avoids requiring that you re-engineer that entire legacy into a new generation of technology or language.***
>
> ***… possibly re-factoring it to be more re-usable, and then augmenting it to more precisely match the requirements of the business design.***
>
> IBM's SOA Foundation Nov 2005

TAKE BACK CONTROL



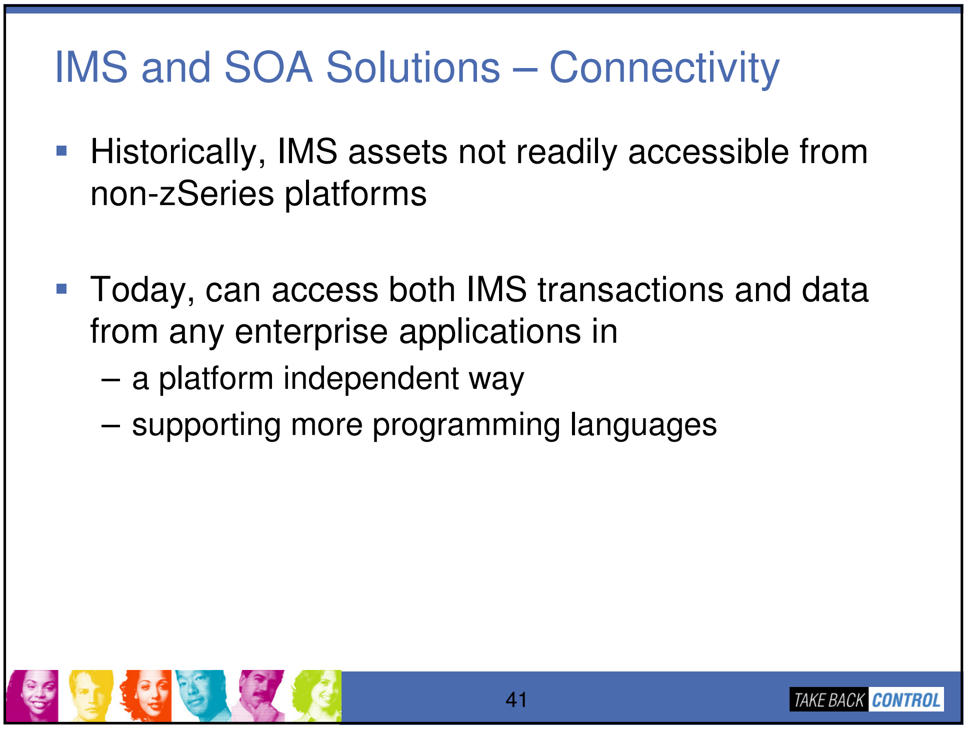
Legacy as the Foundation of SOA

- In most cases, much of your business design is already implemented in existing IT application systems

- Modelling the business design will either validate the existing applications or indicate where change is needed
  - Modifications to existing code, or
  - New application code

- Occasionally, some applications will <u>not</u> lend themselves to re-use
  - e.g. business logic and presentation (user interface) too tightly coupled - Re-engineering will be required

