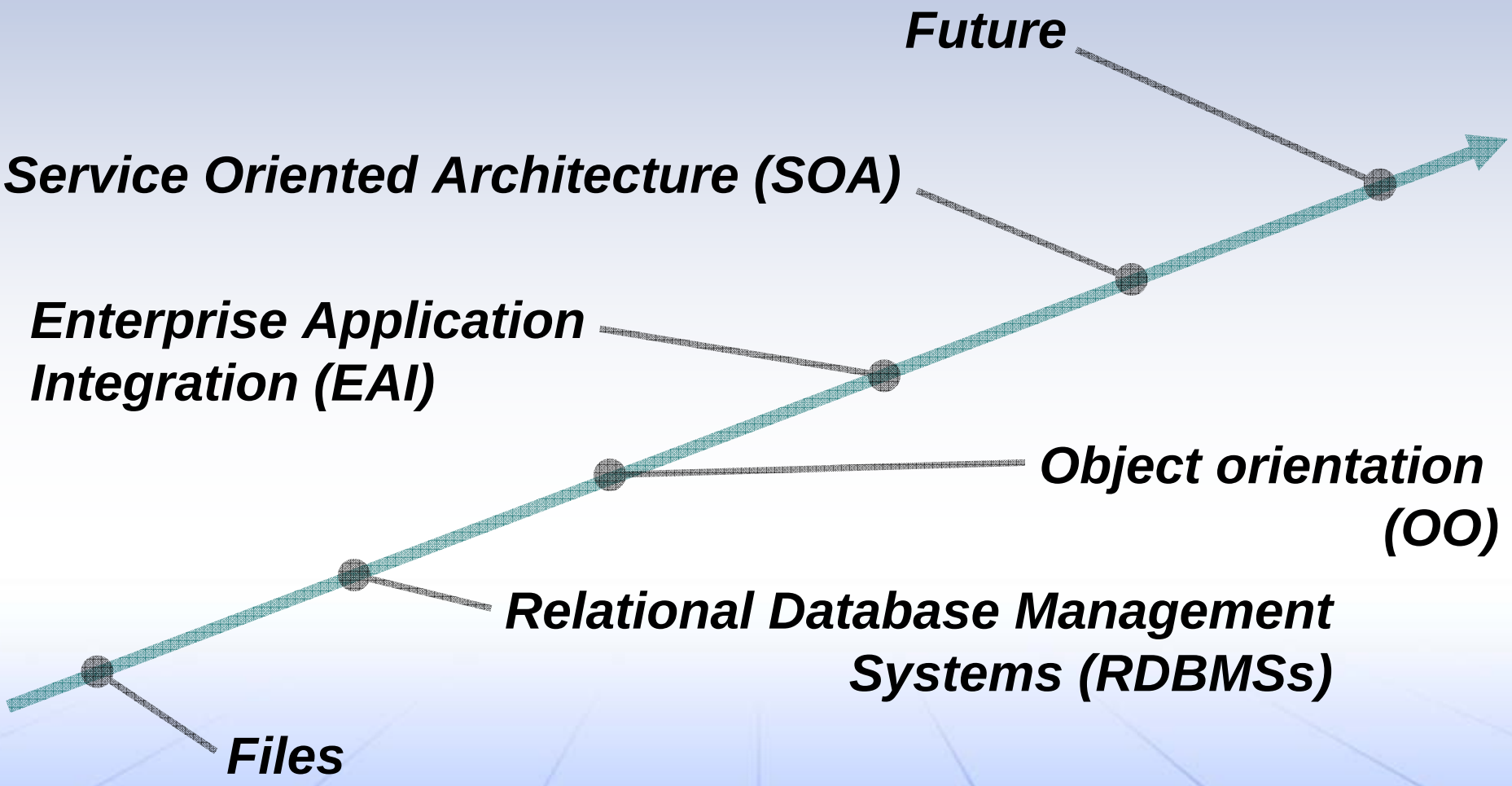
The Evolution of Information as a Service

Dan Wolfson
Distinguished Engineer
IBM Information Management Software

Outline

- ***The evolution of information management and the continuous debate on where 'behavior' should reside***
- ***Service Oriented Architecture with a focus on Information as a Service***
- ***Future directions***

The Evolution of Information Management and the Continuous Debate on Where 'Behavior' Should Reside



The Age of Files – Overview

- ***Characteristics***
  - Flat files persist data
  - Varying forms of structuring the content in the files (e.g. specific characters such as comma, specific length per field, etc.)
  - CRUD operations on file level
- ***Pros***
  - Simplicity
  - Can be very fast for specific use
- ***Cons***
  - Basically no separations of concerns
  - Every application program has redundant functionality to manage data
    - Validation by the application
    - Lack of data manipulation language (DML) / high costs to implement queries
    - Lack of data definition / modeling language (DDL)
    - Lack of transaction control
    - Security limited to operating system
    - Recovery limited to operating system
    - Logging only on the level of files
  - Difficulty of relating, analyzing, and integrating data
  - Scalability dependent on application program design
  - …

The Age of Files – Placement of Behavior

Any type of behavior

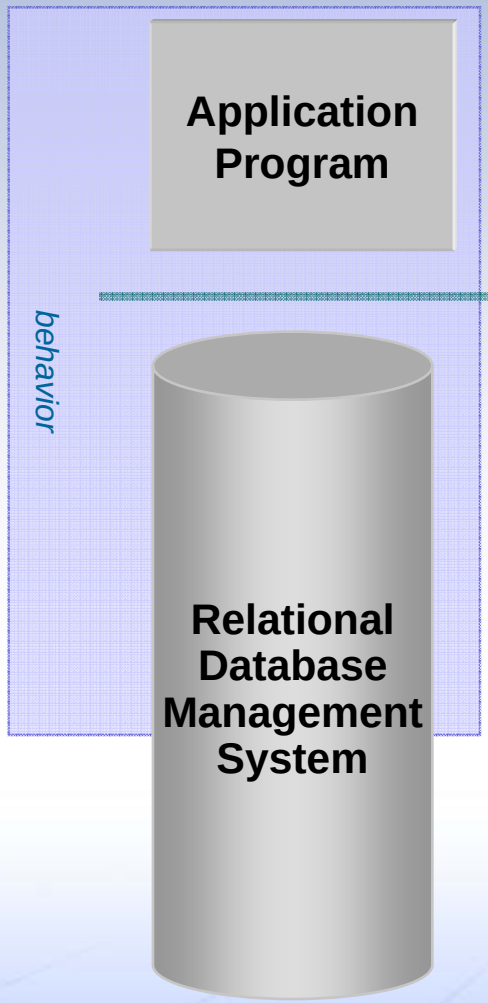
- Application code
- Data manipulation: simple CRUD and more advanced queries
- Implicit data definition / modeling
- Embedded transaction processing and therefore most often limited ACID support
- Embedded security
- …

- Just persistence

The Age of Relational (RDBMSs) – Overview

- ***Characteristics***
  - Data is managed by relational database management systems (RDBMSs)
  - Well-defined, standard-based DDL (SQL) to define the structure of the data
  - Declarative, standard-based DML (SQL) to manipulate & access data
- ***Pros***
  - Analysis and reporting support
  - Logical and physical database design independent of application (separations of concerns)
  - Extensive and powerful DDL
    - Increase in productivity and improvement of common validation and sharing of data
    - Stored procedures introduces operations close to the data (performance, integrity)
    - Well-defined and explicit semantics
  - Declarative DML that allows for query optimization by the RDBMS
  - Centralization of standard data management functionality in one place, i.e. the RDBMS, instead of redundant and inconsistent in every application program
    - Transaction control (ACID), Security, Recovery, Logging
    - Management and scale
- ***Cons***
  - Different programming model

The Age of RDBMSs – Placement of Behavior



- Application code
- Invocation of (complex) queries and stored procedures

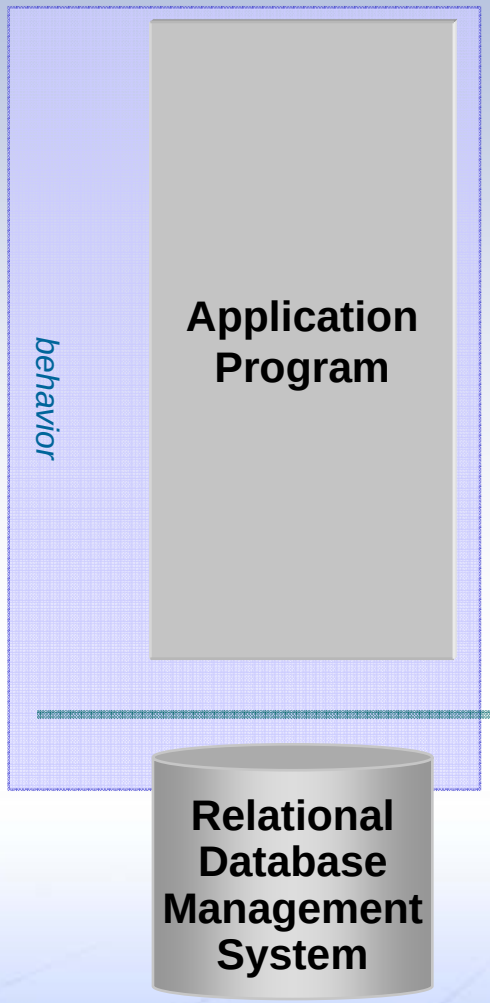
Significant portion of behavior (shift)

- Traditional application-supported behavior
  - Stored procedures
  - Triggers
  - Queries, simple and complex (CRUD)
- Data modeling (DDL)
- Data management functionality
  - Transaction support
  - Concurrency support
  - Persistence

The Age of Object Orientation (OO) – Overview

- ***Characteristics***
  - Powerful OO modeling capabilities
    - Encapsulation introduces the separations of concerns in the application layer
    - Data access encapsulated by dedicated “objects” (EJBs, etc.)
    - Adds behavior to structured types
  - Beginning of the decomposing monolithic applications into modular components with well-defined interfaces
  - Data is still managed primarily by RDBMS
- ***Pros***
  - Well-defined application programming interfaces (APIs) which encapsulate logic behind the scenes
  - Significant improvements to the design and structure of application programs
- ***Cons***
  - Complex and limited adoption of interoperability (CORBA etc.)
  - Mismatch between relational database structure and OO application design
  - Lack of an architecture, methodology, governance to manage the interfaces across an enterprise architecture
  - Sharing of object definitions separate from the RDBMS instance

The Age of OO – Placement of Behavior



Shift back of placing behavior in the application

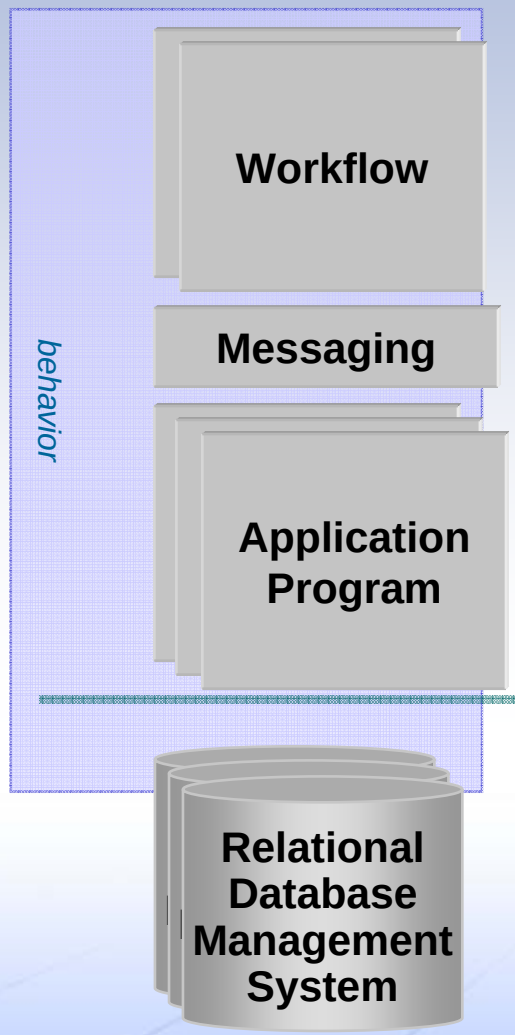
- Object-oriented application code
- Data access strictly encapsulated
- Top-down modeling approach where the object-oriented design dictates database design ("just an object-relational mapping exercise and some tuning to make the database perform")
- Functionality previously implemented in stored procedures, triggers, queries moved (to some extent) back to application (and continuous debate)

- Standard data management functionality
  - Transaction support
  - Concurrency support
  - Persistence

The Age of Enterprise Application Integration (EAI) – Overview

- ***Characteristics***
  - Introduction of an approach to achieve interoperability between applications (application programs) through a messaging infrastructure
  - Mediation and brokering of messages through broad array of styles: direct, broadcast, pub/sub, etc.
  - Typical interface towards a backend/source is its API
- ***Pros***
  - Common infrastructure (hub-and-spoke) replacing proprietary point-to-point connections
  - Extensible, scalable, reliable messaging between heterogeneous participants (i.e. applications)
- ***Cons***
  - Lack of a common standard to define interfaces independent of the programming language (underlying application)
  - Lack of an architecture, methodology, governance to manage the interfaces across an enterprise architecture
  - Limited capabilities for data integration, in particular bulk and requiring complex transformations (addressed by ETL = extract-transform-load)

The Age of EAI – Placement of Behavior



Placing of behavior unchanged

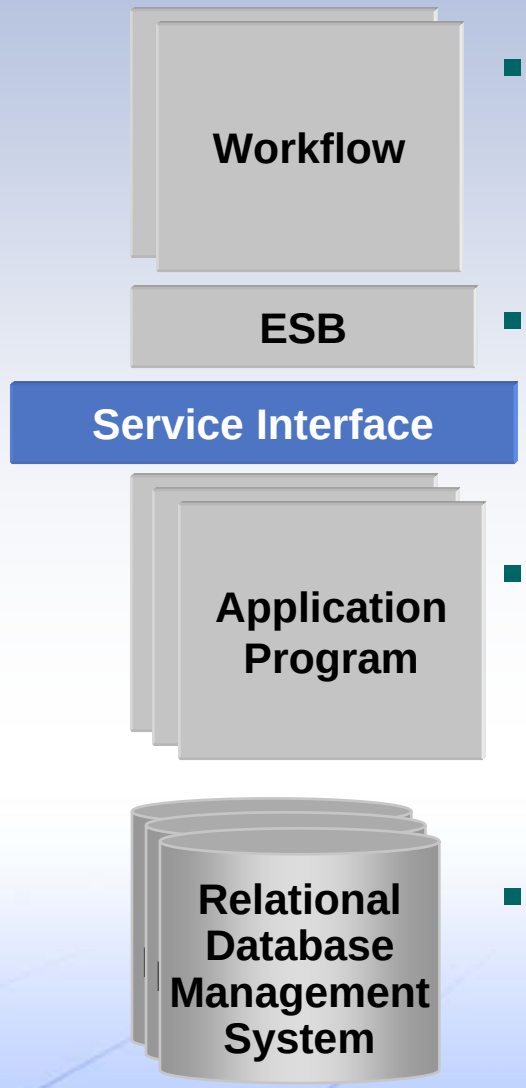
- Separation of workflow (now more formally specified) and application code
- Messaging to support application interoperability
- Increasing importance of the API to become the standard interface to access function and data; data access strictly encapsulated

- Standard data management functionality
  - Transaction support
  - Concurrency support
  - Persistence

The Age of Service Oriented Architecture (SOA) – Overview

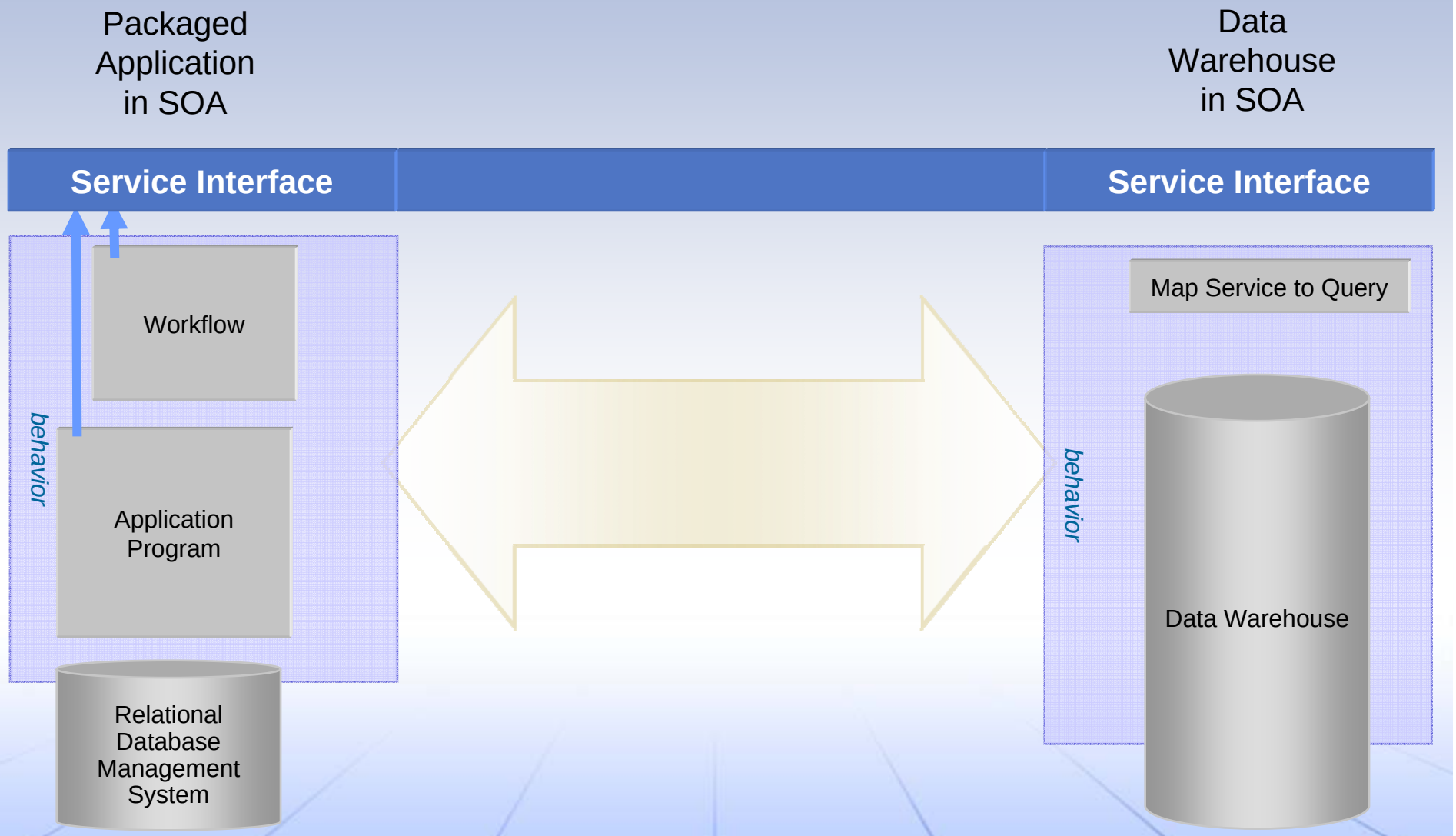
- ***Characteristics***
  - Standard-based definition of a service as the interface between a consumer (requestor) and provider
  - Services (and metadata) can be registered in a registry
- ***Pros***
  - Decoupling of service provider and consumer through well-defined service interface
  - Common standard to define interfaces (agnostic of programming language, protocol/binding, etc.)
  - Holistic architecture, methodology, and governance to manage the interfaces across an enterprise architecture
  - Metadata management and registry (dynamic binding of service consumers to providers)
  - Modular architecture allowing for extensibility of services and supporting functionality (management, security, etc.)
- ***Cons***
  - It doesn't solve everything ☺

The Age of the Service Oriented Architecture



- ***What should be exposed at the service interface?***
- ***Some consumers need access to information such as retrieveCustomerDetail***
- ***Should information access still be strictly guarded and encapsulated within the application layer?***
- ***The placement of behavior is up for debate again***

Placement of Behavior in SOA – A Combination of Complementary Approaches



Placement of Behavior in SOA – Overview of Layers

- ***Business Process Layer***
  - Implements the business processes (workflows)
- ***Service Layer***
  - Provides the formal contracts between the consumer and the provider.
  - Provides the interface for the service component to the consumer
  - Maps the consumer to the service implementation
- ***Service Component Layer***
  - Provides the implementation for services.
  - Many-to-many relationship between service interfaces and service components
- ***Application Service Layer***
  - Provides the application logic
- ***Information Service Layer***
  - Provides access to information
  - Exposes capabilities from information management through service interface

Placement of Behavior in SOA – Architecture Layers and Corresponding Behavior

- ***Business Process Layer***
  - Should be limited to workflow design and implementation, defined by the business analyst
  - Business process should have no visibility or understanding of the underlying operational systems
- ***Service Layer***
  - Should be limited to discovery and allocation of the service request to the appropriate service component(s)
  - Mostly technical rules such as appending access control tokens, logging, error handling, and making routing and binding decisions
- ***Service Component Layer***
  - Predominantly the rules needed to map a service requests to a (sequence of call(s) to operational systems
  - Includes the technical mapping, binding rules, and dependency rules for combining underlying operational capabilities into understandable service interfaces
- ***Application Service Layer***
  - Includes all the business and technical rules already encapsulated into the existing application.
  - Behavior in this layer is constrained by the application context boundaries
- ***Information Service Layer***
  - What should belong in this layer?

Evolution of Information Management

From "just" database management to Information on Demand that includes

- ***Database Management***

  Management of relational data unified with XML data, etc.

- ***Content Management***

  Management of diverse content types, content-centric processes, information capture and classification, content federation, content archiving, records management, etc.