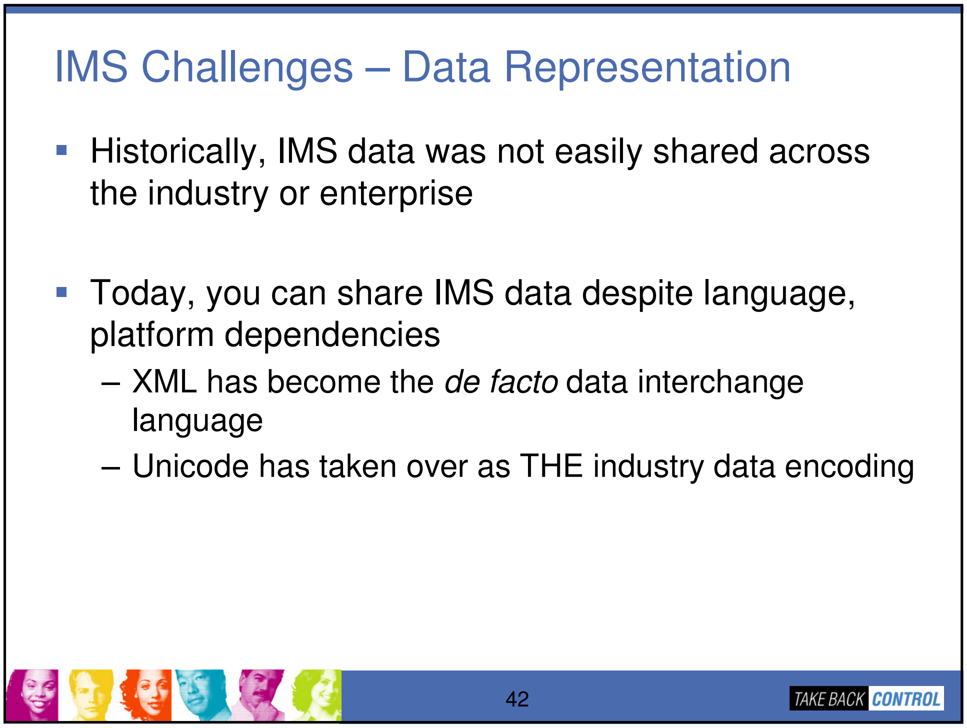
TAKE BACK CONTROL



IMS and SOA Solutions – Connectivity

- Historically, IMS assets not readily accessible from non-zSeries platforms
- Today, can access both IMS transactions and data from any enterprise applications in
  - a platform independent way
  - supporting more programming languages

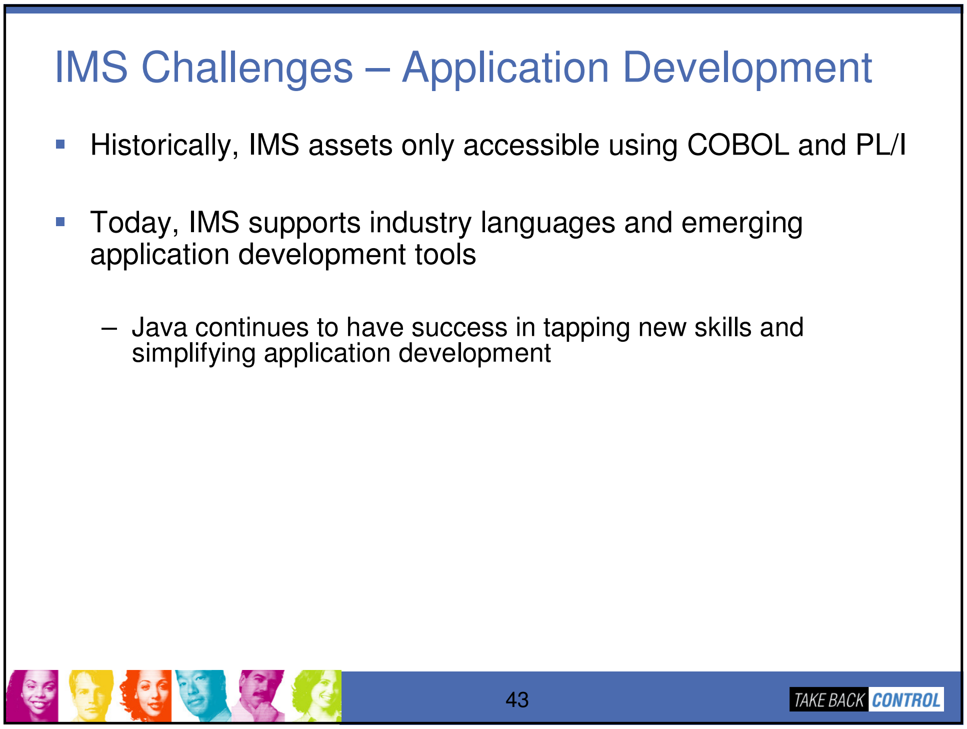
TAKE BACK CONTROL



IMS Challenges – Data Representation

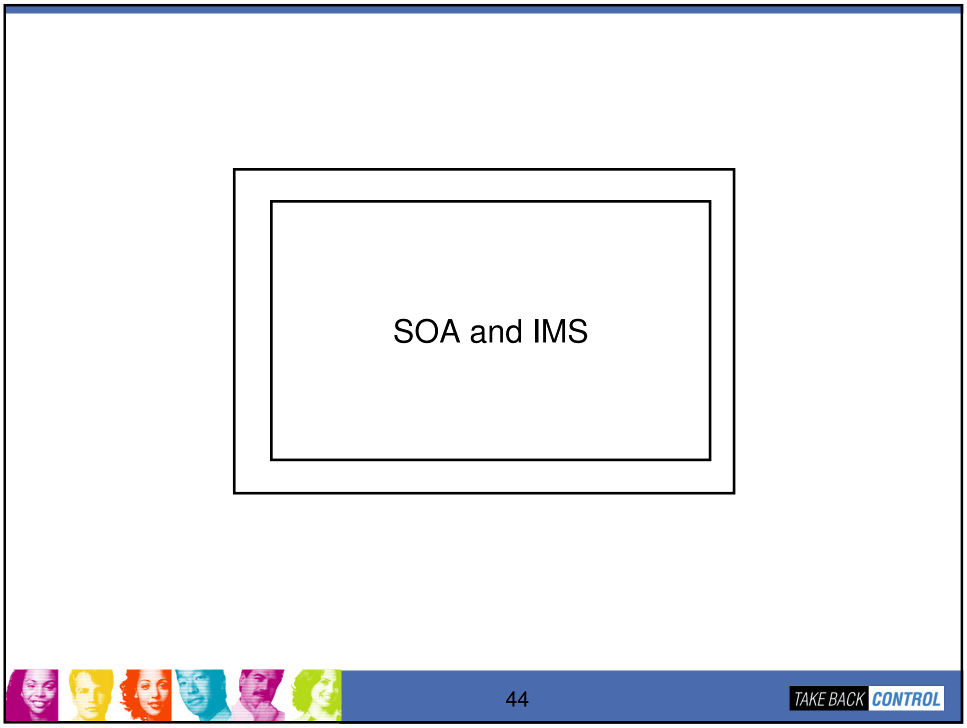
- Historically, IMS data was not easily shared across the industry or enterprise
- Today, you can share IMS data despite language, platform dependencies
  - XML has become the *de facto* data interchange language
  - Unicode has taken over as THE industry data encoding

TAKE BACK CONTROL

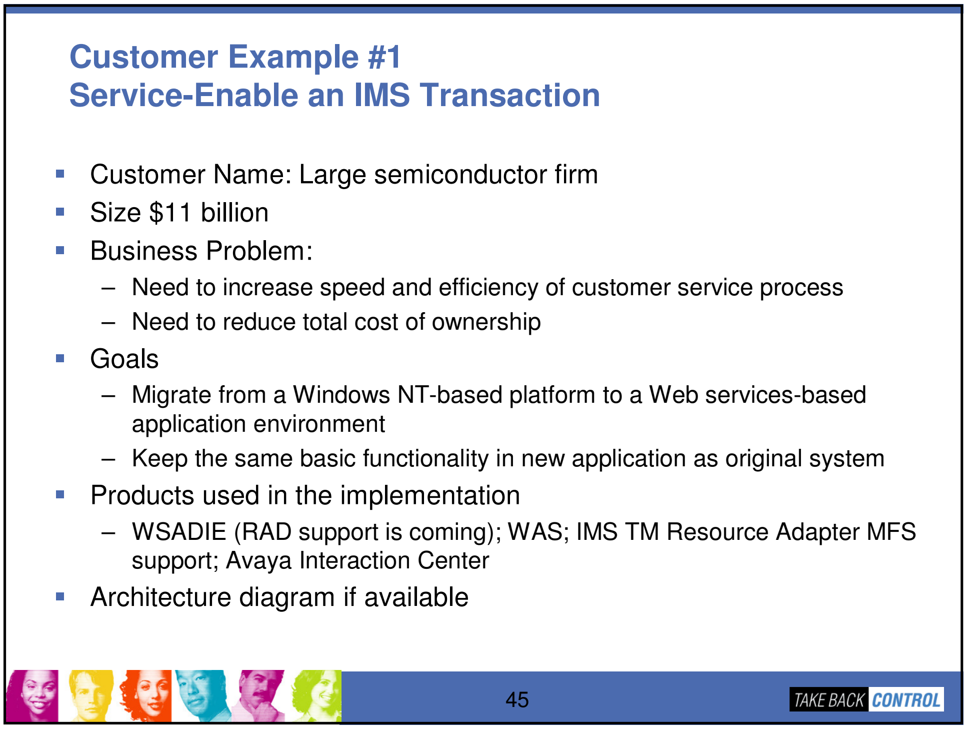


IMS Challenges – Application Development

- Historically, IMS assets only accessible using COBOL and PL/I
- Today, IMS supports industry languages and emerging application development tools
  - Java continues to have success in tapping new skills and simplifying application development