- ***Information Integration***

  Understanding your data (glossary, canonical data modeling, information analysis), data cleansing, data transformation and aggregation (federation and ETL), etc.

- ***Master Data Management***

  Multi-form master data management (operational, collaborative, analytical), management of lifecycle, master data quality, events, hierarchy & relationships, authoring, etc.

- ***Analytics***

  Traditional analytics such as data mining, holistic analytics for structured and unstructured information, real-time analytics, embedded analytics, etc.

- ***Metadata Management***

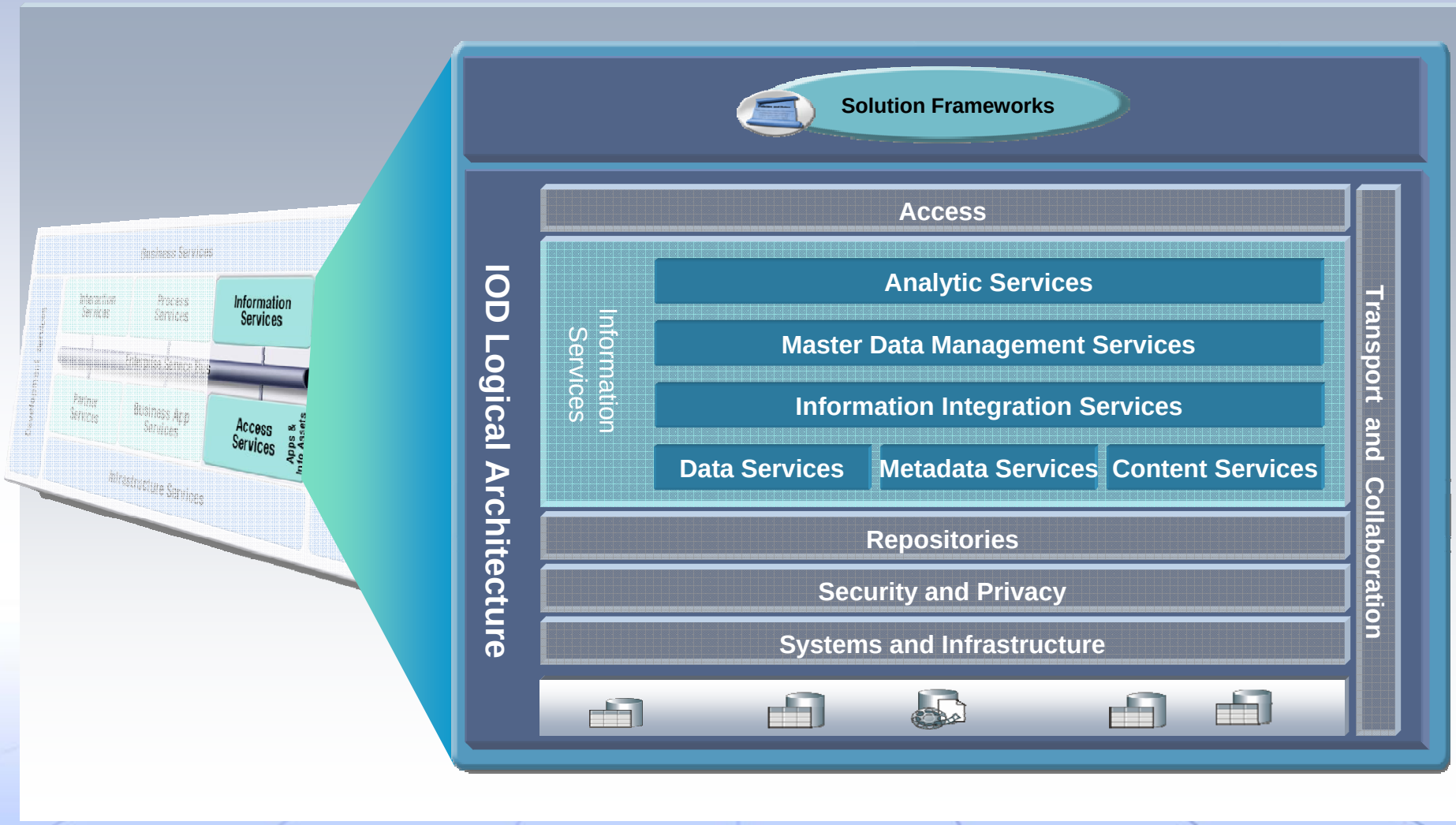
  CRUD operations for metadata that ensure consistency, relating metadata artifacts across disciplines (e.g. data models, glossary, data integration mappings), advanced functionality such as lineage, etc.

- ***Data Governance***

  Managing change, assigning stewardship, identifying and protecting trusted sources, etc.

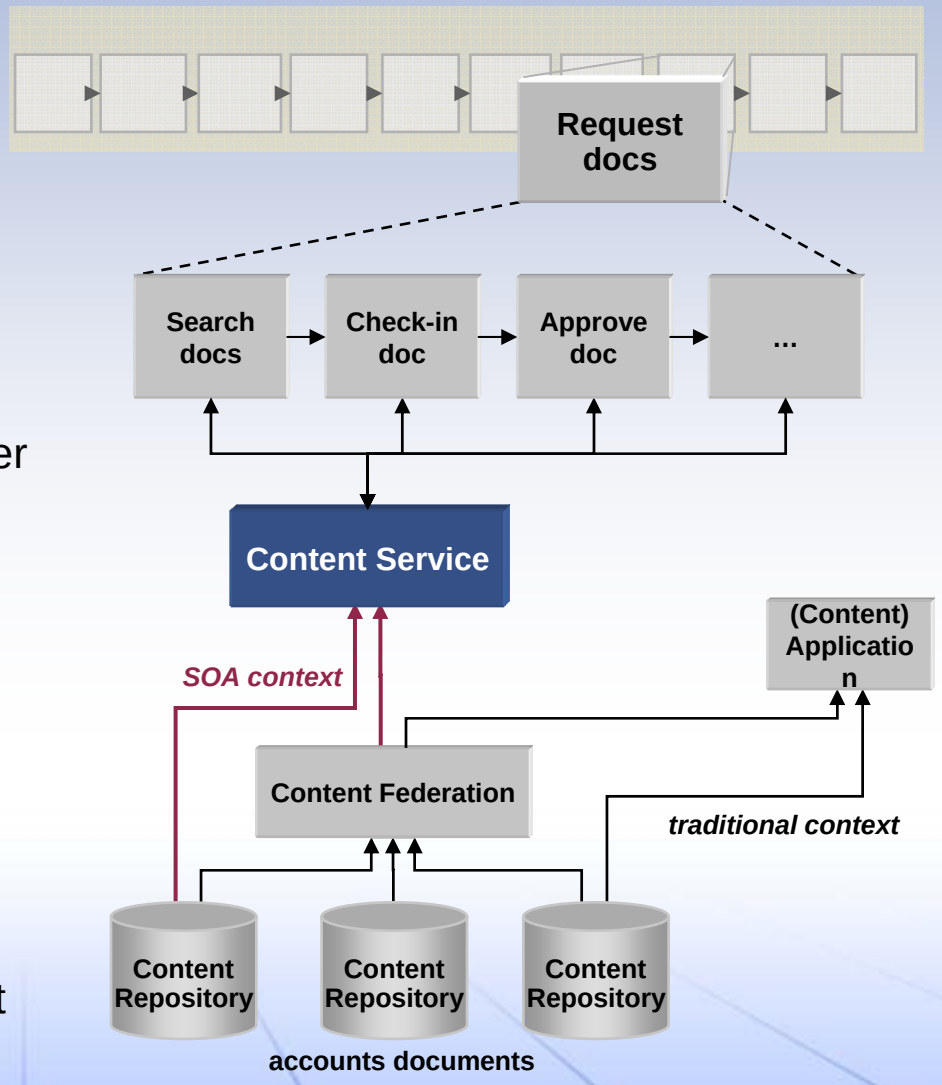
- ***…***

Components of Information as a Service – Information Services from Information on Demand



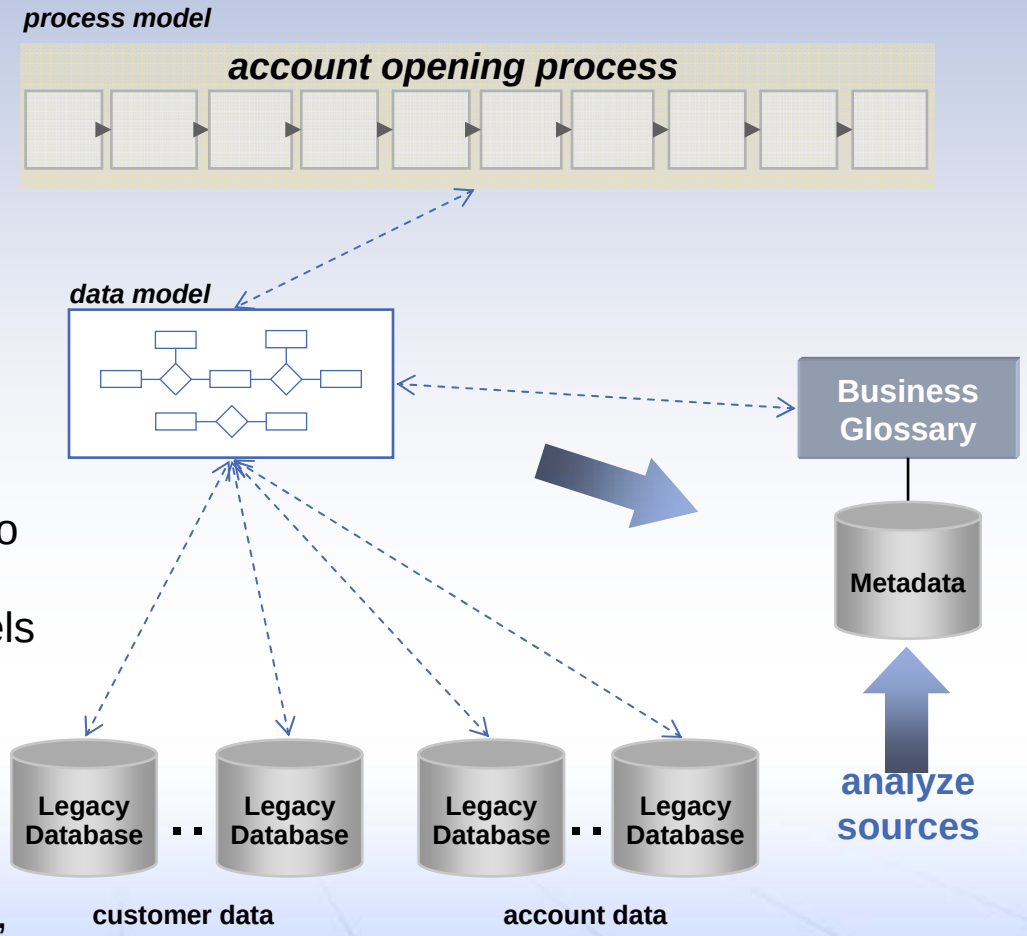
Content Services

- **As Is Environment**
  - Partially paper-based processes
  - Inefficient processes to manage documents
  - Disparate content repositories, point-to-point implementations to access content
- **Solution Characteristics**
  - Decouple content consumer from provider through reusable services, leveraging:
    - *Content management functionality*
    - *Content federation from disparate repositories into common virtual view*
  - Optimized content-centric processes
- **Results**
  - Optimized process to manage content
  - Content-centric process aligned with overall business process
  - Access content through single consistent content service interface



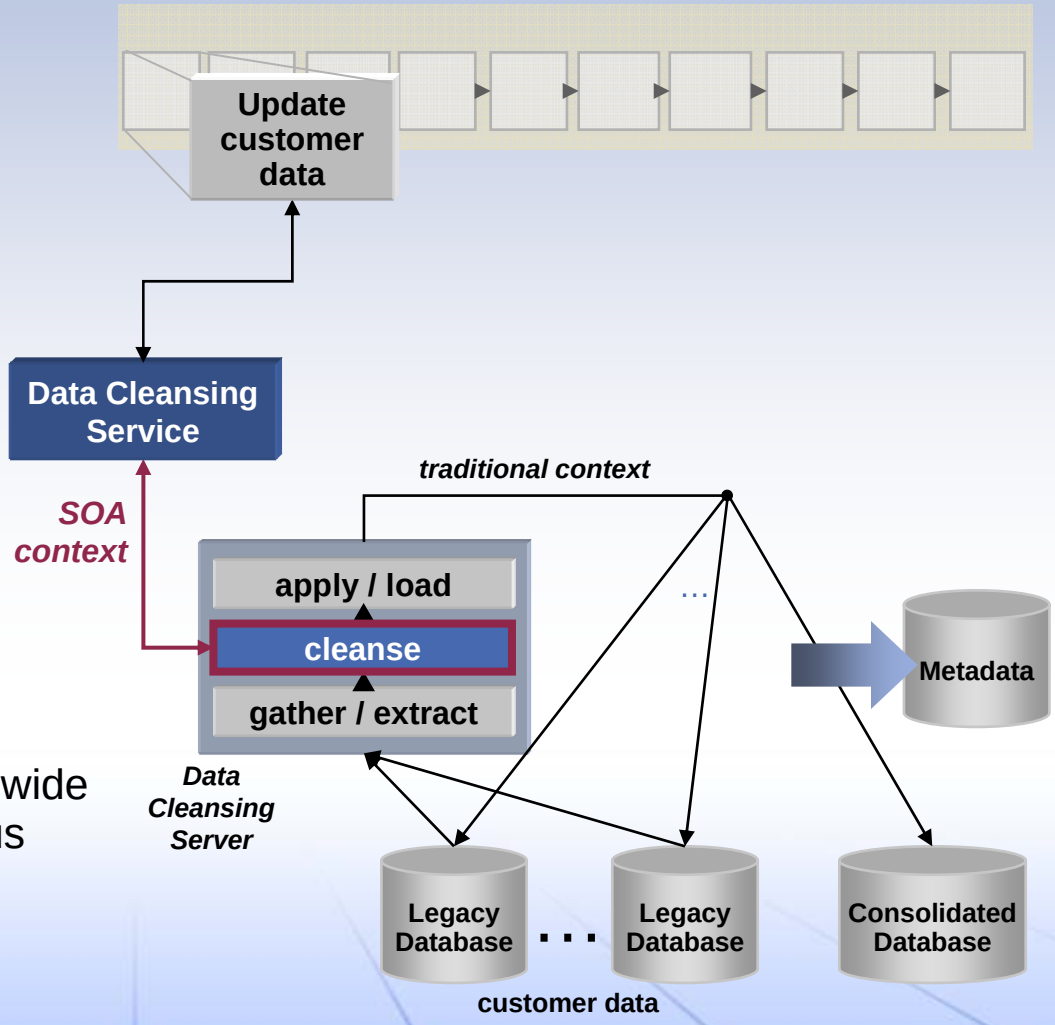
Information Integration Services – Design

- **As-Is Environment**
  - Unclear business terms; inconsistency between business and technical terms
  - Uncertainty of data quality
  - Inconsistency between process models and data models
- **Solution Characteristics**
  - Common business glossary
  - Assess and analyze data sources to determine degree of data quality
  - Align data, process & service models
- **Results**
  - Common definition of key terms
  - Discover structural inconsistencies and anomalies in data formats
  - Consistent data format across data, service and process layers



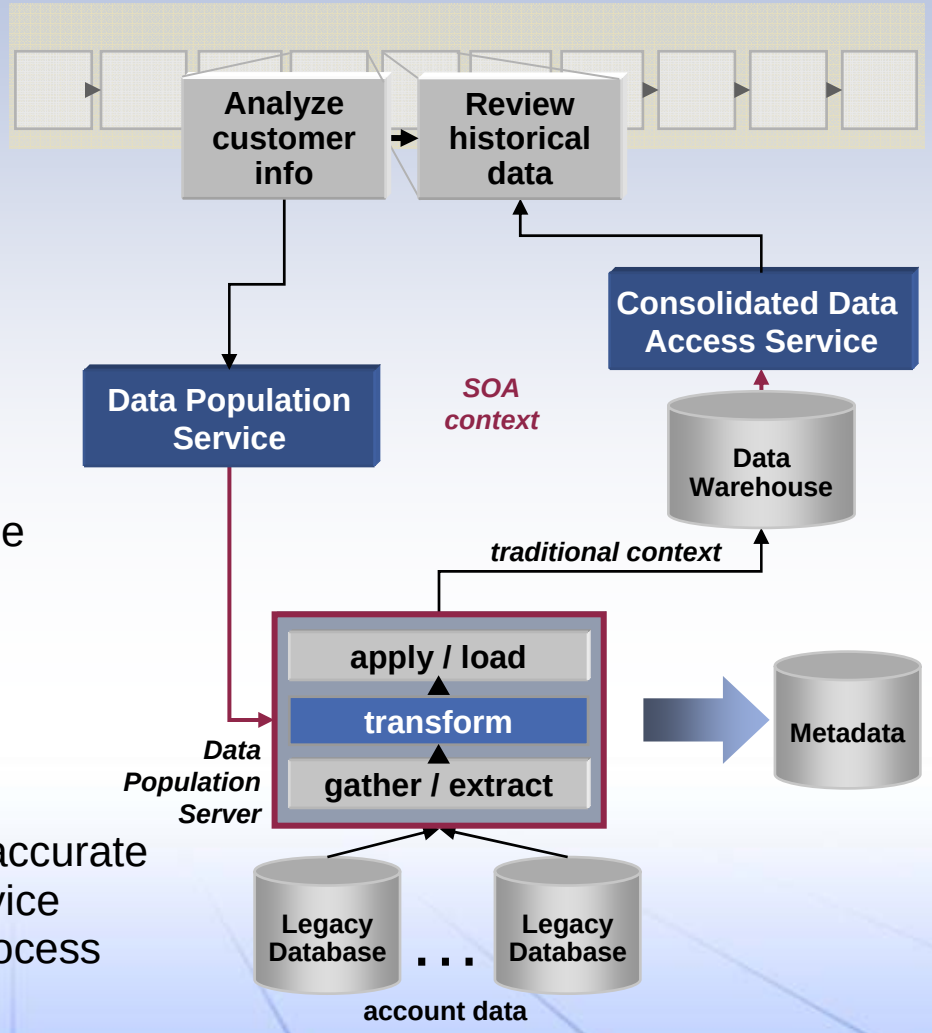
Information Integration Services – Cleansing

- **As-Is Environment**
  - Lack of data standards
  - Data inconsistencies
  - Data redundancies
- **Solution Characteristics**
  - Standardization of data formats
  - Data enrichment
  - Duplicate identification & removal
  - Apply same data cleansing rules against persisted data and expose as services
- **Results**
  - Single & consistent definition of cleansing rules, reused enterprise wide
  - Apply against data stored in various customer databases
  - Apply during data entry



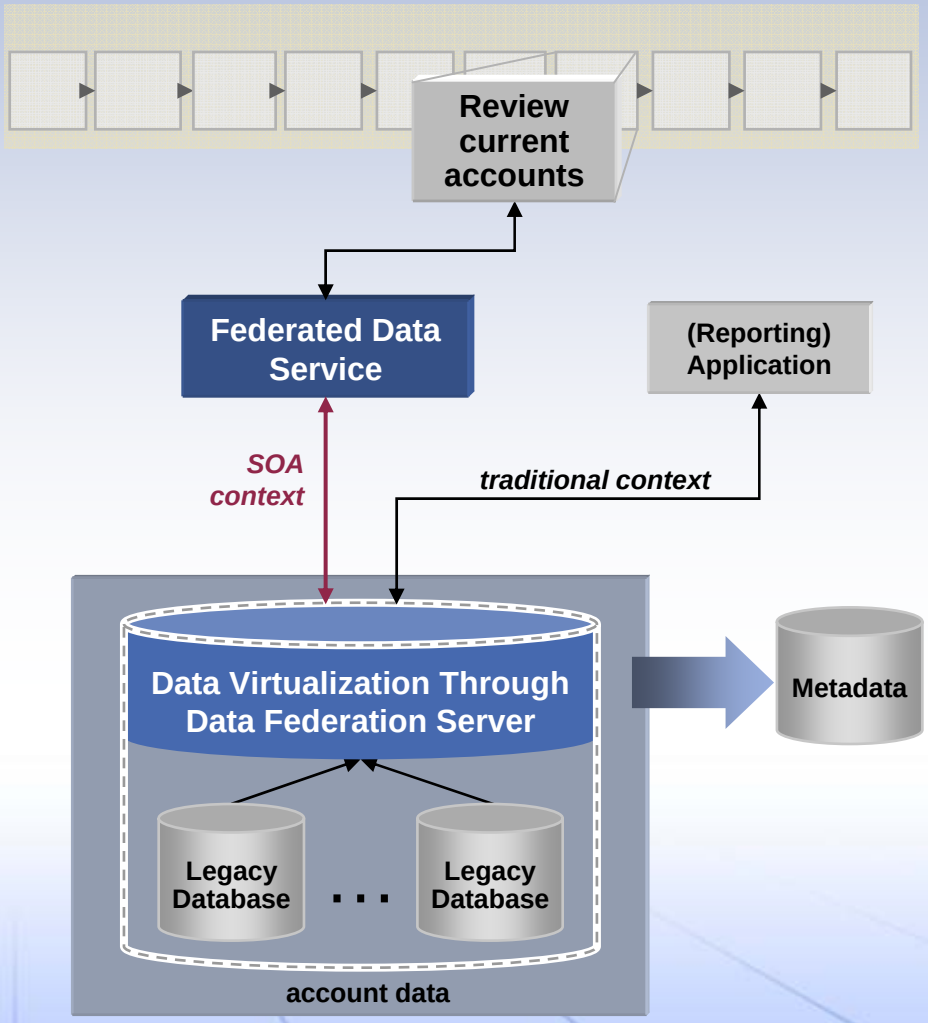
Information Integration Services – Consolidation