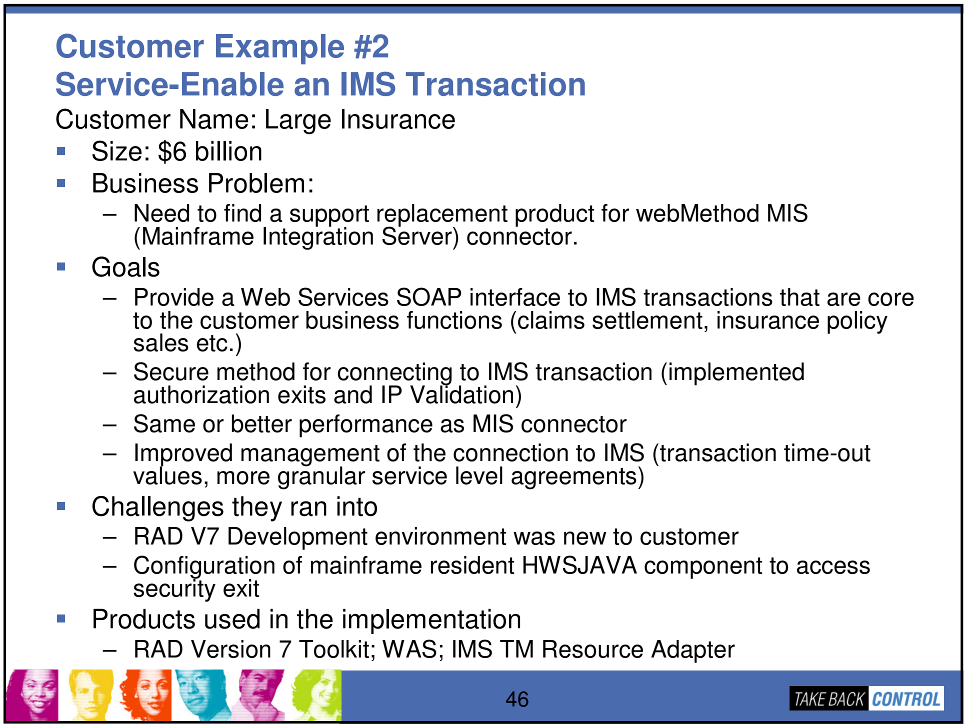


SOA and IMS



Customer Example #1
Service-Enable an IMS Transaction

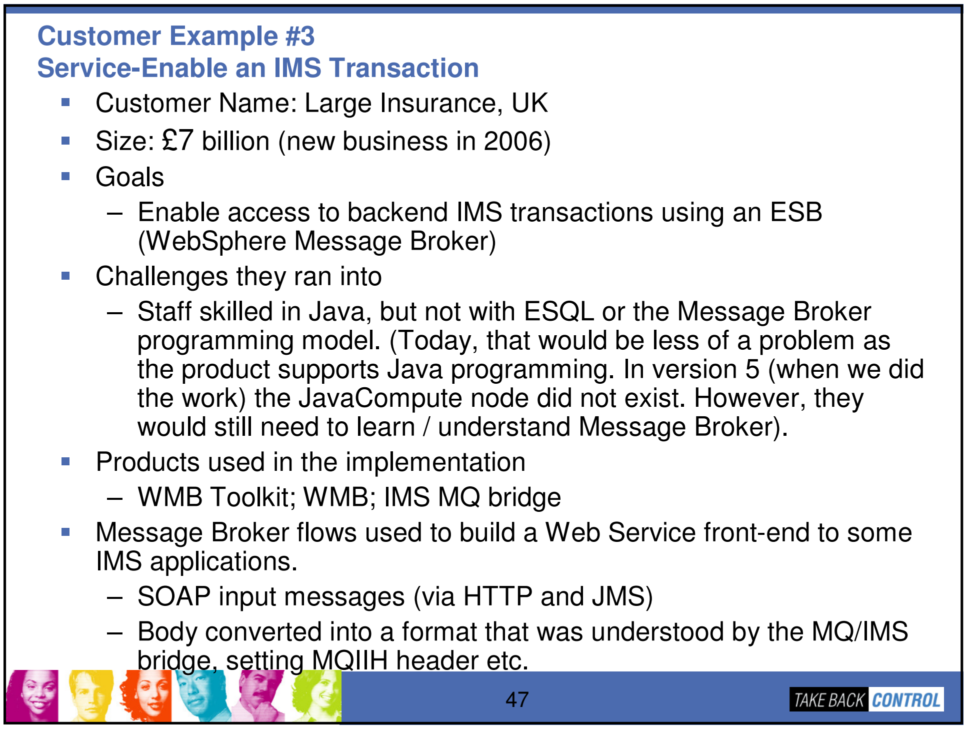
- Customer Name: Large semiconductor firm
- Size $11 billion
- Business Problem:
  - Need to increase speed and efficiency of customer service process
  - Need to reduce total cost of ownership
- Goals
  - Migrate from a Windows NT-based platform to a Web services-based application environment
  - Keep the same basic functionality in new application as original system
- Products used in the implementation
  - WSADIE (RAD support is coming); WAS; IMS TM Resource Adapter MFS support; Avaya Interaction Center
- Architecture diagram if available



Customer Example #2
Service-Enable an IMS Transaction

Customer Name: Large Insurance

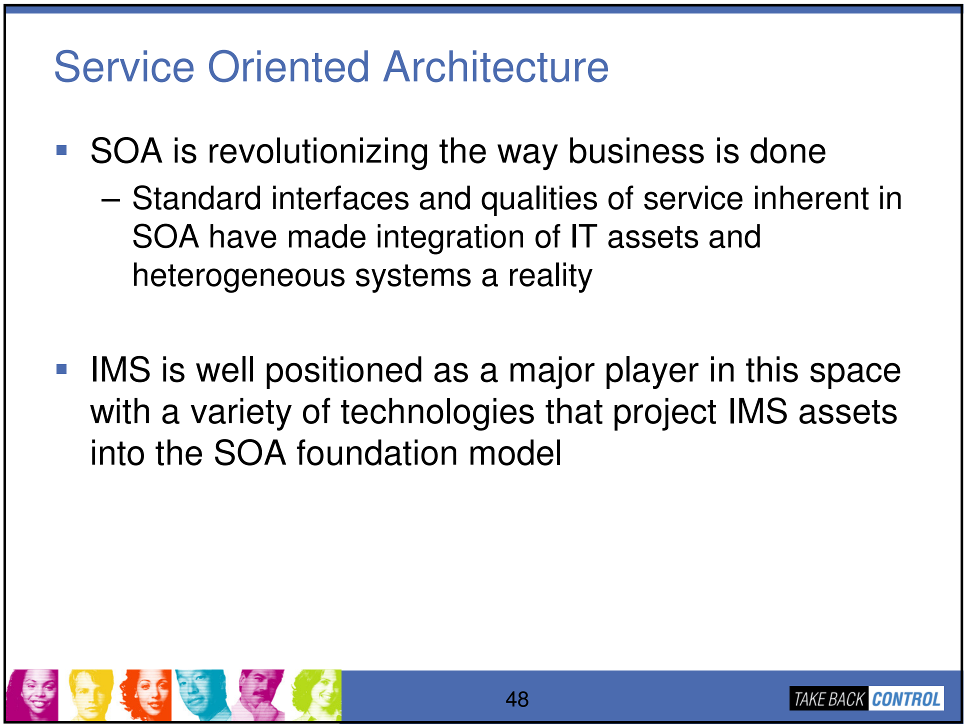
- Size: $6 billion
- Business Problem:
  - Need to find a support replacement product for webMethod MIS (Mainframe Integration Server) connector.
- Goals
  - Provide a Web Services SOAP interface to IMS transactions that are core to the customer business functions (claims settlement, insurance policy sales etc.)
  - Secure method for connecting to IMS transaction (implemented authorization exits and IP Validation)
  - Same or better performance as MIS connector
  - Improved management of the connection to IMS (transaction time-out values, more granular service level agreements)
- Challenges they ran into
  - RAD V7 Development environment was new to customer
  - Configuration of mainframe resident HWSJAVA component to access security exit
- Products used in the implementation
  - RAD Version 7 Toolkit; WAS; IMS TM Resource Adapter