- **As-Is Environment**
  - Data resides in disparate sources
  - Manual & redundant integration of data by multiple consumers results in high costs and inconsistent/inaccurate data
  - Slow response time due to large data volume and complex transformations
- **Solution Characteristics**
  - Apply transformations on extracted source data; copy into consolidated target and expose consolidated data as services
  - Invoke population from business process
- **Results**
  - Multiple consumers can access trusted, accurate and integrated information through a service
  - Data availability aligned with business process



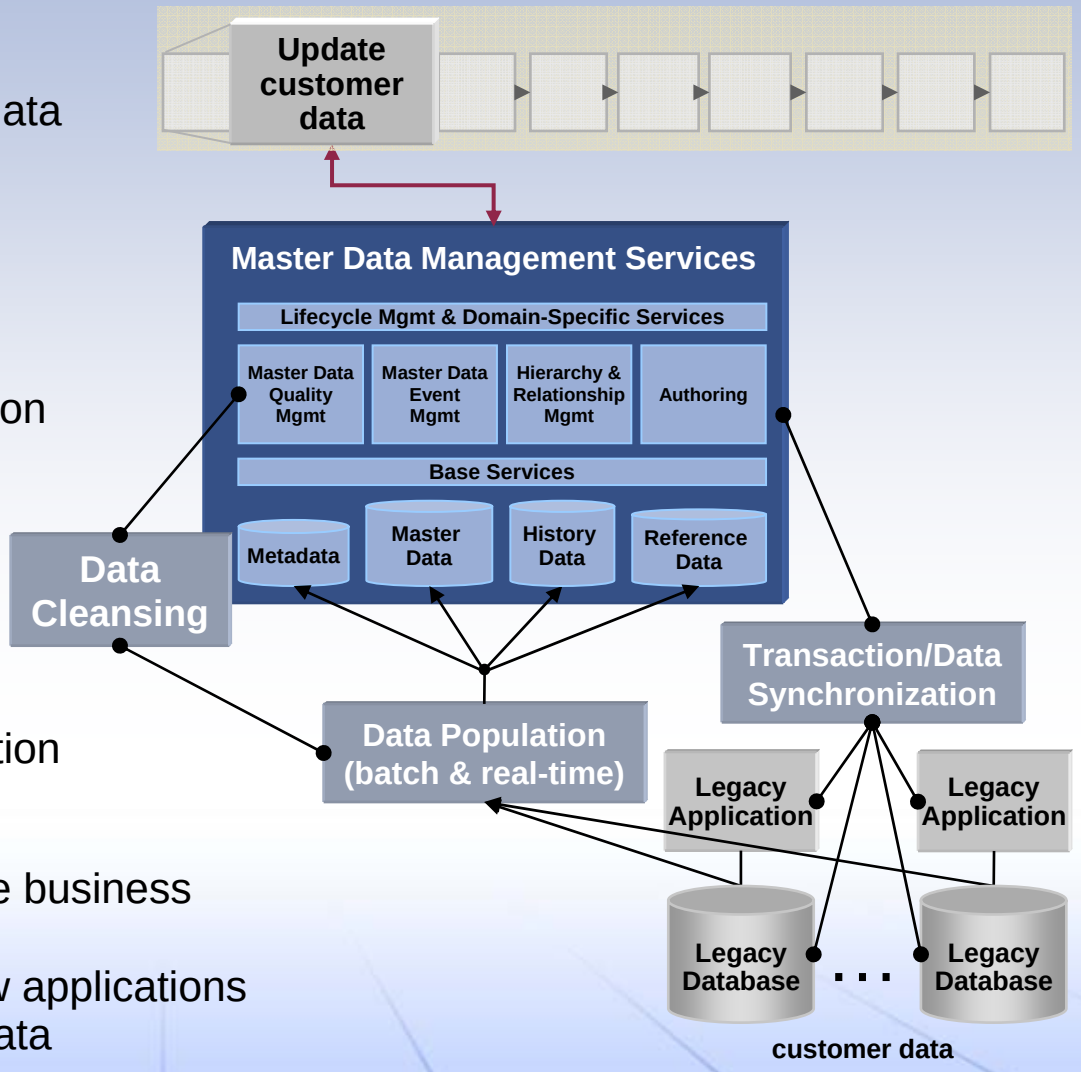
Information Integration Services – Federation

- **As-Is Environment**
  - Data resides in disparate sources
  - Manual & redundant integration of data by multiple consumers results in high costs and inconsistent/inaccurate data
  - Slow response time due to inefficient real-time access
- **Solution Characteristics**
  - On demand integration instead of redundant data
  - Transparent & optimized access to distributed, heterogeneous sources
- **Results**
  - Real-time access to distributed information, fast response time
  - Scalable approach for adding more data sources



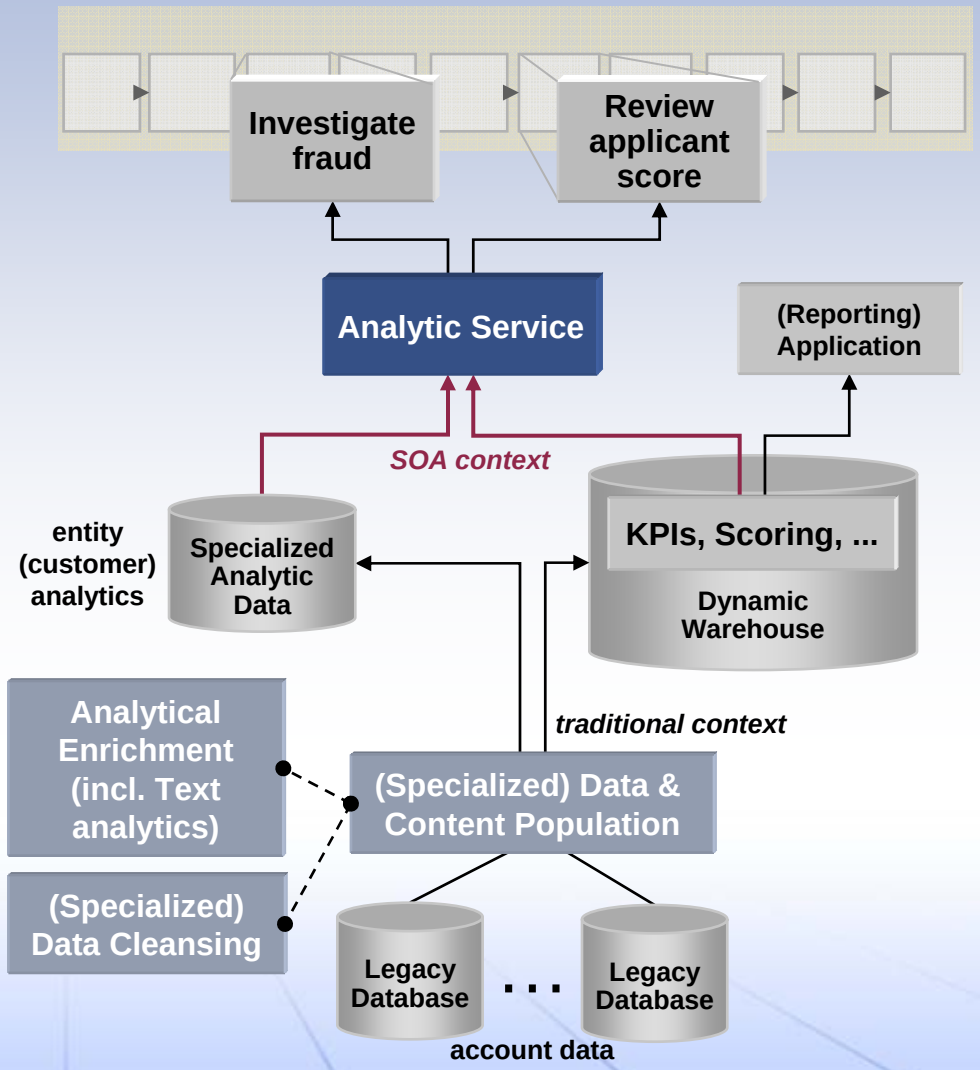
Master Data Management Services

- **As Is Environment**
  - Inconsistent, incomplete master data
  - Redundant and inaccurate implementations to access and integrate master data
- **Solution Characteristics**
  - Integrate data using data population
  - Guarantee quality using data cleansing
  - Master data repository to provide single version of the truth for service access
  - Ensure consistency through transaction and data synchronization
- **Results**
  - Established trusted source of core business information
  - Scalable approach for adding new applications and providing access to trusted data



Analytic Services

- **As Is Environment**
  - High exposure to fraud due to disparate data about individuals
  - Significant overhead to implement compliance guidelines
  - Manual, inefficient implementations to aggregate data for KPIs, scores, etc.
- **Solution Characteristics**
  - Aggregate data by applying data population pattern, cleansing pattern and possibly analytical enrichment
  - Expose analytic insight from warehouses and stores as services
- **Results**
  - Improve analytical insight
  - Time to market improvement; scalability and performance gains



Path to the Future – Expectations on Information Management

- ***Consistency, integrity, and holistic management of information (metadata, structured data, unstructured data)***
- ***Improved governance of data***
- ***Event driven architectures***
- ***Scalability, performance, …***
- ***And and and …***

- ***It is impractical to solve all problems given the resources of a typical organization and the required investment***

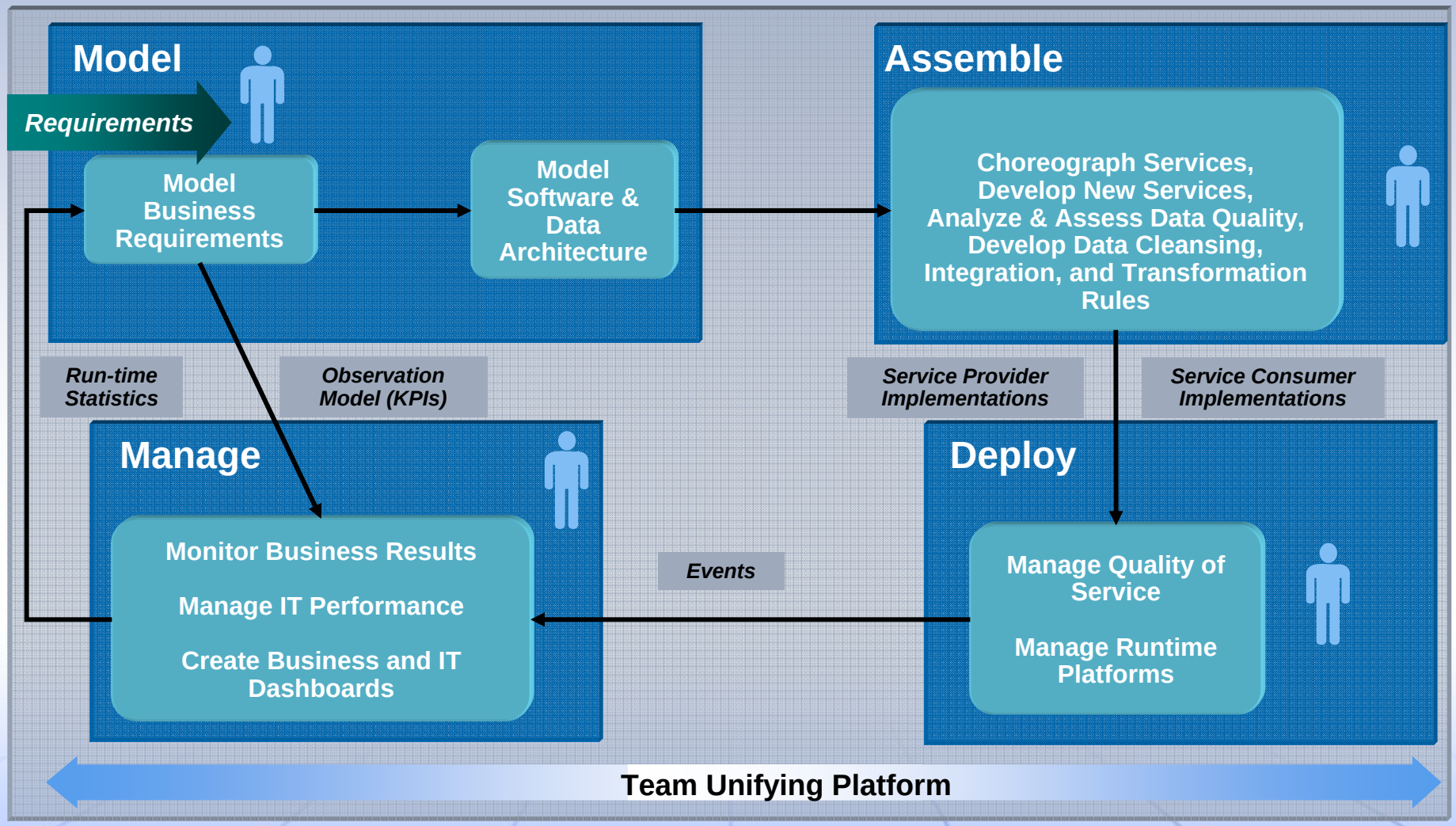
Path to the Future – An Approach

- ***SOA and Information as a Service help to realize "Think big, start small"***
- ***Think big***
  - Decoupling of provider from consumer through a stable service interface allowing to evolve in controlled manner
  - Extensibility to add services based on business need
  - Extensibility to add functionality based on non-functional requirements (ESB, security, management, etc)
  - Governance to manage the change
- ***Start small***
  - Start with a few services that provide the highest value
  - Stable service interface while possibly modifying the service realization: start with what fits current needs and possibly adjust later (e.g., federated data now, consolidate into single system later or proceed with hybrid approach)

Path to the Future – Criteria for Success

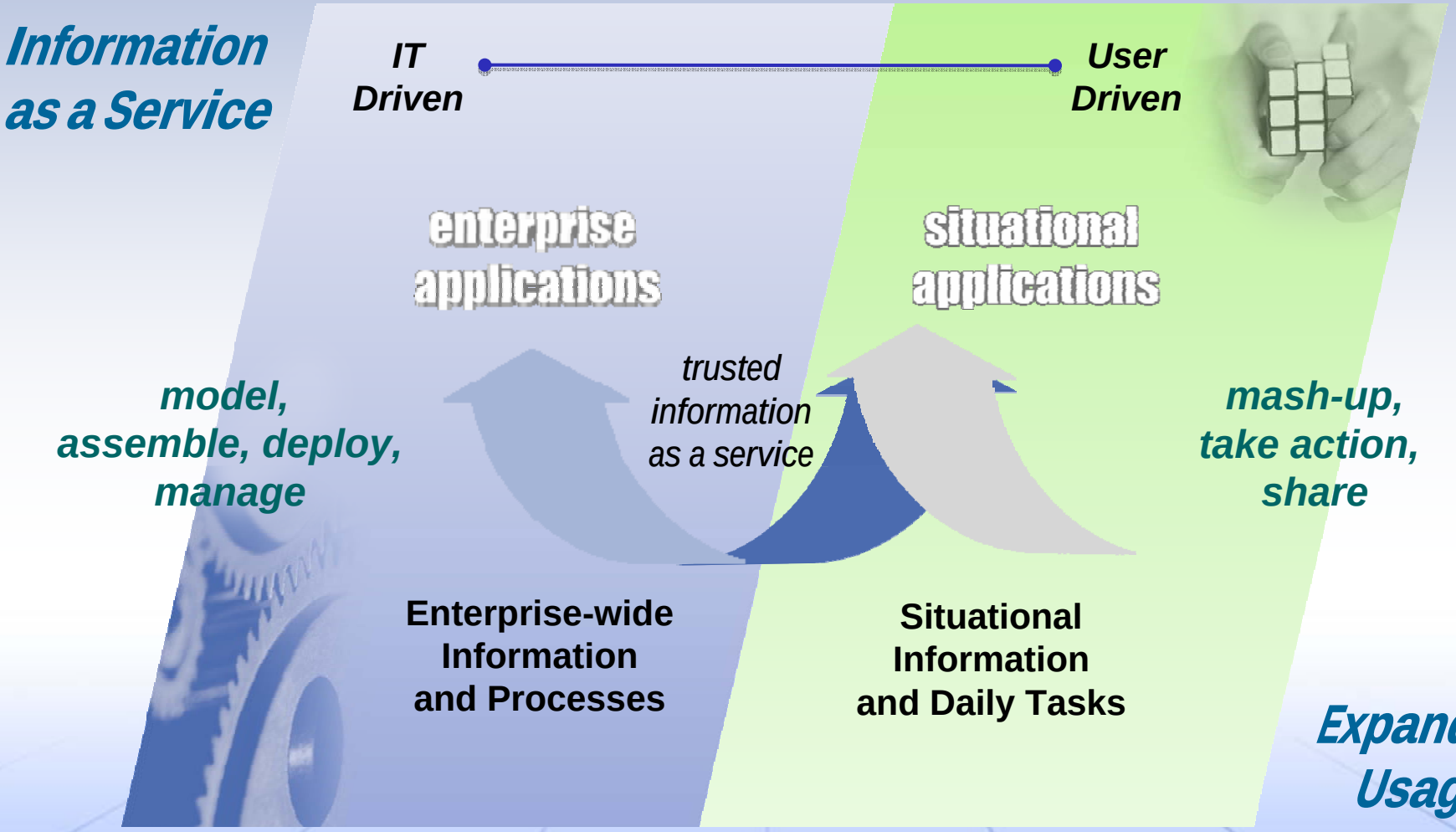
- ***Formally design and model your architecture based on business requirements***
  - You cannot develop a scalable framework if you start by writing lines of code but by developing the overall design and model first
  - Focus primarily on business needs
  - ↳ Business driven development & model driven architectures
- ***Leverage best practices and industry standards***
  - Modeling just a few objects and entities is fairly straightforward
  - Modeling a few objects/entities that can be expanded into an enterprise-wide model without significant change is extremely challenging
  - ↳ Industry models provide this extensive and extensible framework
- ***Manage metadata effectively***
  - After you have modeled your design, you need to efficiently manage and share the artifacts and ensure the consistency across tools and tasks
  - Once you have created the design, you will need to manage its change over time and ensure the consistency over time
  - ↳ Unified metadata management platform & governance