Customer Example #3
Service-Enable an IMS Transaction

- Customer Name: Large Insurance, UK
- Size: £7 billion (new business in 2006)
- Goals
  - Enable access to backend IMS transactions using an ESB (WebSphere Message Broker)
- Challenges they ran into
  - Staff skilled in Java, but not with ESQL or the Message Broker programming model. (Today, that would be less of a problem as the product supports Java programming. In version 5 (when we did the work) the JavaCompute node did not exist. However, they would still need to learn / understand Message Broker).
- Products used in the implementation
  - WMB Toolkit; WMB; IMS MQ bridge
- Message Broker flows used to build a Web Service front-end to some IMS applications.
  - SOAP input messages (via HTTP and JMS)
  - Body converted into a format that was understood by the MQ/IMS bridge, setting MQIIH header etc.

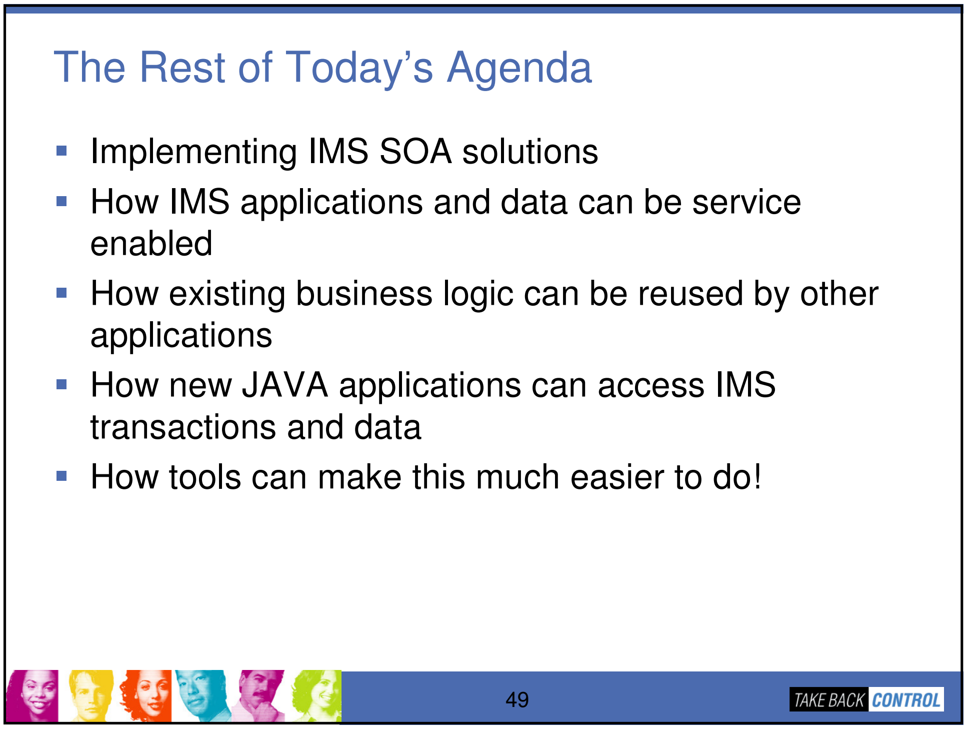
TAKE BACK CONTROL



Service Oriented Architecture

- SOA is revolutionizing the way business is done
  - Standard interfaces and qualities of service inherent in SOA have made integration of IT assets and heterogeneous systems a reality
- IMS is well positioned as a major player in this space with a variety of technologies that project IMS assets into the SOA foundation model

TAKE BACK CONTROL



The Rest of Today’s Agenda

- Implementing IMS SOA solutions
- How IMS applications and data can be service enabled
- How existing business logic can be reused by other applications
- How new JAVA applications can access IMS transactions and data
- How tools can make this much easier to do!