Business Driven Development
An Iterative, Business-focused Development Process



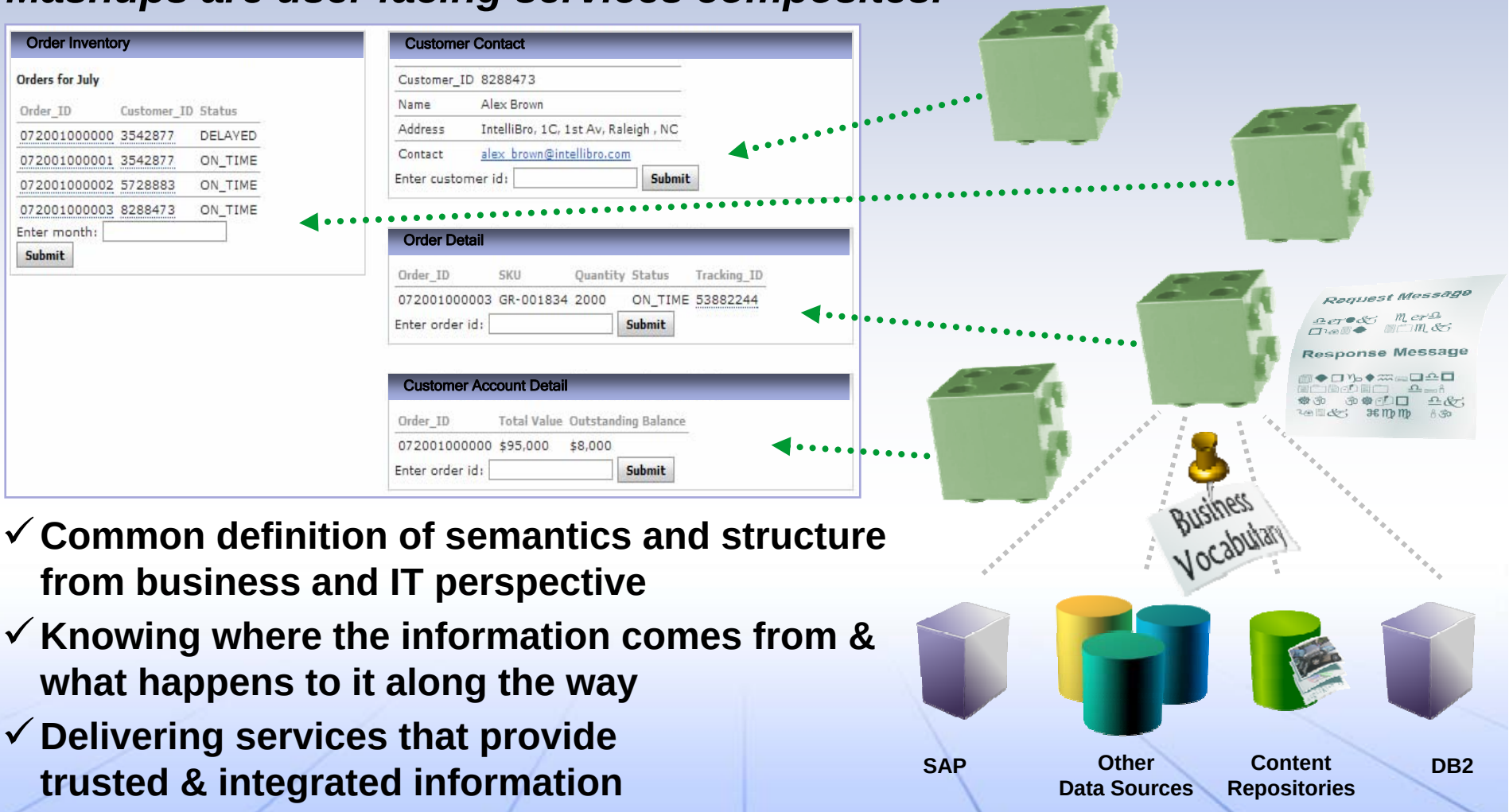
Information as a Service Enables Reuse…

…and Web2.0 will Dramatically Expand Usage Models



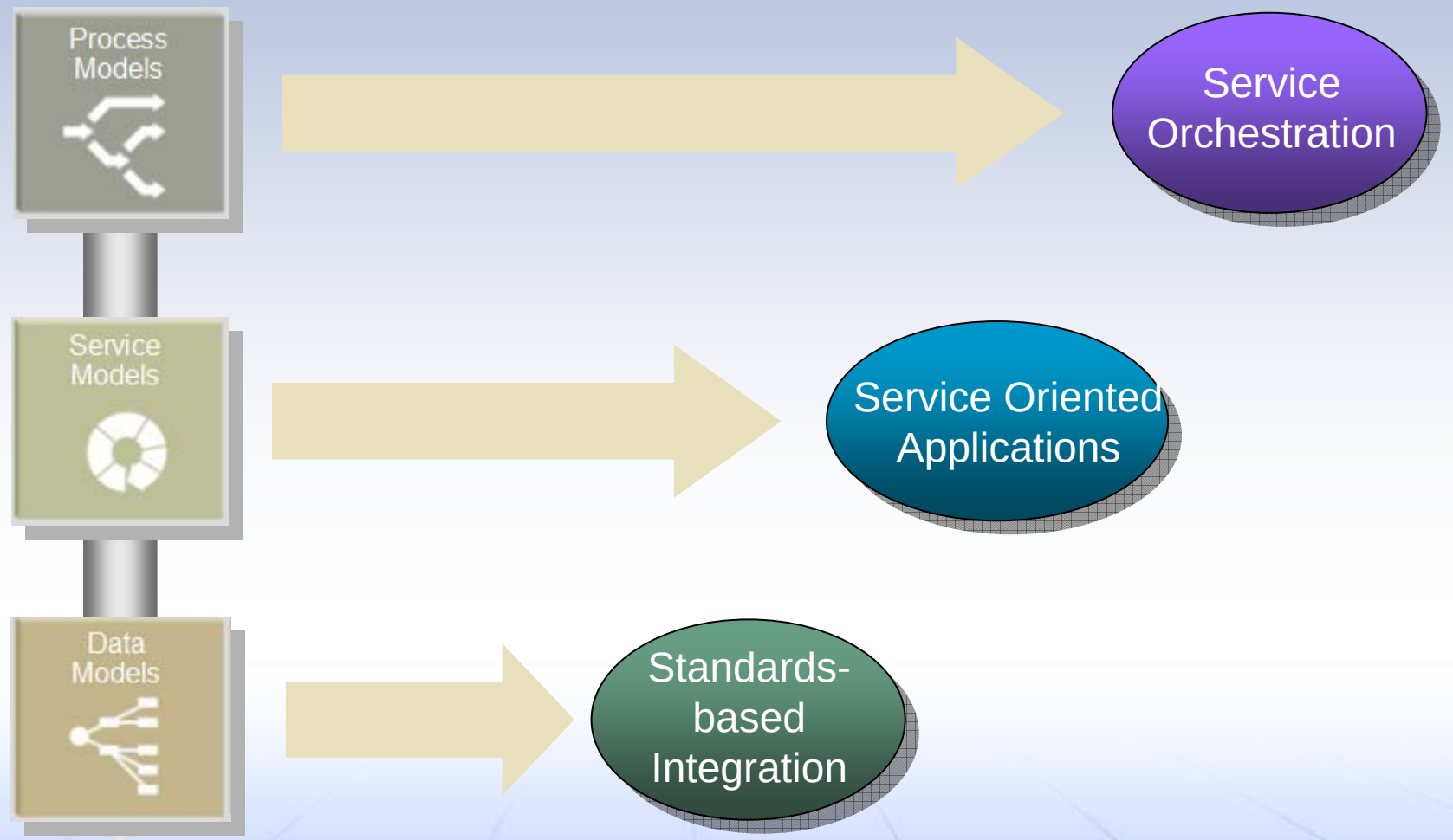
Business Mashups Create Composite Business Services

Portals assemble user interface components (portlets) into mashups. Mashups are user facing services composites.

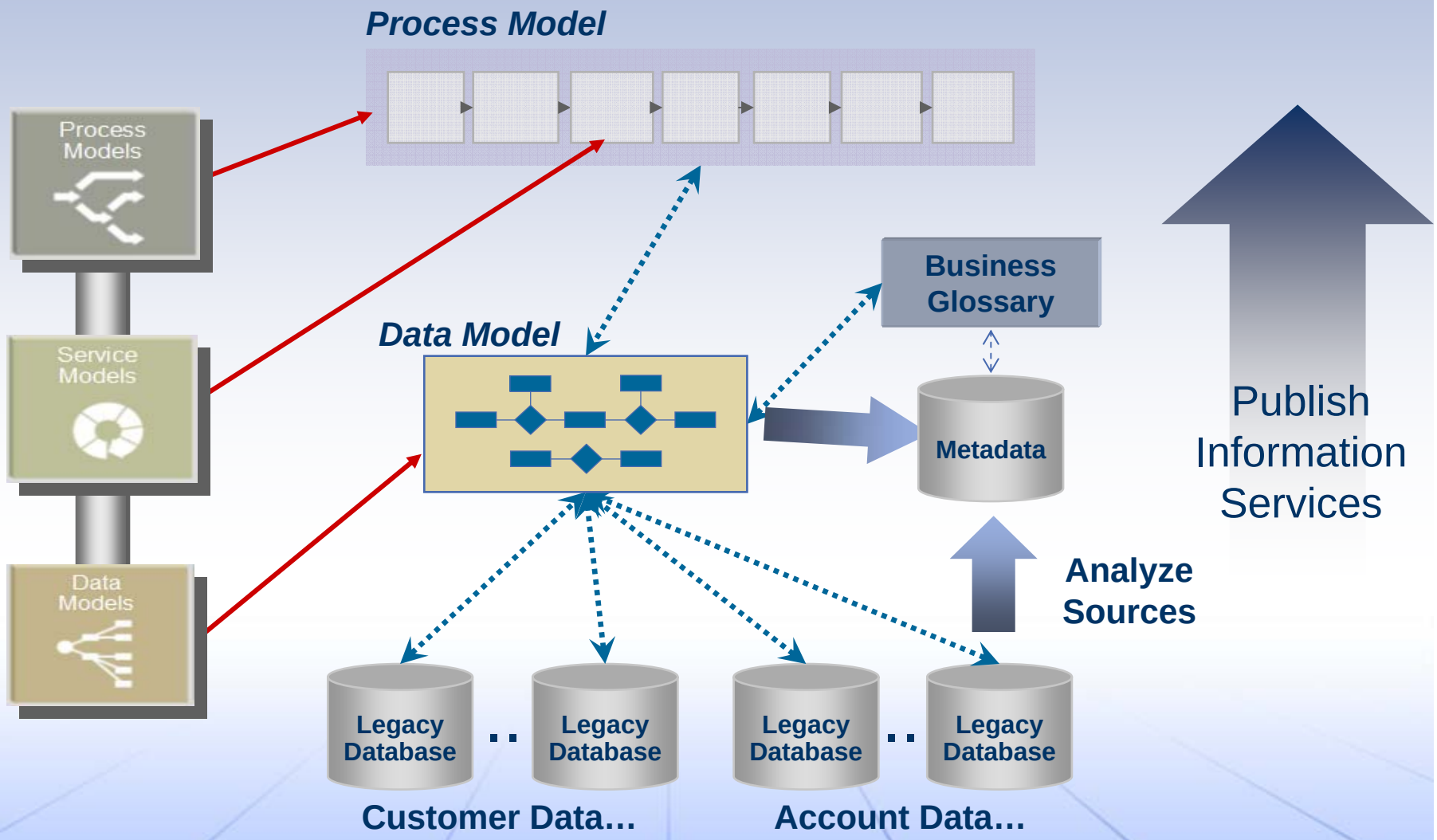


- ✓ **Common definition of semantics and structure from business and IT perspective**
- ✓ **Knowing where the information comes from & what happens to it along the way**
- ✓ **Delivering services that provide trusted & integrated information**

Industry Models



Process, Service & Data Models Need to Cooperate



Metadata Management – Meaning Comes from Many Perspectives

Business User

Business Glossary

Record business terms and taxonomies

Subject Matter Experts

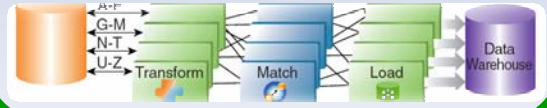
Information Analyzer

Uncover data in source systems and map to business terms

Developers

QualityStage and DataStage

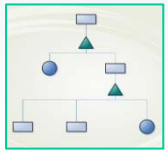
Transforming raw data into trusted data



Data Architects

Rational Data Architect

Data models



Unified Metadata Management

Capturing Design And Operational Metadata

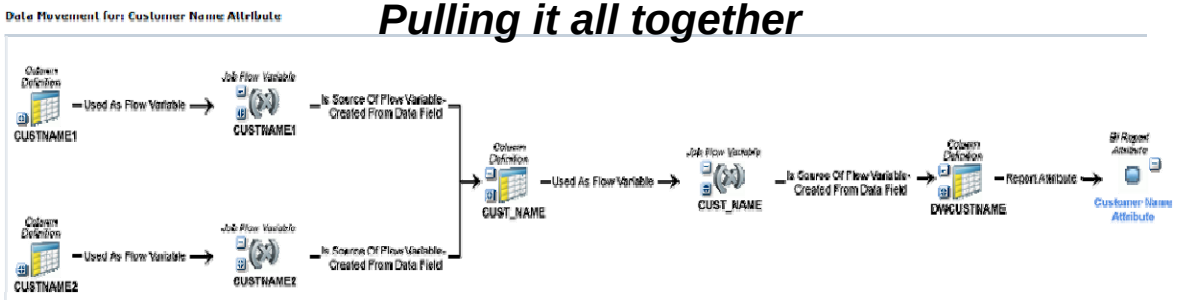
Developers

Information Services Director

Relate service interfaces to trusted data

IBM Metadata Workbench

Pulling it all together

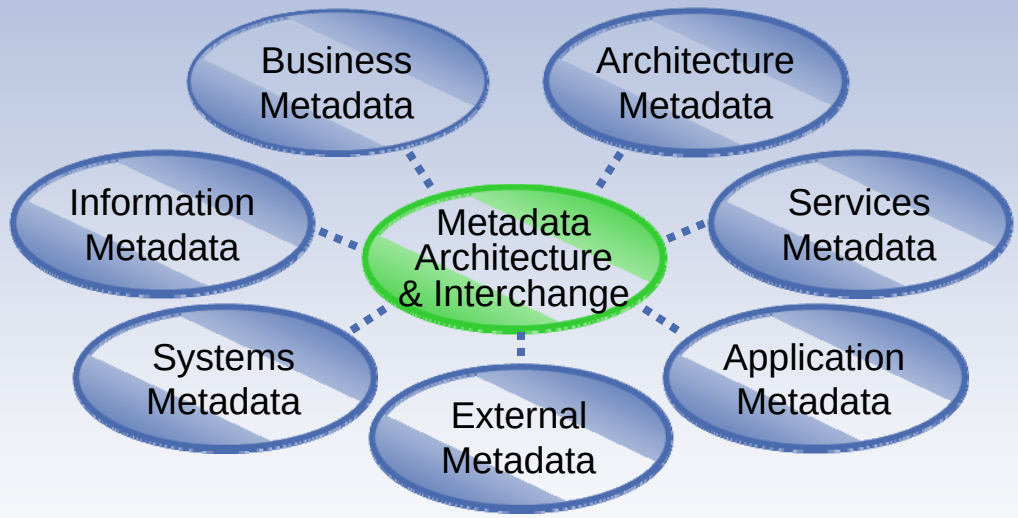


Third Parties

BI reports mapped to trusted data and more…

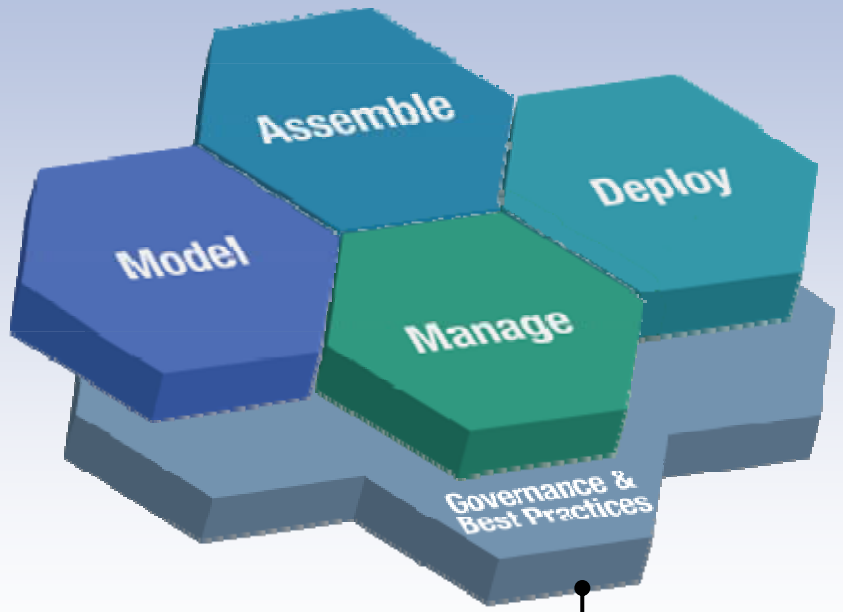
CA · COGNOS · Business Objects

Improved Sharing and Consistency of Metadata



- ***Sharing of metadata within a domain (e.g. "data", "services", etc.) is improving***
- ***Efficient and consistent sharing of metadata – structure and semantics – across domains and organizations will need to be resolved***

SOA governance effectively manages the service lifecycle



Effective SOA governance must:

- Help define guiding decisions around these processes
- Properly enforce these guiding decisions
- Communicate these guiding decisions effectively
- Evolve these guiding decisions with changing needs
- Ensure that the perspective of both service providers and consumers are properly met

The Phases of Methodology; the SOA Governance and Management Method

Design the governance approach

- Define / modify governance processes
- Design policies and enforcement mechanisms
- Identify success factors and metrics
- Identify owners and the funding model
- Charter / refine an SOA center of excellence
- Design the governance IT infrastructure

Put the governance model into action

- Deploy governance mechanisms
- Deploy the governance IT infrastructure
- Educate and deploy on expected behaviors and practices
- Deploy policies

Scope the governance need

- Document and validate the business strategy for IT and SOA
- Assess current IT and SOA capabilities
- Define / refine the SOA vision and strategy
- Review current governance capabilities and arrangements
- Lay out the governance plan

Manage and monitor the governance processes

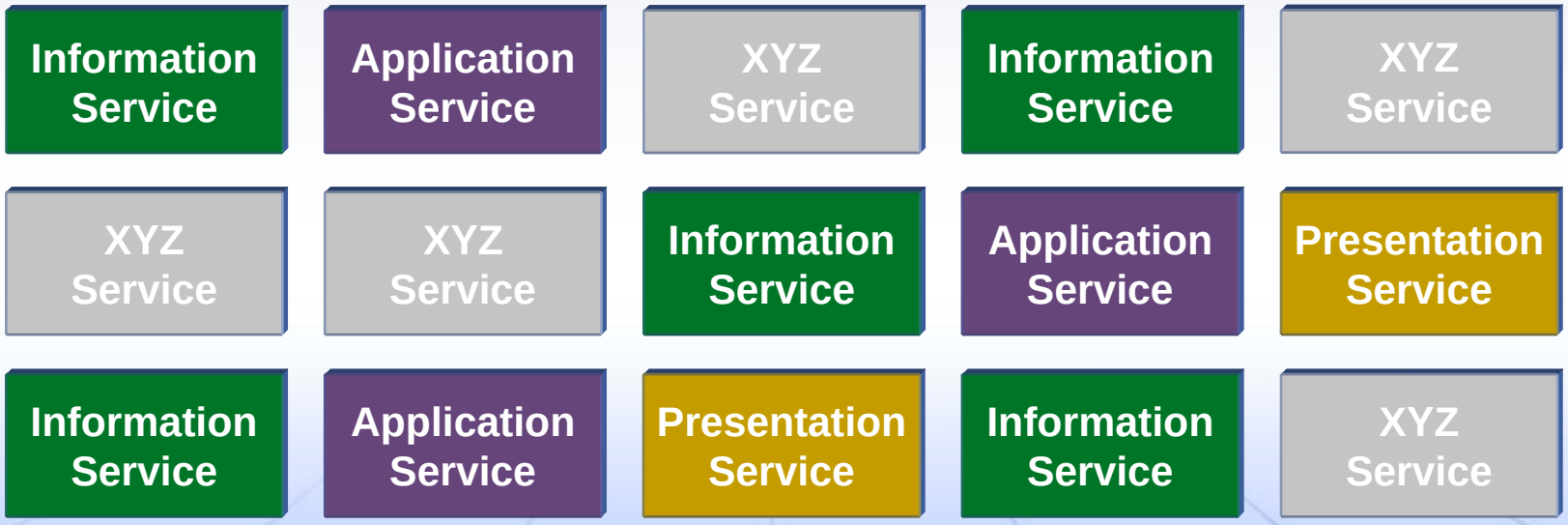
- Monitor compliance with policies
- Monitor compliance with governance arrangements
- Monitor IT effectiveness metrics

Path to the Future – Further Considerations

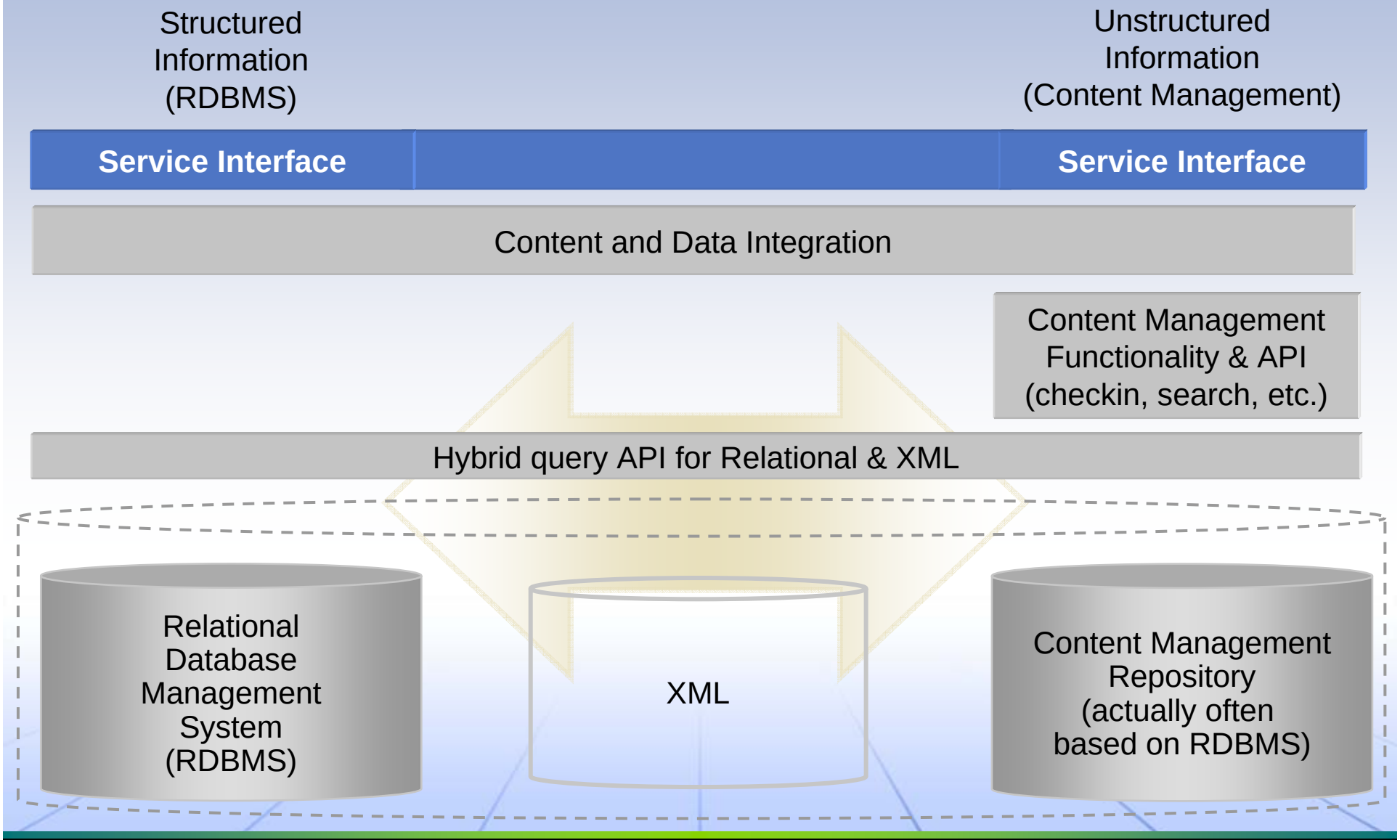
- ***The concept of encapsulation & components and information services***
- ***Evolution of structured and unstructured information in SOA***
- ***Event driven architectures***

The Concept of Encapsulation & Components and Information Services

- ***Concept of encapsulation: hide implementation***
- ***Concept of components: easy to assemble, possibly distributed providers***



Evolution of Structured & Unstructured Information in SOA



Questions

Thank You